A Toolkit for the Visualization of Tensor Fields in Biomedical Finite Element Models

Burkhard Claus Wünsche

A dissertation submitted to the Department of Computer Science of the University of Auckland in partial fulfillment of the requirements for the degree of doctor of philosophy

University of Auckland
May 13, 2004

This thesis was assessed and approved by

Dr. Richard Lobb
Director of the Graphics Group, Department of Computer Science,
University of Auckland, Auckland, New Zealand

Dr. David Kenwright
Senior Research Scientist, Industrial Research Limited, Auckland, New Zealand

---

Prof. Dr. Christopher R. Johnson
Director, School of Computing
Director, Scientific Computing and Imaging Institute (SCI)
Director, Center for Bioelectric Field Modeling, Simulation, and Visualization (NIH NCRR)
Director, Engineering Scholars Program (ESP)
Co-Director, Advanced Visualization Technology Center (AVTC)
Distinguished Professor of Computer Science
Research Professor of Bioengineering
Adjunct Professor of Physics
Faculty Member, Computational Engineering and Science (CES) Program
Co-Founder, Visual Influence Inc.

Abstract

Medical imaging is an essential tool for improving the diagnoses, understanding and treatment of a large variety of diseases. Over the last century technology has advanced from the discovery of x-rays to a variety of 3D imaging tools such as magnetic resonance imaging, computed tomography, positron emission tomography and ultrasonography.

As a consequence the size and complexity of medical data sets has increased tremendously making it ever more difficult to understand, analyze, compare and communicate this data. Visualization is an attempt to simplify these tasks according to the motto "An image says more than a thousand words".

This thesis introduces a toolkit for visualizing biomedical data sets with a particular emphasis on second-order tensors, which are mathematically described by matrices and can be used to express complex tissue properties such as material deformation and water diffusion. The toolkit has a modular design which facilitates the comparison and exploration of multiple data sets. A novel field data structure allows the interactive creation of new measures and boolean filters are introduced as a universal visualization tool. Various new visualization methods are presented including new colour mapping techniques, ellipsoid-based textures and a line integral convolution texture for visualizing tensor fields.

To motivate the design and to assist in the use of the toolkit, guidelines for creating effective visualizations are derived by using perceptual concepts from cognitive science. A new classification for visual attributes according to representational accuracy, perceptual dimension and spatial requirements is presented and the results are used to derive values for the information content and information density of each attribute. A review and a classification of visualization icons completes the theoretical background.

The thesis concludes with two case studies. In the first case study the toolkit is used to visualize the strain tensor field in a healthy and a diseased human left ventricle. New insight into the cardiac mechanics is obtained by applying and modifying techniques traditionally used in solid mechanics and computational fluid dynamics.

The second case study explores ways to obtain in vivo information of the brain anatomy by visualizing and systematically exploring Diffusion Tensor Imaging (DTI) data. Three new techniques for the visualization of DTI data are presented: Barycentric colour maps allow an integrated view of different types of diffusion anisotropy.

Ellipsoid-based textures and Anisotropy Modulated Line Integral Convolution create images segmented by tissue type and incorporating a texture representing the 3D orientation of nerve fibers. The effectiveness of the exploration approach and the new visualization techniques are demonstrated by identifying various anatomical structures and features from a diffusion tensor data set of a healthy brain.

Acknowledgment

A large number of people have helped in various ways during the work for this thesis. I would like to thank in particular the following people:

- My supervisor, Dr. Richard Lobb, who provided essential advice, guidance, and encouragement.
- Prof. Peter J. Hunter and Assoc. Prof. Andrew J. Pullan from the Bioengineering Group of the University of Auckland, Auckland, New Zealand, for valuable discussions and for providing us with a numerical model of a dog heart.
- Dr. Alistair A. Young from the Department of Physiology of the University of Auckland, Auckland, New Zealand, for valuable discussions and for providing us with the models of the human left ventricle.
- Dr. Peter J. Basser from the National Institute of Health, Bethesda, Maryland, USA, for valuable discussions and Dr. Carlo Pierpaoli for providing us with the diffusion tensor data set of a healthy brain.
- Dr. Richard White of the Cleveland Clinic, Cleveland, Ohio, USA, who kindly provided tagged MRI data of a human left ventricle diagnosed with dilated cardiomyopathy.
- Prof. Richard L. M. Faull from the Department of Anatomy with Radiology of the University of Auckland, Auckland, New Zealand, for discussing with us the neuroanatomy of the brain and the resulting visualizations of it.
- Prof. Dr. Hans Hagen and Dr. Dieter Laaser from the University of Kaiserslautern, Kaiserslautern, Germany, for introducing me to the fields of Computer Graphics and CAGD.
- The staff in the Computer Science Department for providing advice and needed resources and making the workplace environment most pleasant.
- The University of Auckland for employing me since January 2001 as a full time lecturer and for allowing me to complete my PhD in the (rare) teaching free periods.

- And last but not least my family and friends for their support during my years at the University of Auckland.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Motivation

During the past 100 years medical imaging has advanced from Roentgen's original discovery of x-rays to a variety of 3D imaging tools such as magnetic resonance imaging, computed tomography, positron emission tomography and ultrasonography. Consequently the available medical data sets now comprise a diverse range of measurements such as tissue densities, blood flow velocity, and material strain. The development of mathematical models for organs and body parts has further increased the range of available data.

The size and complexity of biomedical data sets makes it increasingly difficult to understand, compare, analyze and communicate the data. Visualization is an attempt to simplify these tasks according to the motto "An image says more than a thousand words". Representing complex material properties, such as strains, as a single image improves the perception of features and patterns in the data, enables the recognition of relationships between different measures and facilitates the navigation through and interaction with complex and disparate sets of data.

The aim of this thesis is the development of a toolkit for visualizing biomedical data sets with a particular emphasis on second-order tensor fields which are a fundamental entity in engineering, physical sciences and biomedicine. Examples are stresses and strains in solids and viscous stresses and velocity gradients in fluid flows. Large amounts of tensor data are particularly difficult to interpret since an n-dimensional second-order tensor has the same complexity as an $n \times n$ matrix. An example for a biomedical tensor field is the strain field in the heart which describes the deformation of the heart muscle. It has been reported that abnormalities in the myocardial strain are visible before first symptoms of a heart attack occur [GZM97]. The goal of recording and visualizing cardiac data sets is to recognize and predict heart diseases which remain the biggest killer in the western world [MYPF00]. Understanding the deformation behaviour of the heart represented by the myocardial strain constitutes a major step towards this goal.

1.2 Contributions

The research accomplished in this thesis contributes to the disciplines of scientific visualization and biomedicine.

Chapter 4 summarizes results from cognitive science and offers guidelines for creating effective visualizations by applying perceptual concepts. We extend the traditional pipeline model for visualizing data to include two additional stages that take place within the observer: *visual perception* by the visual system and *cognition* by the human brain. An essential part of this model are visual attributes which we classify according to representational accuracy, perceptual dimension and spatial requirements. From these measures we obtain values for the information content and information density of a visual attribute.

Chapter 4 also presents a survey and an extended classification of visualization icons. By extracting the main visual attributes used for information mapping of each icon an informal measure of the suitability of an icon for different visualization tasks is obtained. The classifications for visual attributes and visualization icons combined with additional guidelines proposed in this chapter provide the scientist with a useful tool for selecting appropriate techniques for a given visualization task.

Chapter 5 presents a toolkit developed for exploring complex biomedical data sets. The contributions of this part are threefold: we suggest a modular design which facilitates the comparison and exploration of multiple data sets and visualizations. We also introduce a novel field data structure which allows interactive creation of new fields and we present boolean filters as a universal visualization tool. Our design incorporates finite element data structures and allows the definition of tissue properties in material coordinates, enables the selection of important structural components of the modeled organ (such as the inside or outside surface of the heart) and facilitates the computation of performance measures. We also suggest several improvements to some common visualization icons, e.g., we introduce cyclical colour maps and we describe tensor ellipsoids which encode the sign of an eigenvalue.

Chapter 6 and 7 present two case studies of practical importance. In the first case study we apply numerical concepts and visualization techniques traditionally used in solid mechanics and computational fluid dynamics to visualize models of a healthy and a diseased human left ventricle. We obtain new insight into the mechanics of the healthy and the diseased left ventricle and we facilitate the understanding of the complex deformation of the heart muscle by creating novel visualizations. Previously recorded results published as statistical data are confirmed and represented in an effective visual form. We also suggest a new hypothesis explaining the pumping behaviour of a left ventricle diagnosed with dilated cardiomyopathy.

The second case study presents visualizations of Diffusion Tensor Imaging (DTI) data. We propose (concurrently with another research group) a new method to extract and visualize nerve fiber tracts in the brain by using streamline integration. The nerve fibers tracts are visualized by streamtubes with a constant diameter or by hyperstreamlines which encode additionally the cellular water diffusion transverse to the fiber tract direction. Three additional novel visualization techniques

are presented: *Barycentric colour maps* allow an integrated view of different types of diffusion anisotropy; *ellipsoid-based textures* indicate diffusion direction and allow differentiation of tissue types but suffer from a lack of visual continuity; *Anisotropy Modulated Line Integral Convolution* (AMLIC) creates an image segmented by tissue type where white matter regions incorporate a texture which indicates the 3D orientation of nerve fiber tracts. The quality of our exploration approach and new visualization techniques is demonstrated by identifying various anatomical structures and features in a diffusion tensor data set of a healthy brain.

1.3 Thesis Overview

This thesis is divided into three parts. The first part comprises the chapters 2–4 and gives background information necessary for the understanding of this thesis from the fields of continuum mechanics, anatomy and physiology, medical imaging, bioengineering and scientific visualization. The second part consists of chapter 5 and introduces a visualization toolkit specifically designed for biomedical models and data sets. The capabilities of this toolkit are demonstrated in the third part of this work which comprises chapter 6 and 7 and contains case studies of biomedical data sets of the heart and the brain, respectively.

Chapter 2 introduces the finite element method which is frequently used to create biomedical models. We concentrate on the finite element representation of objects, i.e., the description of the object geometry and associated data fields by sample values and interpolation functions. The application of finite element analysis, i.e., finding a numerical solution to a set of partial differential equations governing the behaviour of a model, is inconsequential for the visualization process and is instead explained in appendix D. An important aspect of finite element modelling in the context of this thesis is the concept of material coordinates which are inherent to the modeled object and deform with it. As part of this opening chapter we also introduce the notions of stress and strain which are examples of second-order tensors and occur frequently in biological tissues.

Chapter 3 introduces two biomedical structures, the heart and the brain, which are used in the case studies presented later in this thesis. For each organ we describe its anatomy and where relevant its functioning, followed by a description of the corresponding biomedical models and associated data sets. The chapter is important in the context of this thesis for several reasons: it improves the understanding of the subsequent case studies and it enables the reader to compare visualization results with the actual organ anatomy and physiology. Furthermore the chapter helps to motivate the design and it helps to identify the required functionality of a visualization environment for biomedical structures. Finally the chapter introduces two examples of biomedical tensor data whose visualization is one of the main objectives of this work.

