Three Dimensional Digital Image Correlation for Dynamic Measurements

by

Dylan Dowling

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Civil Engineering

Waterloo, Ontario, Canada, 2019

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Structural inspections have traditionally been completed by inspectors using simple tools to determine the health of a structure. The limitations using these techniques led to the development of quantitative sensors such as strain gauges, displacement transducers, and accelerometers. These sensors are still limited by requiring physical access to the structure and extensive cabling and setup. Optical techniques, such as laser Doppler vibrometry and photogrammetry, have been developed to allow technicians to inspect the structure with little or no physical access to the structure and minimal cabling. Digital Image Correlation (DIC) is a photogrammetry technique that involves tracking multiple subsets through a series of images to determine full field displacements. DIC can take the form of 2D DIC using one camera or 3D DIC using more than one camera.

DIC use has typically focused on strain analysis, although researchers have recently begun to use DIC for full field dynamic analysis. Commercial packages have focused on strain analysis, with few software packages having any modal analysis capabilities built in. Commercial packages are also quite expensive. The goal of this work is to provide an open-source 3D DIC software package with built in modal analysis functionality. The software package will allow for use of various interpolation functions, shape functions, and de-noising options working within an Inverse Compositional Gauss-Newton (IC-GN) registration algorithm.

A numeric study was undertaken to determine optimal 2D and 3D DIC analysis parameters using the IC-GN algorithm. The numeric study made use of simulated images to provide improved control over image quality and to eliminate or control error sources. Images were simulated using 8px diameter speckles and a subpixel displacement from 0 to 1px. Left and right images were generated with a known horizontal displacement to simulate a classical stereo vision setup. For 2D analysis, a bicubic interpolation function with a 5x5 Gaussian blur is recommended. If the displacement field is known, then a matching shape function should be used. Otherwise, a 2$^{\text{nd}}$ order shape function should be used to ensure the displacements are captured. A subset size between 41x41px and 61x61px is recommended for 2D analysis. For 3D analysis, a bicubic interpolation function with no Gaussian blur is recommended. A higher order interpolation function was found to not improve error levels, and the Gaussian blur increased the random errors without improving the bias errors. A 2$^{\text{nd}}$ order shape function is recommended when matching between left/right images to capture deformation with imperfectly aligned cameras. A matching shape function is recommended for analyzing the sequential images if the displacement field is known, otherwise a 2$^{\text{nd}}$ order shape function should be used for analyzing sequential images as well. A 41x41px to 61x61px subset size is also recommended for 3D DIC analysis.

An experimental validation test was conducted on a thin, vibrating steel plate. This test demonstrated the capabilities of the 3D DIC system in conducting real world modal analyses and served to verify the system measurements against a reference laser vibrometer. Two high speed, high resolution cameras captured one side of the plate, while a laser vibrometer captured the other side. Excitation was provided by an impact hammer. The displacement signals were generally found to be 97 to 98% cross correlated. The mean error in the magnitude of the displacement signal peaks was approximately 1.5 to 1.7% for most test runs. The DIC system demonstrated very high accuracy in measuring displacements using the laser vibrometer as a baseline. The modal analysis revealed very similar fundamental frequencies measured by the DIC system and laser vibrometer, with the largest difference being only 0.08Hz. The DIC system was also able to extract the mode shapes of the plate, which matched favourably with the results from finite element modal analysis. Overall, 3D DIC is a very valuable tool in modal analysis due to its quick setup, good accuracy, and additional information in the form of mode shapes and damping ratios.

Acknowledgements

Thank you to my supervisors, Dr. Scott Walbridge and Dr. Sriram Narasimhan for their assistance and allowing me to take on a unique project that aligned with my personal interests. I would also like to thank my thesis reviewers, Dr. Chul Min Yeum and Dr. Bruce MacVicar, for their time and comments.

I have to give a giant thank you to the lab technicians, Doug Hirst, Peter Volcic, and especially Richard Morrison. Whether I needed help in the lab, with student teams, or just to borrow tools and get advice, you guys were always there to help.

I of course have to thank both of my research groups. Whether it was only once a term or every day in the lab, you guys were always there to talk to and discuss problems with me. I especially want to thank Rajdip Nayek for all of his patient explanations of experimental modal analyses, Nick Charron and Evan MacLaughlin for their many discussions with me about imaging, and James St. Onge for the many and varied things he assisted me with.

Thank you to the students and professors who allowed me to teach and learn from them, most especially Dr. Rania Al-Hammoud. Being able to teach helped me find who I am as a person and helped me find a passion I never knew I had.

Last, but definitely not least, thank you to everyone who I have met and befriended through the last 2-7 years. You are too numerous to name, but your friendships have helped put a smile on my face every day. I especially want to thank my office mates and friends who put up with my puns for the past few years. I honestly have no idea how or why any of you put up with my jokes, but the laughs and groans I got helped me smile through everything!

Table of Contents

List of Tables xv

List of Figures xvii

1 Introduction 1

1.1 Sensing Technologies 2

1.1.1 Conventional Sensors 2

1.1.2 Optical Sensors 2

1.2 Camera Technologies 4

1.2.1 Digital Imaging 5

1.3 Motivation and Objectives 7

2 Literature Review 9

2.1 DIC Background 9

2.1.1 Correlation Criteria 11

2.1.2 Shape Functions 13

2.1.3 Subpixel Interpolation Functions 17

2.1.4 Subpixel Registration Algorithms 25

2.2 Digital Image Correlation in the Fourier Domain 32

2.2.1 Fourier Transform of Digital Images 32

2.2.2 Correlation Using Discrete Fourier Transforms 35

2.3 Camera Calibration . . . 35

2.4 3 Dimensional Vision . . . 39

2.4.1 Stereoscopy . . . 40

2.4.2 Epipolar Geometry . . . 41

2.5 Noise Reduction . . . 45

2.5.1 Noise Sources . . . 45

2.5.2 Gaussian Pre-Filtering . . . 46

2.6 Speckle Patterns . . . 46

2.6.1 Speckle Pattern Simulation . . . 48

2.7 Applications of DIC . . . 49

2.7.1 Laboratory Usage . . . 49

2.7.2 Field Usage in Civil Engineering . . . 51

2.8 Errors in Digital Image Correlation . . . 54

2.8.1 Errors in 2D DIC . . . 54

2.8.2 Errors in 3D DIC . . . 57

3 Experimental Setup **61**

3.1 Numeric Testing . . . 61

3.1.1 Image Simulation . . . 61

3.1.2 Test Procedure . . . 62

3.2 Laboratory Testing . . . 63

3.2.1 Equipment . . . 63

3.2.2 Test Procedure . . . 70

3.2.3 Modal Analysis . . . 70

3.3 Software Architecture . . . 71

3.3.1 Calibration Model . . . 73

4 Results 75

4.1 Numeric Results 75

4.1.1 Speckle Size 75

4.1.2 Shape Function 77

4.1.3 Interpolation Function 77

4.1.4 Noise 78

4.1.5 Noise Reduction 79

4.1.6 Subset Size 79

4.2 Laboratory Results 80

4.2.1 Camera Noise 80

4.2.2 DIC Analysis Settings 80

4.2.3 Displacements 82

4.2.4 Modal Analysis Results 84

5 Analysis & Discussion 91

5.1 Numeric Discussion 91

5.1.1 Speckle Size 91

5.1.2 Shape Function 92

5.1.3 Interpolation Function 92

5.1.4 Noise 93

5.1.5 Noise Reduction 94

5.1.6 Subset Size 97

5.1.7 Recommended Analysis Parameters 98

5.2 Laboratory Discussion 99

5.2.1 Comparison to Laser Vibrometer 100

5.2.2 Modal Analysis 101

5.2.3 Testing Errors 107

6 Conclusions and Recommendations 109
6.1 Numeric Results 109
6.1.1 Speckle Size 109
6.1.2 2 Dimensional DIC 110
6.1.3 3 Dimensional DIC 110
6.2 Laboratory Results 111
6.2.1 Accuracy of System 111
6.2.2 Modal Analysis 112
6.3 Future Work 113
6.3.1 Applications 113
6.3.2 DIC System Improvements 113

References 117

APPENDICES 125

A Derivations 127
A.1 Derivation of Warp Functions 127
A.1.1 0^{th} Order Shape Function 127
A.1.2 1^{st} Order Shape Function 128
A.1.3 2^{nd} Order Shape Function 128
A.2 Derivations of Explicit Warp Inversion 132
A.2.1 0^{th} Order Shape Function 132
A.2.2 1^{st} Order Shape Function 134
A.3 Derivations of B-Spline Kernels 135
A.3.1 Bilinear B-Spline 135
A.3.2 Bicubic B-Spline 136
A.3.3 Biquintic B-Spline 137

B Graphs **141**

B.1 Numeric Results 142

B.1.1 Shape Function 142

B.1.2 Interpolation 146

B.1.3 Noise Level 150

B.1.4 Noise Reduction (Gaussian Pre-Filtering) 154

B.2 Laboratory Results 171

B.2.1 Raw Data Graphs 171

B.2.2 Displacement Comparison Graphs 180

B.2.3 DIC Stabilization Diagrams 186

B.2.4 Laser Vibrometer FFTs 191

B.2.5 Mode Shapes 194

List of Tables

3.1 Simulated image test matrix. 63

3.2 Individual camera calibration parameters. 66

3.3 Stereo camera calibration parameters. 67

4.1 Fundamental frequencies from DIC (full field). 84

4.2 Damping ratios from DIC (full field). 87

4.3 Fundamental frequencies from LDV (single point). 88

4.4 Fundamental frequency comparison with FEA model. 88

5.1 Recommended analysis parameters for 2D and 3D DIC. 98

5.2 Differences between the DIC and LDV systems. 101

5.3 Statistical significance test of fundamental frequencies. 102

List of Figures

1.1 Parts of a camera. 5

2.1 Typical shape functions used in DIC. 14

2.2 B-Spline kernels for linear, cubic, and quintic splines. 20

2.3 Bidirectional spline interpolation functions. 20

2.4 Forward Additive Newton-Raphson algorithm (1^{st} order shape function). . 29

2.5 Inverse Compositional Gauss-Newton algorithm (1^{st} order shape function). 31

2.6 Example of a 2D Fourier transform of an image. 34

2.7 Typical pinhole camera model. 36

2.8 Coordinate systems present in pinhole camera model. 37

2.9 Intrinsic camera parameters found through calibration. 38

2.10 Basic two camera layout for three dimensional vision. 40

2.11 Simple case of epipolar geometry with co-planar camera sensors. 42

2.12 Error due to quantization of camera sensor. 42

2.13 Epipolar lines resulting from a point, x. 43

2.14 Example of a speckle pattern. 47

2.15 Bias error vs. subpixel displacement. 57

3.1 Example simulated image. 62

3.2 3D displacement errors. 64

3.3 JAI SP-12000M high speed cameras with ZEISS 50mm f/2.2 lenses. 65

3.4 JAI SP-12000M high speed camera setup. 66

3.5 Polytec PDV100 laser vibrometer. 68

3.6 Laser vibrometer setup. 69

3.7 Stainless steel plate hanging from support (speckle side shown). 71

3.8 Main GUI of the DIC system. 72

3.9 Image order in DIC system. 73

4.1 Peak bias error vs. speckle size. 76

4.2 Mean bias error vs. speckle size. 76

4.3 Region of interest used in testing. 81

4.4 Example of displacements found by the DIC system. 82

4.5 Example of velocities found by the LDV. 83

4.6 Example of numerically integrated displacements found by the LDV. . . . 83

4.7 Stabilization diagram from the experimental testing, Run 1. 85

4.8 First four mode shapes found from the experimental modal analysis, Run 3. 86

4.9 Power spectral density (PSD) curve for the LDV from Run 3. 87

4.10 ABAQUS model and first three mode shapes. 89

5.1 Bias errors present in analysis due to 5x5 Gaussian blur. 96

5.2 Displacements from the DIC system and numerically integrated LDV signal. 100

5.3 Graphical comparison of LDV and DIC frequencies. 104

5.4 Mode shapes from DIC and FEA model. 106

5.5 Error in Run 1. 108

B.1 Errors present in 2D analysis due to shape function (no blur added). 142

B.2 Errors present in 2D analysis due to shape function (3x3 blur added). . . . 143

B.3 Errors present in 3D analysis due to shape function (no blur added). 144

B.4 Errors present in 3D analysis due to shape function (3x3 blur added). . . . 145

B.5 Errors present in 2D analysis due to interpolation (no blur added). 146

B.6 Errors present in 2D analysis due to interpolation (3x3 blur added). 147
B.7 Errors present in 3D analysis due to interpolation (no blur added). 148
B.8 Errors present in 3D analysis due to interpolation (3x3 blur added). 149
B.9 Errors present in 2D analysis due to noise. 150
B.10 Errors present in 2D analysis due to noise. 151
B.11 Errors present in 3D analysis due to noise. 152
B.12 Errors present in 3D analysis due to noise. 153
B.13 Bias errors present in 2D analysis due to Gaussian blur (0% noise). 154
B.14 Standard deviation errors present in 2D analysis due to Gaussian blur (0% noise). 155
B.15 Bias errors present in 2D analysis due to Gaussian blur (2% noise). 156
B.16 Standard deviation errors present in 2D analysis due to Gaussian blur (2% noise). 157
B.17 Bias errors present in 2D analysis due to Gaussian blur (4% noise). 158
B.18 Standard deviation errors present in 2D analysis due to Gaussian blur (4% noise). 159
B.19 Bias errors present in analysis 3D due to Gaussian blur (0% noise). 160
B.20 Standard deviation errors present in 3D analysis due to Gaussian blur (0% noise). 161
B.21 Bias errors present in 3D analysis due to Gaussian blur (2% noise). 162
B.22 Standard deviation errors present in 3D analysis due to Gaussian blur (2% noise). 163
B.23 Bias errors present in 3D analysis due to Gaussian blur (4% noise). 164
B.24 Standard deviation errors present in 3D analysis due to Gaussian blur (4% noise). 165
B.25 Bias errors due to subset size in 2D. 166
B.26 Random errors due to subset size in 2D. 167
B.27 Bias errors due to subset size in 3D. 168
B.28 Random errors due to subset size in 3D. 169

B.29 Relative computational time of subsets. 170

B.30 DIC displacement time series from Run 1. 171

B.31 DIC displacement time series from Run 2. 172

B.32 DIC displacement time series from Run 3. 172

B.33 DIC displacement time series from Run 4. 173

B.34 DIC displacement time series from Run 5. 173

B.35 LDV velocity time series from Run 1. 174

B.36 LDV velocity time series from Run 2. 175

B.37 LDV velocity time series from Run 3. 175

B.38 LDV velocity time series from Run 4. 176

B.39 LDV velocity time series from Run 5. 176

B.40 Impact hammer force time series from Run 1. 177

B.41 Impact hammer force time series from Run 2. 178

B.42 Impact hammer force time series from Run 3. 178

B.43 Impact hammer force time series from Run 4. 179

B.44 Impact hammer force time series from Run 5. 179

B.45 LDV (integrated) vs. DIC displacement measurements from Run 1. 180

B.46 LDV (integrated) vs. DIC displacement measurements from Run 2. 181

B.47 LDV (integrated) vs. DIC displacement measurements from Run 3. 181

B.48 LDV (integrated) vs. DIC displacement measurements from Run 4. 182

B.49 LDV (integrated) vs. DIC displacement measurements from Run 5. 182

B.50 LDV (integrated) vs. bias corrected DIC displacement measurements from Run 1. 183

B.51 LDV (integrated) vs. bias corrected DIC displacement measurements from Run 2. 184

B.52 LDV (integrated) vs. bias corrected DIC displacement measurements from Run 3. 184

B.53 LDV (integrated) vs. bias corrected DIC displacement measurements from Run 4. 185

B.54 LDV (integrated) vs. bias corrected DIC displacement measurements from Run 5. 185

B.55 Stabilization diagram from Run 1. 186

B.56 Stabilization diagram from Run 2. 187

B.57 Stabilization diagram from Run 3. 188

B.58 Stabilization diagram from Run 4. 189

B.59 Stabilization diagram from Run 5. 190

B.60 FFT of LDV measurements from Run 1. 191

B.61 FFT of LDV measurements from Run 2. 191

B.62 FFT of LDV measurements from Run 3. 192

B.63 FFT of LDV measurements from Run 4. 192

B.64 FFT of LDV measurements from Run 5. 193

B.65 Mode shapes of Run 1. 194

B.66 Mode shapes of Run 2. 195

B.67 Mode shapes of Run 3. 196

B.68 Mode shapes of Run 4. 197

B.69 Mode shapes of Run 5. 198

B.70 FEA mode shapes 199

Chapter 1

Introduction

Structural inspections, which are critical to ensuring long term functionality and safety, have traditionally been completed by human inspectors using their vision and simple tools such as rulers, measuring tapes, levels, and analog displacement gauges. There are significant limitations to such tools, which have resulted in more modern technologies being developed to allow for more accurate, quantitative measurements. These technologies, such as strain gauges, linear variable displacement transducers (LVDTs), and accelerometers, still require physical access to the structure and often require extensive cabling and setup. Modern technology has enabled the development of newer systems that require limited or no access to the structure, such as optical and light based methods. These methods allow inspectors and technicians to inspect the structure with minimal or no physical access to the structure and no need for a stable ground reference under the structure. Light based sensors make use of lasers to record the movement of structures in the direction of the laser, while optical sensors make use of cameras and image processing to track various points on a structure. Digital image correlation (DIC) is an image processing method that tracks points between images for displacement measurements. However, current implementations of DIC systems are typically proprietary, resulting in higher costs and limited adaptability. An open-source DIC system would resolve the issues of cost while also allowing adaptation to new analysis and inspection needs. An open-source DIC system would also require validation tests and error measures to provide users with confidence in the system and to demonstrate its abilities in real world experiments.

1.1 Sensing Technologies

1.1.1 Conventional Sensors

Sensors are often required to measure displacements, strains, and/or accelerations in structures. Conventional sensors, such as strain gauges, LVDTs, stringpots, and accelerometers have been widely used to monitor static, quasi-static, and dynamic loading. These sensors are each suited for different purposes based on their nature (e.g., accelerometers to measure dynamic motion in the form of accelerations). Conventional sensors typically require physical application to a structure through the use of glues, waxes, or clamps in order to perform measurements. Unfortunately, many structures have poor or limited access, making it difficult to apply carefully calibrated, sensitive sensors to the structure precisely without affecting the measurement system. Many traditional sensors also require extensive cabling that can quickly add complexity to setup and increase setup costs. Some of these sensors, such as LVDTs, also require a ground reference below the structure, which may be difficult in some field environments. Bridges in particular suffer from issues of access due to typically spanning waterways, roadways, and other significant obstacles.

Conventional sensors have been researched and developed extensively in past decades, and their use is commonplace in industry as well as in academia. The reliability of these sensors is quite high when they are correctly applied to the structure, but the application process remains an area of variability in their usage. The alternative to conventional, contact based sensors are non-contact sensors such as the global positioning system (GPS), radar based technologies, and optical systems.

1.1.2 Optical Sensors

To overcome the shortcomings of conventional sensors, non-contact optical sensing methods have gained significant popularity amongst researchers and practitioners in recent times. These methods vary widely in technology, analysis, and applicability, but can be broadly grouped under the areas of pattern interferometry, laser Doppler vibrometry (LDV), and photogrammetry. These methods involve relatively fast set up times, with minimal cabling and often no physical access required, and can provide highly accurate results. However, they are still an active field of research, with some methods being quite expensive and others having limited field validation. Data storage concerns also exist for the imaging based systems due to the large file sizes of images, but storage has become less of a concern with the ever-decreasing cost of data storage.

Laser Based Methods

Pattern interferometry makes use of an interference fringe pattern created by superposing a pair of light patterns and measuring the difference between the patterns [6]. A laser pattern is split, with part of the signal going directly to the sensor and the other part reflecting off of the measured object. The two laser patterns are then recombined prior to reaching the camera sensor, and the resulting amplitudes are read by the camera and used to measure the displacements. Early use of pattern interferometry made use of photographic plates, but this was very time consuming and quickly replaced by electronic speckle pattern interferometry (ESPI) using digital cameras. ESPI has been used to conduct accurate dynamic measurements, but is very sensitive to ambient vibration [6].To combat the sensitivity to ambient vibration, researchers developed digital speckle shearography (DSS). The laser pattern is instead shone directly on the sample, and the reflected pattern is split instead of the original laser signal [6]. These measurement techniques were originally limited to measuring only a single axis of displacement, but ESPI has been successfully used to measure three axes through the use of three cameras and beams [6].

Instead of measuring the phase shift to determine displacements, laser Doppler vibrometry measures the frequency shift to determine velocities [6]. LDV is very commonly used to measure structures, but is limited to measuring a single point in a single direction at a given time. This can be extended to 3D measurements of a single point through the use of three LDVs. To generate full field measurements, multiple lasers or a scanning LDV are required. However, scanning LDVs measure points sequentially rather than simultaneously, and the excitation or response may change during a given measurement time step. This may limit their usage in situations where loading may change rapidly during measurement [6].

Photogrammetry

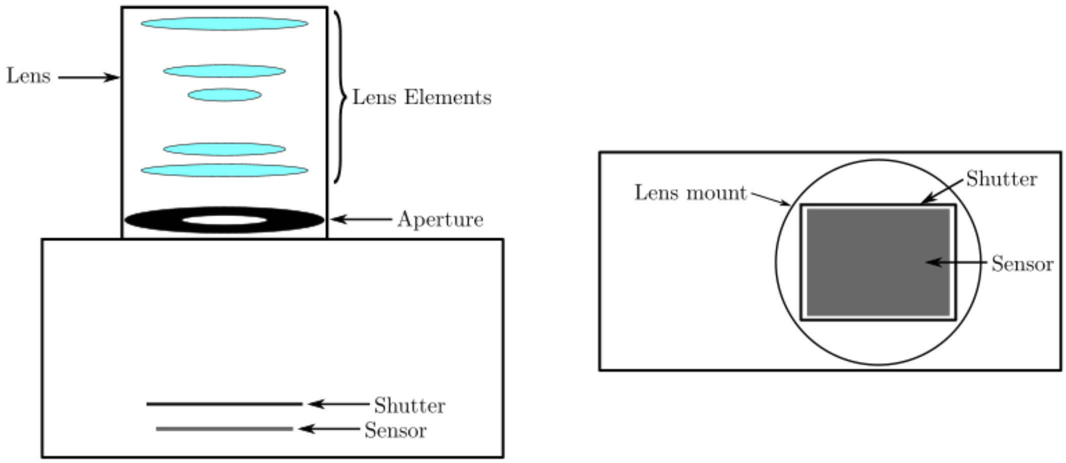
Photogrammetry focuses on determining and tracking the precise positions of points in images and photographs. This can be as simple as a 2D analysis of an image with a known scale or as complex as a multi-camera system to recreate 3D point clouds of items of interest. Early work using photogrammetry for mechanical analysis was completed by Peters and Ranson [39] in the 1980s, which allowed for calculating stress and strain in a simple tension test. The concept was soon extended to digital image correlation (DIC) for use in mechanical analyses by Sutton et al [53] and Chu et al [13].

DIC is a subfield of photogrammetry that involves the ability to measure full field displacements and calculate full field strains of specimens. DIC works by comparing a subset of the first (reference) image to a subset in each subsequent (analysis) image and optimizing a correlation score between these subsets to find the new position of the reference subset. Early implementations of DIC investigated simple 2D DIC using a single camera [13, 53]. 2D DIC systems can measure a single planar surface approximately parallel to the image sensor. This technology was then extended to 3D in the early 1990s through the use of early 3D point tracking and stereo vision principles [25]. 3D DIC is typically performed with a pair of cameras in a stereo vision setup rather than a single camera, although pseudo-stereo systems also exist using a single camera with the use of mirrors or colour filters [67]. Researchers also began to examine the errors present in DIC and their sources in the early 2000s, such as the errors present due to interpolation [44]. As camera technology progressed, researchers also began to investigate their use for dynamic or high speed measurements [18, 21]. DIC is typically performed through the use of non-linear iterative solvers [10, 28, 32, 52], although some researchers make use of other methods such as discrete Fourier transform (DFT) based DIC [7].

1.2 Camera Technologies

Cameras are optical devices composed of many parts, of which the key components are the aperture and the recording media. The aperture can be as simple as a small pinhole to allow light through, but it is more often an opening of variable, controllable size. The recording media can take many forms, such as photographic plates, film, or digital sensors. Lenses are also frequently used to magnify or focus the image, and are typically comprised of magnification, focus, and chromatic aberration reduction elements in front of the aperture. Digital cameras also contain a processor and sometimes a mechanical shutter. Single lens reflex (SLR) cameras contain a mirror and a pentaprism or pentamirror to allow viewing through a viewfinder, but these are absent in most research and industrial cameras. Digital sensors are comprised of millions of individual photosites that correspond to individual pixels, with each photosite made of a photosensitive material. As light reaches the photosensitive material, it charges a capacitor, which is released into the processor following a shutter completion. Figure 1.1 contains a labelled display of the main parts present in a camera.

Sensors can be broadly divided into charged coupled device (CCD) and complementary metal oxide semiconductor (CMOS) sensors, also known as active pixel sensors. CCD sensors contain a photosensitive layer and a transmission layer. The capacitors in the photosensitive layer are charged by photons of light striking the photosensitive layer. Following an exposure, a control circuit causes the charge in a capacitor to shift to the neighbouring



(a) Top down view of camera.

(b) Front view of camera, shutter open without a lens.

Figure 1.1: Parts of a camera.

capacitor through the transmission layer, with the capacitor nearest to the edge discharging into a charge amplifier, where it gets converted to voltage, and subsequently to a digital signal through an analog-to-digital conversion. CMOS sensors also make use of a photosensitive layer, but do not make use of a transmission layer. Instead, pixels in a CMOS sensor contain individual active amplifiers. In the past, CCD sensors have had higher quality than CMOS sensors due to more of the sensor space being comprised of photosensitive materials, but this gap has narrowed over time. CMOS sensors have many advantages, including a lower production cost and much less blooming (spilling into neighbouring pixels due to overcharged photosites) in images due to there being no transmission layer. However, the additional components in a CMOS sensor take up more space on the sensor that is not found in CCD sensors, marginally reducing the photosensitivity of CMOS sensors relative to comparably sized CCD sensors.

1.2.1 Digital Imaging

Digital images are discrete measurements of the light found in a given scene. These discrete measurements are made at the individual photosites in the camera sensor, and are converted to digital pixels in an image. The pixels can be seen as similar to a matrix, where the value at each (i, j) location in the matrix corresponds to the light captured by a pixel at that location. Greyscale images contain a single channel consisting of intensity values from black to white. Colour images make use of three colour channels corresponding to the red, green, and blue light channels, which results in a three layer matrix of values. Colour images are created by using photosites that are sensitive to specific light wavelengths. A colour image is then reconstructed from the various colour intensities at each pixel location in a process known as debayering or demosaicing. Demosaicing uses interpolation to estimate the pixel values of each colour at photosites that are sensitive to the other colours (i.e., to estimate the red values at green and blue photosites) to form a full image. This can lead to some aliasing, so an optical anti-aliasing filter is often used at the back of lenses to reduce artifacts. For this reason, greyscale cameras are still preferred in many high precision applications. Some colour sensors use a stacked sensor approach, where the various wavelengths are absorbed by different layers in the sensor, but these sensors are uncommon.

Once digitized, images are often stored as 8 bit values ranging from 0 to 255, with 0 corresponding to no light (black) and 255 corresponding to maximum light (white). Images may also be stored as 10 bit values, which instead range from 0 (black) to 1023 (white), or 12 bit values, which range from 0 (black) to 4095 (white). 10 and 12 bit images allow for a finer gradation of intensity values, but come at the cost of increased storage and processing requirements, and remain uncommon in many applications.

Once captured, images may be left in a raw format or compressed to reduce the storage requirements. Raw images retain unprocessed pixel values, and each pixel retains a value from 0 to 255 (for 8-bit images). Compression may take the form of lossy or lossless compression. Lossy compression schemes, such as JPEG, remove information that will not be perceptible to the human eye to reduce file size through processes such as discrete cosine transforms. Lossless compression, such as PNG, reduces the file size without removing any information from the camera. Some file formats, such as TIFF, are able to use lossy or lossless compression as dictated by the user. Lossy compression formats are not recommended for photogrammetry applications. While the information removed may be imperceptible to the human eye, it may still be usable by a computer to improve registration or correspondence.

1.3 Motivation and Objectives

Existing DIC systems are typically expensive, have limited adaptability, and are sometimes linked to proprietary cameras. Most systems focus on measuring displacements for strain analysis, and few systems are able to use the measured displacements for other uses like modal analysis despite many systems boasting framerates in excess of 200fps. The vibration characteristics of various materials and structures are also of interest to engineers, and should be incorporated into DIC software. This project is aimed at producing an open-source 3D DIC system that can provide full-field displacement measurements for both static and dynamic testing. Open-source software will allow for lowered costs and the ability to use generic cameras, while a modal analysis submodule will allow for the extraction of fundamental frequencies and mode shapes of tested structures. The primary objectives are to:

- Develop an open-source, adaptable, modern DIC system using current algorithms
- Quantify the error levels present in the developed system and determine ideal parameters to limit errors
- Demonstrate the accuracy and real world applicability of the system through validation displacement tests and dynamic measurements of a plate structure for modal analysis

Chapter 2

Literature Review

2.1 DIC Background

The earliest implementations of DIC worked in the spatial domain in two dimensions, computing integer pixel displacements along a planar surface [39]. Subpixel interpolation methods, such as the use of bilinear interpolators and cubic polynomials, were quickly developed, which allowed for accuracy below that of integer pixel values [13, 51, 53]. More advanced interpolators, such as higher order B-splines, were later used for improved accuracy [12, 44]. Spatial domain methods made use of cost or correlation functions, which were typically normalized cross correlation functions [23, 34, 52]. An efficient cross correlation method was proposed by Lewis [23] that made use of running sums to improve speed, which is particularly useful for calculating the initial guess of a subset displacement. Early DIC implementations made use of a coarse-fine search subpixel registration algorithm [13, 39, 51, 53], which was superseded by the Forward Additive Newton-Raphson (FA-NR) method [10, 31, 34]. Pan et al [32] recently proposed the use of an Inverse Compositional Gauss-Newton (IC-GN) subpixel registration algorithm for improved efficiency in place of the FA-NR.

Modern implementations of DIC consist of four primary components: a correlation function, a subpixel interpolation function, a shape function, and an iterative subpixel registration algorithm [29, 34]. The correlation function is used to determine how well two subsets match, and typically takes the form of a normalized cross correlation or normalized sum of squared differences [29]. Image interpolation is completed in 2 dimensions, and the interpolation functions are often made of bilinear or 2 dimensional B-spline interpolants [9, 12]. Shape functions are used to account for displacements and deformations within subsets between images. The subpixel registration algorithm is used to solve for the unknown displacement deformation parameters, and usually takes the form of a non-linear, iterative solver. The most popular registration algorithms are currently the Forward Additive Newton-Raphson (FA-NR) and recently developed Inverse Compositional Gauss-Newton (IC-GN) [10, 29, 32, 38].

The first step in the DIC analysis is the creation of a reference subset, which is typically a square set of pixel values surrounding the point of interest in the first (reference) image. This reference subset is compared to a target subset in a subsequent (current) image through use of a correlation function. A shape function consisting of deformation parameters is used to account for displacements and deformations in the subset between the reference and current image. The target subset in the current image is shifted and deformed in order to optimize the correlation score using an iterative process referred to as the subpixel registration algorithm [29, 34]. Optimization using the subpixel registration algorithm solves for the unknown displacement and deformation parameters that relate the reference and target subsets. In each iteration, the target subset is updated by mapping the (x, y) locations in the reference subset to new (x', y') points in the target subset using the shape function parameters. The (x', y') points are typically not at integer locations, and are interpolated based on the chosen subpixel interpolation function [29]. The target subset can then be used to calculate the updated correlation score or used to directly update the deformation parameters for the next iteration [32]. An initial guess is often required for the subpixel registration algorithms to accurately converge, so an integer based pixel search can be used to efficiently determine an initial guess for each reference subset in each image.

The simplest DIC setups make use of a single camera measuring in 2 dimensions perpendicular to the camera's optical axis. This setup is limited to only measuring in 2D and can be accurate only if the image sensor is placed approximately parallel to the surface being measured. To alleviate these constraints, more than one camera can be used to capture multiple images, which can be used to calculate 3 dimensional motion. The calculation of 3D motion requires the use of stereo vision principles and an accurate calibration that captures the translation and rotation between the cameras well. Recently, researchers have also been making inroads into using a single camera for 3D measurements, typically through the use of multiple mirrors or multiple colour channels to allow for the same image to be captured from different positions [67]. These systems still make use of the same stereo principles, but work by segmenting the image sensor into multiple parts rather than using multiple full sensors.

2.1.1 Correlation Criteria

Correlation criteria are used to determine how closely two subsets match. The two subsets being compared consist of a reference subset from the reference image and a target subset from the current analysis image. The optimal displacement and deformation parameters of the shape functions are found through the subpixel registration algorithm using the correlation score as the optimization index. Early works made use of simple sum of squared difference or cross correlation measures [39, 53], but issues could arise if lighting levels changed or subsets crossed areas of high brightness rather than uniform brightness. For example, as a subset crosses a bright area, a significantly higher cross correlation could be registered simply due to the high intensity values found in the bright area despite being potentially quite different. Zero mean normalized cross correlation and zero mean normalized sum of squared difference criteria were developed to alleviate such issues [34].

Sum of Squared Difference

A simple of sum of squared difference can be used as a correlation score, but does not account for shifts in intensity between the various images in a sequence, so a zero mean normalized sum of squared differences (ZNSSD) is more commonly used to account for intensity shifts between images [34]. The optimization goal when determining the shape function parameters using the sum of squared difference is to minimize the difference to 0, which denotes no difference between the reference and current subsets. The most significant advantage of the sum of squared difference criteria is that the optimization goal is to always minimize. The normalized sum of squared differences criteria is shown in Equation 2.1 [34]:

$$C_{ZNSSD} = \sum \left(\frac{f(x,y) - \bar{f}}{\sum (f(x,y) - \bar{f})^2} - \frac{g(x,y) - \bar{g}}{\sum (g(x,y) - \bar{g})^2} \right)^2 \tag{2.1}$$

where $f(x,y)$ is the pixel value in the reference image at point (x,y), $g(x,y)$ is the pixel value in the current image at point (x,y), $\bar{f}$ is the average intensity of the reference subset, denoted by Equation 2.2:

$$\bar{f} = \frac{\sum f(x,y)}{N} \tag{2.2}$$

$\bar{g}$ is the average intensity of the current subset given by Equation 2.3:

$$\bar{g} = \frac{\sum g(x,y)}{N} \tag{2.3}$$

N is the number of pixels in the subset, the summations are performed over the full subset, and the other operations are performed element-by-element. For a typical square subset, $N = n \times n$, where n is the width or height of the subset.

Cross Correlation

Similar to the sum of squared differences, a simple cross correlation measure can be used but does not account for intensity shifts, so a zero mean normalized cross correlation (ZNCC) score is used to account for intensity shifts [36]. The goal of optimization using cross correlation is to maximize the correlation score. The primary advantage of the normalized cross correlation score is that it varies between two known values of 0 and 1. The normalized cross correlation can be used to evaluate the accuracy of any correspondence between a reference and a current subset, and allows for comparison between different points for which has the highest accuracy. In turn, this normalized measure of correspondence allows points to be sorted by level of accuracy, which can then be used as the initial guess in following correlations. The normalized cross correlation score is shown in Equation 2.4 [34]:

$$C_{ZNCC} = \frac{\sum(f(x,y) - \bar{f})(g(x,y) - \bar{g})}{\sqrt{\sum(f(x,y) - \bar{f})^2 \sum(g(x,y) - \bar{g})^2}} \tag{2.4}$$

where $f(x,y)$ is the pixel value in the reference image, $\bar{f}$ is the average intensity of the reference subset given by Equation 2.2, $g(x,y)$ is the pixel value in the current image, $\bar{g}$ is the average intensity of the current subset given by Equation 2.3.

Parametric

The parametric cost function is similar to a simple sum of squared differences, but adds either one or two parameters that account for offset and scale changes in image intensity. The parameters a and b are used to capture the scale and offset changes of the intensity of the target subset, respectively [29, 36]. The parametric cost function is shown in Equation 2.5 [29, 36]:

$$C_{PSSDab} = \sum(af(x,y) + b - g(x,y))^2 \tag{2.5}$$

where $f(x,y)$ is the pixel value in the reference image, $g(x,y)$ is the pixel value in the current image, a is the scale parameter, and b is the offset parameter.

Equivalency of Cost Functions

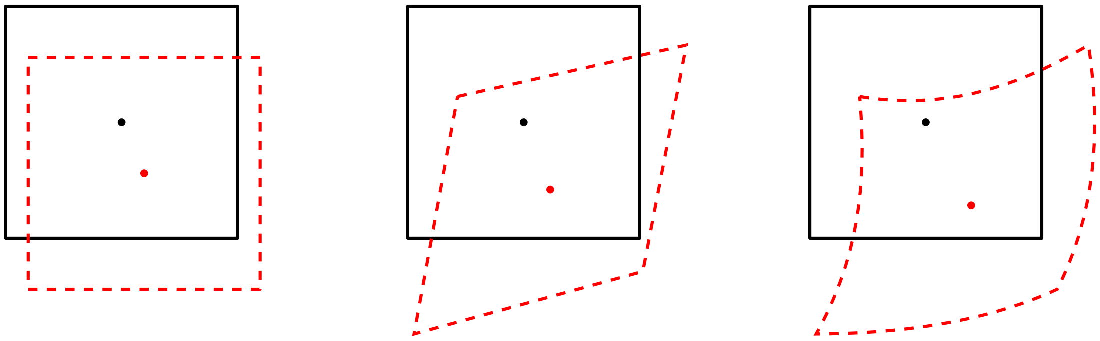
Relationships exist between the cost functions shown in Equations 2.1, 2.4, and 2.5. These equivalencies allow for simplified computation of correlation values for each pixel if required. This can allow for the use of one cost function to optimize the correspondence between subsets, while using another cost function, such as normalized cross correlation, to display the accuracy or to guide which subset should be analyzed next [28]. Equations 2.6 and 2.7 display the relationships between cost functions [36]:

$$C_{ZNSSD} = 2(1 - C_{ZNCC}) \tag{2.6}$$

$$C_{PSSDab} = \sum (g(x,y) - \bar{g})^2 (1 - C_{ZNCC}^2) \tag{2.7}$$

2.1.2 Shape Functions

Shape functions are used in DIC to represent different types of deformations between images and can be derived from a Taylor series expansion of a bidirectional deformation field. The typical shape functions are the 0^{th} order, 1^{st} order, and 2^{nd} order shape functions, with the 1^{st} order shape function being the most popular. A 0^{th} order shape function only captures rigid body displacements. A 1^{st} order shape function captures overall subset displacement, but also allows for the subset to experience shearing and elongation/compression deformations in each direction. Lastly, a 2^{nd} order shape function can account for curvature changes in a subset. Examples of a 0^{th}, 1^{st}, and 2^{nd} order shape function can be seen in Figure 2.1.



(a) 0th order. (b) 1st order. (c) 2nd order.

Figure 2.1: Typical shape functions used in DIC.

The displacement in the horizontal and vertical directions are typically referred to as u and v, respectively. The corresponding elongation and shearing displacements are provided by their first derivatives, where u_x and v_y are the elongations in the horizontal and vertical directions, and u_y and v_x are the shearing deformations in the horizontal and vertical directions. The second order derivatives, u_{xx}, u_{xy}, u_{yy}, v_{xx}, v_{xy}, and v_{yy}, account for the curvature changes. Even for the simplest 0th order shape function, there are at least two displacement variables (u and v) that must be solved for, while the 1st and 2nd order shape functions contain 6 and 12 variables to solve for, respectively. This has led to the use of non-linear optimization algorithms, such as the FA-NR and IC-GN, being commonly used for modern digital image correlation. The 0th order through 2nd order shape functions can be found in Equations 2.8 to 2.10:

$$\begin{aligned} x' &= x + u \\ y' &= y + v \end{aligned} \tag{2.8}$$

$$\begin{aligned} x' &= u_x x + x + u_x y + u \\ y' &= v_x x + v_y y + y + v \end{aligned} \tag{2.9}$$

$$\begin{aligned} x' &= \frac{1}{2}u_{xx}x^2 + u_{xy}xy + \frac{1}{2}u_{yy}y^2 + u_x x + x + u_y y + u \\ y' &= \frac{1}{2}v_{xx}x^2 + v_{xy}xy + \frac{1}{2}v_{yy}y^2 + v_x x + v_y y + y + v \end{aligned} \tag{2.10}$$

where x and y are the horizontal and vertical locations of the point of interest in the reference image, x' and y' are the horizontal and vertical locations of the point of interest in the current image, u and v are the horizontal and vertical shifts, u_x, u_y, v_x, and v_y are the first order partial derivatives of u and v with respect to x and y, and u_{xx}, u_{xy}, u_{yy}, v_{xx}, v_{xy}, and v_{yy} are the second order partial derivatives of u and v with respect to x and y. The first order partial derivatives account for elongation and shearing deformations of the subset, while the second order partial derivatives account for rotational deformations of the subset.

Equation 2.8 clearly shows that only the original coordinate and any displacement in the given direction is used to determine the new coordinate for a given point for 0th order shape functions. A 0th order shape function will work well in cases where a rigid body is translated, but will perform quite poorly in full field, non-rigid displacement tests with deformation of the target subsets. The 1st order shape function will perform much more accurately in these situations due to the presence of the shearing and elongation/compression partial derivative terms. Lastly, the 2nd order shape function shown in Equation 2.10 accounts for the curvature of the function, but can potentially contain additional error compared to the 1st order shape function [24, 60, 66]. A shape function is deemed to be matched to the displacement field when both are of the same order (i.e., a 0th order shape function being used to measure rigid translation). A lower order shape function than the displacement field (i.e., a 0th shape function being used to measure an elongating specimen) is referred to as undermatched, while using a higher order shape function than the displacement field is referred to as being overmatched. Recent work has shown that an overmatched shape function is often preferred when the order of the displacement field is unknown to minimize errors due to undermatching [60, 66].

The basic Taylor series expansion forms of the shape functions shown in Equation 2.8 to 2.10 can be used directly in many of the subpixel registration algorithms, but cannot be directly used in the IC-GN algorithm. The IC-GN algorithm uses a warp function based on the shape function, and the algorithm requires that a warp function matrix be invertible to apply the warp function update as discussed in Section 2.1.4 [4, 14, 32]. Therefore, the warping functions for the 0th to 2nd order shape functions for use in the IC-GN algorithm can be seen in Equations 2.11 to 2.13:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & u \\ 0 & 1 & v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2.11}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1+u_x & u_y & u \\ v_x & 1+v_y & v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2.12}$$

$$\begin{bmatrix} x'^2 \\ x'y' \\ y'^2 \\ x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} A_{00} & A_{01} & A_{02} & 2u(u_x+1) & 2uu_y & u^2 \\ A_{10} & A_{11} & A_{12} & u_xv+uv_x+v & u_yv+uv_y+u & uv \\ A_{20} & A_{21} & A_{22} & 2vv_x & 2v(v_y+1) & v^2 \\ \frac{1}{2}u_{xx} & u_{xy} & \frac{1}{2}u_{yy} & 1+u_x & u_y & u \\ \frac{1}{2}v_{xx} & v_{xy} & \frac{1}{2}v_{yy} & v_x & 1+v_y & v \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^2 \\ xy \\ y^2 \\ x \\ y \\ 1 \end{bmatrix} \tag{2.13}$$

where A_{00} to A_{22} can be found in Equation 2.14:

$$\begin{gathered} A_{00} = 1 + uu_{xx} + u_x(u_x+2) \\ A_{01} = 2u_y(u_x+1) + 2uu_{xy} \\ A_{02} = u_y^2 + uu_{yy} \\ A_{10} = \frac{1}{2}(uv_{xx} + u_{xx}v) + v_x(u_x+1) \\ A_{11} = 1 + u_x + v_y + u_xv_y + u_yv_x + uv_{xy} + u_{xy}v \\ A_{12} = \frac{1}{2}(uv_{yy} + u_{yy}v) + u_y(v_y+1) \\ A_{20} = v_x^2 + vv_{xx} \\ A_{21} = 2v_x(v_y+1) + 2vv_{xy} \\ A_{22} = 1 + vv_{yy} + v_y(v_y+2) \end{gathered} \tag{2.14}$$

The full derivations of the invertible shape functions can be found in Appendix A. The second order shape function shown in Equation 2.13 ignores higher order terms with powers greater than 2 for computation and simplicity [4, 14].

The Jacobian and Hessian matrices of the shape functions are used in recent subpixel registration algorithms to determine the optimal correlation solution. Different shape functions have different forms of the Jacobian, and therefore of the Hessian, although the Hessian is derived from the Jacobian using the same method in all cases. The Jacobian can be found by taking the partial derivative of the warp (shape function) with respect to the incremental deformation parameter, Δp. The Jacobian matrices for the 0th order (2.15a), 1st order (2.15b), and 2nd order (2.15c) shape functions are shown in Equation 2.15:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{2.15a}$$

$$\begin{bmatrix} 1 & \Delta x & \Delta y & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta x & \Delta y \end{bmatrix} \tag{2.15b}$$

$$\begin{bmatrix} 1 & \Delta x & \Delta y & \frac{1}{2}\Delta x^2 & \Delta x\Delta y & \frac{1}{2}\Delta y^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta x & \Delta y & \frac{1}{2}\Delta x^2 & \Delta x\Delta y & \frac{1}{2}\Delta y^2 \end{bmatrix} \tag{2.15c}$$

where Δx and Δy are incremental updates to the x and y locations.

The Jacobian can then be used to find the Hessian of a chosen correlation function with respect to the deformation parameter as shown in Equation 2.16:

$$H = \sum_{x=-M}^{M} \sum_{y=-M}^{M} \left(\left(\nabla f \frac{\partial W}{\partial \Delta p} \right)^T \times \left(\nabla f \frac{\partial W}{\partial \Delta p} \right) \right) \tag{2.16}$$

where ∇f is the gradients of the reference image and is given by $\nabla f = [f_x, f_y]$ and $\frac{\partial W}{\partial \Delta p}$ is the Jacobian given by Equation 2.15. The Hessian is then used in the IC-GN registration algorithm to determine update parameters for the shape function, which can be used to update the target subset for the next iteration.

2.1.3 Subpixel Interpolation Functions

Images are comprised of discretized intensity values, while displacements occur on a continuous spectrum. Displacements typically do not get measured at exact integer pixel shifts, resulting in displacements that are a fraction of a pixel being measured during testing. Subpixel interpolation algorithms allow for measurement of non-integer values, which improves the measurement precision and allows for capture of small displacements and higher frequency vibration. Small movements are quite common in structural engineering, and measuring to a fraction of a millimetre is common with some existing measurement techniques. Subpixel interpolation functions are primarily used in DIC during the registration process when the reference subset coordinates are mapped to updated target subset coordinates. The mapping process rarely results in integer coordinate values, and the interpolation function is then used to provide intensity values between the discrete pixel values. Multiple subpixel interpolation schemes exist, with varying computational requirements, complexity, and accuracy. Examples of interpolation schemes include nearest neighbour, linear interpolation, and b-splines.

Nearest Neighbour

Nearest neighbour is the simplest form of interpolation found in computer graphics and vision. Nearest neighbour interpolation determines the nearest pixel to the point of interest and then assigns that pixel intensity to the point of interest. This process takes very little computational power, but functionally does not improve the resolution of the image. Instead, nearest neighbour interpolation simply changes the discretized signal to a continuously defined piece-wise signal with constant value in the area surrounding each pixel.

Linear Interpolation

Linear interpolation is only slightly more complex than nearest neighbour interpolation, but it is more accurate due to determining values between the grid points based on the surrounding information [54]. In essence, linear interpolation draws a straight line between known values and uses this to find the value at a given subpixel location. When extended to 2 dimensions, linear interpolation is performed in both directions, otherwise known as bilinear interpolation, given by Equation 2.17:

$$f(x,y) = a_0 + a_1x + a_2y + a_3xyf(x,y) = \begin{bmatrix} a_0 & a_1 & a_2 & a_3 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ y \\ xy \end{bmatrix} \tag{2.17}$$

where the a values are the interpolation constants, x is the location in the horizontal direction, and y is the location in the vertical direction. Bilinear interpolation can be thought of as performing a linear interpolation in one direction to find two intermediate values, which are then linearly interpolated between in the other direction (the direction order does not matter). The coefficients present in each grid location can be solved for using Equation 2.18:

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1y_1 \\ 1 & x_1 & y_2 & x_1y_2 \\ 1 & x_2 & y_1 & x_2y_1 \\ 1 & x_2 & y_2 & x_2y_2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} f(Q_{11}) \\ f(Q_{12}) \\ f(Q_{21}) \\ f(Q_{22}) \end{bmatrix} \tag{2.18}$$

where Q_{ij} are the corner pixel locations, $f(Q_{ij})$ is the value at each corner location, x and y are the location of the corners, and the a constants are the interpolation coefficients.

General B-Splines

B-splines are a class of piecewise polynomials for each section of a function between two data points. They are very useful for interpolating between points where the underlying function shape may be unknown due to their versatility. Splines have seen wide use in many fields, including data analysis, computer graphics, and image resampling in popular commercial programs such as Adobe Photoshop. B-spline interpolation in imaging generally takes a separable form wherein operations can be performed on the columns, then rows (or vice versa) separately. This equates to performing a b-spline interpolation along the columns to find values at a given y coordinate followed by interpolating between those y coordinates to find the value at the specified x coordinate in the row. B-splines take the form shown in Equations 2.19 and 2.20 [54, 57, 58]:

$$\beta^0(x) = \begin{cases} 1 & \text{for} |x| < \frac{1}{2} \\ \frac{1}{2} & \text{for} |x| = \frac{1}{2} \\ 0 & \text{for} |x| > \frac{1}{2} \end{cases} \tag{2.19}$$

$$\begin{aligned} \beta^n(x) &= \sum_{k=0}^{n+1} \frac{(-1)^k (n+1)}{(n+1-k)!\,k!} \Big(\frac{n+1}{2} + x - k\Big)_+^n \\ &= \frac{1}{n!} \sum_{k=0}^{n+1} \binom{n+1}{k} (-1)^k \Big(x - k + \frac{n+1}{2}\Big)_+^n \end{aligned} \tag{2.20}$$

where n is the order of b-spline, k is the summation index, x is the distance along the interpolated section, and the + subscript indicates that only the positive portion of the bracket shall be used. This results in a piecewise function that changes at integer points for odd ordered b-splines and at the midpoints for even ordered b-splines. Either version of Equation 2.20 may be used to determine the relevant b-spline equations. A b-spline of order $n = 1$ is equivalent to a linear interpolation, a b-spline of order $n = 3$ is known as a cubic b-spline, and a b-spline of order $n = 5$ is known as a quintic b-spline. In two dimensions, these are known as bilinear, bicubic, and biquintic spline interpolation, respectively. Figure 2.2 displays the kernels of the linear spline, cubic spline, and quintic spline and Figure 2.3 displays bilinear, bicubic, and biquintic interpolations. The bidirectional interpolations are performed by first interpolating in one direction, then interpolating between the results in the other direction. Figure 2.3 interpolates first in the direction of the red lines, followed by between the red lines in the direction of the blue lines.

Cubic B-Spline

The cubic b-spline is the result of setting $n = 3$ in the general b-spline equation given in Equation 2.20. This form of b-spline is more accurate than bilinear interpolation, but is still relatively easy to implement [58]. The high accuracy and easy implementation make it very popular in computer graphics and applications. A bicubic interpolation avoids artifacts such as blocking and smoothing that are present in bilinear and nearest neighbour

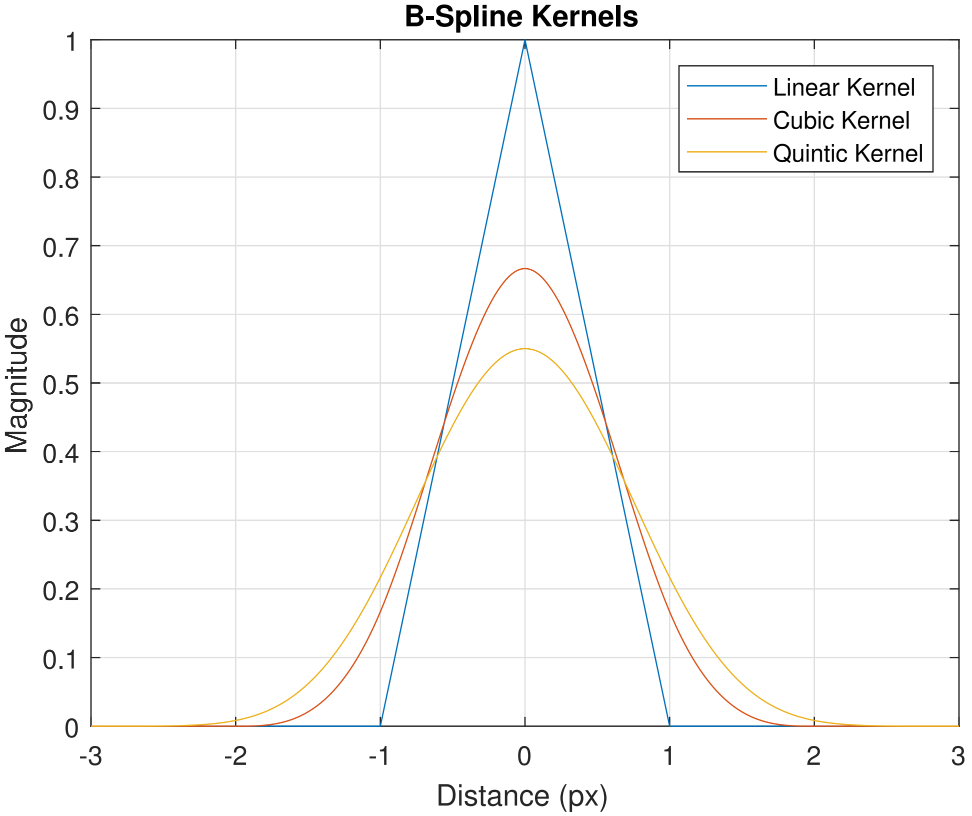
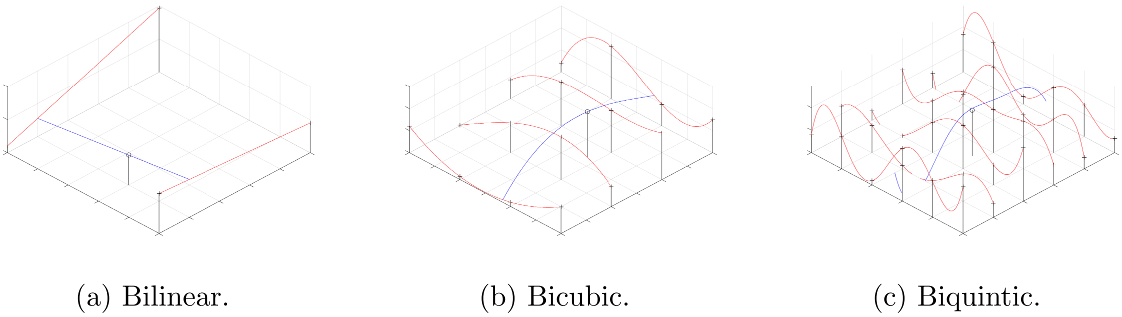


Figure 2.2: B-Spline kernels for linear, cubic, and quintic splines.



(a) Bilinear.

(b) Bicubic.

(c) Biquintic.

Figure 2.3: Bidirectional spline interpolation functions.

interpolation [54]. The expanded form of a cubic b-spline is shown in Equation 2.21:

$$\beta^3 = \begin{cases} \frac{1}{6}x^3 + x^2 + 2x + \frac{4}{3} & -2 \le x < -1 \\ -\frac{1}{2}x^3 - x^2 + \frac{2}{3} & -1 \le x < 0 \\ \frac{1}{2}x^3 - x^2 + \frac{2}{3} & 0 \le x < 1 \\ -\frac{1}{6}x^3 + x^2 - 2x + \frac{4}{3} & 1 \le x < 2 \end{cases} \tag{2.21}$$

The pixel intensity at a given (x, y) location can be found from Equation 2.22:

$$\begin{aligned} p(x,y) &= \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}x^i y^j \\ &= \begin{bmatrix} 1 & x & x^2 & x^3 \end{bmatrix} \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} 1 \\ y \\ y^2 \\ y^3 \end{bmatrix} \end{aligned} \tag{2.22}$$

where x and y are the distances from the center of the kernel. The deterministic solution for the bicubic interpolation coefficients, a_{00} to a_{33}, can be found through Equation 2.23:

$$\begin{aligned} \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} &= A \begin{bmatrix} f(0,0) & f(0,1) & f_y(0,0) & f_y(0,1) \\ f(1,0) & f(1,1) & f_y(1,0) & f_y(1,1) \\ f_x(0,0) & f_x(0,1) & f_{xy}(0,0) & f_{xy}(0,1) \\ f_x(1,0) & f_x(1,1) & f_{xy}(1,0) & f_{xy}(1,1) \end{bmatrix} A^T \\ \text{where } A &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix} \end{aligned} \tag{2.23}$$

a_{00} through a_{33} are the interpolation coefficients, $f(x, y)$ is the image at location (x, y), $f_x(x, y)$ is the x derivative at (x, y), $f_y(x, y)$ is the y derivative at (x, y), and $f_{xy}(x, y)$ is the mixed partial derivative at (x, y). This method requires additional calculations to determine the gradients, which may slow down calculation of the coefficients.

Quintic B-Spline

Quintic b-splines correspond to $n = 5$ in the general b-spline equation shown in Equation 2.20. Quintic b-splines are less common in computer graphics and related fields due to their additional computational intensity. Quintic b-splines are generally more accurate than cubic b-splines [26, 44], but require 6 constants compared to the 4 required for cubic interpolation. Extending to 2D, biquintic interpolation requires 36 constants, while bicubic only requires 16. Quintic b-splines are also more sensitive to high frequency noise than lower order interpolators, and may experience poor results in environments with excessive noise and no noise reduction filtering [30]. The expanded form of the quintic b-spline is shown in Equation 2.24:

$$\beta^5 = \begin{cases} \frac{1}{120}x^5 + \frac{1}{8}x^4 + \frac{3}{4}x^3 + \frac{9}{4}x^2 + \frac{27}{8}x + \frac{81}{40} & -3 \leq x < -2 \\ -\frac{1}{24}x^5 - \frac{3}{8}x^4 - \frac{5}{4}x^3 - \frac{7}{4}x^2 - \frac{5}{8}x + \frac{17}{40} & -2 \leq x < -1 \\ \frac{1}{12}x^5 + \frac{1}{4}x^4 - \frac{1}{2}x^2 + \frac{11}{20} & -1 \leq x < 0 \\ -\frac{1}{12}x^5 + \frac{1}{4}x^4 - \frac{1}{2}x^2 + \frac{11}{20} & 0 \leq x < 1 \\ \frac{1}{24}x^5 - \frac{3}{8}x^4 + \frac{5}{4}x^3 - \frac{7}{4}x^2 + \frac{27}{8}x + \frac{17}{40} & 1 \leq x < 2 \\ -\frac{1}{120}x^5 + \frac{1}{8}x^4 - \frac{3}{4}x^3 + \frac{9}{4}x^2 - \frac{27}{8}x + \frac{81}{40} & 2 \leq x < 3 \end{cases} \tag{2.24}$$

Interpolation can be accomplished using Equation 2.25:

$$\begin{aligned} p(x,y) &= \sum_{i=0}^{5}\sum_{j=0}^{5} a_{ij}x^i y^j \\ &= \begin{bmatrix} 1 & x & x^2 & x^3 & x^4 & x^5 \end{bmatrix} \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} & a_{04} & a_{05} \\ a_{10} & a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{20} & a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{30} & a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{40} & a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{50} & a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{bmatrix} \begin{bmatrix} 1 \\ y \\ y^2 \\ y^3 \\ y^4 \\ y^5 \end{bmatrix} \end{aligned} \tag{2.25}$$

where x and y are the distances from the center of the kernel and a_{00} through a_{55} are the interpolation coefficients.

Efficient B-Spline Interpolation

B-spline interpolation can be performed and stored efficiently using discrete Fourier transforms and resampling matrices [9]. Generally speaking, interpolation can be thought of as a convolution operation as shown in Equation 2.26:

$$p(x,y) = \sum_{i=0}^{n}\sum_{j=0}^{n} a_{ij}\beta^n(x,y) \tag{2.26}$$

where $p(x, y)$ is the pixel intensity values at each (x, y) location in the image that is having its coefficients calculated, a_{ij} is the b-spline coefficients, β^n is the b-spline kernel used in convolution (i.e., linear, cubic, or quintic). A convolution operation on large images can be slow, but converting to the Fourier domain allows the convolution operation to become an element by element multiplication, which can be performed rapidly. The coefficients, a_{ij}, can then be found via the deconvolution shown in Equation 2.27 [9]:

$$\begin{aligned} \mathcal{F}(p) &= \mathcal{F}(a) \star \mathcal{F}(\beta) \\ \mathcal{F}(a) &= \frac{\mathcal{F}(p)}{\mathcal{F}(\beta)} \\ a &= \mathcal{F}^{-1}\left(\frac{\mathcal{F}(p)}{\mathcal{F}(\beta)}\right) \end{aligned} \tag{2.27}$$

where $\mathcal{F}$ denotes a Fourier transform, $\mathcal{F}^{-1}$ denotes the inverse Fourier transform, and the division is performed element by element. The b-spline kernel used in deconvolution can be found by sampling the b-spline curves shown in Figure 2.2 at integer locations. This results in the kernels, b_0, for bilinear, bicubic, and biquintic interpolation shown in Equation 2.28:

$$\begin{aligned} b_0^1 &= \begin{bmatrix} 1 \end{bmatrix} \\ b_0^3 &= \begin{bmatrix} \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{bmatrix} \\ b_0^5 &= \begin{bmatrix} \frac{1}{120} & \frac{13}{60} & \frac{11}{20} & \frac{13}{60} & \frac{1}{120} \end{bmatrix} \end{aligned} \tag{2.28}$$

The kernels are typically zero padded to the length of the image for use in the deconvolution process.

Sampling using the kernels shown in Equation 2.28 corresponds to the value at a location if the image were sampled with no subpixel shift. To account for shifts, the resampling matrices shown in Equation 2.29:

$$\begin{aligned} Q^1 &= \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \\ Q^3 &= \begin{bmatrix} \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & -1 & \frac{1}{2} & 0 \\ -\frac{1}{6} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{6} \end{bmatrix} \\ Q^5 &= \begin{bmatrix} \frac{1}{120} & \frac{13}{60} & \frac{11}{20} & \frac{13}{60} & \frac{1}{120} & 0 \\ -\frac{1}{24} & -\frac{5}{12} & 0 & \frac{5}{12} & \frac{1}{24} & 0 \\ \frac{1}{12} & \frac{1}{6} & -\frac{1}{2} & \frac{1}{6} & \frac{1}{12} & 0 \\ -\frac{1}{12} & \frac{1}{6} & 0 & -\frac{1}{6} & \frac{1}{12} & 0 \\ \frac{1}{24} & -\frac{1}{6} & \frac{1}{4} & -\frac{1}{6} & \frac{1}{24} & 0 \\ -\frac{1}{120} & \frac{1}{24} & -\frac{1}{12} & \frac{1}{24} & -\frac{1}{24} & \frac{1}{120} \end{bmatrix} \end{aligned} \tag{2.29}$$

can pre- and post-multiply the coefficients obtained from the b-spline kernel. This resamples the kernel at the given subpixel shifts of Δx and Δy. The resampling matrices may be derived by imposing a given shift of Δx or Δy on the b-spline kernel and determining the new sampling values as a function of the shift [9]. The coefficients of these matrices corresponding to Δx from the 0^{th} up to the 5^{th} order make up the coefficients of the resampling matrices. The derivations for each of the resampling matrices can be found in Appendix A. Following this, the value at a given subpixel location can be found from Equation 2.30:

$$p(x,y) = \tilde{x} Q C Q^T \tilde{y} \tag{2.30}$$

where Q is given above, C is the coefficient matrix derived from deconvolving the image and relevant b_0 kernel, and $\tilde{x}$ and $\tilde{y}$ are given by Equation 2.31:

$$\begin{aligned} \tilde{x} &= \begin{bmatrix} 1 & \Delta x \end{bmatrix} \\ \tilde{y} &= \begin{bmatrix} 1 & \Delta y \end{bmatrix} \\ \tilde{x} &= \begin{bmatrix} 1 & \Delta x & (\Delta x)^2 & (\Delta x)^3 \end{bmatrix} \\ \tilde{y} &= \begin{bmatrix} 1 & \Delta y & (\Delta y)^2 & (\Delta y)^3 \end{bmatrix} \\ \tilde{x} &= \begin{bmatrix} 1 & \Delta x & (\Delta x)^2 & (\Delta x)^3 & (\Delta x)^4 & (\Delta x)^5 \end{bmatrix} \\ \tilde{y} &= \begin{bmatrix} 1 & \Delta y & (\Delta y)^2 & (\Delta y)^3 & (\Delta y)^4 & (\Delta y)^5 \end{bmatrix} \end{aligned} \tag{2.31}$$

Equation 2.30 can be thought of as resampling along the x direction, followed by resampling along the y direction, similar to how the interpolation can be separated to operate in the x and y directions separately.

2.1.4 Subpixel Registration Algorithms

Correlation algorithms, also known as subpixel registration algorithms, are aimed at providing the ability to converge rapidly and accurately to the peak correlation value between two image subsets. Various subpixel registration algorithms were used by early researchers. These included the FA-NR [10, 29, 31, 44, 52], coarse-fine searching [13, 25, 39, 53, 70], gradient based methods [38, 71], curve fitting, and more. The more popular methods included the FA-NR and gradient based methods due to their accuracy and simplicity [20]. More recently, the IC-GN algorithm was developed and shown to be more computationally efficient and more robust to noise than the classic FA-NR algorithm [32, 35].

Coarse-Fine Search

The coarse-fine search algorithm was an early method to perform DIC prior to the mainstream usage of non-linear iterative algorithms such as the FA-NR [13, 25, 39, 53]. Early researchers found coarse-fine algorithms easy to implement on processors with relatively limited capabilities [13, 25, 39, 53]. The coarse-fine search algorithm worked by creating a grid of possible displacement locations in an image, and a reference subset was compared to target subsets generated at each of the possible grid points to find the maximum correlation. If this correlation was not high enough, a finer search grid was created around the point of maximum correlation. This process was repeated until a correlation or other difference threshold was achieved [53]. If a non-rigid shape function was chosen, then the process would be completed first for the overall rigid displacement, followed by sequentially finding the optimal values for each deformation parameter pair (such as expansion/contraction parameters u_x and v_y together) using a matrix of values [53]. If the deformation parameter pair did not achieve a required correlation score, then the range of the deformation parameters was centered about the optimal point and the range of values was decreased. This was repeated until each deformation parameter also converged [53]. Interpolation can be used throughout this process to provide for grid point values between pixels [53]. The first level of a coarse-fine search may also be used by other methods to estimate the integer pixel displacement for use as an initial guess, followed by use of the other registration algorithms [70].

Gradient Based Methods

Gradient based methods were initially developed as a form of optical flow [38, 71]. Gradient based methods necessitate the assumption that, if a given subset is small enough, that subset will translate as a rigid body and not deform [20, 38]. By doing so, the deformation field can be assumed to take the form of a 0^{th} order shape function shown in Equation 2.32 [38]:

$$g(x', y') = g(x + u, y + v) + \Delta x g_x(x + u, y + v) + \Delta y g_y(x + u, y + v) \tag{2.32}$$

where $g(x', y')$ is the image intensity in the current image at updated point (x', y'), $g(x + u, y + v)$ is the image intensity in the current image at integer location $(x+u, y+v)$, g_x and g_y are the x and y image gradients, and Δx and Δy are the subpixel displacement components of the displacement. The nearest integer location can be found through integer pixel based correlation, but the subpixel component requires a further solution. The solution to Equation 2.32 for the subpixel displacement components can be given in a closed form in terms of g, g_x, and g_y, allowing for a convenient solution for the subpixel displacement components.

