



The  
University  
Of  
Sheffield.

# PERSONALISED FINITE- ELEMENT MODELS USING IMAGE REGISTRATION IN PARAMETRIC SPACE

Development of Computational Workflows for Cardiac Resynchronisation Therapy

Daniel Alejandro Silva Soto

Submitted for the degree of Doctor of Philosophy

*Medical Physics Group  
Department of Cardiovascular Science  
Faculty of Medicine, Dentistry and Health*

*September 2014*

## ABSTRACT

Heart failure (HF) is a chronic clinical condition in which the heart fails to pump enough blood to meet the metabolic needs of the body. Patients have reduced physical performance and can see their quality of life severely impaired; around 40-70% of patients diagnosed of HF die within the first year following diagnosis. It is underestimated that 900,000 people in the UK currently suffer from HF. HF has a big impact on the NHS, representing 1 million inpatient bed, 5% of all emergency medical admission to hospitals and costs 2% of the total NHS budget. The annual incidence of new diagnoses is reported as 93,000 people in England alone – and this figure is already increasing at a rate above that at which population is ageing [1]. Cardiac resynchronisation therapy (CRT) has become established as an effective solution to treat selected patients with HF.

The research presented in this thesis has been conducted as part of a large EPSRC-Funded project on the theme of Grand Challenges in Healthcare, with co-investigators from King's College London (KCL), Imperial College London, University College London (UCL) and the University of Sheffield. The aim is to develop and to apply modelling techniques to simulate ventricular mechanics and CRT therapy in patient cohorts from Guy's Hospital (London) and from the Sheffield Teaching Hospitals Trust. This will lead to improved understanding of cardiac physiological behaviour and how diseases affect normal cardiac performance, and to improved therapy planning by allowing candidate interventions to be simulated before they are applied on patients.

The clinical workflow within the hospital manages the patient through the processes of diagnosis, therapy planning and follow-up. The first part of this thesis focuses on the development of a formal process for the integration of a computational analysis workflow, including medical imaging, segmentation, model construction, model execution and analysis, into the clinical workflow. During the early stages of the project, as the analysis workflow was being compiled, a major bottle-neck was identified regarding the time required to build accurate, patient-specific geometrical meshes from the segmented images. The second part of this thesis focuses on the development of a novel approach based on the use of image registration to improve the process of construction of a high-quality personalised finite element mesh for an individual patient.

Chapter 1 summarises the clinical context and introduces the tools and processes that are applied in this thesis. Chapter 2 describes the challenges and the implementation of a

computational analysis workflow and its integration into a clinical environment. Chapter 3 describes the theoretical underpinnings of the image registration algorithm that has been developed to address the problem of construction of high-quality personalised meshes. The approach includes the use of regularisation terms that are designed to improve the mesh quality. The selection and implementation of the regularisation terms is discussed in detail in Chapter 4. Chapter 5 describes the application of the method to a series of test problems, whilst Chapter 6 describes the application to the patient cohort in the clinical study. Chapter 7 demonstrates that the method, developed for robust mesh construction, can readily be applied to determine boundary conditions for computational fluid dynamics (CFD) analysis. Chapter 8 provides a summary of the achievements of the thesis, together with suggestions for further work.

# TABLE OF CONTENTS

<b>ABSTRACT</b>	<b>I</b>
<b>TABLE OF CONTENTS</b>	<b>III</b>
<b>LIST OF FIGURES</b>	<b>VI</b>
<b>LIST OF TABLES</b>	<b>XVI</b>
<b>CHAPTER 1 INTRODUCTION</b>	<b>1</b>
1.1 Clinical Context	1
1.1.1 The Failing Heart	1
1.1.2 Cardiac Resynchronisation Therapy	5
1.1.3 Imaging techniques for clinical assessment	7
1.2 The Simulation Workflow	8
1.2.1 Overview	8
1.2.2 Imaging and Segmentation	9
1.2.3 Models and Meshes	15
1.2.4 Material Properties and Boundary Conditions	17
1.2.5 Solution	19
1.2.6 Analysis and Interpretation	23
1.3 Contributions of this thesis to the simulation workflow	25
1.3.1 Integration with Clinical Workflows	26
1.3.2 Mesh Quality	27
1.4 Thesis organisation	28
<b>CHAPTER 2 CLINICAL INFORMATICS: MEDICAL DATA AND WORKFLOWS</b>	<b>29</b>
2.1 Introduction	29
2.2 The medical record and clinical information systems	32
2.2.1 Medical record	32
2.2.2 Electronic patient record	32
2.2.3 Information systems	33
2.2.4 Dicom standard	34
2.2.5 Databases	36
2.2.6 Clinical ethics	36
2.2.7 Workflows	37
2.3 Data provenance	38
2.4 Workflows: principles and clinical applications	43
2.4.1 Computational Workflows	44
2.4.2 Petri Nets	45
2.4.3 Clinical Workflows	49
2.4.4 Implementing Workflows	55
2.5 Implementation of clinical informatics to support the simulation workflow	56
2.5.1 The clinical database	56
2.5.2 The workflow system	59

2.6	Summary	63
<b>CHAPTER 3</b>	<b>IMAGE REGISTRATION</b>	<b>64</b>
3.1	Introduction	64
3.2	Components of Image Registration	65
3.2.1	Transformation model type	66
3.2.2	Transformation models	67
3.2.3	Similarity metric	68
3.2.4	Optimisation	69
3.3	Development of the registration algorithm	69
3.3.1	Intensity-based registration in Cartesian space	69
3.3.2	Intensity-based registration in parametric space	78
3.3.3	Linear least squares formulation	89
3.4	Discussion	92
<b>CHAPTER 4</b>	<b>REGULARISATIONS TO THE REGISTRATION</b>	<b>93</b>
4.1	Ill-conditioning	93
4.2	Tikhonov regularisation of the linear least squares formulation	95
4.3	Options for Tikhonov matrices	96
4.3.1	Identity matrix	96
4.3.2	Laplacian matrix	96
4.4	Options for Tikhonov matrices based on mesh quality metrics	98
4.4.1	Mesh quality metrics	98
4.4.2	Scalar product as a metric of element orthogonality	100
4.4.3	A Jacobian-based metric of element orthogonality	103
4.5	Iterative registration, multiple increment	107
4.6	Testing of regularisation options	108
4.7	Discussion	115
4.8	Summary	119
<b>CHAPTER 5</b>	<b>IMPLEMENTATION AND VALIDATION ON SYNTHETIC DATA</b>	<b>120</b>
5.1	Introduction	120
5.2	Implementation	120
5.2.1	Overview	120
5.2.2	Pre-processing stage	121
5.2.3	Iteration stage	124
5.2.4	Choice of programming language	127
5.2.5	Numerical correctness	128
5.3	Proof-of-concept	128
5.4	Validation	130
5.4.1	Assessment criteria	130
5.4.2	Experimental data	132
5.4.3	Experiment: Number of sampling points	134
5.4.4	Experiment: Laplacian regularisation	136

5.4.5	Experiment: Laplacian memory	137
5.4.6	Experiment: Scalar product regularisation	138
5.4.7	Experiment: Jacobian regularisation	140
5.4.8	Experiment: Strain regularisation	142
5.4.9	Experiment: Number of padding elements and trimmed base	143
5.5	Conclusions	144
<b>CHAPTER 6</b>	<b>PERSONALISED FINITE-ELEMENT MESHES OF THE LEFT VENTRICLE</b>	<b>146</b>
6.1	Introduction	146
6.1.1	Review of approaches to mesh construction	146
6.2	Image acquisition and segmentation	151
6.2.1	Image acquisition	151
6.2.2	Image segmentation	154
6.3	Preparation for image registration	160
6.3.1	Pre-processing of segmented images	160
6.3.2	Template Mesh	164
6.3.3	Quality Metrics	167
6.4	Registration parameters	170
6.5	Results and discussion	171
6.6	Conclusions	180
<b>CHAPTER 7</b>	<b>IMAGE-DERIVED WALL MOTION FOR HAEMODYNAMIC ANALYSIS</b>	<b>183</b>
7.1	Motivation	183
7.2	Method overview	185
7.3	Registration	186
7.4	Simulation setup	188
7.4.1	Multi-field simulation	188
7.4.2	Fluid flow simulation	189
7.4.3	Mechanical simulation	191
7.5	Results	193
7.6	Discussion	197
7.7	Conclusion	199
<b>CHAPTER 8</b>	<b>CONCLUSIONS AND FUTURE WORK</b>	<b>200</b>
8.1	Summary	200
8.2	Limitations and Future Work	202
<b>PUBLICATIONS</b>		<b>204</b>
<b>REFERENCES</b>		<b>205</b>

# LIST OF FIGURES

Figure 1: Diagram of the human heart	1
Figure 2: (a) the electrical conduction system, and (b) an annotated ECG signal. The form of the signal changes depending on the location of the measuring leads. The P wave represents atrial systole and QRS represents ventricular systole. At time R, the mitral valve closes and at time S, the aortic valve opens and closes again late into the T wave. At the end of the T wave, the mitral valve opens.	4
Figure 3: The recursive process of ventricular remodelling. Schematic representation (left) and its effect on chamber size and shape after MI (right)	5
Figure 4: Conceptual flowchart for simulation. Medical specialities collect different information in the patient history, as they find relevant. Example of medical testing include blood, urine and hormone tests.	9
Figure 5: Different views of the heart during ventricular systole using cine MRI. The image on the left is a 2-chamber view, showing the left ventricle and atrium. The image on the right is a 4-chamber view.	12
Figure 6: 3D MRI of the heart. The image on the left shows the left ventricle of the heart; the image on the right shows the left and right ventricles, the left atrium and part of the aortic arch. Both slices are part of the same 3D volume.	12
Figure 7: 3D MRI of the chest (left) and segmentation of main structures (right) presented using a labelled mask. The segmentation was acquired using the tool described in [22] and shows the left ventricular myocardium (blue), left ventricular volume (green), right ventricular volume (yellow), right atrium (orange), left atrium (turquoise), aorta (red) and vena cava (red).	14
Figure 8: Representative mesh geometry of the heart ventricles. (Adapted with permission under Creative Commons BY-SA 3.0 from [33], formatted horizontally for better presentation.)	17
Figure 9: Image preparation and segmentation process by Sermesant et al. A volume MR of the thorax is acquired but filtering is required to identify structures, the ventricular blood pools in this case. With this information a surface representing the ventricles is automatically generated and is then manually corrected. (reproduced with permission from [34])	18
Figure 10: Schematics of the tool chain for modelling the electrophysiology (using the Cardiac Arrhythmia Research Package) and mechanics (using the Continuum Mechanics, Imaging, Signal processing and System identification environment) of the heart (adapted from [44])	26

Figure 11: User interface of the ASPIRE software developed by Taylor et al. allows basic virtual surgical intervention to be performed on the vessel surface to investigate the resulting changes in pressure and flow to aid surgical planning. (Reproduced with permission from [54])	31
Figure 12: General networking setup to manage image data within the hospital.	35
Figure 13: The taxonomy of provenance describes points to consider when defining a provenance model [71]	41
Figure 14: The provenance concept map shows the different ways provenance is used and how it links together different data and process management concepts [72]	43
Figure 15: Basic illustration of a Petri net. (a) Shows basic token consumption after the firing of an event. (b) Shows branching at transitions and is an example of concurrency. (c) Shows branching at places and is an example of an unambiguous system. (d) Shows a situation of confusion.	46
Figure 16: Purpose, automation, control, and design of workflows shown using Petri net notation. The left-hand panels illustrate the difference between a state machine and a sequential workflow.	47
Figure 17: A sample workflow used in a scientific laboratory. It performs some data manipulation, executes some tests, and queries a database to analyse results (adapted with permission under Creative Commons BY-SA 3.0 from [78], split into two panels)	50
Figure 18: Due to the multiple branching points and points of re-entry into the system, workflows for a single clinical department can be troublesome to model sequentially.	51
Figure 19: When the number of permutations that may happen is high, as in a hospital, the state machine paradigm makes more sense.	52
Figure 20: The ArQ Designer interface. On the left, the list of plug-ins can be seen. On the right, the relationships between database fields can be seen on the right.	57
Figure 21: the “visits” page of the clinical database, showing (a) the multiple table select plug-in and (b) the page selector tree view. Combining multiple visits (of different assessment types) into one folder is an interesting feature because each visit type uses a different table in the underlying database structure.	58
Figure 22: The ArQ Dicom study anonymisation plug-in (left) and the behind-the-scene properties of the plug-in. Users click on the button, navigate to a directory in their system and a	



drop-down with the Dicom studies present in it appear. The chosen study is copied to the storage directory and anonymised.	59
Figure 23: Illustration of the type of layered architecture that might be employed in a workflow system	61
Figure 24: Components of image registration.	65
Figure 25: Forward and inverse registration mapping. The top distorted grid relates to image [M] and the bottom one to [F]. (Figure generated using SHIRT)	70
Figure 26: Re-sampling of moved image after mapping to fixed points on original Cartesian grid. The thick lines delineate the voxel area, the intermittent lines connect the voxel centres (green). The moving voxel (red) must have undergone a mapping $\{u,v\}$ , based on the position and gradients at $f(x,y)$ , to arrive at $m_1(x,y)$ .	72
Figure 27: Voxel centres, where the residuals are calculated, become unaligned during the registration. The figure shows (a) the moving image, (b) the fixed image, (c) the initial overlap and (d) the overlap and distorted grid after one iteration.	73
Figure 28: Schematic of pulling a voxel of $m$ onto $f$ . The thick lines delineate the voxel area, the intermittent lines connect the voxel centres (green). The moving voxel (red) undergoes the mapping $\{u,v\}$ , based on its position and the gradients at $m(x,y)$ , to take it to $m(x',y') = m_1(x,y)$ .	75
Figure 29: Summarises the iterative update of the moving image, the residual converges to a minimum with successive iterations	76
Figure 30: Flowchart showing the iterative registration process described in this chapter.	77
Figure 31: Comparison of grid types (unstructured and structured) with the structured nature of a digital image, which is typically described in Cartesian space.	79
Figure 32: Local coordinate system (left) and its location within the Cartesian image	79
Figure 33: Image of 48 x 48 pixels with a 4x4 grid (5x5 nodes) superimposed	81
Figure 34: A four element grid with 8x8 pixels in each element	81
Figure 35: An element of the parametric moving image (blue) overlaid on the fixed image (green). The centres of the voxels are indicated by the coloured dots	82
Figure 36: A quadrilateral element defined by nodes $N_1$ to $N_4$ and sampling point, P	83
Figure 37: A hexahedral element defined by nodes $N_1$ to $N_8$ and sampling point, P	87

Figure 38: The translating square problem and two valid solutions (translation and rigid body) that maximise image similarity.	94
Figure 39: Geometric measures of element quality	98
Figure 40: Jacobian-based energy terms	100
Figure 41: The degrees of freedom of the nodes that form a junction	101
Figure 42: Variation of the element Jacobian for quadrilaterals under deformation	103
Figure 43: Input data for the regularisation tests showing (a) the small square, (b) the larger rectangle, and (c) both images superimposed. The overlap of the green and red images appears yellow.	108
Figure 44: Effects of common regularisations used in image registration. The moving image (green) and the registration grid is presented on top of the fixed image (red). The images are blended to show the areas of overlap (yellow). The algorithm runs multiple times with decreasing node spacing in the registration grid, each row represents the last iteration at each grid spacing.	109
Figure 45: The effect of the scalar product regularisation on a distorted regular grid, allowing a different number of nodes to move (left-most column). The resulting nodal positions are shown after one, two, five and ten iterations.	111
Figure 46: The effect of the scalar product regularisation with Laplacian smoothing on a distorted regular grid, allowing a different number of nodes to move (left-most column). The resulting nodal positions are shown after one, two, five and ten iterations.	112
Figure 47: The initial setup of the distorted 27-element mesh used to test the Jacobian regularisation	113
Figure 48: Node positions during the correction of the mesh shown in Figure 47 using the element Jacobian regularisation, at iteration numbers 1(a) to 6(f) and 15(g).	114
Figure 49: Flowchart showing the different stages of the registration program described in section 5.2	120
Figure 50: Consistent node numbering of mesh elements. The element in red contains extended neighbours of the centre-most element	122
Figure 51: The layout of the sampling points, showing the effect of varying the size of the elements has on the distances of points across elements	124

Figure 52: Typical methods for interpolating data, in increasing order of complexity: nearest neighbour, bilinear interpolation, bicubic interpolation and cubic spline interpolation. Panels (a) to (d) use a coarse, 4-by-4 sub-pixel sampling grid. Panels (e) to (h) use a fine, 400-by-400 sub-pixel sampling grid. Cubic spline interpolation is efficient for fine sub-pixel sampling however this work does not benefit such refinement, in which bilinear is the most appropriate sub-pixel sampling method. 126

Figure 53: Registration of a hexagon to a circle showing a) the moving image, b), the fixed image, c) the images overlay, d) the registration result using fixed edges, and e) the registration result using free edges. 129

Figure 54: Registration of a rectangle to a circle showing a) the moving image, b), the fixed image, c) the images overlay, d) the registration result using fixed edges, and e) the registration result using free edges. 129

Figure 55: Registration of a “w-like” complex shape to a similar shape, which has been distorted considerably, showing a) the moving image, b), the fixed image, c) the images overlay, d) the registration result using fixed edges, and e) the registration result using free edges. 129

Figure 56: Registration of two slices of a cardiac MRI showing a) the moving image, b), the fixed image, c) the images overlay, d) the registration result using fixed edges, and e) the registration result using free edges. 130

Figure 57: The setup of the synthetic tests. Panels (a) to (c) show the initial mesh and image set up and panels (d) to (f) show the result after registration. The axis denotes the extent of the image, which is shown as a red surface. 133

Figure 58: The solution to registering a rotated cube to a cube aligned with the Cartesian grid is driven by the choice of the regularisation term. Panel (a) shows the initial set up. The image is shown as a red surface and the axis shows the extent of the image. Panel (b) shows the result after registration, which might not be the desired outcome to the process. 134

Figure 59: The mean surface distance (left) and image similarity (right) after registration as a function of sampling points used. This figure was created using the results presented in Table 8 135

Figure 60: Resulting meshes when registering a 9 element cube to the segmentation of a cube with and without the memory term, using a  $\lambda$  value of 1.0. 138

Figure 61: Resulting meshes when registering a 9 element cube to the segmentation of a cube using the scalar product regularisation and different values for  $\lambda$ . 140

Figure 62: Evolution of the quality of the mesh, with appropriate and inappropriate scaling of the scalar product regularisation	140
Figure 63: Resulting meshes and the evolution of element quality when registering a 9 element cube to the segmentation of a cube using the Jacobian regularisation and different values for $\lambda$ .	141
Figure 64: Resulting meshes when registering a 9 element cube to the segmentation of a cube using the strain regularisation and different values for $\lambda$ .	142
Figure 65: Resulting mesh and fitting distance when registering the ventricular mesh to the idealised geometry using a variable number of padding elements.	143
Figure 66: Image of the idealised ventricle and the registered mesh. The second row shows a histogram of the distance of isosurface points from the idealised mesh. All plots are colour-coded by distance.	144
Figure 67: Schematic of the method described by Jones et al. The row labelled 'reference' corresponds to the patient geometry. A volumetric MR of the patient's thorax was manually segmented. An idealised aortic geometrical mesh was built and converted to a binary volumetric image, which was registered to the segmentation. The map resulting from the registration process was then applied on the idealised mesh to produce a patient-specific mesh. (reproduced with permission from [85])	149
Figure 68: Input and processed data using the method described by Barber et al. (reproduced with permission from [86])	149
Figure 69: Flowchart of the process of cubic Hermite mesh generation described in Lamata et al. To personalise the template mesh, this is binarised and registered to the segmentation of the patient's myocardium. The resulting displacement field is applied on the nodes of the mesh, and the faces of the cubic Hermite elements are warped to the closest surface in the patient's segmentation to produce the personalised mesh. (reproduced with permission from [87])	151
Figure 70: Sagittal slices of the volunteer scan taken at Guy's Hospital, London	153
Figure 71: Sagittal slices of the volunteer scan taken at the Northern General Hospital, Sheffield	153
Figure 72: Scans of multiple patients taken at Guy's Hospital, London	154
Figure 73: Scans of multiple patients taken at the Northern General Hospital, Sheffield	154

Figure 74: Manual segmentation of the left ventricular myocardium (top) and aorta (bottom) of healthy volunteer 2	156
Figure 75: The triangular mesh generated by the KCL segmentation tool. The LV myocardial volume mesh is shown alongside the patient MRI	157
Figure 76: UCL automatic segmentations corresponding to the patient scans shown in Figure 73	158
Figure 77: Manual segmentations corresponding to the patient scans shown in Figure 73	158
Figure 78: Visual comparison of selected segmentation results. The red and green outlines correspond to the UCL and manual segmentations, respectively	159
Figure 79: Rejected UCL segmentations from the London set of cases. The figure shows opaque and translucent renders of cases number (a) 3, (b) 7 and (c) 19.	160
Figure 80: (a, d) Manual segmentation, (b, e) UCL and (c, f) KCL automatic segmentations. Slice through (a-c) and 3D render (d-f).	161
Figure 81: The effect of 3D image filters on a segmented volume. (1) The raw volume is a manual segmentation, produced by the author, using the tool shown in Figure 74 whilst trying to avoid the fallacies of manual segmentations shown in Figure 77 and Figure 80. (2) Gaussian filter. (3) Median filter. (4) Mean filter. (From left to right)	162
Figure 82: Comparison of the pre- (red) and post-processed (blue) segmentations: (a) isosurfaces, and (b)-(d) accuracy of the volumes, compared in orthogonal planes.	163
Figure 83: Processing of 3 patient segmentations: the raw volume (top), and filtered and cut volume (bottom)	164
Figure 84: Parameterisation of the template mesh using characteristics of the ventricular geometry.	165
Figure 85: A typical template mesh viewed from different angles.	165
Figure 86: Two possible configurations to arrange the elements at the apex of the ventricular meshes	166
Figure 87: Patient-specific template meshes; Sheffield cases numbers: 1, 7, 14 and 20 (left to right)	167
Figure 88: Schematic of the surface distance measurements. The mesh faces (a) are broken into face triangles (b). For each point in the surface of the segmentation, the closest point on the	

face triangles is computed, which could be on the triangle face (c), edge(d), node (e) or plane (f).	168
Figure 89: Evolution the mesh of case MAN19; initial template, after alignment, quality and conformance. In the cross-sections, the segmentation appears black and the mesh appears red.	172
Figure 90: Resulting surface distance for all cases executed in this work, using the conformance and quality mesh fitting protocols.	173
Figure 91: Resulting mesh degradation for all cases executed in this work, using the conformance and quality mesh fitting protocols.	173
Figure 92: Resulting number of degenerate mesh elements for all cases executed in this work, using the conformance and quality mesh fitting protocols.	173
Figure 93: Processing results for case MAN13. The figure shows a side view of the mesh is shown with the target isosurface (left-most column). A view of the mesh with badly deformed elements highlighted is shown (top row) and the mesh with the points in the segmentation isosurface that are more than the mean voxel size distance away from the mesh highlighted (bottom row). The columns are ordered by start-end position, conforming and quality mesh (left to right).	175
Figure 94: Processing results for case SHF16 The figure shows a side view of the mesh is shown with the target isosurface (left-most column). A view of the mesh with badly deformed elements highlighted is shown (top row) and the mesh with the points in the segmentation isosurface that are more than the mean voxel size distance away from the mesh highlighted (bottom row). The columns are ordered by start-end position, conforming and quality mesh (left to right).	176
Figure 95: Degenerate elements of the resulting mesh for case LDN06, using the conformance protocol (a) and the quality protocol (b)	177
Figure 96: Cross-section of the template and fitted meshes for cases SHF15 (a) and MAN12 (b)	177
Figure 97: Comparison of the fitting distances and mesh degradation achieved when using initial templates with different numbers of elements.	178
Figure 98: Binary images before and after registration using ShIRT, 3D (top) and slice (bottom) views	180
Figure 99: Resulting mesh fitting for thinner (left) and thicker (right) target geometries using same sized meshes.	181

Figure 100: Overview of a typical aortic simulation workflow, highlighting the 3D mesh [136]. The imaging modalities used include 3D MR and CT for extracting the vessel geometry, and MR phase contrast for the inlet boundary condition and tuning the 0D systemic models.	184
Figure 101: Schematic of the process described in this chapter. The anatomical mesh is static, in contrast with the ventricular region of the mesh that will deform at each time step.	185
Figure 102: 3D + t segmentation of the left ventricular volume and aortic valve (left) and surface mesh of the ventricular volume, aortic valve and aorta (right). One volume dataset, courtesy of Peters and Weese	186
Figure 103: Processing the images, before (blue) and after (red) being truncated at the cut plane. The fitted mesh is shown on the first time step.	187
Figure 104: The original (orange) and processed (grey) images, and the inner surface (green) of the registered mesh	188
Figure 105: Flowchart showing the coupling set-up for the simulation method used in this chapter.	189
Figure 106: Side and top views of the surface mesh, including the moving ventricular wall (red), before meshing the inside volume for the fluid simulation.	190
Figure 107: Resulting enclosing volume for the simulation and the segmentation (orange) at different timesteps. The mesh is coloured by section: base (blue) and walls (white)	190
Figure 108: ANSYS Mechanical and Multi-field solver script. Note that the displacement values and the commands to apply them are stored in separate files to aid legibility.	192
Figure 109: Resulting flow at the ascending aorta	193
Figure 110: Velocity profile at the ascending aorta. Panel (d) shows the instant of maximum flow. The cross-sectional flow is clearly not plug-shaped. The side views of the outlet show there is a large amount of flow going back into the ventricle.	194
Figure 111: Velocity vectors in the fluid domain. Panel (d) shows the instant of maximum flow. The velocity distribution provides a visual proof the simulation is behaving correctly; in particular, the flow is pushed directly out into the aorta and the acceleration of flow near the narrowing at the outlet	195
Figure 112: Streamlines in the fluid domain (initiated from the ventricular wall). Panel (d) shows the instant of maximum flow. The streamlines provide a visual proof the simulation is behaving	

correctly; in particular, the flow is pushed directly out into the aorta and there is an area of recirculation behind the valve. 196

Figure 113: Sensitivity of results and computational requirements of the simulation for different mesh densities. 198



## LIST OF TABLES

Table 1: NYHA classification and its relationship with the ACC/AHA Stages of HF [5]	3
Table 2: Comparison of different commercial database systems [30]	36
Table 3: Differences between clinical and scientific workflow paradigms	49
Table 4: Clinical data storage in current study before implementation of the alternative to digital database system	58
Table 5: The derivatives of the iso-parametric bilinear shape functions	86
Table 6: The trilinear formulation shape function derivatives	89
Table 7: Summary of the regularised transformation model used in this thesis	117
Table 8: Summary of the assessment of the effect of changing the number of sampling points used in the registration process	135
Table 9: Summary of the assessment of the effect of using the identity and the Laplacian matrices as the Tikhonov matrices for image registration	136
Table 10: Summary of the assessment of using the Laplacian memory to regularise the registration, compared to using the Laplacian matrix with no memory	137
Table 11: Summary of the assessment of using the scalar product regularisation	139
Table 12: Summary of registration using the Jacobian regulariser. $\lambda_1$ is the weight of regularisers to image similarity and $\lambda_2$ is the weight of the Laplacian to Jacobian regulariser.	141
Table 13: Summary of the results using the strain regulariser	142
Table 14: Comparison of the multi-centre cMRI equipment used	152
Table 15: Comparison of the multi-centre cMRI image properties of the patient scans	153
Table 16: Comparison of the multi-centre cMRI image properties of the volunteer scans	153
Table 17: Summary of the available segmentations	160
Table 18: Summary of the registration parameters used in the fitting stages	170
Table 19: Summary of the improvement in metrics, broken down by segmentation type	174
Table 20: Detailed results (Figure 90, Figure 91 and Figure 92)	179
Table 21: Summary of the registration parameters used for tracking the ventricular wall.	187



# CHAPTER 1

## INTRODUCTION

### 1.1 Clinical Context

#### 1.1.1 The Failing Heart

The heart is the main organ in the cardiovascular system; slightly larger in size than a clenched fist, its main function is pumping blood around the body. The cardiac system is composed of two circuits: the pulmonary (connecting the right ventricle with the lungs and ending at the left atrium) and the systemic (connecting the left ventricle with the rest of the body and ending at the right atrium). Oxygenated blood is delivered to the heart by a network of coronary vessels that starts at the base of the ascending aorta, splits into the left and right coronary branches and ends at the right atrium. Blood flows along the circuits through vessels called arteries, veins and capillaries.

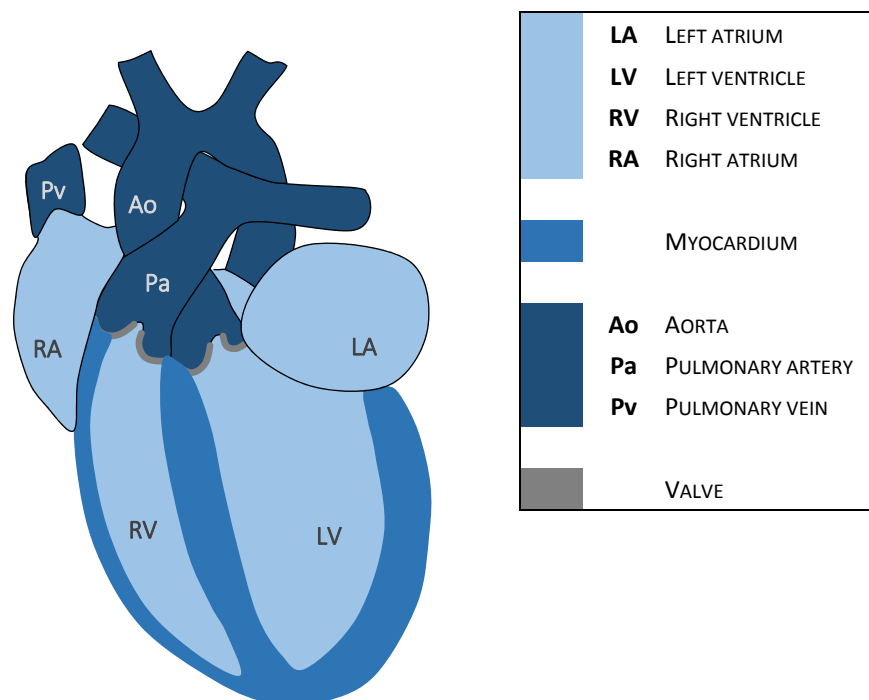


Figure 1: Diagram of the human heart

The heart is divided into four chambers: the left and right atria, and the left and right ventricles. Atria receive and accumulate blood; ventricles discharge the blood away from the heart. The left chambers have significantly thicker walls than the right chambers. Chambers are divided by

muscular walls: the inter-atrial septum and the inter-ventricular septum. The heart is composed mostly of cardiac muscle, as well as soft tissue and a fibrous skeleton, which provides structural support for muscle attachment and the valve's ring-like shape. Cardiac muscle wraps around and spirals around the ventricular walls. The walls of the heart are divided into 3 layers: the epicardium (outermost), the myocardium and the endocardium (innermost). The trabeculae carneae lie along the inside of the endocardium.

The heart contracts and relaxes in a cyclical fashion called the cardiac cycle. The muscles of the heart contract in response to an action potential, caused by ionic currents that flow through the membranes of the cells that make up the myocardium. The signal originates at the sino-atrial (SA) node and travels through the series of specialised cells that form the electrical conduction network of the heart (Figure 2 (a)). This network of cells originates at the SA node, which spontaneously discharges, then travels along the right and left atria into the atrio-ventricular (AV) node, where the network synchronises. The signal splits into the left and right bundle branches into each respective ventricle. The bundle branches into smaller fibres called the Purkinje fibres from which the signal is propagated over the soft tissue of the heart. Cardiomyocytes, or cardiac muscle cells, have a fibrous structure and striated arrangement. When the signal reaches the cardiomyocytes, the action potential causes the cells to contract. The blood flow is driven by pressure gradients that arise from the changing volume of the contracting/expanding chambers. Blood flow is restricted to one direction by the presence of valves. Valves open and close by the same pressure gradient that drives blood flow: no muscle is directly involved in their functioning. The cardiac cycle is divided into two parts, the contraction phase (systole) and the relaxation phase (diastole). During systole, the atria relax as blood arrives from the systemic circuit along the vena cava into the right atrium and from the pulmonary circuit along the pulmonary vein into the left atrium. At the same time, the ventricles have contracted thus pushing blood into the pulmonary circuit along the pulmonary artery from the right ventricle and into the systemic circuit along the aorta from the left ventricle. During diastole, the left and right atria contract pushing blood into the left and right ventricles, respectively. The cardiac cycle is repeated at a rate of around 60 beats per minute in an average healthy adult at rest.

Heart failure is a complex, chronic and progressive clinical condition in which the heart fails to supply the body with an adequate cardiac output for its metabolic needs, which can negatively impact on patients' physical and mental function [2]. HF affects about 2% of all western population and 10% of over 75s [3]. HF most commonly affects the functioning of the left

ventricle (LV) – known as LV dysfunction – and often leads to right ventricular dysfunction. Left ventricular function impairment, however, is not only caused by heart failure.

A symptomatic definition of HF can be given using the New York Heart Association (NYHA) heart failure classification – ranking patients in stages that progress with the development of the disease. The NYHA-HF classification is based on a patient’s feeling of fatigue when undertaking habitual activities. It does not, however, differentiate the presence of structural heart disease on those who are asymptomatic, or prescribe the appropriate level of treatment for each state. For this reason the NYHA classification is commonly reported alongside the 2001 ACC/AHA Stages of Heart Failure classification which is based on the association between the structural effects of the disease on the patient’s heart, the likelihood of patients reporting symptoms and the required medical treatment [4]. Table 1 shows the relationship between the two classifications using simple descriptions.

2001 ACC/AHA STAGES OF HF		NEW YORK HEART ASSOCIATION HEAR FAILURE CLASSIFICATION	
Stage	Description	Stage	Description
A	No reported symptoms but high risk of developing structural heart disease	-	-
B	No reported symptoms but detected structural heart disease	I	Without limitations of physical activity. Ordinary physical activity does not cause undue fatigue, palpitation, or dyspnoea
C	Reported symptoms and structural heart disease	II	Slight limitation of physical activity. They are comfortable at rest. Ordinary physical activity results in fatigue, palpitation, or dyspnoea
		III	Marked limitation of physical activity. They are comfortable at rest. Less than ordinary activity causes fatigue, palpitation, or dyspnoea
		IV	Inability to carry on any physical activity without discomfort. Symptoms are present even at rest or minimal exertion
D	Advanced structural heart disease requiring specialised interventions	IV	

Table 1: NYHA classification and its relationship with the ACC/AHA Stages of HF [5]

Common measurements used to quantify and assess the functioning of the heart exist and are associated with the natural and efficient cardiac cycle. Left ventricular ejection fraction (LVEF or EF) is the percentage volume of blood leaving the left ventricle at the end of systole. The complexity of the flow inside the ventricle dictates that to maximise cardiac efficiency EF is not 100%, but extremely low ejection fractions (<30%) are a sign of cardiac dysfunction and might be associated with a structurally-diseased heart. Another marker of systolic dysfunction is left

ventricular end-systolic volume (LVESV); this is the mean volume of blood left in the left ventricle at the end of systole. It is related to EF by the total left ventricular volume – LVESV value is typically 97mL/m<sup>2</sup> on a healthy heart (normalised by body surface area to account for different heart sizes) [3].

Electrocardiography is the measurement of electrical activity in the heart. Figure 2(b) shows the typical signal measured by the electrodes, placed on the surface of the patient’s skin. Electrocardiogram (ECG) measurements report data on overall cardiac function and provide hints on possible diseases. These can be reported as irregular pulse, delayed time to peak QRS (delayed AV node activation), reduced QRS amplitude, wrong T wave (recovery) functioning etc. Chamber pressure and volume variation plots can also provide useful information for clinicians such as the possibility of mitral regurgitation or systolic dysfunction.

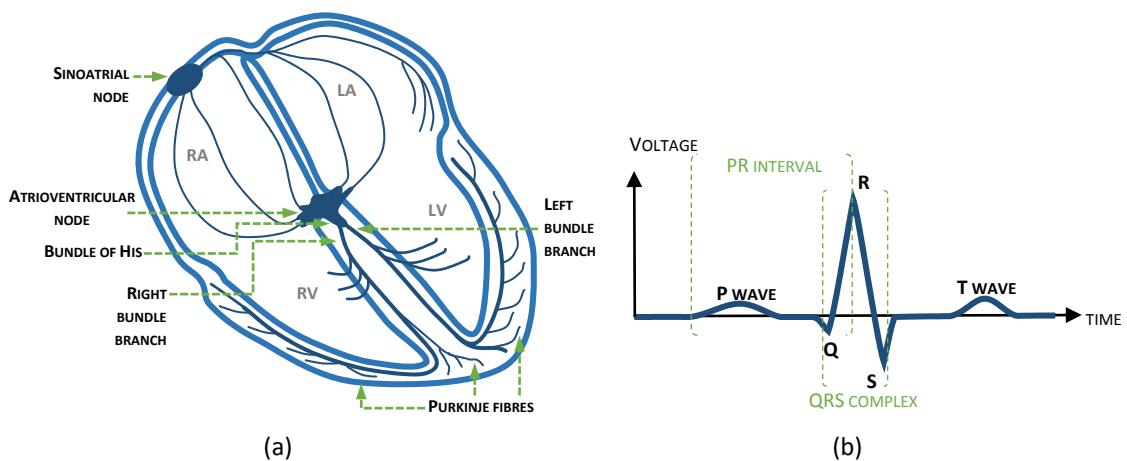


Figure 2: (a) the electrical conduction system, and (b) an annotated ECG signal. The form of the signal changes depending on the location of the measuring leads. The P wave represents atrial systole and QRS represents ventricular systole. At time R, the mitral valve closes and at time S, the aortic valve opens and closes again late into the T wave. At the end of the T wave, the mitral valve opens.

Ventricular remodelling is a phenomenon with geometrical and functional effects – most commonly – on the left ventricle which is associated with a poor ejection fraction [2]. Whilst remodelling can occur during growth, it is most commonly a result of acute myocardial infarction – other causes being chronic hypertension and valvular heart disease [6]. Ventricular remodelling is a homeostatic response to an increased wall strain resulting in fibrosis of the myocardial muscle and an associated wall thickening (decreasing wall stress), leading to ventricular size and shape changes. This process dilates (inside volume) and hypertrophies (wall) the ventricular chamber, tending towards a spherical shape. It accomplishes the maintenance of ventricular function at the expense of a lower ejection fraction. Ventricular remodelling is a progressive process in the sense that it is both continuous (triggered by just a single insult to the

myocardium) and cumulative (escalating effects with increased causes). It is also a recursive process despite pharmacological treatment – although this might decelerate the rate of worsening [7]. Patients with a remodelled ventricle might not initially complain of any of the associated symptoms but these do worsen with time. This is because LV reverse remodelling does not predict symptomatic response however it does improve cardiac performance and hence likelihood of a prolonged life expectancy [6]. Whilst LV remodelling is a chronic and essentially irreversible disease (see Figure 3), reverse remodelling is possible to an extent [7].

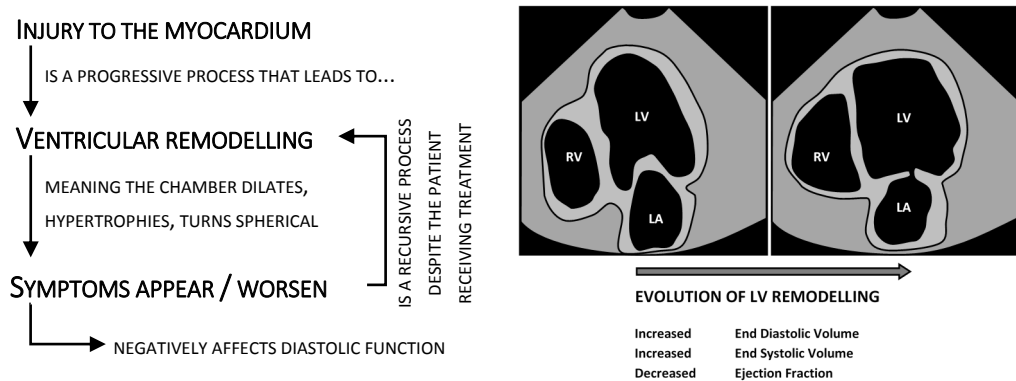


Figure 3: The recursive process of ventricular remodelling. Schematic representation (left) and its effect on chamber size and shape after MI (right)

Cardiac de-synchrony refers to the asynchronous contraction of the ventricles, which is most commonly due to a delay in left ventricular activation. Causes of cardiac de-synchrony may be due to fibrosis in cardiac tissue, damaged Purkinje fibres or scar tissue (typically after myocardial infarction) affecting the propagation of the electrical pulse that initiates ventricular contraction [8]. Cardiac de-synchrony affects the efficiency of the heart's work (as well as lowering cardiac output) by disrupting valve timing (similar to the tuning of the timing in a car's engine valve opening). Cardiac output is reduced and wall strain increased as the heart's workload increases. This condition causes a delay in the QRS period – measurable using a standard ECG. The heart undergoes remodelling to reduce the extra strain caused by de-synchrony. Ventricular de-synchrony is hence strongly associated with reduced EF, large bundle scar, increased wall strain and increased LVESV.

### 1.1.2 Cardiac Resynchronisation Therapy

Cardiac Resynchronisation Therapy (CRT) attempts to correct the delay in activation time to help the heart recuperate. By installing an atrial-synchronised bi-ventricular pacemaker, clinicians can overrule the AV node pacing directly at the site of contraction. On-site stimulation has the

advantage of bypassing the injury causing disruption, achieving normal heart efficiency. It is worth noting that since the rhythm the heart beats (i.e. SA node firing) changes to adapt to physiological demand, CRT is only effective under certain physiological conditions meaning a patient fitted with a bi-ventricular assist device cannot be expected to, say, run long distances or carry out exhausting work. In order to maximise response rate and lower risk, patients are carefully selected for CRT upon meeting the following requirements [9]:

- NYHA stage III or IV;
- QRS delay > 150 (or >120 and dyssynchrony shown by ultrasound);
- Reduced LV ejection fraction (<25%).

It has been recognised that the benefit of CRT is on a par with that of widely accepted drugs for use in patients suffering from HF, such as B-blockers and ACE inhibitors. Due to strict inclusion criteria, the proportion of patients with HF likely to be eligible for CRT has been reported to be 15% to 20% of the patients observed in specialized HF clinics. It is important to note that, despite the aforementioned strictness in the inclusion criteria to be considered for CRT, only two-thirds of patients respond to the treatment [9].

CRT improves patients' quality of life, increases exercise capacity, reduces hospitalisation and repairs the under-lying cardiac injury. Quality of life is measured using the Minnesota Living with Heart Failure Questionnaire (MLWHFQ) – where patients answer a series of questions on their perceived medical improvement and ability to perform tasks. Exercise capacity is seen by an improvement in NYHA functional class and by an increase in the 6-minute walking test results. The 6-minute walking test is the total walking distance travelled during the allowed time – generally performed on a treadmill – complementing the subjectiveness of NYHA classes.

Clinical measurements of improvement are decreased LVESV – threshold of successful response being  $\geq 15\%$  - and increased LVEF – usually around 3% with successful therapy; although this has highly variability and doesn't necessarily recover to healthy values. It has been suggested that short term (1 year) LV reverse remodelling is sustained in the long term (5 years) in "most" patients [10], or 83% for symptomatic non-responders and 87% for responders [6]. Further benefits of CRT, compared to receiving no treatment, include [11]:

- Improvement in the MLWHFQ by 8pts
- Around 20% more patients improve at least one NYHA functional class within the first 6 months
- Decreased hospitalisation by 37%



- Decrease all-cause mortality by 22%
- Implant success rate was 93% and 0.3% peri-implantation death

### 1.1.3 Imaging techniques for clinical assessment

Measurements of chamber volume and shape, such as for the clinical assessment of patients described in sections 1.1.1 and 1.1.2, are performed using medical imaging techniques; these include ultrasound imaging (also called cardiac echo, echocardiography or sonogram), magnetic resonance imaging, X-ray imaging and computed tomography. The use of magnetic resonance imaging is discussed in section 1.2.2 however an extensive discussion of imaging techniques is beyond the scope of this work.

From a clinical perspective, these techniques are assessed in terms of ease of acquisition, image quality and contraindications. Ultrasound imaging is generally risk free, however overuse is discouraged (as is any type of over medication and medical testing). The safety of magnetic resonance imaging is considered to be similar to ultrasound imaging; in both cases, the use of contrast agents does have contraindications. Ultrasound has low image quality (noisy) but is a non-invasive technique, fast to perform and allows for relatively portable devices (including hand-held). The quality of magnetic resonance images varies between applications but generally is much higher than ultrasound imaging. It is a non-invasive technique however acquisition requires many members of staff to be present and takes a considerable amount of time. The equipment used may not be accessible to patients with severe physical illness or that suffer claustrophobia. X-ray imaging and computed tomography (which uses X-radiation) are considered carcinogens and generally avoided for patients with certain conditions (such as pregnancy) however for patients with severe medical conditions or when used sparsely, the exposure to radiation is outweighed by the benefit of better diagnosis. Both techniques are faster to perform and more accessible than magnetic resonance imaging.

In addition to image quality, medical images differ in spatial dimensionality and resolution, and in the content imaged. Ultrasound has trouble imaging through bone and distinguishing between different types of soft tissue, it has a low penetration depth and low field of view. It may however be used to measure fluid flow. It can achieve real-time, high spatial resolution (although not for imaging the beating heart), and may produce 2D and 3D images. Magnetic resonance imaging allows imaging all types of soft tissue, bone and fluid. It has a high field of view and contrast is excellent for different tissue types, but may not be acceptable for tissue of similar density. It is commonly used to produce 2D, 2D+t and 3D images, but can also be used to produce 3D+t images. X-ray imaging is generally used to image bone whereas computed

tomography can be used to image soft tissue as well as bone. X-ray imaging produces a 2D projection of the object imaged whereas computed tomography produces high quality 3D images with high spatial resolution.

Clearly, there are benefits and challenges with the different imaging techniques available, and this trade-off carries on to the diagnostic use of the images. 2-dimensional images are easier to interpret however they may not show all details of interest. 3-dimensional images are harder to interpret and may require specialist software and processing time to produce a correct analysis. Fast, cheap and accessible imaging techniques are usually favoured however they may not be appropriate for diagnosis. For example, it has been documented that it is not appropriate to employ ultrasound images (or any 2D image modality) for measuring ventricular end-diastolic volume because the image does not allow an accurate measurement of the 3D volume [12]. To this effect, the values for abnormal EF and normal LVESV quoted in sections 1.1.1 and 1.1.2 are a reflection of the conservative thresholds that need to be employed in to avoid inaccurate diagnoses. Consequently, normal EF is considered to be  $> 50\%$  [13].

Another important aspect of clinical measurements using medical imaging is inter- and intra-observer subjectively at the time of interpreting medical images, which is relevant but out of the scope of this thesis [14], [15]. In brief, medical imaging is a fundamental tool for clinical diagnosis however medical image analysis is limited by human interpretation. Automatic computer techniques can achieve higher accuracy but, at present, are not able to perform the same type of complex analysis as humans. Computational image analysis allows removing the “human factor” and is the starting point for the engineering analysis techniques described in the following section.

## 1.2 The Simulation Workflow

### 1.2.1 Overview

The basic workflow of modelling, in the medical context, is the process of construction, execution and analysis of a model that represents an individual patient. The models in this project include patient-specific anatomies extracted from medical images and material properties and boundary conditions that might be associated with information in the clinical record. The steps in the modelling process are summarised in Figure 4. A model comprises both the geometrical mesh and the laws of physics that will govern its behaviour. To create the geometrical model, the image is segmented to isolate the object of interest from the rest of the image. A simulation (or number of) is performed to gain more understanding of the functioning

of the patient’s organ or to predict the outcome of therapy, and finally, an analysis of the results is presented back to the clinical staff.

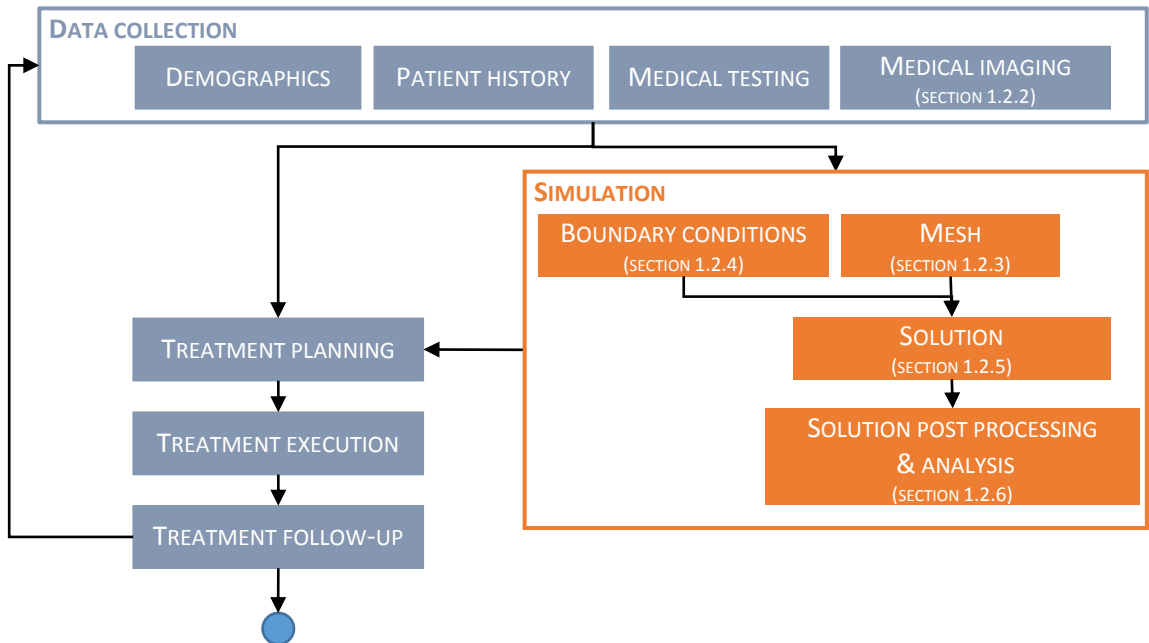


Figure 4: Conceptual flowchart for simulation. Medical specialities collect different information in the patient history, as they find relevant. Example of medical testing include blood, urine and hormone tests.

## 1.2.2 Imaging and Segmentation

Medical imaging is one of the most important tools used during patient diagnosis and treatment. The human body can be imaged using a number of techniques, depending on the object of interest, and is often limited by technical challenges. In the Grand Challenge project, a large dataset of images was collected using magnetic resonance imaging (MRI).

Magnetic resonance imaging (MRI) is a technique used to image the human body. It can also be used to assess the function of the cardiovascular system in a non-invasive manner. The principle of MRI uses a magnetic field to energise  $1H$  nuclei. When the nuclei release this energy it is detected and measured. The position of the emitting nuclei can then be reconstructed. The concentration of  $1H$  nuclei varies across different tissue types and the amount of energy emitted varies accordingly. The data is then sampled in a regular grid, typically using a non-linear interpolation scheme, which in turn becomes the image voxel grid [16].

Image quality is assessed by:

- the accuracy of the image, which determines the faithfulness of the image-representation of the physical object;
- the contrast of the image, which allows the different structures to be distinguished;
- the signal-to-noise ratio, noise is the manifestation of grain-like defects scattered randomly across the image;
- the amount of blur, which distorts sharp edges of the physical object.

The cardiovascular system periodically pulsates due to the natural beating of the heart with consequent transmission of pressure and flow waves through the vessels. Images, particularly those based on magnetic resonance imaging, take time to acquire, and the averaging effect over the cardiac and respiratory cycles can produce significant artefacts. Cardiac motion artefacts can be minimised using imaging sequences that are aligned with, or triggered by, the subject's ECG signal; this is called ECG gating. A commonly used sequence to acquire images (based on this principle) is called balanced steady state free precession (bSSFP) however different manufacturers implement bSSFP differently, such as TrueFISP (Siemens), b-FFE (Philips) and Fiesta (GE).

In addition to motion, other factors that contribute to image quality degradation include under-sampling and using a low power magnetic coil. Typical scanners use a 1.5 Tesla coil, however 3 Tesla scanners are becoming increasingly available in developed countries and higher power scanners have been developed. Increasing the density of the sampling can reduce the signal-to-noise ratio, but also reduces the field of view of the image.

Images may be 2- or 3- dimensional, they can be static or dynamic. The gradients of the magnetic field are used to create a representation of the domain in k-space (frequency domain) and then mapped into either a slice or volume (spatial domain). Typically, for static images, the heart is imaged during diastole, because it is both at rest and is a larger object to image. Images acquired at multiple time points in the cardiac cycle, based on the ECG gating, can be concatenated to produce cine images (4D, or 3D+t).

Differentiation of thin connective tissue (such as the different layers of the heart) and fat is often tricky and represents a major challenge in cardiac imaging. Injecting a contrast agent into the patient's blood is a common method of imaging particular structures of the heart. The use of contrast agents is a highly controlled procedure due to the complications that might arise were the patient to have an allergic reaction to the agent. For non-critical imaging purposes, the use of agents is discouraged or even forbidden. Contrast agents injected into the blood pool can improve the contrast enormously, and are routinely used for imaging of the cardiac chambers

and the vasculature. Contrast agents are transported by the blood into tissues during perfusion. Due to the fact that different tissues are perfused at different rates, this imaging technique enhances the contrast between similar looking tissues. For example, scar is identified at non-functional muscular areas because they are not properly perfused. Fat is identified as low-density tissue, and blood pools behave as high-density tissue.

Assessment of cardiac function is typically a derived measurement from a set of images. Comparisons may be related to the absolute volume of a structure and change (or rate of change) of volume of structures. Comparisons may be made across images acquired during resting and stressed conditions. For logistical reasons, stress is often induced using drugs. Quantification of perfusion (and rate of perfusion) can also be a useful measure.

In routine clinical practice, imaging ill patients is problematic because of particular pathologies that contribute to image degradation, such as:

- high, unsteady heart rate;
- high amount of fat coating the muscular structures;
- some patients are disproportionately large, making them unable to fit inside the machine;
- inability of the patient to hold their breath or to lie still in the machine;
- unsuitable to imaging with contrast agents due to an allergy;

Quite often, the patient might suffer claustrophobia due to the conditions inside the imaging device. In patients with advanced heart disease, patients are likely to have been fitted with artificial implants, such as pacemakers and cardioverter-defibrillators. MR cannot be used on patients with these implants (artificial implants create artefacts on CT scans).

Scans require the availability of a scanner and skilled technicians to acquire and interpret the images. The cost of MRI is high when compared with other imaging techniques. Whilst a trained human technician is able to identify the different structures in images that often have medium to low quality, image quality is a major drawback in image processing.

Figure 5 and Figure 6 are images taken for clinical diagnosis as part of the study. Figure 5 shows two type of cine images (time varying, planar images) and Figure 6 shows two slices through a volume image of the thorax. Acquiring planar images is usually a faster procedure than the acquisition of volumetric images, however modern imaging techniques make the latter a process that takes less than half a minute. Figure 5 has better contrast and time resolution than Figure 6, which has a greater field of view and an extra spatial dimension.

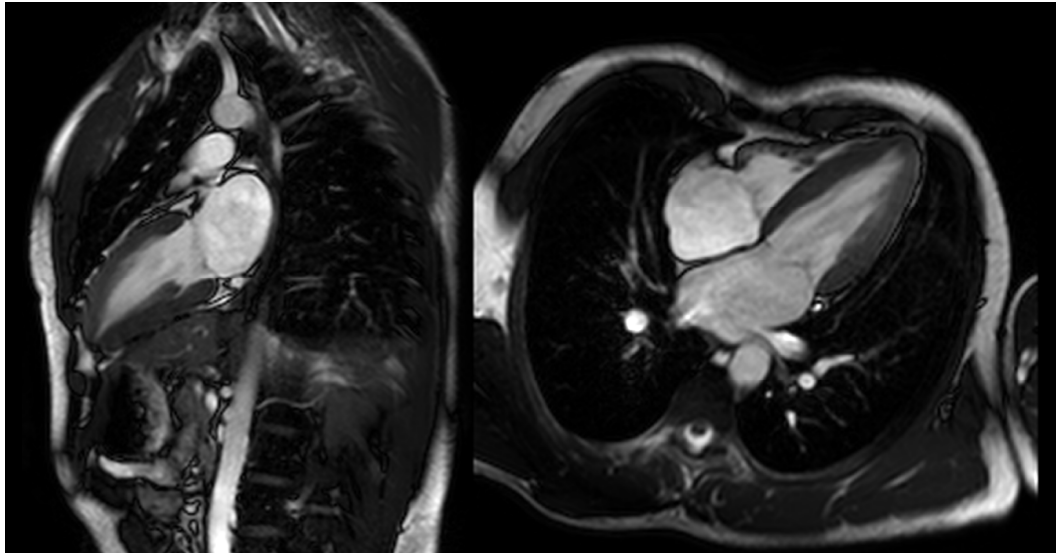


Figure 5: Different views of the heart during ventricular systole using cine MRI. The image on the left is a 2-chamber view, showing the left ventricle and atrium. The image on the right is a 4-chamber view.

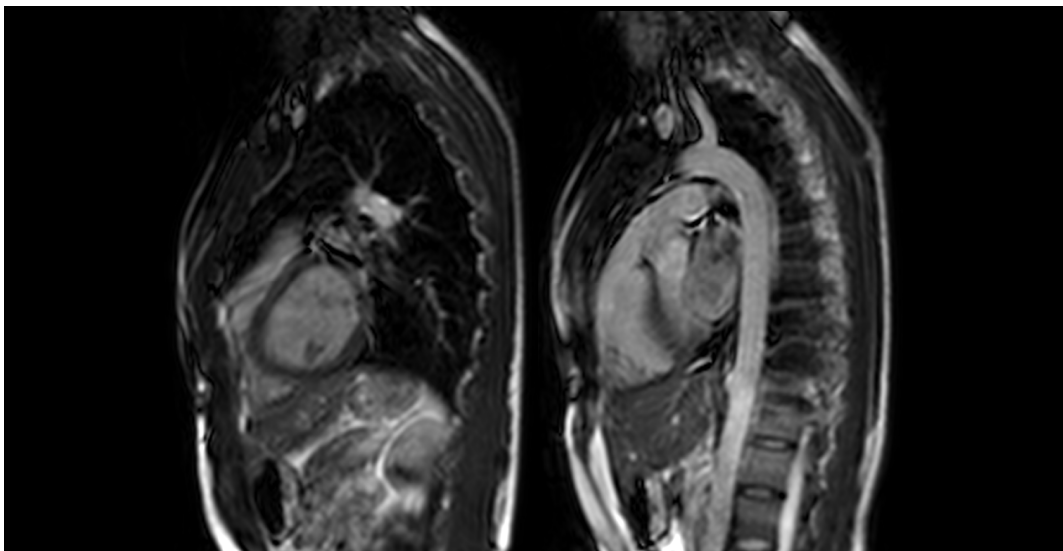


Figure 6: 3D MRI of the heart. The image on the left shows the left ventricle of the heart; the image on the right shows the left and right ventricles, the left atrium and part of the aortic arch. Both slices are part of the same 3D volume.

Image segmentation is the identification of distinct objects that appear in an image. In particular, it is the selection of pixels that correspond to the same physical structure, meaning that the output of the process is an array of ones and zeros (a binary mask). A more useful representation are labelled masks, which allow storing segmentations of different objects in the same image object by using different colour intensities. (The difference here is using integer values instead of binary values to populate the pixel array.) Another benefit of labelled masks is that they allow regions of overlaps. Nevertheless, it is common to represent a segmentation using a surface function that encloses the pixels of interest.

Image segmentation is an essential tool in image processing and recognition. ([17], [18]) It tells computers (and users) what objects appear in an image and its location. It can be the result of, or a means to shape and object recognition. It is essentially the computational equivalent of object identification, which is an instinctive process to humans. Upon successful segmentation, the derived mask can be used to extract information about the objects in the image.

Image segmentation is an important and active area of research in computer science and biomedical engineering. Methods for image segmentation can be broadly divided into two categories: manual and automatic segmentation. Methods that require manual initialisation are typically considered automatic, whereas manual methods that are computer assisted (real time corrections) are considered semi-automatic.

Manual segmentation methods are entirely driven by a human operator and as such, are considered the gold standard when performed by a skilled operator. These methods differ by the way the operator interacts with the computer, such as contour delineation, colour fill or point selection to construct an enclosing polygon. For 3D images, these methods can only be performed in a slice-wise manner.

Manual segmentation has the benefit of the power of human interpretation to infer data that is not visible and discard noise. On the other hand, it is very sensitive to inter- and intra- operator variability and is extremely time consuming for 3D images. Inter-operator variability refers to different human observers interpreting images differently; intra-operator variability refers to the fact that the same operator might change its interpretation of the image in a repeat attempt. Despite the power of human interpretation and inference, manual segmentation requires highly trained operators. Visual aids, such as visualising multiple planes in a volume at the same time, help operators navigate through the volume and therefore improve interpretation.

Automatic methods are otherwise general-purpose algorithms that method building applications to segment specific structures from well-defined images. These methods include thresholding pixels based on their intensities, landmark or edge detection algorithms (e.g. Sobel filter, difference of Gaussian), shape matching (e.g. Hough transform), shape morphing (e.g. snakes, deformable models [19]), seed growing algorithms [20] and registration of template images [21], [22].

Automatic methods may be faster to complete than manual methods for large datasets, and define/rely on immovable definitions for what are the structures to be segmented, thusly removing the human variability factor. When these definitions are well defined, automatic methods achieve greater precision (for comparison, a human might be disorientated when

zooming too far into an image); on the other hand, they are less accurate (for example, they might be misguided by artefacts in the image).

Automatic methods may be driven by a human operator, in which case they are considered semi-automatic, or interactive. [23] An example of a semi-automatic method are live wires, where the operator draws lines, or “wires”, on the image and the wires automatically grow into adjacent pixels based on their intensities. [24] Automatic segmentation tools can become very robust tools when manually driven or manually initialised. This is particularly useful when the target image cannot be characterised fully due to, for example, noise, poor contrast or overlapping objects.

Figure 7 shows an automatic image segmentation driven by registration of pre-segmented templates [22], discussed in chapter 6, section 6.2.2. The tool produces a labelled mask where different intensity values represent different structures of the heart; in this form, the image is much easier to interpret by both humans and computers than the raw medical image.

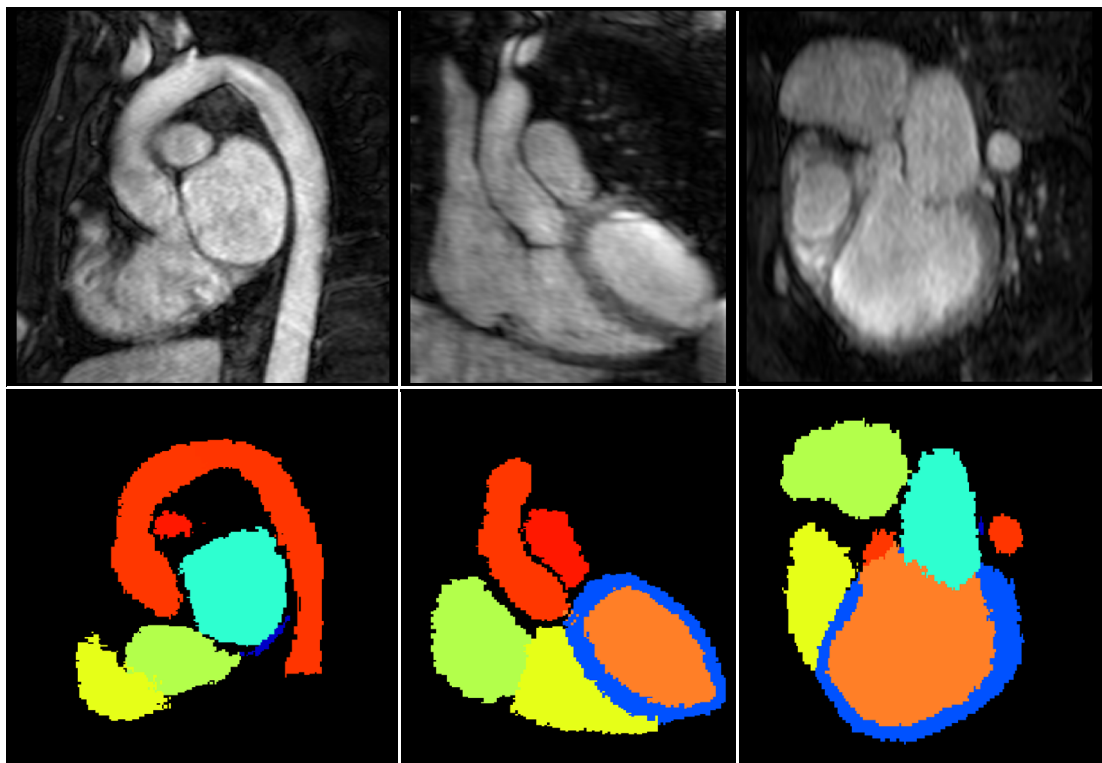


Figure 7: 3D MRI of the chest (left) and segmentation of main structures (right) presented using a labelled mask. The segmentation was acquired using the tool described in [22] and shows the left ventricular myocardium (blue), left ventricular volume (green), right ventricular volume (yellow), right atrium (orange), left atrium (turquoise), aorta (red) and vena cava (red).



### 1.2.3 Models and Meshes

Computer simulations are used to model real world phenomena. Examples of computer simulation methods include the finite element (FE) method, used for structural, thermal and electrical analysis, and computational fluid dynamics (CFD), used to study fluid flow. Generally, the systems are described by a series of differential and algebraic equations that enforce conservation of some properties such as energy, mass or momentum. [25] Only rarely, usually for geometrically simple systems, do the equations yield analytical solutions, and instead they are discretised by sampling at points within the domain to make them tractable for computational solution. Discretisation approaches include finite difference, finite element and finite volume methods. The relationship between displacements or velocities and applied stresses are described by constitutive equations that represent the material properties. The system is completed by a set of boundary conditions. In computational biomedicine, the combination of material properties and boundary conditions can be used to describe the unique physiological conditions to make the model patient specific. There is always a balance between the desire to make the model as accurate and patient-specific as possible and the requirement to make increased number of measurements (sometimes invasive) to achieve this accuracy. [26] In principle, the recommended approach is to make optimal use of those data that are already collected in the course of the clinical workflow for the patient. Additional measurements not only require specific ethical approval, but they are often costly and should only be made when they add significant value to the research study. Analysis protocols that require additional measurements will be more challenging to introduce into clinical practice; this is especially true when such measurements are invasive.

Computational modelling is part of the standard industry practice in many fields of engineering (for familiar examples think of the motorsports (Virgin F1 team) and aerospace (Airbus A380) industries). [27] It is important to remark that it does not fully replace physical prototyping and testing but it does permit improvements in the design of products and optimisation of processes by providing greater insight into the physics of the system. In the medical field, engineering concepts such as stress analysis and CFD aided design can be applied to clinical conditions resulting in, for example, relationships between material elasticity and fluid flow with cardiac tissue strain, obstruction of blood flow and impaired perfusion. Computational models can deliver increased confidence in diagnosis and prognostic indicators by improving understanding and by predicting both the time course of the disease and the potential effects of interventions (including, potentially, cardiac resynchronisation therapy). Once the models are appropriately validated, they can 'measure' quantities that cannot be measured directly, with high spatial and

temporal resolution. Furthermore, they are intrinsically safe as they involve no physical intervention on patients, although of course they might use information that comes from such interventions.

One of the challenges in the planning of modelling workflows for clinical application is to make optimal use of the data that are already collected as part of the clinical pathway. A model should capture the appropriate level of detail at the appropriate complexity. In general, 0D compartment models are useful to provide information on overall distributions of parameters such as pressure, flow or species concentrations, and these might be completely adequate for some purposes. 0D (and 1D) models are increasingly used to provide boundary conditions for locally-detailed 3D models. 1D models are used for the arterial system where knowledge of the distribution throughout the vasculature is required, and particularly when wave transmission effects, including reflections, might be important. They are also sometimes used as transition components between 3D and 0D model compartments. Detailed reviews of these models are provided by van de Vosse and Stergiopoulos [28], and by Shi et al. [29].

In the context of the application of models to heart failure, Shi et al. developed a numerical lumped model of the cardiovascular system with a ventricular assist device [30]. Parametric tuning was manually performed until it matched average physiological measures. Whilst this was a generic model it indicated a path towards patient specific therapy optimisation using a model of reduced dimensions. Furthermore, it provides excellent groundwork for the model to be coupled with higher order models to determine appropriate boundary conditions. Its parameters, whilst having physiological meaning, don't relate to easily obtainable measurements – this being a main limitation to its use. Kim et al. successfully coupled a 0D heart model with a 3D CFD aorta model showing that, regardless of complexity, areas where detailed solutions are not required can be reduced to lower dimension models [31]. The authors created a lumped model of the heart which was tuneable to different cardiac conditions (rest and exercise) and showed the benefits of modelling by components by having two models for the aortic arch (pre- and post-intervention). They demonstrated that the detail of the 3D model is not compromised by the coupling to the lumped model in fact, constraining the boundary conditions created a robust and stable model. However, tuning the parameter values was blinded to physiological measures making it a complex and time consuming task.

3D/4D models provide detailed information on the behaviour of structures and fluids. There is always a danger that increasing the complexity of a model, whilst theoretically improving its fidelity, introduces many new challenges including the need for additional measurements for parameter tuning, often leading to reduced numerical stability and inevitably extended

(sometimes daunting) computational resource requirements. One of the motivations for the work presented in this thesis is the improvement of mesh quality in patient specific cardiac models because it has been demonstrated that the complex nonlinear structural models that are executed within the CRT analysis workflow are particularly sensitive to mesh quality [32]. Figure 8 shows example meshes used by the simulation group in the Grand Challenge project.



Figure 8: Representative mesh geometry of the heart ventricles. (Adapted with permission under Creative Commons BY-SA 3.0 from [33], formatted horizontally for better presentation.)

#### 1.2.4 Material Properties and Boundary Conditions

Material properties can be determined empirically (perhaps known prior experience), deduced from first principles or directly measured. The argument for using approximated values (such as population averages or measurements made on a patient at an earlier point in time) for material properties and boundary conditions lies on the ability to collect more precise readings and the level of confidence in the measurements. On the other hand, the simulation needs to be specific enough to describe the feature of interest correctly.

It is common practice to make a number of assumptions during the simulation process to reduce the complexity of the problem, thus reducing the expense of the computational task. Another argument for making assumptions is being uncertain about the correct properties of the system. Moreover, the effort to collect data (acquisition and processing) involves patients and medical staff, and requires specialised equipment and expertise. Patient specific properties should be used if they significantly impact on the engineering analysis. Using erroneous parameters can lead to making the simulation less specific to the patient. Despite successful simulation of healthy conditions, representing diseased behaviour introduces new challenges in terms of boundary conditions and model complexity.

In CFD analysis, the most obvious boundary condition is the wall. The wall enforces the condition where no fluid can pass through; where the fluid is in direct contact with the wall, it is stationary (the no-slip condition). A model's geometrical boundaries can be approximated using a generic

geometrical model from our knowledge of human anatomy. For a personalised simulation, the geometry is determined using medical imaging through the process of image segmentation, which is a large area of research. A typical processing pipeline of geometrical information involves image acquisition (e.g. cardiac MRI scan), imaging filtering, structure identification, image segmentation and image registration.

Patient specific models require extensive data to be acquired, specifically in vivo measurements which present obvious limitations for clinical uptake. In vivo measurements provide precise raw measurements increasing model accuracy but carry important draw backs, such as expense, clinical centre capability and health risks. In addition, in vivo measurements taken under surgical conditions limit the physiological conditions that the patient can be subjected to. Alternatively, data may be approximated using generic trends based on patient demographics, estimated using measurements from clinical images or inferred from secondary measurements.

Coupled electrophysiological and biomechanical models have been developed to simulate cardiac behaviour [34], [35]. The geometry of the models is extracted from volume MR images as shown in Figure 9. Similarly, a structural model of the heart might be coupled with a pattern of electrical activation, blood flow model and the simulation of a pacing device. Sermesant et al. obtained CT, MRI images and in vivo measurements, and estimated fibre direction by fitting data from a generic database to patient information [34]. In addition, despite large efforts to obtain a patient-specific cardiac structure, electrical activation maps and fibre orientations were based on generic trends. With their models, the authors were able to visualise electrical conductivity in time and space and to predict flow measures (volume discharge and pressure).

