An Numerical Elastic Model for Deforming Bat Pinnae

Sreenath Balakrishnan

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
In
Mechanical Engineering

Müller, Rolf , Chair
Priya, Shashank ,
Socha, John

December 10, 2010
Blacksburg, Virginia

Keywords: Deformable models, bat pinnae, ear movements, spatial hearing , biosonar

An Numerical Elastic Model for Deforming Bat Pinnae

Sreenath Balakrishnan

(ABSTRACT)

In bats, the directivity patterns for reception are shaped by the surface geometry of the pinnae. Since many bat species are capable of large ear deformations, these beampatterns can be time-variant. To investigate this time-variance using numerical methods, a digital model that is capable of representing the pinna geometry during the entire deformation cycle has been developed.

Due to large deformations and occlusions, some of the surfaces relevant to sound diffraction may not be visible and the geometry of the entire pinna has to be computed from limited data. This has been achieved by combining a complete digital model of the pinna in one position with time-variant sparse sets of three dimensional landmark data. The landmark positions were estimated using stereo vision methods. A finite element model based on elasticity was constructed from CT scans of the pinna post mortem. This elastic model was deformed to provide a good fit to the positions of the landmarks and retain values of smoothness and surface energy comparable to life. This model was able to handle ratios of data to degrees of freedom around 1:5000 and still effect life-like deformations with an acceptable goodness of fit.

Dedication

This thesis is dedicated to the memory of my grandmothers T.V. Ikkalikutty Warassiar and T.V. Kochammini Warassiar who brought me up with great love and affection

Acknowledgments

Firstly I would like to thank my advisor, Dr.Rolf Müller, for teaching me the power of sustained hard work and the value of time apart from all the other wonderful technical things that he taught me. I am extremely thankful for the time he spent discussing the research with me and for his excellent guidance throughout my masters. I would like to credit him for all the originality in this research.

I am extremely grateful to my chief collaborator, Li Gao, for being a great colleague and a wonderful friend. I would also like to thank all the students at the Bio-inspired Physics Lab at Shandong University who obtained the bats and performed the experiments without which this work would not have been possible.

I also want to thank my other committee members, Dr.Shashank Priya and Dr.Jake Socha for their invaluable support and guidance.

I am greatly indebted to my friends at IALR, especially my roommate Sujay Kawale, who made my stay in Danville very interesting and rewarding.

Finally, I thank my parents and my brother for their love, and Supriya for her constant moral support.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

Nature is an abundant source of inspiration and knowledge for science. Many scientific discoveries have drawn inspiration from this marvelous source of knowledge. There has been much interest in understanding scientific principles from animals and using the learned principles as inspirations for improving technological capabilities - sometimes called bio-inspired technology. Due to the innate complexity of biological systems, individual studies of such kind are usually confined to a particular system in a specific animal family, genus or species but the principles learnt could have far wider implications. The research presented here is a study of the biosonar system in bats - an active behavioral pattern in a particular species of bat (*Rhinolophus ferrumequinum*) and its connection to echolocation. But the principles learnt could change the popular perception about general hearing systems or any wave emission receiver system like radar, sonar, etc.

Bats are some of the most abundantly found mammalian species in the world. Bats are known to survive in almost all parts of the world except for the polar ice caps and some remote islands [1]. There are around 1100 species of bats which make up for around 20% of all mammalian species [2]. Bats are very diverse, with a variety of feeding habits between different species like fruits, nectar, pollen, insects, vertebrates, arthropods, birds, blood, other bats, etc.[1]. In comparison to other mammals, bats have been considered as highly successful because of these qualities - abundance, diversity and global distribution. Due to all these, bats can be considered as an excellent model of bio-inspiration.

Bats have evolved highly effective sensing methodologies, especially in their use of the biosonar, which has helped them to thrive in various parts of the world [3, 4]. Bats are known to use biosonar for the acquisition of food and navigation tasks such as obstacle avoidance and - possibly - for the recognition of landmarks. Interspecific adaptations of the biosonar can be seen in bat species with different habitats, food habits etc. These adaptations can be seen as evolutionary solutions to specific sensory requirements arising from the ecology of the bat. Understanding the principles behind these adaptations can hence provide insights into improving the capabilities of technological systems.

One of the important components in biological hearing systems is the outer ear. The outer ear (pinna) shape helps to encode information on the sound sources at each ear [5, 6, 7, 8, 9, 10]. This is accomplished by direction and frequency dependent diffraction of the sound at the pinna. Such a diffraction process depends completely on the shape of the pinna and therefore the pinna shape serves an important function in any biological hearing system. Changing the shape of the pinna could hence be a mode of tuning the hearing system to specific sensory requirements arising out of the ecology, prey, habitat etc. of the animal. This could be the reason for the markedly different pinna shapes found between various mammalian species including bats [11].

Any wave emission or receiver system (like ear) is characterized by its beampattern. Beampattern of a receiver describes the distribution of receiver sensitivity over direction at a given frequency. The biosonar beampatterns of the pinnae (receiver) of various bat species have been estimated using various experimental and numerical methods[12]. Previous studies have shown various qualitative differences in the beampatterns of the pinna shapes from different bat species. This suggests the presence of an evolutionary adaptation mechanism in the biosonar system of bats through changes in the shape of the pinna.

Bats are frequently grouped into fm (frequency-modulated) and cf-fm (constant-frequency – frequency-modulated) bats based on their emitted pulse waveform[13]. In fm bats, the instantaneous frequency of the biosonar pulse changes continuously over the entire pulse duration, whereas in cf-fm bats, it remains constant for a significant portion of the pulse duration. Cf-fm bats and horseshoe bats in particular have long been known for their conspicuous pinna movements [14, 15]. Early anatomical work [14] has demonstrated the presence of specialized muscular actuation mechanisms in the pinna of these animals. Experimental results have demonstrated that immobilizing the pinnae of these bats degrades target localization in the vertical direction [16]. The target direction cues generated by rigid rotation of a receiver and its static beampattern in target localization have also been analyzed already [17]. The pinna movements observed on the greater horseshoe bat (*Rhinolophus ferrumequinum*) seems to have a significant amount of deformation in addition to rigid rotation. The wavelengths in the strongest harmonic of the biosonar pulse used by this bat is approximately from 4.3 - 5.7 mm. Hence a deformation of the same order ($\sim>2$ mm) can cause considerable changes in the phase relationship between wave field contributions from different parts of the pinna and alter the beampattern substantially. If such deformation can produce consistent qualitative changes in the beampattern, it would mean that a mechanism to adapt the pinna beampattern exists at the individual behavioral level in addition to the known evolutionary interspecies level of adaptation.

In order to investigate the effect of the ear movements on the beampattern of the pinna, time-varying complete 3D digital models of the pinnae of live and behaving bats have to be built. Time varying physical configuration of dynamic systems can be captured well using high speed computer vision. But during the pinna movements, large parts of the pinna get occluded because of which the three dimensional shape of the pinna reconstructed by using computer vision could be incomplete. In contrast, computer tomography can construct a complete high resolution three dimensional model of the pinna. But it takes quite some time to scan the total length of the pinna. Since the movements occur very fast (around 200ms) it would not be possible to obtain the time-varying three dimensional models of the pinna by computer tomography alone. A combination of both these techniques, employing the ability of computer vision to capture the dynamics and of computer tomography to obtain complete high resolution three dimensional models seems to be a better approach. Computer Tomography can be used to obtain the complete geometry of the pinna in one position. Computer vision can be used to obtain sparse information about the pinna during the in vivo ear movements. The complete in vivo pinna geometry during the ear movements can be estimated by combining the complete pinna geometry obtained from computer tomography with the sparse information obtained from computer vision.

The work on the experiments with the bats, stereo triangulation, CT model reconstruction and acoustic analysis was chiefly done by other collaborators. The methods, results and conclusions are presented here for completeness and providing the context of the research.

1.2 Objective

To build time-varying complete three-dimensional models of behaving bat ears by combining the complete geometric shape in one physical configuration obtained using computer tomography with the sparse dynamic information obtained from stereo vision.

1.3 Organization

The thesis focusses on the main personal contribution - building time-varying three dimensional models of bat ears from computer tomographic and stereo data. Obtaining the required data - bat experiments, stereo triangulation and CT model reconstruction and further acoustical analysis of the bat ear models were chiefly worked upon by other collaborators. But the methods, results and conclusions on all these are presented here for completeness and for providing proper context of the research and its outcomes.

The thesis is organized as follows:

Chapter 2 reviews previous work done on making time-varying three-dimensional models

Chapter 3 describes the concepts and methods used for making time-varying three-dimensional models of actively behaving bat ears.

Chapter 4 presents the results obtained by applying the methods described in the previous chapter.

Chapter 5 discusses the methods and results and makes conclusions on both.

Chapter 6 describes the methods, results and conclusion from the acoustical analysis.

Appendix A explains the implementation of the methods by describing the main computer programs developed.

Chapter 2

State of the art

Simulating the time-varying physical configuration of objects undergoing deformation using information from various sources like computer vision, computer tomography, MR scans, etc. is usually referred to in literature as deformable models. Deformable model formulations have been used for a variety of applications [18] like pattern recognition [19, 20], computer animation [21], tracking [22], image segmentation [23, 24, 25, 26, 27], surgery simulation [28] etc.

In the case of modelling ear deformations, due to the folded shape of the ear, many parts (back, inside etc) of the ear will be occluded from the camera view. For these portions which are completely hidden from the camera view, there would not be any dynamic information obtained from stereo vision. Hence the time-varying dynamic position of these portions need to be extrapolated from the available dynamic information of the parts visible in the camera view using *a priori* knowledge on the material behavior and the geometric information from computer tomography. Such a formulation would be aided by modelling the physics of the deformation. Hence only physics based deformable model formulations were considered.

2.1 Mass-spring models

In mass-spring systems, the objects are modelled as a collection of point masses in a regular lattice structure connected to each other through springs [29] . The springs are usually assumed to be linear [30] (force is directly proportional to the change in the length of the spring). The point masses could be regularly spaced and can be connected together in the form of a hexahedron (s. Fig. 2.1). In addition to longitudinal springs, which resist expansion and contraction of the hexahedron, shear springs connecting opposite vertices can be used to resist shear deformation [31] (s. Fig. 2.1).

In the case of static problems, nodal displacements which balance the forcing function (nodal

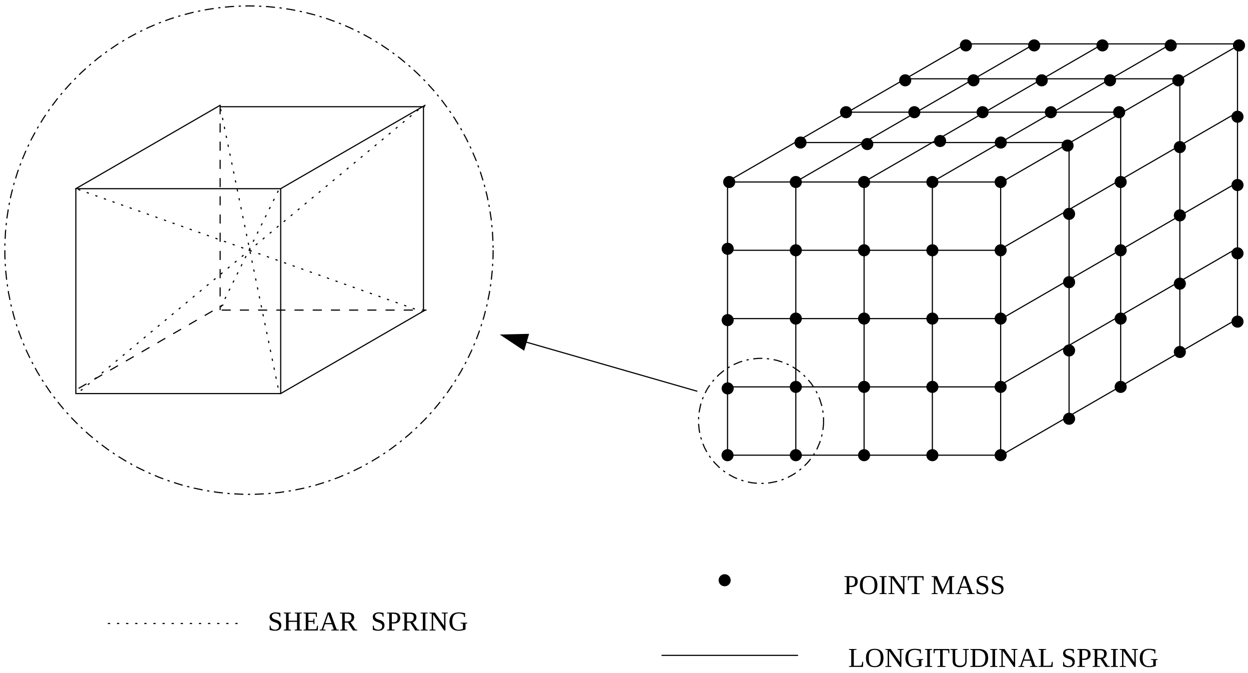

Figure 2.1: Example for a mass-spring system

forces) with the internal restoring forces in the springs is estimated according to the relation

$$[\mathbf{K}][\mathbf{U}] = [\mathbf{F}] \tag{2.1}$$

where $[\mathbf{K}]$ is the global stiffness matrix assembled from the elemental stiffnesses due to the springs, $[\mathbf{U}]$ is the vector of displacements and $[\mathbf{F}]$ the force vector at the point masses. The force vector is derived from the data of the particular deformation that needs to be done. Such formulations were first introduced for facial modelling [32, 33]

For dynamic systems [34, 35, 36], the motion of each point mass i is governed by Newton's second law of motion $\mathbf{f}_i = m_i\ddot{\mathbf{x}}_i$. The forces are a net result of the restoring forces in the springs, damping forces if any, external applied forces etc. The equation of motion of each point mass can be assembled together to form the equation of motion of the total mass-spring system,

$$\mathbf{M}\ddot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{v}) \tag{2.2}$$

where $[\mathbf{M}]$ is a $3n \times 3n$ diagonal mass matrix (n is the number of point masses) and the force vector $\mathbf{f}$, in general, is a function of the position $\mathbf{x}$ and velocity $\mathbf{v}$ of the point masses. The velocity term is included in the force for special cases when there is a need to simulate loss of energy during deformation (viscoelastic or damping effects). In such cases, the springs are assumed to exert a viscous force, proportional to the difference in velocities of the two point masses to which it is attached, in addition to the longitudinal forces.

$$\mathbf{f}^v_{ij} = k_d(\mathbf{v}_j - \mathbf{v}_i) \tag{2.3}$$

where $\mathbf{f}_{ij}^{v}$ is the viscous/damping force on mass i due to the spring that connects it to mass j, k_d is the coefficient of damping and $\mathbf{v}_i$ and $\mathbf{v}_j$ are the velocities of the masses i and j respectively.

The mass-spring system can be generalized into a particle system, which retains the point masses and the connectivity between the point masses but replaces the springs with explicit constraint equations [37, 38, 39]. For a set of constraints, $\mathbf{C}(\mathbf{x})$=0, an associated deformation energy is defined as,

$$\mathbf{E} = k_x \left(\frac{1}{2} \mathbf{C}^T(\mathbf{x}) \mathbf{C}(\mathbf{x}) \right) \tag{2.4}$$

The forces at the nodes are then computed as the derivatives of the energies with respect to the particle positions

$$\mathbf{f}_i = -\frac{\partial \mathbf{E}}{\partial \mathbf{x}_i} \tag{2.5}$$

In the case of a distance constraint between two point masses, the system reduces to one where the masses are connected to each other by a linear spring.

Once the governing dynamics equations of the system is derived (mass-spring or the more general particle system), the physical configuration of the system at discrete points in time can be found out using a suitable time integration scheme.

Mass spring systems have been widely used for modelling skin, fat and muscle [34, 35, 36], locomotion of snakes, worms and fish [40, 41], cloth [38] etc. Mass spring systems (also particle systems) are very intuitive, easy to implement and computationally efficient. But they may not be accurate since the underlying material properties (like elasticity) are not exactly modelled. Most mass-spring systems are not convergent, ie, they do not converge to the true solution on refining the mesh, but are dependent on the resolution of the mesh [29]. Such systems may be suitable for animation films, computer games etc., where the primary concern is a visually convincing result. But for other applications like medical, garment simulation etc., where the chief objective is to produce a physically accurtate animation, such a formulation may not be the ideal choice.

2.2 Finite Element Method

Applying Newton's second law of motion to an elastic material gives the partial differential equation governing the dynamics as [42]

$$\rho \ddot{\mathbf{x}} = \nabla \cdot \sigma + \mathbf{f} \tag{2.6}$$

where ρ is the density of the material, ∇ is the divergence operator, σ is the 3×3 stress tensor and $\mathbf{f}$ is the applied forces. The divergence operator acts on the 3×3 stress tensor to give a column vector which represents the resultant forces due to stress in each of the coordinate directions [42].

$$\nabla \cdot \sigma = \begin{bmatrix} \frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{yx}}{\partial y} + \frac{\partial \sigma_{zx}}{\partial z} \\ \frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \sigma_{zy}}{\partial z} \\ \frac{\partial \sigma_{xz}}{\partial x} + \frac{\partial \sigma_{yz}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} \end{bmatrix} \tag{2.7}$$

In finite element method (FEM), the continuous domain is split up into a number of elements (mesh) and displacement functions satisfying the differential equation (Eq. (2.6)) in an integral sense inside the domain of each of the elements is obtained.

The displacement function in the domain of each element is usually approximated by interpolating it from the nodal displacement values [43] by

$$\tilde{\mathbf{u}}(\mathbf{m}, t) = \sum_i \mathbf{u}_i(t)\mathbf{b}_i(\mathbf{m}) \tag{2.8}$$

where $\mathbf{b}_i(\cdot)$ are the shape functions used for interpolating the displacement function in the elemental domain from the nodal displacement values. The shape functions $\mathbf{b}_i(\cdot)$ are chosen such that they have a value equal to unity at node i and zero at all the other nodes in the element. The problem is now reduced from finding the infinite dimensional continuous displacement field $\mathbf{u}(\mathbf{m}, t)$ to the finite dimensional displacement values at the nodes of the mesh $\mathbf{u}_i(t)$

In deformable models a simple form of FEM, sometimes called *explicit* FEM is used wherein equivalent nodal forces equivalent to the elastic forces due to deformation are computed locally from the displacement field [44, 45, 46]. From a displacement field $\mathbf{u}(\mathbf{m})$ expressed as an interpolation from the nodal displacements Eq (2.8) , the strain $\epsilon(\mathbf{m})$ and stress $\sigma(\mathbf{m})$ tensors are calculated. The deformation energy for the object is then given by

$$E = \int_V \epsilon(\mathbf{m}) \cdot \sigma(\mathbf{m}) d\mathbf{m} \tag{2.9}$$

The forces at the nodes are now computed as the derivatives of the deformation energy with respect to the nodal displacements. Assuming linear relationships for strain-displacement and stress-strain equations, the nodal forces can be obtained as a linear combination of the nodal displacements [29],

$$\mathbf{f} = \mathbf{K}\mathbf{u} \tag{2.10}$$

The dynamic equation giving the Newton's second law of motion for elastic materials (Eq. (2.6)) can then be rewritten in terms of nodal displacement and its derivatives as

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{D}\dot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{f}_{ext} \tag{2.11}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{D}$ the damping matrix (both 3n×3n and usually diagonal) and $\mathbf{f}_{ext}$ is vector of external forces (of dimension 3n). The dynamic finite dimensional equations can be integrated over time using a suitable numerical time integration scheme to get the deformed configuration of the object at discrete points in time.

FEM based deformable model formulations have been used in a variety of simulations such as brittle [44, 47, 48] and ductile fractures [49], surgery [50, 51, 52, 53, 54, 28, 55, 56, 57] etc. Deformable models based on finite element method can produce realistic and visually pleasing effects [29]. But they can be computationally expensive as, in the general case (except for small deformations), the stiffness matrix needs to be computed at each time step [29] and hence there may not be suitable for interactive or real-time use.

2.3 Linear Modal Analysis

Modal analysis is a method for reducing the dimension of the deformation problem from the displacements of all the nodes (or point masses) in the mesh to a much smaller set of weights of the dominant mode shapes [58]. For an object with N nodes with undeformed position vectors $\mathbf{p} = [\mathbf{p}_1, \ldots, \mathbf{p}_N]$, the deformed position vectors $\mathbf{p}'$ can be obtained by a adding to the undeformed position vectors a linear combination of its first M dominant mode shapes [58]

$$\mathbf{p}' = \mathbf{p} + \mathbf{U}_M \mathbf{q} \tag{2.12}$$

where each column of $\mathbf{U}_M$ is a mode shape $\mathbf{U}_i$ of the deformation problem and $\mathbf{q}$ is the weights (the reduced or modal coordinates).

For a dynamic system (Eq. (2.11)) with mass , damping and stiffness matrices $\mathbf{M}$,$\mathbf{D}$,$\mathbf{K}$ respectively, the set of equations can be decoupled by solving the generalized eigen value problem [59]

$$\mathbf{M}\Phi\Lambda = \mathbf{K}\Phi \tag{2.13}$$

where Λ (diagonal matrix with eigen values) and Φ (eigen vectors or mode shapes) are such that $\Phi^T\mathbf{M}\Phi = \mathbf{I}$ and $\Phi^T\mathbf{K}\Phi = \Lambda$. The unknown displacements at the nodes are further assumed to be a linear combination of the modes [60].

$$\mathbf{u}(t) = \Phi\mathbf{q}(t) \tag{2.14}$$

where $\mathbf{u}(t)$ and $\mathbf{q}(t)$ are the time-varying nodal displacements and mode weights respectively.

The relation for the displacements as a linear combination of the modes (Eq. (2.14)) is substituted in the dynamic equation Eq. (2.11) and premultiplied by the transpose of the mode shape matrix to get

$$\Phi^T\mathbf{M}\Phi\ddot{\mathbf{q}} + \Phi^T\mathbf{D}\Phi\dot{\mathbf{q}} + \Phi^T\mathbf{K}\Phi\mathbf{q} = \Phi^T\mathbf{f}_{ext} \tag{2.15}$$

By assuming a proportional (Raleigh) damping $\mathbf{C} = \alpha\mathbf{M} + \beta\mathbf{K}$ [60], the equations Eq. (2.15) are decoupled as,

$$\mathbf{I}\ddot{\mathbf{q}} + (\alpha\mathbf{I} + \beta\Lambda)\dot{\mathbf{q}} + \Lambda\mathbf{q} = \mathbf{g} \tag{2.16}$$

where $\mathbf{g} = \Phi^T\mathbf{f}_{ext}$. The modal coordinates q_i in the vector $\mathbf{q}$ is found by solving the second order differential equation

$$\ddot{q}_i + (\alpha + \beta\lambda_i)\dot{q}_i + \lambda_i q_i = g_i \tag{2.17}$$

The solution can be approximated by considering only the modal coordinates (q_i) corresponding to the larger eigen values. This usually results in considerable reduction in the number of unknowns to be estimated and saves computation time. Increasing the number of modes used increases the level of detail in the simulation. The number of nodes used can be adaptive during the simulation, ie, for portions where high detail is not required, computational costs could be saved by reducing the number of modes and for portions requiring high detail, additional modes could be included. Thus a modal representation is ideal for controlling the level of detail in the deformable object simulation.

After the introduction of modal analysis for deformable model simulations in 1989 [60], it has been used for simulating a variety of scenarios like tree branches subject to turbulence [61], linear and non-linear deformations of simple objects like cantilevers and spoons [62] etc.

2.4 Time Integration

For dynamic models, once a constitutive material and force model has been chosen, the time-dependent world coordinates $\mathbf{x}(\mathbf{m}, t)$ of all the points in the model can be calculated by a suitable numerical time integration scheme.

The dynamic equation of the model (usually Newton's second law of motion) is split up into a coupled set of two first order equations given by [29],

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= F(\mathbf{v}, \mathbf{x}) \end{aligned} \tag{2.18}$$

On integrating this system of equations, a set of values for the nodal positions $\mathbf{x}(0)$, $\mathbf{x}(\Delta t)$, $\mathbf{x}(2\Delta t)$ etc. at the discrete instants of time $0, \Delta t, 2\Delta t$ can be obtained. The nodal position values can then be used for animating the deformation of the object.

The simplest time-integration scheme is the explicit Euler integration scheme. In this scheme, the time derivatives of position and velocity are replaced by its forward finite differences [63, 64].

$$
\begin{aligned}
\dot{\mathbf{x}}(t) &= \frac{\mathbf{x}(t+\Delta t)-\mathbf{x}(t)}{\Delta t}\\
\dot{\mathbf{v}}(t) &= \frac{\mathbf{v}(t+\Delta t)-\mathbf{v}(t)}{\Delta t}
\end{aligned} \tag{2.19}
$$

The finite difference relations for the time-derivatives of the position and velocity can be substituted in the dynamic equations (Eq. (2.18)) and rearranged to get the values of position and velocity at the next time step.

$$
\begin{aligned}
\mathbf{x}(t+\Delta t) &= \mathbf{x}(t) + \Delta t\ \mathbf{v}(t)\\
\mathbf{v}(t+\Delta t) &= \mathbf{v}(t) + \Delta t\ F(\mathbf{v}(t), \mathbf{x}(t))
\end{aligned} \tag{2.20}
$$

This integration scheme is referred to as explicit because the values at the subsequent time step can be calculated directly from the values at the current time step. Such an integration scheme is stable only for time step values smaller than a particular stability threshold [65]. For stiff objects this stability threshold value can be very small which will increase the computational expense for simulating the deformation [29].

The problem with having to take smaller time steps can be overcome by changing the integration scheme to an implicit one [29]. In this scheme the right hand side of the update equations (Eq. (2.20)) is replaced with values of position and velocity at the subsequent time step [63, 64],

$$
\begin{aligned}
\mathbf{x}(t+\Delta t) &= \mathbf{x}(t) + \Delta t\ \mathbf{v}(t+\Delta t)\\
\mathbf{v}(t+\Delta t) &= \mathbf{v}(t) + \Delta t\ F(\mathbf{v}(t+\Delta t), \mathbf{x}(t+\Delta t))
\end{aligned} \tag{2.21}
$$

By solving these set of equations (linear if F is linear and non-linear otherwise) the values of position and velocity at the subsequent time step can be obtained. The scheme derives its name from the fact that the value of position and velocity in the subsequent time step is implicitly defined in the relation (Eq. (2.21)). This integration scheme is stable for arbitrarily large time steps [63, 64], but at each time step a set of equations has to be solved.

Chapter 3

Methods

3.1 Obtaining the CT ear model and the sparse 3D *in vivo* information

3.1.1 Bat experiments

Adult greater horseshoe bats (*Rhinolophus ferrumequinum*), both male and female individuals, were taken from caves in the vicinity of the laboratory in Jinan, China, where the experiments were conducted. They were housed in the laboratory for a few days prior to recording of the pinna deformations and fed on a diet of meal worms and water (available ad libitum). Recordings of pinna deformations were evaluated for a total of five individuals. Multiple pinna deformation behaviors (up to 20) were recorded for each individual.

Colored dots serving as artificial landmarks were painted on the pinna using a mixture of visible bright pigments (nail polish) and an X-ray contrast agent (barium sulfate). Around 60 landmarks were uniformly distributed over the entire visible portion of the frontal pinna (Fig. 3.1a,b). The diameters of these landmark dots typically ranged from approximately 0.3 mm to 1 mm with a few outliers reaching diameters up to 1.6 mm. The animal was subsequently transferred to a padded holder which restrained the body but allowed for free movements of the head. The movements of the painted pinna of each individual tested were recorded with a synchronized stereo pair of high-speed video cameras (GigaView, Southern Vision Systems with Rodenstock Rodagon 50 mm-lens with Navitar 25 mm modular focus block) at frame rates of 200 or 300 Hz. All components of the experimental setup, the two high-speed video cameras, camera lights (key light and multiple fill lights), and the bat holder were mounted on a vibration-damped table. Biosonar emissions of the bat were monitored using a bat detector (D1000X, Pettersson Elektronik AB).