Chapter 4 is the final chapter of the introductory part of this thesis and it reviews the current state of the art of scientific visualization with an emphasis on tensor field visualization. The chapter starts with an overview of challenges encoun-

tered when visualizing multidimensional data and presents a summary of perceptual issues relevant for the design of a visualization. The next section introduces data transformation as a tool to simplify, expand, or modify data in order to make it more suitable for the visualization process. It is followed by a survey of existing visualization techniques for scalar, vector, and tensor fields. Methods which are important in the context of this thesis are dealt with in more detail. We conclude with a classification of visualization algorithms and summarize issues relevant for combining multiple visualization techniques into an effective visualization of a complex data set.

Chapter 5 constitutes the second and main part of this thesis. It describes an integrated visualization environment which we developed to investigate tensor field visualization methods. Although just a prototype implementation, the environment is a powerful visualization toolkit that allows the user to visualize complex biomedical data sets. In addition the application makes it possible to investigate and compare existing and novel visualization concepts. The chapter commences with an introduction of the top-level design of the toolkit and a description of the novel field data structure employed. Subsequently we explain various tools for user interaction, colour map design and the placement of visualization icons. We conclude with a summary of implemented visualization algorithms and tools for improving interaction with and perception of the data.

The final part of this work presents case studies of two biomedical data sets. In chapter 6 we evaluate the deformation of a healthy and a diseased human left ventricle. We present and explain various visualizations of the model and introduce methods for the efficient computation of performance measures from the finite element representation. We conclude with a discussion of our results and mention avenues for future research.

In chapter 7 we visualize the brain anatomy using Diffusion Tensor Imaging (DTI) data. Our approach starts with slice images familiar to the medical specialist and progressively expands the dimension and abstraction level of the representation in order to provide new insight into the data. The quality of our exploration approach and new visualization techniques are demonstrated by identifying various anatomical structures of the healthy brain.

CHAPTER 2

Strain, Stress, and Finite Element Modelling

During the past decade physically based modelling has emerged as an important new technique in biomedicine and computer science. An important subfield is the modelling of elastic bodies as used, for example, in computer animation [PW96] and surgical simulation [SBM+94, KGPG96, CDA99]. A mathematical description of elastic bodies is given by the theory of *elasticity*, the study of the deformation of a solid body under loading together with the resulting stresses and strains. The resulting mathematical models can be solved numerically using the finite element method (FEM).

This thesis explores the visualization of biomedical structures represented by finite element models. Consequently we are mainly interested in the finite element (FE) representation of objects, i.e., the description of their geometry and data fields by sample values and interpolation functions. Also important in the context of this thesis is the concept of material coordinates which are inherent to the modeled object and deform with it.

This chapter first introduces mathematical notations and definitions used in this thesis. Next we give a short introduction into strains, stresses and the theory of elasticity suitable for the computer scientist without an engineering background. Then follows an introduction to the FEM with a particular emphasis on the FE approximation of objects. The mathematical equations underlying the FE model and the employed solution procedures are summarized briefly and their application is demonstrated by two examples in appendix D.

The concepts introduced in this chapter are incorporated in the design of our visualization environment and are used in the case studies in chapter 6 and 7.

2.1 Notations and Definitions

Vectors are written in small bold letters and matrices in bold capital letters or small bold Greek letters. The components of a vector $\mathbf{u}$ are $u_i, \ldots, u_n$, or, if necessary to avoid confusion, u_x, u_y, u_z (in three dimensions). The components of a matrix $\mathbf{M}$ are m_{ij} $(i, j = 1, \ldots, n)$ so that the matrix $\mathbf{M}$ can also be expressed as (m_{ij}).

If the basis vectors of a vector space are constant, i.e., they have a fixed length and direction, the basis is called *Cartesian*. If the basis vectors are additionally unit and orthogonal the basis system is called *rectangular Cartesian* or just *Cartesian*. Unless stated otherwise we use a rectangular Cartesian coordinate system with the basis vectors $\mathbf{e}_i$, $i = 1, \ldots, n$. Vectors are by default taken to be column vectors. The vector $\mathbf{p}$ from the origin to an arbitrary point

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

is given by

$$\mathbf{p} = \sum_{i=0}^{n} x_i \mathbf{e}_i \tag{2.1}$$

The absolute value is denoted by $|\cdot|$, the determinant of a matrix by *det* or $|\cdot|$, the Euclidean norm by $||\cdot||$ and the gradient operator by $\nabla = \left(\frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_n}\right)^T$. Matrix transposition is indicated by ^{T}. Hence $\mathbf{u}^T\mathbf{v}$ is the dot product of the vectors $\mathbf{u}$ and $\mathbf{v}$ and ∇^T is the divergence operator, i.e., $\nabla^T\mathbf{f} = \frac{\partial f_1}{\partial x_1} + \ldots + \frac{\partial f_n}{\partial x_n}$. Furthermore $\mathbf{u}\mathbf{v}^T = \mathbf{W}$ defines the outer product of two vectors where $w_{ij} = u_i v_j$.

2.1.1 Tensors

A k-th rank tensor (or tensor of *order* k) in n-space is a set of n^k quantities which obey certain rules of transformation when the coordinate axes are rotated [App98, Wei]. A scalar is a tensor of order zero and a vector is a tensor of order one.

This thesis discusses the visualization of second-order tensors which are linear transformations between vectors and are represented by matrices. In the following unless stated otherwise the term *tensor* refers to a second-order tensor. Examples for tensors are stresses and strains which are explained in the next sections.

An important property of an n-dimensional symmetric second-order tensor $\mathbf{T}$ is that there always exist n *eigenvalues* λ_i and n mutually perpendicular *eigenvectors* $\mathbf{v}_i$ such that

$$\mathbf{T}\mathbf{v}_i = \lambda_i \mathbf{v}_i \quad i = 1, \ldots, n \tag{2.2}$$

The above equations can be rewritten as

$$(\mathbf{T} - \lambda_i \mathbf{I})\,\mathbf{v}_i = 0 \quad i = 1, \ldots, n$$

The resulting linear system of equations has non-trivial solutions if and only if the matrix $(\mathbf{T} - \lambda_i \mathbf{I})$ is invertible. Hence the eigenvectors of the tensors are given by the roots of the *characteristic polynomial*

$$det\,(\mathbf{T} - \lambda \mathbf{I}) = 0$$

The matrix representation of a tensor is dependent on the reference coordinate system used. Suppose $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ and $\{\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2, \hat{\mathbf{e}}_3\}$ are unit vectors spanning two Cartesian coordinate systems (i.e., the unit vectors are orthonormal). The components of a tensor $\mathbf{T}$ with respect to $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ are expressed with respect to $\{\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2, \hat{\mathbf{e}}_3\}$ by the transformation [LRK86]

$$\hat{\mathbf{T}} = \mathbf{Q}^T \mathbf{T} \mathbf{Q} \tag{2.3}$$

where $\mathbf{Q} = (q_{ij})$ is the coordinate transformation matrix defined through the equations

$$\hat{\mathbf{e}}_i = \mathbf{Q}\mathbf{e}_i \quad i = 1, \ldots, 3$$

Note that

$$q_{ij} = q_{ij}\, \mathbf{e}_i^T \mathbf{e}_i = \sum_j q_{ij}\, \mathbf{e}_i^T \mathbf{e}_j = \mathbf{e}_i^T \mathbf{Q} \mathbf{e}_j = \mathbf{e}_i^T \hat{\mathbf{e}}_j = \cos(\mathbf{e}_i, \hat{\mathbf{e}}_j)$$

since $\mathbf{e}_i^T \mathbf{e}_j = \delta_{ij}$.

Several coordinate system independent measures can be derived from a tensor. In three dimensions these measures include the *tensor invariants* [LRK86]:

$$I_1 = trace(\mathbf{T}) = T_{11} + T_{22} + T_{33} \tag{2.4}$$

$$I_2 = \begin{vmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{vmatrix} + \begin{vmatrix} T_{11} & T_{13} \\ T_{31} & T_{33} \end{vmatrix} + \begin{vmatrix} T_{22} & T_{23} \\ T_{32} & T_{33} \end{vmatrix} \tag{2.5}$$

$$I_3 = |\mathbf{T}| \tag{2.6}$$

The three tensor invariants are the coefficients of the cubic equation

$$\lambda^3 + I_1 \lambda^2 + I_2 \lambda + I_3 = 0$$

whose solutions are the eigenvalues of $\mathbf{T}$.

Note that the first tensor invariant can also be expressed as $I_1 = \lambda_1 + \lambda_2 + \lambda_3$. For a proof choose the eigenvectors of the tensor as a reference coordinate system. Representing the tensor in the resulting coordinate system gives a matrix with zero off-diagonal elements and the eigenvalues as diagonal elements.

If we represent the tensor by an ellipsoid with the lengths and directions of its principal axes given by the eigenvalues and eigenvectors then the first tensor invariant is proportional to the average of the axis lengths, the third invariant is proportional to the ellipsoid volume and the second invariant can be interpreted as a surface measure [Ale00].

2.2 Curvilinear Coordinates

While many modeling problems are formulated in three-dimensional Cartesian space, some problems, especially those involving curved geometric bodies, are better posed in a non-Cartesian *curvilinear* coordinate system. This subsection introduces curvilinear coordinate systems and shows how to transform between them and Cartesian coordinates.

Given an n-dimensional rectangular Cartesian space, a point $\mathbf{p}$ is given by a set of rectangular Cartesian coordinates $(r_1, \ldots, r_n)$. A new set of coordinates $(q_1, \ldots, q_n)$ can be defined by the transformation

$$q_i = q_i(r_1, \ldots, r_n) \quad i = 1, \ldots, n$$

If the Jacobian does not vanish, i.e.,

$$|\mathbf{J}| = \left|\frac{\partial q_i}{\partial r_j}\right| = \begin{vmatrix} \frac{\partial q_1}{\partial r_1} & \cdots & \frac{\partial q_1}{\partial r_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial q_n}{\partial r_1} & \vdots & \frac{\partial q_n}{\partial r_n} \end{vmatrix} \neq 0 \quad (i, j = 1, \ldots, n)$$

an inverse transformation

$$r_j = r_j(q_1, \ldots, q_n) \quad j = 1, \ldots, n$$

does exists [Heu81, p. 300] with $(\frac{\partial r_j}{\partial q_i}) = \mathbf{J}^{-1}$.