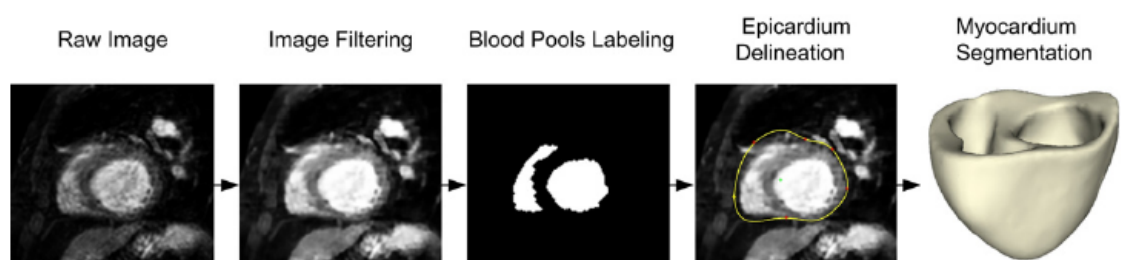


Figure 9: Image preparation and segmentation process by Sermesant et al. A volume MR of the thorax is acquired but filtering is required to identify structures, the ventricular blood pools in this case. With this information a surface representing the ventricles is automatically generated and is then manually corrected. (reproduced with permission from [34])

Auricchio et al. used in vivo measurements to characterise LV activation in patients with heart failure and left bundle branch block (LBBB) [36]. Precise reproduction of the activation wave maps was achieved by adding ECG data to the in vivo measurements. Despite the fact that no

geometrical information was collected, the authors showed the effect of LBBB on activation pulse propagation and hence presented a tool to improve lead location for successful CRT. The potential of their work is limited by the need for contact measurements to achieve accurate maps. Their work however provides evidence of the importance of the electrical activation pattern and the ability to use this to precisely locate scar tissue – a very important consideration for CRT planning. Deftereos et al. showed that scar volume is higher in CRT non-responders than in responders [37].

Niederer et al. addressed the issue of incomplete selection criteria parameters by building patient-specific models and performing a parameter sensitivity analysis [35]. Niederer's models were created from patient cardiac MRI scans and special attention was taken to reproduce myocardial scar on the cardiac tissue. The models were coupled to electrical activation maps allowing a view of the propagation of the activation wave across the surface. The authors found that non-responders could be determined by selecting patients with low or no wall tension regulation.

Whilst taking extensive invasive data of the cardiac activation, Niederer et al. made use of generic activations maps acquired from commercial software, and Auricchio concluded that invasive data is crucial for precise patient specific activation functions. Due to the requirements of invasive data it is unfeasible to transfer this technology into the clinic as it stands.

### 1.2.5 Solution

Solving the governing (equilibrium or conservation) equations that describe the physics phenomena that are to be simulated requires solution of large systems of partial differential equations and numerical methods are employed. In finite element and finite volume methods, degrees of freedom are defined at discrete points (nodes) and the material constitutive equations are used to relate the physical characteristics by which the nodes interact (relationships between stress and strain, or stress and strain rate). Other modelling techniques, such as agent based modelling and molecular dynamics will not be considered in this thesis. Historically, computational structural analysis has usually been performed using the finite element method whereas the finite volume method has more often been applied for modelling fluid flow, although there are several finite element fluids analysis codes. Other physics phenomena such as heat transfer and electromagnetism are typically modelled using the finite element method. [38]

Discretisation is the process of taking a continuous function and describing it at distinct, or discrete, points. The governing equations are discretised in both space and time, with an associated accuracy and precision.

Discretisation of the governing equations in time is required to produce a numerical solution that can be solved. It involves the application of a numerical approximation of the derivatives in the equations. The temporal derivatives are usually written in finite difference form, and the accuracy of the temporal discretisation scheme is related to the number of terms used in the Taylor series expansion; for this reason, the error of discretisation schemes, expressed as powers of the time step is called the truncation error. For 1<sup>st</sup> order approximations the Euler method is used; this can be thought as an extrapolation using a first order Taylor series approximation and an error of the order one time step.

In an explicit method the continuous equation is discretised in such a way that there is an explicit equation for each degree of freedom at a time step as a function only of variables computed at previous time steps. In implicit methods the discretisation produces simultaneous equations for the degrees of freedom at the new time step that need to be solved together. 1st order forward and backwards discretisation are referred to as Euler methods. Explicit methods are easier to implement and solve than implicit methods, however they are less stable.

Central difference schemes use information from time steps prior and posterior to the current time step. In comparison to forwards and backwards schemes, central finite difference schemes achieve a greater degree of accuracy for the same number of Taylor series terms used. This becomes important to consider when extending the Euler method to higher order numerical approximations; alternatively, Runge-Kutta methods can be used.

The precision of the discretisation scheme is associated with the time step used, and to a lesser degree to the computer's floating point precision. An important consideration regarding time step length is that if the time step is too large the solver might not converge. On the other hand, the computational cost of small time steps may be prohibitive. Whilst the transient precision of the simulation can be increased by reducing the simulation time step, it is essentially dependant on the physics of the problem and the grid spacing.

Discretisation in space is performed by the generation of a computational mesh (section 1.2.3); usually the material properties are defined in elements and the degrees of freedom are usually defined at the nodes. The accuracy of the discretisation is related to the element type used (linear or nonlinear) and the precision is related to the element size. An element type describes the shape function used for spatial discretisation between the nodes. Different shape functions

may be used to describe the variability of the nodal properties and discretisation of space. Isoparametric shape functions are the special case when the shape function that describes the geometry of the element as a function of the nodal values of the co-ordinates is the same as the function that describes the displacement throughout the element as a function of the displacement at the nodes.

The main challenge of the meshing process is accurately capturing the shape of the domain being simulated using good quality elements. The quality of mesh elements is a property associated with the relationship between the local (or parametric) variability of nodal properties inside elements and the global nodal values.

The use of smaller element sizes may produce a more precise discretisation of the geometry as well as better quality elements, but in turn produces more points at which the equations need to be solved, driving up the computational cost of the simulation.

Solving single physics equations may require solving multiple governing equations. A tightly-coupled solver is one in which all of the degrees of freedom are expressed in one monolithic set of equations. A segregated solver is one in which they are separated and solved independently, whether sequentially or iteratively. When there is coupling between domains, the same degrees of freedom exist on the border between the domains. For example on the coupling between a 3D and 1D fluid model, both include pressure and flow at the interface. In an explicit method the pressure in the 3D domain at the interface within a time step might be fixed, determined by solution of the 1D equation at the previous time step. In an implicit method the pressure might be updated within the time step by operation of the 1D model, which is computationally expensive.

Computational structural analysis comprises the determination of displacements at discretised points in structures. The equilibrium equations can be described in terms of a minimisation of the total potential energy of the system, made up of the work done by the external forces and the internal strain energy. Computational fluid dynamics involves the determination of velocity field at discretised points by integration of the equations for the conservation of momentum, mass and energy, based on the flux of these variables in each element. For this type of analysis, some schemes solve the equations for momentum and continuity independently and use pressure-correction functions to attempt to balance the results at each time step. Famously, the SIMPLE (Semi-Implicit Pressure Linked Equations) algorithm, and variations thereof, have been employed through the years.

As described in section 1.2.4, the conservation equations are solved for a set of boundary conditions. Boundary conditions may be specified, possibly time-varying, values for displacement, velocity, pressure or derived properties. However some physical scenarios require the boundary conditions to be computed by another model, including the possibility of being comprised of a reduced order model (e.g. a Windkessel model). In such scenarios where the boundary conditions need to be resolved, implicit or explicit coupling between the separate models might be used. Implicit coupling schemes update the boundary conditions during the time step to provide a converged solution for the whole system at the time step. In explicit schemes, the boundary condition is updated a-posteriori to the 3D domain.

Simulations where multiple physics phenomena are considered are called multi-physics simulations. Since each physical domain is described using different governing equations and may be solved using different discretisation and coupling schemes, the systems are solved independently and the environments are updated to reflect the interaction between each domain. An example of multi-physics simulations are fluid-structure interactions, which involve both computational structure analysis and computational fluid dynamics.

An important technical consideration is that, if the different solvers use separate meshes of independent 3D domains (as often the case in Fluid-Structure Interaction), the coupling is simpler if the meshes are conforming. Conforming meshes share nodes at the boundaries where they interface, as opposed to a node from one domain lying on the face of an element in the other mesh. Modern solvers use interpolation schemes to deal with non-conforming meshes but these may make a number of assumptions when transferring the properties across domains that need to be considered. Further complications may arise if the quality of a mesh degrades due to the deformations produced by the coupled simulation. [39]

Solving the large system of equations governing the physics involves finding the correct solution within a vast parameter space. Analytical solutions to the equations do not exist for most, if not all practical situations.

Direct methods for solving, such as Gaussian elimination and matrix decomposition methods, whilst providing exact solutions to the discretised equations (within numerical precision), are computationally unfeasible for large models in terms of large memory requirements and the number of computational operations required. For this reason, iterative methods are often preferred. These provide an approximation that converges towards the exact solution if stable, which is not necessarily the case in the applications described in this section. [40]



Iterative methods are also simpler to implement in a parallel fashion, which considerably speeds up solving the equations. There are three types of parallelisation options for computer algorithms that are used for computer simulation codes. Single-processor architecture in which multiple threads are used are faster and simplest to implement but are limited by the power of the single computer that is used. A similar limitation is encountered if execution is shared across multiple CPUs, using either shared or distributed Random Access Memory and high speed bus communication.

A more sophisticated parallelisation scheme uses network communication between computer clusters to share data across threads. This allows cheaper construction of a parallel environment to execute the simulation however are bottlenecked by the speed of network communication and the communication protocols used, as the threads must synchronise relatively often during the simulation.

### 1.2.6 Analysis and Interpretation

Using mathematical equations to model cardiac tissue has the disadvantage that variables have unrelated physiologic meaning. A model derived from on physiological data familiar to, and routinely collected in, health centre environments would be of increased interest as it would simplify modelling building and increase usability. This is a similar issue to the lack of established relationships between cardiac performance, clinical measurements and patient symptomatic response (i.e. the vast amount of data that can be derived from models may not add to the clinical understanding of diseases). Computer simulations have the potential to use routinely collected data to model patient-specific behaviour and broaden our understanding. Simulation results will aid diagnosis by providing clinicians and researchers with information that is currently unavailable in a consistent manner (using computational algorithms to merge information, whilst a hard task, removes the subjectivity of human interpretation).

There is a two-fold advantage of computer simulations: personalised analyses and bulk processing. In other words, not only can the physiology of individual patients be better understood, but we now have tools to analyse big datasets representing larger proportions of the population. This will undoubtedly provide a greater understanding of the functioning of the human body. Given that the techniques that are now available to analyse data have not been used in medicine before, and that our ability to collect data continues to increase (as does the quality of data), it is hard to imagine the limits of computerised biomedicine.

Current clinical knowledge can be directly translated into the analysis of the results of simulations. For instance, outcomes derived from the PROSPECT study include the importance

of intraventricular dyssynchrony for successful CRT response (QRS duration > 120ms) and the absence of a single parameter that fully defines this through echocardiography [41]. Imaging shows that high total scar bundle (>1) near lead placement is directly related with failure to respond to CRT. Instead, they suggest that leads should be placed directly in areas of late response. Despite the short follow-up period (3 to 6 months) these results have been demonstrated in other investigations. The authors also highlighted the importance of relating therapy planning with appropriate cardiac imaging and the need to improve selection criteria. Computational techniques could be used to test improved lead placement for individual patients, or to find additional markers that increase the specificity of the dyssynchrony condition.

Wall shear stress plays an important role in aortic lumen rupture (leading to the formation of dissections). It is believed that high shear stress initiates rupture, although its contribution towards developing the disease is a contested hypothesis. [42], [43] Using computer simulations to aid the clinical decision process can provide an analysis of areas of risk due to wall shear stress (high or low). To determine the correlation of wall shear stress and lumen rupture progression, an extensive analysis of aortic flow in patients with and without lumen rupture must be carried out, however the wall shear stress analysis will undoubtedly be measured using computed simulated flow; this is a case for the adoption of computer simulations in the clinic to develop clinical knowledge.

Computer simulations used for medical analysis are still isolated from the patient, in terms of disease pathology, history and clinical decision making. The patient conditions during data acquisition are not entirely treated as an integral part of the data acquired.

For instance, clinical experience has shown that the performance of the heart is not the same when the patient is stressed. As such, it has become common practice to measure cardiac function as a function of the state of rest/exercise of the patient during the analysis. The haemodynamic conditions of the patient simply cannot be the same at rest in a horizontal pose and when under stress (walking fast-paced). An improvement to the data acquisition process could include detailed information regarding the medication intake and physical conditions of the patient and could allow better analysis of the simulation.

With the advent of new research techniques, such as “big data”, analysing a comprehensive aggregation of patient data becomes a powerful tool. For instance, to find patterns of particular diseases associated with certain professions (e.g. miners and pulmonary dysfunction) or improved screening techniques for early cancer detection. For this reason, the full patient

record, as collected in the routine clinical practice, should be treated as a valuable item in the acquired data set.

### 1.3 Contributions of this thesis to the simulation workflow

The simulation tool chain of the Grand Challenge project was a long process which included two major simulation steps (the electrophysiology and mechanics modelling). A flowchart of the tool chain is shown in Figure 10; clinical data is coloured green, processed data is coloured blue and processing tasks labelled on the arrows (which in turn represent data flow). The tool chain included extensive validation steps, utilising data from different medical tests.

The modelling workflow consumes multiple types of medical data and utilises many computational tools. The stage that this thesis is most concerned with is the generation of a mesh that accurately represents the geometry of the patient's heart. Starting from a volumetric MR image of the patient's thorax, manual and automatic tools are used to segment the myocardium. A template mesh is then fitted using a variational method. From this geometrical mesh, a mechanics and an electrophysiology mesh are generated. It's important to note that, as the geometrical mesh has multiple uses, errors at this stage may manifest for into the simulation workflow. Using invasive electrophysiological measurements, the electrophysiology of the patient is modelled using the Cardiac Arrhythmia Research Package (CARP). Using ultrasound images, ventricular volume transients are derived and used along-side the results of the electrophysiology to simulate the mechanics of the cardiac muscle, using the Continuum Mechanics, Imaging, Signal processing and System identification environment (CMISS). The results of the mechanics simulation is validated using invasive pressure catheter measurements and wall motion derived from cine MRI and tagged cine MRI.

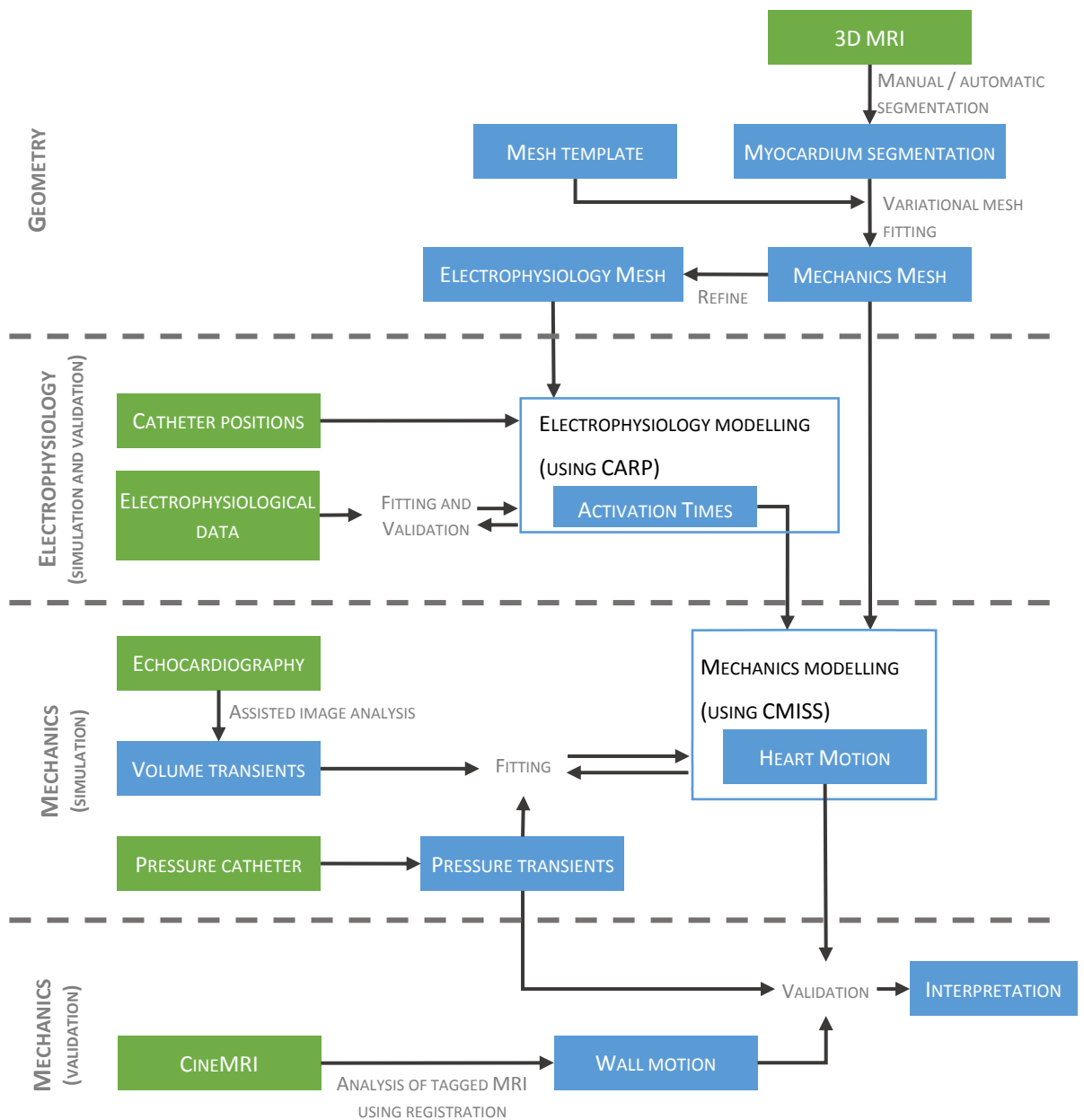


Figure 10: Schematics of the tool chain for modelling the electrophysiology (using the Cardiac Arrhythmia Research Package) and mechanics (using the Continuum Mechanics, Imaging, Signal processing and System identification environment) of the heart (adapted from [44])

### 1.3.1 Integration with Clinical Workflows

The purpose of introducing bio-modelling into the clinical decision making protocol is to make clinical assessment more efficient and accurate. In the workflow presented in Figure 10, the aim is to predict the effectiveness of CRT on prospective patients by simulating therapy effects. It makes the clinical process more efficient because careful selection of input parameters can reduce the amount of tests and modelling results have the potential to be more significant

(provide more reliable information) than other assessments. Other developments include improvement of therapy by simulating different cases before a concrete therapy plan is devised. The simulation workflow involves many inputs and processing tools, a lot of intermediate data items are produced. The workflow data needs to be managed for the users of the workflow to consume and interpret it. This includes modellers (running the tools), researchers (creating and improving the tools) and clinicians (at either end of the engineering workflow). Data needs to be validated before being consumed by a process, a task which more often than not requires human intervention. The automation of these tools relies on storage of resources in a unified and structured manner. The collection of data must integrate with clinical systems: patient database, image stores, test results, appointment system, etc.

Further requirements are needed in the clinical environment, such as a proper audit of data authoring. For the project to be successful, a clear understanding was needed of how both engineering and clinical environments should come together. Chapter 2 is a presentation of the framework that will enable translation of scientific / engineering workflows into operation in the clinical environment.

### 1.3.2 Mesh Quality

The inputs to the simulations of the electrophysiology and mechanics of the heart are produced using a mesh that is personalised to the patient's cardiac geometry. To create the geometrical mesh, a template mesh is fitted to the patient's segmented geometry using the variational mesh fitting process; see Figure 10 for an overall summary of the workflow and Chapter 6 for a more detailed discussion on the mesh personalisation method. A coarse mesh (in the order of tens of elements) is built using cubic Hermite elements and is used for the mechanics simulation. This is then refined to produce a linear tetrahedral mesh which contains of the order of 100 million elements. The cubic mesh is needed to fully capture the large, non-linear deformations of the mechanics simulation. The electrophysiology simulation involves a rapidly advancing electrical wave and thus requires small elements (edge length  $\sim 75\mu\text{m}$ ) to capture the high gradients at the location of the wave front. Both simulations require a smooth topology to avoid spurious results appearing at points of sharp discontinuity.

The simulations take of the order weeks to complete (per simulation), meaning that the time lost in ensuring a poor mesh has not corrupted the results became the major bottle neck in the simulation tool-chain. Optimising the mesh quality of the cubic Hermite mesh, and hence the stability of the simulation, became a major requirement of the project. Whilst a number of descriptors may be used to quantify mesh quality, Lamata et al. demonstrated that the stability

of the simulation is best predicted using Jacobian based descriptors, as proposed by Knupp et al. [32], [45]. The mesh fitting pipeline makes use of the Sheffield Image Registration Toolkit (SHIRT) and the obvious extension to the registration is to directly control the quality of the resulting mesh using geometrical descriptors. SHIRT works in Cartesian space however the quality metrics of unstructured meshes are better described in parametric space. The bulk of the work presented in this thesis focuses on the development and implementation of an image registration algorithm that works in parametric space, and its application to generating high quality cardiac meshes.

## 1.4 Thesis organisation

This thesis addresses the challenges outlined in section 1.3. The construction of a simulation workflow suitable for integration into the clinical environment and clinical workflow is the subject of Chapter 2.

The main contribution of this thesis, presented in Chapters 3 to 6, focuses on the development, implementation and testing of a method, based on image registration, to improve the quality of meshes derived from the images of individual patients. Chapter 6 presents the complete processing tool chain, from medical image to personalised mesh.

Chapter 7 demonstrates that the method, developed for robust mesh construction, can readily be applied to determine boundary conditions for computational fluid dynamics analysis.

Chapter 8 provides a summary of the achievements of the thesis, together with suggestions for further work.

# CHAPTER 2

## CLINICAL INFORMATICS: MEDICAL DATA AND WORKFLOWS

The focus of this chapter is on the computational aspect of medical and scientific data and workflows. An introduction to the field is presented in section 2.1. Section 2.2 will introduce some important concepts regarding the data used in clinical information systems. Section 2.3 will describe in depth provenance and section 2.4 will discuss workflows. Following the review sections, the concepts are put into context in section 2.5 with a description of the implementation of a research clinical system.

The content of sections 2.3 and 2.4, which has been adapted for inclusion in this thesis, was published in Coveney et al. [46] and reproduced by permission of Oxford University Press.

The systems described in this chapter are essential to a productive day-to-day running of the clinic, and key to the adoption to the clinical applications discussed in this thesis (chapters 6 and 7) as well as the overview described in the introduction of this thesis (chapter 1).

### 2.1 Introduction

The purpose of health care is to cure and/or prevent people (the patient) from suffering from illnesses. In order to accomplish this, practitioners must collect a massive amount of information to have a wide understanding of the different diseases that humans can suffer from (informed decision making). Our ability to collect data and develop computational tools to make use of the data is now mature enough to really challenge the common practice of health delivery.

The end point for much research in computational biomedicine is expected to be the provision of clinical tools that will eventually be fully integrated into a primary or secondary healthcare environment. Haux defined medical informatics as “the discipline concerned with the systematic processing of data, information and knowledge in medicine and health care” [47]. Haux’s suggestions of aims of health informatics include:

1. Diagnosis
2. Therapy (reducing strain on patients during medical intervention)
3. Therapy simulation
4. Early recognition and prevention
5. Compensating physical handicaps
6. Health consulting (the informed patient)

7. Health reporting
8. Health care information systems
9. Medical documentation
10. Comprehensive documentation of medical knowledge and knowledge-based decision support

Health informatics brings together expertise in the fields of computer science, medicine, engineering and theoretical and medical physics [48]; however there are arguments for considering it as a subject of its own [47], [49]–[51]. The dependency of health informatics on its parent fields is undeniable but the need for independent research and development within its own scope is a reality. Likewise, the interdependence of technological development and organisation adoption is the key for the success of the field [52].

Lærum et al. [53] performed a number of surveys on electronic health record adoption in Norwegian hospitals and reported that physicians are less satisfied of electronic data recording (opposite opinion for data retrieval), emphasising the need to work closer to the clinical environment. They also reported a lower response from physicians than medical secretaries to the survey, which could suggest less willingness to adopt new technologies.

Whilst the scope of the field is vast, the scope of this work concentrates on introducing modelling techniques into common clinical practice. Taylor et al. provide an excellent example to illustrate the value of computational techniques and demonstrate successful application of a particular solution [54]. The authors employed a segmentation tool to build a 3D model of a region of the cardiovascular system and created a tool to visualise different surgical situations; gaining understanding of the situation specific to the patient, selecting the most favourable surgical conditions for the patient and taking advantage of 3D model visualisation during surgery, proving the power of building robust modelling workflows clinical use. The application of this tool is to aid surgical planning using working patterns adopted from the engineering industry, an example of computer (simulation) aided design. Here, multiple solutions (surgical interventions) are virtual tested (flow resolved using the finite element method) to determine which one provides the best flow pattern in the patient's vessel after intervention. Taylor et al. report that four vascular surgeons were able to use the tool after 20 minutes of training but do not comment on the response of the surgeons to the tools.



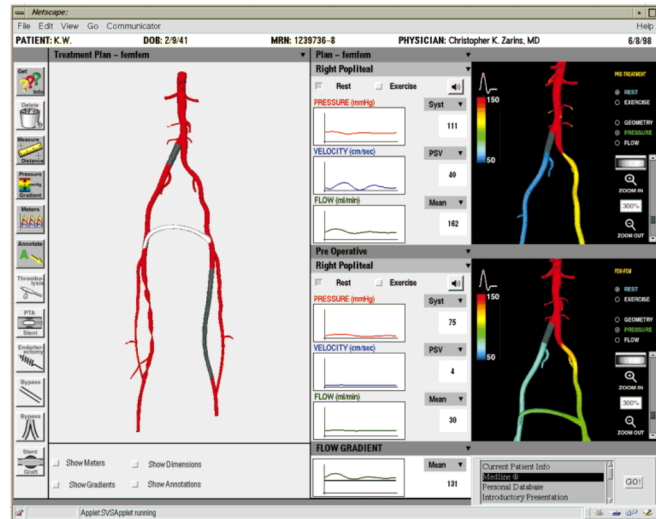


Figure 11: User interface of the ASPIRE software developed by Taylor et al. allows basic virtual surgical intervention to be performed on the vessel surface to investigate the resulting changes in pressure and flow to aid surgical planning. (Reproduced with permission from [54])

Ortega et al. tackled the issues of inter-clinician professional opinion variability and the need for user interfaces aimed at clinicians [55]. Their “Sirius” software is a web-based, plug-in orientated application designed to increase efficiency in routine clinical practice. Instead of the cardiovascular system (which is the main interest of the work presented in this thesis), “Sirius” is a tool for retinal image analysis. Ortega’s et al. work is interesting because of their vision of the extendibility of service beyond the image analysis tool, such as collaborative image-processing and analysis, and remote tool execution.

Similar to Taylor et al., a major goal in this work is to develop tools that bring engineering modelling to the clinical environment, providing clinicians with the necessary tools to make use of available information used for diagnosis and present them with extended knowledge not available otherwise. An important part of informatics is designing computational environments that provide the practical means for other specialists to implement their tools.

The main computational endpoint of the Grand Challenge project, of which this work is part of, is to integrate engineering workflows with clinical workflows to allow collaborative work and parallel development of the fields. Issues of focus to achieve these aims include:

- Automation of computational tasks in the clinic.
- Integration of computationally intensive processes in long running workflows.
- Improvement of multicentre/cross-departmental work collaborations.
- Translation of engineering and computing specialties into hospital environment.

- Integration of individual requirements, such as audit of responsibilities, ethical considerations and validation of work done.
- Fluid adoption of technology.

## 2.2 The medical record and clinical information systems

### 2.2.1 Medical record

The medical record is a piece of documentation available for every patient containing: medical history, medical care received (including test results, medication issued) and observations. Observations may be dictated in a formal or informal nature, and may contain observations that are not directly relevant or routinely noted within the practitioner's field but that might be particularly relevant. For instance, recommendations made to the patient that are not necessarily aligned with the reason for visiting the practitioner. The medical record will be updated through the different stages of hospitalisation, visit or operative treatment: admission, on-service, progress and discharge. The patient history includes medical, surgical, medication, allergies and social details of the patient and close family. The usual contents of a Subjective, Objective, Assessment and Plan (SOAP) note include a presentation of the illness, supporting measurements, possible diagnoses and plan for treatment (this may include further tests, referrals, education, or treatments).

The medical record is central to patient treatment and essential to informed care ensuring practitioners fully understand the patient's condition and patients understand the rationale for the treatment. The medical record belongs to patients (not institutions) and has historically acted as a manner for clinical knowledge to be developed. In practice, medical records are kept in paper-based notes, rarely released by the owning institutions and contain "noisy data". This means patients may have multiple medical records, with repetitive or conflicting information. Patient records contain sensitive personal information so security considerations have to be made for accessing information, the physical storage of data, and proper disposal.

### 2.2.2 Electronic patient record

The problem with paper-based records, by far the most common management system used in clinics, is that they reside in isolation (in departments or wards), which makes it hard to find and share information, and reduces the long-term value of data. The digital representation of the medical record is commonly referred to as the Electronic Health Record (EHR) or Electronic

Patient Record (EPR). The information contained in the record is part of the following categories: visit, event and measurement.

EPRs have emerged as a solution to handling of the increasing amount of patient information being recorded, with the added benefit of improved sharing and integration of information with researchers and non-clinical stakeholders. EPRs have important benefits - such as cheaper and more durable storage, better quality data and clearer to use and being more portable. The clinical environment requires the ability to quickly search and retrieve information, meaning the quality of query results should be a prime consideration. Also, EPRs should be flexible, allowing for multidisciplinary integration and remain unbounded to specific computational formats [52].

Where data is shared amongst practitioners, the meaning of data must be un-ambiguous. Data stored in computer systems is usually stored in structured form; in the clinical scenario this is beneficial due to the many ways data may be expressed. This may limit, however, the physician's ability to fully express patient symptoms or diagnosis. For this reason unified models have been developed to address the issue of sharing information between different specialities (e.g. HL7 [56, p. 7], DICOM). Unified clinical definition models such as SNOMED CT attempt to reference terminology and clinical terms in a standardised structured manner. [57]

The data audit and provenance requirements means that whenever data is stored or modified there should be a log of these changes, whether this is addition or removal of data. These concepts become increasingly complex with the growth of multi-disciplinary situations. For instance, if computer models are used for patient diagnosis one must know what information was used to create the model, who created the model, what algorithms were used and who had access to the files or made any alterations.

### 2.2.3 Information systems

The umbrella of clinical information technologies is often referred to as the Hospital Information System (HIS). These are systems involved in the process of data collection and presentation for the purpose of diagnosis, management, education, service improvement, etc., that need to interact with each other (e.g. prescription ordering) and with humans (e.g. decision support). There are a significant number of stakeholders involved: medical, admin (e.g. personnel, stock), financial, legal, scientists and patients.

For instance, radiology is not just concerned with image acquisition, it also encompasses image interpretation, suggestive diagnosis, developing new techniques (e.g. measurements, quantification) and treatment (e.g. nuclear medicine). As such, a typical Radiology Information System (RIS) will manage the visit timetable (including daily patient listing) reports (ranging from

structured databases to scanned hand-written reports and dictations) on top of being an image store. Other systems used include Laboratory Information Management Systems (LMS).

Central to the electronic revolution in the clinic is the Picture Archiving and Communication System (PACS) in charge of storing and transmission of imaging data. It offers big institutions huge advantages over film in terms of finance (cost to set-up and stock disposables), storing (space, ease of access, safety), quality of information (viewing, image degradation) and management. For small centres, the costs of the infrastructure doesn't always scale down in a favourable manner. Electronic imaging is key for many modern medical diagnostic and research techniques. PACS use the Dicom standard to communicate with modalities and workstations.

Figure 12 describes the general infrastructure of a PACS system. Modality refers to the different acquisition devices that may add images to the system. A workstation is a computer used for viewing, measuring or processing the images. Typically, there is a workstation in the vicinity of the acquisition device that allows radiographers to repeat the scan if the acquired image does not allow diagnosis. Off-site backup protects patient data in case of a major catastrophe, such as a fire or a natural disaster.

#### 2.2.4 Dicom standard

Digital Imaging and Communications in Medicine (DICOM) is a standard, developed by the National Electrical Manufacturers Association (NEMA, US), used to describe electronic file storage, transfer and interaction between compliant devices within the clinical environment. The standard defines a medical imaging file format and a communication protocol. The Dicom standard is widely adopted worldwide and is valid for all imaging modalities across medical disciplines. It describes the relationship between a patient and the file in a hierarchical fashion: from patient to study to series to instance (the file). [58]

A DICOM file contains more information other than just the image; different content is described using DICOM tags. The tags are a standard list of hexadecimal numbers with precise definitions defined by the standard, making the DICOM format more descriptive than a generic container. The file format is valid for different imaging modalities, scanners, processing software, etc. and defines the appropriate technical and patient information stored within the file. This helps to provide a full understanding of a file's provenance and contents from outside the acquisition scope. On the other hand, vendors are free to utilise private tags (with undefined, often unpublished meaning) and the abuse of these reduces the interoperability of the standard [58]–[60].

One of the tags in the standard defines the "pixel data" of the image, and the combination of this with the description of the image in other tags allows reconstruction of the image itself. This way, the standard is not confined to 2D images. Besides image data and technical information about the acquisition, DICOM files allow storage of data describing the patient, ranging from patient identifiers to measurement results. An important aspect to note is the high amount of data redundancy which in turn allows full categorisation of the file in an up-down (patient to image) direction from the information contained in the image alone. A somewhat old fashioned comparison is, being in possession of an unidentified photograph, the ability to fully describe the owner and the information described in said photograph. In practice, the image is a measurement made on a patient and is used for diagnostics purposes; the redundancy of information allows unmistakable positioning of the image in the timeline of a patient's medical history.

On top of that, the standard also specifies a number of services for file transferring (query, retrieve), file storage, and modality listing; adherence to this standard allows modalities and workstations from different vendors to interact. An extension to the standard called Web Access to DICOM Objects (WADO) allows retrieving snapshots of DICOM contents over web transfer. [61] The devices shown in Figure 12 communicate using the DICOM standard.

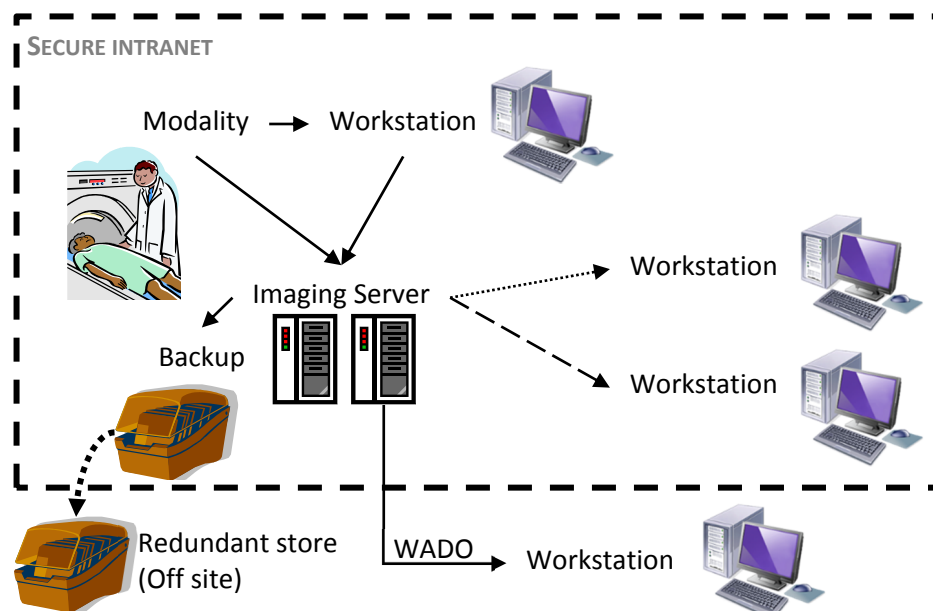


Figure 12: General networking setup to manage image data within the hospital.

### 2.2.5 Databases

The issues of storage include space, location and access permissions. Patient records might contain over hundreds of images making the space requirements of health-centres a considerable challenge. The location of data servers also poses issues as they require a dedicated technical team for IT support, and special facilities for server storage. Companies offering remote data storage are growing in number and offer specialist management of storage servers; however these pose ethical issues (effective ownership of information) and might pose technical issues in the case of loss of communication routes (e.g. data connection being down). The manner data is stored is also an issue and depends on the size and type of data. For text-based information, relational databases allow efficient mechanisms of data storage and retrieval (querying); different solutions exist, each of which adapt to different database sizes. A comparison of different relational database management systems can be found in Table 2.

Clinical applications range in size, one would expect to find many database systems being used in a hospital. For instance, a small research project might use an Access database. This is appropriate because the number of users will be very low, perhaps a single research nurse or junior medic, who on top of data collection, must design and maintain the database and perform the analysis on data. The more sophisticated database systems are considerably more expensive and have more complex and advanced tools for managing databases. On the other side of the spectrum of database size one might find the imaging system, which may use a commercial system from Table 2, or implement its own database system.

DATABASE	MAX DB SIZE	MAX TABLE SIZE	MAX COLUMNS	MAX NUMBER SIZE	MAX COLUMN NAME SIZE
Access	2 GB	2 GB	255	32 bits	64
SQL Server	524258 TB	524258 TB	30000	126 bits (10 <sup>38</sup> )	128
MySQL	Unlimited	256 TB	4096	64 bits	64
Oracle	Unlimited	4 GB	1000	126 bits (10 <sup>38</sup> )	30

Table 2: Comparison of different commercial database systems [30]

### 2.2.6 Clinical ethics

The identity of patients and all their medical data is confidential; this is a right protected by national regulation [62]. No information that would allow association between a patient and their clinical data should be disclosed, or be protected negligently [63]. The right of clinical workers to access patient information is restricted to only the relevant practitioners and members of staff. When information is not used for medical care – such as research or audit -

patients have the right to choose whether to share their information. This is done by giving informed written consent - meaning the patient is aware of what use is going to be made of the data and the possible outcomes of the research project. Patient data cannot be collected into a research database prior to having granted informed written consent. In addition, data must be anonymised appropriately prior to being used, allowing only practitioners involved in direct medical care access to the full dataset.

Anonymisation or de-identification is the process of making patient identity not back-traceable. In practice, this task is complicated by the redundant nature of patient information across files/forms, existence of annotations embedded directly in images and the inconsistency between manufacturers and clinicians when completing DICOM tags. The DICOM standard requires 18 fields being removed (plus all private data fields) to ensure anonymisation; however practice has shown that it usually requires extra work to ensure true de-identification.

Other considerations must be taken. For instance, if a trial involves innovative treatment, patients must be informed that such a protocol has not been approved for standard practice. Also, patient data cannot be removed from hospital premises; this includes electronic media moved from the clinic's computer system [64].

### 2.2.7 Workflows

The workflow system works at a layer above other clinical systems; integrating systems such as the imaging servers (PACS), patient record (EPR), prescription services, and check-in kiosks. The use of workflow-driven technology brings different benefits to the stakeholders of the clinical environment. The overall benefit is a more pleasant and more competitive work environment.

To managers, it means tracking and reporting performance, identifying business weaknesses and better planning ahead. To non-clinical staff, it helps managing their tasks and balancing workload across staff members, improving communication and access to the relevant data. To clinical staff, it means better managing (and perhaps more availability) of patient data, as well as better use of patient data. To patients, it lowers the length of stay in the clinic and allows to keep them better informed of their health; which directly increases satisfaction with the service received. To bio-medical researchers, it has the potential to bring them closer to actual patient data and real clinical practice.

Section 2.4 will discuss the principles and clinical applications of workflow systems. Section 2.5.2 presents details regarding the implementation of a workflow system.

Throughout the following sections, references are made to the Windows Workflow Foundation [65] and the Taverna environment [66]. The first is a set of libraries within the .NET framework to build workflow systems, the second is a data-driven workflow system and a popular tool within the scientific bioinformatics community. A review of these tools is beyond the scope of this work however they serve as a helpful way of contextualising certain ideas.

### 2.3 Data provenance

This section is a conceptual review of data provenance, an essential idea to understand in order to build useful clinical database and workflow systems. The concept of provenance also serves as a bridging topic between these two systems. Practical implementations of data provenance may come at the database or workflow level but must describe the interface between the two systems. A basic implementation of data provenance is implicit in most database and workflow systems but the bigger discussion presented in this section enables a better understanding of the challenges of implementing software solutions for clinical systems. “Provenance is everywhere”. [67]

The term provenance is generally used to refer to the context and particularly to the history of an object. In computer science, the word can be used to refer to the cause (and ownership) of an event, process, or data item. It therefore relates to the consumption or creation of data, rather than the content of the data itself. It is essentially a historical track of how, why, and when a particular piece of data was created and who or what were involved in the process [68]. It changes its meaning subtly depending on the subject matter: the provenance of a database entry, for example, will include who inserted the data, when, and possibly by what means, whereas the provenance of a data measurement (such as the amount of a chemical) might contain information on the method and equipment used and the conditions under which the measurement was taken. It might also include the name of the technician who took the measurement and the time at which it was taken.

Provenance is a crucial part of scientific workflow design and is required to meet clinical research medical standards. The provenance of a scientific workflow can be divided into two categories, as follows.

- Process provenance refers to the execution of a single process (which might be a client process or server-client interaction) and to the logical trace of the workflow (i.e. which steps of the workflow were taken and for what reasons). This two-scale level allows the tracking



of processes within the scope of the workflow and interactions between the workflow and data and events held outside it. This is also referred to as coarse provenance.

- Data provenance is used to refer to the derivation route of a particular data product which might be processed entirely within the workflow or serve as an input to the workflow or the invocation of an external code. The depth of data provenance refers to whether this information has an immediate relationship with the data or is a recursive property. This varies from application to application and is also known as fine-grain provenance.

It has been suggested that the term 'provenance' can be over-used [69]. It is therefore useful to consider which aspects of workflow design are best suited to the provenance approach.

Workflow provenance should address the follow aspects.

- Data quality: accurate determination of the quality of data is already very important for determining the reliability of data sources and how data should be used. This is increasingly so with the current rapid increases in the amount of data (whether this is experimental results, raw data, or published data) that is available to modellers and with the integration of computer modelling with experimental science.
- Data audit trails and attribution: closer links between academic research and the business and clinical communities mean that academics increasingly need to think along the same lines as industrial scientists and clinicians in providing legally valid audit trails for their data. Scientific workflows, in particular, cannot be exempt from this requirement as they are typically extremely complex, involving large numbers of processes and operators and generating vast volumes of data. The use of provenance provides a solution that addresses this need in programming terms. Data attribution, which refers simply to giving official credit for work done, assessing individual contribution to a project, or measuring work performance, is also now hugely important in many academic and business situations.
- Reusability, reproducibility, and repeatability of measurements and data: repetition of experiments has always been part of the scientific method. If workflows can provide a means to quickly and efficiently repeat tasks (which may be trivial or complex, immediate, or long-running) this will be an exemplary use of technology in the development of the scientific process. Since provenance can be broken down to process independent blocks, it may also be used to reuse parts of experiments in different contexts and to demonstrate the reproducibility of measurements or new evidence for a hypothesis (e.g. peer review).
- Revision: if the experimental parameters used to obtain a particular result can be retrieved, this will offer steps in post-processing and reviewing each stage of an experiment or model.

Similarly, recovering the logical steps followed or processes involved in a workflow are also important during the process of optimizing procedures and improving a workflow. This may also include determining sources of error or reasons why processes fail. Furthermore, provenance can be used to compare similar workflows that produce differing or diverging data in a more meaningful way than if the only information available came from the final results of the workflow.

- Annotation: annotating data with associated metadata not only expands the meaning of the original data but may also add value to it that can be used for other purposes. Metadata may be designed either mainly for human interpretation (often as free text) or for computers (in a structured format).

The recording of provenance during a workflow operation can give rise to other secondary considerations that may still have an impact on the design of the provenance system. These include the following.

- Migration and storage: this refers to the place where the provenance information is stored, which may or may not be the same location as that of the data to which it refers. Links between the data and the description of its provenance need to remain when that data is moved or consumed. It becomes an architectural problem when data is moved across domains, shared between different platforms, or stored in different databases. Issues with specific data standards can arise if provenance information is added to the metadata of each file. Consider, for example, an imaging file in DICOM format. It is not clear where in this file type it is possible to store a formal provenance trail, and although it is possible to add such information in private fields this can lead to problems if the data is later anonymized. The process of anonymisation, which is designed to remove any information that links the data to an individual, will explicitly remove all data marked as private.
- Querying data: the most useful data is data that can be queried in a variety of ways, with the output presented either graphically or in text form. In a good provenance model, the provenance information that is attached to the data should aid the formation of complex and useful queries and improve response speed.
- Data tracking: the ability to query data (and its status) from within a workflow without having to load it into memory can improve both the speed and the overall performance of data queries. A variety of efficient workflow tracking services are available, including the standard Windows Workflow Foundation persistence and tracking services. Similar services are available in Taverna.

Figure 13 illustrates the taxonomy of provenance and highlights the issues that should be considered when selecting a provenance model for a workflow.

Just as it is hard to reach a consensus on a workflow environment that is perfect for all possible workflow designs, or even for all those that are appropriate in computational biomedicine, it is hard to reach consensus on the perfect provenance model. The choice of a provenance model depends on factors such as the computer standards to be adopted, the requirements of the amount of data and data types to be collected, and optimization. For instance, provenance models and workflow models that are strongly coupled together will use fewer computing overheads, be more automated, and may make better use of the available data. However, these models may have less flexibility, scalability, or variability than a more loosely coupled approach.

In the most general terms, an abstract provenance system is expected to [70, p. 2]:

- provide an open and interoperable interface to collect provenance information;
- track the workflow and data in virtual organizations that are independent of workflow model and data format;
- minimize the performance overhead and necessary modifications to the workflow components;
- provide an open and interoperable interface to query provenance information.

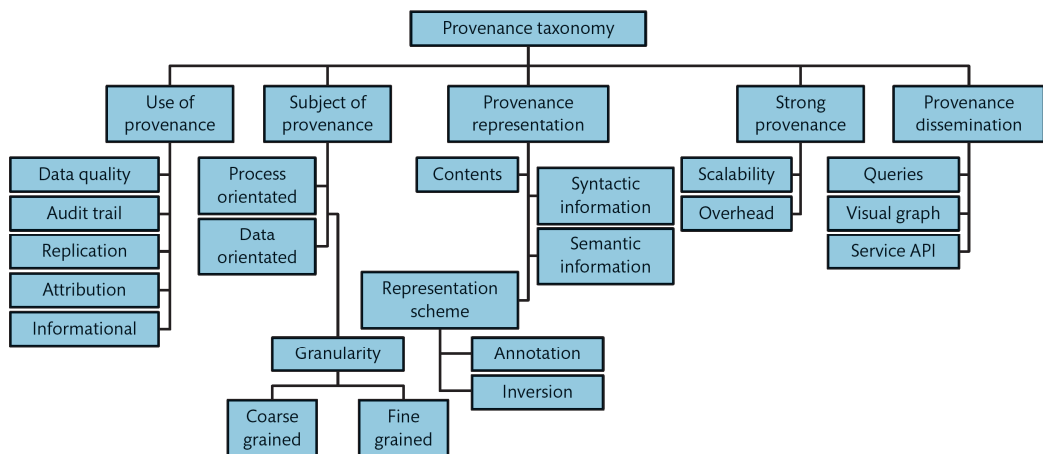


Figure 13: The taxonomy of provenance describes points to consider when defining a provenance model [71]

Despite attempts to work towards a common workflow provenance model that can be applied within all disciplines, most projects still implement their own provenance model. This might well be due to the conundrum of which comes first, workflow design or provenance model? It should

be possible to develop a standard common provenance model that is independent of any specific workflow, but this will require a much greater consensus for any such standard to be developed. Currently, many provenance models are considered to be incomplete, unreliable, and non-portable [67].

The Open Provenance Model (OPM) has been developed as a standard interoperable model that can understand, collect, and represent provenance. This defines the concepts of agents (actors), processes (actions), artefacts (data), and p-assertions (provenance-related connections between these three). The connections between all these are described in OPM graphs, which can be considered as similar to database table schemas. In addition, the OPM specifies a standard XML-based schema for representing provenance and Java-based libraries to convert XML documents to and from OPM graphs, Figure 14 shows the concepts used in this schema [68], [72]. An important issue in this model is the lack of control of the provenance information concerning data objects before they enter the workflow domain. A similar issue arises with the versioning of external web services. The system, in fact, relies on good coding practice, and it can be argued that if this practice were always followed there would be less need for provenance information. It will always be easiest if all data and processes can be managed within the hosting environment. Taverna has recently been extended to support the OPM, but it has not yet been fully integrated into the workflow system.

An excellent example of an implementation of the OPM is given by the Karma project, which was developed at Indiana University.<sup>1</sup> This takes the form of a stand-alone Java-based service to decouple workflow execution and tracking. Karma captures streaming information and defines two abstract layers, termed the execution layer and the registry layer. Streams are then further abstracted into data products, workflow types, services, and methods, and these can be linked to individual workflow instances or runs. Karma provenance data are stored using MySQL with an extension for extracting specific workflow provenance in XML format.

---

<sup>1</sup> The project website can be found here: [http://d2i.indiana.edu/provenance\\_karma](http://d2i.indiana.edu/provenance_karma).

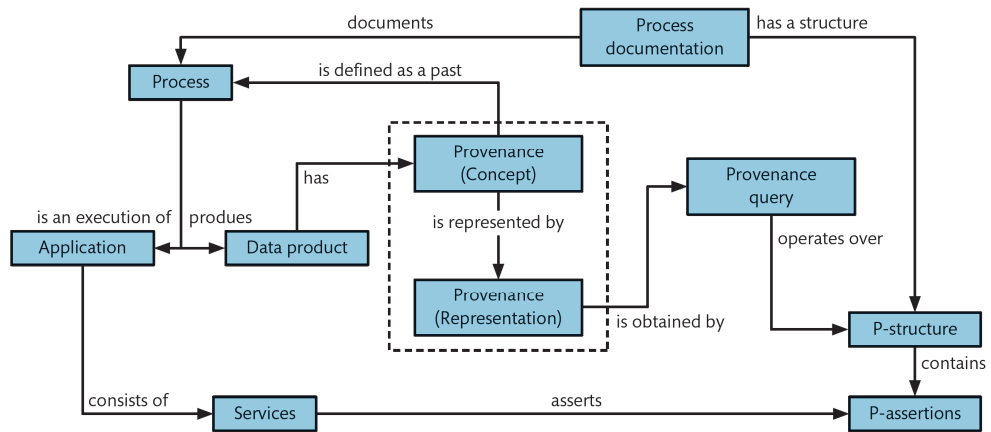


Figure 14: The provenance concept map shows the different ways provenance is used and how it links together different data and process management concepts [72]

The subject of workflow provenance is an extensive one, and it has only been possible to touch briefly on it here. Further open-source alternatives to the OPM, including operating-system-based provenance tracking systems that track program execution at a low level. Other issues that are worth considering, although rather beyond the scope of this chapter, are the efficiency of provenance querying, further privacy and security issues associated with provenance access, and the use of web semantics for provenance annotation [73], [74].

It is also worth noting that for an effective provenance model all processes, and commonly also all executables, must be unambiguously identifiable; this includes the versions of each code that are to be executed. A detailed discussion of this is beyond the scope of this thesis; many tools including git and subversion are available to assist with this management process.

## 2.4 Workflows: principles and clinical applications

The dictionary definition of the term *workflow* is quite simple. It is:

***“the sequence of industrial, administrative, or other processes through which a piece of work passes from initiation to completion”<sup>2</sup>***

This rather vague statement illustrates why the term ends up meaning different things to different people. It can clearly be applied in many different contexts, some of which have nothing at all to do with scientific computing.

<sup>2</sup> This definition can be found here: <http://oxforddictionaries.com/definition/english/workflow>.

### 2.4.1 Computational Workflows

The basic definition of a workflow, is a set of events or tasks that are arranged to be executed in a particular order. Anyone who has studied logic or computer science, even at the most basic level, will be aware that there is a similarity between this and tool-chains and flowcharts. However, the concept of a computational workflow differs subtly from the other two concepts. If a flowchart is a way of representing a particular procedure—perhaps logical thinking, process control, or data manipulation— the computational workflows that are the subject of this chapter are simply implementations of these flowcharts in a programming language. Workflows consist of a set of states (which encapsulate sets of actions), state transactions (which define transitions between states), and rules (which control actions or transaction logic based on process data).

There are two main types of workflow implementation, and these have been termed sequential and state machine. Most of the commonly used scientific workflows are initially defined as sequential but it will be seen later how this often changes as the workflow gets closer to a formal implementation.

Workflow-oriented programming can often be more efficient as it enables users to manage several jobs simultaneously, separate logic implementation from code writing, and rerun common procedures automatically.

Workflows, therefore, are not intended to run as stand-alone programs [whether console- or graphic-user-interface (GUI)-based] but as a set of ‘standard’ instructions that can be run constantly or periodically or repeated as required. This introduces the concepts of the workflow type and workflow instance. The former term is given to the set of flow rules that are defined at workflow level, and latter to a particular instance or ‘run’ of these rules.

A workflow engine controls the running of individual workflow instances; this is analogous to processor multithreading although it is implemented at a much higher level. Individual instances of a workflow are therefore accessed through this main engine, which is free to queue calls to instances; this is analogous to a switch or router. Higher resource efficiency can be achieved if individual workflow instances are able to persist in a database when they are idle without using the computer’s working memory. This is analogous to the well-known procedure of suspending or hibernating an idle job in an operating system such as Unix.

The concept of abstraction, as applied to workflows, refers to the need for end users to be able to manipulate the behaviour of the workflow even if they have no knowledge of programming languages (or even just of the language(s) that the workflow is written in). This is built into the workflow by providing a user interface that allows the modification of workflow logic.

Abstracting the workflow logic and processes from flow control (think of a simple flowchart) has the added benefit that workflow definitions will not need to be recompiled when particular activities or services are changed, or when changes are made to the logic or rules.

It is very likely that workflow design is divided into a number of separate specialist areas when working on a big interdisciplinary modelling project that involves a complex workflow. These might involve, for example, programming the workflow environment, coding specialist activities, the design of subsets of the workflow, and managing its day-to-day running. Workflows may be designed so that specific users or types of user are given or denied access to particular workflow tasks or routines.

### 2.4.2 Petri Nets

The mathematical concept of a Petri net provides a formal definition and a graphical representation for the execution of distributed processes. It can, therefore, be a useful way of representing the formation of a workflow. Petri nets consist of places, transition nodes, directed arcs, and tokens. In this formulation, places represent systems, transition nodes represent rules for changes in place to happen, directed arcs represent actions by connecting places with transitions, and each token represents a current location or active process. Progress through the process (which might, but need not, be a workflow) can therefore be represented by the consumption of tokens.

Figure 15 (a) is an example of a Petri net that shows the consumption (red arrows represent time) of tokens by an event firing (green). There are two possible ways arcs could branch, at places or transitions. Figure 15 (b) shows branching at transitions. Since there is no conflict, concurrency is possible. However, the specific state of the system is ambiguous. This representation in which the pipeline is pre-set and depends on the release of tokens by the places may be used to describe a data-driven system. Figure 15 (c) shows branching at places. There is a clear situation of conflict, so the ability for concurrency must be suppressed. The plus side of this is having the system represented at a specific state at any point in time. This representation, in which the pipeline is not known until completely executed and in which it depends on the particular events that are triggered represents an event-driven system. The certainty of the state of a system is lost with concurrency.

Conflict is the situation in which token transitions are not certain. If concurrency and conflict occur at the same time it is called confusion. Figure 15 (d) shows a state of confusion in which both transitions are ready to fire but the actions of one cancel the other.

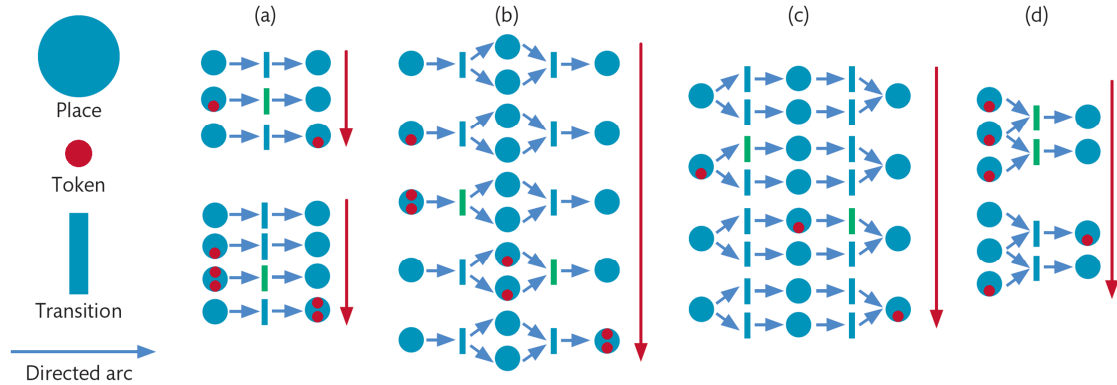


Figure 15: Basic illustration of a Petri net. (a) Shows basic token consumption after the firing of an event. (b) Shows branching at transitions and is an example of concurrency. (c) Shows branching at places and is an example of an unambiguous system. (d) Shows a situation of confusion.

We may therefore use Petri net notation to represent a workflow graphically by mapping each Petri net concept onto the components of the workflow as follows:

- places represent states or activities (circles);
- transition nodes are state change rules (bars);
- directed arcs are processes (arrows);
- tokens mark the current state of the system (dots).

If a workflow is to be valid, the presence of a token must allow for deterministic execution; this means that the execution of that task is fully described by the workflow rules, the initial conditions and formal decision points within the workflow logic. If the situation is ambiguous—that is, if a transition does not define a single, unique, and available place—then that transition is said to be in conflict. This might occur when a place is connected with more than one transition. If there are tokens in two different places at the same time, the workflow is said to be concurrent, with two processes being executed at the same time.

During the execution of a workflow one may need to know the state of the workflow at any given point in time. This means knowing where tokens will be allowed to transition next. If this is not clear, the workflow cannot execute and it can be said that the workflow is in a state of confusion. This happens when there is both concurrency and conflict, creating ambiguity during interpretation and the implementation of logic. It is an unacceptable situation for a computational implementation.

To avoid this state, the Petri net that defines a workflow may be constrained to allow only one phenomenon. How this is achieved depends on the type of workflow (see Figure 16). In a sequential workflow, each place is limited to one input and one output which ensures that there



can be no conflict in logic flow, but concurrency is allowed. In contrast, in a state machine each transition is limited to one input and one output, which ensures that there is no controversy.

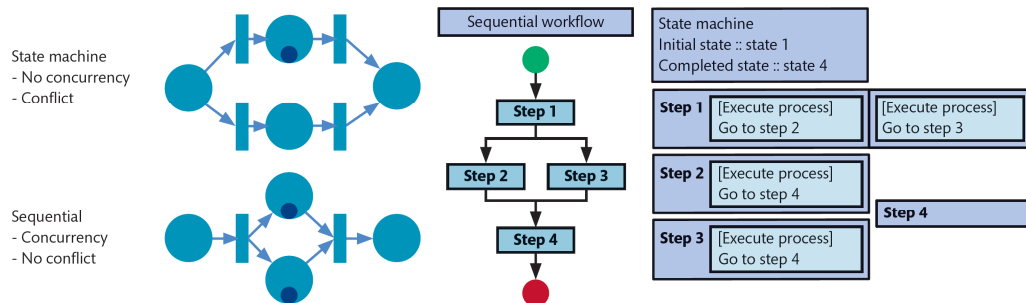


Figure 16: Purpose, automation, control, and design of workflows shown using Petri net notation. The left-hand panels illustrate the difference between a state machine and a sequential workflow.

The use of Petri net notation allows us to understand the limitations of workflow engines. However, while the execution order of a flowchart might be obvious to a human, the workflow itself will be interpreted by a machine. A flowchart must therefore be translated into a discrete set of rules that a machine can understand. This provides a framework behind the design of workflows. Also, it is often required that the workflow engine does many different tasks, including checking data types, validating the workflow itself, and executing complex sub-workflows. One of the most basic points to consider when implementing a workflow system is the selection of the most appropriate model for the workflow: the major paradigms being sequential and state machine-driven.

Many workflows are implemented using distributed computer systems, in which more than one action—often many actions—can be automatically performed at the same time. This means that a flowchart procedure that needs to be heavily automated should be implemented as a sequential workflow. In contrast, a state machine is designed to exist in a state where it waits for an event to be triggered by another part of the workflow or by human intervention before a transition can take place [75], [76]. Translating Petri net concepts into workflow terminology, a place refers to an activity (which could be a task and/or state the workflow is in) and flowchart-style arrows replace transitions and directed arcs. The interpretation of this control logic is dependent on the workflow engine.

A sequential workflow may be easier to conceptualize in that it represents a sequence of actions or activities that are dependent on each other and that must be executed in a particular order. This assured execution order gives confidence in data linkage and the implementation of logic.

However, it is important to work out the sequential order of all possible actions at the beginning of the process when designing a sequential workflow. This means that it is difficult to use this type of workflow to model behaviour that is highly conditional.

State machine workflows are those in which the workflow waits for events to take place in order to trigger workflow activity. Even though events may commonly occur in a given order, this cannot be relied on. The workflow logic must ensure that any interdependencies between workflow activities are met under all possible combinations of external events. This flexibility makes checking errors in logic and data in state machines a harder task.

In practical state machine implementations, activity interdependency may exist due to shared data; quite commonly an activity will work on a file that was created by another activity. Another common type of activity interdependency arises when a task depends on multiple decisions being made beforehand. The user must ensure that the logic is able to handle all combinations of external events.

It is possible to translate any formal Petri net graph into a general workflow design. A formal representation like this is generic and may seem a little abstract, but it is important to realise that these are the logical foundation on which all workflow engines are constructed. Their understanding is critical to being able to evaluate critically the limitations of different workflow implementations.

It is also possible to define a given scientific workflow using several different paradigms. One common task for a workflow involves data processing and mining. This clearly shows the importance of checking data types, the interdependence of events during the workflow (as each output serves as the input for the next process), and the importance of defining the sequence of processes. A given situation might require the execution of several tasks in parallel, or it might involve the execution of several extremely computationally intense programs or sub-workflows. A different situation arises when the motive of the workflow is a time-dependent simulation (an in-silico experiment). This type of workflow is commonly used in chemical processing; one example of a scientific workflow of this type involving modelling at the molecular level is a molecular dynamics simulation (or a sequence of such simulations). These examples suggest that the workflow solution to a specific problem will need to be individually tailored to the specific situation.

### 2.4.3 Clinical Workflows

There is a clear distinction between a workflow that is designed to implement a series of tasks in scientific computing and one that is designed to be used in the clinic. A clinical workflow is likely to be closer to a business workflow than to a scientific one: essentially, it can be thought of as a customized implementation of a business workflow. This type of workflow will be driven by fairly static business logic and may fragment into many sub-workflows and processes. It is likely to be written in a language that has been customized for business use. The Business Process Execution Language (BPEL) is probably the most commonly used example of a language that is used to represent sequential workflows in a business environment. In addition, business workflows often require human interaction, and this can be difficult to manage in this system. The introduction of an extension to BPEL, called BPEL4People, demonstrates the specific difficulties that this type of workflow originally had when dealing with humans instead of machines. [77]

WORKFLOW CONTEXT	CLINICAL WORKFLOWS	SCIENTIFIC WORKFLOWS
Purpose	Long-running patient monitoring	Computational intensive data manipulation
Automation	Human interaction	Highly automated
Control	Flow control	Data flow
Design	Stable design	Evolving design

Table 3: Differences between clinical and scientific workflow paradigms

The differences between clinical and scientific workflow designs (Table 3) are an essential idea to understand given the differences between the tasks that they are used to automate. However, these differences make the integration of scientific and clinical workflows within the same framework particularly difficult. Scientific workflows that are intended for clinical use are data-flow-oriented, and are created to automate and expedite the manipulation or analysis of data, particularly where this involves tedious repetitive processes. Individual functions are arranged into a pipeline of tools. Each of these tools will manipulate a data object by adding or extracting information and then present it to the user (or, of course, to another process). In effect, the tools can be thought of as a single process with one input and one output.

Most clinical workflows in current use are designed mainly to automate important non-scientific procedures, such as patient tracking and health centre management. No workflows have yet been produced that can effectively carry out diagnosis and clinical decision making, particularly

as the condition of patients entering a hospital, for instance, cannot be assumed. The most pressing need is to integrate workflows into larger and more open organizations (such as a group of hospitals). Data will need to be shared between these organizations, although the tools used to analyse it may differ; in addition, data formats will differ depending on whether the data is meant to be analysed by humans or machines, and there might still be good reasons why the access to some data will need to be strictly controlled.

Clinical workflows are generally more static than scientific workflows, which will often need to change as the scientists involved in the modelling explore new and better methods. Both types of workflow will need to incorporate many different levels of user; in the scientific environment, for example principal researchers, computer technicians, postdoctoral researchers, and students, are bound to have different access needs. In a health centre, the users of a workflow are likely to include clinicians, IT administrators, and software engineers.

Scientific workflows need to be flexible and might change in order to explore new and better methods. In contrast, clinical protocols are often long established making the workflow design static. Added considerations include the workflow design approval process; this indeed reflects the difference between the different user levels. In a scientific environment, users are more likely to be involved or require access to different levels of the environment than in a health centre, where users range from clinicians to IT administrators to software engineers.

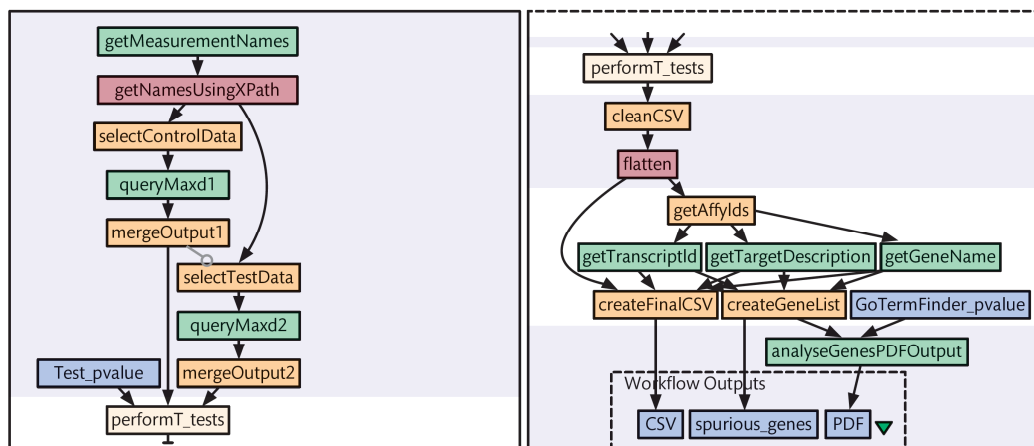


Figure 17: A sample workflow used in a scientific laboratory. It performs some data manipulation, executes some tests, and queries a database to analyse results (adapted with permission under Creative Commons BY-SA 3.0 from [78], split into two panels)

Figure 17, Figure 18 and Figure 19 illustrate typical scientific and clinical workflow scenarios. The illustrations of clinical workflows (Figure 18 and Figure 19) show that it is not always easier to

represent all processes as sequential and that state machines provide a much simpler representation for some types of process.

Figure 19 illustrates the possible pathways a patient may get treated in the STH cardiology clinic, from admission to dispatch. The first row shows the different sources patients may arrive from. From the perspective of the cardiology clinic, patient treatment falls under the responsibility of a cardiologist or a general practitioner (second row). As a result of a clinical appointment, a number of medical examinations can be ordered, the patient might be dispatched, selected for therapy or surgery, or referred to an in-house or external clinic.

The significance of the different entry points in the flowchart is that the clinic has different information regarding the patient and that their possible illnesses are of different urgency. Different events in the pathway last different lengths of time and involve different clinical staff. In most cases, decisions are made by the doctors, therefore events such as tests and surgery follow-up lead back to them. Lastly is the responsibility for treating the patient may change throughout the flowchart (i.e. a blood test might suggest treatment of an illness outside the cardiology unit should be prioritised, before the patient returns for treatment).

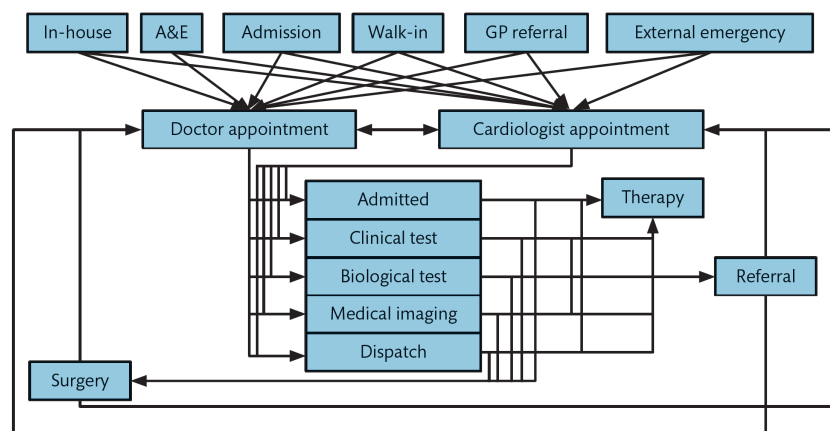


Figure 18: Due to the multiple branching points and points of re-entry into the system, workflows for a single clinical department can be troublesome to model sequentially.

Scientific and clinical workflows also differ in their scalability. While hospitals need to keep audits for long periods and thus host workflows that (at least ideally) will run for a very long time without stopping, a typical scientific workflow will have a much shorter timescale. A clinic or hospital will typically run a large number of workflows, each with a relatively low demand for computational resources, and these will need to keep track of a large quantity of patient data as they accumulate over time.

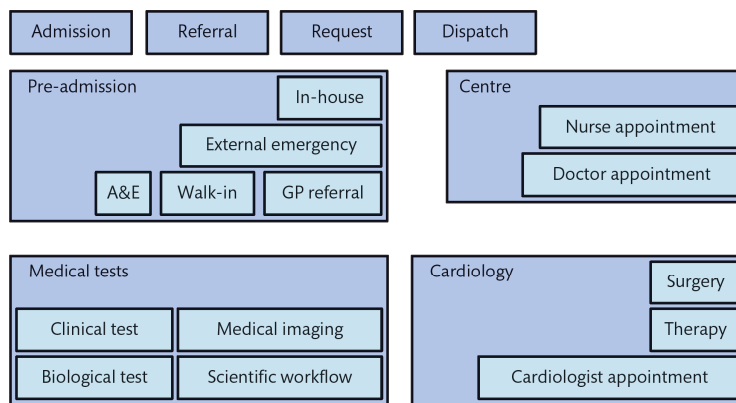


Figure 19: When the number of permutations that may happen is high, as in a hospital, the state machine paradigm makes more sense.

Scientific workflows tend to be involved with the manipulation of data rather than with making judgements, which is important in clinical workflows. These are conceptually similar, but if decision making is to be involved the workflow implementation will need to be modified to take the interaction with humans into account. Scientists will tend to use workflows to reproduce or model processes while clinicians and managers of hospitals will have a stronger requirement to audit them. However, this is not necessarily a clear-cut definition, and the most powerful workflow software will of course provide the functionality for both.

In recent years, the research environment—particularly the applied research environment—has become more pressurised, with industry always pressing to shorten the time between a discovery and its exploitation. Medical researchers suffer from a lack of real clinical data that is both available and appropriate for their research. Clinicians, even research clinicians, quite reasonably expect higher levels of data confidentiality than scientists, as they will often be working with data from individual patients. It is becoming increasingly obvious that researchers and clinicians will need to work even more closely together—and, therefore, to understand each other’s mind-sets—if the results of that research are to be applied successfully in a real clinical environment.