As part of each experiment, multiple video images (around 14 for each configuration) were

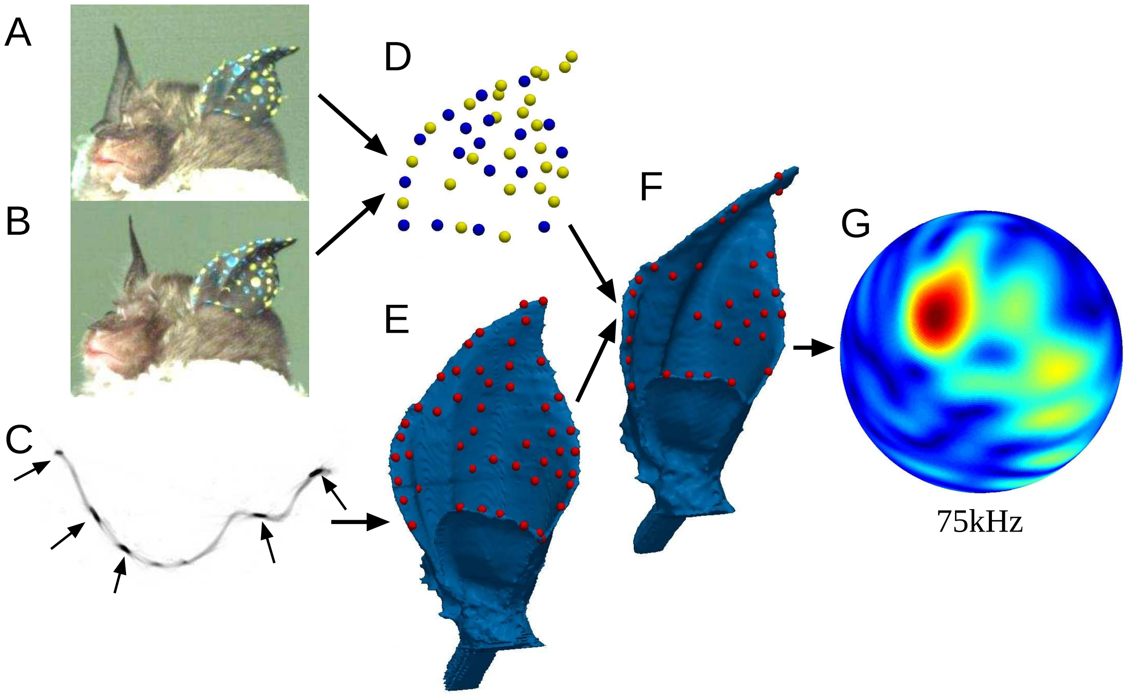

Figure 3.1: Overview of the experimental work flow: a,b) stereo pair of images of a deforming pinna with landmarks captured by the left (a) and right (b) camera, c) A sample tomographic cross-section image of the pinna. Landmarks (arrows) appear in darker gray value than the pinna tissue, d) The three-dimensional location of the landmarks obtained through stereo triangulation from the left and right camera images. Due to occlusions and poor contrast in the images, the three-dimensional locations of some of the landmarks could not be obtained, e) Complete digital model of the pinna with the landmarks obtained from the cross-section images. As with the stereo images, not all landmarks could be identified with a sufficient level of certainty. f) The complete shape model of the pinna during the deformation cycle obtained by fitting the CT model to the estimated locations of the landmarks. Only landmarks clearly identified as identical in the CT model and video images were used. g) A beampattern of the deformed shape estimated using numerical acoustics.

taken of a planar black-and-white chessboard calibration pattern (9×7, edge length of each square 14.1 mm). The board was placed in the same space that was occupied by the bat during the experiment and was oriented in various directions to create a comprehensive calibration dataset.

After obtaining satisfactory high-speed video recordings from an individual bat, the animal was sacrificed and the pinna from which the recordings had been obtained was mounted on a sample holder in an upright position. A high-resolution computer tomographic model (isotropic resolution $35.78\,\mu$ m, Fig. 3.1c,e) of the pinna was then obtained using a μCT machine (Skyscan 1172).

3.1.2 Stereo Triangulation and reconstruction of the μCT images

The head of the bat was positioned at a distance of approximately 60 cm from the image planes of the two video cameras. The distance between the principal points of the two cameras was approximately 10 cm. In this configuration, the landmarks on the pinna remained visible in each of the images of the stereo pairs captured by the two cameras while at the same time providing a stereo basis large enough for accurate position estimation. Each frame in the stream of high-speed video recordings had a resolution of 1280 × 1024 pixels. At this resolution and the configuration of the stereo camera setup used, one pixel in the video sequence images corresponded to ∼0.1 mm in the object plane.

The cameras and the stereo system setup were first calibrated using the images of a chessboard pattern. The calibration images were used to estimate the intrinsic parameters of each camera as well as the geometric relationship between the two camera coordinate systems. The intrinsic parameters estimated for each camera were: the focal lengths in the x and y directions of the image, the image coordinates of the principal point, a skewness coefficient that defines the angle between the x and y axis of the image plane, radial distortion coefficients that describe distortion due to unequal bending of light rays near and away from the lens center and tangential distortion coefficients that describe the distortion due to the lens and image planes not being exactly parallel. All of these intrinsic parameters were obtained independently for each camera using well-established methods [66]. To characterize the stereo setup, translation vectors and rotational matrices to transform the coordinate system of one camera into that of the other were estimated. This was accomplished by first finding the geometric relationship (translation and rotation) between the chessboard coordinate system and the coordinate system of each camera. From these two relationships to the same chessboard coordinate system, the geometric relationship between the two camera coordinate systems was estimated [67, chap. 11].

Based on the estimates of intrinsic camera and stereo configuration parameters, the positions of the landmarks in three-dimensional space were obtained by means of stereo triangulation. As a first step in the triangulation, the stereo images obtained from the setup were rectified using Bouguet's algorithm [67, chap. 12]. The results were equivalent images that would have been obtained from a frontal parallel configuration (principal axes of the camera pair parallel, the image planes coplanar and the corresponding points in the images always in the same pixel row [67, chap. 12]). Corresponding landmark points in the rectified images were picked manually and the three-dimensional coordinates of the landmarks in the camera coordinate systems were calculated using stereo triangulation relations for frontal parallel stereo systems [67, chap. 12]

The μCT model of the pinna was obtained for a single (upright or close to upright) position of each analyzed pinna sample. Cross-sections of the samples were constructed from sets of its X-ray shadow images using a cone-beam volumetric reconstruction method (Feldkamp algorithm). The reconstructed model was down sampled by a factor of three to an isotropic resolution of 107.34 μm. The reduction in resolution was done so as to reduce the computational time and memory required for At this resolution, the structure was sampled with 40 elements per wavelength at the highest frequency simulated (80 kHz, i.e., $\sim$4.3 mm wavelength).

The differences in the X-ray absorptions of the pinna tissue and the landmarks were manifest in different gray-scale intensities in the cross-section images. The landmark voxels were separated from the pinna tissue voxels by thresholding at different gray-scale intensity levels. For all the landmarks, the centroid of the voxels representing the landmark was obtained and the center of the voxel containing this centroid was designated as the position of the landmark. This is converted to a mesh by replacing each voxel in the digital representation by a cubic element with nodes at each corner of the voxel. A list of the three dimensional coordinates of each node, an element wise list of the nodes making up each element and a list of the element numbers whose centers represent the landmark was stored in the mesh file.

3.1.3 Computing the displacements at the landmarks

The CT model is in the μCT coordinates while the landmark locations are in the camera coordinates. Before the displacements at the landmarks on the CT model can be calculated, these coordinate frames need to be matched (s. Fig. 3.2). The CT model was aligned with the landmark positions for the most upright position of the pinna during the deformation cycle by rigid registration (Iterative Closest Point method [68]). The required correspondences between the landmarks on the CT model and the stereo-triangulated landmarks during the deformation cycle were then established manually. The difference between the three-dimensional location of the stereo-triangulated landmarks and the corresponding ones in the rigid registered CT model was used as an estimate of the displacements at the landmarks.

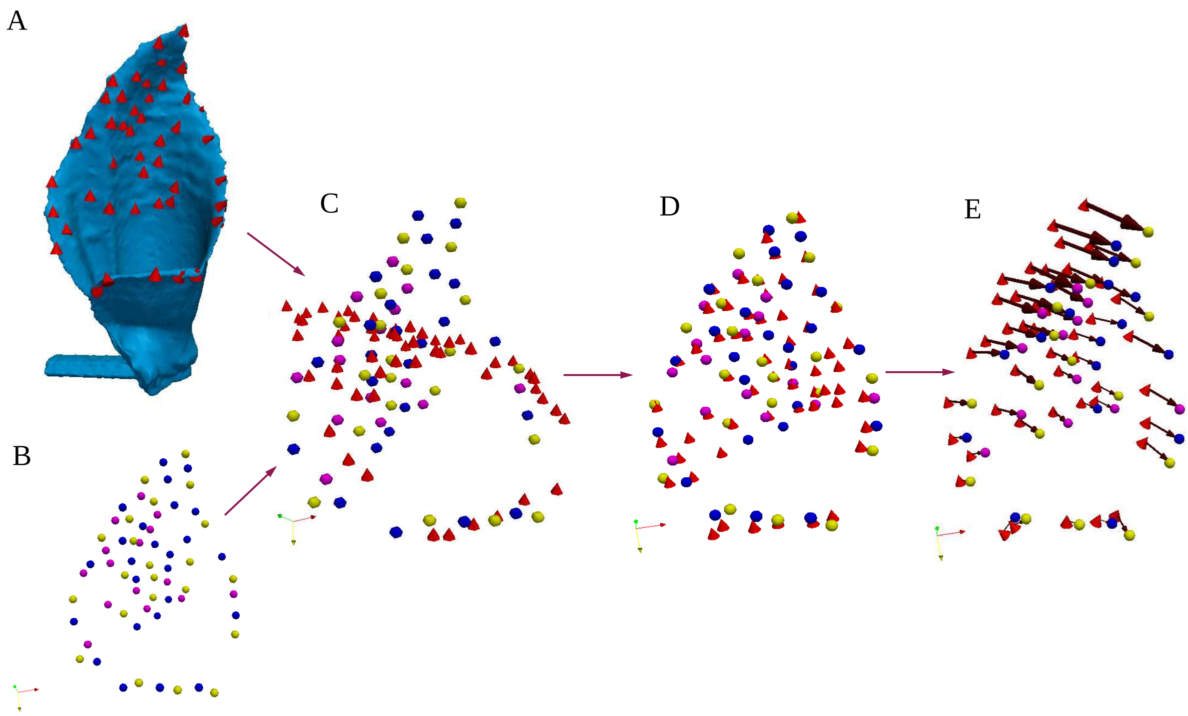

Figure 3.2: Calculating the displacements at the landmarks: a) CT model (blue) with the landmarks (red cones) b) Stereo triangulated landmarks for the most upright position of the pinna during the ear movements (blue, yellow and magenta spheres) c) Landmarks in the CT model (a) and stereo-triangulated landmarks for the most upright position of the pinna (b) translated such that their centroids are coincident d) Landmarks in the CT model (a) and stereo-triangulated landmarks for the most upright position of the pinna (b) aligned using the Iterative Closest Point method e) The displacements at the landmarks calculated for deforming the rigid registered CT model to a configuration of the pinna during the ear movements

3.2 Deformable Model Formulation

The complete *in vivo* shape of the pinna during the ear deformations is obtained by combining the CT model with the sparse set of landmark displacements through a deformable model formulation. Combining the information from these two sources (static shape model and stereo triangulation) can be treated as an interpolation or extrapolation problem where the unknown displacements of the pinna tissue between landmarks are estimated from the known displacements at the landmarks. This approach is aided by the fact that occlusions are mostly limited to the base of the pinna where only small displacements can be expected since the pinna is attached to the skull in this region. In the regions near the tip of the pinna where large pinna deformations can occur, a large number of landmark positions is available for interpolation and extrapolation of the pinna surface deformations. Under these circumstances, it is not necessary to model the mechanical tissue properties in detail. Instead, a simple linearly elastic model for the pinna tissue can be used. The linearity of this model refers to the relationship between the stress and strain components which can be given as [42]:

$$\begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{xy} \\ \sigma_{yz} \\ \sigma_{zx} \end{bmatrix} = \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix} \begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \epsilon_{zz} \\ 2\epsilon_{xy} \\ 2\epsilon_{yz} \\ 2\epsilon_{zx} \end{bmatrix}, \tag{3.1}$$

where the σ_{ij} refers to the stress components, λ and μ are Lame's constants and the ϵ_{ij} refer to the strain components. The Lame's constants are related to the Young's modulus Y and the Poisson's ratio ν according to the relations [42]

$$\lambda = \frac{\mathrm{Y}\nu}{(1+\nu)(1-2\nu)}, \quad \mu = \frac{\mathrm{Y}}{2(1+\nu)} \tag{3.2}$$

The stress strain relationship (Eq. (3.1)) can be written in matrix-vector form as

$$[\boldsymbol{\sigma}] = [\mathbf{C}]\,[\boldsymbol{\epsilon}]\,, \tag{3.3}$$

where $[\boldsymbol{\sigma}]$,$[\mathbf{C}]$ and $[\boldsymbol{\epsilon}]$ respectively are the stress vector, material matrix, and strain vector that are also given in Eq. (3.1).

The strain components are related to the first derivatives of the displacements by [42]

$$\begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \epsilon_{zz} \\ 2\epsilon_{xy} \\ 2\epsilon_{yz} \\ 2\epsilon_{zx} \end{bmatrix} = \begin{bmatrix} \frac{\partial u_x}{\partial x} \\ \frac{\partial u_y}{\partial y} \\ \frac{\partial u_z}{\partial z} \\ \frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x} \\ \frac{\partial u_y}{\partial z} + \frac{\partial u_z}{\partial y} \\ \frac{\partial u_z}{\partial x} + \frac{\partial u_x}{\partial z} \end{bmatrix}, \tag{3.4}$$

where u_i is the displacement in direction i. A finite-element model with linear cube elements was used to evaluate the linear elastic model numerically. The displacement components at any point in the model's domain are interpolated linearly from the values at the nodes of the finite-element meshes immediately surrounding the point according to the relation [43],

$$u_i = \sum_{p=1}^{8} N_p u_{ip}, \tag{3.5}$$

where N_p is the p^{th} first-order shape function associated with the element and u_{ip} refers to the displacement in the i direction at node number p. Differentiating Eq. (3.5), the partials of the displacements are given by

$$\frac{\partial u_i}{\partial j} = \sum_{p=1}^{8} \frac{\partial N_p}{\partial j} u_{ip}, \tag{3.6}$$

where $\frac{\partial u_i}{\partial j}$ is the partial derivative in direction j of the displacement in direction i and $\frac{\partial N_p}{\partial j}$ is the partial derivative in direction j of the p^{th} first-order shape function. Eq. (3.4) can be combined with Eq. (3.6) and the strain vector can be written as

$$[\boldsymbol{\epsilon}] = [\mathbf{dN}]\,[\mathbf{u}]\,, \tag{3.7}$$

where [**dN**] is a matrix of the partial derivatives of the shape functions and [**u**] is a vector of the displacements (in all the three directions - x, y, and z) at the nodes of the cube elements.

The shape function derivatives are arranged in [**dN**] according to

$$[\mathbf{dN}] = \begin{bmatrix} \frac{\partial N_1}{\partial x} & 0 & 0 & \frac{\partial N_2}{\partial x} & 0 & 0 & \dots\dots & \frac{\partial N_8}{\partial x} & 0 & 0 \\ 0 & \frac{\partial N_1}{\partial y} & 0 & 0 & \frac{\partial N_2}{\partial y} & 0 & \dots\dots & 0 & \frac{\partial N_8}{\partial y} & 0 \\ 0 & 0 & \frac{\partial N_1}{\partial z} & 0 & 0 & \frac{\partial N_2}{\partial z} & \dots\dots & 0 & 0 & \frac{\partial N_8}{\partial z} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial x} & 0 & \frac{\partial N_2}{\partial y} & \frac{\partial N_2}{\partial x} & 0 & \dots\dots & \frac{\partial N_8}{\partial y} & \frac{\partial N_8}{\partial x} & 0 \\ 0 & \frac{\partial N_1}{\partial z} & \frac{\partial N_1}{\partial y} & 0 & \frac{\partial N_2}{\partial z} & \frac{\partial N_2}{\partial z} & \dots\dots & 0 & \frac{\partial N_8}{\partial z} & \frac{\partial N_8}{\partial y} \\ \frac{\partial N_1}{\partial z} & 0 & \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial z} & 0 & \frac{\partial N_2}{\partial x} & \dots\dots & \frac{\partial N_8}{\partial z} & 0 & \frac{\partial N_8}{\partial x} \end{bmatrix} \tag{3.8}$$

The deformation strain energy per volume (e) at any point in the domain is given by the sum of the product of the stress components with their respective strains [42].

$$\begin{aligned} e =& \frac{1}{2}\big[\sigma_{xx}\epsilon_{xx} + \sigma_{yy}\epsilon_{yy} + \sigma_{zz}\epsilon_{zz} + \sigma_{xy}\epsilon_{xy} + \sigma_{yx}\epsilon_{yx} + \\ & \sigma_{yz}\epsilon_{yz} + \sigma_{zy}\epsilon_{zy} + \sigma_{zx}\epsilon_{zx} + \sigma_{xz}\epsilon_{xz}\big] \end{aligned} \tag{3.9}$$

From the symmetries $\sigma_{ij} = \sigma_{ji}$ [42]and $\epsilon_{ij} = \epsilon_{ji}$, Eq. (3.9) can be simplified to

$$\begin{aligned} e =& \frac{1}{2}\big[\sigma_{xx}\epsilon_{xx} + \sigma_{yy}\epsilon_{yy} + \sigma_{zz}\epsilon_{zz} + \underbrace{\sigma_{xy}\epsilon_{xy} + \sigma_{yx}\epsilon_{yx}}_{2\sigma_{xy}\epsilon_{xy}} + \\ & \underbrace{\sigma_{yz}\epsilon_{yz} + \sigma_{zy}\epsilon_{zy}}_{2\sigma_{yz}\epsilon_{yz}} + \underbrace{\sigma_{zx}\epsilon_{zx} + \sigma_{xz}\epsilon_{xz}}_{2\sigma_{zx}\epsilon_{zx}}\big] \\ =& \frac{1}{2}\left\{\sigma_{xx}\epsilon_{xx} + \sigma_{yy}\epsilon_{yy} + \sigma_{zz}\epsilon_{zz} + 2\sigma_{xy}\epsilon_{xy} + 2\sigma_{yz}\epsilon_{yz} + 2\sigma_{zx}\epsilon_{zx}\right\} \end{aligned} \tag{3.10}$$

From Eqs. (3.1) and (3.10), the following expression for the strain energy can be obtained

$$\begin{aligned} e =& \frac{1}{2}\begin{bmatrix}\epsilon_{xx} & \epsilon_{yy} & \epsilon_{zz} & 2\epsilon_{xy} & 2\epsilon_{yz} & 2\epsilon_{zx}\end{bmatrix} \begin{bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \sigma_{zz}\\ \sigma_{xy}\\ \sigma_{yz}\\ \sigma_{zx}\end{bmatrix} \\ =& \frac{1}{2}\begin{bmatrix}\epsilon_{xx} & \epsilon_{yy} & \epsilon_{zz} & 2\epsilon_{xy} & 2\epsilon_{yz} & 2\epsilon_{zx}\end{bmatrix} \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix} \begin{bmatrix}\epsilon_{xx}\\ \epsilon_{yy}\\ \epsilon_{zz}\\ 2\epsilon_{xy}\\ 2\epsilon_{yz}\\ 2\epsilon_{zx}\end{bmatrix} \end{aligned} \tag{3.11}$$

which is the same as

$$e = \frac{1}{2} [\boldsymbol{\epsilon}]^T [\mathbf{C}] [\boldsymbol{\epsilon}] \tag{3.12}$$

Substituting for $[\boldsymbol{\epsilon}]$ from Eq. (3.7) results in

$$e = \frac{1}{2} [\mathbf{u}]^T [\mathbf{dN}]^T [\mathbf{C}] [\mathbf{dN}] [\mathbf{u}] . \tag{3.13}$$

The deformation strain-energy per volume given in Eq. (3.13) can be integrated over the volume of the element to arrive at the total deformation strain-energy of the element [43].

$$\begin{aligned} E_{\text{element}} &= \oint_{\mathbf{v}} e dv \\ &= \frac{1}{2} [\mathbf{u}]^T \left\{ \oint_{\mathbf{v}} [\mathbf{dN}]^T [\mathbf{C}] [\mathbf{dN}] \, dv \right\} [\mathbf{u}] \end{aligned} \tag{3.14}$$

Each entry in the matrix $[\mathbf{dN}]^T [\mathbf{C}] [\mathbf{dN}]$ can be integrated over the volume of the element [43] and the resulting matrix is called the elemental stiffness matrix $[\mathbf{k}]$. Eq. (3.14) can then be written as [43]

$$E_{\text{element}} = \frac{1}{2} [\mathbf{u}]^T [\mathbf{k}] [\mathbf{u}] \tag{3.15}$$

The total deformation energy E can be obtained by adding the E_{element} of every element. The $[\mathbf{u}]$ and $[\mathbf{k}]$ from the various elements can be combined to a global $[\mathbf{U}]$ and $[\mathbf{K}]$ and the total deformation energy E can then be written as [43]

$$E = \frac{1}{2} [\mathbf{U}]^T [\mathbf{K}] [\mathbf{U}] \tag{3.16}$$

The dimension of the matrix $[\mathbf{K}]$ is $3n \times 3n$ and the length of the vector $[\mathbf{U}]$ is $3n$ where n is the total number of nodes in the mesh.

The displacements of the landmarks are constraints imposed on the deformation of the pinna. This can be written as

$$[\mathbf{H}] [\mathbf{U}] = [\mathbf{D}] , \tag{3.17}$$

where $[\mathbf{H}]$ is a selection matrix (dimension $3l \times 3n$) that selects the displacements of the l landmarks from the vector of all n displacements ($[\mathbf{U}]$) resulting in a vector of displacements at the landmarks ($[\mathbf{D}]$, of length $3l$). Each row of the $[\mathbf{H}]$ matrix multiplies with the vector of displacements ($[\mathbf{U}]$) to get the displacement of a landmark in one of the coordinate directions. Ie, each row of $[\mathbf{H}]$ matrix contains a value of $1/8$ at the columns which will multiply with the displacements of the nodes that make up a landmark element.

Hence, the problem of interpolating the landmarks smoothly can be stated as:

$$\text{Minimize } \frac{1}{2} [\mathbf{U}]^T [\mathbf{K}] [\mathbf{U}] \, \text{subject to the constraints } \; [\mathbf{H}] [\mathbf{U}] = [\mathbf{D}] \tag{3.18}$$

This problem can be formulated by combining the function to be minimized and the constraints through a vector of Lagrangian multipliers $[\mathbf{\Lambda}]$ as

$$\text{Minimize} \quad W = \frac{1}{2}[\mathbf{U}]^T[\mathbf{K}][\mathbf{U}] + [\mathbf{\Lambda}]^T([\mathbf{H}][\mathbf{U}] - [\mathbf{D}]) \tag{3.19}$$

The minimum can be found by solving

$$\frac{\partial W}{\partial \mathbf{U}} = 0 \Rightarrow [\mathbf{K}][\mathbf{U}] + [\mathbf{H}]^T[\mathbf{\Lambda}] = 0 \tag{3.20}$$

$$\frac{\partial W}{\partial \mathbf{\Lambda}} = 0 \Rightarrow [\mathbf{H}][\mathbf{U}] - [\mathbf{D}] = 0 \tag{3.21}$$

or in matrix form:

$$\begin{bmatrix} \mathbf{K} & \mathbf{H}^T \\ \mathbf{H} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{U} \\ \mathbf{\Lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{D} \end{bmatrix} \tag{3.22}$$

$$\Rightarrow \begin{bmatrix} \mathbf{U} \\ \mathbf{\Lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{K} & \mathbf{H}^T \\ \mathbf{H} & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{D} \end{bmatrix} \tag{3.23}$$

Solving Eq. (3.19) results in an exact interpolation where all landmarks in the deformable model will be displaced by $[\mathbf{D}]$ to match their positions as determined by the stereo triangulation. Since these position estimates are noisy experimental data, it was chosen not to match them exactly, but rather in a least-squares sense. This can be accomplished by including a penalty term [69] which was defined as

$$E_{\text{penalty}} = \frac{1}{2}([\mathbf{H}][\mathbf{U}] - [\mathbf{D}])^{\mathrm{T}}[\mathbf{S}]([\mathbf{H}][\mathbf{U}] - [\mathbf{D}]) \tag{3.24}$$

where $[\mathbf{S}]$ is an identity matrix of dimension $3l \times 3l$

The penalty is a quadratic function of the difference between the landmark displacements estimated from the images ($[\mathbf{D}]$) and the landmark displacements of the model ($[\mathbf{H}][\mathbf{U}]$). Using this squared-error concept, the interpolation problem can be formulated as a minimization of a weighted sum of the deformation energy and the penalty:

$$\begin{aligned} \text{Minimize} \quad W &= E + \Theta E_{\text{penalty}} \\ &= \frac{1}{2}[\mathbf{U}]^T[\mathbf{K}][\mathbf{U}] + \Theta\left\{\frac{1}{2}([\mathbf{H}][\mathbf{U}] - [\mathbf{D}])^{\mathrm{T}}[\mathbf{S}]([\mathbf{H}][\mathbf{U}] - [\mathbf{D}])\right\} \end{aligned} \tag{3.25}$$

where Θ is a scalar weighing parameter. To adjust the weighting between the two energy terms, a scaled version θ of the matching parameter Θ was used:

$$\theta = \Theta \Big/ \frac{\text{trace}([\mathbf{K}])}{nl} \tag{3.26}$$

In this scaling, Θ was divided by the average stiffness of the nodes ($\frac{\text{trace}([\mathbf{K}])}{n}$) divided among the l landmarks. This normalization reduced the dependence of the weighting parameter on the individual model and hence facilitated comparison across the models of different pinnae.

The minimization problem in Eq. (3.25) can be solved by evaluating the analytical expression:

$$\frac{\partial W}{\partial \mathbf{U}} = 0$$

$$\Rightarrow \quad [\mathbf{K}]\,[\mathbf{U}] + \Theta\,[\mathbf{H}]^{\mathrm{T}}\,[\mathbf{S}]\,[\mathbf{H}]\,[\mathbf{U}] = \Theta\,[\mathbf{H}]^{\mathrm{T}}\,[\mathbf{S}]\,[\mathbf{D}]$$

$$\Rightarrow \quad [\mathbf{U}] = \left\{[\mathbf{K}] + \Theta\,[\mathbf{H}]^{\mathrm{T}}\,[\mathbf{S}]\,[\mathbf{H}]\right\}^{-1} \Theta\,[\mathbf{H}]^{\mathrm{T}}\,[\mathbf{S}]\,[\mathbf{D}] \tag{3.27}$$

3.3 Deformation Quality Measures

To select a θ value, two quality measures for the deformed pinna shape were used: one to quantify the match to the measured location of the landmarks (improves as θ increases) and another to represent the life-likeness (smoothness) of the model surfaces (improves as θ decreases).

