

Novel approaches to statistical shape modelling of bone

by
Angelo Tardugno

A thesis submitted in fulfilment of requirements for the degrees of:
Doctor of Philosophy & Diploma of Imperial College

Department of Bioengineering
Imperial College London

2010

How I need a drink, alcoholic of course, after the tough
 chapters involving *femoral modelling*. One is, yes,
adequate even enough to induce some fun and pleasure
 for an instant, miserably brief.

3.1415926535897932384626433832795

Abstract

The femur is the longest bone in the human body and serves the important purposes of load-bearing and allowing bipedal locomotion.

Accurate modelling of the variation in shape within the healthy adult population can be useful for a variety of applications: from the mere anatomical description of its features, in order to better understand its function, to more complex tasks such as pathology detection or surgical planning.

Statistical Shape Modelling (SSM) is a well-established technique that enables to capture the variability within a set of training shapes and describes it with a reduced set of variables.

The aim of this thesis is to evaluate the performance of a SSM based on a point cloud representation of shape, and introduce and test subsequent improvements to the modelling process that can increase its clinical relevance and scope of application.

The standard approach to SSM employs a dimension-reducing technique, generally by means of Principal Component Analysis (PCA). However, this approach favours the compactness of the model, thus not focusing on other aspects that may be more relevant to clinical practice.

Although rotation of the principal components is commonly performed as a post-processing step in statistical analysis involving PCA, it is not routinely applied in SSM.

By applying this class of rotation, the components' effects are more localised, allowing a better interpretation, understanding and classification of

pathological deformities.

Among other possible representations, the Medial Axis Transform (MAT) could offer a further insight into shape modelling, since it allows the information about thickness to be decoupled from the rest of the shape.

SSMs based on this representation can lead to a different perspective on the understanding of femoral anatomy and function, and can also enable the reconstruction of the complete anatomy starting from a reduced set of features, with diverse applications in the fields of surgical planning, forensic science and paleontology.

Acknowledgements

I would like to thank my supervisor, Professor Anthony M. J. Bull, for his support and guidance during the last few years. His passion and enthusiasm have certainly been a source of great motivation.

Many thanks to YuHui Yang for her invaluable help, especially during the first months.

I want to express my special gratitude to Alessandra Carriero, who has always been available both for technical and “non-technical” support.

Many thanks to Alia and Wei, with whom I shared all the excitement of starting a PhD; I am sure things would have been more difficult without them around.

Thanks to all the PhD students I have met in the musculoskeletal group, especially Clarice, Spyros, Andy and Dom, for the working-time spent together, and all the happy moments. I extremely enjoyed working with you all.

Thanks to Iris, Johnny, Colin, Elpida, Andy, Tom, and Thomas. Even though I met you in different circumstances, you have all made my life in London really memorable.

I’d like to say “thank you” to Angela, who has been more like a sister than a simple friend.

I would also like to thank all my Italian friends: Gianluca, Anna, Fra, Cesco, Teo, Matteo, Olga, Federico, Silvia, Laura, Lissi, Floriano and Francesca. I am glad I had the chance to meet you and I know you will always be there

to support me and cheer me up whenever I need it.

My immense gratitude goes to my parents and my brother: you have always been a huge inspiration for me and without your love and help none of this could have been possible.

Contents

Abstract	3
Acknowledgements	5
Contents	7
List of Figures	11
List of Tables	17
Nomenclature	19
Quick Access Nomenclature	21
1 Introduction	23
1.1 Contributions	26
1.2 Thesis Outline	27
2 Femoral Anatomy and Description	29
2.1 Femoral Growth: Endochondral Ossification	32
2.2 Femoral Deformities	34
2.3 Femoral Fractures	38
3 Statistical Shape Modelling	41
3.1 Descriptors of Shape	42

Contents

3.1.1	Point cloud representation	42
3.2	Rigid Registration	47
3.3	Scaling	49
3.4	Feature Matching	51
3.5	Modelling	55
4	Statistical Shape Model of Cloud of Points	63
4.1	Introduction	64
4.2	Statistical Shape Model	65
4.2.1	Shape Modelling	65
4.2.2	Rotation of the PCs	72
4.3	Measurement Method	77
4.3.1	Neck Axis	86
4.3.2	Bicondylar angle	88
4.3.3	Head Radius	88
4.4	Comparison with previous studies	89
4.5	Results	92
5	Medial Axis Transform	113
5.1	Approximation of the MA from the Voronoi Graph	120
5.2	Matching Features on the MAs	121
5.3	Results	128
5.3.1	Medial Axis Transform Model	128
5.3.2	Medial Axis Model	131
5.3.3	Radius Model	151
5.4	Rotated Models	156
5.4.1	Rotated MAT Model	157
5.4.2	Rotated MA Model	163
5.4.3	Rotated Radius Model	167
5.5	Discussion	171

6	Reconstruction of Femoral Morphology	175
6.1	Introduction	176
6.2	Methods	177
6.3	Results	186
6.4	Discussion	194
7	Discussion	197
7.1	Thesis conclusions	197
7.2	Further Considerations	201
7.3	Future Work	205
Appendix:		
A	Muscles Originating or Inserting on the Femur	207
A.1	The hip and gluteus muscles: the inner hip muscles	211
A.2	The hip and gluteus muscles: the outer hip muscles	212
A.3	The Anterior Thigh Muscles: the adductor group	215
A.4	The Anterior Thigh Muscles: the extensor group	216
A.5	The Posterior Thigh Muscles: the flexor group	218
A.6	The Leg Muscles: the posterior compartment	218
B	Code	221
B.1	Reconstruction of Shape from MAT	221
B.2	Medial Axis Computation	224
B.2.1	Computing Normal to Points	224
B.2.2	Compute Medial Axis Transform	227
B.2.3	MAT Simplification	234
B.3	Anatomical Measurements	241
	References	269

Contents

List of Figures

2.1	Femoral Anatomy	31
2.2	Long Bone Development	33
2.3	Example of femoral shaft fracture	40
3.1	Point cloud representation of a human femur	43
3.2	Medial representation of cylindrical shapes	50
3.3	PCA performed on a two-points dataset	59
3.4	PCA with reduced number of examples	61
4.1	Mean shape of the point cloud representation	68
4.2	Scree Graph of Standard SSM	69
4.3	Histograms of the projections of the data-set on the 1 st (a) and 2 nd (b) PCs	71
4.4	Estimate of the probability density functions in PC space (1 of 2)	72
4.5	Estimate of the probability density functions in PC space (2 of 2)	73
4.6	Redistribution of variability after PCs rotation	76
4.7	Neck Shaft Angle	78
4.8	Version Angle	78
4.9	Bicondylar Angle	79
4.10	Head Radius	80
4.11	Intercondylar Distance	80

List of Figures

4.12	Heuristic method to determine anterior-posterior direction . . .	83
4.13	Iterative algorithm to determine the most posterior condylar points	85
4.14	Assignment of Coordinate System to femoral shape	87
4.15	Selection of Neck Area: Step 1	88
4.16	Selection of Neck Area: Step 2	89
4.17	Bicondylar angle	90
4.18	Transverse view of the effects of the first 3 PCs on the shape (standard approach)	93
4.19	Coronal and sagittal views of the effects of the first 3 PCs on the shape (standard approach)	94
4.20	Effects of PCs (1-2) on morphological measures (standard ap- proach)	95
4.21	Effects of PCs (3-4) on morphological measures (standard ap- proach)	96
4.22	Effects of PCs (5-6) on morphological measures (standard ap- proach)	97
4.23	Effects of PCs (7-8) on morphological measures (standard ap- proach)	98
4.24	Relative effects of components on measures (standard approach)	99
4.25	Trend of some parameters in pathology	100
4.26	Effect of Orthomax rotations on PC	102
4.27	Localisation of effects of PC	103
4.28	“Heatmap” of loadings distribution before and after rotation .	104
4.29	Transverse view of the effects of the first 3 PCs on the shape (rotated model)	105
4.30	Coronal and sagittal views of the effects of the first 3 PCs on the shape (rotated model)	106
4.31	Effects of PCs (1-2) on morphological measures (orthomax approach)	107

List of Figures

4.32	Effects of PCs (3-4) on morphological measures (orthomax approach)	108
4.33	Effects of PCs (5-6) on morphological measures (orthomax approach)	109
4.34	Effects of PCs (7-8) on morphological measures (orthomax approach)	110
4.35	Relative effects of components on measures (orthomax approach)	111
5.1	Example of internal and external Medial Axis	115
5.2	Examples of Shapes and Medial Axis	116
5.3	Example of Voronoi Graph	117
5.4	Example of Delaunay Triangle	118
5.5	Approximation of MA from sampled shape	120
5.6	Angle Condition for computing the MA	122
5.7	Metric to identify intersecting spheres	123
5.8	Dendrogram relative to an agglomerative clustering process . .	125
5.9	Mean shape computed from mismatched femurs	127
5.10	Scree graph of MAT model	129
5.11	Effect of the PC on the shape (SSM of the MATs)	131
5.12	Effect of the MAT model PCs on shape (1 of 3)	132
5.13	Effect of the MAT model PCs on shape (2 of 3)	133
5.14	Effect of the MAT model PCs on shape (3 of 3)	134
5.15	Effects of the PCs of the MAT model (1-2) on the morphological measures	135
5.16	Effects of the PCs of the MAT model (3-4) on the morphological measures	136
5.17	Effects of the PCs of the MAT model (5-6) on the morphological measures	137
5.18	Effects of the PCs of the MAT model (7-8) on the morphological measures	138
5.19	Scree Graph of MA model	140

List of Figures

5.20	Comparison between MA and MAT model	141
5.21	Effect of the MA model PCs on shape (1 of 3)	142
5.22	Effect of the MA model PCs on shape (2 of 3)	143
5.23	Effect of the MA model PCs on shape (3 of 3)	144
5.24	Effects of the principal component (1-2) on the morphological measures (MA model)	145
5.25	Effects of the principal component (3-4) on the morphological measures (MA model)	146
5.26	Effects of the principal component (5-6) on the morphological measures (MA model)	147
5.27	Effects of the principal component (7-8) on the morphological measures (MA model)	148
5.28	Comparison between MA and MAT model based on the measures	149
5.29	Scree Graph of the radius model	152
5.30	Effect of the radius model PCs on shape (1 of 3)	153
5.31	Effect of the radius model PCs on shape (2 of 3)	154
5.32	Effect of the radius-model PCs on shape (3 of 3)	155
5.33	Effect of the rotated MAT model PCs on shape (1 of 3)	158
5.34	Effect of the rotated MAT model PCs on shape (2 of 3)	159
5.35	Effect of the rotated MAT model PCs on shape (3 of 3)	160
5.36	Effect of the rotated MA model PCs on shape (1 of 3)	164
5.37	Effect of the rotated MA model PCs on shape (2 of 3)	165
5.38	Effect of the rotated MA model PCs on shape (3 of 3)	166
5.39	Effect of the rotated radius model PCs on shape (1 of 3) . . .	168
5.40	Effect of the rotated radius model PCs on shape (2 of 3) . . .	169
5.41	Effect of the rotated radius model PCs on shape (3 of 3) . . .	170
6.1	Box plot representation of a normal distribution	183
6.2	Main properties of a statistical predictor	185
6.3	Selection of points for reconstruction	187
6.4	Reconstructed and original shapes	188

List of Figures

6.5	Comparison between the original shape, the prediction and the mean-shape	189
6.6	Prediction of NSA	190
6.7	Prediction of VA	191
6.8	Prediction of BA	191
6.9	Prediction of ICD	192
6.10	Prediction of HR	192
A.1	Anatomical Planes	208
A.2	Movements at the hip joint: Extension/Flexion and Abduction/Adduction	209
A.3	Movements at the hip joint: Internal/External rotations	209
A.4	Thigh muscles	210
A.5	Inner Hip Muscles	211
A.6	Vertically Oriented Outer Hip Muscles	213
A.7	Horizontally-oriented Outer Hip Muscles	214
A.8	Adductor Group Muscles	215
A.9	Hip Extensor Muscles	217
A.10	Posterior Thigh Muscles	219
A.11	Posterior Leg Muscles	220

List of Figures

List of Tables

2.1	Femoral regions and muscular groups	35
4.1	Comparison of geometrical parameters gathered during different studies.	91
4.2	Morphological measure ranges (standard approach)	112
4.3	Morphological measure ranges (orthomax approach)	112
5.1	Qualitative analysis of the MAT model	139
5.2	Qualitative analysis of the MA model	150
5.3	Qualitative analysis of the rotated MAT model	161
5.4	Femoral regions and muscular groups	162
5.5	Qualitative analysis of the rotated MA model	167
5.6	Qualitative analysis of the rotated radius-model	171
6.1	Predictions errors of the mean shape and the model-based prediction	193

List of Tables

Nomenclature

- **SSM**: Statistical Shape Modelling
- **PCA**: Principal Component Analysis
- **PC**: Principal Component
- **MA**: Medial Axis
- **MAT**: Medial Axis Transform
- **NSA**: Neck Shaft Angle, angle between capito-collared axis and shaft axis
- **VA**: Version Angle, angle between projection of the capito-collared axis on the transverse plane and the condylar axis
- **BA**: Bicondylar Angle, angle between condylar axis and shaft axis
- **HR**: Head Radius, radius of the head of the femur
- **ICD**: Inter Condylar Distance, distance between the two most posterior points of each condyle
- **LMD**: Lateral Medial Direction, direction of the coordinate frame defined on the femur going from the lateral to the medial position
- **ISD**: Inferior Superior Direction, direction of the coordinate frame defined on the femur going from the inferior to the superior position

List of Tables

- **PAD:** Posterior Anterior Direction, direction of the coordinate frame defined on the femur going from the posterior to the anterior position
- **CT:** Computed Tomography
- **MRI:** Magnetic Resonance Imaging
- **DDH:** Developmental Dysplasia of the Hip, a congenital or acquired deformation or misalignment of the hip joint

-MAT: *Medial Axis Transform*

- **ICD:** *Inter Condylar Distance*, distance between the two most posterior points of each condyle

- **PAD:** *Posterior Anterior Direction*, direction of the coordinate frame defined on the femur going from the posterior to the anterior position

- **DDH:** *Developmental Dysplasia of the Hip*, a congenital or acquired deformation or misalignment of the hip joint.

List of Tables

Chapter 1

Introduction

The concept of shape is difficult to define unequivocally in words, yet its meaning is clear to almost everybody. It relates to the properties of three-dimensional objects and the relationships of the different parts of the object between themselves.

When it comes to biological structures, shape assumes an important relevance as it is strongly related to function; all the organs in a living organism have a precise shape so that they can accomplish a specific task. The bones in the human skeleton serve different purposes, from supporting body weight to protecting vital organs.

Through the evolutionary process, the shape of bones such as the femur has changed in response to different postures and forms of locomotion and has been optimised for the main purposes of load-bearing and bipedal walking (Shefelbine *et al.*, 2002).

Since each individual is different in terms of height, weight and posture, there are different femoral shapes that should be considered normal. In order to detect pathological abnormalities, the concept of “normality” should be defined.

The introduction of medical imaging techniques allowed for a non-invasive evaluation of bone shape, however, before the introduction of tomography,

only two-dimensional projections of internal structures were available. In order to describe the shape of bones, a set of geometrical parameters (angles and distances) was introduced, thus effectively mapping the shape in a relatively high-dimensional space. This mapping is somehow important, as it allows relationships between the variables to be made easier to understand. However, the choice of the variables to employ originated from the empirical expertise of orthopaedics and not from a mathematical analysis of shape.

Due to their intrinsic two-dimensional nature, these sets of parameters lack the power to capture the complexity of a three-dimensional structure; even though this approach was the best available at the dawn of medical imaging, the introduction of tomographic techniques allowed for a complete three-dimensional reconstruction of structures, hence allowing for a more rigorous description of shape.

The approach that is going to be followed in this thesis is not completely different from the one that doctors initially employed; the shape should be mapped onto a higher-dimensional space, in the hope that the relationships between the variables comprising this space would be linear and therefore easier to interpret.

The main idea is to try to limit the amount of preprocessing performed by an expert, and use statistical techniques to capture the features of the shape modelled. In order to achieve this, the shape must be represented by a set of basic variables that just try to represent the shape as a whole, not discarding any kind of information.

Points sampling the surface of the object have been used by many authors for the purpose of model computation. However, due to the high number of variables, the statistical significance of the model might be questioned as, inevitably, the number of shapes available for the modelling process is limited. Thus, a trade-off between the number of variables and accuracy of the representation should be found.

A first model was built based on the points lying on the surface of a set

of femurs taken from a healthy human being. Different representations of shape were also considered and the so-called medial-axis transform (MAT) was then employed for computation of an alternative model.

Regardless of the type of representation, in order to reduce the dimensionality of the space and capture the meaningful relationships between shapes, a standard dimensionality-reduction technique called Principal Component Analysis (PCA) was employed. The advantage of this technique consists of grouping together those variables that have correlated variations across the data-set; shape variability can therefore be modelled in a compact way.

Despite being optimal in terms of compactness, this approach leads to models that cannot be applied to a pathology-detection scenario in a straightforward way. Since the model is based on healthy subjects, the set of Principal Components (PCs), i.e. the reduced set of variables introduced by PCA, is optimised for the description of healthy shape.

Pathological deformities, excluding those that alter considerably the shape of the bones like tumours or serious traumatic events, are usually characterised by a *local* change in shape that affects well-defined structures and alters the relative position or size of them compared to the rest of the shape.

Starting from a standard SSM, these local changes can be described by grouping together many PCs; however this might need supervision by an expert, as a description involving many parameters is generally not easy to understand. A different approach, based on a particular class of rotations (orthomax rotations), can maintain the compactness of the model and favour a local description and enable an effective description of pathological deformity without expert supervision and arbitrary definition of ranges of normality.

It should be said that the standard PCA model and its rotated counterpart cannot be compared directly in terms of performances, since they respond to different purposes. The standard model naturally describes those aspects of shape that characterise healthy femurs, whilst the rotation effec-

tively identifies important structures comprising the shape, thus introducing a new set of measures that is best suited for the description of local bone deformations.

Whenever the deformation affecting the bone is so severe to disrupt the shape of the bone completely (like in the presence of tumours or fractures), trying to describe the shape using the PCs or the rotated PCs computed from a healthy training set could not be possible. Moreover, since in these cases more invasive procedures might be needed to restore functionality, the aim would be to assess the shape prior to the traumatic event or tumour formation.

One possible approach would be to consider the contralateral limb, but since perfect symmetry cannot be guaranteed, predicting the whole shape from those parts of the shape that can be considered normal can be a viable possibility.

Mathematically, this is possible as the SSM is nothing but a representation of the shape in terms of an over-determined system; provided there is a sufficient portion of the femur that can be used for the prediction, a good estimate of the real shape could be obtained.

Reconstruction of the whole shape can be used either to plan reconstructive or implant surgery or to assess the severity of fractures in order to determine the optimal procedure to employ.

1.1 Contributions

This thesis introduces some novel techniques to the description of femoral anatomy.

These include the first SSM based on a medial representation of a three-dimensional shape; despite medial representations have been used in previous works (Styner, 2001), their role was limited to complete the description provided by a standard SSM based on a point cloud representation. Since

traditionally the properties of long bones are defined by axes of symmetry, the medial representation seems the most appropriate approach for describing their shape. This thesis proves the viability of the computation of a SSM based on a medial representation, thus possibly allowing an improved understanding of function and pathology.

Another important contribution consists in the use of orthomax rotations for a model describing a three-dimensional shape; previous work on this field was focused on two-dimensional shapes (or stacks of two dimensional shapes). A clear definition of the statistical limits for this alternative modelling is also provided, allowing the employment of this model as an automatic tool for pathology detection.

The possibility of predicting the complete shape of long bones from fragment is also investigated; apart from medical-related applications in surgical planning and fracture treatment, such a prediction can be employed in fields such as forensic science and paleontology, where the problems of reconstruction of bones from fragments is particularly relevant.

1.2 Thesis Outline

Chapter 2 focuses on the main aspects of femoral anatomy and bone development. A general background about femoral fractures is also presented, in order to understand the motivations for a tool to reconstruct femoral anatomy from partial information. A review of the muscles which are connected to the femur is given in Appendix A.

In Chapter 3, the general theory of SSMs is presented, starting from the different options available for shape representation and then introducing all the different steps needed to perform the actual model computation. Among the techniques to represent shape, the point cloud representation and the medial axis transform are those that are employed in Chapters 4 and 5, respectively.

Chapter 4 also introduces the concept of orthomax rotation and compares the results of a standard model and its rotated counterpart. These results will be analysed using a semi-automatic tool developed to compute geometrical parameters that are used in clinical practice.

An alternative approach to shape representation, based on the Medial Axis Transform, is presented in Chapter 5 and SSMs derived from this representation are evaluated and compared to the standard approach.

In Chapter 6 an application of SSM for shape reconstruction is tested in a femoral-fracture scenario.

Chapter 7 draws conclusions from this work and considers future developments and possible applications of the tools and models that were computed.

Chapter 2

Femoral Anatomy and Description

In order to understand completely any kind of model describing femoral shape, some knowledge about its anatomy is essential. The main anatomical features of the femur will now be introduced, an overview of the muscles that are attached to the femur, and are likely to influence its shape, is covered in Appendix A. By considering the developmental process by which long bones are formed (endochondral ossification) together with the lines of action of the relevant muscles, it is possible to have a clearer picture of the process by which femoral shape develops, hence understanding its connection with function.

The femur is the bone of the thigh, its name originates from Latin where the expression “os femoris” actually means bone of the thigh, an anterior and a posterior view of the femur is shown in Fig. 2.1.

It is the longest bone in the body and its proximal end is formed by the head and the neck and two structures called the greater and lesser trochanters, which provide the points of attachment of the major muscles involved in the movement of the hip joint.

The greater trochanter has a ridge on its anterolateral surface that pro-

vides the insertion point for the gluteus minimus and a corresponding posterior ridge where the gluteus medius inserts.

Other muscles inserting in this region are the obturator interni and the associated gemelli muscles.

The lesser trochanter is smaller than the greater trochanter and has a conical shape. The combined tendons of the iliacus and psoas major attach to the femur in this region.

The head has a spherical shape and it articulates with the acetabulum of the pelvic bone, the concave surface where the head of the femur meets with the pelvis to form the hip joint. Since the head of the femur is longer than the depth of the cavity of the acetabulum, the joint is not stable. Stability is provided by a fibrocartilage lip (labrum), which effectively increases the depth of the acetabulum, and by the action of specific muscular groups, which are described in Appendix A.

The neck is a pyramidal structure that connects the head to the shaft. The orientation of this structure relative to the shaft influences the range of movement of the hip joint, thus making this angle an important parameter in measuring functionality.

The shaft of the femur is not perpendicular to the articulating plane of the knee, as the distal shaft is closer to the mid-line of the body than the lateral shaft.

The posterior shaft shows a raised ridge called *linea aspera* that represents a major site for muscular attachment.

The distal part of the femur is characterised by two large structures, the condyles, which articulate with the tibia. The two condyles are separated in the posterior region by the intercondylar fossa whilst anteriorly they are joined and articulate with the patella, commonly known as knee-cap.

The shape of the femur is affected by many factors; the muscles originating and inserting on the femur play an important role in determining the shape of the thigh bone. A complete description of these muscles is given in

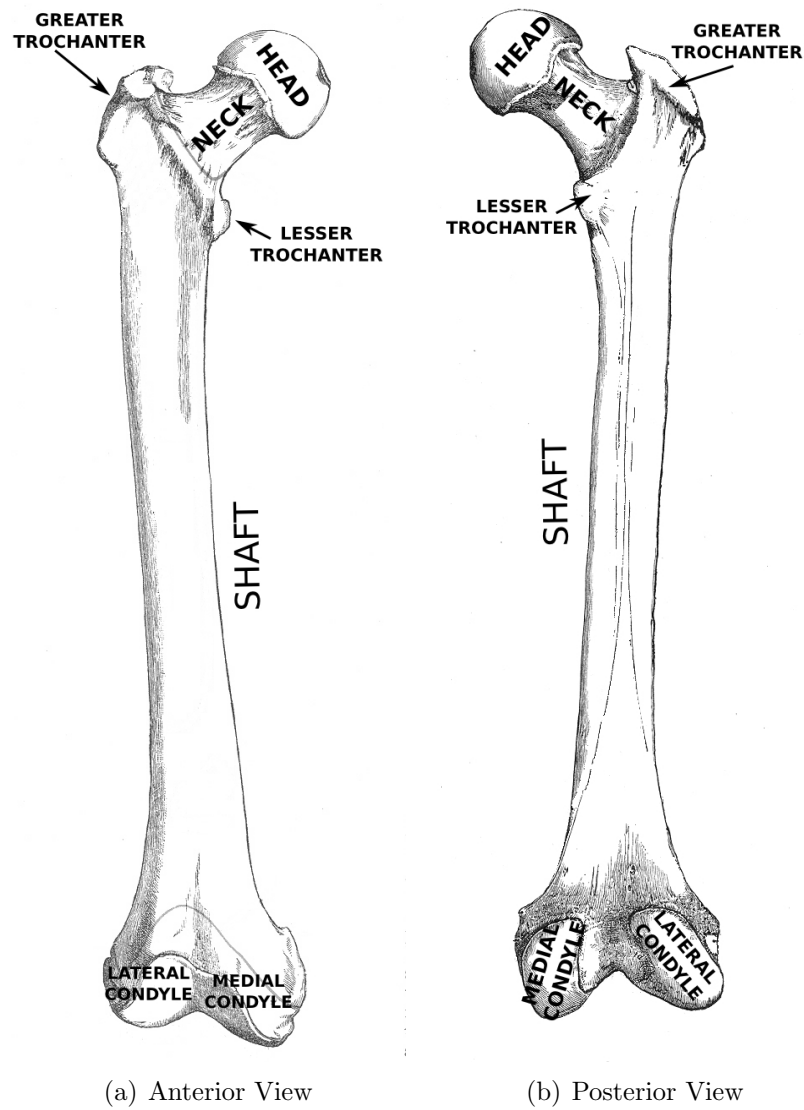


Figure 2.1: Femoral Anatomy (modified from Gray)

Appendix A.

Although imbalances in these muscles can alter bone shape also in adulthood, their effect is greater during the early stages of life, as developing bone has a particular configuration that makes it more susceptible to shape changes due to external loads. A brief overview of the process of bone growth is given in the next section; the focus is on those aspects that are more relevant to the purpose of this work rather than on the complete description of the process.

2.1 Femoral Growth: Endochondral Ossification

Bone develops in the body from the differentiation of pluri-potential tissues. Different sites experience distinct types of differentiation: long bones, including the femur, are created by conversion of a cartilage anlage (template) into well-vascularised bone; this process goes under the name of *endochondral ossification*, which means ossification within the cartilage.

The early stages of this process resemble the ossification of flat bones, such as the scapula; a discrete bone collar forms in the region of the mid-shaft of the long bone and a nutrient artery develops to provide the means for the bone cells to invade the cartilage anlage (Fig. 2.2).

In the femur, this bony collar appears at weeks 7-8 in uterus (Scheuer & Black, 2004). The proper endochondral ossification process occurs when the cells in the middle of the cartilaginous region begin to enlarge and characteristic structures called growth plates are formed. Bone comprises both a cellular component (osteoblasts, osteocytes and osteoclasts) and an amorphous component, called the matrix. Since this matrix is rigid, bone growth cannot happen by simple cell proliferation. The longitudinal growth and diametric expansion of long bones is due to particular structures called growth plates.

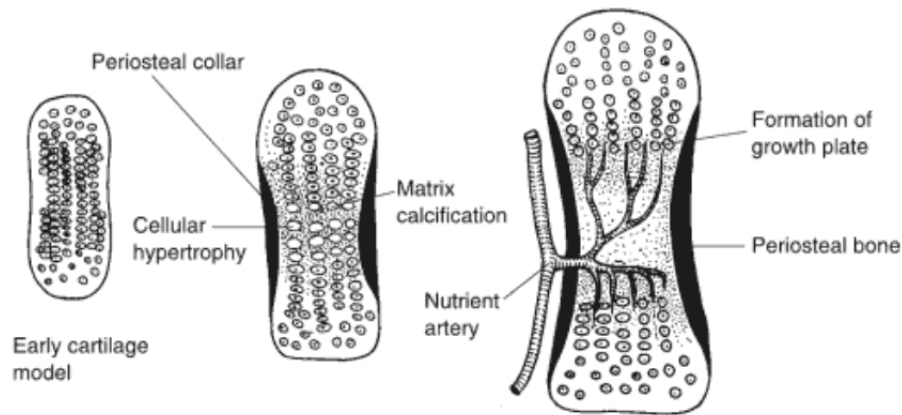


Figure 2.2: Early stages of Long Bones Development (Scheuer & Black, 2004)

The first site of ossification occurs in the primary centre of ossification, which is located in the middle of the shaft. In this region the chondrocytes (cartilage cells) start to grow and produce substances that will promote the creation of the matrix. Meanwhile osteoprogenitor cells are brought in the cartilage which will then lead to the formation of trabecular bone, that is a kind of woven bone found in the internal part of long bones especially in the distal and proximal ends.

At the time of birth secondary ossification centres develop at each end of the femur and ossification continues in the growth plates that are located between the secondary ossification centres and the primary ossification centres.