If q_i is changed and q_j $(j \neq i)$ is held constant then the vector $\mathbf{r}$ will vary along a curve. This curve is called the *coordinate curve* for q_i. When the above transformation is linear the new coordinate system is again Cartesian, though not necessarily orthogonal or normalized. When the above transformation is non-linear the new coordinate system is called *curvilinear* [Bat82]. An example of curvilinear coordinates is explained in detail in appendix C.2.

The position vector $\mathbf{p}$ can now be represented as a function of the new coordinates

$$\mathbf{p} = \mathbf{p}(q_1, \ldots, q_n)$$

The partial derivatives

$$\frac{\partial \mathbf{p}}{\partial q_i} \equiv \hat{\mathbf{e}}_i \quad i = 1, \ldots, n$$

are tangent vectors of the coordinate curves q_i. The vectors $\hat{\mathbf{e}}_i$ form a basis, the so-called *unitary system*, for the coordinates $(q_1, \ldots, q_n)$. In general the vectors are neither orthogonal nor normalized and using equation 2.1 can be expressed in terms of the Cartesian basis as

$$\hat{\mathbf{e}}_i = \frac{\partial \mathbf{p}}{\partial q_i} = \sum_{j=1}^{n} \frac{\partial r_j}{\partial q_i} \mathbf{e}_j \tag{2.7}$$

Consequently a vector

$$\hat{\mathbf{v}} = \sum_{i=1}^{n} \hat{v}_i \hat{\mathbf{e}}_i$$

expressed in curvilinear coordinates can be written with respect to the original basis as

$$\hat{\mathbf{v}} = \sum_{i=1}^{n} \hat{v}_i \hat{\mathbf{e}}_i = \sum_{i=1}^{n} \hat{v}_i \left(\sum_{j=1}^{n} \frac{\partial r_j}{\partial q_i} \mathbf{e}_j \right) = \sum_{j=1}^{n} \sum_{i=1}^{n} \frac{\partial r_j}{\partial q_i} \hat{v}_i \mathbf{e}_j = \sum_{j=1}^{n} \bar{v}_j \mathbf{e}_j$$

where

$$\bar{v}_j = \sum_{i=1}^{n} \frac{\partial r_j}{\partial q_i} \hat{v}_i$$

are the components of the vector in Cartesian coordinates. In matrix form this simplifies to

$$\bar{\mathbf{v}} = \mathbf{J}^{-1} \hat{\mathbf{v}} \tag{2.8}$$

An example of the representation of a vector with respect to different coordinate systems is given in subsection C.2.1.

Similarly a second-order tensor

$$\hat{\mathbf{T}} = \sum_{i=1}^{n} \sum_{j=1}^{n} \hat{t}_{ij} \hat{\mathbf{e}}_i \hat{\mathbf{e}}_j$$

expressed in curvilinear coordinates is written with respect to the Cartesian basis as

$$\hat{\mathbf{T}} = \sum_{i=1}^{n} \sum_{j=1}^{n} \hat{t}_{ij} \left(\sum_{k=1}^{n} \frac{\partial r_k}{\partial q_i} \mathbf{e}_k \right) \left(\sum_{l=1}^{n} \frac{\partial r_l}{\partial q_j} \mathbf{e}_l \right) = \sum_{k=1}^{n} \sum_{l=1}^{n} \bar{t}_{kl} \mathbf{e}_k \mathbf{e}_l$$

where

$$\bar{t}_{kl} = \sum_{i=1}^{n} \sum_{j=1}^{n} \hat{t}_{ij} \frac{\partial r_k}{\partial q_i} \frac{\partial r_l}{\partial q_j}$$

are the components of the tensor in Cartesian coordinates. In matrix form this simplifies to

$$\bar{\mathbf{T}} = \mathbf{J}^{-1} \hat{\mathbf{T}} (\mathbf{J}^{-1})^T \tag{2.9}$$

2.3 Linear Elasticity

2.3.1 Displacement and Strain

An elastic body under an applied load deforms into a new shape. The theory of linear elasticity provides a mathematical description for the displacement the body undergoes. For a one-dimensional example consider a thin rubber band as pictured in figure 2.1. Two arbitrary points P and Q are marked at the positions x and $x + \delta x$, respectively. After deformation these points move to the positions $x + u$ and $x + u + \delta x + \delta u$, respectively, where u is called the displacement. The total length increase (displacement) between these points is δu. The *strain* is now defined as the increase per unit length, i.e., it is the *displacement gradient*, which in one dimension is defined as

$$\epsilon_x = \lim_{\delta x \to 0} \frac{\delta u}{\delta x} = \frac{du}{dx}$$

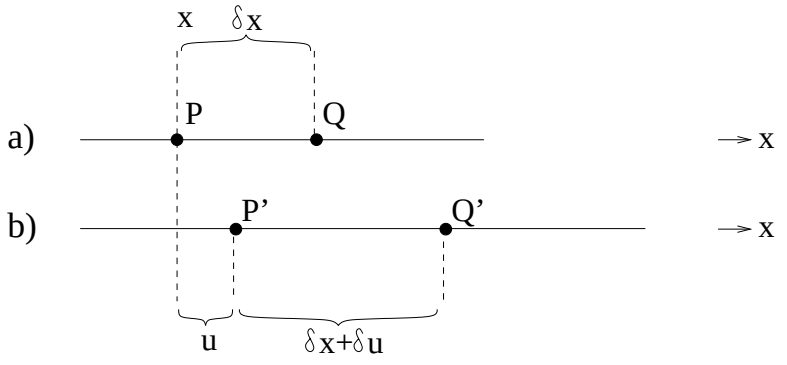


Figure 2.1. *A rubberband before (a) and after (b) stretching.*

In higher dimensions the situation is more complicated. Figure 2.2 shows a body before and after deformation. Under deformation the points P and Q move to position $\mathbf{x}' = \mathbf{x} + \mathbf{u}(\mathbf{x})$ and $\mathbf{x}' + d\mathbf{x}' = \mathbf{x} + d\mathbf{x} + \mathbf{u}(\mathbf{x} + d\mathbf{x})$, respectively, where $\mathbf{u}$ is called the *displacement field*.

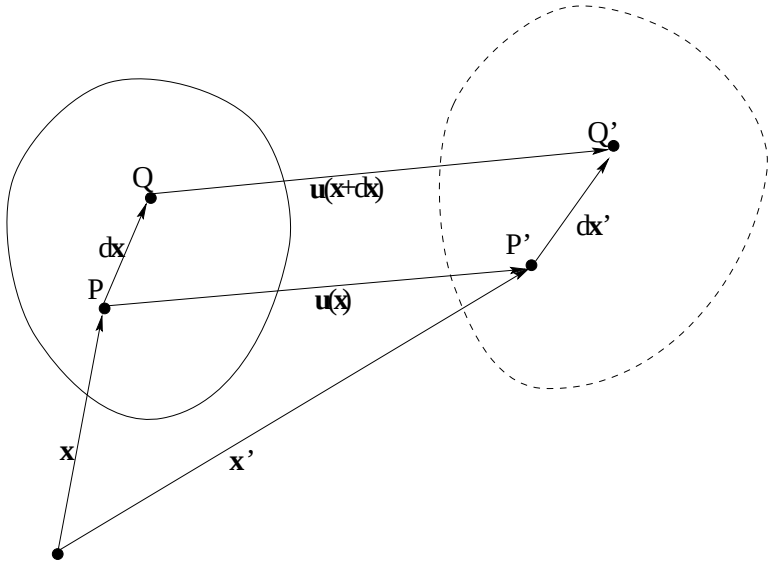


Figure 2.2. *A body before and after deformation.*

If the points are only an infinitesimal distance apart the vector between the deformed points

$$d\mathbf{x}' = d\mathbf{x} + \mathbf{u}(\mathbf{x} + d\mathbf{x}) - \mathbf{u}(\mathbf{x})$$

can be written as [LRK86]

$$d\mathbf{x}' = d\mathbf{x} + (\nabla \mathbf{u}) d\mathbf{x}$$

where the second-order tensor

$$\nabla \mathbf{u} = \begin{pmatrix} \frac{\partial u_1}{\partial x_1} & \frac{\partial u_1}{\partial x_2} & \frac{\partial u_1}{\partial x_3} \\ \frac{\partial u_2}{\partial x_1} & \frac{\partial u_2}{\partial x_2} & \frac{\partial u_2}{\partial x_3} \\ \frac{\partial u_3}{\partial x_1} & \frac{\partial u_3}{\partial x_2} & \frac{\partial u_3}{\partial x_3} \end{pmatrix} \tag{2.10}$$

is known as the *displacement gradient*.

It can be seen that if $\nabla \mathbf{u} = 0$ then $d\mathbf{x}' = d\mathbf{x}$ and the motion in the neighborhood of point P is that of a rigid body translation. The information about the material deformation around P is contained in $\nabla \mathbf{u}$. It is desirable to define an entity which contains only information about deformation, but not about rotation. To do this consider two material vectors $d\mathbf{x}_1$ and $d\mathbf{x}_2$ issuing from point P. Their dot product after transformation is ([LRK86])

$$(d\mathbf{x}_1')^T d\mathbf{x}_2' = d\mathbf{x}_1^T d\mathbf{x}_2 + 2 d\mathbf{x}_1^T \mathbf{E}^* d\mathbf{x}_2$$

where the symmetric second-order tensor

$$\mathbf{E}^* = \frac{1}{2}\left((\nabla \mathbf{u}) + (\nabla \mathbf{u})^T + (\nabla \mathbf{u})^T (\nabla \mathbf{u})\right)$$

is the *Lagrangian strain tensor*. Note that if $\mathbf{E}^* = 0$ the lengths and angles between the material vectors $d\mathbf{x}_1$ and $d\mathbf{x}_2$ remain unchanged, i.e., the deformation $\nabla \mathbf{u}$ around point P is a rigid body transformation (i.e., rotation or translation). The components of $\mathbf{E}^*$ are

$$E_{ij}^* = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} + \sum_{k=1}^{3} \frac{\partial u_k}{\partial x_i}\frac{\partial u_k}{\partial x_j}\right)$$

For small deformations the displacement gradients $\partial u_i / \partial x_j$ are small and the quadratic term of $\mathbf{E}^*$ can be neglected giving the *strain tensor* $\boldsymbol{\epsilon}$ with the components

$$\epsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{2.11}$$

Lai et al. show [LRK86] that in this case ϵ_{ii} can be interpreted as the unit elongation (increase per unit length) of a material element in the x_i direction. The components ϵ_{ii} are called the *normal strains*. Furthermore the terms $2\epsilon_{ij}$, $i \neq j$ can be interpreted as the decrease in angle in radians between two material vectors initially in the x_i and x_j directions and are known as *total shear strains*. The components ϵ_{ij} are known as *average shear strains* or just *shear strains* [eFu].