Curve Fitting

The curve fitting method of DIC starts by finding the closest matched integer pixel location to the reference subset. The algorithm then treats that point, as well as the surrounding 8 points, as a curved surface that can be fitted using a two dimensional quadratic given by Equation 2.33 [38]:

$$C(x, y) = a_0 + a_1 x + a_2 y + a_3 x^2 + a_4 xy + a_5 y^2 \tag{2.33}$$

where a_0 through a_5 are the curve fitting parameters and point (x, y) is the center of the deformed subset. The maximum correlation location can be found through an optimization process, which involves taking derivatives with respect to x and y, followed by solving the resulting linear system of equations for the center of the deformed subset, (x, y). u and v can then be found from $u = x - x_0$ and $v = y - y_0$, where x_0 and y_0 are the centers of the undeformed image [38].

Forward Additive Newton-Raphson (FA-NR)

While early DIC researchers often made use of the coarse-fine search method, researchers soon began using a non-linear iterative solver to perform DIC. The classic non-linear iterative algorithm used for DIC has been the Forward Additive Newton-Raphson (FA-NR) algorithm, which was widely seen as the gold-standard of DIC until the early 2010s [10, 29, 31, 44, 52]. In general, it has been found to give very high accuracy, but at the cost of high computational requirements. These requirements stem from calculating the Hessian matrix on every iteration, followed by inverting it. The Hessian is typically comprised of a 2x2, 6x6, or 12x12 matrix corresponding to the 0^{th}, 1^{st}, and 2^{nd} order shape functions. This calculation and inversion of up to a 12x12 matrix is very computationally expensive.

Solving for the optimal solution using the FA-NR algorithm requires the calculation and updating of a displacement parameter, p. The size of p varies with the shape function chosen to represent the subset, with a typical first order approximation taking the form of $p = \begin{bmatrix} u & u_x & u_y & v & v_x & v_y \end{bmatrix}$ [29, 31]. Equation 2.34 shows the parameter updating algorithm of the FA-NR method [29, 31]:

$$\begin{aligned} p_n &= p_{n-1} - \Big(\nabla\nabla C(p_{n-1})\Big)^{-1} \nabla C(p_{n-1}) \\ &= p_{n-1} + \Delta p \end{aligned} \tag{2.34}$$

where p_{n-1} is the previous iteration parameter, p_n is the current iteration parameter, $\nabla C(p_{n-1})$ are the gradients of the correlation coefficient, and $\nabla\nabla C(p_{n-1})$ is the second order derivatives of the correlation coefficient, otherwise known as the Hessian. If p_n is close to the exact solution, then the Hessian can be reduced to an approximation as shown in Equation 2.35:

$$\nabla\nabla C(p_{n-1}) = 2 \sum_{i=-M}^{M} \sum_{j=-M}^{M} \left(\frac{1}{(\Delta g)^2} \nabla g(x', y')^T \nabla g(x', y') \right) \tag{2.35}$$

where g refers to the target image, Δg is the standard deviation of the target subset intensities, and $\nabla g(x', y')$ is the gradients of the target subset and is given by $\nabla g = [g_x, g_y]$. The deformation parameter vector from the previous iteration, p_{n-1}, is used to determine the coordinates (x', y') in the target subset on the next iteration.

The FA-NR algorithm can be seen diagrammatically in Figure 2.4. The reference subset, $f(x + \xi)$ is compared with the original target subset, $g(x + \xi)$, where f is the reference image intensities, g is the target image intensities, x is the location of the subset, and ξ is the locations within the subset. The change in the shape (or deformation) parameters, Δp, is calculated and used to find the shape deformation parameters based on Equation 2.34. The deformation parameters are used to update the original target subset to create $g(x + W(\xi; p_{n-1}))$, where $W(\xi; p_{n-1})$ is known as the warp function. The warp function uses the deformation parameters to determine the new (x', y') pixel locations within the subset, which can be used to create the updated target subset using interpolation. This updated target subset is then compared to the reference subset, finding a new Δp, which in turn updates the previously updated target subset to create $g(x + W(\xi; p_n))$. This process is repeated until a convergence criteria is achieved or a maximum iteration count is passed.

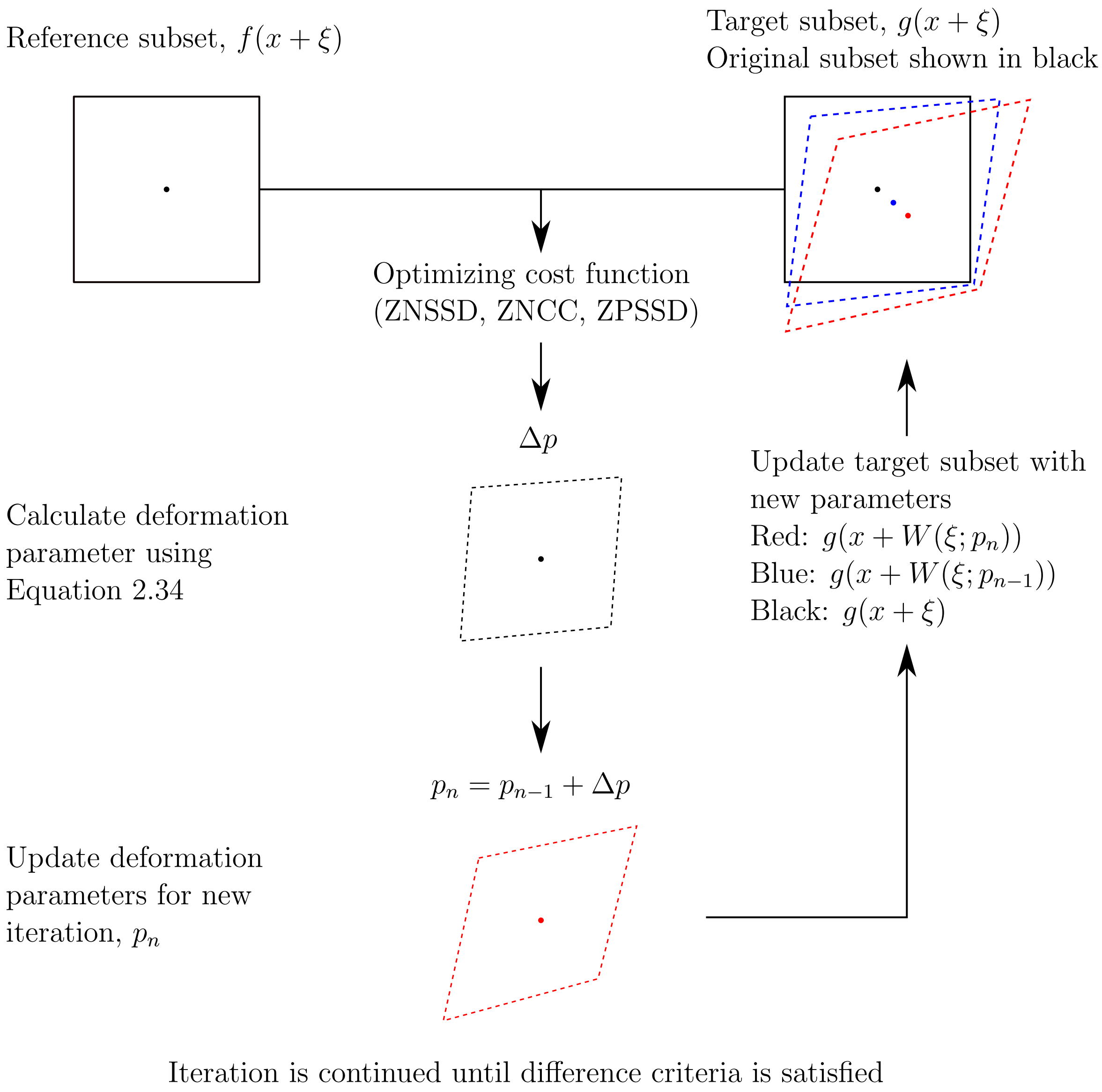

Reference subset, $f(x+\xi)$
Target subset, $g(x+\xi)$
Original subset shown in black
Optimizing cost function
(ZNSSD, ZNCC, ZPSSD)
Δp
Calculate deformation parameter using Equation 2.34
Update target subset with new parameters
Red: $g(x+W(\xi;p_n))$
Blue: $g(x+W(\xi;p_{n-1}))$
Black: $g(x+\xi)$
$p_n = p_{n-1} + \Delta p$
Update deformation parameters for new iteration, p_n
Iteration is continued until difference criteria is satisfied

Figure 2.4: Forward Additive Newton-Raphson algorithm (1st order shape function).

Inverse Compositional Gauss-Newton (IC-GN)

The IC-GN algorithm is a modified version of the FA-NR algorithm that requires fewer computations of the Hessian matrix, and therefore fewer computations of its inverse [32]. Where the FA-NR algorithm warps the target subset to match the reference subset, the IC-GN actually warps the reference subset to match the target subset. The calculated warp is subsequently inverted to find the deformation of the target subset relative to the reference subset. This method can make use of any correlation function, which includes robust criteria such as the C_{ZNSSD} or C_{ZNCC} functions. The IC-GN algorithm requires that both the Hessian and applied warp functions are invertible, and so modifications must be made to the shape function to allow for it to be invertible as discussed in Section 2.1.2 [4, 14, 32]. An incremental warp gets applied to the previous warp function according to Equation 2.36 [32]:

$$W(\xi;p)_n = W(\xi;p)_{n-1} \circ W^{-1}(\xi;\Delta p) \tag{2.36}$$

where $W(\xi;\Delta p)$ is the incremental warp function, $W(\xi;p)$ is the warp function, and n is the current iteration. In general, the algorithm can be seen to follow the process shown in Figure 2.5. The reference subset, $f(x+\xi)$ is compared to the original target subset, $g(x+\xi)$, where f is the reference image intensities, g is the target image intensities, x is the location of the subset, and ξ is the locations within the subset. The reference subset is warped to better match the target subset, and the incremental warp is then inverted to find the incremental warp of the target subset. The incremental warp of the target subset is composed with the previous estimate to give an updated warping parameter, which can be used to build an updated target subset, $g(W(\xi;p)_{n-1})$. The reference subset is then compared to this updated target subset to find a new incremental warp, which is again inverted and composed to find the next updated target subset, $g(W(\xi;p_n)$. This process is repeated until a convergence criteria is achieved or a maximum iteration count is passed. The incremental warp function can be found in each loop by calculating the incremental deformation parameter, Δp, as shown in Equation 2.37 [32]:

$$\Delta p = -H^{-1}\sum_{x=-M}^{M}\sum_{y=-M}^{M}\Big((\nabla f\frac{\partial W}{\partial \Delta p})^T \times [(f(x+\xi)-\bar{f}) - \frac{\Delta f}{\Delta g}(g(x+W(\xi;p))-\bar{g})]\Big) \tag{2.37}$$

where H is the Hessian matrix as described in Section 2.1.2, ∇f is the image intensity gradients in the x and y directions, $\frac{\partial W}{\partial \Delta p}$ is the Jacobian of the cost function with respect to the deformation parameter, Δf and Δg are the standard deviations of the reference and current subset intensities, and $\bar{f}$ and $\bar{g}$ are the means of the reference and current subset intensities [29, 31, 32]. The Hessian and gradients are calculated for the reference subset in the IC-GN algorith, allowing them to be calculated prior to analysis.

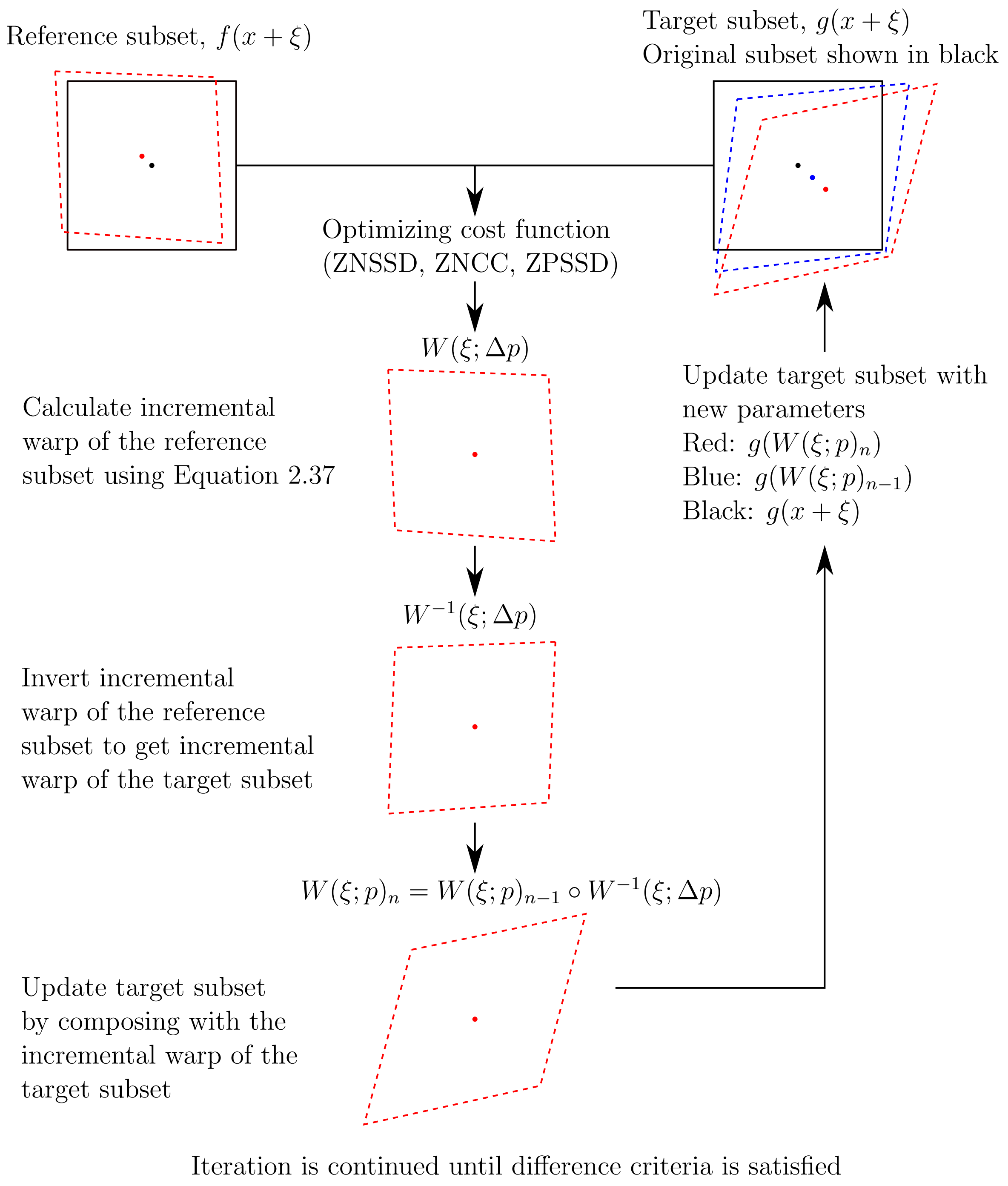

Reference subset, $f(x+\xi)$
Target subset, $g(x+\xi)$
Original subset shown in black
Optimizing cost function
(ZNSSD, ZNCC, ZPSSD)
$W(\xi;\Delta p)$
Calculate incremental
warp of the reference
subset using Equation 2.37
Update target subset with
new parameters
Red: $g(W(\xi;p)_n)$
Blue: $g(W(\xi;p)_{n-1})$
Black: $g(x+\xi)$
$W^{-1}(\xi;\Delta p)$
Invert incremental
warp of the reference
subset to get incremental
warp of the target subset
$W(\xi;p)_n = W(\xi;p)_{n-1} \circ W^{-1}(\xi;\Delta p)$
Update target subset
by composing with the
incremental warp of the
target subset
Iteration is continued until difference criteria is satisfied

Figure 2.5: Inverse Compositional Gauss-Newton algorithm (1st order shape function).

2.2 Digital Image Correlation in the Fourier Domain

DIC methods based in the image domain are quite accurate, but oftentimes are slow to implement due to the iterative nature and point by point calculations. Fourier domain based DIC has been used by a number of researchers to speed up calculation time [7, 15]. This is due to the nature of convolution, which is used in the correlation calculation, being a simple matter of element-wise multiplication in the Fourier domain. The primary limitations of DIC based in the Fourier domain are related to the accuracy found. While subpixel interpolation can be performed through expansion of the image being correlated against [7, 15], interpolation is still somewhat limited by computation power. There are also difficulties in accounting for non-rigid deformations and rotations of the target subset.

2.2.1 Fourier Transform of Digital Images

The Fourier transform of a 2D image is similar in concept to the 1D Fourier transform found in time signal analysis. A typical 1D Fourier transform converts time series data to frequency domain data consisting of a series of complex number that correspond to the amplitude and phase of a signal. This signal is symmetric about a frequency related to the sampling frequency, commonly known as the Shannon-Nyquist frequency [27, 46]. Images are comprised of discrete pixel intensity values, so discrete Fourier transforms (DFTs) are used. The common form of a 1D DFT is given by Equation 2.38:

$$X(a) = \sum_{u=0}^{M-1} x(u) e^{-j2\pi\left(\frac{ua}{M}\right)} \tag{2.38}$$

where $X(a)$ is the discrete Fourier transform of $x(u)$, $x(u)$ is the original discretized function, u is a spatial dimension or time, depending on the signal, and M is the total number of points (or pixels) available.

A 2D Fourier transform is similar, but operates along both dimensions instead of along just one, which is applicable to 2D images. A 2D DFT can also be separated into symmetric quadrants due to two Nyquist frequencies existing (one for each of the horizontal and vertical directions). The common form of a 2D DFT is given by Equation 2.39:

$$X(a,b) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} x(u,v) e^{-j2\pi\left(\frac{ua}{M}+\frac{vb}{N}\right)} \tag{2.39}$$

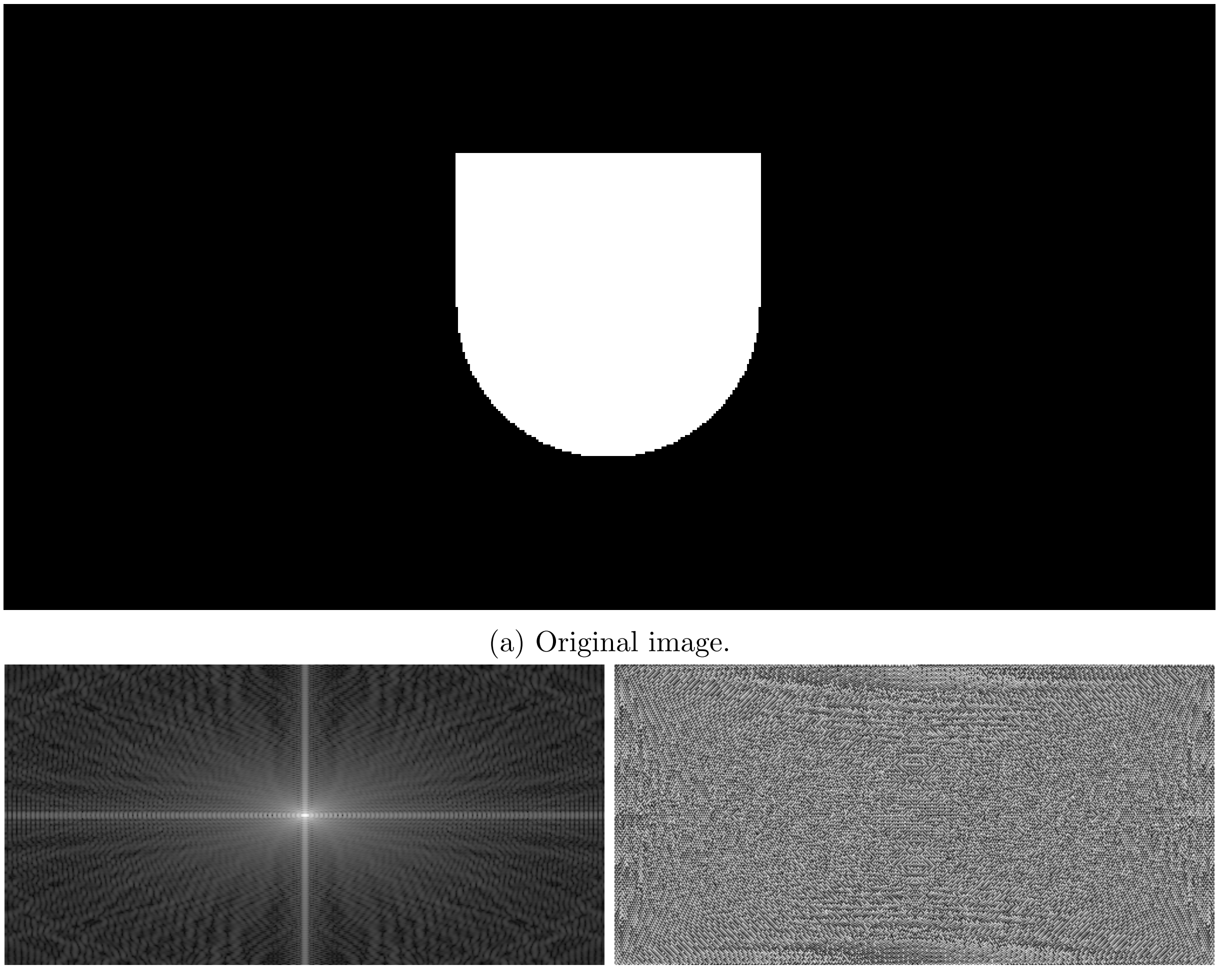
where $X(a,b)$ is the discrete Fourier transform of $x(u,v)$, $x(u,v)$ is the original discretized function or image, M is the number of pixels in the direction of u, and N is the number of pixels in the direction of v.

Figure 2.6 contains an image and associated amplitude and phase plots. Due to the DC component dominating the amplitude plots in Figure 2.6b when linearly scaled, the amplitude plot has been logarithmically scaled. The center of the plot corresponds to the lower frequency components, with the very center being the DC component (or mean intensity of the image). The brightness in the 2D DFT amplitude plot indicates a higher level of that frequency content in the image. Low frequencies refer to subtle or slow changes in pixel intensities, while higher frequencies refer to sudden or sharper changes in pixel intensities. The direction of lines in the amplitude plot also relate to the direction in which the frequency change is occurring; a vertical line of high amplitude frequency components relates to a hard horizontal line in the image. The phase of the 2D DFT shown in Figure 2.6c is difficult to interpret intuitively, but is required for successful reconstruction of an image. It can also be used to perform phase based correlation [7]. Phase based correlation has an advantage in that it takes place along a continuous phase spectrum, allowing subpixel displacement to be calculated in the Fourier domain prior to returning to the image domain. The amplitudes of the various frequencies should not change significantly during displacement provided the average pixel intensity and gradients stay constant, but the phase will change to reflect the displacement of the pixels. However, phase based correlation is only applicable in cases where the subset displaces rigidly. The phase portion of the DFT may also be used to provide subpixel shifts in numerically simulated images [41].

Fourier transforms of 2D images are separable, which allows users to calculate the Fourier transform of the image in each direction independently, saving on computational time and allowing operations to be performed in each direction separately. This can be expressed by Equation 2.40:

$$\begin{aligned} Y(u,b) &= \sum_{v=0}^{N-1} x(u,v)e^{-j2\pi\left(\frac{vb}{N}\right)} \\ X(a,b) &= \sum_{u=0}^{M-1} Y(u,b)e^{-j2\pi\left(\frac{ua}{M}\right)} \end{aligned} \tag{2.40}$$

where $X(a,b)$ is the 2D discrete Fourier transform of $x(u,v)$, $x(u,v)$ is the original discretized function or image, M is the number of pixels in the direction of u, N is the number of pixels in the direction of v, and $Y(u,b)$ is the discrete Fourier transform of $x(u,v)$ applied in only the v direction.



(a) Original image.

(b) Amplitude of DFT.

(c) Phase of DFT.

Figure 2.6: Example of a 2D Fourier transform of an image.

2.2.2 Correlation Using Discrete Fourier Transforms

There are multiple techniques that make use of Fourier transforms to compute the correlation between two images. One method is to use a phase shift difference to find the resulting displacement between two images [7], but this is limited to shifts in just the x or y directions. Rigid body displacements may be applicable for tracking individual targets or if the subsets chosen are small enough, similar to that assumed by the gradient based traditional DIC method. A second method is to make use of convolution to find the peak correlation, but this is limited to finding displacements at only integer pixel values. Methods have been proposed to improve the resolution to subpixel levels [15], including upsampling the entire image using DFTs, upsampling a local neighbourhood around the initial estimate of the peak, and multiple upsampling steps for a local neighbourhood around the initial estimate of the peak [15]. Efficient upsampling techniques can drastically improve speeds, but nonlinear optimization approaches are still found to be more accurate than DFT approaches in general [15].

2.3 Camera Calibration

Camera calibration is a critical process for optical measuring systems. 3D systems require knowledge of the relative camera locations, which can be provided through a stereoscopic calibration process. Calibration can also compensate for various distortions and errors caused by cameras and lenses. Proper calibration can lead to high accuracy measurements, while improper calibration may result in large errors.

Calibration accounts for the intrinsic properties of cameras and the distortions created by lenses. The intrinsic properties of a camera include the focal length of the camera (typically normalized to pixel units), the optical center (which may or may not be coincident with the sensor center), and pixel skew (which is when the camera pixels are not truly square). Calibration can take the form of a single camera calibration, which provides the lens and camera intrinsics of a single camera, or a multiple camera calibration, which provides the camera intrinsics for each camera as well as the external geometric relationship between the two.

The calibration process assumes a pinhole camera model as shown in Figure 2.7. An idealized pinhole camera is one in which the image is passed through a single point aperture to the image sensor with no use of magnification or focusing lenses. A pinhole camera model can use the intrinsic matrix of the camera to convert between the camera coordinate system and the image coordinate system [25]. This model assumes that:

- the aperture is a pinhole without diffraction, and therefore the depth of field is infinite and all points are perfectly focused on the image sensor
- there is no lens present to distort the image radially or tangentially
- the aperture is located along the Z axis projected from the camera center (i.e., the optical center coincides with the center of the sensor)

Camera calibration can be used to compensate for some of these assumptions, such as the lack of radial and tangential distortion and the location of the aperture being along the Z axis.

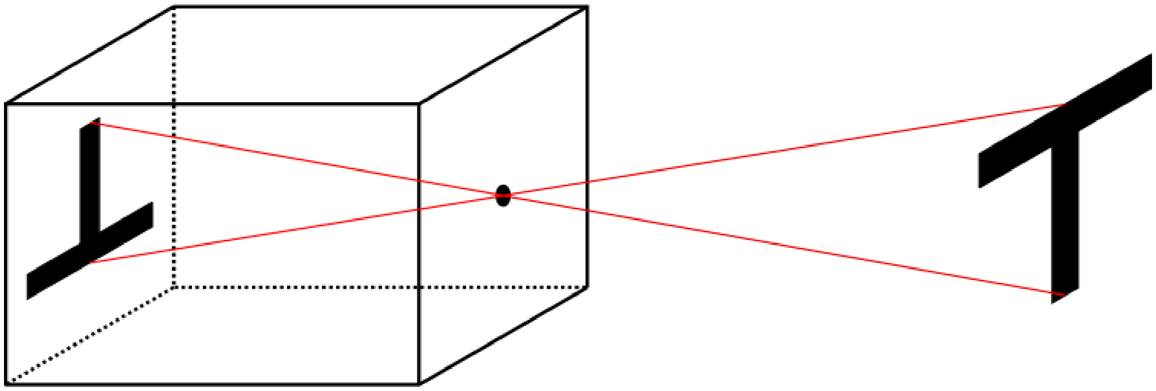


Figure 2.7: Typical pinhole camera model.

Three coordinates systems are present in camera imaging as shown in Figure 2.8. The image coordinates are denoted by (x, y), the camera coordinates are denoted by (X_c, Y_c, Z_c), and the world coordinates are denoted by (X_w, Y_w, Z_w). The image coordinates are expressed in units of pixels (px), while the camera and world coordinates are expressed in real world units such as millimetres (mm). The camera coordinates are measured from the center of the camera, with the Z_c axis projecting outwards through the lens, perpendicular to the imaging plane. The imaging plane is located at the focal distance, f, from the camera center along the Z_c axis. The image coordinates are typically measured from the top corner of the image. The pinhole camera model assumes that all points from the object are projected through the projection center and onto the imaging plane using a straight line [17, 63]. The camera intrinsics are used to convert between the image coordinates and camera coordinates, while extrinsic parameters are used to convert between camera coordinates and world coordinates.

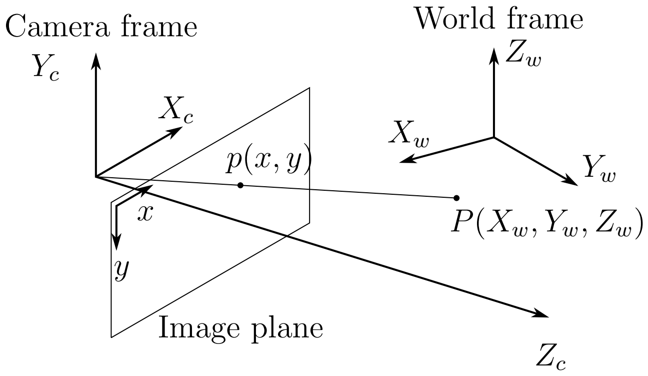


Figure 2.8: Coordinate systems present in pinhole camera model.

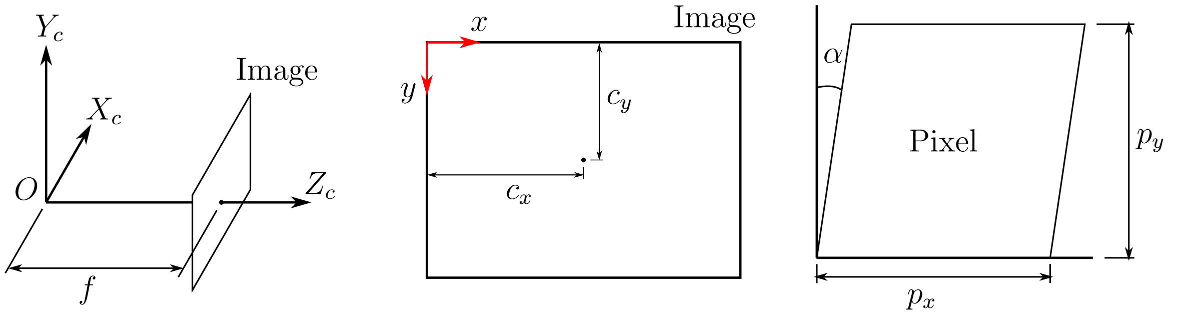
If a given image is located at a focal length f away from the image sensor, then the relationships in Equation 2.41 can be derived geometrically:

$$\begin{gathered}\frac{x}{f} = \frac{X_w}{Z_w} \\ \frac{y}{f} = \frac{Y_w}{Z_w} \\ \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \frac{1}{Z_w} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix}\end{gathered} \tag{2.41}$$

The intrinsic parameters are found through optimizing the reprojection error of a known pattern, such as a checkerboard. The general camera intrinsics matrix can be found in Equation 2.42:

$$K = \begin{bmatrix} f_x & 0 & 0 \\ s & f_y & 0 \\ c_x & c_y & 1 \end{bmatrix} \tag{2.42}$$

where f_x is the normalized focal length in the x direction, f_y is the normalized focal length in the y direction, c_x is the optical center in the x axis, c_y is the optical center in the y axis, and s is the skew coefficient that can be found from $s = f\tan(\alpha)$. Figure 2.9 shows the different parameters. The focal lengths in the x and y directions are typically the same, in particular if the pixels have a square aspect ratio. The focal lengths are normalized relative to the physical size of the pixels, and therefore expressed in units of pixels. The optical center refers to the center of the lens and where the optical, or Z, axis passes through in pixel coordinates. The skew coefficient accounts for if the x and y axes of the camera pixels are not perfectly perpendicular; skew coefficients are occasionally omitted and assumed to be 0. The intrinsics matrix may be transposed depending on the initial formulation, but contains the same information regardless of orientation.



(a) Focal length.

(b) Optical center.

(c) Skew coefficient.

Figure 2.9: Intrinsic camera parameters found through calibration.

The camera calibration also contains information for the camera extrinsics, which measure the rotation and translation relative to a given reference in world coordinates. This allows conversion from world coordinates (X_w, Y_w, Z_w) to 3D camera coordinates (X_c, Y_c, Z_c), at which point the camera intrinsics can convert from 3D camera coordinates to image plane coordinates (x, y). This allows for the perspective transformation given by Equation 2.43:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = K \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_1 \\ R_{21} & R_{22} & R_{23} & t_2 \\ R_{31} & R_{32} & R_{33} & t_3 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.43}$$

Radial and tangential distortions are related to the distortions present in the specific lens/camera combination. Radial distortion occurs when the lens magnifies differently at the outside edges of the frame than the center, preventing straight lines in the real world from remaining straight in captured images. These distortions are assumed to be symmetric about the optical axis [63]. Radial distortion is referred to as barrel distortion when the distortion results in negative radial displacement of image points, and referred to as pincushion distortion when the distortion results in positive radial displacement of image points in the resultant images [63]. Radial distortion can be modelled using Equation 2.44:

$$\begin{aligned} x_{distorted} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_{distorted} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{aligned} \tag{2.44}$$

where $r^2 = x^2 + y^2$, x is the pixel coordinate in the horizontal direction, y is the pixel coordinate in the vertical direction, and k_1 to k_3 are the radial distortion coefficients (k_3 is optional for small distortions). Tangential distortion occurs when the camera sensor and the back of the lens are not planar [17]. Tangential distortion appears as though one corner or side of the image is leaning towards the sensor while the opposing side is leaning away from the sensor. Tangential distortion can be modelled through Equation 2.45:

$$\begin{aligned} x_{distorted} &= x + [2p_1 xy + p_2(r^2 + 2x^2)] \\ y_{distorted} &= y + [p_1(r^2 + 2y^2) + 2p_2 xy] \end{aligned} \tag{2.45}$$

where r, x, and y are as above and p_1 and p_2 are the tangential distortion coefficients.

Multiple cameras can also be calibrated relative to each other for stereoscopic measurement. Additional translation and rotation matrices are determined that are related to the translation and rotation between the two cameras. These translation and rotation matrices can be used to determine the fundamental and essential matrices, which are discussed in Section 2.4. The relationship between the cameras can be used to estimate vision in 3 dimensions, allowing for calculation of X_w, Y_w, and Z_w coordinates based on a given left-right pair of images.

2.4 3 Dimensional Vision

A single camera pointing at an object has difficulty accurately predicting 3 dimensional movements. 3D motion can be estimated by a single camera using changes in the object scale, but any rotation or deformation during capture would prevent accurate measurement. Different scales would also be required for different points or objects. 3D point tracking instead relies on multiple camera positions to determine the location of each point. Figure 2.10 shows how a pair of cameras can determine the location of a point in 3D space, relative to one of the cameras or a reference point. Points O and O' refer to the camera centers, planes I and I' are the imaging planes, x and x' are where the point X appears in each image, e and e' are known as the epipoles, and l and l' are known as the epipolar lines. This concept is generally known as stereo vision or stereoscopy.

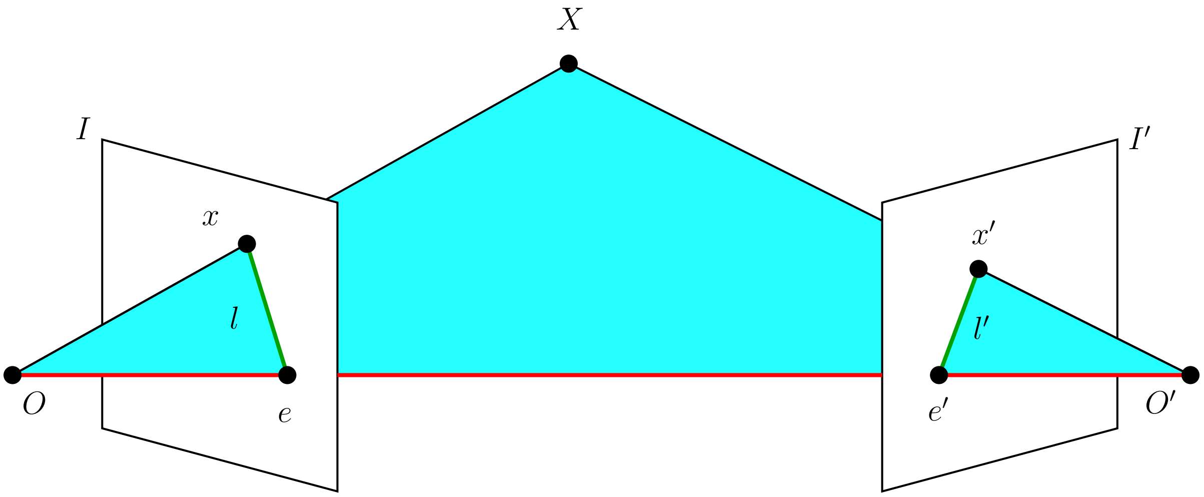


Figure 2.10: Basic two camera layout for three dimensional vision.

2.4.1 Stereoscopy

Humans see in 3D due to each eye seeing slightly different images. The discrepancy between what each eye sees is processed by the brain, resulting in perception of depth and movement in all directions. The same concept can be applied to cameras. In the case of humans, the brain knows the distance between the optic nerves (and thus the baseline of the triangle shown in Figure 2.10) required to perceive depth, but cameras do not inherently know this measure. A stereo calibration must be carried out using both cameras simultaneously in order to minimize the error associated with reprojecting the corrected points. The calibration provides the translation and any rotation between the cameras, which in turn allows the software to calculate the depth to a given point. These parameters are typically described by a 3x3 matrix of rotations shown in Equation 2.46 and a 3x1 vector of translations shown in Equation 2.47:

$$R = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \tag{2.46}$$

$$T = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{2.47}$$

where R_{11} through R_{33} are the rotations of the second camera relative to the first camera and t_x, t_y, and t_z are the translations between the cameras in the x, y, and z directions.

2.4.2 Epipolar Geometry

Epipolar geometry is simply the geometry behind stereo vision, and is related to solely the camera properties and relative pose (i.e., the relative translation and rotation) of the cameras [16]. Epipolar geometry can be subdivided into situations with calibrated cameras and situations with uncalibrated cameras. Cameras may be calibrated either prior to or following capture when conducting digital image correlation, so only the calibrated case will be discussed. The assumptions underlying the calibrated case are that the intrinsic parameters, extrinsic parameters, and relationship between the cameras are known, and that the cameras follow the pinhole camera model.

The simplest case of epipolar geometry is the case where the camera sensors are coplanar with the same lens and only a horizontal shift between the cameras as shown in Figure 2.11. Based on the geometry shown, and assuming that the distance between the cameras is known, the depth can be calculated by Equation 2.48:

$$Z = \frac{fb}{d} \tag{2.48}$$

where Z is the distance from the camera plane, f is the focal length, b is the distance between the cameras, and d is the disparty and is equal to $d = x_R - x_L$. Error can occur in the depth measurement due to the camera sensor discretizing the image as shown in Figure 2.12. The error in disparity between the left and right image can be taken as $\Delta d = \Delta x_R - \Delta x_L$. Assuming the error in each of Δx_R and Δx_L is independent and has a mean of 0, the mean of the disparity error can be found to be 0 as well from Equation 2.49 [16]:

$$\begin{gathered} \Delta d = \Delta x_R - \Delta x_L \\ E[\Delta d] = E[\Delta x_R] - E[\Delta x_L] = 0 - 0 = 0 \end{gathered} \tag{2.49}$$

The depth, Z, contains no uncertainty due to focal length or distance between cameras because the focal lengths do not change and the cameras are stationary. Therefore, the depth has uncertainty only due to the disparity, d, which allows the error in depth, ΔZ,

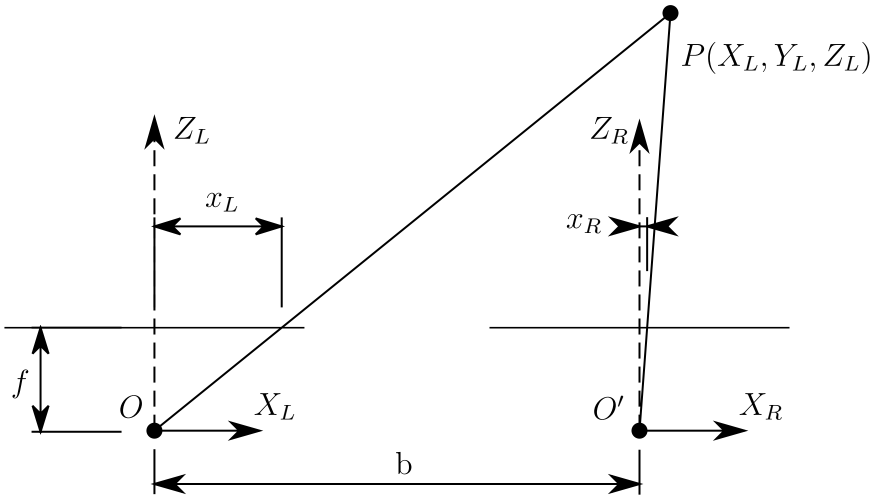


Figure 2.11: Simple case of epipolar geometry with co-planar camera sensors.

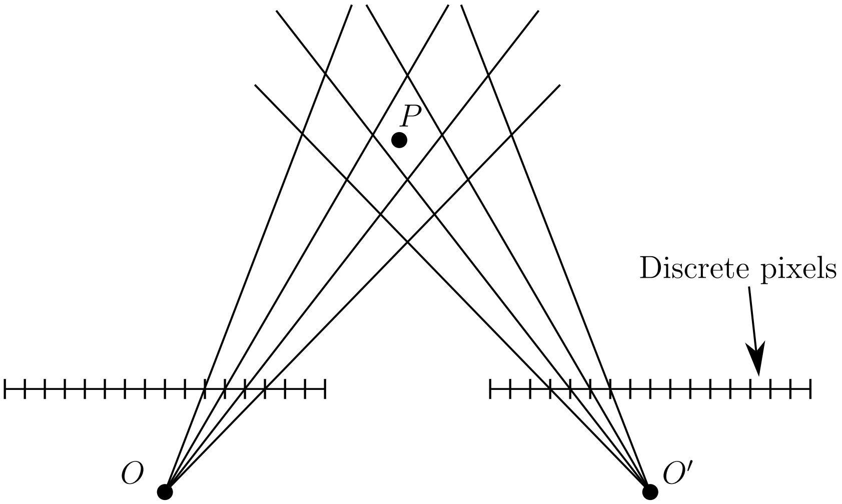


Figure 2.12: Error due to quantization of camera sensor.

to be found from Equation 2.50. The mean error in depth can also be found to be 0 from Equation 2.50 [16]:

$$\begin{aligned} \Delta Z &= -\frac{fb}{d^2}\Delta d \\ E[\Delta Z] &= -\frac{fb}{d^2}E[\Delta d] = -\frac{fb}{d^2} * 0 = 0 \end{aligned} \tag{2.50}$$

Fundamental Matrix

The fundamental matrix, F, has properties derived from epipolar geometry, and may be thought of the algebraic version of epipolar geometry [16]. Figure 2.13 shows that for every point, x, in the left image there exists an epipolar line, l' in the right image (and vice versa). The ray from the point of interest, X, to the image plane location, x and passing through the camera center can then be projected to the other image plane, resulting in the epipolar line, l'. Therefore, a transformation from the location in the left image to the corresponding epipolar line in the right image exists [16].

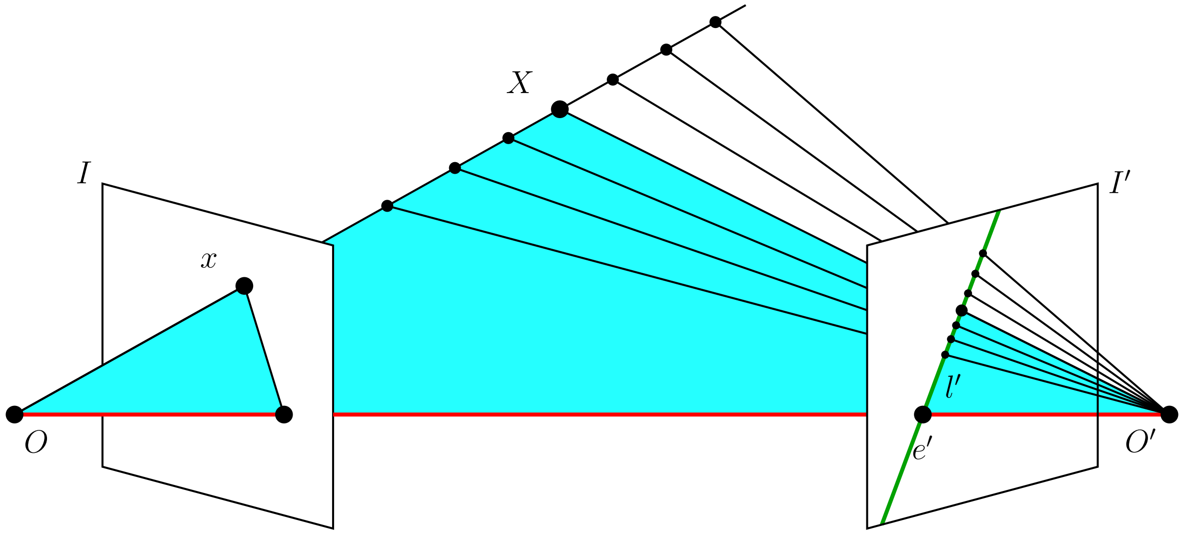


Figure 2.13: Epipolar lines resulting from a point, x.

Assuming a 2D homography, H_π, between the left and right images that transfers points from x to x' such that $x' = H_\pi x$, then the epipolar line l' passing through x' and e' can be written as Equation 2.51 [16]:

$$\begin{aligned} l' &= e' \times x' = [e']_\times x' \\ l' &= [e']_\times H_\pi x = Fx \end{aligned} \tag{2.51}$$

where $[e']_\times$ refers to the skew symmetric matrix for e' that provides the equivalent multiplication as the cross product [16]. Equation 2.51 can be rewritten by substituting $x' = H_\pi x$, which can in turn be rewritten to find a matrix, F, otherwise known as the fundamental matrix [16]. The fundamental matrix is essentially a mapping from the 2D plane of the first image to determine the epipolar line through the second image epipole, e'.

The fundamental matrix has the property that for any and all pairs of corresponding points in the two images, $x'^T Fx = 0$, and the rays defined by these points are coplanar [16]. The fundamental matrix has seven degrees of freedom, and hence requires at least seven points of known correspondence to calculate [16].

The essential matrix is closely related to the fundamental matrix. It is a special case of the fundamental matrix where it is expressed in normalized image coordinates [16]. The fundamental matrix does not assume that the cameras are calibrated, but the essential matrix requires that the cameras be calibrated [16]. The normalized coordinates are expressed by Equation 2.52:

$$E = [t]_\times R \tag{2.52}$$

where R is the rotation matrix, and $[t]_\times$ is the skew-symmetric matrix for the translation vector, t, that provides the equivalent multiplication as the cross product.

If a camera matrix is given by $P = K[R|t]$ and a point x is given by $x = PX$, then the normalized coordinates $\hat{x}$ can be given by $\hat{x} = K^{-1}x = [R|t]X$. Equation 2.52 can be found by taking the normalized camera matrices $P = [I_3|0_3]$ and $P' = [R|t]$ for the fundamental matrix shown in Equation 2.51 [16], where I_3 is the 3x3 identity matrix and 0_3 is a 3x1 0 vector. Similarly to the fundamental matrix, a key property of the essential matrix is that $\hat{x}'^T E\hat{x} = 0$ for all normalized, corresponding points $\hat{x}$ and $\hat{x}'$. This leads to Equation 2.53 [16]:

$$\begin{gathered} \hat{x}'^T E\hat{x} = 0 \\ x'^T K'^{-1} E K^{-1} x = 0 \\ x'^T K'^{-1} E K^{-1} x = x'^T Fx \\ E = K'^T FK \end{gathered} \tag{2.53}$$

where K is the camera intrinsics matrix. Camera calibration software, such as that provided with MATLAB, can be used in conjunction with a checkerboard pattern to determine the fundamental and essential matrices.

2.5 Noise Reduction

2.5.1 Noise Sources

An ideal camera with ideal lighting would result in no noise. However, cameras have a number of noise sources, which contaminate the signal coming from the camera sensor. These noise sources include shot noise, quantization noise, and read noise.

Shot noise is related to the number of photons that actually strike the image sensor. The number of photons given off by a constant light source is not constant in every direction, leading to variation in the number of photons that strike the sensor throughout a test, resulting in some variation in the signal. As the number of photons striking the sensor increases, the variation decreases in relative magnitude and there is a more consistent result that more closely matches the true brightness of the pixel. Therefore, brighter light sources are generally recommended to reduce shot noise.

Quantization noise is the result of cameras measuring in discrete intervals, such as from 0 to 255 (8-bit), which results in round off errors between the true, continuous signal and the reported, quantized signal. The quantization error is then the difference between the true signal and the rounded signal. Quantization error can be reduced in relative magnitude through the use of 10- or 12-bit images.

Read noise is related to a number of factors, including the ambient temperature, camera temperature, and how many electrons are actually created by a given number of photons striking the imaging sensor. Each photon striking the image sensor would generate a single electron in a perfect sensor, which would then be converted to a voltage signal. This voltage is converted to a digital signal that is related to the raw value at that point. Thermal shifts and interference from the surrounding environment can also introduce errors in the analog to digital conversion of the voltage signal to a pixel value. The analog to digital converter should ideally be shielded to limit interference and the cameras left to reach a steady state temperature prior to measuring. The read noise can be determined at a given steady state temperature by taking a very short exposure image with the lens cap on and any external lights off.

Another form of noise, commonly known as salt and pepper noise, is when dark pixels appear in bright areas and bright pixels appear in dark areas. Salt and pepper noise may be caused by stuck or dead pixels, or it may be caused by other components always reading a full or empty charge regardless of actual light level. It can be mitigated through the use of non-linear median filtering.

2.5.2 Gaussian Pre-Filtering

High order b-spline interpolants, such as quintic b-splines, have been found to be highly accurate in numerical simulations without the presence of noise [30]. These higher order b-splines interpolants are sensitive to high frequency noise which can add both bias error and variability to the displacement measurements. A larger subset can be used to reduce the variability, but it is unable to affect the bias error [30].

Pan [30] investigated the use of low pass Gaussian filters to reduce the bias error when using higher order b-splines interpolants. Pan [30] examined numerically generated images with varying noise levels as well as experimental images using the FA-NR algorithm before and after applying a smoothing Gaussian filter. The filters varied in size from 3x3 to 9x9 pixels and were applied to all images. The application of a Gaussian filter acts as a blurring effect and reduces the image gradients, which reduces the accuracy of the algorithm as discussed in Section 2.6. All of the Gaussian filters reduced the bias errors of the analysis, with the 3x3 filter having the lowest effect and the 5x5, 7x7, and 9x9 filters all having similar effects [30]. The larger filters reduced the detail though, which increased the variability (or standard deviation error) of the results. Therefore, Pan [30] recommended the use of a 5x5 Gaussian filter to remove bias error while limiting the standard deviation error increase. The application of the 5x5 filter reduced bias error to near 0 while only marginally increasing the standard deviation error. This was further verified by experimental testing, and also showed that Gaussian pre-filtering combined with a bicubic interpolation scheme could be as accurate as biquintic interpolation regardless of whether the biquintic interpolation used pre-filtering [30].

2.6 Speckle Patterns

DIC requires a unique pattern or shape to be present in an image to accurately track points. In the case where only a single or small number of points are being tracked, targets or natural features on the structures can be used for tracking. Full field displacements, however, require a large number of unique features dispersed across a given area. These unique features are typically created artificially using a painted on speckle pattern, although it is possible to use naturally occurring features in highly variable materials, such as rough brick or masonry [3]. An example speckle pattern is shown in Figure 2.14.

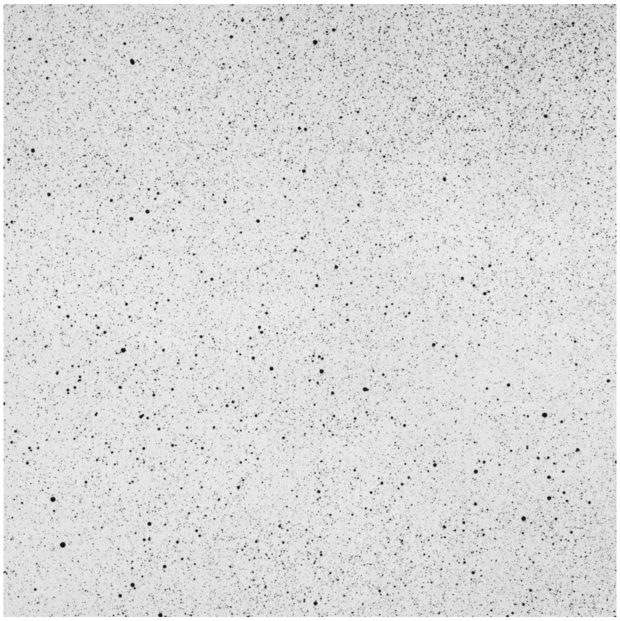


Figure 2.14: Example of a speckle pattern.

The quality of the speckle pattern applied to an object can have a significant impact on the accuracy of the resulting DIC measurements [33]. Poor speckle patterns can lead to significantly increased errors, resulting in very different measured displacement fields for the same specimen [33]. If the speckle pattern is smudged or blurred, it may not provide enough unique features to allow the algorithm to match accurately. If the speckles are too large, they may take up too much of the subset area and consequently result in false matches or lack of convergence. Finally, if the speckles are too small, they may not appear as unique features in the pixels due to them being much much smaller than the pixels. High contrast, such as black speckles on a white background, should be used in speckle patterns to ensure high accuracy [37].

There are numerous ways of quantitatively evaluating the quality of the speckles. Local methods include subset entropy [64] and the sum of squares of subset intensity gradient (SSSIG) [37]. The SSSIG criteria can be seen in Equation 2.54 [37]:

$$\begin{aligned} SSSIG_x &= \sum (f_x)^2 \\ SSSIG_y &= \sum (f_y)^2 \end{aligned} \tag{2.54}$$

where f_x is the gradient of the subset in the x direction, f_y is the gradient of the subset in the y direction, and the summation is performed over the entire subset. For a randomly distributed speckle pattern, the SSSIG should be approximately equal in each direction. If the SSSIG is too small, there is not enough variation in the subset to ensure accurate and unique matches, resulting in larger errors [37]. The SSSIG can be increased by increasing the contrast of the speckle pattern, increasing the bit depth of the images (i.e., using 10- or 12-bit images instead of 8- bit images), or decreasing the noise in an image. A recommended subset size can also be determined by calculating the smallest subset that achieves a threshold SSSIG value for correlation.

An alternative to local subset based speckle evaluation is to use global speckle pattern quality measures. One form of global quality evaluation is the mean intensity gradient (MIG) [29, 33]. The MIG is conceptually similar to the SSSIG, but is an average of the gradients of the whole region of interest rather than of an individual subset. The MIG can be calculated by Equation 2.55:

$$MIG = \sum \frac{|\nabla f|}{WH} \tag{2.55}$$

where $|\nabla f|= \sqrt{(f_x)^2 + (f_y)^2}$, W is the width of the region of interest, and H is the height of the region of interest. If the MIG is too low, it is a sign that there is not enough variation in the subset to ensure accurate matches [33]. The SSSIG of a local subset can also be approximated from the MIG according to Equation 2.56 [29]:

$$\sqrt{SSSIG} \approx N \times MIG \tag{2.56}$$

where N is the size of a subset, and the sum is performed over the entire region of interest or image. The MIG and SSSIG are easily implemented and valuable in evaluating and comparing different speckle patterns or images.

2.6.1 Speckle Pattern Simulation

A number of researchers have proposed different ways to simulate speckle patterns. Zhou and Goodson [71] used various numbers and sizes of speckles and found that the best results were achieved by speckle sizes between 2 and 5 pixels. Zhou and Goodson [71] chose a speckle size and number and then randomly assigned locations to each of the speckles, which were then used to build the image. Undersampling of the image occurred when a speckle size of 1px or smaller was used, which resulted in increased errors. Larger speckles were seen to perform well, but only to an upper limit, at which point errors again rose [71]. The ideal range of 2 to 5 pixels when using the FA-NR generally agreed with previous experimental results showing 2 to 3 pixels was ideal [10]. Other authors have also suggested a range of 2 to 7 pixels in previous years [68].

2.7 Applications of DIC

DIC has been used for applications ranging from simple 2D tensile tests to large scale, 3D tests on full scale structures like bridges and buildings. Most applications in civil engineering to date have used commercial software such as VIC-2D and VIC-3D from Correlated Solutions, Istra4D from Dantec Dynamics, ARAMIS from GOM, Video Gauge from Imetrum, and MatchID, although others have made use of open-source or research oriented software such as NCORR, DICe, GeoPIV, and YADICS. A very small number of research groups have created their own software packages for internal use or development as well.

2.7.1 Laboratory Usage

The earliest work in DIC was typically completed in 2 dimensions, and was primarily focused on laboratory work. One of the earliest forms of DIC was used by Peters and Ranson [39], who used an early form of DIC that was able to account for a rigid body translation using a coarse-fine search method. They verified their algorithms by measuring a small aluminum plate under uniform tension. A shift of 0.1419 inches was found before the images could not correlate [39]. Sutton et al [53] incorporated an early bilinear interpolation scheme. This system also made use of a 1st order shape function, allowing it to track shearing and elongation as well as rigid body displacement [53]. Sutton et al [53] made use a form of coarse-fine search where the initial parameters were initially estimated using a coarse grid, and were estimated in pairs (u and v, u_x and v_y, and u_y and v_x). The estimates were then refined in a local area. A validation test performed on a cantilever found results close to beam theory, with errors less than 5% [53]. Horizontal displacements were found to be very inaccurate due to the small displacements, and the authors estimated a 0.10px threshold for accurate measurement in their system. Chu, Ranson, and Sutton [13] expanded on the interpolation used by Sutton et al [53] by using both polynomial and bilinear interpolation in 2D DIC. They additionally derived the finite strain equations that can be used to determine the 2D strains (ϵ_{xx}, ϵ_{yy}, ϵ_{xy}) directly from the displacement fields [13]. They found that polynomial interpolation smoothed the data better than bilinear interpolation, resulting in lowered frequency content, and that bilinear interpolation was more accurate than polynomial. The researchers made use of a similar coarse-fine search as Sutton et al [53]. Two specimens were tested in pure translation and pure rotation using DIC and compared to the results from a dial indicator, and errors were below 10% for strain values from 0 to 0.03 [13]. Additional work by Sutton et al [52] examined the use of smoothing functions for 2D displacement fields, and found a point-to-point strain error in the order of 150 $\mu\epsilon$, although the average strain error was in the order of 30 $\mu\epsilon$. The displacement noise was found to be approximately 0.01px [52].

An early 3D DIC system was developed by Luo et al [25]. Stereo vision was used for 3D vision through the use of a stereo pair of CCD cameras. The cameras were calibrated, so the camera parameters and relative pose were used to determine the 3D coordinates, and DIC algorithms with an accuracy within 0.05px were used to determine the shifts within image planes and to match paired subsets. Luo et al conducted experimental tests on both a cantilever beam, which matched closely to theory, and a cracked 304L stainless steel specimen.

Zhou and Goodson [71] developed a DIC method that used a non-linear iterative solver and optical flow based interpolation to determine displacements. They also numerically simulated speckle patterns by randomly assigning locations of speckles of various sizes. Speckle sizes ranging from 1 to 14px were examined, with the number of speckles varying as the speckles increased in size. The simulated images were used to determine ideal speckle parameters, and the algorithm was validated using an experimental test that evaluated various rigid body translations and rotations with good results [71].

Trebuna and Hagara [55] completed work on extracting the mode shapes of a vibrating plate using DIC and a laser vibrometer. A DIC analysis was performed on a vibrating plate using Istra4D, and a custom MATLAB script was used to extract the natural frequencies and mode shapes of the vibrating plate. The custom script took a given set of loads (f) and displacements (x, y, z) at each load step, found the related DFTs (F_{fft}, X_{fft}, Y_{fft}, Z_{fft}), then found the transfer functions (H_x, H_y, H_z). The normal mode indicator function (NMIF) and complex mode indicator function (CMIF) were calculated for each spectral line, and the maxima of the CMIF were taken as the fundamental frequencies of the vibrating plate and used to calculate the mode shapes [55]. The technique was verified by experiments conducted on two thin steel plates. The detected frequencies and the mode shapes were similar to that of the verification system.

Javh, Slavic, and Boltezar [21] completed vibration tests and conducted modal analyses using 2D DIC and identified modal information up to 10 kHz. They made use of a hybrid system consisting of an accelerometer and high speed cameras combined using a simplified gradient-based optical flow that generated full field 2D displacements. Experimental work was conducted on a steel beam with free-free conditions. A large force was applied using an impulse hammer to assist in higher frequencies being visible in the optical system. The DIC system picked up the first three modes up to 2730Hz, after which taccelerometer was used to detect the modes. The first 8 modes were produced by plotting the amplitudes of the spectra measured by the camera at each of the identified frequencies. The first four modes appeared clearly, while the higher modes were more difficult to see. The addition of an accelerometer to detect the eigenvalues was beneficial when combined with high speed cameras to measure the mode shapes [21].

Reu et al [42] examined 3D DIC for modal measurements and compared to a scanning LDV. The primary advantage that DIC found over the scanning LDV was that the DIC system was able to calculate displacements for all points in the structure at the same time, whereas the LDV was only able to measure sequentially. Both were found to have noise floors in the range of nanometers (approximately 1/10,000th of a pixel), allowing for the ability to pick up very slight differences in displacement to show the relevant mode shapes. The modal properties of a square aluminum plate were tested using a scanning LDV and pair of stereo cameras to capture simultaneously under the same excitation. VIC-3D was used to determine the displacement field. The 3D scanning LDV used 3 separate PSV-500 lasers to create 3D measurements. The scanning LDVs and the 3D DIC system found the modal frequencies to be within 0.02%, although the damping ratios found varied by up to 50% [42]. The mode shapes were also found to be quite comparable except for the 4th mode, where the DIC amplitudes were lower than the scanning LDV, which may be due to the small response of mode 4 that was only slightly above the noise floor of the DIC system. The strain fields resulting from each technique were compared and found to vary noticeably, primarily due to DIC measuring the direct response to surface displacements whereas the points that the LDVs are measuring may shift during testing. The noise floors of each technique were compared, and the scanning LDV was found to be better for out of plane movements. Both methods were comparable for in plane movements [42]. Reu et al noted that a scanning LDV is still preferred for out of plane modal analysis, but that 3D DIC is preferred if in plane displacements or strains are also required [42].

2.7.2 Field Usage in Civil Engineering

While early usage of DIC focused on material testing, it has also been used to monitor a number of real world civil structures. As early as 1993, Stephen, Brownjohn, and Taylor [48] adopted vision based methods for use in both dynamic and static measurements of a bridge structure. A single camera was used to measure the Humber Bridge in England, which is a 2,220m suspension bridge with a 1,410m long main span. This early application of optical measurements made use of a telescope sighting an area of high contrast, which in this case was artificially applied in the form of black and reflective rings clamped to the structure [48]. The targets and camera were aligned to measure the vertical and lateral motion of the structure. The algorithm calculated a correlation between the chosen subset and each integer pixel value in a chosen window slightly larger than the subset to determine the peak correlation point, which was taken as the displacement. Subpixel interpolation was not used, but it was noted that interpolation may improve the results [48]. A double-integrated acceleration signal was used to compare the displacement results, and the peak-to-peak amplitudes were found to be in agreement, although the detailed trends differed. Vertical and lateral vibration modes of the structures were found using the optical system and compared to the results of a modal survey of the structure. In general, the vertical frequencies aligned almost exactly, while the lateral frequencies were similar but varied more significantly compared to the modal survey [48]. Mode shapes passing through midspan were not apparent from the optical system, which only measured at midspan.

Yoneyama [65] examined bridge deflections using 2D DIC. A random pattern was applied to one external girder of a bridge and natural features were used on the other. This structure was loaded with a heavy cargo truck and measured at various locations along the bridge. The measured girders were well illuminated by artificial light due to the test being conducted overnight. Yoneyama [65] found that DIC was able to determine deflections at the subpixel level, and the measured deflections were comparable to displacement transducers, although there were minor discrepancies with the transducers at one location. Some minor scattering at the midspan of the bridge on the side using natural girder features was seen, resulting in the displacements at this point containing relatively large errors, although enough points were present along the rest of the girder to estimate the midspan deflections using a third order polynomial fit between the measured points [65].

Busca et al [11] completed vibration studies on full scale bridge structures using pattern matching, edge detection and DIC, all of which were compared to a laser interferometer as a reference sensor. The optical methods all made use of 2D imaging, although two different cameras were used to determine the effect the cameras had on the measurements. The camera frame rates during testing were limited to 17fps and 25fps, limiting the measurable fundamental frequencies to 9.5Hz and 12.5Hz, respectively, due to the Shannon-Nyquist sampling criterion. DIC was performed using the commerical program VIC-2D, while edge detection and pattern matching were performed using LabView 2010 edge detection and pattern matching tools. Busca et al [11] also examined various zoom levels, which affected the pixel to real world unit scale of their images and allowed for an evaluation of the scale factor affect on measurement accuracy. A 50m steel truss bridge in Italy was chosen as the test bridge for this study, and images were captured as trains drove over the bridge. The laser interferometer was mounted below the bridge at midspan, while three different measurement setups were used for the optical measurements: frontal measurements with targets, frontal measurements without targets, and in-axis measurements with targets. The measurement uncertainty using edge detection and pattern matching were both shown to only vary with the camera settings and scaling factor, rather than the vibration characteristics of the bridge. A number of targets were placed at different depths, and only some of the targets remained within the focused depth of field, but this did not affect the measurement uncertainty [11]. The targetless measuring approach was found to have higher uncertainty as expected. The targetless areas were noted to contain only small intensity gradients, which can affect performance [33, 29].

Ribeiro et al [43] performed dynamic and static tests both in laboratory and in the field monitoring railway structures. The laboratory tests assessed various factors, such as lighting, stiffness of the stand, and magnification (zoom) level, while the field test evaluated the field performance of the system. Targets were glued to a metallic plate to provide tracking targets on the structures, which were then illuminated by a light source. Focal lengths from 400mm to 3200mm were examined, and found to impact the measurement resolution. However, no interpolation beyond "1/2 pixel" interpolation was used [43]. SAPERA vision software and custom C++ functions were used for target tracking on the structures. No distortion correction appeared to be carried out, although the targets were kept to the center of the field of view to minimize distortional effects [43]. Errors present under incandescent lighting approached 0.02mm, while errors present under LED lighting remained close to 0.005mm. A stiff tripod was recommended to reduce errors associated with camera shake, particularly under air flow. Air flow rates of up to 10m/s were tested, and the error was found to approach 0.10mm. Ribeiro et al [43] compared their system to an LVDT, and found absolute errors in the order of 0.018 to 0.042mm for distances from 3 to 15m from the target. These distances correspond to 15px/mm to 25px/mm. Following the laboratory testing, a 42m long bowstring arch rail bridge was tested in Portugal. Distances from 5 to 25m were examined, relating to 12.5px/mm to 21px/mm. The results from the video sequence generally matched closely to the LVDT results. The 25m distance contained the largest error, which is possibly due to a large digital gain being applied in conjunction with both 2x and 4x extender tubes, resulting in a much lower signal to noise ratio and possibly higher distortion from the extender tubes.

Hoag [19] used 2D DIC to measure the lateral and vertical displacements of railway structures. These tests were performed under both static and dynamic conditions, and made use of live traffic loading. An open-source MATLAB based 2D DIC program, GeoPIV, was used for displacement measurements [19]. The structures tested ranged from single spans of steel railway bridges to a large steel bascule lift bridge in Kingston. Only some of the measurements were compared to other sensors, but those that were compared tended to match well with the other sensor(s) [19].

2.8 Errors in Digital Image Correlation

The errors present in DIC that make use of the FA-NR, IC-GN, or related algorithms are presented here. The errors from other registration algorithms, such as curve fitting or gradient based methods, are not covered here. The errors in curve fitting, gradient based, and other registration methods have generally been found to be larger than for the non-linear iterative solving methods [38].

2.8.1 Errors in 2D DIC

Various error quantifications have been completed for DIC techniques. The primary errors in the DIC algorithms have been found to be related to the interpolation [44] and shape [45, 60, 61, 66] functions used in the algorithms. The interpolation errors are generally related to the interpolation function used, with higher order interpolation giving more accurate results but being more sensitive to noise [44, 61]. The shape function errors are generally related to whether the shape function adequately captures the deformation field of a specimen, and can be seen as undermatched, matched, or overmatched.