In the growth plates chondrocytes will keep on growing and eventually they will be replaced by calcified bone. This process will result in an increase in length of the bone.

At birth the femur has two growth plates, at 3-4 years the proximal growth plate separates into two distinct plates and a new ossification centre is formed within the femoral head.

The presence of these growth plates and the action of the leg muscles will affect the final shape of the femur.

It is not surprising that the geometry of the femur changes significantly in response to walking. The effect of these changes depends on the kind of stimuli provided. If for instance the muscles are not fully developed, this will result in a deformed femur. Jonkers & Shefelbine (2010) show that abnormal loading conditions can lead to significant changes in femoral shape over a period of five months.

One of the aims of this thesis was to provide a model that could describe differences in shape corresponding to the different growth plates and be able to distinguish between the different muscle groups acting on the bone. Such a model could describe the effects of primary or secondary (for instance due to nerve damage) muscle abnormalities, which are considered to be the underlying cause of bone deformities.

Table 2.1 groups the different muscles according to the structure they are most likely to influence.

The lines of action of these muscles are not constant, but change according to the position of the joints in the lower limb. However, since their insertion/origin regions are fixed, they will likely influence femoral shape in the areas shown in the table.

2.2 Femoral Deformities

The stimuli from normal musculature are necessary to attain normal bone formation; whenever muscles are not fully-functional during childhood, bones cannot develop properly and shape abnormalities occur.

Since in adulthood the shape of bones has already been determined, a radical change in shape is no longer possible although bone can change its architecture, thus leading to other kinds of pathologies that can eventually result in fractures.

STRUCTURE	MUSCLE
GREATER TROCHANTER	M. gluteus medius
	M. gluteus minimus
	M. piriformis
	M. vastus lateralis
	M. obturatorius internus
	Ms. gemelli
	M. quadratus femoralis
LESSER TROCHANTER	M. iliacus
	M. psoas major
	M. psoas minor
SHAFT (proximal third)	M. gluteus maximus
	M. pectineus
	M. adductor minimus
	M. adductor brevis
SHAFT (middle third)	M. adductor magnus
	M. adductor longus
	M. vastus medialis
	M. vastus lateralis
	M. vastus intermedius
MEDIAL CONDYLE	M. biceps femoris
	M. adductor magnus
	M. triceps surae
	M. plantaris
LATERAL CONDYLE	M. gracilis
	M. triceps surae
	M. tensor fasciae latae (Tractus iliotibialis)
	M. popliteus

Table 2.1: Regions of the femur and principal muscles that are attached or in contact with these

In order to detect deformities, some geometrical measures are usually evaluated; if these measures do not fall within the range of normality then the bone will be considered abnormal. However, it is difficult to define this range, especially since there are different definitions of the same measures that often cannot be compared and sometimes the technique used for the evaluation is not reliable (Murphy *et al.*, 1987).

Usually, in examining the shape of the femur, the focus is on the angular position of the neck-head relative to axes passing through the condyles and the shaft. This comes as no surprise, since a correct alignment of the neck-head structure enables the load to be transferred correctly to the leg and also allows for a complete range of motion of the limb. From a mechanical point of view, the mechanical axis of the femur, defined as the line passing through the centre of the femoral head and the centre of the knee (see Moreland *et al.* (1987)), should be aligned with the axis of the tibia allowing an optimal load transfer.

The angles that are usually considered to assess femoral geometry are:

- *Neck Shaft Angle (NSA)*: This angle measures the inclination of the head and neck of the femur relative to the shaft axis. Some authors consider this angle to be the angle formed by the projections of the neck and shaft axes onto the coronal plane. However, as the change of version angle leads to a change in neck shaft angle if this definition is used, the 3D angle between the two axes was considered instead, since it is not affected by this problem.
- *Version Angle (VA)*: This measures the position of the neck axis relative to the medial lateral direction in the transverse plane.
- *Bicondylar Angle (BA)*: This parameter measures the inclination of the shaft axis relative to a line perpendicular to the line passing through the most inferior points of the femoral condyles.

The main angular deformities involving the femur that are associated with these angles are:

- *Coxa Vara / Valga*, this deformity affects the head and neck of the femur. If the neck-shaft angle is lower/higher than normal then the condition is called coxa vara/valga. This kind of deformity can be either congenital, meaning that it developed during fetal life, developmental or result from traumatic events. If the condition affects one limb only, the two legs will have different lengths thus resulting in limping and other secondary effects.
- *Version Abnormalities*, if the VA has values greater or lower than normal. When VA is very small or even negative the femur will be retroverted and will result in altered gait.

Angular abnormalities are not the only kind of deformities that can be found in bones: for instance in cerebral palsy bones appear gracile and sometimes exhibit an enlargement of the extremities (Miller, 2005). In order to evaluate these kinds of deformities various measures can be introduced. Since this study was not addressing any particular pathology, two measures were considered that relate to the size of both the proximal and distal ends of the femur:

- *Head Radius (HR)*, that is the radius of the best fitting sphere on the articular surface of the head.
- *Intercondylar distance (ID)*, that measures the distance between the two most posterior points on each condyle.

Although other measures might be introduced, those introduced here were deemed sufficient for the purposes of this thesis.

2.3 Femoral Fractures

Femoral geometry can be disrupted not only because the femur had been subjected to abnormal loading during growth; traumatic events can lead to fractures of the bone that need to be treated either by means of a cast splintage (or other forms of external fixations) or surgically.

In the case of a complete fracture, in which bone fragments separate completely, surgery is needed to place the fragments together to restore the original geometry. As a consequence of a complete fracture the segments are displaced, due to the force of the injury but also due to the action of gravity or the muscles attached to the bone.

To assess the pattern of the fracture, usually x-rays are taken in the region of interest. CT and MRI are also useful and 3D reconstructed images are recommended.

It is important to evaluate in a quantitative way how characteristic structures of the bone are displaced from their original position, or relative to other structures after the bone is fractured. This information allows for better planning of the surgical fixation, thus improving the outcome.

However, the whole shape of the bone would be needed to achieve accurate measurements. Despite some work that has suggested that in long bones proximal features relate to distal features, thus allowing the measurement of standard morphological parameters (Robertson *et al.*, 2000), these results are only valid in a weak statistical sense, and therefore not very reliable.

Statistical Shape Models could provide a way of reconstructing the whole geometry starting from an incomplete shape (using the fractured region to reconstruct the shape is obviously not possible). However, standard point cloud representations can be impractical for this purpose as will be seen in Chapter 6, where models based on different representations are tested in order to assess the feasibility of this kind of prediction.

This predictive tool might prove useful in preventing and assessing malunion of fractures, which is one of the possible delayed complications (Solomon

et al., 2005); a fracture is said to be malunited whenever the fragments are joined in an unsatisfactory position or the stiffness of the bone is insufficient to transmit physiological loads.

Possible causes of this complication are a bad initial reduction of the bone or failure to maintain this reduction during the healing process; hence the importance of obtaining a reduction as accurate as possible.

However, the assessment of malunion is not straightforward: rotational deformities in long bones are usually missed, unless a comparison with the unfractured bone is performed. Such a comparison is not always possible, as both the bones might be fractured. Moreover, some studies suggest a significant difference between left and right bones (Da Silva *et al.*, 2003), thus making this comparison not always reliable.

Since the comparison with the contralateral limb is not possible, a statistical model could be the most viable technique to estimate the original bone geometry prior to fracture.

Femoral fractures are usually classified according to the region of occurrence.

One possible classification is the following (Solomon *et al.*, 2005):

- Fracture of the Femoral Neck
- Intertrochanteric Fractures
- Subtrochanteric Fractures
- Femoral Shaft Fractures
- Supercondylar Fractures
- Condylar Fractures

An example of a femoral shaft fracture is shown in Fig. 2.3; here the fracture has healed in a slightly tilted direction. However it might be difficult to evaluate whether corrective surgery would be needed to correct the alignment based solely on x-ray images.

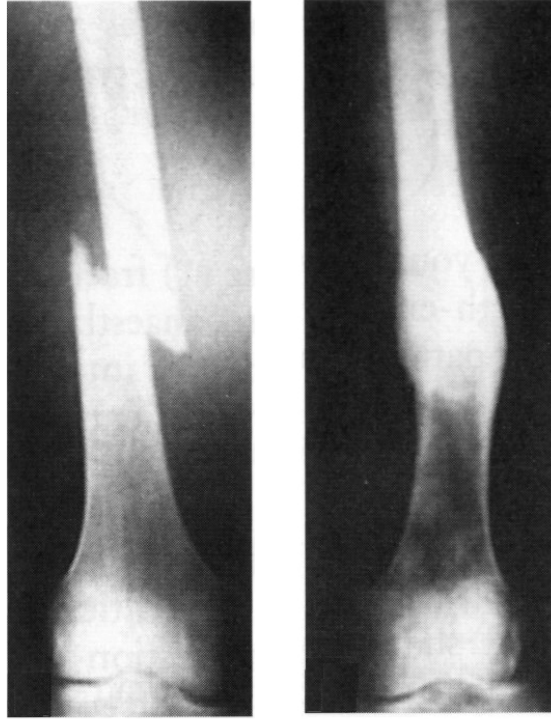


Figure 2.3: Femoral shaft fracture before and after the healing process has taken place (Solomon *et al.*, 2005). Although the misalignment of the healed bone is visible, it is difficult to quantify its entity.

The complete description of femoral fractures falls outside the scope of this work. However, regardless of the location of the fracture, the description of the geometry of the femur before the traumatic event can help in assessing the severity of the fracture. Furthermore this can assist clinicians in the choice of the optimal treatment and in monitoring the healing process in a more quantitative way, thus avoiding possible complications.

Chapter 3

Statistical Shape Modelling

Statistics is the science of making effective use of numerical data representing a population, i.e. a homogeneous group of individuals sharing common characteristics.

When it comes to shape modelling, it is possible to use statistics in order to capture the differences in shape that are observed within the data-set under consideration with a reduced, yet meaningful, set of variables.

SSMs have been employed to describe a variety of shapes, in addition to applications in the recognition of faces and movements (Quan *et al.*, 2009; Ahmad *et al.*, 1997).

Davies *et al.* (2003) computed a SSM representing the anatomy of the hippocampus, a major component of the human brain. This same structure had been studied previously by Shenton *et al.* (2002), and the computed SSM was used to investigate shape abnormalities in schizophrenia.

Among other biological structures that have been successfully described using SSM there are the liver (Lamecker *et al.*, 2002), the pelvic bone (Lamecker *et al.*, 2004), and the humerus (Yang *et al.*, 2006).

In order to build a model of shape that employs the powerful tools of statistics, the first important step is to decide what kind of descriptor of shape should be employed to represent each instance of shape within the

data-set.

Different approaches to describing shape are available and usually they are motivated by what is easily available. The evolution of modern technologies to acquire representation of 3D objects have increased the popularity of the so-called point cloud representation of shape, simply consisting in point sampling the surface enclosing the object. However, this representation is intrinsically redundant since spatially-close points are seldom independent of each other. Other representations, which can reduce these redundancies to different extents, are available.

A brief overview of the most common representations is given in Section 3.1.

3.1 Descriptors of Shape

3.1.1 Point cloud representation

Among the possible representations of three-dimensional shapes, the point cloud representation is one of the most widely employed. It consists of a set of points sampling the surface enclosing the volume occupied by a given object; no information about the connectivity between the points is generally included in this representation, although there are some algorithms that allow the reconstruction of a “watertight” surface, i.e. including the connectivity information, based on the points or tomographic images (Lorensen & Cline, 1987; Amenta & Bern, 1999). An example of a point cloud representation of a human femur is shown in Fig. 3.1.

Despite being effective in representing the shape of an object, the point cloud representation suffers from the limitation of being “local”. Each point is completely independent from the others and there is no intrinsic limitation in the extent of deformation that will disrupt the shape.

This kind of representation was employed successfully in the computation of Statistical Shape Models (SSMs) of a set of homogeneous shapes (Yang

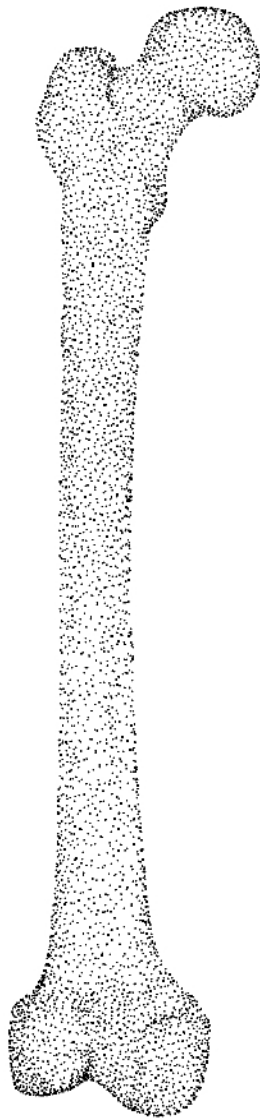


Figure 3.1: Point cloud representation of a human femur

et al., 2006). In such a scenario, the modelling process itself provides a set of statistical constraints that limits the range of movement a single point can experience, thus effectively representing the shape. Assuming the shapes follow a normal distribution, the SSM would define a set of vectors for each point on the shape. These vectors will represent the allowed directions of movement for each point and the amount of movement is determined by the standard deviation associated with the vector itself.

However, since no *a priori* constraint is introduced between the points, the degrees of freedom of this representation are too many to lead to very strong statistical conclusions, unless the number of available samples is relatively high. As the availability of samples is usually limited by cost or ethical-issues, other representations that limit the degrees of freedom of the problem might prove a viable alternative.

Another possible description of 3D shapes involves a parametric representation. When the shape has a spherical topology there are different approaches that can be effectively employed.

Topology is the branch of mathematics that studies those spatial properties that are preserved under continuous deformations.

A 3D surface has a spherical topology when it can be deformed continuously, i.e. without tearing (creating holes) and glueing (creating one object from two or more) to a spherical shape.

Most biological shapes are spherical in topology, since at a macroscopic level they do not have holes, so the focus is on those techniques to represent this particular class of shapes.

A parametric description of a 3D surface consists of a map, $\mathcal{M}$, of a two-dimensional parameter space (u, v) onto the three-dimensional Euclidean space, as described in Eq.3.1.

$$(u, v) \xrightarrow{\mathcal{M}} R^3 \quad (3.1)$$

Among the possible representations, the ones more suitable for modelling

biological shapes are the following (Styner, 2001):

- *Superquadratics (Bajcsy & Solina, 1987)*: This is a popular approach in describing 3D shapes. When representing biological shape, the only viable approach would employ the superquadratic ellipsoids, $\mathbf{v}$, defined in Eq.3.2.

$$\mathbf{v}(a, a_1, a_2, a_3, \epsilon_1, \epsilon_2, u, v) = a \begin{pmatrix} a_1 C_u^{\epsilon_1} C_v^{\epsilon_2} \\ a_2 C_u^{\epsilon_1} S_v^{\epsilon_2} \\ a_3 S_u^{\epsilon_1} \end{pmatrix} \quad (3.2)$$

The parameters u and v are subjected to the limits $\pi/2 \leq u \leq \pi/2$ and $\pi \leq v < \pi$ and the functions C_w^ϵ and S_w^ϵ are defined in Eq.3.4.

$$C_w^\epsilon = \text{sgn}(\cos(w)) | \cos(w) |^\epsilon \quad (3.3)$$

$$S_w^\epsilon = \text{sgn}(\sin(w)) | \sin(w) |^\epsilon \quad (3.4)$$

where sgn represents the sign function, returning the sign of its argument.

All the parameters $a, a_1, a_2, a_3 > 0$ and they define the aspect ratios whilst the parameters ϵ_1, ϵ_2 define the “squareness” of the superquadratic ellipsoids. This kind of representation seems more suited to CAD than to the description of biological structures; although possible in theory, for shapes like that of a femur, different structures would need to be represented by different superquadratics curves that then need patching, i.e. linking together to assure the smoothness of the resulting shape.

The problem of choosing where to patch these superquadratics is not trivial. Moreover, since this process needs iterating over the whole data-set of shapes it could be time consuming. Thus, although widely

used, this approach does not seem well-suited for the computation of SSMs of bones.

- *Spherical Harmonics (SPHARMs)*: one of the possible approaches to describe the boundaries of an object consists of employing a set of function defined on some space parameters, called basis functions. Among the possible bases, SPHARMs allow shape deformations to be expressed in a compact ways, as stated by Kelemen *et al.* (1999).

The main idea is to represent the coordinates of the points as the sum of harmonic functions up to a certain level of degree: higher degrees would add finer features to the shape, thus allowing for a different level of detail to be considered.

Styner used this representation to compute a model of shape of the amygdala/hippocampus complex of the brain (Styner, 2001). The harmonic of order 0 (corresponding to an ellipsoid) was used to perform the registration of all the shapes in a common coordinate frame, this corresponds to registering the shapes according to their principal axes. Such an approach would not work for the femur as, despite having a clear major principal direction, there is no guarantee that the two other directions would always be consistent throughout all the different instances of shapes (Section 4.3).

Moreover, the approach followed by Styner just uses the SPHARMs' properties for this registration purpose and the simplification of shapes, as the model relies on points sampled on the surface obtained from the SPHARM representation.

Although using the parameters of SPHARMs or the superquadratic ellipsoid representations to build a model could be a viable option, it would suffer from the fact that a change of the parameters would affect the shape as a whole.

However, by simply using points on the surface the model would just be

a point cloud based model as nothing about the initial representation is retained.

Despite being possible representations, the parametric representations in general would not allow for the construction of a SSM. Given the complexity of some geometrical shapes, the reduction in the degree of freedoms introduced by the parametric representation could in fact introduce unwanted non-linearities, thus making the model difficult to understand.

Alongside the model based on the point cloud representation, Styner (2001) introduced the idea of a medial-representation of the shape. However, this medial representation is used as a complement to the standard SSM and not as a representation considered to be sufficient on its own.

A complete review of medial representations is given in Chapter 5, however it is worth noticing that this representation reduces the degrees of freedom by grouping together points on the surface that lie on the same sphere tangent to the shape; the locality of the point-cloud representation is retained but the introduction of weaker (compared to the parametric representation) constraints are introduced so that the statistical strength of the model is increased without any non-linear complication.

3.2 Rigid Registration

In order to investigate changes in morphology within a data-set, the different shapes need to be “comparable”. Since shape representation needs the definition of a coordinate system, the coordinate system must be the same for all the instances of shape considered.

The properties of shape are invariant to rigid transformations, therefore they do not depend on the particular choice of coordinate system employed. Once a particular shape is chosen, the remaining shapes can undergo a rigid transformation to obtain alignment.

A general point $P \in \mathcal{R}^3$ can be represented in the so called homogeneous coordinates $\mathbf{p}^H$, defined in Eq.3.5.

$$\mathbf{p}^H = (x_P \ y_P \ z_P \ 1)^T \quad (3.5)$$

Points represented in this way can undergo a rigid transformation (i.e. a rotation and translation) through the homogeneous transformation matrix $\mathbf{H}$, as Eq. 3.6 shows.

$$\mathbf{p}^t = \mathbf{H}\mathbf{p}^H = \begin{bmatrix} \mathbf{R}_1^H & \mathbf{o}_1^H \\ \mathbf{0}^T & 1 \end{bmatrix} \quad (3.6)$$

The last row of the matrix $\mathbf{H}$ in Eq.3.6 never changes, whilst the sub-matrix $\mathbf{R}_1^H$ represents a rotation and vector $\mathbf{o}_1^H$ a translation.

The Iterative Closest Point (ICP, Besl & McKay (1992)) algorithm tries to optimise this transformation so that the sum of the square errors between corresponding points between the reference shape and each test shape is minimised.

The minimisation is obtained using a gradient descent approach; the gradient of the function to minimise is approximated using the values of the function in the neighbourhood of a starting point, and a new point is then considered in the direction of biggest decrease.

As with any gradient descent method, in order to avoid being trapped in local minima the ICP should initiate from a relatively good starting point; therefore before applying the registration all the shapes need to be adjusted so that they are reasonably aligned.

The first step in the vtk's implementation of the ICP algorithm is to match the centroids of the reference and the test shape examined; then the transformation matrix is changed so to minimise the distance between each point in the test shape to its closest point on the reference shape, until convergence is reached.

3.3 Scaling

The shapes that were examined correspond to adult femurs and therefore the differences in length were not as important as it would have been expected for a data-set relative to growing children.

Nevertheless, changes in femoral lengths are still likely to be the most statistically significant. Anthropometric studies consider the length of the femur to be approximately $L_{femur} = 0.245 \times H$, where H represents the height of the person considered (Ethier & Simmons, 2007). So, even in the adult population, differences in femoral length are still predominant relative to all other changes in shape.

Since dealing with variations in length is relatively easy, it was decided to exclude this kind of information from the model. The advantages are two-fold. Firstly, by disregarding length, finer shape changes would be more evident. Secondly, by neglecting size it was easier to employ the medial description for shape representation; the femur can be considered to have an approximated cylindrical shape and the medial representation of a cylinder consists simply of its axis (Fig. 3.2, and see Chapter 5).

The techniques available for the computation of the medial representation of 3D shapes do not result in a continuous surface, but rather in a set of points sampling the actual surface of the medial representation. The number of these points cannot be controlled directly, but it depends on the initial number of points sampling the boundary of the 3D shape considered.

Assuming a constant sampling density (i.e. number of points considered per unit of area), the medial representation of a longer cylinder would consist of a larger number of points. The matching of features between two sets of medial points with considerably different cardinalities is not straightforward. Furthermore, since the shape of the femur is more complicated than a simple cylinder, there is a risk that some important medial points for the description of the shape are neglected in the process.

For all these reasons, each shape was scaled to one shape considered to

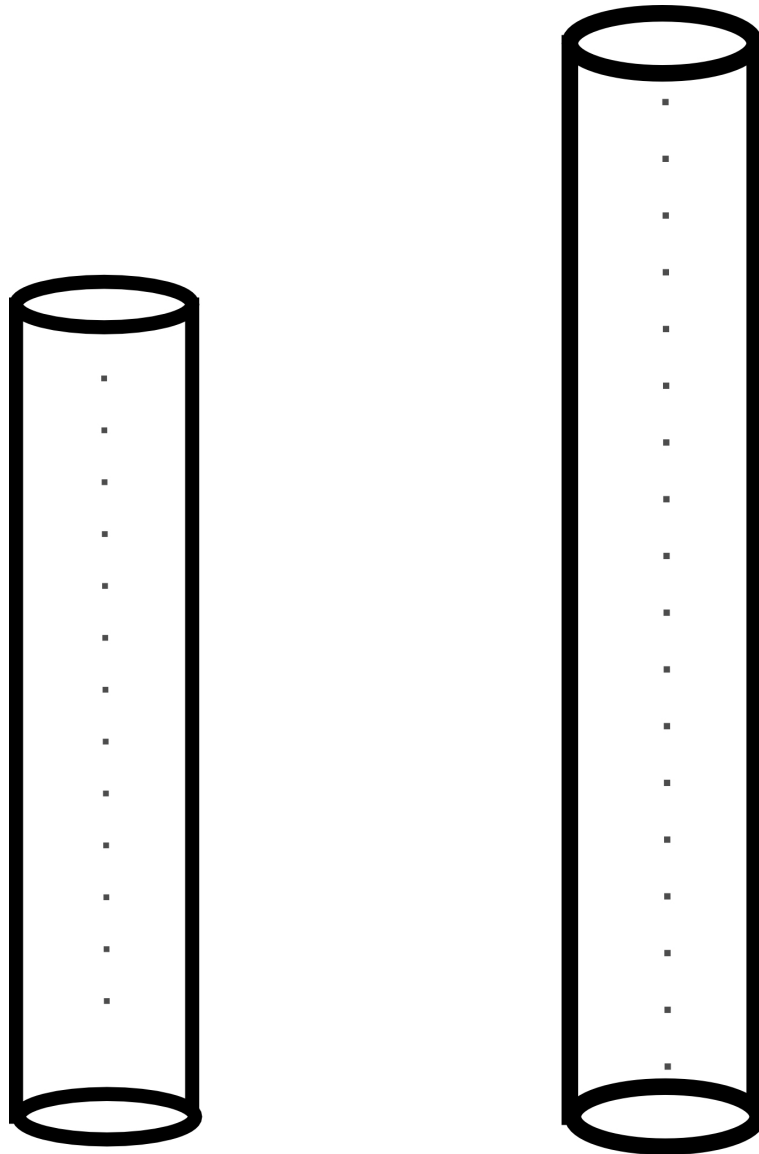


Figure 3.2: Medial representation of cylindrical shapes: the dots represent a sampling of the medial representation

be the reference shape, thus focusing on less evident aspects of shape and simplifying the matching process for the medial representation.

The scaling process relies on the definition of the bounding box of the shape. The bounding box of a set of points, $S \in \mathcal{R}^3$, is defined as the rectangular box that contains all the points in $p \in S$ and has minimum volume. From a computational point of view, this problem is usually solved in an approximated fashion, thus finding the rectangular box that contains all the point and has volume $V = (1 + \epsilon)V_{min}$, where V_{min} is the volume of the real bounding box and $0 < \epsilon < \epsilon_{max}$, with ϵ_{max} an upper limited deemed good enough for the specific application (Barequet & Har-Peled, 2001).

In order to compute the bounding box of each shape the vtk function *vtkBoundingBox* was employed (Schroeder, 2003). For the purpose of scaling, the properties of the bounding box of the reference shape and each other shape in the data-set were considered.

Let $\Delta x_1, \Delta x_2, \Delta x_3$ be the dimensions of the reference bounding box, so that the condition $\Delta x_1 < \Delta x_2 < \Delta x_3$ is verified. Now, by considering the general bounding box of a test shape t , its dimension being labelled $\Delta x_1^t, \Delta x_2^t, \Delta x_3^t$ (again under the assumption $\Delta x_1^t < \Delta x_2^t < \Delta x_3^t$), it is possible to perform the scaling by considering a scaling factor $\alpha = \Delta x_3 / \Delta x_3^t$ that is used for all the three dimensions. The scaled shapes can then be used for the remaining steps of the modelling process.

3.4 Feature Matching

The registered shapes need to be processed further in order to perform the modelling. After the roto-translation, all the shapes will be referred to the same coordinate system, but with no correspondence yet established between the points on each surface.

For two-dimensional shapes, one possible approach is to manually perform the matching by labelling characteristic points on the shape, such as points of

maximum curvatures or end points. However, for three-dimensional shapes such an operation could be both time-consuming and prone to error.

Furthermore, the different shapes in the data set are represented by sets of points with different cardinalities, thus making it difficult to decide which points should be retained and which should be discarded.

In order to perform the matching of features and solve the problem of the different cardinalities in the representations, one possible approach could be to deform a reference shape so that it approximates a given target shape; the information about shape should be preserved, provided the number of points in the reference shape is high enough, and the cardinalities of the different shapes would be homogeneous throughout the data set.

Lorenz & Krahnstöver (2000) proposed a semi-automatic approach in which some key-features are manually selected by the user on all the shapes. Subsequently, one shape acts as the template (source shape) which has to “coat” all the remaining shapes (target shapes). First the selected features are matched as closely as possible, then the mesh of the surface coating the target shape is relaxed, in order to make the point distribution resemble the source mesh as much as possible, still maintaining the matched features on the target shape.

Although this approach proved successful in modelling the vertebrae, femoral shape does not provide as many key feature points. Moreover, manual labelling is time consuming and dependent on the user so a completely automatic matching procedure would be desirable. A completely automatic procedure allows an easier extension of the model whenever a new instance of shape is available.

Automatic matching can be achieved through a grid-based deformation of a source shape onto different targets. Cootes *et al.* (2004) describe the computation of a statistical model based on diffeomorphic (i.e. non-folding) deformation fields defined on the nodes of a grid (two or three-dimensional).

These deformation fields can be represented by composing relatively sim-

ple basis functions. An obvious choice would be the B-splines used by Rueckert *et al.* (1999). Splines are a family of functions defined piece-wise by polynomials. Rueckert introduces the concept of Free-Form Deformations (FFD), based on B-splines, (Rueckert *et al.*, 1999) which have been successfully employed for feature matching and the computation of SSMs of biological structures (Yang *et al.*, 2006; Rueckert *et al.*, 1999, 2006).

The main idea consists of embedding the surface in a three-dimensional mesh and then defining a deformation field on this grid. This deformation field is modelled using B-splines that have the points of the mesh as their control points. The mesh is subsequently subdivided into finer resolution levels by increasing the number of control points, thus decreasing the mesh space.

Given a three-dimensional surface defined by points lying on it and connectivity between these points, the domain of the surface Ω would be formed by triplets of coordinates that fall within a rectangular parallelepiped, as shown in Eq.3.7

$$\Omega = \{(x, y, z) \mid X_{min} \leq x \leq X_{max}, Y_{min} \leq y \leq Y_{max}, Z_{min} \leq z \leq Z_{max}\} \quad (3.7)$$

where $X_{min}, Y_{min}, X_{min}$ ($X_{max}, Y_{max}, X_{max}$) are the lower (upper) limits of the bounding box containing the shape.

Given this domain, a first uniformly-spaced mesh Φ^1 of $n_x \times n_y \times n_z$ control points Φ_{ijk} can be defined and a deformation field can be created by using interpolating B-splines based on these points.