There are two clear advantages to adopting a workflow system in the clinic: patient monitoring and the integration of clinical systems. A typical clinic will be full of specialist ad hoc solutions for obtaining, storing, and manipulating clinical data; a comprehensive clinical workflow system will need to be able to access all these. The key benefit of bringing all these solutions together under one common framework is that while the independent systems might have had minimal

ability to communicate between themselves, the workflow system will take centralized control of the whole system.

If a system for monitoring patients is incorporated into a workflow, this will produce data and information that will be extremely valuable for future analysis. It will produce, for example, a labelled timeline of each patient's condition and how he or she has been treated: an idealized, computer-based form of a patient's medical record. The staff responsible for the day-to-day running of a clinic will be more able to plan and balance their often hectic workloads, and it will be easier to monitor and control the capacity of the clinic. The resulting data will be valuable for treatment planning, optimizing working protocols and improving the pathways through which patients with a particular condition will be treated. There are many more possible uses for the data that can be obtained and stored by clinical workflows, and these may be financial—such as auditing—or involve the communication of data and protocols outside the hospital, in, for example, research papers or publications aimed at patients or the public.

Ideally, therefore, the outcome of incorporating workflows into a clinical environment will have direct benefits for that institution's patients. These are not all clinical: they may include, for example, reductions in the time spent on treatment as bureaucracy is reduced and the whole environment becomes more efficient.

The benefits of introducing workflows into the clinic may be extended further if research-based or engineering workflows (including workflows set up for modelling projects along the lines of those discussed in Chapter 7 of this thesis) are fully integrated with those clinical workflows. The most immediately obvious of these may be further opportunities for collaboration between basic biomedical and clinical researchers, and the parallel development of these fields. There are, however, many additional benefits, and these may include:

- Automation of computational tasks in the clinic;
- Integration of computationally intensive processes into long-running workflows;
- Improving the prospects for collaboration between clinical centres and departments;
- Translating engineering and computing specialties into the hospital environment;
- Improving the speed and ease of the adoption of new technology in the clinic.

When selecting a workflow paradigm to solve a particular problem, whether in basic biomedical or clinical research, it is important to take the human element into account. The following statement may be a useful one to note in this context:

### **Humans can just be considered as a special type of long-running and unreliable task**

One value of this statement is that it helps in reminding us to assess processes involving humans (rather than machines) alongside the other computational tasks in the workflow, instead of treating them separately.

However, the real power of this approach lies in the realization that complex computational tasks share some properties with humans. This might sound odd, but it can be explained using an example. Computational fluid dynamics (or CFD) simulations can be unreliable in much the same way that humans can. They take weeks or even months to run, and their output can turn out to be nonsense, or the code can simply stop executing at some point if the problem becomes unstable. These are not uncommon attributes for complex scientific codes, but can lead to problems if the workflow in which they are embedded is defined in a sequential way. In many cases this problem is addressed by setting one or more points in the workflow at which the setup is validated by a human operator. This step, whilst it might even be thought of as obvious, builds a number of human interaction tasks into the majority of workflows. While there are often ways to 'coerce' sequential workflow engines to deal with breaks in the process at points when human interaction is needed, these do tend to limit the flexibility of the computational set-up. Dealing with this type of problem is even more complex in a state machine-type workflow, so there seems to be no general solution that will be optimal for all workflows likely to be encountered in biomedical and clinical research.

It is also important to realize that when using one of the well-established methods for defining a workflow process, such as the BPEL, the output will not be tied explicitly to a single workflow application. It will, therefore, be possible to change approaches even during the course of a project. It often makes sense to use a highly flexible system, such as Taverna, during the initial development phases of the process. However, once testing has been completed and the workflow can move into a production-type environment, the definition of the process can be imported into another more suitable framework. This strategy can work very well, but only if it is well prepared and the researchers have ensured that the short-term development environment is compatible with the long-term production system. The main constraint is that there should be semantic compatibility between the two systems being used; that is, that they should share a common representation of control flow and logic. If this is true then often an automated transformation process can be applied to achieve syntactic compatibility reducing the amount of work required and the errors that often accompany such complex manual transcriptions.



#### 2.4.4 Implementing Workflows

The procedures involved with setting up and running a workflow can be divided into three main stages. These are roughly sequential, and represent different levels of abstraction between the user and the components or tools involved. They are [79]:

1. The design or composition of the basic shape and logic of the workflow: ideally, this is the only level at which a scientific researcher is involved. Workflow design differs from general computer programming in that the whole purpose is to assemble tools that solve specific tasks together with rules to determine their execution. A component of a workflow may be another workflow. Most workflow systems include a graphical interface for viewing workflows, typically using the directed graph metaphor. Systems developed for scientific workflows focus on the flow of data across workflow components; the workflow shown in Figure 17 was composed in Taverna. Whilst some systems may allow graphical editing of workflows, it is important to note that the graphical tools produce workflow definition files (for consumption by the execution system), meaning that if the format of the definition is known they can be generated using other technologies.
2. The lower-level planning of the workflow execution: this stage defines, for example, where— that is, on which hardware— each component of the workflow will run. There might be cost implications in this: some commercial systems will charge for any use, and different levels and priorities of use might attract different costs in some models. Workflow mapping refers to the process of generating an executable workflow from a workflow definition that is abstract and resource-independent. The stages between the design and execution of a workflow include validating the workflow, parameter binding, optimization, and allocating resources and scheduling. However, not all details of the execution environment are known at the design stage. Different levels of automation are available in different workflow systems, according to the intended use of the system.
3. The final stage simply involves executing each stage of the code on its specified resource, checking its progress, and starting the next stage of the workflow when it is complete. Workflow systems usually monitor workflow execution by capturing provenance information about resolution of execution rules, task execution statistics, data consumption and system variables.

## 2.5 Implementation of clinical informatics to support the simulation workflow

This section refers to the specific implementation of a database and workflow system within the context of this project, to integrate the simulation workflow into the clinic.

### 2.5.1 The clinical database

ArQ is an application developed by the Scientific Computing and Informatics group at the Royal Hallamshire Hospital aimed to provide a flexible yet robust electronic health record. The software allows different investigators in a project to share a common patient record yet it is designed to be used as a clinical system; it provides access to demographic information (anonymised in non-clinician view), PACS data, patient record, and appointment information and assessment results, all available through the same software. It presents personalised databases through a rich graphical interface.

ArQ is a database front-end and allows connection to a number of database back-ends, such as Access, Microsoft SQL Server and MySQL - it does not replace existing database systems, instead it provides clean access. This means that databases can be adapted to user needs and available resources. Also, it includes the ability to access multiple databases, meaning that all patient data does not necessarily need to be stored in the same server or even domain. Security credentials for accessing the ArQ front end can be both a corporative token or individual access can be granted in a traditional username-password manner.

Designed with a plug-in based approach, the ArQ designer allows administrators to personalise the look and feel of the interface whilst developers are able to add their own functionality and access the database through a generic interface provided for such purpose. Such functionalities would expand the available components for data input (textboxes, drop-down menus, checkboxes, time, etc.), design (logos, labels, group boxes, etc.) and data visualisation (ECG viewer, Integrated Clinical Environment Web viewer, PACS image list, Patient Administration System connector). Figure 20 shows the designer interface; including the list of plug-ins (left), plug-in properties (bottom-left), link manager (right) and main interface (centre).

Another component in ArQ is the drag-and-drop database query builder which provides a simple interface to construct sets of queries to be used across different databases. It provides strong querying tools for users with no experience with query languages. It also eliminates the boundaries of different query languages used across databases.

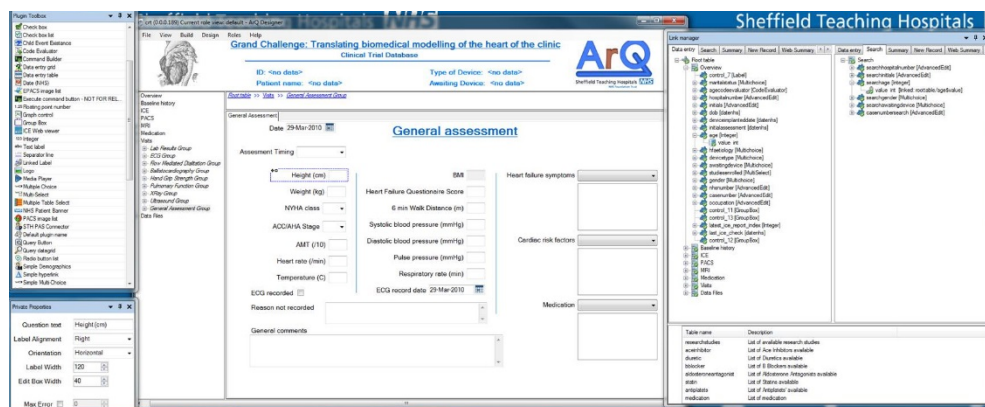


Figure 20: The ArQ Designer interface. On the left, the list of plug-ins can be seen. On the right, the relationships between database fields can be seen on the right.

A preliminary specification for a cardiology database was developed by St. Thomas Hospital (London, UK) – this was developed into an equivalent ArQ database, taking data management, efficiency and clinical use into account. Added features include field descriptions and improved layout design and visualisation aids. The field descriptions are part of the effort to build a comprehensive reference information model for CRT with bio-modelling aided decision protocol. Where appropriate, the database allows modification without compromising existing data (for instance, expanding a medication list). The database now also allows direct access to imaging data whilst ensuring a structured file system. ArQ does not open images natively; instead it opens the appropriate patient images in the local image viewer. Upon creating a new patient record, the program will attempt to find existing patient data on any accessible hospital system and fill the appropriate fields automatically.

Given that some current hospitals system are not electronically based (or are scanned images of paper forms) the process of automatic retrieval is not possible. In such cases, ArQ becomes the new database for such results – during the trial this means duplicate work. Table 4 shows the clinical assessment from a data collection and storage perspective. Note that the clinical perspective of assessing the patient is indifferent of the computer storage systems including the laptop, which is not connected to the hospital network, and the paper-based records. Another important aspect is the structure of data. Clinicians most commonly record data as free text, or note form, which is not problem when data is interpreted by a human. In this format, the contents of records is likely to include non-standard terminology and unexpected observations but is a very natural way of recording information. On the other hand, structured storage is more useful for automating tasks using computers.

ASSESSMENT	STORAGE SYSTEM	STORAGE FORMAT
Demographics	PAS	Structured
History	Paper record	Free form
Image data	PACS	Structured
Image results	PACS	Free form
Blood results	ICE	Structured
Flow-mediated dilatation (FMD)	Laptop	Free form
HF Questionnaire	Paper record	Structured
Exercise tests	Paper record	Both

Table 4: Clinical data storage in current study before implementation of the alternative to digital database system

Figure 21 shows the deployed ArQ database interface, the left pane tree view shows the sections of the patient record. The database is divided into sections, such as patient information, medications, MRI, visits (or assessments), etc. Each section maps into a database table. In the case of assessments, patients make multiple visits to the clinic, each of which may be a different assessment type. In order to display this in a manner that is clinically meaningful, a plug-in was developed to extend the functionality of the database system. Figure 21 shows the “multiple table select” plug-in, displaying multiple visits sorted by assessment type (can also be sorted by appointment date). One of the visit type is the general assessment (shown in the Figure 20).

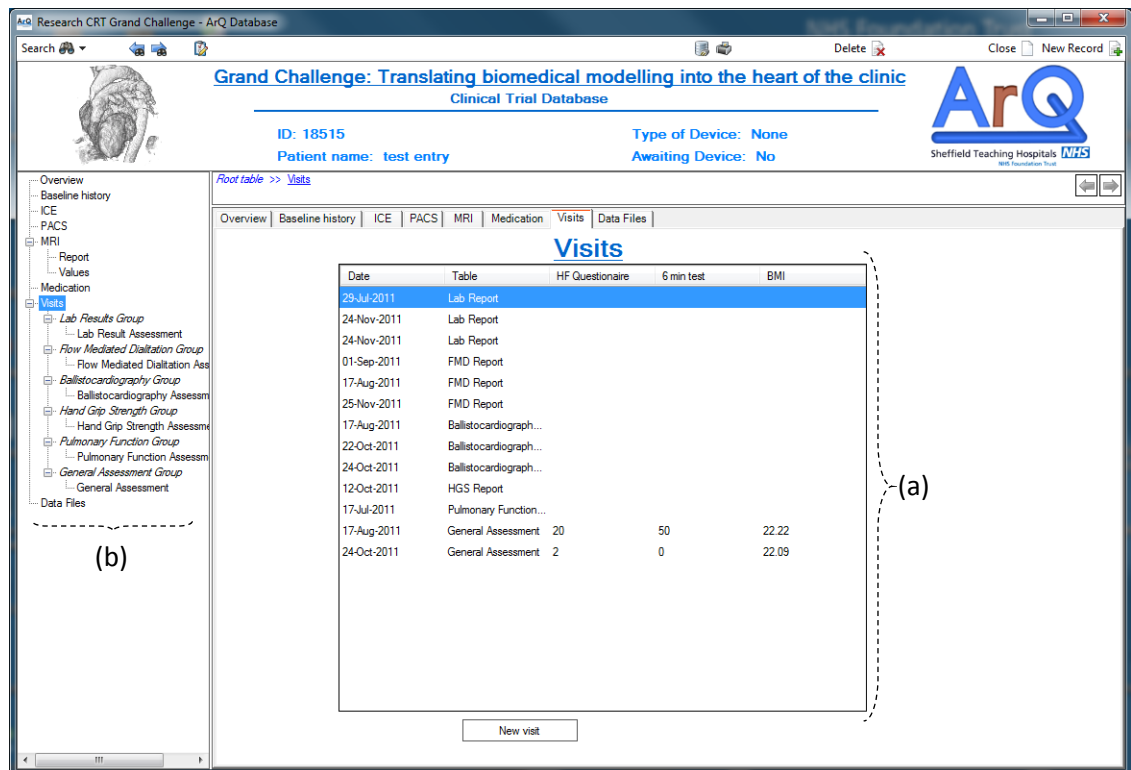


Figure 21: the “visits” page of the clinical database, showing (a) the multiple table select plug-in and (b) the page selector tree view. Combining multiple visits (of different assessment types) into one folder is an interesting feature because each visit type uses a different table in the underlying database structure.

Patient information must be anonymised to grant researchers access to the information. An ArQ plug-in was developed that allowed database designers to point at patient directories and create anonymised copies for researchers. The code implemented the ClearCanvas C# libraries but added functionalities, such as adding a research patient ID, keeping data series and studies under the same ID and setting dates and times to random values. This enables linking files to research cases, viewing images in a logical and ordered manner, and avoiding creating confusion to possible viewing software (which require key values to be present to view DICOM files).

The Grand Challenge CRT protocol utilises two image modalities: cardiac magnetic resonance (cMR) and cardiac echo (ultrasound, US). These are a subset of a larger set of possible images that may be acquired for diagnosis. The combination of MRI and US, required for the simulation workflow shown in Figure 10, is not commonly used for patient diagnostic. The workflow therefore needs to be able to locate the appropriate images in the hospital system and pull them into the research server; this will be done by inspecting the Dicom header tags. Once in the research server, files will be stored in a structured manner to allow the ArQ database to find them automatically. Figure 22 shows the Dicom anonymisation plugin.

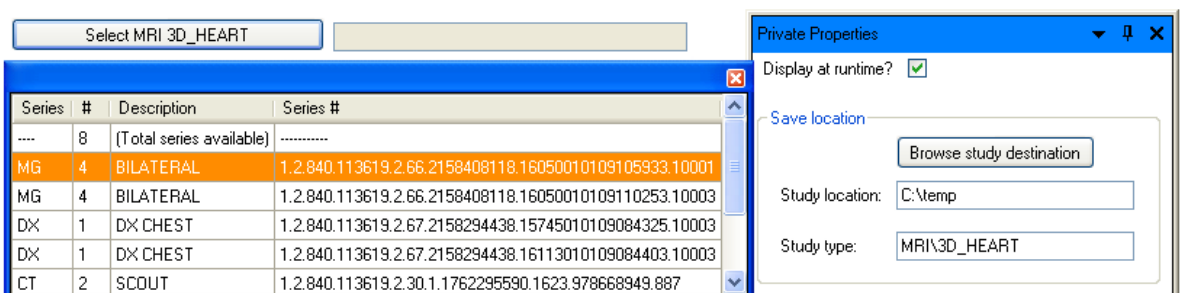


Figure 22: The ArQ Dicom study anonymisation plug-in (left) and the behind-the-scene properties of the plug-in. Users click on the button, navigate to a directory in their system and a drop-down with the Dicom studies present in it appear. The chosen study is copied to the storage directory and anonymised.

## 2.5.2 The workflow system

Windows Workflow Foundation (WF) is part of Microsoft developments of the .Net framework. Rather than a software bundle, WF is a set of libraries that allows developers to create their own programs, representing the wide and abstract capabilities of computational workflows. Whilst this means it is not a direct solution, it allows custom implementation of workflow engine capabilities. WF is intended to be as general as possible to maximise the customisable aspects developers can make use of; applications range from ordering systems to web applications [80]. In addition, WF libraries include extremely useful concepts that are not generally found in many

other frameworks. WF is provided as a set of C# /C++ libraries, natively supported by the most recent Windows operating systems, and highly documented in the MSDN community.

Regardless of workflow type, WF workflows are intended to be designed as logic implementations (decision making) rather than tool-chained processes; the workflow should make a decision and call an external program or service to run. Small procedures, such as string adjustment or simple maths should be done by the workflow. The graphic designer allows a familiar and intuitive drag-and-drop interface that builds workflow logic; this can be then stored as a mark-up file, code file, or a combination of both. Mark-up files don't require any special development platform and can be programmatically created.

Although not integrated as part of its free release version (Express), Microsoft Visual Studio provides full support for graphical development and debugging. The graphic designer however, doesn't fully provide the user experience one would expect. Customisation and the added .Net integration, such as SQL server, WCF and SharePoint, is what makes WF stand stronger than other competitors (mostly Java based libraries). Cross-platform integration can be achieved using Mono (implementation of WF v4 announced but not delivered [81]) and other tools, such as windows services for UNIX. Some important libraries in WF are the tracking, communication and persistence services. The tracking service adds event handlers to state flow allowing implementing a log of when and how a workflow state changed. The communication service aids the execution of external services using WCF security. More importantly, the persistence service allows workflows to be saved in memory whilst it waits for the external execution to finish or an event arrives, hence saving processing capacity, allowing up to thousands of workflows to run at one particular time and adding a safety feature in case of hardware failing.

Services are the command line processes optimised for background operation by Windows. They are designed to run without user intervention and provide task automation. WF allows development of custom services which can be called by the workflow or creating workflows to run as custom services – extending workflow programming concepts into core Windows components.

An overview of the WF architecture can be seen in Figure 23. The behaviour of such instances is controlled by a core workflow engine that manages - loads, runs, handles events and implements services of - the particular workflow instances. This also allows different types of workflows to be run at one particular time. The workflow environment developed for this project has been designed and coded using WF.

WF is by no means a perfect implementation but rather a good starting point for developers and an excellent example of a well-thought-out workflow library. Microsoft commercialises a product called SharePoint, that is developed using WF, however it is aimed at corporate management and has no added value to the scientific community [82].

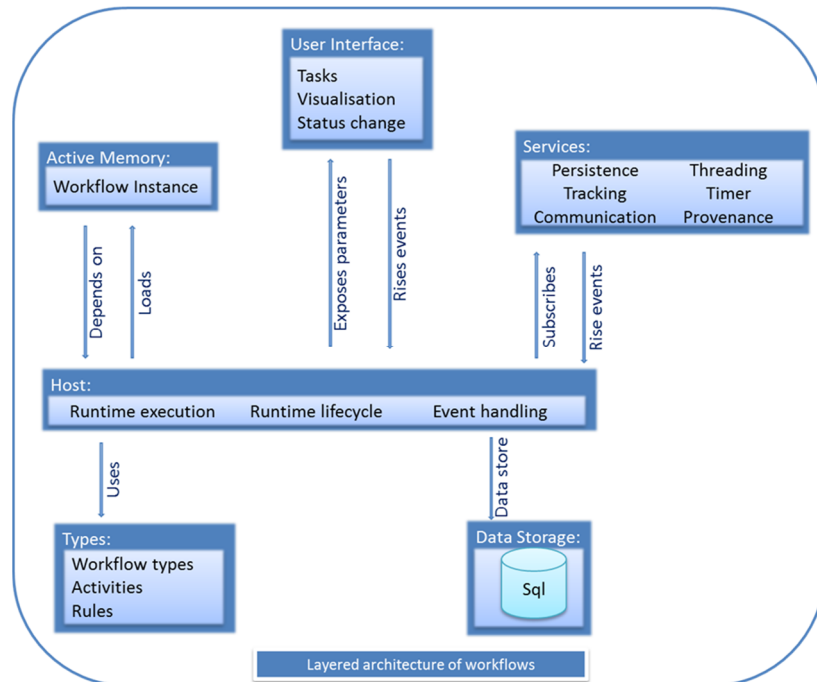


Figure 23: Illustration of the type of layered architecture that might be employed in a workflow system

The Pathfinder workflow engine and user interface has been developed by the Scientific Computing and Informatics group (NHS). At its core, it is built using WF and it provides a web interface designed to allow human interaction with long-running workflows. The workflow engine is strongly coupled with a structured SQL database that stores information allowing workflow definitions, data and execution to be loosely coupled [82], [83]. This design means Pathfinder only needs to be provided with one single workflow definition to deploy hundreds of patient workflows. Decoupling of patient data and workflow instances means the data does not require loading a workflow instance to be loaded into memory – due to WF design it also makes workflow instances much lighter. The workflow is processed in entirely web-based services extending workflow connectivity beyond internal networking. Whilst remaining extremely simple and generic, it provides the necessary solution for the project requirements. On the other hand, Pathfinder is not prepared or designed to handle scientific workflows – it will run sequential workflows, but will not be able to provide interaction or reporting functionalities. Also, Pathfinder is not a completed software bundle and as such still requires work on its user

interface and information presentation. The Pathfinder security model relies entirely on the STH corporate active directory system, meaning it may not be accessed from unmanaged computers. Lastly, Pathfinder compatible workflows are developed in Visual Studio and compiled as dynamic-link libraries thus requiring highly skilled and laborious work to develop workflow definitions.

A specific development introduced to the workflow engine is the ability to accept workflow definitions in markup language – as opposed to C# compiled code. The markup language is in fact XAML (Extensible Application Markup Language), which follows a Microsoft defined schema, but makes it directly compatible with the .NET framework libraries. XOML files (file extension for workflow definitions in XAML language) can contain a list of the workflow tasks, task parameter information and the implementation logic. Task parameters can be dynamically bound. The schema also defines a rules structure to allow coding of conditional statements into the markup files and a layout structure, which is used by the layout libraries to render a visual image of the workflow (in practice, this is not needed). Markup files may not contain raw code, meaning all execution code needs to be contained within the tasks – an obvious workaround is to develop a complex task that parses and compiles raw code but this becomes very impractical.

Advantages of markup files include lighter storage (a whole definition can be stored as a string entry in a database), easy modification (as they are treated as text files), no recompiling needed, parsing with external programs (to view, modify or validate) and dynamic file generation (e.g., by concatenating workflow sections stored in a database with one query). However, these definitions still make use of external libraries that hold task definitions and have a predefined schema. This limitation must be noted if external workflow definitions want to be run using Pathfinder: an external workflow can be parsed into a definition compatible with the engine; however one must ensure availability of equivalent tasks that produce the same output as the definition expects.

A workflow has been designed in conjunction with clinical fellows that allows clinicians to track patient progress and manage active workflows [84]. The workflow engine used will be Pathfinder because it addresses the needs for clinical deployment. For instance, the security model makes use of an LDAP server so it integrates easily for clinical use. In contrast to the scientific applications, there is no desire to allow workflow manipulation in the hospital environment. However, for workflow design and authorisation, a major modification to the Pathfinder core functionality was made to allow it to accept mark-up defined workflow definitions to be loaded into the engine. The workflow engine is integrated with ArQ by means of an intermediate service that inspects for new occurrences in the database and start workflows appropriately.



## 2.6 Summary

More often than not, the biomedical researcher finds the need to write software to manage databases or integrate their research in a wider clinical environment. The different working objectives of researchers and clinicians means that there are different working paradigms and technical requirements. This chapter has outlined these differences, exemplified in database and workflow technologies.

The ideas presented in this chapter are relevant because they set the foundations for the future of healthcare. A hospital can be thought of as a conglomerate of highly specialised clinics; which rarely work together but rather just happen to be in close location to each other. However, health is more complex than treating pathologies in isolation. Looking back at the adoption of informatics in healthcare, a first generation of tools have been built to empower humans; these include ontological searches, and comprehensive encyclopaedias of human anatomy and disease. A second generation of technologies are now emerging, these include tools that instead of empowering humans to do more complex work they replace humans at doing complex tasks (and are better at it). Data mining for instance is the process of finding patterns in large datasets using computational methods. Traditionally, simple predictors such as demographics (e.g. age, sex, ethnic background) have been used to classify patients for diagnosis, however the idea of data mining is gaining track (through other industrial advances) and will redefine the approach to delivering healthcare. We are at the dawn of a new approach towards medicine and healthcare, and it is data centric.

# CHAPTER 3

## IMAGE REGISTRATION

This chapter describes the theoretical development of an image registration algorithm in parametric space. Section 3.1 introduces image registration and section 3.2 describes the components of image registration methods. Section 3.3 presents the theoretical development of the algorithm, first in Cartesian space (section 3.3.1) and then in parametric space (section 3.3.2).

### 3.1 Introduction

Image registration is the process of bringing two images into alignment. A general description of the registration process can be broken down into three main stages: a transformation, a similarity measure and an optimisation. Each of these is taken in sequence in the review in the following paragraphs. In practice, these concepts are not implemented in isolation making classification of image registration processes not a straight-forward task. More formally, image registration is the search for a set of parameters of a transformation model that when applied to an image maximises its similarity with respect to another image.

The challenge addressed in the remainder of this thesis is the morphing of a template model to a patient image (see chapter 1, section 3, Figure 10). Image registration is a natural approach to solving this. This thesis builds on the basic idea of registering the image of a reference subject to a patient. If then the image of the reference subject is segmented, however this is achieved, the resulting model can be morphed to create a patient model by using the registration field derived from the images [21], [85].

A similar approach is to register the binary image of a template mesh to the segmentation of the anatomical structure of the patient, and using the deformation field to morph the mesh [86], [87].

The next evolution of this approach is to make the simulation mesh the central component of the simulation. The simulation mesh is treated as a binary image, with the important difference that it is no longer defined using a regular grid. The mesh is then registered to the segmentation of the patient's anatomical structure. This chapter will describe this process.

### 3.2 Components of Image Registration

Extensive surveys of image registration methods can be found in the literature: Brown [88] surveyed the literature for techniques for general image registration whilst Maintz and Viergever [89] surveyed specifically medical image registration techniques. A more recent review focusing on medical image registration by Sotiras et al. [19] is structured around the components of image registration described in Figure 24.

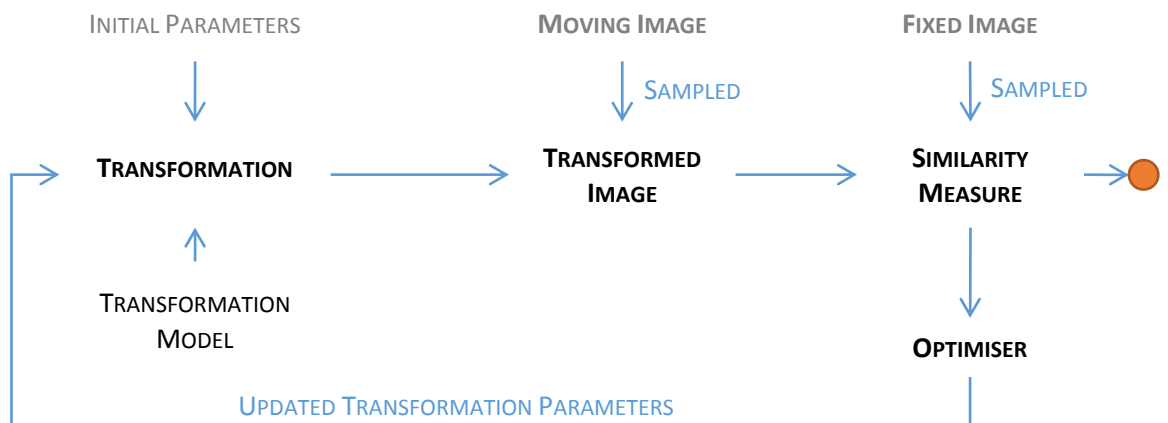


Figure 24: Components of image registration.

The source of mis-registration is the primary criteria for selecting a particular registration procedure. The reason for registration is determining the difference between two images (after all, the output of the process is a deformation field), so it is important that images are matched only with regards to the mis-registration source. For instance, using image registration for particle tracking would most likely favour a rigid/linear transformation because the shape of the particle doesn't change, whereas tracking the muscle of the beating heart certainly requires a non-linear transformation because of the lifting and twisting of the apex, as well as the changing width of the muscular wall. This is an important aspect to take into account when comparing image registration algorithms because the trivial solution to registration is changing each voxel in the image to match the intensity of the target.

In general, two transformation types can be encountered in the literature and provide the most general classification of image registration methods.

### 3.2.1 Transformation model type

#### 3.2.1.1 Feature detection

Landmark or feature detection methods register images by aligning a number of control points defined by features in the image; most commonly these are points, edges or corners however more complex structures can be used. Some or all of the landmarks may need to be located manually. This is more likely needed when complex structures are used as control points. A typical filtering procedure required is the elimination of outlying landmarks. Feature extraction is a process independent from the alignment meaning the registration can be performed across images of different modalities.

#### 3.2.1.2 Intensity based registration

Intensity based registration methods map image pixels based on their intensity values or intensity pattern. Intensity based registration methods sometimes require a rigid-shift pre-processing step to align the images. This may be automated using image metadata or features from the images themselves, but in practice is often most robustly achieved manually. This kind of transformation model generally assumes there are small differences between the images. Images of different modalities encode information with different intensity maps, as a result meaningful multi-modality image registration is not trivially implemented using intensity based transformation models.

Typically, and the value of this should not be underestimated, intensity based registrations are performed multiple times with increasing complexity of the transformation model. For instance, by first using a rigid transformation model, then a linear model and then a non-linear model. This slowly brings the images into registration at a lower computational cost and may remove the interference of noisy images.

The transformation required to achieve registration is usually calculated at the nodes of the registration grid, which may be coarser than the image pixel grid; the displacement of each voxel is then obtained by interpolation. Another common approach is to vary the registration grid spacing; this is called the multi-level or multi-grid approach by different authors. Essentially, the number of degrees of freedom is increased as the registration progresses to achieve a similar effect as before.

### 3.2.2 Transformation models

The deformation model defines the transformation function applied to the moving image. It is often a more complex form than the elementary transformations (translation, rotation, affine). The choice of deformation model comes down to image type and problem definition.

Transformation models may be global or local. Typically, global transformations are either rigid or affine and local transformations are free-form. Global transformations seek a general function (i.e. the same for all voxels) which is then applied over the whole image domain. Function variables may be location dependent. In local transformations the mathematical functions applied are dependent on the voxel location. Local transformations allow for locations to have transformation models with different set of parameters. Thus, a global transformation may be described at the image corners yet use a polynomial interpolation scheme to define the voxel-wise transformation [19].

This should not be confused with whether the transformation variables are locally or globally computed. Global variable computations (parameter optimisation) include information from the whole image to determine the transformation parameters, as opposed to employing only local regions. This thesis uses piecewise continuous registration fields defined over a series of discrete elements, in direct analogy with the finite element method; it is a global transformation model where the transformation parameters are calculated locally.

The simplest models are rigid body and affine. Rigid body transformations allow only translation and rotation of the images whilst affine transformations allow translation, rotation, skewing and scaling.

Common rigid body algorithms for surfaces include:

- Head-to-hat, where the reference surface is high-resolution (i.e. fine detail) and sets of points define a rigid structure. The points are unconnected but are kept in position relative to each other. The registration moves the points to reduce a norm of the distance from the points to the surface. Computing the distance transform of the images may significantly reduce the computational time of head-to-hat. Distance transforms are a derived representation of the images where each voxel is given a value relative to its distance from the closest boundary.
- Iterative closest point, similar to head-to-hat but both images are treated as a cloud of points. A least square rigid body transformation is applied to fit the points to the triangles.
- Crest lines, for which one needs smooth surfaces (differentiable to the 3rd order). First one defines principal curvatures of the surfaces, then finds the loci of the surfaces (called the

crest lines), and finally aligns the crest lines. This is particularly useful for images where intensity has real meaning (e.g. tissue properties), the surfaces are trivially calculated (e.g. CT bone).

Rigid-body or affine registration may employ a point based registration method which involves identifying corresponding points (homologous landmarks) in the images; the registration then aligns these points and optimises the alignment of other voxels by least squares [90]. Alternatively, an image may be constrained to transform in a rigid-fashion by a general vector field [21].

Free-form transformations are defined by polynomials in terms of the individual pixel location. For instance, elastic models are inspired by linear or non-linear elastic theory and provide a far more flexible model than rigid body or affine. A more detailed discussion of this type of transformation model can be found in Chapter 4.

### 3.2.3 Similarity metric

The similarity metric is a critical component of the registration process. It is necessary to define the criteria by which two images are judged to be similar. There are two basic approaches, one that measures the difference in image intensities across the whole image and the other that uses select image features. Hybrid methods that combine both approaches also exist.

Iconic methods evaluate the intensities across the whole image domain. The representation of intensities varies across image modalities, thus iconic methods may be mono-modal or multi-modal. The latter can be reduced to a mono-modal case by transforming one image to the other's domain, or by transforming both into a third domain.

A vector can be constructed, each entry of which is the difference in intensity at one voxel in the image. Any vector norm can be used to provide a similarity measure. Defining A and B as the images, i as the index of individual pixels and N as the total number of pixels allows mathematical definitions of each metric. The simplest approach is using the direct intensity difference between images, all voxels contribute equally towards the measure:  $DIR = \frac{1}{N} \sum_i |A(x_i) - B_t(x_i)|$ . The most commonly used vector norm is the sum of the squares:  $SSD = \frac{1}{N} \sum_i |A(x_i) - B_t(x_i)|^2$ .

More complex approaches use the cross-correlation of intensities  $\sum_i (A(x_i) * B_t(x_i))$ , the normalised cross-correlation of intensities  $\frac{\sum_i (A(x_i) - \bar{A})(B_t(x_i) - \bar{B}_t)}{\sqrt{\sum_i (A(x_i) - \bar{A})^2 \sum_i (B_t(x_i) - \bar{B}_t)^2}}$  or the ratio of image

uniformity  $\frac{1}{R} \sqrt{\frac{1}{N} \sum_i \left( \sum_i \frac{A(x_i)}{B_t(x_i)} - \overline{\sum_i \frac{A(x_i)}{B_t(x_i)}} \right)^2}$  which are (normalised) measures of the deviation of

voxel intensities in the images. These approaches tend to reduce small intensity changes which may be useful for noisy images.

### 3.2.4 Optimisation

This refers to the process of updating the parameters of the transformation function to improve the similarity metric. Physical and biophysical models iteratively solve a minimisation problem so that as the registration progresses the parameters are updated towards an optimal solution. The pre-conditioned conjugate gradient method is commonly regarded as the best optimisation method in this category. This is a variation of the gradient descent method which offers improved convergence rates and is less sensitive to local minima. In situations where the preconditioning of the method is not done appropriately it may perform worse than the gradient descent method [40]. On the other hand, interpolation methods employ non-differentiable transformation models. As such, these methods rely on discrete methods to find the parameters that solve the registration. They perform a global search over the parameter space and employ complex optimisation methods that are out of the scope of this thesis.

## 3.3 Development of the registration algorithm

### 3.3.1 Intensity-based registration in Cartesian space

At the beginning of this work, the Department of Cardiovascular Science already had a registration toolkit (ShIRT, the Sheffield Image Registration Toolkit, [21]), developed over the last decade, based on an optical flow algorithm. Indeed, this toolkit was already in use in the Grand Challenge project as part of the process of morphing meshes developed at KCL [33], [87], [91]. As discussed previously, the earlier application used the existing ShIRT software to register a template image, defined on a regular Cartesian grid, to the patient image. The software developed in this thesis is also based on optical flow, but with the important distinction that the fundamental representation of the image to be mapped (the ‘moved’ or ‘moving’ image) is described in parametric space, with direct analogy to the parametric series of finite elements. This section develops the equations to be solved to generate the mapping between moving and target images. The derivations are presented in two dimensions for the sake of clarity, but extension to 3D is straightforward.

Given two images,  $f$  (fixed or target image) and  $m$  (moved, moving or source image), a spatial mapping is sought that, applied to the moving image, would make the two images identical, or more likely would minimise a measure of the differences between them. It might be convenient to describe the mapping as an operator,  $U_{m \rightarrow f}$ , so that:

$$f(x, y) = U_{m \rightarrow f}(m(x, y)) \quad (1)$$

The application of  $U_{m \rightarrow f}$  to  $m$  produces  $f$ .

Similarly,  $f$  could be mapped to  $m$  using another operator,  $U_{f \rightarrow m}$ :

$$m(x, y) = U_{f \rightarrow m}(f(x, y)) \quad (2)$$

Figure 25 illustrates the processes of mapping between two images,  $f$  and  $m$  (sampled on a finite grid).

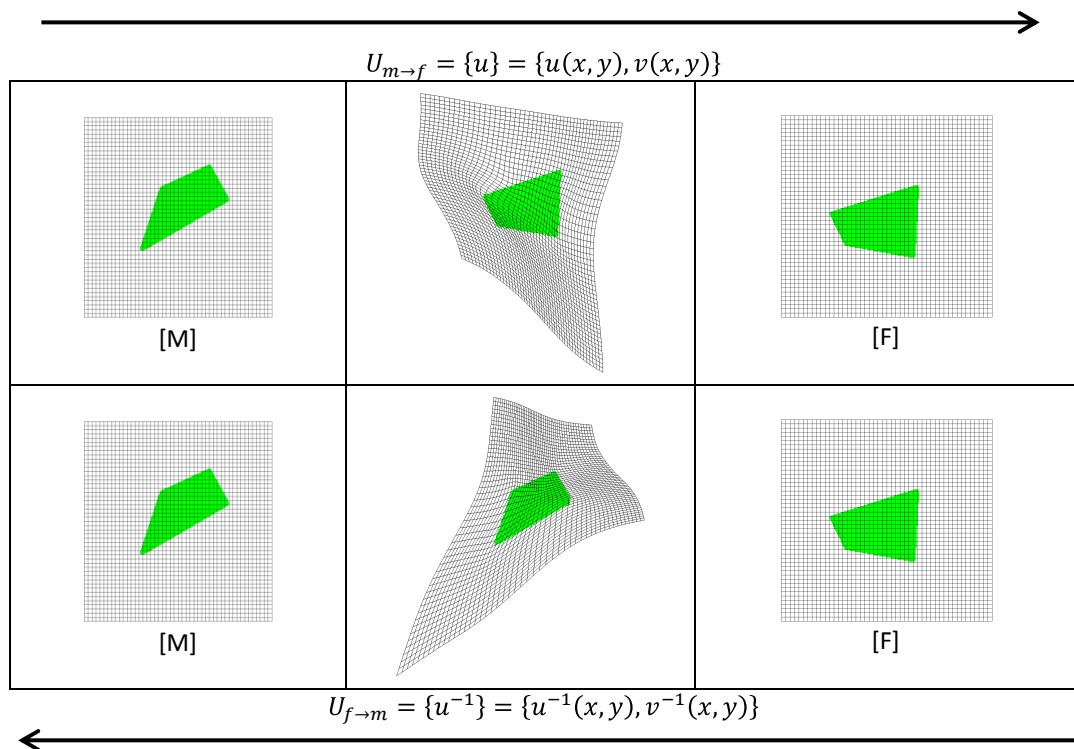


Figure 25: Forward and inverse registration mapping. The top distorted grid relates to image [M] and the bottom one to [F]. (Figure generated using SHIRT)

Important characteristics of the mapping  $U$  might include continuity, differentiability and invertibility. A diffeomorphic mapping is one that is continuous, one-to-one and differentiable:



such a mapping maintains the topology of the imaged structures [92]. In such case, it is required that:

$$m(x, y) = U_{m \rightarrow f}^{-1}(f(x, y)) \quad (3)$$

For a diffeomorphic transformation

$$U_{m \rightarrow f}^{-1} \equiv U_{f \rightarrow m} \quad (4)$$

### 3.3.1.1 Pushing m onto f

In this expression of the problem one looks for a point  $(x, y)$  in the image  $f(x, y)$ , that has the same intensity as the image  $m(x, y)$  at  $(x', y')$ . The moved image is to be transformed under the mapping  $\{u\}$  so that it matches as closely as possible the fixed image. In other words, one must find the vector  $\{u\}$ , evaluated at each of the points in the Cartesian grid on which the image is defined, that minimises the intensity difference  $[m(x, y) - f(x, y)]$  between the images.

The intensity,  $f(x', y')$ , of the fixed image in the vicinity of  $(x, y)$  can be determined from a 1<sup>st</sup> order Taylor series expansion:

$$f(x', y') = f(x+u(x, y), y+v(x, y)) = f(x, y) + u(x, y) \frac{\partial f(x, y)}{\partial x} + v(x, y) \frac{\partial f(x, y)}{\partial y} \quad (5)$$

After the mapping:

$$m_1(x, y) = f(x', y') \quad (6)$$

The difference between  $f(x', y')$  and  $m_1(x, y)$  is the residual,  $R(x, y)$  at this point in the image after the mapping has taken place. It should be noted that in this formulation the residual is evaluated at the positions to which the moved image has been pulled rather than at the points on the original grid.

Using a shorthand form, given that  $f, m, u$  and  $v$  are always evaluated at  $(x, y)$ :

$$R = f - m_1 = f - m + u \frac{\partial f}{\partial x} + v \frac{\partial f}{\partial y} \quad (7)$$

An alternative is to evaluate the residual,  $R$ , at the original grid points. In order to do this the image  $m_1$  has to be re-sampled at these positions, or, to look at the problem another way, the value of  $m$  at the point  $(x', y')$  that will be moved to  $(x, y)$  needs to be determined. This is illustrated in Figure 26.

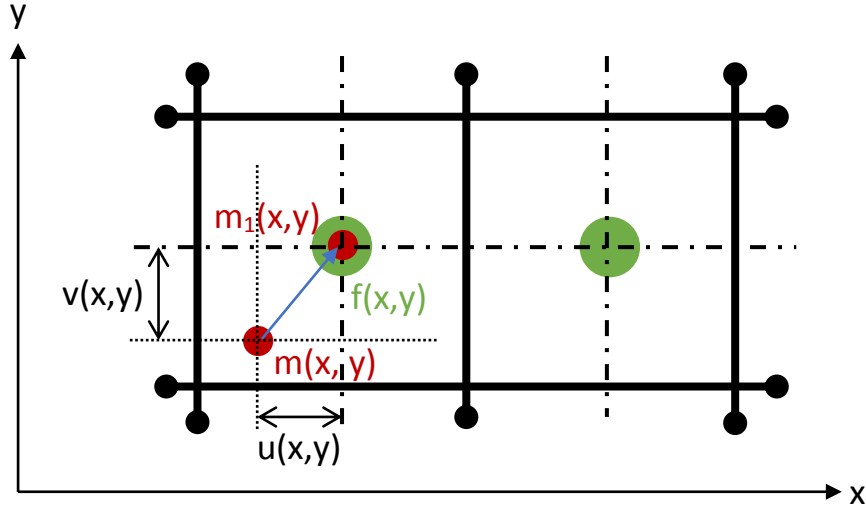


Figure 26: Re-sampling of moved image after mapping to fixed points on original Cartesian grid. The thick lines delineate the voxel area, the intermittent lines connect the voxel centres (green). The moving voxel (red) must have undergone a mapping  $\{u, v\}$ , based on the position and gradients at  $f(x, y)$ , to arrive at  $m_1(x, y)$ .

Again using the Taylor series expansion about  $(x, y)$ :

$$m_1(x', y') = m(x - u, y - v) = m - u \frac{\partial m}{\partial x} - v \frac{\partial m}{\partial y} \quad (8)$$

The residual at the original image points is then:

$$R = f - m_1 = f - m + u \frac{\partial m}{\partial x} + v \frac{\partial m}{\partial y} \quad (9)$$

The average of the residual evaluated at the points  $(x, y)$  and  $(x', y')$  is:

$$\bar{R} = \frac{R + R}{2} = f - m + \frac{1}{2} \left( u \frac{\partial f}{\partial x} + u \frac{\partial m}{\partial x} \right) + \frac{1}{2} \left( v \frac{\partial f}{\partial y} + v \frac{\partial m}{\partial y} \right)$$

$$\bar{R} = f - m + \frac{u}{2} \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) + \frac{v}{2} \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \quad (10)$$

The averaging of the residual at the two points might be regarded as a dangerous process because in principle large positive and large negative residuals could cancel, but in practice this is unlikely to happen because the gradients of the displacement fields are small. In practice Sheffield's experience in the development and testing of ShIRT has been that equation (10) works better than the earlier expressions for the residual in achieving optimal registration in the fewest iterations, perhaps because it uses spatial gradient information from both images.

The intensity values are defined at the voxel centres. At the beginning of the registration process, for two images that are aligned in space, calculating the residual at the centre of the image voxels is trivial. In further iterations, to calculate the residual in a consistent manner images need to be resampled because the voxel centres become un-aligned. Equally, since the points at which the residual is calculated are not the same as the image voxels locations, the correct intensity values need to be calculated. This process is referred to as resampling. Figure 27 shows the misalignment between the voxel locations and the points at which the residuals are calculated; the right-most panel shows the movement of the voxel centres of the moving image.

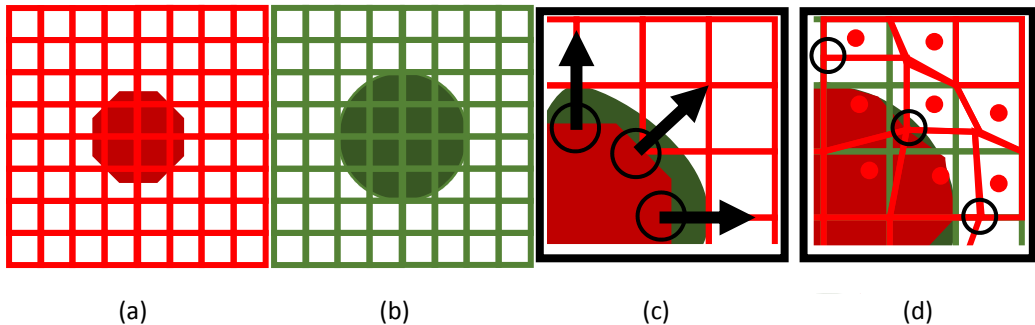


Figure 27: Voxel centres, where the residuals are calculated, become unaligned during the registration. The figure shows (a) the moving image, (b) the fixed image, (c) the initial overlap and (d) the overlap and distorted grid after one iteration.

An alternative way to consider equation (9) is that resampling of the mapped image is such that, if the mapping achieves the alignment of  $m$  with  $f$ , it is exactly the negative of the inverse transform; i.e. it is the mapping that takes  $f$  onto  $m$  rather than  $m$  onto  $f$ . In these terms:

$$R = f - m_1 = f - m - u^{-1} \frac{\partial m}{\partial x} + v^{-1} \frac{\partial m}{\partial y} \quad (11)$$

The average of the residual evaluated at the points  $(x, y)$  and  $(x', y')$  is:

$$\bar{R} = \frac{R + R'}{2} = f - m + \frac{1}{2} \left( u \frac{\partial f}{\partial x} - u^{-1} \frac{\partial m}{\partial x} \right) + \frac{1}{2} \left( v \frac{\partial f}{\partial y} - v^{-1} \frac{\partial m}{\partial y} \right) \quad (12)$$

For small displacements,  $\{u\} = -\{u^{-1}\}$  and equation (12) reduces to equation (10).

The expressions for  $U$  and  $U^{-1}$ , as in equation (12), could be maintained as independent operators, with additional constraints imposed to ensure consistency of the inverse operation. Better still the residuals can be treated as separate cost functions, as in equations (7) and (11), again with additional inverse-consistency constraints. This is essentially the approach described by [92], extended to large displacements by [93]. In a sense equation (10) is invoking a small displacement inverse consistency formulation because it makes an assumption about the relationship between the mapping and its inverse.

If, however, the point of view of the moving image is taken and the residual is consistently worked out at the centre of the moving image's voxels, there is no need for the resampling process.

### 3.3.1.2 Pulling $m$ onto $f$

This alternative expression focuses on a point  $(x', y')$  in the image  $m(x, y)$  that has the same intensity as the image  $f(x, y)$  at  $(x, y)$ .

In Figure 28, functions  $f, m, u, v$  are defined at all points  $(x, y)$ .  $f, m, u$  and  $v$  are the particular values that they take at the point  $(x, y)$  on the fixed Cartesian grid on which the image is defined. The dark green dot and dark red dot illustrate the values of  $f$  and  $m$  respectively at  $(x, y)$ . The pale red dot illustrates the value of  $m$  at position  $(x-u, y-v)$ , in the vicinity of  $(x, y)$ , at which the intensity of  $m$  has the same value as that of  $f(x, y)$ . The arrow indicates that the moved image will be pulled from  $(x-u, y-v)$  to  $(x, y)$  under the mapping.

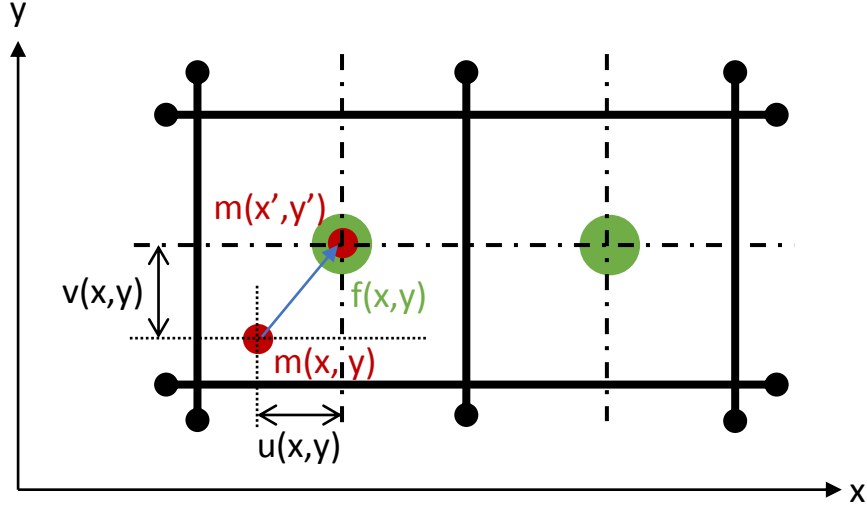


Figure 28: Schematic of pulling a voxel of  $m$  onto  $f$ . The thick lines delineate the voxel area, the intermittent lines connect the voxel centres (green). The moving voxel (red) undergoes the mapping  $\{u, v\}$ , based on its position and the gradients at  $m(x, y)$ , to take it to  $m(x', y') = m_1(x, y)$ .

The intensity,  $m(x', y')$ , of the moving image in the vicinity of  $(x, y)$  can be determined from a Taylor series expansion:

$$\begin{aligned} m(x', y') &= m(x - u(x, y), y - v(x, y)) \\ &= m(x, y) - u(x, y) \frac{\partial m(x, y)}{\partial x} - v(x, y) \frac{\partial m(x, y)}{\partial y} \end{aligned} \quad (13)$$

After the (imperfect) mapping:

$$m_1(x, y) = m(x', y')$$

$$m_1(x, y) = m(x, y) - u(x, y) \frac{\partial m(x, y)}{\partial x} - v(x, y) \frac{\partial m(x, y)}{\partial y} \quad (14)$$

This repeats (8). The difference between  $f(x, y)$  and  $m_1(x, y)$  is the residual,  $R(x, y)$  at the centre of the pixel after the mapping has taken place. In this formulation of the problem the residual is evaluated immediately at the grid points.

$$R = f - m_1 = f - m + u \frac{\partial m}{\partial x} + v \frac{\partial m}{\partial y} \quad (15)$$

Equation (8) is identical to equation (14), and again it could be expressed in terms of the inverse transform as in equation (7). As before, the residual could also be evaluated at the point to which  $m$  is mapped under the transformation, and the result is again equation (9). The averaged residual is again identical. Hence it has been demonstrated that the computed residual is independent of the expression of the problem: pushing  $m$  onto  $f$  and then interpolating back to the fixed grid produces the same result as pulling  $m$  onto  $f$ .

The derivation of the residuals includes the seeking of a point in one image, in the vicinity of a point in the other, that has the same intensity. The identification of such a point is based on a Taylor series expansion, which holds only in the near vicinity of the starting point (potentially extended by the adoption of higher order terms in the expansion). After the application of the mapping field to the moving image, a new image,  $m_1$ , is derived. This can be interpolated back to the fixed points of the original image to become  $m_1$ , and then the process can be repeated iteratively until the residual is minimised. The interpolation process is not entirely trivial because the image between the sampling points is defined in linear patches, and it is first necessary to identify which patch overlays the relevant fixed point. Furthermore the accumulation of the displacements at each increment is also not trivial: the displacement at the second iteration is computed at the fixed point, whilst the point that was originally there has moved under the mapping. Once again interpolation is required to integrate the mapping field over a series of iterations. Figure 29 illustrates how the displacement field computed at each iteration transforms the moving image until the difference in similarity reaches a minimum.


	ITERATION	RESIDUAL
	$m_1(x, y) = U_1(f(x, y))$	$R_1$
	$m_2(x, y) = U_2(m_1(x, y))$	$R_2$
	$m_3(x, y) = U_3(m_2(x, y))$	$R_3$
	...	
	$m_p(x, y) = U_p(m_{p-1}(x, y))$	$R_{min}$

Figure 29: Summarises the iterative update of the moving image, the residual converges to a minimum with successive iterations

At the point where the residual becomes a minimum, we can state that:

$$f(x, y) \approx m_p(x, y)$$

And the registration has come to an end. The simplified schematic of the iterative process is shown below:

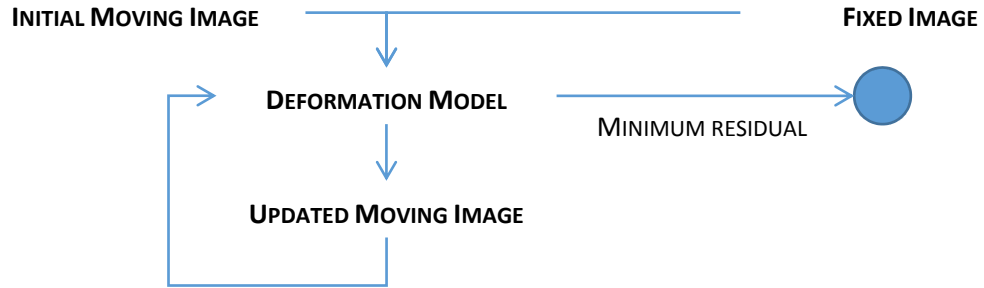


Figure 30: Flowchart showing the iterative registration process described in this chapter.

The extension to 3D is trivial, and the residual becomes:

$$R = f - m + \frac{u}{2} \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) + \frac{v}{2} \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) + \frac{w}{2} \left( \frac{\partial f}{\partial z} + \frac{\partial m}{\partial z} \right) \quad (16)$$

The optimisation of the parameters of the transformation matrix is an ill-posed process and regularisation is therefore used to improve the stability of the registration process.

ShIRT deals with this by expanding the image dimensionality by one, so that all images become binary, with the additional dimension representing intensity. This has the effect that any image can, in principle, be mapped exactly to any other image simply by adjustments in the intensity dimension. In ShIRT, the intensity is a variable and it contributes to calculating the value of the residual. This solution is not, however, unique and many other solutions exist in this higher dimensional representation. The selection of one particular solution is governed by the regularisation term.

Some registration processes, such as deformable models employ regularisers that enforce physiologically plausible deformations [19]. In other words, regularisation can be used to make the registration behave in a particular manner. The downside of these methods is that the regularisers contain a high number of parameters which can make them less practical.

The application of this work focused on employing image registration methods as a way of generating anatomical models for computer simulations. For this application, mesh quality is a crucial property. It dictates whether the solution will converge at all, and if it does how accurate it will be, the rate at which the solver will iterate towards the solution and as a result, the required computational time.

Several mesh quality metrics have been proposed and are used to determine the likely stability and accuracy of the solution [94]–[96]. Commercial finite element analysis systems include guidance on acceptable ranges for mesh quality metrics for specific applications. Factors affecting mesh quality are related to the geometry of the elements. For instance, the more skewed or the higher the aspect ratio of an element, the worse its quality. However, additional factors such as smooth distribution of element sizes are known to impact the overall mesh quality.

A least-squares approach is used to optimise the registration parameters. Finding a stable solution is not that simple. In practical image registration problems, a typical problem is finding local minima with the resulting effect of the optimisation process not completing the registration. To overcome this and make the application of image registration more robust, a number of considerations are taken.

Pre-registration of the images using a model of reduced complexity (i.e. rigid or affine) to bring the images into coarse alignment. Using reduced models in distant images may also avoid undesired (e.g. unsmooth) non-linear maps. Choosing a stricter (or more rigid) regularisation and relaxing the condition once the images are closer in similarity is another approach, this relies on the regularisation term not depending on image intensities. For instance, one might use a stiffer registration grid where large deformations are required to avoid foldings in the grid. Image smoothing removes image details (which is equivalent to an averaging process helping avoid local minima) and decreases the magnitude of the image gradients (which increases the validity of the first order Taylor approximation). Hierarchical optimisation (also called multi-grid or multi-resolution) is the process of using a coarse registration grid and increasing the number of nodes as the images become increasingly similar. A side effect is that it reduces the computational time of the overall process [97].

### 3.3.2 Intensity-based registration in parametric space

As discussed above, the fundamental process, when applied to a moving image that is defined always on a fixed Cartesian grid, involves a series of interpolation operations, both of the images themselves and of the mapping field. The natural development for the current application is to define the moving image in a parameterised space in which the voxels are translated by each mapping operation. This solves the need to resample the moving image after each iteration (although it will be necessary to sample the fixed Cartesian image onto the moving image, this is a trivial operation). As will be discussed later, there is another major advantage of this approach, which is that the quality of elements of the mesh that make up the image in



parameterised space can readily be determined, and these quality measures can be used as regularisation parameters in the control of the mapping process to prevent shape degradation under the mapping.

Typically in the representation of an image it is assumed that the intensities at the centre of each voxel are representative of the value through the voxel, i.e. it is assumed for display purposes that the intensity is constant within the voxel. In the derivations outlined above, the image at all points in space is defined by interpolation from its sampled values at the Cartesian locations. In parametric space a similar interpolation is made, but in terms of parametric rather than Cartesian co-ordinates. Where regular images are divided into voxels in a structured and regular sized grid, this new image is defined in an unstructured and irregular grid, which might be called 'nodes'. The nodal values are interpolated at discrete points called sampling points. The sampling points are defined in a structured and regular grid within the element, in the element space. Since the nodes are defined in a coordinate system that can be compared with the Cartesian image, the sampling points can be mapped into the image space.

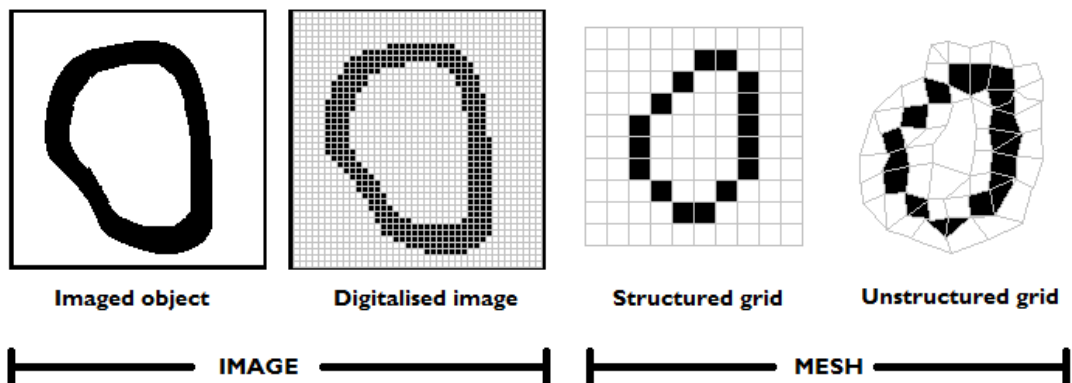


Figure 31: Comparison of grid types (unstructured and structured) with the structured nature of a digital image, which is typically described in Cartesian space.

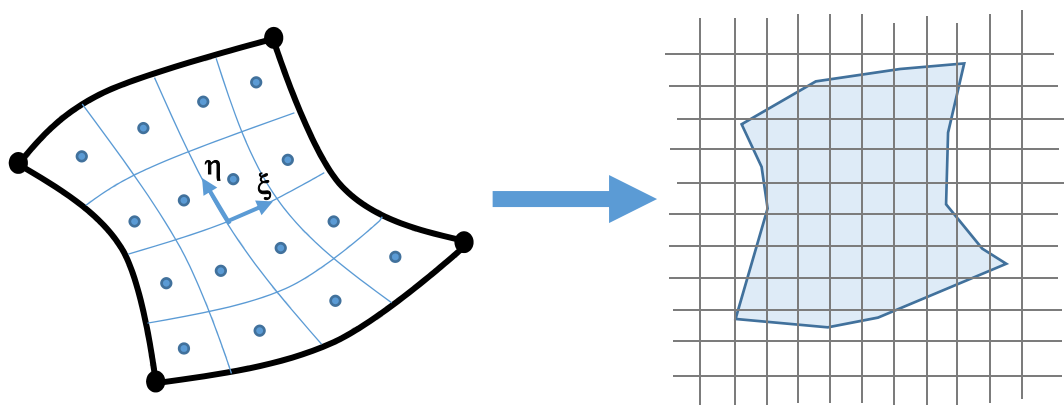


Figure 32: Local coordinate system (left) and its location within the Cartesian image

The aim is to develop the operation to register an image that is defined in parametric space to one that is defined in Cartesian space. The displacement field at the sampling points  $(u(x, y), v(x, y))$  is defined by the values  $(u, v)$  at the node points plus the description of the shape function,  $N$ .

A mapping that takes all of the blue dots in the image in parametric space to points in the fixed image that have the same intensity is required. At each of the blue points the co-ordinates in Cartesian space are  $x = [N]\{x\}$ , and  $m$  is defined at all of these points. (See Figure 32).

The displacements  $u = [N]\{u\}$  (the shape functions are the same as those for the spatial interpolation if an isoparametric mapping is chosen) are required such that:

$$f(x + u(x, y), y + v(x, y)) = m(x, y) \quad (17)$$

Taking a first order Taylor series expansion as before:

$$f(x + u(x, y), y + v(x, y)) = f(x, y) + u(x, y) \frac{\partial f(x, y)}{\partial x} + v(x, y) \frac{\partial f(x, y)}{\partial y} \quad (18)$$

Similarly, the residual is:

$$R(x + u, y + v) = f(x, y) + u(x, y) \frac{\partial f(x, y)}{\partial x} + v(x, y) \frac{\partial f(x, y)}{\partial y} - m(x, y) \quad (19)$$

Hence the operation in parametric space is just the same as that in Cartesian space, except that now  $(x, y)$  are the co-ordinates of the points in the image defined in parametric space, so  $f$  and its derivatives will have to be re-sampled to each of these points.

It must be noted that the information is extracted from the images by defining the residual at all voxel centres; however the displacements do not have to be independently defined at each voxel. The voxel displacements can be described in terms of the values at nodes in the parametric image, interpolated to the voxels using the shape functions.

In the implementation of the Sheffield Image Registration Toolkit (ShIRT) the mapping is described at a series of nodes that are regularly distributed in the Cartesian reference frame of the image (a uniform grid), and the mapping at all other points is defined by a linear interpolation between them. The mapping is thus piecewise continuous but the gradients are discontinuous across the lines between the nodes. In 2D the interpolation functions are bilinear. Figure 33

illustrates an image of 48 x 48 pixels with a 5 x 5 array of nodes across the domain. The registration vector,  $\{u_i\} = \{u_i, v_i\}$ , at each node is defined by two degrees of freedom,  $u_i$  and  $v_i$ . There are thus  $2n^2$  degrees of freedom in the registration vector, where  $n$  is the number of nodes in each direction. More generally, if there are  $m$  nodes in the  $x$  direction and  $n$  in the  $y$  direction, the number of unknowns is  $2mn$ . These can be written in a single vector  $\{u\}$ , of length  $2mn$ .

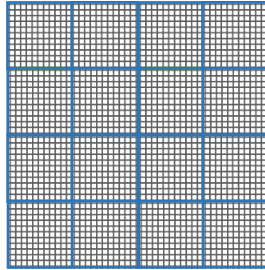


Figure 33: Image of 48 x 48 pixels with a 4x4 grid (5x5 nodes) superimposed

The images and gradients are changing at all points, and therefore it makes sense to evaluate the residual at all points in the original image to maintain all information from the image.

In Figure 33 the focus is on the image pixels and the relationship between the pixel grid and the registration grid. Another point of view is that of the registration grid and the elements that it is made of. Figure 34 highlights this in a new grid, with regular elements in Cartesian space.

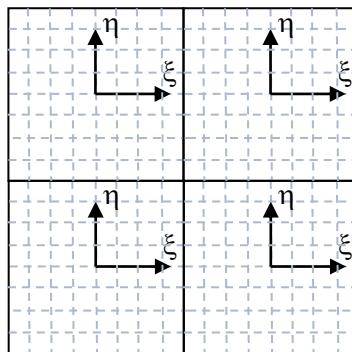


Figure 34: A four element grid with 8x8 pixels in each element

Now within each element,  $u$  is defined in terms of  $\{u\}$  at the nodes. The more general situation, with a parametric grid defined in curvilinear co-ordinates, is illustrated in Figure 35.

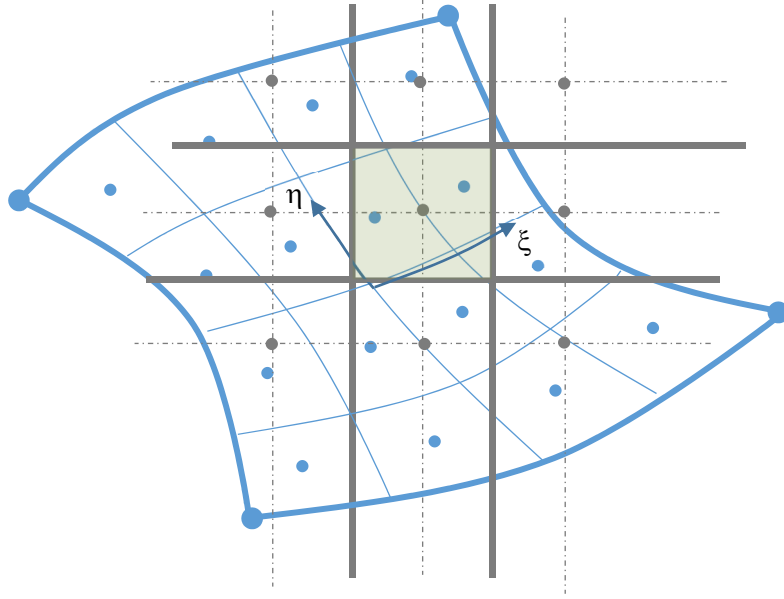


Figure 35: An element of the parametric moving image (blue) overlaid on the fixed image (green). The centres of the voxels are indicated by the coloured dots

The residual is defined at the centres of the voxels in the moving image and is evaluated using essentially (10), repeated below, but now with the images and their gradients evaluated at the centres of the voxels of the moving image.

$$\bar{R} = f - m + \frac{u}{2} \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) + \frac{v}{2} \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \quad (20)$$

There are two necessary processes in the evaluation of this residual at the appropriate points. The first is the interpolation of  $f(x, y)$ , and its gradients, from its known values at the centres of the fixed grid in Cartesian space to the centre of the voxel of the moving image, and the second is the transformation of the gradients of  $m(x, y)$  from their natural definitions in parametric space to their Cartesian equivalents. The interpolation operation can be performed in (at least) three ways: i) the values at the centre of the voxel in Cartesian space in which the centre of the moving voxel currently lies can be taken directly (nearest neighbour), ii) an interpolation can be performed using an assumed form (e.g. trilinear) of the variation of  $f(x, y)$  in Cartesian space, iii) an interpolation can be performed based on a Taylor series expansion about the centre of the voxel of the fixed image. Each of these has advantages and disadvantages, the method in this thesis uses trilinear interpolation. The nearest neighbour approach is the simplest to implement but might be unstable if an image point lies close to a voxel boundary and flips either side of it during iterations of the solution. The interpolation is better in principle but is more complex to

implement and, in a standard trilinear implementation, suffers from discontinuity of the image gradient and therefore the potential for instability in regions close to discontinuities. The Taylor series has the advantage that it is consistent with the residual derivation, but of course might require higher order expansions if the moving image voxel centre is far from that of the fixed image. The transformation operation is performed by pre-multiplication of the vector of gradients of the moving image with respect to the parametric co-ordinates by the Jacobian relating parametric space to Cartesian space.

### 3.3.2.1 Gradients

The fixed image is described on a regular Cartesian grid and hence the intensity gradient can be easily calculated using the second order approximation

$$\frac{\partial f}{\partial x_i} = \frac{f(x_{i+1}, y) - f(x_{i-1}, y)}{x_{i+1} - x_{i-1}} \quad (21)$$

The moving image  $m(x, y)$  is described as an un-structured mesh in parametric space. The parametric gradient needs to be transformed into Cartesian space. The intensity gradient in Cartesian space  $\frac{\partial m}{\partial x}$  is related to the gradient  $\frac{\partial m}{\partial \xi}$  in parametric space through the Jacobian matrix of the element.

### 3.3.2.2 Iso-parametric bilinear formulation

For linear, quadrilateral elements, a bilinear formulation is commonly used. This is the combination of linear interpolation performed in each of the coordinate directions; however the overall interpolation is a non-linear.

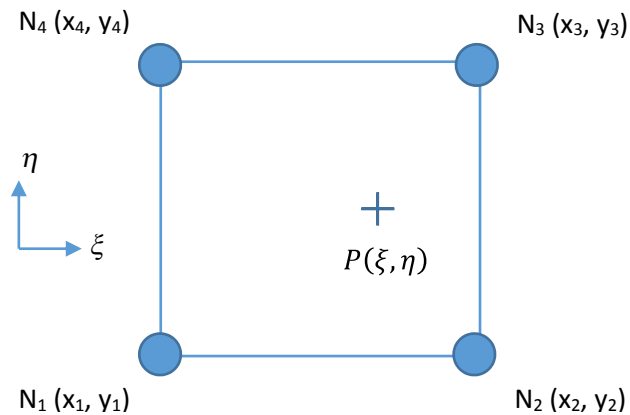


Figure 36: A quadrilateral element defined by nodes N<sub>1</sub> to N<sub>4</sub> and sampling point, P

Point P has local coordinates  $(\xi, \eta)$  and global coordinates  $(x, y)$ . The relationship between the two coordinate pair is dependent on the interpolation formulation used, that is, the contribution

of each node towards the value of P. Using the bilinear formulation, the contribution drops linearly as the distance of P from a node increases. The overall interpolation is a degree 2 polynomial:

$$x = [N_1 \quad N_2 \quad N_3 \quad N_4] \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} \quad (22)$$

Where:

$$\begin{aligned} N_1 &= \frac{1}{4}(1 - \xi)(1 - \eta) \\ N_2 &= \frac{1}{4}(1 + \xi)(1 - \eta) \\ N_3 &= \frac{1}{4}(1 + \xi)(1 + \eta) \\ N_4 &= \frac{1}{4}(1 - \xi)(1 + \eta) \end{aligned} \quad (23)$$

The same idea can be applied to work out the contribution of the residual at each sampling point towards the value of the residual at each of the nodes. Again, using an iso-parametric bilinear formulation:

$$\{u\} = \begin{Bmatrix} u \\ v \end{Bmatrix} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 \end{bmatrix} \begin{Bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{Bmatrix} = [N]\{u\} \quad (24)$$

At any point in the image the residual is:

$$R = f - m + \{NG\}_u^T \{u\} + \{NG\}_v^T \{u\} \quad (25)$$

Where:

$$\begin{aligned} \{NG\}_u^T &= \frac{1}{2} \left[ N_1 \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) \quad 0 \quad N_2 \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) \quad 0 \quad N_3 \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) \quad 0 \quad N_4 \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) \quad 0 \right] \\ \{NG\}_v^T &= \frac{1}{2} \left[ 0 \quad N_1 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \quad 0 \quad N_2 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \quad 0 \quad N_3 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \quad 0 \quad N_4 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \right] \end{aligned} \quad (26)$$

And the vector  $\{u\}$  contains the (unknown) displacements at the nodes of the registration element within which the image point lies.