Much of the work on error quantification in DIC has examined the effect of different shape functions on different deformation fields [45, 60, 61, 66]. In an ideal experiment, the displacement field shape would be known (i.e., whether it is rigid, linear, or includes curvatures, allowing operators to use 0^{th}, 1^{st}, or 2^{nd} order shape functions), but this is rarely the case in practice. Schreier and Sutton [45] examined the errors related to undermatched shape functions in DIC. This occurs when a shape function is chosen that is unable to capture the full effects of the displacement field, such as a 0^{th} order shape function being used to analyze linearly deforming materials. Schreier and Sutton [45] created a quadratic displacement field and analyzed it using both a first and second order shape function with biquintic interpolation. Using a first order shape function should result in systemic bias errors due to not capturing the displacement field correctly, but a second order shape function should be able to capture the displacements correctly and result in no systemic errors. For small subsets, the second order shape function was found to have almost double the random error of the first order, but otherwise had less random error. In general though, a similar level of random error was found for each, but the second order shape function that accurately captured the displacement field had fewer systematic errors present [45]. The authors suggested the use of the smallest possible subsets while meeting other criteria (such as a minimum size to correlate correctly) [45]. This would avoid accidentally undermatching the shape function, but may overmatch the shape function for some displacement fields and increase the random error slightly.

Yu and Pan [66] and Wang and Pan [60] examined the effects of overmatched or matched shape functions on DIC, and Wang et al [61] examined the errors present in a matched rigid body shape function. Overmatched shape functions result from the use of a higher order shape function than the displacement field, such as using a 2nd order shape function to measure rigid body displacements. Overmatched shape functions have been found to give error results similar to matched shape functions, and 2nd order shape functions have generally been seen to give approximately double the random error of 0th and 1st order shape functions [60, 61, 66]. The random errors due to a matched or overmatched shape function have been shown to vary relative to the random noise in the image and the SSSIG of the subsets being considered. This holds true for 0th to 2nd order shape functions, and theoretical work has shown that the 2nd order shape function has twice the random error of the 0th and 1st order shape functions [60, 61, 66]. The random error associated with a matched 0th order shape function can be found in Equation 2.57a [61], with a matched or overmatched 1st order shape function can be found in Equation 2.57b [60, 66], and with a matched or overmatched 2nd order shape function in Equation 2.57c [60, 66]:

$$Std[u] \approx \frac{\sqrt{2}\sigma}{\sqrt{\sum\sum(f_x)^2}} \approx \frac{\sqrt{2}\sigma}{\sqrt{\sum\sum(g_x)^2}} \tag{2.57a}$$

$$Std[u] \approx \frac{\sqrt{2}\sigma}{\sqrt{\sum\sum(f_x)^2}} \approx \frac{\sqrt{2}\sigma}{\sqrt{\sum\sum(g_x)^2}} \tag{2.57b}$$

$$Std[u] \approx \frac{2\sqrt{2}\sigma}{\sqrt{\sum\sum(f_x)^2}} \approx \frac{2\sqrt{2}\sigma}{\sqrt{\sum\sum(g_x)^2}} \tag{2.57c}$$

where u is the displacement under consideration, σ is the standard deviation of the 0 mean random Gaussian noise in the image, $\sum\sum(f_x)^2$ is the SSSIG of the reference subset, and $\sum\sum(g_x)^2$ is the SSSIG of the current subset being analyzed, which is approximately the same as the SSSIG of the reference subset. The derivations of Equation 2.57 assumed perfect interpolation and that the image noise was much smaller than the image intensities, which is generally true in digital imaging. Wang and Pan [60] validated the matched and overmatched results using a numeric study, which found that the the standard deviation error was not affected by the subpixel displacement (as expected based on previous works), and varied approximately linearly with the standard deviation of the added noise. This agreed with Equation 2.57, which also varies linearly with the standard deviation of the added noise. Additionally, a homogenous strain profile was tested numerically, and the numeric results revealed that the undermatched 0th order shape function had error increase almost linearly, while the 1st (matched) and 2nd (overmatched) order errors remained relatively constant. The magnitude of the 2nd order errors was approximately double the magnitude of the 1st order errors, although both were still below 0.01px. In both validation tests, an inverse relationship between the subset size, which is related to the SSSIG, and standard deviation error was found. This matches with Equation 2.57, which predicted an inverse relationship between SSSIG and the standard deviation error. The 2nd order shape function also matched Equation 2.57c well under a quadratic deformation field [60]. The increased standard deviation error caused by the use of a second order shape function can be compensated for using a larger subset to improve the SSSIG score in the denominator [66]. Overall, the errors from overmatched shape functions were much smaller than those from undermatched shape functions, leading Yu and Pan [66] to recommend the use of a 2nd order shape function with a subset between 31x31 and 71x71 pixels if the deformation field is unknown.

Schreier, Braasch, and Sutton [44] examined the errors due to interpolation in the FA-NR algorithm. A bias error was found for both bicubic and biquintic interpolation functions, which has been found to be related to the interpolation functions used [44]. This result was also found by Wang et al [61] when examining linear and cubic interpolants and confirmed by Su et al [49]. The mean bias error was found to be a sinusoid that varies according to the subpixel displacement as shown in Figure 2.15, and that is 0 at the integer pixel positions and 0.5px displacement. Su et al [49] derived a theoretical measure of the interpolation that took the form of $u_e = Csin(\pi u_0)$, where u_0 is the subpixel displacement level. The constant, C, was found to vary directly with the interpolation method used [49]. The magnitude of the bias error due to interpolation depends on the interpolant used as well as the intensity distribution of the analyzed images [44, 49, 61]. High frequency components of the speckle pattern were found to significantly contribute to the bias error measure due to interpolation [49]. One way to avoid interpolation bias is the use of the *sinc* function to interpolate, but the infinite nature and slow decay of *sinc* means it is impractical [49]. An ideal interpolation would not cause aliasing, and the transfer function would stretch from -0.5 to 0.5 pixels around a given point. This is not the case with practical interpolants, which partially overlap beyond the ±0.5px boundary [49]. A low pass filter can be used to reduce interpolation bias errors through suppressing the aliasing effects by reducing high frequency content [49].

The FA-NR and IC-GN algorithms use image gradient values differently in their formulation. The FA-NR algorithm requires interpolation of the target subset from which the gradients are calculated, while the IC-GN algorithm is able to directly calculate the gradients from integer pixel values due to the inverse composition involved [35]. Shao, Dai, and He [47] found that noise does not impact the bias error in the IC-GN algorithm due to it sampling at integer pixel locations, but does impact the bias error of the FA-NR algorithm due to interpolation of the noise of the target subset. Shao et al [47] examined a rigid

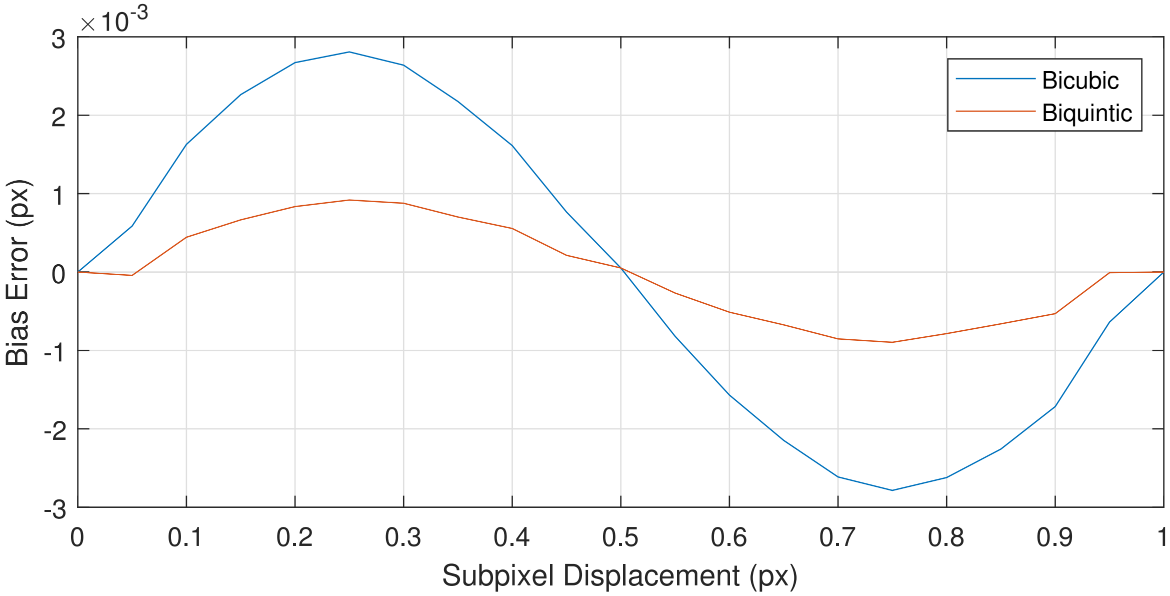


Figure 2.15: Bias error vs. subpixel displacement.

shape function however, so Pan and Wang [35] expanded on their work and derived the theoretical measures of error in the IC-GN framework for common interpolants. Su et al [50] investigated the effect of the discrete gradient estimators and found that the gradient estimator has a different impact on interpolation bias error in the IC-GN algorithm than in the FA-NR algorithm. The bias error in the IC-GN framework has been found to depend directly on the form of gradient estimator used in the analysis [50]. Su et al [50] noted that the Barron gradient operator is approximately similar to results obtained from the FA-NR algorithms previously, but that the Prewitt operator outperformed both in limiting bias error. Su et al [50] also found that noise level had a negligible effect on bias error for the IC-GN algorithm, although it did significantly affect standard deviation error.

2.8.2 Errors in 3D DIC

The errors in 2D DIC have received much of the literature focus, with only a few papers detailing the errors present in 3D DIC systems. Errors can occur in 3D DIC due to many sources, including noise and interpolation biases, but also from errors related to the camera calibration process. Early work by Becker et al [8] demonstrated that calibration errors can introduce systemic errors into a 3D DIC system, and showed that larger subsets may actually increase errors in some situations. This is due to poorly capturing the deformation field, although this is more likely an effect of undermatching the shape functions between the left and right images. Other authors [5, 14] have since recommended the use of a 2$^{\text{nd}}$ order shape function when interpolating between left-right paired images to avoid this undermatching effect, although a 1$^{\text{st}}$ order shape function may still be applicable to matching between left-left or right-right image sets.

Gao et al [14] examined the use of both the FA-NR and the IC-GN subpixel registration algorithms for 3D DIC. They used both algorithms with both 1$^{\text{st}}$ and 2$^{\text{nd}}$ order shape functions, and created two simulated tests using a sinusoidal deformation field and a planar homography deformation field. The planar homography deformation field was used to simulate the shift from the left camera to the right camera. Gao et al [14] found that the computation speed of the IC-GN was much faster than FA-NR in all cases, and found that the 2$^{\text{nd}}$ order IC-GN was only marginally slower than the 1$^{\text{st}}$ order version due to taking more time to compute each iteration, but requiring fewer iterations to correlate. Errors were found to be much lower in the tests when using the 2$^{\text{nd}}$ order algorithms, which is due to the non linear deformation between images.

Balcaen et al [5] created a digital image generator based on a finite element (FE) mesh to create unbiased images for use in numerical 3D DIC studies. This generator maps a known speckle pattern to an FE mesh, which can then be deformed and projected into the camera views. This generator was used to evaluate errors due to the 3D DIC setup, calibration, and image aliasing. As the distance from the center of the field of view (FOV) increased, uncertainy was found to increase in the X direction, but remained relatively constant in the Y and Z directions. Most of the error found was able to be attributed to the quality of the calibration. For the X and Y directions, the camera centers and distortion parameters were found to have the most effect, while the Z direction depended most heavily on the relative camera pose [5].

Recommendations from various authors include using as much of the FOV as possible in calibration to better account for distortion, use up to 50 image pairs to calibrate, and to use the center of the cameras' FOV to measure when possible in order to limit the effects of distortion [5]. A stereo angle greater than 10° is also recommended to allow for an increased baseline between the cameras, which can significantly improve the accuracy [5]. A higher stereo angle in general decreases the uncertainty in the Z direction, but increases the uncertainty in the X and Y directions due to the quantization errors shown in Figure 2.12. To avoid aliasing issues, it is also recommended to use a higher resolution camera, move the cameras closer to the test piece, use a higher focal length, or use a pre-analysis low pass filter to minimize the effects of aliasing [5].

Wang et al [62] and Ke et al [22] examined errors in stereo DIC extensively across two papers, and found general formulae for bias and variance errors in 3D position. The bias generally took the form of $E = Asin(w(X)X)$. In general, the variance of the 3D position was found to depend on the baseline, focal length, and angle between the cameras by Equation 2.58 [62]:

$$Var[P] = \frac{L^2}{f^2 cos(\theta)} \tag{2.58}$$

where P is the point of interest, L is the baseline, f is the focal length, and θ is the angle between the cameras. They found through both simulation and experiments that the in plane variability was slightly impacted by the angle between the cameras, but the out of plane variability was significantly improved with larger angles between the cameras. Provided a good camera calibration is used, the angle between cameras can be tailored to whether in plane or out of plane accuracy is more desired by investigators [62].

The theoretical results were confirmed by Ke et al [22], who performed a number of validation experiments and compared to the theoretical derivations. Up to approximately 3% noise, the theoretical and experimental values were in good alignment, although beyond 3% image noise the experimental variance was found to increase beyond that predicted by theory. Ke et al [22] found that areas away from the camera center generally exhibited higher uncertainty even without distortion present, which is likely due to the quantization errors. Experiments with noisy images found that 2D errors had randomly varying bias in either direction, with a peak magnitude up to 0.01px. The 3D errors matched theory quite well, and the bias present is likely due to the intensity interpolation and the subset quality.

Chapter 3

Experimental Setup

3.1 Numeric Testing

3.1.1 Image Simulation

Images for the numerical analysis were simulated using the process detailed by Reu [41] with speckle sizes ranging between 1 and 18 pixels. Images were generated with a 450 x 350 resolution, then windowed using a Tukey window [56]. These images were then transformed to the Fourier domain, where they were translated in a single direction using Equation 3.1 [41]:

$$\begin{aligned} X_m &= \sum_{i=0}^{L-1} x_i e^{-j2\pi mi/L} \\ Y_m &= X_m e^{-j2\pi mk/L} \\ y_m &= \mathcal{F}^{-1}(Y_m) \end{aligned} \tag{3.1}$$

where x_i is a given row/column, m is the Fourier index, L is the length of the row/column, and k is the imposed shift. The images were then transformed back to the spatial domain and cropped to a resolution of 400 x 300 to remove the edge effects caused by the Tukey windowing. A simulated image is shown in Figure 3.1. A subpixel shift of 0.05px was used between images, and a series of 21 images was used to capture subpixel displacements from 0 to 1px. Left/right image pairs were generated by displacing the right images 5px in the horizontal direction.

The simulated images were first analyzed to determine which speckle size would be used for the remainder of the analysis, which resulted in 8 pixel speckles being used to

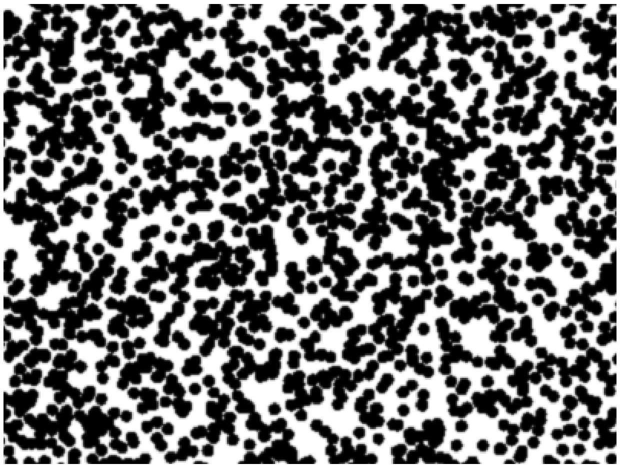


Figure 3.1: Example simulated image.

determine the other optimal parameters of the system. An 8 pixel speckle is similar to the range of 2 to 7 pixels that has been recommended as best practice for practical use of the FA-NR algorithm [10, 71].

3.1.2 Test Procedure

A series of tests were completed while varying the shape function, interpolation method, Gaussian pre-blur, subset size, speckle size, and deformation field. Table 3.1 displays the variables and ranges tested. A regular grid pattern was used comprised of 651 (31x21) analysis points, which were used across all tests. These analysis points were used to determine the bias and random (standard deviation) errors discussed in Chapter 4. The mean bias error was taken by finding the measured error between the correlated displacement and the exact subpixel displacement at each pixel location and averaging across the ROI. Similarly, the standard deviation errors were calculated by taking the standard deviation of the error across all pixel locations in the ROI. A simple rigid translation in the vertical direction was used to determine the effects of each variable on the system. This allows for easier comparison to previous works, which have most often made use of rigid translations.

The 2D analyses were completed using pixel coordinates to provide a normalized error measure that is applicable to any camera system and geometric setup. The 2D analyses involved a 0 to 1 subpixel displacement. The 3D analyses were completed using real world coordinates. The subpixel displacement shown for the x-axis of the 3D results

Table 3.1: Simulated image test matrix.

Variable	**Details**
Shape Function	1st order, 2nd order
Interpolation Method	Bicubic, biquintic
Noise	Gaussian distributed; $\sigma = 0\%$ to 4%; 1% increment
Gaussian Blur	1x1 (no blur) to 7x7 kernels
Subset Size	21x21px to 81x81px; 10px increment
Speckle Size	1px to 18px; 1px increment
Deformation Field	Rigid

is the subpixel change given in the y direction, and corresponds to a shift from 0 to 0.22mm in the y direction. The approximate distance between the planar cameras and the target is 2000mm. Figure 3.2 displays the 3D displacement error results from a 1st order shape function with bicubic interpolation, and shows that the Z displacement bias error is significantly larger than the X or Y displacement components in the presence of noise. The 3D analyses were therefore conducted on just the Z (out of plane) displacements.

3.2 Laboratory Testing

3.2.1 Equipment

High Speed Cameras

A pair of high speed cameras were used to capture images. These cameras were both JAI SP-12000M high speed monochrome digital cameras, with a resolution of 12MP (4096px x 3072px) shown in Figure 3.3. The cameras have 22.53x16.90mm APS-C sensors, which allow for low noise levels in good lighting. The maximum capture rate at full resolution is 189fps when connected by four coaxial cables to the frame grabber boards (subject to memory/processor limitations in the computer). The cameras captured at a sampling rate of 30fps or 30Hz. The camera settings were controlled via the coaxial cable from the computer, and the first camera (in this case, the left camera) was triggered by the computer. The signal was passed through the first camera via an output connection to the input connection of the second camera, causing it to trigger simultaneously. Both cameras made use of the same fixed focal length ZEISS 50mm f/2.2 manual focus lenses. The focus ring was able to be locked down by a thumb screw to ensure focus did not shift during

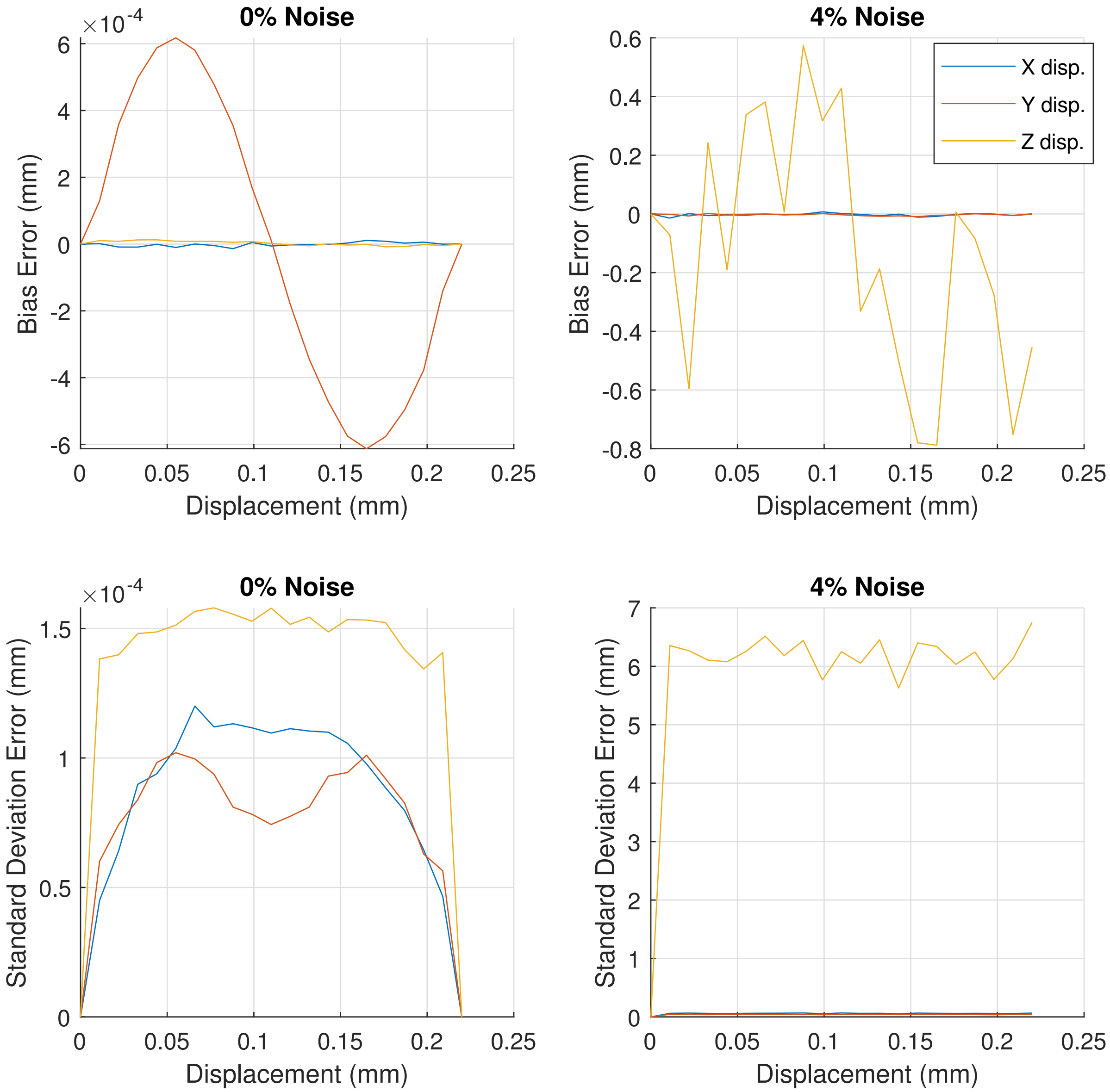


Figure 3.2: 3D displacement errors.

capture. The cameras were mounted to a 1/2" (12.7mm) aluminum plate by four M3 screws each to ensure they did not shift relative to each other during testing, and the aluminum plate was mounted to a Cameron T310BH aluminum ballhead tripod using a standard 1/4" (6.35mm) UNC threaded hole. Figure 3.4 shows the camera setup during the planar testing.

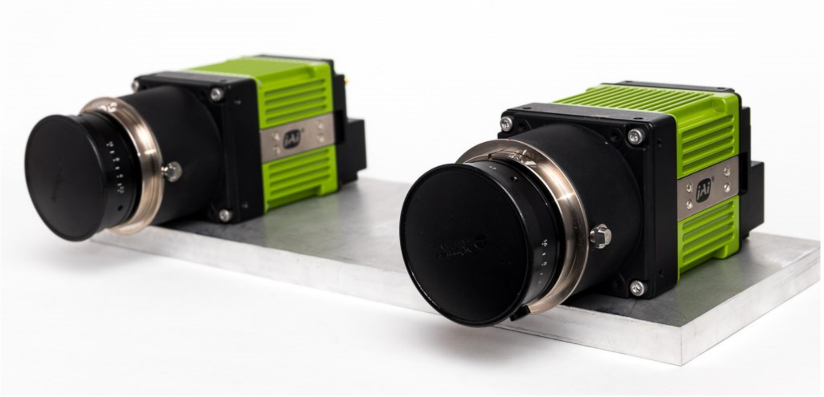


Figure 3.3: JAI SP-12000M high speed cameras with ZEISS 50mm f/2.2 lenses.

The lens apertures were both set to f/5.6 for calibration and measurement. This aperture was chosen because it possessed a reasonable depth of field and high sharpness without being too small and limiting the light entry too much. Supplemental lighting was present during the tests, so wider apertures were not required. Smaller apertures were also avoided to prevent diffraction issues. The calibration parameters are displayed in Tables 3.2 and 3.3. The calibration model used 3 radial distortion parameters and accounted for pixel skew and tangential distortion. The supplemental lighting was provided by a 1500 Lumen Husky LED work light and a Neewer CN-160 video light mounted on light stands.

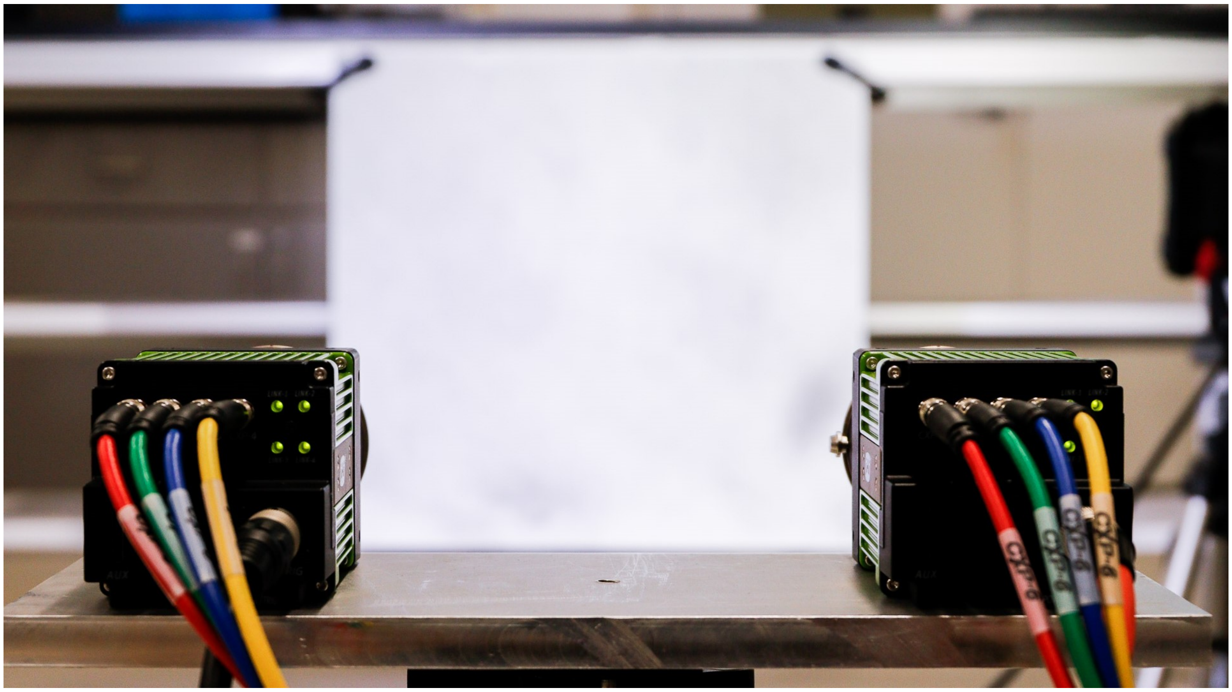


Figure 3.4: JAI SP-12000M high speed camera setup.

Table 3.2: Individual camera calibration parameters.

Camera	Parameter		RMS Error (px)
Left	Intrinsics, K	$\begin{bmatrix} 9534.5 & 0 & 0 \\ -19.42 & 9538.9 & 0 \\ 2118.4 & 1500.6 & 1 \end{bmatrix}$	0.673
	Radial Dist.	$\begin{bmatrix} 0.0234 & 0.1567 \end{bmatrix}$	
	Tangential Dist.	$\begin{bmatrix} -0.00027 & 0.0038 \end{bmatrix}$	
Right	Intrinsics, K	$\begin{bmatrix} 9555.0 & 0 & 0 \\ -15.16 & 9559.7 & 0 \\ 2108.6 & 1654.2 & 1 \end{bmatrix}$	0.662
	Radial Dist.	$\begin{bmatrix} 0.0286 & 0.0672 \end{bmatrix}$	
	Tangential Dist.	$\begin{bmatrix} 0.0040 & 0.0011 \end{bmatrix}$	

Table 3.3: Stereo camera calibration parameters.

Camera	Parameter	
Left	Rotation Matrix, R	$\begin{bmatrix} 1.0000 & 0.0032 & -0.0010 \\ -0.0032 & 0.9998 & 0.0187 \\ 0.0010 & -0.0187 & 0.9998 \end{bmatrix}$
	Translation Vector, t	$\begin{bmatrix} -201.8120 & -2.4650 & -1.5298 \end{bmatrix}$
	Essential Matrix, E	$\begin{bmatrix} 0.0073 & 1.4835 & -2.4932 \\ -1.7246 & 3.7714 & 201.7752 \\ 1.8157 & -201.7837 & 3.7684 \end{bmatrix}$
	Fundamental Matrix, F	$\begin{bmatrix} 0.0000 & 0.0000 & -0.0003 \\ -0.0000 & 0.0000 & 0.0211 \\ 0.0002 & -0.0213 & 0.8330 \end{bmatrix}$
Stereo RMS Error (px)		0.667

Laser Vibrometer

A Polytec PDV 100 laser vibrometer shown in Figure 3.5 was used to provide a baseline comparison for the tests, albeit at a single point roughly in the middle of the vibrating plate. The laser vibrometer was mounted on a Benro TMA37AL Mach3 aluminum tripod to ensure stability during testing, and faced the backside of the plate that the cameras measured. This was to ensure that the surface of the plate could be properly prepared for each instrument separately. Figure 3.6 shows the setup for the laser vibrometer. The LDV data was captured with a sampling rate of 240Hz. No highpass filter was used for the LDV, and a 1kHz lowpass filter was used by the LDV hardware. The LDV required a lowpass filter to be selected for operation, and a 1kHz lowpass filter was the default setting.

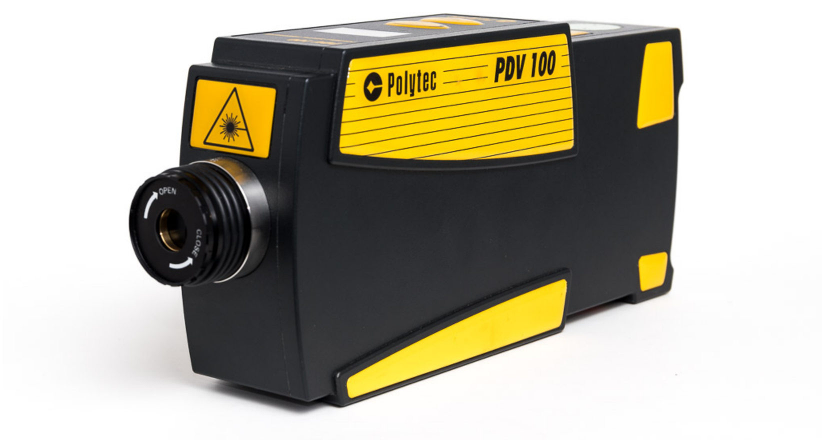


Figure 3.5: Polytec PDV100 laser vibrometer.

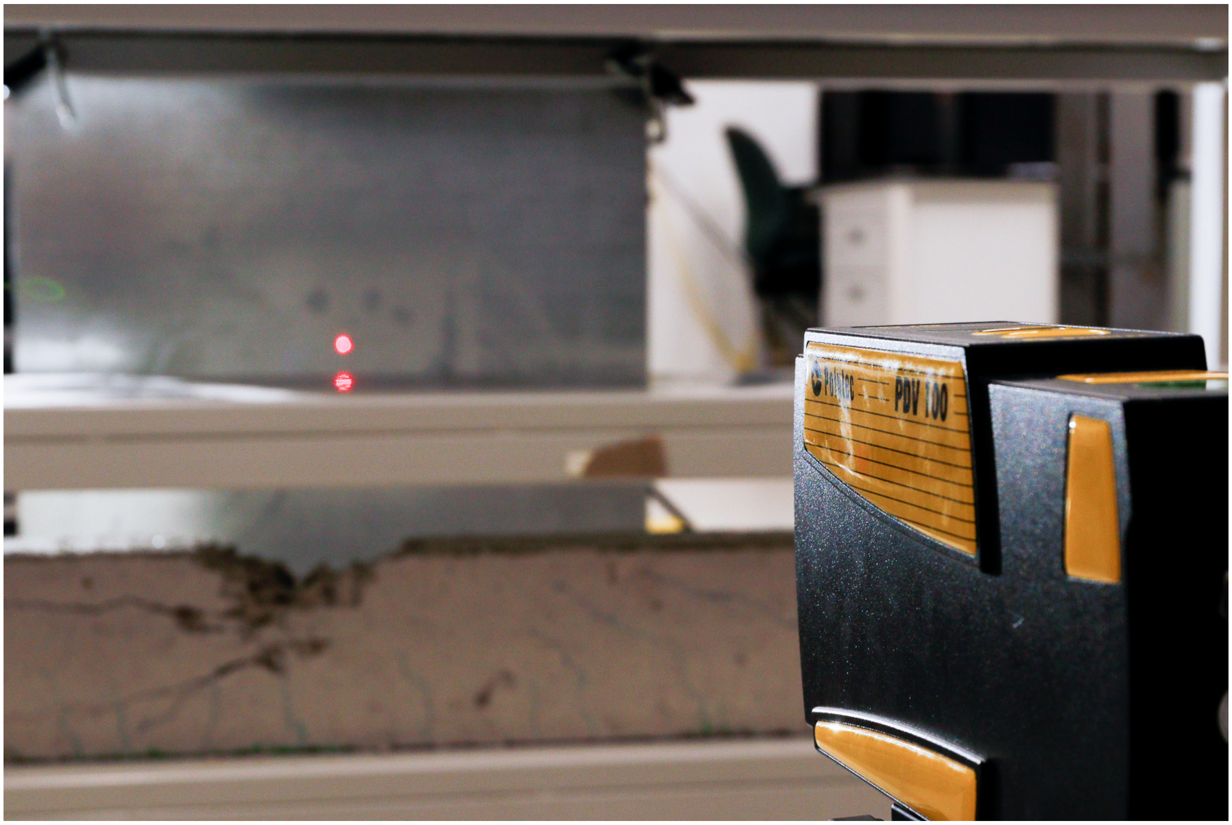


Figure 3.6: Laser vibrometer setup.

Impact Hammer

A Dytran Dynapulse 5800-B4 impact hammer was used to excite the plate. The impact hammer data was collected through the use of a 32 channel Siemens LMS Scadas Mobile system. The LMS system was connected to the same laptop as the laser vibrometer, but the signals were not hardware synchronized. The impact hammer data was collected at a sampling rate of 512Hz. The impact hammer had a voltage to force conversion factor of 2.473mv/N.

3.2.2 Test Procedure

A 596.9mm (23½”) wide by 558.8mm (22”) tall steel plate was hung from a support as shown in Figure 3.7. The high speed cameras and laser vibrometer were set up on opposite sides of it. The cameras measured the entire plate area, while the laser vibrometer was used to measure a single point at the approximate middle of the plate (276.23mm from the top edge and 282.58mm from the left edge). The impact hammer was used to excite the plate by striking it at the bottom edge of the plate on the side of the high speed cameras (22.39 mm from the bottom and 536.41 mm from the left edge). Five trials were completed using this setup. The cameras captured 1200 images during each test.The high speed cameras had an exposure time of 10000μs and a framerate of 30fps (30Hz) to ensure sharp images and that the cameras captured at least the first 4 modes of the plate.

3.2.3 Modal Analysis

The experimental modal analysis was completed through the use of built-in MATLAB functions ($modalfrf, modalsd, modalfit$). A rectangular window was used on the data to avoid cutting off or lowering the amplitude of the impulse load in the test. The frequency response functions (FRFs) of the plate were estimated using Welch’s averaged, modified periodogram method [2]. The FRFs were then used to build a stabilization diagram to determine the physical frequencies of the structure. The least squares complex exponential (LSCE) algorithm was used to determine the modal parameters from the FRFs [2]. The calculated modal parameters include the fundamental frequencies, damping ratios, and mode shapes of each mode.

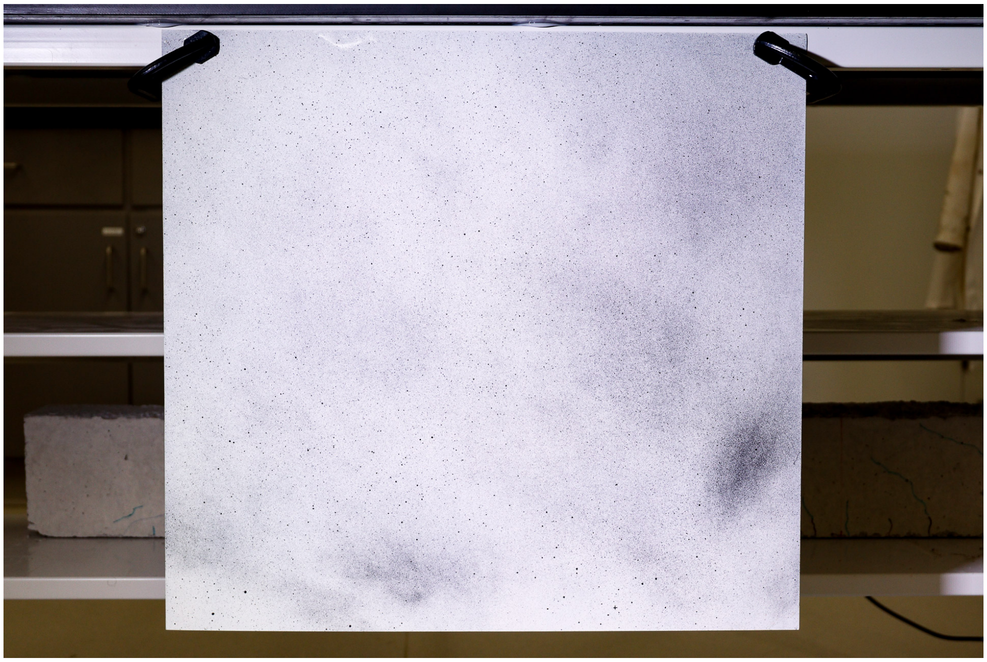
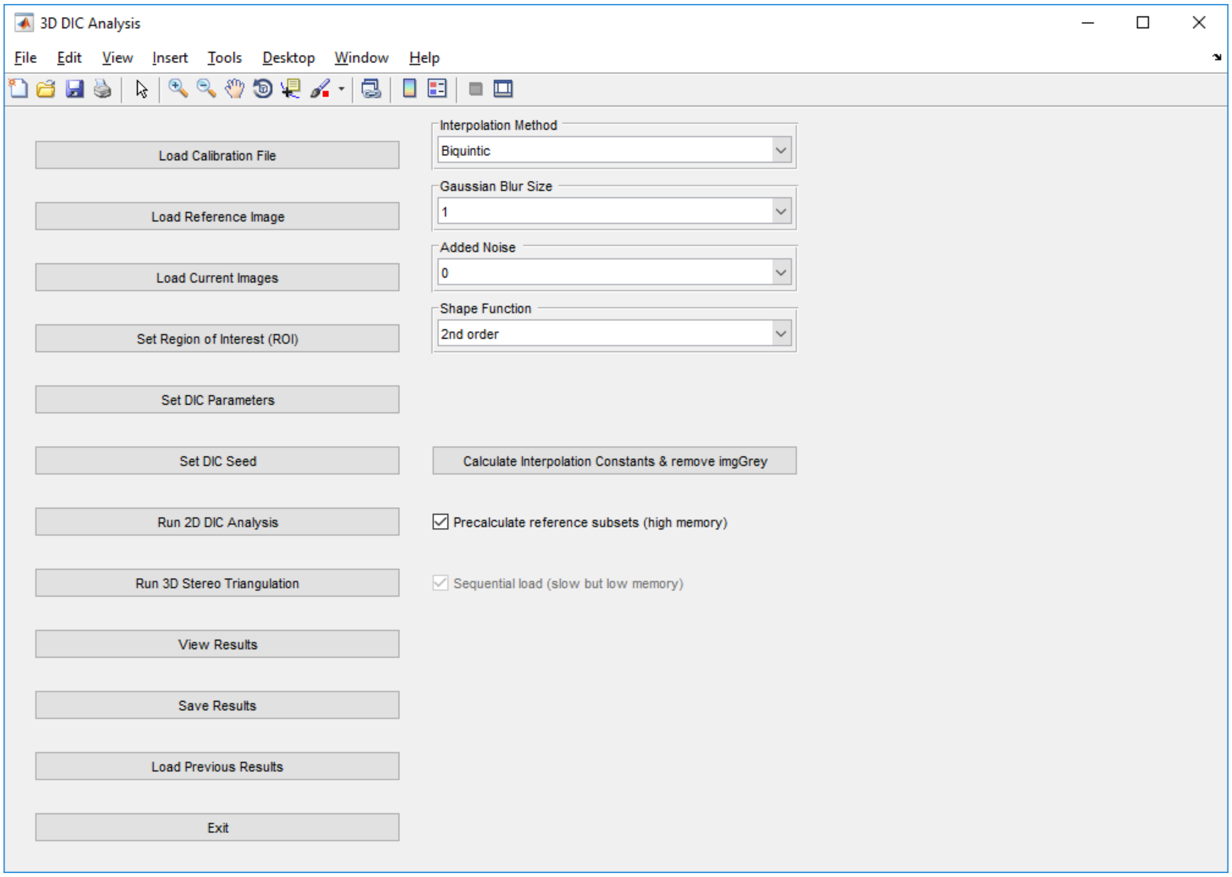


Figure 3.7: Stainless steel plate hanging from support (speckle side shown).

3.3 Software Architecture

The DIC software was developed using MATLAB R2017a [2], and is compatible with versions R2014a onward. MATLAB allowed for easy creation of a graphic user interface (GUI) while also allowing for use of MATLAB's many built-in functions and ability to compute matrix calculations efficiently. The created software is capable of analyzing either a 2D or 3D image set and saving the resulting displacement fields. Figure 3.8 shows the main interface of the GUI. Users start by selecting relevant settings such as the interpolation type, Gaussian blur size, and shape function before loading the stereo calibration and loading undistorted images. Users then select a region of interest (ROI) in the left reference image, from which future calculations stem. A seed point is used as the initial guess in each analysis image. The location of the seed point in the left reference image is selected by the user, and then the seed points are calculated automatically for the remaining images by maximizing the normalized cross correlation using 2D DFTs. The user may then run the correlation analysis in the image planes, determine the 3D triangulation, and output the results, which may be saved in .mat file format or viewed graphically in the program. The program is also able to load results from previous analyses for viewing. Optional settings



Interpolation Method: Biquintic

Gaussian Blur Size: 1

Added Noise: 0

Shape Function: 2nd order

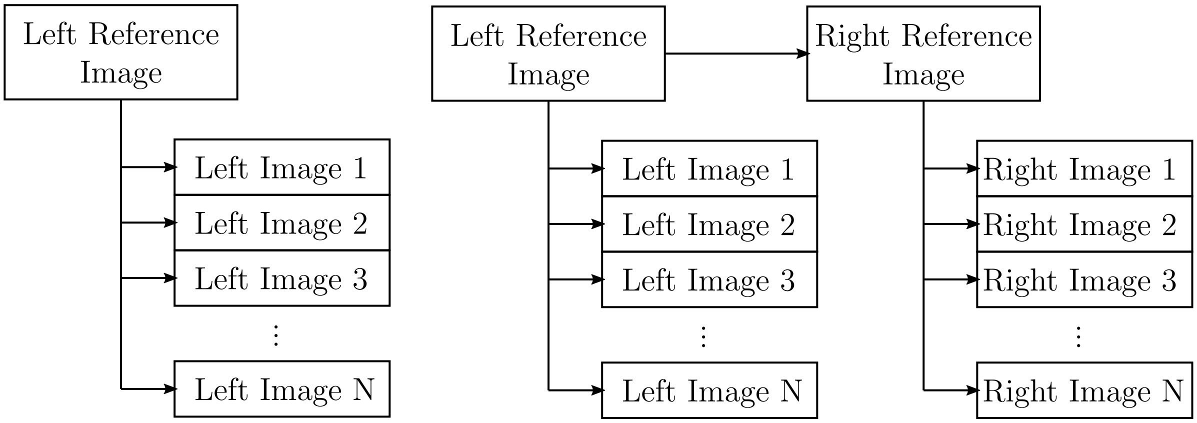
☑ Precalculate reference subsets (high memory)

☑ Sequential load (slow but low memory)

Figure 3.8: Main GUI of the DIC system.

include sequential loading for large image sets to avoid memory issues, pre-calculation of the right reference image subsets to speed right image correlation, and pre-calculation of interpolation constants and clearing of greyscale images to save memory. The option to add a randomly generated noise at a level from 0 to 4% (of an 8-bit image) was added for examining the effect of noise on numeric images, but is not intended for general usage.

The software makes use of the inverse compositional Gauss-Newton (IC-GN) non-linear iterative solver, and is capable of using 0^{th}, 1^{st}, or 2^{nd} order shape functions. Bilinear, bicubic, and biqunitic b-splines are available for interpolation, which is conducted using the efficient manner described in Chapter 2. Gaussian pre-filtering is completed using the built in MATLAB function, $imgaussfilt$, where the standard deviation of the filter is determined by $\sigma = \frac{\sqrt{h}-1}{4}$ for a filter size of h [2, 30]. The image analysis order is shown graphically in Figure 3.9. The 2D images are analyzed sequentially, while the 3D images are analyzed in three steps. The first step consists of correlation among the left images, the second step correlates the left reference image to the right reference image, and the third step correlates the right images to the right reference image at points chosen based on the second step. This limits the only correlation between different camera views to the left and right reference images, improving computation time and limiting error [14]. The 2$^{\text{nd}}$ order shape function is generally recommended when computing between camera views, and this allows it to be limited to a single step when not required in general [14].



(a) 2D analysis.

(b) 3D analysis.

Figure 3.9: Image order in DIC system.

3.3.1 Calibration Model

The DIC system assumes the use of calibrated stereo cameras, so an adequate stereo calibration should be performed before use. Camera calibration was completed through the use of the MATLAB built in stereo calibration function [2]. The camera calibration makes use of the pinhole camera model, and is generally suitable for up to a 95° FOV. A series of stereo-paired images were loaded into the function, which operates in two steps. The first step assumes that the lens distortion is 0 and uses closed form solutions to generate initial guesses for the camera intrinsic and extrinsic parameters. The initial guesses are then used for the full calibration, which makes use of Levenberg-Marquardt non-linear optimization to minimize the reprojection error [2]. Lens distortion is also accounted for in this step.

Chapter 4

Results

Select results are presented in this chapter to validate the DIC system developed during this thesis. The full set of results may be found in Appendix B for both the numeric and laboratory test sets.

4.1 Numeric Results

4.1.1 Speckle Size

The results of the investigation into speckle size can be found in Figures 4.1 and 4.2. The results shown are for rigid body displacement from 0 to 1px using 31x31px square subsets, a 10px step, and a convergence criteria of $|\Delta p| < 10^{-6}$. The images were correlated and errors evaluated in 2D for the speckle size analysis. All points were found using a grid of 31x21 points, for a total of 651 analysis points in each image. Correlation deteriorated starting at a pixel size of 14px and failed to correlate beyond 16px; hence, only results up to 16px are plotted.

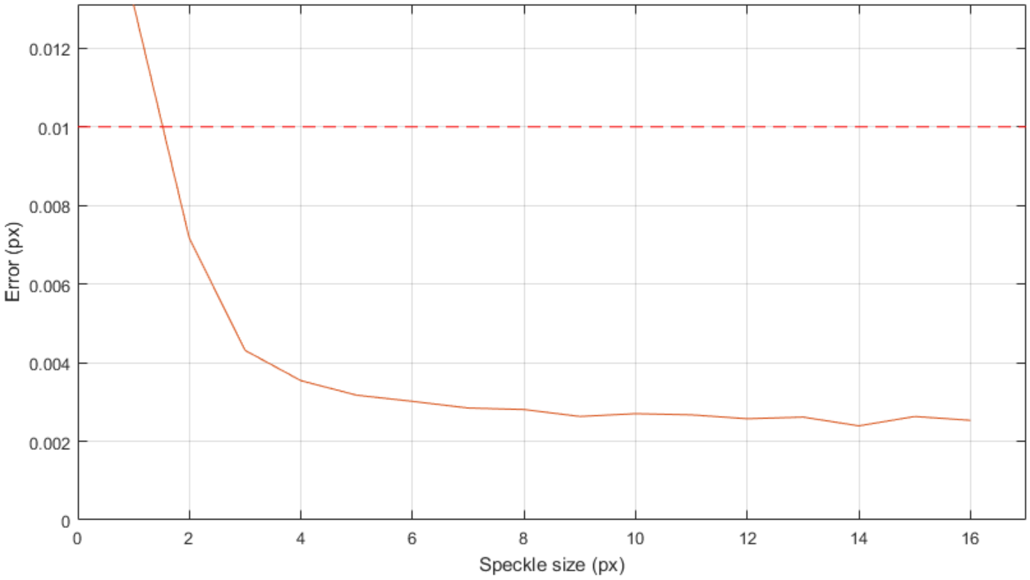


Figure 4.1: Peak bias error vs. speckle size.

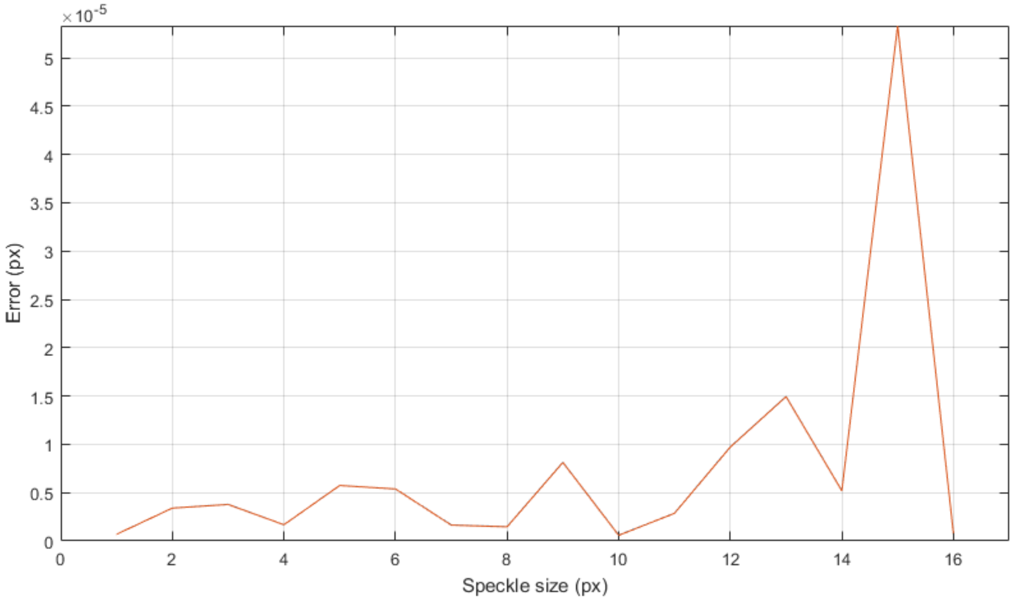


Figure 4.2: Mean bias error vs. speckle size.

4.1.2 Shape Function

1st and 2nd order shape functions were examined during the numerical simulation. This was to match the real world practice of typically using at least a 1st order shape function, even for rigid deformations. Figure B.1 shows that the shape function does not have any significant effect on the bias error. There is no difference in bias error when using a 1st or a 2nd order shape function without image noise. With image noise, there are slight differences between a 1st and 2nd order shape function, but they are relatively similar in both magnitude and shape. The largest difference between the shape functions can be seen in the standard deviation error, where the 2nd order shape function gives twice the standard deviation error of the 1st order shape function for the same noise level.

Figure B.2 contains similar plots with a 3x3 Gaussian blur added prior to analysis. The effects of the blur in general are discussed in more detail in Section 4.1.5, but the results here show that the Gaussian blur does not affect the errors differently for each shape function. The difference between the shape functions can again be seen primarily in the standard deviation error, where the 2nd order shape function again gives twice the standard deviation error of the 1st order shape function at the same noise level.

Figures B.3 and B.4 show that the shape function does not significantly impact the bias errors in 3D DIC. Without noise, the errors are extremely small, in the order of 10^{-5}mm. The shape function did have a significant impact on the 3D standard deviation error, with the 2nd order shape function having approximately double the standard deviation error of the 1st order shape function. The 3D errors due to the shape function chosen also do not appear to be influenced by whether a Gaussian blur is applied.

4.1.3 Interpolation Function

The interpolation function results in a sinusoidal bias error in the 2D case as shown in Figure B.5, the magnitude of which depends on the interpolation method chosen. The sinusoid is equal to zero at both the integer pixel locations and at the 0.5px shift. The magnitude of the bias error of the bicubic interpolation was found to be larger than the biquintic regardless of shape function. The addition of image noise increased the peak magnitude of the bias errors and reduced the smoothness of the sinusoidal bias error due to random variation. The noiseless biquintic interpolation was found to have small errors ($< 1.0 \times 10^{-3}$px) even before the addition of a Gaussian blur, which reduced it further ($< 3.0 \times 10^{-4}$px). These values are both smaller than the noiseless bicubic bias errors, which were found to be $< 3.0 \times 10^{-3}$px before filtering and $< 1.5 \times 10^{-3}$px after filtering. The presence of noise increases the bias error significantly; the biquintic peak bias increased from $< 1.0 \times 10^{-3}$px to 4×10^{-3}px and the bicubic interpolation increased from $< 3.0 \times 10^{-3}$px to $< 7.5 \times 10^{-3}$px. The standard deviation errors are approximately the same regardless of which interpolation method is used (when considering the same shape function), although the biquintic standard deviation error is marginally smaller than the bicubic standard deviation error.

The interpolation function did not appear to significantly affect the bias error or standard deviation errors in the 3D analysis as shown in Figure B.7. Additionally, the sinusoidal shape present in the 2D analysis largely disappears in the 3D analysis. In a perfectly noiseless case, the sinusoidal bias error can still be faintly seen, but this is removed entirely at even 1% noise. The biquintic standard deviation error is again marginally smaller than the bicubic standard deviation error, but they are very similar when using the same shape function.

4.1.4 Noise

The 2D bias error due to additional noise is very small for the IC-GN algorithm. Even with large noise levels (4% of an 8-bit max value 255, or 10.2 intensity levels), the error does not increase appreciably. Biquintic interpolation generally had a smaller magnitude of 2D bias error under comparable noise levels, but did seem to increase more relative to the noiseless case than bicubic interpolation did. The standard deviation increases directly with the addition of noise (as expected by [60, 61, 66]). In general, the 2$^{\text{nd}}$ order shape function has twice the standard deviation error of the 1$^{\text{st}}$ order shape function, but the interpolation function does not appear to affect the standard deviation error at all.

The 3D bias error was significantly affected by the addition of image noise. With no noise, the analysis is incredibly accurate (to approximately 10^{-5}mm), but the introduction of image noise causes significantly more bias error. Even a 1% noise results in ±0.2mm of error, and larger noise levels result in up to ±1mm of bias error. However, it appears that the peak magnitude of the bias error is unrelated to the noise level beyond 1%, with the 2-4% noise levels giving similar bias error results. The standard deviation error was affected by noise similarly to the 2D results, with an increasing noise level increasing the standard deviation noise level approximately linearly. The 2$^{\text{nd}}$ order shape function additionally had approximately double the standard deviation error of the 1$^{\text{st}}$ order shape function.

4.1.5 Noise Reduction

The addition of a Gaussian blur prior to DIC analysis was used to reduce noise and to smooth edges in the images. Figures B.13 to B.18 display the effects of applying a Gaussian filter in the presence of 0%, 2%, and 4% noise in the images. In the case of noiseless bicubic images, the 5x5 Gaussian blur was most successful at reducing the bias error in the images, and actually outperformed the stronger 7x7 blur. In the case of noiseless biquintic images, the 3x3 and 5x5 Gaussian blurs performed very similarly, and both out performed the 7x7 blur, although the differences were very small in all cases. In both the bicubic and biquintic cases, the Gaussian blur increased the standard deviation error, with the 7x7 blur increasing the standard deviation error the most and the 3x3 blur increasing the standard deviation the least. In general, the 3x3 blur had only a very slight impact on the standard deviation. As the noise level increased, the benefits of applying a Gaussian blur became much less pronounced, but the relative impact to the standard deviation of the error also became less pronounced. At the 2% noise level shown in Figures B.15 and B.16, the 5x5 and 7x7 blurs still performed best for bicubic interpolation, and all blurs performed approximately equally well for biquintic interpolation. At the 4% noise level shown in Figures B.17 and B.18, the 5x5 and 7x7 filters still had a small effect on the bicubic bias error, but much less pronounced than at lower noise levels, and there was no improvement in the biquintic bias error.

The shape of the 3D bias errors appear to be random, and the magnitude of the errors is not changed with larger blur levels at noise levels from 0% to 4%. Figures B.19 to B.23 demonstrate that the bias errors are approximately the same across all blur levels investigated. The standard deviation error levels were found to increase slightly as the blur size increased, similarly to the 2D results. The noise level seemed to much more strongly affect the standard deviation errors in 3D as well.

4.1.6 Subset Size

The effect of subset size on both the bias and random errors in 2D and 3D can be seen in Figures B.25 to B.28. The plots show a slight negative relation between bias error and subset size in the noiseless case, and no relation between bias error and subset size for the noisy case. The plots do show a strong relation between standard deviation error and subset size for both the noiseless and noisy cases. Both the 2D and 3D random errors decrease in an approximately exponential manner as the subset size gets larger. The 2$^{\text{nd}}$ order shape functions had poor correlation at a 21x21 pixel subset size, and so sizes from 31x31 to 81x81 are plotted for the 2$^{\text{nd}}$ order shape function. The magnitude of the random errors in the 2$^{\text{nd}}$ order shape function is approximately double that of the 1$^{\text{st}}$ order shape functions. The relative computation time for each subset size is shown in Figure B.29. The computation time depends on the power of the computer and processors used and will vary significantly between different hardware configurations. Therefore, the relative time using a single computer was used to demonstrate the increase in computation time. The computation time increases in an exponential manner as the subset size increases.

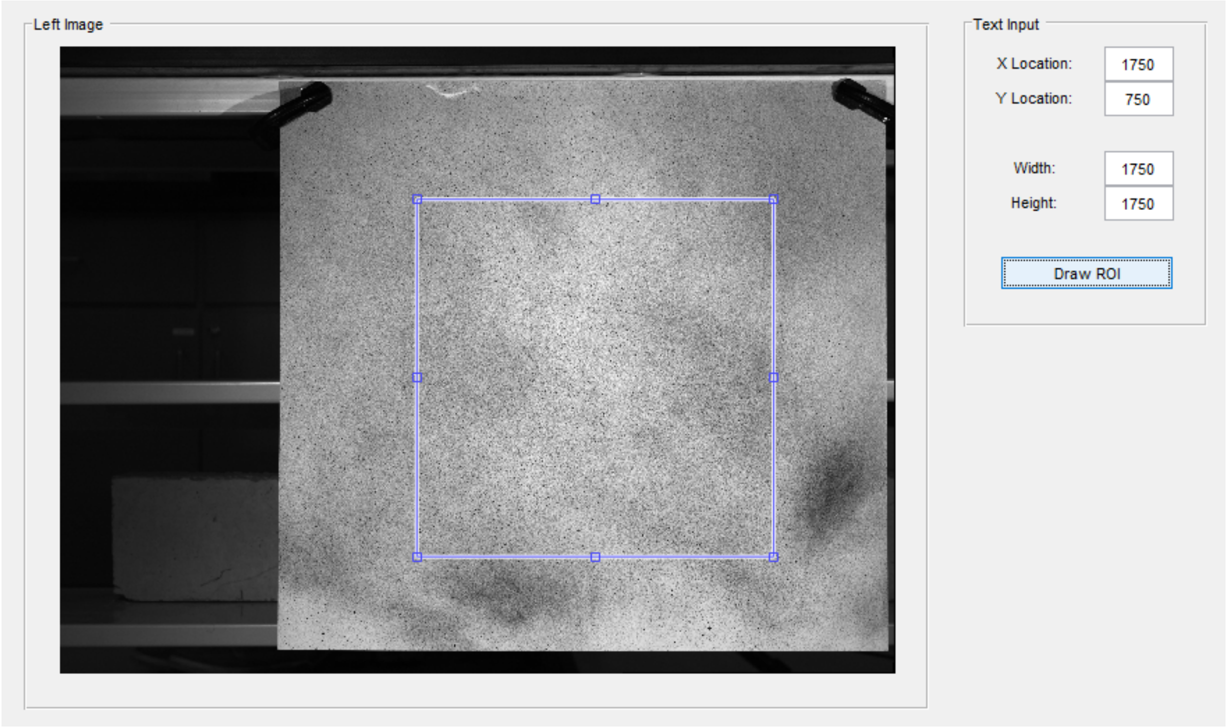
4.2 Laboratory Results

4.2.1 Camera Noise

20 bias images were captured using each camera. Each camera was evaluated independently to account for any variation between the cameras. Noise levels were evaluated using the standard deviation of the pixel intensities, which can in turn be used to simulate a Gaussian noise. The camera used as the left camera in all tests was found to have a standard deviation or noise level that ranged from 0.5846 to 0.5925 and a mean of 0.5877. The camera used as the right camera in all tests was found to have a noise level that ranged from 0.6316 to 0.6382, with a mean of 0.6346. These levels of noise correspond to mean values of 0.23% and 0.25% of the 8-bit values.

4.2.2 DIC Analysis Settings

The images were analyzed using a subset size of 51x51 pixels, a subset step of 75px, and a convergence criteria of $|p|= 10^{-5}$. Bicubic interpolation with no Gaussian blur was used in the correlation process. The subset step of 75px was chosen to allow for faster computation of the results considering the 12MP images being used. The region of interest used in the test is shown in Figure 4.3. The top left corner was located at (1750, 750) in the left image and the ROI had a width and height of 1750px. Images were captured at a rate of 30fps (or 30Hz). The initial seed point in each image was chosen at (2625,1625), which is the center of the ROI. This resulted in a 24x24 grid of analysis points, or 576 total analysis points.



X Location: 1750

Y Location: 750

Width: 1750

Height: 1750

Figure 4.3: Region of interest used in testing.

4.2.3 Displacements

An example plot of the displacement found by the DIC system in Run 3 is presented in Figure 4.4 at a single point corresponding to the LDV measurement location. The corresponding LDV velocity profile is shown in Figure 4.5, which was in turn highpass filtered at 0.25Hz and numerically integrated to generate the LDV displacement time series shown in Figure 4.6. The DIC point grid did not perfectly align with the LDV measurement location, and bilinear interpolation was used between the four points nearest to the LDV measurement location to calculate the resulting DIC displacement time series. The measurements found by both the DIC system and LDV followed the steady decay expected of the plate motion due to an impulse load.

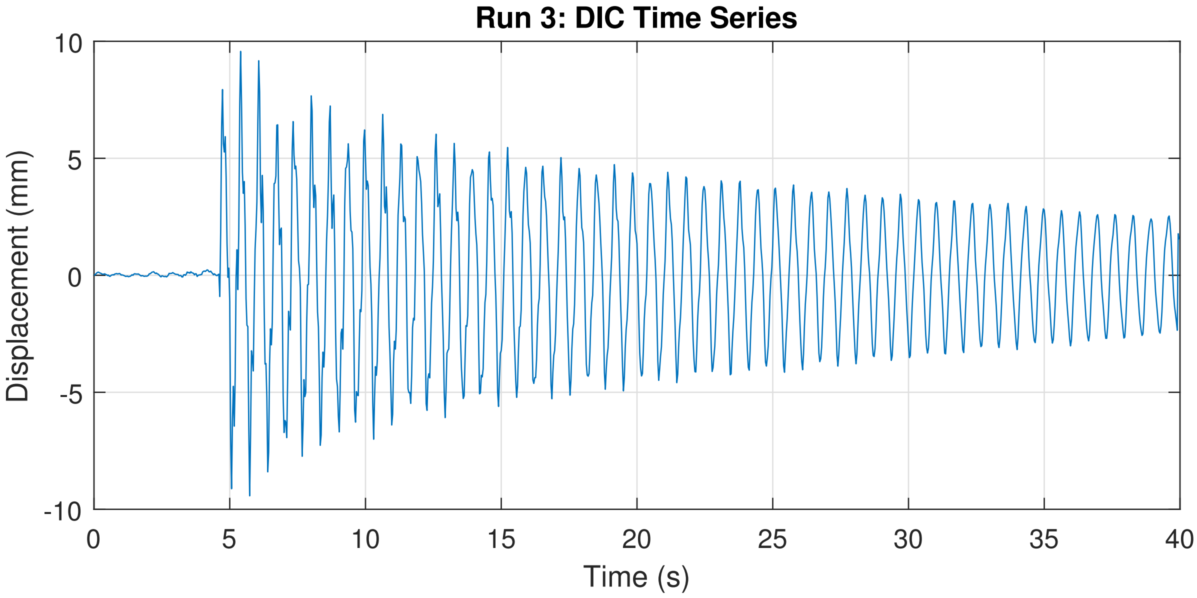


Figure 4.4: Example of displacements found by the DIC system.

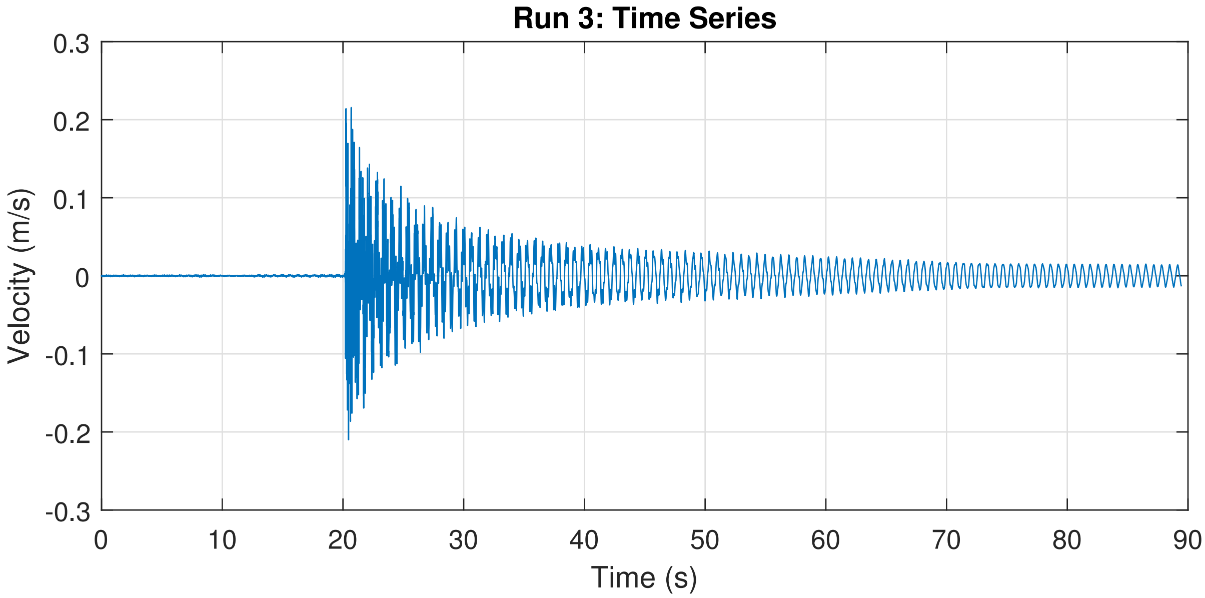


Figure 4.5: Example of velocities found by the LDV.