The deformation field $\mathbf{T}_{local}^1$, relative to the mesh Φ^1 , can be expressed in terms of a 3D tensor product of cubic B-spline functions (Eq. 3.8).

$$\mathbf{T}_{local}^1(x, y, z) = \sum_{l=0}^3 \sum_{m=0}^3 \sum_{n=0}^3 B_l(u)B_m(v)B_n(w)\Phi_{i+l,j+m,k+n} \quad (3.8)$$

where $i = \lfloor x/n_x \rfloor - 1$, $j = \lfloor y/n_y \rfloor - 1$, $k = \lfloor z/n_z \rfloor - 1$, $u = x/n_x - \lfloor x/n_x \rfloor$, $v = y/n_y - \lfloor y/n_y \rfloor$, $w = z/n_z - \lfloor z/n_z \rfloor$ and $B_l(u)$ represents the l^{th} basis function of the B-spline give in Eq. 3.9

$$\begin{aligned}
 B_0(u) &= (1 - u)^3/6 \\
 B_1(u) &= (3u^3 - 6u^2 + 4)/6 \\
 B_2(u) &= (-3u^3 + 3u^2 + 3u + 1)/6 \\
 B_3(u) &= u^3/6
 \end{aligned} \tag{3.9}$$

The control points of the mesh are the parameters of the B-spline and the degree of deformation depends on the resolution of Φ . Finer meshes allow for local non-rigid deformations at the expense of computational complexity. However, B-splines have the property of being locally controlled, so they are computational efficient even for a large number of control points.

The mesh of control points can be subsequently refined, thus leading to progressively finer meshes $\Phi^1, \Phi^2, \Phi^3, \dots, \Phi^L$, satisfying the condition that the spacing δ decreases from Φ^l to Φ^{l+1} . Different local transformations $\mathbf{T}^l$ can then be defined for a specific mesh Φ^l , and the global local transformation can be computed by adding the individual ones, as shown in Eq. 3.10.

$$\mathbf{T}_{local}(x, y, z) = \sum_{l=1}^L \mathbf{T}^l(x, y, z) \tag{3.10}$$

The values of the control points are adjusted so that the sum of the distances between each point in the reference shape and its closest point on the target shape is minimised with a procedure described by Lee *et al.* (1997).

By applying the local transformation to all the points in the target shape and preserving the connectivity information, a uniform representation within the data-set can be achieved, hence allowing for the computation of the SSM.

3.5 Modelling

Once a uniform and comparable representation across the data-set is achieved, it is possible to compute the proper SSM.

After the feature matching, each shape, $\mathbf{x}_i$, can be represented by a vector obtained by concatenating the coordinates of each point sampling its surface (Eq. 3.11); coordinates occupying the same position throughout the different vectors will be considered matched.

$$\mathbf{x}_i = \left(\overbrace{x_1^i \ y_1^i \ z_1^i}^{1^{st} \ point} \ \overbrace{x_2^i \ y_2^i \ z_2^i}^{2^{nd} \ point} \ \dots \ \overbrace{x_n^i \ y_n^i \ z_n^i}^{n^{th} \ point} \right) \quad (3.11)$$

The mean shape can be computed by averaging all the shape vectors in the data-set; assuming there are m shapes in the data-set, the expression for $\mathbf{x}_{\text{mean}}$ would be given by Eq. 3.12.

$$\mathbf{x}_{\text{mean}} = \frac{1}{m} \sum_{i=1}^m \mathbf{x}_i \quad (3.12)$$

Using the mean shape it is possible to translate the data-set in shape-space so that all the points will be centred on the origin, thus allowing for a simple description of variation; once the model is computed the variation in shape can be modelled by moving away from the origin along the so-called principal directions.

The data-set can be represented by a matrix $\mathbf{X}$ in which each column represents a shape after the mean has been subtracted.

Principal Component Analysis (PCA) aims to reduce the dimensionality of a data-set comprising many interrelated variables whilst retaining the majority of the variation observed. The original variables are transformed into a new set of variables, called Principal Components (PCs) that are a linear combination of the original ones. These new variables are uncorrelated, and

once the mean value has been subtracted, they satisfy the property shown in Eq.3.13 for all the pairs (X, Y) of PCs considered.

$$E(X, Y) = E(X)E(Y) \quad (3.13)$$

Where $E(X, Y)$ represents the expected value of the PCs considered together and $E(X)$ is the expected value of the PC considered.

The main focus of PCA is variance, thus the first step consists of finding a linear combination $\mathbf{PC}_1 = \alpha'_1 \mathbf{x}$ of the initial variables $\mathbf{x}$ with the maximum variance. Subsequently, among the linear combinations that are uncorrelated to $\mathbf{PC}_1$ the one that maximises the variance is computed ($\mathbf{PC}_2$). The procedure is iterated and each $\mathbf{PC}_k$ is computed so that its correlation with all $\mathbf{PC}_l$, with $l < k$ is zero and that the variance is maximised.

The computation of the PCs can be performed by solving the eigenproblem associated to the covariance matrix $\mathbf{C}$ of the dataset, which is defined in Eq.3.14.

$$\mathbf{C} = \frac{1}{m} \mathbf{X} \mathbf{X}^T = \mathbf{A} \mathbf{A}^T \quad (3.14)$$

The PCs describing the data-set correspond to the eigen-vectors of the matrix $\mathbf{A} \mathbf{A}^T$, i.e. those vectors $\mathbf{v}_i$ that satisfy Eq.3.15.

$$\mathbf{A} \mathbf{A}^T \mathbf{v}_i = \mu_i \mathbf{v}_i \quad (3.15)$$

When the cardinality m of the data-set exceeds the dimensionality n of the original space, it is possible to compute n PCs. In this case, PCA can be considered as a rotation of the original coordinate system that aligns the axes with the directions of maximum variation.

When $m < n$, although the computation of n PCs is still possible, only $m - 1$ eigenvectors are meaningful, as the covariance matrix is rank-deficient.

Turk & Pentland (1991) showed that the eigenvectors $\mathbf{u}_i$ of the $m \times m$ matrix $\mathbf{A}^T \mathbf{A}$ can be transformed into these meaningful eigenvectors of $\mathbf{C}$

(Eq. 3.16).

$$\mathbf{v}_i = A\mathbf{u}_i \quad (3.16)$$

This approach can save computational time and can be seen as a projection of the data-set onto the $(m-1)$ -dimensional hyper-plane defined by the data-points, and the subsequent definition of a coordinate system that maximises the variance observed alongside each of the axes.

The number of PCs that are computed can be as high as $m-1$, the cardinality of the data-set minus one, however it is common practise to retain just the first k PCs that account for most of the variation in the data-set.

In this scenario, each element of the data-set can be expressed in terms of the PCs. All the k PCs can be assembled into the matrix Φ , as shown in Eq. 3.17.

$$\Phi = [\mathbf{PC}_1 \quad \mathbf{PC}_2 \quad \dots \quad \mathbf{PC}_k] \quad (3.17)$$

Using this matrix it is possible to represent the data-set with a reduced set of variables, called b-values, which are the projections of each vector $\mathbf{x}$ along each of the PCs. Since not every PC is employed in this description, an additional term ϵ needs to be introduced to ensure a perfect representation. However, one can safely neglect this term assuming it is due to noise and errors in the process of shape representation, thus leading to Eq. 3.18.

$$\underbrace{\mathbf{x}}_{n \times 1} = \underbrace{\Phi}_{n \times k} \underbrace{\mathbf{b}}_{k \times 1} + \underbrace{\epsilon}_{n \times 1} \approx \Phi \mathbf{b} \quad (3.18)$$

In order to compute the PCs of a given data-set, the empirical covariance matrix of the examples is computed; each vector in the data-set, after the mean $\mathbf{x}_{mean}$ has been subtracted, will form one of the columns of the data matrix $\mathbf{D}$ (Eq. 3.19).

$$\mathbf{X} = [\mathbf{x}_1 - \mathbf{x}_{mean} \quad \mathbf{x}_2 - \mathbf{x}_{mean} \quad \dots \quad \mathbf{x}_m - \mathbf{x}_{mean}] \quad (3.19)$$

The empirical covariance matrix Σ can be expressed in terms of the centred data-matrix, as shown in Eq. 3.20.

$$\Sigma = \frac{1}{m} \sum \mathbf{X} \cdot \mathbf{X}' \quad (3.20)$$

The PCs can then be calculated by finding the matrix $\mathbf{V}$ that renders the covariance matrix diagonal (Eq 3.21).

$$\mathbf{V}'\Sigma\mathbf{V} = \mathbf{\Lambda} \quad (3.21)$$

The columns of $\mathbf{V}$ correspond to the eigenvectors of Σ and matrix $\mathbf{\Lambda}$ is a diagonal matrix with elements λ_{ii} each of which represents the eigenvalue associated to the i^{th} eigenvector.

In order to determine how many PCs should be retained, the cumulative variability is computed. Each PC is responsible for a percentage of variability expressed by Eq. 3.22, where the denominator represents the sum of all the eigenvalues.

$$V_i = \frac{\lambda_i}{\sum_{l=1}^L \lambda_l} \quad (3.22)$$

The cumulative variability can be defined as a non-decreasing function $C(i)$, that assumes values given by the expression in Eq.3.23.

$$C(i) = \sum_{l=1}^i V_l \quad \forall i = 1, \dots, L \quad (3.23)$$

The residual variance can be then computed by considering the following equation (Eq. 3.24).

$$R(i) = 1 - C(i) \quad (3.24)$$

The scree graph relative to the set of PCs can then be obtained by simply plotting the residual variance against the number of components. Gener-

ally, the first k components are retained so that a great proportion of the variability (for example 95%) is taken into account.

If the number of shapes in the data-set is limited, it is not possible to guarantee that the number m of examples would be higher than the dimension of the shape-space $d_s = 3 \times n$.

Whenever $m < d_s$, the points representing each shape in shape space will define a hyper-plane which is at most $(m - 1)$ -dimensional. Considering a fictional three-dimensional shape-space, if the available data-set comprises 2 shapes only, then the only principal direction that could be found would be that corresponding to a line passing through the points (Fig. 3.3).

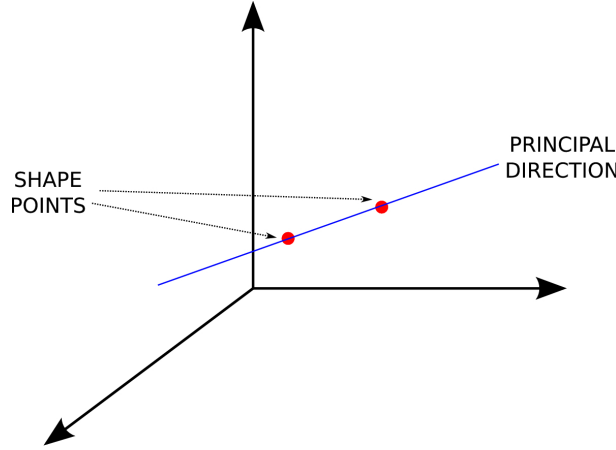


Figure 3.3: Principal Component Analysis performed on two three-dimensional points. Since the number of points available is less than the dimensionality of the space, the principal direction will define a line passing through the points

In extreme cases like the basic example just presented, PCA would not be able to produce a generic model but would simply give a compact representation that fits all the data perfectly, by means of a set of $(m - 1)$ orthogonal d_s -dimensional vectors that span the hyperplane containing the data points which are oriented along the direction of maximum variability.

In case the number of examples cannot exceed the dimensionality of the space introduced by the representation employed, the generality of the SSM must be questioned. As said earlier, only a subset of the PCs is usually retained, so that most of the variation is taken into account. However, if most of the computed PCs should be considered to describe this variability, then it is very likely that the model would have poor generality; the representation can still be used to describe the variability within the data-set, but not for other instances of shape.

However, if a smaller percentage of the components is needed to describe most of the variability, then the number of examples can be considered enough to capture the general aspects of shape.

No absolute criteria can be given in terms of an optimal data-set cardinality for the computation of a SSM, as this depends on the shape modelled and the representation used for the description.

A three-dimensional representation of the modelling process in shape-space is shown in Fig. 3.4. The first principal components, accounting for most of the variability (although not greater than 95%) are projected onto the unidimensional axis **I**, the remaining components that bring the overall variability beyond 95% are represented by axis **II** while the third axis represents all the remaining components.

Unless the set of examples used to compute the SSM is not (almost) perfectly aligned along the first axis, it would always be possible to detect the second direction, as the associated variability will be greater than the one observed along the third axis. From a practical point of view, if the available PCs allow variation to be detected that can be safely considered noise-related, then the model can be considered to be general enough to represent the population the data-set belongs to.

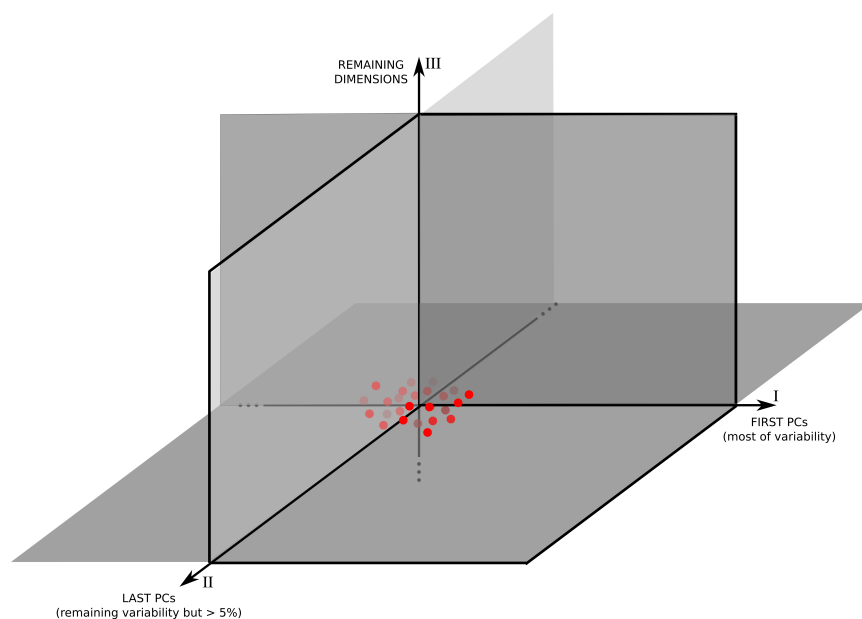


Figure 3.4: Principal Component Analysis performed in a space with higher dimensionality compared to the number of examples in the data-set

Chapter 4

Statistical Shape Model of Cloud of Points

In Chapter 2 the traditional approach to the description of femoral geometry was introduced. However, although this approach is motivated by what are the most evident changes in shape, it originates from the analysis of two-dimensional data that do not take into account the complexity of the three-dimensional space. The geometrical parameters considered in clinical practice are usually based on projections of the femur onto the anatomical planes; the reasons behind this approach are mainly due to the fact that planar images were available since the introduction of x-ray imaging. However, these parameters are not sufficient for the complete description of the three-dimensional anatomy and different definitions are needed according to the specific task to employ.

A statistical approach, performed not on the parameters but rather on the shape itself, would enable the description of the femur efficiently using a different set of parameters that has not been introduced by an expert, but is justified by a more rigorous statistical analysis of the femoral shape itself.

4.1 Introduction

Since the introduction of imaging methods in medicine, it has been important to describe quantitatively the shape of internal organs and structures, thus enabling the definition of what should be considered normal and what pathological. Many improvements had been introduced throughout the years in this field, 3D imaging being probably one of the greatest ones. However, most of the techniques employed to characterise biological shapes in clinical practice still rely on simple angle and distance measures taken on planar images, i.e. planar projections of three-dimensional structures.

Although these measures have been used for many years, they somehow lack the power to describe the overall complexity of a three-dimensional shape. On top of that, very often measures that are described with the same name are not directly comparable, as they are not referred to the same coordinate system and might not be independent from each other.

Statistical Shape Models (SSMs) have already been used to capture the complexity of three-dimensional shapes (Lamecker *et al.*, 2002; Shenton *et al.*, 2002; Davies *et al.*, 2003; Lamecker *et al.*, 2004; Yang *et al.*, 2006) with a reduced set of meaningful parameters, thus favouring compactness. The standard approach consists of representing the shapes as sets of matching features (i.e. points on the surface) and then mapping each instance of shape to a high-dimensional feature space. Provided that the modelled shapes are similar enough, some kind of dimensionality reduction technique, generally in the form of Principal Component Analysis (PCA), can be used to determine the principal modes of variation in feature space.

Despite being optimal in terms of compactness of representation, PCA alone might not be flexible enough to describe shapes that do not follow the patterns of the modelled population. Since the effects of pathology and trauma are mostly localised, a model that allows for a local description of shape changes would be preferable in a diagnostic context.

By rotating the Principal Components (PCs) using orthomax rotations,

it is possible to achieve the desired localised effects, retaining the overall compactness of the model brought about by PCA.

A standard SSM and its rotated version were created based on a data-set of 23 segmented three-dimensional left femurs. In order to evaluate the performances of these two models, a semi-automatic algorithm was implemented that extracts some geometrical parameters describing the geometry of the femur. Each component of the models was sampled at statistically significant intervals and the algorithm applied to each new instance of shape, allowing for a comparison between the two models

4.2 Statistical Shape Model

4.2.1 Shape Modelling

A set of 23 anonymised femurs was scanned using a CT scanner. The scanning parameters were the following: slice thickness 1mm; image matrix dimension 512×512 , with in-plane resolution $0.39mm \times 0.39mm$. All the bones were segmented using automatic thresholding in Mimics[®] (Materialise, 2009), and the reconstructed three-dimensional shapes were classified as normal by an orthopaedic surgeon.

These three-dimensional shapes formed the data-set on which the modelling methodology was applied. All the shapes were exported in *stl* format and then converted to *vtk* format (Schroeder, 2003).

In order to build a SSM it is necessary to have a comparable description of all the items comprising the data-set. It was decided to exclude the size component from the model, as size would be represented by the first principal component accounting for most of the variability. Including size is still possible and the results will be the same as the one presented here, the only difference being that the total variability within the training set would be considerably higher.

Once all the bounding boxes containing the object were scaled, one shape

amongst the data-set was chosen to be the reference shape. The effects of the choice of this reference on the resulting model has not yet been investigated, but it is unlikely they would be significant; this follows current practice in SSM (Yang *et al.*, 2006).

Each shape was then rigidly transformed in order to overlap it onto the reference shape, hence creating a common coordinate system. This task was accomplished via the VTK’s implementation of the iterative closest point (ICP) algorithm which effectively register 3D shapes, as shown by Besl & McKay (1992).

In order to perform PCA on the samples, each instance of shape should be represented by the same number of variables. This was accomplished using Rueckert’s Free Form Deformation algorithm introduced in Section 3.4; the reference shape is embedded in a three-dimensional mesh-grid. The grid, and the underlying space, can be deformed using b-splines having the points of the mesh as control points. The grid is progressively refined leading to an overall smooth deformation. The reference shape is then deformed so that its outer shape approximates each of the other shapes in the data-set (Rueckert *et al.*, 1999).

Since the chosen reference shape was represented by 4999 points, after the FFD is performed, each shape was described with the same number of points.

Starting from this representation, each shape can be now described by its associated shape vector, i.e. the vector built by concatenating the coordinates of each surface point.

$$\mathbf{X} = \begin{pmatrix} x_1^1 & y_1^1 & z_1^1 & x_2^1 & y_2^1 & z_2^1 & \cdots & x_n^1 & y_n^1 & z_n^1 \\ x_1^2 & y_1^2 & z_1^2 & x_2^2 & y_2^2 & z_2^2 & \cdots & x_n^2 & y_n^2 & z_n^2 \\ \vdots & & & & & & & & & \\ x_1^m & y_1^m & z_1^m & x_2^m & y_2^m & z_2^m & \cdots & x_n^m & y_n^m & z_n^m \end{pmatrix}^T \quad (4.1)$$

Where n is the number of points sampling each surface and m is the total number of shapes considered.

Given this representation, each instance of shape was represented as a point in a $3 \times n$ dimensional space. Since the number of points was 4999, then the shape space considered had 14,997 dimensions.

The main assumption in SSMs is that the points form a multi-variate Gaussian distribution in shape-space. Such an assumption cannot be verified until the actual model is built, since the dimensionality of the shape space is too high.

If the points indeed satisfy the Gaussian assumption, it is possible to capture the major modes of variation of the data set using Principal Component Analysis.

The mean shape was computed by averaging all the shape vectors, the resulting shape maintained the main characteristics of a femur (Fig. 4.1), thus partially justifying the Gaussian assumption. Since the properties of a multivariate Gaussian distribution are completely defined by the mean and the variances, having a mean that maintains the characteristics of the shape modelled gives some support to the assumption made.

Using the *princomp* function in MATLAB[®] (The MathWorks, 2009), principal component analysis (PCA) was performed on the data-set.

The principal components are ordered based on the variability explained; the first principal component accounting for most of the variability in the data-set. The overall variability of the model ($V = \sum_{i=1}^m \lambda_i$) is represented



Figure 4.1: Mean shape obtained from the point cloud representation. The main features of the femur are preserved and can be clearly identified.

by the sum of all the eigen-values of the components, thus the percentage of variability explained by a component j is expressed by Eq.4.2.

$$V_j = \frac{\lambda_j}{\sum_{i=1}^m \lambda_i} = \frac{\lambda_j}{V} \quad (4.2)$$

Usually not all the components extracted are retained (see Section 3.5), as the ones with small explained variance are considered to be influenced mainly by noise. The criterion followed for choosing how many PCs to considered was to retain a subset of the PCs that leaves a residual variance of less than 5%. By plotting the scree graph it was possible to determine that 9 components satisfied this criterion (Fig. 4.2).

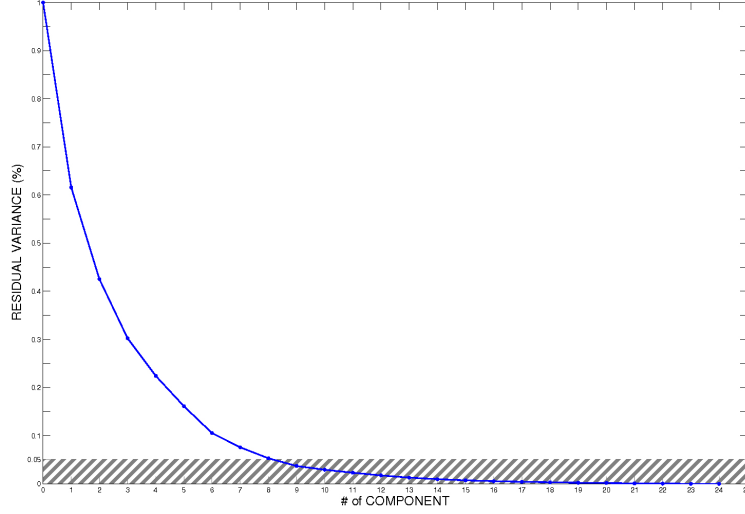


Figure 4.2: Scree Graph for the standard PC model

Once these components were selected they provided an effective and compact way to describe the shape of a healthy femur. Each shape could be represented by the following expression:

$$\mathbf{x} = \bar{\mathbf{x}} + \Phi \mathbf{b} + \epsilon \quad (4.3)$$

Where $\bar{x}$ represents the mean shape, b is the set of coordinates in the principal component space and Φ is a $n \times p$ matrix where each row corresponds to one PC. The term epsilon is considered to be some noise and it is neglected.

The coordinates of a new instance of shape in the PC shape can be calculated using the pseudoinverse Φ^{-1} of the principal component matrix.

$$\mathbf{b} \approx \Phi^{-1}(\mathbf{x} - \bar{\mathbf{x}}) \quad (4.4)$$

Using the values of $\mathbf{b}$ it is possible to estimate the probability density function of the shape vectors in PC space.

Since the number of examples in the data-set is limited, unidimensional histograms cannot be employed to verify the Gaussian assumption. Izenman (1991) obtained a robust expression to evaluate the optimal bin size W to use in histogram computation. This bin size depends on the inter-quartile range (see Section 6.2) and N , the cardinality of the data-set (Eq.4.5).

$$W = 2 \cdot IQR \cdot N^{-\frac{1}{3}} \quad (4.5)$$

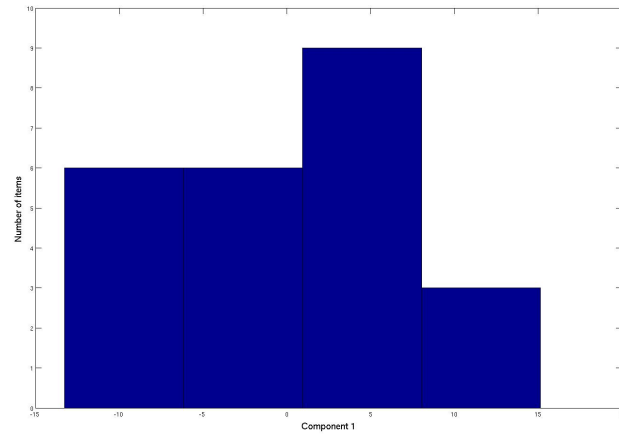
When considering the projections of the items in the available data-set on each of the PCs, the values of W lead to a number of bins ranging from 4 to 6, thus making the evaluation of the underlying probability density function inconclusive, as it can be seen in Fig.4.3.

Botev *et al.* (2010) developed a MATLAB script that computes an estimate of the probability density function of a dataset using a kernel method. The probability density function is approximated by the sum of kernel functions evaluated at each data point and a bandwidth parameter is optimised to obtain a smooth function.

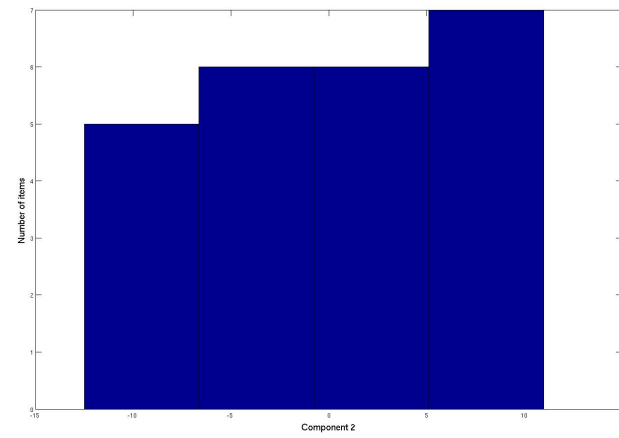
Plots of the density estimates for the projections on different PCs is given in Fig.4.4.

The different density functions confirm that the data-set can be considered approximately Gaussian. The peak values of the distribution is always close to the centre of the axes and the probability density function decreases monotonically while moving away from this point.

These qualitative properties justify the approach of describing the data-set using the mean shape and the variances associated with each PC that is used in Section 4.5.



(a)



(b)

Figure 4.3: Histograms of the projections of the data-set on the 1st (a) and 2nd (b) PCs. The optimal bin size does not allow to draw conclusions on the probability distribution.

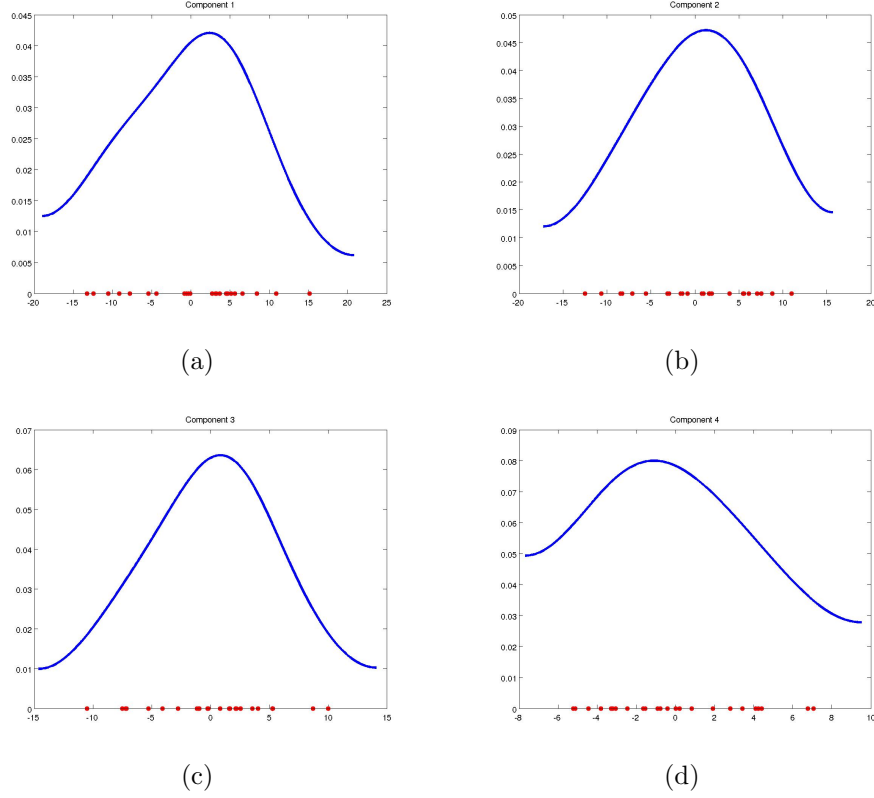


Figure 4.4: Estimate of the probability density functions in PC space. a) PC1 b) PC2 c) PC3 d) PC4

4.2.2 Rotation of the PCs

The resulting description of the shape is optimal in terms of compactness, however, if the aim is to have a model describing pathological changes in shape another approach might be more appropriate.