3.3.1 Matching RMS Error

The error in matching the landmarks was defined as the root-mean-square value of the distance between the location of the landmarks in the deformed model and the location obtained from the stereo-triangulation:

$$err_{\text{match}} = \sqrt{\frac{1}{l}\left([\mathbf{H}]\,[\mathbf{U}] - [\mathbf{D}]\right)^{\mathrm{T}}\left([\mathbf{H}]\,[\mathbf{U}] - [\mathbf{D}]\right)} \tag{3.28}$$

This quantity measures the conformance of the position of the landmarks in the estimated deformation to the stereo-triangulated location of the landmarks.

3.3.2 Smoothness Measures

Smoothness measures were devised such that it would measure the life-likeness of a deformation. Two smoothness measures - one using the curvature of the pinna surface and the other quantifying the amount of stretching of the pinna surface were used.

Curvature : Curvature is an intrinsic property of a surface that is independent of the coordinate system used to describe the surface. Through every point on any continuous surface, an infinite number of planes perpendicular to the tangent plane at that point can be constructed. The intersection of each of these perpendicular planes with the original surface produces a curve. By calculating the curvatures of each of these curves at the point, an infinite set curvature values at the point is obtained. The maximum and minimum of this infinite set of curvature values are called principal curvatures.

To compute the curvature, the surface of the pinna model was represented by a triangular mesh using the marching cubes algorithm [70] implemented in the *Visualization Toolkit* [71]. Only the top portion of the pinna model (above the tip of the antitragus) was used in the analysis, since the base of the pinna deforms very little and hence had a very low susceptibility to the appearance of artifacts. For each node in the selected portion of the surface mesh, nodes from a 3-neighborhood were used to fit a third-degree implicit surface that passes through the current node and fitted its neighbors in a weighted least-squares sense (s. Fig. 3.3). A Gaussian weighting function was used [72]

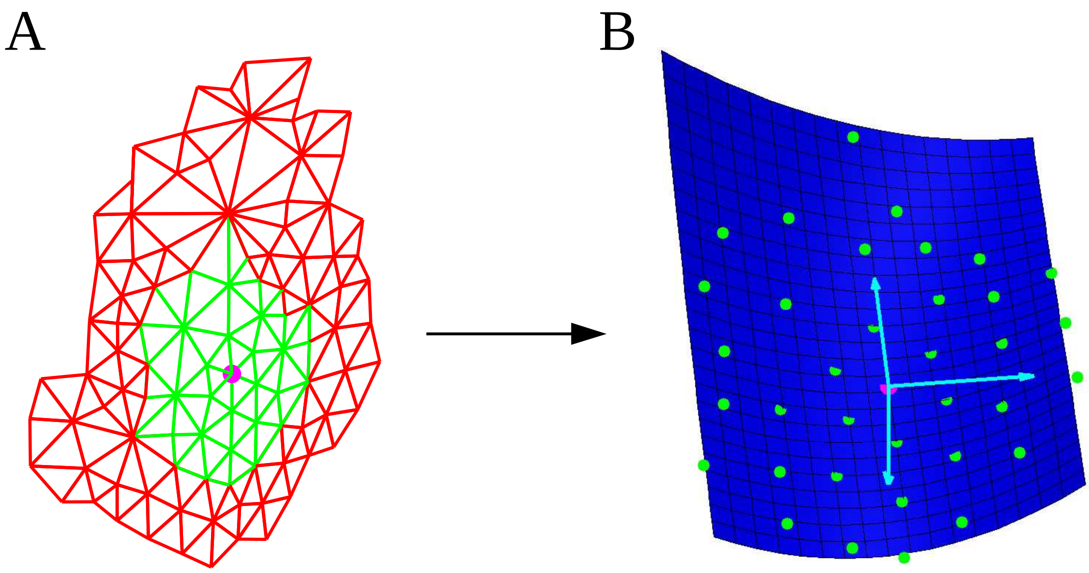

Figure 3.3: Fitting surface to mesh: a) A mesh node (magenta) and its 3-neighborhood (green) b) A third degree implicit surface (blue) passing through the node (magenta) and weighted least squares fitting its 3-neighborhood nodes.

$$W_i = e^{-\frac{s_i^2}{h^2}} \tag{3.29}$$

where W_i is the weight of the i$^{\text{th}}$ neighboring node, s_i is the distance of the i$^{\text{th}}$ neighboring node from the central node (the node for which the curvature is being calculated - point marked in magenta in Fig. 3.3) and h is the width of the Gaussian weighting function (was taken as one-third of the distance from the central node to its farthest neighbor).

The maximum principle curvature [73] of the implicit surface at the central node was then assigned to central node as a local measure of smoothness [72]. The size of the neighborhood and the degree of the implicit surface were selected to provide some noise removal. The mean of the curvature at the nodes of the surface mesh was taken as the smoothness measure of the deformation.

Surface Energy : An energy term which represents the stretching of the pinna surface under a deformation was formulated to serve as a smoothness measure. At each point **P** on the CT model surface a coordinate reference frame **uvw** is defined such that **w** is perpendicular to the surface at point **P** and **u** and **v** are two mutually perpendicular vectors in the plane tangent to the surface at **P**. Under the deformation, the point **P** goes to **P'**. The components of the deformation from **P** to **P'** along the **u**, **v**, **w** directions are ϵ, η, ζ. The strains due to stretching in the tangent plane at the point **P** while deforming to **P'** is $[\frac{\partial\epsilon}{\partial u} \quad \frac{\partial\eta}{\partial v}]$. The stresses due to stretching can be obtained from the strains by multiplying with suitable material constants [74].

$$\begin{aligned}\sigma_u &= \alpha_{11}\frac{\partial\epsilon}{\partial u}\\ \sigma_v &= \alpha_{22}\frac{\partial\eta}{\partial v}\end{aligned} \tag{3.30}$$

where α_{11} and α_{22} are the material constants. The infinitesimal strain energy due to stretching at point **P** is the sum of the products of the stress components with their respective strains [74]

$$\begin{aligned}E_{\text{stretch}} &= \frac{1}{2}\left(\alpha_{11}\left(\frac{\partial\epsilon}{\partial u}\right)^2 + \alpha_{11}\left(\frac{\partial\eta}{\partial v}\right)^2\right)dA\\ &= \frac{1}{2}\begin{bmatrix}\frac{\partial\epsilon}{\partial u} & \frac{\partial\eta}{\partial v}\end{bmatrix}\begin{bmatrix}\alpha_{11} & 0\\ 0 & \alpha_{22}\end{bmatrix}\begin{bmatrix}\frac{\partial\epsilon}{\partial u}\\ \frac{\partial\eta}{\partial v}\end{bmatrix}dA\\ &= \frac{1}{2}[\boldsymbol{\epsilon}]^T[\boldsymbol{\alpha}][\boldsymbol{\epsilon}]\end{aligned} \tag{3.31}$$

where $[\boldsymbol{\epsilon}]$ is the stretching strain vector,where the combined material matrix is $[\boldsymbol{\alpha}]$. From expression for the stretching energy Eq. (3.31) it can be seen that the material matrix $[\boldsymbol{\alpha}]$ acts just like a scaling factor. Hence for simplicity, it was chosen to be an identity matrix

The total stretching energy can be found out by integrating the stretching energy given in Eq. (3.31) over the total surface.

$$\mathbf{E}_{\text{stretch}} = \int_\sigma \frac{1}{2}[\boldsymbol{\epsilon}]^T[\boldsymbol{\alpha}][\boldsymbol{\epsilon}]dA \tag{3.32}$$

where σ is the total surface domain.

The continuous domain σ is split up into finite triangular elements and the integral given in Eq. (3.32) is evaluated for each element and added up to get the total stretching energy [74].

$$\mathbf{E}_{\text{stretch}} = \sum_{n=1}^{nels}\int_{el_n}\frac{1}{2}[\boldsymbol{\epsilon}]^T[\boldsymbol{\alpha}][\boldsymbol{\epsilon}]dA \tag{3.33}$$

where $nels$ is the number of elements and el_n is the n$^{\text{th}}$ element.

The surface mesh of the top portion (part of the ear above the antitragus) of the ear is extracted from the original CT model mesh. The faces of the cube mesh which are on the outer surface of the ear are obtained and each of these square faces are further split into two triangles by dividing along one of the diagonals.

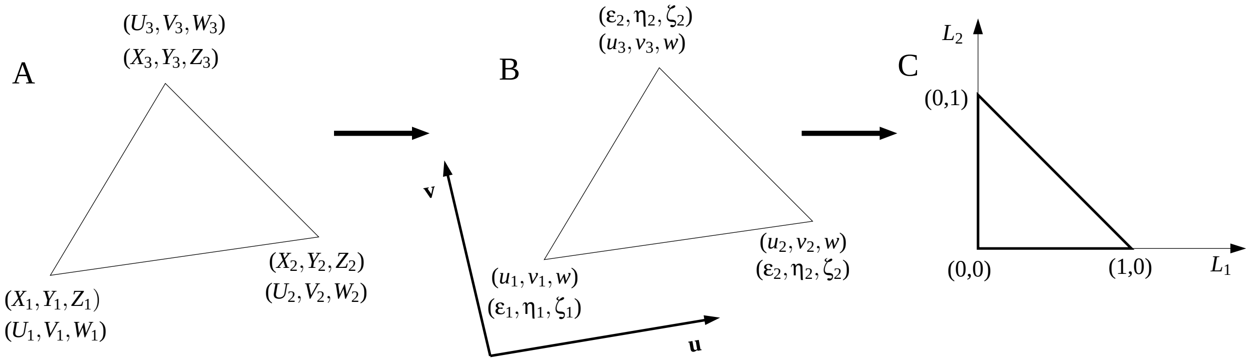

Figure 3.4: Surface mesh triangle transformations: a) A triangle in the surface mesh of the CT model with its nodal coordinates and displacements in the **XYZ** coordinate system b) The nodal coordinates and displacements transformed to a **uvw** coordinate system where **w** is perpendicular to the triangular surface and **u** and **v** are two mutually perpendicular vectors in the plane of the triangle c) The triangle in the **uvw** coordinate system is further mapped into a standard right angled triangle described in a barycentric coordinate system $\mathbf{L_1L_2}$

The ordering of the nodes in the triangles were chosen such that the normal (cross product of the vector from node 1 to 2 with the vector from node 1 to 3) points away from the pinna surface. A coordinate system **uvw** is defined such that the **w** axis is along the normal of the triangle and **u**,**v** are two mutually perpendicular axes in the plane of the triangle. The **u** direction was conveniently chosen as parallel to the vector from node 1 to node 2. Subsequently **v** was chosen such that the coordinate system **uvw** would be right handed. The coordinates (X_r, Y_r, Z_r) and the displacements (U_r, V_r, W_r) at the nodes in the original **XYZ** coordinate system were transformed (by rotation) into coordinates (u_r, v_r, w_r) and displacements $(\epsilon_r, \eta_r, \zeta_r)$ in the **uvw** coordinate system [74].

$$\begin{bmatrix} u_1 \\ v_1 \\ w_1 \\ u_2 \\ v_2 \\ w_2 \\ u_3 \\ v_3 \\ w_3 \end{bmatrix} = \begin{bmatrix} [R] & 0 & 0 \\ 0 & [R] & 0 \\ 0 & 0 & [R] \end{bmatrix} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ X_3 \\ Y_3 \\ Z_3 \end{bmatrix}$$

$$\begin{bmatrix} \epsilon_1 \\ \eta_1 \\ \zeta_1 \\ \epsilon_2 \\ \eta_2 \\ \zeta_2 \\ \epsilon_3 \\ \eta_3 \\ \zeta_3 \end{bmatrix} = \begin{bmatrix} [R] & 0 & 0 \\ 0 & [R] & 0 \\ 0 & 0 & [R] \end{bmatrix} \begin{bmatrix} U_1 \\ V_1 \\ W_1 \\ U_2 \\ V_2 \\ W_2 \\ U_3 \\ V_3 \\ W_3 \end{bmatrix} \tag{3.34}$$

where [R] is the 3×3 rotation matrix which transforms (only rotation no translation) the **XYZ** coordinate system to the **uvw** coordinate system. The displacement mapping can be written in a simpler matrix form as

$$[\mathbf{d_{uvw}}] = [\mathbf{R}][\mathbf{d_{XYZ}}] \tag{3.35}$$

where $[\mathbf{d_{uvw}}]$ and $[\mathbf{d_{XYZ}}]$ are the displacements in the **uvw** and **XYZ** coordinate systems respectively and $[\mathbf{R}]$ is the 9×9 transformation matrix obtained by assembling the 3×3 [R] matrices as given in Eq. (3.34).

Since the three nodes define a plane, and the **w** direction is perpendicular to the plane, all the nodes have the same **w** coordinate value (s. Fig. 3.4).

The displacements inside the element were interpolated from the nodal displacements using first order shape functions [74].

$$\begin{aligned} \epsilon &= N_1\epsilon_1 + N_2\epsilon_2 + N_3\epsilon_3 \\ \eta &= N_1\eta_1 + N_2\eta_2 + N_3\eta_3 \\ \zeta &= N_1\zeta_1 + N_2\zeta_2 + N_3\zeta_3 \end{aligned} \tag{3.36}$$

The required tangential strain components can be then obtained by differentiation given as

$$\begin{aligned} \frac{\partial \epsilon}{\partial u} &= \frac{\partial N_1}{\partial u}\epsilon_1 + \frac{\partial N_2}{\partial u}\epsilon_2 + \frac{\partial N_3}{\partial u}\epsilon_3 \\ \frac{\partial \eta}{\partial v} &= \frac{\partial N_1}{\partial v}\eta_1 + \frac{\partial N_2}{\partial v}\eta_2 + \frac{\partial N_3}{\partial v}\eta_3 \end{aligned} \tag{3.37}$$

This can be written in matrix form as (similar to Eq. (3.7)),

$$[\boldsymbol{\epsilon}] = [\mathbf{dN}][\mathbf{d_{uvw}}] \tag{3.38}$$

where $[\boldsymbol{\epsilon}]$ is the strain matrix and $[\mathbf{dN}]$ is a matrix of the derivatives of shape functions.

Using (Eqs. (3.33) (3.38)), the elemental surface energy due to stretching can be written as

$$E_{stretch} = \iint \frac{1}{2}[\mathbf{d_{uvw}}]^T[\mathbf{dN}]^T[\boldsymbol{\alpha}][\mathbf{dN}][\mathbf{d_{uvw}}]dudv \tag{3.39}$$

The displacements in the **uvw** coordinate system $[\mathbf{d_{uvw}}]$ can be substituted with those in the **XYZ** coordinate system $[\mathbf{d_{XYZ}}]$(Eq. (3.35)) and can be taken outside the integral as they are constants and do not vary inside the triangle. Eq. (3.39) can then be written as

$$E_{stretch} = \frac{1}{2}[\mathbf{d_{XYZ}}]^T[\mathbf{R}]^T \left(\iint [\mathbf{dN}]^T[\boldsymbol{\alpha}][\mathbf{dN}]dudv \right) [\mathbf{R}][\mathbf{d_{XYZ}}] \tag{3.40}$$

The integrand inside the braces in Eq. (3.40) expands to a 9×9 matrix and is obtained by integrating each of its elements. The integral is calculated by mapping the triangle in the **uvw** coordinate system into a standard right angled triangle with the length of the perpendicular sides equal to unity (s. Fig. 3.4). This triangle is represented in a barycentric coordinate system $\mathbf{L_1L_2}$. The mapping relationship is given as [74]

$$\begin{aligned} u &= N_1u_1 + N_2u_2 + N_3u_3 \\ v &= N_1v_1 + N_2v_2 + N_3v_3 \end{aligned} \tag{3.41}$$

where N_r's are the first order shape functions given as [74]

$$N_1 = 1 - L_1 - L_2 \quad N_2 = L_1, \quad N_3 = L_2 \tag{3.42}$$

The displacements inside the triangle are interpolated from the nodal displacements $(\epsilon_r, \eta_r, \zeta_r)$ by using the same first order shape functions Eqs. (3.36), (3.42).

The derivatives of the displacements $(\epsilon_r, \eta_r, \zeta_r)$ with respect to the **uvw** coordinate system can be transformed to derivatives with respect to the $\mathbf{L_1L_2}$ coordinate system by using the chain rule.

$$\begin{aligned} \frac{\partial \epsilon}{\partial u} &= \frac{\partial \epsilon}{\partial L_1}\frac{\partial L_1}{\partial u} + \frac{\partial \epsilon}{\partial L_2}\frac{\partial L_2}{\partial u} \\ \frac{\partial \eta}{\partial v} &= \frac{\partial \eta}{\partial L_1}\frac{\partial L_1}{\partial v} + \frac{\partial \eta}{\partial L_2}\frac{\partial L_2}{\partial v} \end{aligned} \tag{3.43}$$

Using the relations given in Eqs. (3.41), (3.42), (3.36), (3.43) the integral inside the braces in Eq. (3.40) was transformed to an integral on the standard right angled triangle. The integral was evaluated approximately using three point gauss quadrature.

This integrated matrix is pre- and post-multiplied with the rotation matrix ($[\mathbf{R}]$) to get an elemental surface stiffness matrix for stretching $[\mathrm{K_{srf}}]$ [74].

$$E_{stretch} = \frac{1}{2}[\mathbf{d_{XYZ}}]^T[\mathrm{K_{srf}}][\mathbf{d_{XYZ}}] \tag{3.44}$$

The stretching surface energy of each elements can be added up to get the total surface energy by assembling the elemental surface stiffness matrices into a global surface stiffness matrix and taking its quadratic product with the displacements of all nodes [74].

$$\mathbf{E}_{stretch} = \frac{1}{2}[\mathbf{D_{XYZ}}]^T[\mathbf{K_{srf}}][\mathbf{D_{XYZ}}] \tag{3.45}$$

where $[\mathbf{K_{srf}}]$ is the global surface stiffness matrix and $[\mathbf{D_{XYZ}}]$ is the displacement vector in the **XYZ** coordinate system of all the nodes in the CT model mesh. The $[\mathbf{D_{XYZ}}]$ vector is same as the vector $[\mathbf{U}]$ in Eq. (3.16).

3.3.3 Stereo triangulation error modelling

For stereo-triangulated landmark locations, some coordinates may be more error prone than others. For stereo camera configurations with both the cameras looking in approximately the same direction (s. Fig. 3.5), it can be seen that as the point to be stereo triangulated moves away from the cameras, the region of uncertainity (shaded region in Fig. 3.5) will get elongated in the direction of the camera view (**z** direction in Fig. 3.5). Hence it can be asserted that for such configurations, the stereo triangulation of the points located far from the cameras is more error prone in the direction along the camera view in comparison to the other directions (s. Fig. 3.5).

The penalty term used in the deformable model formulation (Eq. (3.24)) penalizes the error in matching the landmarks along all the coordinate directions equally. By modelling the error in stereo triangulation, the penalty term could be changed to better suit the errors in estimation of the landmark locations.

The stereo triangulation error modelling is done for the frontal parallel camera system (image planes parallel and epipolar lines collinear) obtained after image rectification [75]. The image coordinates of a corresponding pair of points in the left and right camera coordinate systems is given as $[x_l, y_l]$ and $[x_r, y_r]$ respectively. The distribution of these points is assumed to be normal with means μ_l, μ_r and covariances Σ_l,Σ_r respectively. The coordinates of the three-dimensional triangulated point (in the right camera coordinate system) can be obtained using the ideal noise-free triangulation relations $P = [X_r, Y_r, Z_r]^T = f(l, r)$ given by [75]

$$\begin{aligned} X_r &= \frac{dx_r}{(x_l - x_r)} \\ Y_r &= \frac{dy_r}{(x_l - x_r)} \\ Z_r &= \frac{d}{(x_l - x_r)} \end{aligned} \tag{3.46}$$

where d is the distance between the projection centers of the left and right camera. The stereo-triangulated point can be approximated to be normally distributed with mean $\mu_P = f(\mu_l, \mu_r)$ and covariance [75]

$$\Sigma_P = J \begin{bmatrix} \Sigma_l & 0 \\ 0 & \Sigma_r \end{bmatrix} J^T \tag{3.47}$$

where J is the Jacobian of the stereo-triangulation relations f(l,r) given by [75]

$$J = \begin{bmatrix} \frac{\partial X_r}{\partial x_l} & \frac{\partial X_r}{\partial y_l} & \frac{\partial X_r}{\partial x_r} & \frac{\partial X_r}{\partial y_r} \\ \frac{\partial Y_r}{\partial x_l} & \frac{\partial Y_r}{\partial y_l} & \frac{\partial Y_r}{\partial x_r} & \frac{\partial Y_r}{\partial y_r} \\ \frac{\partial Z_r}{\partial x_l} & \frac{\partial Z_r}{\partial y_l} & \frac{\partial Z_r}{\partial x_r} & \frac{\partial Z_r}{\partial y_r} \end{bmatrix} \tag{3.48}$$

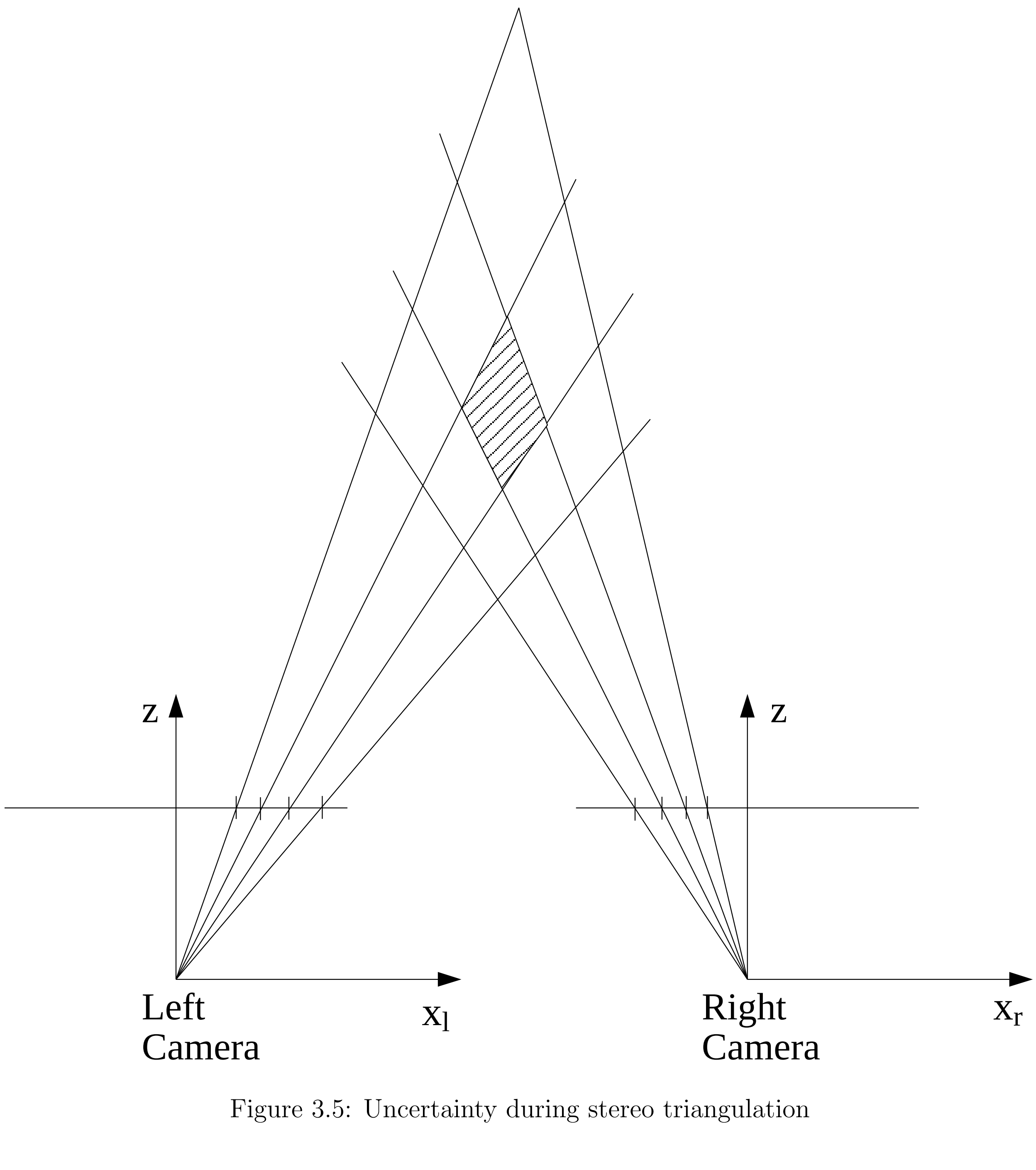

Figure 3.5: Uncertainty during stereo triangulation

The means of the image coordinates can be approximated as the point chosen during stereo matching and the covariances as identity matrices since their true values are not known [75]. Assuming the covariance of the image coordinates as identity matrices means that the image coordinates are assumed to be uncorrelated with a variance of one pixel in both the x and y directions. The mean and covariance of the stereo-triangulated three-dimensional points can be calculated by using the above mentioned values for the mean and covariance of the image coordinates in Eqns. (3.46) and (3.47)

For a multivariate normal distribution (like the three-dimensional location of the stereo-triangulated landmarks) the probability density function p is given by,

$$p(X) = \frac{1}{(2\pi)^{k/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(X-\mu)^T\Sigma^{-1}(X-\mu)} \tag{3.49}$$

$$\Rightarrow \quad \frac{1}{2}(X-\mu)^T\Sigma^{-1}(X-\mu) = \log\left(\frac{1}{p(2\pi)^{k/2}|\Sigma|^{1/2}}\right) \tag{3.50}$$

where k is the dimension , μ the mean and Σ the covariance of the variable X (k=3 when X is three dimensional coordinates). From Eq. (3.49) it is evident that the constant probability contours of the variable are along the contours of the function $(X-\mu)'\Sigma^{-1}(X-\mu)$.

μ is the location of the stereo triangulated landmark. Hence $(X-\mu)$ is the vector error in matching the landmark. Hence by pre and post multiplying the inverse of the covariance matrix with the error vector (quadratic product) a function approximately inversely propotional to the probability is obtained, ie, as the probability decreases the quadratic product increases and vice versa. This means that if we weight the vector error in matching the landmarks $(X-\mu)$ with the inverse of the co-variance matrix, we get a quantity that increases as the probability of the point in being the real landmark location decreases. The inverse covariance penalises the error in matching according to the probability of the point.

This insight can be used to improve the formulation by incorporating the inverse covariances of the stereo-triangulated point in the [**S**] matrix given in Eq.(3.24). When the matrix [**S**] is taken to be an identity matrix, the penalty is proportional to the square of the distance between the stereo-triangulated landmark and the corresponding point in the deformed model. Hence the constant penalty contours are spheres centered at the stereo-triangulated landmark. The constant penalty contours can be made identical to the constant probability contours around the stereo-triangulated landmark by defining the [**S**] matrix as

$$[\mathbf{S}] = \begin{bmatrix} [\Sigma_1]^{-1} & 0 & 0 & \cdots & 0 \\ 0 & [\Sigma_2]^{-1} & 0 & \cdots & 0 \\ \vdots & & \ddots & & \vdots \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & [\Sigma_l]^{-1} \end{bmatrix} \tag{3.51}$$

where Σ_r is the three dimensional covariance matrix of the $\mathrm{r^{th}}$ landmark.

Chapter 4

Results

Experimental data (stereo video sequences and CT model) was obtained from five individuals. Many ear deformation stereo video sequences (up to 20) were recorded for each of these individuals. Out of these video sequences, eight were selected for deformable model fitting and further acoustic analysis. Only video sequences with large ear deformations were selected. Each chosen video sequence was one deformation cycle, ie, the pinna goes from an upright state to a maximally bent state and back. Four deformation cycles (from three individuals) out of these eight chosen cycles, were used to establish the deforming methodology. Each deformation cycle was sampled at 9 or 10 representative instants. The complete shape of the pinna at these instants were computed by deforming the CT model to the landmark locations estimated by triangulation of the stereo pictures at those instants.