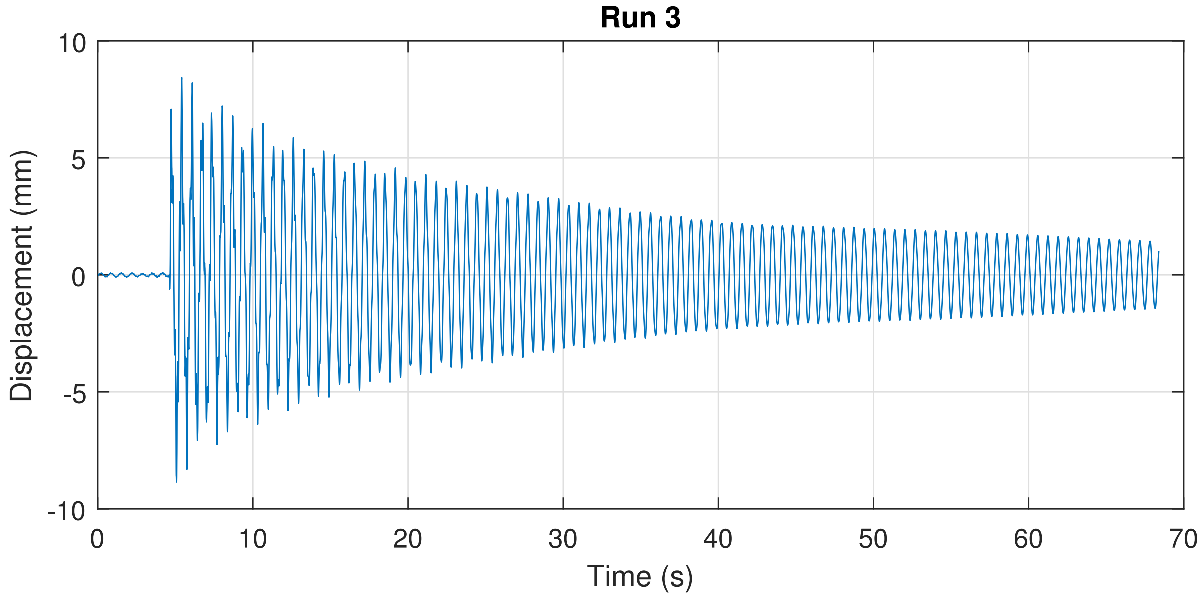


Figure 4.6: Example of numerically integrated displacements found by the LDV.

4.2.4 Modal Analysis Results

The DIC system provided full field displacements at known grid points and was excited at a single point by an impulse from an impulse hammer, so the system was treated as single input, multiple output (SIMO). The fundamental frequencies, damping ratios, and mode shapes were then determined through the use of built in MATLAB functions [2]. Figure 4.7 displays the stabilization diagram from Run 1 of the experimental testing. All of the stabilization diagrams may be found in Appendix B. The fundamental frequencies and damping ratios found through the experimental modal analysis using DIC can be seen in Tables 4.1 and 4.2. The stabilization diagrams clearly show the fundamental frequencies shown in Table 4.1. When fundamental frequencies for a given run were found to be very close, the mode with a reasonable damping ratio (typically below 1%) was chosen as the correct mode. The plate was only intrinsically damped, and was thus considered a very lightly damped system. Examples of the first 4 mode shapes from Run 3 can also be seen in Figure 4.8. The mode shape output was complex valued, so the amplitude of each point was taken to determine the amplitude of the vibration in a given direction. The mode shapes shown are for the Z-direction, which is approximately perpendicular to the plate surface. The values for the X- and Y-directions were found to be heavily contaminated by noise, resulting in unclear mode shapes.

Table 4.1: Fundamental frequencies from DIC (full field).

Mode	**Frequency (Hz)**						
	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**	**Average**	**Std. Dev.**
1	1.53	1.53	1.53	1.52	1.52	1.53	0.005
2	4.63	4.56	4.60	4.61	4.59	4.60	0.026
3	7.32	7.13	7.22	7.28	7.18	7.23	0.078
4	9.38	9.13	9.22	9.22	9.29	9.25	0.092
5	11.4	11.6	11.6	11.6	11.7	11.6	0.097
6	13.0	12.8	12.9	12.9	12.8	12.9	0.102

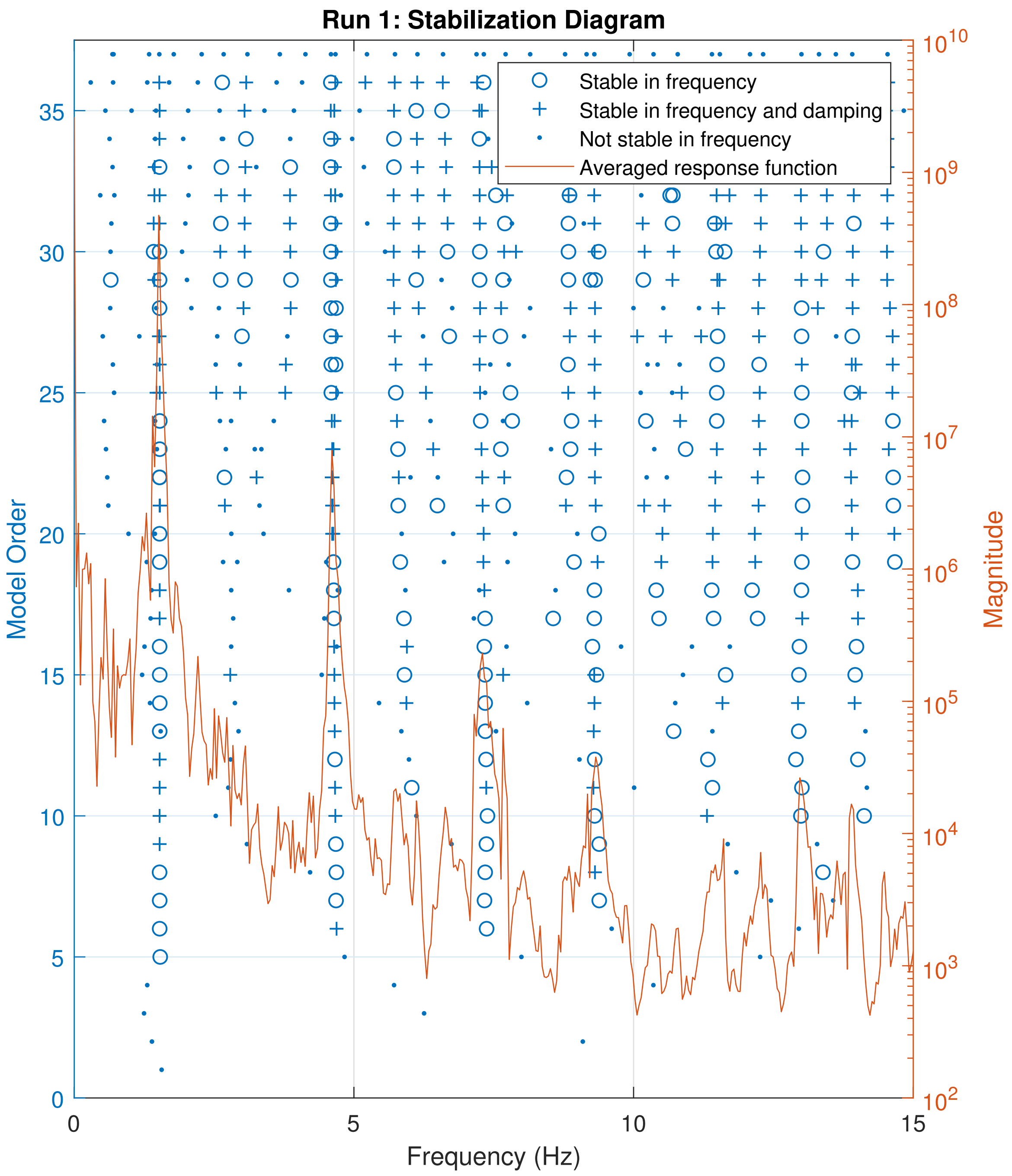
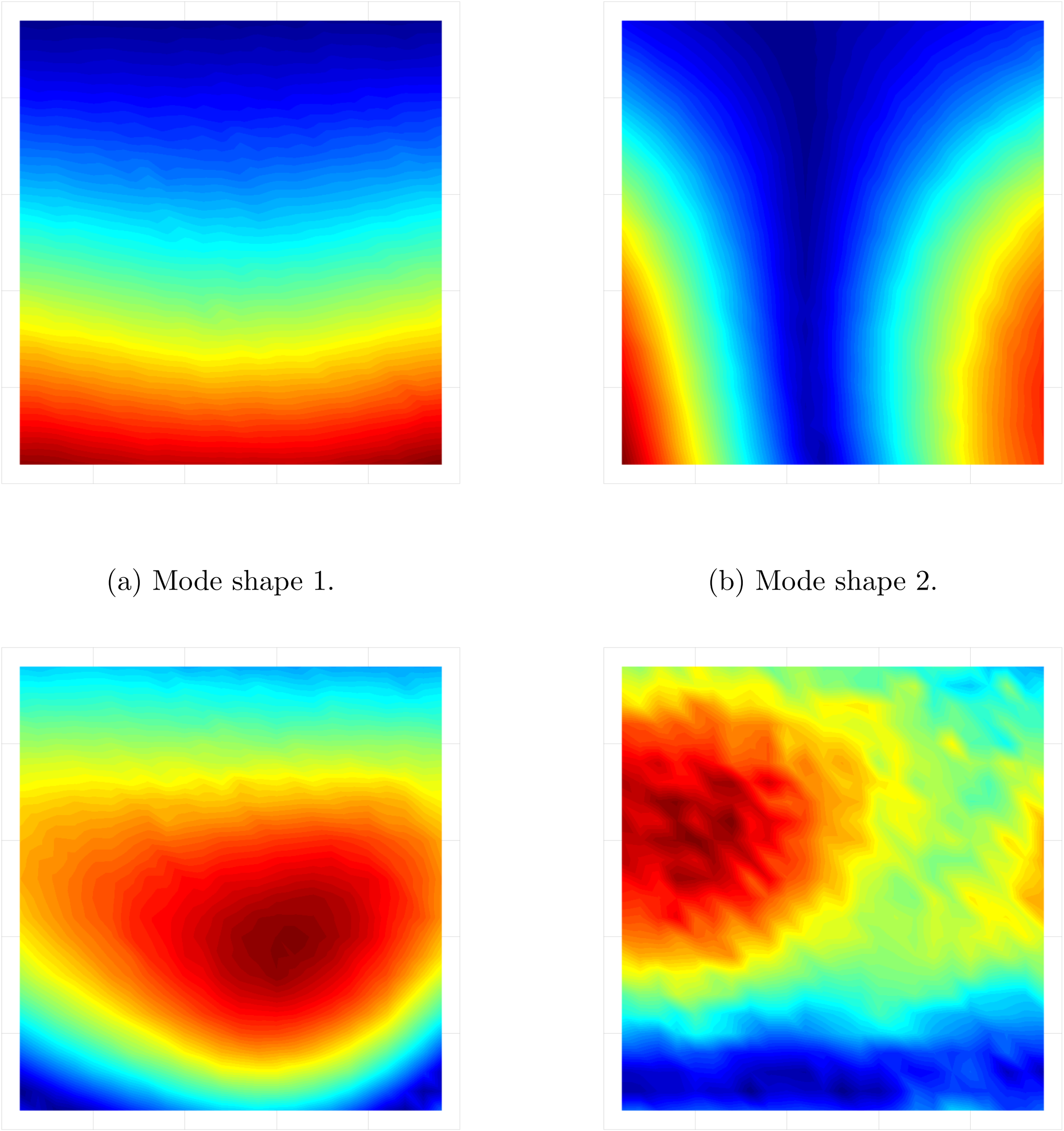


Figure 4.7: Stabilization diagram from the experimental testing, Run 1.



(a) Mode shape 1.

(b) Mode shape 2.

(c) Mode shape 3.

(d) Mode shape 4.

Figure 4.8: First four mode shapes found from the experimental modal analysis, Run 3.

Table 4.2: Damping ratios from DIC (full field).

Mode	**Damping Ratio (%)**						
	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**	**Average**	**Std. Dev.**
1	0.74	0.22	0.32	0.50	0.01	0.36	0.28
2	0.49	0.27	0.40	0.36	0.16	0.34	0.13
3	0.76	0.10	0.37	0.35	0.44	0.41	0.24
4	1.70	1.28	1.44	1.64	2.06	1.62	0.30
5	0.69	0.30	0.81	0.67	0.50	0.59	0.20
6	0.52	0.26	1.10	0.80	0.71	0.68	0.31

Figure 4.9 displays a typical power spectral density curve for the LDV signal from Run 3. The power spectral density curve was found by taking the amplitude of the FFT of the velocity. The peaks of the power spectral density curves were then taken as the fundamental frequencies from the LDV signal. The fundamental frequencies found from analyzing the LDV signal can be seen in Table 4.3.

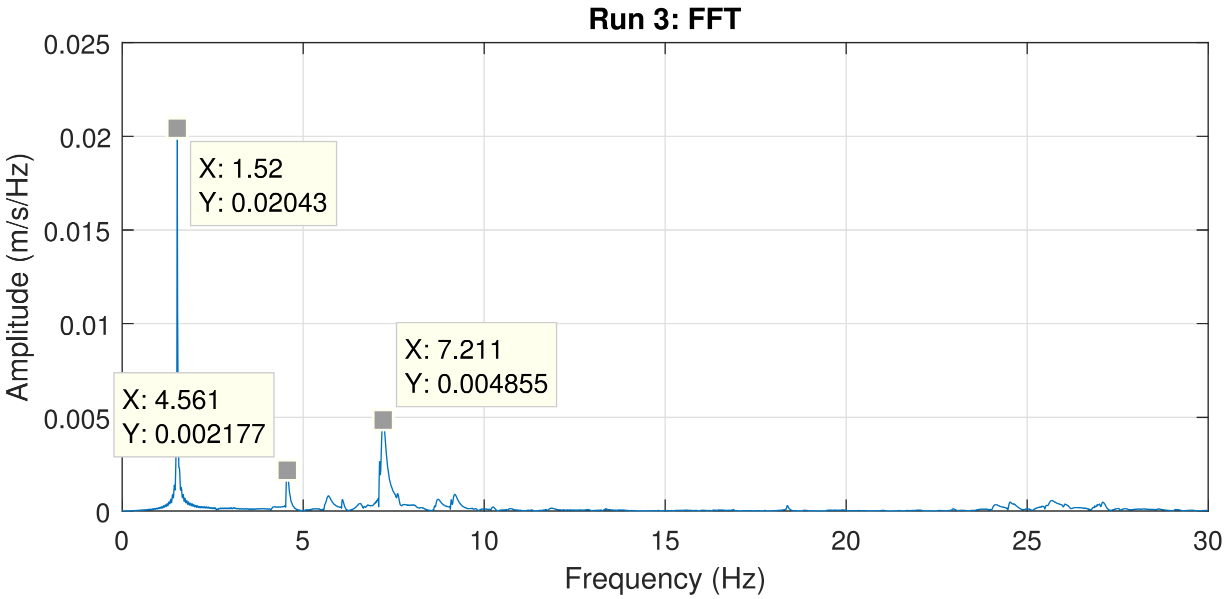


Figure 4.9: Power spectral density (PSD) curve for the LDV from Run 3.

Table 4.3: Fundamental frequencies from LDV (single point).

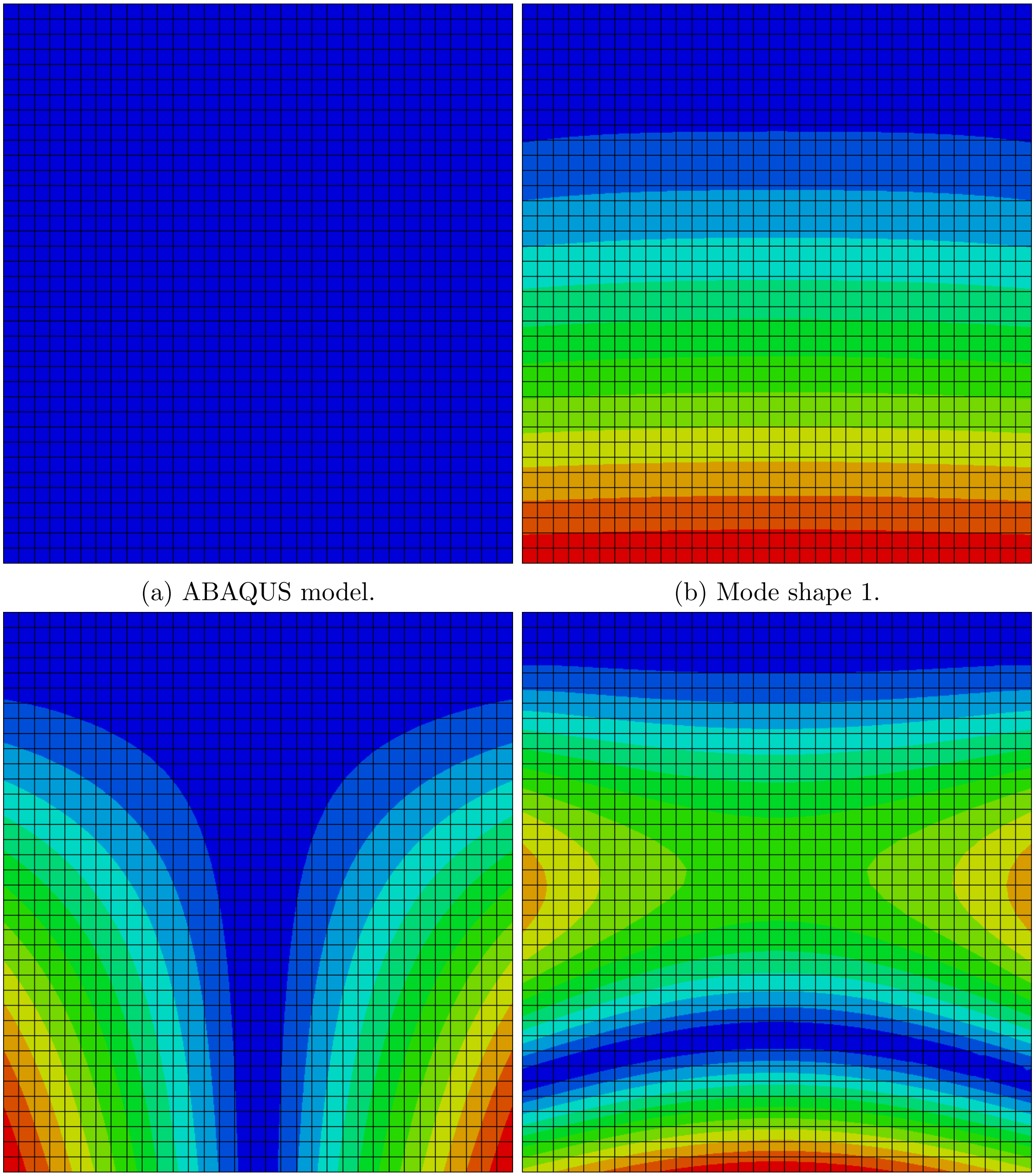
Mode	**Frequency (Hz)**						
	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**	**Average**	**Std. Dev.**
1	1.53	1.53	1.52	1.52	1.52	1.52	0.005
2	4.55	4.54	4.56	4.57	4.56	4.56	0.013
3	7.30	7.12	7.21	7.21	7.16	7.20	0.067
4	9.30	N/A	N/A	N/A	N/A	9.30	N/A

Finite Element Analysis Results

A finite element analysis (FEA) model was created in ABAQUS CAE Version 6.13-4 to compare to the fundamental frequencies and mode shapes of the experimental testing [1]. The plate was created through the use of linear shell (S4) elements with a 5 layer deep integration. The element size was approximately 15x15mm. The use of quadratic shell elements (S8) and an increase or decrease in the shell layers were briefly investigated, and found to not significantly affect the result. The plate thickness was measured using calipers, and found to average 0.62mm. The corners of the plate were fixed in all three directions and all three rotations. A modal analysis was conducted using the subspace model. The results from the FEA model are summarized in Table 4.4. The FEA model and first three modes found using the FEA analysis are shown in Figure 4.10. The first two mode shapes matched very closely to the experimental mode shapes from DIC, while the third FEA mode shape was similar to the fourth DIC mode in some runs. The fundamental frequencies of the first two modes are approximately 5 to 7% different than the experimental frequencies, but are still relatively similar.

Table 4.4: Fundamental frequency comparison with FEA model.

Mode	**Frequencies (Hz)**					
	FEA Freq.	**DIC Freq.**	**% Diff.**	**FEA Freq.**	**LDV Freq.**	**% Diff.**
1	1.63	1.53	+6.35%	1.63	1.52	+6.54%
2	4.35	4.60	-5.65%	4.35	4.56	-4.69%
3	N/A	7.23	N/A	N/A	7.20	N/A
4	10.1	9.25	+8.51%	10.1	9.30	+7.98%



(a) ABAQUS model.

(b) Mode shape 1.

(c) Mode shape 2.

(d) Mode shape 3.

Figure 4.10: ABAQUS model and first three mode shapes.

Chapter 5

Analysis & Discussion

5.1 Numeric Discussion

5.1.1 Speckle Size

The size of speckles used in numeric images can affect the results significantly. Previous studies have recommended between 2 and 7 pixels as the ideal size for speckles when using the FA-NR algorithm [10, 68, 71], but the ideal speckle range may be different for the IC-GN. Figure 4.1 shows that at very small speckle sizes, such as 1 or 2px, the error is relatively high, which was expected due to undersampling or aliasing as discussed by Zhou and Goodson [71]. Beyond 6px speckle size, the peak bias errors remain relatively similar (below 0.003px) until the correlation values degrade in images beyond 16px. Figure 4.2 shows that at large speckle sizes, the mean bias error increases, even though the overall peak bias error remains relatively constant. When the pixels approach larger sizes, the subset (which in this case was 31x31px) may start to become dominated by individual pixels, resulting in fewer unique features to correlate in the scene. While the peak bias error may remain similar, showing that the worst correlation in each case is similar in bias error magnitude, each individual subset shows more bias error. For this reason, a speckle size above 4px and below approximately 12px is recommended for use. These tests were conducted using a 31x31 pixel subset, and smaller subsets may have poor correlation when using 12px speckles, so speckles below 8px are further recommended for smaller subsets. Future analyses utilize an 8px speckle size to ensure that the speckle size falls in the ideal range while also remaining close to the previously recommended 2 to 7px for the FA-NR algorithm.

5.1.2 Shape Function

The numerical simulations presented here involve pure translation. The matching shape function for this is a 0^{th} order (rigid) shape function, which means that the 1^{st} and 2^{nd} order shape functions examined here are both overmatched shape functions. An overmatched shape function is recommended by Yu and Pan [66] if the deformation field is unknown, and Yu and Pan, Wang and Pan [60], and Wang et al [61] show that the errors for a matched and overmatched shape function are similar in magnitude. The 2D errors shown here also correspond with the theoretical derivations provided by Wang et al. [61] and Wang and Pan [60] for the standard deviation error associated with matched or overmatched shape functions. Even when using different interpolation functions, the 2^{nd} order shape function shows approximately the same bias error and approximately twice the random error of the 1^{st} order shape function in 2D. This demonstrates that the effects of interpolation method are independent of the shape function used in 2D.

Unlike in the 2D case, a 0^{th} order shape function will not typically capture any small translations and rotations between cameras, so a 1^{st} or 2^{nd} order shape function should always be used with stereoscopic DIC. Previous authors have recommended using a 2^{nd} order shape function to match the left and right images at a minimum [14]. The results from the 3D numerical simulations show that the shape order does affect the bias error slightly, but the errors are in the same range of ±1mm, showing that the effect is very minimal. However, the standard deviation errors approximately double by using a 2^{nd} order shape function instead of a 1^{st} order shape function. This result is similar to the 2D results, and shows that the standard deviation errors due to shape function are not related to errors specific to 3D analysis. A larger subset could be used to reduce the standard deviation error with the 2^{nd} order shape function.

5.1.3 Interpolation Function

The biquintic bias errors were found to be much smaller than the bicubic bias errors in 2D. This is due to the biquintic interpolation allowing more accurate modelling of the intensity changes between pixels than the bicubic interpolation and matches with previous work [44]. Higher order interpolants, such as biseptic or higher, would be expected to continue the trend, although at higher computational cost. Biquintic interpolation bias errors appear to be more sensitive to noise, likely due to the noise causing additional high frequency content to appear in the images that is picked up by the biquintic interpolation kernel. As the noise approaches 4%, the 2D bias error present due to biquintic and bicubic interpolation begin to approach similar magnitudes, although biquintic does remain lower.

The standard deviation errors for both interpolation methods are quite small without noise, with biquintic being slightly smaller. However, the addition of any noise overshadows any differences in the standard deviation errors, and the standard deviation errors can be seen as generally the same for both interpolation methods in 2D.

Adding a Gaussian blur to remove some of the higher frequency content in the images appeared to reduce the 2D bias error for both methods by approximately the same amount in very noisy images, while less noisy images saw more accuracy improvement in the bicubic method, which is discussed in more depth in Section 4.1.5.

Interestingly, in the 3D analysis, the use of a bicubic or a biquintic interpolation function did not appear to affect the bias or standard deviation errors. This is likely due to the error being dominated by the noise level present resulting in lower quality triangulation rather than the interpolation function. For this reason, a bicubic interpolation function is actually recommended for 3D works to allow for faster computation with minimal impact to result accuracy.

5.1.4 Noise

The 2D bias error was found to increase marginally with the addition of image noise, but did not generally increase the magnitude of error significantly. Low noise levels, such as 0 to 2% showed particularly strong invariance to image noise. The IC-GN algorithm is known to be relatively noise invariant in terms of bias error, while previous algorithms such as the FA-NR may increase bias errors by an order of magnitude or more [35]. The 2D standard deviation errors were expected to increase with increasing noise due the additional random variance introduced in the final correlations due to the noise. This was shown to be the case, with the standard deviation error increasing linearly with increasing noise. Previous authors [60, 66] have found similar results, with theoretical derivations showing that the standard deviation error should be linearly proportional to the image noise. The noise levels encountered also agree closely with the theoretical standard deviation errors. The 2$^{\text{nd}}$ order shape function had standard deviation errors double that of the 1$^{\text{st}}$ order shape function, but the standard deviation errors still increased linearly with the noise as expected.

The 2D bias and standard deviation errors demonstrate that the proposed system matches closely with expected theoretical results. The noise invariance of the bias error measures shows that the systemic errors in the system due to noise are very small. The random errors in the system vary proportionally to the image noise however, so image noise should still be reduced when possible.

The 3D analysis was particularly impacted by the presence of noise in the images. The level of noise directly impacts not only the standard deviation error as in the 2D analysis, but also the bias error. The inclusion of even 1% of noise changes the error from almost imperceptible to over 1mm in magnitude (for this geometry and situation). The sinusoidal pattern commonly seen in 2D disappeared with the inclusion of even 1% noise. This is likely due to 3D specific errors from triangulation, where otherwise small errors from image noise cause the position estimate in each image plane to vary slightly. The variation magnitudes are different between each image, causing the triangulation estimate to vary as well. The standard deviation errors increased linearly with the noise level as expected, similar to the 2D result.

Image noise has by far the most significant impact on 3D displacement error results, so it is critical that noise levels are reduced in 3D imaging systems. Image noise can be reduced by ensuring adequate light reaches the sensors to reduce shot (photon) noise, using higher bit depth images to reduce quantization noise, ensuring gains are set to the base (native) value, ensuring that the camera is left running to reach a steady state temperature to reduce read noise, and using shielded cables or avoiding interference laden areas to reduce read noise. Higher quality imaging sensors and lenses may also be able to reduce errors and provide much better results.

5.1.5 Noise Reduction

Applying a Gaussian filter to reduce noise in images is recommended by previous authors using the FA-NR algorithm to reduce bias errors in DIC [30]. The IC-GN algorithm is generally less sensitive to noise though [35], and so the effects of the Gaussian pre-filtering may differ.

In the 2D analysis, a 5x5 Gaussian filter was found to provide the greatest decrease in the bias error for the images when using bicubic interpolation in both 1st and 2nd order overmatched shape functions. This held true for both the noiseless and noisy image cases, although the improvement was smaller in the noisy image case. The 5x5 filter was found to be similar or even better than the 7x7 filter at reducing the bias error. However, the use of a Gaussian filter blurs edges in the image to reduce the high frequency content (acting as a low pass filter to reduce possible aliasing issues [49]), but in doing so decreases the image intensity gradients present. Decreasing the image intensity gradients results in lower SSSIG scores. The SSSIG score has previously been found to be inversely proportional to the standard deviation errors in the analysis, so decreasing the SSSIG will increase the standard deviation errors [60, 61, 66]. This was found to be the case for the simulation images, and the theoretical standard deviations errors as predicted by previous authors [60, 61, 66] match quite closely to those seen in the numerical simulations. In general, the larger the Gaussian blur, the higher the standard deviation error, in particular for noiseless images. In the presence of noise, the increased standard deviation error due to the Gaussian blur was mostly overshadowed by the increased error due to noise.

For biquintic interpolation in both the 1st and 2nd order overmatched shape functions in 2D, a 3x3 Gaussian filter was found to provide the greatest decrease in the bias error for the images. However, this was only found to be the case for noiseless images, and all of the noisy images demonstrated a similar level of bias error regardless of which filter was applied. Similar to bicubic interpolation, the larger Gaussian filters caused a larger increase in standard deviation errors, particularly for the noiseless images. In the presence of noise, the increased error due to the Gaussian filter was mostly overshadowed by the increases due to noise.

Overall, Gaussian filters were found to improve the bias error of bicubic interpolation at small noise levels in 2D measurements. As the noise level increased to 4%, the effects of noise were found to overwhelm the effects of the Gaussian filter, particularly for the 2nd order shape function. The 2nd order shape function was found to have correlation difficulties when using a 7x7 blur, with approximately 1 to 5 subsets (of 651 possible points) not converging in some images. The Gaussian filters were found to slightly improve the 2D bias error performance of biquintic interpolation in noiseless images, but were not found to appreciably improve the bias error performance of biquintic interpolation at real world noise levels. This is in contrast to work done by Pan [30] on Gaussian blurs in the FA-NR algorithm, where biquintic interpolation was still found to benefit significantly from filtering. However, the bias errors shown in Figure B.17 are of a similar magnitude to those found by Pan when using a 5x5 or larger filter [30]. In both cases, the standard deviation errors increased by a similar amount between bicubic and biquintic interpolation. These increases can be significant for noiseless images relative to the unfiltered standard deviation errors, but are generally insignificant relative to the impact of noise on the measurements.

A 5x5 Gaussian filter may be beneficial in 2D DIC analysis when used in conjunction with bicubic interpolation in relatively noise-free images ($< 2\%$ random noise), and can allow the bias error to approach or surpass the accuracy found by unfiltered biquintic interpolation as shown in Figure 5.1. The increase in standard deviation error can be an acceptable tradeoff for the improved computational speed of the bicubic interpolation. The increased standard deviation error may also be partially compensated for by using a slightly larger subset. No Gaussian filter is recommended for use with biquintic interpolation for any images due to the poor improvements in bias error and related increases in the standard deviation error.

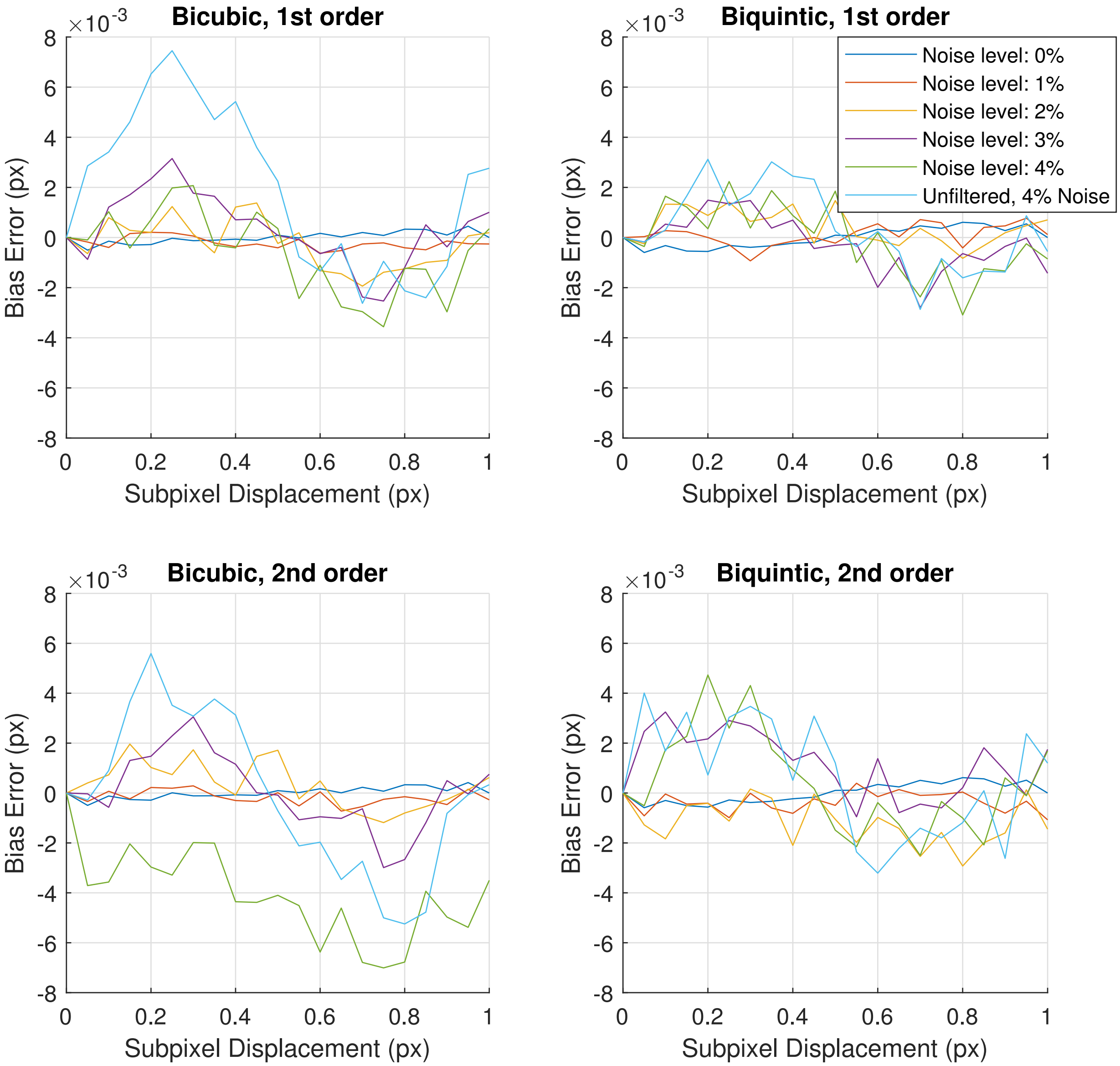


Figure 5.1: Bias errors present in analysis due to 5x5 Gaussian blur.

The Gaussian pre-blur did not appear to significantly affect the bias error at any noise level or for any shape function or interpolant when conducting 3D DIC analyses. No ideal blur size was found in reducing the bias errors for 3D analysis. However, the Gaussian blur did increase the 3D standard deviation errors as the blur increased in size. The 3D standard deviation errors did not appear to increase as much as the 2D result did, but still did increase noticeably. The increase in random error is due to decreasing the SSSIG of the subset, which results in larger random errors [35, 60].

The pre-blur does not appear to improve the bias error of any of the analysis, but does increase the standard deviation errors significantly. A larger subset could be used to reduce the standard deviation errors, but this would lead to increased computation time for larger subsets with no benefit to bias error. A Gaussian pre-blur may still be beneficial in 3D analysis to improve accuracy within each camera plane, but is not required or recommended for purely 3D analysis. This is likely due to the 3D triangulation being affected much more strongly by any noise in the system rather than by the Gaussian blur. The triangulation errors due to noise at each point largely overshadow the other error sources.

5.1.6 Subset Size

The bias errors for the IC-GN algorithm can be divided into bias errors due to noise and bias errors due to interpolation [35]. The interpolation component of the bias error decreases as the subset size increases, while the noise component is unaffected by subset size. This is generally shown by the 2D results, which show a slight decrease in peak bias error in all of the noiseless cases. However, the noise induced bias error component appears to be equally as strong as the interpolation induced bias error component, and the results with 4% image noise are less consistent, although they do trend slightly downwards in all cases but the 2nd order shape function with bicubic interpolation. The 3D peak bias errors do not show any consistent trends relative to the subset size, which is likely due to the triangulation errors caused by slightly random variations being much larger than the other error sources.

The standard deviation errors are inversely related to the SSSIG, which is in turn related to the subset size [29, 35, 60]. Therefore, the standard deviation errors are expected to decrease inversely to the subset size increase. Both the 2D random errors and 3D random errors follow this trend very closely for both the noisy and noiseless cases. The standard deviation errors are once again approximately double for 2nd order shape functions relative to the 1st order shape functions. The random errors using a 1st order shape function show good results when the subset is larger than approximately 41x41 pixels, while the 2nd order shape function begins to show good results when the subset is larger than approximately 61x61 pixels.

The relative computation time increases drastically as the subset grows, primarily due to the requirement to interpolate and correlate N^2 pixels, where N is the subset size. This makes smaller subsets exponentially faster than larger subsets, particularly in the time intensive interpolation step. However, using a smaller subset results in increased errors, and can possibly result in additional iterations prior to convergence. Subset sizes in excess of 61x61 pixels take a very long time to calculate and may add additional error if the shape function is not able to adequately capture the subset deformations, while subset sizes below 41x41 pixels result in large random errors. Therefore, a subset size between 41x41 and 61x61 pixels is recommended for analysis, although larger subsets are recommended for 2^{nd} order shape functions if computation time allows for it. This matches well to previous work recommending a 31x31px to 71x71px subset [66].

5.1.7 Recommended Analysis Parameters

Certain parameters are critical in DIC analysis, such as the subset size, speckle pattern, interpolation function, and shape function. Table 5.1 summarizes the recommended parameters for 2D and 3D DIC analysis.

Table 5.1: Recommended analysis parameters for 2D and 3D DIC.

Parameter	**2D DIC**	**3D DIC**
Speckle Size	4 to 8px	4 to 8px
Interpolation Function	Bicubic	Bicubic
Shape Function	Field known: 1^{st} Field unknown: 2^{nd}	Sequentially: Same as 2D Between cameras: 2^{nd}
Gaussian Blur	5x5 filter	No filter
Subset Size	41x41 to 61x61px	41x41 to 61x61px

The speckle pattern can be defined by the size of speckles, which are recommended to be between 4 and 12px. Previous authors have recommended between 2 and 7px speckles when using the FA-NR algorithm [71, 68], but the best results appear to fall in the 4 to 12px range for the IC-GN algorithm. Larger speckle sizes may result in poor correlation when using smaller subsets, so a speckle size of between 4 and 8 pixels is recommended for images.

The recommendations for 2D and 3D analyses differ. 2D analysis should make use of a bicubic interpolation function with a 5x5 Gaussian blur, which allows for the bicubic interpolation to approach the accuracy of biquintic interpolation while allowing for faster analysis. This comes at the expense of slightly increased random errors, which can be partially compensated for by using a slightly larger subset. Biquintic interpolation could also be used, but should not make use of any Gaussian blur if it does. A 1st order shape function should be used if it is known that the displacement field will be linear or rigid since a 2^{2nd} order shape function contains twice the random error without any improvement in bias error. If the displacement field is unknown or includes non-linear components, then a 2nd order shape function should be used to ensure the displacement field is captured accurately. The subset size should range between 41x41 and 61x61 pixels to balance the magnitude of random error with the computation time.

A 3D analysis should make use of bicubic interpolation, since both biquintic and bicubic interpolation had similar bias and random error levels and bicubic interpolation requires fewer computations. No Gaussian blur is recommended for 3D analysis due to Gaussian blurs increasing the random errors without improving the bias error result. A 1st order shape function should be used if possible, particularly between the sequential images from each camera, due to the lower random errors present when using a 1st order shape function. However, if the displacement field is unknown, a 2nd order shape function should always be used. Additionally, if the cameras are angled and the displacement field is unknown or likely to be linear, a second order shape function should be used to match between the left-right image pairs. This will allow for warping based on the different camera perspectives, which a 1st order shape function may not be able to capture. The subset size should range between 41x41 and 61x61 pixels to balance the magnitude of random error with the computation time.

5.2 Laboratory Discussion

Of the 5 runs completed for the modal analysis experiment, Run 2 was found to be affected significantly by noise or other errors prior to the impact hammer striking the plate, resulting in a bias shift for the run. The other 4 runs were generally of high quality, although slight bias shifts existed for each of the runs depending on the initial ambient motion of the plate and when the reference image was captured during the ambient motion. This is an issue with controlled vibration experiments in general, but one that is easily remedied by subtracting the mean of the signal from each signal. This bias shift is likely at least partially due to small ambient vibrations disturbing the plate and the reference images not being captured when the plate was at exactly 0 displacement.

5.2.1 Comparison to Laser Vibrometer

The DIC system displayed a slight mean bias in the signals that depended on when the reference image was captured. Once this mean was corrected, the DIC system displacement signal and the LDV displacement signals were synchronized through maximizing the cross correlation between the signals, which resulted in excellent agreement for Runs 1, 3, 4, and 5. Run 2 found a false peak correlation due to the presence of a disturbance at the start of the signal, and therefore the second highest correlation score was used to synchronize the time series signals. Due to the sampling rate mismatch, the DIC results were resampled from 30Hz to 240Hz by an integer increase of 8. Resampling was accomplished using a built in MATLAB function which applies a lowpass FIR filter to avoid aliasing and accounts for lag due to the filtering [2]. An example of the synchronized and overlaid time series for Run 3 can be seen in Figure 5.2.

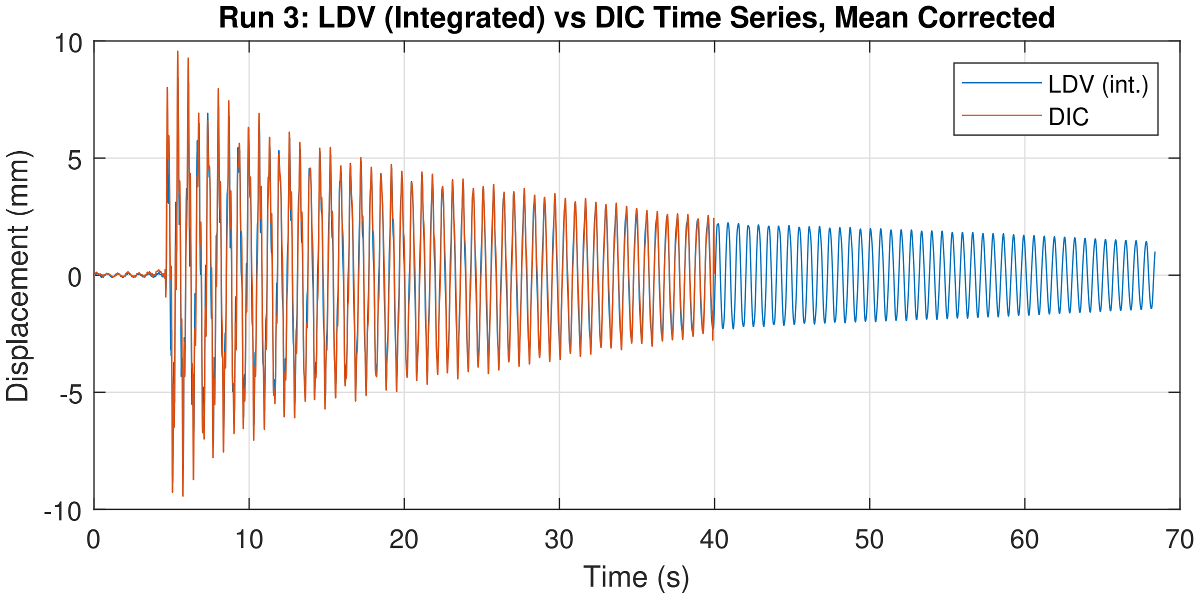


Figure 5.2: Displacements from the DIC system and numerically integrated LDV signal.

The DIC displacement signal at the LDV measurement point matched very well with the displacement signal from the LDV in all runs. Both the phase and the magnitudes of the signals are almost identical, although the DIC system does exhibit slightly larger peaks than the LDV. The largest peak difference and the average absolute peak difference are shown in Table 5.2. In each run, there appeared to be one point with a significantly larger DIC displacement, while the remainder of points were much more similar. This is also shown by the maximum differences of 6 to 14% of the total displacement while the mean difference was below 2% in the final three runs and approximately 3% for the first two runs. The normalized cross correlation of Run 2 was lowest, with a score of 92%, while Runs 1, 3, 4, and 5 had cross correlation scores from 97% to 98%. The noticeably lower cross correlation score in Run 2 is due at least in part to the significant noise present in the DIC signal prior to the impact hammer strike. Overall, the DIC system matches very closely with the LDV signal, reinforcing the accuracy of the DIC system in real world dynamic applications. These tests were performed on a relatively flexible plate that exhibited noticeable deformation, and similar tests with more stiff structures may find more noise contamination due to the smaller displacements to be measured.

Table 5.2: Differences between the DIC and LDV systems.

Run	**Max Peak Diff. (mm)**	**% of Signal**	**Mean Peak Diff. (mm)**	**% of Signal**
1	2.081	14.3	0.462	3.18
2	1.776	12.2	0.511	3.52
3	1.133	7.80	0.234	1.61
4	1.334	9.18	0.242	1.67
5	0.906	6.24	0.227	1.56

The other differences that are readily apparent are at the start of the signal for Runs 1 and 2 and at the end of the signal in Run 1. There is additional noise in the DIC signal that is not present in the LDV signal at the start of Runs 1 and 2. There is also a slight desynchronization between the DIC and LDV signals at the end of Run 1, which is due to a memory issue in testing. These issues are discussed in depth in Section 5.2.3.

5.2.2 Modal Analysis

A preliminary examination of the stabilization diagrams from the DIC analysis and power spectral density plots from the LDV measurements show good agreement in the fundamental frequencies. The stabilization diagrams calculated for the DIC analysis show strong peaks at approximately 1.5Hz, 4.5Hz, 7.2Hz, and 9.2Hz, which match closely with the fundamental frequencies calculated via MATLAB's $modalfit$ using the least squares complex exponential method [2]. These were also the frequencies that exhibited reasonably low levels of damping. This implies that these modes are the true physical modes of the structure. The power spectral density plots for the LDV measurement also show clear peaks at approximately 1.5Hz, 4.5Hz, and 7.2Hz. There is a fourth peak at approximately 9.3Hz in the first run power spectral density plot, but it is not a clear frequency in the later runs. Two additional modes can also be seen in the stabilization diagrams from the DIC measurement at approximately 11.5Hz and 13.0Hz. The mode at 11.5Hz was found to produce an expected mode shape as well, which reinforces the 11.5Hz mode being a physical mode of the structure.

Fundamental Frequencies

The frequencies found from the DIC analysis and the LDV measurements were found to align quite closely. A statistical evaluation of the difference was conducted using a paired, two tailed t-test. The variance in testing was assumed to be normally distributed due to the Gaussian nature of image noise, random nature of the 3D DIC bias error, and the constant 3D DIC standard deviation error found from numeric testing. The null hypothesis was chosen to be that the means of the two samples do not differ (i.e., $\mu_d = \mu_1 - \mu_2 = 0$). The test could only be completed for Modes 1 to 3 due to a lack of data for the 4th mode in the LDV measurements, and the results are summarized in Table 5.3, where $\bar{d}$ is the mean of the differences between the pairs, s_d is the standard deviation of the differences between the pairs, t is the calculated t value, and p is the corresponding probability.

Table 5.3: Statistical significance test of fundamental frequencies.

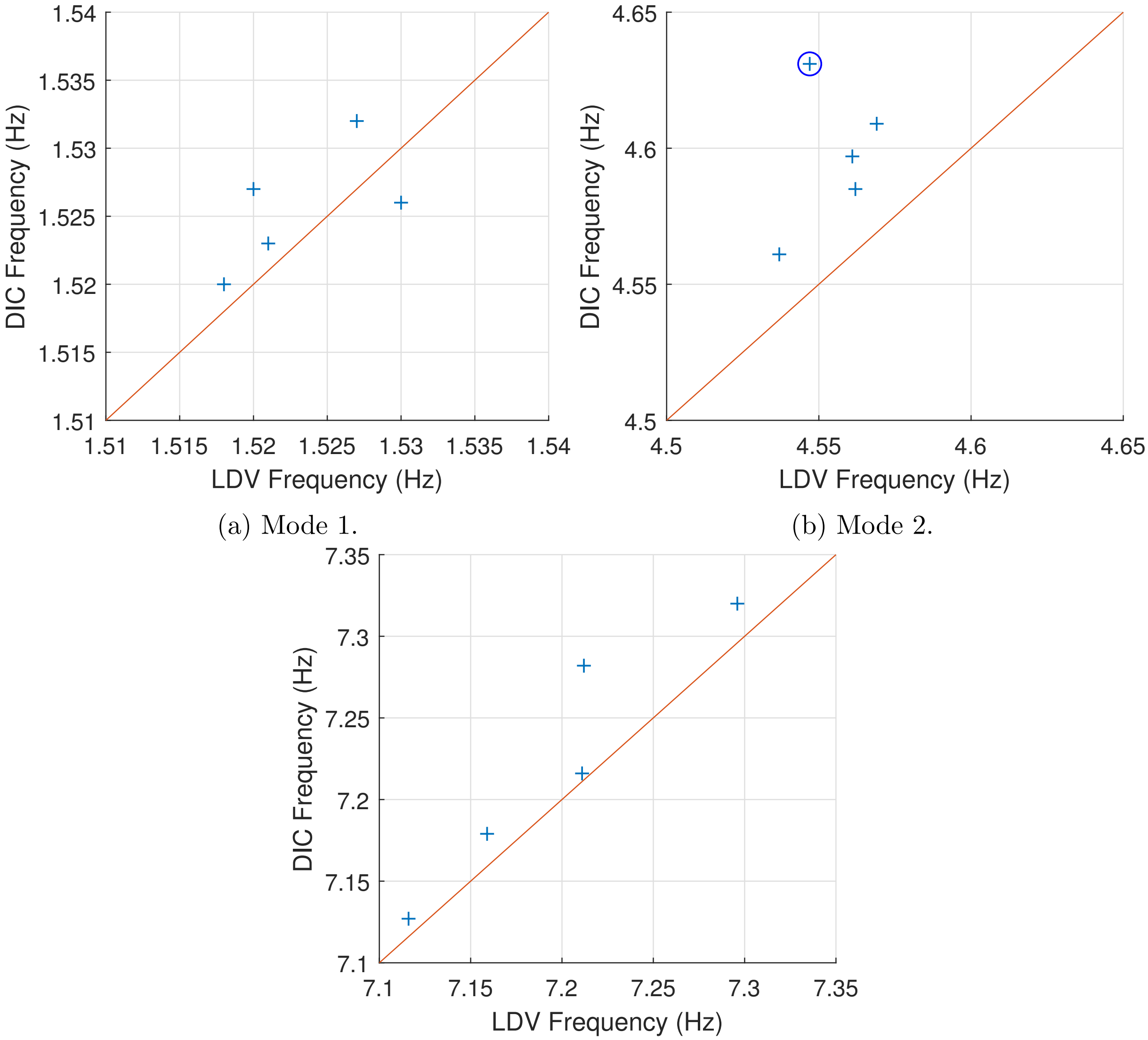
Mode	$\bar{d}$	s_d	t	p	**Reject Null?**
1	0.0024	0.0042	1.290	0.266	No
2	0.0414	0.0249	3.712	0.021	Yes
3	0.0260	0.0257	2.262	0.086	No

Modes 1 and 3 both result in failing to reject the null hypothesis. This means that the means are not different at a statistically significant level ($p > 0.05$). Mode 2 resulted in a rejection of the null hypothesis, but it should be noted that $p = 0.021$, so the null hypothesis would not be rejected for a more stringent condition of $p < 0.01$. This means that the mean values are different at a statistically significant level ($p < 0.05$). It should also be noted that the results only show a mean difference of approximately 0.041Hz, or less than 1% of the frequency measured, so there is little practical difference between the means in the context of operational modal analysis. To evaluate the means as truly different, the sample difference should be both statistically significant and practically different in the context of the problem [59]. Therefore, the mean frequencies from Mode 2 are found to be functionally the same, although with a slightly larger difference than Modes 1 or 3. The mean frequencies for Modes 1 and 3 are significantly the same between the DIC system and LDV measurement.

The LDV and DIC frequencies were plotted against each other to compare to an ideal fit. Ideally, the points should all fall on the line shown, which would indicate exact agreement between the DIC system and LDV. Practically speaking, an excellent result would be one where the points fall almost on the line and do not consistently fall above or below the ideal fit line, indicating minor random errors and no systemic errors [40]. Figure 5.3 shows that the DIC and LDV results are in very good agreement for Modes 1 and 3, although the DIC system very slightly overpredicted the frequencies. Mode 2 showed the furthest deviation from the ideal fit, with all 5 frequencies overpredicted by the DIC system. One data point (circled) corresponding to Run 1 noticeably deviates from the trend as well. That point shows the DIC system overpredicting the frequency by much more than the other data points, although still only by about 0.08Hz.

Overall, the results from the LDV and DIC are generally in very good agreement, with only very minor differences between the frequencies. There is some indication of a very slight systematic bias error in the DIC that overpredicts the fundamental frequencies, but it is typically in the order of 0.02Hz or less.

The FEA model gave similar results for the first two modes, although a model calibration was not carried out to improve the model accuracy. The FEA model found the first two frequencies to approximately ±6.5%, and the mode shapes matched very closely. The FEA model did not provide the third mode found by the DIC and LDV analyses, but did find the fourth mode. However, the fourth mode was overestimated by the FEA model by approximately 8.5%. Higher order modes were generally found to be overestimated by the FEA model as well.



(a) Mode 1.

(b) Mode 2.

(c) Mode 3.

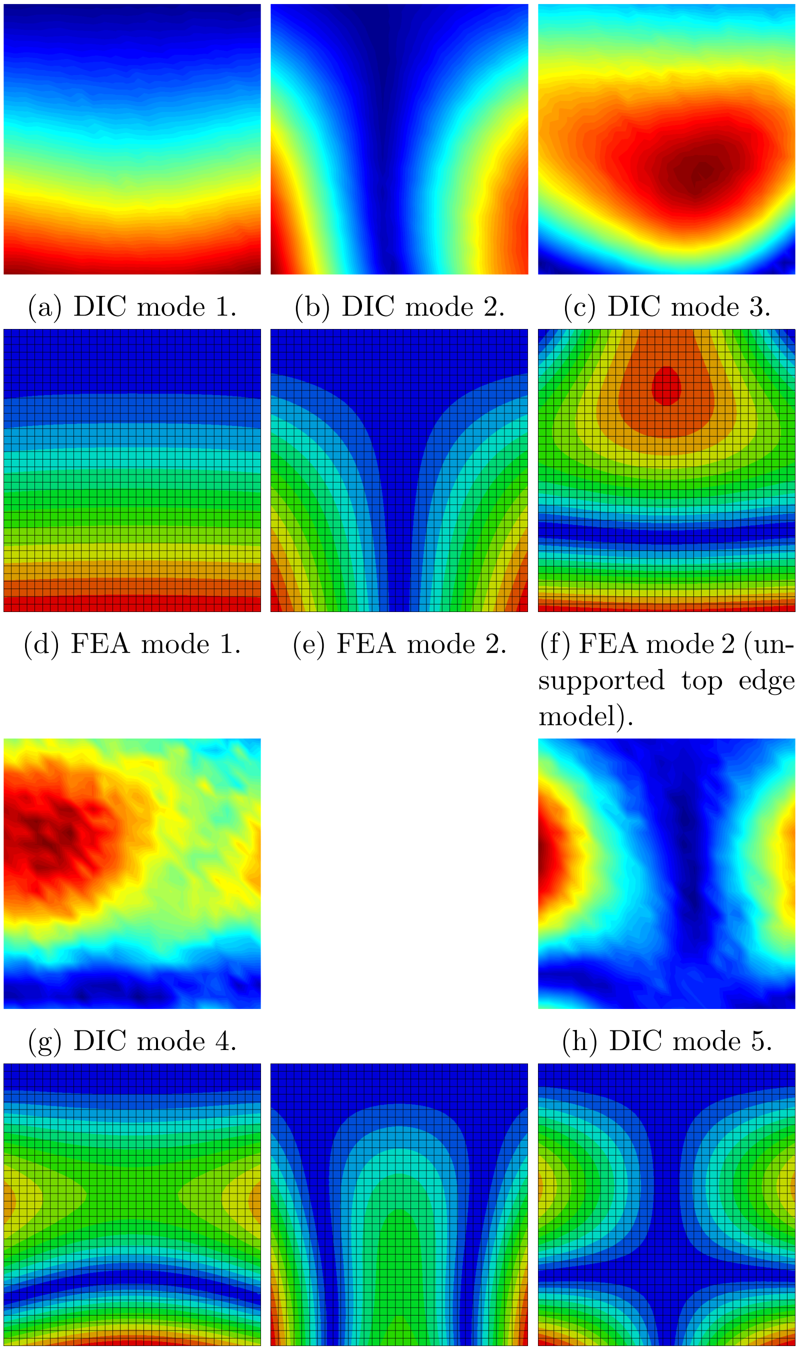
Figure 5.3: Graphical comparison of LDV and DIC frequencies.

Mode Shapes

The mode shapes from the DIC system and FEA analysis can be found in Figure 5.4. It should be noted that the very top and bottom edges of the plate were not analyzed in the experimental test, so the 0 displacement point is actually slightly above the mode shapes shown for the DIC system. The first 2 mode shapes were found to be very similar throughout the 5 test runs, as well as being very similar to the FEA mode shapes. Mode 1 from both systems is the first flexural mode in the vertical direction, and both mode shapes increase from 0 at the top (clamped) edge to a maximum at the bottom (free) edge. The FEA mode shape contains almost flat gradients in the horizontal direction, while the DIC mode shapes display a slight curve in the gradients along the horizontal direction. Mode 2 from both systems is a twisting of the bottom two corners. Both mode shapes display peak displacements at the bottom corners and 0 displacement at the top corners. The FEA mode shape is perfectly symmetric, while the DIC mode shape is slightly to the left of the vertical symmetry axis. This may be due to the hammer excitation occurring on the right half of the plate, slight variation in the material properties across the plate, or minor differences in support at the top corners.

The third DIC mode shape does not match with any of the FEA mode shapes. The frequency found for Mode 3 was also not found in the FEA analysis. This mode was captured by both the DIC and LDV systems, implying that it did occur in the experimental testing. The errors may be due to the FEA boundary conditions being idealized, since a similar model with no support condition along the top edge (shown in Figure 5.4f) found a third mode shape similar to that found by the DIC system. The fundamental frequency for the mode shape was also close (FEA: 6.32Hz, DIC: 7.23Hz), although all frequencies were underpredicted by the unsupported top edge model discussed. The mode shape still differs along the top and bottom edges however, with the DIC mode shape displaying no displacement along the top edge and bottom corners, while the FEA mode shape allows free movement along the top and bottom edges. The top edge may have been partially supported in the experimental testing, allowing the middle to deform more freely while preventing motion along the very top edge. The bottom edge may have deformed during testing, but the deformed area may have fallen outside of the ROI in the DIC analysis.

The fourth DIC mode shape is somewhat similar to the third FEA mode shape in some runs, although much more contaminated by experimental noise. The fourth DIC mode shape exhibits its peak displacements at the middle of the left and right edges, while the FEA mode shape has significant displacement at the same locations. The FEA mode shape has a large displacement along the bottom edge in a narrow band, while the DIC mode shape does not show the displacement along that bottom edge. This may be due to



(a) DIC mode 1. (b) DIC mode 2. (c) DIC mode 3.

(d) FEA mode 1. (e) FEA mode 2. (f) FEA mode 2 (unsupported top edge model).

(g) DIC mode 4. (h) DIC mode 5.

(i) FEA mode 3. (j) FEA mode 4. (k) FEA mode 5.

Figure 5.4: Mode shapes from DIC and FEA model.

the peak displacement point at the bottom being slightly outside of the ROI of the DIC analysis. This mode shape was the least consistent between test runs, and some test runs did not match the FEA mode shape.

The fourth FEA mode shape does not have a corresponding DIC mode shape, which may be due to FEA analysis error or difficulties capturing the displacements using the DIC system. The fourth and fifth FEA fundamental frequencies were very similar and at approximately 15Hz however, so the experimental plate may have exhibited the theoretical fifth mode at just below 15Hz and the theoretical fourth mode at just above 15Hz, preventing it from being captured during testing.

The fifth DIC and FEA mode shapes match up quite closely, although the fifth DIC mode shape shows some noise contamination. However, both mode shapes display displacements along the left and right edges, as well as in the bottom corners, with the area along the vertical line of symmetry showing no displacement. The DIC analysis did not extend to the very corners of the plates, so the displacements at the very bottom corners were not captured. Both also show no displacement in the top left and right corners.

In general, the DIC system was able to accurately capture the first 2 modes in every test run, as well as possibly capturing a third mode that was not present in the FEA model. It was also able to capture the fourth and fifth modes of the vibrating plate, although lower signal-to-noise ratios in the fourth and fifth modes resulted in noisy mode shapes. This was particularly true for Runs 1 and 5, which had difficulty finding higher order modes beyond the third mode. Overall, the DIC system was most accurate at finding the first three modes of the system, and was useful in finding higher order modes in some cases.

5.2.3 Testing Errors

The largest error encountered during testing can be seen in the final 1 to 2 seconds of Figure 5.5. The cameras were initially set to capture 1250 images per camera per run, but the end of testing typically featured a significant drop in framerate. The likely cause of this framerate drop is due to there being no contiguous system memory (RAM) available to store pictures. When the images are captured, they are sent to the framegrabber cards, which piece the images together and store them in RAM. However, the computer capturing images was limited to 32GB of RAM, some of which was additionally used by the operating system and by the camera control system. As the number of images approached 1200, the computer began resorting information on the RAM as required to store the next images, but in doing so caused frames to drop. The other 4 runs were completed with a limit of 1200 images per camera per run, which resolved the issue.

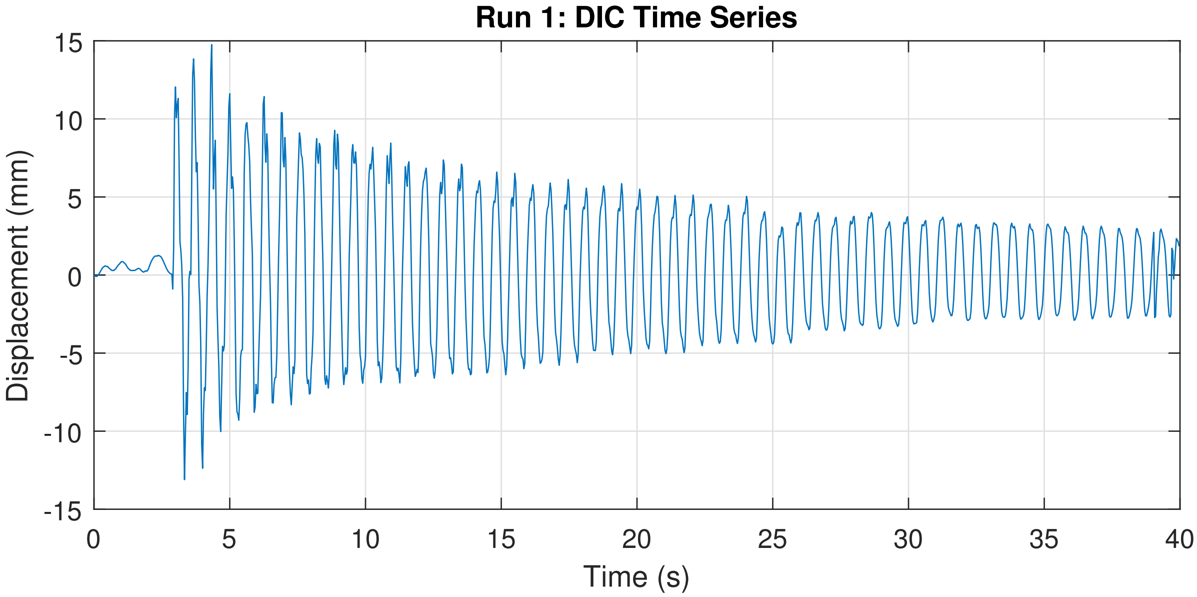


Figure 5.5: Error in Run 1.

The plate did not return fully to rest between the tests or may have been lightly excited by ambient vibration as shown by the first few seconds of each test. Both the LDV and the DIC system picked up on very small displacements during this time. This may have led to a small mean bias error occurring in the DIC measurements based on when the reference image was initially chosen. If the reference DIC image was taken while the plate was closer to the cameras, the recorded measurements would show larger positive displacements (away from the camera) and smaller negative displacements (towards the camera). This bias error was easily removed by subtracting the mean of the signal from each point, which gave almost identical results in both magnitude and phase for the LDV and DIC displacement signals.

Run 1 and Run 2 both had additional errors occur just prior to the test starting in the DIC system. These errors are likely attributable to human errors, such as the operator contacting the plate or the plate support and causing minor fluctuations prior to applying the impact hammer. Additionally, significant shadowing of the test surface caused by the operator setting up to use the impact hammer may have caused significant changes in lighting temporarily prior to impact, causing more extensive errors, although the normalized correlation functions should minimize the effects of lighting changes during testing.

Chapter 6

Conclusions and Recommendations

A 3D DIC system was created using the MATLAB programming language and modern DIC algorithms. This system will be provided open-source, and was written in MATLAB to allow for easy modification. A numeric study using simulated images was conducted to determine the bias and random errors present within the system and to determine the ideal parameters for both 2D and 3D DIC analysis. A real world verification experiment was then conducted on a thin, vibrating plate. The vibrating plate test was used to demonstrate the accuracy of the system through comparison to an LDV and to demonstrate the real world applicability and advantages of 3D DIC for modal analysis.

6.1 Numeric Results

6.1.1 Speckle Size

A speckle pattern of between 4 and 12px was found to provide the best results for simulated images in DIC analysis. Speckles sizes below this found large peak bias errors, while speckle sizes above this found large mean bias errors. Between 4 and 8px is recommended for the speckle size for simulated images in case a small subset, such as 11x11 or 15x15, is used in the analysis. This would prevent a 12px speckle size from dominating the subset and causing decorrelation.

6.1.2 2 Dimensional DIC

The 2D numeric results were very similar to previous literature results for both the FA-NR and IC-GN algorithms. The 2$^{\text{nd}}$ order shape function had twice the random error as the 1$^{\text{st}}$ order shape function, and the error levels for each were very similar to theoretically derived results. This led to the recommendation of using a matched shape function if possible, although a 2$^{\text{nd}}$ order shape function was recommended in the event of an unknown displacement field. The bias errors due to biquintic interpolation were smaller than for bicubic interpolation as expected, and the interpolation method did not significantly affect the random errors. This held true for both 1$^{\text{st}}$ and 2$^{\text{nd}}$ order shape functions as well as for noisy and noiseless images. The effects of image noise were similar to previous works on the IC-GN algorithm as well, where an increase in image noise resulted in a linear increase to random error and only slight effects to bias error. The use of a Gaussian blur for noise reduction acted as expected, resulting in reduced bias errors, especially for the bicubic interpolation, and a small increase in random error. The random errors increased slightly due to the Gaussian blurs, but were more strongly affected by the increased noise levels. A 5x5 Gaussian blur with bicubic interpolation was recommended for future 2D DIC analysis, which is in line with recommendations by previous authors using the FA-NR algorithm. The subset size was recommended to be between 41x41px and 61x61px, which is similar to the range of 31x31px to 71x71px found by previous authors.

6.1.3 3 Dimensional DIC

3D DIC errors have not been studied in depth to date, so there is little literature to compare to. However, comparison to the 2D results reveals a number of insights. The shape function (which was overmatched in this case) was found to act similarly in 3D and 2D, where it had minimal effect on the bias errors, but a 2$^{\text{nd}}$ order shape function had twice the random error of a 1$^{\text{st}}$ order shape function. A 1$^{\text{st}}$ order shape function is recommended for registration between all of the left or all of the right images, but a 2$^{\text{nd}}$ order shape function is still recommended for left-right image registration based on previous author recommendations. A significant difference between 2D and 3D analysis was shown for the interpolation function. A biquintic interpolation function was found to have similar bias errors as a bicubic interpolation function in 3D. This contrasts sharply with the 2D result, where a bicubic interpolation function had larger bias errors than a biquintic function. This is likely due to errors specific to a 3D analysis, such as errors in the triangulation process. The interpolation bias took the form of a sinusoid for noiseless images, but it did not take the shape of a sinusoid when any noise was present. This further implied that slight image noise caused each analysis point to triangulate slightly differently. The standard deviation errors were not noticeably affected by the interpolation function though, which is similar to 2D, so it is not due to random error or noise. A bicubic interpolation is recommended for 3D analysis due to similar bias error levels as the biquintic interpolation at a lower computational cost. The addition of image noise caused the bias errors to significantly increase in the 3D analysis, unlike in 2D where the image noise only slightly affected the bias errors. This is again likely due to errors specific to 3D, such as triangulation errors due to noise. The random errors were affected similarly however, and increased linearly with an increasing noise level. This highlights the importance of low noise levels in 3D DIC applications. The noise can be reduced through the use of high quality sensors, good lighting, and a good speckle pattern when possible. The Gaussian blur did not appear to significantly affect 3D analysis. The 3D specific errors appeared to dominate the bias error results, and did not appear to be affected by the Gaussian blurs. This is unlike 2D, where a Gaussian blur was found to reduce the bias errors significantly. The random errors were also increased slightly by the Gaussian blur, although the random errors were still dominated by the image noise level. Therefore, no Gaussian blur was recommended for 3D analysis due to a Gaussian blur increasing random errors without decreasing bias errors.