Combining the parts of the  $\{NG\}$  vector:

$$R = f - m + \{NG\}^T \{u\} \quad (27)$$

Where

$$\{NG\}^T = \frac{1}{2} \begin{bmatrix} N_1 \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) & N_1 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) & N_2 \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) & N_2 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \\ N_3 \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) & N_3 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) & N_4 \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) & N_4 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \end{bmatrix} \quad (28)$$

The residual at each voxel is now expressed as a contribution towards the overall displacement of the registration grid. To transform the two dimensional parametric gradient into Cartesian space the Jacobian matrix is used.

$$\begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{pmatrix} \frac{\partial m}{\partial x} \\ \frac{\partial m}{\partial y} \end{pmatrix} = \begin{pmatrix} \frac{\partial m}{\partial \xi} \\ \frac{\partial m}{\partial \eta} \end{pmatrix} \quad (29)$$

Here the inverse of the Jacobian is a function of the Jacobian derivatives:

$$J = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \rightarrow J^{-1} = \frac{1}{\frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi} \frac{\partial x}{\partial \eta}} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} \\ -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} \end{bmatrix} \quad (30)$$

$$\begin{pmatrix} \frac{\partial m}{\partial x} \\ \frac{\partial m}{\partial y} \end{pmatrix} = J^{-1} \begin{pmatrix} \frac{\partial m}{\partial \xi} \\ \frac{\partial m}{\partial \eta} \end{pmatrix} \quad (31)$$

The Jacobian derivatives need to be calculated, recall:

$$x = N_1x_1 + N_2x_2 + N_3x_3 + N_4x_4$$

$$\frac{\partial x}{\partial \xi} = \left(\frac{\partial N_1}{\partial \xi}\right)x_1 + \left(\frac{\partial N_2}{\partial \xi}\right)x_2 + \left(\frac{\partial N_3}{\partial \xi}\right)x_3 + \left(\frac{\partial N_4}{\partial \xi}\right)x_4$$

$$\frac{\partial x}{\partial \xi} = (\eta - 1)x_1 + (1 - \eta)x_2 + (1 + \eta)x_3 + (-1 - \eta)x_4$$

(32)

$$\frac{\partial x}{\partial \eta} = \left(\frac{\partial N_1}{\partial \eta}\right)x_1 + \left(\frac{\partial N_2}{\partial \eta}\right)x_2 + \left(\frac{\partial N_3}{\partial \eta}\right)x_3 + \left(\frac{\partial N_4}{\partial \eta}\right)x_4$$

$$\frac{\partial x}{\partial \eta} = (\xi - 1)x_1 + (-1 - \xi)x_2 + (1 + \xi)x_3 + (1 - \xi)x_4$$

The other derivatives are calculated in a similar fashion. The derivatives of the shape functions are shown in Table 5.

$i$	$N_i$	$\frac{\partial N_i}{\partial \xi}$	$\frac{\partial N_i}{\partial \eta}$
1	$\frac{1}{4}(1 - \xi)(1 - \eta)$	$\frac{1}{4}(\eta - 1)$	$\frac{1}{4}(\xi - 1)$
2	$\frac{1}{4}(1 + \xi)(1 - \eta)$	$\frac{1}{4}(1 - \eta)$	$\frac{1}{4}(-1 - \xi)$
3	$\frac{1}{4}(1 + \xi)(1 + \eta)$	$\frac{1}{4}(\eta + 1)$	$\frac{1}{4}(\xi + 1)$
4	$\frac{1}{4}(1 - \xi)(1 + \eta)$	$\frac{1}{4}(-\eta - 1)$	$\frac{1}{4}(1 - \xi)$

Table 5: The derivatives of the iso-parametric bilinear shape functions

### 3.3.2.3 Iso-parametric trilinear formulation

Three dimensional images are very common in medical imaging and indeed computer simulations are performed in 3 dimensions. The two dimensional theory introduced above is readily extended to three dimensions. Taking the element numbered as illustrated below:



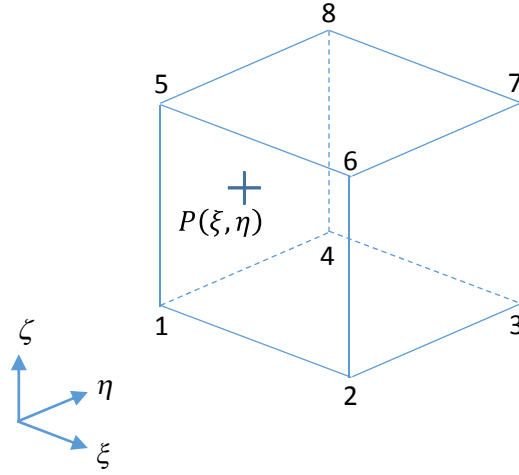


Figure 37: A hexahedral element defined by nodes  $N_1$  to  $N_8$  and sampling point,  $P$

The second degree polynomial is now defined by a 3-by-24 matrix and the number of unknowns collected in the 24-by-1 vector,  $\{u\}$ .

$$\{u\} = \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} = \begin{bmatrix} N_1 & 0 & 0 & \dots & \dots & N_8 & 0 & 0 \\ 0 & N_1 & 0 & \dots & \dots & 0 & N_8 & 0 \\ 0 & 0 & N_1 & \dots & \dots & 0 & 0 & N_8 \end{bmatrix} \begin{Bmatrix} u_1 \\ v_1 \\ w_1 \\ \vdots \\ \vdots \\ u_8 \\ v_8 \\ w_8 \end{Bmatrix} = [N]\{u\} \quad (33)$$

The complete trilinear shape functions are as follows:

$$\begin{aligned} N_1 &= \frac{1}{8}(1-\xi)(1-\eta)(1-\zeta) \\ N_2 &= \frac{1}{8}(1+\xi)(1-\eta)(1-\zeta) \\ N_3 &= \frac{1}{8}(1+\xi)(1+\eta)(1-\zeta) \\ N_4 &= \frac{1}{8}(1-\xi)(1+\eta)(1-\zeta) \\ N_5 &= \frac{1}{8}(1-\xi)(1-\eta)(1+\zeta) \\ N_6 &= \frac{1}{8}(1+\xi)(1-\eta)(1+\zeta) \\ N_7 &= \frac{1}{8}(1+\xi)(1+\eta)(1+\zeta) \\ N_8 &= \frac{1}{8}(1-\xi)(1+\eta)(1+\zeta) \end{aligned} \quad (34)$$

For any point in the image, the residual is defined:

$$R = f - m + \{NG\}_u^T \{u\} + \{NG\}_v^T \{u\} + \{NG\}_w^T \{u\} \quad (35)$$

Where now the  $\{NG\}$  vectors are:

$$\begin{aligned}
& \{NG\}_u^T \\
&= \frac{1}{2} \left[ N_1 \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) \quad 0 \quad 0 \quad N_2 \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) \quad 0 \quad 0 \quad \dots \quad \dots \quad N_8 \left( \frac{\partial f}{\partial x} + \frac{\partial m}{\partial x} \right) \quad 0 \quad 0 \right] \\
& \{NG\}_v^T \\
&= \frac{1}{2} \left[ 0 \quad N_1 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \quad 0 \quad 0 \quad N_2 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \quad 0 \quad \dots \quad \dots \quad 0 \quad N_8 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \quad 0 \right] \quad (36) \\
& \{NG\}_w^T \\
&= \frac{1}{2} \left[ 0 \quad 0 \quad N_1 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \quad 0 \quad 0 \quad N_2 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \quad \dots \quad \dots \quad 0 \quad 0 \quad N_8 \left( \frac{\partial f}{\partial y} + \frac{\partial m}{\partial y} \right) \right]
\end{aligned}$$

And the  $\{NG\}$  vectors combined into the  $\{NG_{3D}\}$  matrix:

$$R = f - m + \{NG_{3D}\}^T \{u\} \quad (37)$$

Once again, to transform the three dimensional parametric gradients into Cartesian space the Jacobian matrix is used,

$$\begin{bmatrix} \partial x / \partial \xi & \partial y / \partial \xi & \partial z / \partial \xi \\ \partial x / \partial \eta & \partial y / \partial \eta & \partial z / \partial \eta \\ \partial x / \partial \varphi & \partial y / \partial \varphi & \partial z / \partial \varphi \end{bmatrix} \begin{pmatrix} \partial m / \partial x \\ \partial m / \partial y \\ \partial m / \partial z \end{pmatrix} = \begin{pmatrix} \partial m / \partial \xi \\ \partial m / \partial \eta \\ \partial m / \partial \varphi \end{pmatrix} \quad (38)$$

And the inverse Jacobian remains a function of the Jacobian derivatives

$$J = \begin{bmatrix} \partial x / \partial \xi & \partial y / \partial \xi & \partial z / \partial \xi \\ \partial x / \partial \eta & \partial y / \partial \eta & \partial z / \partial \eta \\ \partial x / \partial \varphi & \partial y / \partial \varphi & \partial z / \partial \varphi \end{bmatrix} \rightarrow \begin{pmatrix} \partial m / \partial x \\ \partial m / \partial y \\ \partial m / \partial z \end{pmatrix} = J^{-1} \begin{pmatrix} \partial m / \partial \xi \\ \partial m / \partial \eta \\ \partial m / \partial \varphi \end{pmatrix} \quad (39)$$

To calculate the Jacobian derivatives, recall

$$x = N_1x_1 + N_2x_2 + N_3x_3 + N_4x_4 + N_5x_5 + N_6x_6 + N_7x_7 + N_8x_8$$

$$\frac{\partial x}{\partial \xi} = \left(\frac{\partial N_1}{\partial \xi}\right)x_1 + \left(\frac{\partial N_2}{\partial \xi}\right)x_2 + \left(\frac{\partial N_3}{\partial \xi}\right)x_3 + \left(\frac{\partial N_4}{\partial \xi}\right)x_4 + \left(\frac{\partial N_5}{\partial \xi}\right)x_5 + \left(\frac{\partial N_6}{\partial \xi}\right)x_6 + \left(\frac{\partial N_7}{\partial \xi}\right)x_7 + \left(\frac{\partial N_8}{\partial \xi}\right)x_8 \quad (40)$$

$$\begin{aligned} \frac{\partial x}{\partial \xi} = & (-1 + \varphi + \eta - \varphi\eta)x_1 + (1 - \varphi - \eta + \varphi\eta)x_2 + (1 - \varphi + \eta - \varphi\eta)x_3 \\ & + (-1 + \varphi - \eta + \varphi\eta)x_4 + (-1 - \varphi + \eta + \varphi\eta)x_5 + (1 + \varphi - \eta - \varphi\eta)x_6 \\ & + (1 + \varphi + \eta + \varphi\eta)x_7 + (-1 - \varphi - \eta - \varphi\eta)x_8 \end{aligned}$$

The other derivatives are summarised in Table 6 and are calculated similarly.

$i$	$N_i$			$\frac{\partial N_i}{\partial \xi}$	$\frac{\partial N_i}{\partial \eta}$	$\frac{\partial N_i}{\partial \varphi}$
1	$1/8(1-\xi)$	$(1-\eta)$	$(1-\varphi)$	$1/8(-1+\varphi+\eta-\varphi\eta)$	$1/8(-1+\varphi+\xi-\varphi\xi)$	$1/8(-1+\eta+\xi-\eta\xi)$
2	$1/8(1+\xi)$	$(1-\eta)$	$(1-\varphi)$	$1/8(1-\varphi-\eta+\varphi\eta)$	$1/8(-1+\varphi-\xi+\varphi\xi)$	$1/8(-1+\eta-\xi+\eta\xi)$
3	$1/8(1+\xi)$	$(1+\eta)$	$(1-\varphi)$	$1/8(1-\varphi+\eta-\varphi\eta)$	$1/8(1-\varphi+\xi-\varphi\xi)$	$1/8(-1-\eta-\xi-\eta\xi)$
4	$1/8(1-\xi)$	$(1+\eta)$	$(1-\varphi)$	$1/8(-1+\varphi-\eta+\varphi\eta)$	$1/8(1-\varphi-\xi+\varphi\xi)$	$1/8(-1-\eta+\xi+\eta\xi)$
5	$1/8(1-\xi)$	$(1-\eta)$	$(1+\varphi)$	$1/8(-1-\varphi+\eta+\varphi\eta)$	$1/8(-1-\varphi+\xi+\varphi\xi)$	$1/8(1-\eta-\xi+\eta\xi)$
6	$1/8(1+\xi)$	$(1-\eta)$	$(1+\varphi)$	$1/8(1+\varphi-\eta-\varphi\eta)$	$1/8(-1-\varphi-\xi-\varphi\xi)$	$1/8(1-\eta+\xi-\eta\xi)$
7	$1/8(1+\xi)$	$(1+\eta)$	$(1+\varphi)$	$1/8(1+\varphi+\eta+\varphi\eta)$	$1/8(1+\varphi+\xi+\varphi\xi)$	$1/8(1+\eta+\xi+\eta\xi)$
8	$1/8(1-\xi)$	$(1+\eta)$	$(1+\varphi)$	$1/8(-1-\varphi-\eta-\varphi\eta)$	$1/8(1+\varphi-\xi-\varphi\xi)$	$1/8(1+\eta-\xi-\eta\xi)$

Table 6: The trilinear formulation shape function derivatives

### 3.3.3 Linear least squares formulation

The above derivations are expressed in terms of the errors at each voxel and the movement at the appropriate nodes.

$$R = f - m + \{NG\}^T\{u\} \quad (41)$$

The equations are more conveniently presented in matrix form, the aggregated error over the whole image domain and the movement of all the nodes in the mesh.

$$\{R\} = \{f - m\} + [NG]\{u\} \quad (42)$$

$\{R\}$  is a vector with the error at each mesh voxel after the movement; it has an entry for each image voxel sampled.

$\{f-m\}$  is a vector with the difference in intensity between each image location; it has an entry for each image voxel sampled.

$\{u\}$  is the nodal displacement vector and contains all degrees of freedom; it has an entry for each degree of freedom at each node in the mesh.

$[NG]$  is a matrix which relates the contribution to each node of the image intensity derivatives; it has a row for each image voxel sampled and a column for each degree of freedom at every node in the mesh.

The  $[NG]$  matrix has as many non-zero terms in every row as degrees of freedom associated with the voxel represented by the row (the nodes the voxel contributes towards), and as many non-zero terms in every column as voxels associated with the relevant node in the registration grid. For any real problem  $[NG]$  will be a sparse matrix.

There are several issues associated with the fact that each node requires a linearly independent equation for each degree of freedom it possesses. Firstly, there is a minimum number of sample points required for the system to be solvable. More precisely, there is a minimum number of sample points describing each node for the system to be solvable. Moreover, from the equations described above, sample points where the intensity gradients at both the image and mesh are zero (highly common in binary images) provide no information for solving the system. One should like the residual to be zero at every voxel in the images, but this is in general not possible because there are not enough degrees of freedom. It is safe so say that the system will always be built from an excess of equations and therefore one seeks a solution that minimises some chosen norm of the error vector.

In standard form:

$$\{R\} = [A]\{u\} + \{b\} \quad (43)$$

Where  $\{b\} = \{f - m\}$  and  $[A] = [NG]$ . Each entry of  $[A]$ ,  $a_{ij}$ , is the term associated with the contribution of the  $i^{\text{th}}$  pixel to the  $j^{\text{th}}$  degree of freedom.

One possible norm of this vector is the sum of the squares of the residual over the whole image domain:

$$\|R\|_2 = \sum_{i=1}^I (b_i + \{a\}_i^T \{u\})^2 \quad (44)$$

Where  $\{a\}_i$  is the  $i^{\text{th}}$  row of the  $[A]$  matrix. Expanding the vectors:

$$\|R\|_2 = \sum_{i=1}^I \left( b_i + \sum_{n=1}^N a_{in} u_n \right)^2 \quad (45)$$

Where  $a_{in}$  is the component of the vector  $\{a\}$  at pixel  $i$  that is associated with the degree of freedom  $n$ . Multiplying out:

$$\|R\|_2 = \sum_{i=1}^I \left( b_i^2 + 2b_i \sum_{n=1}^N a_{in} u_n + \left( \sum_{n=1}^N a_{in} u_n \right)^2 \right) \quad (46)$$

We can take the derivative of this vector norm with respect to each of the degrees of freedom:

$$\frac{\partial \|R\|_2}{\partial u_n} = \sum_{i=1}^I \left( 2b_i a_{in} + 2a_{in} \sum_{m=1}^N a_{im} u_m \right) \quad (47)$$

And the norm is minimised when each of these derivatives is zero, so the  $n$ th equation is:

$$0 = \sum_{i=1}^I b_i a_{in} + \sum_{i=1}^I \left( a_{in} \sum_{m=1}^N a_{im} u_m \right) \quad (48)$$

Alternatively, reversing the order of summation in the second term yields:

$$0 = \sum_{i=1}^I b_i a_{in} + \sum_{m=1}^N \left( \sum_{i=1}^I a_{in} a_{im} \right) u_m \quad (49)$$

There are  $N$  simultaneous equations in the  $N$  unknowns. In expanded form:

$$\begin{bmatrix} \sum_{i=1}^I a_{i1} a_{i1} & \sum_{i=1}^I a_{i1} a_{i2} & \sum_{i=1}^I a_{i1} a_{im} & \sum_{i=1}^I a_{i1} a_{iN} \\ \sum_{i=1}^I a_{i2} a_{i1} & \sum_{i=1}^I a_{i2} a_{i2} & \dots & \sum_{i=1}^I a_{i2} a_{iN} \\ \dots & \dots & \dots & \dots \\ \sum_{i=1}^I a_{im} a_{i1} & \sum_{i=1}^I a_{im} a_{i2} & \dots & \sum_{i=1}^I a_{im} a_{iN} \\ \dots & \dots & \dots & \dots \\ \sum_{i=1}^I a_{iN} a_{i1} & \sum_{i=1}^I a_{iN} a_{i2} & \sum_{i=1}^I a_{iN} a_{im} & \sum_{i=1}^I a_{iN} a_{iN} \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ \dots \\ u_m \\ \dots \\ u_N \end{Bmatrix} = \begin{Bmatrix} -\sum_{i=1}^I b_i a_{i1} \\ -\sum_{i=1}^I b_i a_{i2} \\ \dots \\ -\sum_{i=1}^I b_i a_{im} \\ \dots \\ -\sum_{i=1}^I b_i a_{iN} \end{Bmatrix} \quad (50)$$

This equation can be conveniently written in terms of the original matrices and vectors from equation (43).

$$[A]^T[A]\{u\} = -[A]^T\{b\} \quad (51)$$

### 3.4 Discussion

This chapter has described the components of image registration and described in depth the derivation of the registration process, based on optical flow, used in this thesis.

In particular, Section 3.3.1.1 describes the rationale behind pushing the moving image onto the fixed image. As explained in section 3.3.1.2, describing the process as pulling the moving image into the fixed image removes the need to resample the residual. This idea is key for implementing the method using unstructured grids as the moving image, as shown in section 3.3.2.

The key feature of the image registration method described in this chapter is the parametrisation of the moving image. This idea works in the inverse direction, allowing unstructured grids to be treated as images. Equation (16) describes the residual of registration metric after a registration iteration in terms of the voxel intensity values and the derivatives at the sampling locations, in both images. In theory, the registration can be described in terms of the intensity derivatives at either of the images but introducing information from both images makes the process more robust. The risk of averaging the derivatives is that it allows them to cancel each other however in practice and for an over-determined system of equations, as discussed in sections 3.3.2 and 3.3.3, this problem is not met.

Sparse images, such as segmentations, are commonly encountered in the field of medical imaging. Such images are distinct in that they contain very few intensity gradients. When registration methods based in optical flow are employed, they yield ill-conditioned systems of equations because the images do not contain enough “information” for robust registration. By employing gradients from both images, the method described in this chapter overcomes this problem. In practice, binary images are blurred prior to registration to increase the number of gradients in the image.

Despite the aforementioned characteristic of the registration method, the system of equations remain ill-conditioned. In the following chapter the regularisation of the registration will be explored in depth, and the benefits of parametrising the moving image will be discussed.

# CHAPTER 4

## REGULARISATIONS TO THE REGISTRATION

The image registration algorithm presented in the previous chapter, which is based on the computation of optical flow, faces the problem that it is under constrained. Typical motion estimation algorithms use regularisations to control the structure of the motion [98] taking advantage of the circumstance that a combination of two, ill-posed systems of equations may create a solvable system. This chapter describes the effect of different regularisations on the registration process.

Section 4.1 describes the ill-conditioning of image registration methods, including the method presented in chapter 3 and section 4.2 presents an approach to regularise the system of equations. Section 4.3 and 4.4 discuss a number of regularisation options, which are then tested in section 4.5. Section 4.6 discusses the importance of this chapter and contextualises the work presented in chapters 3 and 4 with other image registration methods found in the literature.

The first regularisation introduced is the Laplacian operator which is commonly used in image registration methods to produce smooth results. Two alternative regularisations are presented, one is based on the scalar product of the element edges and the other on the Jacobian distribution within the element. These alternative regularisations are derived from metrics that measure mesh quality better than the smoothness measure and should theoretically produce higher quality meshes. The assumption is that by introducing these metrics into the registration process, the quality of the mesh can be optimised along-side the image similarity metric (and the smoothing metric).

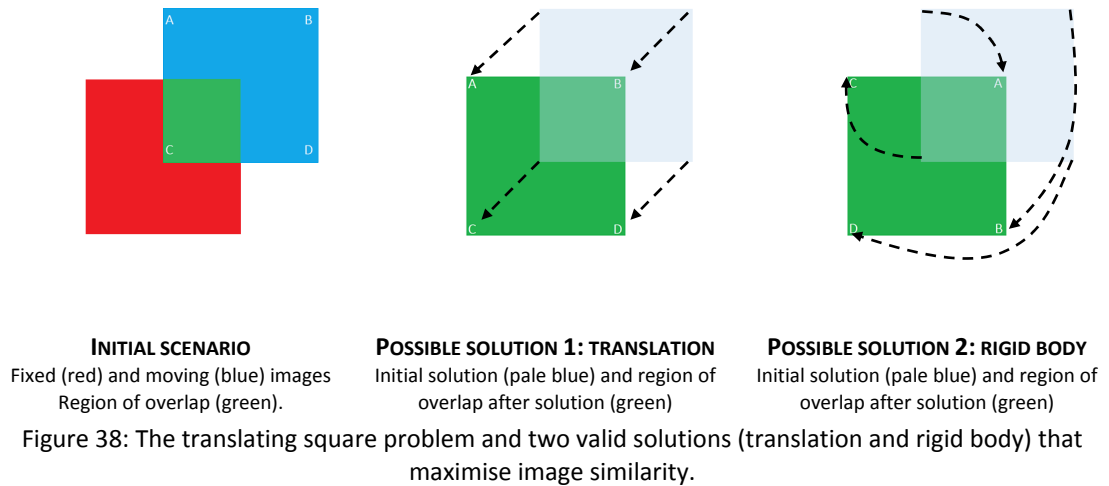
### 4.1 Ill-conditioning

In the last chapter the fundamental equations to represent the residual at all points in the registration field were developed, together with the linear least squares formulation of the optimisation problem. In practice this problem is almost always ill-conditioned.

Ill-conditioning can be described in terms of the image process. For instance, if registration is the process of matching the objects that appear in two images, the solution is uncertain when certain objects only appear on one of the images.

Another way of looking at it is the translating square problem (shown in Figure 38): both images in question contain a square of same size but in different locations [97]. One solution is to

translate the square on one image so that the bottom right corners match. Another perfectly valid solution is rotating the square so that the bottom corner matches the top left corner. The resulting images will match but the underlying deformations are significantly different. In this example however, both solutions can be achieved using rigid registration models.



Yet another way of looking at image registration is in terms of increasing image similarity. However, one could define a map that took all points of the same intensity into a point of the same intensity in the second image. In the translating square problem described above, take all the points in one square into the same location in the other square. The similarity metric would be maximised but the real application of the registration problem would not be solved [99].

Mathematically speaking, the optimisation of the parameters of the transformation matrix is an ill-posed process. For the process to be well posed,

- 1) the transformation must exist,
- 2) the transformation must be unique and
- 3) the transformation depends continuously on the images.

Regularisation is therefore used to improve the stability of the registration process and to introduce additional information about the process.

Generally speaking, not all pixels in the moving image can be mapped from one image to the other. The reason for this, in the case of anatomical images for example, is due to the contents of the images not being the same, due to similar objects appearing with different intensities and because of the images being noisy. In practice, this does not majorly affect the process of registration and good matches can be found using image registration.



The non-uniqueness of the system is a bigger problem [100]. Generally, image registration is massively under-constrained and the solution that best fits the application is prioritised. In other words, there are many solutions to the image registration problem but the preferred solution is the source of mis-alignment. For example, in the case of phase-contrast images and segmentations (binary images), the former contain mostly noise whilst the later contain mostly "empty" voxels. They contain very little "information" to drive the registration.

## 4.2 Tikhonov regularisation of the linear least squares formulation

Equation (51) at the end of Chapter 3 is generally ill-conditioned because  $[A]^T[A]$  is singular, or close to it. For instance, in the medical field one is likely to work with segmentations, and these sparse images are more likely to produce singular systems. The reason for this is that most intensity gradients in the image are zero. Therefore a regularisation term is added. Tikhonov regularisation is a general method of improving the conditioning of the linear least squares problem. An additional term is added to the cost function so that:

$$\|R\|_2 = \|[A]\{u\} + \{b\}\|_2 + \|\Gamma\{u\}\|_2 \quad (52)$$

where  $\Gamma$  is a Tikhonov matrix which is chosen to impose particular characteristics on the solution.

The system that minimises this cost function now becomes:

$$\left[ [A]^T[A] + \lambda[\Gamma]^T[\Gamma] \right] \{u\} = -[A]^T\{b\} \quad (53)$$

The weighting factor,  $\lambda$ , allows changing the effect of the regularisation term. Setting this value to 0 will result in an un-regularised solution, provided  $([A]^T[A])^{-1}$  exists. Increasing the value of  $\lambda$  will result in a solution that favours the regularisation term rather than the condition of image similarity. In practice, this value is determined by experimentation and can be used between sets of similar images. ShIRT and other toolkits [101] select the value of  $\lambda$  that most improves the conditioning of the system.

## 4.3 Options for Tikhonov matrices

### 4.3.1 Identity matrix

A common choice for the Tikhonov matrix is the identity matrix [101]. This solution is trivial to implement however faces the problem that it does not add any useful information to the registration process; it simply gives preference to solutions with displacements of smallest magnitude. Negative and positive displacements cancel out, meaning the node displacement is aware of the global displacements but not the displacement of the local-neighbours; in practice it may lead to very uneven results. Under this regularisation, the overall Tikhonov system takes the following form:

$$\left[ [A]^T[A] + \lambda[I] \right] \{u\} = -[A]^T\{b\} \quad (54)$$

### 4.3.2 Laplacian matrix

Operators that penalise uneven solutions also favour the solutions of smallest magnitude, making them a far better choice for image registration methods. A common choice for imposing a smoothness constraint is penalising high curvatures in the displacement field. Choosing the Laplacian filter for the Tikhonov matrix produces the following:

$$\left[ [A]^T[A] + \lambda[\nabla^2]^T[\nabla^2] \right] \{u\} = -[A]^T\{b\} \quad (55)$$

The Laplacian describes the second derivative of the displacement field:

$$\nabla^2\{u\} = \sum_{i=1}^3 \frac{\partial^2\{u\}}{\partial x_i^2} \quad (56)$$

On a parametric grid with curvilinear co-ordinates this can be evaluated by re-casting in terms of the parametric co-ordinates. A simpler alternative is to use the graph Laplacian, which penalises high curvatures along the edges of the mesh (as opposed to the Cartesian directions):

$$\nabla^2\{u\} = \sum_{i=1}^3 \frac{\partial^2\{u\}}{\partial \eta_i^2} \quad (57)$$

The graph Laplacian is not equal to the second derivatives in Cartesian space, however given the complexity of the later matrix (when expressed in parametric terms) it allows for a more effective implementation. For reference, the Laplacian expressed in parametric terms is:

$$\begin{aligned}
\nabla^2\{u\} = \sum_{i=1}^3 \frac{\partial^2\{u\}}{\partial x_i^2} = & \left[ \frac{\partial^2\xi}{\partial x^2} \frac{\partial u}{\partial \xi} + \frac{\partial^2\eta}{\partial x^2} \frac{\partial u}{\partial \eta} + \frac{\partial^2\eta}{\partial x^2} \frac{\partial^2 u}{\partial \eta^2} + \frac{\partial^2\xi}{\partial x^2} \frac{\partial^2 u}{\partial \xi^2} + \left( \frac{\partial\eta}{\partial x} \frac{\partial\xi}{\partial x} + \frac{\partial\xi}{\partial x} \frac{\partial\eta}{\partial x} \right) \frac{\partial^2 u}{\partial \xi \partial \eta} \right] \\
& + \left[ \frac{\partial^2\xi}{\partial y^2} \frac{\partial u}{\partial \xi} + \frac{\partial^2\eta}{\partial y^2} \frac{\partial u}{\partial \eta} + \frac{\partial^2\eta}{\partial y^2} \frac{\partial^2 u}{\partial \eta^2} + \frac{\partial^2\xi}{\partial y^2} \frac{\partial^2 u}{\partial \xi^2} + \left( \frac{\partial\eta}{\partial y} \frac{\partial\xi}{\partial y} + \frac{\partial\xi}{\partial y} \frac{\partial\eta}{\partial y} \right) \frac{\partial^2 u}{\partial \xi \partial \eta} \right] \\
& + \left[ \frac{\partial^2\xi}{\partial z^2} \frac{\partial u}{\partial \xi} + \frac{\partial^2\eta}{\partial z^2} \frac{\partial u}{\partial \eta} + \frac{\partial^2\eta}{\partial z^2} \frac{\partial^2 u}{\partial \eta^2} + \frac{\partial^2\xi}{\partial z^2} \frac{\partial^2 u}{\partial \xi^2} + \left( \frac{\partial\eta}{\partial z} \frac{\partial\xi}{\partial z} + \frac{\partial\xi}{\partial z} \frac{\partial\eta}{\partial z} \right) \frac{\partial^2 u}{\partial \xi \partial \eta} \right] \\
& + \left[ \frac{\partial^2\xi}{\partial x^2} \frac{\partial v}{\partial \xi} + \frac{\partial^2\eta}{\partial x^2} \frac{\partial v}{\partial \eta} + \frac{\partial^2\eta}{\partial x^2} \frac{\partial^2 v}{\partial \eta^2} + \frac{\partial^2\xi}{\partial x^2} \frac{\partial^2 v}{\partial \xi^2} + \left( \frac{\partial\eta}{\partial x} \frac{\partial\xi}{\partial x} + \frac{\partial\xi}{\partial x} \frac{\partial\eta}{\partial x} \right) \frac{\partial^2 v}{\partial \xi \partial \eta} \right] \\
& + \left[ \frac{\partial^2\xi}{\partial y^2} \frac{\partial v}{\partial \xi} + \frac{\partial^2\eta}{\partial y^2} \frac{\partial v}{\partial \eta} + \frac{\partial^2\eta}{\partial y^2} \frac{\partial^2 v}{\partial \eta^2} + \frac{\partial^2\xi}{\partial y^2} \frac{\partial^2 v}{\partial \xi^2} + \left( \frac{\partial\eta}{\partial y} \frac{\partial\xi}{\partial y} + \frac{\partial\xi}{\partial y} \frac{\partial\eta}{\partial y} \right) \frac{\partial^2 v}{\partial \xi \partial \eta} \right] \\
& + \left[ \frac{\partial^2\xi}{\partial z^2} \frac{\partial v}{\partial \xi} + \frac{\partial^2\eta}{\partial z^2} \frac{\partial v}{\partial \eta} + \frac{\partial^2\eta}{\partial z^2} \frac{\partial^2 v}{\partial \eta^2} + \frac{\partial^2\xi}{\partial z^2} \frac{\partial^2 v}{\partial \xi^2} + \left( \frac{\partial\eta}{\partial z} \frac{\partial\xi}{\partial z} + \frac{\partial\xi}{\partial z} \frac{\partial\eta}{\partial z} \right) \frac{\partial^2 v}{\partial \xi \partial \eta} \right] \\
& + \left[ \frac{\partial^2\xi}{\partial x^2} \frac{\partial w}{\partial \xi} + \frac{\partial^2\eta}{\partial x^2} \frac{\partial w}{\partial \eta} + \frac{\partial^2\eta}{\partial x^2} \frac{\partial^2 w}{\partial \eta^2} + \frac{\partial^2\xi}{\partial x^2} \frac{\partial^2 w}{\partial \xi^2} + \left( \frac{\partial\eta}{\partial x} \frac{\partial\xi}{\partial x} + \frac{\partial\xi}{\partial x} \frac{\partial\eta}{\partial x} \right) \frac{\partial^2 w}{\partial \xi \partial \eta} \right] \\
& + \left[ \frac{\partial^2\xi}{\partial y^2} \frac{\partial w}{\partial \xi} + \frac{\partial^2\eta}{\partial y^2} \frac{\partial w}{\partial \eta} + \frac{\partial^2\eta}{\partial y^2} \frac{\partial^2 w}{\partial \eta^2} + \frac{\partial^2\xi}{\partial y^2} \frac{\partial^2 w}{\partial \xi^2} + \left( \frac{\partial\eta}{\partial y} \frac{\partial\xi}{\partial y} + \frac{\partial\xi}{\partial y} \frac{\partial\eta}{\partial y} \right) \frac{\partial^2 w}{\partial \xi \partial \eta} \right] \\
& + \left[ \frac{\partial^2\xi}{\partial z^2} \frac{\partial w}{\partial \xi} + \frac{\partial^2\eta}{\partial z^2} \frac{\partial w}{\partial \eta} + \frac{\partial^2\eta}{\partial z^2} \frac{\partial^2 w}{\partial \eta^2} + \frac{\partial^2\xi}{\partial z^2} \frac{\partial^2 w}{\partial \xi^2} + \left( \frac{\partial\eta}{\partial z} \frac{\partial\xi}{\partial z} + \frac{\partial\xi}{\partial z} \frac{\partial\eta}{\partial z} \right) \frac{\partial^2 w}{\partial \xi \partial \eta} \right]
\end{aligned} \tag{58}$$

Construction of the graph Laplacian requires the mesh adjacency matrix. The mesh adjacency matrix represents the connectivity of nodes. This can be extended to form a discrete representation of the second order spatial derivative.

At each node, the discrete representation becomes the combination of the second order central approximation of the second degree derivative of the individual parametric directions. This means a weighting of -1 is given to each adjacent node and the weighting to the central node is equal to the number of adjacent nodes. The number of adjacent nodes is dependent on element (shape) type and by the dimensionality of the representation.

For nodes that lie at the edge of the mesh, where the adjacency matrix is asymmetrical, a zero gradient condition is used instead. This means that edge nodes are moved by the same amount as their connecting nodes. In practical terms, the adjacent node is assigned a weighting of -2 and the central node is assigned a weighting of 2.

The interpretation of the Laplacian on a structured grid is clear, and the graph Laplacian is just a simpler implementation of a form of Laplacian. In the current application the mesh that describes the geometry of the heart is unstructured. To represent the left ventricle, a number of pentahedral elements are placed around the apical region. Pentahedral elements are

degenerate hexahedra in which two parallel edges are each reduced to a point. The apical node is hence surrounded by multiple nodes that do not conform to the three parametric directions. In this case, the apical node is treated like an edge termination.

Other kernel shapes for the discrete Laplace operator were explored. These variations included second-order connectivity (diagonals) and higher-order discrete approximations of the second-order spatial derivative. In early testing they were found not to offer any advantage for the current application over the simpler form described above. The computational time required to build this regularisation using a naïve implementation is negligible, even for big meshes.

## 4.4 Options for Tikhonov matrices based on mesh quality metrics

### 4.4.1 Mesh quality metrics

#### 4.4.1.1 Orthogonality-based metrics of mesh quality

A number of geometric metrics are generally accepted by the modelling community to assess the fitness of meshes for mechanic and flow simulations [102]. Examples of such geometric metrics include the element aspect ratio and skew angle, these are shown in Figure 39. The aspect ratio of a 2D element is defined as the ratio of its shortest edge length to its longest edge length. This can be extended to hexahedral elements: a hexahedron of equal-length edges (a cube) has an aspect ratio of 1; as it deviates from the cubic shape, the aspect ratio diverges away from unity. The skew angle represents the absolute value of the difference between a reference angle and 90 degrees. The reference angle is formed by the two lines which pass through the midpoints of the sides of the quadrilateral. A rectangle has the skew angle of 0 since the formed angle between lines is 90 degrees. [38] Figure 39 shows how the skew angle is found for a quadrilateral element. For hexahedral elements, one calculates the skew angle for each of the element's faces and assigns it the value with the largest deviation from unity.

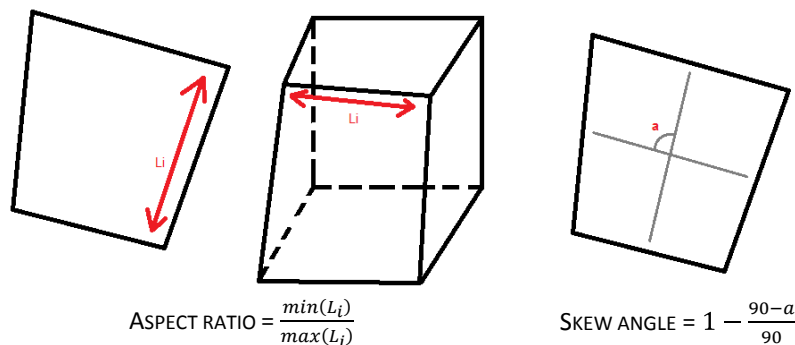


Figure 39: Geometric measures of element quality

#### 4.4.1.2 Jacobian-based metrics of grid deformation

The Jacobian defines the transformation from the local coordinate space to the global space, and its variation across the element represents a measure of how deformed the element is with respect to a unit cube. The Jacobian itself encapsulates shape and volume information of mesh elements. The determinant of the Jacobian can be computed at any point inside an element, and will vary according to the degree of local distortion.

$$[J] = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \varphi} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} & \frac{\partial y}{\partial \varphi} \\ \frac{\partial z}{\partial \xi} & \frac{\partial z}{\partial \eta} & \frac{\partial z}{\partial \varphi} \end{bmatrix} \quad (59)$$

This transformation is heavily used in computational methods, such as the finite element method. The element stiffness matrix is usually computed using a numerical integration based on a weighted sum of evaluations at specific locations (the Gauss points) using a Gaussian quadrature. The integration is performed at specific locations in parametric space, with the transformation between integrals in spatial and parametric co-ordinates defined by the determinant of the Jacobian. The order of integration is usually chosen based on the shape functions of the elements, and the integration becomes inaccurate if there are large variations in the determinant of the Jacobian across the element. Indeed, it has been shown that elements with high variation of the Jacobian across their volume are unstable [32].

If an element is folded, it will have a region of negative volume. In such locations, the determinant of the Jacobian will be negative. One approach that is used to prevent negative Jacobians is to check the determinant at all points in the registration grid a posteriori of the registration iteration and to redo the last registration step, usually with a smaller increment of displacement, if a negative value is found [96], [103]. This approach tends to improve, but does not guarantee, the quality of the elements.

Rohlfing et al. recognise the effect of constraining the local Jacobian of the deformation field [104]. In fact, they use both the Jacobian constraint and a smoothness constraint (Laplacian). They used the voxel locations to calculate this functional. In their paper, they describe a number of different variations to the constraint:

$E_{Jacobian} = \int_{V_r}  J_T(x) - 1  dx$	Penalises local expansion and compression. Minimises to zero.
$E_{Jacobian} = \int_{V_r}  \log(J_T(x))  dx$	Symmetrically weight local expansion / compression.
$E_{Jacobian} = \int_{V_r} \left  \frac{J_T(x) + 1}{J_T(x) - 2} \right  dx$	Assign no penalty for local volume preservation.
$E_{Jacobian} = \left  \int_{V_r} \log(J_T(x)) dx \right $	Penalises only global volume change

Figure 40: Jacobian-based energy terms

Loeckx et al. expand on Rohlfing et al. by changing the local rigidity of the registration field by applying a varying weighting factor. They impose the orthogonality condition  $J_T J_T^T = 1$  as this effectively constrains  $|J_T| = 1$  [105].

#### 4.4.1.3 Jacobian-based metric for element quality

The element Jacobian, or element Jacobian ratio, is more precisely defined as the ratio of maximum to minimum Jacobian determinant in an element. The Jacobian calculation is done at the integration points of elements commonly known as Gauss points. At each integration point, the Jacobian determinant is calculated, and the element Jacobian is found. If the quadrilateral element is not convex, a negative Jacobian ratio will be obtained.

#### 4.4.2 Scalar product as a metric of element orthogonality

With these geometric metrics in mind, a method was developed that improves both the aspect ratio and the skew angle, in particular, the orthogonality of all node connections are maximised. For a given node, there must be at least one node junction:

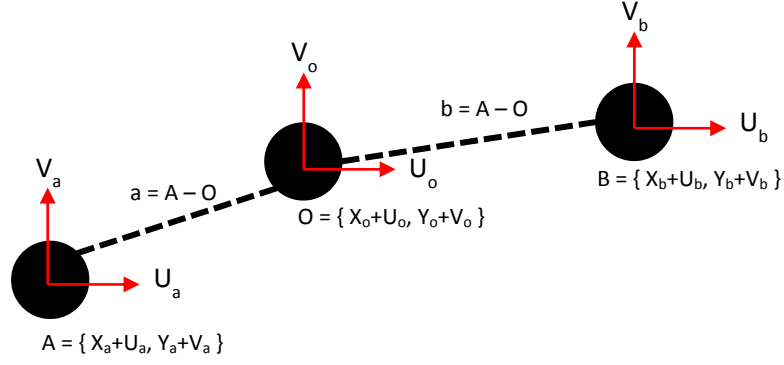


Figure 41: The degrees of freedom of the nodes that form a junction

Taking the dot product of vectors  $\vec{a}$  and  $\vec{b}$ :

$$\vec{a} \cdot \vec{b} = (x_b + u_b - x_o - u_o)(x_a + u_a - x_o - u_o) + (y_b + v_b - y_o - v_o)(y_a + v_a - y_o - v_o) \quad (60)$$

Expanding terms:

$$\begin{aligned} \vec{a} \cdot \vec{b} = & x_b x_a + x_b u_a - x_b x_o - x_b u_o + u_b x_a + u_b u_a - u_b x_o - u_b u_o \\ & - x_o x_a - x_o u_a + x_o^2 + x_o u_o - u_o x_a - u_o u_a + u_o x_o \\ & + u_o^2 + y_b y_a + y_b v_a - y_b y_o - y_b v_o + v_b y_a + v_b v_a \\ & - v_b y_o - v_b v_o - y_o y_a - y_o v_a + y_o^2 + y_o v_o - v_o y_a \\ & - v_o v_a + v_o y_o + v_o^2 \end{aligned} \quad (61)$$

Collecting terms:

$$\vec{a} \cdot \vec{b} = \begin{pmatrix} x_b - x_o \\ -x_o \\ x_o \\ y_b - y_o \\ -y_o \\ y_o \end{pmatrix}^T \begin{pmatrix} x_a \\ x_b \\ x_o \\ y_a \\ y_b \\ y_o \end{pmatrix} + \begin{pmatrix} x_b - x_o \\ x_a - x_o \\ 2x_o - x_a - x_b \\ y_b - y_o \\ y_a - y_o \\ 2y_o - y_a - y_b \end{pmatrix}^T \begin{pmatrix} u_a \\ u_b \\ u_o \\ v_a \\ v_b \\ v_o \end{pmatrix} + \begin{pmatrix} u_b \\ -u_o \\ u_o - u_a \\ v_b \\ -v_o \\ v_o - v_a \end{pmatrix}^T \begin{pmatrix} u_a \\ u_b \\ u_o \\ v_a \\ v_b \\ v_o \end{pmatrix} \quad (62)$$

An angle is formed at every node for every two adjacent nodes. In a 2D structured grid, nodes can therefore have one, two or four angles that need to be constrained. The above equation represents just one of these angles.

To construct the system that will be minimised, the equation is linearised by dropping the 2<sup>nd</sup> order terms.

$$\vec{a} \cdot \vec{b} = \{D\}^T \{u\} + \{c\}^T \quad (63)$$

The coefficients of  $u$  form a matrix,  $[D]$ , with dimensions  $n \times d$  (where  $n$  is the number of nodes of the mesh and  $d$  is the number of degrees of freedom), and the constant is an  $n \times 1$  vector. Each row represents a node  $n_0$ , and each term in the above equation is positioned in the  $i^{\text{th}}$  column for the corresponding degree of freedom. The system can be solved using the linear least squares approach.

$$\frac{\partial \|[D]\{u\} + \{c\}\|^2}{\partial u_n} = 0 \quad (64)$$

There are two approaches to couple this system with the registration system. One can either solve to minimise the sum of the two residuals together, or alternatively solve to minimise both residuals independently. The former approach would require solving  $\frac{\partial \|[A]\{u\} + \{b\} + [D]\{u\} + \{c\}\|^2}{\partial u_n} = 0$  which would inevitably give rise to cross terms. Alternatively, the latter option seeks to minimise a weighted sum of the residuals of the differences in image intensities and the orthogonality criterion:

$$\frac{\partial \|[A]\{u\} + \{b\}\|^2}{\partial u_n} + \lambda \frac{\partial \|[D]\{u\} + \{c\}\|^2}{\partial u_n} \quad (65)$$

Expanding both derivatives,

$$\frac{\partial \{\{u\}[A]^T[A]\{u\} + 2\{b\}[A]\{u\} + \{b\}^2\}}{\partial u_n} + \lambda \frac{\partial \{\{u\}[D]^T[D]\{u\} + 2\{c\}[D]\{u\} + \{c\}^2\}}{\partial u_n} = 0 \quad (66)$$

$$[A]^T[A]\{u\} + [A]^T\{b\} + \lambda[D]^T[D]\{u\} + \lambda[D]^T\{c\} = 0 \quad (67)$$

Combining the two expressions together,

$$\{u\} = -[A]^T[A] + \lambda[D]^T[D]]^{-1} [[A]^T\{b\} + \lambda[D]^T\{c\}] \quad (68)$$

The key to an efficient implementation of this regularisation is accessing  $[D]^T$  sparsely and efficiently. In order to do this, starting from the node adjacency matrix, one must index all angles formed by connected nodes. Efficient code that loops through all nodes and construct  $[D]^T\{c\}\{x\}$  and  $[D]^T[D]$  can then be written. (Note that for  $[D]^T(\{c\}\{x\})$  the order matters for computational speed.) Only the angles centred on nodes that are not edge nodes and that are not part of a collapsed element are regularised.



The scalar product regulariser seeks to minimise the scalar product of the lines defining the element corners. This can indeed be minimised by making the lines orthogonal, but it can also be minimised by reducing the lengths of the lines. A more sophisticated approach to regularisation would use the orthogonality of the unit vectors along the line edges (insensitive to lengths), but an obvious alternative is to use the scalar product in combination with the Laplacian and thus to benefit from both. This theme is developed in the next section, in the context of another regularisation term that seeks to control the mesh quality using the Jacobian directly. It also suggests that the further addition of a volume-conserving regularisation might also provide benefit.

#### 4.4.3 A Jacobian-based metric of element orthogonality

Figure 42 shows the change of the Jacobian-based metric as a mesh element deforms. As a corner node moves near the centre, the Jacobian ratio climbs. Eventually, any further movement will break the element.

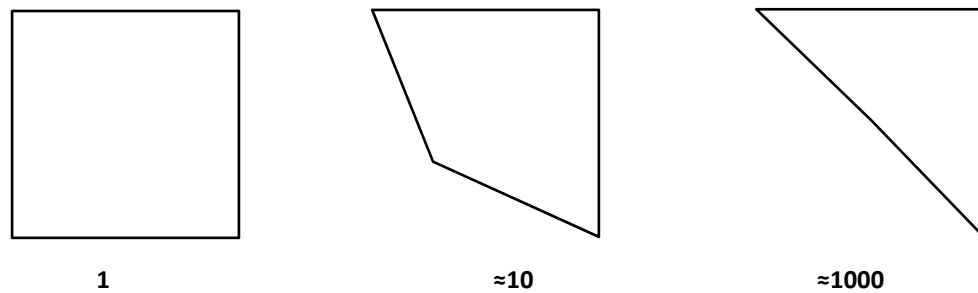


Figure 42: Variation of the element Jacobian for quadrilaterals under deformation

It is useful to express the Jacobian determinant at the local Gauss points in terms of the element nodes. For any point in an element, the Jacobian determinant is defined as

$$|J| = \begin{vmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \varphi} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} & \frac{\partial y}{\partial \varphi} \\ \frac{\partial z}{\partial \xi} & \frac{\partial z}{\partial \eta} & \frac{\partial z}{\partial \varphi} \end{vmatrix} = \frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \eta} \frac{\partial z}{\partial \varphi} + \frac{\partial x}{\partial \eta} \frac{\partial y}{\partial \varphi} \frac{\partial z}{\partial \xi} + \frac{\partial x}{\partial \varphi} \frac{\partial y}{\partial \xi} \frac{\partial z}{\partial \eta} - \frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \varphi} \frac{\partial z}{\partial \eta} - \frac{\partial x}{\partial \eta} \frac{\partial y}{\partial \xi} \frac{\partial z}{\partial \varphi} - \frac{\partial x}{\partial \varphi} \frac{\partial y}{\partial \eta} \frac{\partial z}{\partial \xi} \quad (69)$$

Of course the derivatives of the shape functions allow the following

$$x = \sum_{i=1}^8 N_i x_i \rightarrow \frac{\partial x}{\partial \xi} = \sum_{i=1}^8 \frac{\partial N_i}{\partial \xi} x_i \quad (70)$$

$$\begin{aligned} |J| = & \sum_{i=1}^8 \sum_{j=1}^8 \sum_{k=1}^8 \left( \frac{\partial N_i}{\partial \xi} x_i \frac{\partial N_j}{\partial \eta} y_j \frac{\partial N_k}{\partial \varphi} z_k \right) + \sum_{i=1}^8 \sum_{j=1}^8 \sum_{k=1}^8 \left( \frac{\partial N_i}{\partial \eta} x_i \frac{\partial N_j}{\partial \varphi} y_j \frac{\partial N_k}{\partial \xi} z_k \right) \\ & + \sum_{i=1}^8 \sum_{j=1}^8 \sum_{k=1}^8 \left( \frac{\partial N_i}{\partial \varphi} x_i \frac{\partial N_j}{\partial \xi} y_j \frac{\partial N_k}{\partial \eta} z_k \right) \\ & - \sum_{i=1}^8 \sum_{j=1}^8 \sum_{k=1}^8 \left( \frac{\partial N_i}{\partial \xi} x_i \frac{\partial N_j}{\partial \varphi} y_j \frac{\partial N_k}{\partial \eta} z_k \right) \\ & - \sum_{i=1}^8 \sum_{j=1}^8 \sum_{k=1}^8 \left( \frac{\partial N_i}{\partial \eta} x_i \frac{\partial N_j}{\partial \xi} y_j \frac{\partial N_k}{\partial \varphi} z_k \right) \\ & - \sum_{i=1}^8 \sum_{j=1}^8 \sum_{k=1}^8 \left( \frac{\partial N_i}{\partial \varphi} x_i \frac{\partial N_j}{\partial \eta} y_j \frac{\partial N_k}{\partial \xi} z_k \right) \end{aligned} \quad (71)$$

Finally, collecting terms together

$$\begin{aligned} |J| = & \sum_{i=1}^8 \sum_{j=1}^8 \sum_{k=1}^8 \left( \frac{\partial N_i}{\partial \xi} \frac{\partial N_j}{\partial \eta} \frac{\partial N_k}{\partial \varphi} + \frac{\partial N_i}{\partial \eta} \frac{\partial N_j}{\partial \varphi} \frac{\partial N_k}{\partial \xi} + \frac{\partial N_i}{\partial \varphi} \frac{\partial N_j}{\partial \xi} \frac{\partial N_k}{\partial \eta} \right. \\ & \left. - \frac{\partial N_i}{\partial \xi} \frac{\partial N_j}{\partial \varphi} \frac{\partial N_k}{\partial \eta} - \frac{\partial N_i}{\partial \eta} \frac{\partial N_j}{\partial \xi} \frac{\partial N_k}{\partial \varphi} - \frac{\partial N_i}{\partial \varphi} \frac{\partial N_j}{\partial \eta} \frac{\partial N_k}{\partial \xi} \right) (x_i y_j z_k) \end{aligned} \quad (72)$$

The Jacobian determinant of a Gauss point under nodal displacement  $\{u_i, v_i, w_i\}$  can be shown to be,

$$\begin{aligned} |J|_u = & \sum_{i=1}^8 \sum_{j=1}^8 \sum_{k=1}^8 \left( \frac{\partial N_i}{\partial \xi} \frac{\partial N_j}{\partial \eta} \frac{\partial N_k}{\partial \varphi} + \frac{\partial N_i}{\partial \eta} \frac{\partial N_j}{\partial \varphi} \frac{\partial N_k}{\partial \xi} + \frac{\partial N_i}{\partial \varphi} \frac{\partial N_j}{\partial \xi} \frac{\partial N_k}{\partial \eta} - \frac{\partial N_i}{\partial \xi} \frac{\partial N_j}{\partial \varphi} \frac{\partial N_k}{\partial \eta} \right. \\ & \left. - \frac{\partial N_i}{\partial \eta} \frac{\partial N_j}{\partial \xi} \frac{\partial N_k}{\partial \varphi} - \frac{\partial N_i}{\partial \varphi} \frac{\partial N_j}{\partial \eta} \frac{\partial N_k}{\partial \xi} \right) ((x_i + u_i)(y_j + v_j)(z_k + w_k)) \end{aligned} \quad (73)$$

The Element Jacobian measure is non-linear,

$$q_e = \frac{|J|_{min}}{|J|_{max}} \quad (74)$$

Furthermore, the extreme values of the Jacobian will be found at the element nodes, not the Gauss points. A useful residual measure is the change of the Jacobian ratio within the element measured at the nodes, but this is difficult to linearise. An alternative is to use the determinant at the centre of the element to normalise. This would have the advantage that instead of seeking to preserve the quality of the elements it would actually attempt to improve them. This will be called the Jacobian Ratio regularisation term. In this form, where  $det[J]_{i,u}$  is the determinant at node  $i$  after application of the displacement field  $u$ :

$$JR_i = \frac{J|_{i,u} - J|_{0,u}}{J|_{0,u}} \quad (75)$$

The determinant at the centre in the current iteration, neglecting higher order terms, is:

$$\begin{aligned} J|_{0,u} = J|_{0,0} &+ \sum_{p=1}^8 \sum_{q=1}^8 \sum_{r=1}^8 \frac{\partial N_p}{\partial \xi} \frac{\partial N_q}{\partial \eta} \frac{\partial N_r}{\partial \zeta} + \frac{\partial N_p}{\partial \eta} \frac{\partial N_q}{\partial \zeta} \frac{\partial N_r}{\partial \xi} + \frac{\partial N_p}{\partial \zeta} \frac{\partial N_q}{\partial \xi} \frac{\partial N_r}{\partial \eta} \\ &- \left( \frac{\partial N_p}{\partial \xi} \frac{\partial N_q}{\partial \zeta} \frac{\partial N_r}{\partial \eta} + \frac{\partial N_p}{\partial \eta} \frac{\partial N_q}{\partial \xi} \frac{\partial N_r}{\partial \zeta} + \frac{\partial N_p}{\partial \zeta} \frac{\partial N_q}{\partial \eta} \frac{\partial N_r}{\partial \xi} \right) \Big|_0 \\ &\cdot \left( (y_q z_r) u_p + (x_p z_r) v_q + (x_p y_q) w_r \right) \end{aligned} \quad (76)$$

Implementation of the Jacobian Ratio regularisation equation requires a linear approximation of  $1/J|_{0,u}$ . Using a binomial expansion and neglecting higher order terms:

$$\begin{aligned} \frac{1}{J|_{0,u}} = \frac{1}{J|_{0,0}} &\left( 1 - \frac{1}{J|_{0,0}} \sum_{p=1}^8 \sum_{q=1}^8 \sum_{r=1}^8 \frac{\partial N_p}{\partial \xi} \frac{\partial N_q}{\partial \eta} \frac{\partial N_r}{\partial \zeta} + \frac{\partial N_p}{\partial \eta} \frac{\partial N_q}{\partial \zeta} \frac{\partial N_r}{\partial \xi} + \frac{\partial N_p}{\partial \zeta} \frac{\partial N_q}{\partial \xi} \frac{\partial N_r}{\partial \eta} \right. \\ &- \left. \left( \frac{\partial N_p}{\partial \xi} \frac{\partial N_q}{\partial \zeta} \frac{\partial N_r}{\partial \eta} + \frac{\partial N_p}{\partial \eta} \frac{\partial N_q}{\partial \xi} \frac{\partial N_r}{\partial \zeta} + \frac{\partial N_p}{\partial \zeta} \frac{\partial N_q}{\partial \eta} \frac{\partial N_r}{\partial \xi} \right) \Big|_0 \right. \\ &\left. \cdot \left( (y_q z_r) u_p + (x_p z_r) v_q + (x_p y_q) w_r \right) \right) \end{aligned} \quad (77)$$

And so, again neglecting higher order terms:

$$\begin{aligned}
\frac{|J|_{i,u} - |J|_{0,u}}{|J|_{0,u}} &= \frac{1}{|J|_{0,0}} \left( |J|_{i,0} - |J|_{0,0} \right. \\
&+ \sum_{p=1}^8 \sum_{q=1}^8 \sum_{r=1}^8 \left[ \left( \frac{\partial N_p}{\partial \xi} \frac{\partial N_q}{\partial \eta} \frac{\partial N_r}{\partial \zeta} + \frac{\partial N_p}{\partial \eta} \frac{\partial N_q}{\partial \zeta} \frac{\partial N_r}{\partial \xi} + \frac{\partial N_p}{\partial \zeta} \frac{\partial N_q}{\partial \xi} \frac{\partial N_r}{\partial \eta} \right. \right. \\
&- \left. \left. \left( \frac{\partial N_p}{\partial \xi} \frac{\partial N_q}{\partial \zeta} \frac{\partial N_r}{\partial \eta} + \frac{\partial N_p}{\partial \eta} \frac{\partial N_q}{\partial \xi} \frac{\partial N_r}{\partial \zeta} + \frac{\partial N_p}{\partial \zeta} \frac{\partial N_q}{\partial \eta} \frac{\partial N_r}{\partial \xi} \right) \right] \right) \\
&- \frac{\det[J]_{i,0}}{\det[J]_{0,0}} \left( \frac{\partial N_p}{\partial \xi} \frac{\partial N_q}{\partial \eta} \frac{\partial N_r}{\partial \zeta} + \frac{\partial N_p}{\partial \eta} \frac{\partial N_q}{\partial \zeta} \frac{\partial N_r}{\partial \xi} + \frac{\partial N_p}{\partial \zeta} \frac{\partial N_q}{\partial \xi} \frac{\partial N_r}{\partial \eta} \right. \\
&- \left. \left. \left( \frac{\partial N_p}{\partial \xi} \frac{\partial N_q}{\partial \zeta} \frac{\partial N_r}{\partial \eta} + \frac{\partial N_p}{\partial \eta} \frac{\partial N_q}{\partial \xi} \frac{\partial N_r}{\partial \zeta} + \frac{\partial N_p}{\partial \zeta} \frac{\partial N_q}{\partial \eta} \frac{\partial N_r}{\partial \xi} \right) \right] \right) \\
&\cdot \left( (y_q z_r) u_p + (x_p z_r) v_q + (x_p y_q) w_r \right)
\end{aligned} \tag{78}$$

This expression is valid at every point in the domain, and so could readily be evaluated at each voxel in the image. The derivatives of the shape functions take on a specific numerical value at each position and, when multiplied by the appropriate co-ordinate terms, provide the coefficients of the displacements, or the entries in  $[\Gamma]$  in the regularisation process. This regulariser has the advantage that preserving the Jacobian tends to preserve the element volume, but this in itself makes it unsuitable for scaling operations.

This regularisation is considerably more computationally expensive than the previous one. Like the scalar product regularisation, the Jacobian regularisation produces a matrix and a vector and preparing the equations for solving requires inverting a massive matrix. Instead, the solution-ready matrices are built directly. The matrices are sparse and non-zero terms only arise between nodes that belong to the same element. Computation time is further reduced by pre-computing the shape functions and their combinations.

Elements at the edge of the mesh and collapsed elements are not regularised.

The downside to the Jacobian-based regularisation is that the characteristics of the minimisation problem results in rates of convergence of the least-squares solver that are considerably slower than the scalar product regularisation. On top of that, the system of equations is more expensive to set up. On the other hand, it tends to improve the mean element Jacobian of the mesh, as opposed to improving geometrical measures that describe less accurately the quality of the mesh.

## 4.5 Iterative registration, multiple increment

The discussion so far has focused on a single operation to register one image to another. Due to the approximations in the derivation of the equations (starting from the Taylor series expansion), it is likely that the result could be improved. One method is to compute the registration field and apply it to the moved image to create a new moved image, then perform the whole operation again. This can be repeated until there is no further reduction in the residual.

On each iteration, the regularisation term can be computed independently using the increment of displacement at that iteration. Alternatively, it can be computed based on the summed displacements over all iterations including the current one. The later approach produces a memory term that includes knowledge of the previous iterations. Generally, it is likely that this is a good thing because, for example, it is not only the smoothness or change in mesh quality over the last iteration that is important but the aggregate over the whole morphing operation.

There are, however, cases in which it might be better to forget previous iterations, and in some sense this might behave more like a fluid registration algorithm. This is illustrated in the context of the Laplacian regularisation term, in this case, the minimisation equation at iteration,  $p$ , reads:

$$\|R\|_2 = \|[A]\{u_p\} + \{b\}\|_2 + \left\| [\Gamma] \left\{ \{u\}_p + \left( \sum_{q=1}^{p-1} \{u\}_q \right) \right\} \right\|_2 \quad (79)$$

The last term is simply a constant at the current iteration (the sum of all previous iterations) and so, by comparison, the solution is:

$$\{u_p\} = - \left[ [A]^T[A] + \lambda [\nabla^2]^T[\nabla^2] \right]^{-1} \left( [A]^T\{b\} + \lambda [\nabla^2]^T[\nabla^2] \left( \sum_{q=1}^{p-1} \{u\}_q \right) \right) \quad (80)$$

The last term on the right can be regarded as a memory term. If it is included then the regularisation is applied based on the accumulated displacement over all iterations, whilst if it is omitted every iteration operates independently. Thus, using the cumulative Laplacian as the regularisation is expected to produce a solution that is smoother than without it.

## 4.6 Testing of regularisation options

To illustrate the effect of the different matrices described above, a number of tests were performed using each regularisation term. These tests provide valuable insight into the performance of the regularisation terms.

To test the Laplacian regularisation, a 2D implementation of the registration algorithm was coded in Matlab. For efficient and robust computation, the algorithm runs multiple times with decreasing node spacing in the registration grid. In the following test a parametric image of a small square is registered to a Cartesian image of a rectangle of larger dimensions. The edge nodes have been clamped for clarity. Figure 43 shows the experimental setup and Figure 44 summarises the resulting registration grid after successful registration.

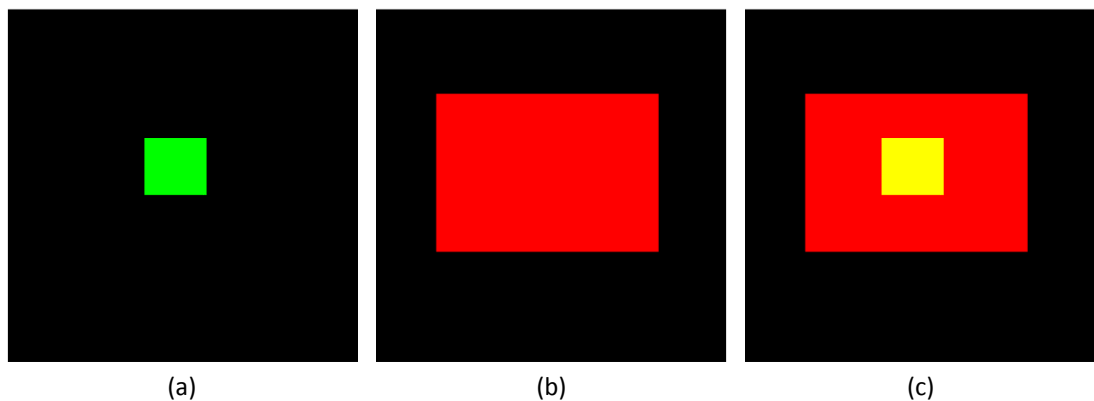


Figure 43: Input data for the regularisation tests showing (a) the small square, (b) the larger rectangle, and (c) both images superimposed. The overlap of the green and red images appears yellow.

It is clear from the images that there are large portions of the images that do not contain gradients. Nodes located in these regions "lack information" meaning the gradients matrix will be empty for these nodes. These nodes may contain intensity differences, those are collected in the right hand side vector in equation (53). Whilst the identity matrix is enough to regularise the otherwise ill-conditioned transformation matrix the solution for the aforementioned nodes is not clearly defined. This leads to folding in the registration grid and uneven element shapes, as seen in Figure 44 (left column).

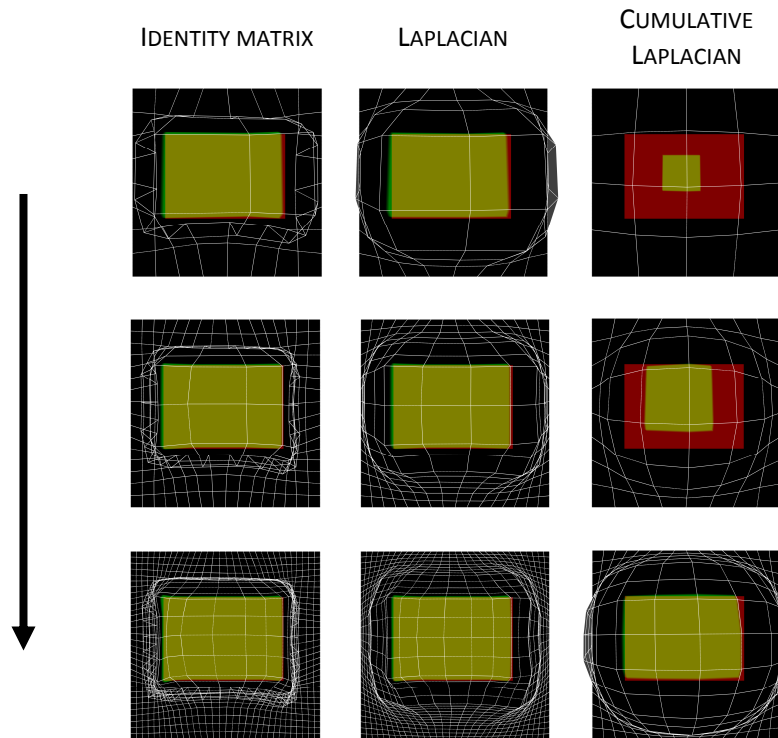


Figure 44: Effects of common regularisations used in image registration. The moving image (green) and the registration grid is presented on top of the fixed image (red). The images are blended to show the areas of overlap (yellow). The algorithm runs multiple times with decreasing node spacing in the registration grid, each row represents the last iteration at each grid spacing.

This problem is asymmetrical because the small quadrilateral is not centred in the middle of the big quadrilateral, nor are the corners of the quadrilaterals in the image aligned at the nodes of the registration grid, this can be seen in Figure 43 (c). If one thinks of this problem as the edges of the quadrilaterals needing to meet, in this case some edges will meet before others. This means each iteration deforms the registration grid in a different direction and therefore the overall grid does not appear smooth (this is clear in the result above). The Laplacian matrix smooths the displacement of the registration grid at each iteration meaning that the nodes with no image information are displaced to produce an even mesh (in practice this is rarely achievable due to image features, numerical errors, etc), this can be seen in Figure 44 (middle column). Another factor contributing to the folding of the grid is the value of  $\lambda$  being too low for some sections of the grid to remain smooth.

The cumulative Laplacian smooths the displacement of the registration grid from its initial position. This produces a grid that is far more evenly spaced as the notion of a smooth displacement field includes information about the previous displacements, as seen in Figure 44 (right-most column). Notice that this regulariser does not stop the grid from folding, although it does reduce the tendency to fold. Another feature of this regulariser is that the application of

the Laplacian across the cumulative displacement limits the ability to conform closely to geometric irregularities in the images - which can be seen both as a downside or an upside depending on whether these features are real or artefactual.

The scalar product regularisation forms an independent system of equations that may be minimised, and thus its performance in improving the angles in an irregular grid can be evaluated independently from the registration process. For this purpose, a basic unit test was implemented in Matlab. As well as evaluating the performance, this was useful to test the numerical correctness of the algorithm before it was implemented in the main Fortran code that is described in the following chapter.

In the following tests, a 25 node, 4-by-4 element 2D mesh is constructed. The nodes are regularly spaced except for the centre-most node, which is displaced in both spatial directions. An algorithm that solves the system presented above was implemented and solved in an iterative fashion; this test doesn't use any intensity values and only solves the regularisation equations. The initial mesh is presented in red (dotted line) and the final position is presented in blue (solid line). For reference, the regular grid is presented in light grey. The test is run for enough iterations for a minimum to be found. It is possible to perform the scalar product regularisation alone (results shown in Figure 45), or in combination with the Laplacian regulariser introduced previously (results shown in Figure 46). It is also valuable to investigate whether the registration is more stable if some of the nodes, for example around the edge of the image are directly constrained not to move. The left-most column indicates how many nodes are unconstrained, and the next four rows illustrate the nodal positions after one, two, five and ten iterations. For clarity, it must be noted that whenever a node or line lies in the same location, the blue grid is drawn on top of the red grid.



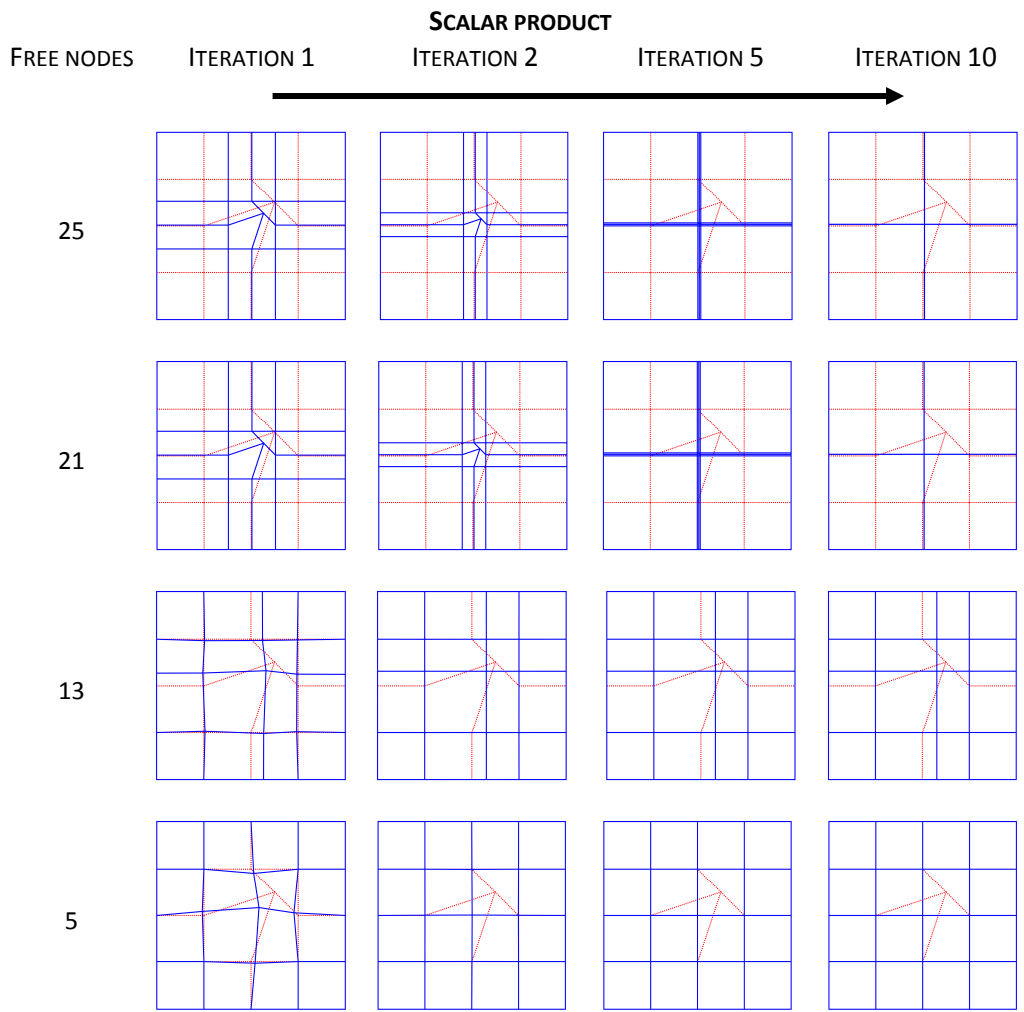


Figure 45: The effect of the scalar product regularisation on a distorted regular grid, allowing a different number of nodes to move (left-most column). The resulting nodal positions are shown after one, two, five and ten iterations.