Performing the deformable model formulation requires computing the stiffness matrix $[\mathbf{K}]$ given in Eq. (3.16). Computing the stiffness matrix requires setting values for Lame's material constants λ and μ. For a deformation formulated as the minimum of a weighted sum of a deformation energy and a penalty function (Eq. (3.25)), the Young's modulus Y acts just like a scaling factor for the deformation energy term. Hence, for simplicity, the value for the Young's modulus was chosen to be equal to unity. Typical values of the Poisson's ratio of cartilage fall into the range from 0.13 to 0.45 [76]. The model fitting was performed for various values of Poisson's ratio in the suggested range and was found to change the shape of the deformed pinna by very little. Hence the Poisson's ratio was assigned a value approximately in the middle of the suggested range (0.3)

The deformation of the mesh for a particular set of landmark displacements can be obtained by solving Eq. (3.27). The particular value of θ to be used in the equation needs to be determined. In the penalty formulation (Eqs. (3.25) and (3.26)), θ acts as a parameter which controls the weighting between the deformation energy and the matching. Larger values of θ increase the importance of penalty over deformation energy and hence the displacements of the landmarks in the deformed shape approach their measured values $[\mathbf{D}]$. However, as the value of θ increases, the diminished importance of the deformation energy leads to

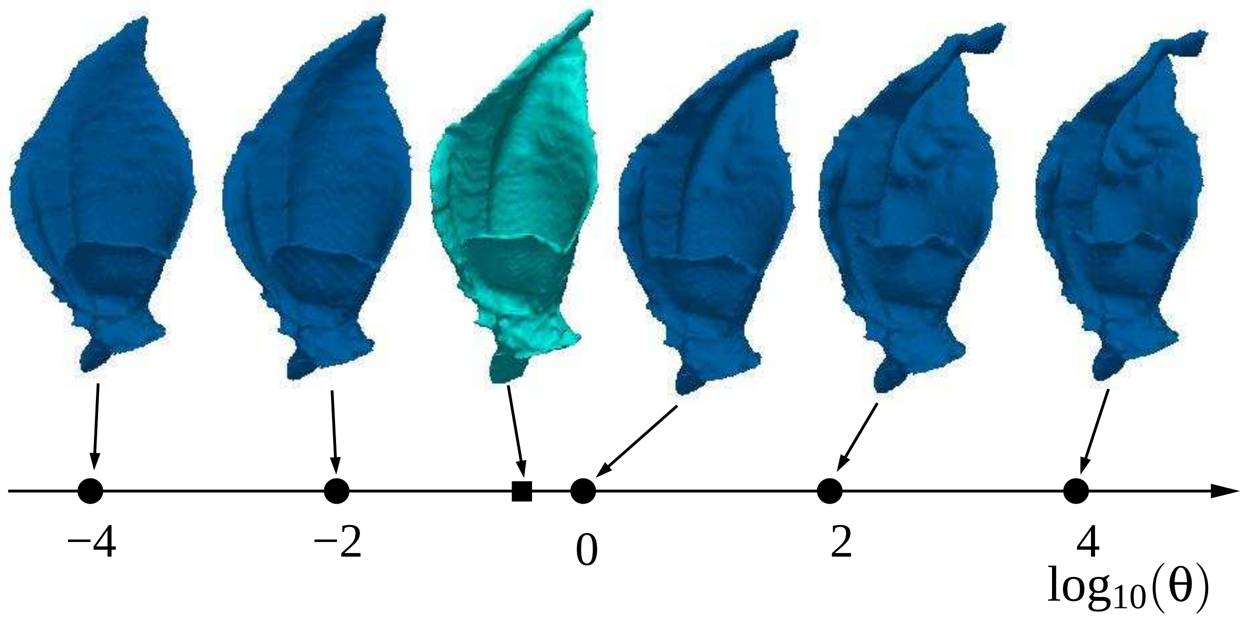

Figure 4.1: Effect of the weighting parameter (θ) on the fitting of the digital ear shape model: Example of one ear model being fitted to same set of landmarks using different values of θ (abscissa). The shape obtained with the θ that was chosen (s. text) for the analysis is highlighted.

the formation of artificial ridges and bumps near the landmarks (Fig. 4.1). The value of θ hence needs to be optimized to yield a good fit to the positions of the landmarks as well as life-like surfaces. The deformation quality measures (matching and smoothness) were used to optimize the value of θ. They are opposing measures; in the sense that while the matching measure improves by increasing θ, the smoothness measure improves by decreasing θ. A criterion on these measures can be used to obtain a good fit of the CT model to the landmarks.

4.1 Using mean curvature as the smoothness measure

The effectiveness of curvature as a smoothness measure is shown in Fig. 4.2. The insets **a** and **b** shows two deformations for the same CT model and landmark locations for different values of θ. The difference in the spatial distribution of curvature across the pinna surface is evident. This difference in the curvature distributions is also reflected in the histograms and is captured using the mean of the distribution (blue and red triangles on the curvature axis).

While deforming a CT model to the same landmark locations for different values of θ, systematic changes in the matching energy and mean curvature was observed (s. Fig. 4.3).

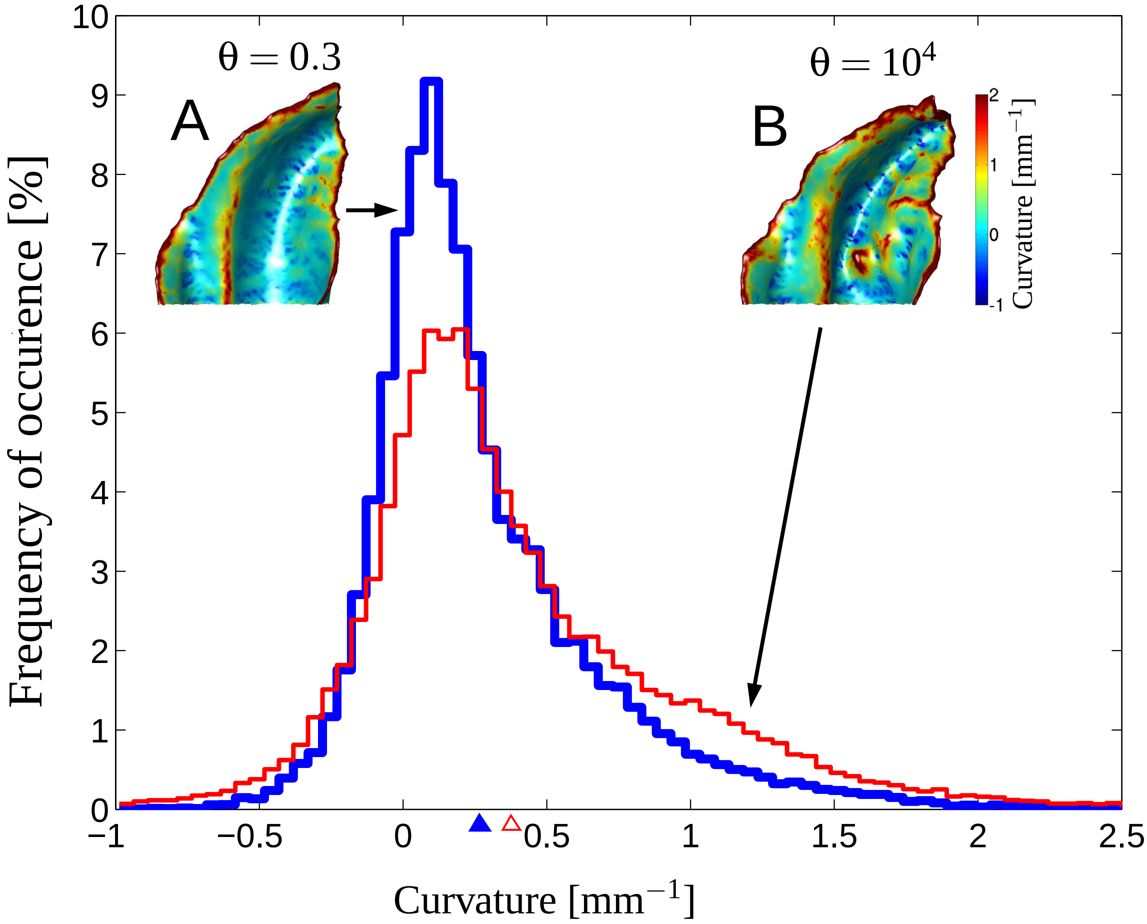

Figure 4.2: Distribution of curvature for an artifact-free pinna (blue) and a pinna with artificial folds (red). The pinna geometries from which these distributions were obtained are shown (insets 1 and 2) along with the value of the weighting parameter (θ, Eq. (3.26)) used.

The matching measure decreased continuously upon increasing θ and approached zero at higher values of θ (Fig. 4.3). When $\theta \rightarrow 0$ the model does not deform at all and the mean curvature of the deformed model is equal to the mean curvature of the CT model. As the value of θ was increased, the mean curvature of the deformed shape also increased and reached a maximum at higher values of θ (Fig. 4.3). All the deformations from the video sequences chosen to establish the deformation methods, showed qualitatively similar results.

For getting good life-like deformations, the matching with the landmark locations should be maximized while avoiding the appearance of artificial ridges and bumps. This means that the model should be deformed until the artificial ridges and bumps appear. Hence a criteria based on maximum tolerable smoothness measure was devised. The strategy was to increase θ until an acceptable value (threshold) for the mean curvature was reached. The smoothness measure threshold was placed at 25% of the span between the mean curvature of the CT

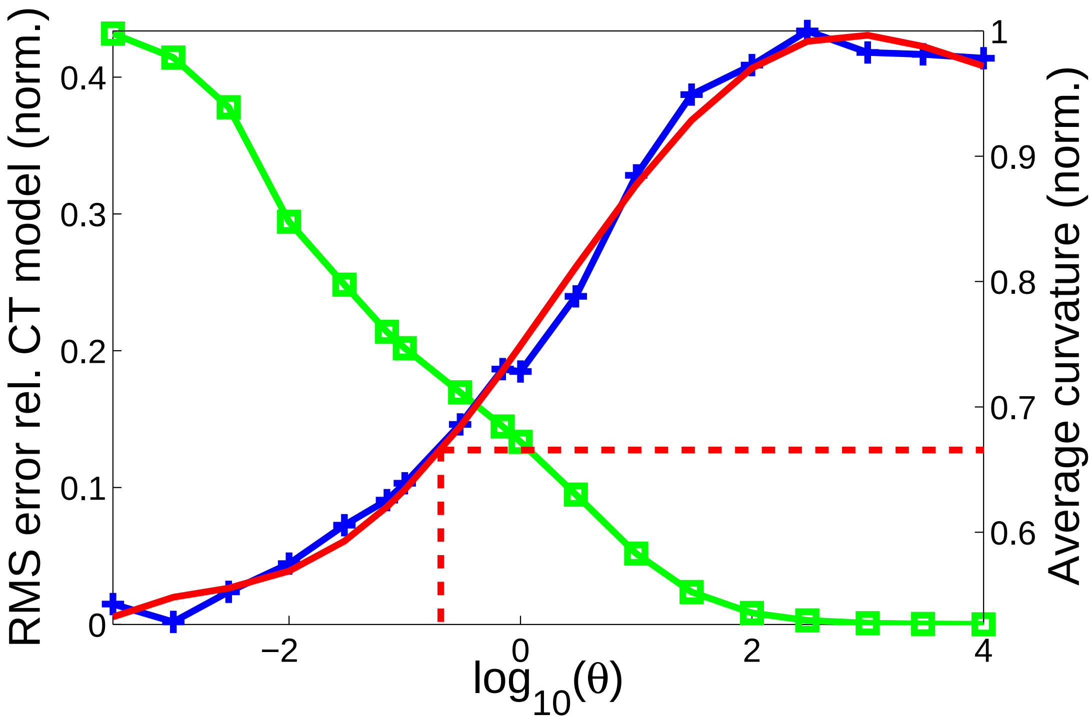

Figure 4.3: Criteria for selection of the weighting parameter (θ, s. Eq. (3.26)): root-mean-square error between landmark positions in the deformed model and the position estimates based on stereo-vision (squares) and average surface curvature (pluses). The interpolation curve (red) is a fifth-order polynomial fitted using the minimum least-squares criterion. The threshold for selecting the value of θ (dashed line) was set at 25% of the range between the average curvature of the undeformed CT model and the maximum curvature value observed.

model (undeformed shape) and the upper saturation values of the smoothness measure (s. Fig. 4.3). At this threshold, the matching measure was sufficiently small. The thresholding value (25%) was a conservative limit based on observations from the deformations in the video sequences chosen to establish the methods. It was observed that at this threshold, all the deformed shapes were artifact free but at higher thresholds, some of the shapes had non-life like ridges and bumps (s. Fig. 4.1). The value of θ was selected as the point along the θ-axis where a fifth-order polynomial fit through the smoothness measure values as a function of θ passed the threshold. The polynomial fit was used in this thresholding operation to prevent noisy cusps in the curve from having a significant effect on the chosen θ value. The deformed shapes obtained for the θ value chosen according to this method is shown in Fig. 4.4

Figure 4.4: Four video sequences obtained from three different individuals and their fitted 3D models. The first two rows from the top are from the same individual.

4.2 Using surface energy as the smoothness measure

The surface energy for the deformations was also calculated and found to be highly correlated with mean curvature. Both these smoothness measures showed similar changes upon changing the value of θ for the deformation. On increasing the value of θ both the smoothness measures increased sharply from a lower saturation threshold and reached a higher saturation threshold (s. Fig. 4.5) The linear correlation coefficient between the mean curvature and

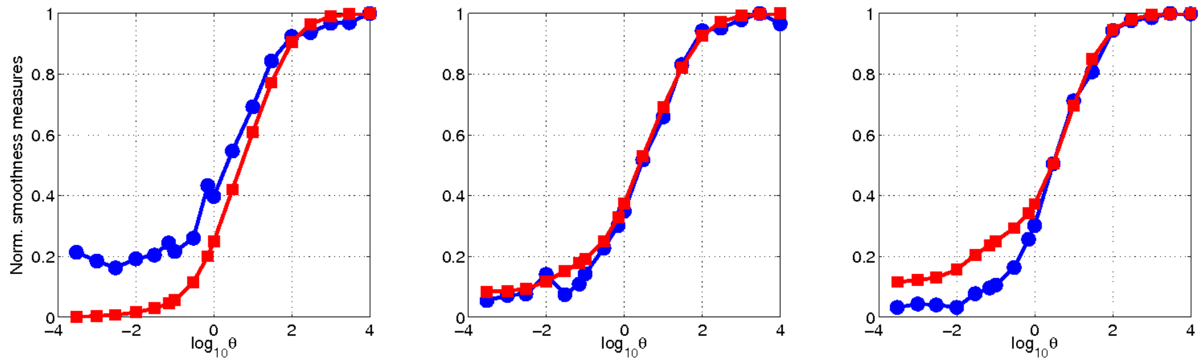

Figure 4.5: Dependence of smoothness measures with θ: Typical variation of curvature (blue circles) and surface energy (red squares) with θ. Curvature is normalized with respect to the mean curvature of the CT model and the upper saturation value. Surface energy was normalized with respect to the surface energy of the CT model (zero) and the upper saturation value

surface energy obtained for different values of θ for the deformations used for establishing the methods was always ≥ 0.97.

Surface energy was also used as the smoothness measure for obtaining the optimal value of the weighting parameter θ. Since the two measures were highly correlated (s. Fig. 4.5), same procedure and threshold (25% of the span between the smoothness measure of the CT model and the upper saturation value) was used for choosing the value of θ suitable for the deformation . Since there is no deformation in the CT model, its surface energy is zero. The deformation shapes obtained by using surface energy as the smoothness measure were very similar to those obtained by using mean curvature as the smoothness measure (s. Fig. 4.6).

The normalized mean curvature of the deformations whose optimal θ value was obtained by thresholding the surface energy smoothness measure was calculated. Even though the two smoothness measures were highly linear correlated, these measures for the same value of θ were not always the equal. The normalized mean curvature of deformations whose normalized surface energy was 0.25 was not always equal to 0.25 (s. Fig. 4.7).

The surface stiffness matrix $[\mathbf{K_{srf}}]$ is estimated from the CT model and hence can be precalculated. The surface energy for a deformation is calculated by taking the quadratic product

Figure 4.6: 3D pinna models fitted using surface energy as the smoothness measure: Four video sequences obtained from three different individuals (the first two from the top are from the same individual). Surface energy was used as the smoothness measure for obtaining the optimal θ value.

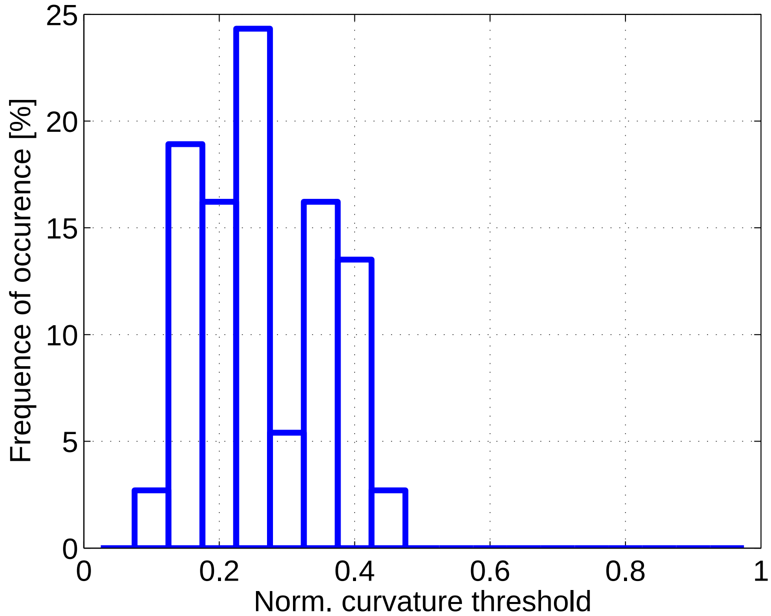

Figure 4.7: Comparison of normalized mean curvature and surface energy. Normalized mean curvature of the deformations whose optimal θ value was obtained by thresholding the normalized surface energy at 0.25

of the nodal displacement values with the surface stiffness matrix (Eq. (3.45)). Calculating the surface energy of a deformation took much less computational time (fraction of a second) when compared to that required for calculating the mean curvature (up to two minutes) on a dual quad core Intel Xeon X5450 3 GHz processor.

4.3 Stereo triangulation error modelling

Firstly, it was checked whether the errors in stereo triangulation is significant. For establishing this, the distribution of the stereo triangulated landmark location needs to be estimated. If the distribution is approximately spherical, there would be no need for including the stereo triangulation error model in the deformable model formulation.

The distribution of the chosen landmark positions in the left and right camera images is assumed to be a Gaussian centered at the chosen point with a variance of one pixel in both the coordinate directions (left to right and top to bottom in the image plane). A large number of points were drawn from this distribution and stereo triangulated to get the distribution of the stereo triangulated landmark (s. Fig. 4.8). The distribution is much wider in the $\mathbf{Z}$ direction (direction of the camera view) compared to the width along the $\mathbf{X}$ (left to right on the image plane) and $\mathbf{Y}$ directions (top to bottom on the image plane) (s. Fig. 4.8).

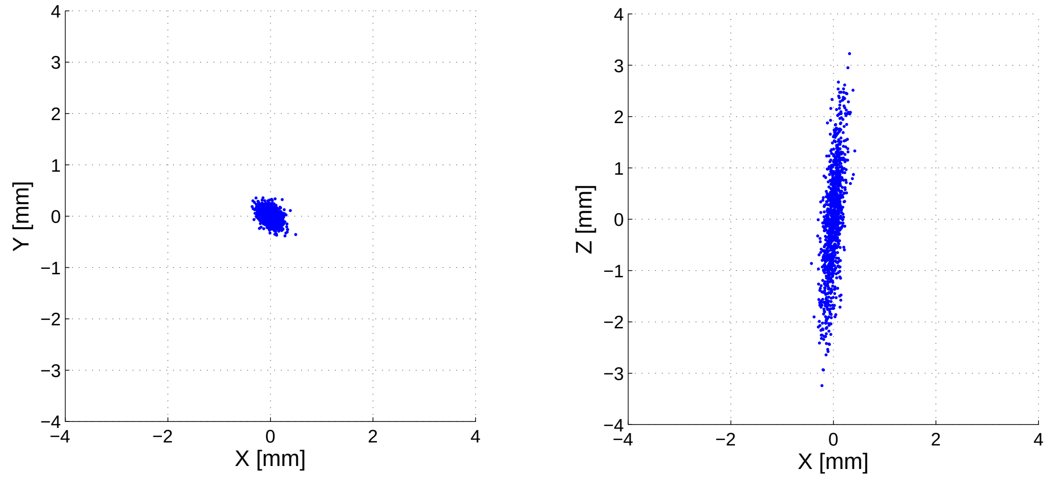

Figure 4.8: Distribution of the stereo triangulated landmark

For comparing the difference in the uncertainty of the **X**, **Y** and **Z** coordinates of the stereo-triangulated landmark, the covariances in those directions needs to be estimated. The covariance of the stereo-triangulated location of the landmarks was calculated (s. Eq. (3.47)) by approximating it to be multivariate normal distributed. For each landmark in a pair of matching stereo images, the covariance matrix was calculated according to Eq. (3.47). The covariance in each direction is normalized by dividing it by the norm of the covariance matrix. The covariances were normalized in order to make it independent of the units used for measuring the landmark location. The distribution of the normalized covariances in the **X**, **Y** and **Z** directions for different landmarks was estimated (s. Fig. 4.9). It can be observed from these distributions that the covariance in the **Z** direction is higher than that along the **X** and **Y** directions by more than an order of magnitude.

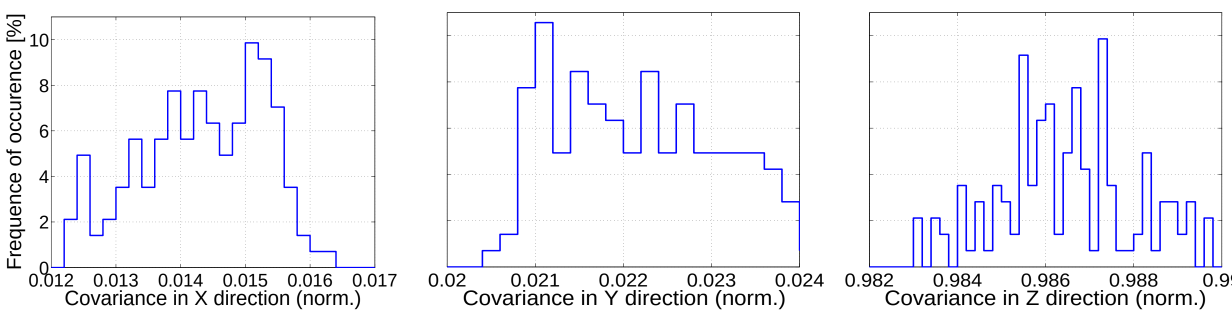

Figure 4.9: Distribution of the covariance of the stereo triangulated three-dimensional location of the landmarks in the **X**, **Y** and **Z** directions. The covariances were calculated for landmarks stereo triangulated from three instants in a video sequence. Contains covariances from a total of 142 landmarks.

The covariance estimates were used to penalize matching errors in the **X**, **Y** and **Z** directions differently by the changing the weighting matrix [**S**] in the deformable model formulation Eq. (3.25) according to Eq. (3.51). Each deformation was estimated for a range of values of the weighting parameter θ and the optimal one was obtained by thresholding the normalized smoothness measure (surface energy) at 0.25.

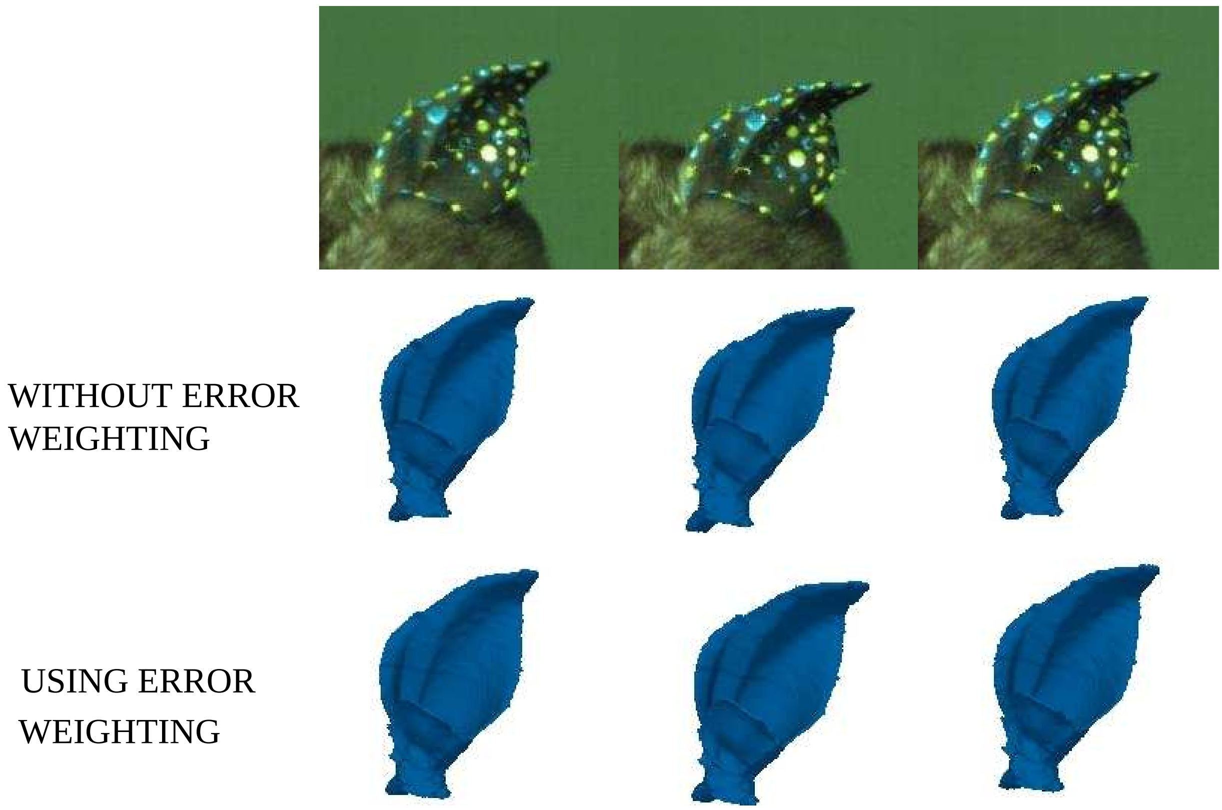

Figure 4.10: Comparison of deformations with and without using stereo error modelling: The video sequence instants are same as those used for calculating the variation in the covariances of the stereo triangulated landmarks (s. Fig. 4.9)

The deformation with and without using stereo error modelling looks very similar (s. Fig. 4.10). Even though, there is a much higher uncertainty in the Z-coordinate of the stereo triangulated landmarks (s. Figs. 4.8, 4.9), penalizing the coordinates differently does not seem to make much difference to the deformation (s. Fig. 4.10).

Chapter 5

Conclusions

The penalty method (Eq. (3.25)) gives good control in imposing displacement constraints from noisy data (s. Fig. 4.1). By increasing the weighting parameter θ, the amount by which the landmarks will conform to the stereo triangulated locations can be adjusted. Imposing the penalties is equivalent to connecting a zero length spring between the stereo triangulated landmark and the landmark position in the CT model. The non-scaled weighting parameter Θ is the stiffness of the spring. By increasing θ (Θ increases proportionately on increasing θ), the force exerted by the spring for an error in matching the landmark location increases. This forces the model to move closer to the landmark locations. But due to the error in the stereo triangulated landmark location, as the model moves very close to the landmarks, non-life like artifacts start to appear (s. Fig 4.1). For very large values of θ (usually $>10^4$), the spring stiffness becomes much higher than the stiffness of the ear and the linear equations become ill conditioned. This is because the stiffer springs have a rigid body deformation component so large that it overwhelms other deformation components needed for accurate computation of strains [43].