Orthomax rotations are used to produce a model with localised effects, allowing for a better detection and description of pathological abnormalities.

A rotation matrix $\mathbf{R}$ is applied to the original PC matrix, thus leading to a new vector basis that spans the reduced PC-space. The rotation is computed to maximise the criterion ψ in Eq.4.6. The matrix $\mathbf{\Phi}$ is orthonormal with

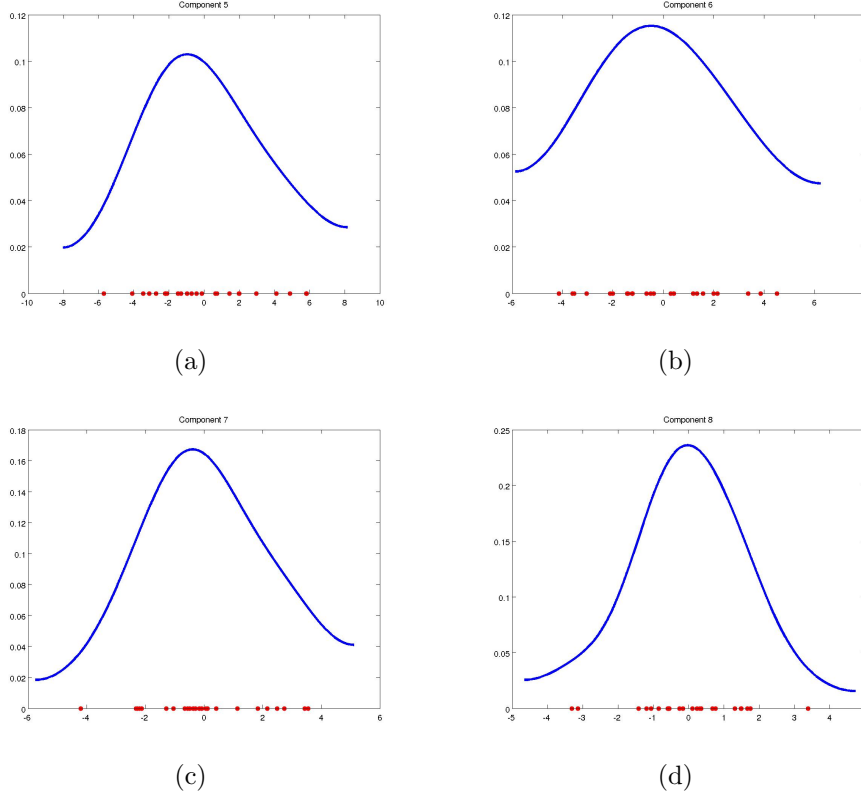


Figure 4.5: Estimate of the probability density functions in PC space. a) PC5 b) PC6 c) PC7 d) PC8

dimension $n \times k$ and $\mathbf{R}$ is an orthonormal rotation in $\mathcal{R}^k$, i.e. satisfying the property $\mathbf{R}\mathbf{R}' = \mathbf{I}^k$, where $\mathbf{I}^k$ is the $k \times k$ identity matrix (Stegmann *et al.*, 2006).

$$\psi = \frac{\sum_{j=1}^k \sum_{i=1}^n (\Phi \mathbf{R})_{ij}^4 - \frac{\lambda}{n} \sum_{j=1}^k (\sum_{i=1}^n (\Phi \mathbf{R})_{ij}^2)^2}{n} \quad (4.6)$$

The choice of the parameter λ determines the type of rotation, with values of $\lambda = 1$ and $\lambda = 0$ corresponding to *varimax* and *quartimax* rotations

respectively. As stated by Stegmann *et al.* (2006) the effect of a varimax rotation will result in an increase in the variance of the loadings within each component. Since $\mathbf{R}$ is an orthonormal rotation it cannot change the square sum of the new basis vectors. Therefore, the only way to increase the variance of the loadings in each column of Φ is to push some loadings close to zero whilst some others grow larger. Conversely, a quartimax rotation will tend to achieve the same results on the row of the matrix Φ thus leading to rotated components that affect different localised aspects of shape in an exclusive fashion.

For the purpose of this study, a varimax rotation seemed to be the best suited as it allows for a separate description of different shape changes that could involve the same aspect of the shape. For example, a change in shape affecting the head and neck of the femur could result in change of version and neck-shaft angle. A varimax rotation could describe these two effects with two different components, whereas a quartimax rotation might tend to group these two effects together, as it favours non-overlapping effects. The orthomax rotation was calculate using rotation optimisation routines developed by Bernaards & Jennrich (2005) in R (R Development Core Team, 2009).

In order to evaluate the statistical boundaries relative to the newly found component, the normal-distribution hypothesis was employed. Since the eigenvalues of the covariance matrix correspond to the variance of each component, the underlying distribution of the data can be modelled as a hyper-ellipsoid with the same dimensionality as the space. The axes of these ellipsoids will have a length equal to the square root of each eigenvalue (the region of space that was considered to be statistically significant actually lies within 3 standard deviations). Once the rotated components have been computed, the intersection of a line through the origin oriented like each rotated component was considered and its intersection with the hyper-ellipsoid was computed.

The equation of a hyper-ellipsoid is presented in Eq.4.7

$$\mathbf{x}^t \mathbf{A} \mathbf{x} = 1 \quad (4.7)$$

where $\mathbf{x}$ is a column vector and $\mathbf{A}$ is a diagonal matrix whose eigenvectors correspond to the principal axes of the ellipse.

If a coordinate system defined by the axes of the ellipsoid is considered, each rotated component will be represented by a new set of coordinates given by Eq.4.8

$$\begin{aligned} i &= 1, \dots, p \\ \widetilde{PC}_{rot}^i &= [PC_{rot}^i \cdot PC^1 \quad PC_{rot}^i \cdot PC^2 \quad \dots \quad PC_{rot}^i \cdot PC^m] \end{aligned} \quad (4.8)$$

where $\widetilde{PC}_{rot}^i$ represents the coordinates for the i^{th} rotated component in the coordinate frame aligned with the axes of the hyper ellipsoid, while the notation $PC_{rot}^i \cdot PC^j$ represents the dot product between the rotated components and the standard principal component, with index $j = 1, \dots, m$, defined in the original space. The effect of this change of coordinates can be visualised in 2D in Fig. 4.6.

The matrix $\mathbf{A}$ in this coordinate system will be diagonal and each element A_{ii} will correspond to the inverse of the eigenvalues of the covariance matrix.

The variability explained by each rotated component will be smaller compared to the original variables. In order to quantify this variability, a line passing through the origin and having the same direction of each rotated principal component is considered.

The parametric equation of a line passing through the origin is given by Eq.4.9:

$$\mathbf{x} = t \cdot \mathbf{d} \quad (4.9)$$

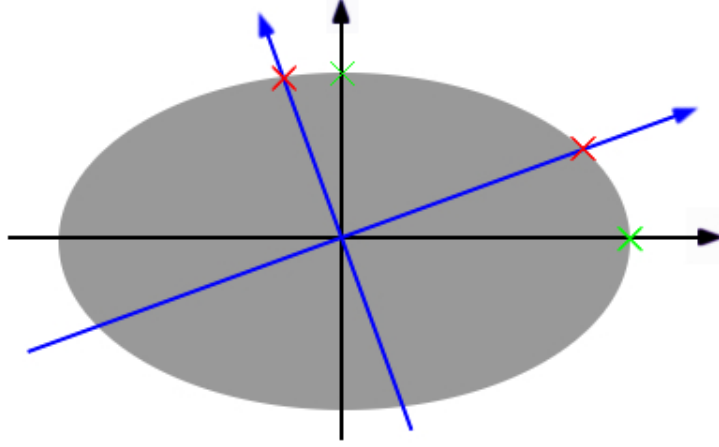


Figure 4.6: Two-dimensional example of how the rotation of the components changes the maximum variability explained along the components. Before the rotation the first components will have a standard deviation related to the length between the origin and the green crosses. After the rotation the standard deviation will be the length of the segments up to the red crosses.

By substituting Eq.4.9 in Eq.4.7 it is possible to find the value for t that corresponds to the intersection of the line with the surface of the ellipse. Provided that $\|d\| = 1$ the value of t corresponds to the variability explained along the rotated component considered (Eq.4.10).

Moving along each rotated component up to a value of $3t$ (both in the positive and negative direction) will give an insight into the effect of the component itself remaining within the region of legal variation consistent with a Gaussian model.

$$t^2 = \frac{1}{\mathbf{d}^T \mathbf{A} \mathbf{d}} \quad (4.10)$$

4.3 Measurement Method

In order to quantify variations in femoral shape, some geometrical parameters linked to the bone's functionality need to be evaluated.

Since the femur is composed of three major structures (the head/neck, the shaft and the condyles) the measures that were considered relate to the relative positions of these structures and some significant parameters related to their size.

Five different parameters were extracted from each shape. Despite there being more parameters that can be introduced to better describe femoral geometry, this subset was considered to be sufficient for the purpose of the study.

The five measurements taken can be divided in two sub-categories:

- Angular Measures
 - Neck Shaft Angle (NSA): This angle measures the inclination of the head and neck of the femur relative to the shaft axis. Some authors consider this angle as the angle formed by the projections of the neck and shaft axes onto the coronal plane. However, as the change of version angle leads to a change in neck shaft angle if this definition is used, the 3D angle between the two axes was considered instead, since it is not affected by this problem (Fig. 4.7).
 - Version Angle (VA): This measures the position of the neck axis relative to the medial lateral direction in the transverse plane (Fig. 4.8).
 - Bicondylar Angle (BA): This parameter measures the inclination of the shaft axis relative to a line perpendicular to the line passing through the most inferior points of the femoral condyles (Fig. 4.9).
- Size Measures

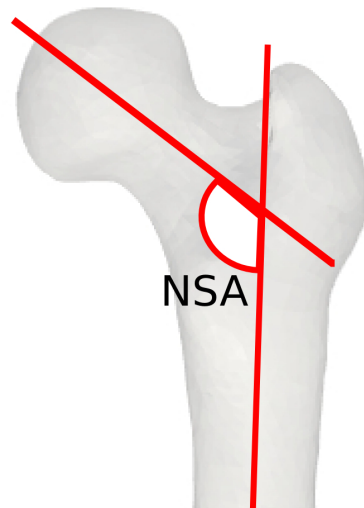


Figure 4.7: Definition of the Neck Shaft Angle

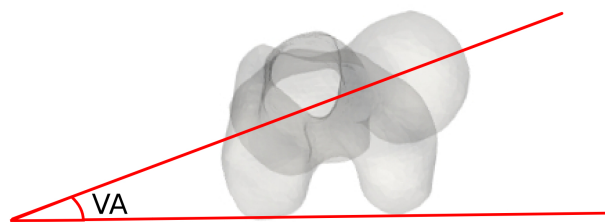


Figure 4.8: Definition of Version Angle



Figure 4.9: Definition of Bicondylar Angle

- Head Radius (HR): the radius of the sphere fitting a sphere on the surface of the head of the femur (Fig. 4.10).
- Intercondylar Distance (ID): the length of the segment connecting the two most inferior points on the condyles (Fig. 4.11).

An accurate angle definition depends on a consistent definition of a coordinate frame. Usually, while acquiring a medical image, the patient is put in an anatomical position. However, the accuracy of this manual positioning is far from perfect and can be influenced by the physical limitations of the patient; it is therefore advisable not to rely on the image's own coordinate system but to build a coordinate system based on the characteristics of the shape itself. The directions that need to be defined are the Lateral-Medial Direction (LMD) the Posterior-Anterior Direction (PAD) and the Inferior-Superior Direction (ISD).

There are different methods described in the literature to define these



Figure 4.10: Definition of Head Radius

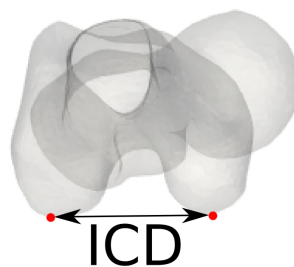


Figure 4.11: Definition of Intercondylar Distance

directions. It was decided to follow the classic table top method, as presented by Kingsley & Olmsted (1948), to determine the so-called posterior condylar axis, i.e. the axis passing through the most posterior points of the two condyles.

The aim is to find the contact points of the posterior aspects of each femur in the data-set with the surface of a table. Since only the three-dimensional reconstructions of the actual femurs are available, the procedure is not actually performed, but only simulated using custom-made Python (van Rossum, 1995) code and using the python vtk (Schroeder, 2003) wrappings for visualisation purposes.

In order to find the actual points of contact with the imaginary table top, an iterative algorithm was developed. First the femoral shape needs to be oriented according to its principal directions, computed using the *prcomp* function in R accessed through python's rpy module (Moriera & Warnes, 2004). The mean point of all the surface points was taken to be the centre of the new coordinate system.

The first principal axis always correspond approximately to the long axis of the femur. However it is not possible to assign the two remaining axes to the LMD or PAD just by considering their associated variances.

A simply heuristic method was introduced to distinguish between the two directions; if only the condyles are examined, by considering those points on their surface that lie on the 1-2 plane (i.e. the plane defined by the 1st and 2nd principal directions) and those that lie on the 1-3 plane, it is possible to identify the anterior/posterior direction by looking for the point corresponding to the intercondylar fossa, the concave region separating the two condyles.

Considering the principal coordinate system by looking at the absolute values of the coordinates of those condylar points that have been selected (those points lying on the 1-2 and 1-3 planes), the one with the lowest value will correspond to the intercondylar fossa (Fig. 4.12). Due to its shape, the

intercondylar fossa corresponds to a depression that can be easily identified by looking at the intersection of the shape with the aforementioned planes.

This assumption was satisfied in all but 3 of the 72 reconstructed shapes¹ that were tested and never failed in finding at least the orientation of the AP direction, since in those 3 cases the point identified lay on the trochlear surface.

Whenever the right direction is not found, it is possible to manually change the orientation, thus establishing the correct orientation of the axis; this was done in the three cases mentioned.

Once the estimate of the anatomical directions had been found, a transformation was applied to the points so that the x,y and z axes will correspond to the rough estimates of the LMD, PAD and ISD.

Starting from the transformed points, the aim was to find a small rotation of the shape that satisfied the condition that 3 points, lying on the greater trochanter and the medial and lateral posterior condyles, had the same coordinate along the axis labelled as PAD.

This problem was solved by employing a two-step iterative procedure. The first step consisted of finding a first approximation of the condylar axis. The femur was rotated about the 1st principal axis until there were two points on the condyles that had the same posterior-related coordinate (since in the coordinate system a point was defined as more posterior the more negative its y-coordinate was, points with the most negative second coordinates were considered).

A dichotomic style search was implemented to speed up the procedure and allowed for results deemed accurate enough for the purposes of this thesis. The procedure is first initialised by considering a rotation of $+10^\circ$ and -10° about the z-axis. The condylar points with the lowest y-coordinate (more negative y-coordinates correspond to more posterior points in the provisional

¹For each of the nine PCs of both the rotated and unrotated model, 4 shapes were extrapolated, thus $9 \times 4 \times 2 = 72$

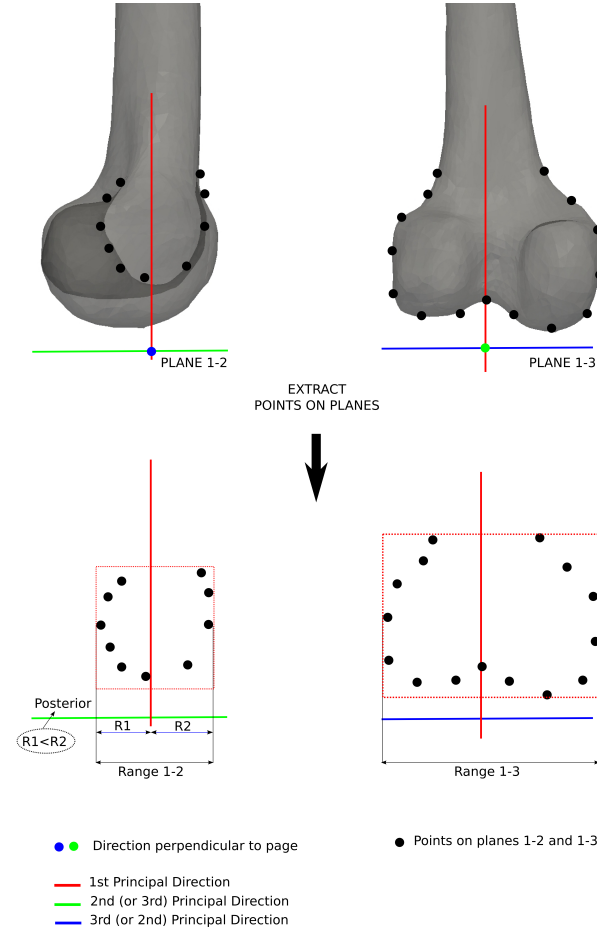


Figure 4.12: Heuristic method to determine anterior-posterior direction. The points on the surface that lie on the 1-2 and 1-3 planes are considered. The range of the coordinates of these points along the 2nd and 3rd principal direction are then evaluated, and the direction with the lowest range is taken as the AP direction. The point with the minimum absolute value of the coordinate is considered as a point on the intercondylar fossa and therefore determines the posterior direction. Whenever this assumption is not satisfied, it is possible to manually swap the Anterior and the Posterior labels. This manual correction was applied in 3 of the 72 cases tested.

coordinate frame) were considered to be the current contact point, i.e. the contact point with a table top if the shape was simply translated along the z-axis until it touched the surface of the table.

The positive rotation was labelled Lateral (L), as it tends to make the contact point lie on the lateral condyle, whereas the negative rotation was labelled Medial (M) for analogous reasons. If the actual contact point was not located accordingly with the label, the angles of rotation were increased by a 1° step, until the labels were consistent.

The unrotated shape was labelled O and corresponds to a mid-rotation between the rotations of the M-shape and the L-shape.

If the O-shape satisfied the condition that the two contact points on the condyles have the same valued y-coordinate, the procedure stopped and it was possible to proceed to the next step.

If there was only one point of contact in the condylar region, then either the M-shape or the L-shape was dropped, according to the position of the contact point on the O-shape; if it was medial then the M-shape was dropped and replaced with the O-shape, otherwise it is the L-shape that would have been replaced. At this point a new O-shape was computed by considering a new angle of rotation by averaging the angles of the new M and L shapes.

This procedure was iterated until the difference between the 2nd coordinate of the contacts points was below a threshold $T = 0.1mm$ (Fig. 4.13).

Once convergence was obtained, it was possible to move on to the following step. The final shape was rotated about a new axis passing through the two condylar contact points, using the rotation matrix provided in Eq.4.11

$$R = \begin{bmatrix} X^2 + (1 - X^2)C_\Theta & XY(1 - C_\Theta) - ZS_\Theta & XZ(1 - C_\Theta) + XS_\Theta \\ XY(1 - C_\Theta) + ZS_\Theta & Y^2 + (1 - Y^2)C_\Theta & YZ(1 - C_\Theta) - XS_\Theta \\ XZ(1 - C_\Theta) - XS_\Theta & YZ(1 - C_\Theta) + XS_\Theta & Z^2(1 - Z^2)C_\Theta \end{bmatrix} \quad (4.11)$$

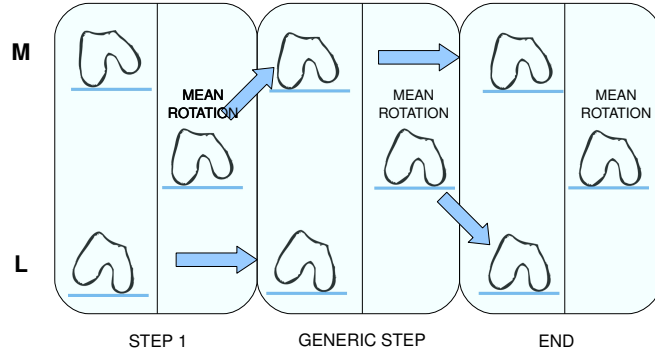


Figure 4.13: Schematic of the iterative procedure employed to determine the most posterior condylar points

where X , Y and Z correspond to the components of the unit vector defining the axis of rotation and C_Θ and S_Θ correspond to the *cos* and *sin* of the angle of rotation Θ . The same procedure described for the rotation about the z -axis was employed, the only difference being that the contact points were labelled Inferior (I) and Superior (S) (the initial inferior contact point will correspond to either one of the contact points found in the 1st step).

When this second step converged, the algorithm was started again from the first step until there was no significant difference in the y -coordinates of all 3 contact points.

The line passing through the final contact points on the posterior aspects of the condyles represents the condylar axis. In order to find the LMD, the long axis of the femur was computed by considering planes perpendicular to the first principal axis of the femur taken from 20 to 80 % of the overall length at 10% intervals.

Ellipses taken in correspondence to each of these planes were fitted using a least square technique (Fitzgibbon *et al.*, 1999) and then a regression line fitting the centres of these ellipses was computed. New cutting planes perpendicular to this newly found direction were considered and new ellipses were fitted at the same levels. The procedure was then iterated until the change in direction was smaller than an appropriately chosen threshold ($T = 0.5^\circ$).

Once the long axis was computed, the reference PAD was computed by considering the cross product between the ISD, given by the long axis, and the condylar axis. The cross product of the PAD with the ISD finally gave the LMD. These 3 directions form a right-handed coordinate system in which LMD, PAD and ISD correspond respectively to the x, y and z axes.

An example of the assigned coordinate system, alongside some of the axes that will be introduced later, is given in Fig. 4.14.

4.3.1 Neck Axis

The parameters that were computed depend on two other axes: the “capito collar” axis and the inferior condylar axis.

Kingsley & Olmsted (1948) suggested that the femoral head should not be included during the calculation of the version angle, however relying only on the neck can lead to difficulties, as this structure might not be developed enough to allow for the determination of its axis. The “capito-collar” axis was used instead, as presented by Pearson & Bell (1919).

Using the Visualization ToolKit (Schroeder, 2003) through its python wrappings, a simple GUI was implemented that allows the selection of a region containing the base of the femoral neck. A rectangle can be manually selected by the user in a rendering window; firstly the femur is shown in a frontal plane and two points can be selected on each end of the base of the femoral neck. The view is then automatically shifted to a plane perpendicular to the line passing through these two points and a third point can be selected in a transverse plane (Figs.4.15,4.16).

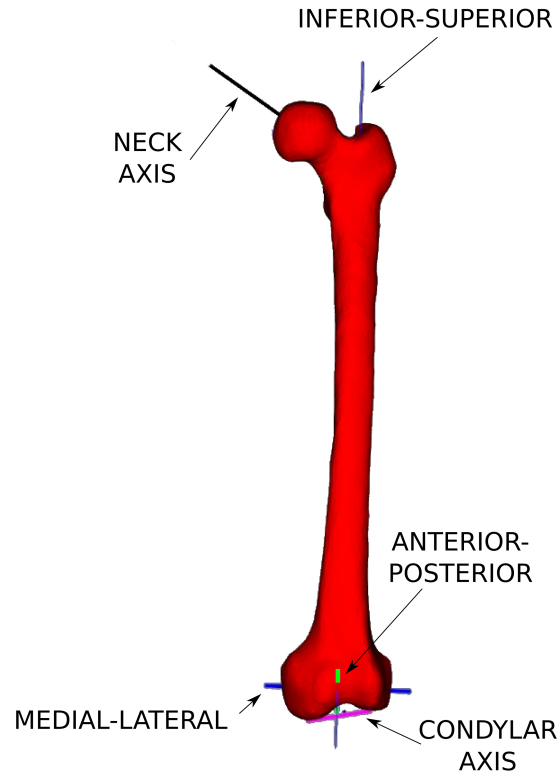


Figure 4.14: Assignment of a coordinate system and computation of main axes relative to femoral shape

Thus a selection rectangle is defined and the perpendicular to this rectangle pointing medially is used as the first approximation of the capito-collar axis. Ellipses are fitted in the region between 20 and 80% of the overall length of the femoral neck and head, a line is then fitted to the centres of these ellipses and new cutting planes perpendicular to this line are considered. The procedure is then iterated until there is no significant change in the direction (less than 0.5°).

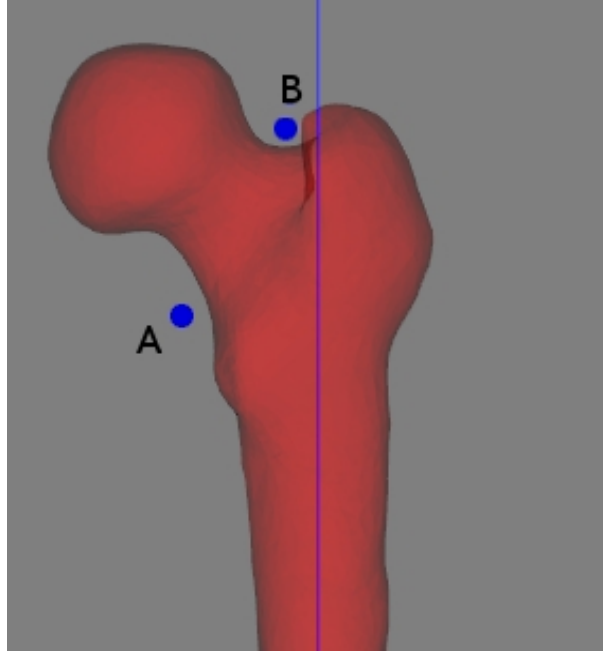


Figure 4.15: First step of the selection of the neck area. Point A is close to the base of the neck in the medial/inferior region whereas point B corresponds to the base of the neck in the lateral/superior position

4.3.2 Bicondylar angle

In order to measure the bicondylar angle a line passing through the two most inferior points was considered; since an appropriate coordinate system had already been defined, the two points with the most negative z-coordinates in the positive and negative x-semispaces were selected. The bicondylar angle was considered to be the angle between this line and the x-axis in the coronal plane (Fig. 4.17).

4.3.3 Head Radius

The computation of the HR is based on the manual selection of points on the articular surface of the femoral head; once these points are selected the

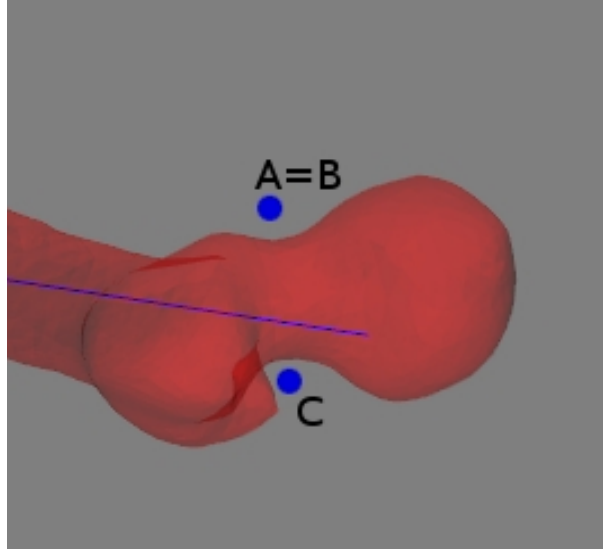


Figure 4.16: Second step of the selection of the neck area. Point A and B will define a line that is now parallel to the view direction. Point C must be selected close to the base of the neck in the posterior direction

best fitting sphere is computed, and its radius considered to be the radius of the femoral head.

4.4 Comparison with previous studies

In order to evaluate the performances of this software, the measure obtained from the data-set were compared to previous studies (Table 4.1); the standard deviations and/or ranges observed are presented when available. The results are in line with those available in the literature, thus proving the reliability of the computed measures. It should be noted that the set of femurs considered by Parsons (1914) also included bones showing a negative VA, thus explaining the lower value of the mean VA observed in this study.

Sugano *et al.* (1998b) used a method that compares directly to the one introduced here. It is worth noticing that the geometrical parameters they

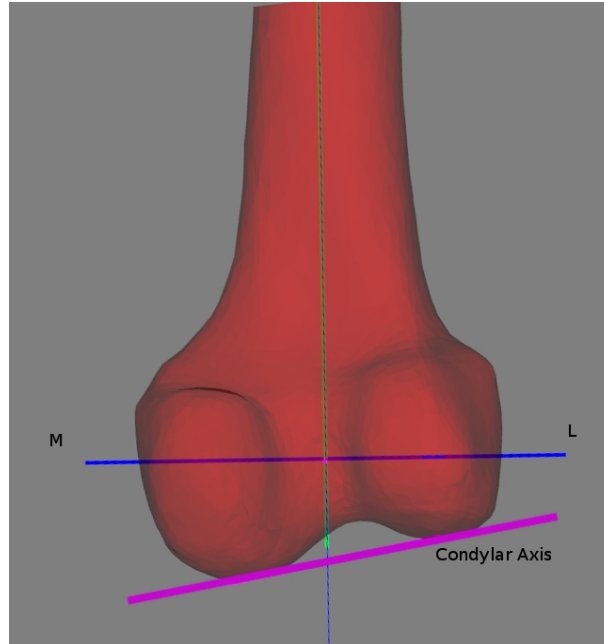


Figure 4.17: The two most inferior points are selected and the condylar axis is defined by the line joining them, the angle between this line and the medial lateral direction forms the bicondylar angle.

found are in line with what has been found by applying the measurement software to the 23 femurs available.