Figure 2.3 illustrates these concepts. In the left part of the figure an infinitesimal quadratic area in the xy-plane has been strained without change of area. The changes in length perpendicular to the given material direction are $\partial u_2 / \partial x_1$ and $\partial u_1 / \partial x_2$, respectively. By rotating the element as shown on the right hand side it can be seen that the deformation has been a simple shear, i.e., the deformation is equivalent to a translation in x_1-direction by an amount proportional to the x_2-coordinate followed by a rotation. The decrease in angle between the axes of the infinitesimal square element is approximately $2\epsilon_{12}$.

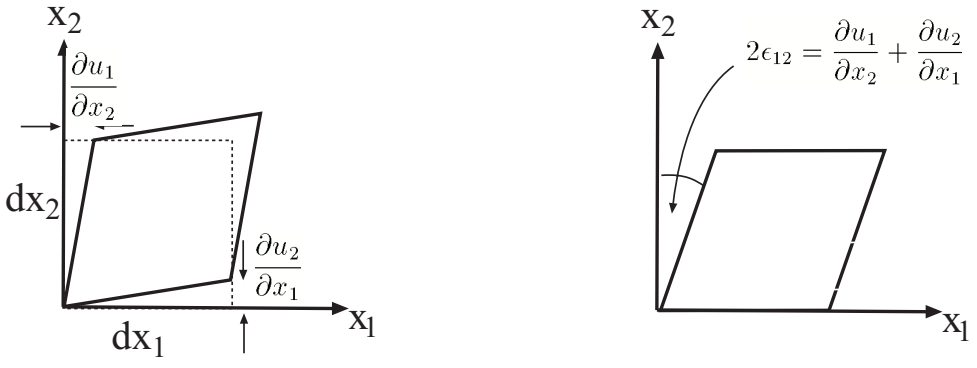


Figure 2.3. *An infinitesimal quadratic area in the xy-plane has been strained without change of area (left). Rotating the element (right) shows that the deformation is equivalent to a translation in x_1-direction by an amount proportional to the x_2-coordinate followed by a rotation. The decrease in angle between the axes of the infinitesimal square element is approximately $2\epsilon_{12}$.*

Note that by definition the strain tensor $\boldsymbol{\epsilon}$ is symmetric so that equation 2.2 holds. The eigenvectors $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$ of $\boldsymbol{\epsilon}$ are the *principal directions* of the strain, i.e., the directions in which there is no shear strain. The eigenvalues λ_1, λ_2, and λ_3 are the *principal strains* and give the unit elongations in the principal directions. The maximum, medium, and minimum eigenvalue are called the *maximum, medium, and minimum principal strain*, respectively.

2.3.2 Stress

The previous subsection gave a purely kinematic description of the motion and deformation of an elastic body without considering the internal and external forces causing it. Internal forces are body forces[1] acting throughout the body and external forces are surface forces acting on a real or imagined surface separating the body. The surface force at a point of the surface is described by a *stress vector*.

Consider a plane S with normal $\mathbf{n}$ through a point P of the elastic body as shown in figure 2.4. Let $\Delta\mathbf{f}$ be the force acting on a small area ΔA containing P. The stress vector $\mathbf{t}_n$ in P is defined as

$$\mathbf{t}_n = \lim_{\Delta A \to 0} \frac{\Delta \mathbf{f}}{\Delta A}$$

In classical continuum theory the resulting stress vector is the same for all surfaces through point P with a tangent plane S at P. It can be shown ([LRK86]) that

[1]Body forces are forces that act on all particles in a body as a result of some external body or effect not in direct contact with the body under consideration. An example of this is the gravitational force exerted on a body. This type of force is defined as a force intensity per unit mass or per unit volume at a point in the continuum. Hence, when, for example, considering gravity the body force (per unit mass) is the gravitational acceleration g.

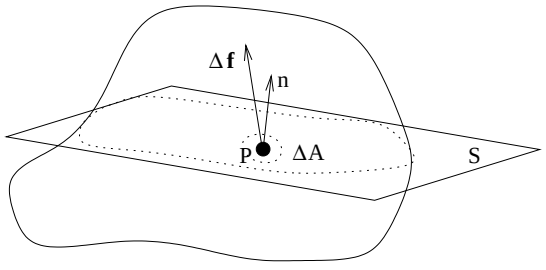


Figure 2.4. *Definition of a stress vector.*

independent of the choice of S

$$\mathbf{t}_n = \boldsymbol{\sigma}\mathbf{n}$$

where the linear operator $\boldsymbol{\sigma}$ is the *stress tensor* in P.

To interpret the components of the stress tensor $\boldsymbol{\sigma}$ consider an infinitesimal small axis-aligned cube as shown in figure 2.5.

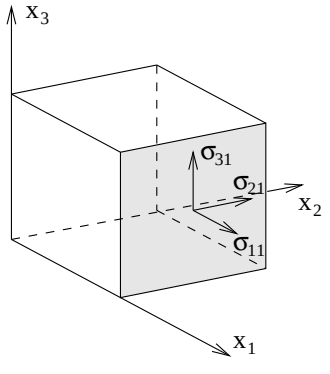


Figure 2.5. *The Stress components σ_{11}, σ_{21}, and σ_{31} are interpreted as components of a force acting on the surface orthogonal to $\mathbf{e}_1$ (shaded) of an infinitesimal small axis-aligned unit cube.*

The stress tensor components σ_{11}, σ_{21}, and σ_{31} are the components of the stress vector $\mathbf{t}_{\mathbf{e}_1}$, i.e., they are the components of the force acting on an infinitesimal surface orthogonal to $\mathbf{e}_1$. The other components of $\boldsymbol{\sigma}$ are interpreted similarly. The diagonal elements σ_{11}, σ_{22}, and σ_{33} are called the *normal stresses* and the off-diagonal elements σ_{12}, σ_{13}, σ_{23}, σ_{21}, σ_{31}, and σ_{32} are called the *shear stresses*. By using the conservation of angular momentum equation it can be shown [HB96] that $\boldsymbol{\sigma}$ is in fact symmetric[2], i.e., $\sigma_{12} = \sigma_{21}$, $\sigma_{13} = \sigma_{31}$, $\sigma_{23} = \sigma_{32}$.

[2]This is not the case if there are body moments per unit volume such as for a polarized anisotropic dielectric solid [LRK86].

In this thesis we concentrate on the visualization of strain tensors and diffusion tensors which are always symmetric. In section 5.9 we use as an example the stress field in a plate under an uniaxial load for which the resulting stress tensors are also symmetric.

As for the strain tensor the three eigenvectors of the symmetric stress tensor $\boldsymbol{\sigma}$ give the *principal directions* of the stress and the eigenvalues give the *principal stresses*. Each principal direction gives the normal direction of a plane on which the shear stresses are zero and the normal stress is the *principal stress*.

2.3.3 Model of a Linear Elastic Solid

Every continuum in motion must fulfill Newton's laws of motion. For steady state solutions the continuum doesn't experience acceleration and the second law of motion reduces to the *equilibrium equations*

$$\sum_{j=1}^{n} \frac{\partial \sigma_{ij}}{\partial x_j} + \rho g_i = \sum_{j=1}^{n} \frac{\partial \sigma_{ij}}{\partial x_j} + f_i = 0 \qquad \text{i=1,...,n} \tag{2.12}$$

where σ is the stress tensor, $\mathbf{g}$ is the body force per unit mass, ρ is the mass density and n is the number of dimensions [Bur87]. The product of body force per unit mass and mass density is often referred to as *internal load* $\mathbf{f}$.

The above system of equations is valid for every continuum. However, in order to describe the response of a specific material under a specific loading additional material parameters are necessary.

Figure 2.6 shows a slender cylinder with length l, diameter d, and cross-sectional area A under a uniaxial load p. If the length of the cylinder increases linearly with the applied load and the diameter decreases linearly with it the body is called a *linear elastic solid*.

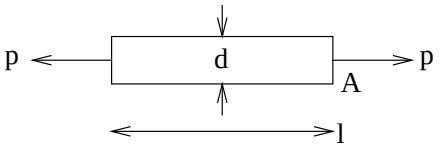


Figure 2.6. *A slender cylinder of a linear-elastic material under a uniaxial load.*

In order to define material properties, material behaviour which is independent of the specimen size must be identified. Appropriate measures are the axial stress

$$\sigma_a = \frac{p}{A}$$

the axial strain

$$\epsilon_a = \frac{\Delta l}{l}$$

and the lateral strain

$$\epsilon_d = \frac{\Delta d}{d}$$

Two material coefficients can now be derived. The *Young's modulus* (or, *modulus of elasticity*) defines the ratio of axial stress to axial strain under uniaxial loading

$$E = \frac{\sigma_a}{\epsilon_a}$$

and the *Poisson's ratio* defines the ratio of lateral strain to axial strain

$$\nu = \frac{\epsilon_d}{\epsilon_a}$$

For some materials (e.g., fibrous materials) the Young's modulus and Poisson's ratio might depend on the orientation of the cylindrical test specimen with respect to the material microstructure. In this case the material is said to be *anisotropic* with respect to its elastic properties. Otherwise the material is *isotropic* with respect to its elastic properties. If the elastic properties are the same over the whole material it is said to be *homogeneous*, whereas if the elastic properties vary from one neighbourhood to another the material is said to be *inhomogeneous*.

For a linear elastic material the relationship between stress and strain is a linear one, i.e.,

$$\sigma_{ij} = \sum_{k=1}^{n} \sum_{l=1}^{n} C_{ijkl} E^*_{kl}$$

or assuming small deformations

$$\sigma_{ij} = \sum_{k=1}^{n} \sum_{l=1}^{n} C_{ijkl} \epsilon_{kl} \tag{2.13}$$

where the fourth-order tensor $\mathbf{C}$ is known as the *elasticity tensor*.