The smoothness measures are capable of quantifying the non-life like bumpy artifacts in the deformed shapes (s. Figs. 4.2, 4.5). The smoothness measure (both mean curvature and surface energy) increases continuously with increasing the value of θ (s. Figs. 4.3, 4.5) and reaches a saturation value. The smoothness measure value for deformed shapes with such artifacts (at higher values of θ) are much higher than those for deformed shapes without such artifacts.

The tolerable absolute smoothness measure needs to be adaptive with the amount of deformation. For larger deformations, the tolerable absolute smoothness measure needs to be higher since the smoothness measure of the optimal deformation will be larger due to the increased bending of the ear. It was found that the saturation value of the smoothness measure on increasing the θ had good linear correlation with amount of deformation in the displacements at the landmarks (s. Fig. 5.1). Hence by choosing the tolerable smoothness measure in relation with the absolute smoothness measure saturation value (at 25 % of the

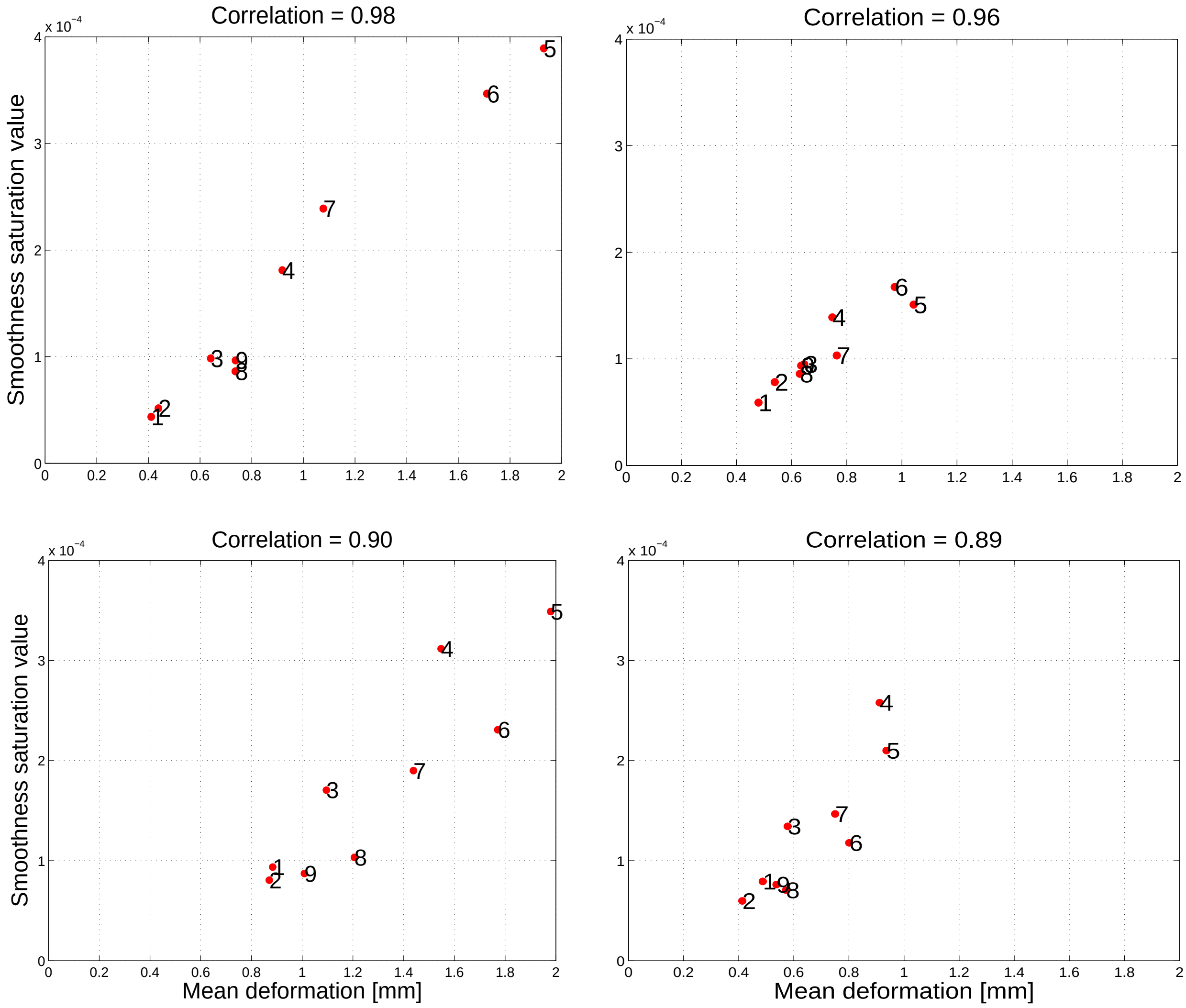

Figure 5.1: Dependence of smoothness measure saturation value with deformation. The landmarks in the CT model is rigid registered with the stereo triangulated landmarks. The mean of the distance between the rigid registered and stereo-triangulated landmarks is taken to be the measure of the deformation in the stereo-triangulated landmarks. The numbers give the position of the deformation in the sampled cycle. The deformation cycles were sampled such that in the images from 1-5 the ear, starting from an upright position, goes to a maximally bend position and in the images from 5-9, the ear goes back to a predominantly upright position

span between the smoothness measure value of the CT model and the absolute smoothness measure saturation value), the tolerable absolute smoothness measure is being made adaptive with the amount of deformation in the displacement at the landmarks. But the adaptive threshold (25 %) was based on experimental observations that; at this value of smoothness measure, the deformed shapes were all artifact free and the landmarks in the deformed model were very close to the stereo triangulated location of the landmarks. The possible values of θ spans a huge range - from 0 to infinity. Only a significant change (around 2-3 times) in the value of θ would make a noticeable difference in the deformed shape of the ear. Hence the chosen value of θ just defines a range of values (around 2 times greater or lesser than the chosen value) around which the deformed shape looks life-like.

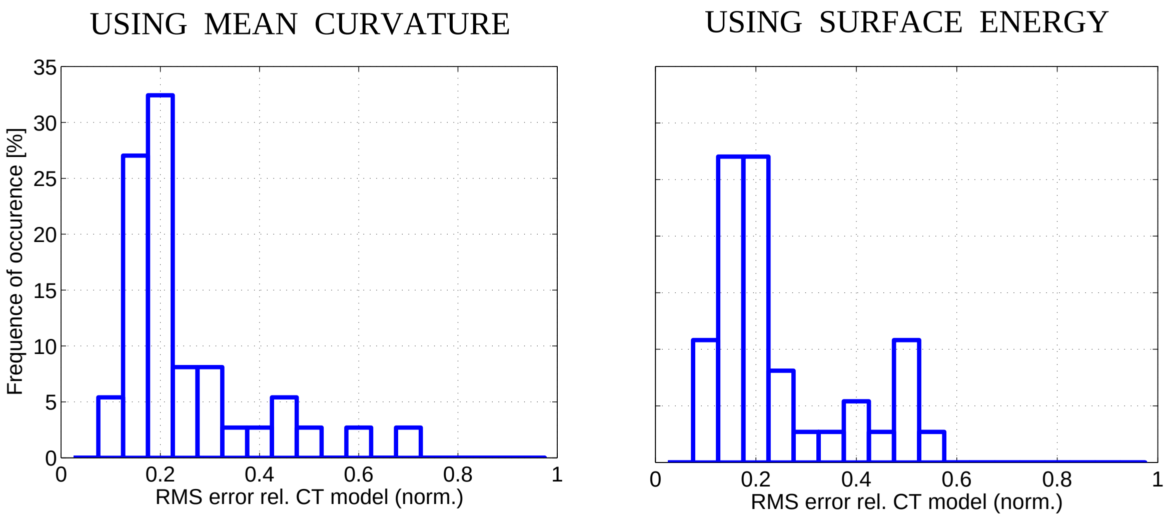

Figure 5.2: Distribution of the relative error in matching the landmarks for the deformations chosen using mean curvature and surface energy. Errors are represented as the ratio of the RMS error in the deformed shape to that of the CT model. The distributions contain a total of 37 deformations

The error in matching the location of the stereo triangulated landmarks for the deformations chosen using mean curvature and surface energy as smoothness measure was analyzed (s. Figs. 5.2, 5.3). It was observed that the distribution of the relative error in matching the stereo triangulated landmark locations in both the cases were very similar. The matching errors were predominantly around 20 % of the matching error of the undeformed CT model for both the methods. There were some outliers (around 20 % which means around 7-8 shapes) whose matching error relative to the CT model is $\geq$40 %. The relative error of the outliers were a little higher when using mean curvature and the number of outliers were marginally higher when using surface energy. It was observed that these shapes were the ones during the beginning or ending of the deformation cycle, ie, the when the ear was mostly upright. This means that the relative error in matching the landmarks were high when the magnitude of the deformation was small. For smaller deformations the matching error in the

CT model is also small. Also, the relative matching error in the deformed shapes is inversely proportional to the matching error in the CT model. Hence, the inverse relationship with the matching errors in the CT model and the smaller matching errors in the CT model for small deformations, could be the reason for high relative errors in these deformations. The distributions of the absolute RMS errors (s. Fig. 5.3) also look very similar with the

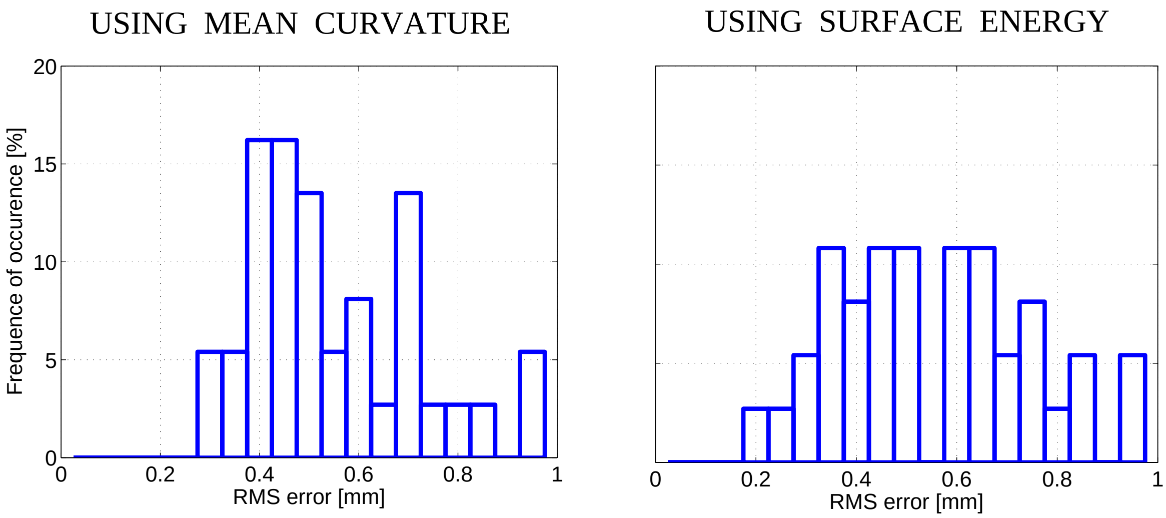

Figure 5.3: Distribution of the absolute RMS error in matching the landmarks for the deformations chosen using mean curvature and surface energy. The distributions contain a total of 37 deformations

absolute RMS error always less than 1 mm. On visual inspection, the deformed shapes obtained using mean curvature and smoothness energy as the smoothness measures were very similar and matched the shape of the ear during the deformation cycle very closely (s. Figs. 4.4, 4.6). Since the real shape of the ear during the ear deformations is unknown, there is no quantitative method to objectively identify which of the two smoothness measures is giving a deformed shape closer to the actual shape. Since there is no reason to use one smoothness measure over the other and the results from using both the measures look very similar, the more convenient one (surface energy) could be used.

The difference in the location of landmarks in the deformed shape from the stereo triangulated landmarks (s. Fig. 5.3) can be assumed to be higher than that between the fitted shape and the actual shape of the ear. Under such an assumption, it can be stated that the overall error in the deformation is less than 1 mm (s. Fig. 5.3). Since these models were to be used for numerical acoustic analysis, only the relative size of the errors in comparison to the wavelength of the simulated sound waves matter. The highest sound frequency used in the simulations were 80 kHz which has the smallest wavelength - 4.3 mm. Since the error in the deformed shapes (< 1 mm) was less than 25 % of the smallest wavelength (4.3 mm) used, the beampattern estimates from these deformed shapes are believed to be fairly accurate.

Usually in biological systems, the behavioral patterns are not always exactly reproduced. For example, there is a lot of variability in the sizes, shapes and behavior among individuals of the same species. Hence the underlying physical principles of biological systems are usually not heavily dependent of the exact reproduction of a specific size, shape or behavior [77]. This means that a very high degree of accuracy in the deformed shape may not be needed to observe a consistent effect in the acoustics.

The error in stereo triangulation along the different coordinate directions was highly skewed. A maximum error of around 0.1 , 0.1 and 1.5 mm would be made in the **X**, **Y** and **Z** directions respectively if an error of one pixel was made in picking the matching points in the stereo images. The **X** direction is from left to right in the image plane, **Y** direction from top to bottom in the image plane and **Z** direction along the camera view. Hence intuitively, the method of including the stereo triangulation error in the deformable model formulation should improve the model fitting. But the shapes obtained when using stereo triangulation errors in the deformable model formulation look very similar to those obtained without using stereo triangulation errors (s. Fig. 4.10). Again, since the real shape of the ear during the ear deformations is unavailable, there is no way to evaluate which of the two methods, using or without using stereo triangulation error in the deformable model formulation, gives a more accurate deformed shape. In the absence of a clear reason to use the slightly more complicated stereo triangulation error modelling method, the simpler methods (without using any stereo triangulation error modelling) were favored.

Chapter 6

Acoustical Analysis

6.1 Methods and Measures

Previously developed finite-element based numerical acoustic methods [78] were used to obtain the beampatterns of the deformed pinna shapes. The beampatterns of the shapes were obtained for sound frequencies from 60 to 80 kHz at 5 kHz intervals (s. Figs. 6.3, 6.5) covering the observed frequency band of the individuals' biosonar (60 to 77 kHz). The important features in beampatterns - the global maximum (mainlobe) and other local maxima (sidelobes) were extracted from a spherical map representation of the beampattern using image processing techniques. In order to quantify the observed effects, the following two measures: (i) a "total sensitivity" of main- and sidelobes and (ii) a "beam overlap" between the lobes of neighboring frequencies were defined.

In both cases, the mainlobe was defined as the solid angle (area on the unit sphere) inside a -3 dB beam-gain contour below to the global maximum that included the global maximum (Fig. 6.1).

For the computation of the total sensitivity, the entire solid angle outside of the mainlobe was considered as potentially belonging to the sidelobe. Areas where the beam gain fell below -15 dB were excluded in order to eliminate low sensitivity areas, e.g., on the back of the ear. It was verified that the contributions from these areas of low sensitivity did not depend significantly on pinna shape.

The total sensitivities of the main- and sidelobes were calculated as

$$\begin{aligned} T_{\text{mainlobe}} &= \oint_{\text{ML}} D(\theta, \phi, f) \sin \phi d\phi d\theta \\ T_{\text{sidelobe}} &= \oint_{\text{SL}} D(\theta, \phi, f) \sin \phi d\phi d\theta \end{aligned} \tag{6.1}$$

where ML and SL are the solid angles of the mainlobe and the sidelobes respectively, θ is

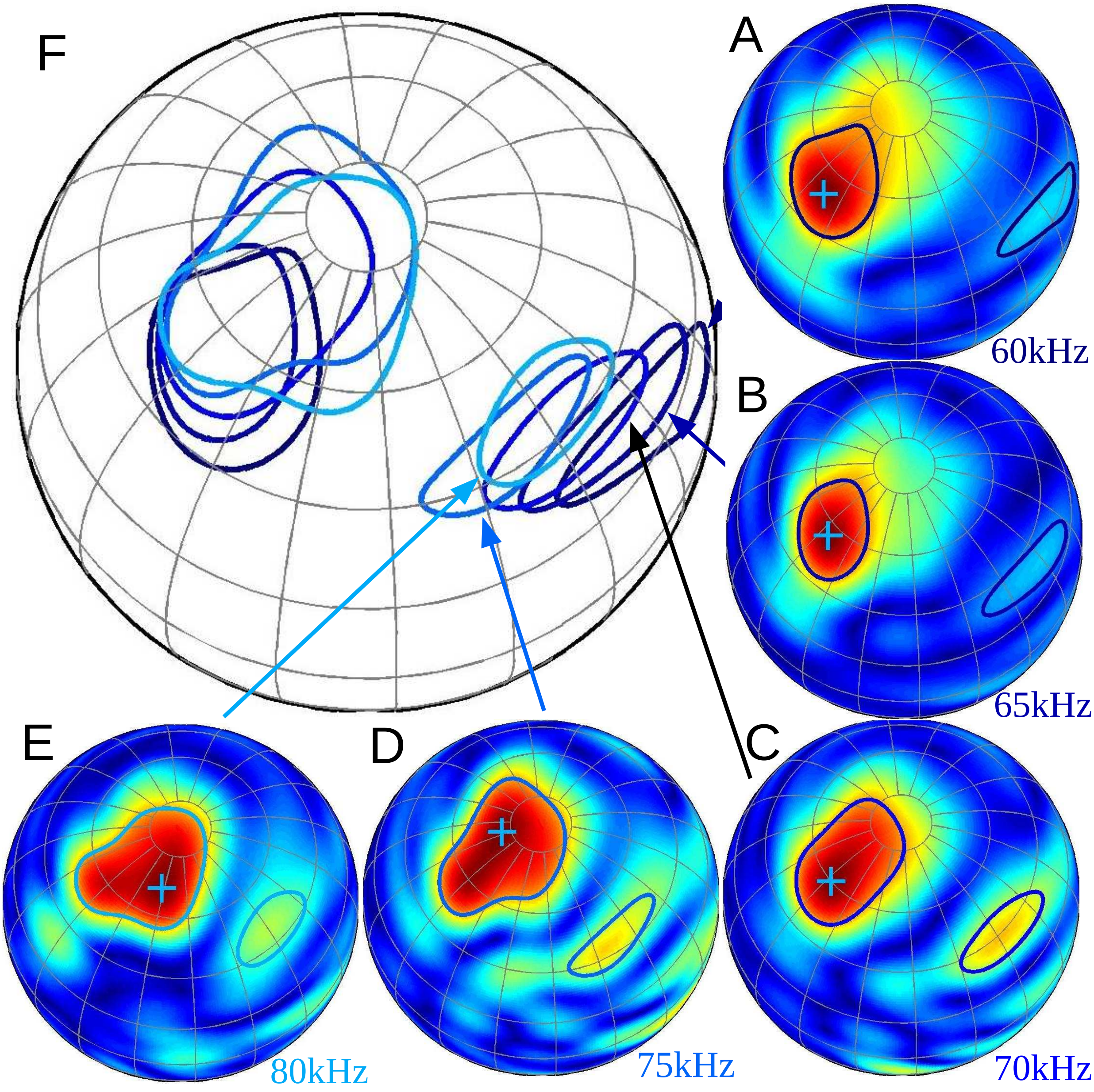

Figure 6.1: Representation of beampatterns by -3 dB contours. a,b,c,d,e) Beampatterns of a pinna geometry at different frequencies with the contours of the mainlobes and sidelobes superimposed. Crosses mark the location of the global gain maximum. f) Contours of different frequencies are superimposed for comparison of direction, size and shape of the lobes. The frequency of the lobes is color coded such that the highest frequency is assigned the lightest color.

the azimuth, and ϕ the elevation angle.

In order to compare the changes in main- and sidelobes with frequency, only beampatterns with strong sidelobes were selected. The strength of the sidelobes was determined by looking at their total sensitivity. Strong sidelobes were defined as having an average total sensitivity across frequency three times greater than that of the mainlobe.

To characterize the changes in sidelobe over frequency, the sidelobe with the largest maximum gain in each beampattern was selected. The solid angle inside a -3 dB sensitivity contour (relative to the sidelobe's maximum gain) was used to represent the shape of the sidelobe (Fig. 6.1).

The overlap between the solid angles occupied by a lobe at different frequencies was taken as a measure of similarity: Overlap was defined as the ratio of the intersection to the union of the solid angles of the two lobes. Beam overlaps were calculated for lobes of adjacent frequency values. Since the beampatterns of each shape were obtained for a total of five frequencies (60 kHz to 80 kHz at 5 kHz intervals), there were four beam overlap values to characterize the change in each mainlobe or sidelobe. The average of these four values was then taken as a single measure of the change in these lobes across frequency.

6.2 Results

Systematic changes in the beampatterns were seen as the pinnae bent and resulted in qualitative differences between the upright and bent pinna geometries: For the upright position of the pinna, the sidelobes of the beampattern were weak compared to the mainlobe (as much as -15 dB difference between maxima). As the pinna deformed, the sensitivity in the sidelobes increased significantly: At maximum, the sensitivities of bent and upright pinnae differed by a factor of about five (s. Fig. 6.6) Some of the sidelobes did even surpass the respective mainlobes to become the global sensitivity maximum. When the pinnae returned from their bent state to an upright position at the end of the deformation cycle, the beampatterns also reverted to their previous configuration (s. Figs. 6.2, 6.3, 6.4, 6.5, 6.6). Regardless of bending state, the beampatterns always contained a strong mainlobe (s. Figs. 6.2 6.3, 6.4, 6.5) with a total sensitivity that remained approximately constant throughout the deformation cycle (s. Fig. 6.6). In general, the direction, width, and shape of the mainlobes changed little across frequency (s. Fig. 6.7) whereas for majority of the sidelobes these properties depended strongly on frequency. Furthermore, the sidelobe locations overlapped much less across frequency than those of the mainlobes (Fig. 6.7, $p < 0.1\%$, two-sample Kolmogorov-Smirnov test). This frequency-dependence manifested itself either in orderly patterns of change, e.g., in lobe direction (Fig. 6.7B) or in less easily comprehensible changes that affected either lobe direction or shape (Fig. 6.7C).

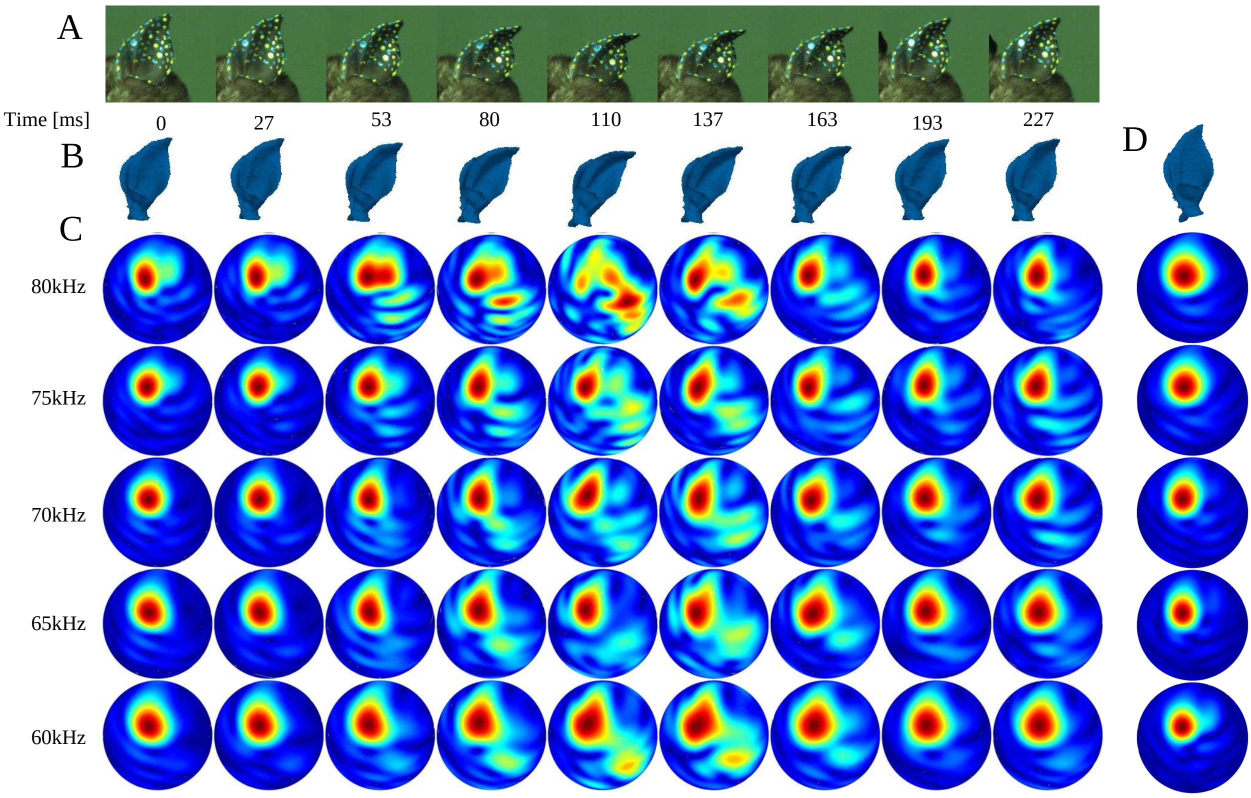

Figure 6.2: Numerical analysis of representative stages in a deformation cycle (first of the four most complete sequences analyzed): a) Video images of the pinna with landmarks from one of the stereo cameras. The recording time (relative to the first frame) is indicated below each frame, b) Renderings of complete digital models of the pinna that were matched to the location of the landmarks of the video frames in the respective columns, c) Numerical beampattern estimates obtained from the complete digital models. Each row shows beampatterns for a different frequency, d) CT model and its beampattern estimates at different frequencies.

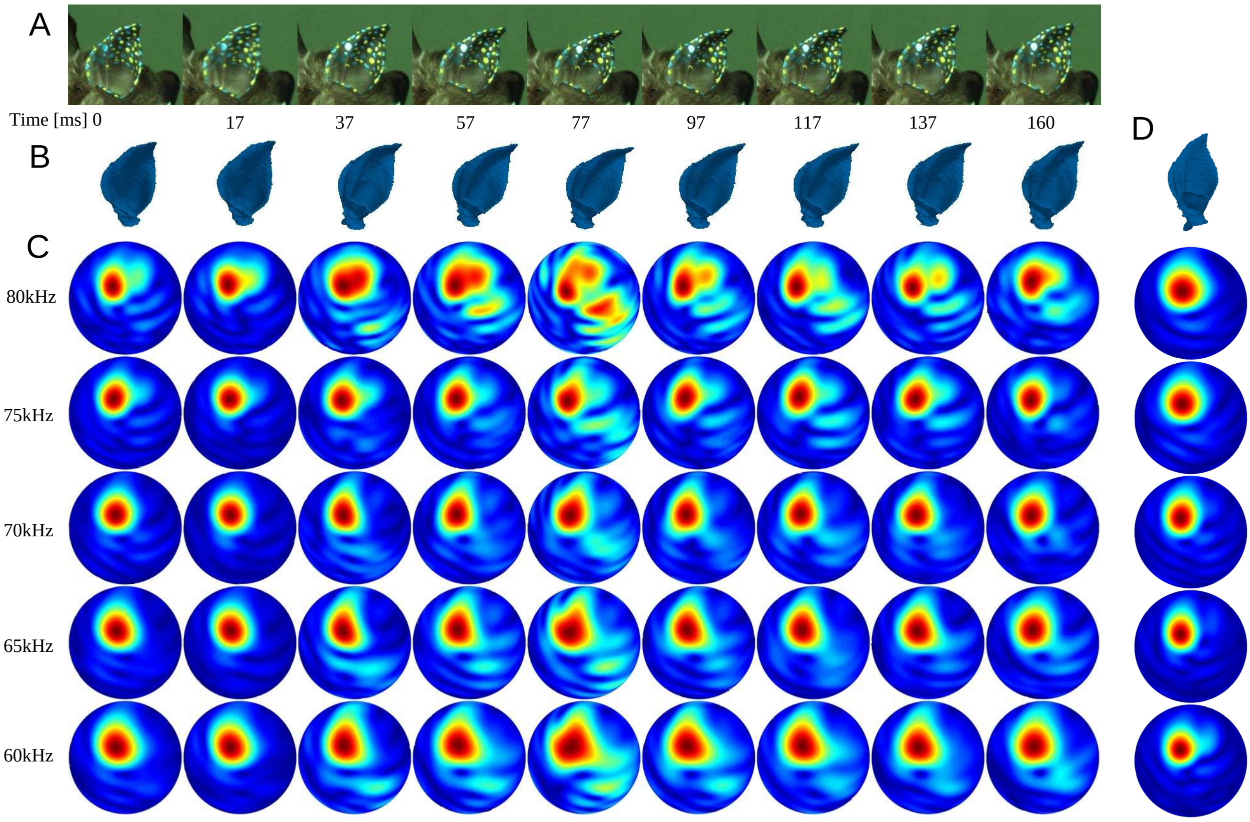

Figure 6.3: Numerical analysis of representative stages in a deformation cycle (second of the four most complete sequences analyzed): a) Video images of the pinna with landmarks from one of the stereo cameras. The recording time (relative to the first frame) is indicated below each frame, b) Renderings of complete digital models of the pinna that were matched to the location of the landmarks of the video frames in the respective columns, c) Numerical beampattern estimates obtained from the complete digital models. Each row shows beampatterns for a different frequency, d) CT model and its beampattern estimates at different frequencies.