	NSA [$^\circ$]	VA [$^\circ$]	BA [$^\circ$]	HR [cm]
3D SHAPES	$126.94^\circ \pm 8.4$ (114.45, 141.42)	26.42 ± 5.71 (16.8192, 37.60)	12.03 ± 1.7 (7.91, 15.90)	2.51 ± 0.17 (2.12, 2.8)
Parsons ^a	$126 \pm ?$ (112, 138)	$13 \pm ?$ (17, -40)	$9 \pm ?$ (14, 15)	$2.45 \pm ?$ (2.05, 2.75)
Sugano ^b	125.8 ± 6.3	22.6 ± 10.6	...	2.05 ± 0.1
Sugano ^c	...	19.8 ± 9.3	...	...
	...	13.4 ± 10.4	...	...
	...	26.0 ± 9.1	...	...
Kersnič ^d	...	...	...	$2.68 \pm ?$ (2.36, 2.89)

^aParsons (1914)

^bSugano *et al.* (1998b)

^cSugano *et al.* (1998a), 3 methods are compared

^dKersnič *et al.* (1996)

Table 4.1: Comparison of geometrical parameters gathered during different studies. Sugano *et al.* (1998a) considered 3 alternative methods from different sources. Numbers in brackets represent the range of the values observed.

4.5 Results

The behaviours of the two models were compared first visually and then in a quantitative way using the set of measures previously introduced.

Since about 99.7% of the values drawn from a normal distribution lie within 3 standard deviations, new instances of shape were recreated by sampling the model at $\pm 1.5\sigma$ and $\pm 3\sigma$, where σ corresponds to the standard deviation of each component. The results of the traditional PCA model are shown in Fig. 4.29.

All the shapes created by the model preserve the structure of a femur and they actually describe changes in shape that represent the main known differences in femoral morphology.

The effect of the first component for instance can be related to a change in neck-shaft angle, however there is also a change in the thickness of the bone that can be noticed in the head radius.

To quantify the effect of each component some geometrical parameters were measured. The changes of these parameters along the first 8 components are presented in Figs. 4.20-4.23.

From Figs. 4.20 and 4.29 it can be seen that the 1st PC accounts for a change in shape affecting different aspects of the shape at the same time; moving along this component results principally in an increase in NSA and measures related to the size of the femur (both ID and HR). However, the changes in the other parameters, albeit being smaller, are not negligible.

By considering the ratio between the range of each measure for all the components and the maximum range in the overall model it is possible to evaluate the relative importance of each parameter compared to the others (Fig. 4.24).

By considering the 1st component it can be seen that there are 3 parameters that are equally important within this component (NSA, HR and ID). This can give further insight into the correlation between these morphological parameters in the normal population.

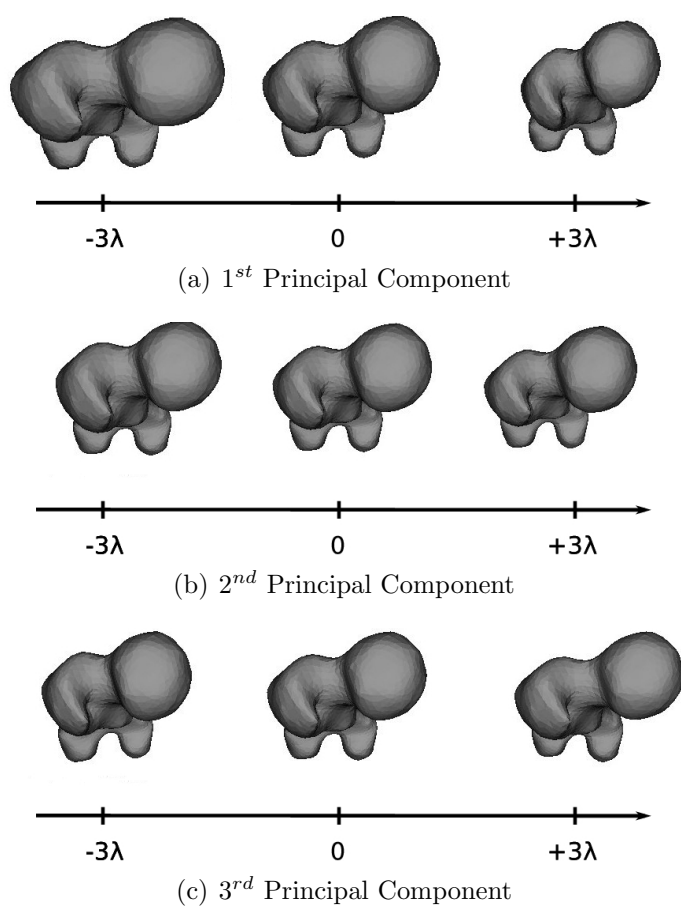
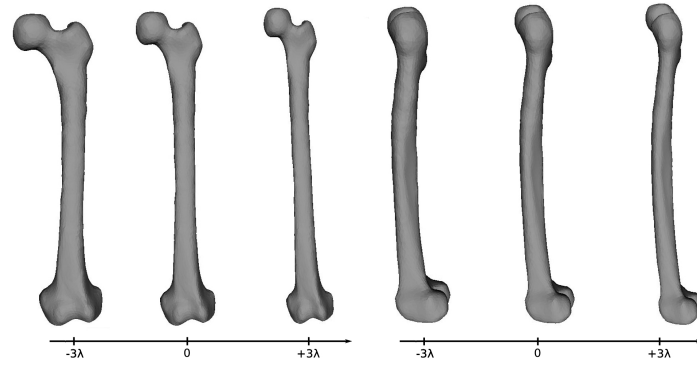
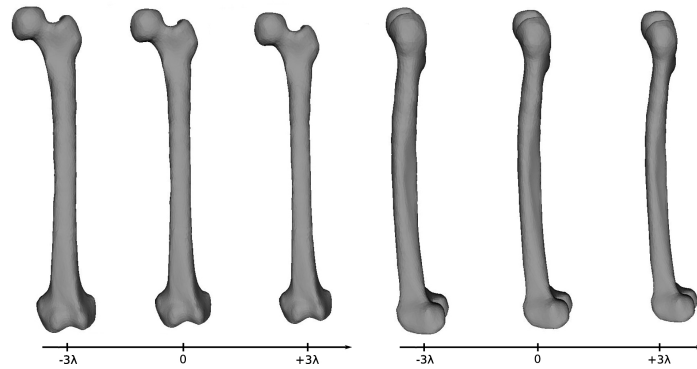


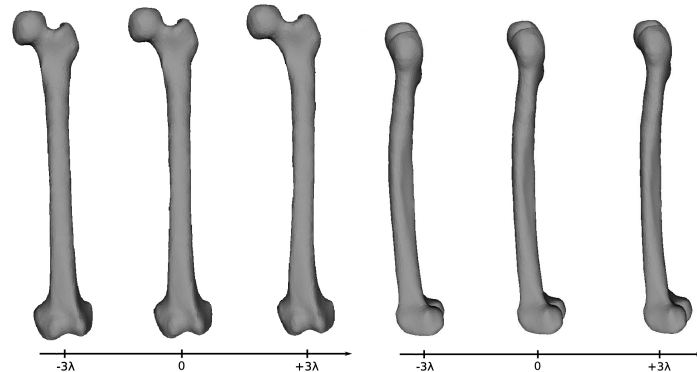
Figure 4.18: Transverse view of the effects of the first 3 principal components on the shape.



(a) 1st Principal Component

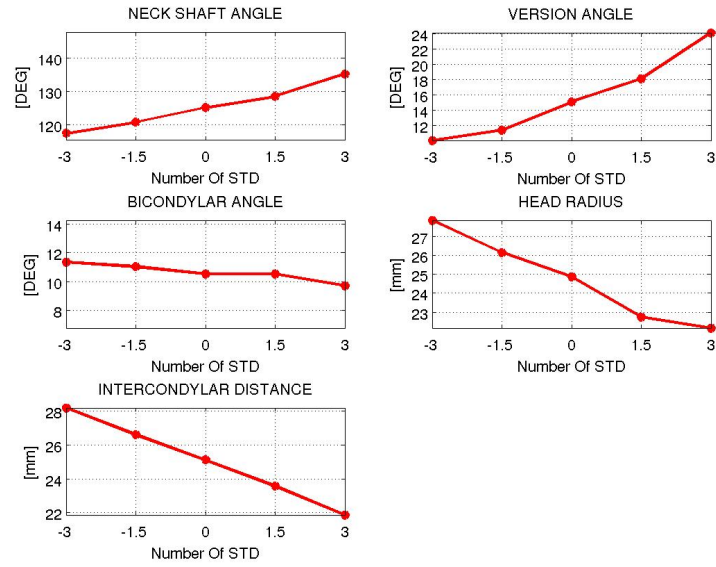


(b) 2nd Principal Component

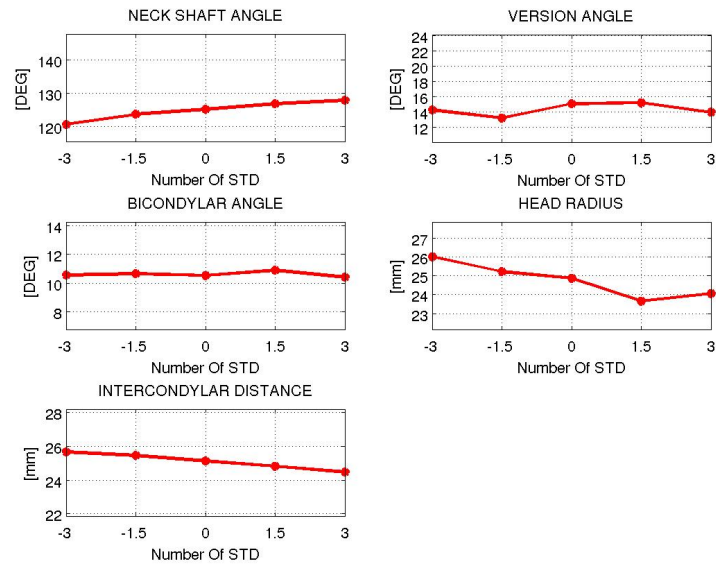


(c) 3rd Principal Component

Figure 4.19: Coronal and sagittal views of the effects of the first 3 principal components on the shape.

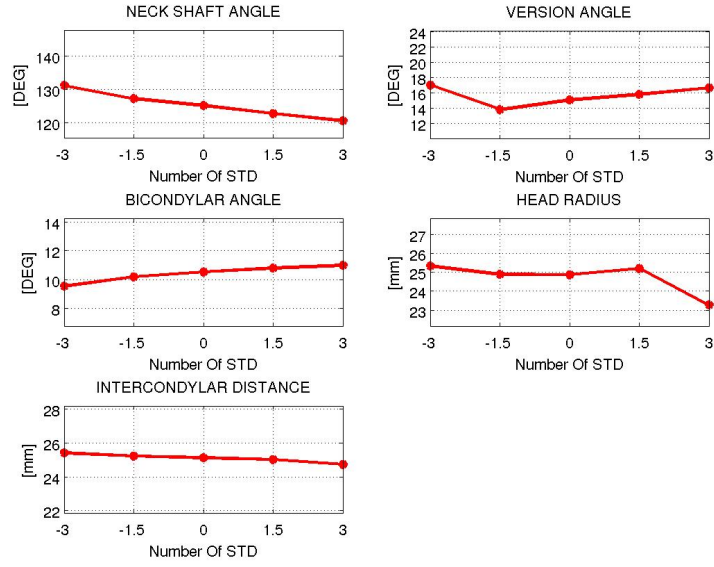


(a) 1st Component

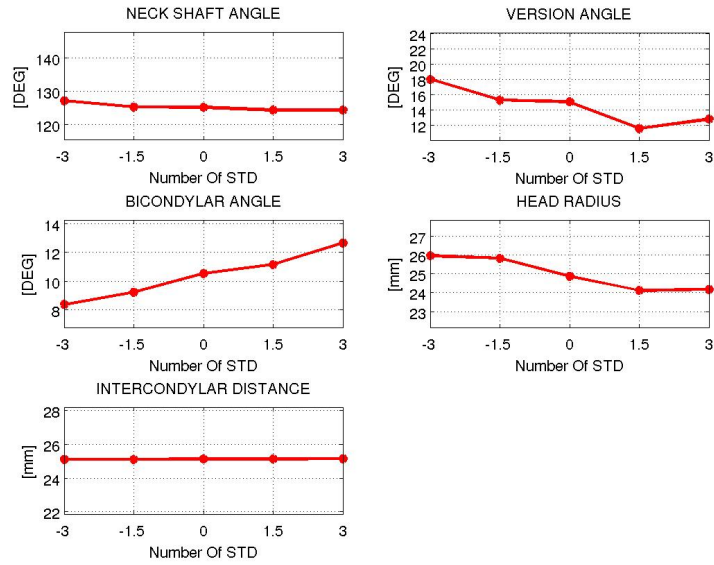


(b) 2nd Component

Figure 4.20: Effects of PCs 1-2 on the morphological measures

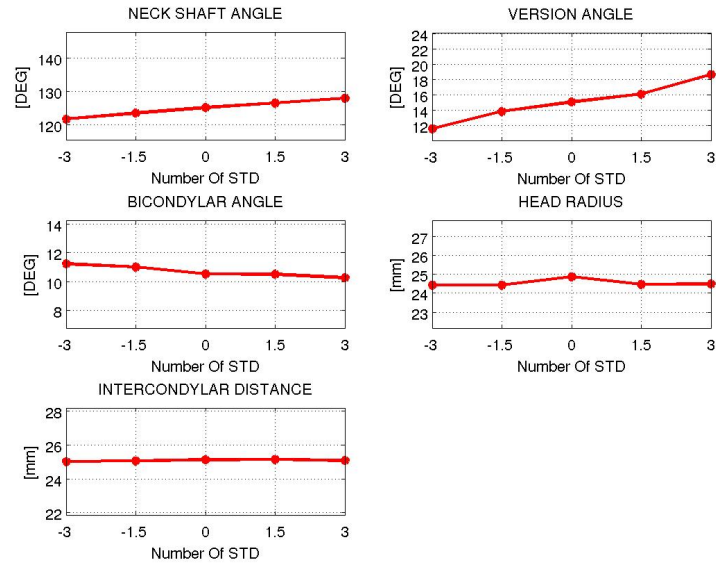


(a) 3rd Component

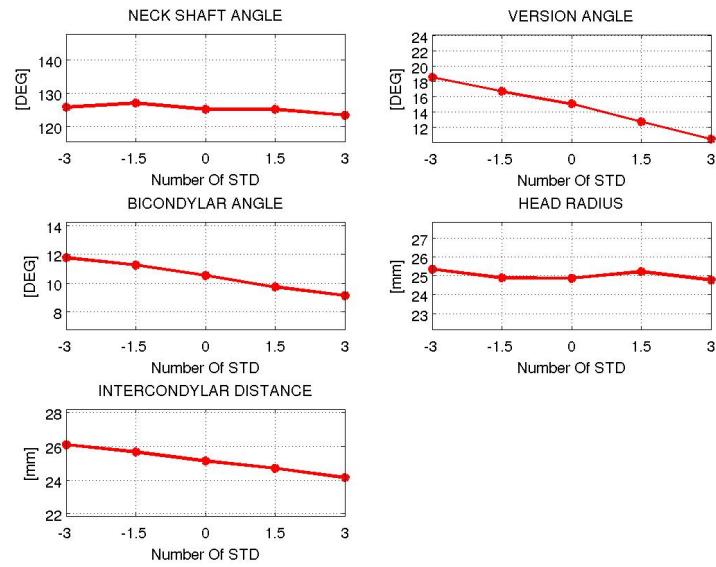


(b) 4th Component

Figure 4.21: Effects of the PCs 3-4 on the morphological measures

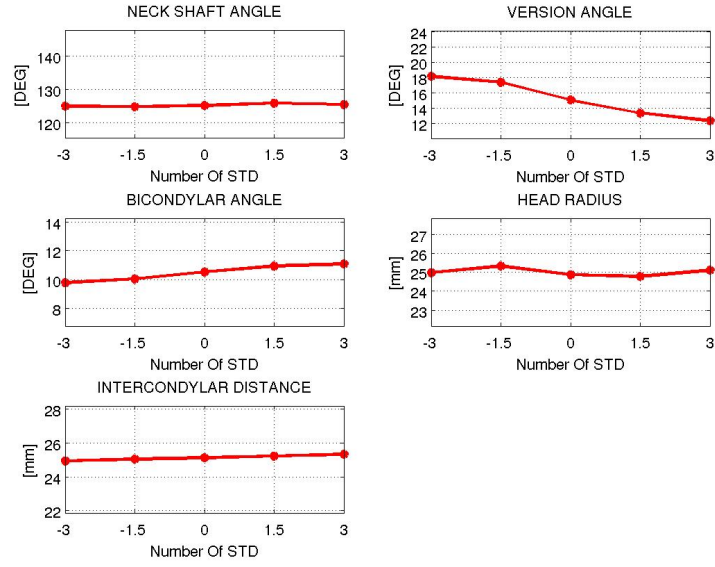


(a) 5th Component

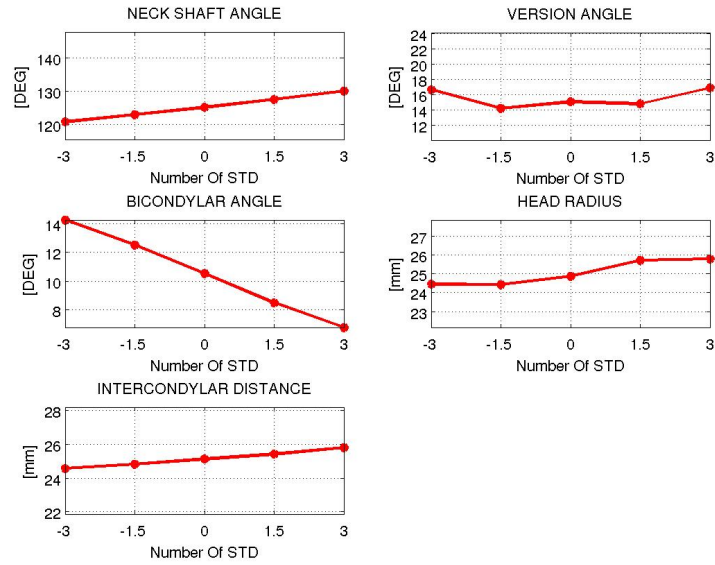


(b) 6th Component

Figure 4.22: Effects of the PCs 5-6 on the morphological measures



(a) 7th Component



(b) 8th Component

Figure 4.23: Effects of the PCs 7-8 on the morphological measures

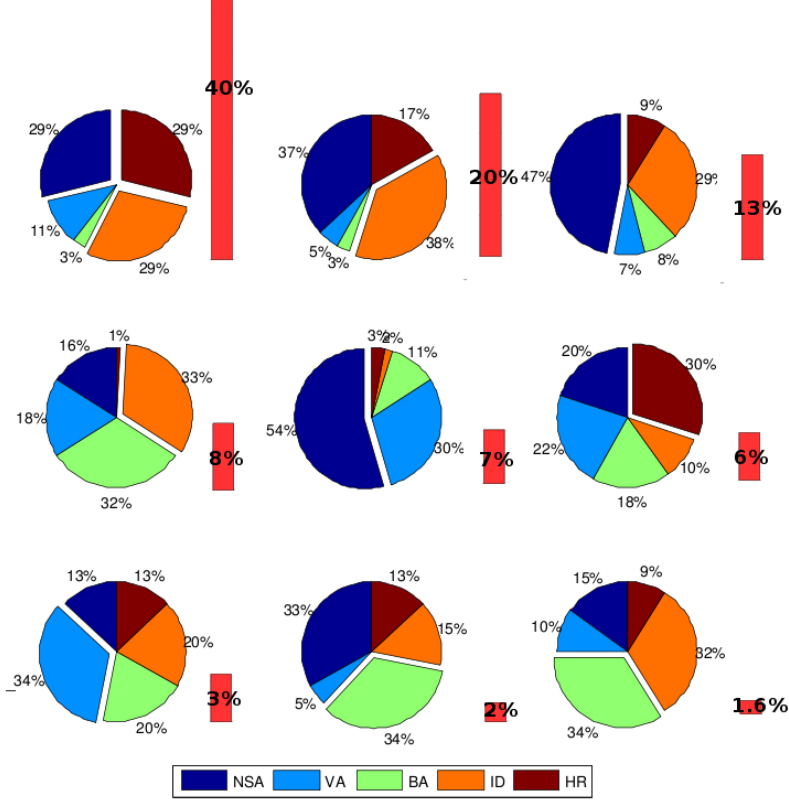


Figure 4.24: Pie charts describing the relative importance of each measure through the 9 components the model comprises

However, when considering pathological deformities, this correlation may no longer be present. From the standard SSM developed, a negative correlation was found to exist between the NSA and the HR. Sugano *et al.* (1998b) studied the morphology of the femur in developmental dysplasia of the hip (DDH), a disorder that represents abnormal development or dislocation of the hip (Miller, 2005, page 182).

Four different classes of DDH are here considered, according to the degree of sub-luxation (from class I to class IV, spanning a range of sub-luxation

from less than 50% to more than 100%). Classes II and III are grouped together thus effectively reducing the classes to three. Each of these classes was considered to be represented by its mean values for NSA and HR.

By considering the differences in NSA and HR between each DDH class and the normal population, the behaviour of the NSA as HR increases can be analysed (Fig. 4.25).

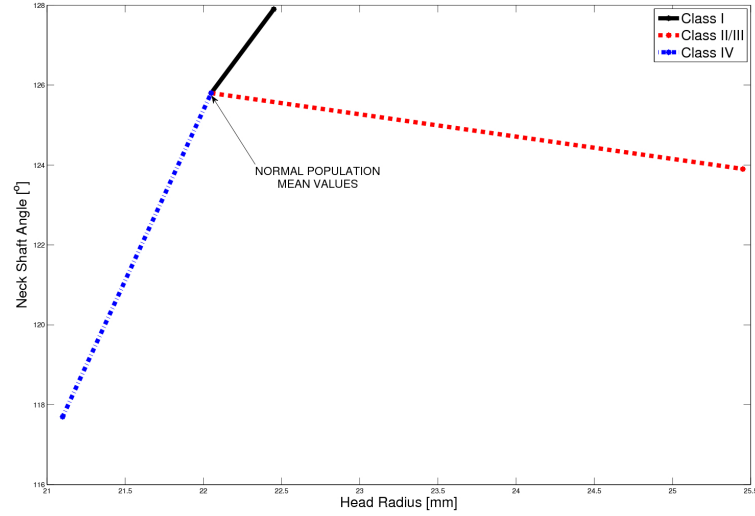


Figure 4.25: HR and NSA trends in DDH-affected subjects (data from Sugano *et al.* (1998b))

The standard SSM showed that, in the normal population, an increase in HR would correspond to a decrease in NSA. However, in the case of DDH subjects such a negative trend can be observed for class II/III.

In the case of DDH the correlation between NSA and HR explained by the SSM is disrupted, and similar trends for other sets of measures can also be lost due to pathology.

Moreover, since pathological deformities are usually characterised by a higher variability than that observed in the normal population, clear trends

cannot always be found.

Although this model captures those changes that tend to be correlated to each other providing a good way of understanding the relationships between the different structures within the shape; it has the limitation of being representative of the normal population only (i.e. the population that is modelled). Thus, as it is this model cannot be used for diagnostic purposes.

In the case of DDH, the results presented by Sugano *et al.* suggest that the correlation between NSA and HR found in healthy subjects is disrupted.

A *varimax* rotation of the PC (Section 4.2.2) led to more localised effects that enabled for more flexibility of the model.

A rotated component can be interpreted as a vector field defined on the points sampling the surface. Both the principal components and their rotated counterparts correspond to a vector field with a magnitude significantly different from zero only in a certain subset of all the points. Moving along each component modulates the effect of this vector field.

By comparing these vector fields before and after the rotation, it is noticeable that the vector field becomes more localised, as can be seen in Fig. 4.26.

Before the rotation the 1st component was mainly affecting the femoral head and the medial condyle, resulting in a change in NSA, VA as well as HR and ID.

After the rotation, the affected region is limited to the head region, therefore the component was expected to influence only the NSA, VA and HR. However, by looking just at the head region (Fig. 4.27) it can be seen that the vectors are almost parallel and not converging to (or diverging from) a common point, like it happens before the rotation.

In the light of this, it can be said that before the rotation, the component shows a change in head volume (change in HR) that does not appear once the rotation is applied. As will be shown later, the 1st rotated component does not involve any significant change in HR.

The effect of the orthomax rotation was to increase the variance of the

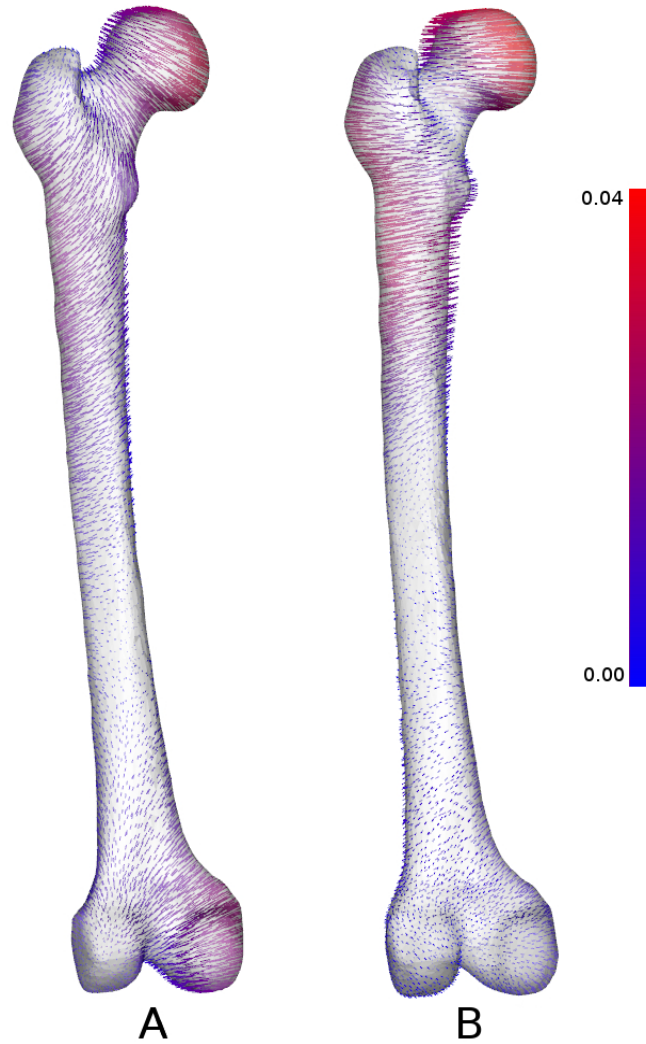


Figure 4.26: Effect of rotation on the vector field associated to the 1st component. Vectors are coloured according to their magnitude.
(A) Effect of the 1st component prior to rotation: the main area influenced are the head and the medial condyle. (B) Effect of the same component after rotation: the effect on the femoral head is stronger whilst there is no effect on the medial condyle

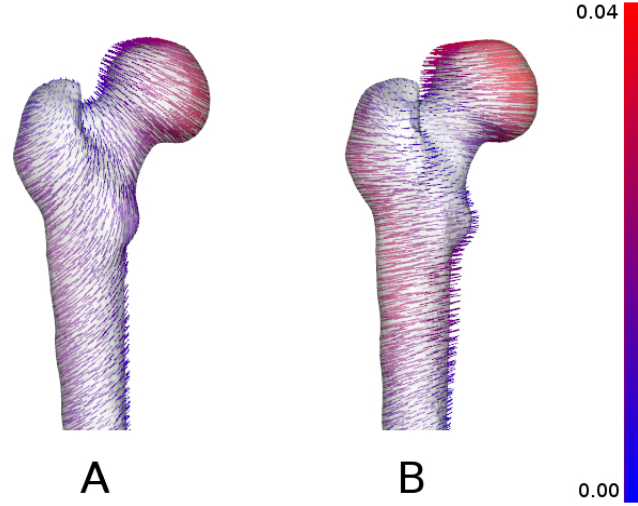


Figure 4.27: Effect of rotation on the vector field associated to the 1st component on the head region only. (A) Unrotated Component (B) Rotated Component

statistical distribution of the loadings within each component; a few loadings became relatively higher, whereas most of the others were practically rendered negligible (Fig. 4.28).

The bright bands in the heatmaps of Fig. 4.28 were more evident after the rotation, and there was also less overlapping between different components.

Analogous to what was done with the principal components, each rotated component was sample at $\pm 1.5\sigma$ and $\pm 3\sigma$, and the geometric parameters were measured. The result of this procedure is presented in Fig. 4.30.

The effect of the first 8 rotated components on the geometrical parameters can be evaluated in Fig. 4.31 4.34.