Since $\boldsymbol{\sigma}$ and $\boldsymbol{\epsilon}$ are symmetric it can be shown that $\mathbf{C}$ contains 36 degrees of freedom, i.e., 36 material constants are necessary to completely describe the elasticity tensor [LRK86]. The stress-strain relationship simplifies considerably if the material is isotropic. In this case the elasticity tensor is *isotropic*, i.e., the tensor is invariant under orthogonal transformations (see glossary) and equation 2.13 can be shown to reduce to

$$\boldsymbol{\sigma} = \lambda(\epsilon_{11} + \epsilon_{22} + \epsilon_{33})\mathbf{I} + 2\mu\boldsymbol{\epsilon} \tag{2.14}$$

i.e., the stress-strain relationship is fully described by two material constants λ and μ [LRK86]. The material constants, known as *Lamé's constants*, can be expressed in terms of the previously introduced Young's modulus and Poisson's ratio as follows

$$\begin{aligned} \lambda &= \frac{\nu E}{(1+\nu)(1-2\nu)} \\ \mu &= \frac{E}{2(1+\nu)} \end{aligned}$$

Assuming an isotropic material equation 2.13 (Hooke's law) simplifies to

$$\begin{pmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{12} \\ \sigma_{23} \\ \sigma_{13} \end{pmatrix} = \frac{E}{(1+\nu)(1-2\nu)} \begin{pmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & 1-2\nu & 0 & 0 \\ 0 & 0 & 0 & 0 & 1-2\nu & 0 \\ 0 & 0 & 0 & 0 & 0 & 1-2\nu \end{pmatrix} \begin{pmatrix} \epsilon_{11} \\ \epsilon_{22} \\ \epsilon_{33} \\ \epsilon_{12} \\ \epsilon_{23} \\ \epsilon_{13} \end{pmatrix}$$

or in matrix form

$$\boldsymbol{\sigma} = \mathbf{C}\boldsymbol{\epsilon} \tag{2.15}$$

where the stress and strain components are represented in vector form as

$$\boldsymbol{\sigma}^T = \begin{pmatrix} \sigma_{11} & \sigma_{22} & \sigma_{33} & \sigma_{12} & \sigma_{23} & \sigma_{13} \end{pmatrix}$$

and

$$\boldsymbol{\epsilon}^T = \begin{pmatrix} \epsilon_{11} & \epsilon_{22} & \epsilon_{33} & \epsilon_{12} & \epsilon_{23} & \epsilon_{13} \end{pmatrix}$$

respectively, and the matrix representation of the elasticity tensor is

$$\mathbf{C} = \frac{E}{(1+\nu)(1-2\nu)} \begin{pmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & 1-2\nu & 0 & 0 \\ 0 & 0 & 0 & 0 & 1-2\nu & 0 \\ 0 & 0 & 0 & 0 & 0 & 1-2\nu \end{pmatrix}$$

The finite element modeller implemented during the course of this thesis allows the user to simulate isotropic linear elastic materials under small deformations. Typical examples of such materials are metals and rubber. Even though biological materials are usually highly anisotropic and have a non-linear stress-strain relationship, the previously described idealized material properties have been used for biomedical applications such as real-time surgery simulation [BNC96] and the modelling of bones [Yet89, chapter 14].

2.4 The Finite Element Method

The Finite Element Method (FEM) became popular in the 1960's as a tool to solve problems in structural mechanics numerically using computers. Solutions are obtained by numerically solving partial differential equations predicting the response of physical systems subjected to external influences [Bur87]. A solution is computed by subdividing the domain of the physical system into a finite element (FE) mesh and by approximating the governing differential equations by integral expressions over mesh elements.

Over the last two decades the FEM has become increasingly popular and is now an accepted tool in the fields of biomedicine (bioengineering) and computer graphics (modelling and animation). Applications include surgical simulation [SBM+94],

muscle modelling [CZ92], cloth modeling [EDC96], scattered data modelling [XZ00] and facial animation [Lif].

This thesis is concerned with the visualization of finite element models of biomedical structures. Two such models are introduced in the next chapter. Since the visualization process only requires knowledge of the representation of the object geometry and the associated data fields this section emphasizes the FE approximation and interpolation rather than the methods of solving the equations. Another feature of the FE method important in this thesis is the definition of so-called material coordinates which reflect the geometry and/or physical properties of the modelled object and deform with it.

This section first introduces concepts used in the FEM. Then follows an introduction to the FE discretization and a description of various FE interpolation functions used to approximate the geometry and data fields of a model. The subsequent subsections introduce the concepts of world and material coordinates and contain an overview of numerical integration techniques used for the evaluation of FE integrals. We conclude this section with a short overview of the FE solution process. A more detailed explanation of the FE solution process using two examples is given in appendix D. An introduction to the FEM suitable for non-experts is found in [HP02, Bur87, HB96].

2.4.1 Concepts

Burnett introduces the following four concepts in any FE problem [Bur87]:

- The *system* is an object composed of various materials whose properties are described by material parameters.
- The *governing equations* define the behaviour of the system and consist usually of differential equations expressing conservation principles of some physical property or variational principles such as the minimization of a physical property. They may also include constitutive equations which contain physical properties of the materials that constitute the system. The free[3] variable or variables in the governing equations represent the unknown FE solution. In the following we assume that governing equation contains just one free variable.
- The *domain* of the problem is the space over which the free variable is defined. Usually the domain is the region of space occupied by the system and/or the time interval over which the system changes its state.
- *Loading conditions* are externally originating physical quantities that interact with the system and cause its state to change. Loads acting in the interior of the domain are called *interior loads* and are part of the governing equations.

[3] In Engineering a quantity which is varied in an experiment is often referred to as *independent variable* [ERC]. Chapter 4 will use a different definition of the term *independent variable* in conjunction with the visualization of multidimensional data sets. We use instead the terms *free variable* or *unknown variable* for the varied quantity.

> Loads acting on the boundary of the domain are called *boundary loads* and form separate equations called *boundary conditions*.

The solution of the FE problem is a solution for the free variable which fulfills the governing equations and the boundary conditions over the domain.

Figure 2.7 explains the above terminology with an example: The system is a thin metal rod which is connected on the left side to a heat source with a constant temperature of $T = 30°C$ and on the right side to an energy source with a *heat flux* (transfer of energy) of $q = 100W/m^2$. The domain is the interval $[0, 2]$ on the x-axis. The governing equation is

$$-\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) = 0$$

where $k = 10Wm^{-1\circ}C^{-1}$ is the thermal conductivity of the metal and the unknown or free variable is the temperature T. Finally the system has no internal loads and it has the boundary conditions

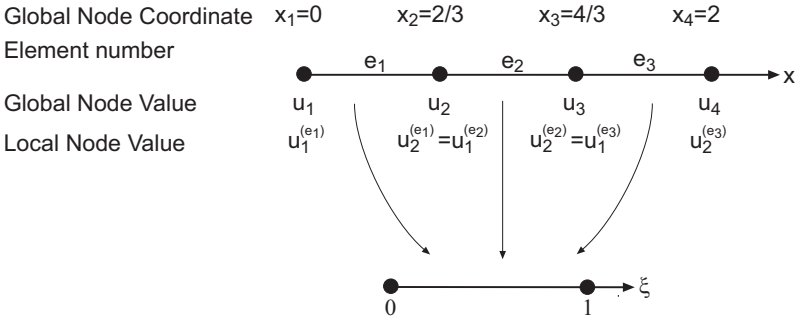
$$\begin{aligned} T &= 30 \quad \text{at} \quad x = 0 \\ q = -k\frac{\partial T}{\partial x} &= 100 \quad \text{at} \quad x = 2 \end{aligned}$$

Figure 2.7. *Mathematical description of the 1D heat conduction problem.*

2.4.2 Finite Element Approximation

A finite element approximation involves discretizing the domain into a *finite element mesh* and representing the solution $u(x)$ for the free variable by values at the nodes of the mesh. The solution is extended over the entire domain by interpolating the nodal values of an element using *element basis functions*. Instead of defining element basis functions for each element it is convenient to define a *parent element* and to define a mapping between it and the element.

As an example consider the metal rod in figure 2.7. In order to use a common notation for the unknown we denote the temperature distribution with $u(x)$ and its finite element approximation with $\tilde{u}(x)$. Figure 2.8 shows the rod approximated by three elements $e_1, \ldots, e_3$ with the nodes $n_1, \ldots, n_4$, which have the nodal coordinates $x_1, \ldots, x_4$ and the nodal values $u_1, \ldots, u_4$, respectively. Each element e_j has two local nodes $n_1^{(e_j)}$ and $n_2^{(e_j)}$ which are associated with their corresponding global node n_i



i=Δ(j,k)	j	k
1	1	1
2	1	2
2	2	1
3	2	2
3	3	1
4	3	2

Figure 2.8. *Finite Element Approximation of the metal rod in figure 2.7 and the corresponding connectivity matrix.*

by a connectivity matrix $i = \Delta(j, k)$ where i is the global node index, j the element index, and k the local node index. Similarly we have local nodal coordinates $x_k^{(e_j)}$ and local nodal values $u_k^{(e_j)}$. Each element e_j is associated with a *parent element* defined by the $[0, 1]$ interval in the ξ parameter space, i.e., $\xi(0) = x_1^{(e_j)}$ and $\xi(1) = x_2^{(e_j)}$. Any point in the domain then has a (global) world coordinate x and an associated ξ-coordinate (*material coordinate* or *element coordinate*) ξ. If the same basis functions are used for the geometry and the unknown variables the mapping $x(\xi)$ from material to world coordinates is called *isoparametric mapping*.

Linear Lagrange Basis Functions

A simple approximation $\tilde{u}^{(e)}$ of the temperature distribution over the element e is obtained by connecting the element nodal values with line segments. Mathematically this is expressed as an interpolation of the nodal values using *Linear Lagrange basis functions* ϕ_i

$$\tilde{u}^{(e)}(\xi) = \sum_{i=1}^{n} u_i^{(e)} \phi_i(\xi) \tag{2.16}$$

where $n = 2$ and

$$\phi_1(\xi) = 1 - \xi \qquad \phi_2(\xi) = \xi \tag{2.17}$$

The world coordinates of the corresponding values are obtained analogously by interpolating the element nodal coordinates

$$\tilde{x}^{(e)}(\xi) = \sum_{i=1}^{n} x_i^{(e)} \phi_i(\xi)$$

The basis functions, displayed in figure 2.9, have the property that $\phi_1(0) = 1$ and $\phi_2(0) = 0$. Hence $\tilde{u}^{(e)}(0) = u_1^{(e)}$ and $\tilde{x}^{(e)}(0) = x_1^{(e)}$. Similarly $\phi_1(1) = 0$ and $\phi_2(1) = 1$ such that $\tilde{u}^{(e)}(1) = u_2^{(e)}$ and $\tilde{x}^{(e)}(1) = x_2^{(e)}$. As a result the interpolation is C^0 continuous across element boundaries if nodes are shared between neighbouring elements.

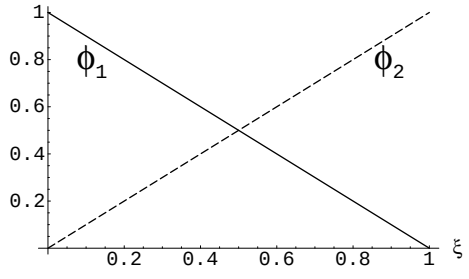


Figure 2.9. *Linear Lagrange basis functions.*

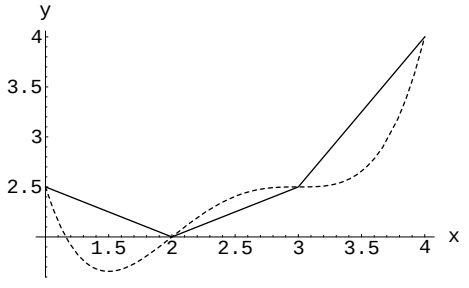


Figure 2.10. *Piecewise linear Lagrange interpolation (solid) of a quartic function (dashed).*

As an example consider the linear Lagrange interpolation of the function $u(x) = 16 - 27x + 18x^2 - 5x^3 + 0.5x^4$ with the nodal coordinates $x_i = i$, $i = 1, \ldots, 4$ and the nodal values $u_i = u(i)$. Figure 2.10 shows the original function as a dashed curve and the piecewise linear interpolation as a solid polyline.