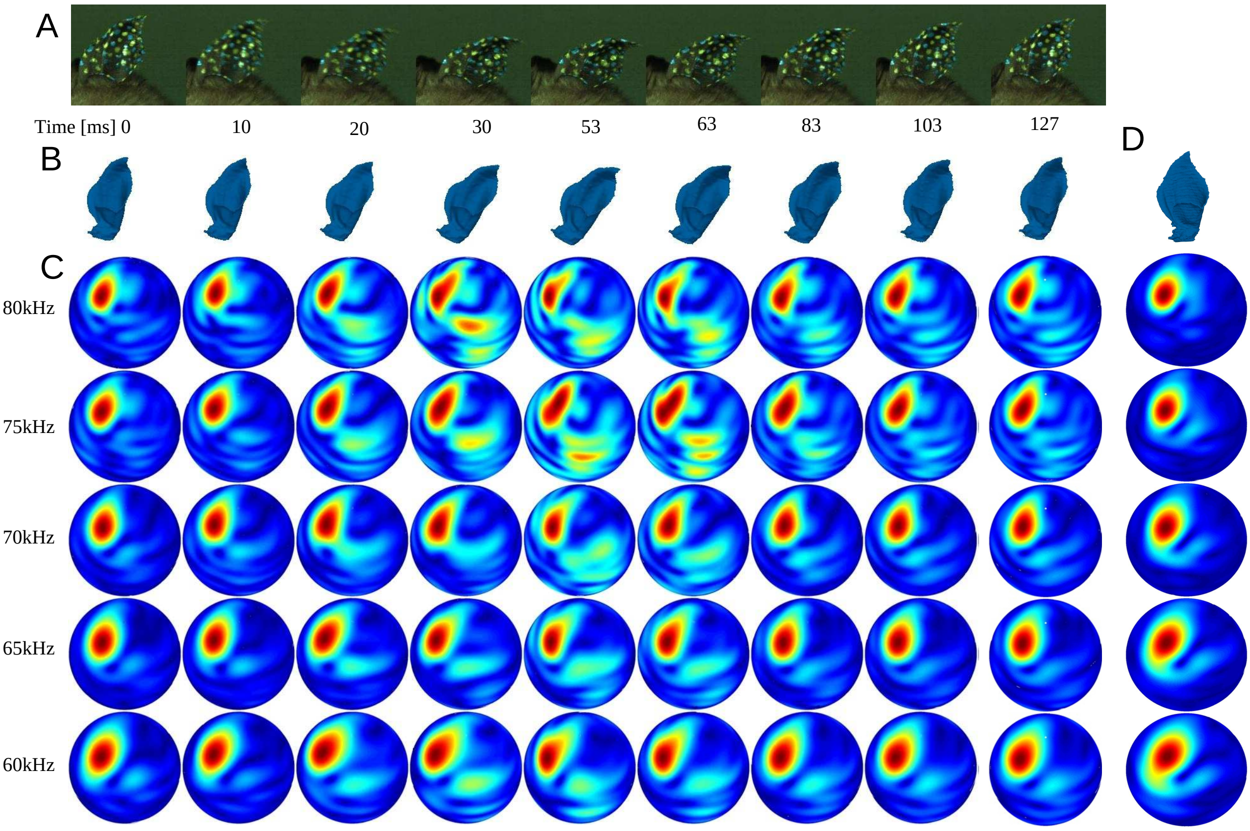

Figure 6.4: Numerical analysis of representative stages in a deformation cycle (third of the four most complete sequences analyzed): a) Video images of the pinna with landmarks from one of the stereo cameras. The recording time (relative to the first frame) is indicated below each frame, b) Renderings of complete digital models of the pinna that were matched to the location of the landmarks of the video frames in the respective columns, c) Numerical beampattern estimates obtained from the complete digital models. Each row shows beampatterns for a different frequency, d) CT model and its beampattern estimates at different frequencies.

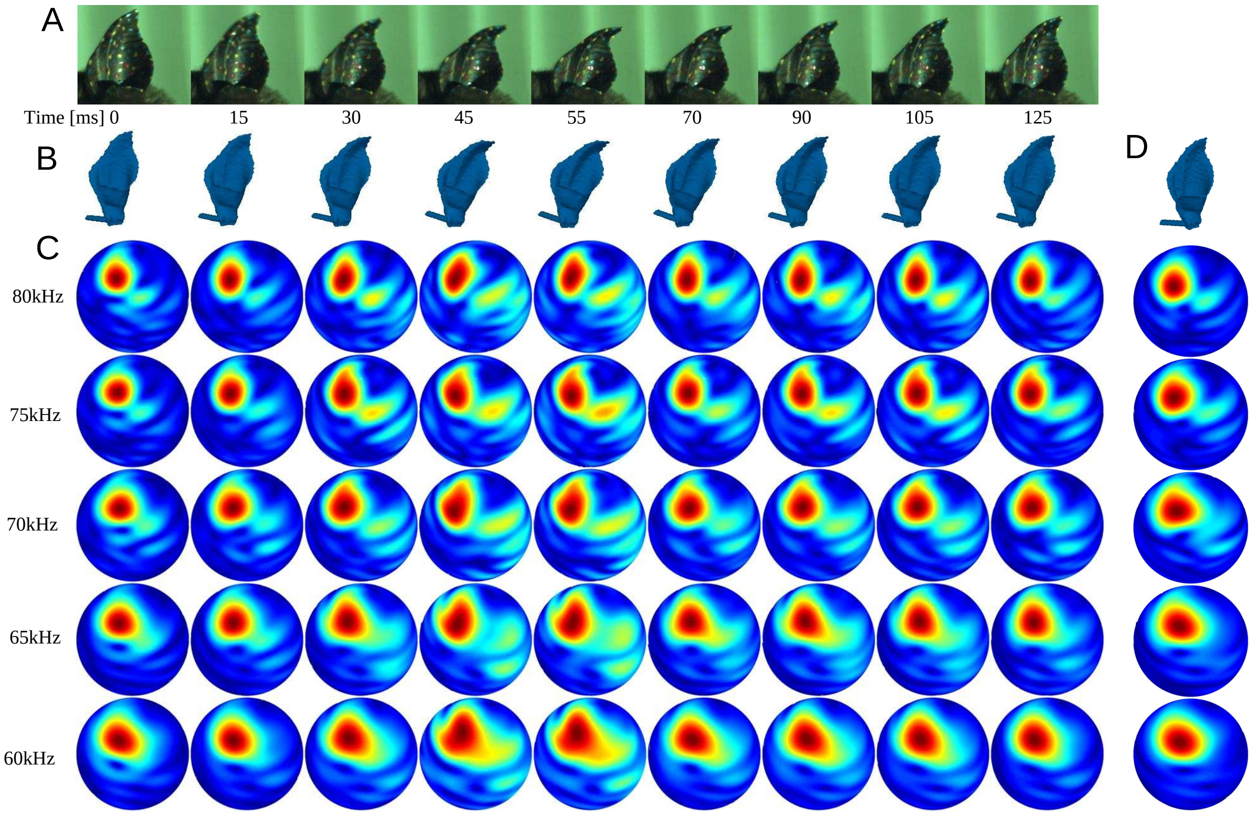

Figure 6.5: Numerical analysis of representative stages in a deformation cycle (fourth of the four most complete sequences analyzed): a) Video images of the pinna with landmarks from one of the stereo cameras. The recording time (relative to the first frame) is indicated below each frame, b) Renderings of complete digital models of the pinna that were matched to the location of the landmarks of the video frames in the respective columns, c) Numerical beampattern estimates obtained from the complete digital models. Each row shows beampatterns for a different frequency, d) CT model and its beampattern estimates at different frequencies.

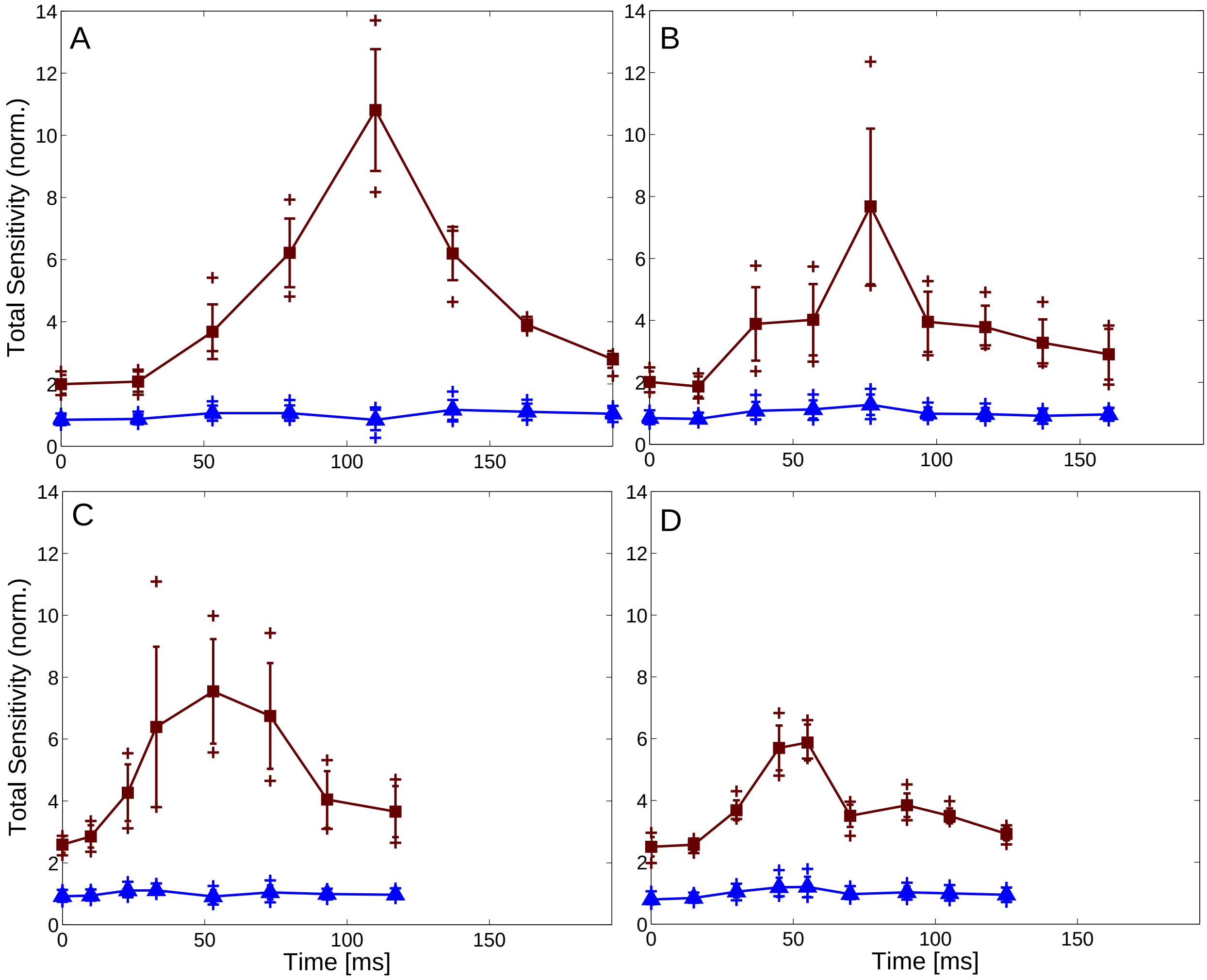

Figure 6.6: Change in total sensitivity of main- (■) and sidelobe (▲) with pinna deformation. Shown for each sequence are: total sensitivity averages (mainlobes: ▲, sidelobes: ■), standard deviation (error bars), extreme values (+) over frequency (60 to 80 kHz in 5 kHz intervals).

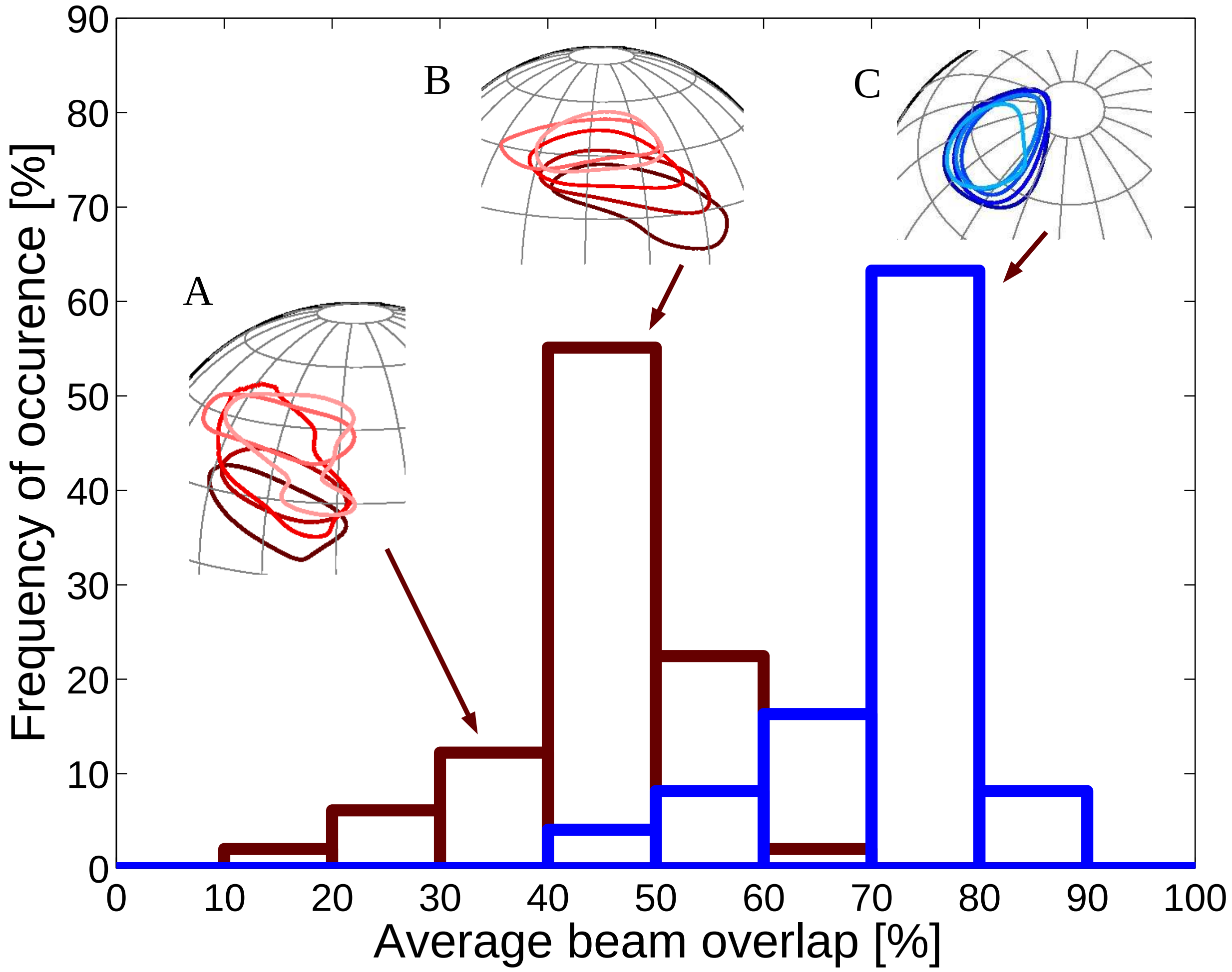

Figure 6.7: Comparison of the frequency dependence of mainlobe and sidelobe: Histograms of the average beam overlap of the mainlobes (blue) and sidelobes (red) in beampatterns for adjacent frequencies. Insets A,B,C show example lobe areas as a function of frequency where frequency is color coded such that highest frequency is the lightest.

6.3 Conclusions

Different beampatterns are better suited for some estimation tasks than for others. For example, beampatterns that concentrate their sensitivity in one and the same direction for all frequencies are well suited for detection of weak signals coming from this direction. They also favor the identification of sound sources from the spectral signature of echoes received from this direction [79]. In this case, the concentration of sensitivity maximizes the signal-to-noise ratio which helps any estimation tasks, including detection and identification. Furthermore, orienting the mainlobes across all frequencies in the same direction results in a flat transfer function for signals from sources in this direction. This facilitates spectral identification, because the spectral signature of the target is not perturbed during echo reception. The beampatterns of all the upright pinnae from the behavioral experiments (s. Fig. 6.8F-J) as well as those of the static shapes from other cf-fm bat species analyzed before (s. Fig. 6.8A-E), fall into this category. Accurate monaural target localization imposes requirements on the beampattern that may conflict with detection and identification [79]: It is favored by beampatterns with frequency-dependent change in the direction of the lobes that encode information on target direction into the echo's spectral profile [80]. Accurate direction finding requires large changes in the beam gain with direction and frequency (Fig. 6.7A 6.7C), but not necessarily in an orderly or easily comprehensible fashion. The beampattern estimates obtained for the pinnae of fm bat species (examples in Fig. 6.8P-T) fall into this category. The beampatterns estimates for the bent pinnae in the behavioral experiments also show the same qualitative features (strong sidelobes with frequency dependent positions and shapes, s. Fig. 6.8K-O).

Our findings suggest that horseshoe bats could use non-rigid deformations of their ears as a mechanism to switch between beampatterns with different qualities and hence different functional roles. How these different patterns could be integrated into the animals' biosonar system is still an open question: Even though attempts to correlate biosonar emissions and ear movements have been made in early studies already [15, 89], it remains to be determined what the time-relationships between ear deformations, received echoes, and informational needs of the animals are. Since the time durations of the deformations and the emitted pulse are fairly similar, transitions between these ear shapes with different acoustic functions could happen either between successive echoes or even within a single echo. The latter would give the monaural hearing in these animals a pronounced non-stationary quality.

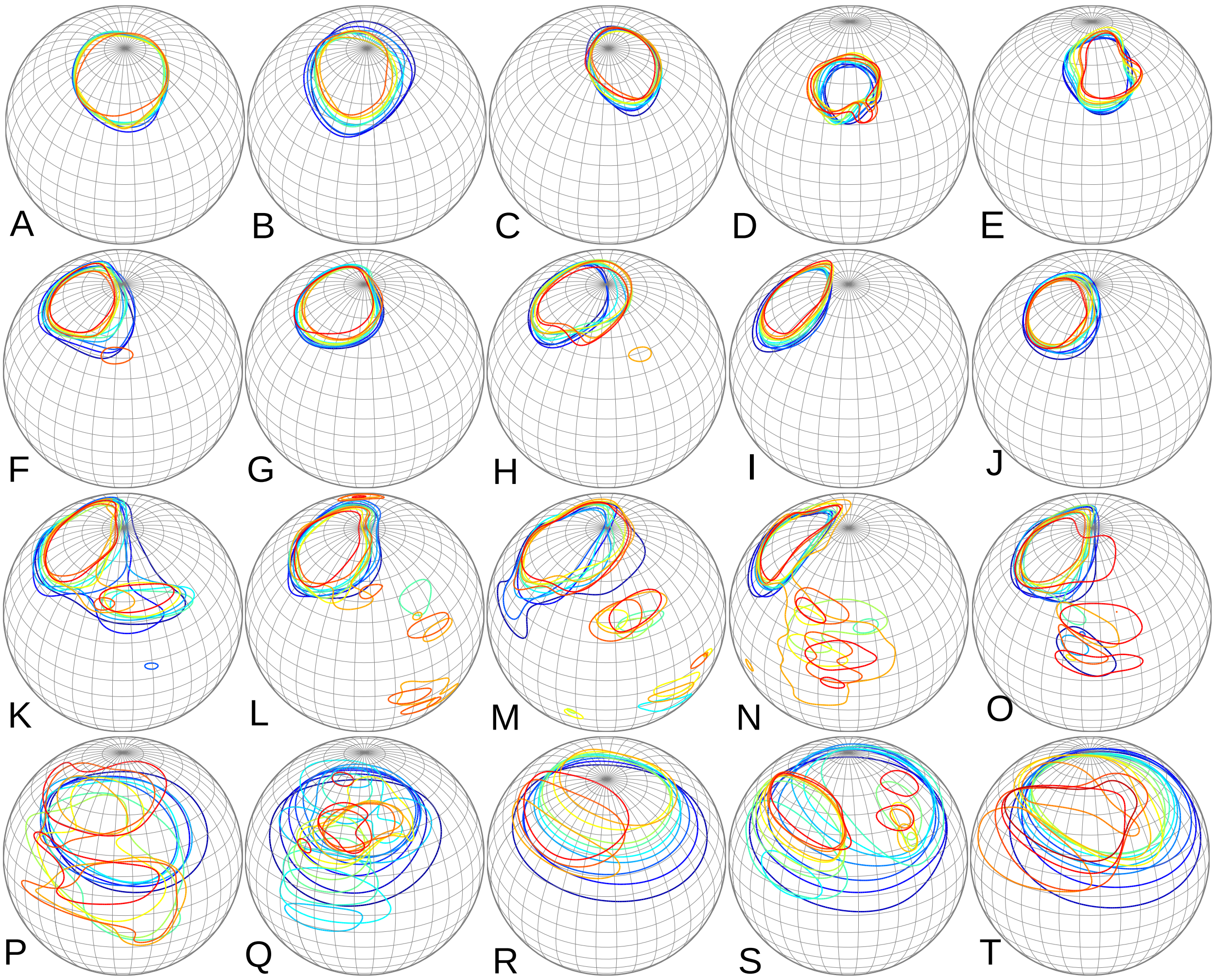

Figure 6.8: Comparison of beampatterns of the upright, bent pinnae of horseshoe bats and the static ear shapes from other cf-fm and fm bat species. Beampatterns of static pinnae from different cf-fm bat species (top row), in-vivo pinna shapes from greater horseshoe bats in an upright (second row) and a bent pose (third row), and fm bats (fourth row): a) Stoliczka's trident bat (*Aselliscus stoliczkanus*, 102 to 120 kHz [81] in steps of 2 kHz), b) Pearson's horseshoe bat (*Rhinolophus pearsonii*, 44 to 62 kHz [82] in steps of 2 kHz), c) Least horseshoe bat (*Rhinolophus pusillus*, 90 to 110 kHz [83] in steps of 2 kHz), d) Pomona roundleaf bat (*Hipposideros pomona*, 98 to 126 kHz [84] in steps of 2 kHz), e) Pratt's roundleaf bat (*Hipposideros pratti*, 50 to 70 kHz [85] in steps of 2 kHz), f,g,h,i,j) Five different individuals of Greater horseshoe bats (*Rhinolophus ferrumequinum*), upright in-vivo shapes (60 to 80 kHz in steps of 2 kHz), k,l,m,n,o) in-vivo bent shapes from the same Greater horseshoe bat individuals and using the same frequencies, p) Common bent-wing bat (*Miniopterus schreibersii*, 47 to 87 kHz [86] in steps of 4 kHz), q) Round-eared tube-nosed bat (*Murina cyclotis*, 38 to 174 kHz [87] in steps of 8 kHz), r) Rickett's big-footed bat (*Myotis ricketti*, 30 to 70 kHz in steps of 4 kHz), s) Pipistrelle (*Pipistrellus sp.*, 35 to 100 kHz [86] in steps of 5 kHz), t) Lesser bamboo bat (*Tylonycteris pachypus*, 50 to 118 kHz [88] in steps of 4 kHz). For each frequency, a single contour line is drawn at a gain level of -6 dB. Frequencies are coded by color from blue for the lowest frequency to red for the highest frequency analyzed.

Bibliography

[1] R.M. Nowak. *Walker's mammals of the world.* Johns Hopkins Univ Pr, 1991.

[2] C. Tudge. *The variety of life.* Oxford Univ. Press, 2000. ISBN 0198503113.

[3] D.R. Griffin and R.B. Lindsay. Listening in the Dark. *Physics Today*, 12:42, 1959.

[4] J.A. Thomas, C.F. Moss, M. Vater, and P.W. Moore. Echolocation in bat and dolphins. *The Journal of the Acoustical Society of America*, 118:2755, 2005.

[5] J. Blauert. *Spatial Hearing - Revised Edition: The Psychophysics of Human Sound Localization.* MIT Press, Cambridge, MA, 1996.

[6] S. Mehrgardt and V. Mellert. Transformation characteristics of the external human ear. *J. Acoust. Soc. Am*, 61(6):1567–1576, June 1977.

[7] A. D. Musicant, J. C.K. Chan, and J. E. Hind. Direction-dependent spectral properties of cat external ear: New data and cross-species comparisons. *J. Acoust. Soc. Am.*, 87 (2):757–781, February 1990.

[8] M. L. Spezio, C. H. Keller, R. T. Marrocco, and T. T. Takahashi. Head-related transfer functions of the Rhesus monkey. *Hearing Research*, 144:73–88, 2000.

[9] M. Aytekin, E. Grassi, M. Sahota, and C. F. Moss. The bat head-related transfer function reveals binaural cues for sound localization in azimuth and elevation. *J. Acoust. Soc. Am.*, 116(6):3594–3605, Dec 2004.

[10] H. U. Schnitzler and A. D. Grinnell. Directional Sensitivity of Echolocation in the Horseshoe Bat, *Rhinolophus ferrumequinum* II. Behavioral Directionality of Hearing. *J. Comp. Physiol. A*, 116:63–76, 1977.

[11] M.A. Obrist, M.B. Fenton, J.L. Eger, and P.A. Schlegel. What ears do for bats: A comparative study of pinna sound pressure transformation in Chiroptera. *J. Exp. Biol.*, 180:119–152, 1993.

[12] R. Müller. Numerical Analysis of Biosonar Beamforming Mechanisms and Strategies in Bats. *J. Acoust. Soc. Am.*, 127(5), May 2010. in press.

[13] G. Jones and E. C. Teeling. The evolution of echolocation in bats. *Trends Ecol. Evol.*, 21(3):149–156, Mar 2006.

[14] H. Schneider and F.P. Möhres. Die Ohrbewegungen der Hufeisennasenfledermäuse (Chiroptera, Rhinolophidae) und der Mechanismus des Bildhörens. *Z. Vergl. Physiol.*, 44 (1):1–40, January 1960.