The relative importance of each measure within each component appears different from what it was observed for the standard PCA model (Fig. 4.35)

In all the 9 components there was only one parameter that clearly experienced the greatest variations compared to the others. Most of the components

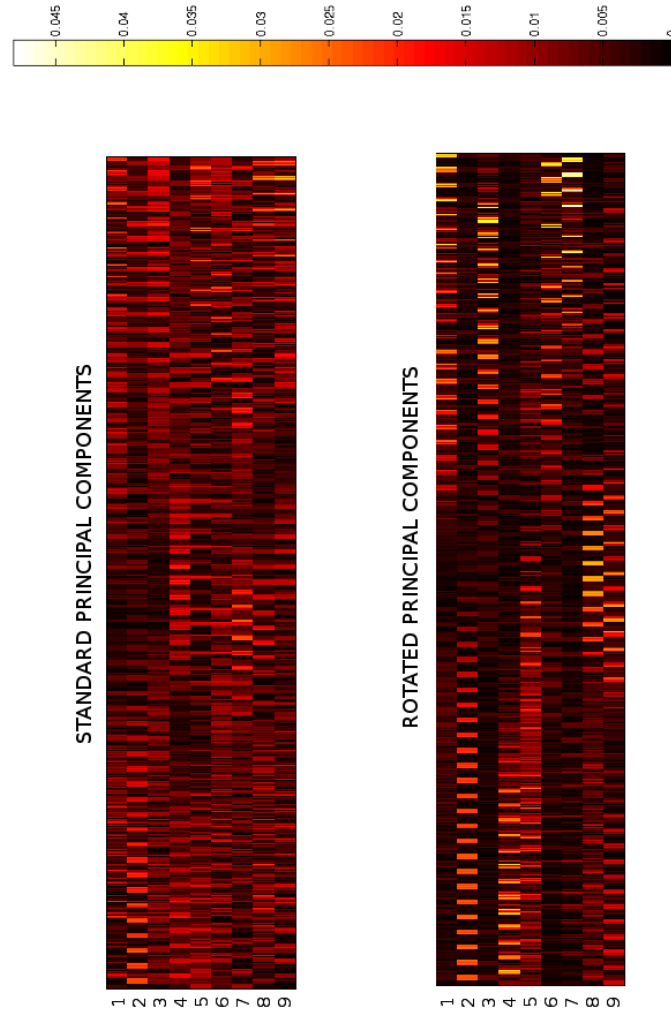
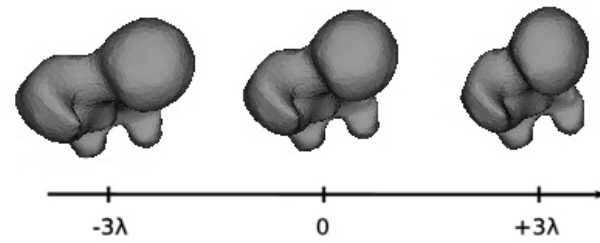
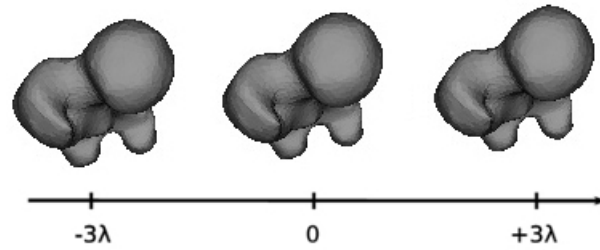


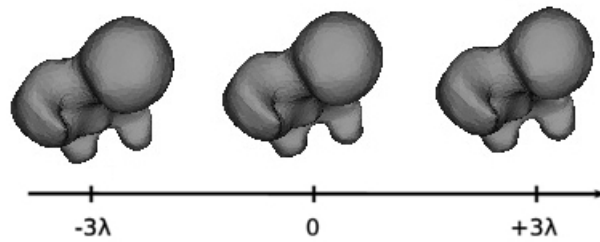
Figure 4.28: Change in the loading distribution after an orthomax rotation. Each row (1-9) of the two heatmaps corresponds to a component. After the rotation a reduced number of loadings becomes higher whilst the remaining are pushed to zero



(a) 1st Principal Component

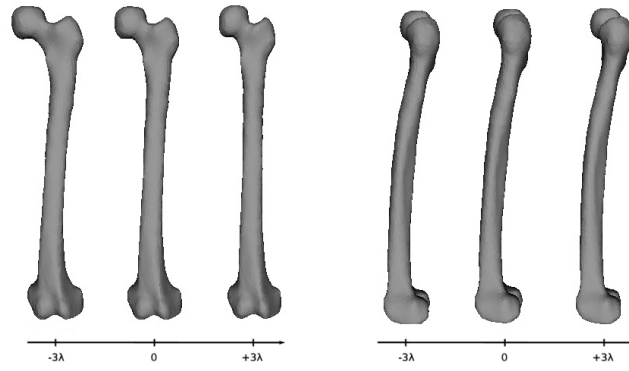


(b) 2nd Principal Component

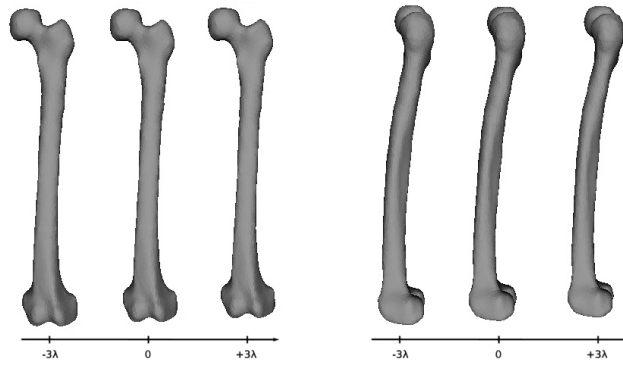


(c) 3rd Principal Component

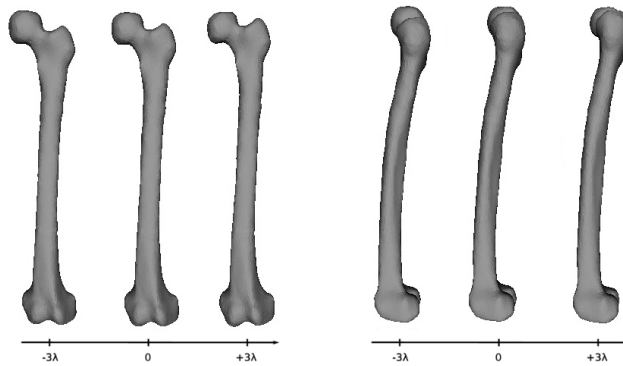
Figure 4.29: Transverse view of the effects of the first 3 rotated principal components on the shape.



(a) 1st Principal Component

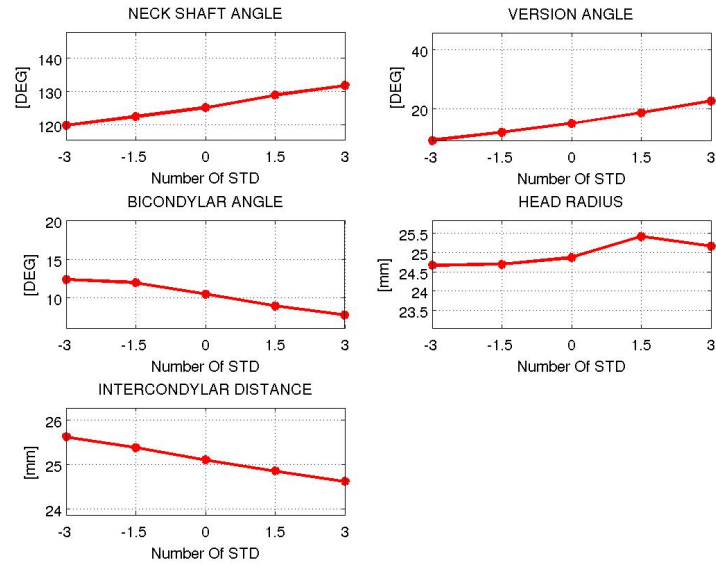


(b) 2nd Principal Component

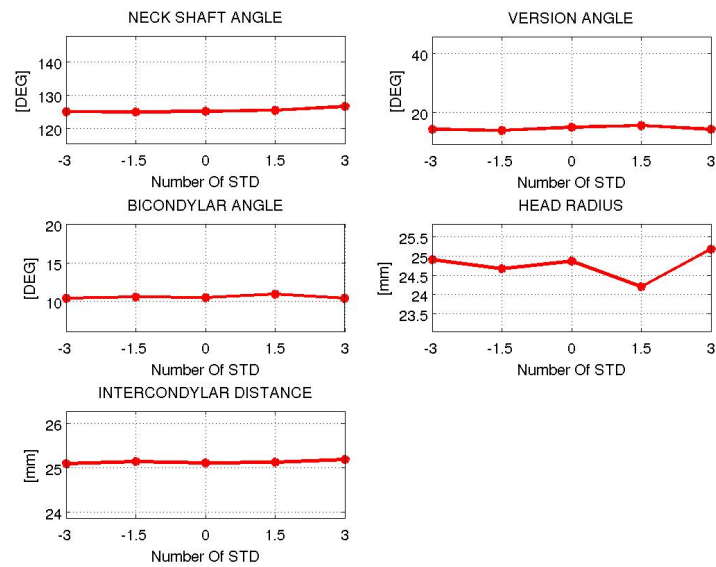


(c) 3rd Principal Component

Figure 4.30: Coronal and sagittal views of the effects of the first 3 rotated principal components on the shape.

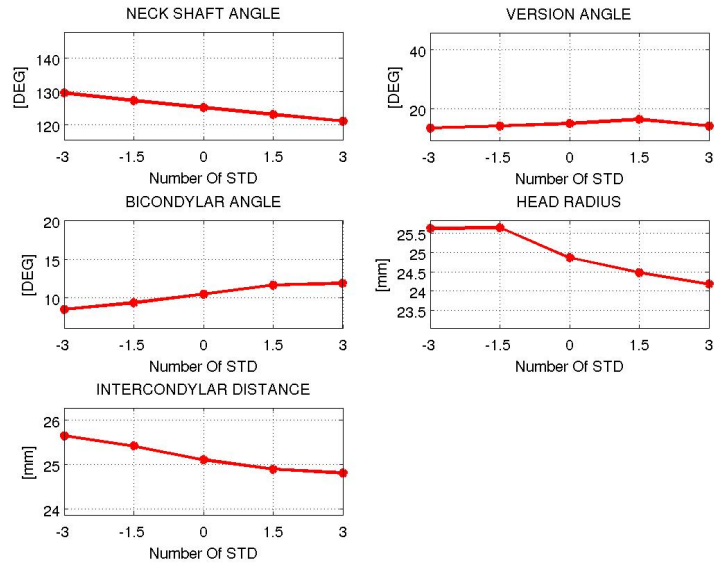


(a) 1st Component

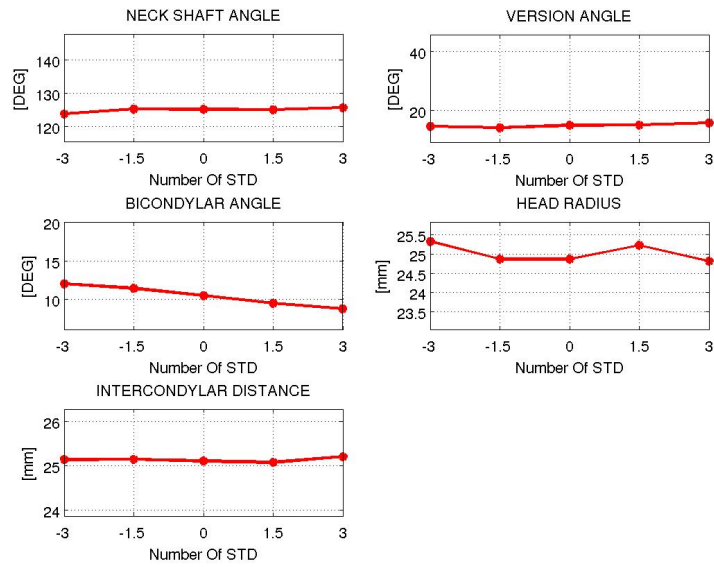


(b) 2nd Component

Figure 4.31: Effects of rotated PCs 1-2 on the morphological measures

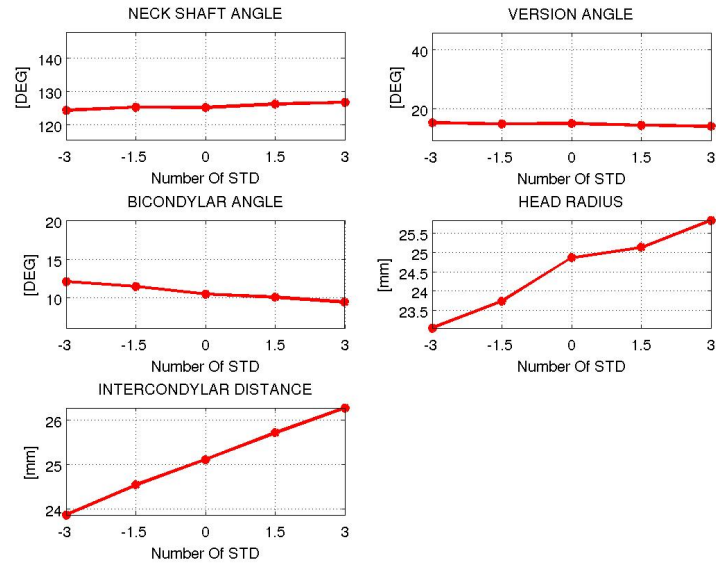


(a) 3rd Component

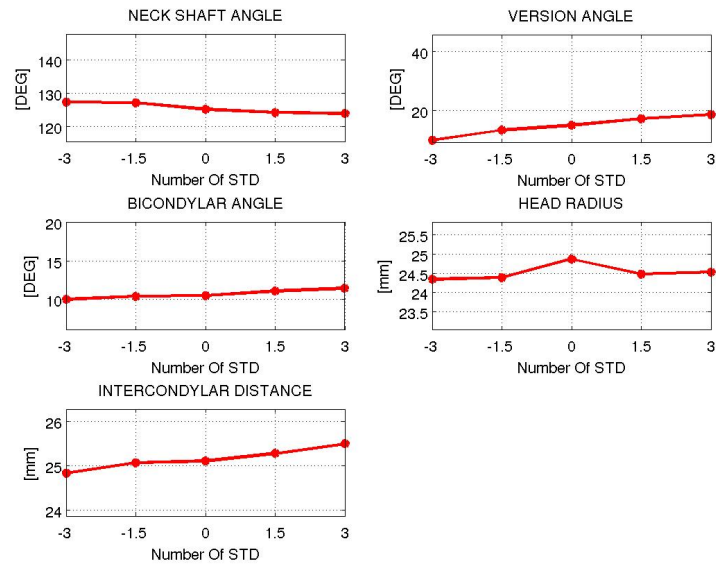


(b) 4th Component

Figure 4.32: Effects of rotated PCs 3-4 on the morphological measures

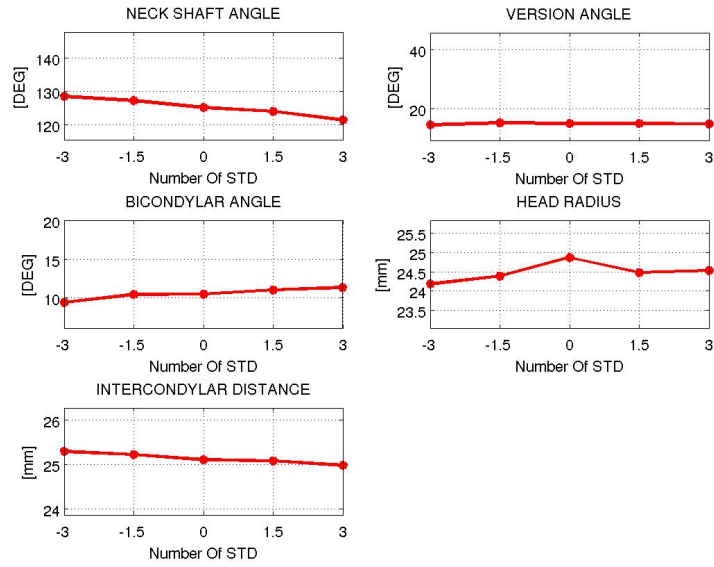


(a) 5th Component

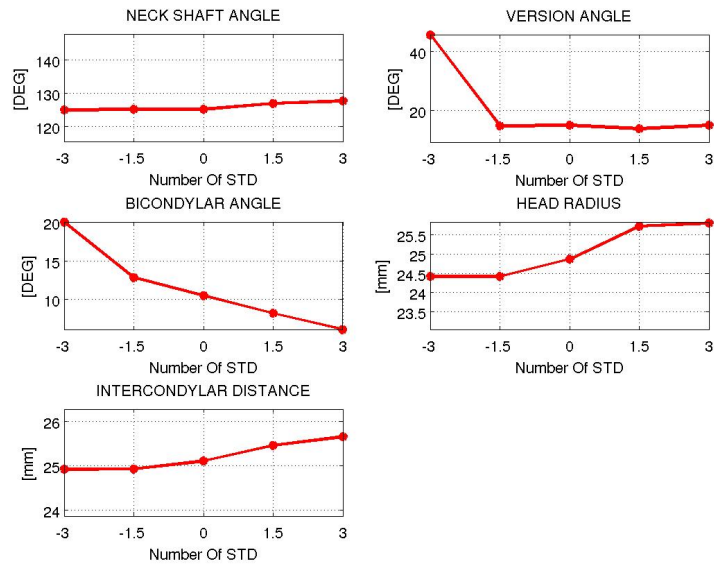


(b) 6th Component

Figure 4.33: Effects of rotated PCs 5-6 on the morphological measures



(a) 7th Component



(b) 8th Component

Figure 4.34: Effects of rotated PCs 7-8 on the morphological measures

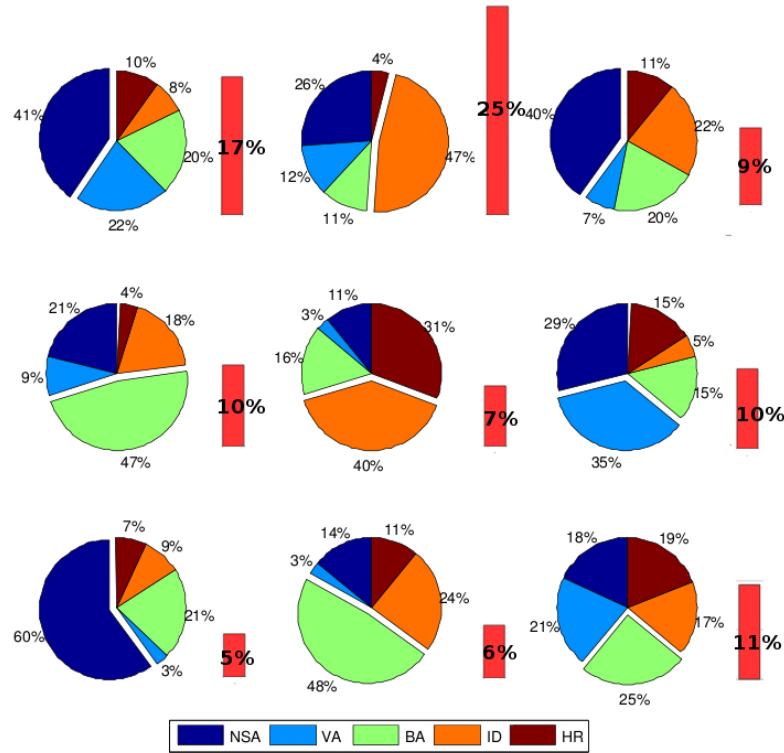


Figure 4.35: Pie charts describing the relative importance of each measure through the 9 components the rotated model comprises

had a predominant structure that experienced more than 40% of the overall variation explained for that particular mode of variation, therefore leading to more “understandable” effects.

The results for both the PCA model and its rotated counterpart are presented in Tables 4.2 and 4.3.

Component #	1	2	3	4	5	6	7	8	9
NSA [$^{\circ}$]	17.73	7.25	10.59	2.78	6.31	3.59	1.09	9.32	2.24
VA [$^{\circ}$]	14.13	2.02	3.23	6.43	7.11	8.12	5.84	2.68	2.89
BA [$^{\circ}$]	1.65	0.47	1.45	4.28	0.95	2.63	1.32	7.47	3.85
ID [mm]	5.7	2.4	2.1	1.85	0.1	0.57	0.56	1.35	1.53
HR [mm]	6.31	1.19	0.68	0.05	0.13	1.95	0.39	1.25	0.46

Table 4.2: Ranges of variation for the parameters for each component in the standard model

Component #	1	2	3	4	5	6	7	8	9
NSA [$^{\circ}$]	11.93	1.71	8.43	1.88	2.42	3.46	7.04	2.5	2.56
VA [$^{\circ}$]	13.32	1.68	2.93	1.62	1.16	8.73	0.63	1.17	6.12
BA [$^{\circ}$]	4.56	0.55	3.35	3.22	2.68	1.45	1.9	6.69	2.69
ID [mm]	0.75	0.98	1.48	0.51	2.81	0.18	0.35	1.39	0.75
HR [mm]	1	0.1	0.83	0.13	2.39	0.66	0.31	0.72	0.94

Table 4.3: Ranges of variation for the parameters for each component in the rotated model

Chapter 5

Medial Axis Transform

Although shape appears to be a very natural concept in everyday language, when it comes to describing it in a quantitative manner there are many different approaches. There are also different possible models of shape, none of them being the best in all circumstances, as it is the final aim of the model that determines what the best option is.

When computing a SSM, it is important to represent each instance of shape as a set of corresponding features, thus providing a way of comparing different shapes allowing for statistical analysis.

In Chapter 3, each shape was described using a set of points lying on its surface. These points were the nodes of a triangular mesh on the surface computed through the marching cube algorithm from the segmented CT scans.

Although this approach gave good results and it had been employed previously in other works in the literature (Yang *et al.*, 2006), it was felt there were other approaches that might increase the statistical strength of the model, by reducing the intrinsic redundancies of this representation.

Blum (1967) introduced a new idea to extract a novel descriptor of shape that can be particularly relevant for branching structures, as the femur or long bones in general. He proposed making the shape interact with itself in

order to find a particular representation that captures the symmetries of the shape and enables an easier detection of its features.

A two-dimensional shape was imagined to be embedded in a plane of dry grass. The outline of the shape was then imagined to be set on fire; eventually the fire would die out in those points corresponding to the medial axis (MA).

A more formal definition describes the MA of a 2D shape as the locus of points formed by the centres of the circles that are tangent to the shape in 2 or more points. If the information about the position of these centres is augmented with the radius of the tangent circles, then all the information about the original shape is preserved.

The MA including the radii forms an invertible transformation, called the Medial Axis Transform (MAT); starting from the MAT it is possible to reconstruct the original shape without any loss of information.

In the case of a closed shape, there are 2 MATs that can be considered: one lying inside the object and the other one outside; the information contained in each one of them is the same, therefore it is possible to consider just the inner MA without losing any information. An example of the two MATs for a two-dimensional shape is shown in Fig. 5.1.

By introducing the MAT, it is possible to define natural shape properties that would have required further analysis or that would not have been available altogether. The two shapes in Fig. 5.1 show both similarities and differences; the number of inflection points is the same but while the shape on the top is straight, the bottom one is bending and shows a different thickness at its ends. Just by considering the contours' properties, the comparison of the two shapes is not straightforward; even by considering differential properties, such as the number of inflection points and the total curvature, it is impossible to simply determine the differences between the two shapes.

However, by considering the two MAs it can be easily seen that the two shapes are similar since their MAs share the same topology, but it is also

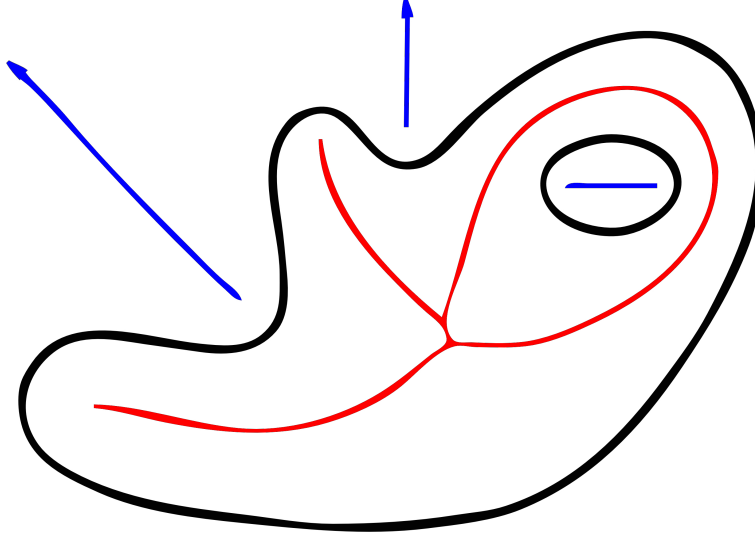


Figure 5.1: An example of the internal (red) and external (blue) MA of a smooth two-dimensional shape (modified from Amenta & Bern (1999)).

easy to capture their differences, since the MAs have different properties.

This ability of the MA to capture interesting features relative to the shape could be used to build a different kind of SSM that could have the advantages of being more statistically significant, as the MA maintains the same amount of information using fewer points.

The computation of a SSM based solely on a medial representation of shape has not been performed before; this work represents the first attempt to compute a SSM of 3D shapes based on their MA representation.

Unfortunately, the exact computation of the MA of a shape can only be performed on special classes of shapes. However, most of the times approximations to the MA can be computed and used instead.

One of the most widespread ways of computing these approximations uses a set of points sampling the surface and then considers the Voronoi diagram of these points, specifically its vertices.

Given a set of m points $\mathcal{P}$ in R^N , the space R^n would be partitioned in m

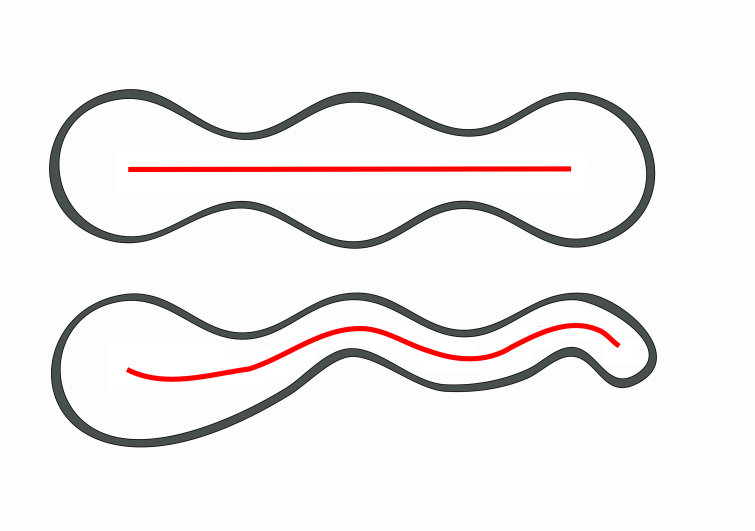


Figure 5.2: Two 2D shapes and their MA (red). The similarity between the two shapes is captured by the topology of the MA; both of them consist of a unidimensional line connecting two end points.

regions of space $V(\mathbf{p}_i)$ with $i = 1, 2, \dots, m$ and where $p_i \in \mathcal{P}$ is a point in the set. Each region would include those points $m \in R^N$ that are closer to the point p_i considered than to any other point in $\mathcal{P}$. A more formal definition is given in Eq.5.1

$$V(\mathbf{p}_i) = \{\mathbf{q} \in R^N, d(\mathbf{q}, \mathbf{p}_i) \leq d(\mathbf{q}, \mathbf{p}_j), \forall j = 1, 2, \dots, m \text{ and } j \neq i\} \quad (5.1)$$

where $\mathbf{q}$ represents a generic point in R^N and $d(.,.)$ denotes the Euclidian distance between two points. The Voronoi graph $Vor(\mathcal{P})$ consists of the boundaries between these regions. These boundaries are defined, in 2D, by segments and points. These points, called the Voronoi Vertices are of particular interest as they can be used to compute an approximation of the MA for 2D shapes. An example of a Voronoi Graph is given in Fig. 5.3.

Attali & Montanvert (1997), reviewed different methods to approximate

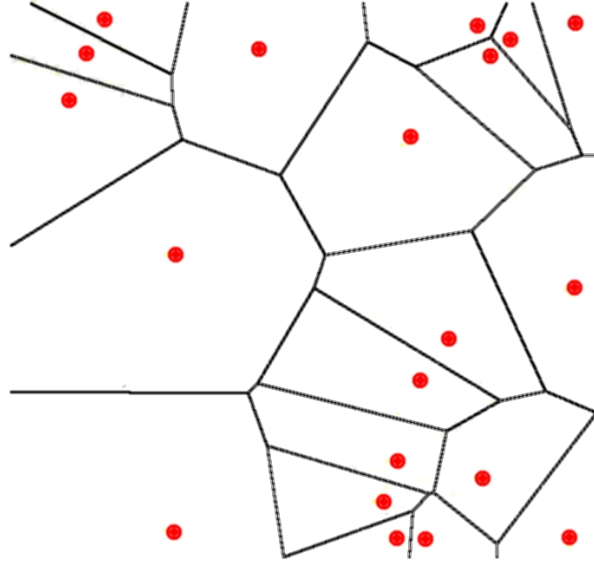


Figure 5.3: Voronoi Graph of a random set of point (red dots)

the MA of 2D-shapes. All of these employ the concept of Voronoi Graph and its dual concept, i.e. the Delaunay triangulation.