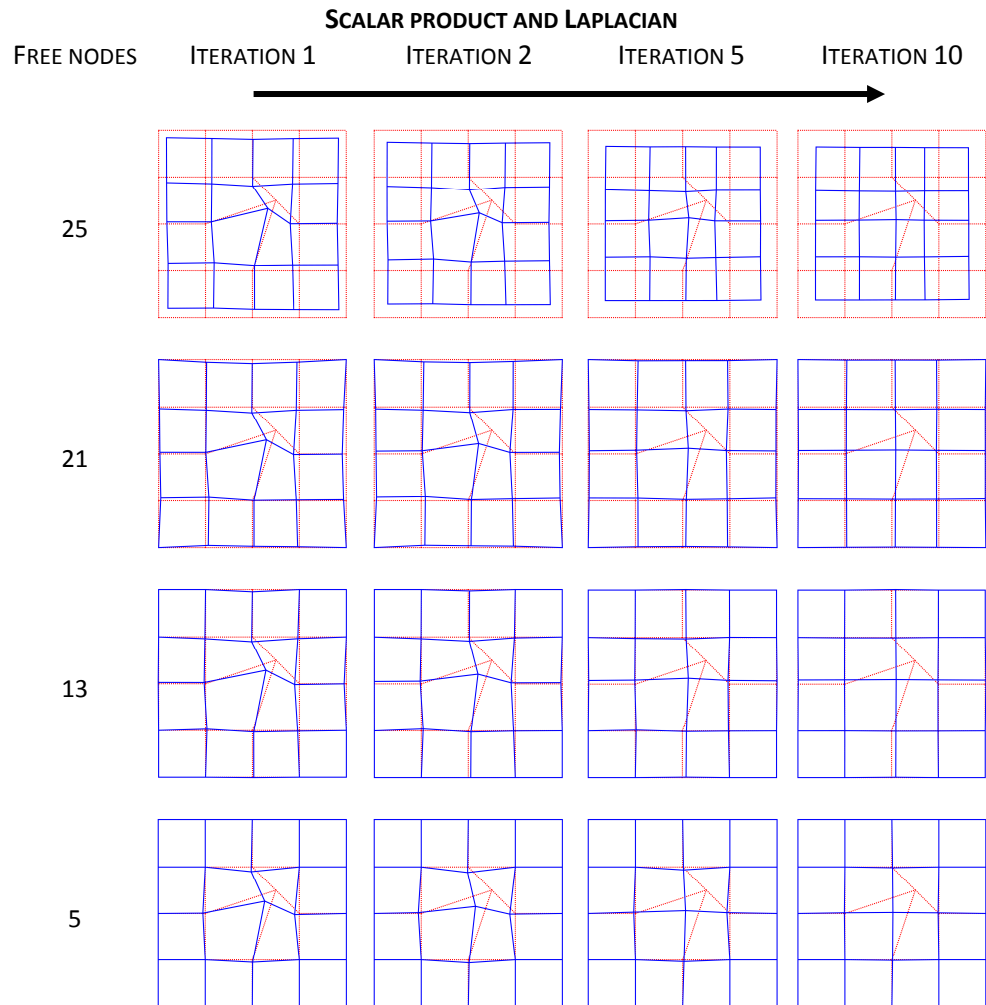


Figure 46: The effect of the scalar product regularisation with Laplacian smoothing on a distorted regular grid, allowing a different number of nodes to move (left-most column). The resulting nodal positions are shown after one, two, five and ten iterations.

If all 25 nodes are allowed to move, the system collapses the interior angles. (This can be seen in the top row of Figure 45) Adding the Laplacian smoothing solves this problem, but it forces nodes that were already forming right angles to move (Figure 46, top row). The same results are seen if 21 nodes are free to move and the outside corner nodes are clamped in position. If 13 nodes are allowed to move, with the nodes lying on the outside edge clamped, the system no longer attempts to collapse the angles. With only five nodes free to move (the centre node and its adjacent nodes), the perfectly regular mesh is generated. Thus it has been demonstrated that decreasing the number of degrees of freedom has the expected result.

As for the scalar product regularisation, the Jacobian regularisation forms an independent system of equations that can be solved directly, thus allowing its evaluation independently from the registration process (i.e. no intensity values are used). Once again, a basic unit test was implemented in Matlab.

In the following tests, a 64 node, 3-by-3-by-3 element 3D mesh is constructed. The nodes are regularly spaced except for the nodes that form the centre-most element, which is highly distorted. An algorithm that solves the system presented above was implemented and solved in an iterative fashion. For clarity, the initial mesh is presented as a blue wireframe in Figure 47, viewed from different camera locations. A 3D wireframe with the centre-most element presented with blue faces is also shown.

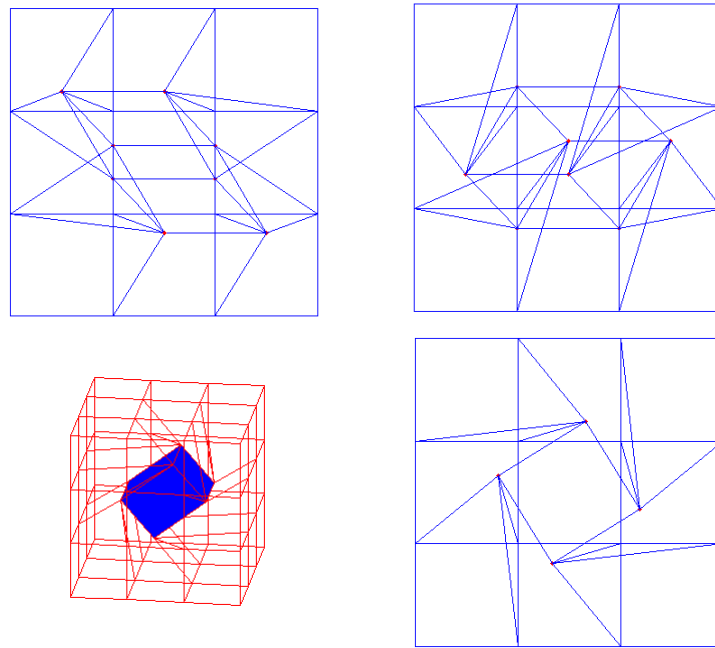


Figure 47: The initial setup of the distorted 27-element mesh used to test the Jacobian regularisation

The test is run for enough iterations for the mesh to settle to a solution that minimises the functional. In the test, the Jacobian regulariser is run with the Laplacian constraint. The nodes that do not form an exterior edge are free to move. Figure 48 shows the first six iterations in panels (a) to (f), and the final result in panel (g).

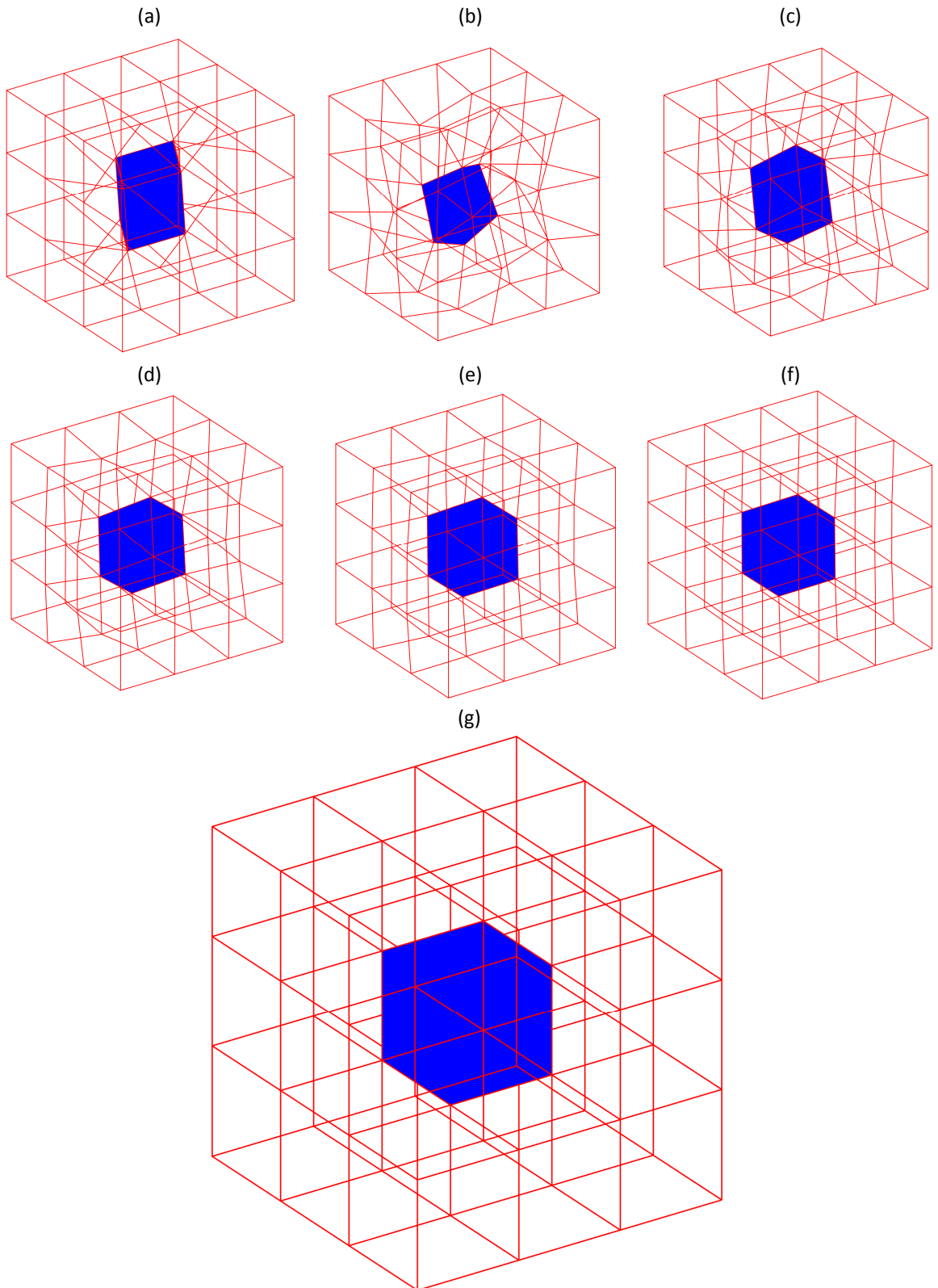


Figure 48: Node positions during the correction of the mesh shown in Figure 47 using the element Jacobian regularisation, at iteration numbers 1(a) to 6(f) and 15(g).

If all nodes are free to move, the system is singular even when using the Laplacian operator. It was found that the minimum constraint was clamping the edge nodes (those that appear in less than 3 elements), under which condition the system always solves to the same solution.

## 4.7 Discussion

Four transformation model types will be discussed, Rueckert's B-Splines, physical models, diffusion models and curvature models. Then, a discussion of the equations used in this chapter will be presented, including the similarities with other methods found in the literature.

Interpolation models assume the displacement can be estimated at a reduced number of points in the image and propagated through the image domain by interpolation. Then a similarity measure can be calculated and the displacements corrected to improve this measure. Rueckert's is a popular journal paper, with over 3200 citations according to Google scholar (as of September 2014). The deformation model described consists of a global affine transformation and a local free-form deformation, and uses B-Spline interpolation between the registration control points [103].

The global transformation is responsible for the general (global) alignment of the images and takes the form,

$$T_{global}(x, y, z) = \begin{pmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \\ \theta_{31} & \theta_{32} & \theta_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} \theta_{14} \\ \theta_{24} \\ \theta_{34} \end{pmatrix}$$

The local transformation is responsible for the fine detail (local) variations between the images, it takes the form,

$$T_{local}(x, y, z) = \sum_{l=0}^3 \sum_{m=0}^3 \sum_{n=0}^3 B_l(u)B_m(v)B_n(w)P_{i+l,j+m,k+n}$$

In the equation above, B represents the B-Spline basis function and P represents the control points.

Free-form deformations are needed to capture the high variability between the images that are to be registered. It is however a well-known fact that attempting to capture high deformations with a very un-regularised transformation can produce undesired results. The global transformation is a means to coarsely bring the images into alignment before the fine details are resolved; in practice, Rueckert first optimised the affine transform and then optimised the complete free-form deformation. The number of control points (or the spacing between them, depending on the view point) determines the "flexibility" (or rigidity) of the registration process.

Rueckert (as other authors, [106], [107]) favoured a coarse-to-fine hierarchical (or multi-resolution) approach. The registration parameters are optimised using an iterative gradient descent technique, balancing the cost of similarity and smoothness cost functions. The strength of this approach is the relative simplicity required to implement the algorithm, the robustness of the algorithm due to the combination of the affine and free-form models, and (very importantly) the implicit smoothness of the B-splines.

Transformation models based on physical model minimise a cost function based on both the deformation and the image similarity [108]. The deformation model can be elastic or fluid, and take the form:

$$\mu \nabla^2(\mathbf{u}) + (\lambda + \mu) \nabla(\nabla \cdot \mathbf{u}) + F = 0$$

In elastic models, this is the Navier-Cauchy equation (or the Lamé equation) where  $\mu$  is the shear modulus and  $\lambda$  the 1<sup>st</sup> Lamé constant (a function of the elastic and shear modulus). In fluid models, this is the Navier-Stokes equation and  $\mu$  is the fluid viscosity.

In both models,  $F$  are the external forces and are derived from the image intensities; typically, this is a form of the image gradients but other methods may be used [109]. The internal forces and the external forces are balanced until the deformed image reaches a state of equilibrium. The elastic properties play a critical part in the solution of the registration problem. The main difference between the models is that elastic implementations solve for the displacements,  $\mathbf{u}$ , and fluid models solve for the velocities,  $\mathbf{u}$ .

Registration methods based on physical models are solved using a variational approach. The resulting effect is that the image forces deform the moving image but the resulting deformation is constrained by the properties of the physical model used. The shear (or viscosity) terms in physical models penalise affine transformations. Methods that penalise affine registrations require an affine pre-registration step to bring images into better alignment, as described [106], [110].

A set of transformation models categorised as diffusion models (also called Thirion's demons) take the form

$$\Delta(\mathbf{u}) + F = 0$$

And curvature models take the form

$$\Delta^2(\mathbf{u}) + F = 0$$

These two methods may be categorised as physical models because of the proposed reasoning behind their derivation however the correctness of such classification has been debated [19], [99]. As physical models, these methods are solved using a variational formulation

(characterised by the Euler-Lagrange equations) leading to systems of non-linear partial differential equations, allowing efficient implementations.

The curvature model (as described in [111], [112]) is an exercise of minimising a criterion based on the grid curvature and the sum square differences:

$$R(u) = \lambda \frac{1}{2} \sum_{l=1}^d \int_{\Omega} (\Delta u_l)^2 dx + \frac{1}{2} \int_{\Omega} \left( T(x - u(x)) - R(x) \right)^2 dx$$

The authors stress that this curvature model does not penalise affine transformations and thus does not require a linear pre-registration step to be successful. In practice, it may still be desirable to perform a linear registration to increase the robustness of the registration process.

Thirion observed that optical flow behaves like both diffusion models and deformable models based on attraction [113]. The regularised equations used in this work are summarised in Table 7.

Laplacian	$[A]\{u\} + \lambda[\nabla^2]\{u\} + \{b\} = 0$
Laplacian + memory	$[A]\{u_p\} + \lambda[\nabla^2]\{u_p\} + \lambda[\nabla^2] \left( \sum_1^{p-1} \{u\}_q \right) + \{b\} = 0$
Scalar product	$[A]\{u\} + \lambda[\nabla^2]\{u\} + \lambda_c[D]\{u\} + \{b\} + \lambda_c\{c\} = 0$
Jacobian	$[A]\{u\} + \lambda[\nabla^2]\{u\} + \lambda_c[J_v]\{u\} + \{b\} + \lambda_c\{J_0\} = 0$

Table 7: Summary of the regularised transformation model used in this thesis

In a similar fashion as described in Chapter 3, a force term can be derived from the first order approximation of the Taylor series expansion; here, A is a matrix composed of the gradients of F:

$$F(u) = \{b\} + [A]\{u\}$$

This way, the Laplacian transformation model shown in Table 7 is expressed as:

$$F(u) + \lambda \nabla^2(u) = 0$$

In other words, the curvature transformation model is equivalent to the Laplacian regularisation term. The difference comes in the way the equations get solved; in [111] the equations are solved using a variational approach. This work instead chooses to minimise the L2-norm of the residual described in Chapter 3. There are several benefits to the approach used in this work. Indeed, for the variational approach gives rise to a set of fourth-order partial differential equations to describe the minimisation of the displacement which requires  $O(N \log N)$  iterations to solve, where N is the number of image pixels. In contrast, the method presented in this thesis solves a linear set of equations that are solved using the conjugate gradient method in  $O(N)$

iterations, where N is the number of degrees of freedom. Typically, there are far less degrees of freedom than voxels in an image.

The second draw-back of the approach presented by [111] is that, by analogy with the work described in [99], the optimisation does not directly optimise the similarity metric employed (the sum of squares difference). Indeed, [99] show that deriving the diffusion equation in a fashion similar to the method presented in this work is a more effective way to minimise the image differences, even if the formulations of the equations are very similar. The method presented in this thesis can be compared with curvature models where the residual does directly increase the similarity between the images. From this point of view, the Laplacian becomes a regularisation to the registration and not the registration model.

To put the other terms into context, it is worth looking at another natural choice for the regularisation term, based on structural considerations. The image that is to be moved, or morphed, to the fixed image can be thought of as an elastic body. The displacement at all points in the body can be described by a three dimensional field with Cartesian components (u, v, w). In a finite element representation the body is divided into a series of elements which define the connectivity between nodes at which the governing parameters (often displacements) are determined.

The three dimensional strain ‘vector’ (a collapsed shorthand representation of the strain tensor) at any point can be written:

$$\{\varepsilon\} = \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{Bmatrix} = \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial w}{\partial z} \\ \frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \\ \frac{\partial w}{\partial y} + \frac{\partial v}{\partial z} \\ \frac{\partial w}{\partial x} + \frac{\partial u}{\partial z} \end{Bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{bmatrix} \{u\} \quad (81)$$

or:

$$\{\varepsilon\} = [B]\{u\} \quad (82)$$

Now returning to the idea of the regularisation of the linear least squares problem, equation (53), one possible choice of the Tikhonov matrix is a linear multiple of the strain matrix. This means that the vector for which the L2 norm is being added is, in fact, proportional to the vector of strain at the node points of the mesh. Then the solution is:

$$\{u\} = - \left[ [A]^T[A] + \lambda[B]^T[B] \right]^{-1} [A]^T\{b\} \quad (83)$$



With this formulation, shear can be penalised if a particular application requires such constraint. This regulariser is implemented in the Sheffield Image Registration Toolkit and the experience of the developers is that it has rarely proven to be useful.

Deformations that preserve the strain on the grid are a common choice in the field of deformable models. From the above equation, it is obvious that the so-called external forces employed in deformable models are analogous to the registration equations and the internal forces of the model are analogous to the constant strain constraint. However, despite the similarities in the constituent equations, it can be shown that diffusion and curvature are not a simplification of the elastic models. This is because both terms of the internal forces in the elastic model contain contributions from the shear terms.

The difference between the external forces that are commonly used in deformable models and the formulation used in this work is that, by considering the model to be an image defined in parametric space, two sets of gradients are used and this makes the registration process much more robust as it is pushed by two sets of gradients.

## 4.8 Summary

This chapter has described the motivation for regularisation of the registration equations, evaluated several regularisation options (including a novel mesh improvement algorithm) and explained the position of this work in the more general context of image registration (including deformable models).

The two novel contributions made by this work are 1) the use of an unstructured grid to define the image without reducing the image to a deformable model, and 2) the introduction of terms to regularise the shape of the unstructured grid to preserve and improve grid quality for finite element analysis. An interesting improvement to this work would be employing cubic basis functions to impose an inherent smoothness in the registration field.

# CHAPTER 5

## IMPLEMENTATION AND VALIDATION ON SYNTHETIC DATA

### 5.1 Introduction

This chapter describes the implementation details of the image registration processes described in Chapter 3 and Chapter 4. Figure 49 summarises the implementation overview presented in section 5.2.1 and expanded-on in the remainder of section 5.2. Section 5.3 shows the results of a proof-of-concept program and section 5.4 shows the results of some tests performed on simplified geometrical shapes. These tests are designed to incrementally test and understand the contribution of different parameters to the registration.

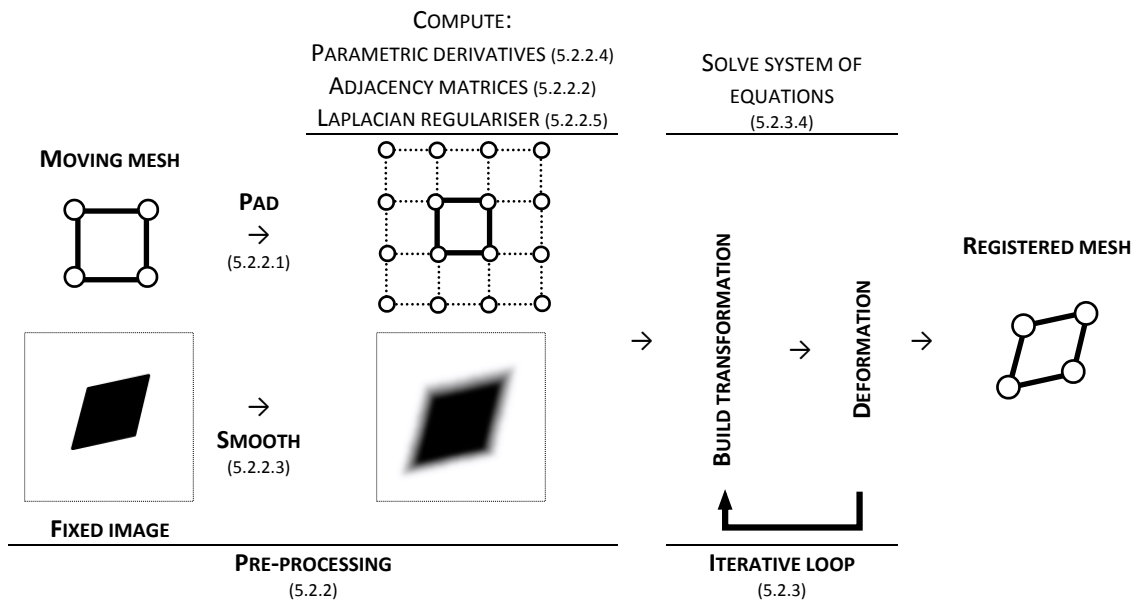


Figure 49: Flowchart showing the different stages of the registration program described in section 5.2

## 5.2 Implementation

### 5.2.1 Overview

The image registration program can be broken down into routines of increasing specificity. For instance, there are a number of main program routines that handle the global program variables which are called only once, and a number of specific numerical routines that are called many times during execution.

The main program routines handle the input and output of data (file reading and writing, command line argument interpreting) and the flow of the program (initialise variables, prepare data, build and solve the system of equations).

The data or operations on data that needs preparing before-hand, as discussed before, includes image smoothing, mesh smoothing (smoothing an image in parametric space), computing the mesh derivatives in parametric space, resolving the sampling points in local coordinates and indexing of nodal connections.

Those pre-processing routines make use of more specific subroutines that construct big data matrices, such as building the transformation matrix and each of the regularisation terms: the identity, Laplacian and strain matrix, and the scalar product and Jacobian-ratio regularisation matrices.

To build the transformation matrices, the image gradients need to be computed, for this a number of specific routines are needed, such as for interpolating the intensity (and gradient) values of mesh and image at specific locations, transforming the parametric gradients into Cartesian gradients, and resolving the shape function values and derivatives at specific locations.

To build the regularisation matrices, the nodes and element adjacency matrices are required, as well as the extended node and element neighbours for the geometric regularisation terms.

A number of routines are required to handle the data matrices stored in sparse mode. In addition to the aforementioned key-value pair to row-indexed sparse storage mode converting routine, routines to remove and/or merge duplicate and/or empty entries are needed. General vector operations include vector sorting (for multiple data-types and dimensions), vector-matrix multiplication, matrix-matrix addition and matrix-matrix multiplication.

### 5.2.2 Pre-processing stage

The initial mesh is created by an external process to the registration. This process can be manual, by defining the coordinates of the points and their connectivity or automatic, by parametrising the definition of the mesh. For example, the cubes used in section 5.4 of this chapter were defined manually. On the other hand, the ventricular meshes used in chapter 6 were automatically made to approximate the shape of the segmentation based on a number of characteristics of the ventricular geometry, as described in chapter 6, section 6.3.2.

### 5.2.2.1 Mesh padding

In the pre-processing stage, some key information about the images is generated and stored for efficient access in the later stage. For instance, the mesh is padded, the nodes and elements are indexed and the intensity derivatives of the moving mesh are calculated.

As already discussed in previous chapters, the driving force of the registration is a function of the intensity differences and the intensity gradients of the images. In the binary mesh (or unstructured, binary image) all voxels have the same value so, to introduce a gradient, the mesh is extended by adding a layer in the outside boundary, and the voxels of the new elements are given a value equivalent to the image background.

To pad the mesh, the nodes on the outer limits of the mesh are selected and then the faces are extruded outwards to form new elements. This is a simple process as one can take advantage of the fact that, in the hexahedral structures used in the current analysis, internal nodes appear on 8 elements (that are not collapsed), and therefore any nodes that feature in fewer elements must lie on outer faces.

### 5.2.2.2 Adjacency matrices

The adjacency matrices describe the connectivity of nodes and elements. The information required to build these matrices is already contained in the definition of the mesh's elements, but reformatting allows for efficient querying of information.

The node adjacency matrix is built by accumulating the node connectivity defined at each element, over all elements. The element adjacency matrix is built by matching faces with the same nodes. In both cases, vectored binary logic makes an efficient implementation possible. The mesh definition must use consistent element node numbering, as illustrated in Figure 50.

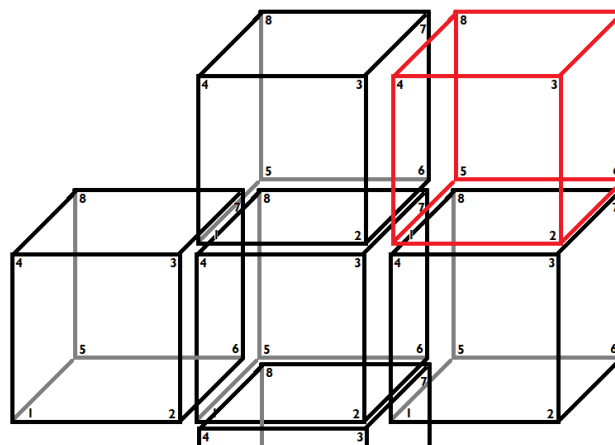


Figure 50: Consistent node numbering of mesh elements. The element in red contains extended neighbours of the centre-most element

### 5.2.2.3 Image smoothing

In terms of the stability of the registration process, image smoothing has three advantages:

- it increases the regions of the image in which gradients are found, thus increasing the distance between the two images that can be accommodated by the registration process;
- the magnitudes of the gradients decrease, making the linear assumption more valid;
- noise is reduced.

On the other hand, image smoothing is a non-reversible operation that leads to local details being lost. However, the averaging assumptions used elsewhere in the program make this detail irrelevant for most applications. It is common practice to use smooth images for registration. A common way of smoothing images is by applying a Gaussian kernel; in this work a Gaussian kernel with a 3x3x3 neighbourhood used.

In the particular application described in this thesis, the fixed image is the binary segmentation of a patient-specific heart and the moving image is the binary parametric-space representation of the template mesh generated. Smoothing the mesh uses the same technique except that care must be taken when the neighbouring voxels of a point are located on a neighbouring element. In an un-smoothed mesh, voxels in the original elements will have a value of 1 and voxels in the padding elements will have a value of 0.

### 5.2.2.4 Image gradients

As discussed in Chapter 3, the residual used in the registration algorithm is sampled at finite locations determined by the elements of the moving image (the mesh). The gradients of the mesh in parametric space do not change as the mesh is deformed, and so they only need to be calculated once. Thus, this is performed in the pre-processing stage.

The intensity gradients are calculated for each element at every sampling point. The sampling points are placed in a regularly spaced sampling grid and a second-order approximation to the first derivative is used.

### 5.2.2.5 Laplacian filter

The Laplacian filter (described in Chapter 4, section 4.3.2) acts on the resulting vector field of the registration process and consequently has the same form throughout the execution of the program. The construction of the Laplacian filter is described in the detail in Chapter 4. One of the choices to be made is how to handle the boundary of the image. In principle the mesh can be completely free, relying on the regularisation term to deal with any singularity of the transformation matrix, or it can be fixed at some or all points on the boundary. Generally it has

been preferred in previous applications of the Sheffield Image Registration Toolkit to allow the boundaries of the moving image to move freely, but it might be more robust in some circumstances to add constraints to prevent this motion. This is discussed and illustrated in some of the test cases that follow.

### 5.2.3 Iteration stage

This part of the program is responsible for constructing the transformation model described in Chapter 3 and finding the optimal registration parameters. The transformation model accumulates the image and mesh information, sampled at the sampling points, at the mesh nodes.

#### 5.2.3.1 Location of sampling points

A key feature of the computational implementation of the method presented in this thesis is that mesh elements can be processed locally, similar to the finite element method. The mesh is described in an irregular, unstructured grid form. The parametric sampling is however regular within each element of the mesh. The implication of this, in terms of the spacing between the sampling points, is that the Cartesian distance between points might not be correctly accounted for at points on the edge of elements.

In Figure 51, the first two elements (counting from the left) have the same length in  $x$ , ensuring the spacing across elements is constant. Conversely, with the last element being longer, the spacing across elements differs by the amount marked in red.

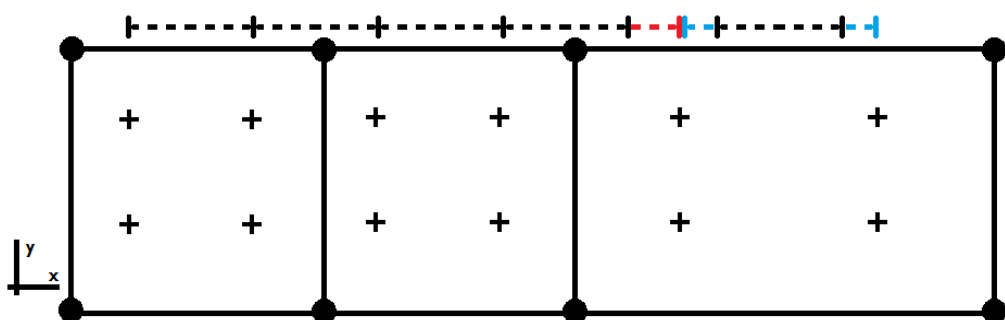


Figure 51: The layout of the sampling points, showing the effect of varying the size of the elements has on the distances of points across elements

The spacing between points in the last element is bigger in  $\{x, y\}$  space (marked in blue) but constant in  $\{\xi, \eta\}$  space; the difference marked in blue is accounted for by the Jacobian when transforming from one space to another. On the other hand, because the elements are

computed in isolation, the difference marked in red will not be accounted for by the Jacobian when being computed from the centre element. This difference in inter-element node spacing can have a significant contribution to the overall error; this is especially true for un-smoothed binary images, where gradients only appear at object edges.

The fixed image (the binary segmented patient image) is described in a regular structured grid form. This means that the voxels are equally sized. When  $\partial f / \partial \mathbf{x}$  (and all dimensions) is worked out, it is assured the  $\partial \mathbf{x}$  component is constant throughout the registration and across all voxels.

### 5.2.3.2 Image interpolation

Since the intensity values are only described at finite points of the fixed image, the values at the sampling points must be interpolated. A tri-linear interpolation scheme was used in this program. Tri-linear interpolation is more computationally intensive than nearest point amongst other things because to perform this type of interpolation, the location and values of the eight enclosing points must be known (as opposed to the single closest point). To interpolate the gradient values, the gradient at all voxel positions was calculated, then each Cartesian component of the gradient is interpolated independently.

Different interpolation methods are demonstrated using 5-by-5 voxel images in Figure 52. The methods shown are nearest neighbour, bilinear, bicubic and cubic spline interpolation (left-to-right). The top row set of images use 4-by-4 sampling points between the voxels; the bottom row set of image use 400-by-400 sampling points between the voxels.

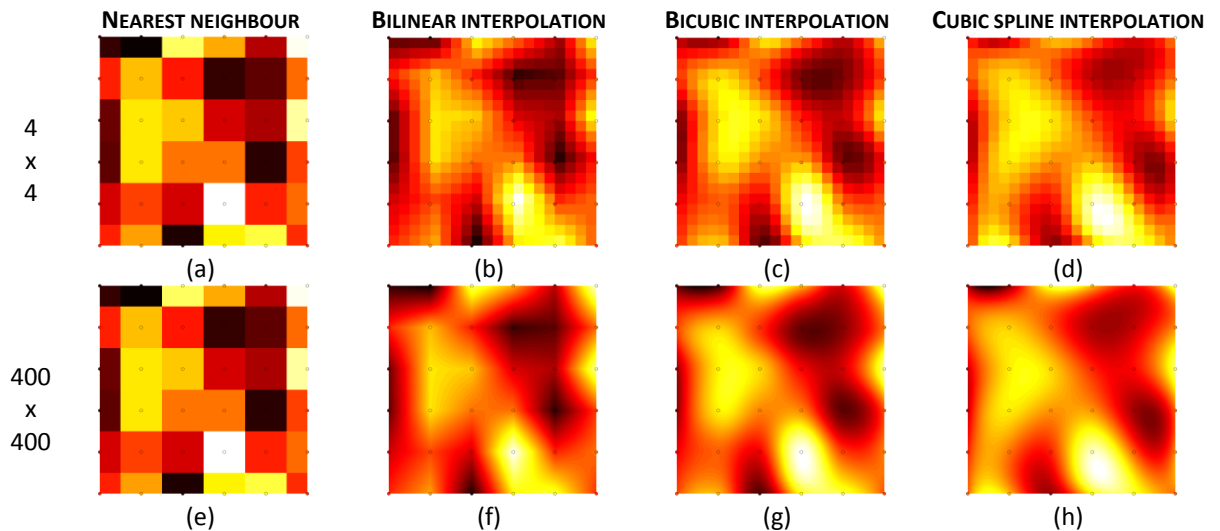


Figure 52: Typical methods for interpolating data, in increasing order of complexity: nearest neighbour, bilinear interpolation, bicubic interpolation and cubic spline interpolation. Panels (a) to (d) use a coarse, 4-by-4 sub-pixel sampling grid. Panels (e) to (h) use a fine, 400-by-400 sub-pixel sampling grid. Cubic spline interpolation is efficient for fine sub-pixel sampling however this work does not benefit such refinement, in which bilinear is the most appropriate sub-pixel sampling method.

The images were interpolated using fine and coarse sub-pixel sampling grids, as seen in Figure 52. First, the fine grid (panels (e) to (h)) is used to show the effect on the data more clearly; the relative computational time for generating the data was 1.00, 2.90, 9.14 and 2.86 for the nearest neighbour, bilinear, bicubic and cubic spline interpolators, respectively. For fine sampling, bilinear interpolation is over twice as computationally expensive as nearest neighbour and as expensive as using cubic splines.

The relative number of sampling points with respect to voxels in the image is much smaller in the application described in this thesis, similar to the coarse grid shown in panels (a) to (d). For coarse sampling, the relative computational time for generating the data was 1.00, 1.16, 1.73 and 7.17. It is clear now that the computational cost of bilinear interpolation is small compared with the improvement in smoothness, and that spline interpolation involves heavy computation. All images were created using Matlab, and the timing-data was averaged over 1000 runs and normalised with respect to nearest neighbour interpolation.

The mesh intensity-gradients described in Cartesian space are dependent on the location of the sampling points. The gradients described in parametric space must be transformed to Cartesian space each time the transformation matrix is built.

### 5.2.3.3 Sparse storage formats

A naïve storage strategy for all program variables would use full-size vectors and matrices to store all values in the transformation and regularisation matrices. The number of entries in these



matrices is equal to the product of the number of degrees of freedom and the number of sampling points. For the size of the problems solved in this work, storing this amount of double-precision floating point data would require several Gigabytes of RAM per matrix. A more efficient strategy is to take advantage of the sparse nature of the matrices.

Following the conventions recommended in the book *“Numerical Recipes in C”*, matrices were stored in Row Indexed Sparse Storage Mode (RISSM) to allow efficient storage and accessing of data by the solution routine [114]. Although this storage mechanism was adopted, in practice it has little advantage over a simpler ‘intermediate sparse formulation’. The intermediate sparse formulation is a straight-forward key-value pair (KVP), where the key is a two element vector representing the row-column index of each non-zero value. The gain in storage space of RISSM compared to KVP is marginal, and the faster access times are due to the indexed nature of the formulation, which can also be achieved in KVP by holding a separate index of the rows. Keeping the matrices in KVP form allows for a much simpler code to both execute and maintain, and might be preferred for future development of the code.

Another important consideration is that the solver requires the computation of the square of each of the matrices. Matrix-matrix multiplication of a matrix of size  $n$ -by- $m$  involves  $n$ -times- $m$ -times- $m$  multiplication and addition operations. For any useful application, this is impractical. The squared representation of the transformation matrix is built directly to avoid the matrix multiplication operation. During assembly of the transformation matrix, each element is processed in turn and the contribution to the transformation matrix is calculated for each node. In addition, the node cross-products are calculated to compute the entries of the square matrix.

#### 5.2.3.4 Linear solution

The equations are solved using an un-optimised implementation of the linear, bi-conjugate gradient method, based on the LINBCG routine found in *“Numerical Recipes in C”*. Unit tests using a variety of matrices were performed to ensure the routine worked as expected. The conjugate gradient method was chosen over Gaussian elimination and SVD decomposition for several reasons, such as robustness, speed and effectiveness of operation with sparse data structures.

#### 5.2.4 Choice of programming language

The registration algorithm was implemented in Fortran 90. Fortran was developed in the 1950s and became the first widely used general-purpose programming language; it remains the reference language for high performance computing. There are two main motivations behind

the language design, numeric computation and high performance; thus, it is perfectly suited for scientific computing [115].

By design, the language allows simple constructs and is generally strong typed; furthermore, it only allows basic pointer logic and dynamic memory allocation. The first two design choices mentioned have lost popularity amongst modern languages, whilst pointer logic still has strong advocacy amongst performance-focussed programmers. Simple constructs and static, type-safe variables support the writing of code that is unambiguous to the compiler; the strength of Fortran is that it can be used with the most efficient compilers. Compilers are programs that convert text written in a particular programming language and convert it to machine code; Fortran is able to produce efficient machine code without having to master assembly language. Modern Fortran (the Fortran 90 specification) lends itself to the writing of code that is easy to read and maintain, and parallelisation can be achieved using the OpenMP extension [116].

The registration program is divided into two stages, the preparation (or pre-processing) stage and the registration (or iteration) stage.

### 5.2.5 Numerical correctness

Compile-time checks ensure the code is syntactically correct however one must still ensure the program builds and solve the correct system of equations. Small numerical inaccuracies might not be obvious by inspection as solving by best fit (linear least squares) smooths the results anyway. It is also hard to follow the step by step process as the system of equations is built directly into a sparse form.

The major components of the code were tested independently, that is, the building of the gradient matrix, the Laplacian, scalar product and Jacobian regularisation matrices, and the conjugate gradient solver. One of the advantages of coding a prototype in Matlab was that the matrices, and other intermediate results, generated in the prototype environment could be compared directly with those generated by the 'production' Fortran code. Although these tests cannot trap algorithmic flaws, they proved very valuable in detecting coding errors.

## 5.3 Proof-of-concept

A proof-of-concept implementation was created using the MATLAB environment. MATLAB is a language designed to allow quick prototyping and to handle low level mathematical operations efficiently [117]. The program registers two two-dimensional images using the transformation model and optimisation procedure described in Chapter 3. The cumulative Laplacian

regularisation scheme described in Chapter 4 (section 5) was used. To improve the robustness of the process, the registration is performed multiple times with a registration grid of increasing resolution. This assists particularly in situations in which the moved image is further from the segmented image, either globally or locally, by bringing them into better alignment, before the operations on the finer grid improve the local fit. The image edges can be clamped or be free to move. In these tests, the moving image is described as a mesh with the nodes aligned in a regular Cartesian arrangement.

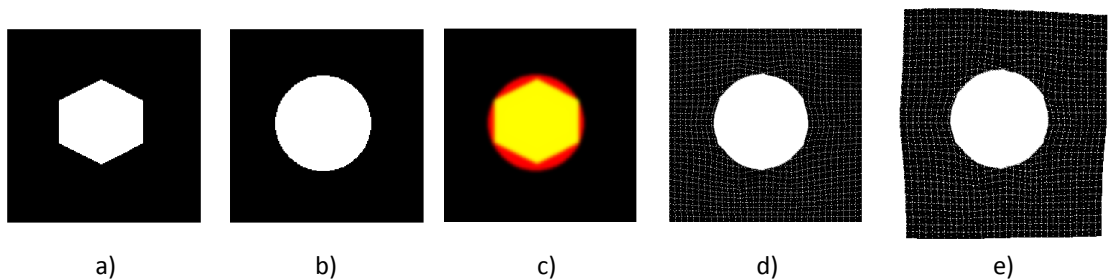


Figure 53: Registration of a hexagon to a circle showing a) the moving image, b), the fixed image, c) the images overlay, d) the registration result using fixed edges, and e) the registration result using free edges.

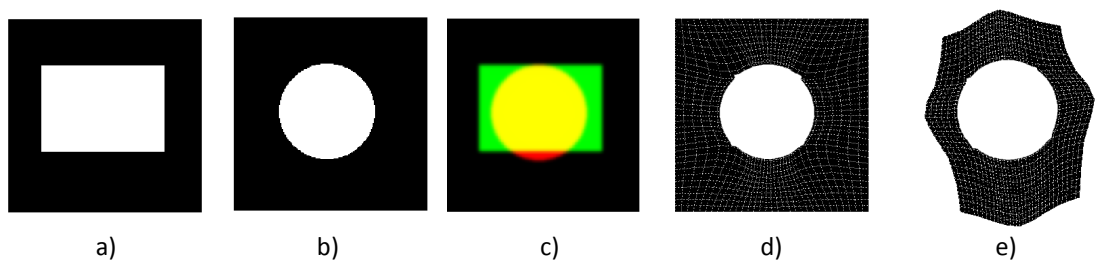


Figure 54: Registration of a rectangle to a circle showing a) the moving image, b), the fixed image, c) the images overlay, d) the registration result using fixed edges, and e) the registration result using free edges.

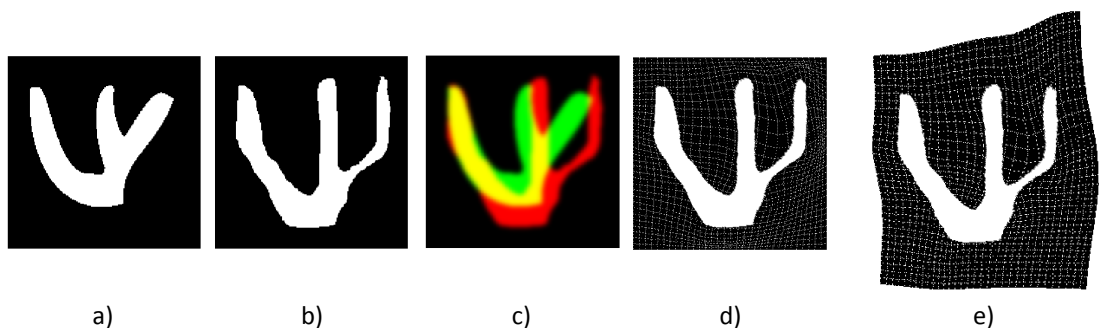


Figure 55: Registration of a “w-like” complex shape to a similar shape, which has been distorted considerably, showing a) the moving image, b), the fixed image, c) the images overlay, d) the registration result using fixed edges, and e) the registration result using free edges.

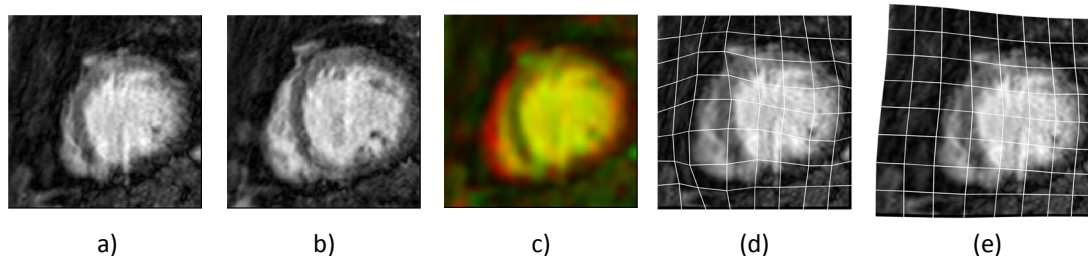


Figure 56: Registration of two slices of a cardiac MRI showing a) the moving image, b) the fixed image, c) the images overlay, d) the registration result using fixed edges, and e) the registration result using free edges.

The results show that the procedure works. In the test shown in Figure 53, the images are very similar initially but this is not the case in the other tests. This is especially obvious in the test shown in Figure 54 where the grid with free edges becomes strongly deformed by the registration operation. It is interesting to note the fixed image (circle) has no particular orientation and therefore the registration could rotate around the centre of the circle and still produce an optimal similarity measure. In the test shown in Figure 55, the right-most branch needs to go through a significant deformation whereas the left-hand-side of the image is relatively similar. The free grid produces a more robust solution in this case because it allows the registration to occur more naturally, without excessive distortion of the grid around the edge (although a good result is shown using the fixed edges, the operation was very sensitive to the weight of the regularisation term). Figure 56 illustrates the operation of the registration process on more typical medical images that contain different intensities. Throughout this work, the images that are registered are binary images (pixels either have a value of 0 or 1) however the underlying algorithm works on the whole range of numbers between 0 and 1. This is clear when one considers that the images are blurred prior to registration (smoothing a binary image turns it into a greyscale image), or indeed in Figure 56 (the grey values are clearly visible in the figure).

## 5.4 Validation

### 5.4.1 Assessment criteria

#### 5.4.1.1 Fitting accuracy

There is no ground truth to compare the registration method against and if there were, one could question why that method was not used in the first place [89], [118]. Instead, the performance of the algorithm is evaluated using a set of accuracy metrics.

The accuracy of the results will be dependent on the number of sampling points used. The registration sampling points are defined at irregular positions that do not coincide with the

image voxels; for this reason for coarse sampling the image edges might not coincide with the mesh edges. Moreover, for the same number of sampling points, bigger mesh elements will result in coarser sampling.

The system will match all the sampling points in the mesh but for binary images the differences are only found at the image edges: within the regions of constant intensity the movement is not defined by the gradients matrix, but rather by the smoothing terms.

In principle, all intensities can be found perhaps not at the locations where the voxels are defined; in practice, this is limited by the available voxels, the ranges of the image and the interpolation scheme. In general, it is safe to assert that in image registration there is no guarantee that every voxel in one image will have a matching partner in the other image, at least not within the validity of the first order Taylor series approximation. In the original Sheffield Image Registration Toolkit this is dealt with by the introduction of an additional dimension, the intensity direction, into all images. The resulting images are all binary, and indeed a special type of binary in which there is a single surface, in  $(n+1)$  dimensional space, separating the voxels with a value of unity from those with a value of zero. Under these conditions it is at least guaranteed that a mapping exists, although there is no unique mapping. The selection of a particular solution is driven by the regularisation term. This higher dimensional approach has not yet been implemented in the current code. Nevertheless the operation on binary images means that there will always be a solution that maps between the two images.

Matching values at sample points is straight forward. The image similarity metric used in this chapter is the standard absolute difference  $SAD = \frac{\sum |F-M|}{N}$ , which corresponds to the cost function used for registration. It is minimum when the images are the same.

The surface distance error is defined as the Euclidean distance from the fixed image surface to the closest point on the surface of the moving mesh. The image surface is defined at the voxel corners where the segmented image has a sharp intensity change. In theory, the limiting factors for this measure will be the distance between sampling points.

#### 5.4.1.2 Mesh quality

The quality of the mesh is a measure of how stable the solution will be in numerical simulations. Mesh quality plays a crucial role in the accuracy, robustness and convergence time of numerical simulations. In fact, a mesh of poor quality will also affect the quality of this registration method. Mesh quality is a geometrical measure and many measures have been proposed to quantify it, such as element skew-ness and element aspect ratio. The element Jacobian combines many geometrical measurements and has been shown to have the best correlation with mesh quality.

(This is introduced in chapter 4 section 4.4.1 and further discussed in chapter 6 sections 6.1 and 6.3.3.)

The elemental Jacobian ratio is defined as [87] :

$$l = \frac{\min(J_i^{3D})}{\max(J_i^{3D})} \quad \forall i \in Q^e$$

Where  $Q^e$  are the sampling points within the element and  $J^{3D}$  are the determinants of the volume Jacobian of the local to global mapping. This ratio describes the geometrical homogeneity of the element.

#### 5.4.2 Experimental data

The synthetic tests will use the following input data.

**Eight and twenty-seven element cube:** These are meshes representing cubes, composed of eight and twenty-seven hexahedral elements. The mesh is aligned with the Cartesian axes but the internal node is offset from the regular grid (eight elements) or the internal element is rotated (twenty-seven elements). In the tests involving these meshes, the fixed image is a cube of larger proportions.

**Idealised ventricle:** This mesh represents an idealised ventricle and is registered to the image of an idealised ventricle.

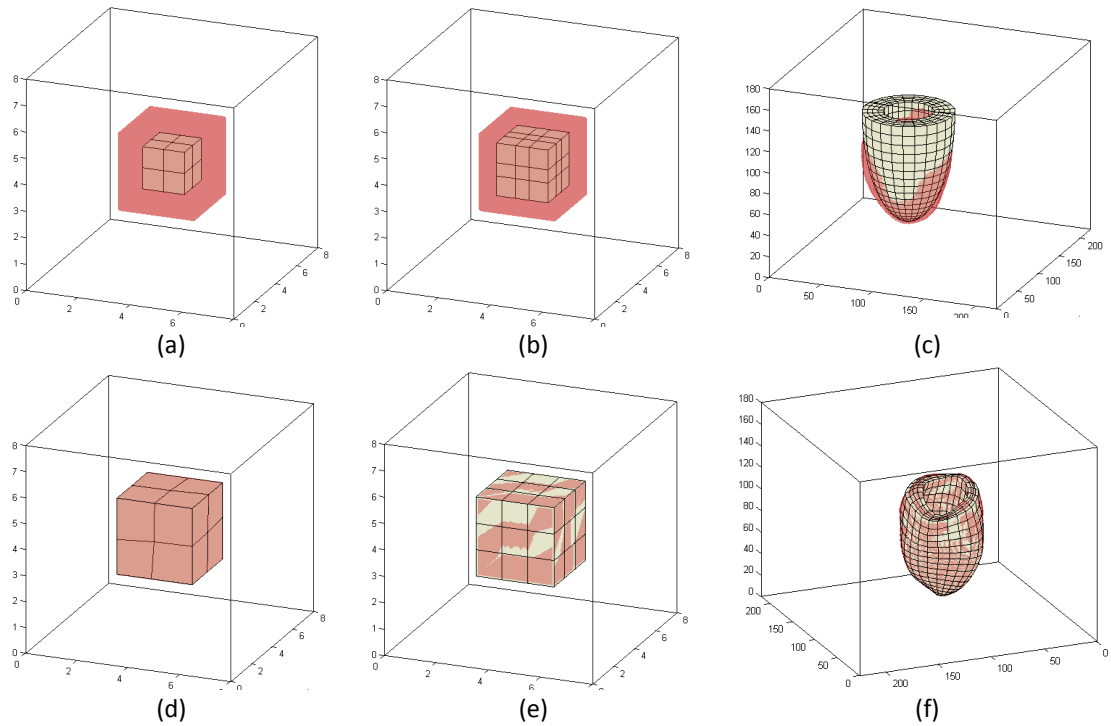


Figure 57: The setup of the synthetic tests. Panels (a) to (c) show the initial mesh and image set up and panels (d) to (f) show the result after registration. The axis denotes the extent of the image, which is shown as a red surface.

Figure 57 shows the setup of the synthetic tests. The axis represents the image boundaries and its isosurface plotted red, before the registration (a-c) and after the deformation (d-f). The first column shows the 8 element cube, the second column the 27 element cube and the third column the idealised ventricle. The reasons for using the simplified geometry are that they require less computational time to be performed and that they have a known solution. It must be remarked that the trivial solution for image registration is not all that trivial. For instance, take the case of the rotated cube shown in Figure 58 (a); a desired solution might be the one presented in the bottom-left panel of the figure above, that is the mesh aligned with the imaged cube object. Depending on the choice of the regularisation term, another registration solution [Figure 58 (b)] may effectively pull the mesh surfaces to the surface of the image taking the shortest possible route. The solution changes if the images are smoothed.

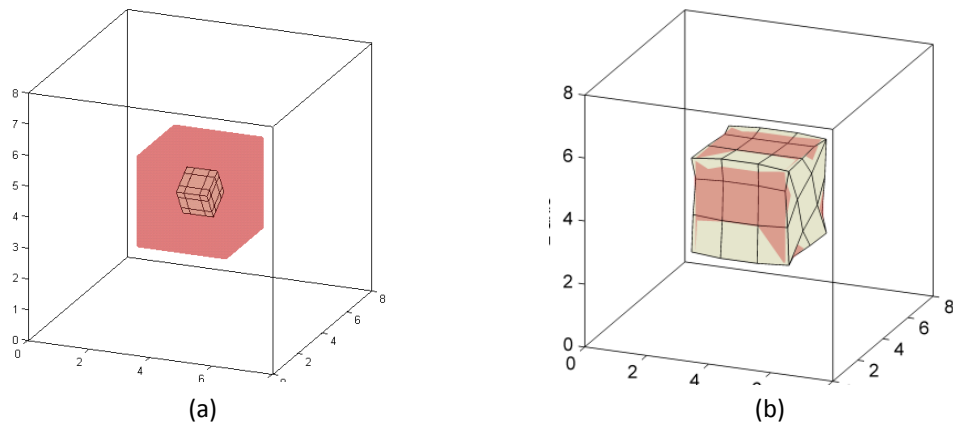


Figure 58: The solution to registering a rotated cube to a cube aligned with the Cartesian grid is driven by the choice of the regularisation term. Panel (a) shows the initial set up. The image is shown as a red surface and the axis shows the extent of the image. Panel (b) shows the result after registration, which might not be the desired outcome to the process.

### 5.4.3 Experiment: Number of sampling points

This experiment tests the effect of changing the number of sampling points used. Sampling points are the number of voxels a mesh has and is also the points at which the registration is calculated. The number of sampling points determines where the intensity is sampled, and therefore might produce a better fit, but it doesn't intrinsically enrich the description of the registration field. The number of sampling points is a variable in the registration method and directly affects the time needed to construct the registration equations; it is important to understand the effect on the registration.

In this experiment, the 8 element mesh is registered to a cube of larger proportions, with increasing number of sampling points. The results are summarised in Table 8 and Figure 59.



Number of sampling points in each element	Mean surface distance after registration (mm)	Mean surface distance (as a percentage of the voxel size of the static image)	Mean iterations for optimisation routine (as percentage of degrees of freedom)	Image similarity
2 <sup>3</sup>	0.040	32%	41.3%	0.000
4 <sup>3</sup>	0.057	45%	32.7%	0.001
8 <sup>3</sup>	0.023	19%	20.7%	0.000
12 <sup>3</sup>	0.021	17%	22.6%	0.000
18 <sup>3</sup>	0.001	1%	27.7%	0.007
24 <sup>3</sup>	0.010	8%	25.3%	0.006
32 <sup>3</sup>	0.020	16%	24.3%	0.005
40 <sup>3</sup>	0.028	23%	21.4%	0.004
48 <sup>3</sup>	0.036	28%	21.8%	0.003

Table 8: Summary of the assessment of the effect of changing the number of sampling points used in the registration process

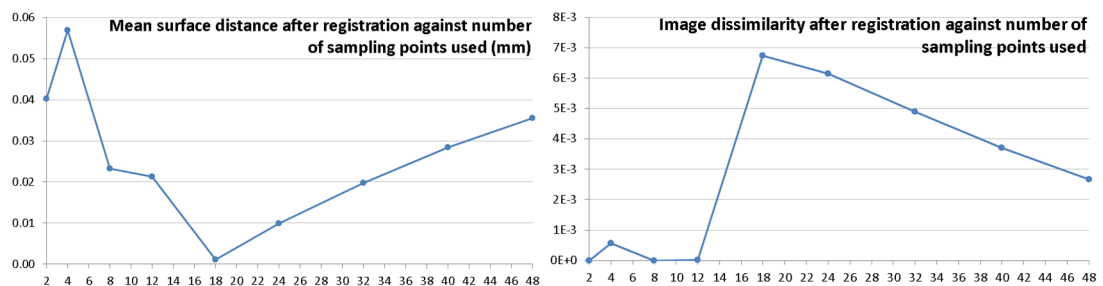


Figure 59: The mean surface distance (left) and image similarity (right) after registration as a function of sampling points used. This figure was created using the results presented in Table 8

It is clear from Figure 59 that both figures have a turning point around the 18<sup>3</sup> sampling point. This coincides with the point when the spacing between the sampling points in the final mesh turn from being spaced closer than the voxel spacing (0.125 mm) to being further. For reference, for the 12<sup>3</sup> sampling points case, the points are expected to be around 0.17 mm apart and for the 18<sup>3</sup>, 0.12 mm. As one would expect, after this point, as the sampling distance is decreased the image similarity increases [as shown by the downwards trend in Figure 59 (r)]. Interestingly, the distance between the surfaces increases with finer sampling, which can only be explained by the fact that image similarity takes all points in the images into account and the surface only measures the image edges; the more sampling points used, the less important they become in measuring image similarity. This is interesting to note because whilst it is important to verify the registration is behaving as expected, ultimately, the surface distance measurement is a more valuable metric to assess the fit of a mesh template to an anatomical structure.

Regardless of the metric used, all results are shorter than the voxel size (0.125 mm). An important practical outcome from this experiment is that the number of iterations required to

solve the equations can be significantly reduced by using more sampling points. This suggests that this variable should be changed in situations where the optimiser fails to converge.

#### 5.4.4 Experiment: Laplacian regularisation

Common Tikhonov regularisers include the identity matrix and the Laplacian matrix. In this experiment, the effect of each of these regularisers is compared by registering the eight and twenty-seven element cubes to a larger cube. It is important to understand the situations in which each of the regularisers are more useful.

8 element mesh			27 element mesh		
$\lambda$	Regulariser used	Mean surface distance after registration (mm)	$\lambda$	Regulariser used	Mean surface distance after registration (mm)
0.1	Identity	[Unstable]	0.1	Identity	[Unstable]
0.1	Laplacian	0.029	0.1	Laplacian	[Unstable]
1	Identity	0.034	1	Identity	[Unstable]
1	Laplacian	0.016	1	Laplacian	0.038
10	Identity	0.023	10	Identity	[Unstable]
10	Laplacian	0.023	10	Laplacian	0.005
100	Identity	0.020	100	Identity	0.027
100	Laplacian	0.018	100	Laplacian	0.008

Table 9: Summary of the assessment of the effect of using the identity and the Laplacian matrices as the Tikhonov matrices for image registration

The results for registration using the identity and Laplacian matrices are summarised in Table 9. In this test the memory term (see Chapter 4, section 5) is not invoked and each increment of the registration is independently regularised. In all cases the mean surface to mesh distance is less than one voxel linear dimension (the voxel size is 0.125 mm). The tests labelled "Unstable" in Table 9 failed to either solve the registration equations or did not converge towards an optimal solution (i.e. the edges of the mesh and image did not align). The most important aspect to notice is that the stability of the registration process increases by using the Laplacian matrix. Furthermore, although not explored in this test, the Laplacian provides a smooth solution for nodes where there is no information from the images (image gradients). This means that in practice, for binary images, the Laplacian matrix provides much more stability to the registration than the identity matrix.

### 5.4.5 Experiment: Laplacian memory

In this experiment, the eight and twenty-seven element meshes are registered using the Laplacian memory regularisation term (Chapter 4, section 4.3.2). The Laplacian memory term is the default regulariser used in SHIRT and provides increased stability to the registration as well as less grid degradation.

It is important to understand this variable because it is the naïve way to improve the quality of the mesh after registration. This method computes quickly and is a trivial extension from the Laplacian matrix, however it assumes the desired solution can be achieved using a smooth displacement field. Additionally, regularising successive iterations by invocation of the memory term has the side effect of decelerating the nodal displacements.

<b>8 element mesh</b>				
$\lambda$	<b>Laplacian with memory</b>		<b>Laplacian</b>	
	<b>Mean surface distance after registration (mm)</b>	<b>Mean element quality after registration</b>	<b>Mean surface distance after registration (mm)</b>	<b>Mean element quality after registration</b>
0.1	0.008	0.700	0.029	0.659
1	0.008	0.700	0.016	0.725
10	0.012	0.699	0.023	0.703
100	0.086	0.687	0.018	0.700
<b>27 element mesh</b>				
$\lambda$	<b>Laplacian with memory</b>		<b>Laplacian</b>	
	<b>Mean surface distance after registration (mm)</b>	<b>Mean element quality after registration</b>	<b>Mean surface distance after registration (mm)</b>	<b>Mean element quality after registration</b>
0.1	Unstable	Unstable	Unstable	Unstable
1	0.008	0.658	0.038	-0.421
10	0.008	0.667	0.005	0.615
100	0.014	0.670	0.007	0.660

Table 10: Summary of the assessment of using the Laplacian memory to regularise the registration, compared to using the Laplacian matrix with no memory

The results for this experiment are summarised in Table 10; no metrics of variability are included because they are not significant for the interpretation of the results. Increasing the value of  $\lambda$  for the memory term further decelerates the displacements. As a result, increasing the value of  $\lambda$  increases the surface distance after registration. The surface distance is also increased by keeping the value of  $\lambda$  the same and using the memory term. There is an exception caused by the fact that a  $\lambda$  value of 1.0 is the optimal value for tests using the memory term (only known after registration). Using a value of 1.0 and no memory term is not a stable process, giving a poor result in terms of quality, as seen in Figure 60.

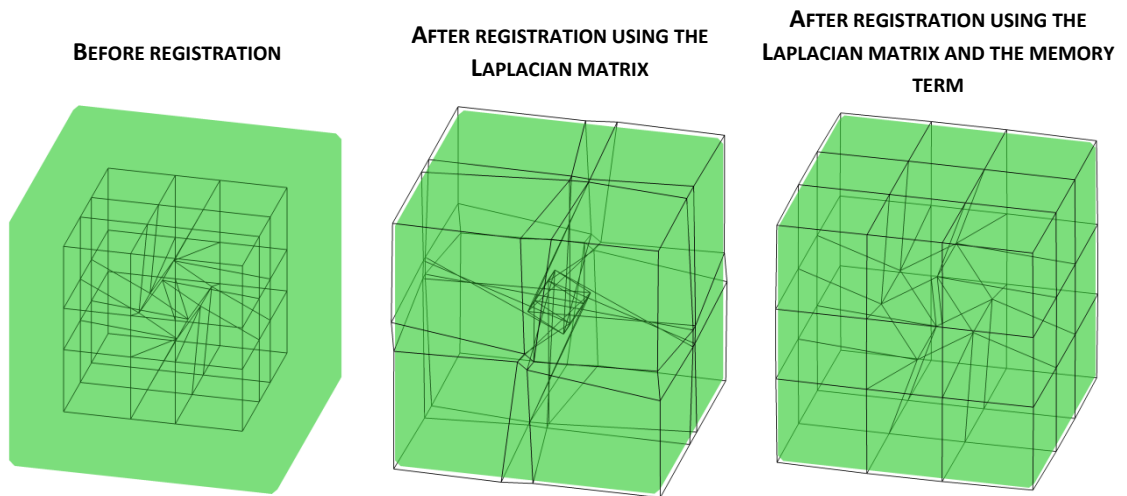


Figure 60: Resulting meshes when registering a 9 element cube to the segmentation of a cube with and without the memory term, using a  $\lambda$  value of 1.0.

As expected, the memory term slows down the nodal displacement but also has the effect of stopping the elements from collapsing. In the case of the twenty-seven element mesh, the centre nodes do not have image information (the gradients are far away) so providing a smooth solution at these locations is crucial for stabilising the registration and the registration grid.

#### 5.4.6 Experiment: Scalar product regularisation

This experiment tests the effectiveness of the scalar product regulariser (chapter 4, section 4.4.2) on improving the quality of a distorted mesh. In this experiment, the eight and twenty seven element meshes will be registered, with a varying influence of the scalar product regulariser.

This regularisation scheme attempts to improve the orthogonality of the sides of the elements but this not exactly the same as improving the quality of the mesh. For this reason, it is important to understand the effect the regulariser has on the mesh. As discussed in chapter 4, section 4.6, the scalar product regulariser is used in conjunction with the Laplacian regulariser. In this test, the relative weight between the scalar product and Laplacian regulariser is kept the same.

The results of the tests are shown in Table 11;  $\lambda_1$  is the weight of the regularisation matrices to the intensity matrix. The quality of the mesh is improved in each of the cases, when compared with using the identity matrix, the Laplacian matrix and the Laplacian matrix with the memory term. In all but one of the tests, the inside element is partially or fully rotated back into alignment with the Cartesian grid, and the mean surface distance after registration was less than the size

of a voxel (0.125 mm). The variability of metrics is not quoted because they do not help interpret the results.

<b>8 element mesh</b>			
$\lambda_1$	Mean surface distance after registration (mm)	Mean element quality after registration	Number of bad quality elements
1	0.072	1.000	0
10	0.015	0.999	0
100	0.012	0.876	0
<b>27 element mesh</b>			
$\lambda_1$	Mean surface distance after registration (mm)	Mean element quality after registration	Number of bad quality elements
1	0.161	2.360	18
10	0.029	0.998	0
100	0.026	0.811	0

Table 11: Summary of the assessment of using the scalar product regularisation

The twenty-seven element mesh registered using a value for  $\lambda$  of 1.0 did not produce a mesh of equally sized elements, as shown in Figure 61. The nodes of the inside elements collapsed. This is not an unexpected result as this satisfies the condition of orthogonality. This is, however, an undesired effect produced by inappropriately scaling the regulariser.

Figure 62 shows the evolution of the mesh element quality over successive registration iterations. The blue line shows the mean mesh quality and the red bars the standard deviation from the mean. Using the Laplacian matrix controls the curvature of the displacement field but does not directly affect the mesh quality, therefore the deviation remains unchanged. The scalar product regulariser improves the orthogonality of all the node junctions in the mesh and this can be seen in the plot (right-most plot). When the regularisation is inappropriately scaled, the scheme successfully aligns the centre element, but fails to stop the nodes from collapsing. This can be seen as an improvement in the mesh quality followed by quality degradation.

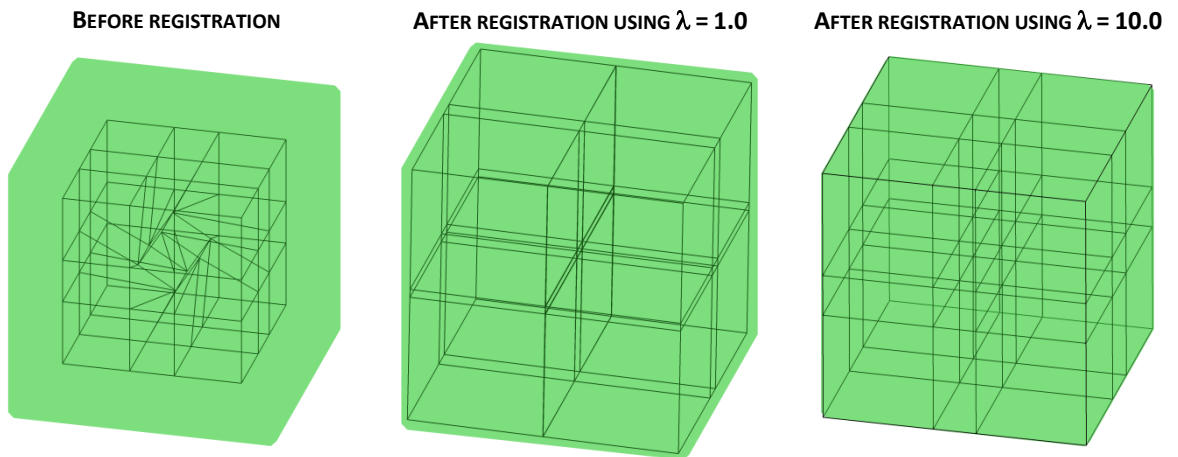


Figure 61: Resulting meshes when registering a 9 element cube to the segmentation of a cube using the scalar product regularisation and different values for  $\lambda$ .

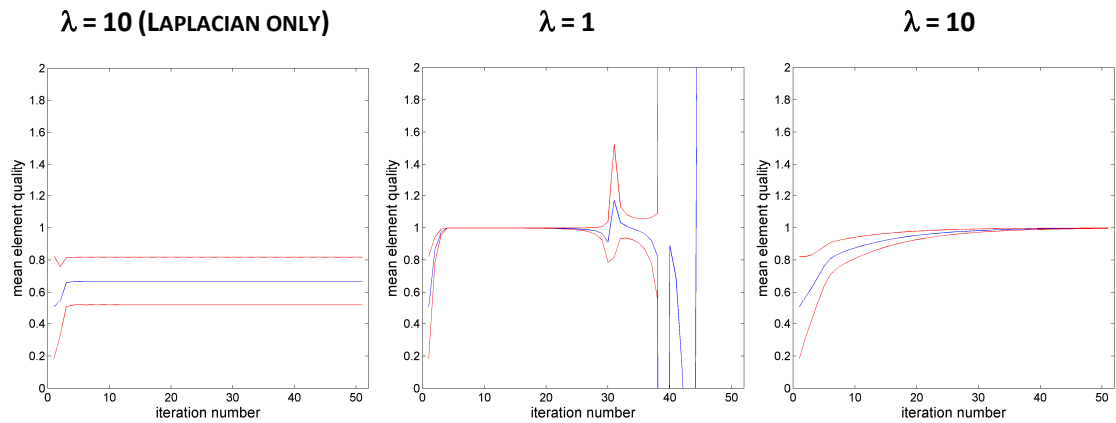


Figure 62: Evolution of the quality of the mesh, with appropriate and inappropriate scaling of the scalar product regularisation

The scalar product regulariser is able to improve the orthogonality of the mesh however it has the side-effect of collapsing the mesh nodes if not smoothed using the Laplacian matrix. As a result, it is quite a complicated regulariser to control. Furthermore, the orthogonality of the mesh is not the same as the quality of the mesh, so this method does not guarantee a mesh of good quality for computer simulations.