[15] D.R. Griffin, D.C. Dunning, D.A. Cahlander, and F.A. Webster. Correlated orientation sounds and ear movements of horseshoe bats. *Nature*, 196:1185–1186, December 1962.

[16] J. Mogdans, J. Ostwald, and H.-U. Schnitzler. The role of pinna movement for the localization of vertical and horizontal wire obstacles in the greater horseshoe bat, *Rhinolopus ferrumequinum*. *J. Acoust. Soc. Am.*, 84(5):1676–1679, November 1988.

[17] V. A. Walker, H. Peremans, and J. C. T. Hallam. One tone, two ears, three dimensions: A robotic investigation of pinnae movements used by rhinolophid and hipposiderid bats. *J. Acoust. Soc. Am.*, 104(1):569–579, July 1998.

[18] J. Montagnat, H. Delingette, and N. Ayache. A review of deformable surfaces: topology, geometry and deformation. *Image and vision computing*, 19(14):1023–1040, 2001.

[19] A.S. Aguado, M.S. Nixon, and M.E. Montiel. Parameterizing arbitrary shapes via Fourier descriptors for evidence-gathering extraction. *Computer Vision and Image Understanding*, 69(2):202–221, 1998.

[20] C. Dorai and A.K. Jain. COSMOS-A representation scheme for 3D free-form objects. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(10):1115–1130, 2002.

[21] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 205–214. ACM, 1987. ISBN 0897912276.

[22] A. Blake and M. Isard. 3D position, attitude and shape input using video tracking of hands and lips. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 185–192. ACM, 1994. ISBN 0897916670.

[23] N. Ayache, P. Cinquin, I. Cohen, L. Cohen, F. Leitner, and O. Monga. Segmentation of complex threedimensional medical objects: a challenge and a requirement for computer-assisted surgery planning and performance. *Computerintegrated surgery: technology and clinical applications*, pages 59–74, 1996.

[24] M. Kamber, R. Shinghal, D.L. Collins, G.S. Francis, and A.C. Evans. Model-based 3-D segmentation of multiple sclerosis lesions in magnetic resonance brain images. *Medical Imaging, IEEE Transactions on*, 14(3):442–453, 2002.

[25] J. Montagnat and H. Delingette. Volumetric medical images segmentation using shape constrained deformable models. In *CVRMed-MRCAS'97*, pages 13–22. Springer, 1997.

[26] A. Moreau-Gaudry, P. Cinquin, and J.P. Baguet. Active model based carotid ultrasonic data segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI99*, pages 176–183. Springer, 1999.

[27] G. Szekely, A. Kelemen, C. Brechbuehler, and G. Gerig. Segmentation of 3D objects from MRI volume data using constrained elastic deformations of flexible Fourier surface models. In *Computer Vision, Virtual Reality and Robotics in Medicine*, pages 493–505. Springer, 1995.

[28] S. Cotin, H. Delingette, and N. Ayache. A hybrid elastic model for real-time cutting, deformations, and force feedback for surgery training and simulation. *The Visual Computer*, 16(8):437–452, 2000.

[29] A. Nealen, M. Müller, R. Keiser, E. Boxerman, and M. Carlson. Physically based deformable models in computer graphics. In *Computer Graphics Forum*, volume 25, pages 809–836. Wiley Online Library, 2006.

[30] S.F.F. Gibson and B. Mirtich. A survey of deformable modeling in computer graphics. *MERL, TR-97*, 19, 1997.

[31] Y. Chen, Q. Zhu, A. Kaufman, and S. Muraki. Physically-based animation of volumetric objects. In *Computer Animation 98. Proceedings*, pages 154–160. IEEE, 2002. ISBN 0818685417.

[32] S.M. Platt and N.I. Badler. Animating facial expressions. *ACM SIGGRAPH computer graphics*, 15(3):245–252, 1981.

[33] K. Waters. A muscle model for animation three-dimensional facial expression. *ACM SIGGRAPH Computer Graphics*, 21(4):17–24, 1987.

[34] J.E. Chadwick, D.R. Haumann, and R.E. Parent. Layered construction for deformable animated characters. In *Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, page 252. ACM, 1989. ISBN 0897913124.

[35] D. Terzopoulos and K. Waters. Physically-based facial modeling, analysis, and animation. *Journal of visualization and Computer Animation*, 1(2):73–80, 1990.

[36] K. Waters and D. Terzopoulos. Modelling and animating faces using scanned data. *The Journal of Visualization and Computer Animation*, 2(4):123–128, 1991.

[37] M. Teschner, B. Heidelberger, M. Müller, and M. Gross. A versatile and robust model for geometrically complex deformable solids. In *Computer Graphics International, 2004. Proceedings*, pages 312–319. IEEE, 2005. ISBN 0769521711.

[38] D.E. Breen, D.H. House, and M.J. Wozny. Predicting the drape of woven cloth using interacting particles. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 365–372. ACM, 1994. ISBN 0897916670.

[39] D. Baraff and A. Witkin. Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 43–54. ACM, 1998. ISBN 0897919998.

[40] G.S.P. Miller. The motion dynamics of snakes and worms. *ACM Siggraph Computer Graphics*, 22(4):169–173, 1988.

[41] X. Tu and D. Terzopoulos. Artificial fishes: Physics, locomotion, perception, behavior. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 43–50. ACM, 1994. ISBN 0897916670.

[42] P.G. Ciarlet. *Mathematical elasticity: three-dimensional elasticity*. North Holland, 1988.

[43] R.D. Cook, D.S. Malkus, M.E. Plesha, and R.J. Witt. *Concepts and Applications of Finite Element Analysis*. John Wiley & Sons, 2007. ISBN 0470088214.

[44] J.F. O'Brien and J.K. Hodgins. Graphical modeling and animation of brittle fracture. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 137–146. ACM Press/Addison-Wesley Publishing Co., 1999. ISBN 0201485605.

[45] G. Debunne, M. Desbrun, M.P. Cani, and A.H. Barr. Dynamic real-time deformations using space & time adaptive sampling. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 31–36. ACM, 2001. ISBN 158113374X.

[46] M. Müller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler. Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation, July*, pages 21–22. Citeseer, 2002.

[47] J. Smith, A. Witkin, and D. Baraff. Fast and controllable simulation of the shattering of brittle objects. In *Computer Graphics Forum*, volume 20, pages 81–91. John Wiley & Sons, 2001.

[48] M. Müller, L. McMillan, J. Dorsey, and R. Jagnow. Real-time simulation of deformation and fracture of stiff materials. In *Proceedings of the Eurographic workshop on Computer animation and simulation*, pages 113–124. Citeseer, 2001. ISBN 3211837116.

[49] J.F. O'Brien, A.W. Bargteil, and J.K. Hodgins. Graphical modeling and animation of ductile fracture. *ACM Transactions on Graphics (TOG)*, 21(3):291–294, 2002.

[50] J.P. Gourret, NM Thalmann, and D. Thalmann. Simulation of object and human skin formations in a grasping task. *ACM Siggraph Computer Graphics*, 23(3):21–30, 1989.

[51] S.A. Cover, N.F. Ezquerra, J.F. O'Brien, R. Rowe, T. Gadacz, and E. Palm. Interactively deformable models for surgery simulation. *Computer Graphics and Applications, IEEE*, 13(6):68–75, 2002.

[52] M.A. Sagar, D. Bullivant, G.D. Mallinson, and P.J. Hunter. A virtual environment and model of the eye for surgical simulation. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 205–212. ACM, 1994. ISBN 0897916670.

[53] R.M. Koch, M.H. Gross, F.R. Carls, D.F. von Büren, G. Fankhauser, and Y.I.H. Parish. Simulating facial surgery using finite element models. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 421–428. ACM, 1996. ISBN 0897917464.

[54] S. Cotin, H. Delingette, and N. Ayache. Real-time elastic deformations of soft tissues for surgery simulation. *Visualization and Computer Graphics, IEEE Transactions on*, 5(1):62–73, 2002.

[55] G. Picinbono, H. Delingette, and N. Ayache. Real-time large displacement elasticity for surgery simulation: Non-linear tensor-mass model. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2000*. Springer, 2000.

[56] G. Picinbono, H. Delingette, and N. Ayache. Nonlinear and anisotropic elastic soft tissue models for medical simulation. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 1370–1375. IEEE, 2005. ISBN 0780365763.

[57] C. Laugier, C. Mendoza, and K. Sundaraj. Towards a realistic medical simulator using virtual environments and force feedback. In *International Symposium in Research Robotics ISRR-2001, Australia*, 2001.

[58] D.L. James and D.K. Pai. BD-tree: output-sensitive collision detection for reduced deformable models. In *ACM SIGGRAPH 2004 Papers*, pages 393–398. ACM, 2004.

[59] J. Inman Daniel. *Engineering Vibration*. Prentice Hall, New Jersey, 2001.

[60] A. Pentland and J. Williams. Good vibrations: Modal dynamics for graphics and animation. In *Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, pages 215–222. ACM, 1989. ISBN 0897913124.

[61] J. Stam. Stochastic dynamics: Simulating the effects of turbulence on flexible structures. In *Computer Graphics Forum*, volume 16, pages C159–C164. Wiley Online Library, 1997.

[62] J. Barbič and D.L. James. Real-Time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Transactions on Graphics (TOG)*, 24(3):982–990, 2005.

[63] W.H. Press, SA Teukolsky, WT Vetterling, and BP Flannery. *Numerical Recipes in C: The Art of Scientific Computing (Cambridge*. Cambridge Univ. Press, 1992.

[64] U.M. Ascher and L.R. Petzold. *Computer methods for ordinary differential equations and differential-algebraic equations*. Society for Industrial Mathematics, 1998. ISBN 0898714125.

[65] M. Müller, B. Heidelberger, M. Teschner, and M. Gross. Meshless deformations based on shape matching. *ACM Transactions on Graphics (TOG)*, 24(3):471–478, 2005.

[66] J. Heikkila and O. Silven. A Four-step Camera Calibration Procedure with Implicit Image Correction. pages 1106–1112, Los Alamitos, CA, USA, 1997. IEEE Computer Society.

[67] G.R. Bradski and A. Kaehler. *Learning OpenCV*. O'Reilly, 2008. ISBN 9780596516130.

[68] Z. Zhang. Iterative Point Matching for Registration of Free-Form Curves and Surfaces. *Int. J. Comp. Vis*, 7(3):119–152, 1994.

[69] O. Clatz, H. Delingette, I.F. Talos, A.J Golby, R. Kikinis, F.A Jolesz, N. Ayache, and S.K. Warfield. Robust nonrigid registration to capture brain shift from intraoperative MRI. *IEEE Trans. Med. Imaging*, 24(11):1417–1427, 2005.

[70] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*, 21(4):163–169, 1987.

[71] W. Schroeder, K. Martin, and B. Lorensen. *The Visualization Toolkit - An Object-Oriented Approach to 3D Graphics*. Kitware, Inc., 2006.

[72] S.K. Kim and C.H. Kim. Finding ridges and valleys in a discrete surface using a modified MLS approximation. *Comput. Aided Des.*, 38(2):173–180, 2006.

[73] A.G. Belyaev, A.A. Pasko, and T.L. Kunii. Ridges and ravines on implicit surfaces. In *Computer Graphics International, 1998. Proceedings*, pages 530–535. IEEE, 2002. ISBN 0818684453.

[74] G. Celniker and D. Gossard. Deformable curve and surface finite-elements for free-form shape design. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, pages 257–266. ACM, 1991. ISBN 0897914368.

[75] L. Matthies and S. Shafer. Error modeling in stereo navigation. *Robotics and Automation, IEEE Journal of*, 3(3):239–248, 2002.

[76] V.C. Mow and X.E. Guo. Mechano-Electrochemical Properties of Articular Cartilage: their Inhomogeneities and Anisotropies. *Annual Review of Biomedical engineering*, 4: 175–210, 2002.

[77] X. Wang and R. Müller. Pinna-rim skin folds narrow the sonar beam in the lesser false vampire bat (Megaderma spasma). *The Journal of the Acoustical Society of America*, 126:3311, 2009.

[78] R. Müller. Numerical analysis of biosonar beamforming mechanisms and strategies in bats. *The Journal of the Acoustical Society of America*, 128(3):1414, 2010.

[79] K. R. Rao and J. Ben-Arie. Optimal Head Related Transfer Functions for Localization and Hearing: A Signal Processing Design Perspective. *IEEE Trans. Biomed. Eng.*, 43 (11):1093–1105, Nov 1996.

[80] R. Müller, H. Lu, and J. R. Buck. Sound-diffracting flap in the ear of a bat generates spatial information. *Phys. Rev. Lett.*, 100(10):108701–4, March 2008.

[81] G. Li, B. Liang, Y. Wang, H. Zhao, K. M. Helgen, L. Lin, G. Jones, and S. Zhang. Echolocation calls, diet, and phylogenetic relationships of stoliczka's trident bat, *Aselliscus stoliczkanus* (Hipposideridae). *Journal of Mammalogy*, 88(3):736–744, June 2007.

[82] G. Jones and M. W. Holderied. Bat echolocation calls: adaptation and convergent evolution. *Proc. R. Soc. B*, 274:905–912, 2007.

[83] F. Luo, A. Li, J. Ma, F. Wu, B. Liang, Q. Chen, and S. Zhang. Spectrotemporal characteristics of echolocation pulses and frequency tuning by inferior collicular neuron of *rhinolophus pusillus*. *Acta Zool Sinica*, 52(3):528–535, Jun. 2006.

[84] L. Zhang, G. Jones, J. Zhang, G. Zhu, S. Parsons, S. J. Rossiter, and S. Zhang. Recent surveys of bats (Mammalia: Chiroptera) from China. I. Rhinolophidae and Hipposideridae. *Acta Chiropterologica*, 11(1):71–88, 2009.

[85] H. Zhao, S. Zhang, M. Zuo, and J. Zhou. Correlations between call frequency and ear length in bats belonging to the families rhinolophidae and hipposideridae. *Journal of Zoology*, 259(2):189–195, 2003.

[86] M. K. Obrist, R. Boesch, and P. F. Flückiger. Variability in echolocation call design of 26 Swiss bat species: consequences, limits and options for automated field identification with a synergetic pattern recognition approach. *Mammalia*, 68(4):307–322, 2004.

[87] T. Kingston, G. Jones, Z. Akbar, and T. H. Kunz. Echolocation signal design in Kerivoulinae and Murininae (Chiroptera: Vespertilionidae) from Malaysia. *J. Zool., Lond.*, 249:359–374, 1999.

[88] L. Zhang, B. Liang, S. Parsons, L. Wei, and S. Zhang. Morphology, echolocation and foraging behaviour in two sympatric sibling species of bat (*Tylonycteris pachypus* and *Tylonycteris robustula*) (Chiroptera: Vespertilionidae). *Journal of Zoology*, 271:344–351, 2007.

[89] J. D. Pye and L. H. Roberts. Ear Movements in a Hipposiderid Bat. *Nature*, 225: 285–286, January 1970.

Appendix A

Implementation - Computer codes

A.1 Calculating the displacements at landmarks

A.1.1 Rigid registration

Matlab program *rigidregmsh.m*

USAGE

```
function rcrd = rigidregmsh(mshfnme, ptcld, rev, figflg)
```

INPUTS

Variable	Type	Description
mshfnme	String	Name of the mesh file to be rigid registered
ptcld	Matrix ($N_{pts} \times 3$)	Point cloud to which the landmarks in the mesh needs to be rigid registered
rev	Vector	Eigen vectors to be reversed for rigid registration
figflg	Logical	Flag to plot the result of rigid registration

OUTPUTS

Variable	Type	Description
rcrd	Matrix ($N \times 3$)	Rigid registered mesh nodes

Program Description: The program is used to rigid register the mesh (CT model) with the stereo-triangulated landmarks. The optimal rotation and translation which aligns the landmarks in the mesh with the point cloud (stereo triangulated landmarks) is found out using eigen vector matching and Iterative Closest Point (ICP) algorithm. These optimal rotation and translation is then applied to the whole mesh.

The covariance matrices and their eigen vectors of both the landmarks in the mesh and the point cloud is obtained.

```
% finding the eigen vectors of the landmarks in the mesh
dircrd = undefcrd(dir,:);
dircrdcov = cov(dircrd);
[dir_eigvec,dir_eigval]=eig(dircrdcov);
```

```
% finding the eigen vectors of the point cloud
ptcld_cov = cov(ptcld);
[pt_eigvec, pt_eigval] = eig(ptcld_cov);
```

The negative of the eigen vectors are also eigen vectors hence the eigen vectors calculated by matlab could be the negative of what is actually required. The eigen vectors that need to be reversed is determined by the user and given the `rev` vector.

```
dir_eigvec(:,rev) = -dir_eigvec(:,rev); %User defined
```

The eigen vectors are now matched by finding the rotations which will align these vectors and translating the centre of mass of the landmarks in the mesh to the center of mass of the point cloud.

```
Rin = inv(dir_eigvec')*pt_eigvec';

rcrd = undefcrd*Rin;
rdircrd = rcrd(dir,:);
rotdirmean = mean(rdircrd);

rcrd = rcrd + repmat(pt_mean-rotdirmean,size(undefcrd,1),1);
```

Further alignment is done using the ICP method. It was observed that the ICP method works better when the “model”, to which the “data” is to be aligned, has more points. In our case the mesh (“data”) has to be aligned to the point cloud (“model”). When the number of landmarks in the mesh is more than the number of points in the point cloud, the ICP method is applied using the mesh as the “model” and the reverse of the optimal rotation and translation is applied to the mesh.

```
if size(rdircrd,1)>size(ptcld,1)
[Ricp,Ticp]=icp(rdircrd,ptcld,1000,4,2,1e-15);
rcrd = (inv(Ricp)*(rcrd' - repmat(Ticp,1,size(rcrd,1))))';
else
[Ricp,Ticp] = icp(ptcld,rdircrd,1000,4,2,1e-15);
rcrd = (Ricp*rcrd' + repmat(Ticp,1,size(rcrd,1)))';
```

```
end
```

A.2 Deformable Model

A.2.1 Making the stiffness matrix

Matlab program *makesys.m*

USAGE

```
Ksys = makesys(coordinates,elements,E,nu);
```

INPUTS

Variable	Type	Description
coordinates	Matrix ($N \times 3$)	Nodal coordinates of the mesh
elements	Matrix ($N_{el} \times 8$)	Mesh element connectivity matrix
E	Double	Young' Modulus
nu	Double	Poisson's ratio

OUTPUTS

Variable	Type	Description
`Ksys`	Matrix ($3N \times 3N$)	Stiffness matrix

Program Description: Calculates the stiffness matrix of a hexahedral mesh assuming linear elastic material behavior.

The program loops over all elements, calculates the elemental stiffness matrix of each element and finally assembles all the elemental stiffness matrices into a global stiffness matrix. Each element (in **XYZ** coordinate system) is transformed into a standard cube of dimension $2 \times 2 \times 2$ centered at the origin whose edges are parallel to the transformed coordinate axes ($\boldsymbol{\epsilon\nu\tau}$ coordinate system). The elemental stiffness matrix is calculated by calculating the integral (Eq. (3.14)) on this standard cube using 3 point gauss quadrature along each coordinate axes (ie, $3 \times 3 \times 3 = 27$ gauss points).

```
pts = [-sqrt(0.6) 0 sqrt(0.6)]; %Gauss points along each axis
wts = [5/9 8/9 5/9]; %Weights corresponding to each gauss point
```

Gauss quadrature approximation of the integral is calculated by looping over all the gauss points and taking a weighted sum of the integrand at each of these points.

```
for e=1:size(pts,2)
     for n=1:size(pts,2)
          for t=1:size(pts,2)
```

At each gauss point, the derivatives of each of the linear shape function with respect to the coordinate directions $\boldsymbol{\epsilon}$, $\boldsymbol{\nu}$ and $\boldsymbol{\tau}$ is calculated using the function *dNds.m.* The function *dNds* was generated using the symbolic math toolbox and the point at which the derivatives are to be computed is inputted to the function as a 3 element vector.

```
dNs = dNds([pts(e) pts(n) pts(t)]);
```

The derivatives (dNs) are obtained as a 3×8 matrix

$$\mathrm{dNs} = \begin{bmatrix} \frac{\partial N_1}{\partial \eta} & \frac{\partial N_2}{\partial \eta} & \cdots & \frac{\partial N_8}{\partial \eta} \\ \frac{\partial N_1}{\partial \nu} & \frac{\partial N_2}{\partial \nu} & \cdots & \frac{\partial N_8}{\partial \nu} \\ \frac{\partial N_1}{\partial \tau} & \frac{\partial N_2}{\partial \tau} & \cdots & \frac{\partial N_8}{\partial \tau} \end{bmatrix}$$

The coordinate system transformation from the **XYZ** coordinate system to the $\boldsymbol{\epsilon\nu\tau}$ coordinate system was done using shape functions

$$\begin{aligned} x &= N_1 x_1 + N_2 x_2 + \ldots + N_8 x_8 \\ y &= N_1 y_1 + N_2 y_2 + \ldots + N_8 y_8 \\ z &= N_1 z_1 + N_2 z_2 + \ldots + N_8 z_8 \end{aligned}$$

The Jacobian of the coordinate system transformation is given by

$$\begin{bmatrix} \frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \nu} & \frac{\partial x}{\partial \tau} \\ \frac{\partial y}{\partial \eta} & \frac{\partial y}{\partial \nu} & \frac{\partial y}{\partial \tau} \\ \frac{\partial z}{\partial \eta} & \frac{\partial z}{\partial \nu} & \frac{\partial z}{\partial \tau} \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & \ldots & x_8 \\ y_1 & y_2 & \ldots & y_8 \\ z_1 & z_2 & \ldots & z_8 \end{bmatrix} \begin{bmatrix} \frac{\partial N_1}{\partial \eta} & \frac{\partial N_1}{\partial \nu} & \frac{\partial N_1}{\partial \tau} \\ \frac{\partial N_2}{\partial \eta} & \frac{\partial N_2}{\partial \nu} & \frac{\partial N_2}{\partial \tau} \\ \vdots & \vdots & \vdots \\ \frac{\partial N_8}{\partial \eta} & \frac{\partial N_8}{\partial \nu} & \frac{\partial N_8}{\partial \tau} \end{bmatrix}$$

```
J = (dNs*elcrd)';
```

The strains are functions of the derivatives of the displacement with respect to the **XYZ** coordinate directions (Eq. (3.4)). For example,

$$\begin{aligned}\epsilon_{xx} &= \frac{\partial u_x}{\partial x} \\ &= \frac{\partial N_1}{\partial x}u_{x1} + \frac{\partial N_2}{\partial x}u_{x2} + \ldots\ldots + \frac{\partial N_8}{\partial x}u_{x8}\end{aligned}$$

The partial derivative of the shape functions with respect to the **XYZ** coordinate system is given by chain rule. For example,

$$\frac{\partial N_1}{\partial x} = \frac{\partial N_1}{\partial \eta}\frac{\partial \eta}{\partial x} + \frac{\partial N_1}{\partial \nu}\frac{\partial \nu}{\partial x} + \frac{\partial N_1}{\partial \tau}\frac{\partial \tau}{\partial x}$$

The partial derivatives of the $\boldsymbol{\epsilon\nu\tau}$ coordinates with respect to the **XYZ** coordinates is obtained by taking the inverse of the Jacobian matrix.

```
Jinv = inv(J);
```

Suitably arranged inverse Jacobian matrices can be multiplied with the derivatives of the shape functions with respect to the $\boldsymbol{\epsilon\nu\tau}$ coordinates to get the derivatives of the shape functions with respect to the **XYZ** coordinates. This can further be used to find the strains.

```
Jinv_9x9 =[Jinv  zeros(3,6)
           zeros(3,3)  Jinv  zeros(3,3)
           zeros(3,6)  Jinv];
```

```
dN = zeros(9,24);
dN(1:3,1:3:end) = dNs; dN(4:6,2:3:end) = dNs; dN(7:9,3:3:end) = dNs;
strain = difftostrn*Jinv_9x9*dN;
```

The stresses can be obtained from the strains by multiplying it with the material matrix **C** (Eq. (3.1)) which can further be multiplied with the strains to get the stiffness matrix integrand (Eqs. (3.12), (3.13), (3.14)). The weighted integrand is added over all gauss points to obtain the elemental stiffness matrix. The differential volume in the **XYZ** coordinate system is transformed into a differential volume in the $\boldsymbol{\epsilon\nu\tau}$ coordinate system by multiplying with the determinant of the Jacobian matrix

$$dxdydz = |J|d\eta d\nu d\tau$$

```
stima = stima + wts(e)*wts(n)*wts(t)*strain'*C*strain*det(J);
```

The location (row and column number) and value of the each entry of the elemental stiffness matrix in the global stiffness matrix are stored in three different vectors.

```
I = 3 * elements(n_el,[1,1,1,2,2,2,3,3,3,4,4,4,5,5,5,6,6,6,7,7,7,8,8,8]) - ...
                      [2,1,0,2,1,0,2,1,0,2,1,0,2,1,0,2,1,0,2,1,0,2,1,0];
```

```
rw(:,n_el)=repmat(I',24,1);
cl(:,n_el)=reshape(repmat(I,24,1),24*24,1);
vl(:,n_el)=reshape(stima,24*24,1);
```

Once all the elemental stiffness matrices are calculated, the sparse global stiffness matrix is assembled

```
rw = reshape(rw,24*24*size(elements,1),1);
cl = reshape(cl,24*24* size(elements,1),1);
vl = reshape(vl,24*24* size(elements,1),1);
Ksys = sparse(rw,cl,vl,3*size(coordinates,1),3*size(coordinates,1));
```

A.2.2 Solving for a deformed shape

Matlab program *pnltyfem.m*

USAGE
`pnltyfem(mshfnme,dirfnme,dirvarnme,sysfnme,pnlty,defmshfnme)`

INPUTS

Variable	Type	Description
mshfname	String	Name of the binary file containing the undeformed mesh
dirfnme	String	Name of the MAT file containing the landmark location and the displacement at each of them. A $N_{lnd} \times 4$ matrix with the first column giving the element number whose center is the landmark location and the subsequent three columns give the displacement (in meters) in the **X**, **Y** and **Z** directions
dirvarnme	String	Name of the variable containing the displacements
sysfnme	String	Name of the MAT file containing the stiffness matrix
pnlty	Double	Weight of the penalty term. Θ in Eq. (3.25)
defmshfnme	String	Name of the file to which the deformed mesh should be written to

OUTPUTS
NONE

Program Description: All the required files (undeformed mesh file, stiffness matrix file and displacements file) are read. The landmark displacement selection matrix [**H**], the weighting matrix [**S**] and the displacements vector [**D**] were made.

```
ndirvec = 1:n_dir;
```

```
els = reshape(elmnts(eldir(:,1),:)',1,8*n_dir);

Hrw = reshape(repmat([3*ndirvec-2;3*ndirvec-1;3*ndirvec],8,1),24*n_dir,1);
Hcl = reshape([3*els-2;3*els-1;3*els],24*n_dir,1);
Hval = (1/8)*ones(24*n_dir,1);
H = sparse(Hrw,Hcl,Hval,3*n_dir,3*ncrd);

S = pnlty*sparse(1:3*n_dir,1:3*n_dir,1);

D = reshape(eldir(:,2:4)',3*n_dir,1);
```

The system matrix and forcing vector for solving the system of linear equations (Eq. (3.27)) were made

```
Ksys = Ksys + H'*S*H;
F = H'*S*D;
```

The set of linear equations was solved using the *minres* iterative solver to get the displacements at all the nodes

```
U = minres(Ksys,F,1e-8,5000);
```

The displacements at the nodes were added on to the undeformed coordinates to get the deformed coordinates.

```
defcrd=undefcrd+[U(1:3:end) U(2:3:end) U(3:3:end)];
```

The deformed mesh is written to a binary file using the *mshwrite.m* function

```
mshwrite(defmshfnme,defcrd,elmnts,dirch,neu);
```

A.3 Curvature Calculation

A.3.1 Aligning the deformed mesh with the coordinate axes

Matlab function *autorotmodel.m*

USAGE
`autorotmodel(mshfnme,figflg)`

INPUTS

Variable	Type	Description
mshfnme	String	Name of the deformed mesh file
figflg	Logical	Flag for plotting the aligned mesh

OUTPUTS

NONE

Program Description: The program aligns the principal directions of the mesh with the coordinate axes and translates the center of mass of the mesh to the origin.