The Delaunay triangulation for a set P of points in the plane consists in those triangles (formed by connected any three points of P) that satisfy the condition of having a circumcircle, i.e. the circle passing through the three vertices of the triangle, that does not contain any point of P . An example of a valid and an invalid Delaunay triangle is give in Fig. 5.4.

The Delaunay triangulation is said to be the dual concept of the Voronoi Graph since the latter can be obtained by simply connecting the centres of the circumcircles of the triangulation.

Since all these approximations are computed on points sampling the boundary of the shape, the resulting MA would present artefacts that can be eliminated using appropriate pruning techniques.

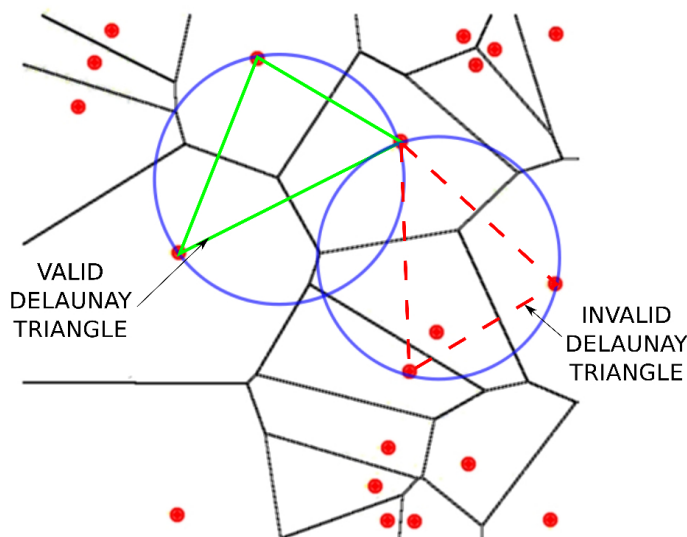


Figure 5.4: Valid (green solid line) and invalid (red dashed line) Delaunay triangles of the same set of points shown in Fig. 5.3

Brandt (1994) proved that for 2D-shapes the set of Voronoi Vertices converges to the real MA as the sample density approaches infinity. Unfortunately, this is not true for 3D-shapes, as there are some vertices that will always be close to the surface of the shape (Amenta *et al.*, 1998), regardless of the sampling density. The validity of the extension to three-dimensional shapes is spoiled by the occurrence of the so-called “slivers” in three-dimensional Delaunay diagrams. A sliver is a tetrahedron whose four vertices are almost co-circular on the boundary. The position of the Voronoi vertex corresponding to a sliver does depend on the four vertices, but it is generally not related to any feature of the shape. Amenta & Bern (1999) found that some of the Voronoi vertices, called “poles”, are guaranteed to be away from the surface. Indeed these poles, that are those Voronoi vertices that are the furthest from the sample point in each Voronoi cell, converge

to the MA as sample density goes to infinity. Amenta *et al.* (2001) proved this by considering the properties of the poles and subsequently proving the equivalence of the shape reconstructed from the poles to the original shape.

Starting from this result, the computation of the MA of 3D-shapes can be computed starting from the poles of the Voronoi Graph and pruning the resulting graph in order to get rid of those branches of the graph due to the sampling of the surface.

By considering the approximation of the MA computed on some points on the outline of a rectangle, it can be seen that provided the sampling density is high enough, a subset of the Voronoi graph includes a good approximation of the real MA (Fig. 5.5).

However, there are some branches in this approximation that are not related to the shape itself but rather to the sampling process.

Some methods have been introduced to perform this pruning (Attali & Lachaud, 2001) which however, as stated by Dey & Zhao (2004), suffer from the problem of being dependent on parameters that cannot be determined beforehand but are instead shape-specific.

In the same paper, Dey and Zhao proposed a new approach that eliminates this problem. Since their method was the one employed in this work, some key features of it are going to be presented in the following section.

Giblin & Kimia (2004) provide a formal classification for MA points of three-dimensional object; based on this classification, Leymarie & Kimia (2001) introduce the concept of medial scaffold. The MA is represented in the form of a graph, and different classes of MA points correspond to different features of shape. Although using this representation would probably simplify the matching process (Section 5.2), it introduces excessive complication in the model, and therefore the Voronoi approach of Dey & Zhao (2004) was the one employed.

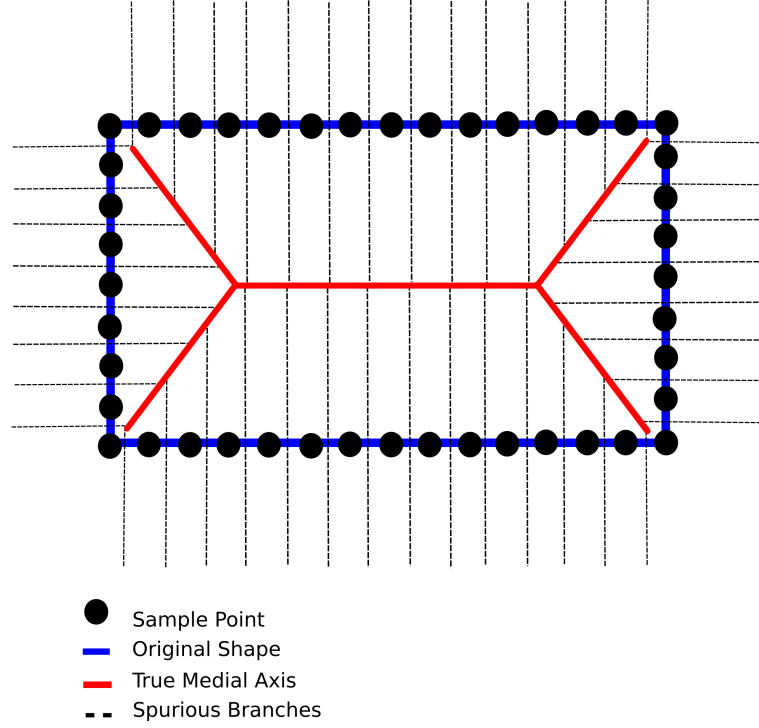


Figure 5.5: Approximation of MA for a rectangular shape. The red part of these approximations are close to the real MA, however the black parts are due to artefacts introduced by the sampling process

5.1 Approximation of the MA from the Voronoi Graph

Starting from a cloud point representation of a 3D-shape S , the aim is to find the set of *medial balls* of the shape itself. A medial ball satisfies the condition of being tangent to the shape in 2 or more points; the MAT of S is approximated by the union of these balls.

Each point of S has two medial balls: one for each side of the surface. By considering shapes topologically akin to a sphere, i.e. those that divide R^3 into two disconnected regions one internal and the other external to the

shape, an internal and external MA can be defined.

By definition, the Voronoi vertices of a shape satisfy the condition of being the centre of a sphere that contains two or more points on the shape. However, in order for this sphere to be medial, it needs to be tangent to the shape in those points.

A key role in the identification of those Voronoi vertices that represent spheres that satisfy this condition is played by the poles of the sample point. These are the points p_i^+ that are the furthest Voronoi vertices from p_i within those contained in the corresponding Voronoi cell. The vector $\mathbf{V}_{p_i} = p_i^+ - p_i$ is called the pole vector and it approximates the normal to the surface of S at p_i . The plane passing through the point p_i and having normal $\mathbf{V}_{p_i}$ is an approximation of the tangent plane to the surface at p_i .

When considering Fig. 5.6, if the poles of a Voronoi graph are taken into account, each one of those will have at least two points on the surface corresponding to it; when more than 2 points lie on the surface of the sphere, the points p_A and p_B are those that form the smallest angle Θ . The point m will be retained as a MA point if the angle Θ is below a certain threshold. Those points m that satisfy this, are said to fulfill the *angle condition*.

Dey & Zhao (2004) also proposed the inclusion of another condition, as the angle condition alone does not give satisfactory results in a density-independent manner. However, since all the shapes used here had the same sampling density, the MAs were computed based on the angle condition only. All geometric computations were performed using C++ code based on the CGAL library (Hemmer, 2009).

5.2 Matching Features on the MAs

Once the MAs of all the shapes have been computed, in order to compute the SSM, corresponding points in each MA have to be matched together. The MAs that are computed using the method just described still contain some

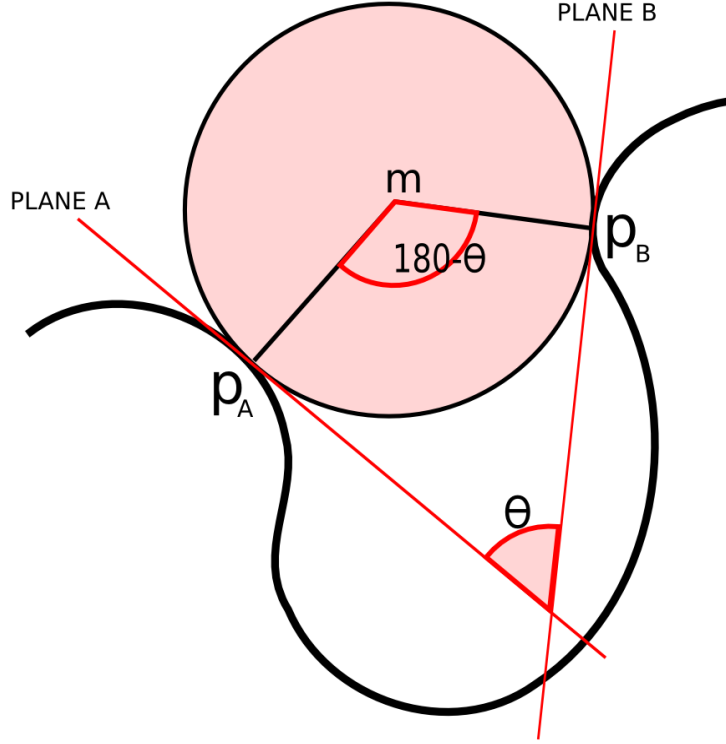


Figure 5.6: Generic Ball associated with Voronoi vertex m . p_A and p_B are two points on the surface and θ is the given by 180 minus the angle formed by the pole vectors of p_A and p_B (Adapted from (Dey & Zhao, 2004))

degree of redundancy. This can be dealt with by introducing a simplification algorithm as proposed by Tam & Fournier (1998). The proposed procedure deals with a general polyball approximation, but can be safely applied to the MATs as they are particular cases of polyball approximations.

The authors proposed the clustering of all the balls representing the shape and then examining the sphericity (ϵ) of each cluster, where $\epsilon = \frac{r}{R}$ is defined as the ratio between the radius of the biggest sphere enclosed in the cluster and the minimum enclosing spheres that contains all the spheres in the cluster itself. It was decided to use a bottom-up hierarchical clustering technique

based on a particular distance. Those groups of spheres that should be simplified need at least to be intersecting. A kind of metric that measures the intersection of two spheres is therefore needed.

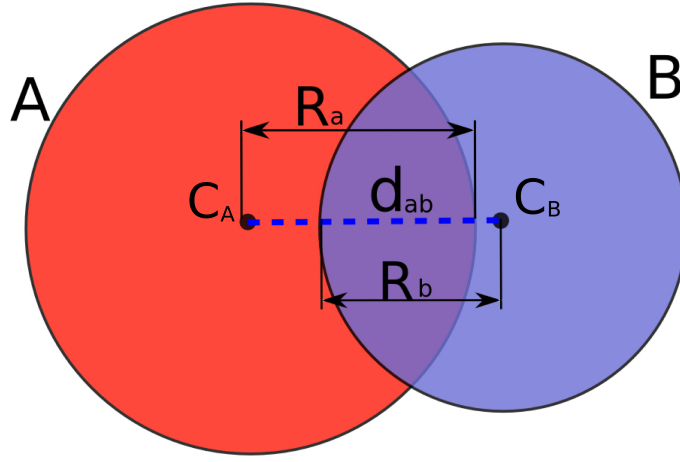


Figure 5.7: Two generic spheres will intersect only if the distance between their centres is less than the sum of their radii. By introducing a new metric Eq. 5.2, those spheres that have a negative-valued distance will intersect

When considering Fig. 5.7, if the distance d_{ab} between the centres of two spheres A and B is smaller than the sum of their radii R_a and R_b then the two spheres are intersecting. A bottom-up hierarchical clustering employing the distance defined in Eq.5.2 can be employed.

$$D(A, B) = d(c_A, c_B) - (R_a + R_b) \quad (5.2)$$

where $d(c_A, c_B)$ represents the standard Euclidian distance between the centres of the spheres and R_a and R_b are the radii of the spheres.

The clustering method consists of starting from the individual elements and then merging them progressively into bigger clusters. Starting from the

set of all the spheres of the MAT, those that are closest to each other will be merged to form a cluster of spheres. Then the distance between the spheres and the newly formed clusters is computed and new clusters are formed. The procedure terminates when all the spheres form a single cluster.

In order to apply this algorithm, the distance between two spheres that has just been introduced needs to be extended to deal with clusters of spheres. Usually either the minimum or maximum distance between each element in a cluster and each element in the other cluster is taken as the intra-cluster distance.

The maximum distance approach, also known as *complete linkage clustering* (Tan *et al.*, 2006), seemed to be the best choice for the simplification for the MATs; the distance in Eq. 5.2 is negative when the two spheres are intersecting. If this distance is employed to measure the distance between two clusters A and B and one of the spheres in cluster A is not intersecting the spheres in cluster B then, following the maximum criterion, the intra-cluster distance will be positive.

Since only clusters of mutually intersecting spheres need to be considered for simplification, the complete linkage clustering will successfully identify those clusters that might be simplified as the one that had been merged based on a negative intra-cluster distance.

When considering Fig. 5.8, the initial hierarchy, or level 1, will consider each of the elements of the set as unitary clusters. At the second level, the elements (C),(D) and (E),(F) will be merged together to form two new clusters (CD) and (EF). The procedure is repeated, until a single cluster (ABCDEF) is created.

The representation in Fig. 5.8 is called a dendrogram. According to the needs, the dendrogram can be “cut” at different levels. For the purposes of MATs simplification, the dendrogram related to each MAT was cut so that no cluster was formed by merging two clusters with a positive intra-cluster distance, thus considering only groups of spheres that were intersecting.

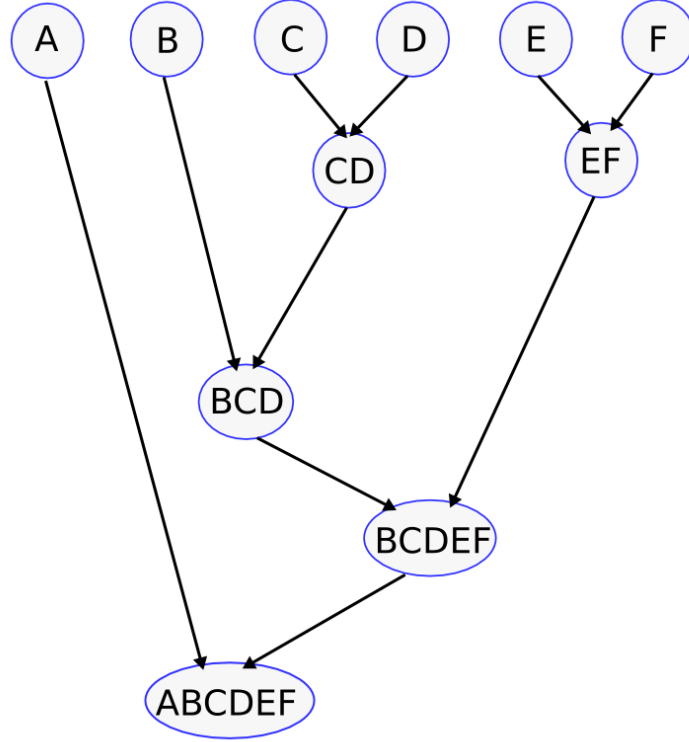


Figure 5.8: Results of agglomerative clustering are usually represented in a tree-like graph called dendrogram. The main idea of this kind of algorithm is that the tree can be cut at different levels according to a specific criterion

The sphericity ϵ of each cluster was computed, and those clusters with $\epsilon > 0.95$ were approximated by the biggest sphere in the cluster itself.

Whenever this condition was not verified for a particular cluster $C_{<\epsilon}$, smaller clusters within the cluster itself were considered and their sphericity was computed. This procedure was repeated until no further simplification could be performed.

After the medial representations were simplified, the different MATs needed to be matched so that corresponding spheres could be compared. One pos-

sible approach was presented by Tam & Fournier (1998). In their paper, a technique for interpolation between images based on union of spheres was developed. Each image was transformed into a three-dimensional representation that was subsequently approximated by a union of spheres.

Although the results of this matching allowed for a good interpolation between two 3D shapes, they were not good enough for the purpose of building a SSM; despite trying different weights on the distance function proposed, the matching were never good enough as proved by the resulting mean shape obtained after the matching (Fig. 5.9).

Moreover, the matching of two bipartite graphs relies on the Hungarian algorithm (Kuhn, 1955) that, despite being more efficient than a brute-force approach, still requires long computations.

A different approach was employed to match the spheres between two MATs that was both faster and gave results that were good enough to build a SSM based on the MAs and MATs.

After the MATs of each femur in the data-set were computed, a Free Form Deformation (FFD), as proposed by Rueckert *et al.* (1999), was applied on the shape whose MAT was composed of the smallest number of points (reference shape) in order to approximate all the other shapes in the data-set. The final transformation was then applied to the centres of the spheres of the MA of the reference shape.

Each of these deformed MAs formed an intermediate approximated MA that allowed for the proper matching to be performed. For all the points of the deformed MA the closest point on the real MA was found and the position of its centre and the magnitude of its radius was considered to be the real MAT point corresponding to the point of the reference MAT.

In order to be more accurate, it might be possible to consider not just the Euclidean distance of the centres of the spheres of the MATs, but a weighted distance that also includes the difference between the radii of the spheres. However, the results were already good enough and this variation did not



Figure 5.9: The mean shape computed from the matching proposed by Tam & Fournier (1998). Although this approach gives good results for linear interpolation between two shapes it is not robust enough for the computation of SSMs

contribute any substantial improvement.

After the matching of the MATs is performed, the data set could be represented by a matrix $\mathbf{M}$, the definition of which is given in Eq. 5.3

$$\mathbf{M} = \begin{bmatrix} x_1^1 & x_1^2 & \cdots & x_1^k \\ y_1^1 & y_1^2 & \cdots & y_1^k \\ z_1^1 & z_1^2 & \cdots & z_1^k \\ R_1^1 & R_1^2 & \cdots & R_1^k \\ \vdots & \vdots & \ddots & \vdots \\ x_N^1 & x_N^2 & \cdots & x_N^k \\ y_N^1 & y_N^2 & \cdots & y_N^k \\ z_N^1 & z_N^2 & \cdots & z_N^k \\ R_N^1 & R_N^2 & \cdots & R_N^k \end{bmatrix} \quad (5.3)$$

where each column of $\mathcal{M}$ represents a shape vector that had been obtained by simply concatenating the co-ordinates x, y and z of the centres of each sphere of the MAT and the radius R.

Principal Component Analysis can be performed on this matrix to obtain a SSM based on the MAT representation of shape. Since the MAT contains two different kinds of information, two other models could be built just by considering the position of the centres and the radii of the spheres independently.

5.3 Results

5.3.1 Medial Axis Transform Model

The first model that was considered was based on whole MATs, thus including both the position of the centres of the spheres and their corresponding radii.

The mean shape was computed and then subtracted from each column of $\mathcal{M}$. The principal components and their corresponding variances were

calculating using MATLAB's function *princomp* with the *econ* option, since the number of examples, k , was smaller than the original dimension of shape-space.

The PCs are ordered according to their decreasing variances and then the components numbers are listed on the x-axis while the corresponding variances are represented on the y-axis, thus resulting in the scree graph relative to the PCA of the data (Fig. 5.10).

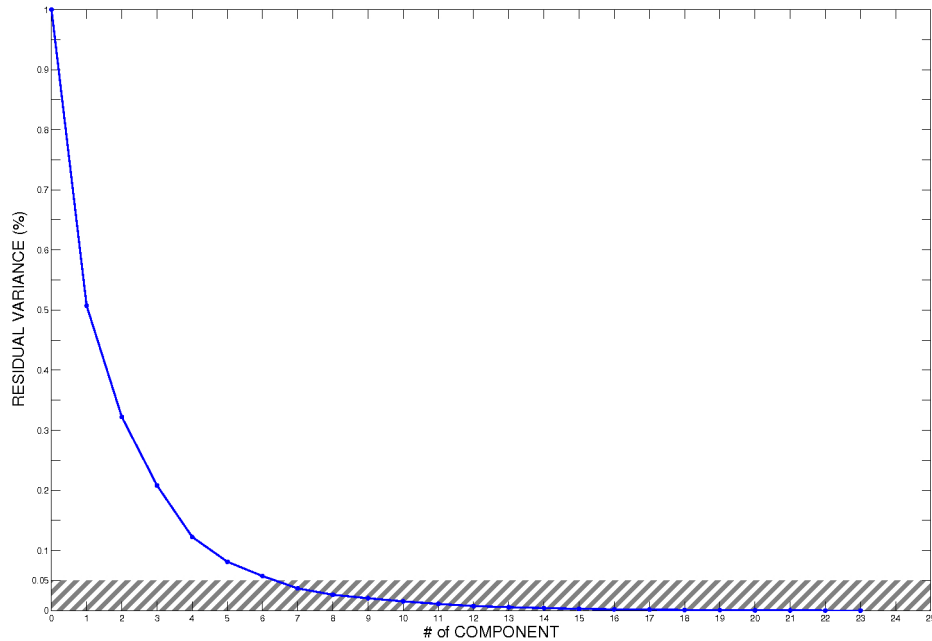


Figure 5.10: Scree graph relative to the MAT model

Analogous to what was stated in Chapter 4 the components that account for 95% of the variability were retained.

In this case, the first 8 PCs are those that account for most of the variability in the model. Since the inclusion of the 9th component in the model

produced no significant gain in variability, the model analysed comprised 8 components.

5.3.1.1 Visual Analysis

The behaviour of the model was first assessed visually; the main hypothesis of the SSM is that the data is distributed as a multivariate Gaussian, therefore the statistical properties of the distribution are fully determined by considering the mean and the standard deviations.

A good representation of the properties of the shape modelled would be given by sampling each component at suitable intervals up to $\pm 3\sigma$, where σ represents the standard deviation associated with the particular PC considered.

From statistical theory, 99.73% of the observation lie in the region between $\pm 3\sigma$. Thus by sampling the regions within these boundaries an overall portrait of the effect of the component examined can be drawn.

The effect of the 1st PC on the femoral shape is presented in Fig. 5.11.

The effect of the component is global and, as the value of this component is increased the femur becomes thicker and the NSA decreases and the medial condyle gets more prominent.

All the remaining 7 components were examined visually in the same way, the results of the qualitative visual assessments are given in Table 5.1.

The images representing the changes in the shapes at the two extremes for each component are presented in Figs.5.12-5.13.

5.3.1.2 Quantifying the effect of the Components

In order to have a quantitative description of the effect of each component on the shape, the same algorithm to extract meaningful geometrical parameters that was presented in Chapter 3 was employed to evaluate the MAT model.

The trends of each of the variables as the movement in the reduced shape space is limited along each PC are depicted in Figs. 5.15-5.18.

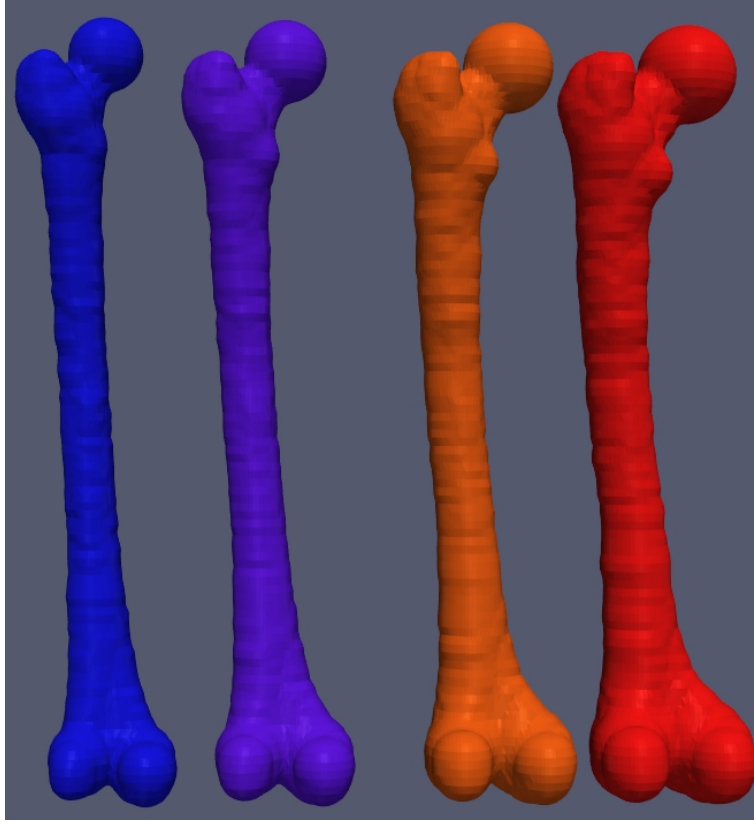


Figure 5.11: Effect of the 1st Principal Component of the shape. The component is sampled at -3σ -1.5σ $+1.5\sigma$ and $+3\sigma$

5.3.2 Medial Axis Model

By its own definition, it is evident that there are 2 different kinds of information contained in the MAT. From a rigorous point of view, the co-ordinates of the centres of the spheres and the corresponding radii are not homogeneous. Therefore a more correct approach would consider these two aspects of the representation separately. This approach is now taken.

The results of the modelling based on the centres of the spheres, i.e. on the MA only, are going to be presented in this section, while the one based on the radii information will be presented in Section 5.3.3.