6.2 Laboratory Results

6.2.1 Accuracy of System

The DIC system was found to be very close to the measurements provided by the LDV system. Run 2 had the lowest cross correlation at 92%, but the other 4 runs had cross correlations in the range of 97% to 98%. The main differences were found in the peak measurements of the signals, where the DIC system showed an average of between 1.5% and 3.5% larger displacements than the LDV system. A single peak in each run was noticeably higher in the DIC measurement by up to 14%. The first two runs may have experienced some operator errors in the DIC measurement based on the initial portion of the runs, which is further borne out by the more consistent accuracy achieved by Runs 3, 4, and 5, which averaged 1.5% to 1.7% differences and had a maximum difference of 6% to 10%.

6.2.2 Modal Analysis

Digital image correlation was shown to be a valuable tool in modal analysis. The setup requirements are very small, only requiring a speckle pattern or other trackable features, setting up 1 or 2 tripods, and setting up the cameras. The cameras are able to capture the full field displacements, allowing for not only out of plane movement from a distance, but also in plane movement. This may allow capture of mode shapes that do not have a large out of plane component. DIC can be used to capture many points in a structure, which allows for more modes to be picked up even when a mode shape amplitude is 0 at a single measurement point. DIC systems also avoid introducing additional mass to systems because no direct contact is required during testing.

The DIC system created here was evaluated through an experimental test of a thin plate structure. The thin plate was clamped to a rigid frame and subjected to an impulse loading. The DIC system and an LDV were used to capture data during excitation, and the LDV was used to provide a baseline comparison to the DIC system measurements.

The DIC system was able to find the first 5 fundamental frequencies of the structure as well as the attendant mode shapes. The calculated fundamental frequencies had a peak difference of 0.08Hz or 1.8% of the results given by a reference LDV, although most of the DIC frequencies were within 1% or less of the LDV frequencies. The first 2 DIC frequencies were within 6.5% of the frequencies given by a finite element model, although the model was not calibrated. The DIC was also able to successfully determine the first 5 mode shapes of the structure. The first two mode shapes corresponded exactly with those given by the FEA model, while the fourth and fifth DIC mode shapes were similar to mode shapes given by the FEA model. The DIC system was unable to pick up one mode shape shown by the FEA model, although it is suspected that that mode shape was slightly above the Shannon-Nyquist frequency for the tests conducted.

Stereoscopic DIC is a very valuable tool in conducting operational modal analysis. Not only is it able to determine the fundamental frequencies of the structure being tested, but it is also able to determine the mode shapes for comparison to an FEA model or for further interpretation by the analyst. The results were found to be within 1% of the reference LDV in all but one case, demonstrating that it has comparable accuracy to an LDV for modal analysis while providing additional information in the form of mode shapes. Considering the comparable setup time of both systems, similar accuracy, and additional mode shape information provided by the DIC system, it is strongly recommended that DIC be used for further modal analysis.

6.3 Future Work

6.3.1 Applications

DIC has widely been used in simulated and laboratory situations, but has only recently been gaining more widespread use in field investigations. More diverse weather conditions, lighting conditions, and structure types need to be monitored by DIC. Weather conditions can play a large role, resulting in significant errors due to precipitation obscuring the target, harsh shadows developing during testing, and more. Lighting conditions in a laboratory setup are often easily implemented and highly idealized, while field conditions may have variable lighting during the course of a test or between tests. This means that field operators must be more aware of conditions, and as such, be able to adjust settings more rapidly. Civil field applications to date have largely focused on bridge structures, but future works should also investigate structures such as walls, slabs, cranes, or even full buildings under various loads. The full field nature of DIC would be very beneficial in tracking displacements or strains through walls and slabs in particular, both in field and laboratory testing. It may also be beneficial in the testing of larger structures that would otherwise be limited to only a few strain measurement points. Additional work on bridges would also be beneficial, including both dynamic and static testing of multiple points in a single structure using a single pair of cameras and natural targets.

6.3.2 DIC System Improvements

A variety of recommendations to improve the DIC system are presented here. These recommendations include functionality improvements, performance improvements, and accuracy improvements.

Functionality Improvements

The first functionality improvement is the addition of a strain module, which has been successfully implemented in other DIC systems but was not investigated in this system. One common use of DIC is to calculate the strains in specimens, which can take many different forms. An in-depth investigation into various strain formulations from a displacement field should be conducted and used to create a strain sub-module that can be accessed directly from the software GUI.

Minor options that would improve the user experience should also be implemented, such as an option to undistort images if necessary based on the calibration parameters. This option avoids users having to undistort images beforehand and would save on storage space at the expense of additional computation requirement.

The last functionality improvement is primarily tied to performance improvements. Real time DIC, even at reduced framerates, reduced resolution, or for limited targets, would allow for better field user experience. The operators would be better able to determine optimal parameters prior to full structure testing. This would likely require simplified calculations (such as a 0^{th} order shape function), cameras that can be directly read during acquisition, and implementation of a highly parallelized or GPU based algorithm.

Performance Improvements

While MATLAB is a convenient programming language to use for research and fast prototyping, it is not as efficient as properly optimized C or C++ programs. Efficiency improvements would almost definitely result from the proper use of C++ code for DIC analysis, and is recommended for future iterations of any DIC packages developed. C++ additionally can make use of OpenCV, which is a large, open-source library with many similar functions to MATLAB's image processing toolbox and camera calibrators.

Parallel processing would also be beneficial in DIC for multiple purposes. The primary bottleneck in DIC occurs during image correlation due to the interpolation of the target subsets. This operation can likely be further optimized by calculating all of the interpolated points in the subset simultaneously using parallel processing rather than determining them sequentially in a loop. Performance could be improved significantly through the use of general purpose graphics processing unit (GPGPU) to parallelize computations, especially computations involving FFTs [47, 69]. The use of GPGPUs and the relevant FFT functions would allow for the use of parallel seed matches, potentially allowing every subset in the first image to make use of an independent seed point to improve calculations, rather than relying on the neighbouring subsets and possibly propagating error. The excellent performance of GPGPUs when calculating FFTs may even allow for the potential use of Fourier domain based interpolation, which is generally seen as a bias-free form of interpolation. This would eliminate or significantly reduce the bias errors present in DIC analysis. The use of GPGPUs could be implemented through the use of OpenCL or CUDA (if an NVIDIA graphics card will be known to be used). A version of the correlation score guided analysis order could also be used with multi-core computer processing units (CPUs) using OpenMP to allow for simultaneous computation of as many valid calculation points as possible simultaneously. An explicit version of the 2^{nd} order warp function inverse may be derived rather than inverting the updated warp function on each iteration. This would remove the 6x6 matrix inversion required in the warp function update. If this inversion can be turned into an explicit solution, then the only inversion required in analysis would be the initial inversion of the reference subset Hessian, which would significantly improvement computation time.

Accuracy Improvements

Noise was found to play a very large role in the accuracy of 3D DIC, and so reducing the effects of noise is a key area of improvement. This could be achieved through the use of various noise filters, such as median filters or geometric mean filters. These filters should be investigated for their effects on both 2D and 3D DIC, since Gaussian filters appear to have been the primary filter investigated to date.

The gradient operator used by the algorithm has been shown to have an effect on the accuarcy of a DIC system by previous authors [50]. Various gradient operators such as Prewitt, Sobel, Roberts, and Central Difference should be investigated to determine their effects on both 2D and 3D DIC.

References

[1] *ABAQUS CAE Version 6.13-4*. Dassault Systèmes, Vélizy-Villacoublay, France, 2013.

[2] *MATLAB Release 2017a*. The MathWorks Inc., Natick, Massachusetts, USA, 2017.

[3] S. Allaoui, A. Rekik, A. Gasser, E. Blond, and K. Andreev. Digital Image Correlation measurements of mortarless joint closure in refractory masonries. *Construction and Building Materials*, 162:334–344, 2018.

[4] R. Bai, H. Jiang, Z. Lei, and W. Li. A novel 2nd-order shape function based digital image correlation method for large deformation measurements. *Optics and Lasers in Engineering*, 90:48–58, 2017.

[5] R. Balcaen, P. L. Reu, P. Lava, and D. Debruyne. Stereo-DIC Uncertainty Quantification based on Simulated Images. *Experimental Mechanics*, 57(6), 2017.

[6] J. Baqersad, P. Poozesh, C. Niezrecki, and P. Avitabile. Photogrammetry and optical methods in structural dynamics A review. *Mechanical Systems and Signal Processing*, 86:17–34, 2017.

[7] F. Barros, P. J. Sousa, P. J. Tavares, and P. M. G. P. Moreira. A DFT-based method for 3D digital image correlation. *Procedia Structural Integrity*, 5:1260–1266, 2017.

[8] T. Becker, K. Splitthof, T. Siebert, and P. Kletting. Error estimations of 3D digital image correlation measurements. In *SPIE 6341, Speckle 06*, Nimes, FR, 2006.

[9] J. Blaber, B. Adair, and A. Antoniou. Ncorr: Open-Source 2D Digital Image Correlation Matlab Software. *Experimental Mechanics*, 55(6):1105–1122, 2015.

[10] H. A. Bruck, S. R. McNeill, M. A. Sutton, and W. H. Peters. Digital image correlation using Newton-Raphson method of partial differential correction. *Experimental Mechanics*, 29(3):261–267, 1989.

[11] G. Busca, A. Cigada, P. Mazzoleni, and E. Zappa. Vibration Monitoring of Multiple Bridge Points by Means of a Unique Vision-Based Measuring System. *Experimental Mechanics*, 54(2):255–271, 2014.

[12] P. Cheng, M. A. Sutton, H. W. Schreier, and S. R. McNeill. Full-field speckle pattern image correlation with B-spline deformation function. *Experimental Mechanics*, 42(3):344–352, 2002.

[13] T. C. Chu, W. F. Ranson, and M. A. Sutton. Applications of digital-image-correlation techniques to experimental mechanics. *Experimental Mechanics*, 25(3):232–244, 1985.

[14] Y. Gao, T. Cheng, Y. Su, X. Xu, Y. Zhang, and Q. Zhang. High-efficiency and high-accuracy digital image correlation for three-dimensional measurement. *Optics and Lasers in Engineering*, 65:73–80, 2015.

[15] M. Guizar-Sicairos, S. T. Thurman, and J. R. Fienup. Efficient subpixel image registration algorithms. *Optics Letters*, 33(2):156, 2008.

[16] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, second edition, 2003.

[17] J. Heikkila and O. Silven. A four-step camera calibration procedure with implicit image correction. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997.

[18] Mark N. Helfrick, Christopher Niezrecki, Peter Avitabile, and Timothy Schmidt. 3D digital image correlation methods for full-field vibration measurement. *Mechanical Systems and Signal Processing*, 25(3):917–927, 2011.

[19] A. J. Hoag. *Monitoring of Dynamic Bridge Behaviour using Digital Image Correlation*. PhD thesis, Queen's University, 2016.

[20] Z. Hu, H. Xie, J. Lu, T. Hua, and J. Zhu. Study of the performance of different subpixel image correlation methods in 3D digital image correlation. *Applied Optics*, 49(21):4044, 2010.

[21] J. Javh, J. Slavič, and M. Boltežar. High frequency modal identification on noisy high-speed camera data. *Mechanical Systems and Signal Processing*, 98:344–351, 2018.

[22] X.D. Ke, H. W. Schreier, M. A. Sutton, and Y. Q. Wang. Error Assessment in Stereo-based Deformation Measurements: Part II. *Experimental Mechanics*, 51(4):423–441, 2011.

[23] J. P. Lewis. Fast Normalized Cross-Correlation. *Vision Interface*, pages 1–7, 2001.

[24] H. Lu and P. D. Cary. Deformation measurements by digital image correlation: Implementation of a second-order displacement gradient. *Experimental Mechanics*, 40(4):393–400, 2000.

[25] P. F. Luo, Y. J. Chao, M. A. Sutton, and W. H. Peters. Accurate measurement of three-dimensional deformations in deformable and rigid bodies using computer vision. *Experimental Mechanics*, 33(2):123–132, 1993.

[26] L. Luu, Z. Wang, M. Vo, T. Hoang, and J. Ma. Accuracy enhancement of digital image correlation with B-spline interpolation. *Optics Letters*, 36(16):3070, 2011.

[27] H. Nyquist. Certain topics in telegraph transmission theory. *Transactions of the American Institute of Electrical Engineers*, 47(2):617–644, 1928.

[28] B. Pan. Reliability-guided digital image correlation for image deformation measurement. *Applied Optics*, 48(8):1535–1542, 2009.

[29] B. Pan. Recent Progress in Digital Image Correlation. *Experimental Mechanics*, 51:1223–1235, 2011.

[30] B. Pan. Bias error reduction of digital image correlation using Gaussian pre-filtering. *Optics and Lasers in Engineering*, 51:1161–1167, 2013.

[31] B. Pan and K. Li. A fast digital image correlation method for deformation measurement. *Optics and Lasers in Engineering*, 49(7):841–847, 2011.

[32] 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.

[33] B. Pan, Z. Lu, and H. Xie. Mean intensity gradient: An effective global parameter for quality assessment of the speckle patterns used in digital image correlation. *Optics and Lasers in Engineering*, 48(4):469–477, 2010.

[34] B. Pan, K. Qian, H. Xie, and A. Asundi. Two-dimensional digital image correlation for in-plane displacement and strain measurement: A review. *Measurement Science and Technology*, 20, 2009.

[35] B. Pan and B. Wang. Digital Image Correlation with Enhanced Accuracy and Efficiency: A Comparison of Two Subpixel Registration Algorithms. *Experimental Mechanics*, 56:1395–1409, 2016.

[36] B. Pan, H. Xie, and Z. Wang. Equivalence of digital image correlation criteria for pattern matching. *Applied Optics*, 49(28):5501–5509, 2010.

[37] B. Pan, H. Xie, Z. Wang, K. Qian, and Z. Wang. Study on subset size selection in digital image correlation for speckle patterns. *Optics Express*, 16(10):7037, 2008.

[38] B. Pan, H. M. Xie, B. Q. Xu, and F. L. Dai. Performance of sub-pixel registration algorithms in digital image correlation. *Measurement Science and Technology*, 17(6):1615–1621, 2006.

[39] W. H. Peters and W. F. Ranson. Digital imaging techniques in experimental stress analysis. *Optical Engineering*, 21(3):427–431, 1982.

[40] C. Rainieri and G. Fabbrocino. *Experimental modal analysis of civil engineering structures*. Springer Science+Business Media, New York, NY, USA, 2014.

[41] P. L. Reu. Experimental and numerical methods for exact subpixel shifting. *Experimental Mechanics*, 51(4):443–452, 2011.

[42] P. L. Reu, D. P. Rohe, and L. D. Jacobs. Comparison of DIC and LDV for practical vibration and modal measurements. *Mechanical Systems and Signal Processing*, 86:2–16, 2017.

[43] D. Ribeiro, R. Calçada, J. Ferreira, and T. Martins. Non-contact measurement of the dynamic displacement of railway bridges using an advanced video-based system. *Engineering Structures*, 75:164–180, 2014.

[44] H. W. Schreier, J. R. Braasch, and M. A. Sutton. Systematic Errors in Digital Image Correlation Caused By Intensity Interpolation. *Optical Engineering*, 39(11):2915–2921, 2000.

[45] H. W. Schreier and M. A. Sutton. Systemic Errors in Digital Image Correlation Due to Undermatched Shape Functions. *Experimental Mechanics*, 42(3):303–310, 2002.

[46] C. E. Shannon. Communication in the Presence of Noise (Classic Paper). *Proceedings of the IRE*, 37(1):10–21, 1949.

[47] X. Shao, X. Dai, and X. He. Noise robustness and parallel computation of the inverse compositional Gauss-Newton algorithm in digital image correlation. *Optics and Lasers in Engineering*, 71:9–19, 2015.

[48] G. A. Stephen, J. M. W. Brownjohn, and C. A. Taylor. Measurements of static and dynamic displacement from visual monitoring of the Humber Bridge. *Engineering Structures*, 15(3):197–208, 1993.

[49] Y. Su, Q. Zhang, Z. Gao, X. Xu, and X. Wu. Fourier-based interpolation bias prediction in digital image correlation. *Optics Express*, 23(15), 2015.

[50] Y. Su, Q. Zhang, X. Xu, Z. Gao, and S. Wu. Interpolation bias for the inverse compositional GaussNewton algorithm in digital image correlation. *Optics and Lasers in Engineering*, 100(March 2017):267–278, 2018.

[51] M. A. Sutton, S. R. McNeill, J. Jang, and M. Babai. Effects of subpixel image restoration on digital correlation error estimates. *Optical Engineering*, 27(10):870–877, 1988.

[52] M. A. Sutton, J. L. Turner, H. A. Bruck, and T. A. Chae. Full Field Representation of the Discretely Sampled Surface Deformations for Displacement and Strain Analysis. *Society of Experimental Mechanics Journal*, 31(June):168–177, 1991.

[53] M. A. Sutton, W. J. Wolters, W. H. Peters, W. F. Ranson, and S. R. McNeill. Determination of displacements using an improved digital correlation method. *Image and Vision Computing*, 1(3):133–139, 1983.

[54] P. Thévenaz, T. Blu, and M. Unser. Image interpolation and resampling. *Handbook of Medical Imaging*, pages 1–39, 2000.

[55] F. Trebua and M. Hagara. Experimental modal analysis performed by high-speed digital image correlation system. *Measurement: Journal of the International Measurement Confederation*, 50(1):78–85, 2014.

[56] J. W. Tukey. *An introduction to the calculations of numerical spectrum analysis.* Wiley, New York, NY, USA, 1967.

[57] M. Unser, A. Aldroubi, and M. Eden. B-spline signal processing: Part I - Theory. *IEEE Transactions on Signal Processing*, 41(2):821–833, 1993.

[58] M. Unser, P. Thévenaz, and L. Yaroslavsky. Convolution-Based Interpolation for Fast, High-Quality Rotation of Images. *IEEE Transactions on Image Processing*, 4(10):1371–1381, 1995.

[59] R. E. Walpole, R. H. Myers, S. L. Myers, and K. Ye. *Probability & Statistics for Engineers & Scientists*. Pearson, Boston, MA, USA, 9th edition, 2012.

[60] B. Wang and B. Pan. Random Errors in Digital Image Correlation Due to Matched or Overmatched Shape Functions. *Experimental Mechanics*, 55(9):1717–1727, 2015.

[61] Y. Q. Wang, M. A. Sutton, H. A. Bruck, and H. W. Schreier. Quantitative error assessment in pattern matching: Effects of intensity pattern noise, interpolation, strain and image contrast on motion measurements. *Strain*, 45(2):160–178, 2009.

[62] Y. Q. Wang, M. A. Sutton, X. D. Ke, H. W. Schreier, P. L. Reu, and T. J. Miller. Error Assessment in Stereo-based Deformation Measurements: Part I. *Experimental Mechanics*, 51(4):405–422, 2011.

[63] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):965980, 1992.

[64] S. Yaofeng and J. H. L. Pang. Study of optimal subset size in digital image correlation of speckle pattern images. *Optics and Lasers in Engineering*, 45(9):967–974, 2007.

[65] S. Yoneyama and A. Kitagawa. Bridge deflection measurement using digital image correlation. *Experimental Techniques*, 31(1):34–40, 2007.

[66] L. Yu and B. Pan. The errors in digital image correlation due to overmatched shape functions. *Measurement Science and Technology*, 26(4), 2015.

[67] L. Yu and B. Pan. Single-camera high-speed stereo-digital image correlation for full-field vibration measurement. *Mechanical Systems and Signal Processing*, 94:374–383, 2017.

[68] E. Zappa, P. Mazzoleni, and A. Matinmanesh. Uncertainty assessment of digital image correlation method in dynamic applications. *Optics and Lasers in Engineering*, 56:140–151, 2014.

[69] L. Zhang, T. Wang, Z. Jiang, Q. Kemao, Y. Liu, Z. Liu, L. Tang, and Dong. S. High accuracy digital image correlation powered by GPU-based parallel computing. *Optics and Lasers in Engineering*, 69(June):7–12, 2015.

[70] Z. F. Zhang, Y. L. Kang, H. W. Wang, Q. H. Qin, Y. Qiu, and X. Q. Li. A novel coarse-fine search scheme for digital image correlation method. *Measurement: Journal of the International Measurement Confederation*, 39(8):710–718, 2006.

[71] P. Zhou and K. E. Goodson. Subpixel displacement and deformation gradient measurement using digital image/speckle correlation (DISC). *Optical Engineering*, 40(8):1613–1620, 2001.

APPENDICES

Appendix A

Derivations

A.1 Derivation of Warp Functions

A.1.1 0^{th} Order Shape Function

The 0^{th} order shape function generally takes the form of Equation A.1, which is the same as the matrix form given in Equation A.2.

$$\begin{aligned} x' &= x + u \\ y' &= y + v \end{aligned} \tag{A.1}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & u \\ 0 & 1 & v \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{A.2}$$

The IC-GN requires an invertible matrix that takes the same form before and after the update step, and so a square, invertible matrix of the form given in Equation A.3 is required.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & u \\ 0 & 1 & v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{A.3}$$

A.1.2 1st Order Shape Function

A similar process was conducted for the 1st order shape function, which typically takes the form shown in Equation A.4.

$$\begin{aligned} x' &= u_x x + x + u_y y + u \\ y' &= v_x x + v_y y + y + v \end{aligned} \tag{A.4}$$

This can be converted to the matrix form shown in Equation A.5.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 + u_x & u_y & u \\ v_x & 1 + v_y & v \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{A.5}$$

However, this form of the equation does not comprise an invertible matrix with the same form before and after the update step, and so is converted to the form shown in Equation A.6. The addition of the bottom row of 0s and 1 allows the matrix to take an invertible form that is the same pre- and post-update.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 + u_x & u_y & u \\ v_x & 1 + v_y & v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{A.6}$$

A.1.3 2nd Order Shape Function

The 2nd order shape function is significantly more involved than the 0th or 1st order shape function. It typically takes the form given by Equation A.7, which results in 12 unknowns (u, v, u_x, u_y, v_x, v_y, u_{xx}, u_{xy}, u_{yy}, v_{xx}, v_{xy}, and v_{yy}).

$$\begin{aligned} x' &= \frac{1}{2} u_{xx} x^2 + u_{xy} xy + \frac{1}{2} u_{yy} y^2 + u_x x + x + u_y y + u \\ y' &= \frac{1}{2} v_{xx} x^2 + v_{xy} xy + \frac{1}{2} v_{yy} y^2 + v_x x + v_y y + y + v \end{aligned} \tag{A.7}$$

The matrix form of this equation takes the form given by Equation A.8. This form is clearly not invertible in the same form as the original, and so must be modified to allow for this step.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \frac{1}{2}u_{xx} & u_{xy} & \frac{1}{2}u_{yy} & 1+u_x & u_y & u \\ \frac{1}{2}v_{xx} & v_{xy} & \frac{1}{2}v_{yy} & v_x & 1+v_y & v \end{bmatrix} \begin{bmatrix} x^2 \\ xy \\ y^2 \\ x \\ y \\ 1 \end{bmatrix} \tag{A.8}$$

In order to generate an invertible matrix of the same form, the matrix shown in Equation A.8 needs to be expanded, which can be done through the addition of the second order deformation terms to the left side and adding the related terms to the warp matrix. This means that the warp function will take the form of Equation A.9.

$$\begin{bmatrix} x'^2 \\ x'y' \\ y'^2 \\ x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} A_{00} & A_{01} & A_{02} & A_{03} & A_{04} & A_{05} \\ A_{10} & A_{11} & A_{12} & A_{13} & A_{14} & A_{15} \\ A_{20} & A_{21} & A_{22} & A_{23} & A_{24} & A_{25} \\ \frac{1}{2}u_{xx} & u_{xy} & \frac{1}{2}u_{yy} & 1+u_x & u_y & u \\ \frac{1}{2}v_{xx} & v_{xy} & \frac{1}{2}v_{yy} & v_x & 1+v_y & v \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^2 \\ xy \\ y^2 \\ x \\ y \\ 1 \end{bmatrix} \tag{A.9}$$

$$\begin{aligned} x'^2 &= \Big(\frac{1}{2}u_{xx}x^2 + u_{xy}xy + \frac{1}{2}u_{yy}y^2 + u_x x + x + u_y y + u\Big)^2 \\ &= \Big(\frac{1}{4}u_{xx}^2\Big)x^4 + \Big(u_{xx}u_{xy}\Big)x^3y + \Big(\frac{1}{2}u_{xx}u_{yy} + u_{xy}^2\Big)x^2y^2 + \Big(u_{xy}u_{yy}\Big)xy^3 \\ &\quad + \Big(\frac{1}{4}u_{yy}^2\Big)y^4 + \Big(u_{xx}(u_x+1)\Big)x^3 + \Big(u_y u_{xx} + 2u_{xy}(u_x+1)\Big)x^2y + \Big(2u_y u_{xy} \\ &\quad + u_{yy}(u_x+1)\Big)xy^2 + \Big(u_y u_{yy}\Big)y^3 + \Big(1 + uu_{xx} + u_x(u_x+2)\Big)x^2 + \Big(2u_y(u_x+1) \\ &\quad + 2uu_{xy}\Big)xy + \Big(u_y^2 + uu_{yy}\Big)y^2 + \Big(2u(u_x+1)\Big)x + \Big(2uu_y\Big)y + u^2 \end{aligned} \tag{A.10}$$

$$\begin{aligned}
y'^2 &= \Big(\frac{1}{2}v_{xx}x^2 + v_{xy}xy + \frac{1}{2}v_{yy}y^2 + v_x x + y + v_y y + v\Big)^2 \\
&= \Big(\frac{1}{4}v_{xx}^2\Big)x^4 + \Big(v_{xx}v_{xy}\Big)x^3y + \Big(\frac{1}{2}v_{xx}v_{yy} + v_{xy}^2\Big)x^2y^2 + \Big(v_{xy}v_{yy}\Big)xy^3 \\
&\quad + \Big(\frac{1}{4}v_{yy}^2\Big)y^4 + \Big(v_x v_{xx}\Big)x^3 + \Big(2v_x v_{xy} + v_{xx}(v_y+1)\Big)x^2y + \Big(v_x v_{yy} \\
&\quad + 2v_{xy}(v_y+1)\Big)xy^2 + \Big(v_{yy}(v_y+1)\Big)y^3 + \Big(v_x^2 + vv_{xx}\Big)x^2 + \Big(2v_x(v_y+1) \\
&\quad + 2vv_{xy}\Big)xy + \Big(1 + vv_{yy} + v_y(v_y+2)\Big)y^2 + \Big(2vv_x\Big)x + \Big(2v(v_y+1)\Big)y + v^2
\end{aligned} \tag{A.11}$$

$$\begin{aligned}
x'y' &= \Big(\frac{1}{2}u_{xx}x^2 + u_{xy}xy + \frac{1}{2}u_{yy}y^2 + u_x x + x + u_y y + u\Big) \\
&\quad \times \Big(\frac{1}{2}v_{xx}x^2 + v_{xy}xy + \frac{1}{2}v_{yy}y^2 + v_x x + y + v_y y + v\Big) \\
&= \frac{1}{4}\Big(u_{xx}v_{xx}\Big)x^4 + \frac{1}{2}\Big(u_{xx}v_{xy} + u_{xy}v_{xx}\Big)x^3y + \frac{1}{4}\Big((u_{xx}v_{yy} + u_{yy}v_{xx}) + 4u_{xy}v_{xy}\Big)x^2y^2 \\
&\quad + \frac{1}{2}\Big(u_{xy}v_{yy} + u_{yy}v_{xy}\Big)xy^3 + \frac{1}{4}\Big(u_{yy}v_{yy}\Big)y^4 + \frac{1}{2}\Big(u_{xx}v_x + v_{xx}(u_x+1)\Big)x^3 \\
&\quad + \Big(\frac{1}{2}u_{xx}(v_y+1) + u_{xy}v_x + \frac{1}{2}u_y v_{xx} + v_{xy}(u_x+1)\Big)x^2y + \Big(\frac{1}{2}v_{yy}(u_x+1) + u_y v_{xy} \\
&\quad + \frac{1}{2}u_{yy}v_x + u_{xy}(v_y+1)\Big)xy^2 + \frac{1}{2}\Big(u_y v_{yy} + u_{yy}(v_y+1)\Big)y^3 + \Big(\frac{1}{2}(uv_{xx} + u_{xx}v) \\
&\quad + v_x(u_x+1)\Big)x^2 + \Big(1 + u_x + v_y + u_x v_y + u_y v_x + uv_{xy} + u_{xy}v\Big)xy \\
&\quad + \Big(\frac{1}{2}(uv_{yy} + u_{yy}v) + u_y(v_y+1)\Big)y^2 + \Big(v + uv_x + u_x v\Big)x + \Big(u + uv_y + u_y v\Big)y + uv
\end{aligned} \tag{A.12}$$

This is clearly quite complex and difficult to implement, so the higher order terms of degree greater than 2 with respect to x and y are ignored. Ignoring these terms (i.e., x^4, x^3y, x^2y^2, xy^3, y^4, x^3, x^2y, xy^2, and y^3), Equations A.10 to A.12 reduce to

$$\begin{aligned}
x'^2 = \Big(1 + uu_{xx} + u_x(u_x+2)\Big)x^2 + \Big(2u_y(u_x+1) + 2uu_{xy}\Big)xy \\
+ \Big(u_y^2 + uu_{yy}\Big)y^2 + \Big(2u(u_x+1)\Big)x + \Big(2uu_y\Big)y + u^2
\end{aligned} \tag{A.13}$$

$$y'^2 = \Big(v_x^2 + vv_{xx}\Big)x^2 + \Big(2v_x(v_y+1) + 2vv_{xy}\Big)xy + \Big(1 + vv_{yy} + v_y(v_y+2)\Big)y^2 + \Big(2vv_x\Big)x + \Big(2v(v_y+1)\Big)y + v^2 \tag{A.14}$$

$$x'y' = \Big(\frac{1}{2}(uv_{xx} + u_{xx}v) + v_x(u_x+1)\Big)x^2 + \Big(1 + u_x + v_y + u_xv_y + u_yv_x + uv_{xy} + u_{xy}v\Big)xy + \Big(\frac{1}{2}(uv_{yy} + u_{yy}v) + u_y(v_y+1)\Big)y^2 + \Big(v + uv_x + u_xv\Big)x + \Big(u + uv_y + u_yv\Big)y + uv \tag{A.15}$$

This can be summarized in a matrix warp form according to Equations A.16 and A.17.

$$\begin{bmatrix} x'^2 \\ x'y' \\ y'^2 \\ x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} A_{00} & A_{01} & A_{02} & 2u(u_x+1) & 2uu_y & u^2 \\ A_{10} & A_{11} & A_{12} & u_xv + uv_x + v & u_yv + uv_y + u & uv \\ A_{20} & A_{21} & A_{22} & 2vv_x & 2v(v_y+1) & v^2 \\ \frac{1}{2}u_{xx} & u_{xy} & \frac{1}{2}u_{yy} & 1+u_x & u_y & u \\ \frac{1}{2}v_{xx} & v_{xy} & \frac{1}{2}v_{yy} & v_x & 1+v_y & v \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^2 \\ xy \\ y^2 \\ x \\ y \\ 1 \end{bmatrix} \tag{A.16}$$

where A_{00} to A_{22} can be found in Equation A.17.

$$\begin{gathered} A_{00} = 1 + uu_{xx} + u_x(u_x+2) \\ A_{01} = 2u_y(u_x+1) + 2uu_{xy} \\ A_{02} = u_y^2 + uu_{yy} \\ A_{10} = \frac{1}{2}(uv_{xx} + u_{xx}v) + v_x(u_x+1) \\ A_{11} = 1 + u_x + v_y + u_xv_y + u_yv_x + uv_{xy} + u_{xy}v \\ A_{12} = \frac{1}{2}(uv_{yy} + u_{yy}v) + u_y(v_y+1) \\ A_{20} = v_x^2 + vv_{xx} \\ A_{21} = 2v_x(v_y+1) + 2vv_{xy} \\ A_{22} = 1 + vv_{yy} + v_y(v_y+2) \end{gathered} \tag{A.17}$$

A.2 Derivations of Explicit Warp Inversion

The IC-GN algorith makes use of an update step in the warp function as shown in Equation A.18:

$$W(\xi;p)_n = W(\xi;p)_{n-1} \circ W^{-1}(\xi;\Delta p) \tag{A.18}$$

This update step varies depending on the shape function chosen, and the warp update for each shape function is shown in Equation A.19a to A.19b:

$$\begin{bmatrix} 1 & 0 & u \\ 0 & 1 & v \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & u \\ 0 & 1 & v \\ 0 & 0 & 1 \end{bmatrix} \circ \begin{bmatrix} 1 & 0 & \Delta u \\ 0 & 1 & \Delta v \\ 0 & 0 & 1 \end{bmatrix}^{-1} \tag{A.19a}$$

$$\begin{bmatrix} 1+u_x & u_y & u \\ v_x & 1+v_y & v \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1+u_x & u_y & u \\ v_x & 1+v_y & v \\ 0 & 0 & 1 \end{bmatrix} \circ \begin{bmatrix} 1+\Delta u_x & \Delta u_y & \Delta u \\ \Delta v_x & 1+\Delta v_y & \Delta v \\ 0 & 0 & 1 \end{bmatrix}^{-1} \tag{A.19b}$$

The explicit forms of these equations can be found by inverting the matrix on the right side to find the updated warp parameters using explicit expressions for speed. The 2nd order shape function makes use of a 6x6 warp matrix, which is very difficult to manually invert. The 2nd order shape function was therefore not explicitly inverted for computation.

A.2.1 0th Order Shape Function

The 0th order shape function makes use of a 3x3 warp matrix in the IC-GN to capture the rigid body translation. The inverse of a 3x3 matrix, M, can be found by calculating the matrix of cofactors, transposing the matrix of cofactors to find the adjugate matrix, followed by dividing by the determinant of the original matrix to find the inverse. In this case, the matrix M is:

$$M = \begin{bmatrix} 1 & 0 & \Delta u \\ 0 & 1 & \Delta v \\ 0 & 0 & 1 \end{bmatrix} \tag{A.20}$$

The matrix of cofactors for the 0th order shape function can be found through taking the determinant of each 2x2 minor matrix and multiplying by $+$ or $-$ depending on location, resulting in:

$$\begin{aligned} M_{cofactor} &= \begin{bmatrix} \begin{vmatrix} 1 & \Delta v \\ 0 & 1 \end{vmatrix} & -\begin{vmatrix} 0 & \Delta v \\ 0 & 1 \end{vmatrix} & \begin{vmatrix} 0 & 1 \\ 0 & 0 \end{vmatrix} \\ -\begin{vmatrix} 0 & \Delta u \\ 0 & 1 \end{vmatrix} & \begin{vmatrix} 1 & \Delta u \\ 0 & 1 \end{vmatrix} & -\begin{vmatrix} 1 & 0 \\ 0 & 0 \end{vmatrix} \\ \begin{vmatrix} 0 & \Delta u \\ 1 & \Delta v \end{vmatrix} & -\begin{vmatrix} 1 & \Delta u \\ 0 & \Delta v \end{vmatrix} & \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\Delta u & -\Delta v & 1 \end{bmatrix} \end{aligned} \tag{A.21}$$

The adjugate matrix can then be found by transposing the cofactor matrix:

$$\begin{aligned} M_{adjugate} &= M_{cofactor}^T \\ &= \begin{bmatrix} 1 & 0 & -\Delta u \\ 0 & 1 & -\Delta v \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{A.22}$$

Finally, the matrix inverse can be found by dividing the adjugate matrix by the determinant of the original matrix. The determinant of the matrix, M, is:

$$\begin{aligned} det(M) &= \begin{vmatrix} 1 & 0 & \Delta u \\ 0 & 1 & \Delta v \\ 0 & 0 & 1 \end{vmatrix} \\ &= 1 \begin{vmatrix} 1 & \Delta v \\ 0 & 1 \end{vmatrix} - 0 \begin{vmatrix} 0 & \Delta v \\ 0 & 1 \end{vmatrix} + u \begin{vmatrix} 0 & 1 \\ 0 & 0 \end{vmatrix} \\ &= 1 \end{aligned} \tag{A.23}$$

The matrix inverse of the 0^{th} order shape function is then:

$$\begin{aligned} W^{-1} &= \frac{1}{det(M)} M_{adjugate} \\ &= \begin{bmatrix} 1 & 0 & -\Delta u \\ 0 & 1 & -\Delta v \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{A.24}$$

A.2.2 1^{st} Order Shape Function

The 1^{st} order shape function makes use of a 3x3 warp matrix in the IC-GN to capture the shearing and elongation/compression deformations. Similarly to the 0^{th} order shape function, the inverse of a 3x3 matrix, M, can be found by calculating the matrix of cofactors, transposing the matrix of cofactors to find the adjugate matrix, followed by dividing by the determinant of the original matrix to find the inverse. In this case, the matrix M is:

$$M = \begin{bmatrix} 1+\Delta u_x & \Delta u_y & \Delta u \\ \Delta v_x & 1+\Delta v_y & \Delta v \\ 0 & 0 & 1 \end{bmatrix} \tag{A.25}$$

The matrix of cofactors for the 1^{st} order shape function can be found through taking the determinant of each 2x2 minor matrix and multiplying by + or − depending on location, resulting in:

$$\begin{aligned} M_{cofactor} &= \begin{bmatrix} \begin{vmatrix} 1+\Delta v_y & \Delta v \\ 0 & 1 \end{vmatrix} & -\begin{vmatrix} \Delta v_x & \Delta v \\ 0 & 1 \end{vmatrix} & \begin{vmatrix} \Delta v_x & 1+\Delta v_y \\ 0 & 0 \end{vmatrix} \\ -\begin{vmatrix} \Delta u_y & \Delta u \\ 0 & 1 \end{vmatrix} & \begin{vmatrix} 1+\Delta u_x & \Delta u \\ 0 & 1 \end{vmatrix} & -\begin{vmatrix} 1+\Delta u_x & \Delta u_y \\ 0 & 0 \end{vmatrix} \\ \begin{vmatrix} \Delta u_y & \Delta u \\ 1+\Delta v_y & \Delta v \end{vmatrix} & -\begin{vmatrix} 1+\Delta u_x & \Delta u \\ \Delta v_x & \Delta v \end{vmatrix} & \begin{vmatrix} 1+\Delta u_x & \Delta u_y \\ \Delta v_x & 1+\Delta v_y \end{vmatrix} \end{bmatrix} \\ &= \begin{bmatrix} 1+\Delta v_y & -\Delta v_x & 0 \\ -\Delta u_y & 1+\Delta u_x & 0 \\ \Delta u_y \Delta v - \Delta u(1+\Delta v_y) & \Delta u \Delta v_x - \Delta v(1+\Delta u_x) & (1+\Delta u_x)(1+\Delta v_y) - \Delta u_y \Delta v_x \end{bmatrix} \end{aligned} \tag{A.26}$$

The adjugate matrix can then be found by transposing the cofactor matrix:

$$\begin{aligned} M_{adjugate} &= M^T_{cofactor} \\ &= \begin{bmatrix} 1+\Delta v_y & -\Delta u_y & \Delta u_y \Delta v - \Delta u(1+\Delta v_y) \\ -\Delta v_x & 1+\Delta u_x & \Delta u \Delta v_x - \Delta v(1+\Delta u_x) \\ 0 & 0 & (1+\Delta u_x)(1+\Delta v_y) - \Delta u_y \Delta v_x \end{bmatrix} \end{aligned} \tag{A.27}$$

Finally, the matrix inverse can be found by dividing the adjugate matrix by the determinant of the original matrix. The determinant of the matrix, M, is:

$$\begin{aligned} det(M) &= \begin{vmatrix} 1+\Delta u_x & \Delta u_y & \Delta u \\ \Delta v_x & 1+\Delta v_y & \Delta v \\ 0 & 0 & 1 \end{vmatrix} \\ &= (1+\Delta u_x)\begin{vmatrix} 1+\Delta v_y & \Delta v \\ 0 & 1 \end{vmatrix} - \Delta u_y \begin{vmatrix} \Delta v_x & \Delta v \\ 0 & 1 \end{vmatrix} + \Delta u \begin{vmatrix} \Delta v_x & 1+\Delta v_y \\ 0 & 0 \end{vmatrix} \\ &= (1+\Delta u_x)(1+\Delta v_y) - \Delta u_y \Delta v_x \end{aligned} \tag{A.28}$$

The matrix inverse of the 1st order shape function is then:

$$\begin{aligned} W^{-1} &= \frac{1}{det(M)} M_{adjugate} \\ &= \frac{1}{(1+\Delta u_x)(1+\Delta v_y) - \Delta u_y \Delta v_x} \begin{bmatrix} 1+\Delta v_y & -\Delta u_y & \Delta u_y \Delta v - \Delta u(1+\Delta v_y) \\ -\Delta v_x & 1+\Delta u_x & \Delta u \Delta v_x - \Delta v(1+\Delta u_x) \\ 0 & 0 & (1+\Delta u_x)(1+\Delta v_y) - \Delta u_y \Delta v_x \end{bmatrix} \end{aligned} \tag{A.29}$$

A.3 Derivations of B-Spline Kernels

The general B-spline equation is shown in Equation A.30 [58]. The resampling matrices can be found by adding a shift of Δx to the B-splines and finding the resultant matrix in terms of the powers of Δx.

$$\begin{aligned} \beta^0(x) &= \begin{cases} 1 & \text{for} |x| < \frac{1}{2} \\ \frac{1}{2} & \text{for} |x| = \frac{1}{2} \\ 0 & \text{for} |x| > \frac{1}{2} \end{cases} \\ \beta^n(x) &= \sum_{k=0}^{n+1} \frac{(-1)^k (n+1)}{(n+1-k)!\,k!} (\frac{n+1}{2} + x - k)_+^n \\ &= \frac{1}{n!} \sum_{k=0}^{n+1} \binom{n+1}{k} (-1)^k (x - k + \frac{n+1}{2})_+^n \end{aligned} \tag{A.30}$$

A.3.1 Bilinear B-Spline

A bilinear B-spline takes the form of a single linear B-spline in each direction, which can be found by setting $n = 1$ in the general equation. The linear B-spline is shown in Equation

A.31.

$$\beta^1 = \begin{cases} x+1 & -1 \leq x < 0 \\ -x+1 & 0 \leq x < 1 \end{cases} \tag{A.31}$$

A shift of Δx to the kernel gives Equation A.32. Sampling at each of the valid integer values of x to find the new shifted interpolation kernel gives Equation A.33

$$\beta^1(x-\Delta x) = \begin{cases} x-\Delta x+1 & -1 \leq x < 0 \\ -x+\Delta x+1 & 0 \leq x < 1 \end{cases} \tag{A.32}$$

$$\beta^1(\Delta x) = \begin{cases} -\Delta x+1 & x = 0 \\ \Delta x & x = 1 \end{cases} \tag{A.33}$$

This can then be summarized in the matrix form of Equation A.34.

$$\beta^1(\Delta x) = \begin{bmatrix} 1 & \Delta x \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix} \tag{A.34}$$

A linear interpolation can then be applied in both directions to form bilinear interpolation. To perform the linear interpolation in the other direction, the same process may be followed for $\beta(y-\Delta y)$, which simply results in the same matrix. To interpolate in both directions, the resampling matrices can be used to pre- and post-multiply the sampling coefficient matrix, resulting in Equation A.35.

$$p(x,y) = \begin{bmatrix} 1 & \Delta x \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix} C \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix}^T \begin{bmatrix} 1 \\ \Delta y \end{bmatrix} \tag{A.35}$$

where C is the 2x2 sampling coefficient matrix, which can be found from deconvolution of the image and the B-spline kernel.

A.3.2 Bicubic B-Spline

A bicubic B-spline takes the form of a cubic B-spline in each direction, which can be found by setting $n = 3$ in the general equation. The cubic B-spline is shown in Equation A.36.

$$\beta^3 = \begin{cases} \frac{1}{6}x^3 + x^2 + 2x + \frac{4}{3} & -2 \leq x < -1 \\ -\frac{1}{2}x^3 - x^2 + \frac{2}{3} & -1 \leq x < 0 \\ \frac{1}{2}x^3 - x^2 + \frac{2}{3} & 0 \leq x < 1 \\ -\frac{1}{6}x^3 + x^2 - 2x + \frac{4}{3} & 1 \leq x < 2 \end{cases} \tag{A.36}$$

A shift of Δx to the kernel gives Equation A.37. Sampling at each of the valid integer values of x to find the new shifted interpolation kernel gives Equation A.38

$$\beta^3(x-\Delta x) = \begin{cases} \frac{1}{6}(x-\Delta x)^3 + (x-\Delta x)^2 + 2(x-\Delta x) + \frac{4}{3} & -2 \le x < -1 \\ -\frac{1}{2}(x-\Delta x)^3 - (x-\Delta x)^2 + \frac{2}{3} & -1 \le x < 0 \\ \frac{1}{2}(x-\Delta x)^3 - (x-\Delta x)^2 + \frac{2}{3} & 0 \le x < 1 \\ -\frac{1}{6}(x-\Delta x)^3 + (x-\Delta x)^2 - 2(x-\Delta x) + \frac{4}{3} & 1 \le x < 2 \end{cases} \tag{A.37}$$

$$\beta^3(\Delta x) = \begin{cases} -\frac{1}{6}\Delta x^3 + \frac{1}{2}\Delta x^2 - \frac{1}{2}\Delta x + \frac{1}{6} & x = -1 \\ \frac{1}{2}\Delta x^3 - \Delta x^2 + \frac{2}{3} & x = 0 \\ -\frac{1}{2}\Delta x^3 + \frac{1}{2}\Delta x^2 + \frac{1}{2}\Delta x + \frac{1}{6} & x = 1 \\ \frac{1}{6}\Delta x^3 & x = 2 \end{cases} \tag{A.38}$$

This can then be summarized in the matrix form of Equation A.39.

$$\beta^3(\Delta x) = \begin{bmatrix} 1 & \Delta x & \Delta x^2 & \Delta x^3 \end{bmatrix} \begin{bmatrix} \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & -1 & \frac{1}{2} & 0 \\ -\frac{1}{6} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{6} \end{bmatrix} \tag{A.39}$$

Cubic interpolation can be completed in each direction to perform a bicubic interpolation. The same process can be followed for the y direction to find $\beta^3(\Delta y)$, which results in the same matrix as for $\beta^3(\Delta x)$. To interpolate in both directions, the resampling matrices can be used to pre- and post-multiply the sampling coefficient matrix, resulting in Equation A.40.

$$p(x,y) = \begin{bmatrix} 1 & \Delta x & \Delta x^2 & \Delta x^3 \end{bmatrix} \begin{bmatrix} \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & -1 & \frac{1}{2} & 0 \\ -\frac{1}{6} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{6} \end{bmatrix} C \begin{bmatrix} \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & -1 & \frac{1}{2} & 0 \\ -\frac{1}{6} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{6} \end{bmatrix}^T \begin{bmatrix} 1 \\ \Delta y \\ \Delta y^2 \\ \Delta y^3 \end{bmatrix} \tag{A.40}$$

where C is the 4x4 sampling coefficient matrix, which can be found from the deconvolution of the image and the B-spline kernel.

A.3.3 Biquintic B-Spline

Biquintic B-splines can be found by taking a quintic B-spline in each direction. The single direction quintic B-spline can be found by setting $n = 5$ in the general B-spline equation.

The 1D quintic B-spline is shown in Equation A.41.

$$\beta^5 = \begin{cases} \frac{1}{120}x^5 + \frac{1}{8}x^4 + \frac{3}{4}x^3 + \frac{9}{4}x^2 + \frac{27}{8}x + \frac{81}{40} & -3 \leq x < -2 \\ -\frac{1}{24}x^5 - \frac{3}{8}x^4 - \frac{5}{4}x^3 - \frac{7}{4}x^2 - \frac{5}{8}x + \frac{17}{40} & -2 \leq x < -1 \\ \frac{1}{12}x^5 + \frac{1}{4}x^4 - \frac{1}{2}x^2 + \frac{11}{20} & -1 \leq x < 0 \\ -\frac{1}{12}x^5 + \frac{1}{4}x^4 - \frac{1}{2}x^2 + \frac{11}{20} & 0 \leq x < 1 \\ \frac{1}{24}x^5 - \frac{3}{8}x^4 + \frac{5}{4}x^3 - \frac{7}{4}x^2 + \frac{27}{8}x + \frac{17}{40} & 1 \leq x < 2 \\ -\frac{1}{120}x^5 + \frac{1}{8}x^4 - \frac{3}{4}x^3 + \frac{9}{4}x^2 - \frac{27}{8}x + \frac{81}{40} & 2 \leq x < 3 \end{cases} \tag{A.41}$$

A shift of Δx to the kernel gives Equation A.42. Sampling at each of the valid integer values of x to find the new shifted interpolation kernel gives Equation A.43

$$\beta^5(x - \Delta x) = \begin{cases} \frac{1}{120}(x-\Delta x)^5 + \frac{1}{8}(x-\Delta x)^4 + \frac{3}{4}(x-\Delta x)^3 + \frac{9}{4}(x-\Delta x)^2 + \frac{27}{8}(x-\Delta x) + \frac{81}{40} & -3 \leq x < -2 \\ -\frac{1}{24}(x-\Delta x)^5 - \frac{3}{8}(x-\Delta x)^4 - \frac{5}{4}(x-\Delta x)^3 - \frac{7}{4}(x-\Delta x)^2 - \frac{5}{8}(x-\Delta x) + \frac{17}{40} & -2 \leq x < -1 \\ \frac{1}{12}(x-\Delta x)^5 + \frac{1}{4}(x-\Delta x)^4 - \frac{1}{2}(x-\Delta x)^2 + \frac{11}{20} & -1 \leq x < 0 \\ -\frac{1}{12}(x-\Delta x)^5 + \frac{1}{4}(x-\Delta x)^4 - \frac{1}{2}(x-\Delta x)^2 + \frac{11}{20} & 0 \leq x < 1 \\ \frac{1}{24}(x-\Delta x)^5 - \frac{3}{8}(x-\Delta x)^4 + \frac{5}{4}(x-\Delta x)^3 - \frac{7}{4}(x-\Delta x)^2 + \frac{27}{8}(x-\Delta x) + \frac{17}{40} & 1 \leq x < 2 \\ -\frac{1}{120}(x-\Delta x)^5 + \frac{1}{8}(x-\Delta x)^4 - \frac{3}{4}(x-\Delta x)^3 + \frac{9}{4}(x-\Delta x)^2 - \frac{27}{8}(x-\Delta x) + \frac{81}{40} & 2 \leq x < 3 \end{cases} \tag{A.42}$$

$$\beta^5(\Delta x) = \begin{cases} -\frac{1}{120}\Delta x^5 + \frac{1}{24}\Delta x^4 - \frac{1}{12}\Delta x^3 + \frac{1}{12}\Delta x^2 - \frac{1}{24}\Delta x + \frac{1}{120} & x = -2 \\ \frac{1}{24}\Delta x^5 - \frac{1}{6}\Delta x^4 + \frac{1}{6}\Delta x^3 + \frac{1}{6}\Delta x^2 - \frac{5}{12}\Delta x + \frac{13}{60} & x = -1 \\ -\frac{1}{12}\Delta x^5 + \frac{1}{4}\Delta x^4 - \frac{1}{2}\Delta x^2 + \frac{11}{20} & x = 0 \\ \frac{1}{12}\Delta x^5 - \frac{1}{6}\Delta x^4 - \frac{1}{6}\Delta x^3 + \frac{1}{6}\Delta x^2 + \frac{5}{12}\Delta x + \frac{13}{60} & x = 1 \\ -\frac{1}{24}\Delta x^5 + \frac{1}{24}\Delta x^4 + \frac{1}{12}\Delta x^3 + \frac{1}{12}\Delta x^2 + \frac{1}{24}\Delta x + \frac{1}{120} & x = 2 \\ \frac{1}{120}\Delta x^5 & x = 3 \end{cases} \tag{A.43}$$

This can then be summarized in the matrix form of Equation A.44.

$$\beta^5(\Delta x) = \begin{bmatrix} 1 & \Delta x & \Delta x^2 & \Delta x^3 & \Delta x^4 & \Delta x^5 \end{bmatrix} \begin{bmatrix} \frac{1}{120} & \frac{13}{60} & \frac{11}{20} & \frac{13}{60} & \frac{1}{120} & 0 \\ -\frac{1}{24} & -\frac{5}{12} & 0 & \frac{5}{12} & \frac{1}{24} & 0 \\ \frac{1}{12} & \frac{1}{6} & -\frac{1}{2} & \frac{1}{6} & \frac{1}{12} & 0 \\ -\frac{1}{12} & \frac{1}{6} & 0 & -\frac{1}{6} & \frac{1}{12} & 0 \\ \frac{1}{24} & -\frac{1}{6} & \frac{1}{4} & -\frac{1}{6} & \frac{1}{24} & 0 \\ -\frac{1}{120} & \frac{1}{24} & -\frac{1}{12} & \frac{1}{24} & -\frac{1}{24} & \frac{1}{120} \end{bmatrix} \tag{A.44}$$

Quintic interpolation can be completed in each direction to perform a biquintic interpolation. The same process can be followed for the y direction to find $\beta^5(\Delta y)$, which results in the same matrix as for $\beta^5(\Delta x)$. To interpolate in both directions, the resampling matrices can be used to pre- and post-multiply the sampling coefficient matrix, resulting in Equation A.45.

$$p(x,y) = \begin{bmatrix} 1 & \Delta x & \Delta x^2 & \Delta x^3 & \Delta x^4 & \Delta x^5 \end{bmatrix} QCQ^T \begin{bmatrix} 1 \\ \Delta y \\ \Delta y^2 \\ \Delta y^3 \\ \Delta y^4 \\ \Delta y^5 \end{bmatrix} \tag{A.45}$$

where C is the 6x6 sampling coefficient matrix, which can be found from the deconvolution of the image and the B-spline kernel, and

$$Q = \begin{bmatrix} \frac{1}{120} & \frac{13}{60} & \frac{11}{20} & \frac{13}{60} & \frac{1}{120} & 0 \\ -\frac{1}{24} & -\frac{5}{12} & 0 & \frac{5}{12} & \frac{1}{24} & 0 \\ \frac{1}{12} & \frac{1}{6} & -\frac{1}{2} & \frac{1}{6} & \frac{1}{12} & 0 \\ -\frac{1}{12} & \frac{1}{6} & 0 & -\frac{1}{6} & \frac{1}{12} & 0 \\ \frac{1}{24} & -\frac{1}{6} & \frac{1}{4} & -\frac{1}{6} & \frac{1}{24} & 0 \\ -\frac{1}{120} & \frac{1}{24} & -\frac{1}{12} & \frac{1}{24} & -\frac{1}{24} & \frac{1}{120} \end{bmatrix}$$

Appendix B

Graphs

B.1 Numeric Results

B.1.1 Shape Function

2D Results

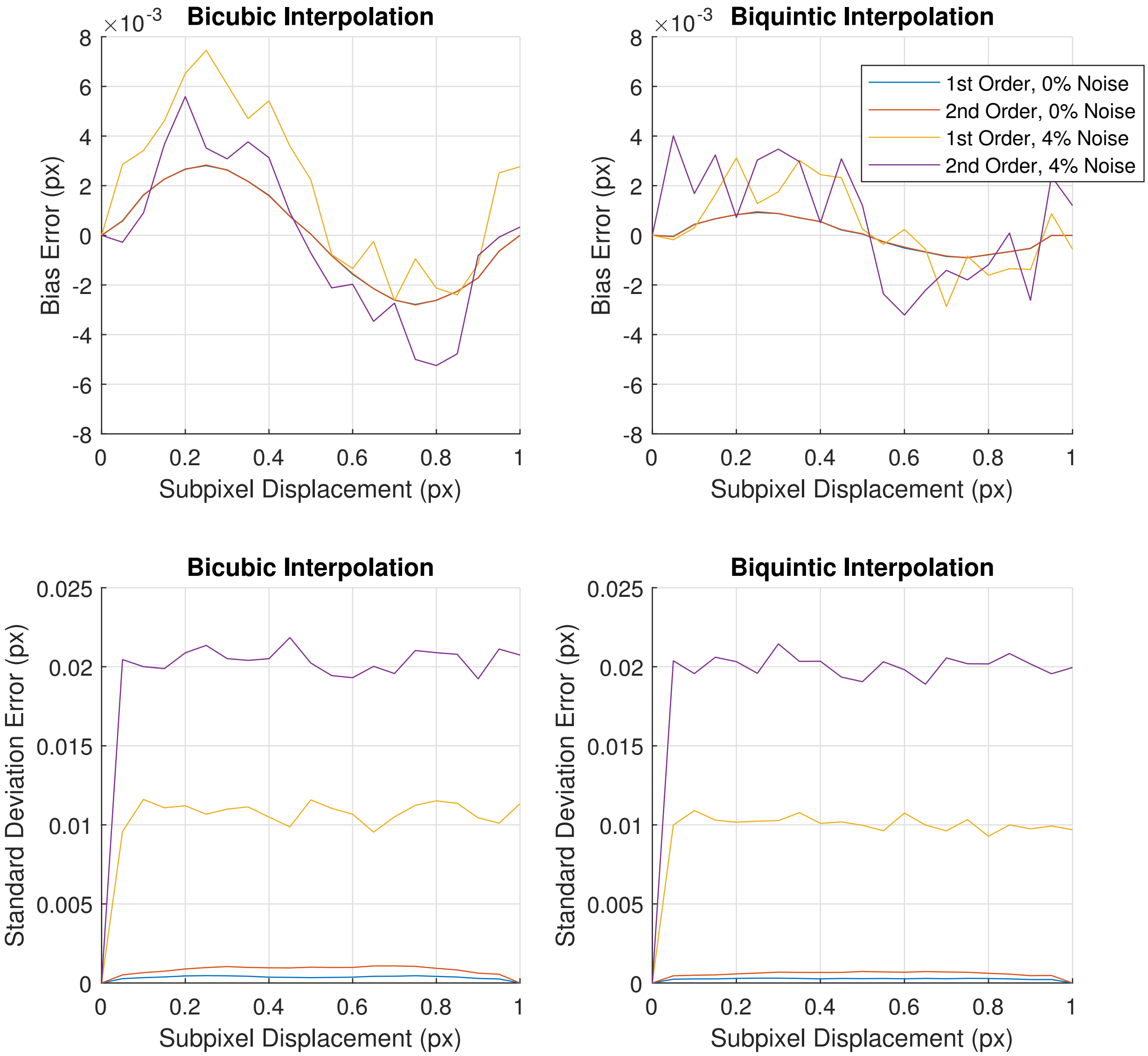


Figure B.1: Errors present in 2D analysis due to shape function (no blur added).

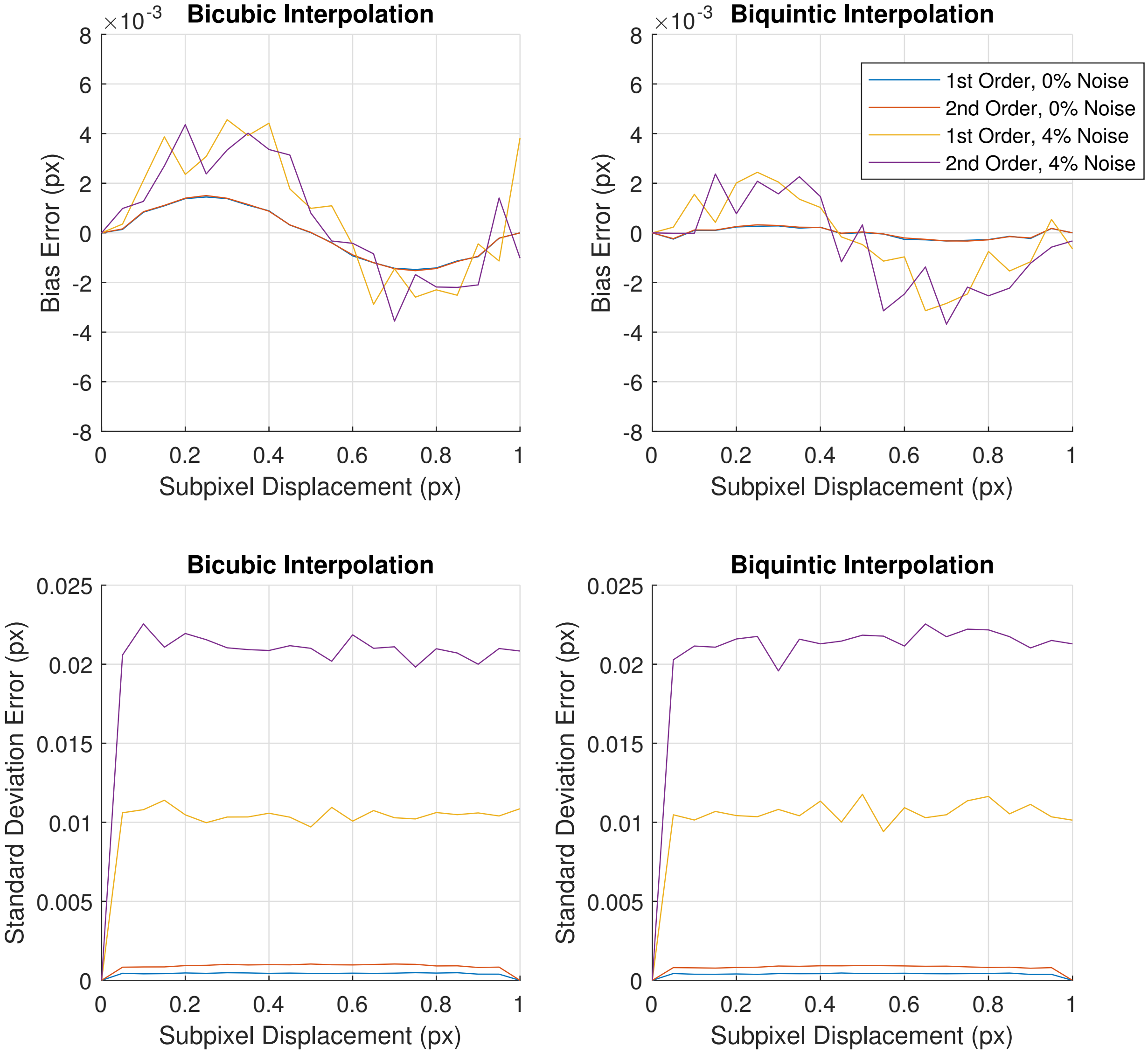


Figure B.2: Errors present in 2D analysis due to shape function (3x3 blur added).

3D Results

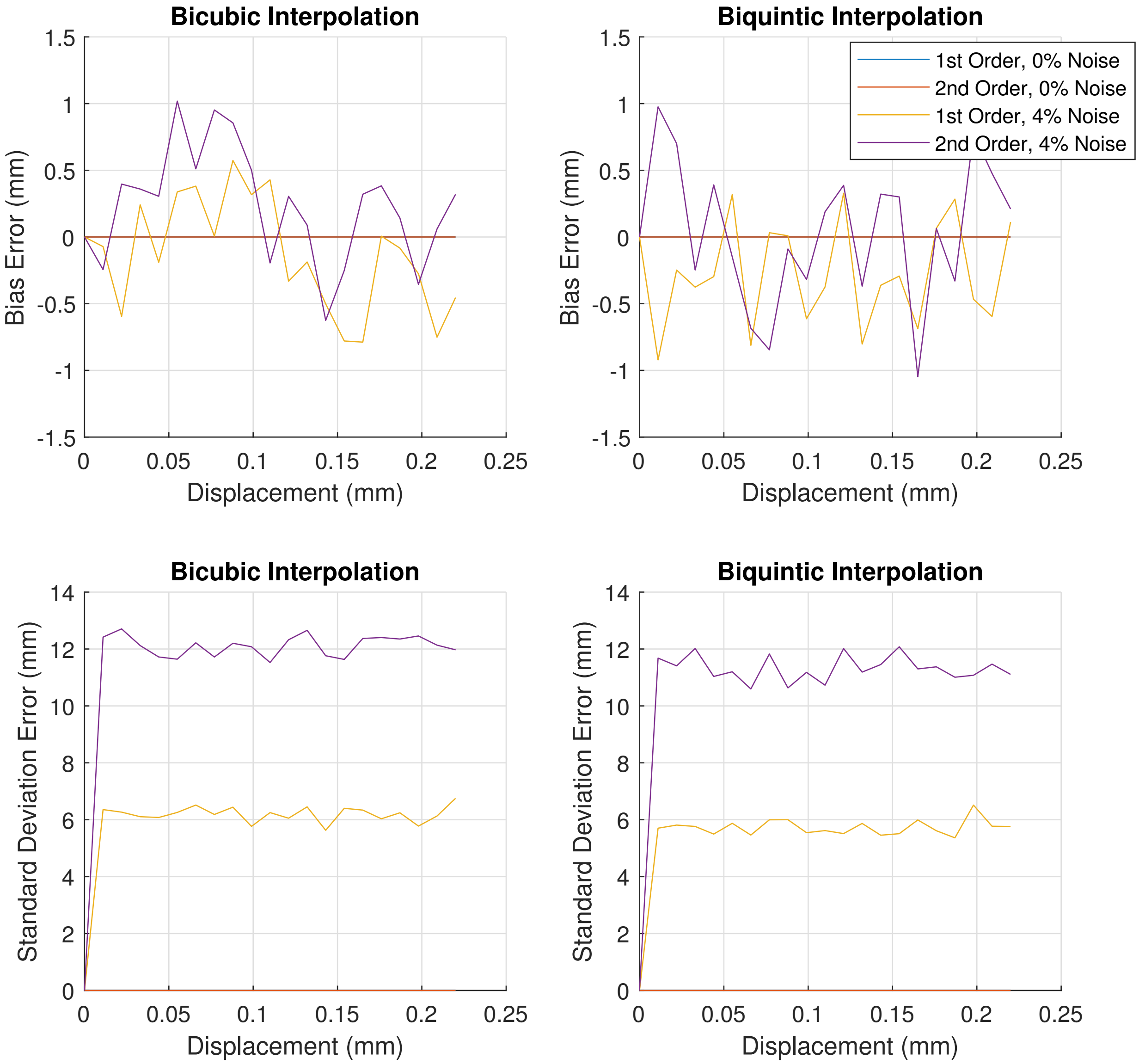


Figure B.3: Errors present in 3D analysis due to shape function (no blur added).