#### 5.4.7 Experiment: Jacobian regularisation

In this experiment, the 27 element cube with the distorted centre element is registered using the Jacobian ratio regulariser and the Laplacian regulariser (chapter 4, sections 4.4.3 and 4.6).

Table 12 shows the results changing the weights of the regularisers;  $\lambda_1$  being the weight of regularisers to image similarity and  $\lambda_2$  being the weight of the Laplacian to Jacobian regulariser.

The variability of metrics is not quoted because they do not help interpret the results. The right-

most column of Figure 63 shows the Jacobian ratio (mean in blue, standard deviation in red) as the iterations progress.

$\lambda_1$	$\lambda_2$	Mean surface distance after registration (mm)	Mean element quality after registration
0.1	100	Unstable	Unstable
<b>0.5</b>	<b>100</b>	<b>0.0003</b>	<b>0.999</b>
5	100	0.0146	0.786
0.5	10	0.0004	0.998
0.5	1000	0.0139	0.787

Table 12: Summary of registration using the Jacobian regulariser.  $\lambda_1$  is the weight of regularisers to image similarity and  $\lambda_2$  is the weight of the Laplacian to Jacobian regulariser.

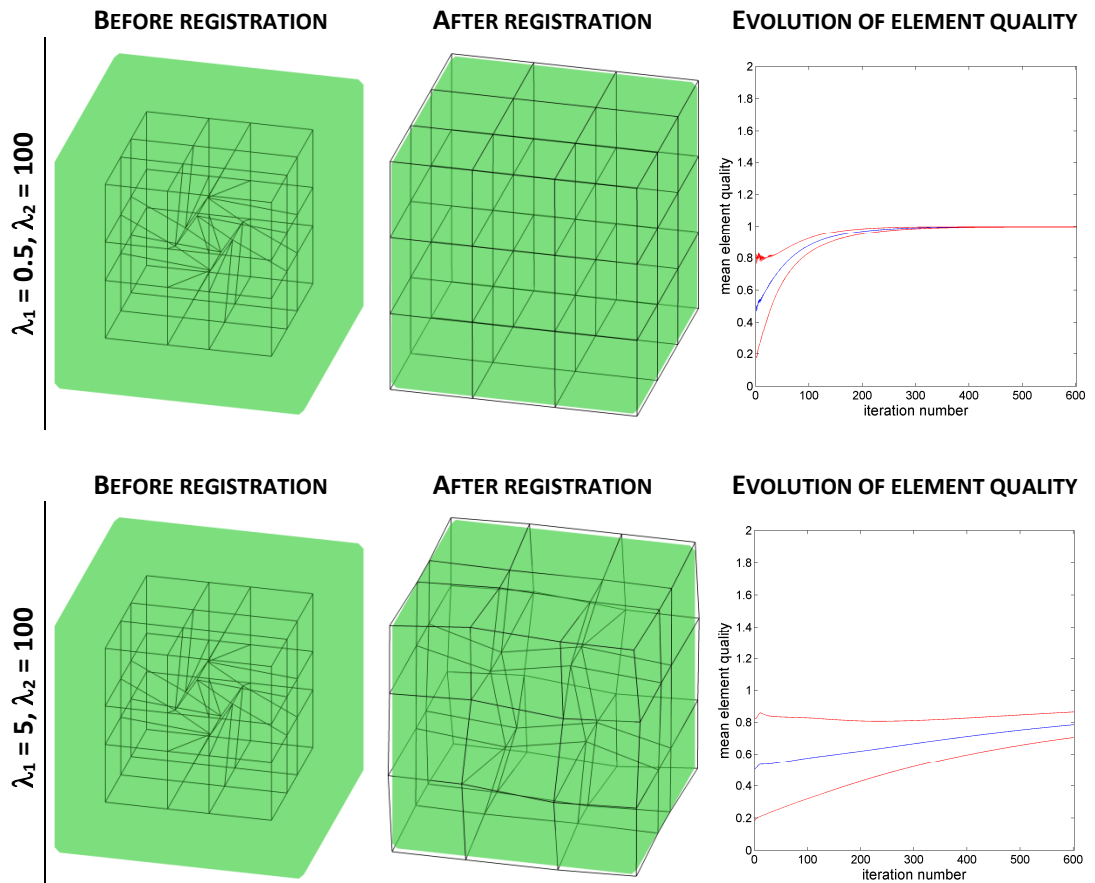


Figure 63: Resulting meshes and the evolution of element quality when registering a 9 element cube to the segmentation of a cube using the Jacobian regularisation and different values for  $\lambda$ .

It is clear that the Jacobian ratio is effective at regularising the registration equations and improving the mean element quality. This regularisation does not suffer the disadvantage of the

elements collapsing to a point, but it's hard to scale. The effect of inappropriate scaling is the registration becoming unstable.

#### 5.4.8 Experiment: Strain regularisation

In this experiment, the cubes are registered using the strain regulariser with a variable  $\lambda$  (see chapter 4, section 4.7). To produce these results, the contribution of the Laplacian and strain regulariser was kept constant. Table 13 shows the mean surface distance after registration using different values for  $\lambda$ .

8 ELEMENT MESH		27 ELEMENT MESH	
$\lambda_1$	Mean surface distance after registration (mm)	$\lambda_1$	Mean surface distance after registration (mm)
0.1	0.035	0.1	0.062
1	0.012	1	0.043
10	0.029	10	0.013
100	0.431	100	0.038

Table 13: Summary of the results using the strain regulariser

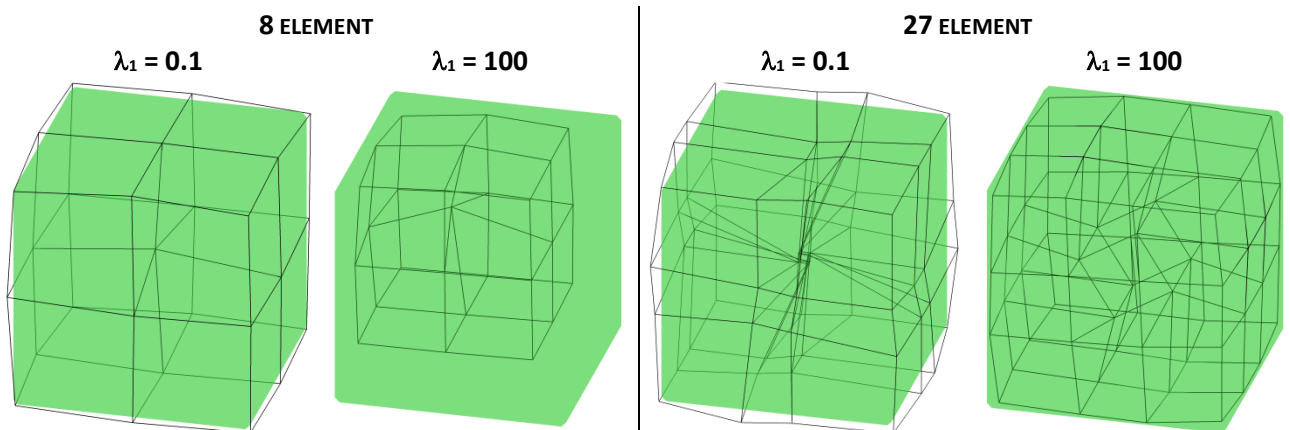


Figure 64: Resulting meshes when registering a 9 element cube to the segmentation of a cube using the strain regularisation and different values for  $\lambda$ .

It is found that the strain regulariser is a useful condition to use to regularise the registration equations. As expected, this regularisation can have a negative effect if set too high by preventing volume change or too low by producing unsmoothed results.



### 5.4.9 Experiment: Number of padding elements and trimmed base

In this experiment, an idealised ventricular mesh is registered to an idealised ventricular geometry. The purpose of these tests is to examine the effect of the number of padding layers (see section 5.2.2 of this chapter) used in the registration. Using more layers increases the number of image voxels that get included in the registration (which is beneficial when the images are not well aligned) but comes at a cost of increased number of degrees of freedom.

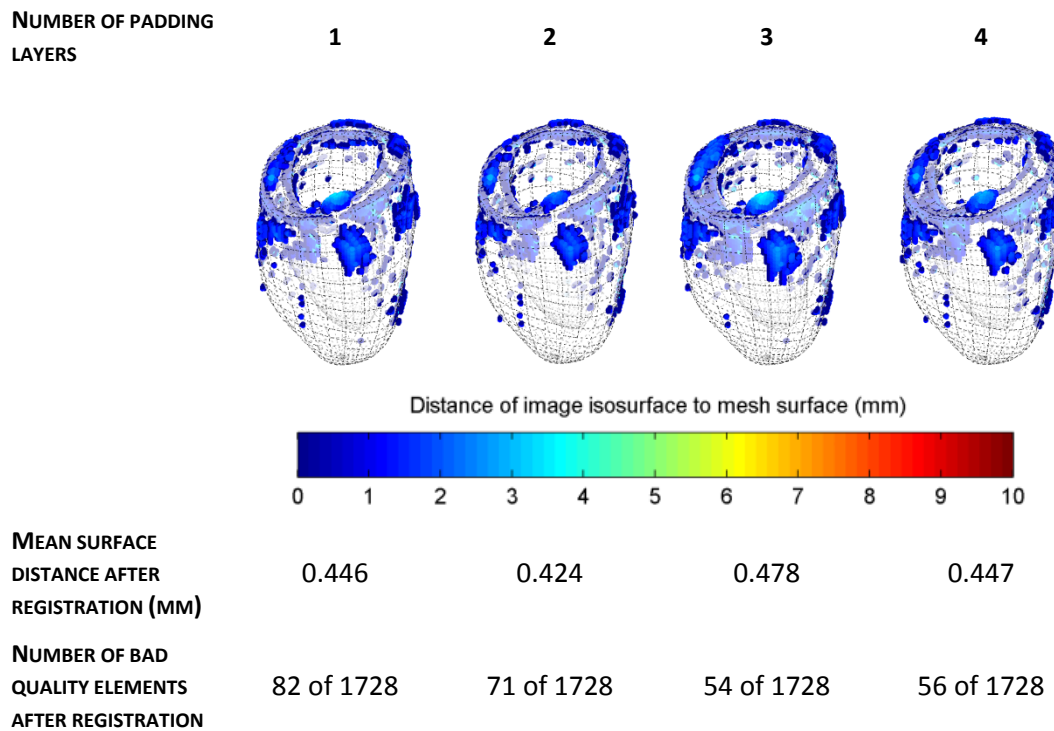


Figure 65: Resulting mesh and fitting distance when registering the ventricular mesh to the idealised geometry using a variable number of padding elements.

Figure 65 shows the results of this experiment. The 3D renders show the points in the image isosurface that are more than one voxel away from the surface of the mesh after registration. For reference, the voxel size of the image is 1.2794 by 1.2794 by 0.88235 mm.

This experiment shows that the number of padding elements does not significantly affect the accuracy of the registration, for all the resulting surface distances are within sub-pixel accuracy. The downside of using a larger number of padding layers is two-fold, in that it increases the number of degrees of freedom, and that it increases the effect of the Laplacian regulariser on the system. This second effect is explained by the fact that, as the images become increasingly similar, the padding nodes only contribute to optimising the regularising metric.

In the above experiment, it can be seen that the highest error in the accuracy appears at the base of the ventricle. At this location, the shape of the idealised mesh differs from the target, and the thickness of the target narrows. This has an effect on the accuracy results presented which negatively biases what might be (in some situations) a desired solution: to not include the narrowing at the base of the ventricle in the personalised mesh used for the simulations.

For this reason, the effect of trimming the base of the ventricle is explored. Registering the idealised mesh to the un-altered ventricular geometry produces a surface distance of  $0.64251\text{mm} \pm 0.70436\text{ mm}$ , compared to  $0.50965\text{ mm} \pm 0.49247\text{ mm}$  when trimmed. The mean surface distance is not greatly reduced; however the deviation from the mean is greatly reduced. This is more obviously seen in the distance plots shown in Figure 66.

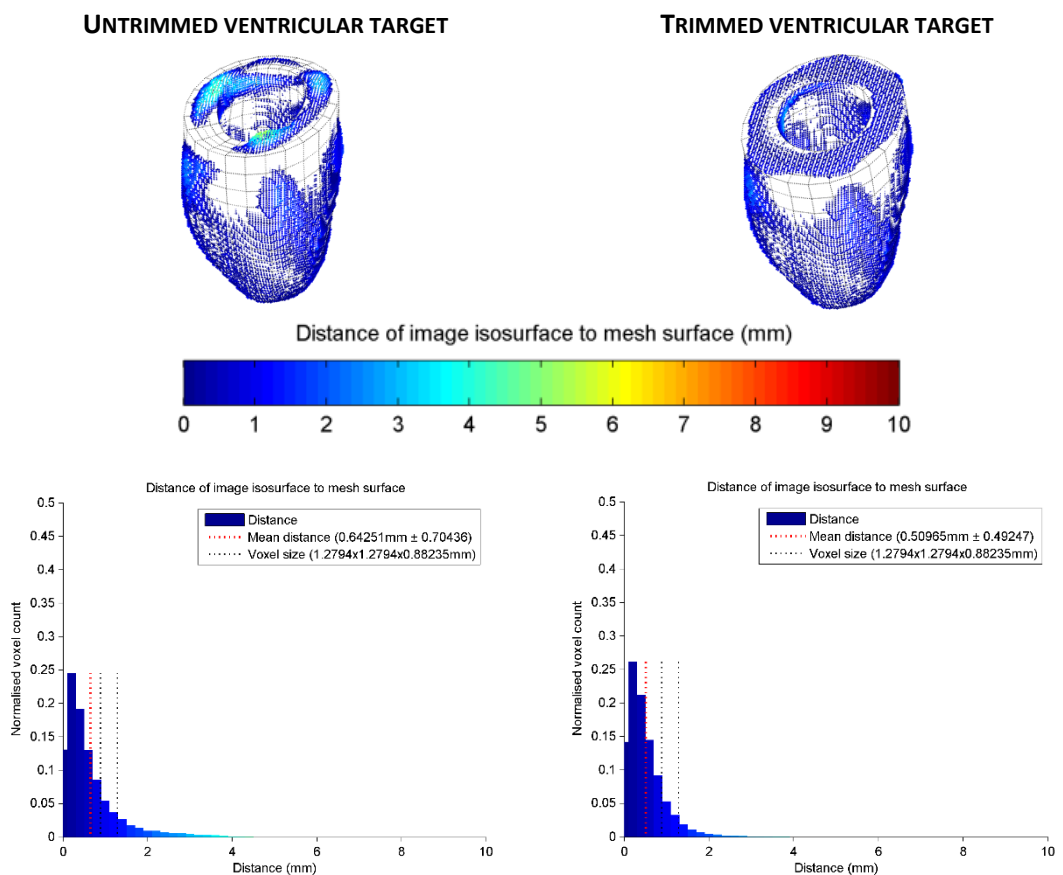


Figure 66: Image of the idealised ventricle and the registered mesh. The second row shows a histogram of the distance of isosurface points from the idealised mesh. All plots are colour-coded by distance.

## 5.5 Conclusions

This chapter has described the implementation and rationale behind some decisions made when implementing the registration method into a software solution.

It has also shown the tests that were performed to ensure the software worked as expected. The tests were performed on simple shapes to gain a deeper understanding on the behaviour of the registration parameters.

The results achieved in this chapter demonstrate the software provides accurate results to sub-pixel accuracy. Whilst using image registration as a tool to create finite element meshes by deforming a template mesh into a patient geometry is a technique described already in the literature; this chapter shows the basic performance of a novel approach to producing a high quality finite element mesh, namely the implementation of mesh quality improving optimisers in the registration process.

The Laplacian is fast to compute and effective at regularising the registration; using the memory term is helpful to preserve the integrity of the mesh elements specially when using binary images. The scalar product and Jacobian regularisers directly improve the quality of the grid but are hard to scale appropriately. Whilst the scalar product regulariser is less computational expensive to compute than the Jacobian regulariser, it is a weaker predictor of mesh quality.

An important aspect in image registration is that it is generally ill-posed (chapter 4, section 4.1) and the fact that the scalar product and Jacobian regularisation terms are singular does not help matters. Concatenating these regularisation terms with the Laplacian matrix, along with appropriate scaling, helps make the regularisation term non-singular and in turn helps to find a solution to the registration process.

To the effect of comparing the method described in this thesis and other methods based on deformable models (see chapter 4, section 4.7), the strain regulariser was tested in section 5.4.8. The resulting mean fitting distances found in the experiment were in the same range as other regularisation options, however the regulariser does not contribute towards improving the quality of the mesh and is highly sensitive to the values of  $\lambda$ .

This chapter has further explored the effect of other variables in the registration and has concluded that the dominant parameters in the registration are the scaling factors of the regularisation terms ( $\lambda$ ).

The execution of these tests has developed the understanding necessary to assemble an appropriate process for the creation of patient-specific cardiac meshes, by deforming an idealised template mesh. Specific choices for the regularisation terms and for the mesh structure (including padding elements and trimming) are discussed in the next chapter.

# CHAPTER 6

## PERSONALISED FINITE-ELEMENT MESHES OF THE LEFT VENTRICLE

### 6.1 Introduction

In previous chapters, an image registration tool has been introduced and its performance has been assessed using simple geometries. In this chapter, the complexity of the problem is elevated by presenting a more complex target image and a bigger mesh. Meshes of an idealised ventricular structure will be morphed to real patient geometries. The practical implication of this application is that no longer is the correspondence between image and mesh trivial.

An additional complication is the uneven nature of the target image, when the mesh is required to be smooth and with good quality elements. Mesh quality, surface smoothing and geometry fitting can rarely be maximised together, there is a trade-off that must be accepted. Typical methods for improving mesh quality operate at the post-processing stage, working on fixed surfaces [96]. In this work, a balance, determined by the Tikhonov multipliers, between mesh quality and mesh-to-image fit is achieved during the registration process.

The remainder of this introduction, section 6.1.1, focuses on background work leading this study. Section 6.2 describes image acquisition and the methods available for segmentation. The preparation of the segmentations for image registration is described in section 6.3. Section 6.4 completes the description of methods by discussing the choice of registration parameters. The results are presented and discussed in section 6.5. Section 6.6 presents the conclusions of this work set in the larger context of image registration and mesh generation tools.

#### 6.1.1 Review of approaches to mesh construction

##### 6.1.1.1 Generating volumetric meshes

The starting point for the generation of a general volumetric mesh is usually the definition of surfaces that form a closed volume. Tetrahedral elements are commonly created using a combination of controlled Voronoi triangularisation and convex Hull operations, whilst hexahedral elements are commonly created using the marching cubes algorithm. There is no published method for generating cubic hexahedral elements from an enclosed surface. Furthermore, these methods have been developed in manufacturing or engineering environments where the structures to be modelled are usually of known shape (by design), or

can easily be measured, rather than approximated from invariably noisy images. This situation is not common in the biomedical environment, where even the imaging techniques of the highest quality require expert interpretation. For this reason, defining the closed surfaces for modelling the complex geometries of the human body can be a laborious, time consuming task. One can remove the complications of generating complex meshes by performing simulations using an idealised geometry. Idealised geometries have the advantage that they are constructed from labelled components: that is, each section of the mesh corresponds to a known physical reality. This is useful for assigning the mesh properties at the simulation stage. On the other hand, idealised geometries are not patient-specific.

An alternative approach to generating volumetric meshes is deforming an idealised geometry to comply with a particular geometry (the patient geometry). Lamata et al. [87] describe the current state-of-art in mesh generation from an idealised geometry. Other mesh generation methods found in the literature include Sermesant et al. [119], however the novelty introduced by Lamata et al. is in the operation on elements of a higher order ([119] use linear tetrahedral elements).

Another published method for generating cubic Hermite cardiac meshes manipulates a manually defined, linear hexahedral mesh to create the cubic Hermite elements [120]. Zhang et al. [121] created a pipeline to create cubic Hermite meshes, but their method does not apply for general geometries.

Other methods reported in the literature involve creating a personalised surface mesh and then creating a volumetric mesh from it. The surface mesh can be generated directly using image processing techniques [122].

#### 6.1.1.2 Mesh fitting using binarised meshes

An optimal mesh deformation can be found using image registration techniques. The advantage of using image registration, as opposed to node/surface projections, is that it is generally a more robust process due to the comparatively information-rich data employed. Jones et al. [85] describe the process of using this method to generate personalised models of the aortic arch. Barber et al. [21] describe the generalisation of the process to other structures of the human body.

To determine the deformation field, both fixed and moved image need to contain similar representations of the structure. The method used by Jones et al. operates on binary images of the patient anatomy (fixed image) and of the model (moved image). The segmentation of the patient image might be achieved using manual or automatic methods, and the result is a binary

image, or binary mask, of the structure of interest (in this case the aorta). An idealised aortic geometry is created, including three branching supra-aortic vessels. Using a grid of equal size to the segmented patient image, the mesh of the idealised model is binarised by assigning a value of unity to voxels that lie inside the mesh elements. Figure 68 shows this process. Barber et al. use the medical image as the target image. To improve the registration process, the moving image is made to match the intensity distribution of the patient image.

With the inputs in a similar state, the binary images are registered using a non-linear registration algorithm (ShIRT). This produces a registration field described by the displacements at registration nodes in the Cartesian grid. Displacements at all other points in the image are determined by trilinear interpolation. Thus the required displacements at the nodes of the model are resolved by interpolating the displacement at the registration nodes within the displacement vector field.

This method suffers from the same drawback as with all atlas-based methods, when severe variations between the idealised and target geometry cannot be accounted for. In this case of the aorta, the location of the supra-aortic vessels is extremely variable across subjects. The topology of patient anatomy is not consistent. Although the most common anatomy of the supra-aortic branches is the three-branch structure used by Jones, many patients have alternative topologies. The practical implication is that the idealised geometry is not a valid representation of the patient anatomy. This weakness of atlas-based methods is typically resolved by taking a multi-atlas approach. In this case, that would involve using several idealised geometries, with a varying supra-aortic vessel location and using a criterion to select the best mesh for each subject.

An alternative approach presented by Barber and Hose [21] involves registering an atlas intensity image to a patient intensity image. If the atlas image is segmented the relevant anatomical structure can be morphed using the registration field. The atlas can be chosen to represent a particular cohort of patients or be synthetically produced by averaging images after registration. The segmentation is only used to build the mesh for the atlas. The atlas image can be morphed to the patient image and the mesh deformed using the deformation field.

This has several advantages, including the fact that the registration can be driven partly by adjacent anatomical structures that do not feature in the model but that provide important landmarks that assist the determination of the optimal registration field, and that the segmented object is often a closer representation of patient anatomy than an idealised model. The drawback is that it is usually more difficult to produce a good quality mesh on the segmented atlas anatomy than on an idealised structure.

Further problems exist due to the fact that the visibility and characteristics of the supra-aortic vessels, as they appear in the medical image, is also highly variable.

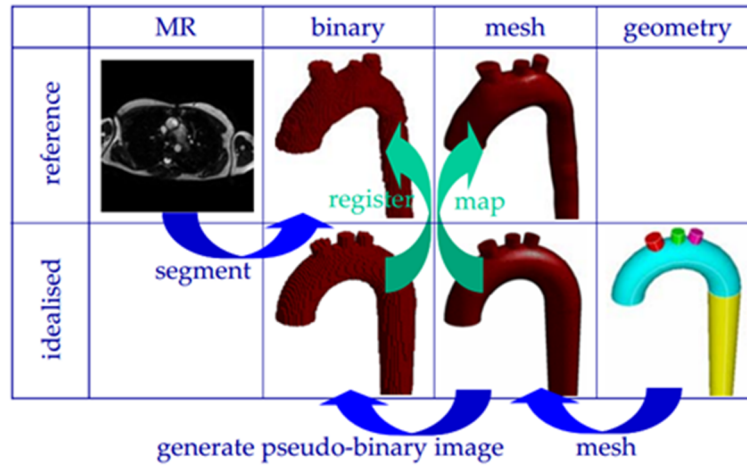


Figure 67: Schematic of the method described by Jones et al. The row labelled 'reference' corresponds to the patient geometry. A volumetric MR of the patient's thorax was manually segmented. An idealised aortic geometrical mesh was built and converted to a binary volumetric image, which was registered to the segmentation. The map resulting from the registration process was then applied on the idealised mesh to produce a patient-specific mesh. (reproduced with permission from [85])

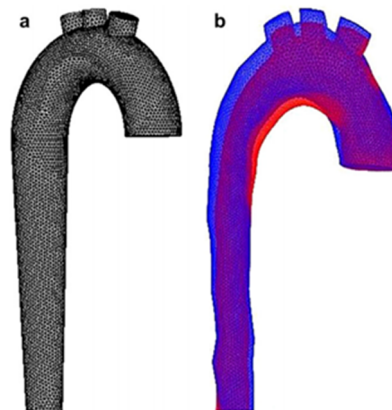


Figure 68: Input and processed data using the method described by Barber et al. (reproduced with permission from [86])

In their paper, Lamata et al. [87] used an idealised mesh and a binary segmentation of the patient image to fit cubic Hermite meshes to patient images. The segmentation is analysed to locate the valve openings and the principal axis is used to determine the base-to-apex direction; this information is used to create an idealised volumetric mesh. A deformation field is used to morph the mesh to the patient geometry at the nodes, then a post-processing stage deals with the Hermite variables and mesh quality. The binarisation process is the same as previously

described and the registration uses SHIRT. This produces a general deformation field which requires mapping to the Hermite parameters.

A variational method is used to deform the mesh. The variational method is an extension of a surface fitting algorithm for cubic Hermite meshes, developed by Professor Peter Hunter's group in Auckland [123], which minimises the distances between two surfaces by optimising the Hermite parameters. The variational method requires manual tuning of a number of geometrical factors affecting face curvature and aspect ratio. It also ensures no element folding is introduced into the mesh. The original motivation for this work was that it has been shown that the Hermite elements offer strong advantages in stability when applied to the nonlinear analysis of ventricular structures. However it is noted that, once a fitted cubic Hermite mesh is generated, its elements can be broken down into linear hexahedral elements or tetrahedral elements. Note the latter option is not explored in the paper. Figure 69 shows this process.

The general idea behind this method has been shown to work to generate different types of geometries, such as vessels [86], the aortic arch [85] and ventricular structures [91]. In addition to the benefits of using a deformed idealised model, the strength of this method relies on employing a fast and robust registration method to produce the displacement field. Even if the target is to generate lower order elements, going through the step of generation of cubic Hermite elements imposes a smoothness on the mesh that also contributes towards the robust generation of high quality elements.

The weakness of the approach used by Lamata et al. is that the registration is performed on a regular spatial grid (natural for the representation of an image, but not of a model) and thus requires the post-processing operation (variational method) to produce the Hermite meshes. The post-processing leads to a loss of correspondence between mesh and geometry, as acknowledged by the authors [87]. Also, mesh quality is optimised independent from the fitting. The process used in this thesis removes these limitations.

The remainder of this chapter will present the mesh generation pipeline. The pipeline includes acquiring the medical images, generating a segmentation, processing the segmentation, creating a template mesh and fitting a template mesh [124], [125].



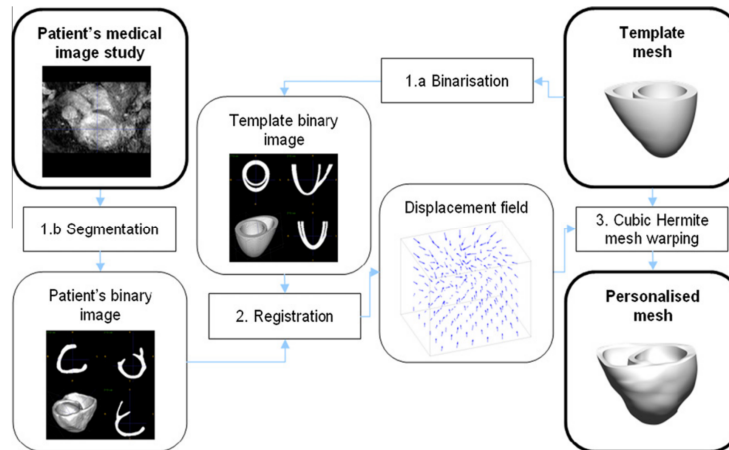


Figure 69: Flowchart of the process of cubic Hermite mesh generation described in Lamata et al. To personalise the template mesh, this is binarised and registered to the segmentation of the patient's myocardium. The resulting displacement field is applied on the nodes of the mesh, and the faces of the cubic Hermite elements are warped to the closest surface in the patient's segmentation to produce the personalised mesh. (reproduced with permission from [87])

## 6.2 Image acquisition and segmentation

### 6.2.1 Image acquisition

Cardiac images were acquired on a cohort of patients diagnosed with advanced heart failure (corresponding to NYHA classes III and IV). The images used a steady state free precession (SSFP) protocol capturing the full chest area of the patient and employed a contrast agent to enhance the myocardial muscle. Two clinical centres participated in the acquisition process, each belonging to separate Foundation Trusts of the National Health Service (NHS FT): the Northern General Hospital (Sheffield Teaching Hospitals NHS FT, Sheffield) and Guy's Hospital (Guy's and St. Thomas NHS FT, London).

Each centre employed the equipment used during routine clinical practice in that centre. Sheffield used a Siemens Avanto 1.5T whilst Guy's used a Philips Achieva 1.5T. Each manufacturer has developed their own implementation of the SSFP protocol. Siemens use a technique called Inversion Recovery True Fast Imaging with Steady-state Precision and Philips use a technique called Turbo-Field Echo-Planar Imaging. The details of the implementations are not published. Table 14 summarises the differences between the cMRI (cardiac Magnetic resonance Imaging) equipment used for the image acquisition.

<b>EQUIPMENT INFORMATION</b>	<b>SHEFFIELD</b>	<b>LONDON</b>
Location	Northern General Hospital (NGH)	Guy's Hospital
Manufacturer	Siemens	Philips Medical Systems
Model name	Avanto	Achieva
Magnetic Field Strength	1.5 Tesla	1.5 Tesla
Acquisition protocol	IR True FISP	TFEPI

Table 14: Comparison of the multi-centre cMRI equipment used

Major contributors to image quality include radiologist and radiographer experience and the field strength of the cMRI machine (both used 1.5T). To analyse the differences in the imaging capabilities at the centres, two healthy individuals volunteered to be scanned twice, once at each centre, to directly compare the images on the same anatomies. Figure 70 and Figure 71 show the scans corresponding to a healthy volunteer in his early-thirties, who happened to be an athlete. A scan of the second volunteer, in his early-twenties, appears in Figure 74. The differences in image properties are summarised in Table 16. The images collected at Guy's are of considerably higher quality in terms of image contrast and signal-to-noise ratio. This might reflect the greater experience of the Guy's team in the acquisition and processing of these images, and was despite on-site training in Sheffield by personnel from Guy's.

Figure 72 and Figure 73 show the scans corresponding to four patient cases. The patients shown are not the same in each clinical centre, for obvious reasons, for each centre an image from a patient collected early in the study and one collected late in the study is illustrated. This gives an indication of how the image quality changed as the study progressed.

Compared with scans from healthy volunteers, patient scans suffer from complications, including motion, due to comparative health and physiological status of the patient and their ability to comply with the imaging protocol. Breath-hold was challenging for some patients. It is apparent that the quality of the patient scans is more similar between the two centres. Despite the complications and reduced image quality, images from both centres were able to be segmented and processed using the workflow developed in the Grand Challenge project. Since the starting point for the current work is the segmented image, the challenges of segmentation can be regarded as a separate issue, although the smoothness of the segmentation clearly does impact on the registration process.

<b>PATIENT CASES</b>	<b>SHEFFIELD</b>	<b>LONDON</b>
Slice size (R x C)	416 x 348	288 x 288
Pixel dimensions (mm)	0.769231 x 0.769231	1.16 x 1.16
Slice thickness (mm)	1.5	0.9
Patient cases shown	1 7 14 20	1 7 14 20

Table 15: Comparison of the multi-centre cMRI image properties of the patient scans

<b>HEALTHY VOLUNTEER</b>	<b>SHEFFIELD</b>	<b>LONDON</b>
Slice size (R x C)	416 x 348	288 x 288
Pixel dimensions (mm)	0.769231 x 0.769231	1.25 x 1.25
Slice thickness (mm)	1.5	6
Volunteer shown	1	1

Table 16: Comparison of the multi-centre cMRI image properties of the volunteer scans

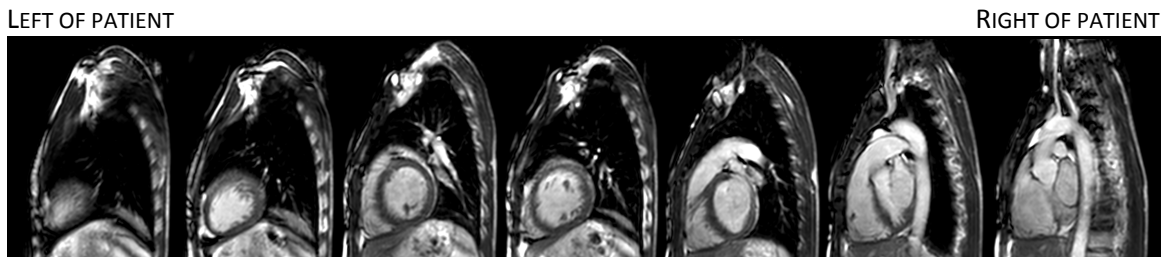


Figure 70: Sagittal slices of the volunteer scan taken at Guy's Hospital, London

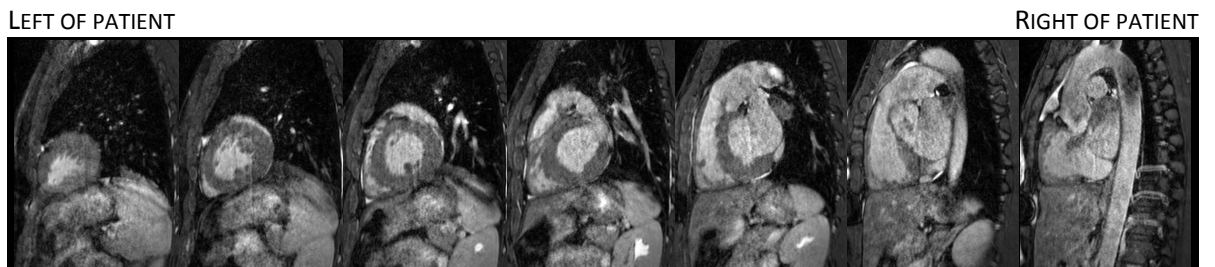


Figure 71: Sagittal slices of the volunteer scan taken at the Northern General Hospital, Sheffield

EARLY IN STUDY

LATE IN STUDY

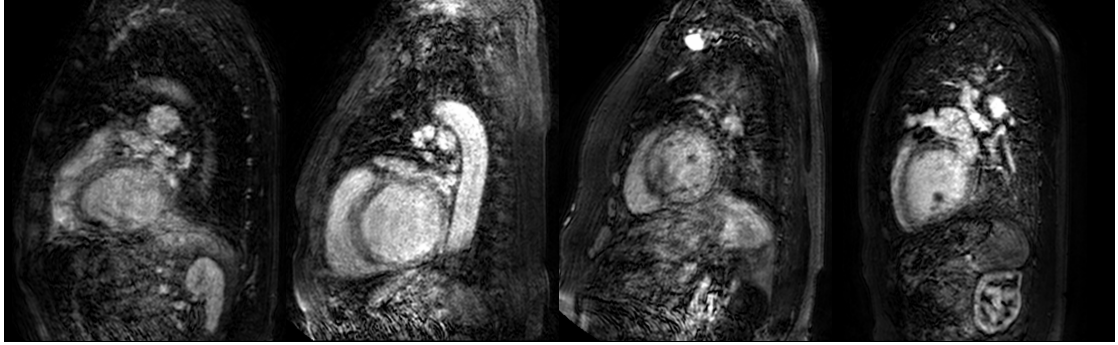


Figure 72: Scans of multiple patients taken at Guy's Hospital, London

EARLY IN STUDY

LATE IN STUDY

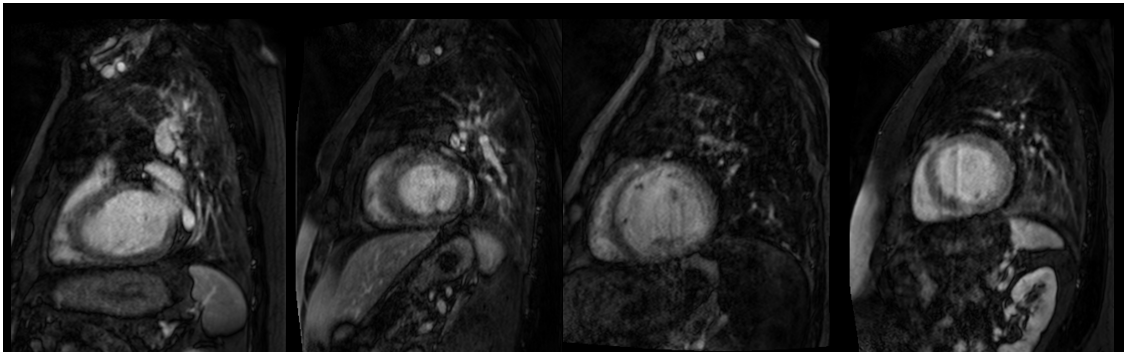


Figure 73: Scans of multiple patients taken at the Northern General Hospital, Sheffield

### 6.2.2 Image segmentation

Three segmentation techniques were employed. One of the goals of the Grand Challenge project was to develop a robust automatic segmentation process. This process, developed at UCL, was applied to all images from both centres. To provide comparisons, the images acquired at Sheffield were manually segmented by a junior registrar in cardiology (Manual Segmentation), and the images acquired at Guy's were segmented using an atlas-based segmentation tool provided to KCL by Philips (KCL Segmentation) [126]. The atlas is composed of the four heart chambers and the aortic arch. To make the process efficient, the model is initialised as a rigid model and then components are given increased freedom in turns. The atlas is first oriented to the medical image using a generalised Hough transformation followed by a global affine registration. The model is then fitted using an elastic, non-linear registration model. The material properties of the atlas are pre-determined, with different atlas components having different elasticity values. As the deformation progresses, the components of the model become aligned with the image objects and are fixed in space for the remaining deformations.

The images acquired at both centres were segmented using a multi-atlas based image segmentation tool developed at University College London (UCL Segmentation) [22]. Each atlas

is composed of a medical image and its corresponding segmentation of the heart chambers and major vessels. Firstly, all the medical images of the atlases are registered to the patient image (the image to be segmented). For all results, regions of the images are scored on image similarity and a compound atlas is created using the best matching regions. The compound image is registered again to the patient image and the segmentation will correspond to the patient image.

In practical terms, image segmentation, and in particularly manual segmentation, is a nondeterministic problem. This is because the user must make a decision on whether a pixel belongs to a region of interest or not, and this is based on the intensity of the pixel. Images with low signal-to-noise ratio, blurry images, images with poor contrast or images with low resolution introduce a level of judgement which is unlikely to be repeatable by another user following the same criteria, or indeed the same user on the same dataset.

What tends to happen is the user will perform a per-slice segmentation in one direction, making judgement on areas of high uncertainty. When the user corrects the segmentation in the remaining two directions, the user introduces more uncertainty to the volume by making more decisions that will likely contradict those made in the first pass. The overall uncertainty in the segmentation process often results in very uneven surfaces, which itself contradicts the fact that human organs are often very smooth.

Commonly available software packages for manual segmentation employ tools such as the paint brush or the lasso. The paint brush allows users to click and select voxels in a very intuitive manner. The disadvantage is the user must colour-in all of the area of interest manually. The lasso tool follows the user's mouse to draw an outline around a region of interest. The advantage over the paint brush is that the tool colour-fills the region automatically. An improvement to the lasso tool is using splines to interpolate between user-selected points. Some solutions use splines that adapt based on the image intensity (e.g. live-wire [127]) whilst other solutions balance image forces and elastic forces (e.g. elastic models, [128]), a more extensive review can be found here [23]. In practice, the performance of these tools vary greatly between images. Figure 74 shows a simpler approach where the user-selected points are joined using cubic splines; showing the user-selected points and connecting splines (left) and the resulting patch (right).

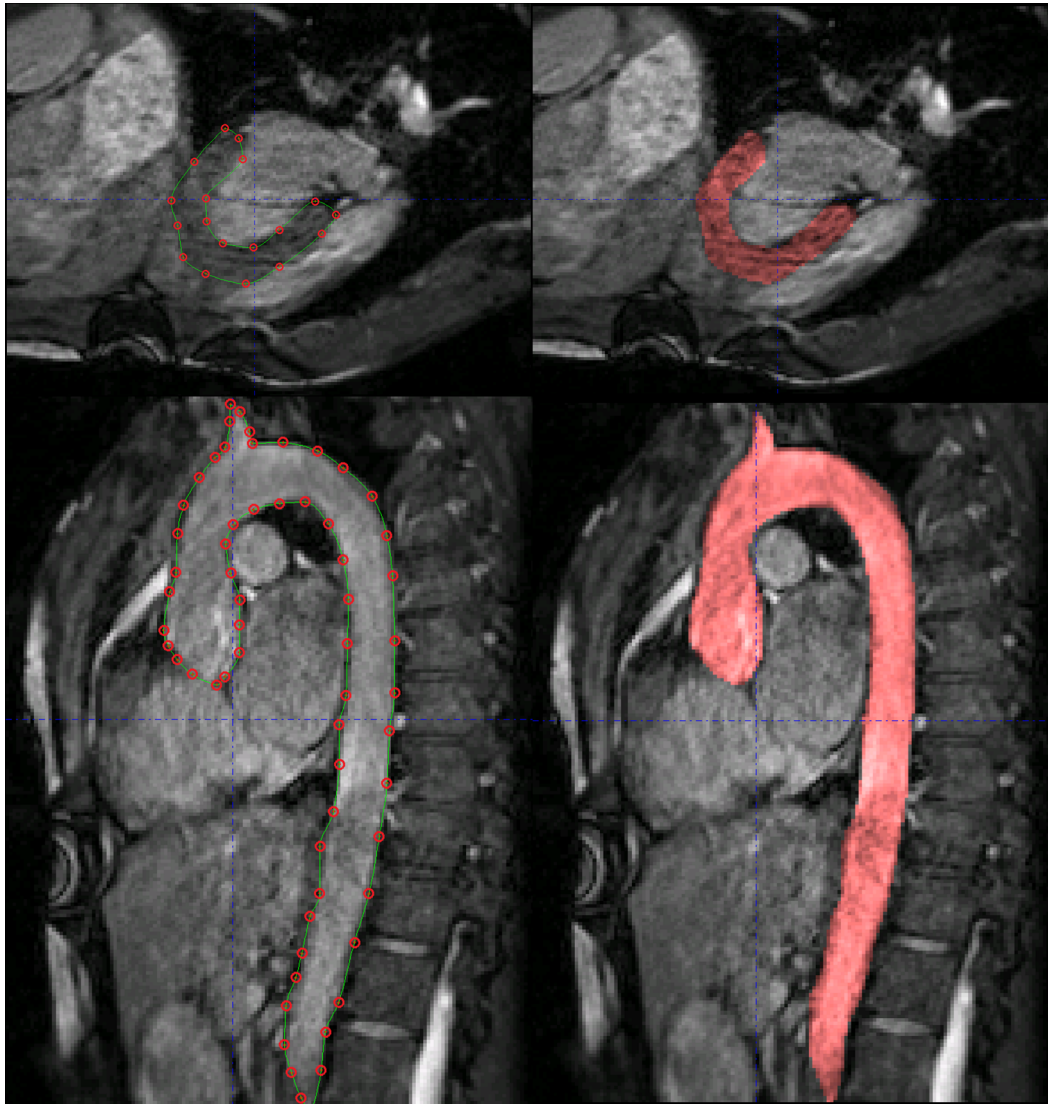


Figure 74: Manual segmentation of the left ventricular myocardium (top) and aorta (bottom) of healthy volunteer 2

The overall time taken to produce the full heart segmentation using the Philips tool (KCL) is less than 30 seconds. The tool has been specifically tuned to work on MR images after the developers originally used CT images, which offer higher contrast, reduced noise and in which the actual intensity value directly maps to the type of tissue imaged [126]. The robustness (and speed) of the tool lies on the multiple levels of detail used. It first uses a Hough transform to get the model into general alignment, and then uses multiple passes of a deformable model algorithm to adapt the model to the image [129]. The accuracy of the tool is generally acceptable however the tetrahedral mesh produced is not guaranteed to be useful for computer simulations (in terms of the correspondence of the element to the anatomy, and element quality). Furthermore, the underlying model guarantees that the resulting mesh will resemble the expected shape of the four chambers regardless of the accuracy of the fit. The adapted LV using this technique is shown in Figure 75.

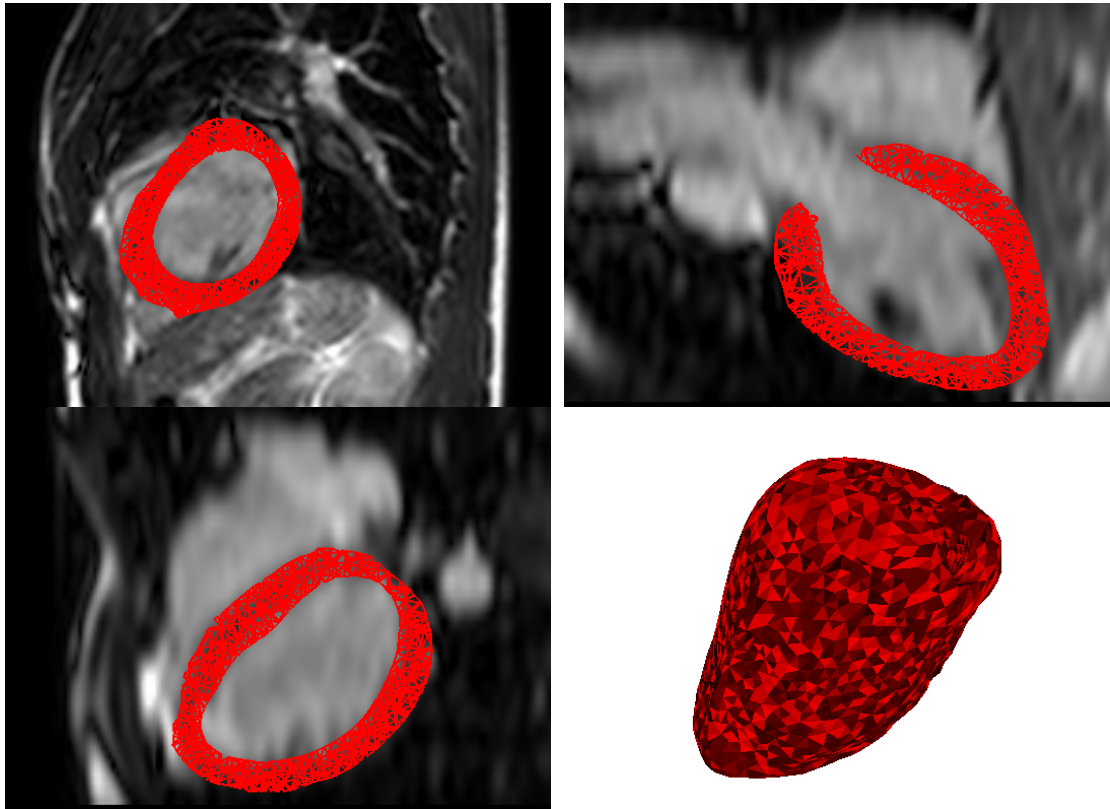


Figure 75: The triangular mesh generated by the KCL segmentation tool. The LV myocardial volume mesh is shown alongside the patient MRI

Comparing the manual and UCL-tool segmentations of the Sheffield images, those produced by the UCL tool are visually more appealing than the manual ones because they include the valve openings at the base of the ventricles. Valves are difficult to segment manually because they do not appear clearly (or are not visible) in the MRI image and are not included in the manual segmentations. The atlas based segmentations provide a rough estimate of the valve locations. Generally speaking, the segmentations generated by the UCL tool appear to be thinner than the actual myocardium in the MRI images. They also have a characteristic flat intra-ventricular wall, in contrast with the manual segmentations which appear rounder along all the surface. The uneven topology and the thinness of the target are challenging features for the application of the mesh generation process described in this thesis.

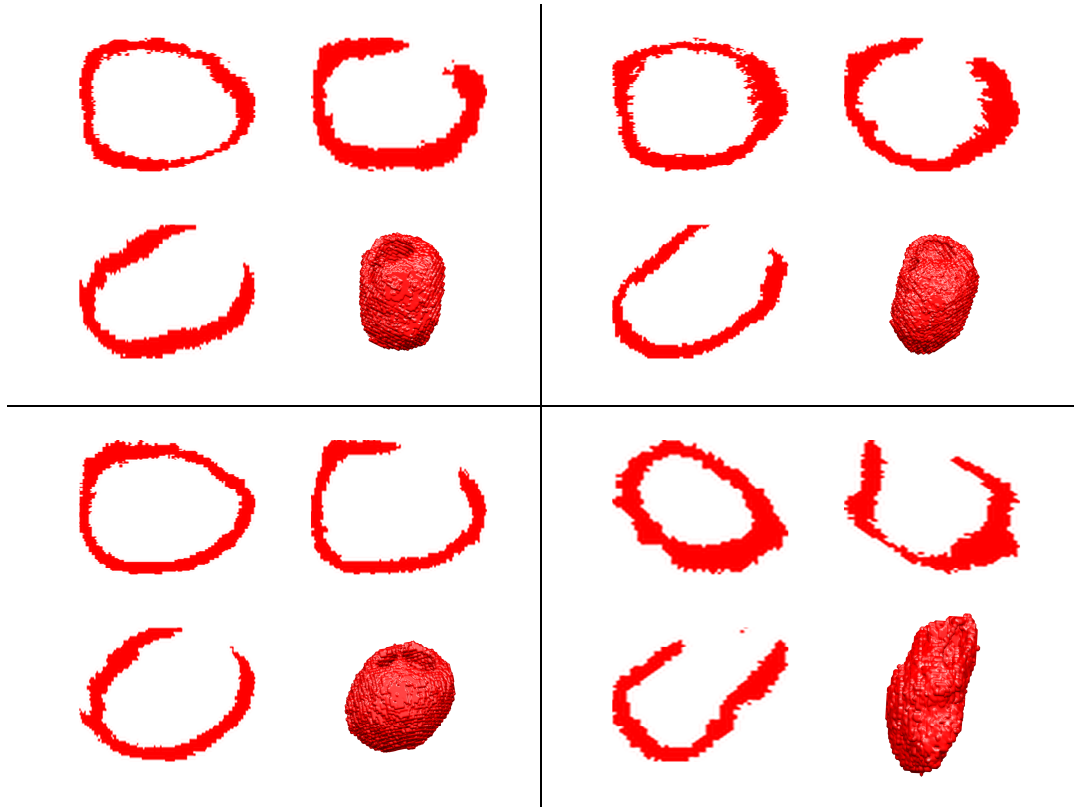


Figure 76: UCL automatic segmentations corresponding to the patient scans shown in Figure 73

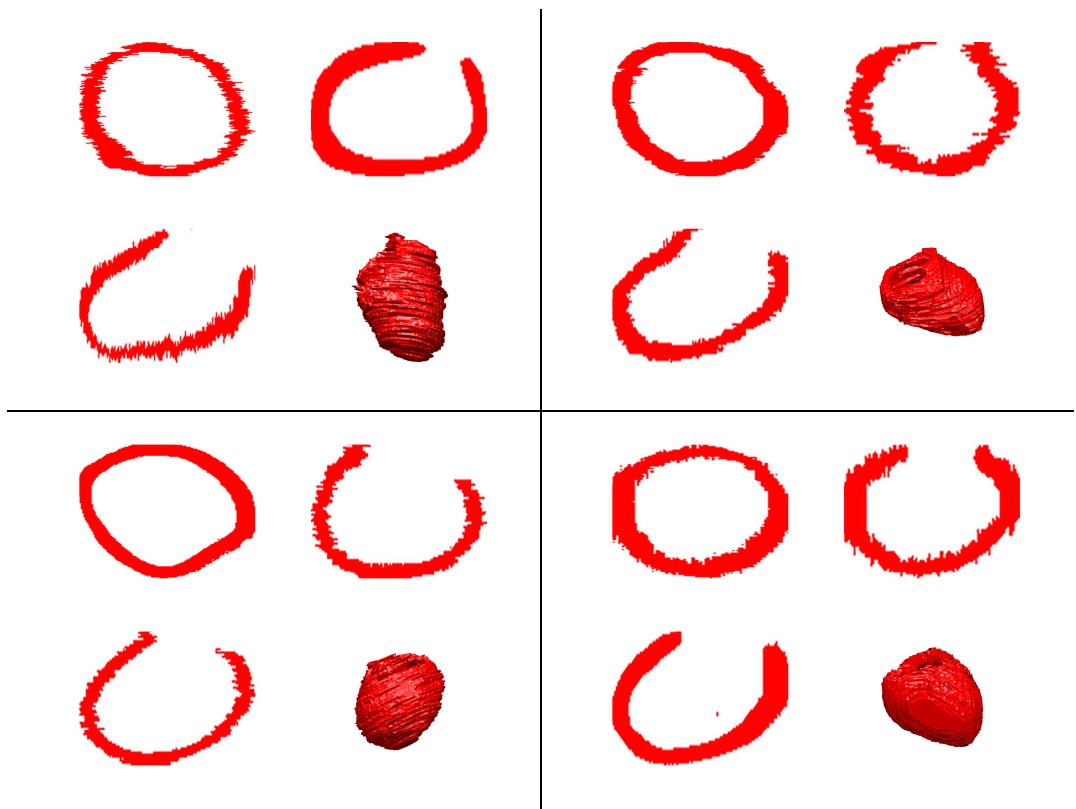


Figure 77: Manual segmentations corresponding to the patient scans shown in Figure 73



For the purposes of this thesis, which is not focused on the quality of the segmentation, quantitative comparisons were not performed. Visual comparisons are provided for context in Figure 78.

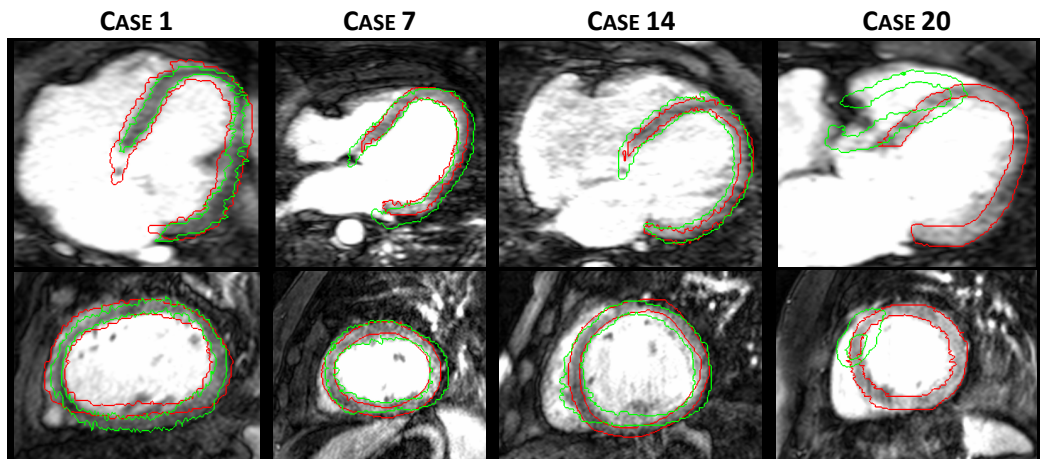


Figure 78: Visual comparison of selected segmentation results. The red and green outlines correspond to the UCL and manual segmentations, respectively

21 patients were recruited to the project in Sheffield but one patient was unable to be scanned. This resulted in a total of 20 cases segmented manually and by the UCL tool. Success is defined by visually inspecting the geometry to ensure the geometry was a viable target for the registration process. The UCL segmentation tool produced a result that failed this visual test. 20 patients were recruited to the project in KCL. The cases were processed inside the institution and only the segmentations were provided (not the corresponding MRI). The segmentations were again judged visually and 18 of the 20 cases segmented using the UCL tool and 4 of 5 cases segmented using the KCL tool were accepted. 2 volunteers were scanned in both centres. Table 17 shows all the segmentations used to create a personalised mesh. Rejected cases are shown in Figure 79 and discussed below.

The segmentations produced using Philips's KCL tool have a voxel size of 2.0 by 2.0 by 2.0 mm; the segmentations produced for the Sheffield cases have a voxel size of 1.5 by 0.77 by 0.77 mm except the volunteer cases which have a voxel size of 1.3 by 0.63 by 0.63 mm; and the segmentations produced for the London cases by the UCL tool have a voxel size of 1.0 by 0.88 by 0.88 mm.

RECRUITMENT CENTRE	LABEL	COHORT	SEGMENTATION TOOL EMPLOYED	NUMBER OF AVAILABLE SEGMENTATIONS	NUMBER OF REJECTED SEGMENTATIONS
Guy's	VOLK	Volunteers	UCL	2	0
NGH	VOLS	Volunteers	Philips	2	0
NGH	MAN	Patients	Manual	20	0
NHG	SHF	Patients	UCL	20	1
Guy's	KCL	Patients	Philips	4	1
Guy's	UCL	Patients	UCL	18	2

Table 17: Summary of the available segmentations

Renders of some rejected cases are shown in Figure 79; to help visualise the wall thickness they are made translucent. Cases number 3, 7 and 19 show holes in the segmentation (the bigger of these appear as fainter patches in (b)). Furthermore, the geometries show a highly irregular topology near the base (3) and apex (7 and 19) of the ventricle. Cases number 3 and 7 were rejected and number 19 was accepted as a challenge (discussed in section 6.5).

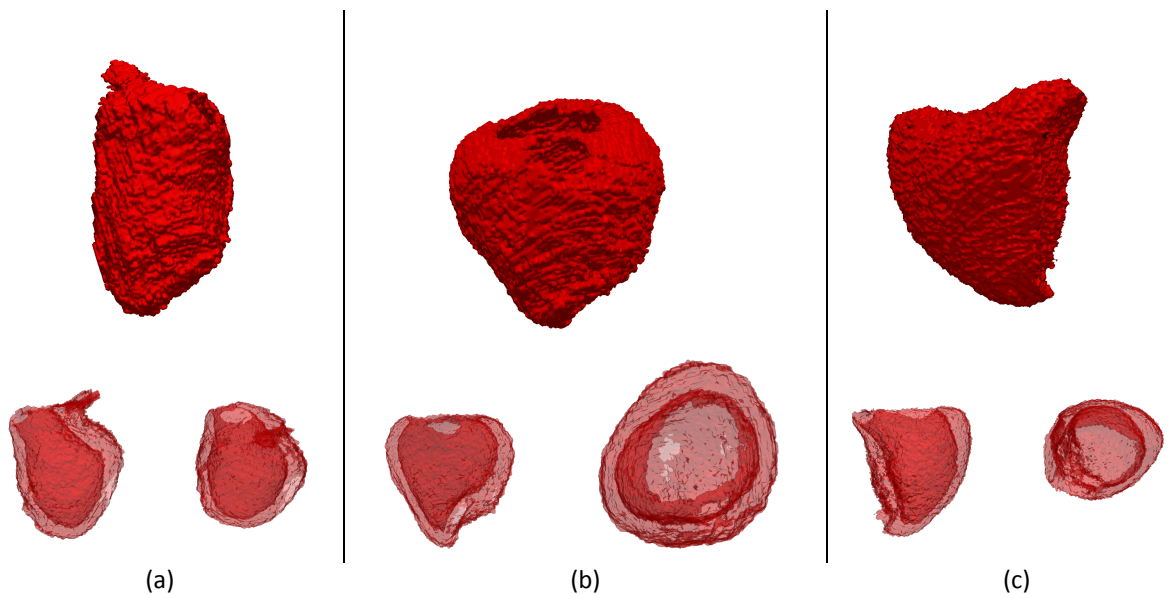


Figure 79: Rejected UCL segmentations from the London set of cases. The figure shows opaque and translucent renders of cases number (a) 3, (b) 7 and (c) 19.

## 6.3 Preparation for image registration

### 6.3.1 Pre-processing of segmented images

The previous section summarises the segmentations that were available for processing, using the registration software developed in this thesis, to develop high-quality patient-specific meshes. In fact these segmentations in their raw state have a number of recurring problems. Figure 80 shows examples of these in arbitrarily chosen cases. The manual segmentation shown

has a very irregular shape due to the fact that it was acquired in a slice-wise fashion. Other problems associated with manual segmentations include unintended artefacts and incomplete regions (holes). The UCL segmentation shown is smoother, but in this particular case the general shape is far from the expected ventricular shape. It is clear from the overlay view (b) that this is due to a segmentation inaccuracy. The KCL segmentation shown is fairly clean but the bitmask contains holes that must be closed. The intra-ventricular wall appears protruded and un-physiologically straight. This is a recurring problem appearing in the atlas-based segmentations used in this thesis; also in the apical region in the UCL segmentation. The reason for the wall deformity is the templates split the intra-ventricular wall into two sections, each belonging to one ventricle.

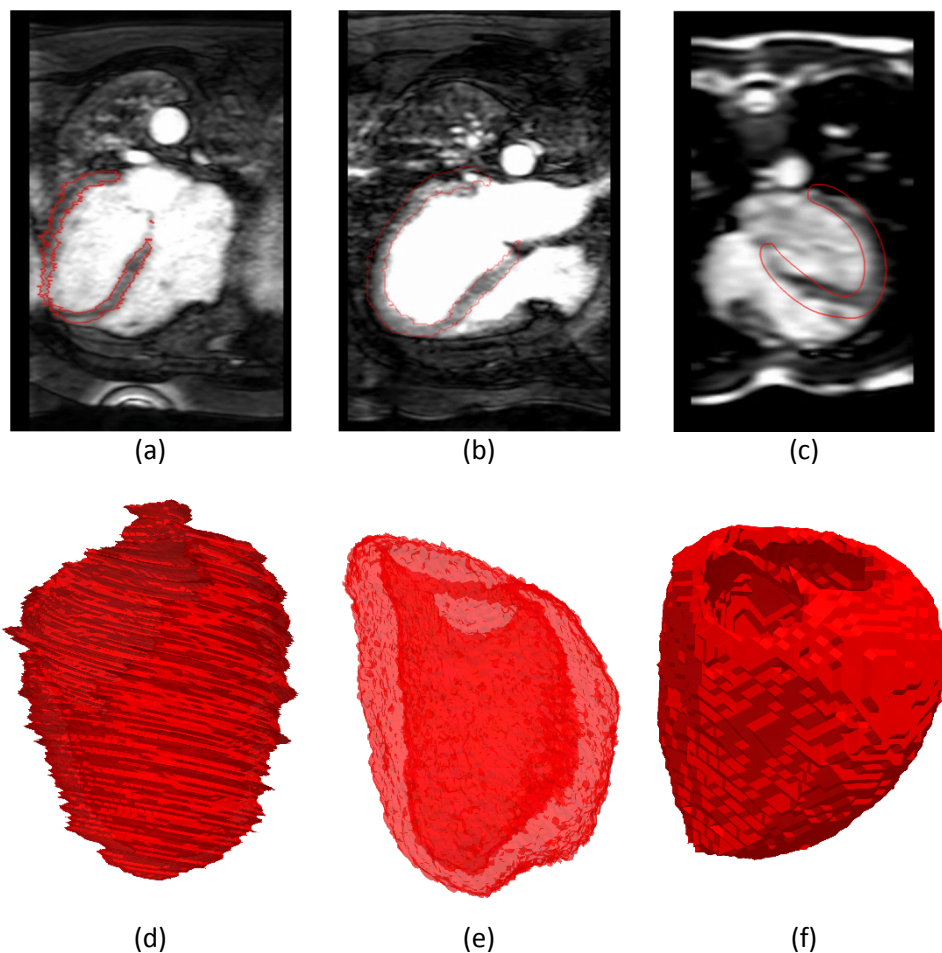


Figure 80: (a, d) Manual segmentation, (b, e) UCL and (c, f) KCL automatic segmentations. Slice through (a-c) and 3D render (d-f).

A 3D image filter is applied to the segmentation to improve the smoothness of the target image. The following selection of possible filters was compared: Gaussian, median and mean. The results were examined by visual inspection; the Gaussian filter changes the edges of the

segmentation the most whilst the mean filter gave the smoothest results. A comparison of the effects of the filters is shown in Figure 81.

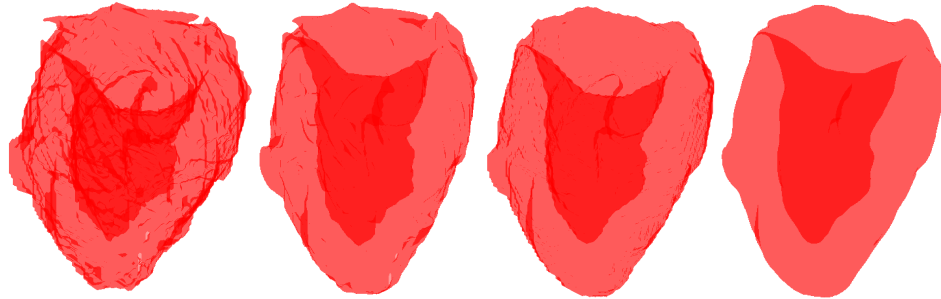


Figure 81: The effect of 3D image filters on a segmented volume. (1) The raw volume is a manual segmentation, produced by the author, using the tool shown in Figure 74 whilst trying to avoid the fallacies of manual segmentations shown in Figure 77 and Figure 80. (2) Gaussian filter. (3) Median filter. (4) Mean filter. (From left to right)

Typical results of the filtering process are illustrated in Figure 82. Generally, the filtering process corrects the jagged shape of the manual segmentations however in regions (or slice orientation) that are already smooth, the process tends to enlarge the segmentation by a couple of pixels; this is most obvious in the left hand side of Figure 82. The mean filter produces monochromatic images (as opposed to binary images) so a threshold is needed to produce the binary mask. The threshold was chosen at a value 0.4 (i.e. slightly lower than 0.5) which generally results in a smoother surface.

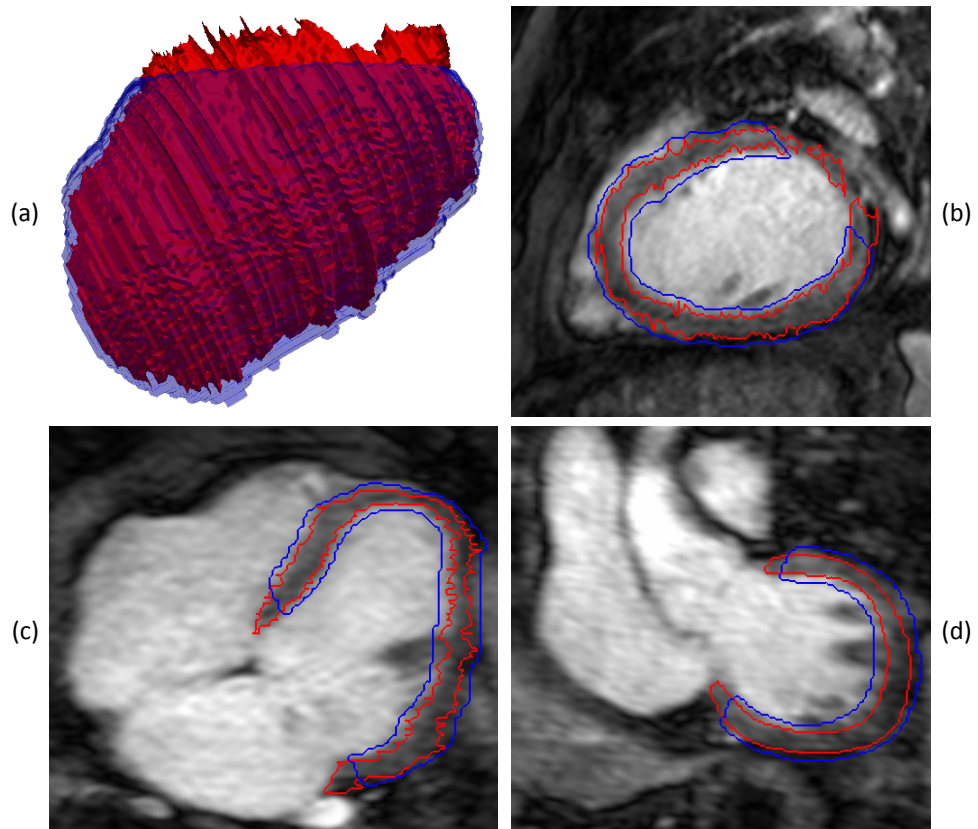


Figure 82: Comparison of the pre- (red) and post-processed (blue) segmentations: (a) isosurfaces, and (b)-(d) accuracy of the volumes, compared in orthogonal planes.

The template mesh of the ventricle does not include features such as valve or the basal wall (atrio-ventricular wall). To ensure correspondence between mesh and image, the segmentation must be trimmed at an appropriate location. The hand segmentations do not consistently define where the ventricular muscle ends, and the position of the valve and atrial wall. In some segmentations, the ventricular wall ends prematurely. For this reason, the trim location is located 5mm away from the manually defined basal plane.

The dilate-mean-erode operation was found to preserve the features better than the classical morphological closing operation (dilate-erode). Theoretically, a sufficiently large morphological closing operation will close all holes in an image, but this is dependent on the size of the holes. For instance, the first case in Figure 83 shows a jagged surface which will require a much larger closing operation than the second and third cases. Instead, the dilate-mean-erode operation captures a larger range of discontinuities in the surface. The morphological operations used a 5 voxel wide kernel. The whole process of filtering the image takes between 2 to 10 seconds, depending on the size of the image, when implemented in MATLAB. Figure 83 shows the processing results; from left to right, cases MAN01, SHF01 and SHF03. In summary, the filter steps are as follows:

- 1) Dilate the image
- 2) Apply a mean filter
- 3) Threshold the image at 0.4
- 4) Erode the image
- 5) Remove all voxels above the manually selected basal plane

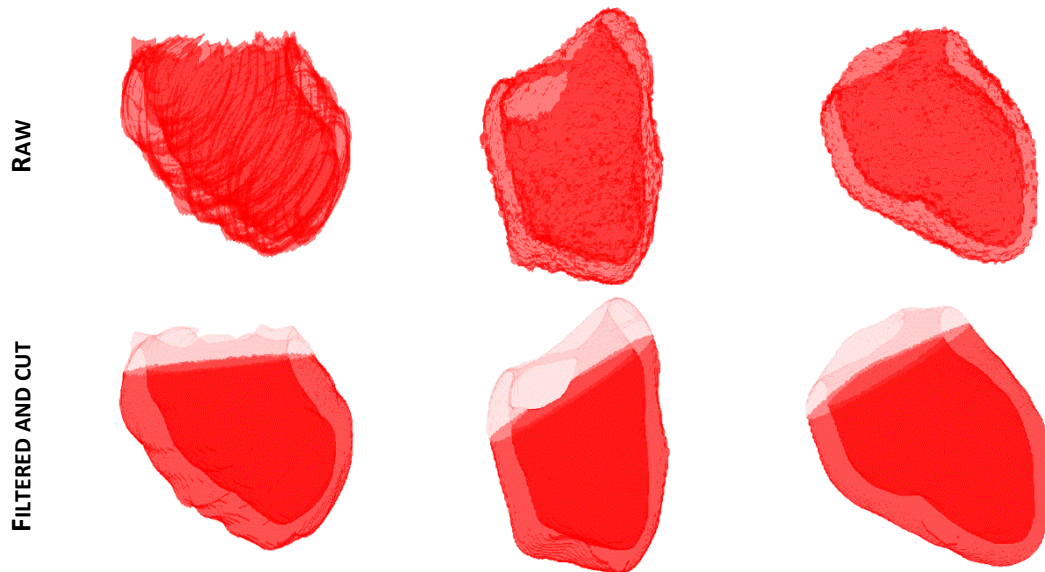


Figure 83: Processing of 3 patient segmentations: the raw volume (top), and filtered and cut volume (bottom)

### 6.3.2 Template Mesh

Image preparation and mesh generation are part of the process but are presented in separate sections for clarity. For each image, a tailored template mesh is automatically created. Image registration works best when the reference and moving images are most similar, so it is useful to construct a patient-specific template mesh. The variation in the templates are described using simple parameters but have the same topology. Figure 84 shows the variables of the template mesh.