Cubic Hermite Basis Functions

A smoother approximation than with the linear Lagrange interpolation is achieved by using a cubic Hermite interpolation. The interpolation requires in addition to the element nodal values u_i and coordinates x_i also the corresponding first derivatives $\left(\frac{du}{d\xi}\right)_i$ and $\left(\frac{dx}{d\xi}\right)_i$. If nodal values and derivatives are shared between elements the interpolation is C^1 continuous across element boundaries. The resulting interpolation function is

$$\tilde{u}^{(e)}(\xi) = \sum_{i=1}^{n} \left(u_i^{(e)} \phi_i^0(\xi) + \left(\frac{du}{d\xi}\right)_i^{(e)} \phi_i^1(\xi) \right) \tag{2.18}$$

where $n = 2$. The element basis functions

$$\begin{array}{ll} \phi_1^0(\xi) = 1 - 3\xi^2 + 2\xi^3 & \phi_2^0(\xi) = \xi^2 (3 - 2\xi) \\ \phi_1^1(\xi) = \xi (\xi - 1)^2 & \phi_2^1(\xi) = \xi^2 (\xi - 1) \end{array} \tag{2.19}$$

are displayed in figure 2.11. The subscript gives the node index and the superscript indicates whether the basis function is associated with a nodal value (0) or a nodal derivative (1).

Note that the basis functions have the properties

$$\phi_i^j(0) = \begin{cases} 1 & \text{if } i = 1,\ j = 0 \\ 0 & \text{otherwise} \end{cases}, \qquad \phi_i^j(1) = \begin{cases} 1 & \text{if } i = 2,\ j = 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\left.\frac{d\phi_i^j}{d\xi}\right|_{\xi=0} = \begin{cases} 1 & \text{if } i = 1,\ j = 1 \\ 0 & \text{otherwise} \end{cases}, \qquad \left.\frac{d\phi_i^j}{d\xi}\right|_{\xi=1} = \begin{cases} 1 & \text{if } i = 2,\ j = 1 \\ 0 & \text{otherwise} \end{cases}$$

and therefore both nodal values and nodal derivatives are interpolated, i.e.,

$$\tilde{u}^{(e)}(0) = u_1^{(e)}, \qquad \tilde{u}^{(e)}(1) = u_2^{(e)}$$
$$\left.\frac{d\tilde{u}}{d\xi}\right|_{\xi=0} = \left(\frac{du}{d\xi}\right)_1^{(e)}, \qquad \left.\frac{d\tilde{u}}{d\xi}\right|_{\xi=1} = \left(\frac{du}{d\xi}\right)_2^{(e)}$$

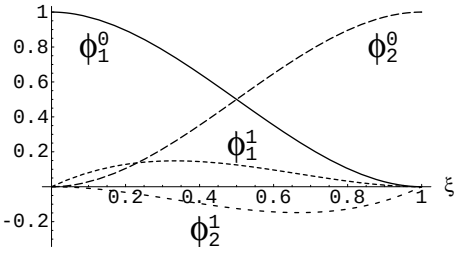


Figure 2.11. *Cubic Hermite basis functions.*

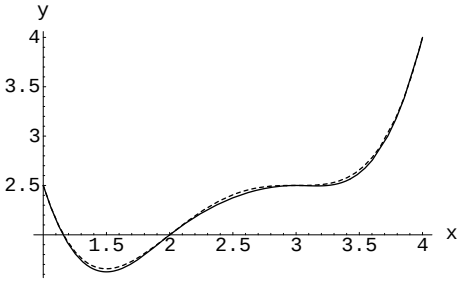


Figure 2.12. *Piecewise cubic Hermite interpolation (solid) of a quartic polynomial (dashed).*

The smoother approximation of the cubic Hermite interpolation is demonstrated by taking again the function $u(x) = 16 - 27x + 18x^2 - 5x^3 + 0.5x^4$ as an example. As previously the coordinates are $x_i = i$, $i = 1, \ldots, 4$ and the nodal values are $u_i = u(i)$. In addition the interpolation requires the nodal derivatives $\left(\frac{du}{d\xi}\right)_i = \left.\frac{du}{d\xi}\right|_{x=i}$. Figure 2.12 shows the original function as a dashed curve and the interpolated function as a solid curve.

The definition of the cubic Hermite interpolation presented above requires that the nodal derivatives are given with respect to the ξ-coordinates. This is inconvenient in some applications since a shared node might need different nodal derivatives for each element in order to achieve C^1 continuity across neighboring elements.

As an example consider the function $u(x) = x^2$ over the element domains $\Omega^{(e_1)} = [0, 1]$ and $\Omega^{(e_2)} = [1, 3]$. The global node n_2 with the coordinate $x_2 = 1$ corresponds to the local nodes $n_2^{(e_1)}$ and $n_1^{(e_2)}$. Introducing an isoparametric mapping gives $u^{(e_1)}(\xi) = \xi^2$ and $u^{(e_2)}(\xi) = (2\xi + 1)^2$ where $0 \leq \xi \leq 1$. The derivatives are $\frac{du^{(e_1)}}{d\xi} = 2\xi$ and $\frac{du^{(e_2)}}{d\xi} = 4(2\xi + 1)$ so that at the common node

$$\left(\frac{du}{d\xi}\right)_2^{(e_1)} = 2 \neq 4 = \left(\frac{du}{d\xi}\right)_1^{(e_2)}$$

A more useful formulation in practice is to define global nodal derivatives $\left(\frac{du}{ds}\right)_I$, where s is an arc length parameter from the isoparametric mapping and I is the global node index. The corresponding element nodal derivatives are then computed as

$$\left(\frac{du}{d\xi}\right)_i^{(e)} = \left(\frac{du}{ds}\right)_I \left(\frac{ds}{d\xi}\right)_i^{(e)}$$

where $\left(\frac{ds}{d\xi}\right)_i^{(e)}$ is an *element scale factor* which scales the arc length derivative of the global node I to the ξ-coordinate derivative of the element node i.

The idea is demonstrated using the previous example. Using the arc length parameterizations $s = \xi$ and $s = 2\xi + 1$ for the elements e_1 and e_2, respectively, yields the global nodal derivative $\left(\frac{du}{ds}\right)_1 = 2$ for the common node. The local nodal derivatives are computed as

$$\begin{aligned}\left(\frac{du}{d\xi}\right)_2^{(e_1)} &= \left(\frac{du}{ds}\right)_2 \left(\frac{ds}{d\xi}\right)_2^{(e_1)} = 2 * 1 = 2 \\ \left(\frac{du}{d\xi}\right)_1^{(e_2)} &= \left(\frac{du}{ds}\right)_2 \left(\frac{ds}{d\xi}\right)_1^{(e_2)} = 2 * 2 = 4\end{aligned}$$

The finite element modeller incorporated into our visualization toolkit implements both finite elements with local and elements with global nodal derivatives.

Multidimensional Basis Functions

Multidimensional finite elements are constructed as products of the underlying 1D elements. The domain of the parent element is formed from the product of the 1D parent element domains and the basis functions are defined as the tensor product of the corresponding 1D basis functions.

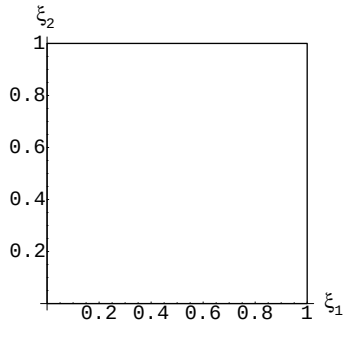


Figure 2.13. *The parent element for a 2D isoparametric quadrilateral.*

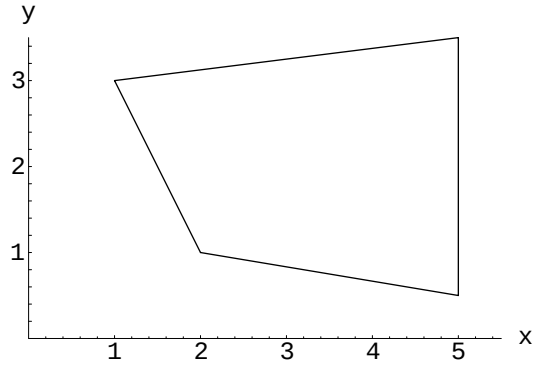


Figure 2.14. *Bilinear element in world coordinates.*

As an example consider a bilinear Lagrange element. The parent element is the unit square shown in figure 2.13. The $n_{LL} = n_L n_L = 4$ basis functions of the bilinear Lagrange element are the tensor products of the $n_L = 2$ basis functions of the linear

Lagrange element given in equation 2.17, i.e.,

$$\begin{aligned}
\phi_1(\xi_1,\xi_2) &= \phi_1(\xi_1)\phi_1(\xi_2) = (1-\xi_1)(1-\xi_2)\\
\phi_2(\xi_1,\xi_2) &= \phi_2(\xi_1)\phi_1(\xi_2) = \xi_1(1-\xi_2)\\
\phi_3(\xi_1,\xi_2) &= \phi_1(\xi_1)\phi_2(\xi_2) = (1-\xi_1)\xi_2\\
\phi_4(\xi_1,\xi_2) &= \phi_2(\xi_1)\phi_2(\xi_2) = \xi_1\xi_2
\end{aligned} \tag{2.20}$$

and are shown in figure 2.15.