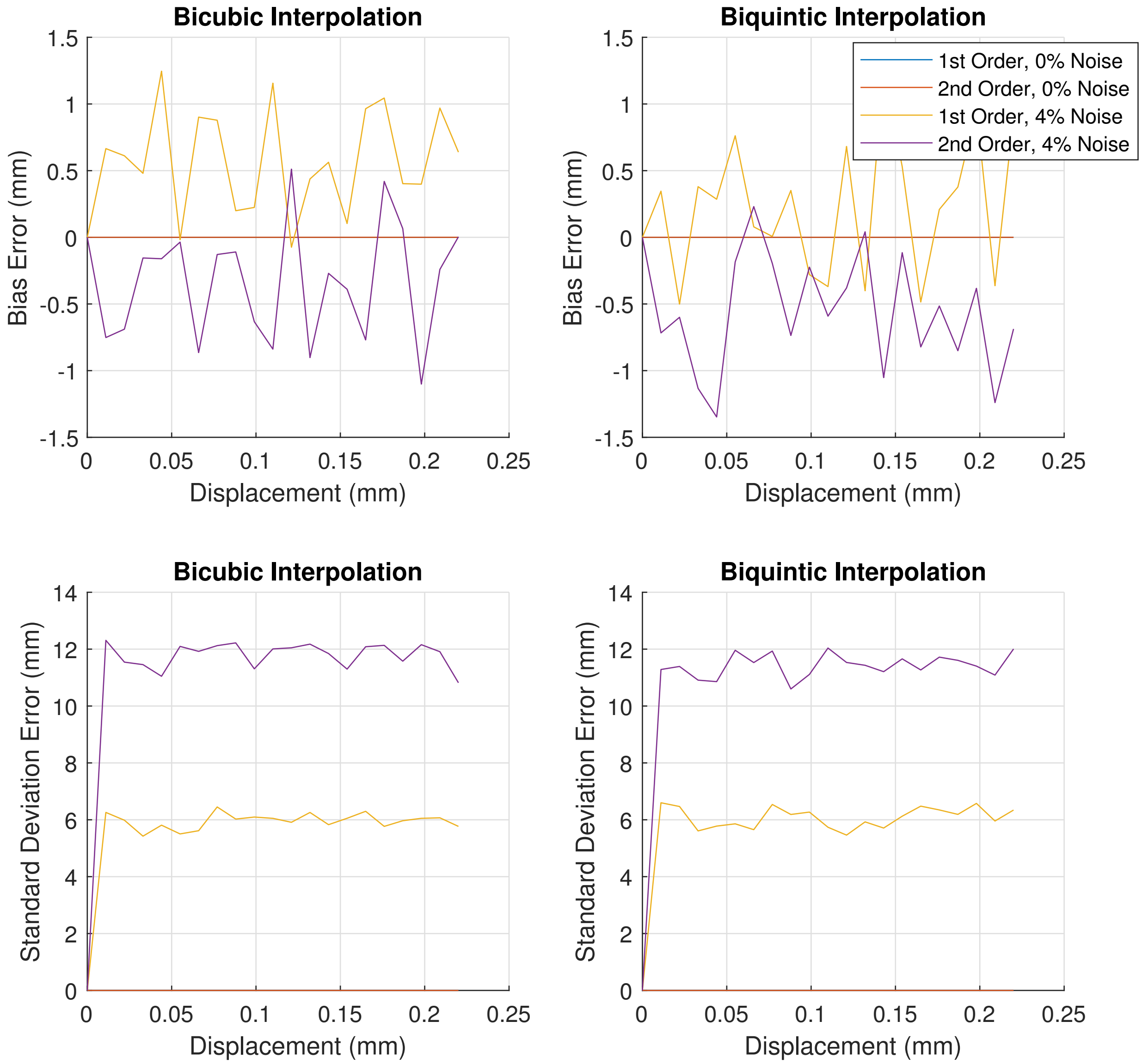


Figure B.4: Errors present in 3D analysis due to shape function (3x3 blur added).

B.1.2 Interpolation

2D Results

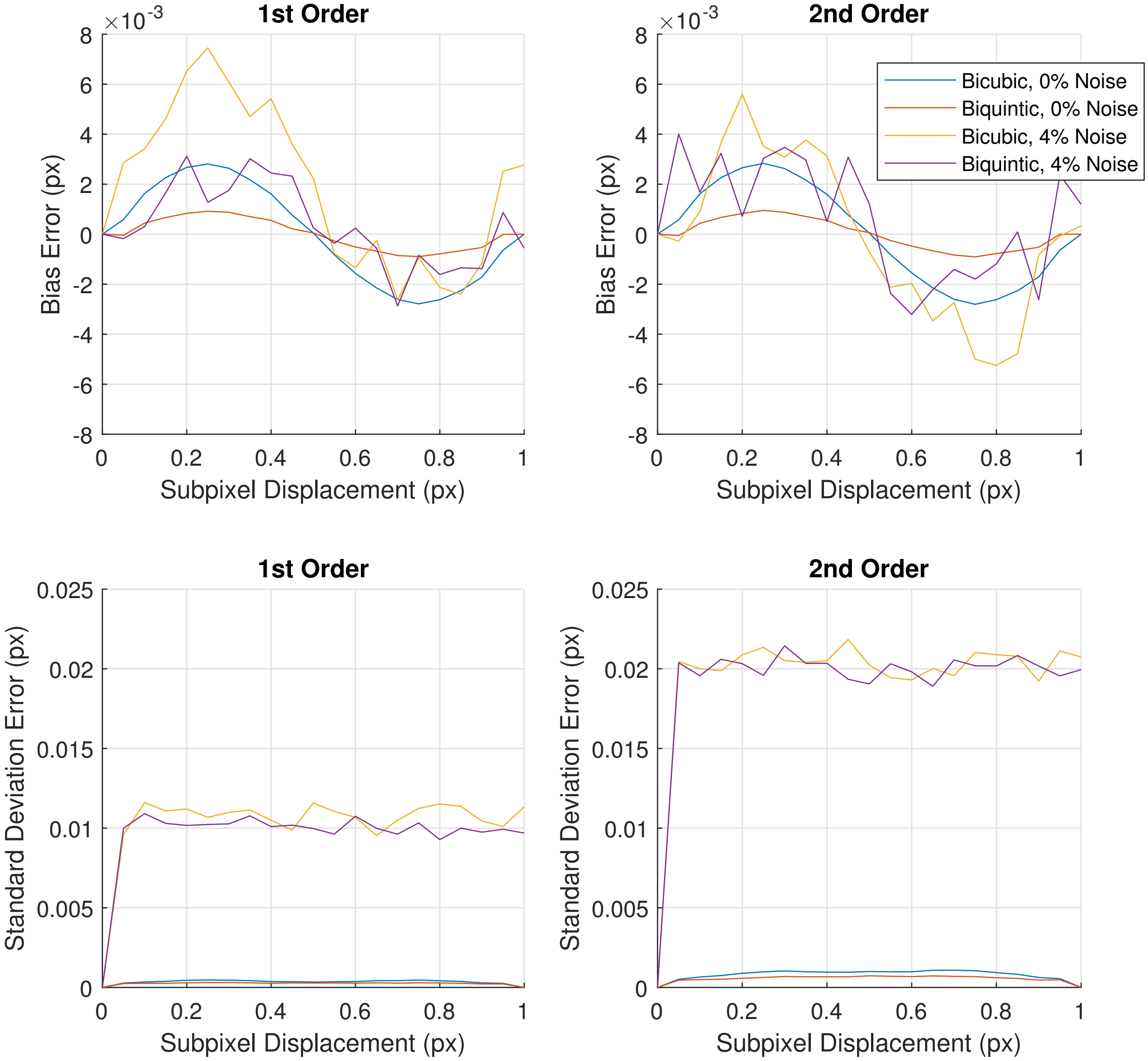


Figure B.5: Errors present in 2D analysis due to interpolation (no blur added).

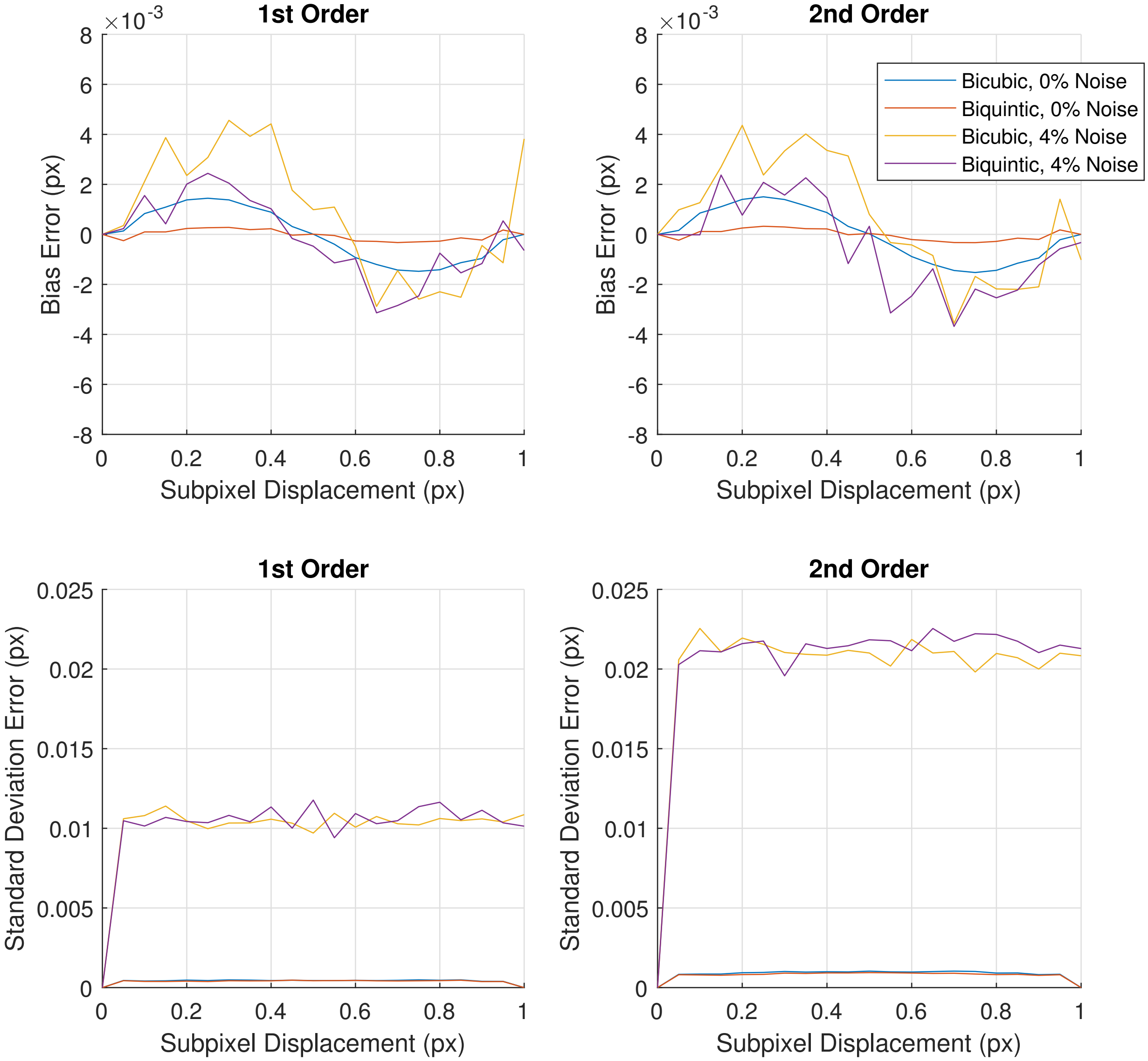


Figure B.6: Errors present in 2D analysis due to interpolation (3x3 blur added).

3D Results

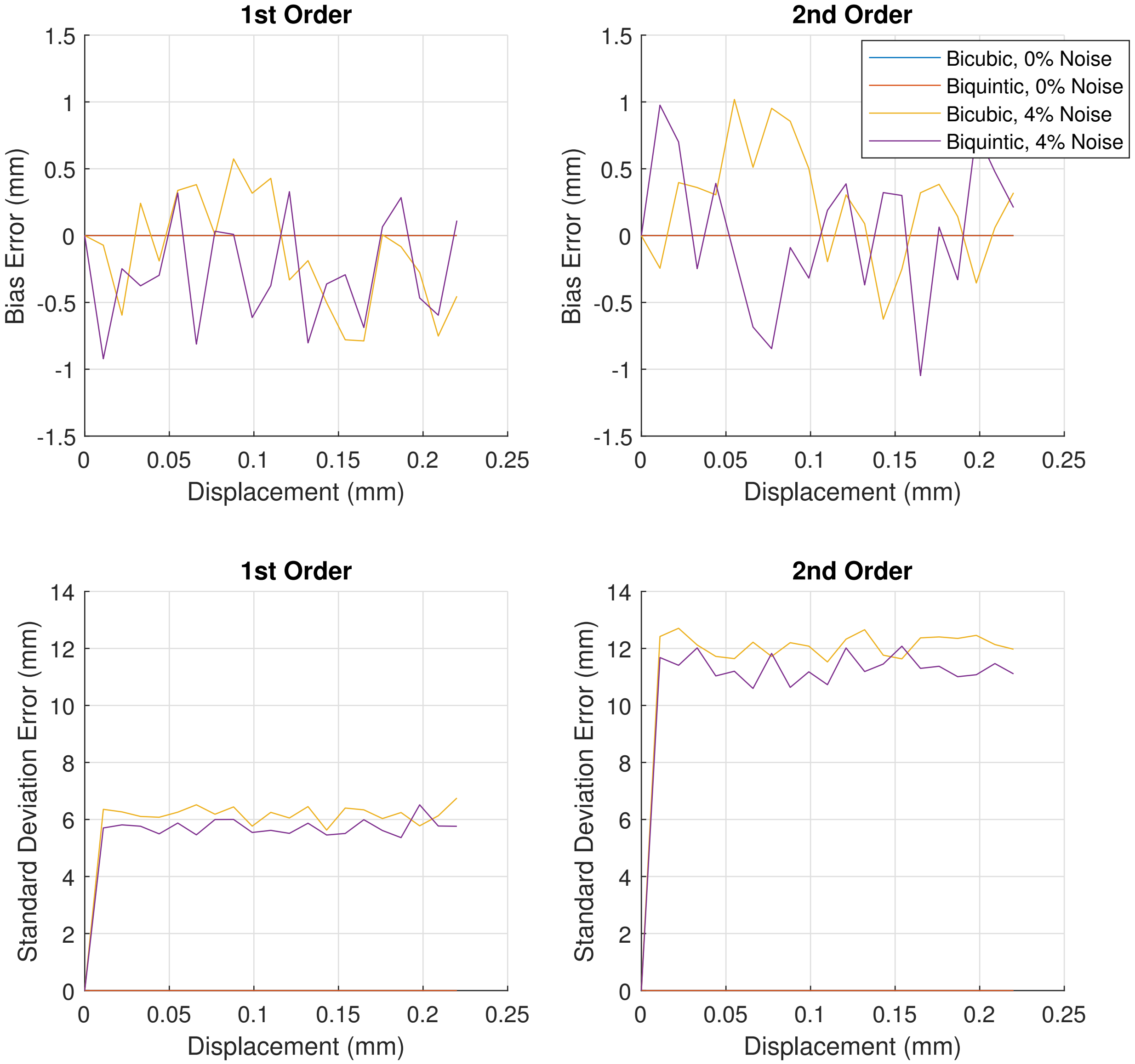


Figure B.7: Errors present in 3D analysis due to interpolation (no blur added).

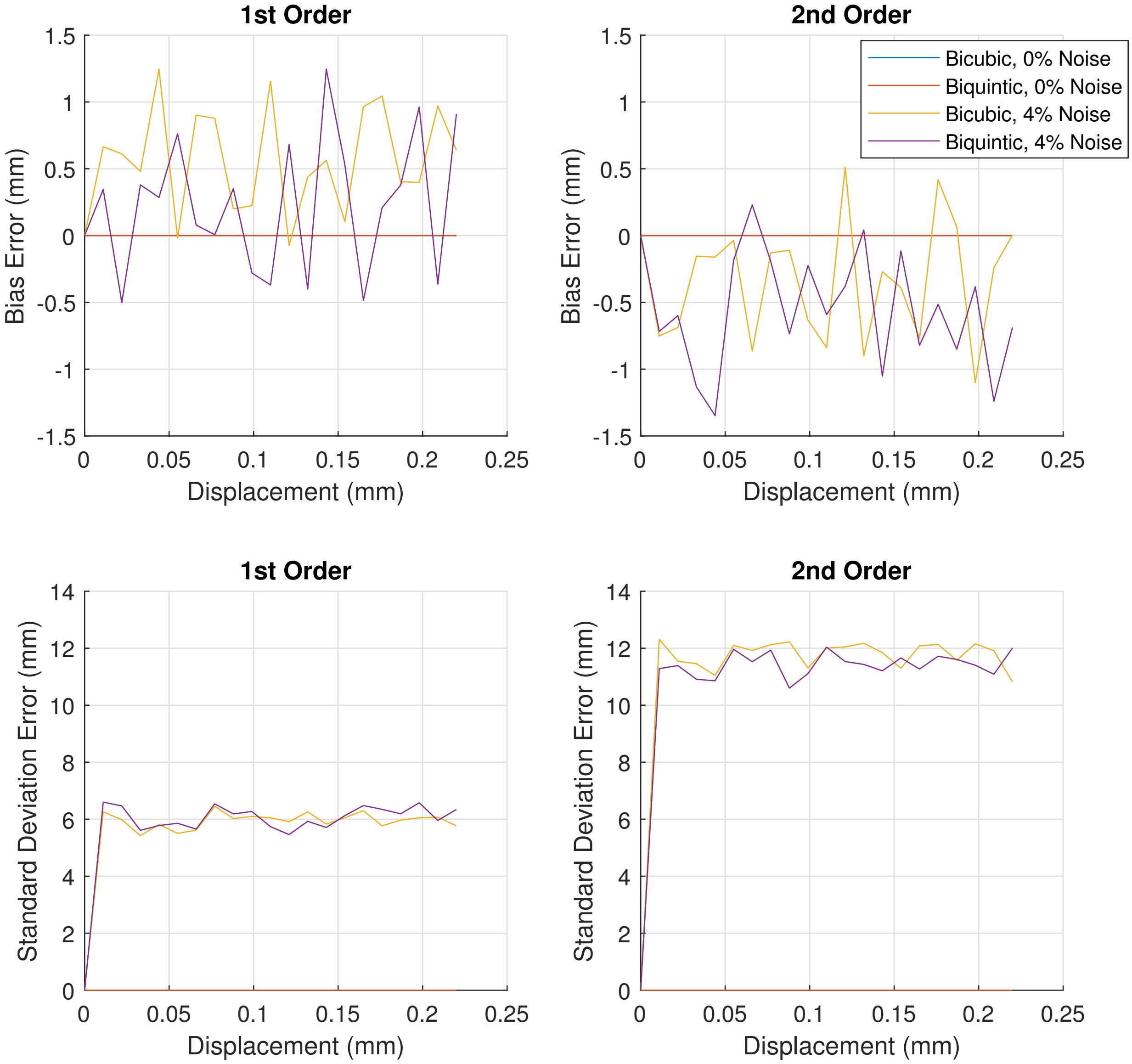


Figure B.8: Errors present in 3D analysis due to interpolation (3x3 blur added).

B.1.3 Noise Level

2D Results

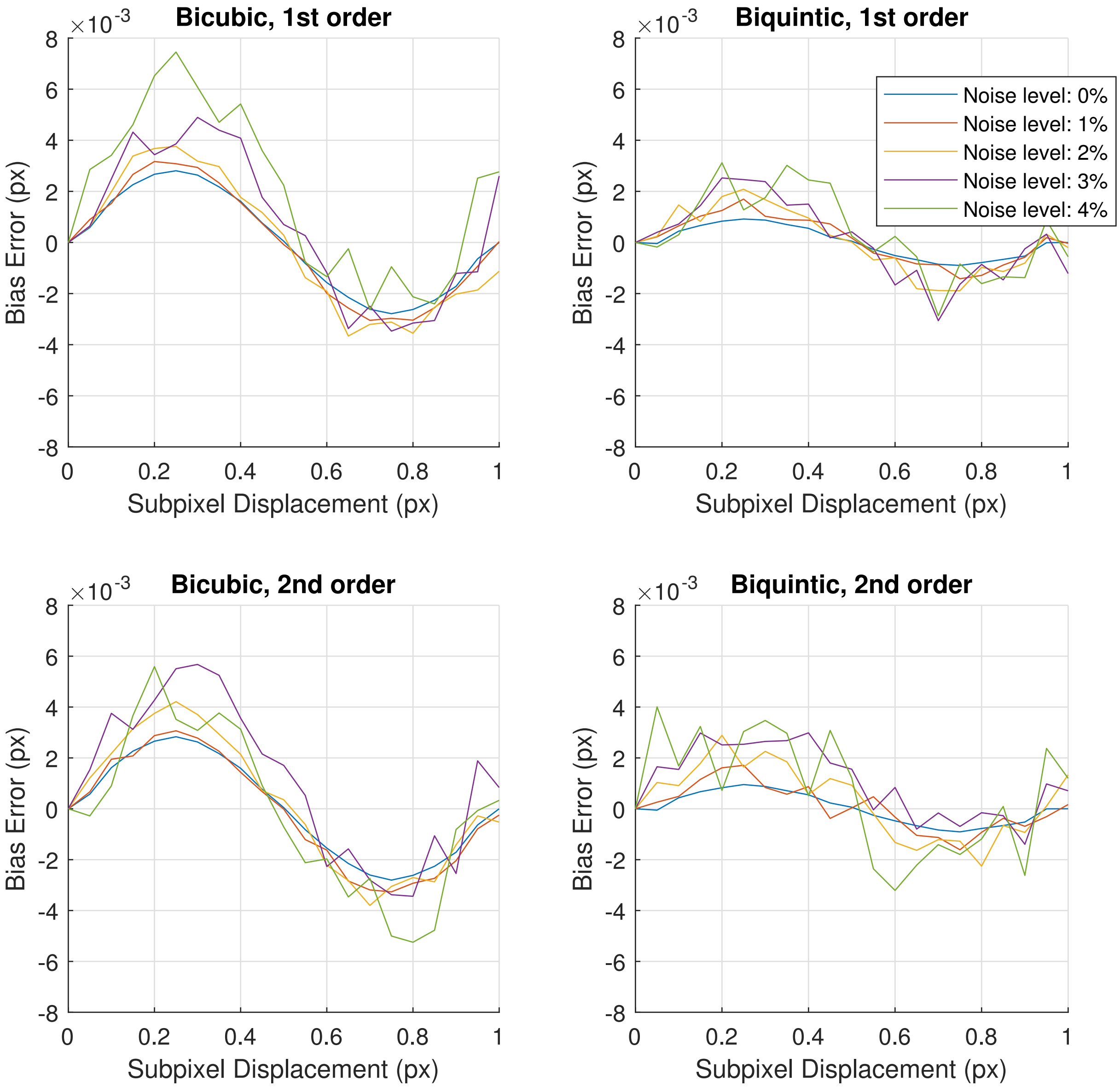


Figure B.9: Errors present in 2D analysis due to noise.

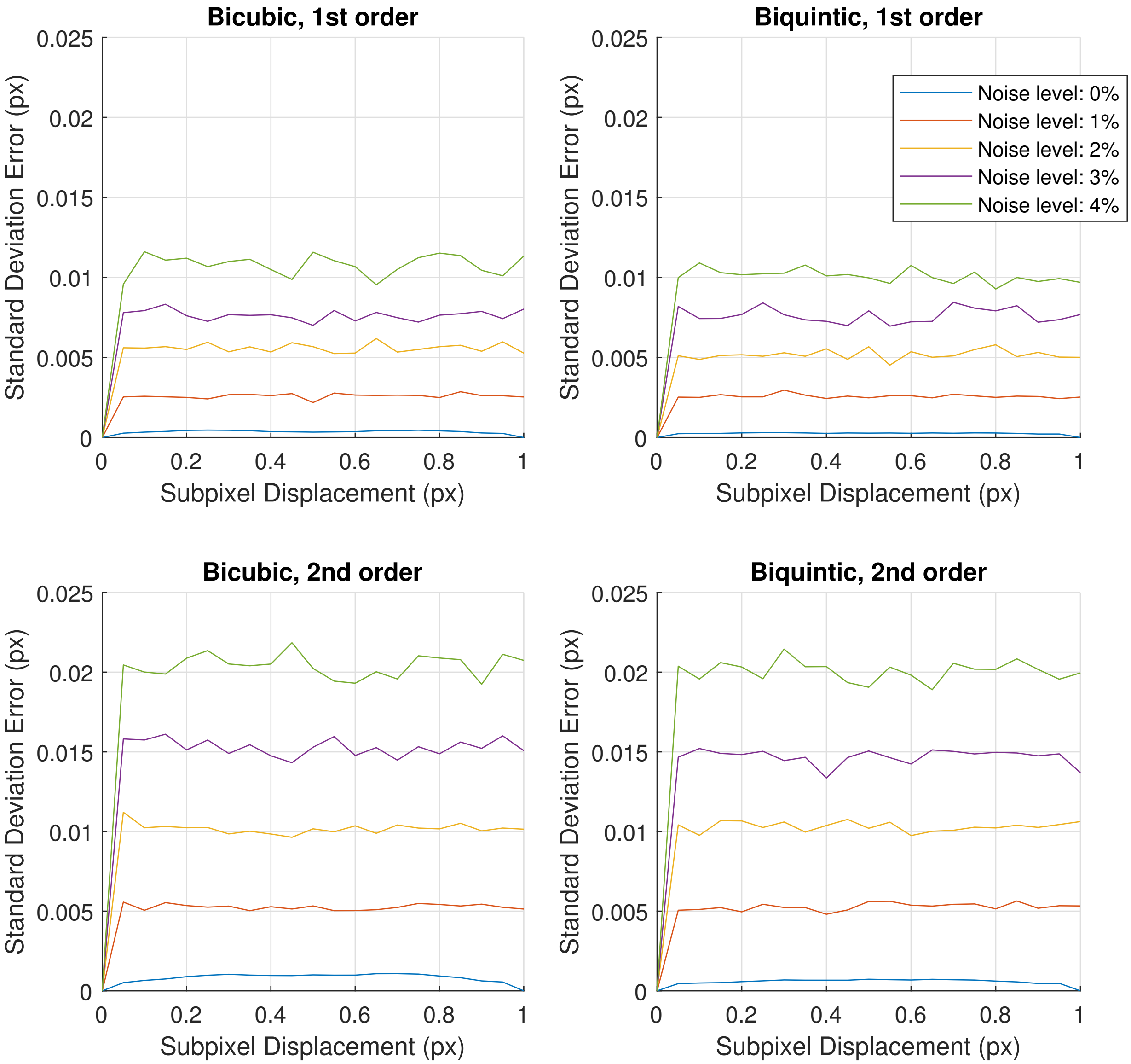


Figure B.10: Errors present in 2D analysis due to noise.

3D Results

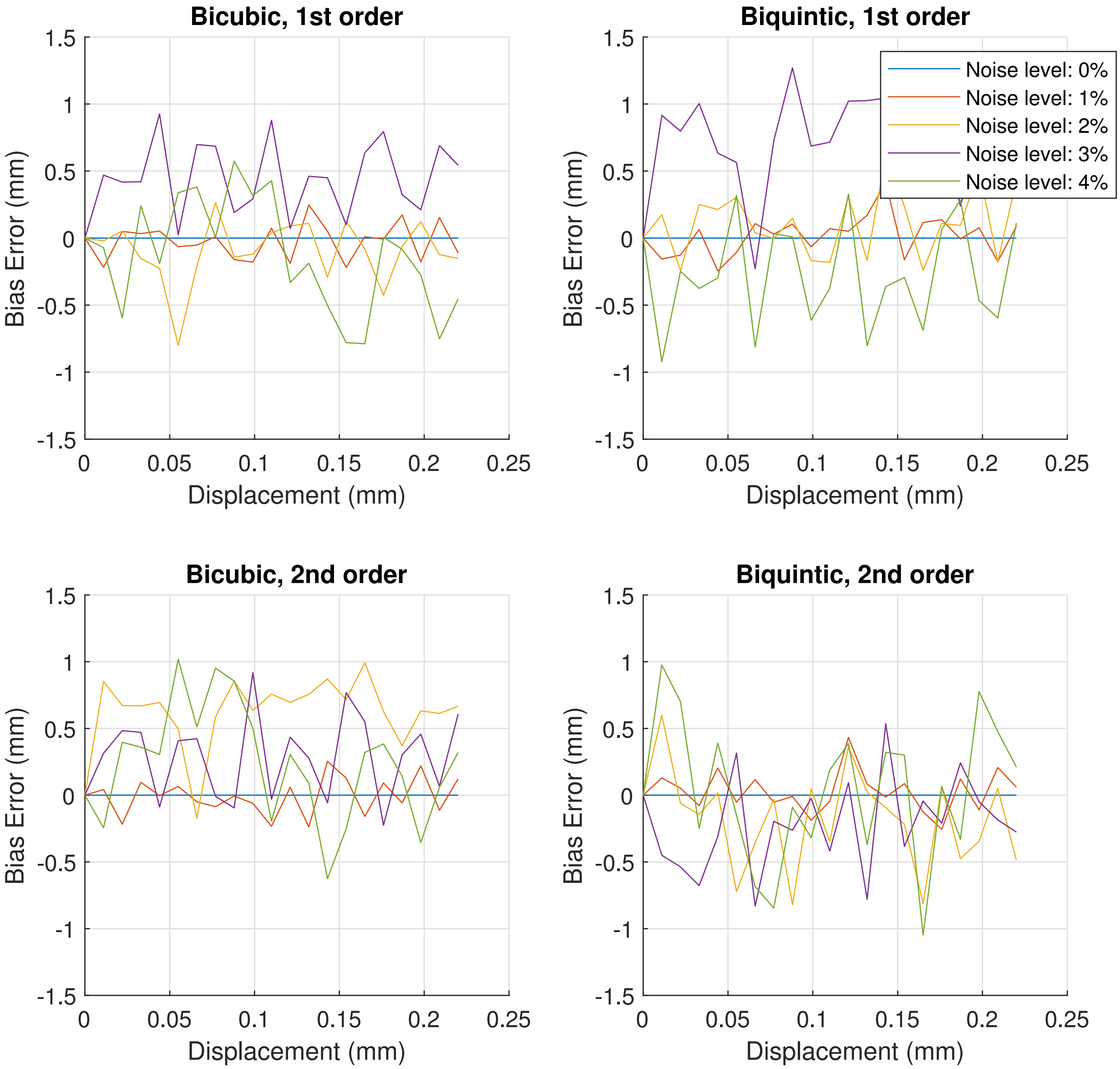


Figure B.11: Errors present in 3D analysis due to noise.

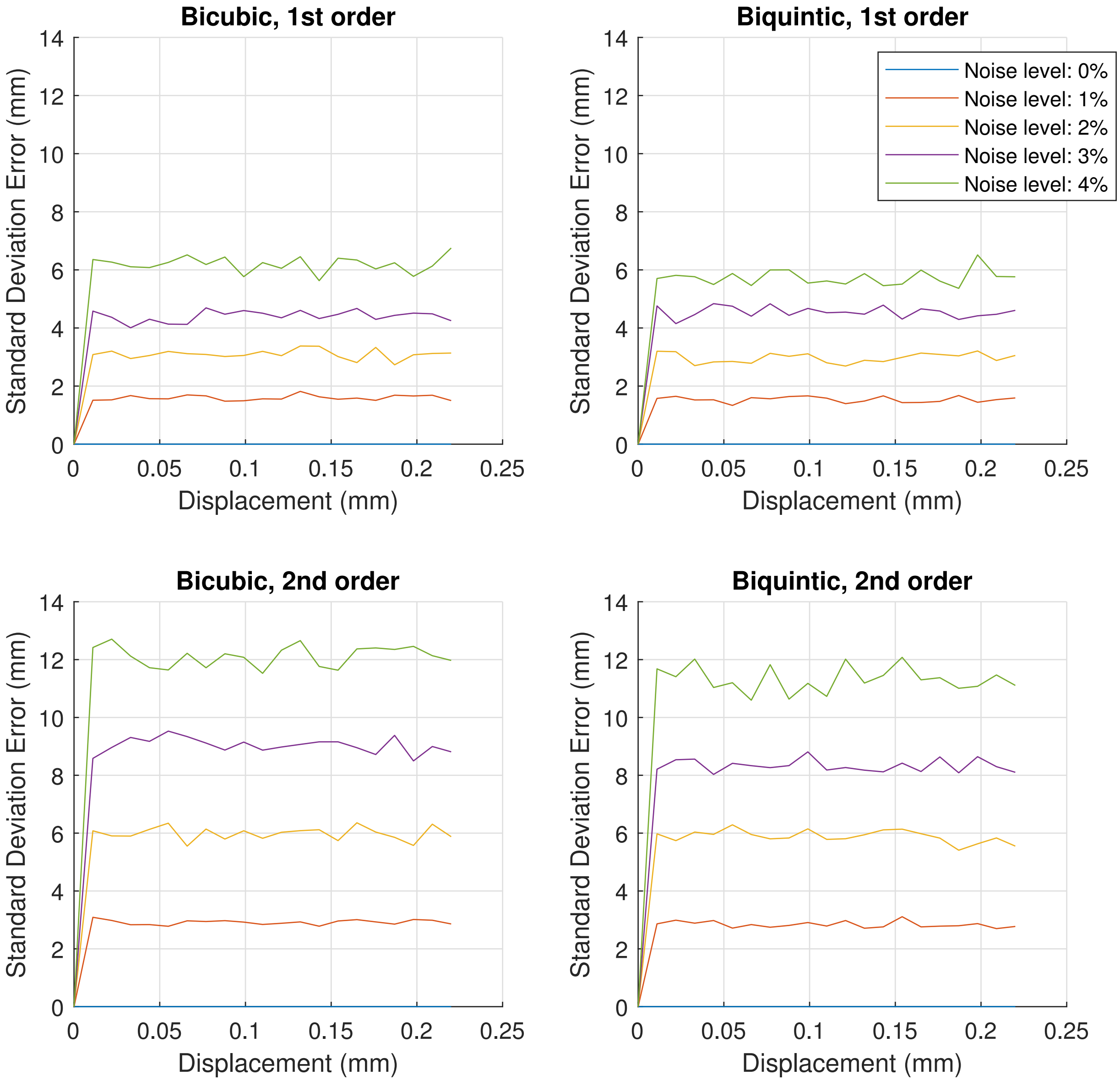


Figure B.12: Errors present in 3D analysis due to noise.

B.1.4 Noise Reduction (Gaussian Pre-Filtering)

2D Results

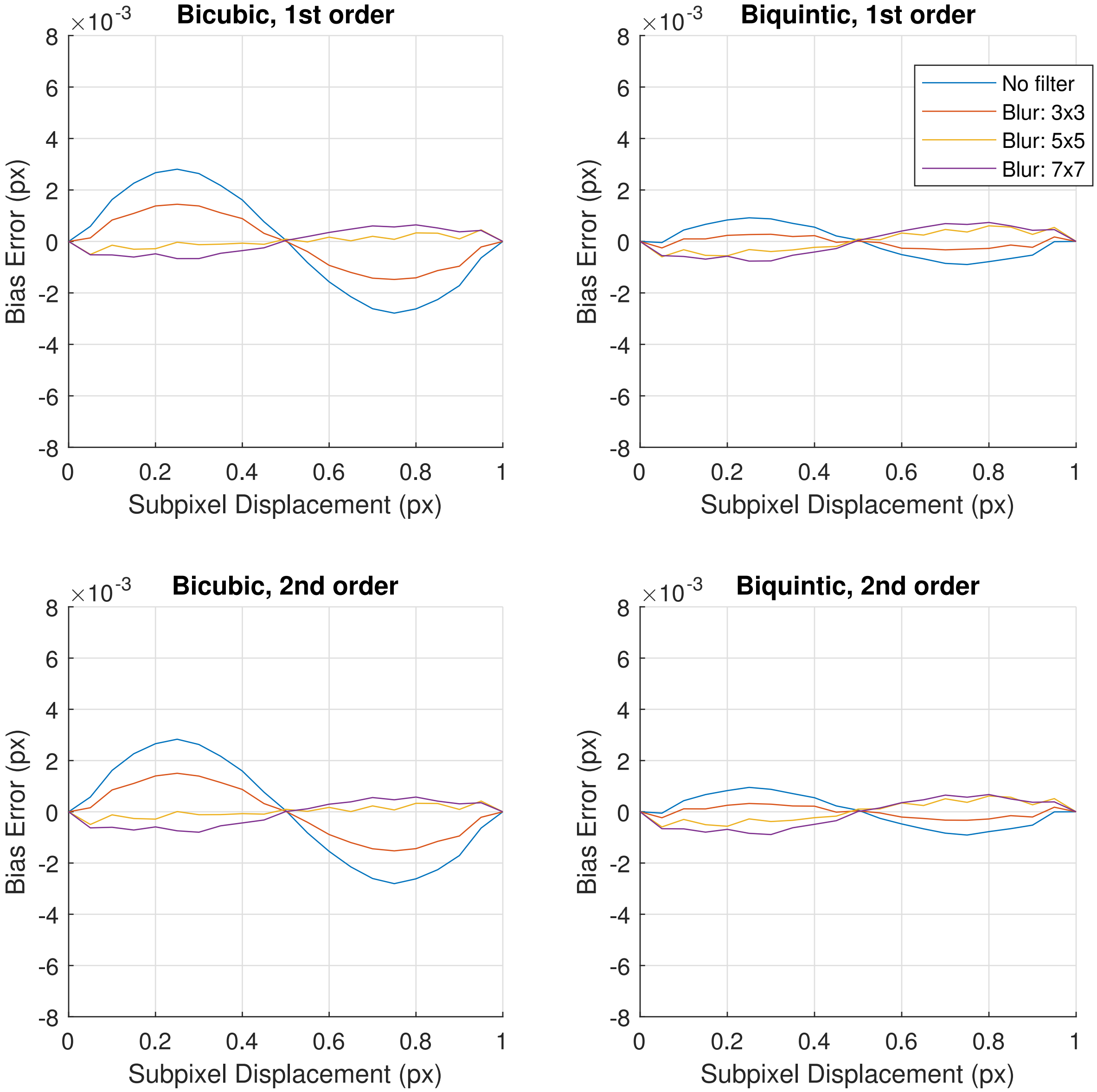


Figure B.13: Bias errors present in 2D analysis due to Gaussian blur (0% noise).

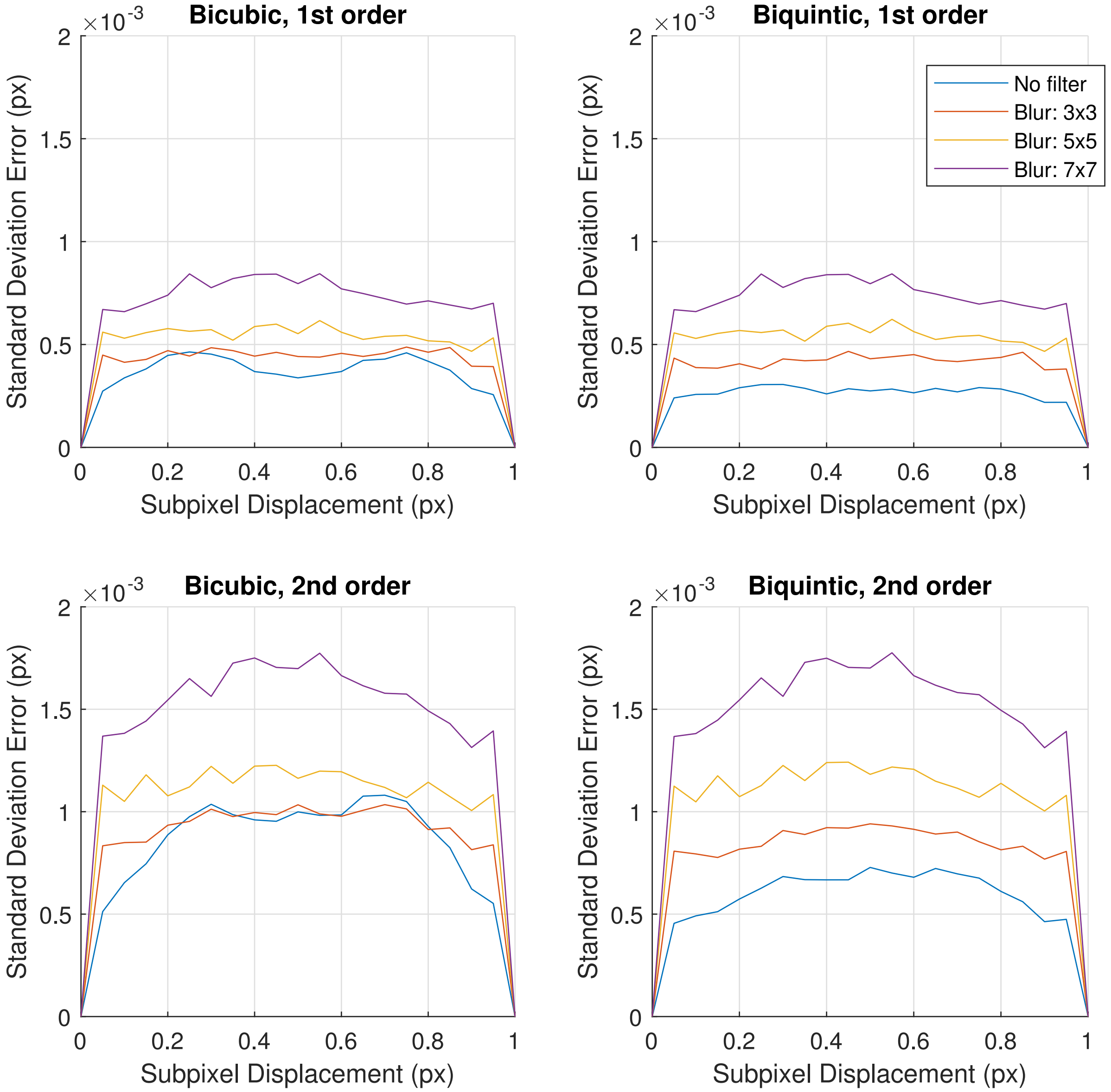


Figure B.14: Standard deviation errors present in 2D analysis due to Gaussian blur (0% noise).

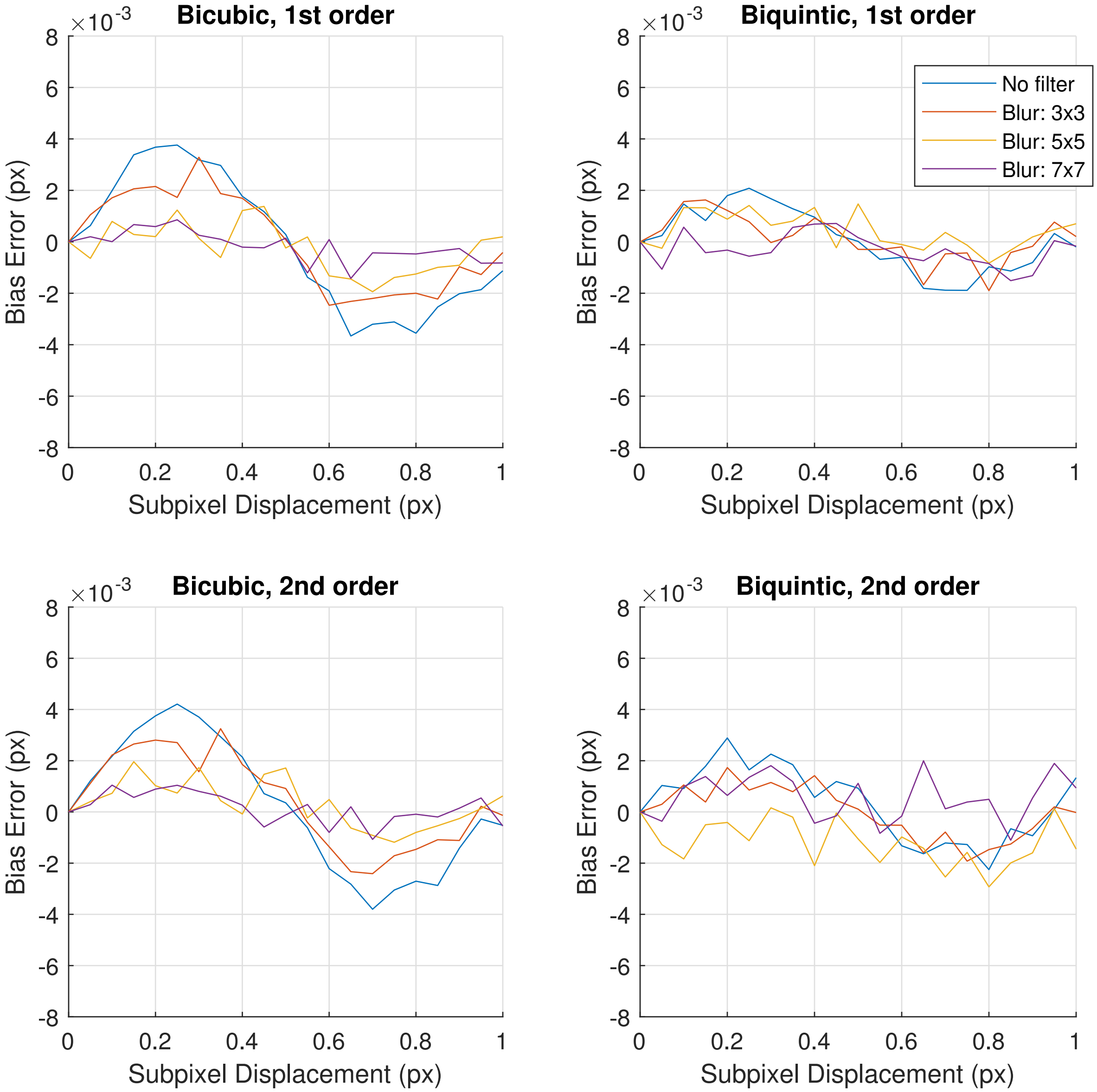


Figure B.15: Bias errors present in 2D analysis due to Gaussian blur (2% noise).

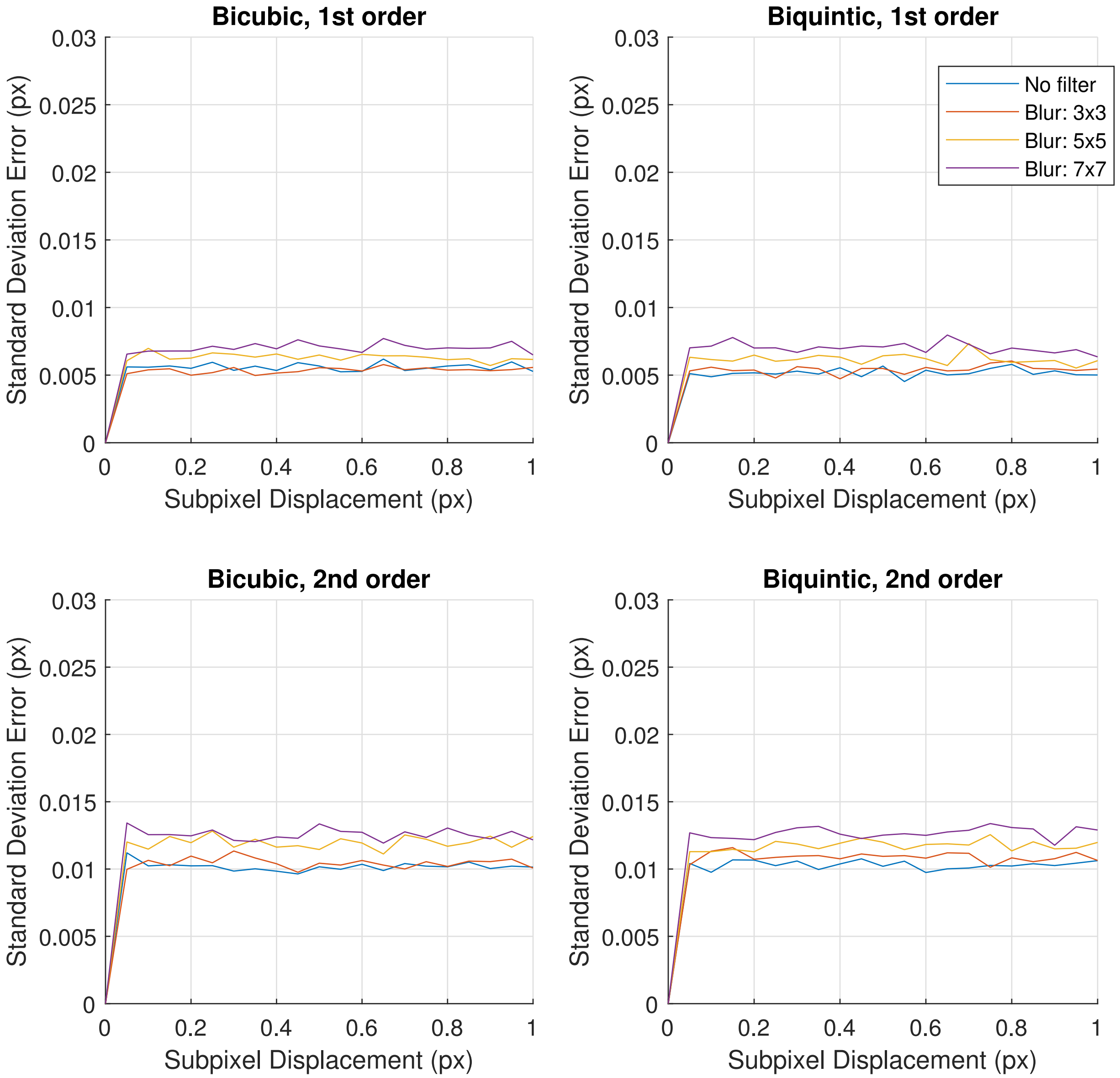


Figure B.16: Standard deviation errors present in 2D analysis due to Gaussian blur (2% noise).

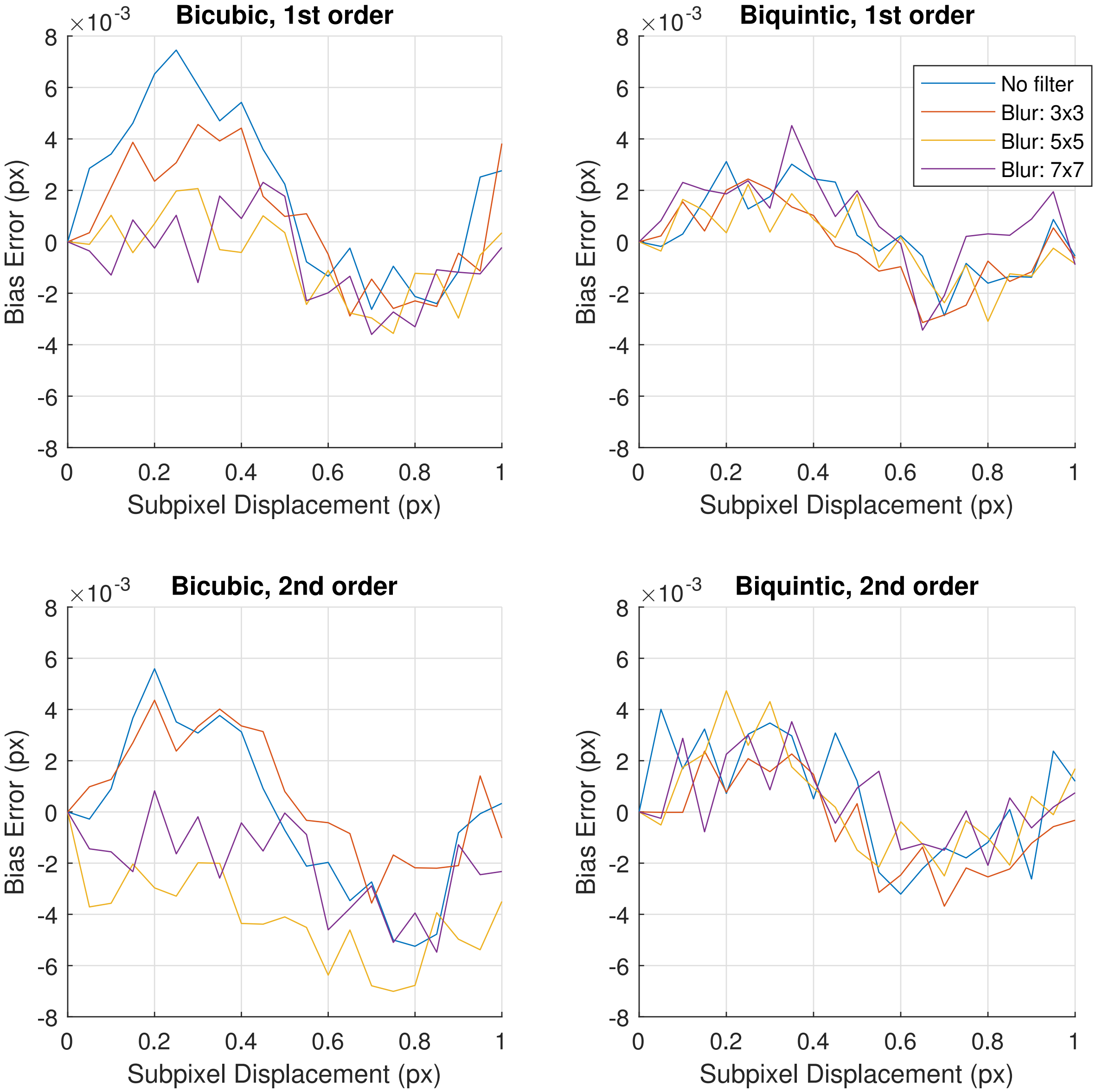


Figure B.17: Bias errors present in 2D analysis due to Gaussian blur (4% noise).

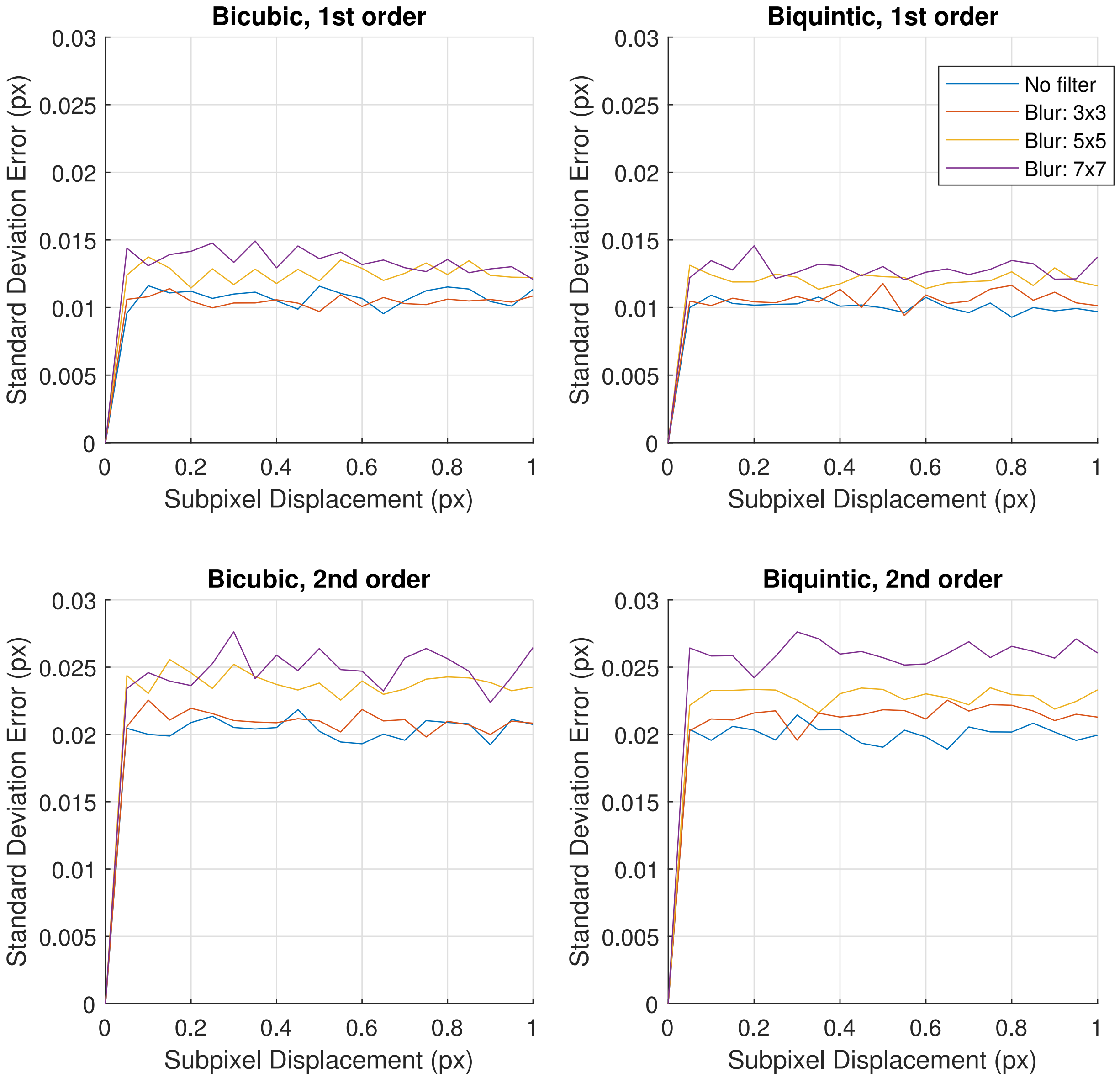


Figure B.18: Standard deviation errors present in 2D analysis due to Gaussian blur (4% noise).

3D Results

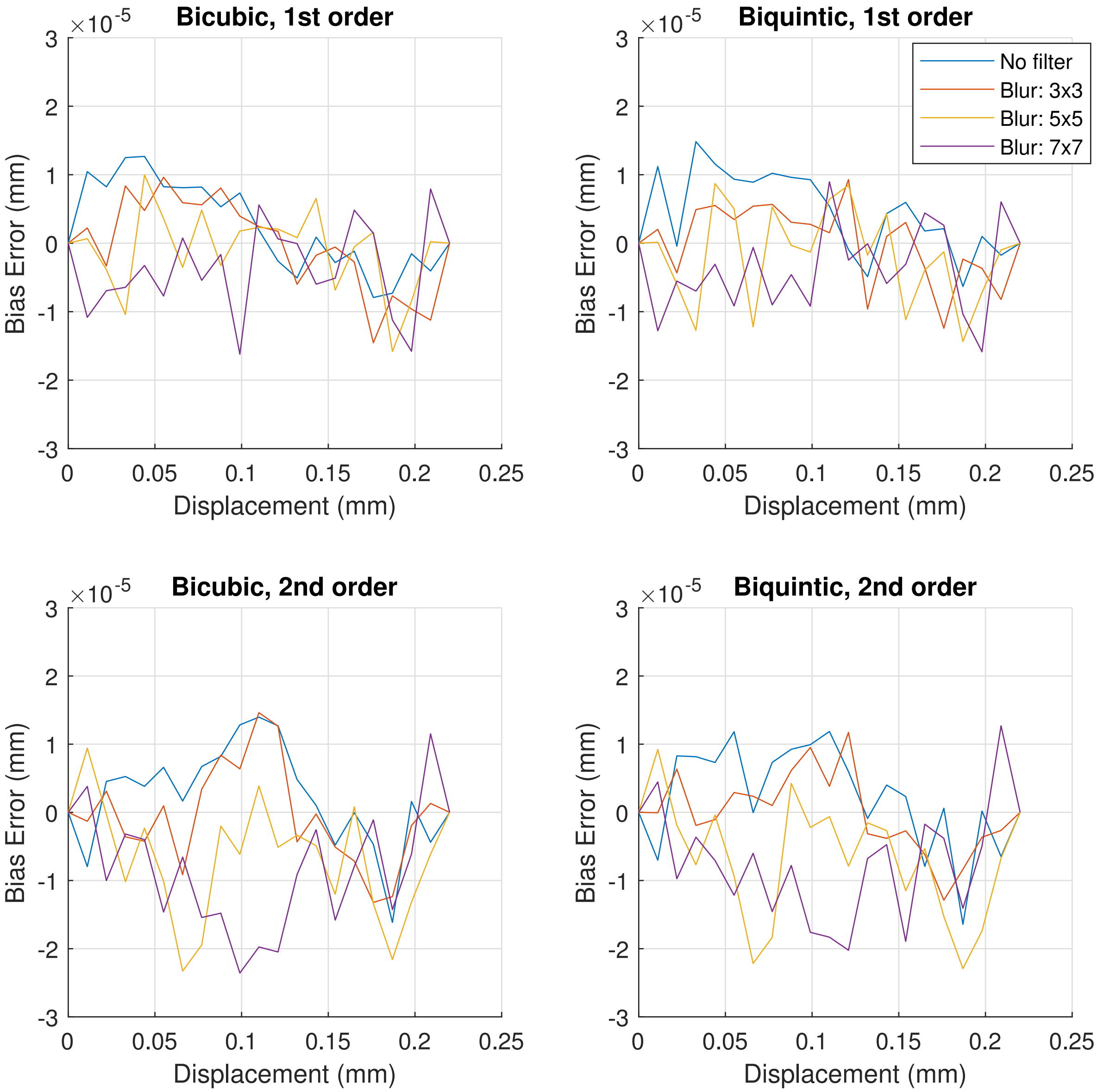


Figure B.19: Bias errors present in analysis 3D due to Gaussian blur (0% noise).

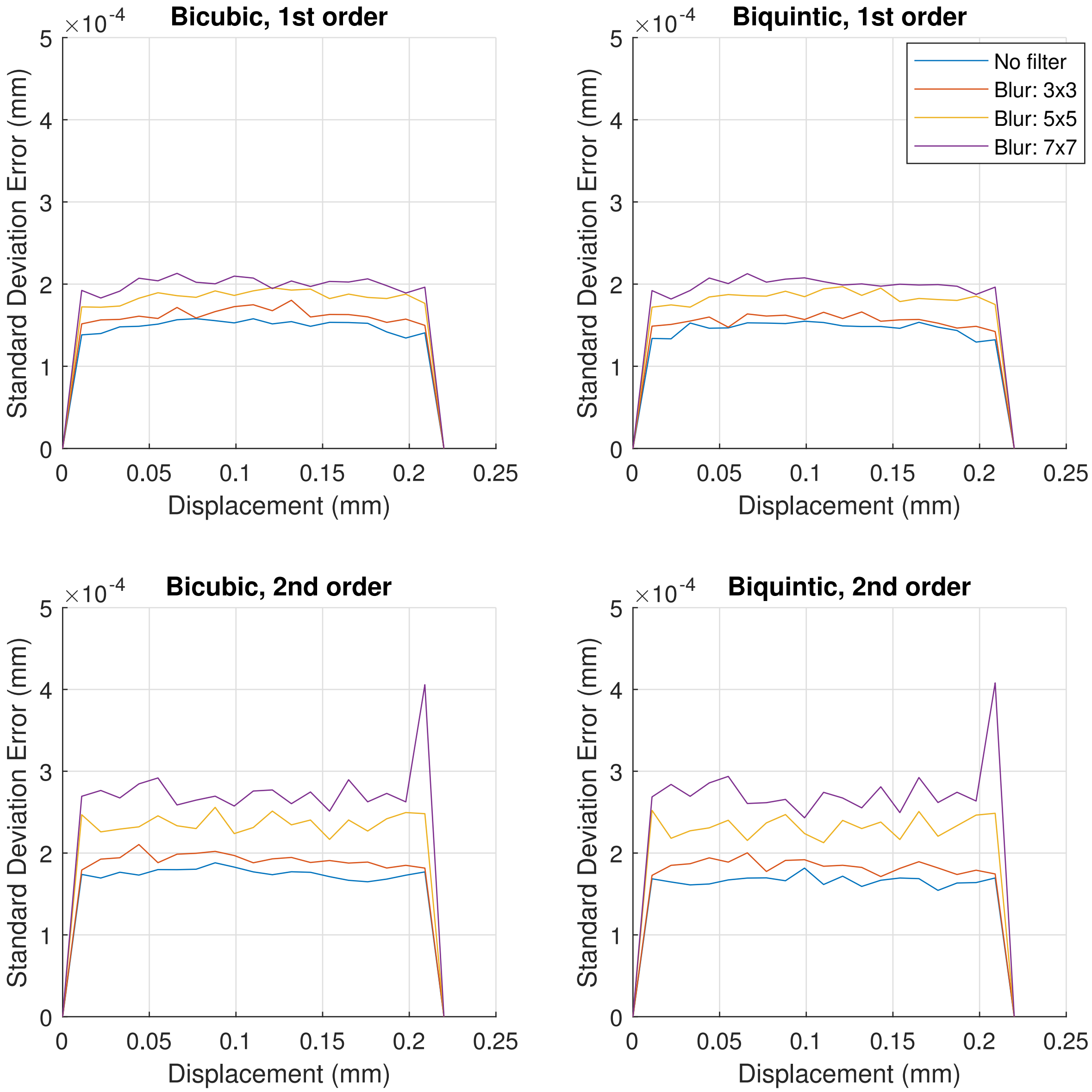


Figure B.20: Standard deviation errors present in 3D analysis due to Gaussian blur (0% noise).

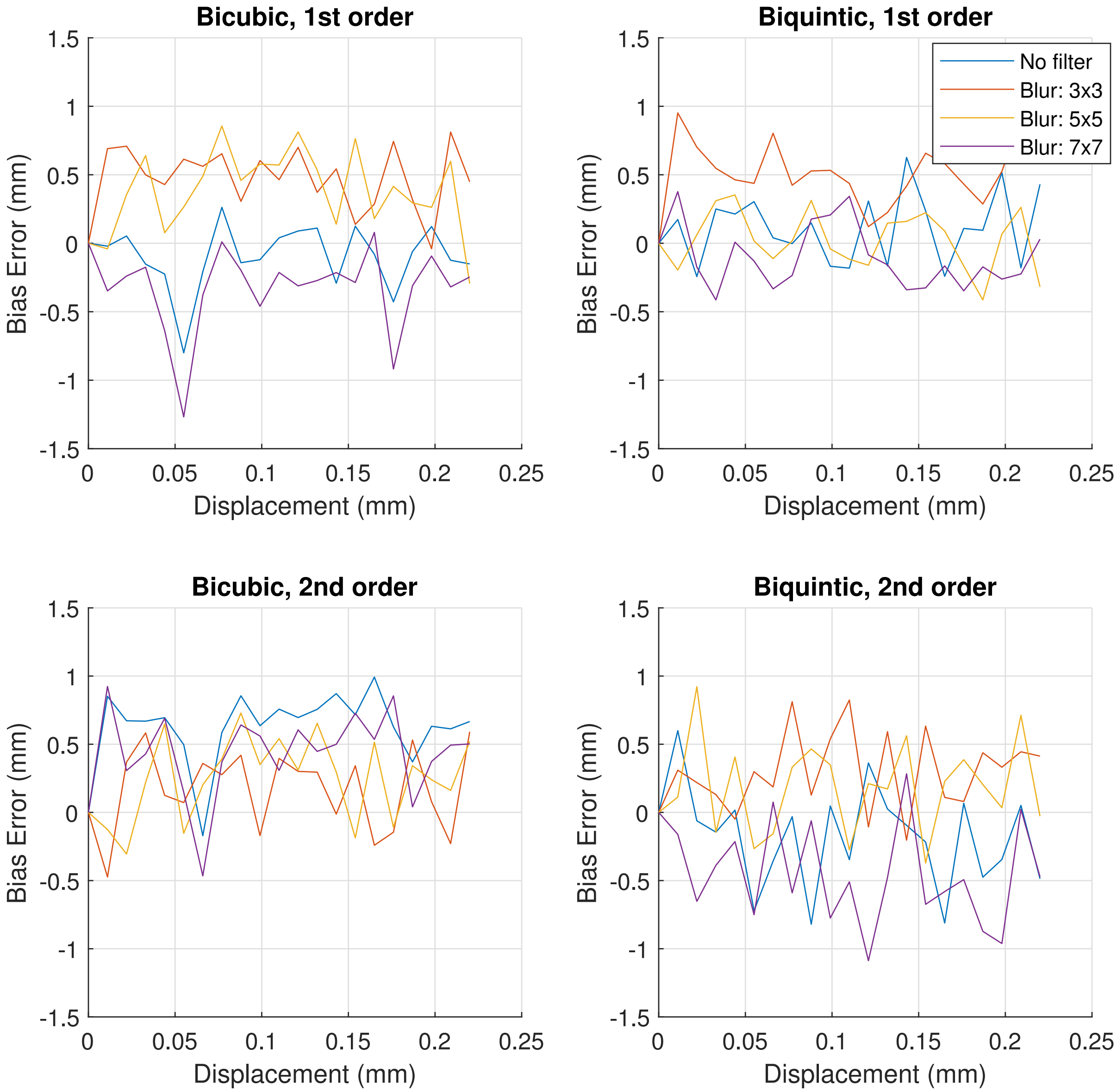


Figure B.21: Bias errors present in 3D analysis due to Gaussian blur (2% noise).