The template mesh is designed only to represent the left ventricle. The truncation at the basal plane removes the region of the valves. This is appropriate for the particular analysis protocol adopted in the Grand Challenge project, in which simplified representations of the valve are prescribed on the basal plane. The methodology can be applied to other templates, including more complete valve representations and two-ventricle models, but these applications are beyond the scope of this thesis. The inclusion of the details of the valves would introduce several

practical problems, including difficulties in imaging the valve leaflets and a tendency to produce distorted elements in this region. Furthermore, the simulation (in the Grand Challenge project) requires a flat basal plane.

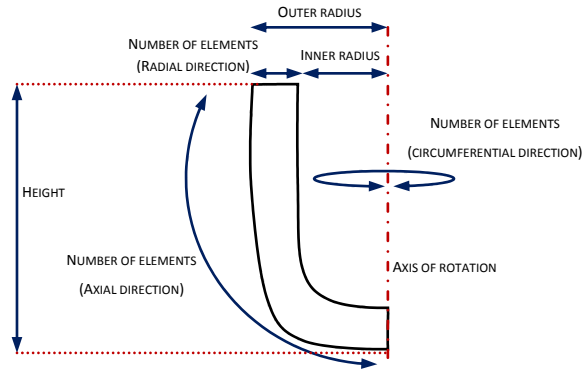


Figure 84: Parameterisation of the template mesh using characteristics of the ventricular geometry.

The user input are three points to define the basal plane and the number of elements of the mesh. The total number of elements is dependent on the number of elements in the axial (base-to-apex direction), circumferential and radial directions. Figure 85 shows the mesh used to produce the results in this chapter; the mesh has 16, 24 and 4 elements in each direction, totalling 1536 elements. Meshes with 2 and 3 elements in the radial direction were also built.

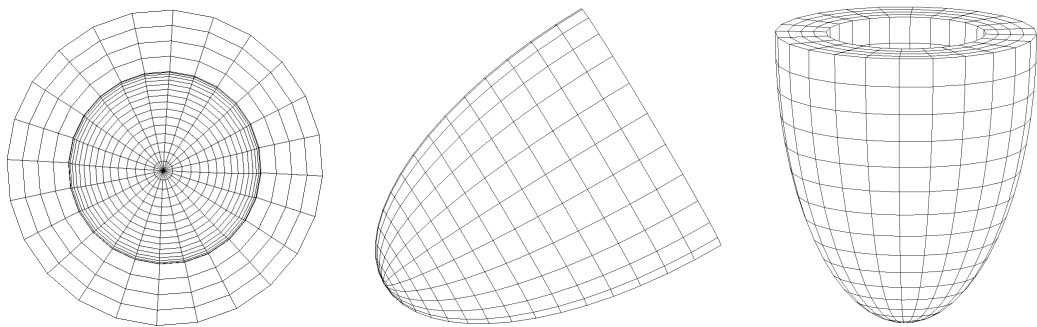


Figure 85: A typical template mesh viewed from different angles.

Firstly, a main axis needs to be defined. In this work, the vector normal to the basal plane and that passes through the centre of gravity of the cloud of points is used. An alternative choice is the principal axis of the cloud of points as determined by principal component analysis, a better choice for generating bi-ventricular meshes.

Next, the location of the apex is estimated by finding the point whose perpendicular distance to the basal plane is furthest away; this determines the height of the ventricle. It also determines

the direction of the ventricle from the basal plane, so the points that belong to the basal wall are removed.

To determine the outer width of the mesh, the point that is furthest away from the main axis is selected. The inner width is selected at an arbitrary 55% of the outer distance to simplify the process (selecting an appropriate wall thickness is not trivial). With these measurements the template mesh is built. Finally, the mesh and segmentation are aligned in a pre-registration step. To do this, the mesh is oriented in the same direction as the main axis of the segmentation. The main axis of the mesh corresponds to the vector that points to the apex of the mesh from the centre of gravity of all nodes. The centres of gravity are aligned and the basal plane of the mesh is oriented to that of the segmentation. The MATLAB implementation of the process completes in roughly 1 second.

Figure 86 shows two valid topologies that may be used to represent the ventricle. The configuration on the right is similar to the 17-segment model for the left ventricle [130], or bull-eye's representation. In order to avoid the singularity at the node in the location of the apex, an alternative "butterfly configuration" (shown on the right) may be used.

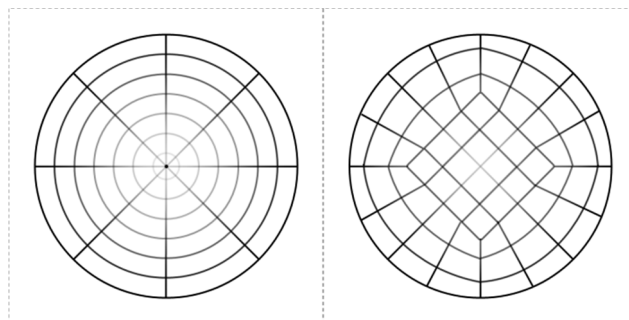


Figure 86: Two possible configurations to arrange the elements at the apex of the ventricular meshes

The topology of the ventricular mesh contains pyramidal elements (two parallel edges of a hexahedron each collapsed to a point) at the apex. This is a deliberate design choice over other topology combinations (e.g. the structure illustrated to the right of Figure 86). Although the alternative structure avoids the use of degenerate elements, it is more difficult to map the results onto an intuitive representation of the ventricle (bull's eye plots, which depend on rotational symmetry, are often used for clinical interpretation [33].) This correspondence is useful for other applications, such as interpreting simulation results and shape analysis, and is the topology of choice of the project partners.



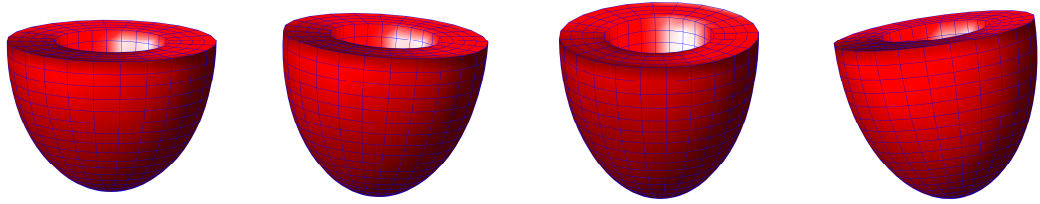


Figure 87: Patient-specific template meshes; Sheffield cases numbers: 1, 7, 14 and 20 (left to right)

Figure 87 illustrates the variation of the patient-specific templates in practice. The most important aspect of this stage is the orientation, alignment (with the basal plane) and mesh height. The current implementation of the process is semi-automatic as the user must input the basal plane locations, however this can be automated.

### 6.3.3 Quality Metrics

In the previous chapter, the concepts of image similarity and element quality were used to assess the results of the registration. The first criterion is useful to describe the difference between images but the distance between two surfaces is best measured differently. Instead, the accuracy of the fitting process will be assessed measuring the distance between the surfaces of the segmentation and mesh. The second criterion is a useful metric but to quantitatively assess the mesh fitting process, the mesh quality will also be assessed in terms of overall mesh deformation.

The mesh surface was produced by breaking each of the surface quadrilaterals into four triangles. This allows comparison with the segmentation for the closest node, edge or plane, as shown in Figure 88. The segmentation surface was constructed using the isosurface function offered in the standard MATLAB package, which produces a tetrahedral mesh. The distances reported are those from each node in the segmentation surface to the mesh surface.

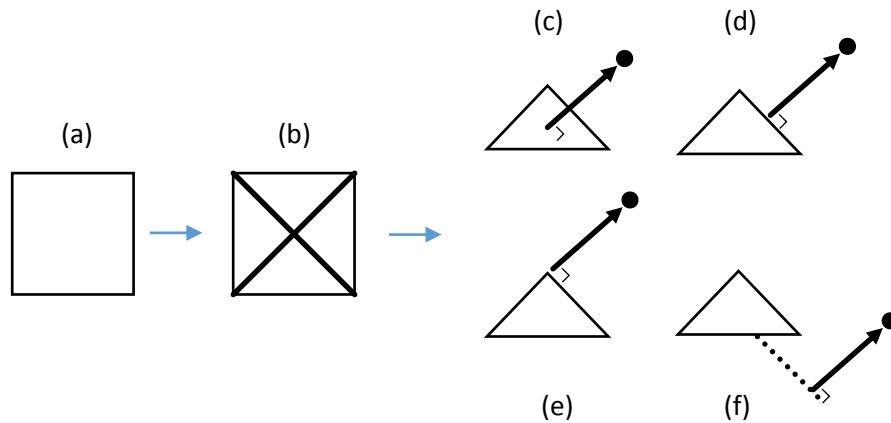


Figure 88: Schematic of the surface distance measurements. The mesh faces (a) are broken into face triangles (b). For each point in the surface of the segmentation, the closest point on the face triangles is computed, which could be on the triangle face (c), edge (d), node (e) or plane (f).

The results obtained are dependent on the way the surfaces are obtained. The mesh surface reformulation was chosen for being the simplest, symmetrical way to break a quadrilateral into triangles. For the segmentation surface, the MATLAB implementation was chosen for convenience. The surface produced using the isosurface function was compared to a naïve implementation using the voxel corners and the differences were found to be insignificant.

The fitting distances are always an absolute value (there is no positive and negative side to the surface because there is no right and wrong side, any deviation from the surface is regarded as a fitting error). At each point of the segmentation isosurface, the error is the L2-norm of the individual Cartesian components; the overall fitting distance is the simple mean of the vector of the distances between the isosurface nodes and the mesh triangular surface. In practice, the resulting fitting distance can never be zero; in fact, there will be a minimum fitting distance that can be reached, due in part to the sparse sampling used inside the mesh (i.e. if the sampling points are 1 mm away from each other one can expect a minimum fitting error around that value).

A further observation is that the surface will be fitted to accommodate the whole domain so if the surface is irregular, the mesh fitting will attempt to satisfy a best fit for all points in the surface. When the distance from the fitted mesh to the segmentation is measured, irregularities are measured as an error. The experiments presented in this chapter will use data that has been considerably smoothed to ensure a meaningful assessment is made. The results of the meshing tool working on the raw segmentation will be presented and described to demonstrate the robustness of the method.

Assessing the element quality by comparing its deviation from the ideal shape is not particularly useful in this scenario since, for any practical application, the elements will never be perfect cubes; it would partly defy the point of generating fitted meshes. A more useful measure of quality, to study the effect of the optimisation process, is the deviation of the quality of the fitted mesh from the quality of the idealised mesh.

The element Jacobian ratio is the ratio of the minimum and maximum Jacobian within the element. For linear hexahedra it can be assumed that both maximum and minimum will be measured at the corners of the element,

$$q = \frac{\min(J)}{\max(J)}$$

$$J_i = \begin{vmatrix} d\xi/dx & d\zeta/dx & d\eta/dx \\ d\xi/dy & d\zeta/dy & d\eta/dy \\ d\xi/dz & d\zeta/dz & d\eta/dz \end{vmatrix}_i \quad (i = 1 \rightarrow 8)$$

The degradation of an element is the ratio of the element Jacobian ratios after and before the registration,

$$l = \frac{q_{morphed}}{q_{template}}$$

The degradation of the mesh is the mean element degradation. The collapsed elements at the apex are not included in the computation as these can be handled differently by different simulation software. More importantly because the collapsed elements are not hexahedra, the quality measurement gives them a score of zero which negatively skews the final measurement.

A truly valuable measure would use the element of worst quality to quantify mesh quality, however it is a difficult optimisation problem to formulate. Instead, one can aim to reduce the number of bad elements to a minimum and make adjustments prior to the simulation. The threshold at which an element is considered of poor quality is a debatable question. Lamata et al. [32] suggest that for Hermite cubic hexahedral elements a threshold of 0.33 (measured at the integration points) is a good estimate of mesh quality. On the other hand, ANSYS [102] quotes a range between 0 and 40 because it differs with the problem type, and because ANSYS works with both linear and higher order elements of different shapes of Lagrangian formulation. ANSYS have determined that measuring the ratio at the integration points provides a robust metric for linear elastic elements but suggest measuring the ratio at the nodal points for “general usage” [38]. The results section of this chapter uses a threshold of bad elements of 0.4 purely for statistical purposes. The template mesh does not contain any elements of quality less

than 0.33 so setting the threshold at a more conservative level demonstrates more clearly the improvement of mesh quality delivered by the use of the Jacobian regularisation term.

## 6.4 Registration parameters

Despite the template mesh being constructed for each segmentation based on geometric features, this is still a global alignment. To make the process more robust, the subsequent fitting process is broken down into two steps: a heavily constrained deformation used to adapt major features of the geometry, followed by a less constrained deformation used to capture local features. Both steps produce non-linear deformations. The heavily constrained deformation is referred to as the adaptation step. The less constrained deformation is referred to as the tuning step. The Tikhonov weights of the smoothness and mesh quality terms in the tuning step can be independently chosen to achieve particular characteristics of the final mesh. In this thesis, two sets of results are presented, one in which the weights are chosen to produce a better surface fit (referred to as the conformance weights) and one in which they are chosen to produce a better mesh quality (referred to as the quality weights).

In summary the process of constructing a patient-specific mesh is:

1. Construction of patient-specific axisymmetric template from a parameterised model for overall sizing and alignment.
2. Adaptation step using strong regularisation to fit to the major geometrical features of the segmented anatomy.
3. Tuning step to fit the fine detail of the segmentation and/or to optimise mesh quality.

Table 18 summarises the values used at each stage of the mesh fitting process.

	<b>ALIGNMENT</b>	<b>CONFORMANCE</b>	<b>QUALITY</b>
Tikhonov Weight (smoothness)	100 (normalised)	10 (normalised)	1 (constant)
Tikhonov Weight (Jacobian)	0	0	1000
Memory	ON	ON	OFF
Sampling points	2 x 2 x 2	4 x 4 x 4	2 x 2 x 2
Smoothing kernel neighbourhood	1	1	0
Typical number of iterations	40	25	15

Table 18: Summary of the registration parameters used in the fitting stages

The Tikhonov weights are chosen to impose particular characteristics on the final mesh, and the magnitudes have been determined by experimentation. The memory term in the Laplacian (chapter 4, section 4.3.2) is used to maintain the integrity of the shape when no quality regularisation scheme is used. The selection of the number of the sampling points is largely dependent on the size of the mesh and the fit required; generally, for a 12 mm wide ventricular wall and a fitted mesh constructed using four elements in the radial direction, using a 2x2x2 sampling grid will place them about 2 mm apart. Using an 8 point sampling grid will place them about 0.5 mm apart. The smoothing kernel can only be as big as the sampling grid, and in practice for coarse alignments the maximum value is typically chosen. In this case, the alignment stage was strongly constrained (as opposed to the image largely smoothed) to make the process easier to follow. Table 18 presents the approximate number of iterations required for each stage. The maximum number of iterations used is application dependent and the process stops if the movement becomes too small.

## 6.5 Results and discussion

The program was tested using a Dell (Precision T500) workstation configured with a 3.47 GHz (Intel Xeon) CPU, 72 GB RAM and Microsoft Windows 7 Pro operating system. The program was compiled using the gfortran compiler (GNU Compiler Collection, version 4.8.1 [131]) on the 64-bit Cygwin environment, with single thread execution [132].

The alignment stage performed well in bringing the templates close to the segmentation by scoring an average  $0.617 \pm 0.685$  mm in surface distance measurements, achieved at the expense of an average mesh degradation of 1.023.

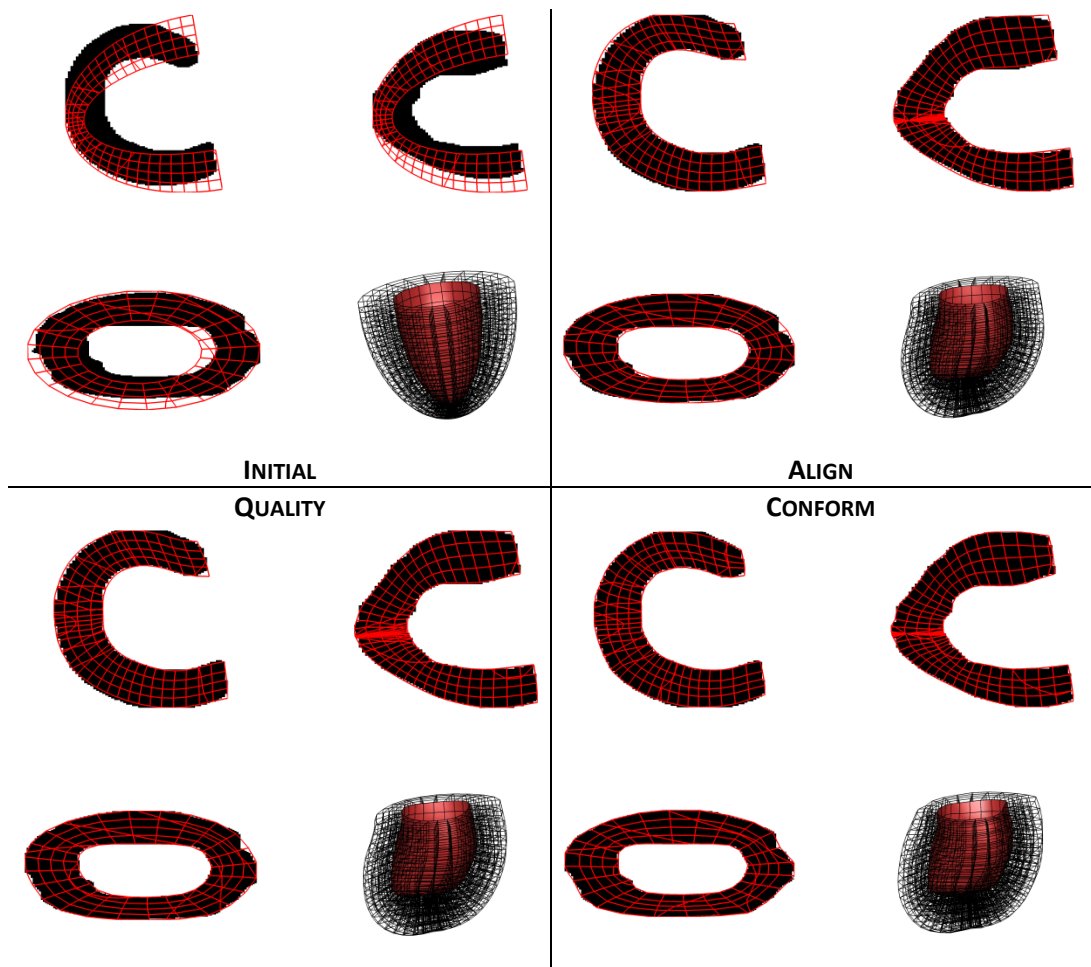


Figure 89: Evolution the mesh of case MAN19; initial template, after alignment, quality and conformance. In the cross-sections, the segmentation appears black and the mesh appears red.

The average mesh fitting distance was  $0.408 \pm 0.479$  mm using the conformance protocol and  $0.517 \pm 0.595$  mm using the quality protocol. Figure 90 shows the mean fitting distances for each case. Disregarding cases LDN11, MAN12 and SHF15, which will be examined in detailed later due to their disproportionately large errors, brings the means down to  $0.331 \pm 0.347$  mm and  $0.442 \pm 0.467$  mm for conformance and quality protocols, respectively.

The mean mesh degradation was 0.997 for the conformance protocol and 1.033 for the quality protocol. The template mesh has 5.88% of elements that are degenerate, that is, elements with a Jacobian ratio less than 0.4. The percentage of degenerate elements was 9.01% and 3.29% for the conformance and quality protocols, respectively.

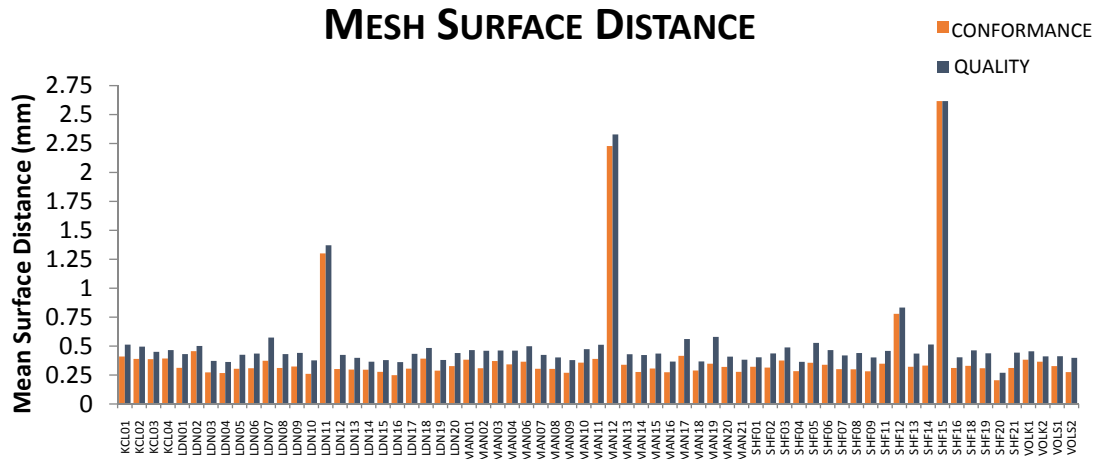


Figure 90: Resulting surface distance for all cases executed in this work, using the conformance and quality mesh fitting protocols.

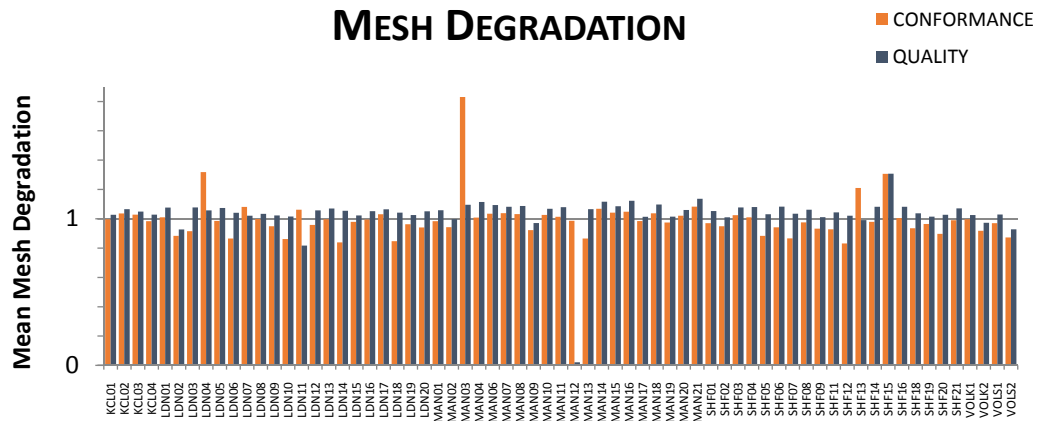


Figure 91: Resulting mesh degradation for all cases executed in this work, using the conformance and quality mesh fitting protocols.

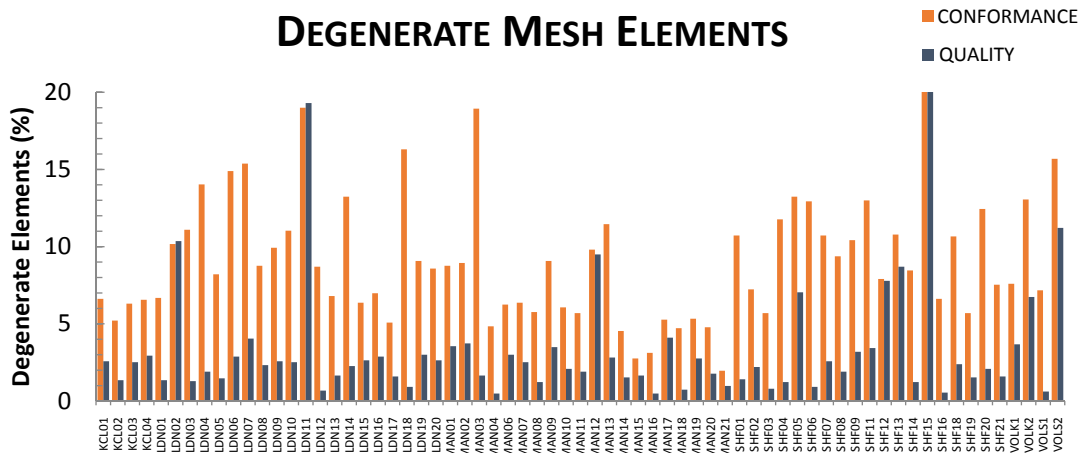


Figure 92: Resulting number of degenerate mesh elements for all cases executed in this work, using the conformance and quality mesh fitting protocols.

It is clear from the mean overall results that each protocol performs as expected by scoring better than the other in their respective performance measure. It is worth remarking that both protocols achieve sub-voxel precision making all meshes representative of the patient geometry. It could be argued that a closer fitting to the geometry makes a better mesh, but when the fitting scores less than the voxel precision there is no longer the certainty that the geometry is in fact more patient-specific. When weighed against the image voxel size, the improvement in fitting accuracy between the quality and the conforming mesh is 10.928% (0.111 mm). On the other hand, the quality protocol averages a 6.866% improvement in degradation and 68.603% less degenerate elements than the template mesh. Cases LDN11, MAN12, SHF15 and VOLS2 were treated as anomalous for statistical purposes.

Table 19 summarises the results by classifying the cases by segmentation type (columns labelled LDN, MAN and SHF), not including the failed cases. The column labelled AVERAGE includes all cases. It is clear that the method presented in this chapter performs well independent of the segmentation type used.

	<b>LDN</b>	<b>MAN</b>	<b>SHF</b>	<b>AVERAGE</b>
Total Segmentations	19	19	18	67
Fit improvement (% , quality to conformance)	11.965	11.691	11.440	10.928
Degradation improvement (% , conformance to quality)	8.443	6.419	9.401	6.866
Decrease in degenerate elements (% , conformance to quality)	72.710	64.074	71.178	68.604

Table 19: Summary of the improvement in metrics, broken down by segmentation type

Figure 93 shows case MAN 13 which is an excellent example of a good result. The fitting distances were  $0.339 \pm 0.366$  mm and  $0.430 \pm 0.478$  mm, and the mesh degradations were 0.865 and 1.065 for the conformance and quality protocols, respectively. This means the mesh quality optimisation is a trade-off of 0.091mm in terms of fitting distance (8.95% of the voxel size) for an improvement of 0.2 in terms of degradation (23.18% less degradation). Most of the degraded elements are fixed using the quality protocol except for a small number of elements due to the local geometry of the target segmentation. It is clear from the distance plot that the only noticeable loss in fitting accuracy happens at the base of the ventricle, where the wall thickness narrows.



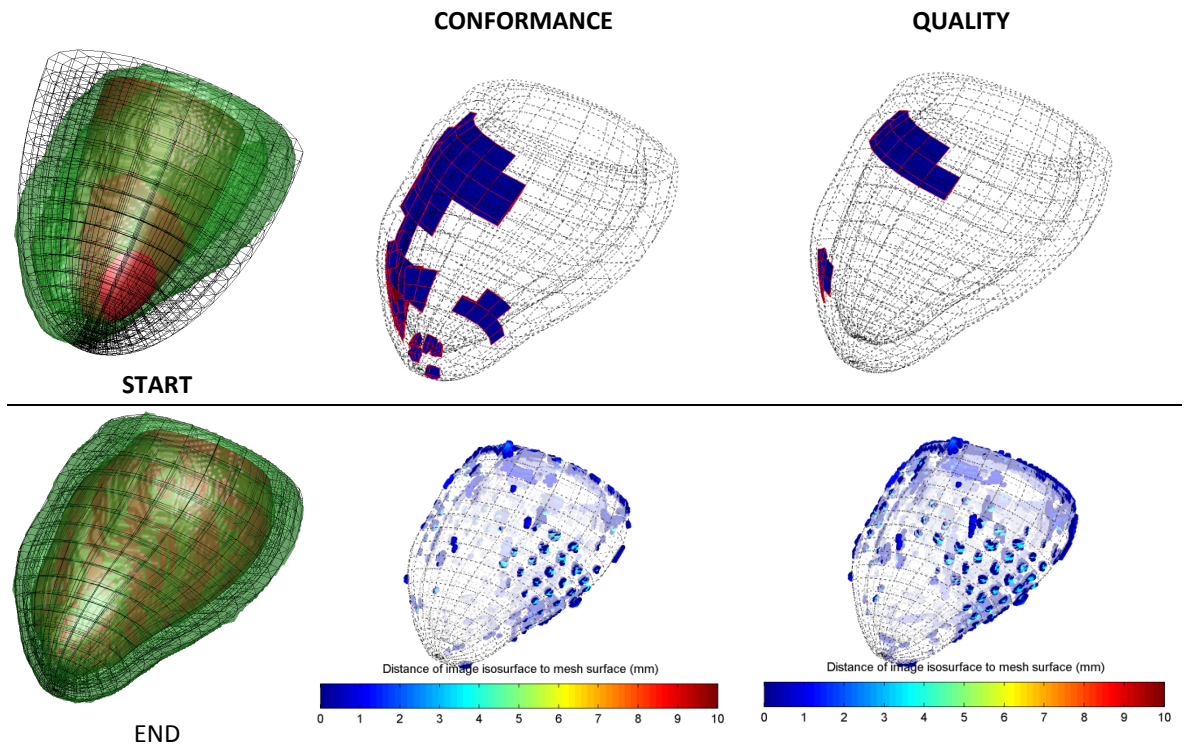


Figure 93: Processing results for case MAN13. The figure shows a side view of the mesh is shown with the target isosurface (left-most column). A view of the mesh with badly deformed elements highlighted is shown (top row) and the mesh with the points in the segmentation isosurface that are more than the mean voxel size distance away from the mesh highlighted (bottom row). The columns are ordered by start-end position, conforming and quality mesh (left to right).

Case SHF 16 (Figure 94) is a very similar shape. As in the previous example a number of degenerate elements appear on the side and apical region of the ventricle however the degenerate elements are not clustered together as above. The quality protocol removes all degenerate elements. It can be seen that in the thicker wall regions (right hand side in the images) a closer fit is obtained than in thinner regions. Overall, the fitting distances were  $0.311 \pm 0.362$  mm and  $0.403 \pm 0.459$  mm, and the mesh degradations were 1.005 and 1.081 for conformance and quality meshes, respectively. This represents a 9.086% (of the mean voxel size) difference in mesh fitting and a 7.660% improvement in mesh degradation.

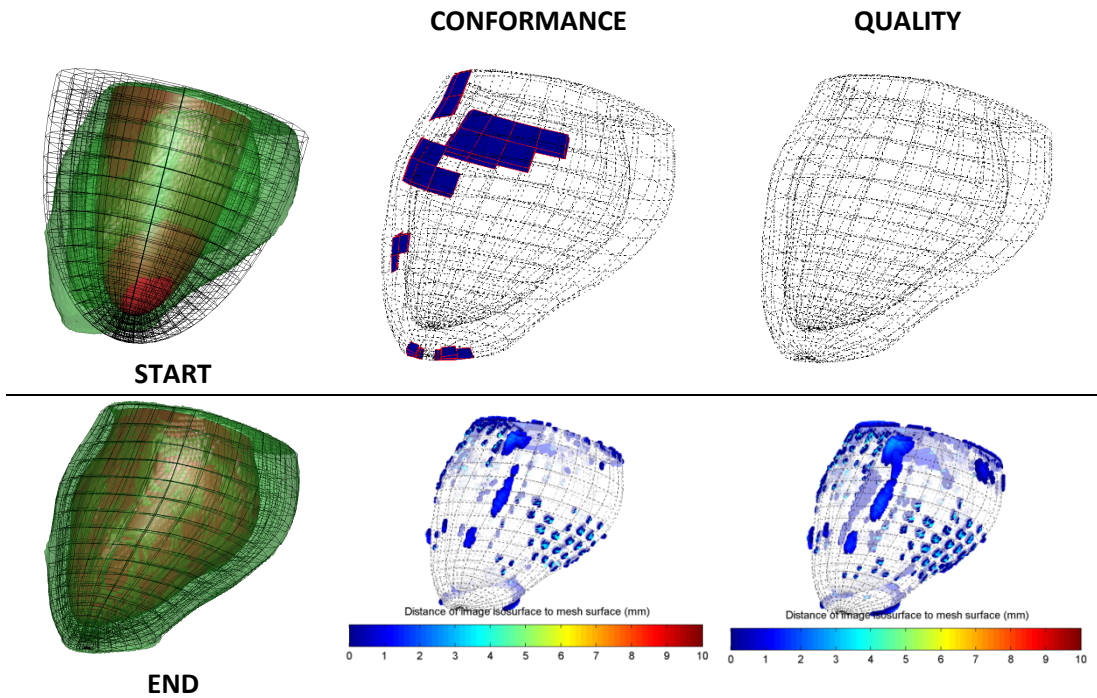


Figure 94: Processing results for case SHF16 The figure shows a side view of the mesh is shown with the target isosurface (left-most column). A view of the mesh with badly deformed elements highlighted is shown (top row) and the mesh with the points in the segmentation isosurface that are more than the mean voxel size distance away from the mesh highlighted (bottom row). The columns are ordered by start-end position, conforming and quality mesh (left to right).

LDN 06 presented as a challenging case due to the narrow geometry of one side of the ventricle. It remains unclear whether this is realistic or an artefact of a failed segmentation; in either case the flat side represents the intraventricular wall. Being able to create a mesh for this geometry shows the robustness of the process but the quality of the elements has to be judged accordingly. There are a high number of elements (14.89%) that are degenerate or have a negative volume, this would not produce a mesh suitable for mechanical simulations. The quality protocol reduces the number of degenerate elements by 80.66% and removes all volumes with negative volume. The mean mesh fitting are  $0.308 \pm 0.420$  mm and  $0.435 \pm 0.514$  mm (13.75% of the mean voxel size) and the mesh degradation are 0.865 and 1.041 (20.38% less degradation) for the conformance and quality meshes, respectively.

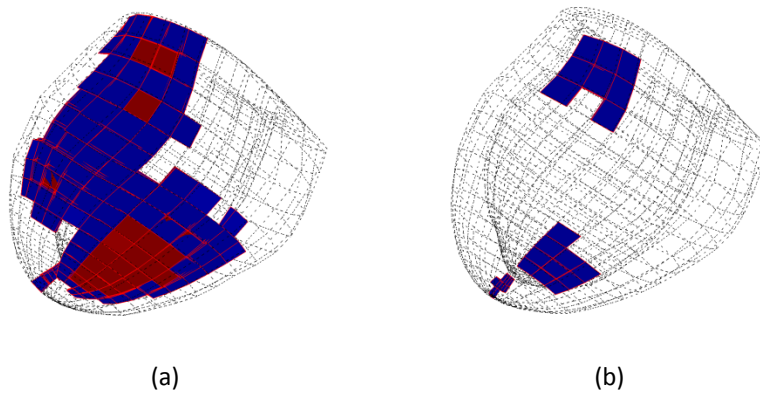


Figure 95: Degenerate elements of the resulting mesh for case LDN06, using the conformance protocol (a) and the quality protocol (b)

Case SHF 15 failed to fully conform to the segmentation, failing at a region where the wall was particularly thin. In addition, the starting position of the mesh resulted in failing elements far away from the target (when compared with the rest of the elements). This segmentation clearly failed so this target is an unfair challenge for the process. This case can be morphed by smoothing the images a larger amount but this leads to poor element quality in the region of the thin ventricular wall.

Case MAN 12 failed as a region of the mesh had inverted elements. The segmentation was a thin target and the initial alignment of mesh and target was optimal except for the region that produced elements of poor quality. When such an initial setup is encountered, the best fitting elements tend to conform to the segmentation before the rest of the elements deform into place (in a zipper-like action). However, in this particular case, it was observed that the system of equations required a large amount of iterations to solve and the registration did not complete.

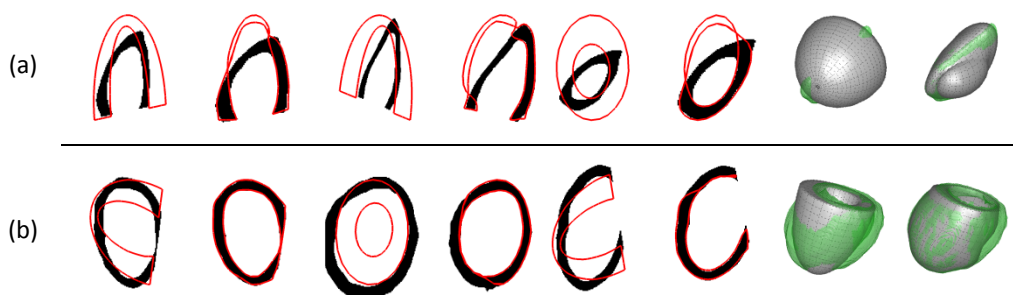


Figure 96: Cross-section of the template and fitted meshes for cases SHF15 (a) and MAN12 (b)

To assess the influence of initial mesh definition, the same segmentation can be used as target for meshes with different initial element count. Three meshes with a varying number of wall elements (2, 3 and 4) were used. Figure 97 shows a summary of the results; the mesh quality is

expressed in terms of mesh degradation (left) and the percentage ratio of degenerate elements after and before the registration process (right). It is clear that the mesh fitting is basically unaffected by the density of the mesh. However the lower the number of elements in the mesh, the more degenerate the elements must become to conform to the target shape.

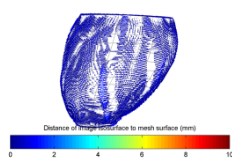
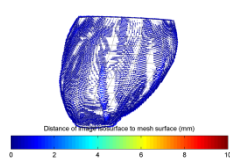
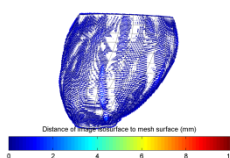
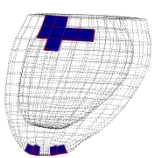
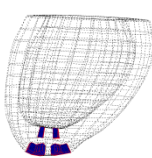
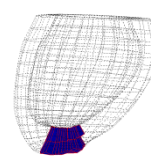
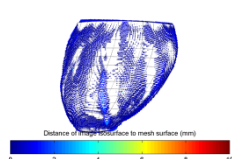
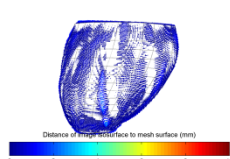
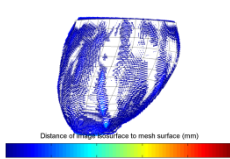
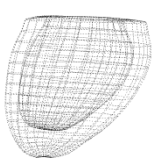
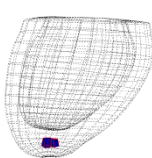
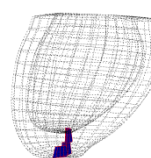
	4 ELEMENT WALL	3 ELEMENT WALL	2 ELEMENT WALL
CONFORMANCE	 <p>Distance of image surface to mesh surface (mm)</p> <p>Mean distance = 0.274 mm</p>	 <p>Distance of image surface to mesh surface (mm)</p> <p>Mean distance = 0.267 mm</p>	 <p>Distance of image surface to mesh surface (mm)</p> <p>Mean distance = 0.267 mm</p>
	 <p>Degradation = 1.049</p> <p>Degenerate elements = 3.1/5.9</p>	 <p>Degradation = 1.085</p> <p>Degenerate elements = 5.5/5.9</p>	 <p>Degradation = 1.115</p> <p>Degenerate elements = 14.7/14.7</p>
QUALITY	 <p>Distance of image surface to mesh surface (mm)</p> <p>Mean distance = 0.367 mm</p>	 <p>Distance of image surface to mesh surface (mm)</p> <p>Mean distance = 0.351 mm</p>	 <p>Distance of image surface to mesh surface (mm)</p> <p>Mean distance = 0.377 mm</p>
	 <p>Degradation = 1.123</p> <p>Degenerate elements = 0.5/5.9</p>	 <p>Degradation = 1.145</p> <p>Degenerate elements = 1.8/5.9</p>	 <p>Degradation = 1.167</p> <p>Degenerate elements = 8.7/14.7</p>

Figure 97: Comparison of the fitting distances and mesh degradation achieved when using initial templates with different numbers of elements.

The following table summarises the results for the four element-thick meshes using the protocols discussed in Table 20. The columns labelled “d” show the surface distances in mm and the columns labelled “q” show the mesh quality degradation.

CASE ID	ALIGNMENT		CONFORMANCE		QUALITY	
	d	q	d	q	d	q
KCL01	0.5873	1.0147	0.4107	0.9960	0.5133	1.0273
KCL02	0.6193	1.0495	0.3897	1.0359	0.4949	1.0655
KCL03	0.5628	1.0346	0.3872	1.0282	0.4510	1.0493
KCL04	0.5294	1.0143	0.3931	0.9835	0.4659	1.0281
LDN01	0.6068	1.0635	0.3123	1.0104	0.4309	1.0768
LDN02	0.5008	0.9268	0.4567	0.8833	0.5008	0.9268
LDN03	0.5962	1.0614	0.2734	0.9158	0.3727	1.0773
LDN04	0.4229	1.0457	0.2670	1.3181	0.3630	1.0569
LDN05	0.4799	1.0660	0.3044	0.9856	0.4248	1.0741
LDN06	0.5542	1.0334	0.3085	0.8649	0.4350	1.0412
LDN07	0.6688	1.0054	0.3731	1.0806	0.5737	1.0211
LDN08	0.5081	1.0234	0.3107	0.9983	0.4312	1.0333
LDN09	0.5342	1.0105	0.3242	0.9489	0.4413	1.0230
LDN10	0.4549	1.0046	0.2601	0.8618	0.3766	1.0154
LDN11	1.3704	0.8168	1.3002	1.0615	1.3704	0.8168
LDN12	0.5176	1.0412	0.3016	0.9582	0.4239	1.0576
LDN13	0.4734	1.0622	0.2972	0.9963	0.3990	1.0699
LDN14	0.4270	1.0360	0.2957	0.8394	0.3650	1.0543
LDN15	0.4473	1.0126	0.2772	0.9792	0.3785	1.0223
LDN16	0.4481	1.0393	0.2494	0.9949	0.3621	1.0522
LDN17	0.7806	1.0558	0.3053	1.0305	0.4331	1.0649
LDN18	0.5780	1.0308	0.3917	0.8474	0.4836	1.0419
LDN19	0.4833	1.0122	0.2883	0.9626	0.3807	1.0255
LDN20	0.5291	1.0410	0.3274	0.9405	0.4397	1.0510
MAN01	0.5515	1.0516	0.3822	0.9837	0.4656	1.0580
MAN02	0.5877	0.9864	0.3084	0.9427	0.4602	0.9973
MAN03	0.5334	1.0869	0.3705	1.8304	0.4620	1.0953
MAN04	0.5617	1.1092	0.3422	1.0085	0.4606	1.1150
MAN06	0.6010	1.0858	0.3650	1.0347	0.4989	1.0940
MAN07	0.6385	1.0696	0.3048	1.0379	0.4238	1.0816
MAN08	0.5600	1.0726	0.3032	1.0318	0.4018	1.0876
MAN09	0.4502	0.9503	0.2697	0.9231	0.3787	0.9705
MAN10	0.6547	1.0582	0.3568	1.0268	0.4731	1.0684
MAN11	0.6021	1.0728	0.3889	1.0137	0.5116	1.0792
MAN12	2.3284	0.0211	2.2274	0.9864	2.3284	0.0211
MAN13	0.4905	1.0524	0.3390	0.8652	0.4297	1.0658
MAN14	0.7222	1.1015	0.2767	1.0681	0.4230	1.1165
MAN15	0.5649	1.0767	0.3062	1.0420	0.4350	1.0851
MAN16	0.4568	1.1139	0.2743	1.0486	0.3666	1.1225
MAN17	0.7564	1.0014	0.4157	0.9845	0.5614	1.0134
MAN18	0.4336	1.0881	0.2897	1.0371	0.3672	1.0973
MAN19	0.7294	0.9931	0.3481	0.9747	0.5792	1.0147
MAN20	0.5339	1.0508	0.3204	1.0208	0.4096	1.0604
MAN21	0.4661	1.1309	0.2774	1.0824	0.3821	1.1367
SHF01	0.4829	1.0404	0.3215	0.9699	0.4031	1.0529

SHF02	0.5679	0.9965	0.3151	0.9492	0.4365	1.0096
SHF03	0.6257	1.0651	0.3758	1.0248	0.4882	1.0775
SHF04	0.4862	1.0737	0.2835	1.0102	0.3636	1.0805
SHF05	0.5277	1.0314	0.3559	0.8837	0.5277	1.0314
SHF06	0.5526	1.0723	0.3393	0.9421	0.4656	1.0831
SHF07	0.5168	1.0238	0.3009	0.8663	0.4198	1.0344
SHF08	0.6051	1.0537	0.2998	0.9757	0.4401	1.0618
SHF09	0.5107	1.0020	0.2825	0.9330	0.4022	1.0106
SHF11	0.5339	1.0352	0.3482	0.9283	0.4591	1.0440
SHF12	0.8334	1.0213	0.7785	0.8320	0.8334	1.0213
SHF13	0.4444	1.0085	0.3211	1.2103	0.4351	0.9907
SHF14	0.7079	1.0757	0.3314	0.9791	0.5143	1.0816
SHF15	2.6157	1.3078	2.6158	1.3069	2.6157	1.3078
SHF16	0.4900	1.0719	0.3111	1.0046	0.4031	1.0815
SHF18	0.5561	1.0257	0.3301	0.9351	0.4636	1.0377
SHF19	0.5516	1.0042	0.3087	0.9645	0.4375	1.0143
SHF20	0.3733	1.0078	0.2052	0.8970	0.2700	1.0272
SHF21	0.5281	1.0672	0.3108	0.9900	0.4431	1.0708
VOLK1	0.4934	1.0124	0.3829	0.9961	0.4548	1.0252
VOLK2	0.4278	0.9539	0.3647	0.9184	0.4108	0.9729
VOLS1	0.4882	1.0124	0.3274	0.9698	0.4126	1.0291
VOLS2	0.4889	0.9026	0.2755	0.8722	0.3985	0.9282
<b>MEAN</b>	<b>0.6166</b>	<b>1.0230</b>	<b>0.4083</b>	<b>0.9974</b>	<b>0.5168</b>	<b>1.0332</b>

Table 20: Detailed results (Figure 90, Figure 91 and Figure 92)

## 6.6 Conclusions

This chapter has presented a pipeline for generating quality personalised finite element meshes of the left ventricle, using the registration algorithm presented in chapters 3 and 4. The registration algorithm is based on ShIRT, however ShIRT is not an ideal tool for the registration of images like the ones used in this thesis. As the binary images have less information to drive the registration, it relies heavily on the regularisation terms. However, because the regularisation used in ShIRT is defined in Cartesian space, it over-penalises the deformations required for the morphing.

The results shown in Figure 98 were achieved early on in the project using ShIRT. The model (green) has a much thinner wall than the patient image (red). The middle picture (below) shows the images after registration and the fit (yellow) is quite good. However the mapping is very distorted (right hand side below) and would not be good for mapping a volume mesh based on the model. The method presented in this chapter represents a major improvement of the algorithm for applications of this nature.

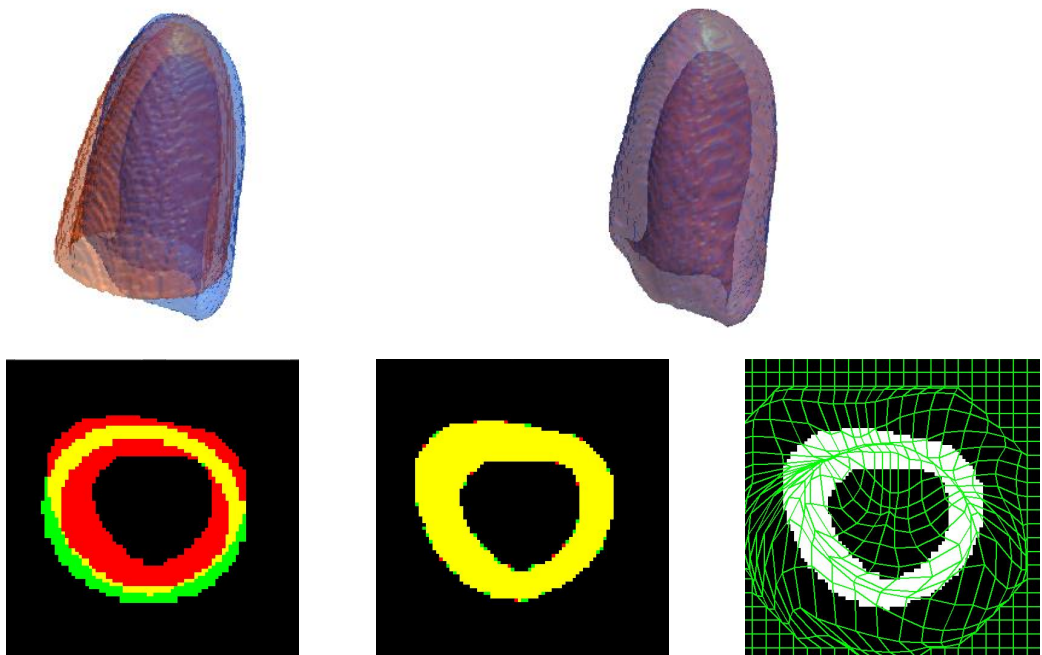


Figure 98: Binary images before and after registration using ShIRT, 3D (top) and slice (bottom) views

The processing pipeline is similar to that described in the leading published work [87] with the differentiation that the registration and mesh optimisation stages are consolidated into a single step. Results are comparable with those presented in the literature in terms of mesh fitting and mesh quality. For instance, using the four element thick templates, the fitting accuracy, based

on the mean distance between the mesh and segmentation surface, was between 18.2% and 82.2% of the mean voxel size with a mean fitting accuracy of 37.5% (this is a nominal distance of 0.385 mm).

Comparatively, in [87] the authors report a fitting accuracy of 0.643 mm for the segmentations of comparable voxel sizes (based on the results shown in Table 1 of their publication). This fitting corresponds to 53% of the mean voxel size of the segmentations. In [33] a median error of 1.18 mm is reported in section E and mean fitting accuracies of values between 0.9 and 2.2 mm are reported in Figure 6 (the voxel sizes are not stated). It should be noted that the linear meshes used in this study have less degrees of freedom and are less able to capture complex topologies than the cubic Hermite meshes. Unlike [87], the improvements shown in this work do not require special treatment of the trans-mural element edges (as described Section C).

A weakness of this chapter is that the only way to determine whether the improved mesh quality reported in this chapter results in improved performance and stability of the simulation is by extensively testing the meshes by running computer simulations with them (as in [32]).

The degradation measure would have provided a better insight to the process if the thickness of the template meshes had been specific to each segmentation. For instance, Figure 99 shows the same segmentation dilated to simulate a thicker myocardium; the thicker segmentation obviously produces a better mesh. However, the template mesh is relatively thin and so the degradation measured is larger for the thicker target.

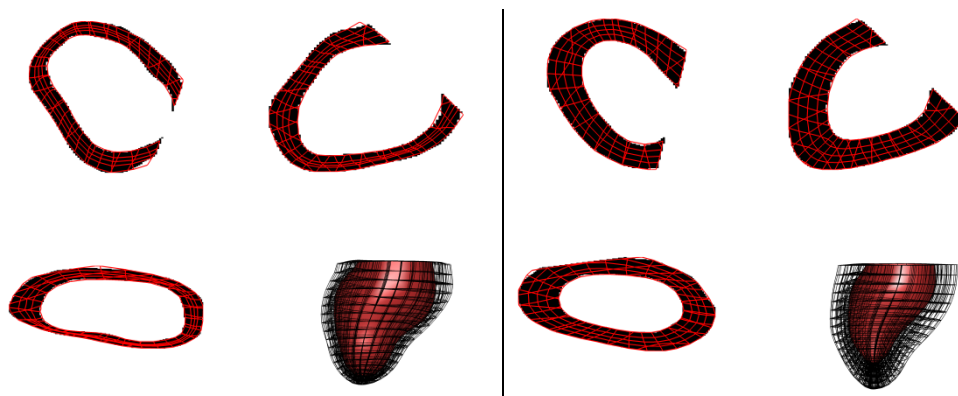


Figure 99: Resulting mesh fitting for thinner (left) and thicker (right) target geometries using same sized meshes.

In terms of execution performance, registration using the alignment and conformation protocols takes between 2-3 minutes whilst using the quality protocol takes between 20-30 minutes, depending on the case. For comparison, the process described in [87] takes 7-9 minutes overall, but does not use a Jacobian-based metric to optimise the mesh quality. In practical terms, both

processes take considerably less time than the multi-physics simulations that the meshes are used for (which may take up to weeks to complete).

In the process used in this work, post-processing the segmentations does run the risk of distorting the segments as it might remove real sharp corners, but such sharp features are unlikely to be found in cardiac tissue. There is an argument that this produces meshes that are less faithful to the true patient geometries, however it is not clear that these smoothing errors introduced are larger than the ones present in the imaging and segmentation processes already. It is demonstrable that smoothing the images makes the fitting more robust.

Limitations of this work include:

- The code has currently been developed only for trilinear elements, compared to the state-of-the-art process which uses Hermite cubic elements.
- The use of a first order Taylor series expansion in the transformation model of the registration process, which limits the range of the reliability of this process to small deformations.
- The mesh quality regularisation term is a linearisation of a differentiable function that is similar to, but not quite the same as, the standard quality metric of the ratio of maximum to minimum Jacobian in the element. The reason for this choice is that the linear analytical derivative fits well with the linear least squares regularisation process and is implemented efficiently in exactly the same way as the Laplacian term.



# CHAPTER 7

## IMAGE-DERIVED WALL MOTION FOR HAEMODYNAMIC ANALYSIS

This chapter presents another application of the registration algorithm which is the derivation of flow at the ascending aorta based on image-derived contraction of the ventricular wall. Section 7.1 presents the background work and motivation for this kind of analysis. The method is described and data used is presented in section 7.2. Section 7.3 describes the preparation of data for analysis, using the image registration method presented in chapter 3. Section 7.4 describes the use of the image derived wall motion to simulate the flow of blood from the ventricle into the aorta and section 7.5 summarises the results of this process. A discussion of the overall method and conclusions from this chapter are presented in sections 7.6 and 7.7, respectively.

### 7.1 Motivation

A comprehensive review of the recent state-of-the-art in patient-specific modelling of cardiovascular mechanics is presented by Taylor and Figueroa [133]. One important issue is the description of the boundary conditions. Fluid-Solid Interaction (FSI) is used to couple the arterial mechanics with the haemodynamics. The registration methods described in this thesis could be used to determine the motion of the surface of the wall, for the ventricle and vessel. The registration field describes the displacement of every point on the wall. Although it is not guaranteed that this displacement field, on a point-by-point basis, is consistent with the mechanics of the wall, the surface motion should be accurately represented.

A particular area of interest is that of aortic flow simulation because it has high clinical value. A common strategy for the haemodynamic simulation of the aorta using a rigid-walled approach is as follows [26], [134]:

- The simulation workflow for aortic flow starts with a segmentation of the aorta to produce a fluid mesh;
- The flow-pressure conditions at the outlet boundaries are regulated using Windkessel models;
- At the inlet, a time-varying, plug-like velocity condition is imposed.

Windkessel models are zero dimensional functions that describe the relationship between flow and pressure in large vessels. They are described using electrical analogies, in its simplest

representation it uses a resistor to model impedance to the flow (such as friction) and a capacitor to describe vessel compliance. [135]

To obtain the magnitude of the inlet velocity, the mean velocity is often measured from a 2D+t phase contrast MR image, measured at a cross-section across the vessel. Figure 100 summarises this simulation workflow [136].

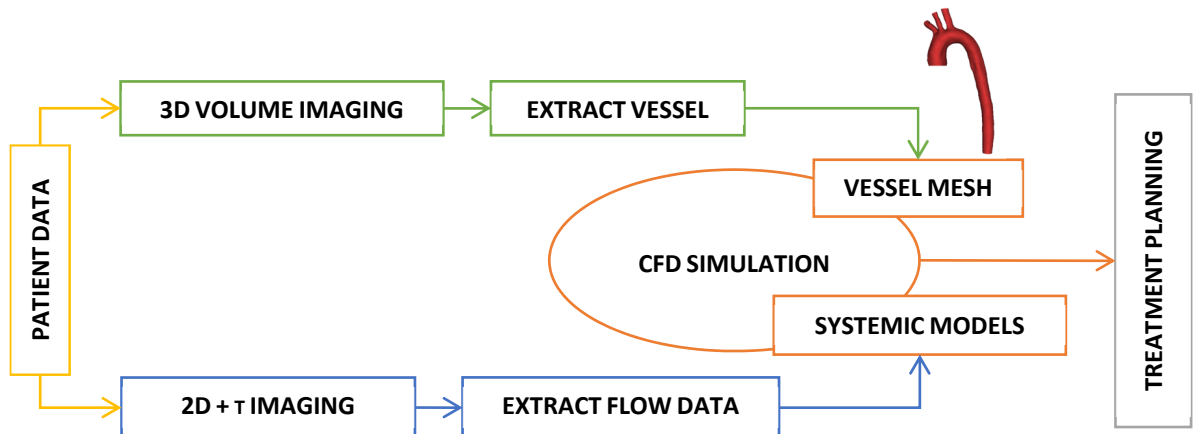


Figure 100: Overview of a typical aortic simulation workflow, highlighting the 3D mesh [136]. The imaging modalities used include 3D MR and CT for extracting the vessel geometry, and MR phase contrast for the inlet boundary condition and tuning the 0D systemic models.

In this chapter, a methodology to replace the inlet plug velocity boundary condition with a more accurate representation based on registration of time-series segmentations of the ventricle is developed and illustrated. Peters et al. present a novel method for segmentation of the ventricle and (open) aortic valve from CT data, and compare the pressure drop computed by computational fluid dynamics with that computed from the simple Bernoulli equation [137]. The inlet to the valve is represented as a cylinder, extruded along the axis of the ventricular inflow tract, with a temporally varying plug flow at the inlet. The temporal profile of the flow was determined from a sparse set of ventricular segmentations performed by the authors of the paper. A natural question that arises is whether a more comprehensive simulation that includes the motion of the left ventricle might have any impact on the computed pressure gradients. The ventricular motion will produce a velocity distribution, including out-of-plane components, that is not the same as the plug flow applied by Peters et al.

## 7.2 Method overview

Peters et al. used a deformable mesh from which the change in ventricular volume could be determined directly, but in the context of this thesis the only requirement is a series of binary segmentations. The registration field is determined by registration of the time-series segmentations using the methods developed and illustrated in chapters 3 to 6. The proposed pipeline begins with the time series segmentation of the pulsating left-ventricular volume, however this is achieved.

In order to describe the pulsation of the ventricle in parametric space, a ventricular mesh is built and conformed to the pulsating segmentation. The ventricular mesh utilised was described previously in chapter 6; this requires the segmentation to be cut below the valves. The template mesh is registered to the first time step. The conformed mesh resulting from the first time step is then registered to the following time steps; effectively tracking the surface of the ventricle through time. The resulting pulsating ventricular sac is stitched to the surface mesh of the valve and ascending aorta. An easy way to perform the haemodynamic analysis of the system is to use the fluid-structure interaction (FSI) facility in the (commercial) ANSYS software suite. The fluid mesh is defined, bounded by a shell that includes the ventricular wall. The motion of the wall is imposed directly at every timestep, based on the registration field. This is not true FSI, but uses conveniently the coupling functionality of the commercial software. The output of the simulation is a description of the fluid flow, which will support the comparison of the plug-inlet assumption and the simulated flow. The modified work flow is shown in Figure 101.

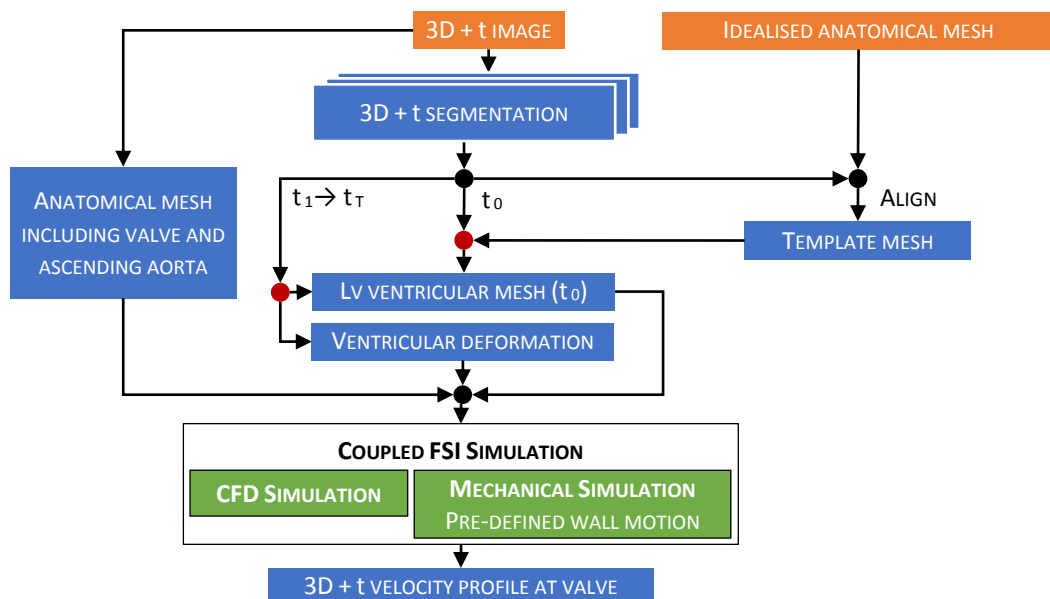


Figure 101: Schematic of the process described in this chapter. The anatomical mesh is static, in contrast with the ventricular region of the mesh that will deform at each time step.

### 7.3 Registration

The toolchain to determine the ventricular deformation starts from the segmentation data and surface mesh, shown in Figure 102. The segmentation data consists of ten binary images of the left ventricle and aortic valve, for this analysis only the six segmentations that correspond to ventricular systole are used. The surface mesh is formed of triangular elements and represent the interior ventricular wall, aortic valve and a section of the ascending aorta. The inter-ventricular cavity is closed.

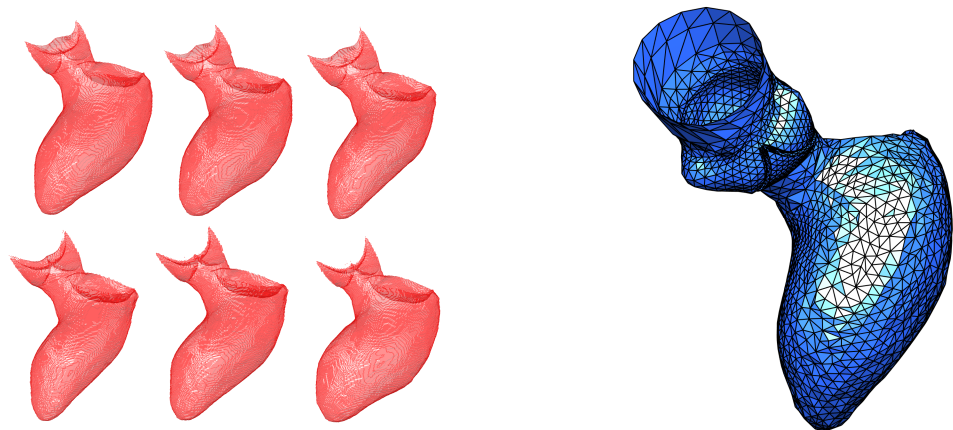


Figure 102: 3D + t segmentation of the left ventricular volume and aortic valve (left) and surface mesh of the ventricular volume, aortic valve and aorta (right). One volume dataset, courtesy of Peters and Weese

Before the ventricular surfaces are tracked through time, the segmented images need to be prepared for registration. To extract the region from the segmented image that matches the template mesh, three points are used to define a cutting plane. The preferred orientation of the plane is normal to the valve-to-valve direction (note that in the data used in this analysis, the atrio-ventricular valve is closed). The truncated region is then dilated using a kernel 20 pixels wide; the original truncated image is subtracted from the dilated region to create a hollow volume. A template mesh is built to the dimensions of the first time step. This process is shown in Figure 103; the top row shows the raw segmentations, the bottom shows the dilated, hollow volumes. This setup is now the same as the meshing of the myocardium in Chapter 6. The size of the segmentation images is  $512 \times 512 \times 278$ , so the processing time was significantly higher than in the chapter 6, taking around 30-40 seconds to process each image.

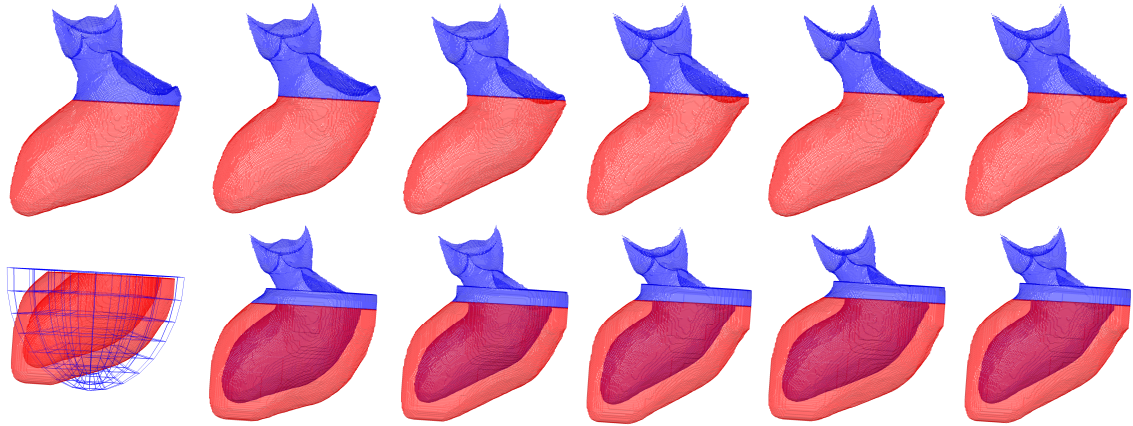


Figure 103: Processing the images, before (blue) and after (red) being truncated at the cut plane. The fitted mesh is shown on the first time step.

The registration is performed in two steps. Firstly, a highly regularised registration is performed to bring the template into alignment with the first time step. Then, the template is made to conform closely to the first time step using a weakly regularised registration. The resulting mesh is made to conform to the following time steps in an incremental fashion. Table 21 details the parameters used.

	FIRST CONFORM	TIME STEP REGISTRATION
Tikhonov Weight (smoothness)	10 (normalised)	1 (normalised)
Tikhonov Weight (quality constraint)	-	-
Memory	ON	ON
Sampling points	4	4
Smoothing kernel neighbourhood	2	0
Typical number of iterations	500	50

Table 21: Summary of the registration parameters used for tracking the ventricular wall.

The images are relatively well aligned between time steps so there is no need to realign them after the first time step. The registration of all time steps takes in the region of 45 minutes to complete. The average surface distance of the registered surfaces is  $0.291 \text{ mm} \pm 0.282 \text{ mm}$  (the voxel dimensions are  $0.48 \times 0.48 \times 0.45 \text{ mm}$ ). The results can be seen in Figure 104.

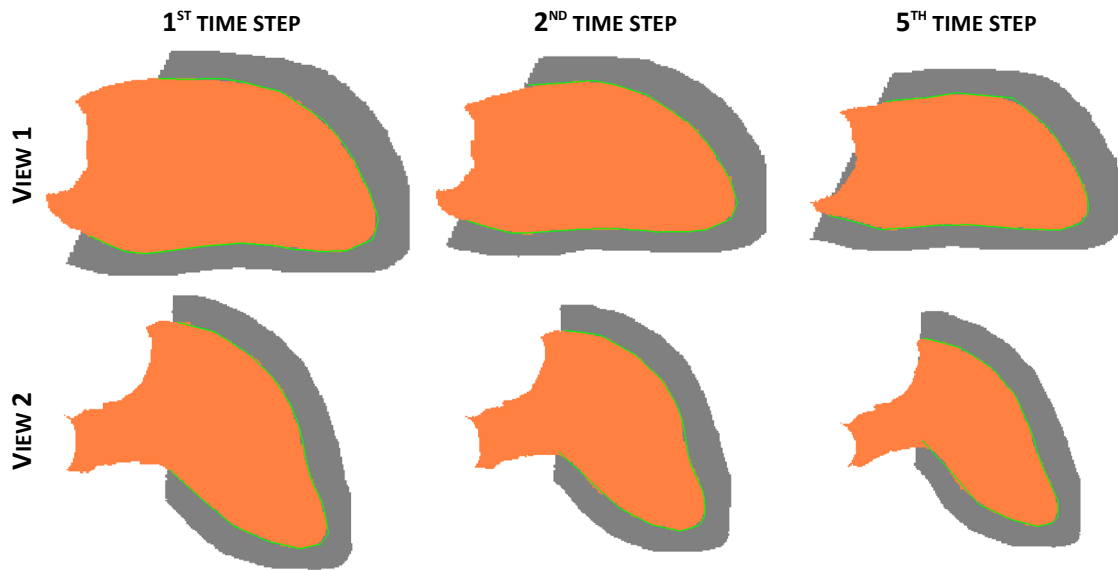


Figure 104: The original (orange) and processed (grey) images, and the inner surface (green) of the registered mesh

## 7.4 Simulation setup

### 7.4.1 Multi-field simulation

Fluid-structure interaction (FSI) are simulations where the physics of a fluid and a solid structure are simulated without disregarding the interactions that may occur between each domain (mesh, material properties, solver properties, boundary conditions). A number of ways the domains interact (or coupling) options exist. An important aspect of this work is that whilst the CFD analysis is required to simulate the blood flow, the mechanical simulation is not required. In full FSI, a mechanical simulation is set up using appropriate material properties and boundary conditions, and the deformation of the solid structure is simulated. Using the image-derived measurements from section 7.3, the structural simulation is avoided, saving computational time as well as increasing the accuracy of the model by avoiding making assumptions of the mechanical properties of the ventricular wall. The software suite commercialised by ANSYS were selected to manage the coupling of the fluid simulation with the deformable model [138].

ANSYS offers a number of options for running FSI simulations [38]. For this work, a multi-field analysis using code coupling was used. This implementation of coupled analysis allows defining the CFD and FEA simulations independently (multi-field) and the multi-field stagger (MFX) transfers information across the environments. This option was chosen as it offers the most flexible user control. The ANSYS Mechanical solver is set-up to behave like the main (or “master”) process and communicate with the CFX simulation (the “slave” process) to form a two

way coupling (see Figure 105). Of course, the mechanical solver simply imposes the image-derived displacement resulting in a one-way coupling.

The other options offered by ANSYS do not allow defining the domains separately, or do not allow manipulating the solving routine (to by-pass the mechanical solve using the calculated displacements).

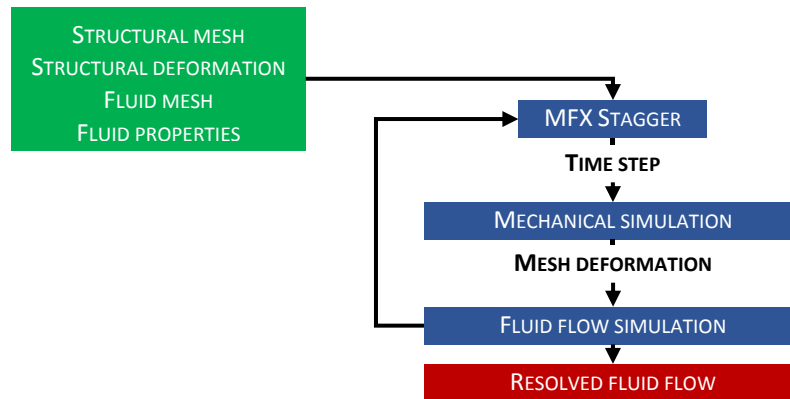


Figure 105: Flowchart showing the coupling set-up for the simulation method used in this chapter.

## 7.4.2 Fluid flow simulation

### 7.4.2.1 CFD mesh

The fluid domain is the volume enclosed by the surface of the ventricular volume, taken to be static, and the dynamic ventricular sac (see Figure 101 and Figure 102). One approach towards the simulation is defining a set of vectors (based on the registration) and allow the coupling software to interpolate the correct displacement on the static ventricular volume. The interpolation is required because the displacement vectors are not defined at the same location as the nodes of the fluid volume; this becomes unstable if the distances between the surface mesh and displacement vectors are large or irregular. To ensure compatibility of the displacement, an alternative approach is removing the ventricular region of the surface mesh and replacing it with the dynamic mesh altogether. The resulting fluid domain is enclosed by the surface, taken to be static, which defines the ventricular inflow tract, the open valve and the ascending aorta, and the dynamic ventricular sac.