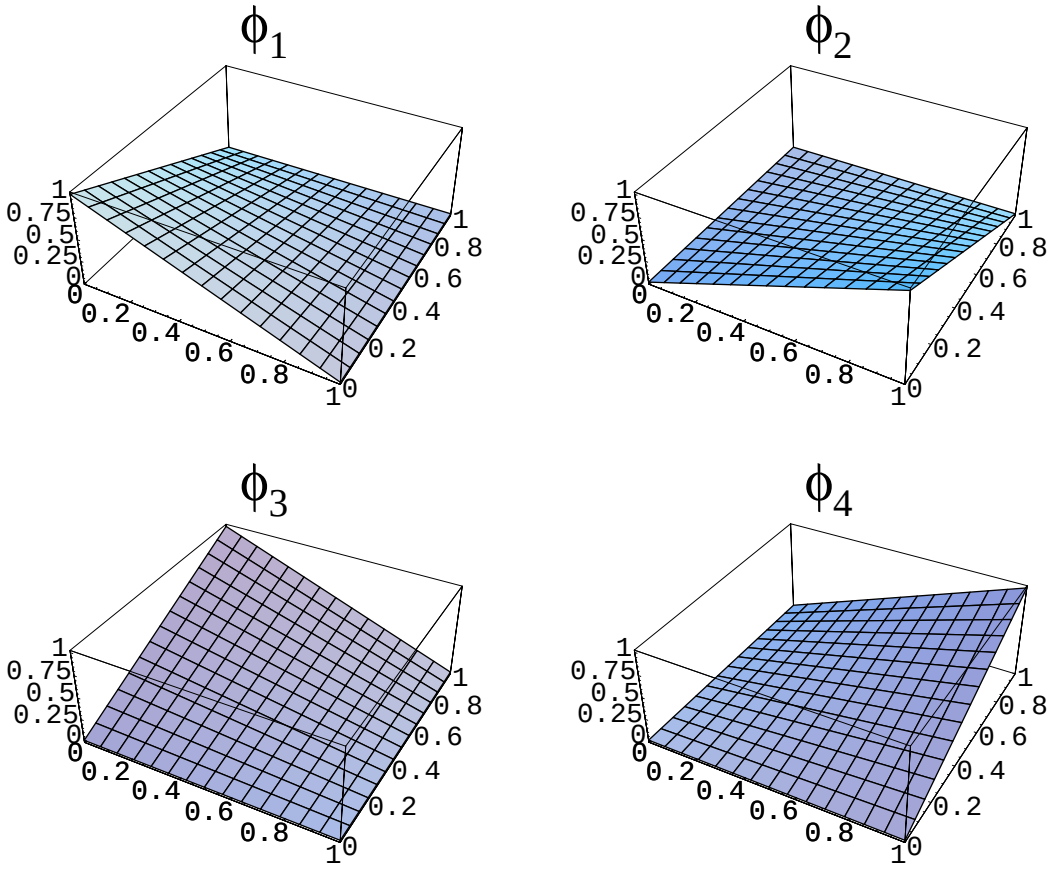


Figure 2.15. *Bilinear Lagrange basis functions.*

As in equation 2.16 the nodal values are interpolated as

$$\tilde{u}^{(e)}(\boldsymbol{\xi}) = \sum_{i=1}^{n} u_i^{(e)} \phi_i(\boldsymbol{\xi}) \tag{2.21}$$

where $n = 4$ and $\boldsymbol{\xi} = (\xi_1, \xi_2)^T$ are the material coordinates. The world coordinates are interpolated analogously, i.e.,

$$\tilde{\mathbf{x}}^{(e)}(\boldsymbol{\xi}) = \sum_{i=1}^{n} \mathbf{x}_i^{(e)} \phi_i(\boldsymbol{\xi}) \tag{2.22}$$

where $\mathbf{x} = (x_1, x_2)^T$ and $\mathbf{x}_i^{(e)}$ are the 2D element nodal coordinates of the element e. Figure 2.14 shows an example for a bilinear element in world coordinates.

Using the same procedure as for the bilinear Lagrange element it is also possible to define bicubic Hermite elements. The parent element is again the unit square shown in figure 2.13 and the $n_{HH} = n_H n_H = 16$ basis functions of the element are the tensor products of the $n_H = 4$ basis functions of the cubic Hermite element, i.e.,

$$
\begin{aligned}
\phi_1^{0,0}(\xi_1,\xi_2) &= \phi_1^0(\xi_1)\phi_1^0(\xi_2) = (1-3\xi_1^2+2\xi_1^3)\,(1-3\xi_2^2+2\xi_2^3)\\
\phi_2^{0,0}(\xi_1,\xi_2) &= \phi_2^0(\xi_1)\phi_1^0(\xi_2) = \xi_1^2\,(3-2\xi_1)\,(1-3\xi_2^2+2\xi_2^3)\\
\phi_3^{0,0}(\xi_1,\xi_2) &= \phi_1^0(\xi_1)\phi_2^0(\xi_2) = (1-3\xi_1^2+2\xi_1^3)\,\xi_2^2\,(3-2\xi_2)\\
\phi_4^{0,0}(\xi_1,\xi_2) &= \phi_2^0(\xi_1)\phi_2^0(\xi_2) = \xi_1^2\,(3-2\xi_1)\,\xi_2^2\,(3-2\xi_2)\\
\phi_1^{0,1}(\xi_1,\xi_2) &= \phi_1^0(\xi_1)\phi_1^1(\xi_2) = (1-3\xi_1^2+2\xi_1^3)\,\xi_2\,(\xi_2-1)^2\\
\phi_2^{0,1}(\xi_1,\xi_2) &= \phi_2^0(\xi_1)\phi_1^1(\xi_2) = \xi_1^2\,(3-2\xi_1)\,\xi_2\,(\xi_2-1)^2\\
\phi_3^{0,1}(\xi_1,\xi_2) &= \phi_1^0(\xi_1)\phi_2^1(\xi_2) = (1-3\xi_1^2+2\xi_1^3)\,\xi_2^2\,(\xi_2-1)\\
\phi_4^{0,1}(\xi_1,\xi_2) &= \phi_2^0(\xi_1)\phi_2^1(\xi_2) = \xi_1^2\,(3-2\xi_1)\,\xi_2^2\,(\xi_2-1)\\
\phi_1^{1,0}(\xi_1,\xi_2) &= \phi_1^1(\xi_1)\phi_1^0(\xi_2) = \xi_1\,(\xi_1-1)^2\,(1-3\xi_2^2+2\xi_2^3)\\
\phi_2^{1,0}(\xi_1,\xi_2) &= \phi_2^1(\xi_1)\phi_1^0(\xi_2) = \xi_1^2\,(\xi_1-1)\,(1-3\xi_2^2+2\xi_2^3)\\
\phi_3^{1,0}(\xi_1,\xi_2) &= \phi_1^1(\xi_1)\phi_2^0(\xi_2) = \xi_1\,(\xi_1-1)^2\,\xi_2^2\,(3-2\xi_2)\\
\phi_4^{1,0}(\xi_1,\xi_2) &= \phi_2^1(\xi_1)\phi_2^0(\xi_2) = \xi_1^2\,(\xi_1-1)\,\xi_2^2\,(3-2\xi_2)\\
\phi_1^{1,1}(\xi_1,\xi_2) &= \phi_1^1(\xi_1)\phi_1^1(\xi_2) = \xi_1\,(\xi_1-1)^2\,\xi_2\,(\xi_2-1)^2\\
\phi_2^{1,1}(\xi_1,\xi_2) &= \phi_2^1(\xi_1)\phi_1^1(\xi_2) = \xi_1^2\,(\xi_1-1)\,\xi_2\,(\xi_2-1)^2\\
\phi_3^{1,1}(\xi_1,\xi_2) &= \phi_1^1(\xi_1)\phi_2^1(\xi_2) = \xi_1\,(\xi_1-1)^2\,\xi_2^2\,(\xi_2-1)\\
\phi_4^{1,1}(\xi_1,\xi_2) &= \phi_2^1(\xi_1)\phi_2^1(\xi_2) = \xi_1^2\,(\xi_1-1)\,\xi_2^2\,(\xi_2-1)
\end{aligned}
$$

The nodal values are interpolated as

$$
\begin{aligned}
\tilde{u}(\xi_1,\xi_2) = \sum_{i=1}^{n} \Bigg(& u_i^{(e)}\phi_i^{0,0}(\xi_1,\xi_2) + \left(\frac{\partial u}{\partial \xi_1}\right)_i^{(e)} \phi_i^{1,0}(\xi_1,\xi_2)\\
& + \left(\frac{\partial u}{\partial \xi_2}\right)_i^{(e)} \phi_i^{0,1}(\xi_1,\xi_2) + \left(\frac{\partial^2 u}{\partial \xi_1 \partial \xi_2}\right)_i^{(e)} \phi_i^{1,1}(\xi_1,\xi_2)\Bigg)
\end{aligned}
$$

where $n = 4$. Note that in addition to the partial derivatives in the ξ_1 and ξ_2-directions at the element nodes the bicubic interpolation requires the mixed partial derivative $\left(\frac{\partial^2 u}{\partial \xi_1 \partial \xi_2}\right)_i^{(e)}$ at each node. An example of a bicubic element in world coordinates is shown in figure 2.16

As in the 1D case it is again more practical to define global nodal derivatives $\left(\frac{\partial u}{\partial s_1}\right)_I$, $\left(\frac{\partial u}{\partial s_2}\right)_I$, and $\left(\frac{\partial^2 u}{\partial s_1 \partial s_2}\right)_I$, where s_1 and s_2 are the arc lengths in the ξ_1 and ξ_2 directions and I is the global node index. The corresponding element nodal derivatives are then computed as

$$
\begin{aligned}
\left(\frac{\partial u}{\partial \xi_1}\right)_i^{(e)} &= \left(\frac{\partial u}{\partial s_1}\right)_I \left(\frac{ds_1}{d\xi_1}\right)_i^{(e)}\\
\left(\frac{\partial u}{\partial \xi_2}\right)_i^{(e)} &= \left(\frac{\partial u}{\partial s_2}\right)_I \left(\frac{ds_2}{d\xi_2}\right)_i^{(e)}
\end{aligned}
$$

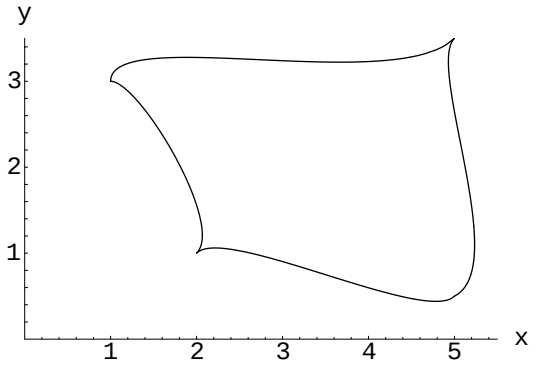


Figure 2.16. *Bicubic Hermite element in world coordinates.*

$$\left(\frac{\partial^2 u}{\partial \xi_1 \partial \xi_2}\right)_i^{(e)} = \left(\frac{\partial^2 u}{\partial s_1 \partial s_2}\right)_I \left(\frac{ds_1}{d\xi_1}\right)_i^{(e)} \left(\frac{ds_2}{d\xi_2}\right)_i^{(e)}$$

where $\left(\frac{ds_1}{d\xi_1}\right)_i^{(e)}$ and $\left(\frac{ds_2}{d\xi_2}\right)_i^{(e)}$ are again the element scale factors.

The definition of higher order elements and mixed elements, e.g., linear Lagrange cubic Hermite elements, proceeds analogously to the presented examples.