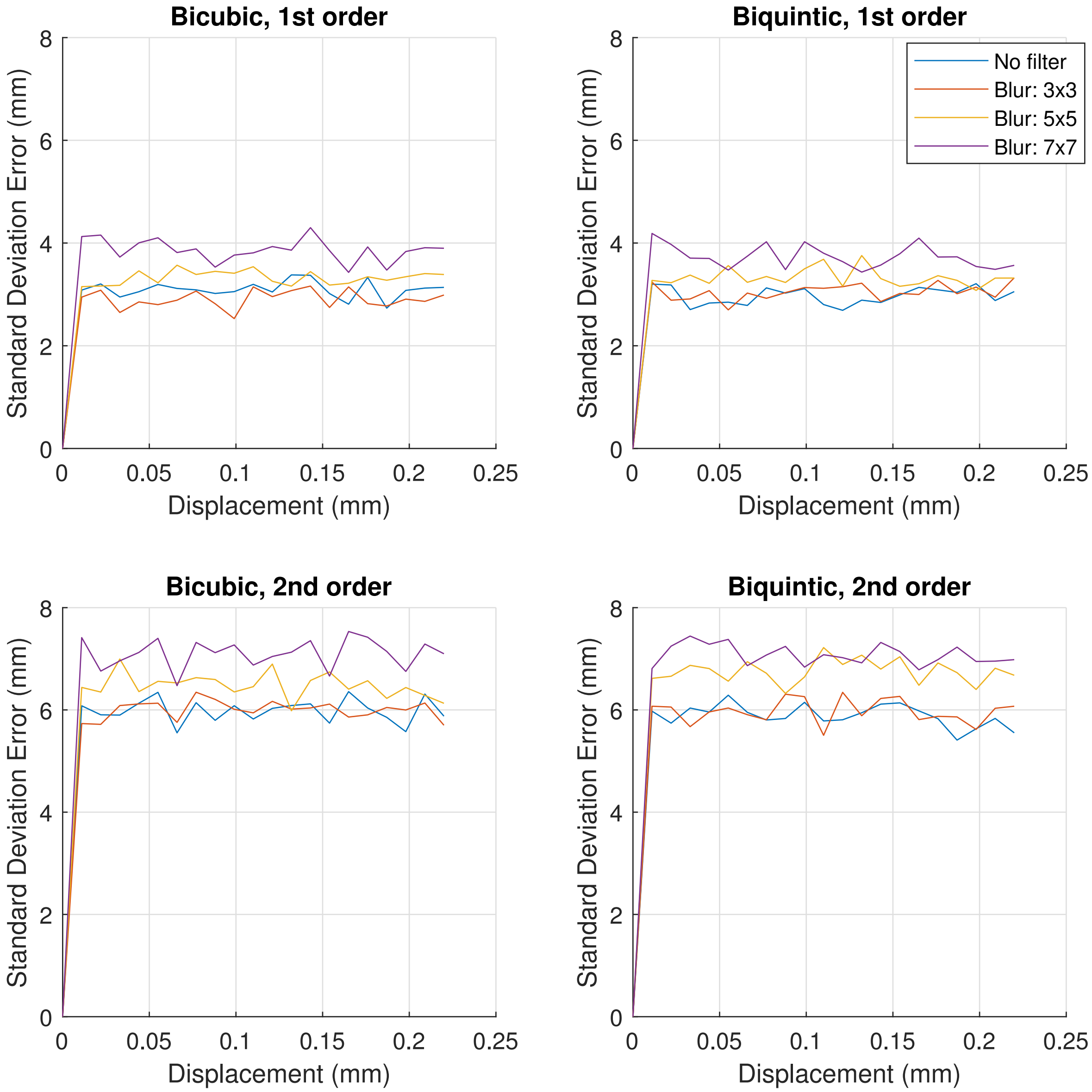


Figure B.22: Standard deviation errors present in 3D analysis due to Gaussian blur (2% noise).

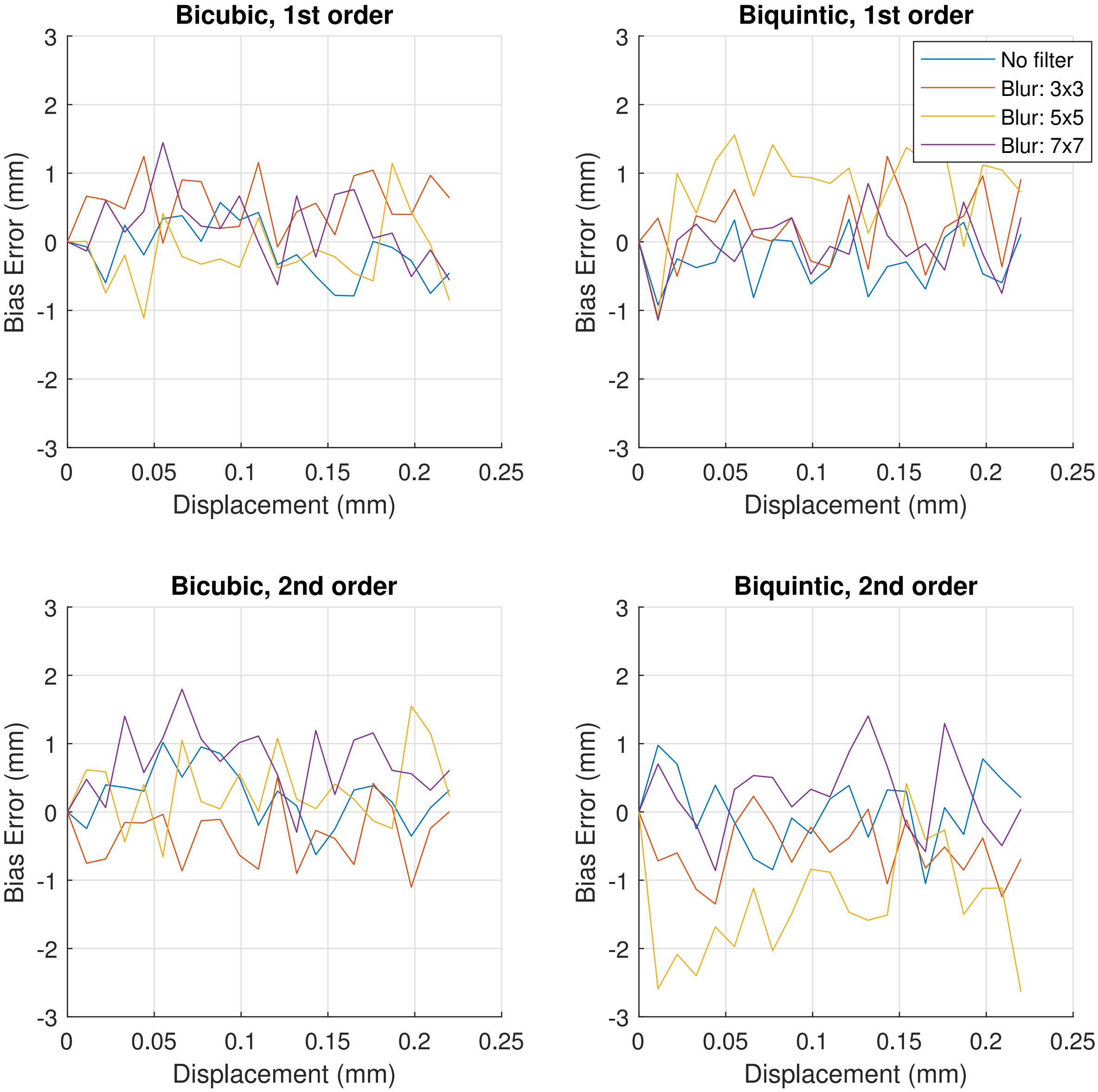


Figure B.23: Bias errors present in 3D analysis due to Gaussian blur (4% noise).

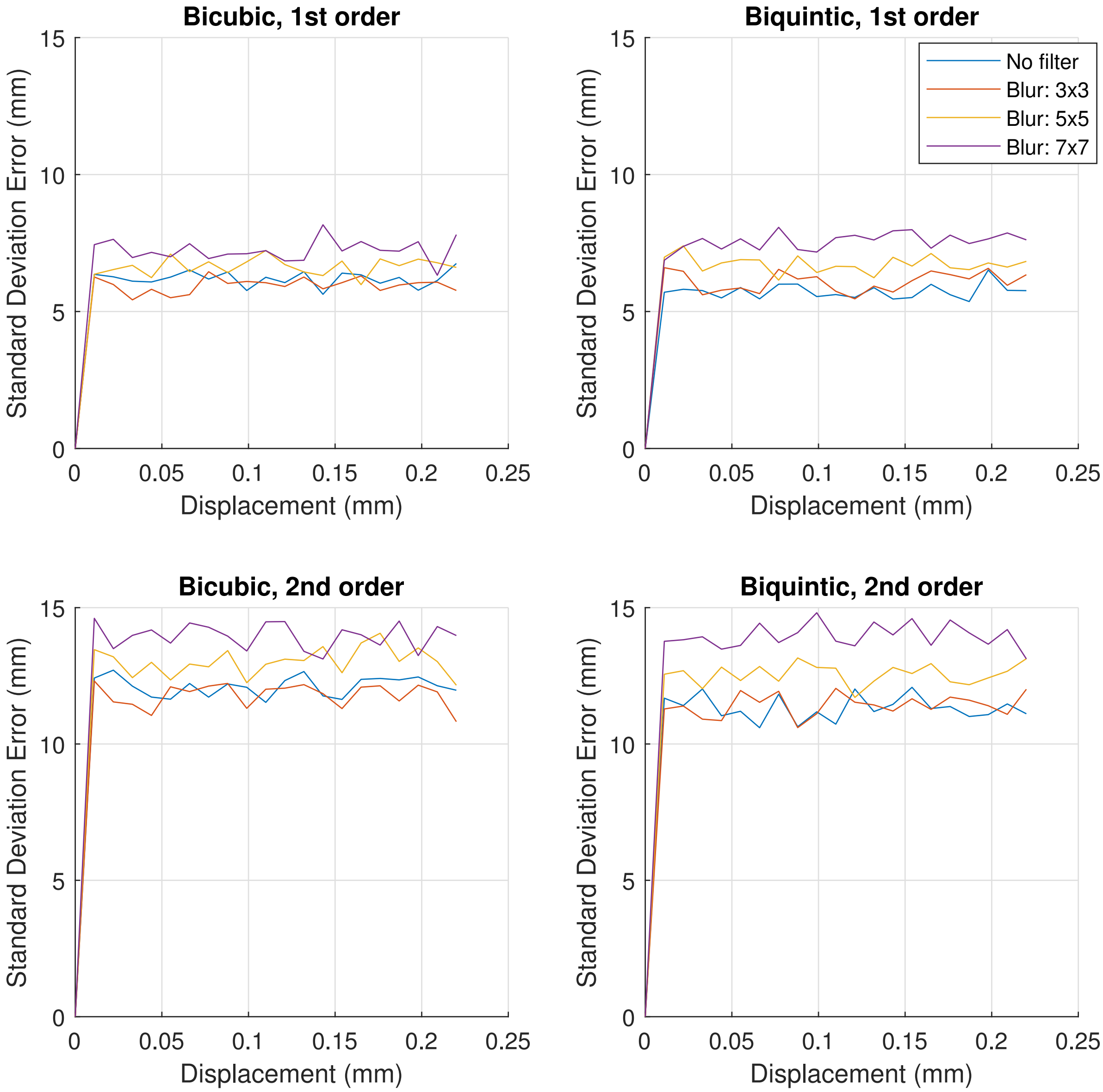


Figure B.24: Standard deviation errors present in 3D analysis due to Gaussian blur (4% noise).

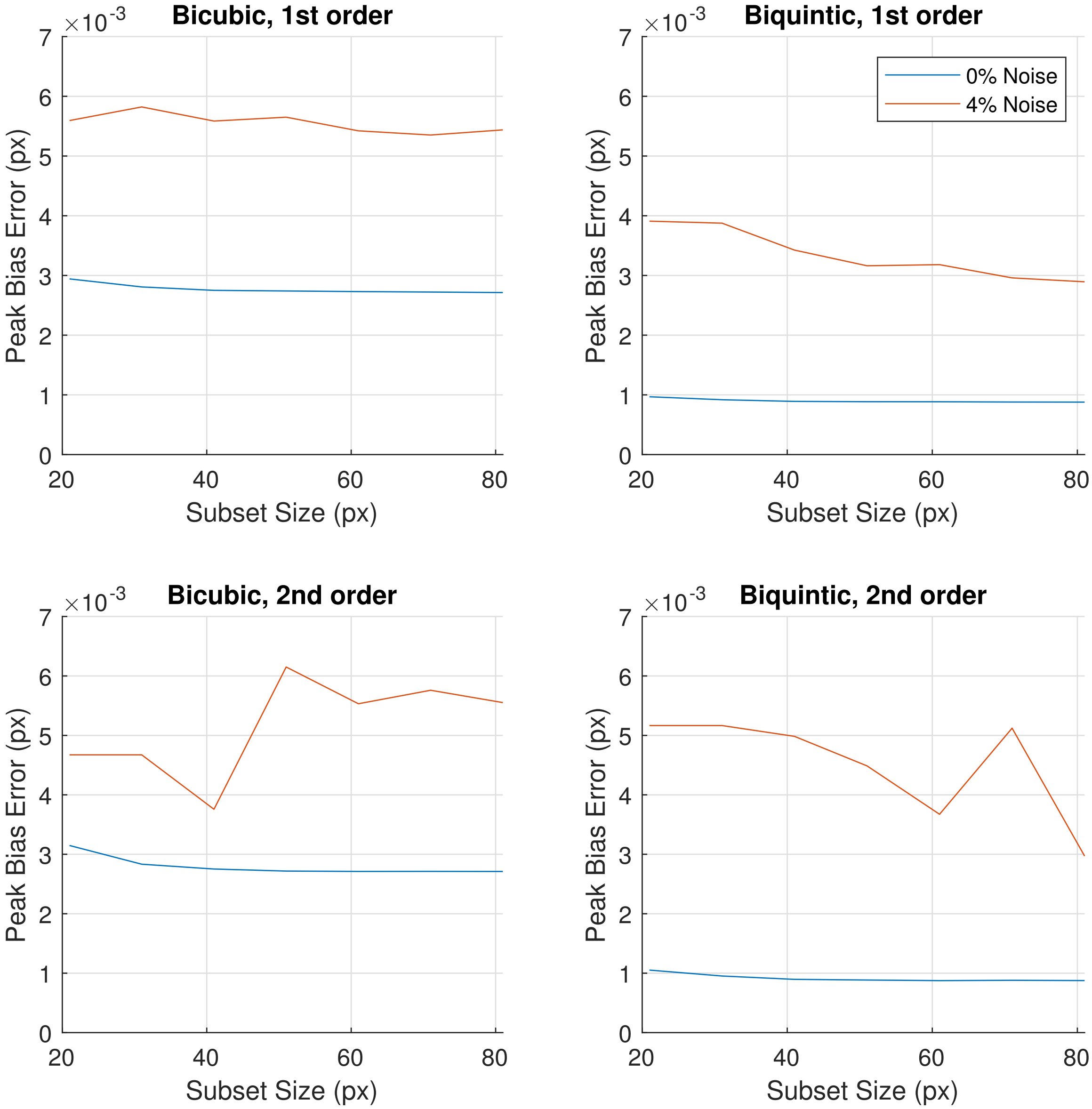


Figure B.25: Bias errors due to subset size in 2D.

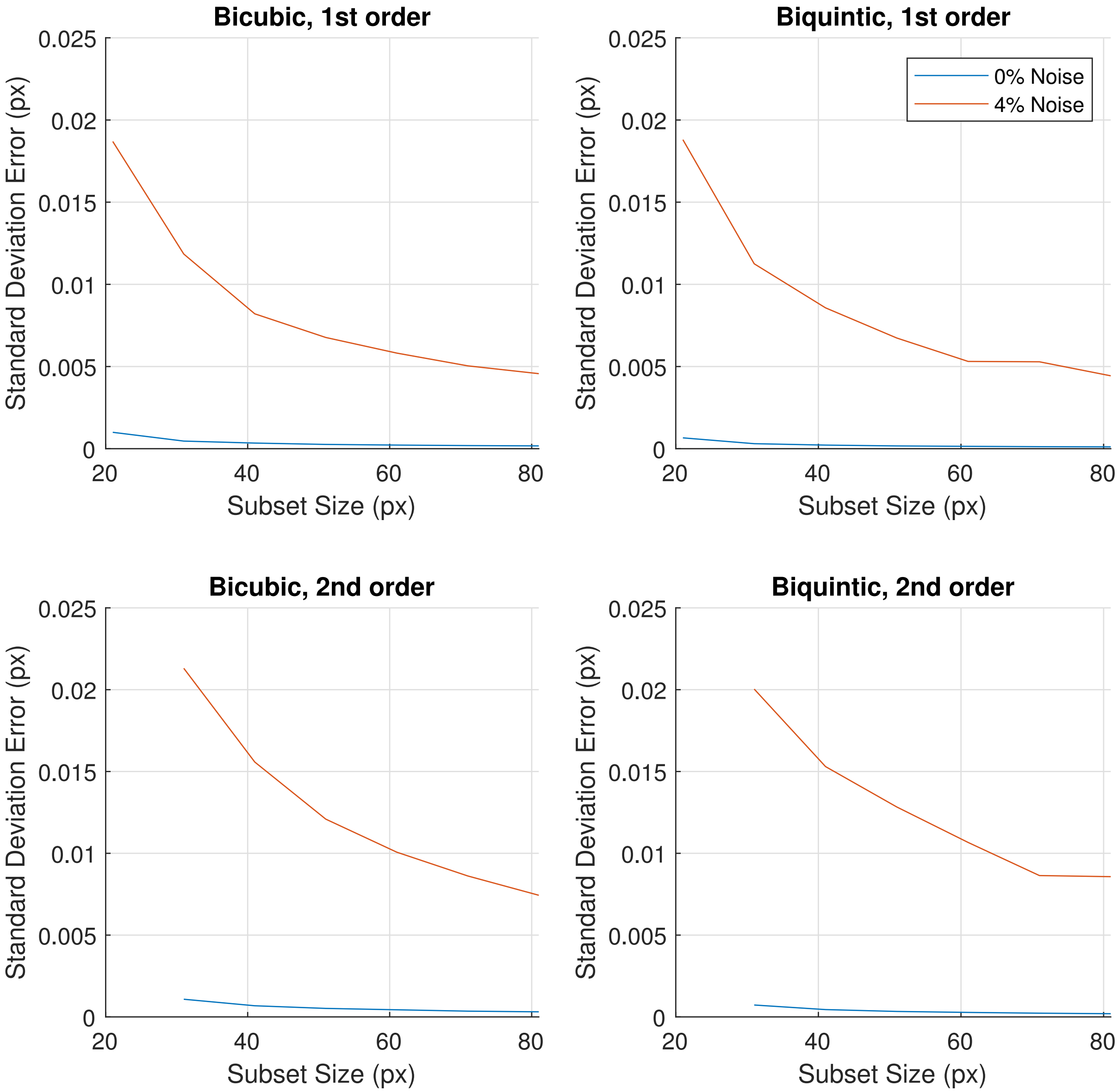


Figure B.26: Random errors due to subset size in 2D.

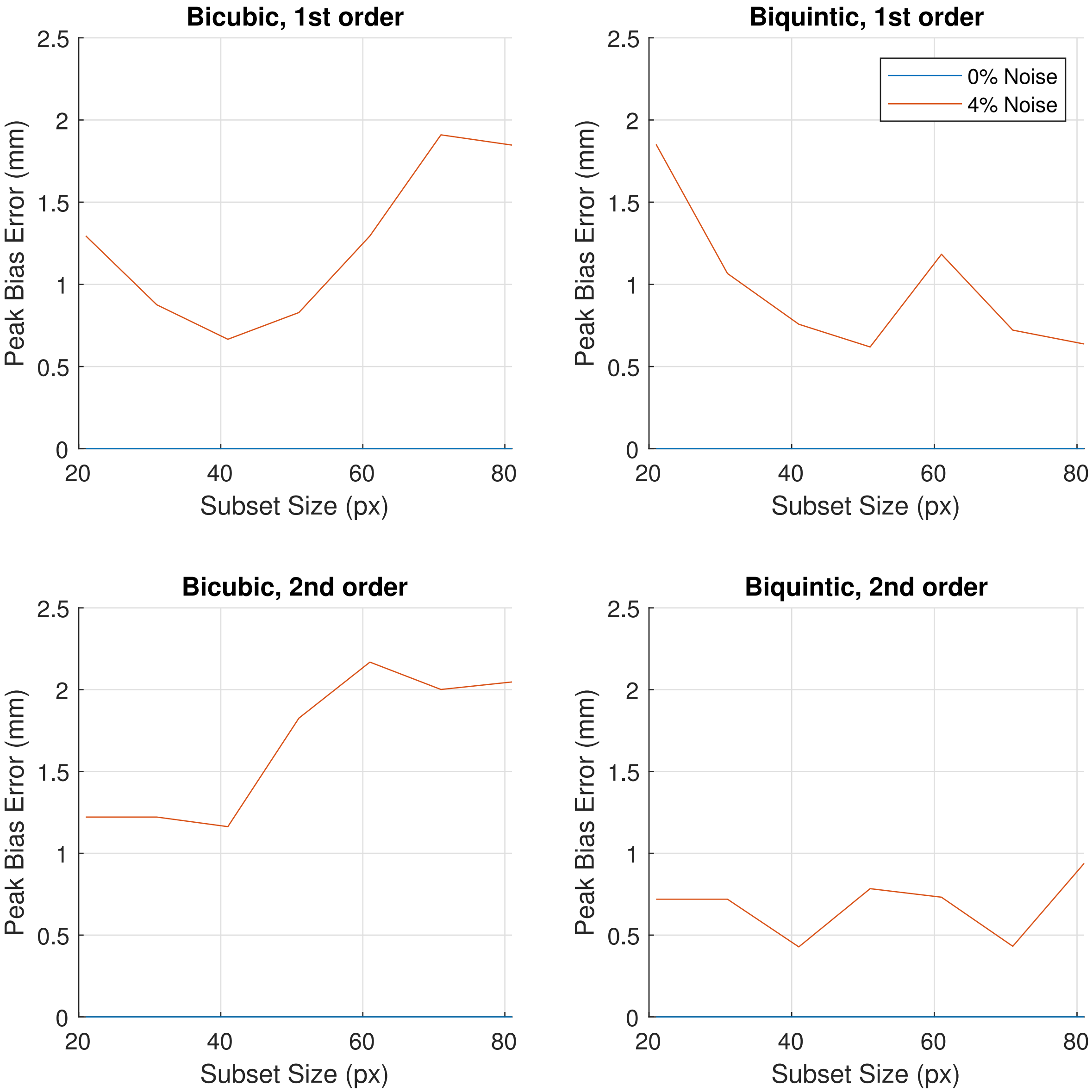


Figure B.27: Bias errors due to subset size in 3D.

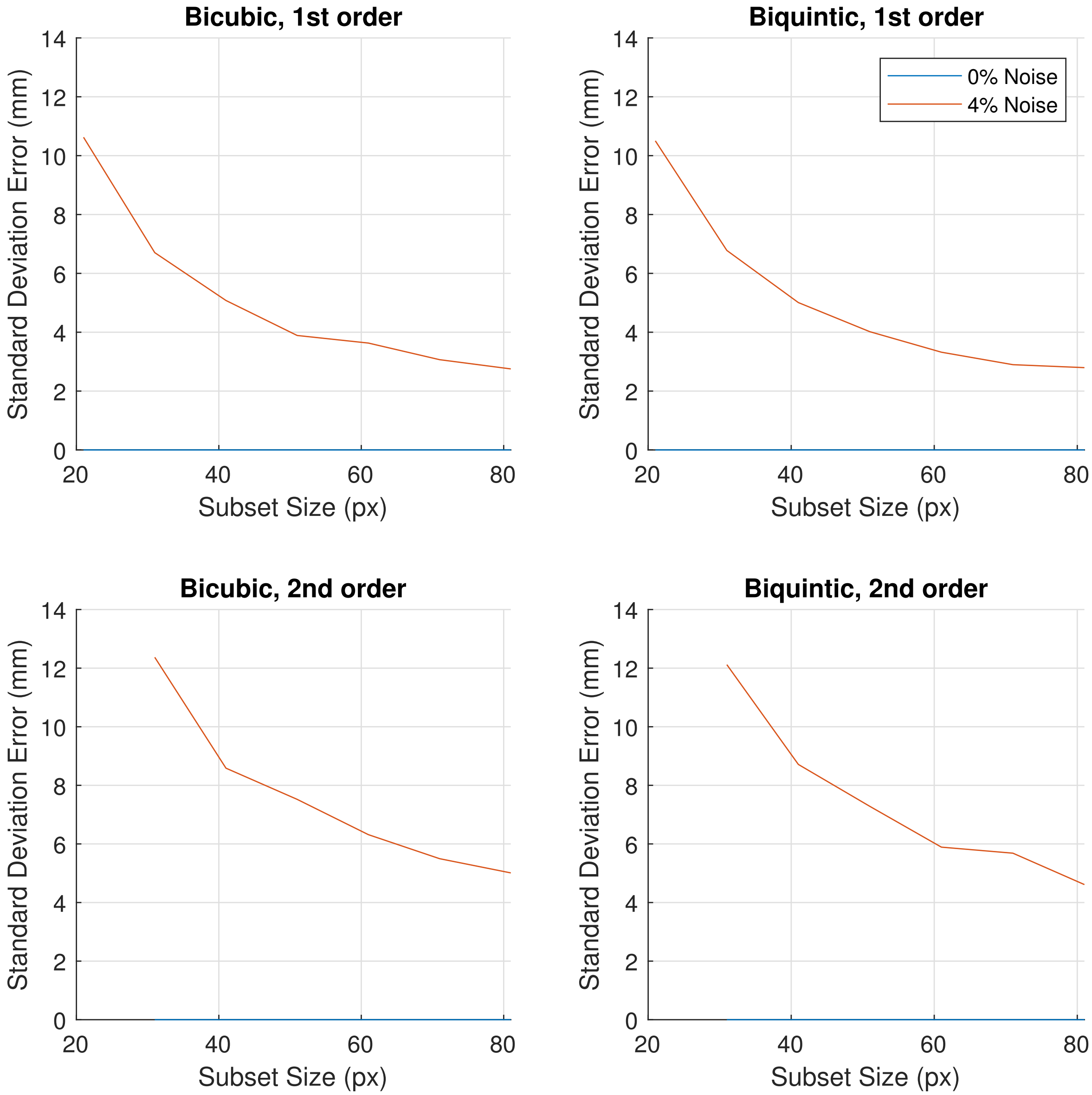


Figure B.28: Random errors due to subset size in 3D.

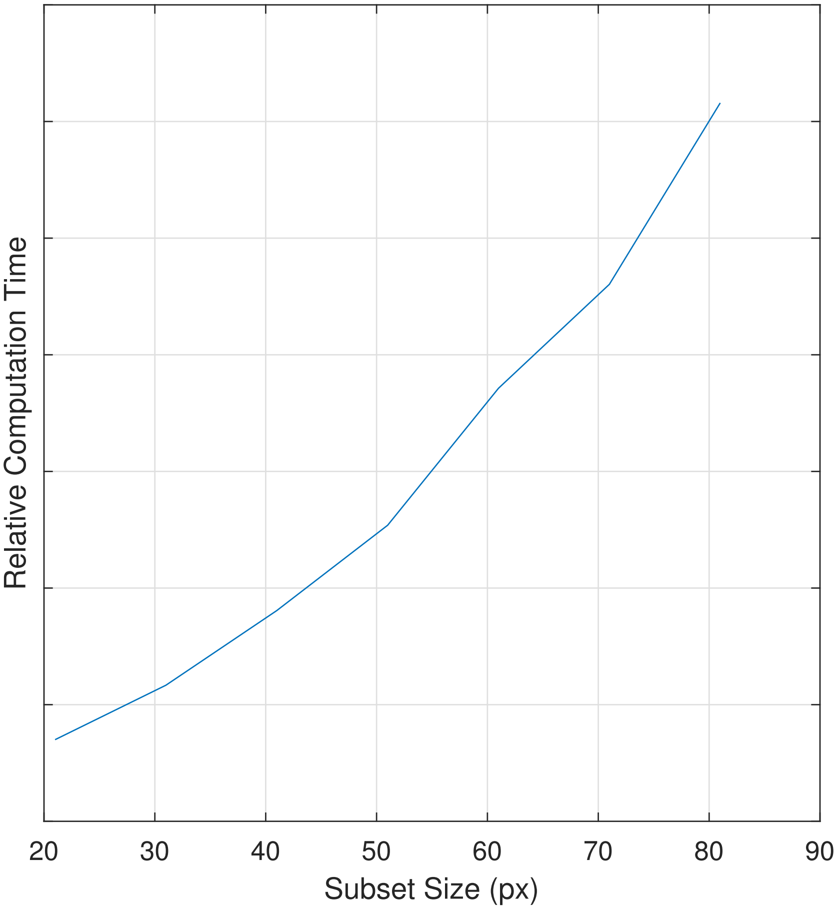


Figure B.29: Relative computational time of subsets.

B.2 Laboratory Results

B.2.1 Raw Data Graphs

DIC Displacement Time Series (Middle Point Selected)

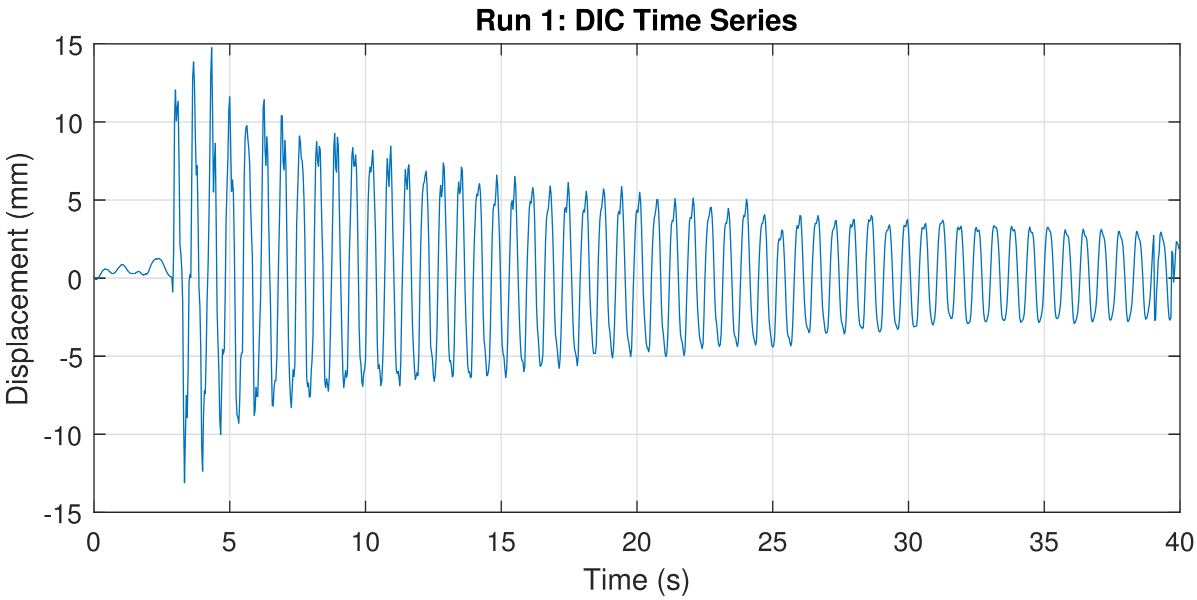


Figure B.30: DIC displacement time series from Run 1.

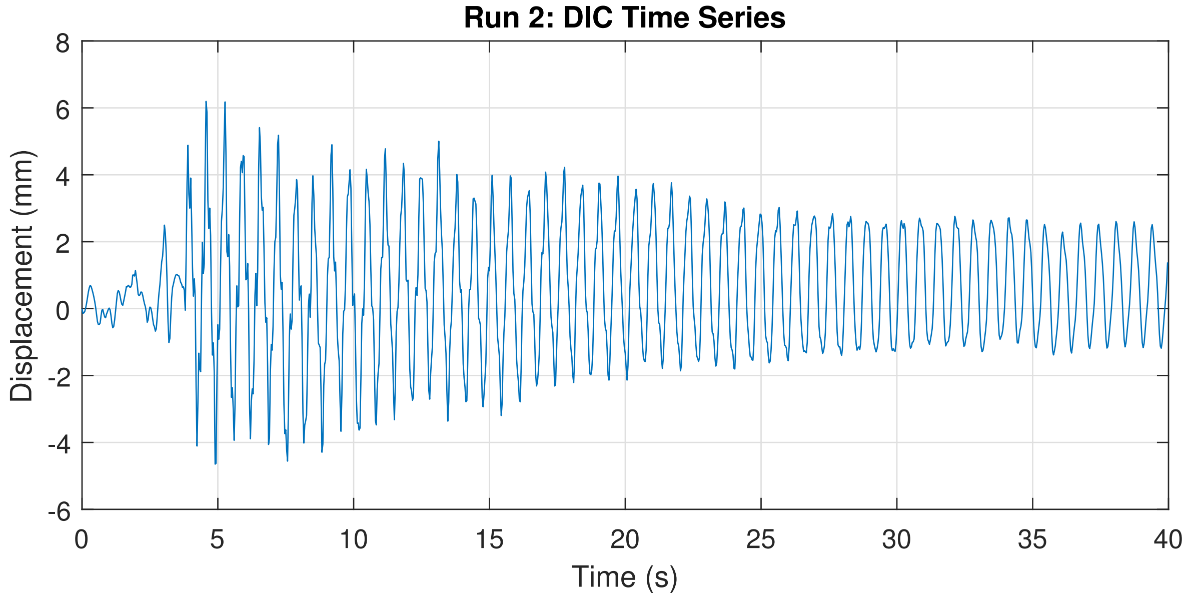


Figure B.31: DIC displacement time series from Run 2.

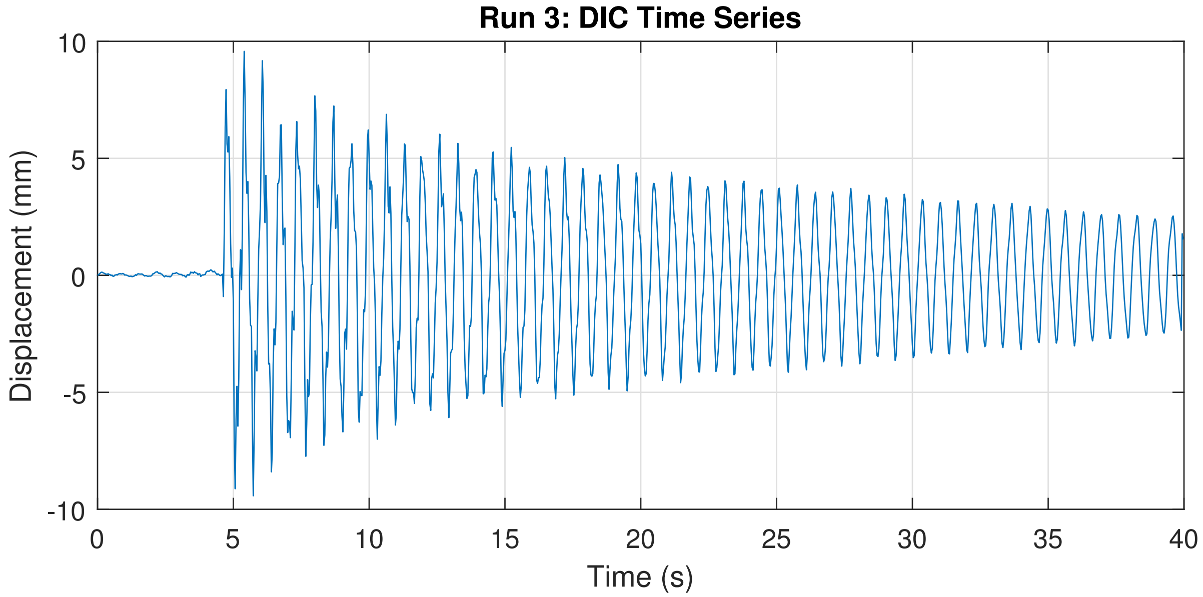


Figure B.32: DIC displacement time series from Run 3.

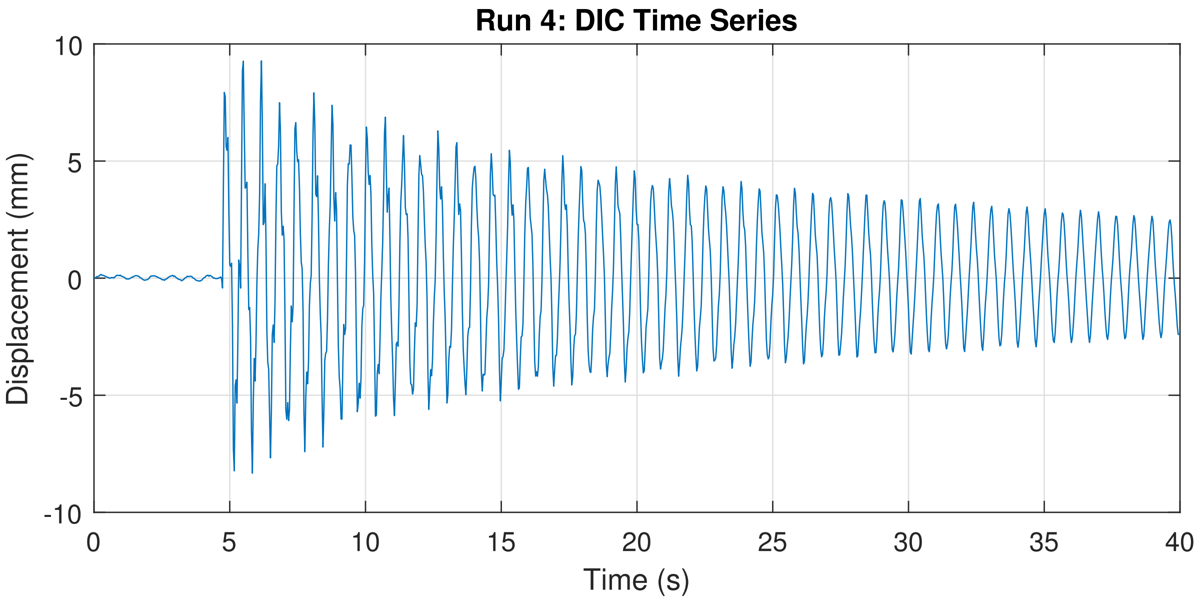


Figure B.33: DIC displacement time series from Run 4.

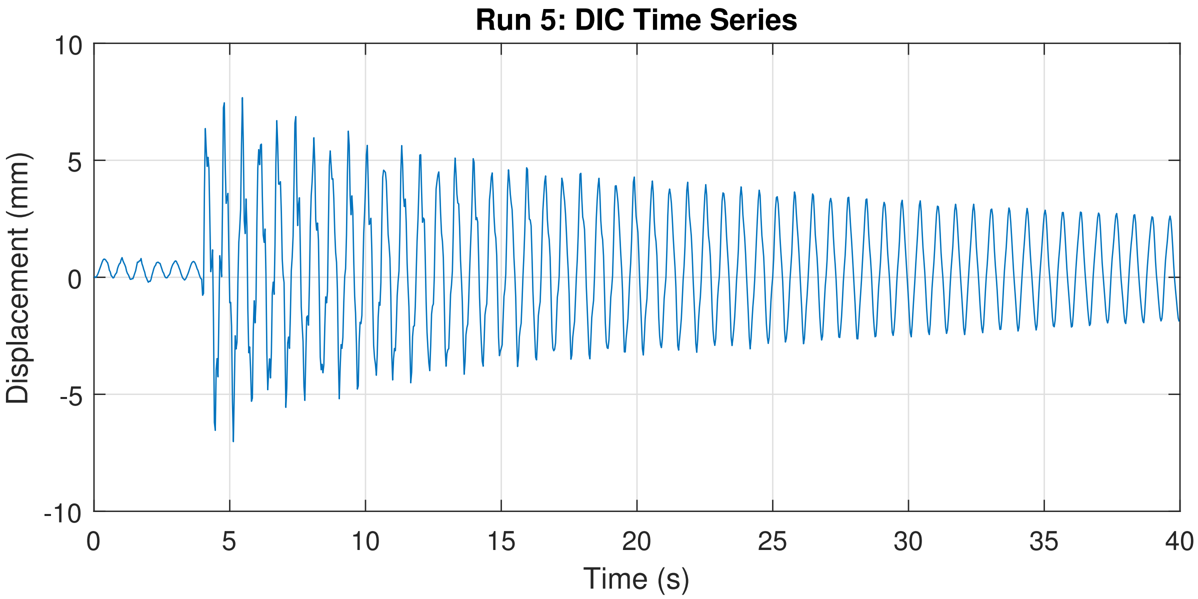


Figure B.34: DIC displacement time series from Run 5.

LDV Velocity Time Series

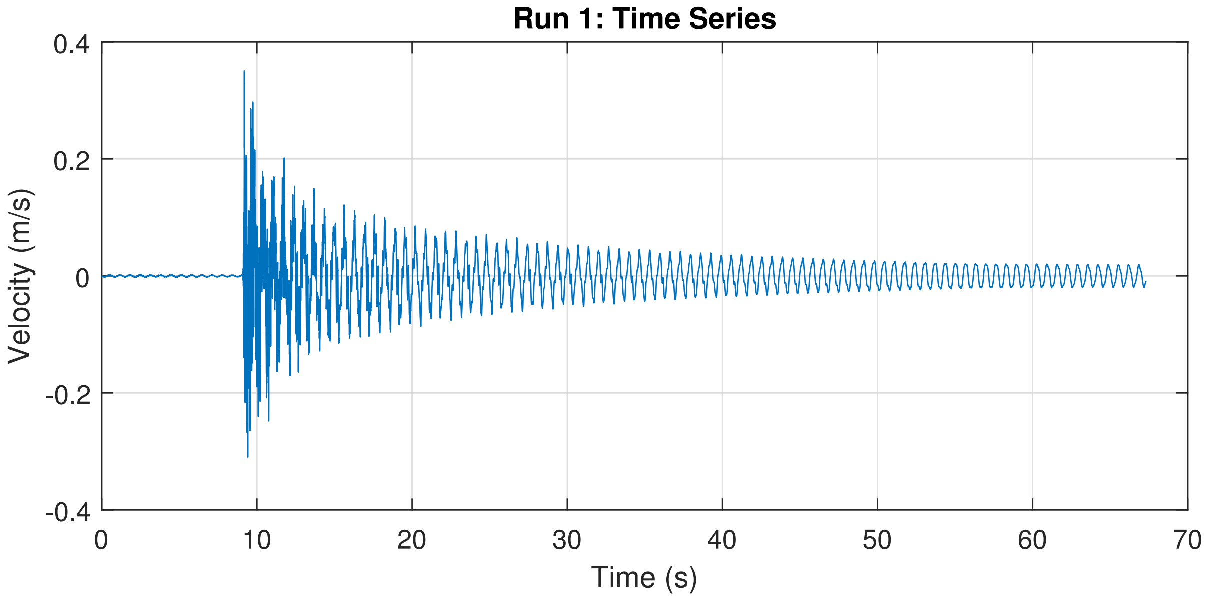


Figure B.35: LDV velocity time series from Run 1.

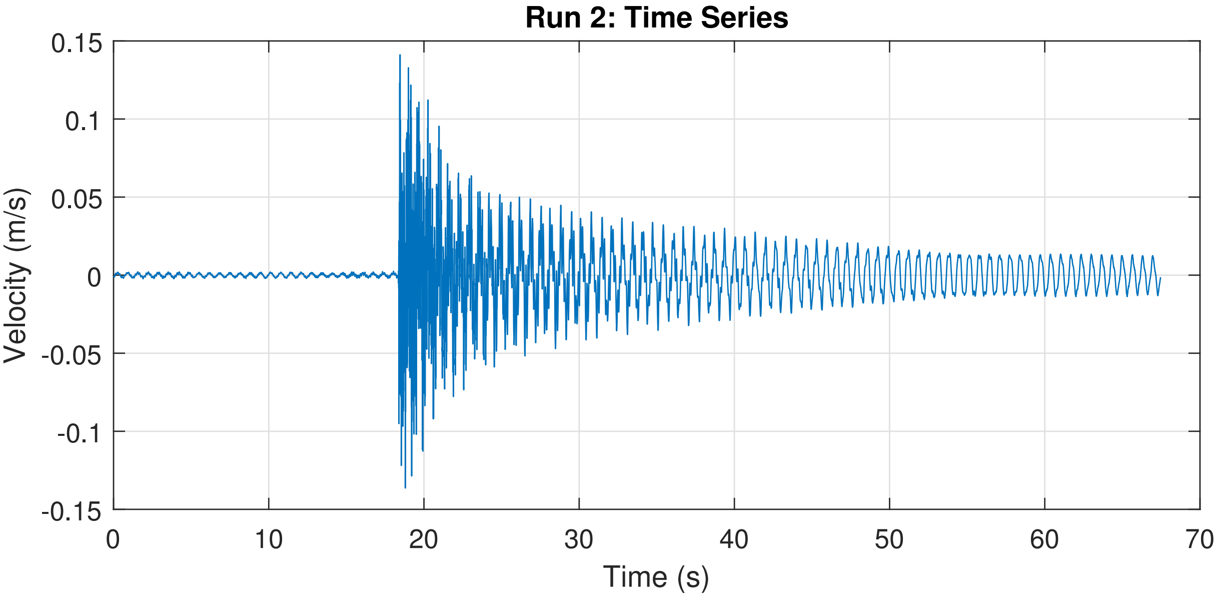


Figure B.36: LDV velocity time series from Run 2.

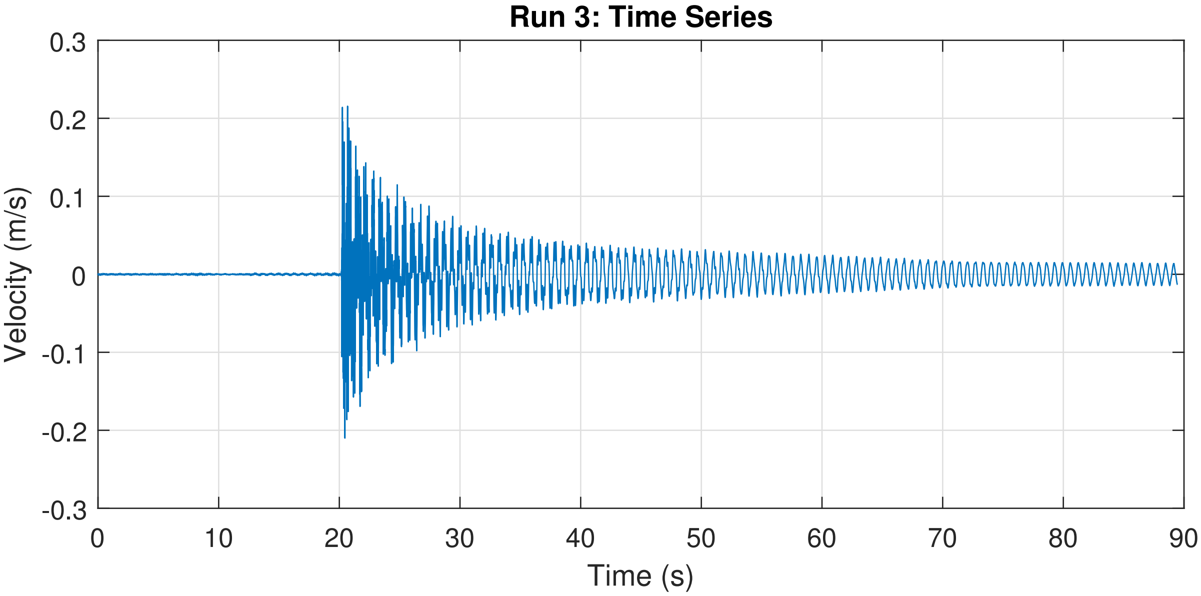


Figure B.37: LDV velocity time series from Run 3.

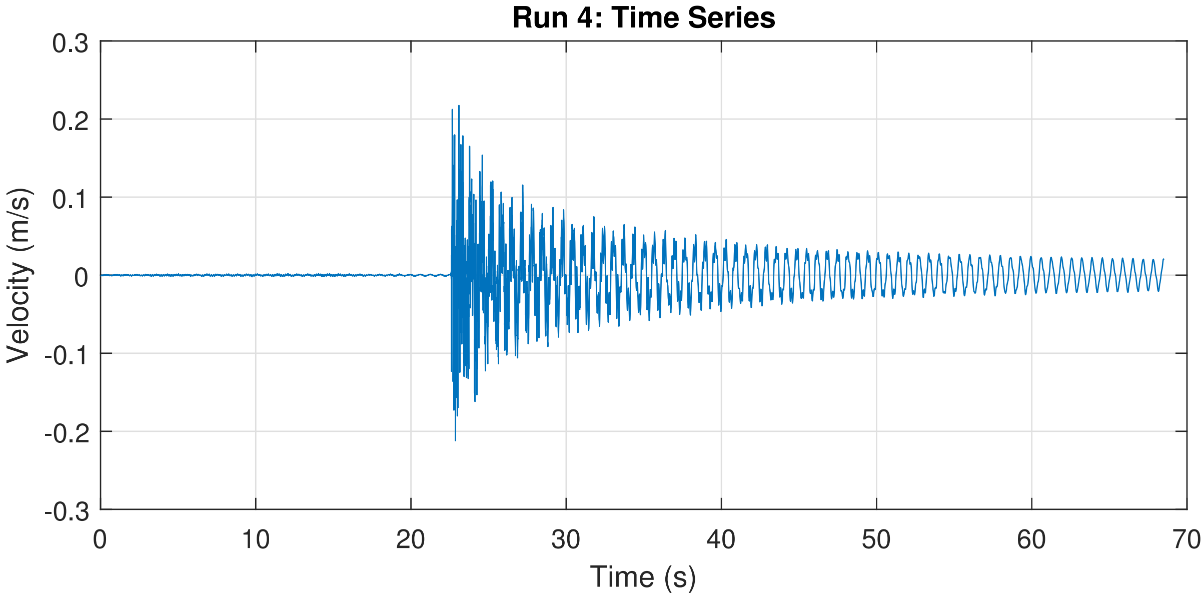


Figure B.38: LDV velocity time series from Run 4.

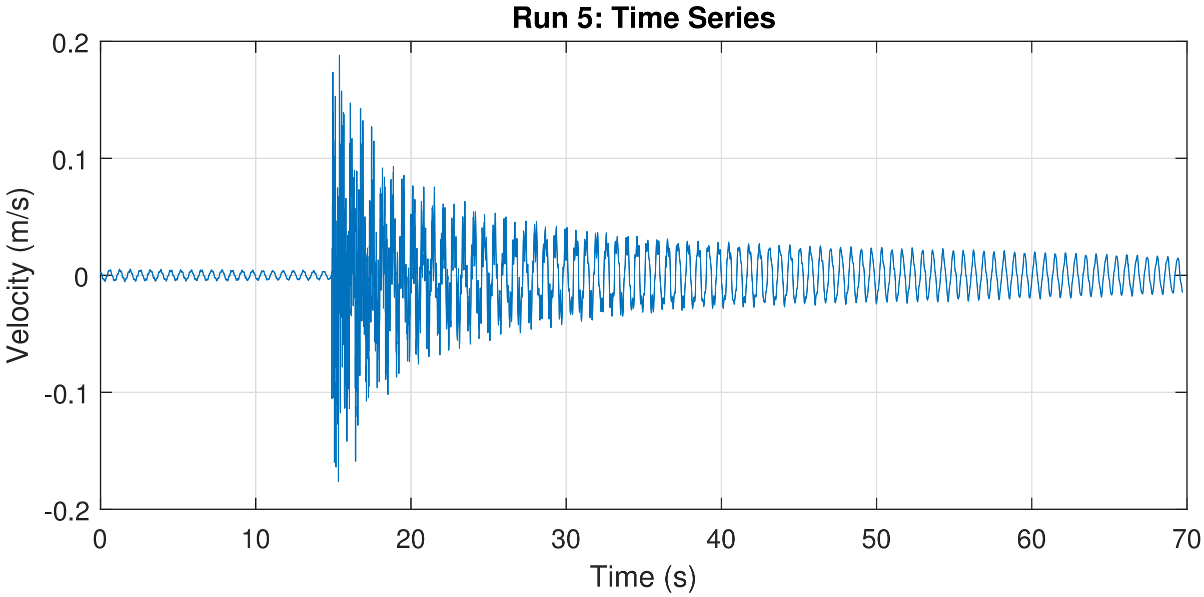


Figure B.39: LDV velocity time series from Run 5.

Impact Hammer Time Series

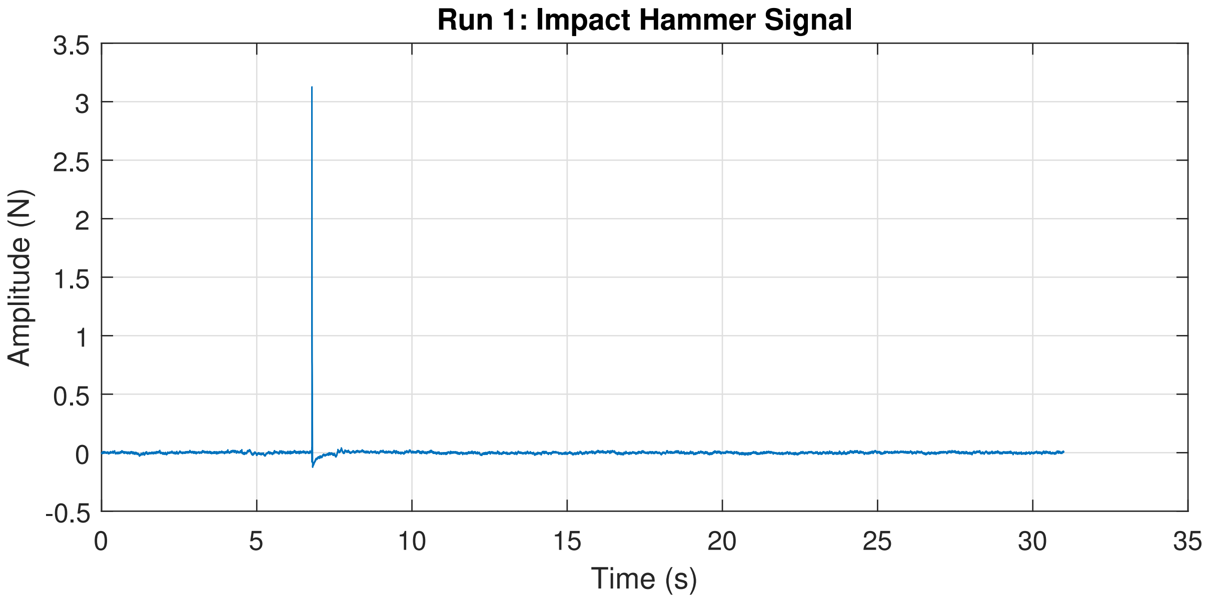


Figure B.40: Impact hammer force time series from Run 1.

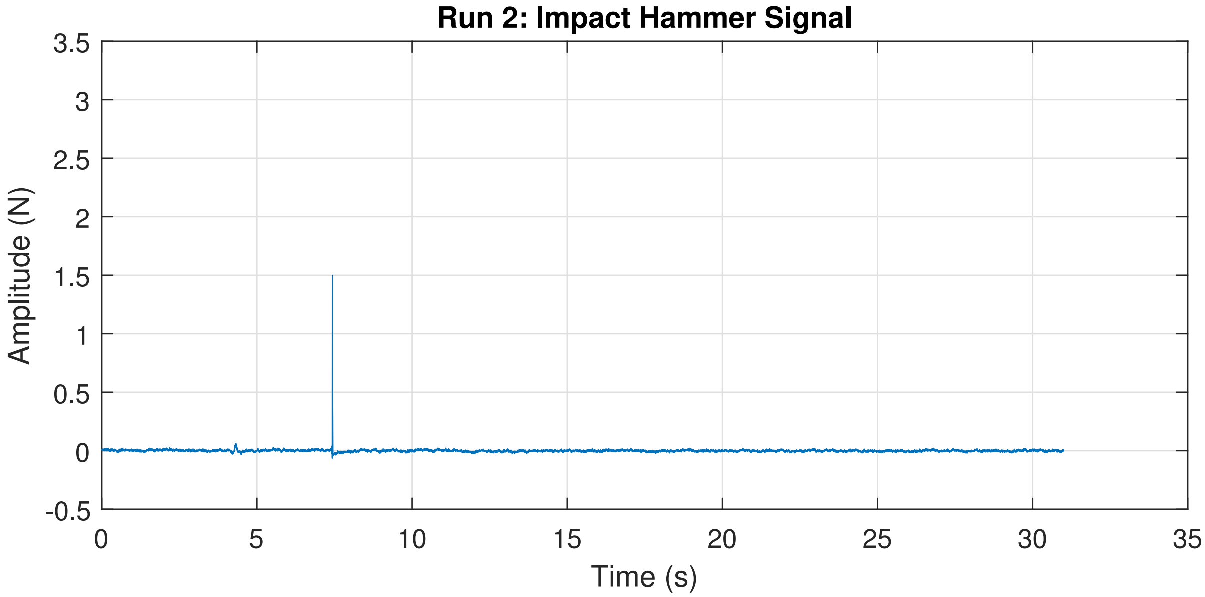


Figure B.41: Impact hammer force time series from Run 2.

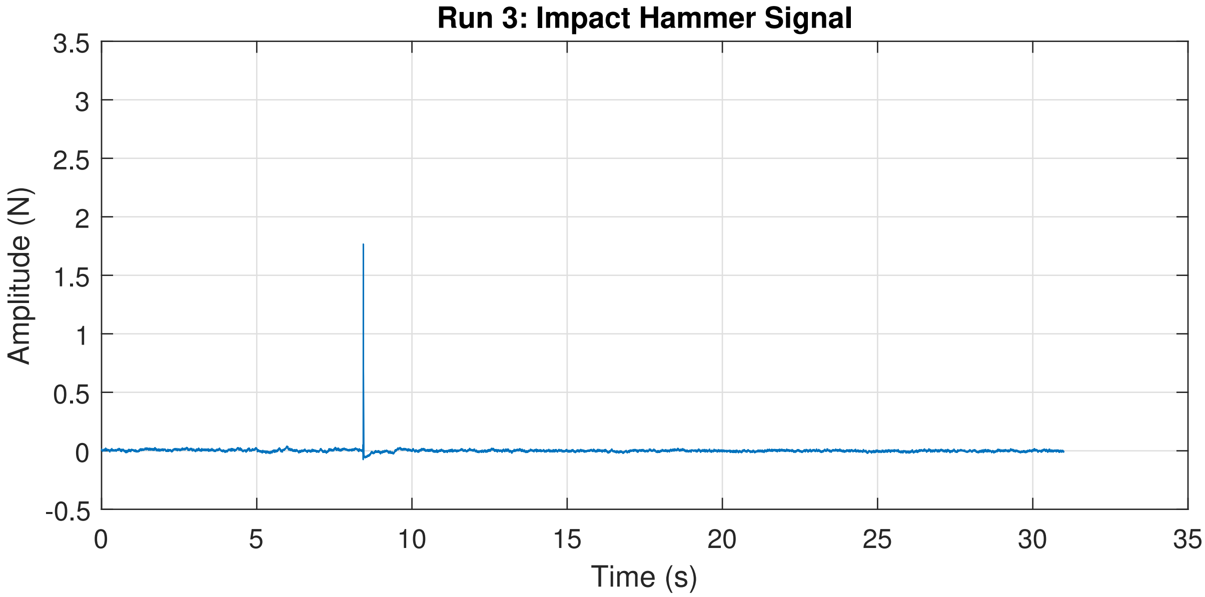


Figure B.42: Impact hammer force time series from Run 3.

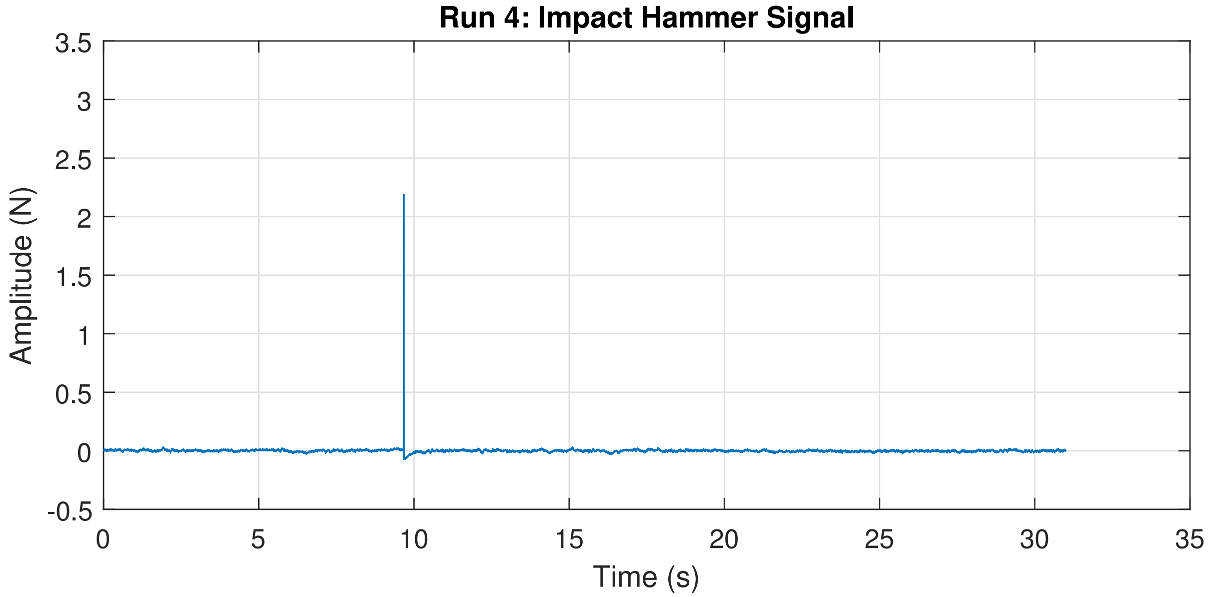


Figure B.43: Impact hammer force time series from Run 4.

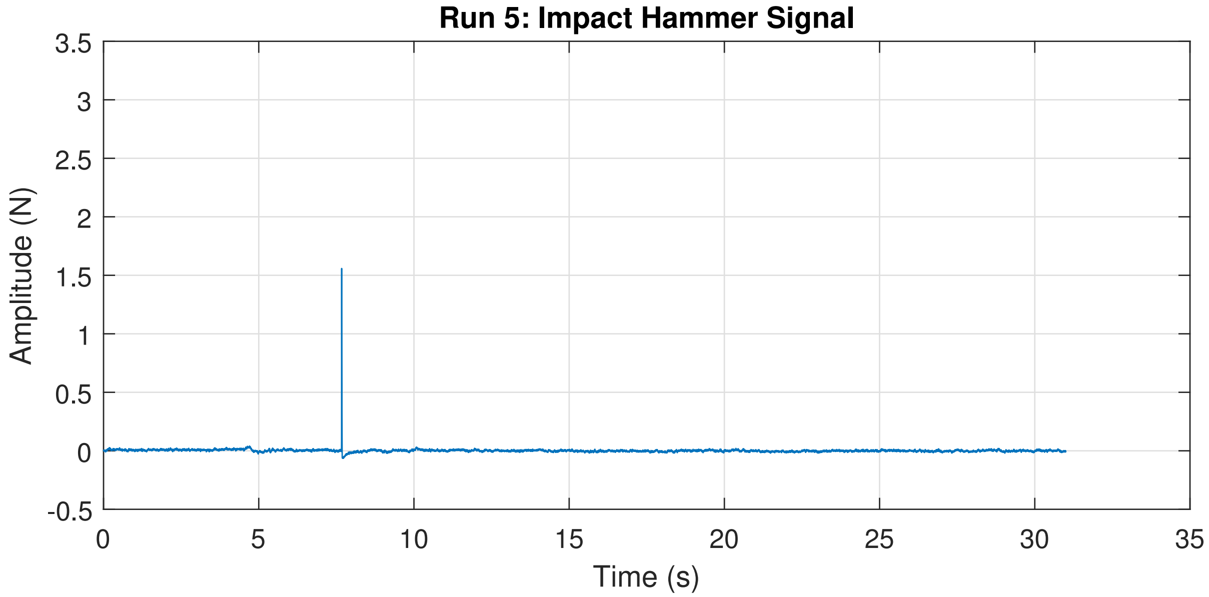


Figure B.44: Impact hammer force time series from Run 5.

B.2.2 Displacement Comparison Graphs

DIC (Mean Bias Uncorrected)

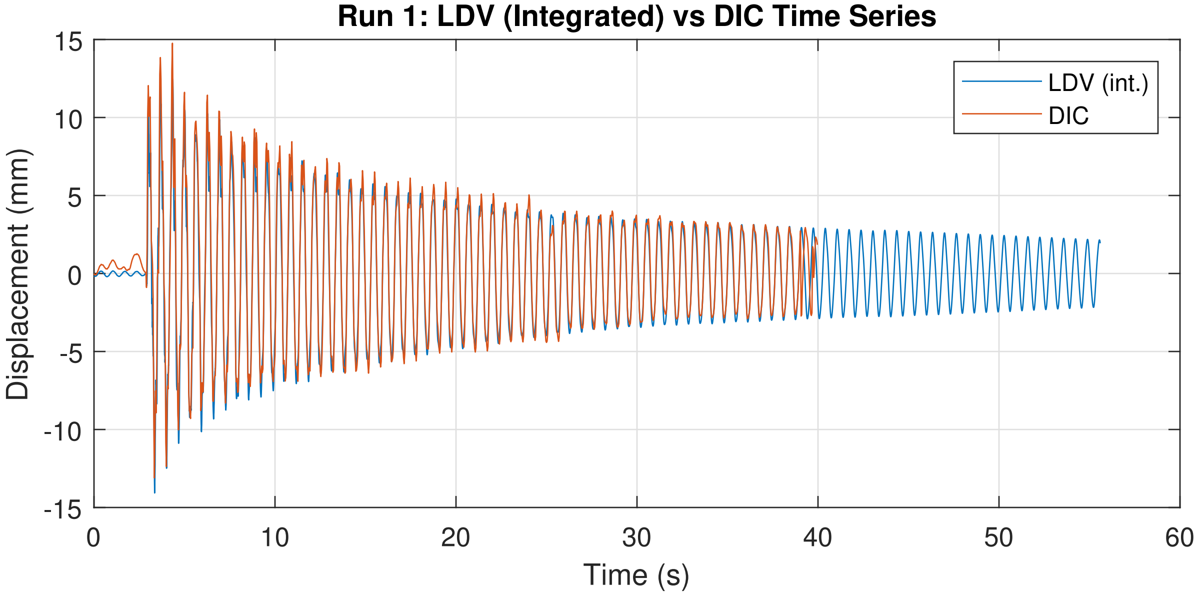


Figure B.45: LDV (integrated) vs. DIC displacement measurements from Run 1.

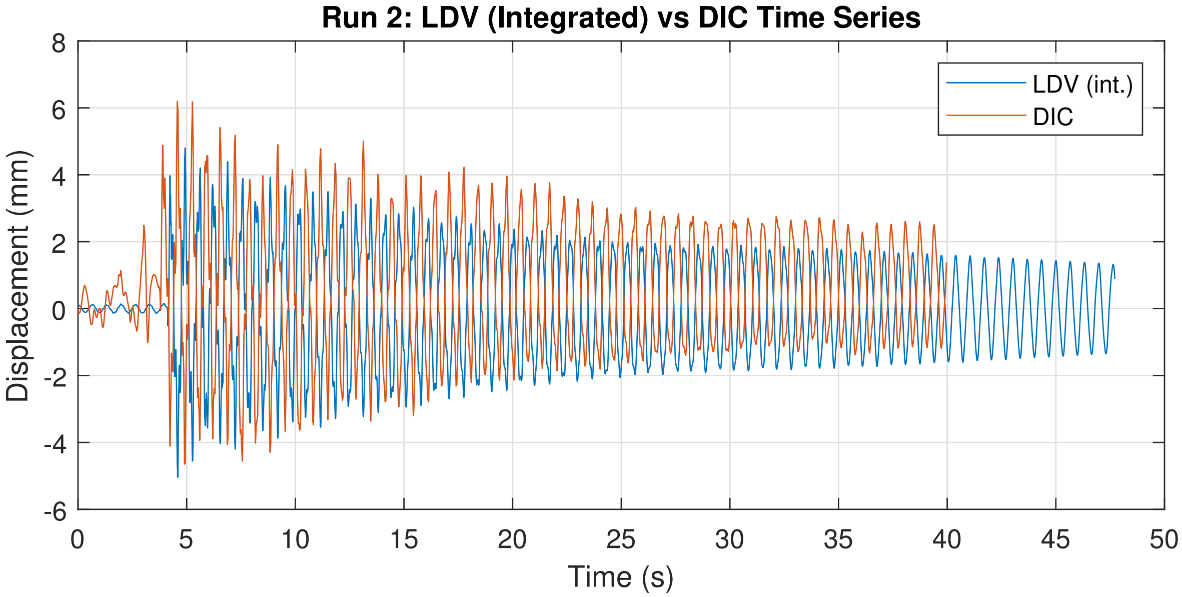


Figure B.46: LDV (integrated) vs. DIC displacement measurements from Run 2.