To mesh the volume, a continuous surface is required. First, all nodes and faces on the static surface mesh that are below the cutting plane are removed. Then, the nodes at the base of the dynamic ventricular sac are translated towards the nearest node or edge in the static surface of the base. Finally, a closing surface is created at the opening in the ascending aorta (this has been removed from Figure 106 for visualisation purposes).

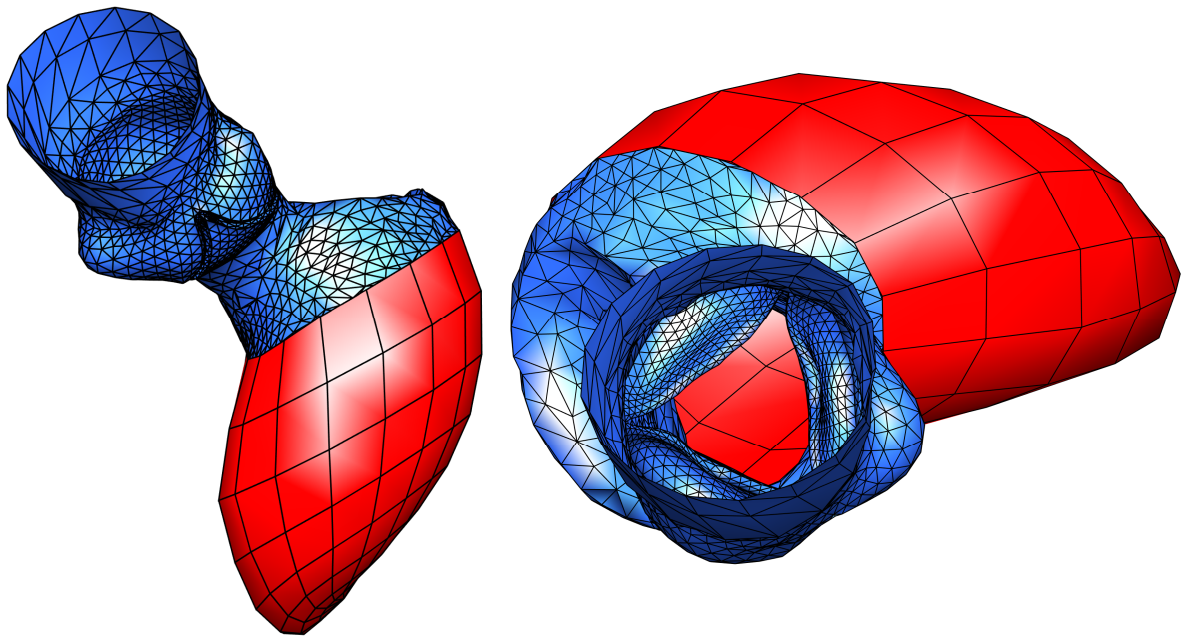


Figure 106: Side and top views of the surface mesh, including the moving ventricular wall (red), before meshing the inside volume for the fluid simulation.

The closed surface is volume meshed using ANSYS ICEM, with a target mesh element edge size of 2 mm [139]. The element size is a trade-off between stability during the fluid simulation and during the deformation at each time step. A total of 251'660 tetrahedral elements were produced.

Figure 107 is similar to Figure 104 however in this case, the static section of the mesh (coloured blue in both Figure 102 and Figure 106) has been added to emphasise the error introduced by the processing, in particular, the joining of the static and dynamics (ventricular) surface meshes.

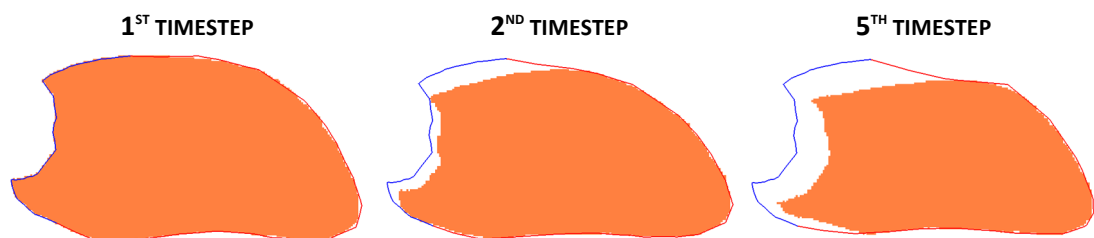


Figure 107: Resulting enclosing volume for the simulation and the segmentation (orange) at different timesteps. The mesh is coloured by section: base (blue) and walls (white)

#### 7.4.2.2 CFD Simulation

ANSYS CFX solves the Navier-Stokes and continuity equations to resolve the fluid flow in the system using a 2<sup>nd</sup> order Backwards Euler scheme. The analysis reported here is a laminar solution, turbulence was not modelled. The blood was modelled as incompressible fluid of



density  $1'056 \text{ kg m}^{-3}$  and with a dynamic viscosity of  $0.0035 \text{ Pa s}$ . A zero-pressure condition was assumed at the outlet (ascending aorta), and a no-slip condition was imposed at all walls (aorta, aortic valve and ventricle). The initial condition of the fluid was set to zero velocity fluid and pressure, a stationary mesh and zero pressure at the outlet. The displacements are prescribed at the surface nodes of the dynamic part of the ventricle, and CFX applies them on all the nodes using a diffusion model. The timing options are dependent of the multi-field coupler, itself based on the data available; for this analysis, the simulation time step was 10 milliseconds. The convergence criteria was set at  $10^{-5}$  (i.e. five orders of magnitude), and a full set of results was exported at each time step.

### 7.4.3 Mechanical simulation

#### 7.4.3.1 Finite element analysis mesh

The tracked templates conform to the dilated version of the ventricular volume (analogous to the surrounding muscle) meaning that the inside surface of the template corresponds to the surface of the ventricular volume. Extracting the nodes of the inside surface is a trivial task and describes the displacements of the volume's surface. The surface mesh has 128 quadrilateral elements.

#### 7.4.3.2 Finite element analysis simulation

The mechanics simulation is programmed using the ANSYS Parametric Design Language (APDL). The script requires commands to describe the mechanics simulation and the multi-field coupling. The scripting strategy was chosen on the premise that creating a Matlab script to generate the APDL script is a trivial task.

The main script, shown in Figure 108, firstly loads the geometry at the first time step, described as a thin shell, then specifies a displacement function at each node. The displacement function is specified at each known time step and is independent for each degree of freedom; the solver uses linear interpolation between the specified times. Lastly, the multi-field variables specify the transient set-up (shared between the fluid and structural simulation) and the load transfer parameters.

/input,ansys_geom,inp	Load the geometry
mp,ex,1,1	Define elastic moduli (not relevant)
mp,nuxy,1,0.4	Define minor Poisson's ratios (not relevant)
keyopt,1,1,1	Define material based on previous properties (not used)
sectype,,SHELL	Define the elements as "shell" type elements
secdata,0.1	Give all elements a thickness (not used)
/solu	
/input,disp_array,inp	Load displacements in the form or an array
/input,disp_solve_single,inp	This applies the displacements on the nodes
allsel	
sf,all,fsin,1	Select nodes for FSI interface, call these fsin
MFAN,ON,MFAX	Set up ansys mfx
MFBU,ON,50,	Options for bucket interpolation (scaling option)
MFTO,OFF,1e-005,ABS	Options for MFLC (bucket tolerance, tolerance type)
MFCI,100,0.1,	Options for conservative interpolation (pixel resolution, separation factor)
MFCL,MFPS	Sequential solve
MFPS,group1,ANSYS	Name the mechanical domain
MFPS,group2,CFX	Name the fluid domain
MFSO,group1,group2	Sort the solve order
MFRE,ALL,1	Relaxation value
MFTI,0.4,	Solution end time
MFDT,0.01,0.01,0.01,0	dts (initial dt, min, max, "time step carry over" on/off)
MFRS,0,SING	Receive simulation time from coupler
MFIT,1,1,1,	Stagger iterations (max,min,target)
MFCO,ALL,0.01,	Convergence criteria (all, value)
MFOUTPUT,1	Write output (frequency: at every time step)

Figure 108: ANSYS Mechanical and Multi-field solver script. Note that the displacement values and the commands to apply them are stored in separate files to aid legibility.

It makes no sense to run the simulation of the whole cardiac cycle because the valve is static and because for the filling phase the aortic valve should close and the mitral valve open. For this reason, only the first 50% of the cardiac cycle was simulated; this is long enough to capture the entirety of ventricular systole which typically accounts for 40% of the cardiac cycle. The cardiac period for this simulation was 0.8 seconds.

To ensure smooth transmission of the mechanical forces to the fluid, the displacements at each node (only known at ten time steps during the cardiac cycle) are interpolated using cubic splines. Spline interpolation ensures continuity of the second derivate of the displacements. There is a minimal change in the integral value of the displacement function, so the function is scaled to preserve the integral value. The mesh is brought to a halt after systole.

## 7.5 Results

The flow at the ascending aorta is obtained by integrating the velocity at the outlet. Figure 109 shows the simulated flow over time, expressed in the typical clinical unit (L/min). A typical flow at the ascending aorta for a young adult is shown to compare with the result.[140] The simulation only comprised of the first half of the cardiac cycle; ventricular systole typically last a 3<sup>rd</sup> of the cardiac cycle. The peak flow was 15.18 L/min. [140]

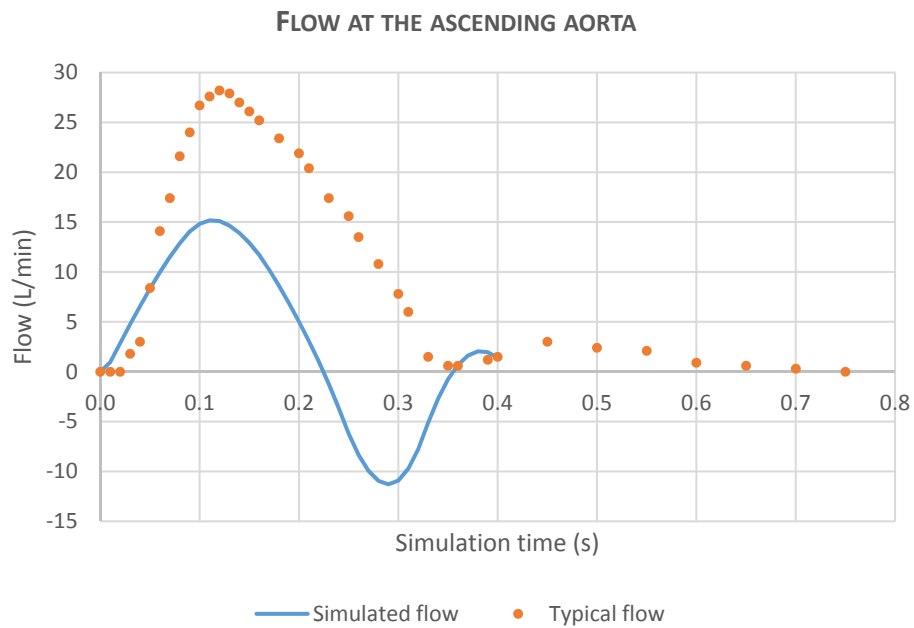


Figure 109: Resulting flow at the ascending aorta

The peak computed flow at the outlet is 15.18 litres per minute at time 0.11 seconds. It must be noted that the static section of the ventricle would contribute further to the outflow, in particular, increase the maximum flow and stop the excessive back flow seen early in the cycle. Figure 110 shows that there is an asymmetric flow profile at the outlet and a significant back flow. The velocity vectors, shown in Figure 111 are used to check the expected patterns in the fluid are captured by the simulation. The streamlines of the velocity field in Figure 112 show a region of recirculation distal to the valve during the cardiac cycle that increases in magnitude after the time of peak flow.

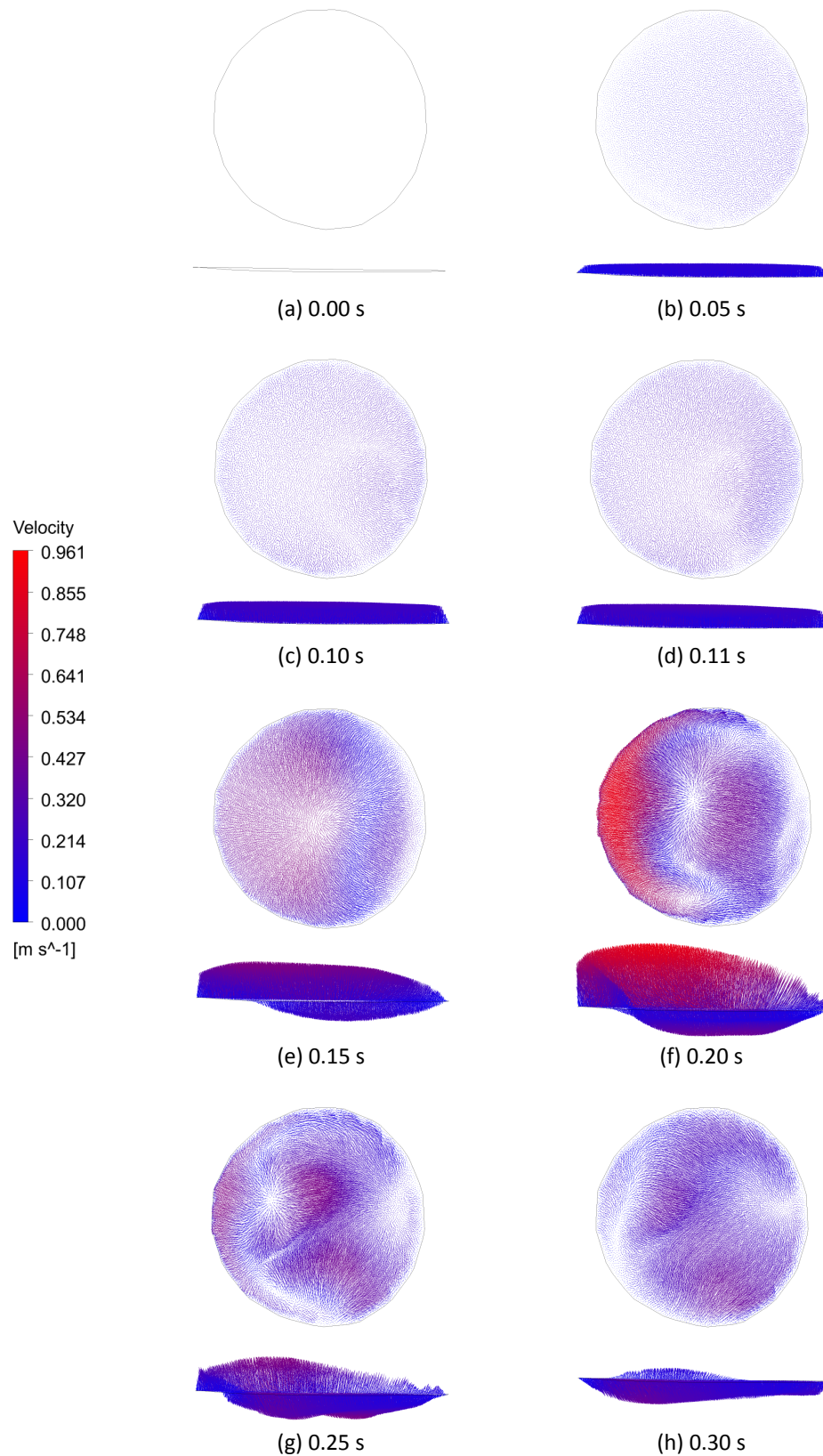


Figure 110: Velocity profile at the ascending aorta. Panel (d) shows the instant of maximum flow. The cross-sectional flow is clearly not plug-shaped. The side views of the outlet show the is a large amount of flow going back into the ventricle.

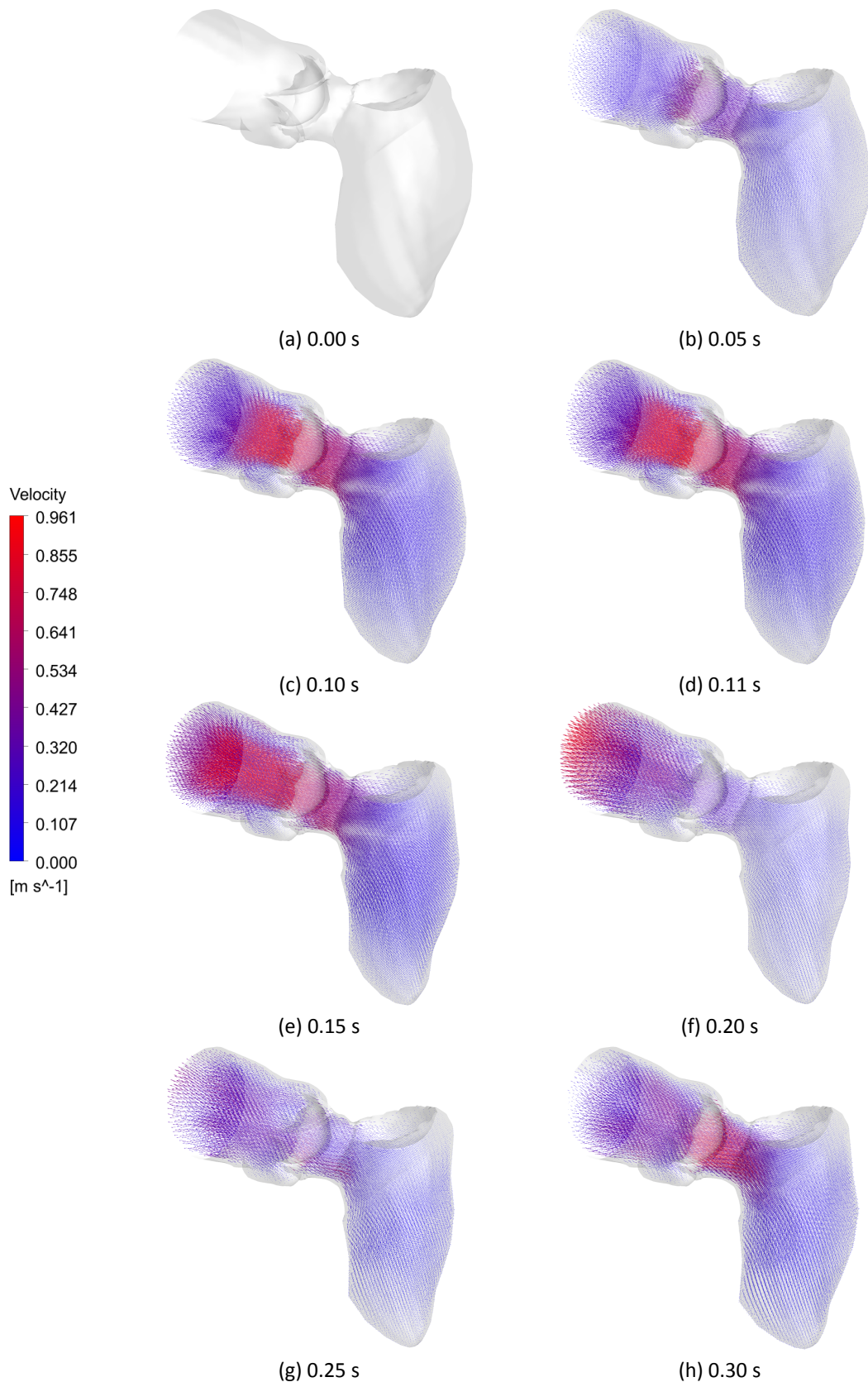


Figure 111: Velocity vectors in the fluid domain. Panel (d) shows the instant of maximum flow. The velocity distribution provides a visual proof the simulation is behaving correctly; in particular, the flow is pushed directly out into the aorta and the acceleration of flow near the narrowing at the outlet

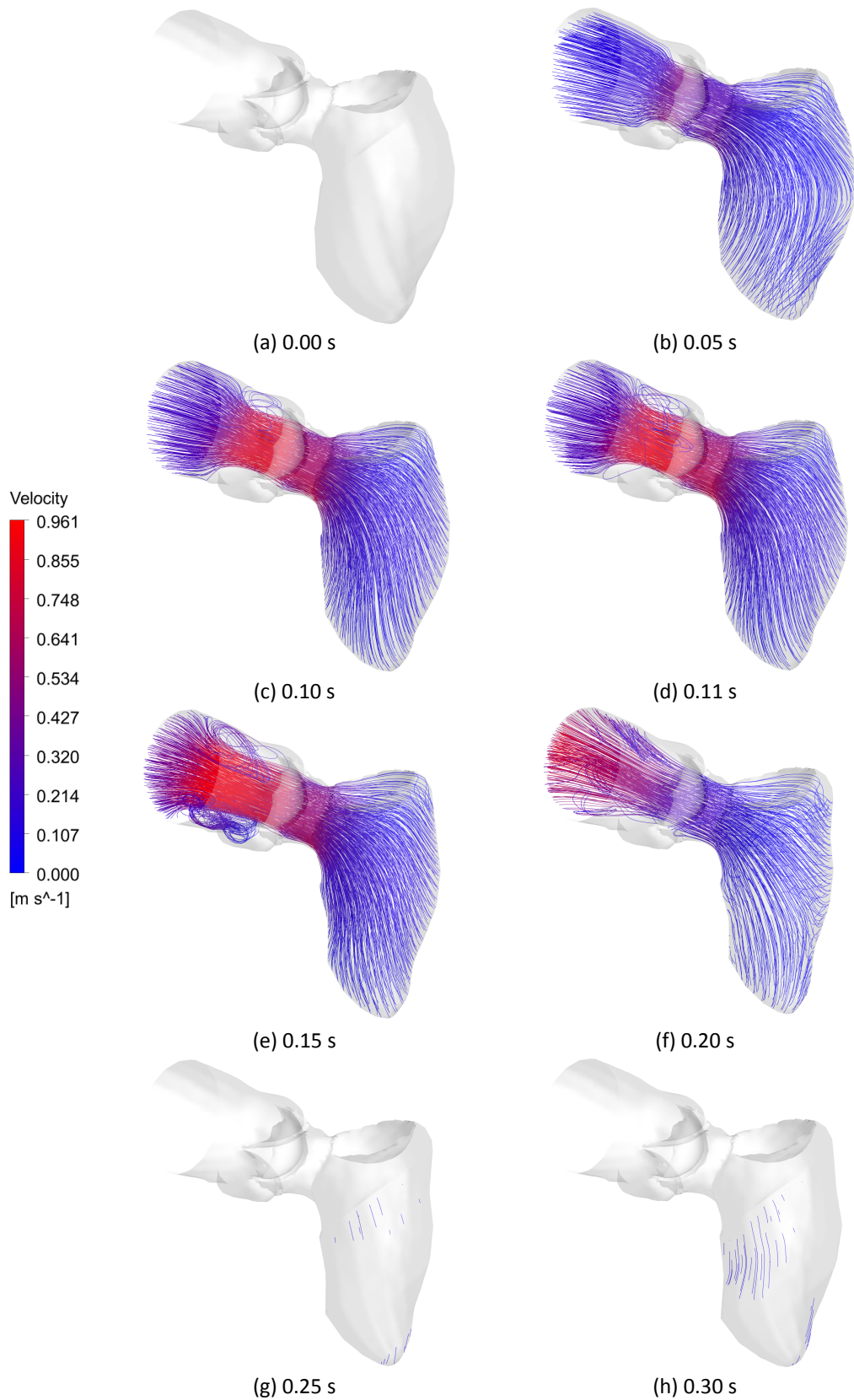


Figure 112: Streamlines in the fluid domain (initiated from the ventricular wall). Panel (d) shows the instant of maximum flow. The streamlines provides a visual proof the simulation is behaving correctly; in particular, the flow is pushed directly out into the aorta and there is an area of recirculation behind the valve.

## 7.6 Discussion

The purpose of this chapter is to report the development of, and to demonstrate, a protocol for the application of the registration scheme that is the main focus of this thesis to an important practical application. Further work is required to interpret the analysis results, and to compare and contrast with those obtained from a simulation of a reduced domain. Nevertheless it is noted that the overall flow features appear to be captured by the simulation, including a speed-up at the valve (Venturi effect), a region of recirculation around the valve and the transient acceleration of the flow from the contracting wall towards the outlet.

The main limitations of this exercise are that the ventricular volume is only partially tracked, and that the valve remains open throughout the simulation. This leads to both the shape of the flow-time signal and the instantaneous maximum flow to be atypical for human aortic flow, as seen in Figure 109. It was hypothesised that modelling ventricular contraction using only part of the ventricle was not a bad assumption because whilst the base of the ventricle moves significantly, it appears to contract little relative to the rest of the ventricle. This assumption affects the ejection volume measured at the ascending aorta and is clearly not appropriate. Should the full ventricular wall be simulated, one would expect the observed total volume and maximum flow to be closer the typical aortic flow shown in Figure 109, and for the duration of systole to be briefly extended (as seen in the flow-time signal). It is hypothesised that whilst the open valve clearly affects the shape of the flow-time signal, this effect is magnified by the previous limitation. The simulation of the closing valve using the method presented in this chapter is considerably more complicated due to the imaging data for valves being of poor quality.

The chosen fluid mesh element size is relatively coarse; Table 22 summarises results with refinement of the mesh from a very coarse model to a moderately coarse one, which increases the peak flow at the outlet by 0.14% but costs 225% more in terms of computational time. No mesh refinement at the wall was employed. For the purposes of this thesis the coarse mesh analysis serves to illustrate the protocol, but a more comprehensive mesh sensitivity test will be required to evaluate the effect on pressure gradient across the valve.

MESHING PARAMETERS (TARGET EDGE LENGTH)		MESH STATISTICS		SIMULATION TIME
SURFACE	VOLUME	# NODES	# ELEMENTS	
5	5	3 366	15 985	3 minutes
4	4	6 322	31 367	4 minutes
3	3	14 272	74 283	8 minutes
3	2	46 133	251 660	26 minutes
2	1.5	106 970	597 010	73 minutes

Table 22: Details of the mesh sensitivity analysis

Figure 113 illustrates the computational requirements of the simulation (to the nearest minute) for each mesh density tested. It also shows the sensitivity to flow at the outlet to mesh density. In terms of absolute values, the difference in maximum flow seen in the sensitivity study is 0.15 litres per minute, or 0.97% of the maximum flow using the coarsest mesh. The change in flow can be explained by the change in volume of the fluid domain during the meshing process. The total fluid volume, plotted against element edge length in Figure 113 appears to converge below the 1 mm element edge length, which suggests it is likely that the maximum flow measurement will also converge to about 15.2 litres per minute. In any case, the magnitude of these changes in flow output are insignificant from a clinical perspective.

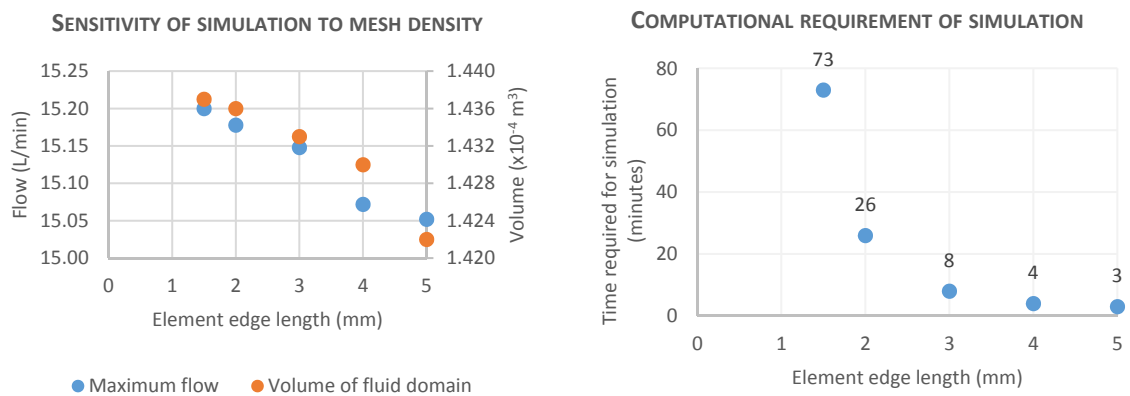


Figure 113: Sensitivity of results and computational requirements of the simulation for different mesh densities.



## 7.7 Conclusion

The complete simulation ran for 26 minutes and the processing pipeline is entirely automated except for the selection of the cutting plane (first step in the pipeline) and the post-processing analysis. This makes this pipeline truly convenient for routine clinical practice. The method still requires segmenting all phases in the MR image however as discussed in chapter 6, methods for automatic segmentation are becoming increasingly accurate and robust. Tracking the non-linear deformation of muscle is a complicated task, in this chapter this is not achieved as the true muscular deformation does not appear in the segmentation. However, as discussed in chapter 3, an extension to the registration method presented in this thesis is to work directly on the volumetric images, were the muscle is visible and theoretically trackable.

The output of the simulation can either be used to examine the cardiac performance of the patient or to produce improved boundary conditions for fluid flow simulations of the aortic arch. It is unclear at this stage whether the pipeline would benefit from extending the geometry to include the aortic arch or there is greater benefit in having separate simulations for each region. It is clear from the results obtained, as well as from clinical flow measurements (using phase contrast imaging) that the flow in the ascending aorta is far from plug-like so an inlet boundary condition produced using this method is a considerable improvement to the setup.

# CHAPTER 8

## CONCLUSIONS AND FUTURE WORK

### 8.1 Summary

This research programme was designed to address the challenges of construction of a work flow to introduce computational analysis and biomedical modelling into the therapy planning pathway, with the specific application of cardiac resynchronisation therapy (CRT). Chapter 2 has covered the general issues of work flow construction and management of clinical data from an integrated perspective (clinical, research, computing), with the end objective of the CRT application.

In the analysis of the work flow and the study of its operation on the study cohort it became apparent that the work flow could not be implemented into a routine clinical environment until a critical bottleneck had been resolved, namely the construction of a high quality mesh of the individual patient that would be numerically stable in the simulation analysis. Image registration is a natural choice of process to morph a template mesh onto a patient image, other choices include projection of mesh surfaces towards the boundaries of the segmentation; the advantage of image registration is that it works over the whole image domain.

ShIRT, based on an optical flow algorithm, has been successfully applied to many clinical applications and, having been developed in the Medical Physics group in Sheffield, was a reasonable starting point for this work. In fact, the current state of the art method of generating patient-specific meshes, discussed in chapter 6, section 6.1 uses ShIRT to compute the morphing field. ShIRT includes a regularisation term that effectively imposes smoothness on the resulting registration field (using the Laplacian regularisation). The formulation of ShIRT, in Cartesian space, does not naturally lend itself to the control of mesh quality because the degrees of freedom are not linked to the element shapes. In fact, the registration maps produced by ShIRT do not reliably maintain the quality of the mesh elements, as demonstrated in chapter 6, section 6.5. The method presented in chapters 3-6 represents a major improvement of the algorithm for applications of this nature.

This thesis has presented a novel implementation of image registration (chapter 3), defined in parametric space to facilitate the direct implementation of a regularisation term to optimise mesh quality (chapter 4). The method involves combining the registration grid, registration mask and moving image into a single object, defined by a finite-element mesh.

As discussed above, the method is based on an optical flow algorithm with the algebra transformed into parametric space. This formulation of the problem emphasises the image processing roots of the procedure, seeking a transformation or registration field that maps a binary image of the template onto the binary segmentation of the patient image. The regularisation term is used to preserve, or to improve, the quality of the mesh. In principle there is strong similarity to another approach, deformable models, in which the mesh is assigned structural or fluid properties and the images are used to derive force terms to deform the template mesh towards a patient representation. However, the basis of deformable models is the balance of forces whereas this method works to increase the intensity-based similarity between two images. This thesis deals with binary images and the methods are very similar, however the method presented in this thesis can be extended to use information from a medical image as the moving image by extending the intensity dimension.

Chapter 5 describes implementation choices and tests performed during the development stages to gain understanding of the algorithm, and its limitations. The experiments performed in this chapter test the main variables of the registration. Notable outcomes of the tests include the relatively minor influence the number of sampling points has on the accuracy of the registration and the sensitivity of the registration process to the regularisation term and its weighting. Another important determinant of the outcome is the combination of the regularisation terms. In normal operation, the registration would be run twice: first with the Laplacian regularisation and memory term, and then with either the mesh improvement regularisation or the Laplacian regularisation and no memory term.

An application of the registration method to produce mechanical meshes of the left ventricular myocardium has been demonstrated (Chapter 6). A total of 66 cases were processed (the segmentations derived using different segmentation algorithms are counted independently because they exhibit considerably different image properties and morphological characteristics). Overall, the mean fitting distance was 0.408 mm for the conformance mesh and 0.517 mm for the quality mesh, and the mesh degradation of the conformance mesh was 0.997 and 1.033 for the quality mesh. The conformance mesh is 10.9% closer to the target surface than the quality mesh, however in both cases the fitting distance was well beyond the voxel size, so the loss in accuracy is negligible. On the other hand, the quality mesh contains 68.6% less degraded elements, which is an important improvement.

The registration method is also suitable for tracking anatomical structures in volume images. Chapter 7 illustrates an application of this by tracking the deformation of the ventricular muscle and using structural deformation to run a fluid flow analysis. A number of limitations of the code

(which will be listed in the next section) make the content of this chapter a proof-of-concept that needs improving, however the chapter successfully demonstrates the registration algorithm performs well for the task.

## 8.2 Limitations and Future Work

Chapter 2 has introduced the concept of informatics as a way to integrate medical data and derived modelling data to improve clinical decision making. Not explored in this work remains the process of producing useful metrics from workflow-derived data and mining the data-rich environment to create a new subset of medical knowledge. This was regarded as out of scope for the current thesis.

There are several limitations of the registration work flow developed in this thesis, and some of them will be addressed in future developments:

- The method presented in chapter 3 is limited by the first order Taylor series approximation made to derive the equations. It is a benefit that the linearisation produced by this approximation leads to major benefits in algorithmic efficiency and shorter compute times. Large displacements are accommodated by repeated operation of the algorithm, updating the moved image at each operation. However this process is not always robust if the displacement between the template and target images is large, and it certainly does not produce a diffeomorphic mapping. In the future higher order approximations will be implemented, and the opportunity to enforce a diffeomorphic constraint will be explored.
- The most significant fact of how the registration performs is the choice of regularisation term and the weight given to the term. This is a general problem, not specific to the registration process presented in this thesis, but nonetheless remains a limitation for practical use of the tool. More work is required to optimise the relative weightings, particularly when multiple regularisation terms (e.g. smoothing and mesh quality) are implemented together.
- In terms of implementation, the code runs single-threaded and has not been optimised for memory use. This is of particular concern with the Jacobian regularisation: the time taken to construct the scalar product and Jacobian regularisation terms can be prohibitive for large meshes, even with the linearization performed. Using the geometrical regularisation terms increases the run time of the image registration program ten-fold.
- The software has been developed to optimise the quality of eight node hexahedral meshes. Other element shapes, such as tetrahedra, might be preferred for some applications. For the ventricular meshes, higher order elements are often used: in particular cubic Hermite

elements have been favoured by the groups from Auckland and from KCL. Preliminary development of the algebraic formulations required to implement these elements into the current work flow has been performed, and the software implementation will be done in future work.

The main drawback of the application to generate patient specific meshes is the requirement for a segmentation of the medical image before commencement of the registration process. Whilst several methods for segmenting medical data exist, the reliance on this data being available is a drawback in terms of practicality, data quality and accuracy. The software developed in this thesis can be applied to general medical images (the images do not have to be binary), but the work flow has not been developed or tested for operation on raw image data. This would lead to significant expansion of the number of potential applications of the code in more general image processing applications. An alternative approach to the binary template-to-segmented image registration performed in this thesis would be to resample a parent image, on which a template has been generated, into the parametric space and then to register the parent image to the patient image, carrying the template along. This might improve robustness because there would be information (such as adjacent structures) in the parent image that does not feature in the template mesh. There are some disadvantages in operation on the full image data, one of which is the variations of intensity between images of different patients (or even time series images of the same patient). In fact the SHIRT code operates on binary images that are constructed by expansion of the dimension of the medical image by one, with the additional dimension representing the intensity direction. This extension supports registration of images that have some differences in intensity ranges, and the same extension could be applied in the new code.

The simulation set up presented in chapter 7 was developed only to demonstrate the use of the registration to produce more realistic boundary conditions and has many limitations: it does not fully track the complete ventricular contraction, does not include both the mitral valve and the aortic valve and is limited to the period of systole, does not take the motion of the ventricular inflow tract and aorta into account, does not take the geometry of the aortic arch into account and, in the current set up, does not allow for a fine mesh resolution to be used in the simulation. All of these issues would need to be resolved to support a comprehensive simulation of the performance of the valve.

# PUBLICATIONS

## Conferences

Appeared:

- Daniel Alejandro Silva Soto, David Jones, D. Rodney Hose and Steven Wood. September, 2012. *Developing a workflow System that integrates scientific development*. Presented at the Virtual Physiological Human 2010, London, England, UK. [84]
- Daniel Alejandro Silva Soto and D. Rodney Hose. September 2013. *Using image registration defined in parametric space to personalise cardiac meshes*. Presented at Bioengineering 2013, Glasgow, Scotland, UK. [124]
- Daniel Alejandro Silva Soto, David C. Barber and D. Rodney Hose. September 2014. *Generating patient-specific finite-element meshes by image registration in parametric space*. Presented at the Virtual Physiological Human 2014, Trondheim, Norway. [125]

## Journal articles

In preparation:

- Generation of high quality finite element meshes using image registration in parametric space.
- Image-derived wall motion for haemodynamic analysis.

## Book chapter

Published:

- Daniel Alejandro Silva Soto, Steven Wood and D. Rodney Hose. 2014. Chapter 8: Workflows: Principles, Tools and Clinical Applications. In: Peter Coveney, Vanessa Diaz, Peter Hunter, and Marco Viceconti, eds. *Computational Biomedicine*. Oxford: Oxford University Press, pages 161-185. [46]

## REFERENCES

- [1] National Institute for Health and Care Excellence, "Chronic heart failure. Management of chronic heart failure in adults in primary and secondary care (CG108)." National Institute for Health and Care Excellence, Aug-2010.
- [2] "Heart failure," *Medical Subject Headings (MeSH)*. National Institutes for Health.
- [3] C. S. P. Lam, E. Donal, E. Kraigher-Krainer, and R. S. Vasan, "Epidemiology and clinical course of heart failure with preserved ejection fraction," *European Journal of Heart Failure*, vol. 13, no. 1, pp. 18–28, Jan. 2011.
- [4] C. Members, S. A. Hunt, D. W. Baker, M. H. Chin, M. P. Cinquegrani, A. M. Feldmanmd, G. S. Francis, T. G. Ganiats, S. Goldstein, G. Gregoratos, M. L. Jessup, R. J. Noble, M. Packer, M. A. Silver, L. W. Stevenson, R. J. Gibbons, E. M. Antman, J. S. Alpert, D. P. Faxon, V. Fuster, G. Gregoratos, A. K. Jacobs, L. F. Hiratzka, R. O. Russell, and S. C. Smith, "ACC/AHA Guidelines for the Evaluation and Management of Chronic Heart Failure in the Adult: Executive Summary A Report of the American College of Cardiology/American Heart Association Task Force on Practice Guidelines (Committee to Revise the 1995 Guidelines for the Evaluation and Management of Heart Failure)," *Circulation*, vol. 104, no. 24, pp. 2996–3007, 2001.
- [5] The Criteria Committee of the New York Heart Association, *Nomenclature and Criteria for Diagnosis of Diseases of the Heart and Great Vessels*, 9th ed. Boston, Mass: Little, Brown & Co, 1994.
- [6] P. W. X. Foley, S. Chalil, K. Khadjooi, N. Irwin, R. E. A. Smith, and F. Leyva, "Left ventricular reverse remodelling, long-term clinical outcome, and mode of death after cardiac resynchronization therapy," *European Journal of Heart Failure*, vol. 13, no. 1, pp. 43–51, Jan. 2011.
- [7] M. G. S. J. Sutton and N. Sharpe, "Left Ventricular Remodeling After Myocardial Infarction : Pathophysiology and Therapy," *Circulation*, vol. 101, no. 25, pp. 2981–2988, Jun. 2000.
- [8] S. F. Nagueh, "Mechanical Dyssynchrony in Congestive Heart Failure," *Journal of the American College of Cardiology*, vol. 51, no. 1, pp. 18–22, Jan. 2008.
- [9] National Institute for Health and Clinical Excellence, "Cardiac resynchronisation therapy for the treatment of heart failure." Jul-2010.
- [10] D. Verhaert, R. A. Grimm, C. Puntawangkoon, K. Wolski, S. De, B. L. Wilkoff, R. C. Starling, W. H. W. Tang, J. D. Thomas, and Z. B. Popovic, "Long-Term Reverse Remodeling With Cardiac Resynchronization Therapy: Results of Extended Echocardiographic Follow-Up," *Journal of the American College of Cardiology*, vol. 55, pp. 1788–1795, 2010.
- [11] F. A. McAlister, J. Ezekowitz, N. Hooton, B. Vandermeer, C. Spooner, D. M. Dryden, R. L. Page, M. A. Hlatky, and B. H. Rowe, "Cardiac Resynchronization Therapy for Patients With Left Ventricular Systolic Dysfunction," *JAMA: The Journal of the American Medical Association*, vol. 297, pp. 2502–2514, Jun. 2007.
- [12] C. Jenkins, K. Bricknell, J. Chan, L. Hanekom, and T. H. Marwick, "Comparison of Two- and Three-Dimensional Echocardiography With Sequential Magnetic Resonance Imaging for Evaluating Left Ventricular Volume and Ejection Fraction Over Time in Patients With Healed Myocardial Infarction," *The American Journal of Cardiology*, vol. 99, no. 3, pp. 300–306, Feb. 2007.
- [13] N. J. Talley and S. O'Connor, *Examination Medicine*. Elsevier Australia, 2009.
- [14] T. A. Foley, S. V. Mankad, N. S. Anavekar, C. R. Bonnichsen, M. F. Morris, T. D. Miller, and P. A. Araoz, "Measuring Left Ventricular Ejection Fraction - Techniques and potential pitfalls," *European Cardiology Review*, vol. 8, no. 2, pp. 108–114, Mar. 2012.

- [15] E. Vallejo, "Variability of serial same-day left ventricular ejection fraction using quantitative gated SPECT," *Journal of Nuclear Cardiology*, vol. 9, no. 4, pp. 377–384, Jul. 2002.
- [16] D. W. McRobbie, *MRI from picture to proton*. Cambridge, UK; New York: Cambridge University Press, 2006.
- [17] H. Nickisch, H. Barschdorf, F. M. Weber, M. W. Krueger, O. Dössel, and J. Weese, "From image to personalized cardiac simulation: encoding anatomical structures into a model-based segmentation framework," in *Statistical Atlases and Computational Models of the Heart. Imaging and Modelling Challenges*, Springer, 2013, pp. 278–287.
- [18] M. Sermesant, K. Rhode, G. I. Sanchez-Ortiz, O. Camara, R. Andriantsimiavona, S. Hegde, D. Rueckert, P. Lambiase, C. Bucknall, E. Rosenthal, H. Delingette, D. L. G. Hill, N. Ayache, and R. Razavi, "Simulation of cardiac pathologies using an electromechanical biventricular model and XMR interventional imaging," *Medical Image Analysis*, vol. 9, no. 5, pp. 467–480, Oct. 2005.
- [19] A. Sotiras, C. Davatzikos, and N. Paragios, "Deformable Medical Image Registration: A Survey," *Medical Imaging, IEEE Transactions on*, vol. 32, no. 7, pp. 1153–1190, Jul. 2013.
- [20] B. Li and S. T. Acton, "Active Contour External Force Using Vector Field Convolution for Image Segmentation," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2096–2106, Aug. 2007.
- [21] D. Barber and D. Hose, "Automatic segmentation of medical images using image registration: diagnostic and simulation applications," *Journal of Medical Engineering & Technology*, vol. 29, no. 2, pp. 53–63, Jan. 2005.
- [22] M. Zuluaga, M. J. Cardoso, M. Modat, and S. Ourselin, "Multi-atlas Propagation Whole Heart Segmentation from MRI and CTA Using a Local Normalised Correlation Coefficient Criterion," in *Functional Imaging and Modeling of the Heart*, vol. 7945, S. Ourselin, D. Rueckert, and N. Smith, Eds. Springer Berlin Heidelberg, 2013, pp. 174–181.
- [23] F. Zhao and X. Xie, "An overview of interactive medical image segmentation," *Annals of the BMVA*, vol. 2013, no. 7, pp. 1–22, 2013.
- [24] A. Schenk, G. Prause, and H.-O. Peitgen, "Efficient semiautomatic segmentation of 3D objects in medical images," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2000*, 2000, pp. 186–195.
- [25] Y. C. Fung, *Biomechanics: Mechanical Properties of Living Tissues*. Springer New York, 1993.
- [26] A. G. Brown, Y. Shi, A. Marzo, C. Staicu, I. Valverde, P. Beerbaum, P. V. Lawford, and D. R. Hose, "Accuracy vs. computational time: Translating aortic simulations to the clinic," *Journal of Biomechanics*, vol. 45, no. 3, pp. 516–523, Feb. 2012.
- [27] D. P. Raymer, *Aircraft Design: A Conceptual Approach*, 4th ed. American Institute of Aeronautics and Astronautics, 2006.
- [28] F. N. van de Vosse and N. Stergiopoulos, "Pulse Wave Propagation in the Arterial Tree," *Annual Review of Fluid Mechanics*, vol. 43, no. 1, pp. 467–499, Jan. 2011.
- [29] Y. Shi, P. Lawford, and R. Hose, "Review of zero-D and 1-D models of blood flow in the cardiovascular system," *Biomed. Eng. Online*, vol. 10, no. 1, p. 33, 2011.
- [30] Y. Shi, P. Lawford, and D. Hose, "Numerical Modeling of Hemodynamics with Pulsatile Impeller Pump Support," *Annals of Biomedical Engineering*, vol. 38, pp. 2621–2634, 2010.
- [31] H. Kim, I. Vignon-Clementel, C. Figueroa, J. LaDisa, K. Jansen, J. Feinstein, and C. Taylor, "On Coupling a Lumped Parameter Heart Model and a Three-Dimensional Finite Element Aorta Model," *Annals of Biomedical Engineering*, vol. 37, pp. 2153–2169, 2009.
- [32] P. Lamata, I. Roy, B. Blazevic, A. Crozier, S. Land, S. A. Niederer, D. R. Hose, and N. P. Smith, "Quality Metrics for High Order Meshes: Analysis of the Mechanical Simulation of the Heart Beat," *IEEE Transactions on Medical Imaging*, vol. 32, no. 1, pp. 130–138, Jan. 2013.



- [33] P. Lamata, M. Sinclair, E. Kerfoot, A. Lee, A. Crozier, B. Blazevic, S. Land, A. J. Lewandowski, D. Barber, S. Niederer, and others undefined, "An automatic service for the personalization of ventricular cardiac meshes," *Journal of The Royal Society Interface*, vol. 11, no. 91, p. 20131023, 2014.
- [34] M. Sermesant, J.-M. Peyrat, P. Chinchapatnam, F. Billet, T. Mansi, K. Rhode, H. Delingette, R. Razavi, and N. Ayache, "Toward Patient-Specific Myocardial Models of the Heart," *Heart Failure Clinics*, vol. 4, pp. 289–301, 2008.
- [35] S. A. Niederer, G. Plank, P. Chinchapatnam, M. Ginks, P. Lamata, K. S. Rhode, C. A. Rinaldi, R. Razavi, and N. P. Smith, "Length-dependent tension in the failing heart and the efficacy of cardiac resynchronization therapy," *Cardiovascular Research*.
- [36] A. Auricchio, C. Fantoni, F. Regoli, C. Carbucicchio, A. Goette, C. Geller, M. Kloss, and H. Klein, "Characterization of Left Ventricular Activation in Patients With Heart Failure and Left Bundle-Branch Block," *Circulation*, vol. 109, pp. 1133–1139, Mar. 2004.
- [37] S. Deftereos, G. Giannopoulos, C. Kossyvakis, K. Raisakis, A. Kaoukis, M. Driva, V. Panagopoulou, O. Ntzouvara, A. Theodorakis, K. Toutouzas, V. Pyrgakis, and C. Stefanadis, "Differential Effect of Biventricular and Right Ventricular DDD Pacing on Coronary Flow Reserve in Patients With Ischemic Cardiomyopathy," *Journal of Cardiovascular Electrophysiology*, vol. 21, pp. 1233–1239, 2010.
- [38] ANSYS Inc., "ANSYS Mechanical APDL Theory Reference, Release 14.5.pdf." .
- [39] ANSYS Inc., "ANSYS Coupled-Field Analysis Guide, Release 12.1." Nov-2009.
- [40] J. R. Shewchuk, "An introduction to the conjugate gradient method without the agonizing pain (Edition 1 1/4)." Carnegie Mellon University, Pittsburgh, PA, 1994.
- [41] J. J. Bax and J. Gorcsan Iii, "Echocardiography and Noninvasive Imaging in Cardiac Resynchronization Therapy: Results of the PROSPECT (Predictors of Response to Cardiac Resynchronization Therapy) Study in Perspective," *Journal of the American College of Cardiology*, vol. 53, pp. 1933–1943, 2009.
- [42] M. Shojima, M. Oshima, K. Takagi, R. Torii, M. Hayakawa, K. Katada, A. Morita, and T. Kirino, "Magnitude and Role of Wall Shear Stress on Cerebral Aneurysm: Computational Fluid Dynamic Study of 20 Middle Cerebral Artery Aneurysms," *Stroke*, vol. 35, no. 11, pp. 2500–2505, Nov. 2004.
- [43] Authors/Task Force members, R. Erbel, V. Aboyans, C. Boileau, E. Bossone, R. D. Bartolomeo, H. Eggebrecht, A. Evangelista, V. Falk, H. Frank, O. Gaemperli, M. Grabenwoger, A. Haverich, B. Iung, A. J. Manolis, F. Meijboom, C. A. Nienaber, M. Roffi, H. Rousseau, U. Sechtem, P. A. Sirnes, R. S. v. Allmen, C. J. M. Vrints, ESC Committee for Practice Guidelines (CPG), J. L. Zamorano, S. Achenbach, H. Baumgartner, J. J. Bax, H. Bueno, V. Dean, C. Deaton, C. Erol, R. Fagard, R. Ferrari, D. Hasdai, A. Hoes, P. Kirchhof, J. Knuuti, P. Kolh, P. Lancellotti, A. Linhart, P. Nihoyannopoulos, M. F. Piepoli, P. Ponikowski, P. A. Sirnes, J. L. Tamargo, M. Tendera, A. Torbicki, W. Wijns, S. Windecker, Document reviewers, P. Nihoyannopoulos, M. Tendera, M. Czerny, J. Deanfield, C. D. Mario, M. Pepi, M. J. S. Taboada, M. R. v. Sambeek, C. Vlachopoulos, J. L. Zamorano, M. Grimm, O. Musayev, A. Pasquet, Z. Ku Ijugi, M. Cikes, G. P. Georghiou, J. Stasek, H. Molgaard, S. Kovask;, V. Kytö, G. Jondeau, Z. Bakhutashvili, Y. von Kodolitsch, C. Tsioufis, A. Temesvari, R. Rubinshtein, F. Antonini-Canterin, O. Lunegova, P. Stradins, E. Chammas, R. Jonkaitiene, A. Cassar, K. Bjornstad, K. Widenka, M. Sousa Uva, D. Lighezan, J. Perunicic, J. Madaric, I. Vilacosta, M. Back, A. Mahdhaoui, R. Demirbag, and I. Kravchenko, "2014 ESC Guidelines on the diagnosis and treatment of aortic diseases: Document covering acute and chronic aortic diseases of the thoracic and abdominal aorta of the adult \* The Task Force for the Diagnosis and Treatment of Aortic Diseases of the European Society of Cardiology (ESC)," *European Heart Journal*, vol. 35, no. 41, pp. 2873–2926, Nov. 2014.
- [44] Bojan Blazevic, "Towards a fast patient-specific model of electrophysiology for the clinic (Ph.D. proposal)." King's College London, UK, 21-Mar-2011.

- [45] P. M. Knupp, "Algebraic mesh quality metrics," *SIAM journal on scientific computing*, vol. 23, no. 1, pp. 193–218, 2001.
- [46] D. A. Silva Soto, S. Wood, and D. R. Hose, "Workflows: Principles, tools and clinical applications," in *Computational Biomedicine*, 1st ed., P. Coveney, V. Díaz-Zuccarini, P. Hunter, and M. Viceconti, Eds. Oxford: Oxford University Press, 2014, pp. 161–177.
- [47] R. Haux, "Aims and tasks of medical informatics," *International Journal of Medical Informatics*, vol. 44, pp. 9–20, 1997.
- [48] T. A. Morris and K. W. McCain, "The Structure of Medical Informatics Journal Literature," *Journal of the American Medical Informatics Association*, vol. 5, pp. 448–466, Sep. 1998.
- [49] A. Hasman, "Challenges for medical informatics in the 21st century," *International Journal of Medical Informatics*, vol. 44, pp. 1–7, 1997.
- [50] R. Haux, "Medical informatics: Past, present, future," *International Journal of Medical Informatics*, vol. 79, pp. 599–610, 2010.
- [51] P. L. Reichertz, "Hospital information systems--Past, present, future," *International Journal of Medical Informatics*, vol. 75, pp. 282–299.
- [52] J. T. Lium and A. Faxvaag, "Removal of paper-based health records from Norwegian hospitals: effects on clinical workflow," *Studies in health technology and informatics*, vol. 124, p. 1031, 2006.
- [53] H. Lærum, T. H. Karlsen, and A. Faxvaag, "Use of and attitudes to a hospital information system by medical secretaries, nurses and physicians deprived of the paper-based medical record: a case report," *BMC Medical Informatics and Decision Making*, vol. 4, p. 18, 2004.
- [54] C. A. Taylor, M. T. Draney, J. P. Ku, D. Parker, B. N. Steele, K. Wang, and C. K. Zarins, "Predictive Medicine: Computational Techniques in Therapeutic Decision-Making," *Computer Aided Surgery*, vol. 4, pp. 231–247, 1999.
- [55] M. Ortega, N. Barreira, J. Novo, M. G. Penedo, A. Pose-Reino, and F. Gómez-Ulla, "Sirius: A web-based system for retinal image analysis," *International Journal of Medical Informatics*, vol. 79, pp. 722–732, 2010.
- [56] Health Level Seven International, "Health Level 7 website." [Online]. Available: <http://www.hl7.org/about/index.cfm?ref=nav>.
- [57] International Health Terminology Standards Development Organisation, "SNOMED CT website." [Online]. Available: <http://www.ihtsdo.org/snomed-ct>.
- [58] National Electrical Manufacturers Association (NEMA), "Digital Imaging and Communications in Medicine (DICOM) website (NEMA)." [Online]. Available: <http://dicom.nema.org/>.
- [59] John Pella, "John Pella on the DICOM Standards," 2009. [Online]. Available: <http://johnpella.com/pella-dicom-tutorial1.htm>.
- [60] S. C. Horii, "A Nontechnical Introduction to DICOM: The Elemental Unit of DICOM." RadioGraphics, 1997.
- [61] "Digital Imaging and COmmunications in Medicine (DICOM) Part 18: Web Access to DICOM Persistent Objects (WADO)." National Electrical Manufacturers Association (NEMA), 18-Mar-2011.
- [62] Ministry of Justice, United Kingdom, "Data Protection Act." HMSO, 1998.
- [63] DH/IPU/Patient Confidentiality, "NHS Confidentiality Code of Practice." Department of Health, Nov-2003.
- [64] DH/Digital Information Policy, "Information Security Management: NHS Code of Practice." Department of Health, Apr-2007.
- [65] Microsoft, "Windows Workflow Foundation on MSDN." [Online]. Available: [https://msdn.microsoft.com/en-us/library/dd489441\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/dd489441(v=vs.110).aspx).

- [66] D. Hull, K. Wolstencroft, R. Stevens, C. Goble, M. R. Pocock, P. Li, and T. Oinn, "Taverna: a tool for building and running workflows of services," *Nucleic Acids Research*, vol. 34, pp. W729–W732, Jul. 2006.
- [67] J. Cheney, S. Chong, N. Foster, M. Seltzer, and S. Vansummeren, "Provenance: A Future History," in *Proceedings of the 24th ACM SIGPLAN Conference Companion on Object Oriented Programming Systems Languages and Applications*, New York, NY, USA, 2009, pp. 957–964.
- [68] L. Moreau, B. Clifford, J. Freire, J. Futrelle, Y. Gil, P. Groth, N. Kwasnikowska, S. Miles, P. Missier, J. Myers, B. Plale, Y. Simmhan, E. Stephan, and J. Van den Bussche, "The Open Provenance Model core specification (v1.1)," *Future Generation Computer Systems*, 2010.
- [69] C. Goble, "Position statement: Musings on provenance, workflow and (semantic web) annotations for bioinformatics," in *Workshop on Data Derivation and Provenance, Chicago*, 2002.
- [70] Y. L. Simmhan, B. Plale, and D. Gannon, "Karma2: Provenance management for data-driven workflows," *International Journal of Web Services Research*, vol. 5, pp. 1–22, 2008.
- [71] Y. L. Simmhan, B. Plale, and D. Gannon, "A survey of data provenance in e-science," *SIGMOD Record*, vol. 34, pp. 31–36, 2005.
- [72] L Moreau et al, "An Open Provenance Model for Scientific Workflows," presented at the High Performance Computing and Grids - HPC'06 Workshop, 2006.
- [73] T. Heinis and G. Alonso, "Efficient lineage tracking for scientific workflows," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 1007–1018.
- [74] P. Missier, N. W. Paton, and K. Belhajjame, "Fine-grained and efficient lineage querying of collection-based workflow provenance," in *Proceedings of the 13th International Conference on Extending Database Technology*, 2010, pp. 299–310.
- [75] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, pp. 541–580, 1989.
- [76] Nick Chapman, "Petri Net Models," *ISE-2 Surprise 97 Project*, 27-May-1997. [Online]. Available: [http://www.doc.ic.ac.uk/~nd/surprise\\_97/journal/vol2/njc1/](http://www.doc.ic.ac.uk/~nd/surprise_97/journal/vol2/njc1/).
- [77] D. Ings, "BPEL4People Website," *OASIS WS-WPEL Extension for People (BPEL4People) TC*. [Online]. Available: [https://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=bpel4people](https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=bpel4people).
- [78] Peter Li, *Identification of differential genes using the LIMMA Bioconductor package within R - workflow preview*. 2008.
- [79] B. Ludäscher, I. Altintas, S. Bowers, J. Cummings, T. Critchlow, E. Deelman, D. D. Roure, J. Freire, C. Goble, M. Jones, and others, "Scientific process automation and workflow management," *Scientific Data Management: Challenges, Existing Technology, and Deployment, Computational Science Series*, pp. 476–508, 2009.
- [80] P. Matt Milner, "A Developer's Introduction to Windows Workflow Foundations (WF) in .NET 4," 2010. [Online]. Available: <https://msdn.microsoft.com/en-us/library/ee342461.aspx>.
- [81] The Mono Project (Xamarin Inc. ), "Mono Project Website." [Online]. Available: <http://www.mono-project.com/>.
- [82] Microsoft, "Introduction to workflows." [Online]. Available: <https://support.office.microsoft.com/en-us/article/Introduction-to-workflows-4b77a2bb-0993-453f-8fca-b6a011833827>.
- [83] Microsoft, "MSDN - SqlWorkflowPersistenceService Class." [Online]. Available: <https://msdn.microsoft.com/en-us/library/system.workflow.runtime.hosting.sqlworkflowpersistence%28v=vs.110%29.aspx>.

- [84] Daniel Alejandro Silva Soto, David Jones, Steven Wood, and Rod Hose, "Developing a workflow System that integrates scientific development," presented at the VPH2012, London.
- [85] Jones, D. M., Hose, D. R., Lawford, P. V., Hill, D. L., Razavi R. S., and Barber D. C., "Creation of patient-specific CFD models by morphing a previously-meshed reference geometry using image registration," presented at the Medical Image Understanding and Analysis, London, UK, 2004, pp. 173–176.
- [86] D. C. Barber, E. Oubel, A. F. Frangi, and D. R. Hose, "Efficient computational fluid dynamics mesh generation by image registration," *Medical Image Analysis*, vol. 11, no. 6, pp. 648–662, Dec. 2007.
- [87] P. Lamata, S. Niederer, D. Nordsletten, D. C. Barber, I. Roy, D. R. Hose, and N. Smith, "An accurate, fast and robust method to generate patient-specific cubic Hermite meshes," *Medical Image Analysis*, vol. 15, no. 6, pp. 801–813, Dec. 2011.
- [88] L. G. Brown, "A survey of image registration techniques," *ACM computing surveys (CSUR)*, vol. 24, no. 4, pp. 325–376, 1992.
- [89] J. Maintz and M. Viergever, "A survey of medical image registration," *Medical Image Analysis*, vol. 2, no. 1, pp. 1–36, Oct. 1997.
- [90] J. M. Fitzpatrick, D. L. Hill, and C. R. Maurer Jr, "Image registration," *Handbook of medical imaging*, vol. 2, pp. 447–513, 2000.
- [91] P. Lamata, S. Niederer, D. Barber, D. Nordsletten, J. Lee, R. Hose, and N. Smith, "Personalization of cubic hermite meshes for efficient biomechanical simulations," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2010*, Springer, 2010, pp. 380–387.
- [92] G. E. Christensen and H. J. Johnson, "Consistent image registration," *Medical Imaging, IEEE Transactions on*, vol. 20, no. 7, pp. 568–582, 2001.
- [93] J. He and G. E. Christensen, "Large deformation inverse consistent elastic image registration," in *Information Processing in Medical Imaging*, 2003, pp. 438–449.
- [94] P. M. Knupp, "Algebraic mesh quality metrics for unstructured initial meshes," *Finite Elements in Analysis and Design*, vol. 39, no. 3, pp. 217–241, 2003.
- [95] P. M. Knupp, "Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities. Part I—a framework for surface mesh optimization," *International Journal for Numerical Methods in Engineering*, vol. 48, no. 3, pp. 401–420, 2000.
- [96] P. M. Knupp, "Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities. Part II—A framework for volume mesh optimization and the condition number of the Jacobian matrix," *Int. J. Numer. Meth. Engng.*, vol. 48, no. 8, pp. 1165–1185, Jul. 2000.
- [97] B. Fischer and J. Modersitzki, "Ill-posed medicine—an introduction to image registration," *Inverse Problems*, vol. 24, no. 3, p. 034008, Jun. 2008.
- [98] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani, "Hierarchical model-based motion estimation," in *Computer Vision—ECCV'92*, 1992, pp. 237–252.
- [99] X. Pennec, P. Cachier, and N. Ayache, "Understanding the 'demon's algorithm': 3D non-rigid registration by gradient descent," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI'99*, 1999, pp. 597–605.
- [100] U. Clarenz, M. Droske, S. Henn, M. Rumpf, and K. Witsch, "Computational Methods for Nonlinear Image Registration," in *Mathematical Models for Registration and Applications to Medical Imaging*, vol. 10, O. Scherzer, Ed. Springer Berlin Heidelberg, 2006, pp. 81–101.
- [101] P. C. Hansen, *The L-curve and its use in the numerical treatment of inverse problems*. IMM, Department of Mathematical Modelling, Technical University of Denmark, 1999.
- [102] ANSYS Inc., "ANSYS Meshing User's Guide, Release 14.5." Oct-2012.

- [103] D. Rueckert, L. I. Sonoda, C. Hayes, D. L. Hill, M. O. Leach, and D. J. Hawkes, "Nonrigid registration using free-form deformations: application to breast MR images," *Medical Imaging, IEEE Transactions on*, vol. 18, no. 8, pp. 712–721, 1999.
- [104] T. Rohlfing, C. R. Maurer, D. A. Bluemke, and M. A. Jacobs, "Volume-preserving nonrigid registration of MR breast images using free-form deformation with an incompressibility constraint," *IEEE Transactions on Medical Imaging*, vol. 22, no. 6, pp. 730–741, Jun. 2003.
- [105] D. Loeckx, F. Maes, D. Vandermeulen, and P. Suetens, "Nonrigid image registration using free-form deformations with a local rigidity constraint," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2004*, Springer, 2004, pp. 639–646.
- [106] R. Bajcsy and S. Kovačič, "Multiresolution elastic matching," *Computer Vision, Graphics, and Image Processing*, vol. 46, no. 1, pp. 1–21, Apr. 1989.
- [107] A. Shalhaf, H. Behnam, Z. Alizade-Sani, and M. Shojaifard, "Automatic Classification of Left Ventricular Regional Wall Motion Abnormalities in Echocardiography Images Using Nonrigid Image Registration," *J Digit Imaging*, vol. 26, no. 5, pp. 909–919, Oct. 2013.
- [108] C. Broit, "Optimal Registration of Deformed Images," University of Pennsylvania, Philadelphia, PA, USA, 1981.
- [109] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International journal of computer vision*, vol. 1, no. 4, pp. 321–331, 1988.
- [110] J. Ashburner, "A fast diffeomorphic image registration algorithm," *NeuroImage*, vol. 38, no. 1, pp. 95–113, Oct. 2007.
- [111] B. Fischer and J. Modersitzki, "Curvature based image registration," *Journal of Mathematical Imaging and Vision*, vol. 18, no. 1, pp. 81–85, 2003.
- [112] T. Rohlfing, "Transformation model and constraints cause bias in statistics on deformation fields," in *Medical Image Computing And Computer-Assisted Intervention—MICCAI 2006*, Springer, 2006, pp. 207–214.
- [113] J.-P. Thirion, "Image matching as a diffusion process: an analogy with Maxwell's demons," *Medical Image Analysis*, vol. 2, no. 3, pp. 243–260, Sep. 1998.
- [114] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing. Second Edition*. 1992.
- [115] I. B. M. Corporation, *FORTRAN: an automatic coding system for the IBM 704*. 1956.
- [116] *Open MP*. OpenMP Architecture Review Board, 2013.
- [117] *MATLAB, version 7.13 (R2011b)*. Natick, Massachusetts: The MathWorks Inc., 2011.
- [118] J. A. Schnabel, C. Tanner, A. D. Castellano-Smith, A. Degenhard, M. O. Leach, D. R. Hose, D. L. G. Hill, and D. J. Hawkes, "Validation of nonrigid image registration using finite-element methods: application to breast MR images," *IEEE Transactions on Medical Imaging*, vol. 22, no. 2, pp. 238–247, Feb. 2003.
- [119] M. Sermesant, C. Forest, X. Pennec, H. Delingette, and N. Ayache, "Deformable biomechanical models: application to 4D cardiac image analysis," *Medical Image Analysis*, vol. 7, no. 4, pp. 475–488, 2003.
- [120] M. J. Gonzales, G. Sturgeon, A. Krishnamurthy, J. Hake, R. Jonas, P. Stark, W.-J. Rappel, S. M. Narayan, Y. Zhang, W. P. Segars, and A. D. McCulloch, "A three-dimensional finite element model of human atrial anatomy: New methods for cubic Hermite meshes with extraordinary vertices," *Medical Image Analysis*, vol. 17, no. 5, pp. 525–537, Jul. 2013.
- [121] Y. Zhang, X. Liang, J. Ma, Y. Jing, M. J. Gonzales, C. Villongco, A. Krishnamurthy, L. R. Frank, V. Nigam, P. Stark, S. M. Narayan, and A. D. McCulloch, "An atlas-based geometry pipeline for cardiac Hermite model construction and diffusion tensor reorientation," *Medical Image Analysis*, vol. 16, no. 6, pp. 1130–1141, Aug. 2012.
- [122] S. Ji, J. C. Ford, R. M. Greenwald, J. G. Beckwith, K. D. Paulsen, L. A. Flashman, and T. W. McAllister, "Automated subject-specific, hexahedral mesh generation via image registration," *Finite Elements in Analysis and Design*, vol. 47, no. 10, pp. 1178–1185, Oct. 2011.

- [123] J. W. Fernandez, P. Mithraratne, S. F. Thrupp, M. H. Tawhai, and P. J. Hunter, "Anatomically based geometric modelling of the musculo-skeletal system and other organs," *Biomechanics and Modeling in Mechanobiology*, vol. 2, no. 3, pp. 139–155, Mar. 2004.
- [124] D. A. Silva Soto and D. R. Hose, "Using image registration defined in parametric space to personalise cardiac meshes," presented at the Bioengineering 2013, Glasgow, UK, 2013.
- [125] D. A. Silva Soto, D. C. Barber, and D. R. Hose, "Generating patient-specific finite-element meshes by image registration in parametric space," presented at the Virtual Physiological Human Conference 2014, Trondheim, Norway, 2014.
- [126] O. Ecabert, J. Peters, H. Schramm, C. Lorenz, J. von Berg, M. J. Walker, M. Vembar, M. E. Olszewski, K. Subramanyan, G. Lavi, and J. Weese, "Automatic Model-Based Segmentation of the Heart in CT Images," *IEEE Transactions on Medical Imaging*, vol. 27, no. 9, pp. 1189–1201, Sep. 2008.
- [127] I. Wolf, M. Vetter, I. Wegner, M. Nolden, T. Bottger, M. Hastenteufel, M. Schobinger, T. Kunert, and H.-P. Meinzer, "The medical imaging interaction toolkit (MITK): a toolkit facilitating the creation of interactive software by extending VTK and ITK," in *Medical Imaging 2004*, 2004, pp. 16–27.
- [128] A. F. Frangi, W. J. Niessen, and M. A. Viergever, "Three-dimensional modeling for functional analysis of cardiac images, a review," *Medical Imaging, IEEE Transactions on*, vol. 20, no. 1, pp. 2–5, 2001.
- [129] J. Peters, O. Ecabert, C. Meyer, H. Schramm, R. Kneser, A. Groth, and J. Weese, "Automatic whole heart segmentation in static magnetic resonance image volumes," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2007*, Springer, 2007, pp. 402–410.
- [130] M. D. Cerqueira, N. J. Weissman, V. Dilsizian, A. K. Jacobs, S. Kaul, W. K. Laskey, D. J. Pennell, J. A. Rumberger, T. Ryan, M. S. Verani, and others, "Standardized myocardial segmentation and nomenclature for tomographic imaging of the heart a statement for healthcare professionals from the cardiac imaging committee of the Council on Clinical Cardiology of the American Heart Association," *Circulation*, vol. 105, no. 4, pp. 539–542, 2002.
- [131] *GNU Fortran*. Free Software Foundation, Inc., 2013.
- [132] *Cywin*. Red Hat, Inc.
- [133] C. A. Taylor and C. A. Figueroa, "Patient-Specific Modeling of Cardiovascular Mechanics," *Annual Review of Biomedical Engineering*, vol. 11, no. 1, pp. 109–134, Aug. 2009.
- [134] D. Chen, M. Müller-Eschner, H. von Tengg-Kobligk, D. Barber, D. Böckler, R. Hose, and Y. Ventikos, "A patient-specific study of type-B aortic dissection: evaluation of true-false lumen blood exchange," *Biomedical engineering online*, vol. 12, no. 1, p. 65, 2013.
- [135] O. Frank, "The basic shape of the arterial pulse. First treatise: Mathematical analysis," *Journal of molecular and cellular cardiology*, vol. 22, no. 3, pp. 255–277, 1990.
- [136] D. A. Silva Soto and D. R. Hose, "Report on Aortic Analysis Workflow Operation on Heidelberg Cases (Dr. Rusche Forschungsprojekt)," The University of Sheffield.
- [137] J. Peters, A. Lungu, F. M. Weber, I. Waechter-Stehle, D. R. Hose, and J. Weese, "Comparison of CFD-based and Bernoulli-based pressure drop estimates across the aortic valve enabled by shape-constrained deformable segmentation of cardiac CT images," presented at the 6th International Symposium on Biomedical Simulation ISBMS'14, London, UK, 2014.
- [138] *ANSYS Mechanical*. ANSYS, Inc.
- [139] *ANSYS ICEM*. ANSYS, Inc.
- [140] I. Voges, M. Jerosch-Herold, J. Hedderich, E. Pardun, C. Hart, D. D. Gabbert, J. H. Hansen, C. Petko, H.-H. Kramer, C. Rickers, and others, "Normal values of aortic dimensions,

distensibility, and pulse wave velocity in children and young adults: a cross-sectional study," *J Cardiovasc Magn Reson*, vol. 14, p. 77, 2012.