The eigen vectors of the covariance of the nodal coordinates are the principle directions of the mesh.

```
[vec, ]=eig(cov(crd));
```

The mesh is rotated to align the eigen vectors along the **XYZ** directions such ear faces the **X** direction and the length of the ear is along the **Z** direction. The eigen vector corresponding to the largest eigen value is aligned with the **Z** direction (bottom-top), eigen vector corresponding to the second highest eigen value is aligned with the **Y** direction left - right and the eiqen vector corresponding to the smallest eigen value is aligned with the **X** direction (back to front).

```
rotdef = crd/vec';
```

The rotated mesh is translated such that the center of mass of the mesh nodes is at the origin

```
rotmean = mean(rotdef);
rotdef = rotdef-repmat(rotmean,size(rotdef,1),1);
```

The negative of the eigen vectors are also eigen vectors hence the eigen vectors calculated by matlab could be the negative of what is actually required. For eg., after aligning the eigen vectors with the coordinate directions, the ear might end up being upside down because the eigen vector corresponding to the largest eigen value was pointing to the bottom instead of pointing to the tip. Hence some eigen vectors might need to be reversed before alignment. Some priori knowledge about the shape of the ear is used to understand which of the eigen vectors need to be reversed.

1. The center of mass of the ear is towards the bottom of the ear. If the sense of the eigen vector corresponding to the largest eigen value is wrong (ie, pointing to the bottom of the ear), then the rotated and translated ear will be longer to the bottom of the Z-plane. Hence if it is observed that the rotated and translated ear is longer below the Z plane, then the eigen vector corresponding to the largest eigen value should be reversed.

   ```
   if abs(min(rotdef(:,3)))>abs(max(rotdef(:,3)))
   vec(:,3)=-vec(:,3);
   end
   ```

2. A horizontal section passing through the center of mass, will be longer towards the front of the ear than to the back. If the sense of the eigen vector corresponding to the smallest eigen value is wrong (ie, pointing to the back of the ear), then for the rotated and translated ear a horizontal cross section at the center of mass will be longer in the negative X-axis. Hence if it is such, then the eigen vector corresponding to the smallest eigen value should be reversed.

   ```
   zcutdf = rotdef(rotdef(:,3)<0.001 & rotdef(:,3)>-0.001,:);
   if abs(min(zcutdf(:,1)))>abs(max(zcutdf(:,1))) vec(:,1)=-vec(:,1);
   end
   ```

3. Since the eigen vectors need to form a right handed coordinate system, the eigen vector corresponding to the second highest eigen value can be obtained by taking the cross product of the eigen vector corresponding to the highest eigen value with that corresponding to the smallest eigen value.

   ```
   vec(:,2) = cross(vec(:,3),vec(:,1));
   ```

Once the correct sense of the eigen vectors are established, the original mesh coordinates are aligned to the coordinate system and translated to the origin based on the corrected eigen vectors.

```
rotdef = crd/vec';
rotmean = mean(rotdef);
rotdef = rotdef-repmat(rotmean,size(rotdef,1),1);
```

The aligned mesh is written into a mesh file using the *plyvxlmshwrite* function

```
plyvxlmshwrite(plyvxlmshfnme,rotdef,elmnts-1);
```

A.3.2 Converting mesh to a voxel representation

C++ program *polytovoxel*. Uses the VTK libraries.

USAGE
```
polytovoxel [options] filename.plyvxlmsh
```

INPUTS

Option	Description
–a	Output file written in ascii format
–b xmin, xmax, ymin, ymax, zmin, zmax	Bounds of the voxel stack. Default bound is the extend of the model + 2 times the sample spacing
–s spc	Set sample spacing (default: 1.0)
–h	Print help
–v	Verbose output

Program Description: Writes a voxel representation of the mesh. Each voxel is assigned a value of 255 (white) if it is outside the mesh and 0 (black) if inside.

The mesh is read from the file and made in VTK. The extend of the mesh is obtained.

```
hexgrd->GetBounds(ext);
```

If the bounds of the voxel representation has not been set by the user, the bounds are taken to be larger than the extend in each direction by two sample spaces

```
 if (!bflg) {
bnd[0] = ext[0]-2*dx;
bnd[1] = ext[1]+2*dx;
bnd[2] = ext[2]-2*dx;
bnd[3] = ext[3]+2*dx;
bnd[4] = ext[4]-2*dx;
bnd[5] = ext[5]+2*dx;
}
```

The number of samples in each direction is calculated

```
 nsmpl[0] = (int)((bnd[1]-bnd[0])/dx)+1;
nsmpl[1] = (int)((bnd[3]-bnd[2])/dx)+1;
nsmpl[2] = (int)((bnd[5]-bnd[4])/dx)+1;
```

The bounds are adjusted to get the required sample spacing.

```
bnd[1] = bnd[0]+(nsmpl[0]-1)*dx;
bnd[3] = bnd[2]+(nsmpl[1]-1)*dx;
bnd[5] = bnd[4]+(nsmpl[2]-1)*dx;
```

Using the *Implicit Modeller* object in VTK, the distance of each point in the voxel representation grid from the mesh is found out.

```
vtkImplicitModeller *vxlmdl = vtkImplicitModeller::New();
vxlmdl->SetInput(hexgrd);
vxlmdl->SetSampleDimensions(nsmpl);
vxlmdl->SetMaximumDistance(3*dx);
vxlmdl->SetModelBounds(bnd);
vxlmdl->SetProcessModeToPerVoxel();
vxlmdl->SetOutputScalarTypeToDouble();
vxlmdl->AdjustBoundsOff();
vxlmdl->Update();
```

The grid points with the distance to the mesh less than a threshold (half of the sample spacing) is assumed to be inside the mesh (value=0) and outside (value=255) otherwise.

```
double *vflt = (double*)(vxlmdl->GetOutput()->GetScalarPointer());

  for (i=0; i<vxlmdl->GetOutput()->GetNumberOfPoints(); i++)
     if (vflt[i]<(0.5*dx))
         vflt[i] = 0;
     else
         vflt[i] = 255;
}
```

The voxel representation is written into a file.

A.3.3 Obtaining the surface mesh of the top of the pinna (part above the antitragus)

C++ program *clipear*. Uses the VTK libraries.

USAGE
`clipear [options] filename.vtk`

INPUTS

Option	Description
–f	Flag for showing the cut model
–h	Print help
–t tol	Tolerance for triangulation
–v	Verbose output

Program Description: The surface mesh of the entire pinna is obtained from the voxel representation by using the *srfmsh* program. A triangular surface mesh of the top portion of the pinna (above the antitragus) is cut off from this surface mesh of the entire pinna using the *clipear* program.

The surface mesh of the entire pinna is first read into the program. This surface mesh is "cleaned" (removing the duplicate points, re-stitching the mesh etc.) using the *CleanPolyData* filter.

```
vtkCleanPolyData *clply1 = vtkCleanPolyData::New();
clply1->AddInput(reader->GetOutput());
clply1->SetTolerance(0.0);
clply1->ConvertPolysToLinesOff();
```

```
clply1->ConvertLinesToPointsOff();
clply1->Update();
```

The largest contiguous mesh is extracted from this "cleaned" mesh using the *PolyDataConnectivityFilter*

```
vtkPolyDataConnectivityFilter *plycnt = vtkPolyDataConnectivityFilter::New();
plycnt->SetInput(clply1->GetOutput());
plycnt->SetExtractionModeToLargestRegion();
plycnt->Update();
plydat = plycnt->GetOutput();
```

A plane perpendicular to **Z** direction for cutting this mesh is made

```
vtkPlane *pln = vtkPlane::New();
pln->SetNormal(0,0,1);
```

Initially, the plane is defined to be somewhere above the antitragus. The plane is brought down in small steps and at each step the number of connected pieces of mesh in the part above the plane is checked. When the number becomes two, it means that the antitragus has been reached. The previous position of the plane is taken to be the position just above the antitragus and used for clipping the mesh.

```
/* Defining the clipping function */
vtkClipPolyData *clp = vtkClipPolyData::New();
clp->SetInput(plydat);
clp->SetClipFunction(pln);

/* Defining the connectivity filter to be used for understanding the number of
regions in the clipped output */
vtkPolyDataConnectivityFilter *clpcnt = vtkPolyDataConnectivityFilter::New();
clpcnt->SetInput(clp->GetOutput());

while (!cntflg) {
      pln->SetOrigin(0,0,bnd[5]-0.008-0.0002*norg);

      //cutting from above the antitragus
      clp->Update();
      clpcnt->Update();

      //if the anti-tragus has been cut, then quit the loop
      if(clpcnt->GetNumberOfExtractedRegions()>=2)
           cntflg=1;
      else
           if(bnd[5]-0.008-0.0002*norg<bnd[4]) {
                fprintf(stderr,"Bottom of the ear reached after %d iterations n",norg)
```

```
            return(EXIT_FAILURE);}
        else
            norg++;
}
// cut with the plane just above the one that cut the antitragus
pln->SetOrigin(0,0,bnd[5]-0.008-0.0002*(norg-1));
clp->Update();
clpcnt->Update();
```

The clipped mesh obtained after cutting the original full pinna mesh with the plane just above the antitragus is written in binary format into a file. An extension *.plydat* was usually given to the file.

A.3.4 Calculating the curvature

Matlab function *getcurv.m*

USAGE

`out=getcurv(tri,pts,ngh)`

INPUTS

Variable	Type	Description
tri	Matrix ($N_{tri} \times$ 3)	Element connectivity of the triangular mesh
pts	Matrix ($N_{pts} \times 3$	Nodal coordinates
ngh	Double	Size of the neigbourhood used for surface fitting

OUTPUTS

Variable	Type	Description
out	Structure	Contains the principle curvature values and directions

Program Description: Calculates the principal curvature values of a triangular mesh.

The approximate normal at each node in the mesh is calculated.

```
nv = vertnorm_dw(tri,pts);
```

The normal at a node is calculated as the weighted average of the normal of the all the triangular elements containing the node. The weight is taken as the inverse of the distance between the node and the centroid of the triangle.

An inverse triangulation, ie a list of all the triangles containing each node, is built.

```
itri = buildInverseTriangulation(tri);
```

A loop through all the nodes of the mesh is taken

```
for n_pts=1:npts
```

For each node in the mesh, the nodes in a neighborhood of size given by `ngh` (default value = 3) is taken

```
crpts = n_pts;
for n_ngh=1:ngh
     % finding the triangles containing the current points (crpts) from
     %the inverse triangulation Note:  contains zeros and duplicates
     itrirs = reshape(itri(crpts,:),size(itri(crpts,:),1),size(itri(crpts,:),2));

     %removing the zeros and duplicates
     ngtris = unique(itrirs(logical(itrirs~=0)));

     %finding the unique vertices in the triangles got above
     crpts = unique(reshape(tri(ngtris,:),length(ngtris),3));
end
crpts = setdiff(crpts,n_pts);
```

An orthogonal frame of reference is made such that the z axis of the new reference frame is along the normal at the node. The origin of the reference frame is at the node. The coordinates of all the points with respect to this coordinate reference is calculated.

```
%make an orthogonal reference frame at the point with z-axis along
%the normal at the point
z_axs = nv(n_pts,:);
if(nv(n_pts,3)~=0) %zcmp =0
     x_zcmp = -1*nv(n_pts,1)/nv(n_pts,3);
     x_axs = [1 0 x_zcmp]/norm([1 0 x_zcmp]);
elseif(nv(n_pts,1)~=0) %xcmp =0
     x_xcmp = -1*nv(n_pts,2)/nv(n_pts,1);
     x_axs = [x_xcmp 1 0]/norm([x_xcmp 1 0]);
elseif(nv(n_pts,2)~=0) %ycmp =0
     x_ycmp = -1*nv(n_pts,3)/nv(n_pts,2);
     x_axs = [0 x_ycmp 1]/norm([0 x_ycmp 1]);
end
y_axs = cross(z_axs,x_axs);

trpts = (pts(crpts,:)-repmat(pts(n_pts,:),length(crpts),1))/[x_axs;y_axs;z_axs];
```

A third degree implicit surface is weighted (Eq. (3.29)) least squares fitted to this transformed coordinates. The equation of the surface is given as,

$$z = a_1x + a_2y + a_3x^2 + a_4xy + a_5y^2 + a_6x^3 + a_7x^2y + a_8xy^2 + a_9y^3$$

The constant term is avoided to make sure that the surface passes through the current node.

```
P = [trpts(:,1) trpts(:,2) trpts(:,1).^2 ...
trpts(:,1).*trpts(:,2) trpts(:,2).^2 trpts(:,1).^3 ...
(trpts(:,1).^2).*trpts(:,2) trpts(:,1).*(trpts(:,2).^2) trpts(:,2).^3];
```

```
%finding the one-third length of the farthest neighbor
h = max(sqrt(trpts(:,1).^2+trpts(:,2).^2+trpts(:,3).^2))/3;

%calculating the weights as W=e^(-s2/h2);
W = diag(exp(-sqrt(trpts(:,1).^2+trpts(:,2).^2+trpts(:,3).^2)/(h^2)));

%finding the polynomial coefficients
ply = (P'*W*P)(P'*W*trpts(:,3));
```

The normal of the surface at the current node is given by the relation [73]

$$\hat{n} = -\frac{\nabla F}{|\nabla F|}$$

where F is the implicit form of the surface

```
dF = [ply(1) ply(2) 1];
nrml(n_pts,:)  = -dF/norm(dF);
```

The matrix $-\nabla\hat{n}$ is calculated according to the relation [73]

$$-\nabla\hat{n} = \frac{1}{|\nabla F|}\left(\mathbf{I} - \hat{n}\cdot\hat{n}^T\right)\mathbf{Hes}(F)$$

where $\mathbf{Hes}(F)$ is the Hessian of the implicit relation given by

$$\mathbf{Hes}(F) = \begin{bmatrix} \frac{\partial^2 F}{\partial x^2} & \frac{\partial^2 F}{\partial x \partial y} & \frac{\partial^2 F}{\partial x \partial z} \\ \frac{\partial^2 F}{\partial y \partial x} & \frac{\partial^2 F}{\partial y^2} & \frac{\partial^2 F}{\partial y \partial z} \\ \frac{\partial^2 F}{\partial z \partial x} & \frac{\partial^2 F}{\partial z \partial y} & \frac{\partial^2 F}{\partial z^2} \end{bmatrix}$$

```
HesF = [2*ply(3) ply(4) 0;
        ply(4) 2*ply(5) 0;
```

```
        0 0 0];
ndeln = (1/norm(dF))*(eye(3)-nrml(n_pts,:)'*nrml(n_pts,:))*HesF;
```

The characteristic polynomial of the $\nabla\hat{n}$ matrix is used to find the mean and Gaussian curvatures. The characteristic polynomial of the matrix is given as [73]

$$\lambda^3 - 2H\lambda^2 + G\lambda$$

```
chply = poly(ndeln); %characteristic polynomial of the matrix
kmean(n_pts)=-chply(2)/2;
kg(n_pts)=chply(3);
```

The mean and Gaussian curvatures can be used to calculate the maximum and minimum principle curvatures [73]

$$k_{max} = H + \sqrt{H^2 - K}, \qquad k_{min} = H - \sqrt{H^2 - K}$$

```
kmax(n_pts)=kmean(n_pts)+sqrt(kmean(n_pts)^2-kg(n_pts));
kmin(n_pts)=kmean(n_pts)-sqrt(kmean(n_pts)^2-kg(n_pts));
```

The vectors containing the maximum and minimum principle curvatures is saved along with the mesh into a structure (`out`) which is returned from the function.

A.4 Surface Energy Calculation

A.4.1 Extract a surface mesh of the top portion of the pinna from the volume mesh

Matlab program *getsrf.m*

USAGE

`edgfcs=getsrf(crd,elmnts,zlw)`

INPUTS

Variable	Type	Description
crd	Matrix ($N\times3$)	Coordinates of the nodes of the mesh
elmnts	Matrix ($N_{el}\times8$)	Mesh element connectivity matrix
zlw	Double	Height of the Z plane used for cutting the mesh

OUTPUTS

Variable	Type	Description
edgfcs	$N_{srf}\times3$	Connectivity of the triangular surface mesh

Program Description: The program can be used to obtain the surface of a pinna mesh aligned with the coordinate axes. The program first removes all the elements below an user given z coordinate value (`zlw`) and then extracts a triangular surface mesh from the clipped mesh.

The nodes whose coordinates are above the `zlw` value are obtained.

```
zinflg = crd(:,3)>zlw;
```

Elements which do not contain any nodes whose z coordinate is below the given z threshold `zlw` is obtained

```
elin = false(size(elmnts,1),1);

for nels=1:size(elmnts,1)
     if all(zinflg(elmnts(nels,:)))
          elin(nels)=true;
     end
end

elsin = elmnts(elin,:);
```

A connectivity list of the faces of the clipped elements is made.

```
fcs = reshape([elsin(:,[1 4 3 2]) elsin(:,[1 2 6 5]) elsin(:,[2 3 7 6]) ...
               elsin(:,[3 4 8 7]) elsin(:,[4 1 5 8]) elsin(:,[5 6 7 8])]',...
               4,size(elsin,1)*6)';
```

The number of elements of which a node is a member is calculated for all the nodes.

```
cncts = zeros(size(crd,1),1);

for nels=1:size(elmnts,1)
     for nnds=1:size(elmnts,2)
          cncts(elmnts(nels,nnds))=cncts(elmnts(nels,nnds))+1;
     end
end
```

The nodes which is a part of less than 8 elements are on the outside of the mesh. A face whose all nodes are members of less than 8 elements is on the surface of the mesh.

```
edgflg =false(size(fcs,1),1);
```

```
for nfcs=1:size(fcs,1)
      if all(cncts(fcs(nfcs,:))~=8)
           edgflg(nfcs)=true;
      end
end
```

Each of these surface faces can be divided along the diagonal to get a triangular surface connectivity.

```
rctedgfcs = fcs(edgflg,:);

edgfcs = reshape([rctedgfcs(:,[1 2 3]) rctedgfcs(:,[1 3 4])]', ...
                  3,size(rctedgfcs,1)*2)';
```

A.4.2 Making the surface stiffness matrix

Matlab program *mkesrfsys.m*

USAGE
`Ksrf = mkesrfsys(crd,elmnts,alpha)`

INPUTS

Variable	Type	Description
crd	Matrix ($N \times 3$)	Nodal coordinates of the mesh
elmnts	Matrix ($N_{el} \times 3$)	Triangular surface mesh connectivity matrix
alpha	Matrix (2×2)	Material matrix

OUTPUTS

Variable	Type	Description
Ksrf	Sparse matrix ($3N \times 3N$)	Surface stiffness matrix

Program Description: Calculates the surface stiffness matrix for the triangular surface mesh.

The program loops over all elements (triangles), calculates the elemental stiffness matrix of each element and finally assembles all the elemental stiffness matrices into a global stiffness matrix.

```
parfor nels=1:size(elmnts,1)
```

The original **XYZ** coordinate system is transformed to a **uvw** coordinate system (s. Fig. 3.4) such that **w** is normal to the element and **u** and **v** are mutually perpendicular vectors perpendicular to **w**.

```
xdir = (crd(elmnts(nels,2),:)-crd(elmnts(nels,1),:))./ ...
        norm(crd(elmnts(nels,2),:)-crd(elmnts(nels,1),:));
zdir = cross(xdir,crd(elmnts(nels,3),:)-crd(elmnts(nels,1),:))./...
       norm(cross(xdir,crd(elmnts(nels,3),:)-crd(elmnts(nels,1),:)));
ydir = cross(zdir,xdir);
rtcrds = [xdir;ydir;zdir]*crd(elmnts(nels,:),:)';
```

The element in the **uvw** coordinate system, is transformed to a standard right triangle (s. Fig. 3.4) for numerical evaluation of the stiffness matrix integral Eq. (3.40) using Gauss quadrature.

```
ptse=[2/3 1/6 1/6]; ptsn=[1/6 2/3 1/6]; %Gauss points
wts=[1/6 1/6 1/6]; %Weights
```

A matrix with the derivatives of the shape functions with respect to the barycentric coordinates is made

$$dNs = \begin{bmatrix} \frac{\partial N_1}{\partial L_1} & \frac{\partial N_2}{\partial L_1} & \frac{\partial N_3}{\partial L_1} \\ \frac{\partial N_1}{\partial L_2} & \frac{\partial N_2}{\partial L_2} & \frac{\partial N_3}{\partial L_2} \end{bmatrix}$$

From the definition of the shape functions Eq. (3.42) the derivatives can be obtained as

```
dNs = [1 0 -1
       0 1 -1];
```

The Jacobian of the transformation from the **uvw** to the barycentric coordinate system is given by,

$$J = \begin{bmatrix} \frac{\partial u}{\partial L_1} & \frac{\partial u}{\partial L_2} \\ \frac{\partial v}{\partial L_1} & \frac{\partial v}{\partial L_2} \end{bmatrix}$$

$$= \begin{bmatrix} u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \end{bmatrix} \begin{bmatrix} \dfrac{\partial N_1}{\partial L_1} & \dfrac{\partial N_1}{\partial L_2} \\ \dfrac{\partial N_2}{\partial L_1} & \dfrac{\partial N_2}{\partial L_2} \\ \dfrac{\partial N_3}{\partial L_1} & \dfrac{\partial N_3}{\partial L_2} \end{bmatrix}$$

The partial derivatives of the barycentric coordinates with respect to the **uvw** coordinate system is obtained by taking the inverse of the Jacobian matrix.

```
Jinv = inv(J);
```

Suitably arranged inverse Jacobian matrices can be multiplied with the derivatives of the shape functions with respect to the barycentric coordinate system to get the derivatives of the shape functions with respect to the **uvw** coordinate system. This can further be used to find the strains.

```
Jinv_4x4 = [Jinv zeros(2,2)
            zeros(2,2) Jinv];
dN = zeros(4,6);
dN(1:2,1:2:end) = dNs;
dN(3:4,2:2:end) = dNs;
strain = difftostrn*Jinv_4x4*dN;
```

The stresses can be obtained from the strains by multiplying it with the material matrix `alpha` (Eq. (3.30)) which can further be multiplied with the strains to get the stiffness matrix integrand (Eq. (3.33)). The weighted integrand is added over all gauss points to obtain the elemental stiffness matrix. The differential volume in the **uvw** coordinate system is transformed into a differential volume in the barycentric coordinate system $\mathbf{L_1L_2}$ by multiplying with the determinant of the Jacobian matrix

$$dudv = |J|dL_1dL_2$$

```
stima2D = stima2D+wts(npts)*strain'*alpha*strain*det(J);
```

The stiffness matrix is rotated to the **XYZ** coordinate system by pre and post multiplying with the rotation matrix (Eq. (3.34))

$$\begin{aligned} \texttt{R} = [&[\texttt{xdir};\texttt{ydir}] \quad \texttt{zeros}(2,6) \\ &\texttt{zeros}(2,3) \quad [\texttt{xdir};\texttt{ydir}] \quad \texttt{zeros}(2,3) \\ &\texttt{zeros}(2,6) \quad [\texttt{xdir};\texttt{ydir}]]; \end{aligned}$$

```
stima = R'*stima2D*R;
```

The location (row and column number) and value of the each entry of the elemental stiffness matrix in the global stiffness matrix are stored in three different vectors.

$$\texttt{I} = 3 * \texttt{elmnts}(\texttt{nels},[1,1,1,2,2,2,3,3,3]) - ...$$
$$[2,1,0,2,1,0,2,1,0];$$

```
rw(:,nels)=repmat(I',9,1);
cl(:,nels)=reshape(repmat(I,9,1),9*9,1);
vl(:,nels)=reshape(stima,9*9,1);
```

Once all the elemental stiffness matrices are calculated, the sparse global stiffness matrix is assembled.

```
rw=reshape(rw,9*9*size(elmnts,1),1);
cl=reshape(cl,9*9*size(elmnts,1),1);
vl=reshape(vl,9*9*size(elmnts,1),1);

Ksrf = sparse(rw,cl,vl,3*size(crd,1),3*size(crd,1));
```

A.5 Stereo triangulation error

A.5.1 Calculating the covariance matrices

Matlab function *calc_cov.m*

USAGE
`sigma=calc_cov(x_left,x_right,fcalib)`

INPUTS

Variable	Type	Description
x_left	Matrix ($2 \times N_{pts}$)	Landmark locations in the left camera image (in pixels)
x_right	Matrix ($2 \times N_{pts}$)	Landmark locations in the right camera image (in pixels)
fcalib	String	Name of the MAT file containing the rectified camera calibration parameters

OUTPUTS

Variable	Type	Description
sigma	Matrix ($3 \times 3 \times N_{pts}$	Covariance of the stereo triangulated landmarks

Program Description: Calculates the covariance of stereo triangulated landmarks from the landmark locations in the left and right camera images and the camera calibration parameters.

All the camera calibration parameters are loaded.

```
load(fcalib);
```

The program loops through all the landmark locations and calculates the covariances using the *gencov* matlab function. The *gencov* function was created by implementing the approximation for covariance (Eqs. (3.47), (3.48)) by using the symbolic math toolbox.

```
n_pts=size(x_left,2);
sigma=zeros(3,3,n_pts);

for npts=1:n_pts
    sigma(:,:,npts) = gencov(x_left(:,npts),x_right(:,npts),...
                             cc_left_new,cc_right_new,fc_left_new,...
                             fc_right_new,norm(T_new));

end
```

`cc_left_new` and `cc_right_new` are the coordinates of the principal point, and `fc_left_new` and `fc_right_new` the focal lengths of the left and right cameras respectively (after rectification). `T_new` is the translation between the camera coordinate systems.

A.5.2 Solving for a deformed shape using stereo error modelling

Matlab program *pnlty_cov_fem.m*

USAGE

```
pnlty_cov_fem(mshfnme,sysfnme,dirfnme,dirvarnme,...
          pnlty,sigma,defmshfnme)
```

INPUTS

Variable	Type	Description
mshfname	String	Name of the binary file containing the undeformed mesh
dirfname	String	Name of the MAT file containing the landmark location and the displacement at each of them. A $N_{lnd} \times 4$ matrix with the first column giving the element number whose center is the landmark location and the subsequent three columns give the displacement (in meters) in the **X**, **Y** and **Z** directions
dirvarname	String	Name of the variable containing the displacements
sysfnme	String	Name of the MAT file containing the stiffness matrix
pnlty	Double	Weight of the penalty term. Θ in Eq. (3.25)
sigma	Matrix ($3 \times 3 \times N_{lnd}$)	The covariances of the stereo triangulated landmarks
defmshfnme	String	Name of the file to which the deformed mesh should be written to

OUTPUTS

NONE

Program Description: The program is very similar to *pnltyfem.m* except for how the weighting matrix [**S**] (Eq. (3.25)) is constructed. Instead of it being an identity matrix, the matrix is made such that the inverse covariances are on the diagonal (Eq. (3.51))

```
Srw = reshape(repmat([3*(1:n_dir)-2;3*(1:n_dir)-1;3*(1:n_dir)],3,...
                     1),9*n_dir,1);

Scl = reshape(repmat(1:3*n_dir,3,1),9*n_dir,1);
Sval = zeros(9*n_dir,1);

for nster=1:n_dir
    isig = inv(sigma(:,:,nster));
    Sval(9*(nster-1)+1:9*nster) = reshape(isig,9,1);
end
```