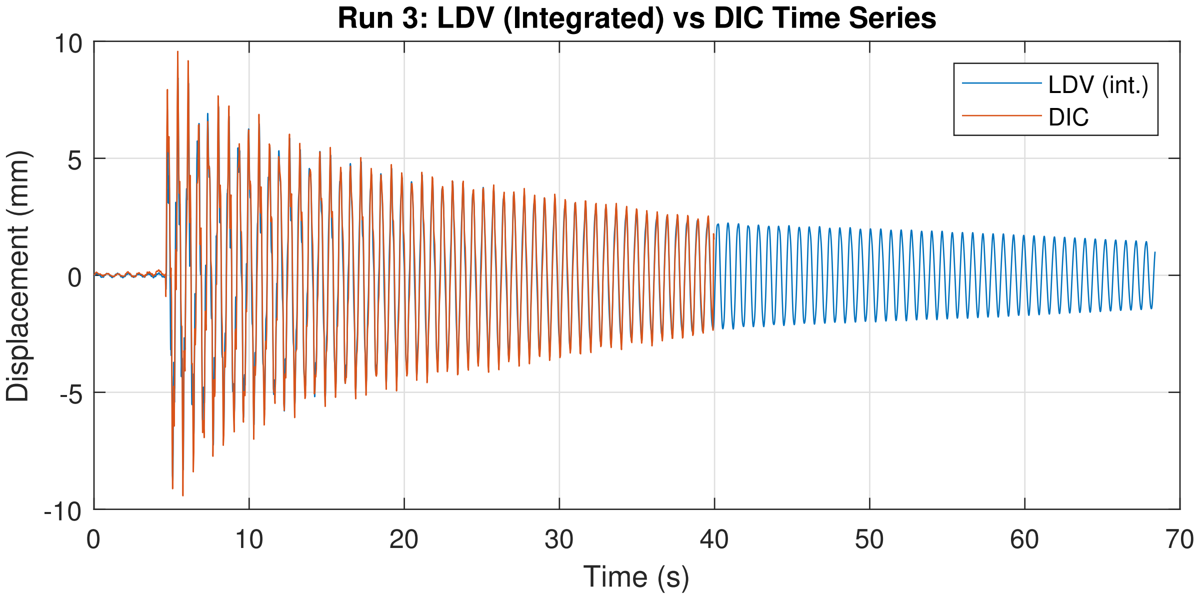


Figure B.47: LDV (integrated) vs. DIC displacement measurements from Run 3.

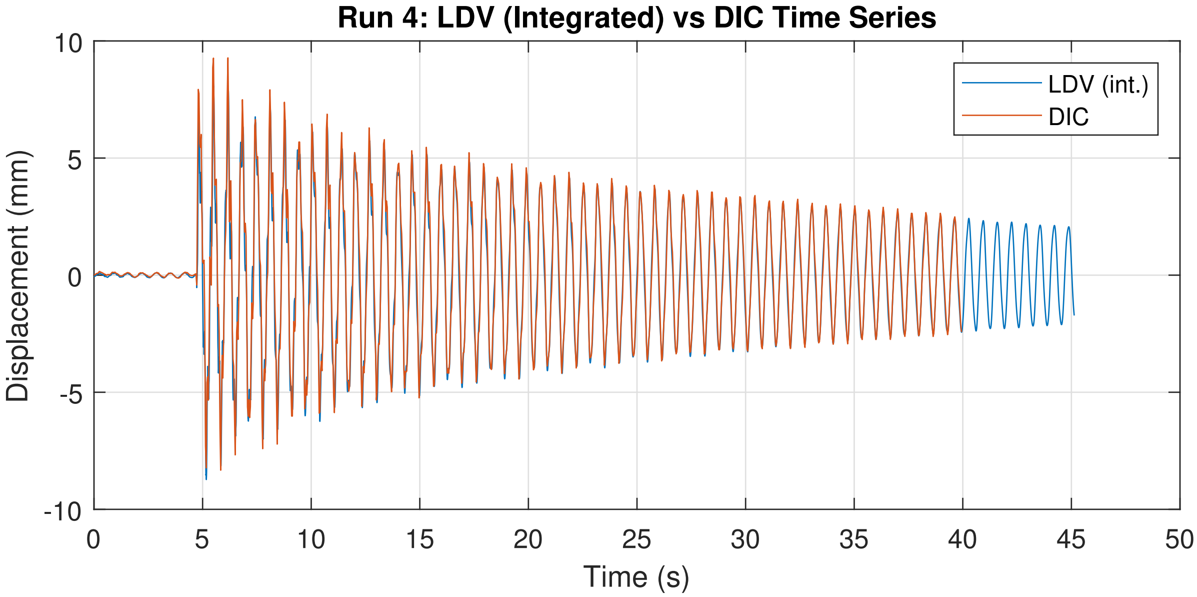


Figure B.48: LDV (integrated) vs. DIC displacement measurements from Run 4.

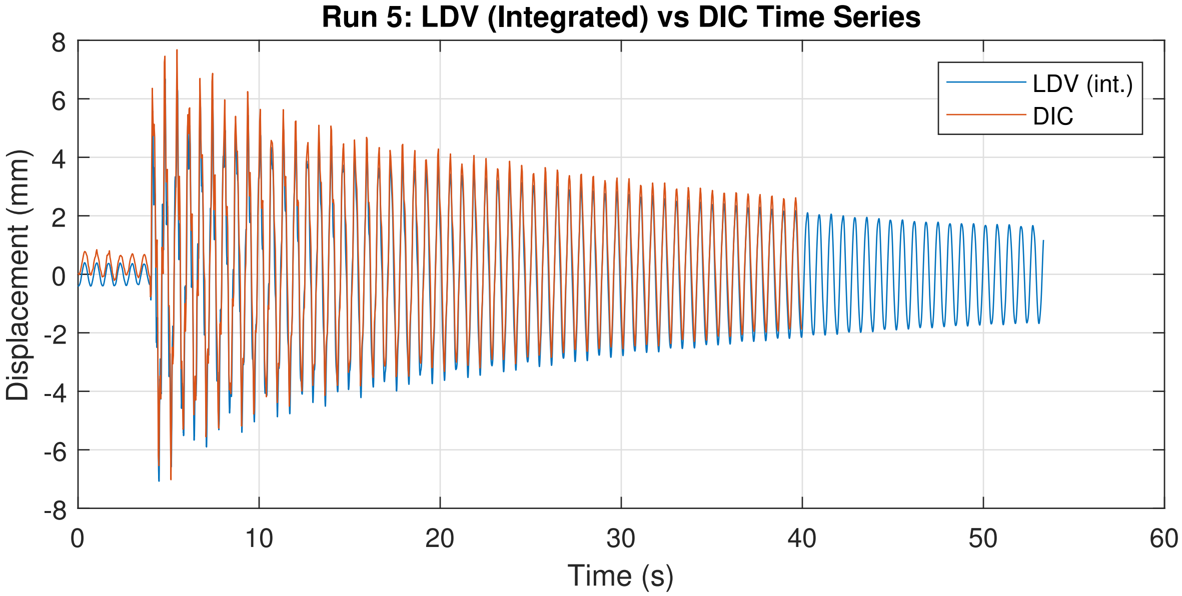


Figure B.49: LDV (integrated) vs. DIC displacement measurements from Run 5.

DIC (Mean Bias Corrected)

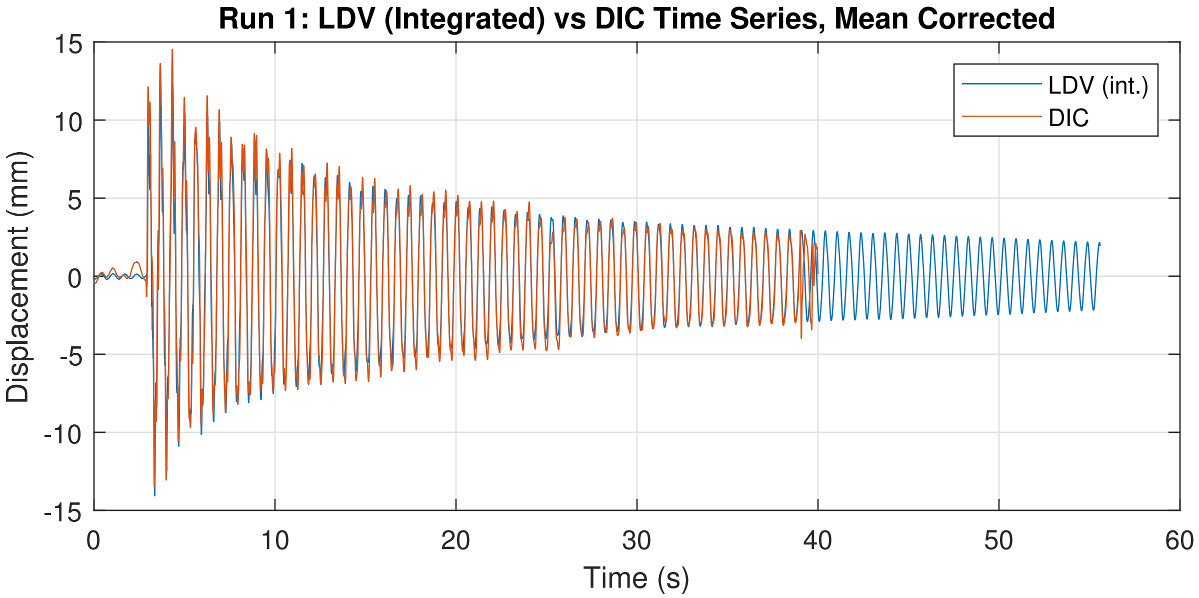


Figure B.50: LDV (integrated) vs. bias corrected DIC displacement measurements from Run 1.

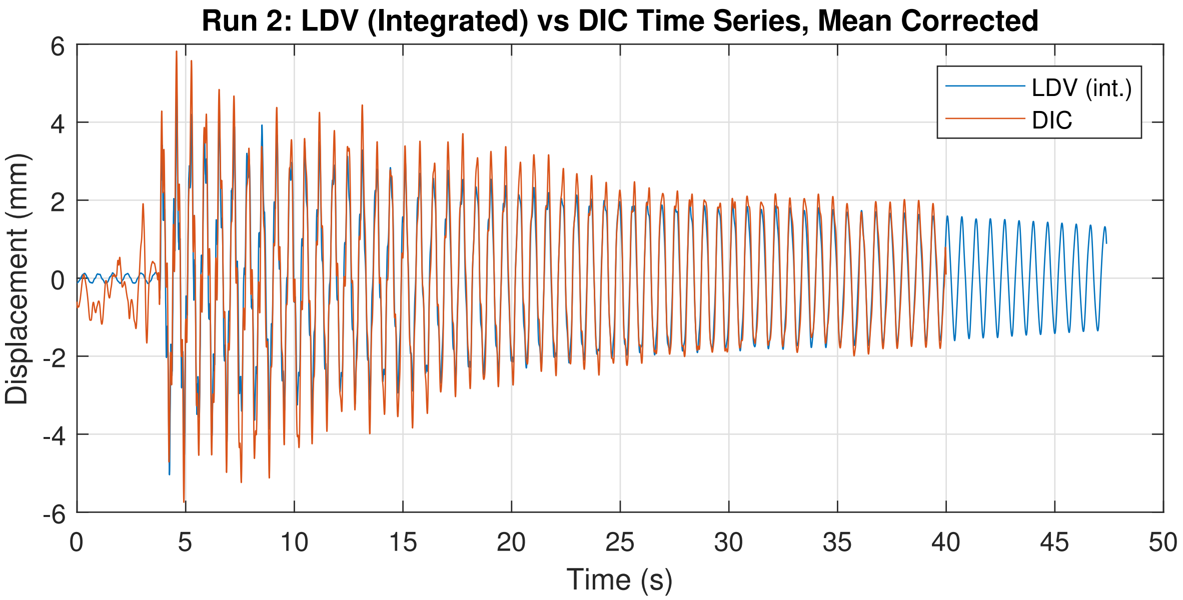


Figure B.51: LDV (integrated) vs. bias corrected DIC displacement measurements from Run 2.

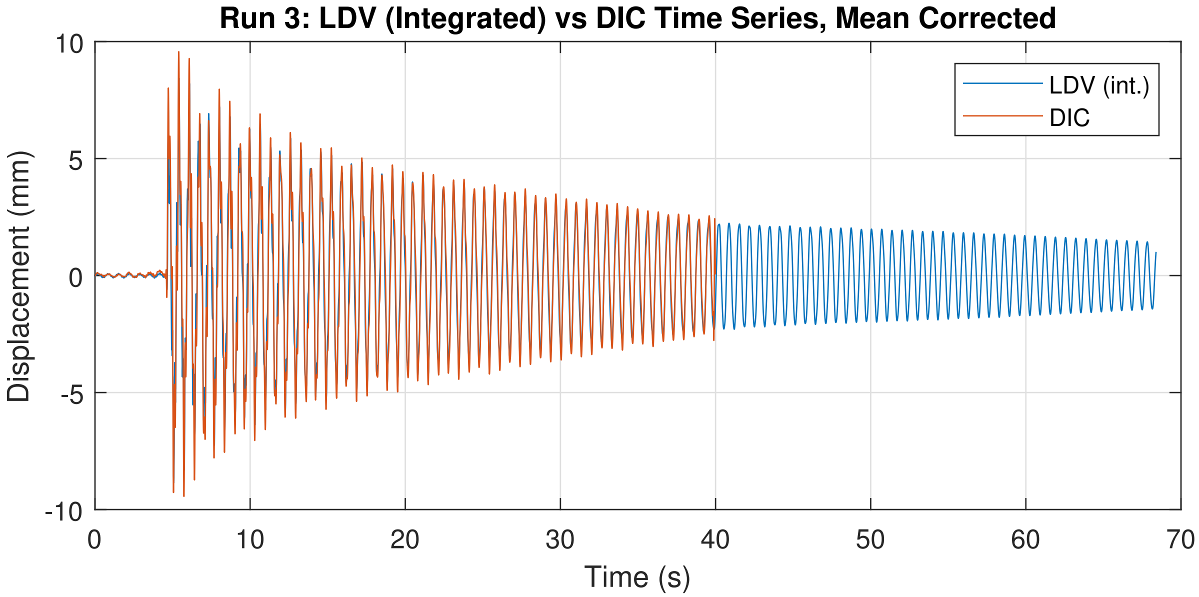


Figure B.52: LDV (integrated) vs. bias corrected DIC displacement measurements from Run 3.

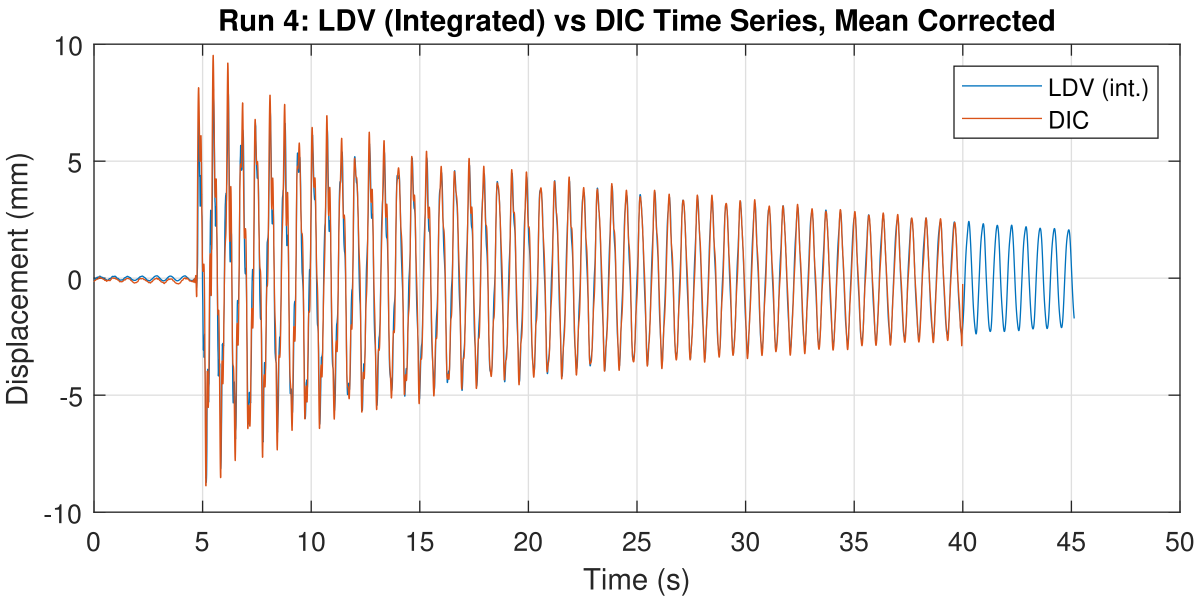


Figure B.53: LDV (integrated) vs. bias corrected DIC displacement measurements from Run 4.

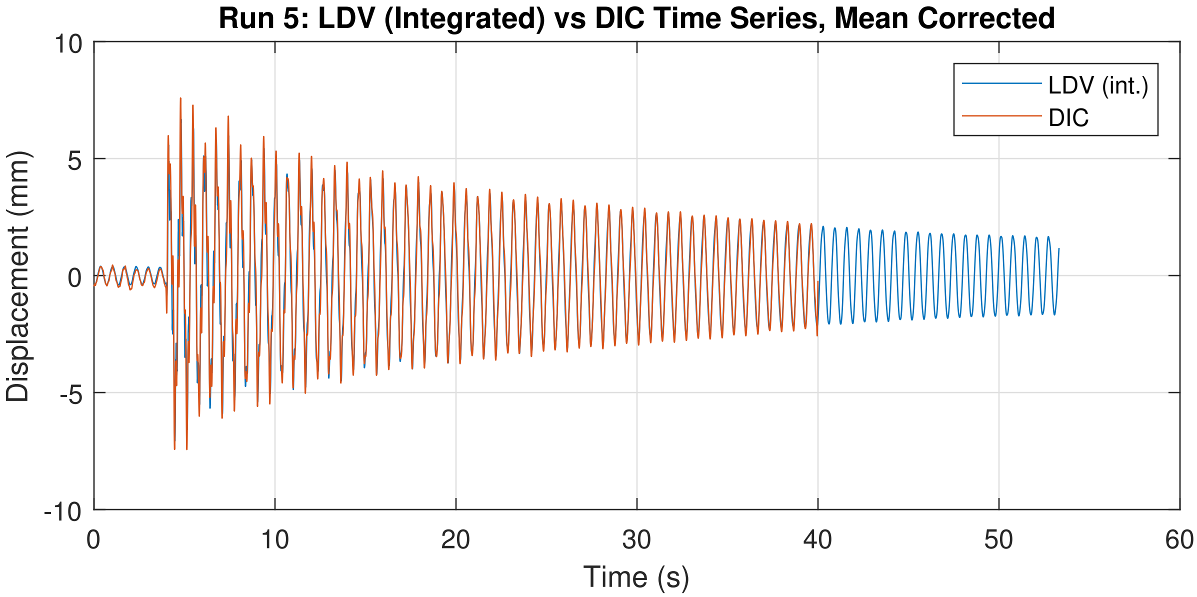


Figure B.54: LDV (integrated) vs. bias corrected DIC displacement measurements from Run 5.

B.2.3 DIC Stabilization Diagrams

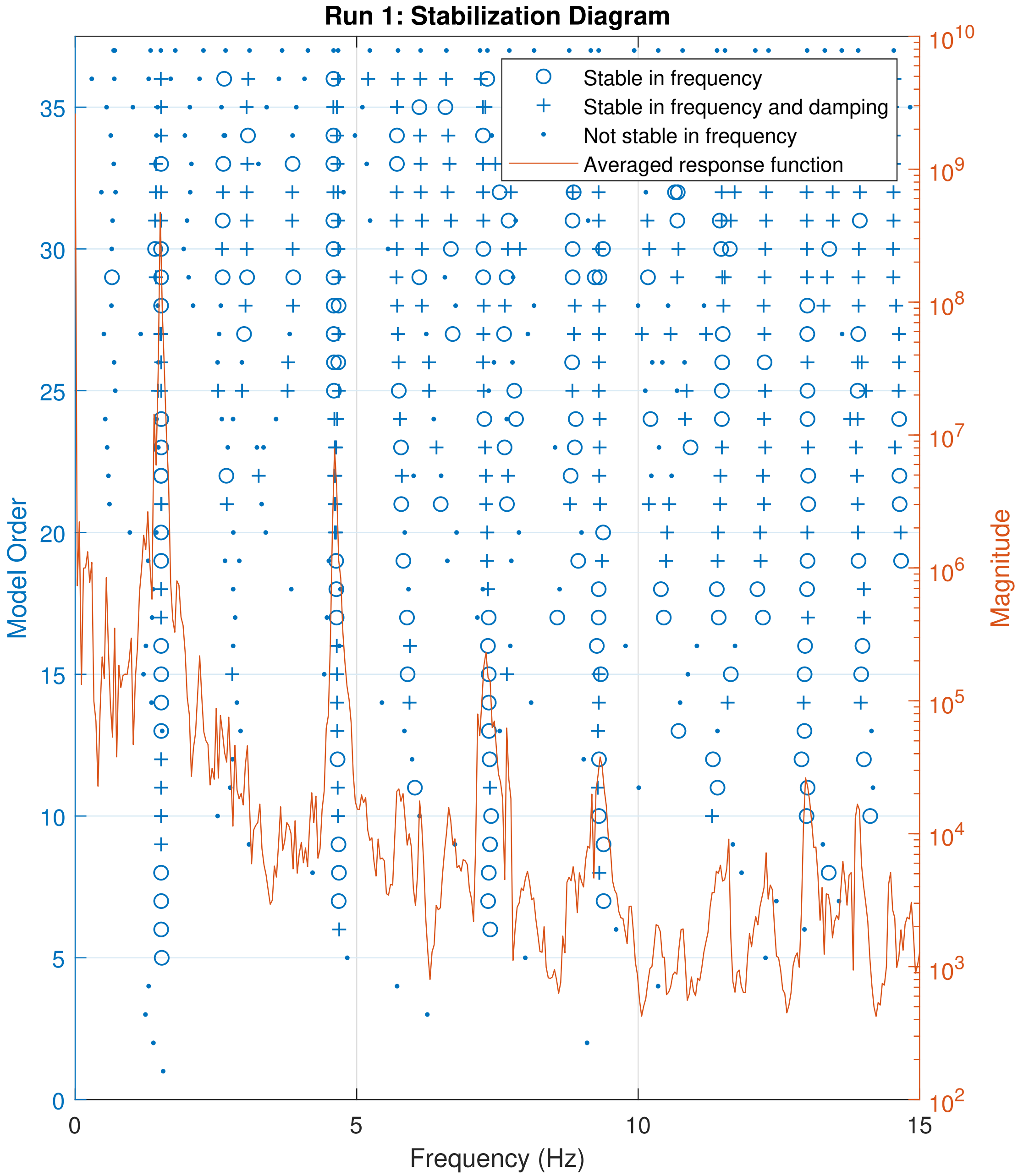


Figure B.55: Stabilization diagram from Run 1.

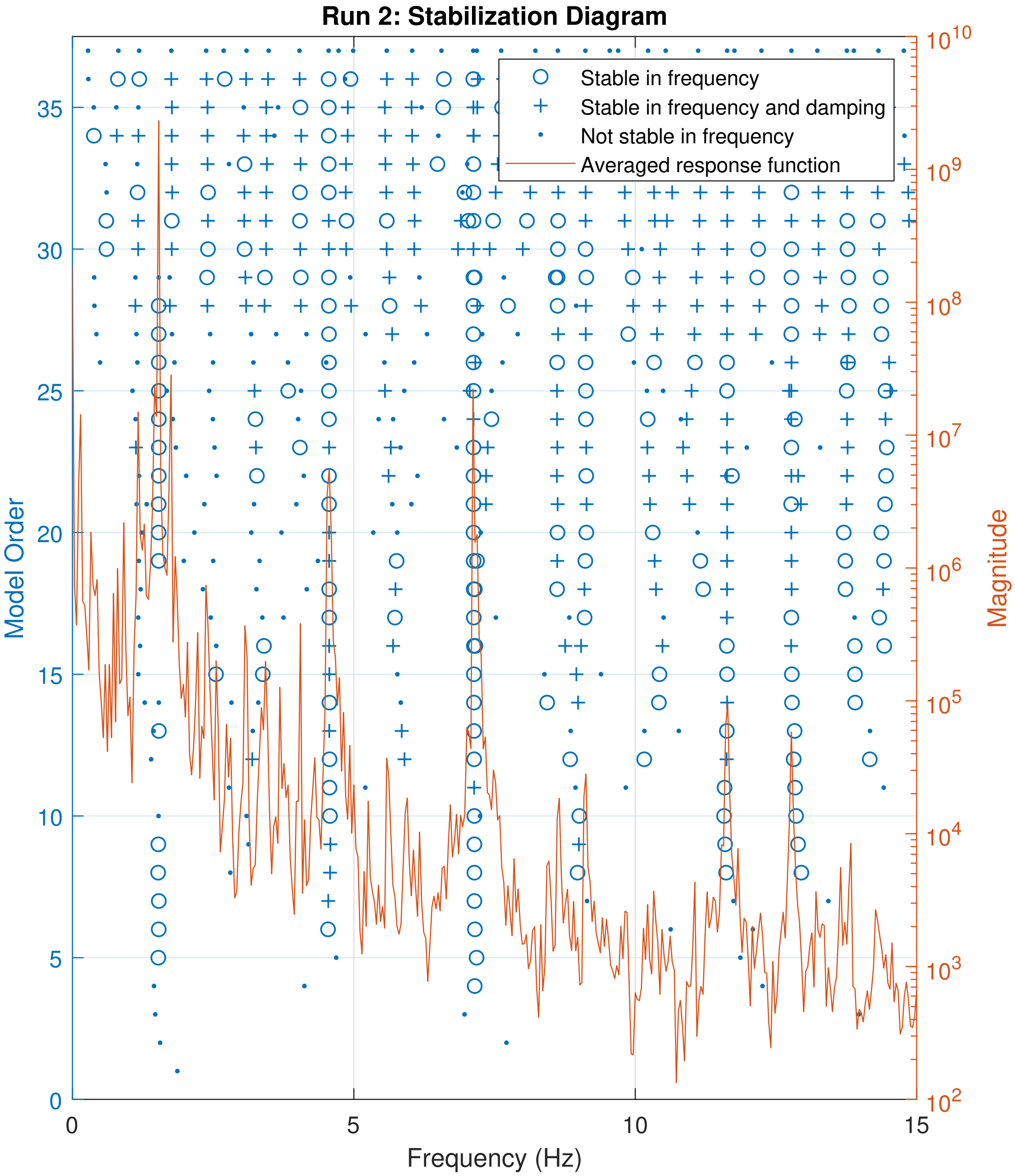


Figure B.56: Stabilization diagram from Run 2.

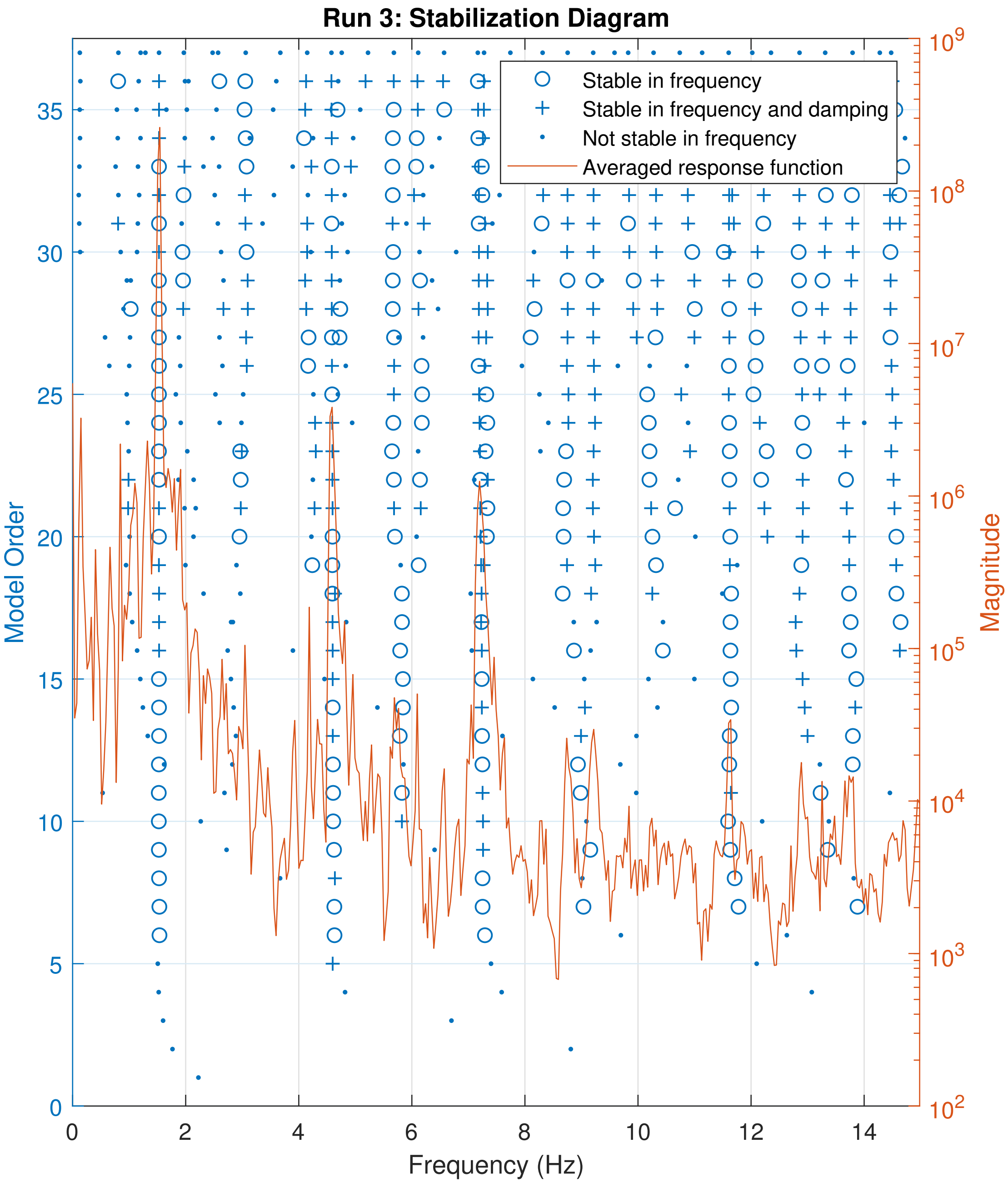


Figure B.57: Stabilization diagram from Run 3.

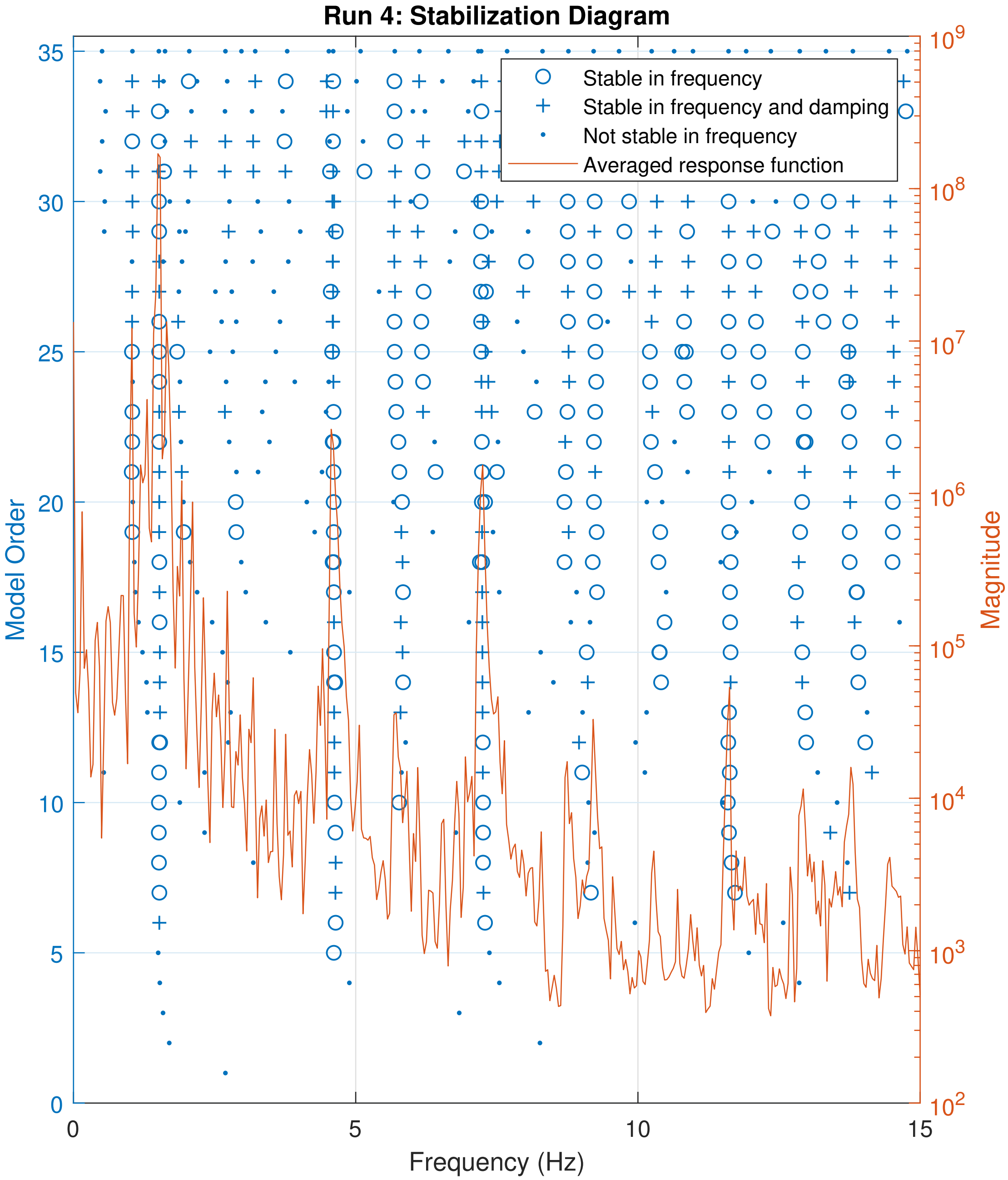


Figure B.58: Stabilization diagram from Run 4.

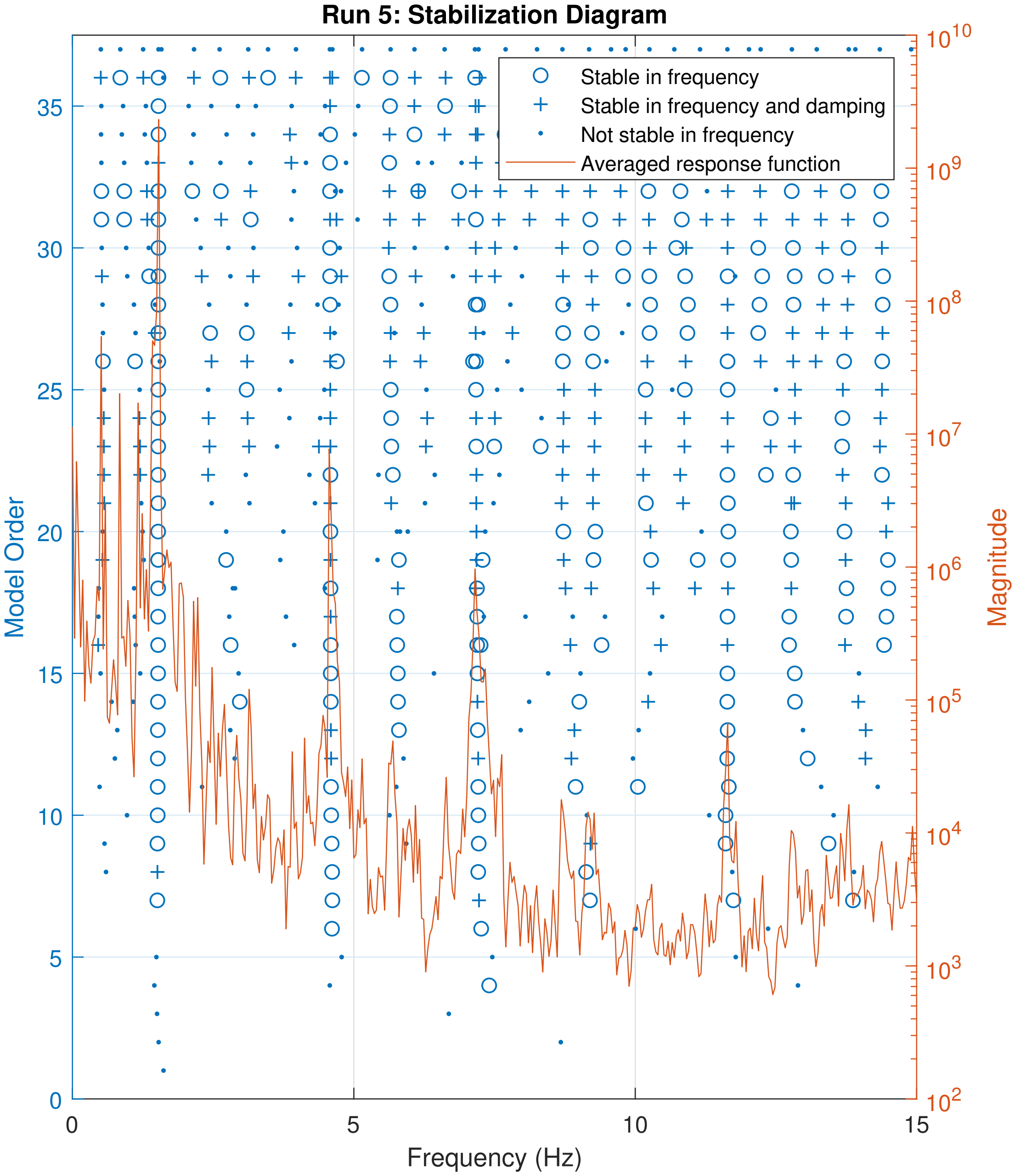


Figure B.59: Stabilization diagram from Run 5.

B.2.4 Laser Vibrometer FFTs

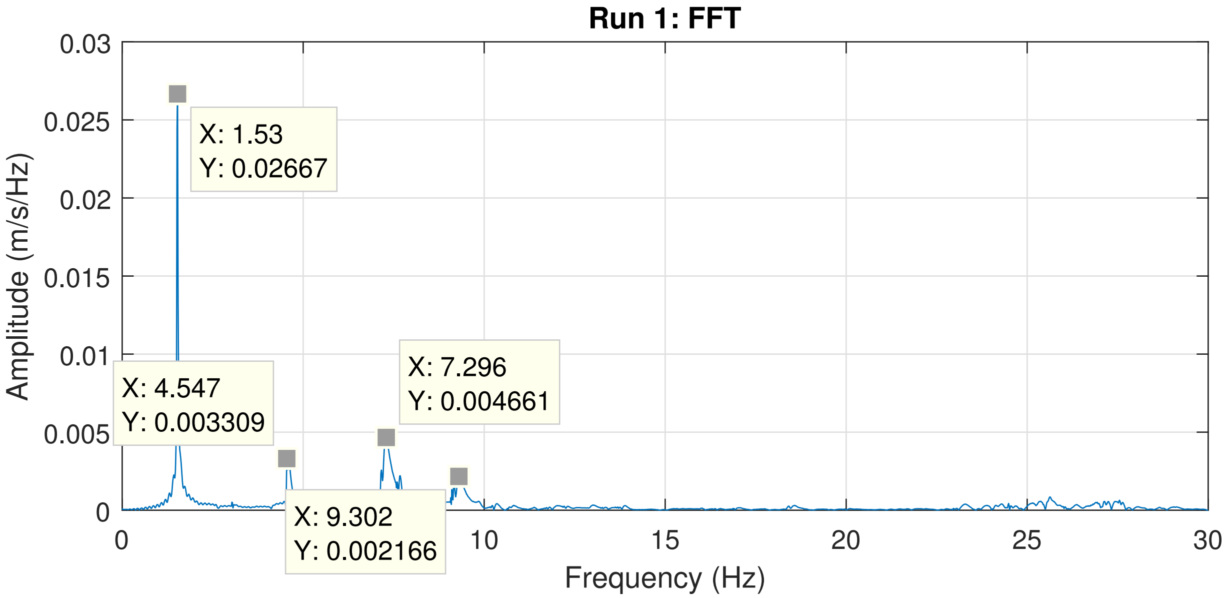


Figure B.60: FFT of LDV measurements from Run 1.

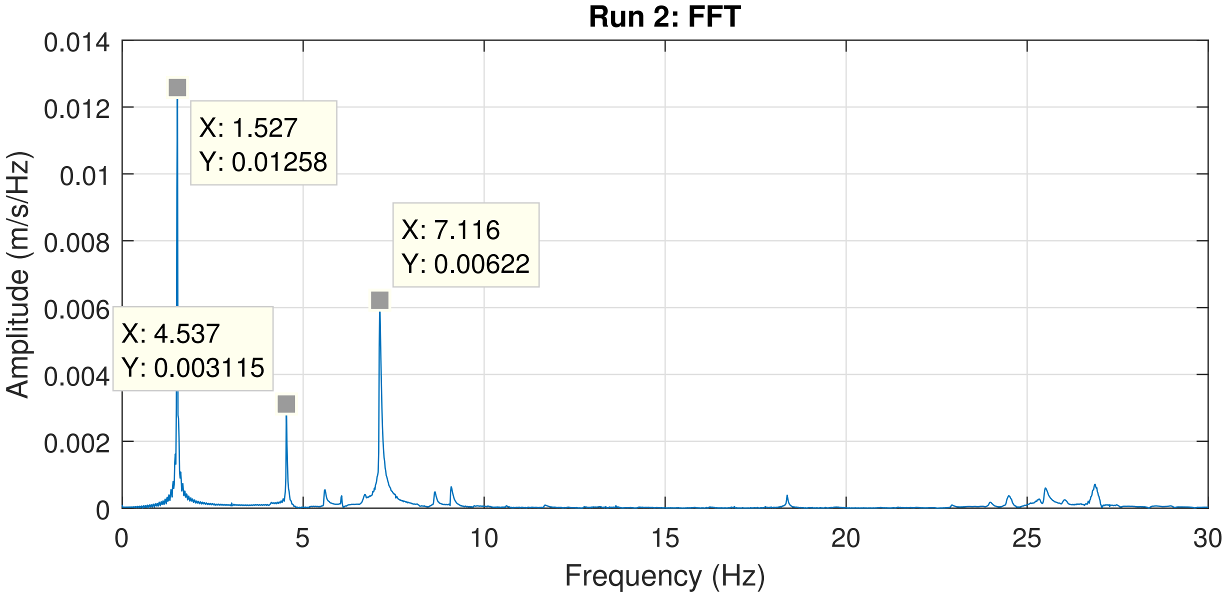


Figure B.61: FFT of LDV measurements from Run 2.

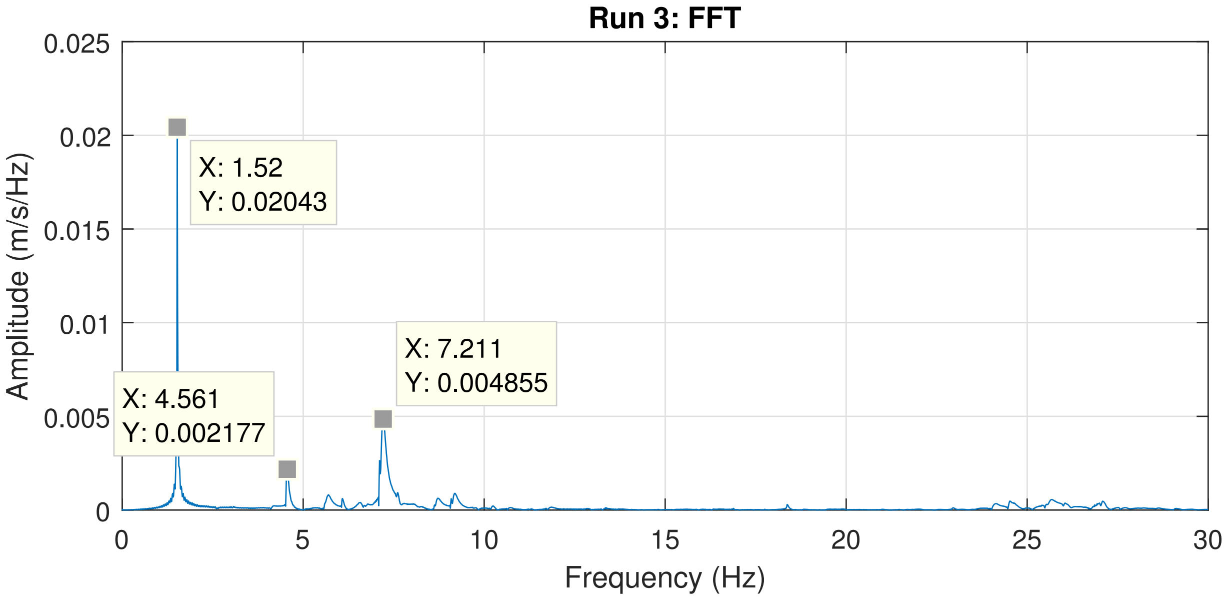


Figure B.62: FFT of LDV measurements from Run 3.

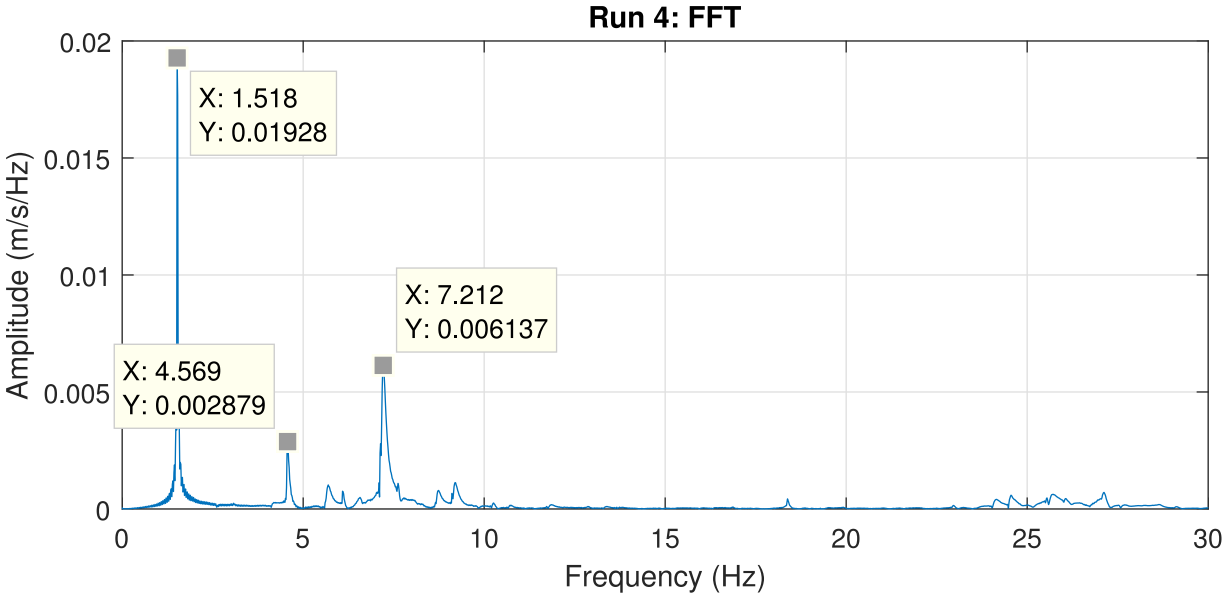


Figure B.63: FFT of LDV measurements from Run 4.

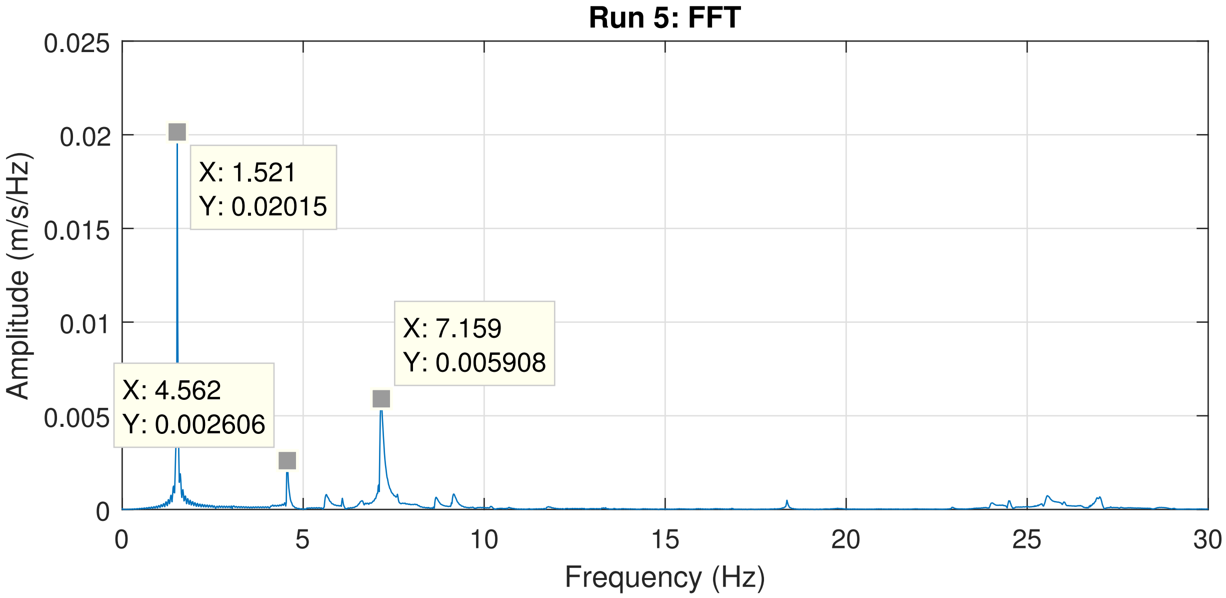
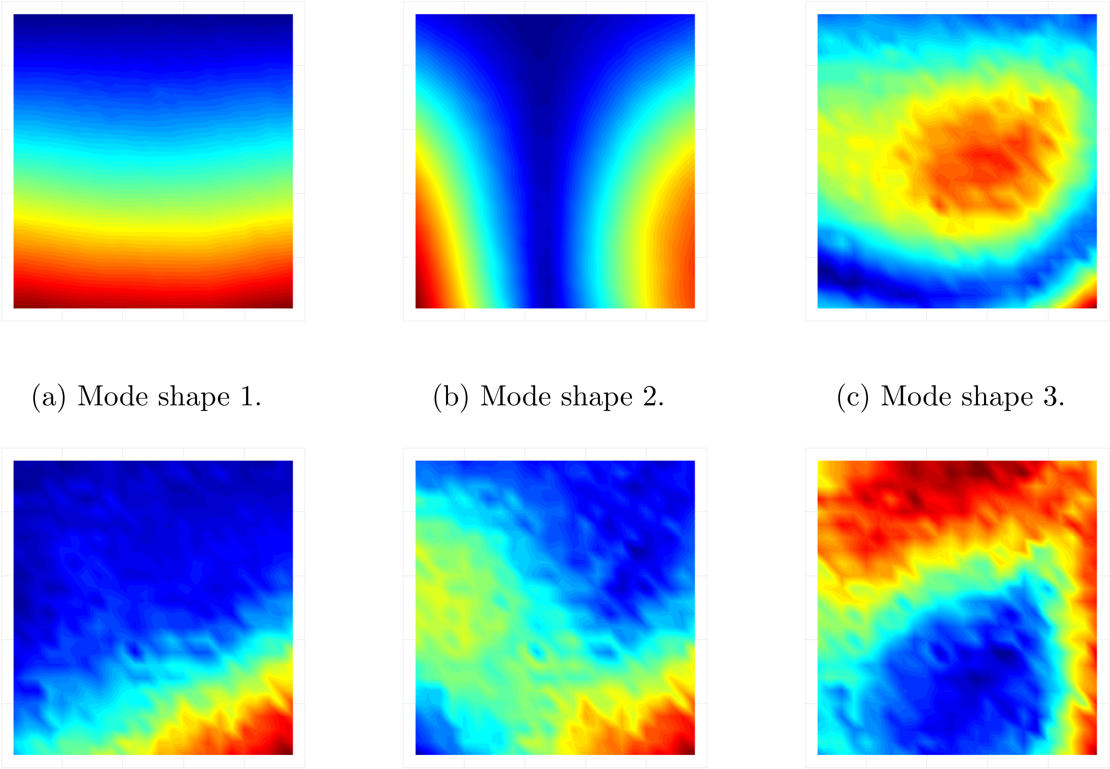


Figure B.64: FFT of LDV measurements from Run 5.

B.2.5 Mode Shapes



(a) Mode shape 1.

(b) Mode shape 2.

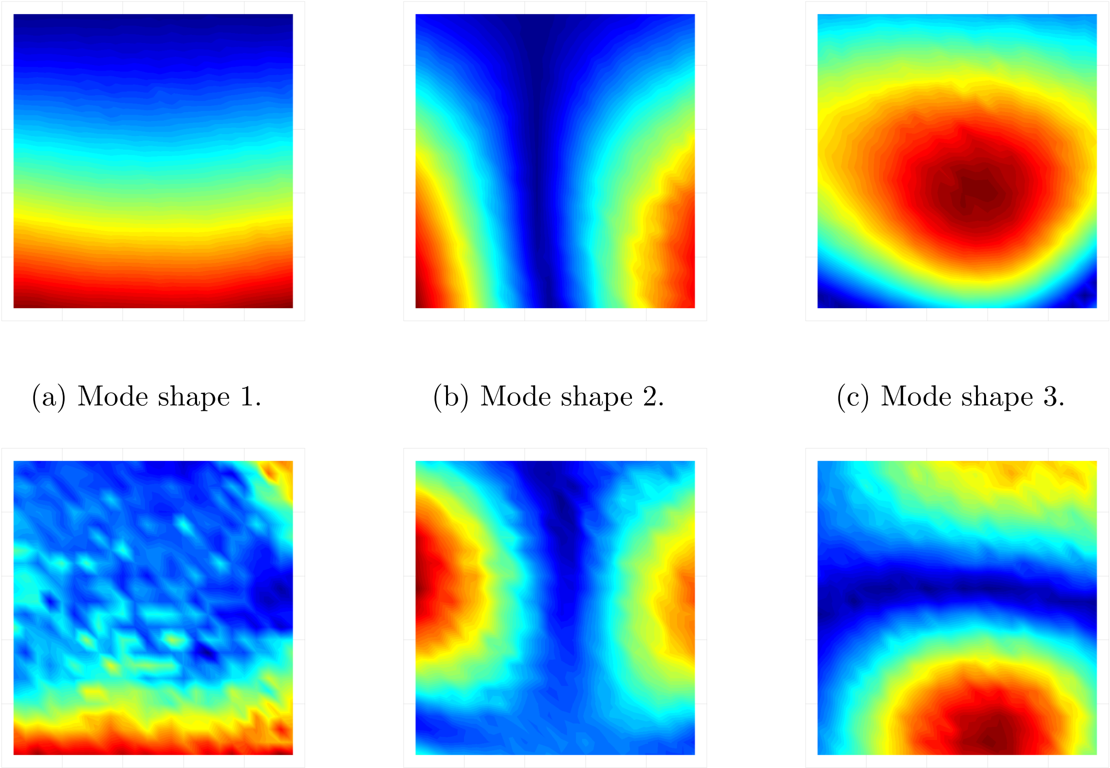
(c) Mode shape 3.

(d) Mode shape 4.

(e) Mode shape 5.

(f) Mode shape 6.

Figure B.65: Mode shapes of Run 1.



(a) Mode shape 1.

(b) Mode shape 2.

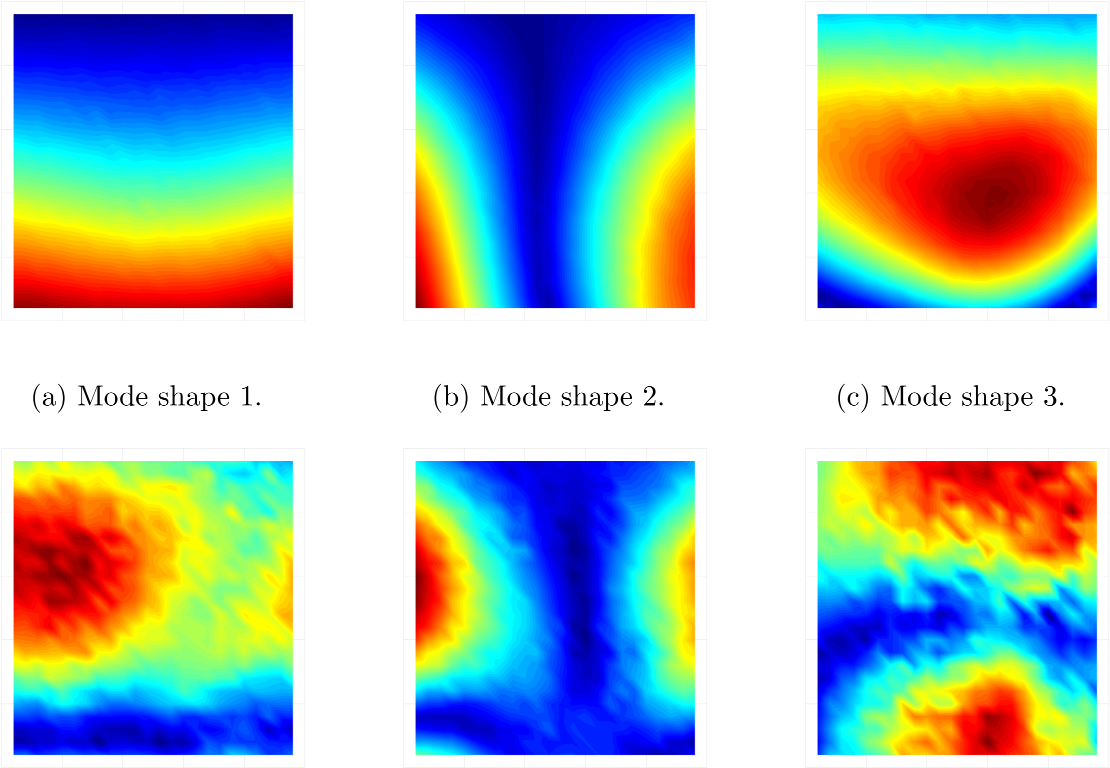
(c) Mode shape 3.

(d) Mode shape 4.

(e) Mode shape 5.

(f) Mode shape 6.

Figure B.66: Mode shapes of Run 2.



(a) Mode shape 1.

(b) Mode shape 2.

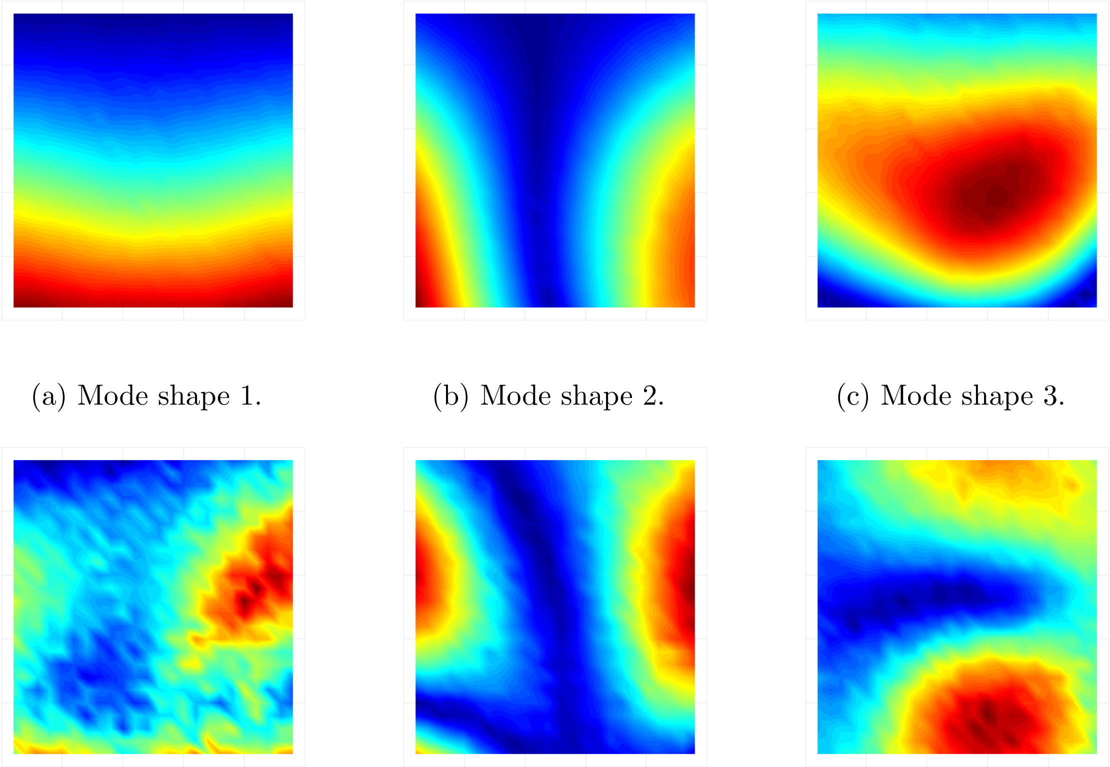
(c) Mode shape 3.

(d) Mode shape 4.

(e) Mode shape 5.

(f) Mode shape 6.

Figure B.67: Mode shapes of Run 3.



(a) Mode shape 1.

(b) Mode shape 2.

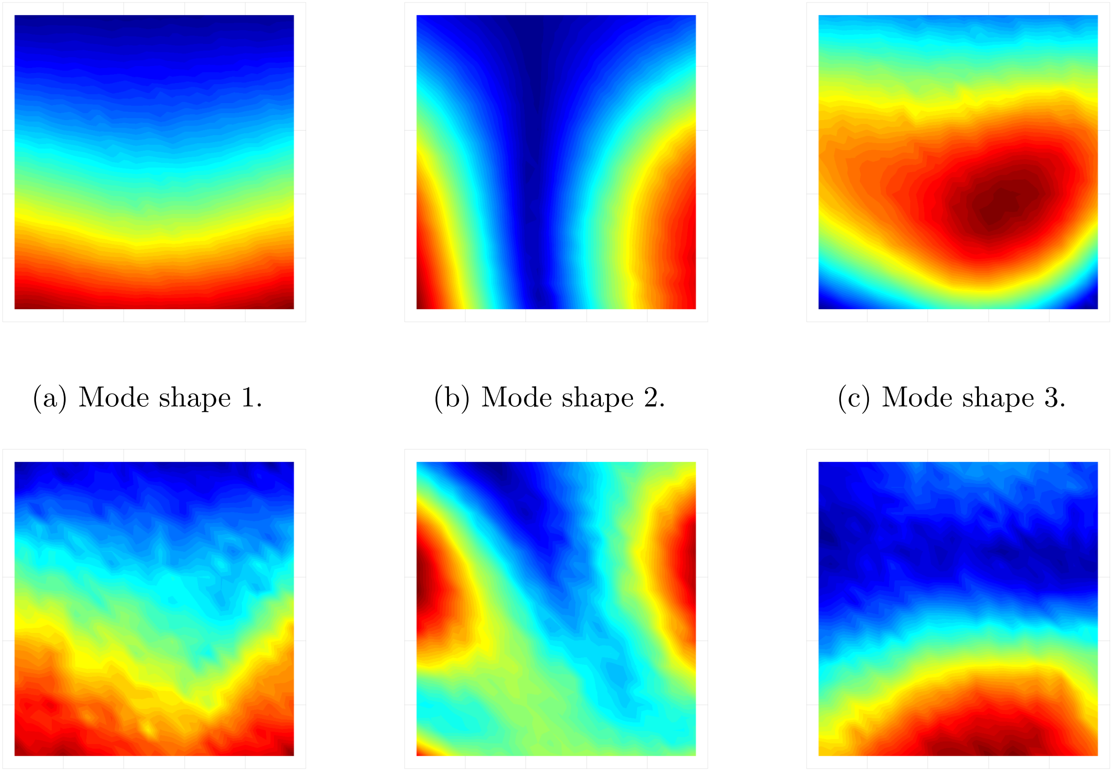
(c) Mode shape 3.

(d) Mode shape 4.

(e) Mode shape 5.

(f) Mode shape 6.

Figure B.68: Mode shapes of Run 4.



(a) Mode shape 1.

(b) Mode shape 2.

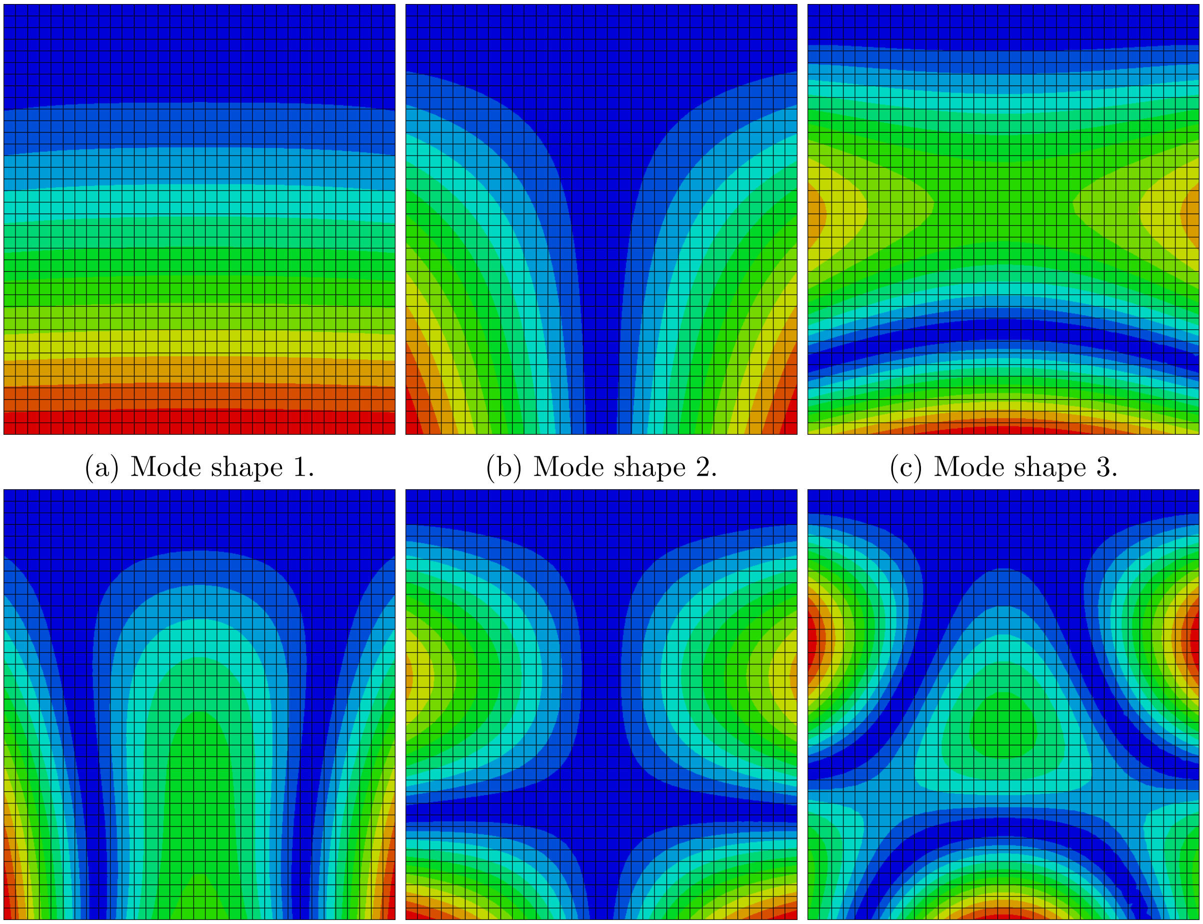
(c) Mode shape 3.

(d) Mode shape 4.

(e) Mode shape 5.

(f) Mode shape 6.

Figure B.69: Mode shapes of Run 5.



(a) Mode shape 1. (b) Mode shape 2. (c) Mode shape 3.

(d) Mode shape 4. (e) Mode shape 5. (f) Mode shape 6.

Figure B.70: FEA mode shapes