2.4.3 Isoparametric Mapping

The previous subsection introduced the mapping of a unit square in ξ-coordinates to a finite element in world coordinates. The resulting *isoparametric map* was motivated by the goal to define the element basis function only once for a parent element. A further advantage of ξ-coordinates is that the coordinate system deforms with the material and as such allows representation of data values with respect to the deformed material. In many instances it is convenient to switch between world coordinates (x-coordinates) and material coordinates (ξ-coordinates). This subsection gives an overview of the mathematical principles involved. For simplicity we omit the element indices '(e)' and the tilde-symbols indicating approximation functions.

Coordinate Transformation of a Point

Assuming a point in material coordinates $\boldsymbol{\xi}$ the corresponding world coordinates $\mathbf{x}$ are obtained by the isoparametric map

$$\mathbf{x}(\boldsymbol{\xi}) = \sum_{i=1}^{n} \bar{\mathbf{x}}_i \phi_i(\boldsymbol{\xi}) \tag{2.23}$$

where $\bar{\mathbf{x}}_i$ are the nodal coordinates. An example is the bilinear interpolation given by equation 2.22. Vice versa material coordinates can be computed from the world coordinates by using numerical approximation techniques such as a multidimensional Newton mesh [PVTF92].

Partial Derivatives with Respect to World and Material Coordinates

In most FE applications data quantities are defined by nodal values and are interpolated over the finite elements. For example, a scalar quantity u defined by the nodal values u_i is interpolated as

$$u(\boldsymbol{\xi}) = \sum_{i=1}^{n} u_i \phi_i(\boldsymbol{\xi})$$

The partial derivatives of u can be computed directly from the interpolation function as

$$\frac{\partial u}{\partial \xi_j} = \sum_{i=1}^{n} u_i \frac{\partial \phi_i}{\partial \xi_j} \tag{2.24}$$

The derivatives $\frac{\partial \phi_i}{\partial \xi_j}$ of the basis functions are generally given as predefined analytic functions. The derivatives with respect to the world coordinate system are defined as

$$\frac{\partial u}{\partial x_j} = \sum_{i=1}^{n} \frac{\partial u}{\partial \xi_i} \frac{\partial \xi_i}{\partial x_j}$$

where $\frac{\partial u}{\partial \xi_i}$ is defined as in equation 2.24 and the partial derivatives $\frac{\partial \xi_j}{\partial x_i}$ are the components of the inverse of the Jacobian of the isoparametric mapping in equation 2.23

$$\mathbf{J} = \left(\frac{\partial x_i}{\partial \xi_j}\right) = \begin{pmatrix} \frac{\partial x_1}{\partial \xi_1} & \frac{\partial x_1}{\partial \xi_2} & \cdots & \frac{\partial x_1}{\partial \xi_n} \\ \frac{\partial x_2}{\partial \xi_1} & \frac{\partial x_2}{\partial \xi_2} & \cdots & \frac{\partial x_2}{\partial \xi_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial x_n}{\partial \xi_1} & \frac{\partial x_n}{\partial \xi_2} & \cdots & \frac{\partial x_n}{\partial \xi_n} \end{pmatrix}, \quad \left(\frac{\partial \xi_j}{\partial x_i}\right) = \mathbf{J}^{-1} \tag{2.25}$$

The introduced concepts can be generalised for vector and tensor quantities. Note, however, that a component based interpolation of vectors and tensors is not necessarily the most suitable one and alternative techniques have been suggested [AB99].

Coordinate Transformation of Vector and Tensor Quantities

Depending on the intended application it can be convenient to specify a quantity in either material or world coordinates. Material coordinates are often more computational efficient and provide a relationship between data values and underlying geometry. World coordinates are usually more convenient from a visualization standpoint. Whereas scalar quantities are coordinate system independent the representation of vector and tensor quantities depends on the coordinate system used.

Section 2.1 introduced the transformation of vector and tensor quantities between two coordinate systems. Using equation 2.8 the representations of a vector in material coordinates $\tilde{\mathbf{v}}$ and in world coordinates $\mathbf{v}$ are converted into each other by

$$\mathbf{v} = \mathbf{J}\tilde{\mathbf{v}} \ , \qquad \tilde{\mathbf{v}} = \mathbf{J}^{-1}\mathbf{v} \tag{2.26}$$

Similarly using equation 2.9 the corresponding formulas for converting tensors are

$$\mathbf{T} = \mathbf{J}\tilde{\mathbf{T}}\mathbf{J}^{T} \ , \qquad \tilde{\mathbf{T}} = \mathbf{J}^{-1}\mathbf{T}(\mathbf{J}^{-1})^{T} \tag{2.27}$$

Note that if the determinant of the Jacobian is zero, i.e., if the finite element is degenerate (e.g., two vertices have the same coordinates), the reverse transformations do not exist.

2.4.4 Gaussian Quadrature

Solving a FE problem requires the solution of complex integral equations involving the finite element basis functions (see appendix D). Similar integrals are encountered when computing model properties such as volume and surface areas (see subsection 5.8.2). Our visualization toolkit was specifically designed for FE models of biomedical structures and the computation and visualization of such derived information is a useful feature. An efficient evaluation of FE integrals is therefore desirable.

A popular method to evaluate integrals arising in 2D and 3D finite element problems is the Gaussian quadrature (Gauss-Legendre quadrature). The idea behind it is to approximate the integral of a function f over a range $[0, 1]$ by weighted samples of $f(\xi)$ taken at points $\xi_1, \ldots, \xi_n$. If f is a polynomial of degree $2n-1$ then it has $2n$ coefficients. It is possible to choose n *weights* w_i and n *gauss points* ξ_i such that the approximation is exact [Bur87], i.e.,

$$\int_0^1 f(\xi)\ d\xi = \sum_{i=1}^{n} w_i f(\xi_i)$$

As an example take $n = 2$: We are looking for w_i, ξ_i, $i = 1, 2$, such that

$$\int_0^1 f(\xi)\ d\xi = w_1 f(\xi_1) + w_2 f(\xi_2) \tag{2.28}$$

where $f(\xi) = a_0 + a_1\xi^1 + a_2\xi^2 + a_3\xi^3$. Expanding the left hand side of equation 2.28

$$\int_0^1 f(\xi)\ d\xi = \sum_{i=0}^{3} a_i \int_0^1 \xi^i\ d\xi$$

and applying equation 2.28 to each term yields a set of four equations

$$\int_0^1 \xi^i\ d\xi = \frac{1}{i+1} = w_1\xi_1^i + w_2\xi_2^i, \quad \text{i=0,...,3}$$

which is satisfied by

$$w_{1,2} = \frac{1}{2}, \quad \xi_{1,2} = \frac{1}{2} \mp \frac{1}{2\sqrt{3}}$$

Similar calculations can be done to obtain the gauss points and weights for higher order polynomials [Bur87, Bat82, AS65]. The corresponding values for a 5^{th} degree polynomial are [HP02]

$$w_{1,3} = \frac{5}{18}, \qquad \xi_{1,3} = \frac{1}{2} \mp \frac{1}{2}\sqrt{\frac{3}{5}}$$
$$w_2 = \frac{4}{9}, \qquad \xi_2 = \frac{1}{2}$$

The Gaussian integration is extended to multi-dimensional integration by defining gauss points whose components contain all combinations of the corresponding 1D gauss points with the corresponding weights multiplied (product rule). For example, choosing two gauss points in each coordinate direction results in the following 2D gauss points and weights

$$w_{11} = w_{12} = w_{21} = w_{22} = \frac{1}{4}$$

$$\boldsymbol{\xi}_{11} = \begin{pmatrix} \frac{1}{2} - \frac{1}{2\sqrt{3}} \\ \frac{1}{2} - \frac{1}{2\sqrt{3}} \end{pmatrix}, \quad \boldsymbol{\xi}_{12} = \begin{pmatrix} \frac{1}{2} - \frac{1}{2\sqrt{3}} \\ \frac{1}{2} + \frac{1}{2\sqrt{3}} \end{pmatrix}$$

$$\boldsymbol{\xi}_{21} = \begin{pmatrix} \frac{1}{2} + \frac{1}{2\sqrt{3}} \\ \frac{1}{2} - \frac{1}{2\sqrt{3}} \end{pmatrix}, \quad \boldsymbol{\xi}_{22} = \begin{pmatrix} \frac{1}{2} + \frac{1}{2\sqrt{3}} \\ \frac{1}{2} + \frac{1}{2\sqrt{3}} \end{pmatrix}$$

The number of gauss points in each coordinate direction depends on the complexity of the element integrals to be evaluated. A complication in multi-dimensional problems is that the element integrals are often not polynomial due to the partial derivatives of the ξ parameter. Also the quadrature error has to be balanced against the discretization error. In general two gauss points are used in coordinate directions with quadratic basis functions and three gauss points in coordinate directions with cubic basis function [HP02]. Guidelines for choosing a suitable type of Gaussian integration are given by Burnett [Bur87] and Bahte [Bat82].

2.4.5 Solving a FE Problem

So far this chapter has described the finite element approximation of the model geometry and associated data fields. The actual solution process for a FE problem is not directly relevant in the context of this thesis since it usually does not influence the visualization process.

We have however implemented a complete FE modeller and employed it for the creation of various physical models used to test and demonstrate our visualization

environment. The following paragraphs give a short overview of the FE solution process.

A FE problem is solved by performing the following basic steps: First the domain is subdivided into smaller regions called *finite elements*. The shapes of the elements are limited to a certain type depending on the particular type of FEM chosen. Each type of element has a parent element and an associated set of basis functions (see subsection 2.4.2). Note that several different element types can be used simultaneously as long as they are compatible with each other.

The governing equation of the FE problem is then transformed into algebraic equations such that the equations are algebraically identical for each element of the same type. The terms of the element equations are numerically evaluated and assembled into an *element stiffness matrix* and an *element load vector* representing a linear system of equations (the so-called *element equations* or *element system*). The resulting matrices and vectors are assembled into one *global stiffness matrix* and *global load vector* which form a *global system*. The global system, which is a linear system of equations, is modified according to the imposed boundary conditions. The resulting equations describe the modeled system over the discretized given domain and are called the *global system after imposing boundary conditions*. The global system can be solved directly, e.g., by a Conjugate Gradient method [PVTF92], to yield the required solution.

For the interested reader the solution process is demonstrated by two examples in the appendix D. The first example describes a 2D heat conduction problem and was chosen for its simplicity. The second example describes a linear elastic solid under an applied load and is formulated in vector notation so that it can be immediately extended to higher dimensions. The 3D version of this example was employed to model a plate with a hole under an uniaxial load, which is used in chapter 5 for demonstrating various features of our visualization toolkit.

The enclosed CD contains an object-oriented FE modeller programmed by us using Microsoft Visual C++ 6.0 [Gre97]. The application implements generalised 2D and 3D versions of the FE problems discussed in appendix D. The graphical user interface was designed using OpenGL [WND97] and FLTK, a LGPL'd C++ graphical user interface toolkit for X (UNIX), OpenGL, and WIN32 [Spi].