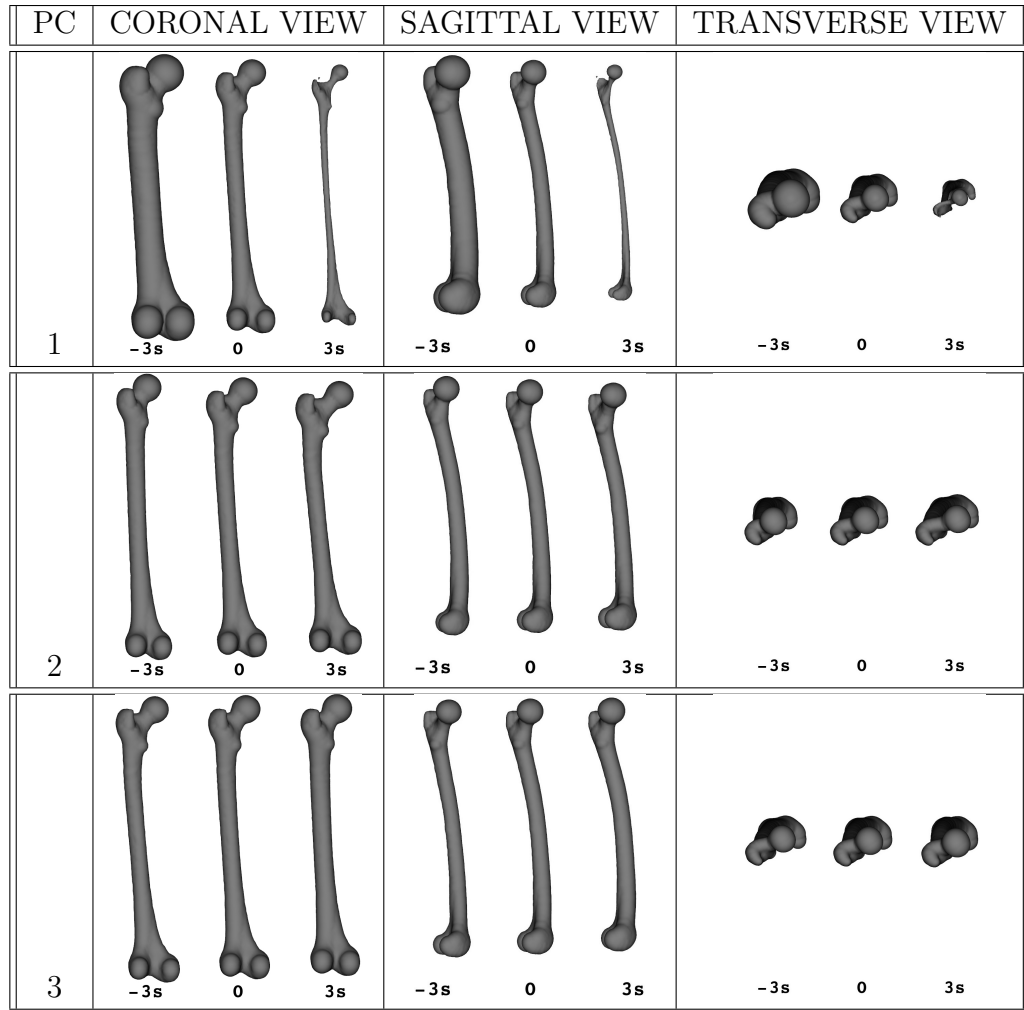


Figure 5.12: Effects of the MAT model PCs on femoral shape (1 of 3)

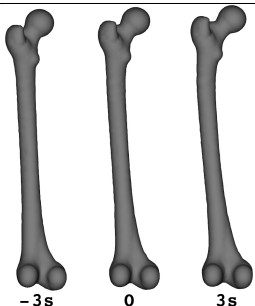
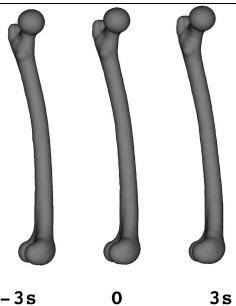
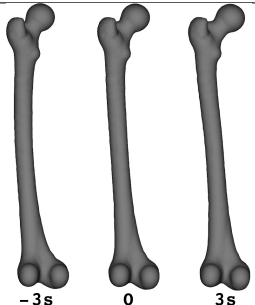
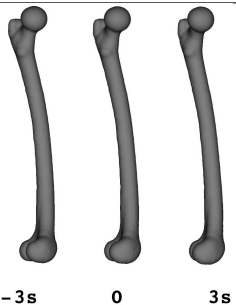
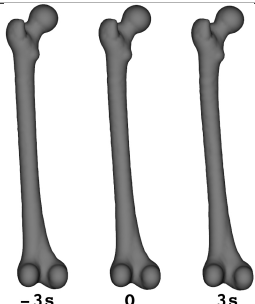
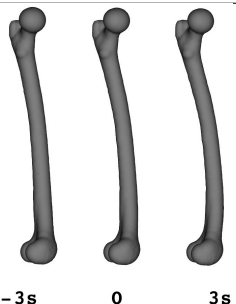
PC	CORONAL VIEW	SAGITTAL VIEW	TRANSVERSE VIEW
4	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
5	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
6	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s

Figure 5.13: Effects of the MAT model PCs on femoral shape (2 of 3)

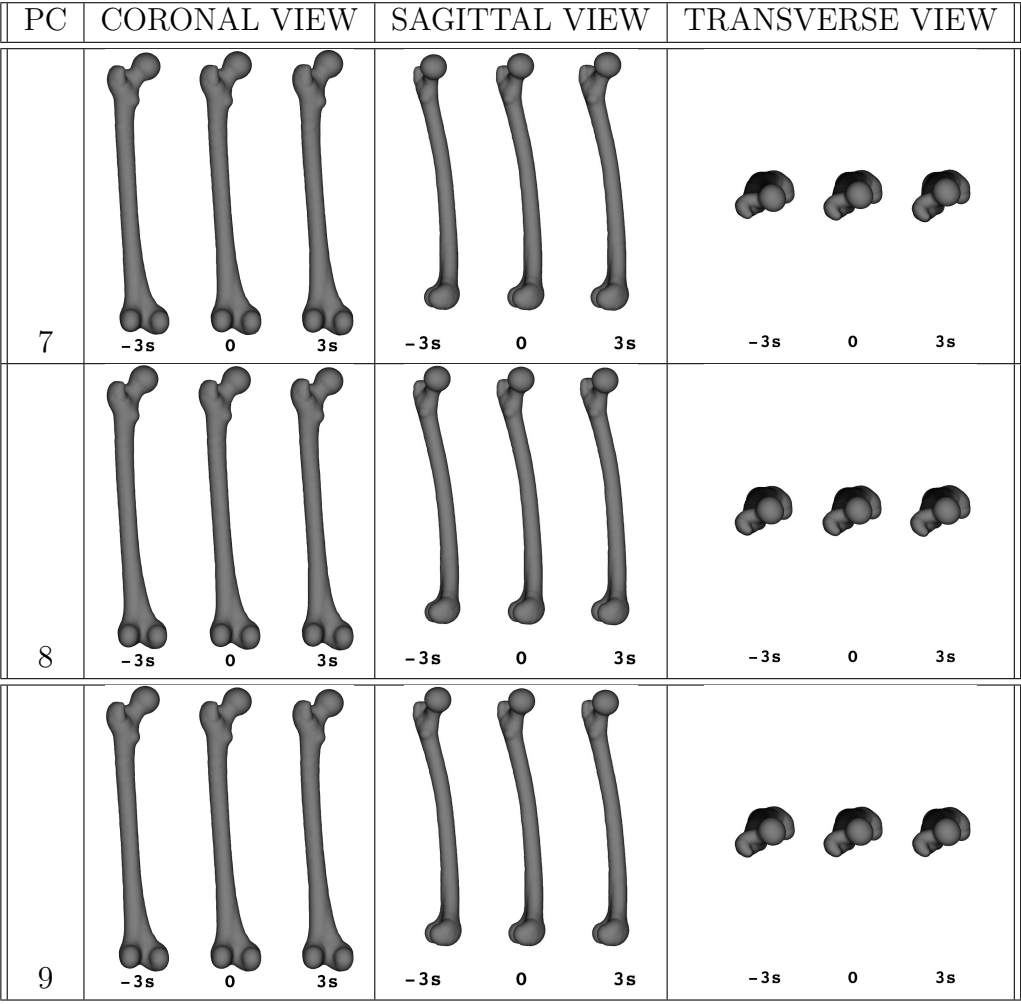
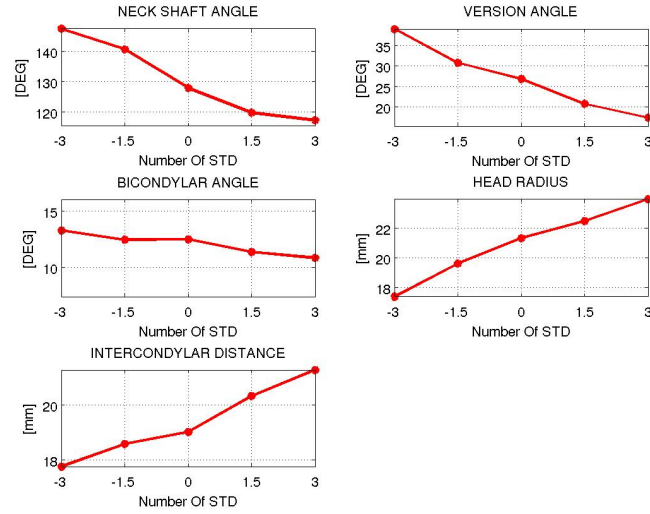
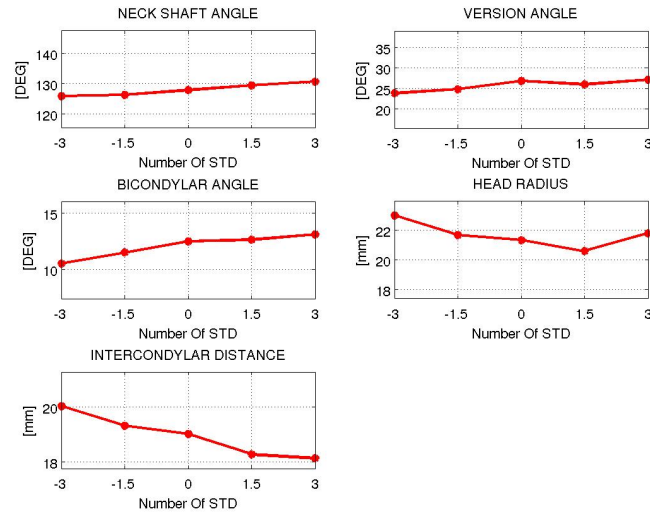


Figure 5.14: Effects of the MAT model PCs on femoral shape (3 of 3)

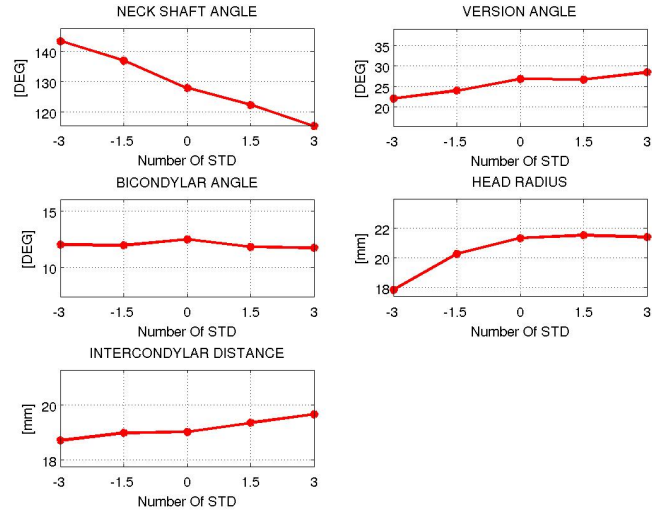


(a) 1st Component

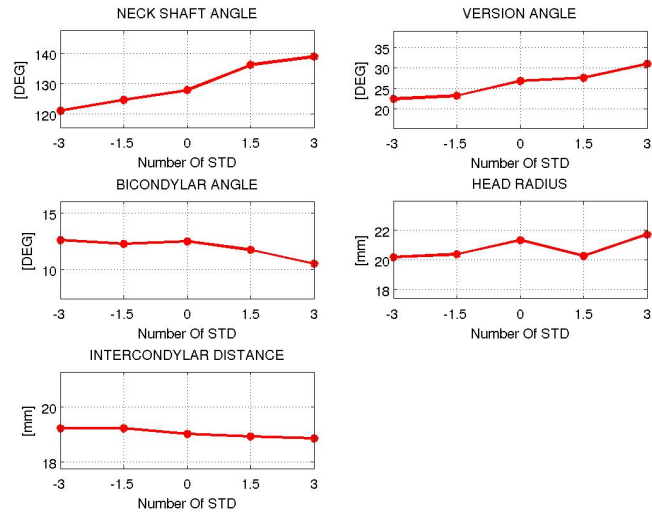


(b) 2nd Component

Figure 5.15: Effects of the principal components of the MAT model (1-2) on the morphological measures

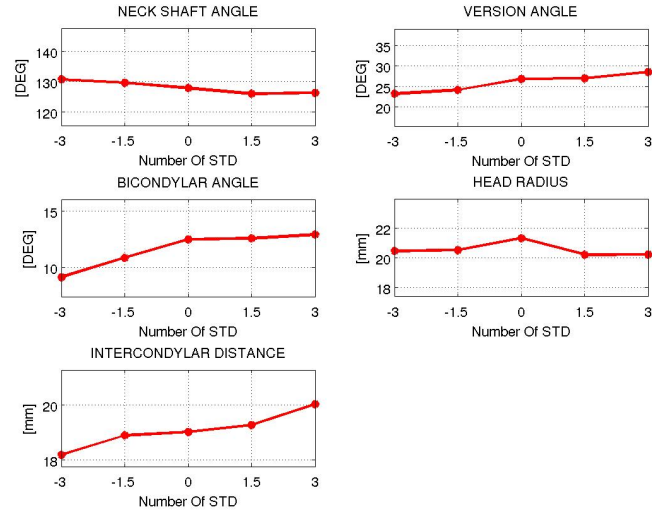


(a) 3rd Component

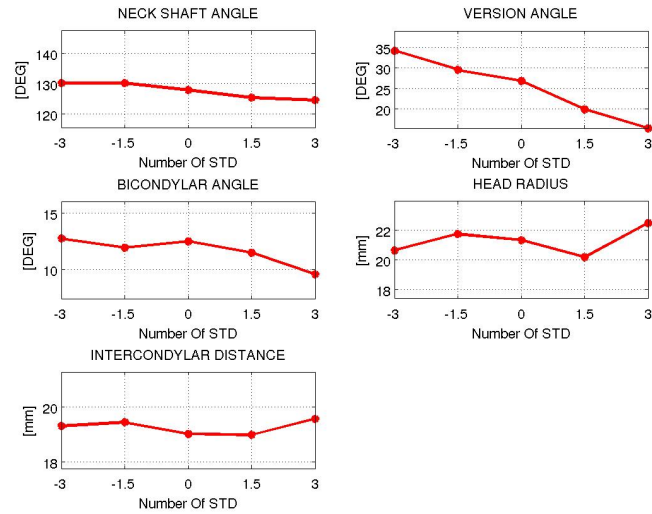


(b) 4th Component

Figure 5.16: Effects of the principal components of the MAT model (3-4) on the morphological measures

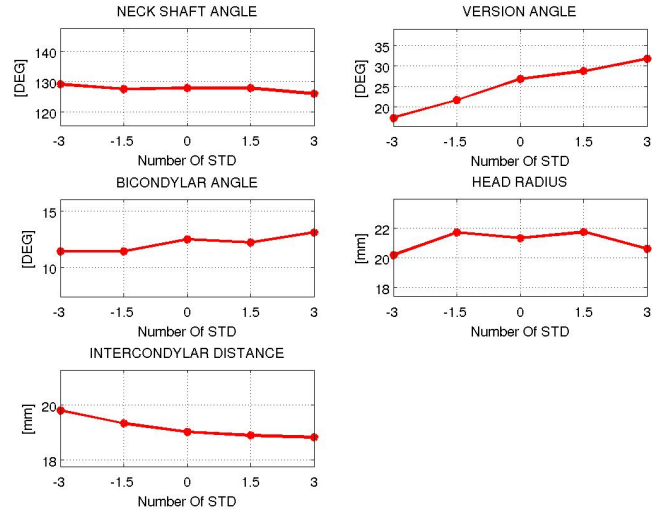


(a) 5th Component

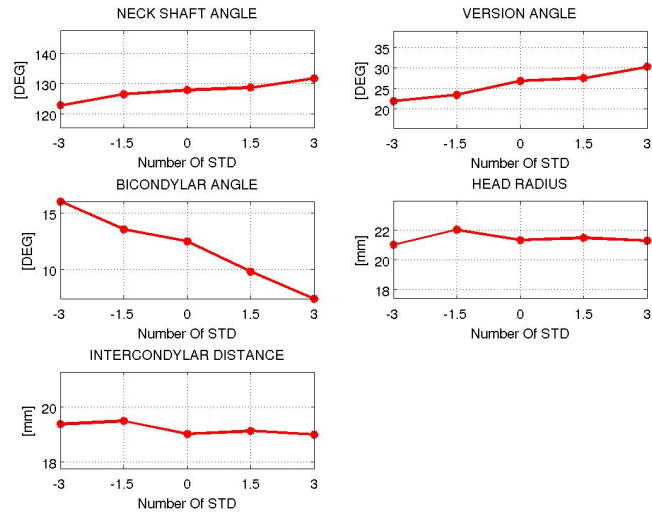


(b) 6th Component

Figure 5.17: Effects of the principal components of the MAT model (5-6) on the morphological measures



(a) 7th Component



(b) 8th Component

Figure 5.18: Effects of the principal components of the MAT model (7-8) on the morphological measures

Component	Effect of the Component
1	Increased thickness, increased NSA. Medial Condyle gets more prominent
2	Shortening of the bone. Condyles become smaller. Greater trochanter changes shape significantly
3	NSA increases in association with a decrease in anteversion
4	The shaft changes its bending, associated changes in proximal structures
5	Shaft gets thicker, condyles and greater trochanter become bigger
6	Version Angle changes and small variation in shaft bending
7	Change in condyle morphology
8	Change in intercondylar distance, head radius and neck length

Table 5.1: Effects of the various components on the femoral shape

The scree graph relative to the MA model (Fig. 5.19) looks very similar to the one for the MAT model (Fig. 5.10).

The scree graphs of the MA and the MAT models appear very similar in shape, although the variances that are involved with the MAT model are marginally higher. A direct comparison between the two scree graphs is given in Fig. 5.20.

Although there are some components that, at least in terms of explained variability, seem to be different from each other, the two models need to be examined in more depth to determine if their differences actually justify the existence of two separate models.

The same kind of visual and quantitative analysis was performed on the MA model in order to assess this.

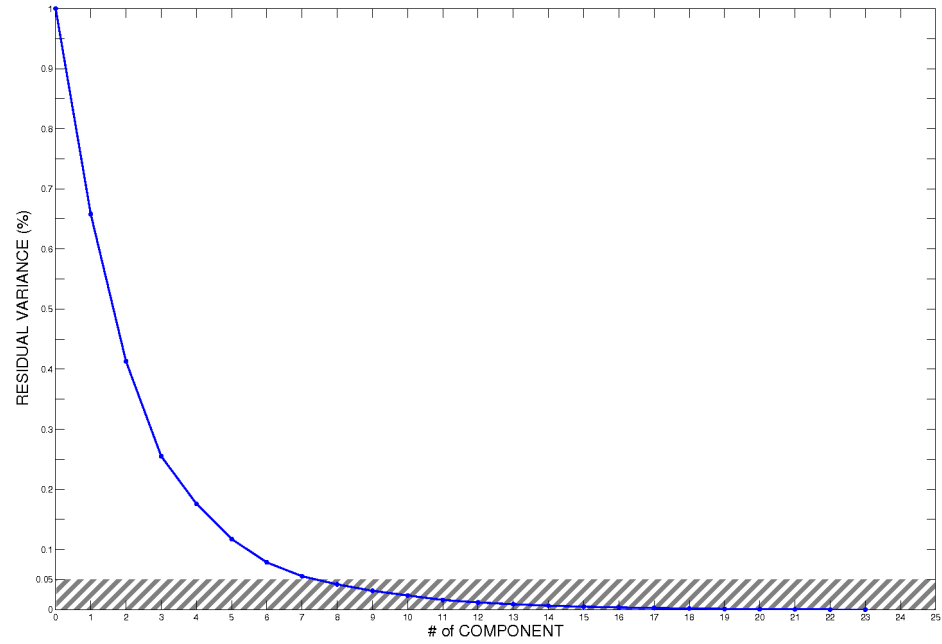


Figure 5.19: Scree Graph relative to the MA model

5.3.2.1 Visual Analysis

The effects of the model were first assessed visually, as was done for the MAT model in Section 5.3.1.1. The results of this qualitative analysis is given in Table 5.2.

The images representing the changes in the shapes at the two extremes for each component are shown in Figs.5.21-5.23.

5.3.2.2 Quantifying the effect of the Components

The morphological parameters were also examined for the MA model, the results are shown in Figs. 5.24-5.27.

By comparing the results of this model to those for the MAT model, it is

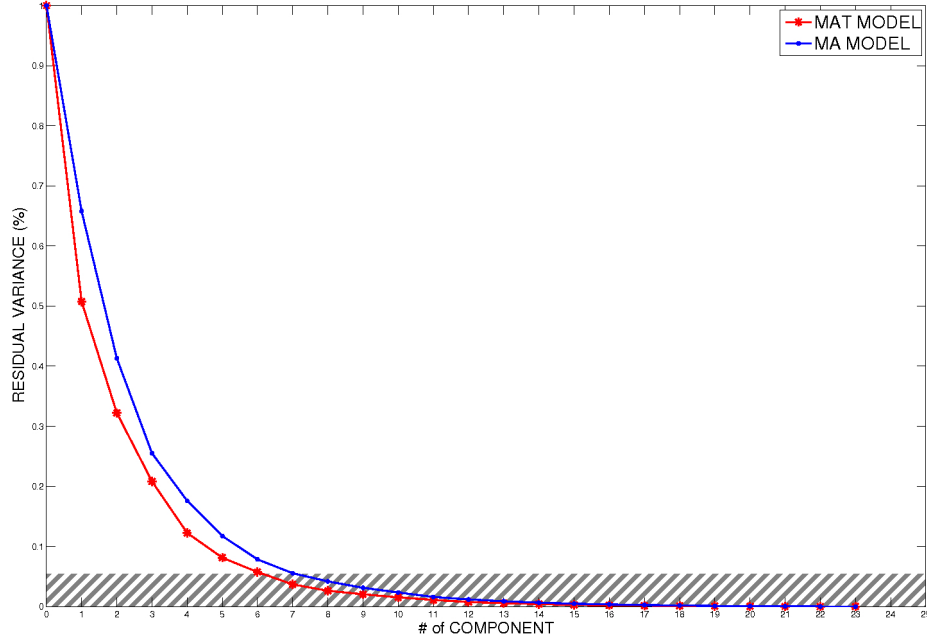


Figure 5.20: Comparison between the scree graphs of the MA (blue/continuous) and MAT (red/dashed) models

noticeable that most of the trends for the parameters are similar. This can be confirmed by considering the correlation coefficient between each of the measures for two corresponding components from the two models. The absolute values of these coefficients are represented as a heat map in Fig. 5.28.

The two models correlate nicely for those measures that are less dependent on the radius of the spheres of the MAT, however this cannot be said for the HR and ICD.

By considering the NSA, it is clear by its definition that its value depends on the capito-collar axis and the long axis of the shaft. These axes depend only on the position of the centres of the spheres not on their radius; the shapes of the neck and head and the shaft present cylindrical symmetry,

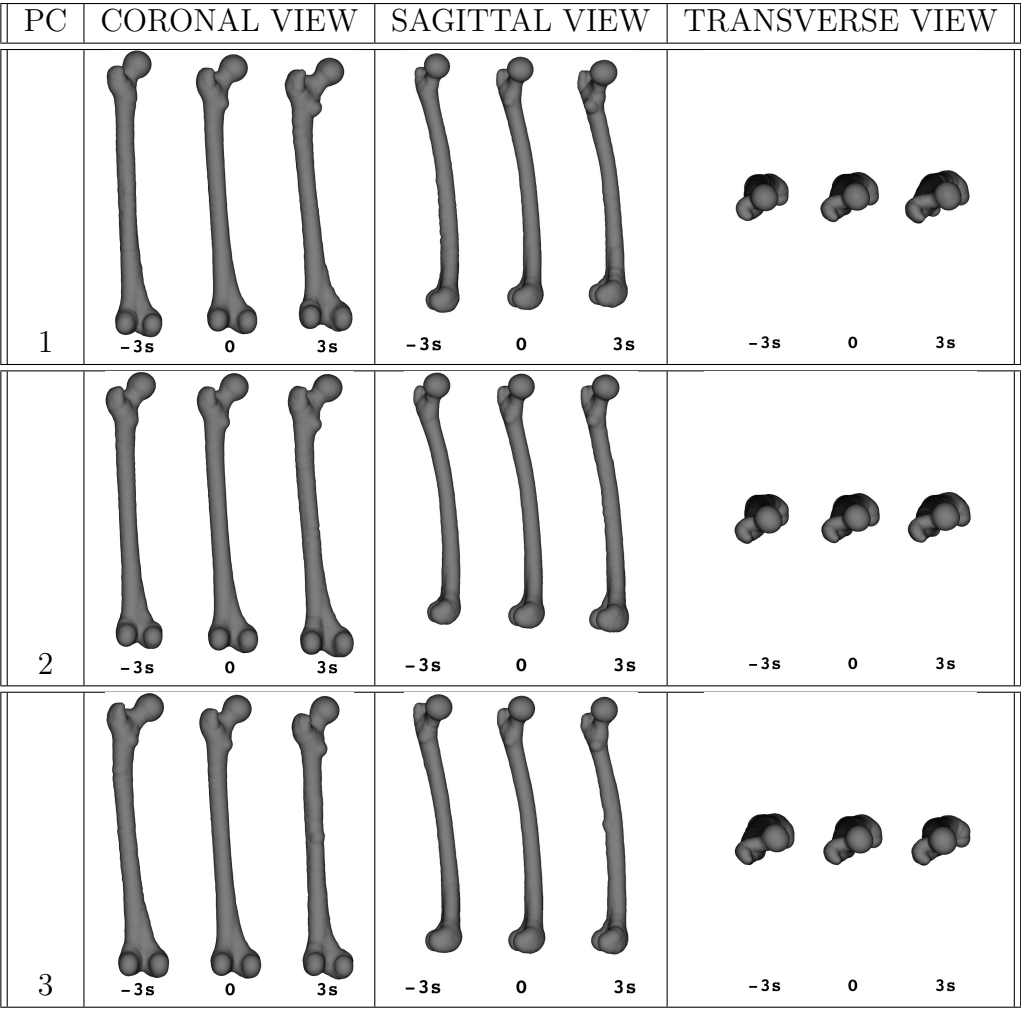


Figure 5.21: Effects of the MA model PCs on femoral shape (1 of 3)

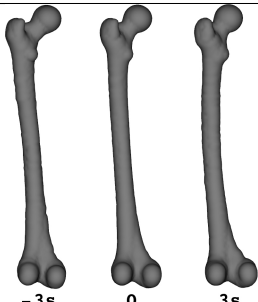
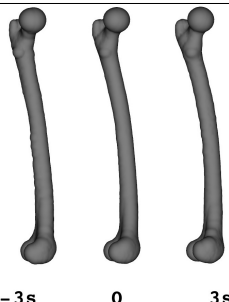
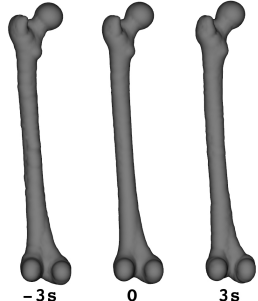
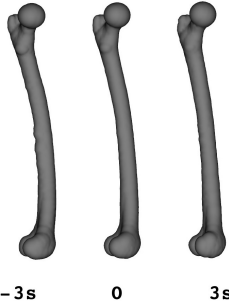
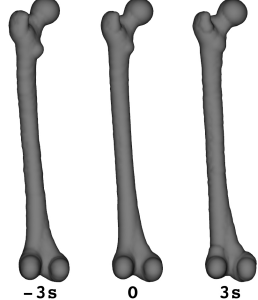
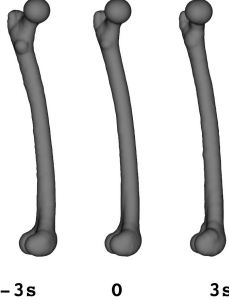
PC	CORONAL VIEW	SAGITTAL VIEW	TRANSVERSE VIEW
4	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
5	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
6	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s

Figure 5.22: Effects of the MA model PCs on femoral shape (2 of 3)

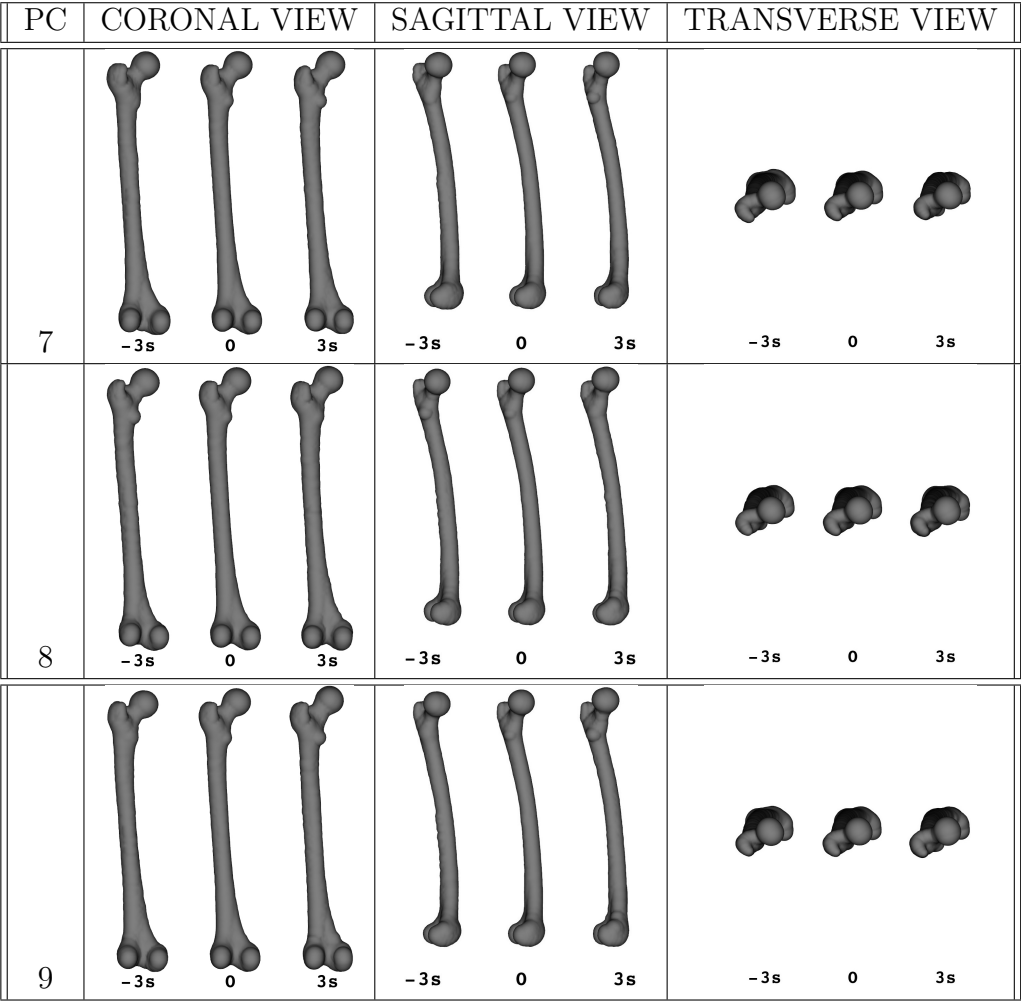
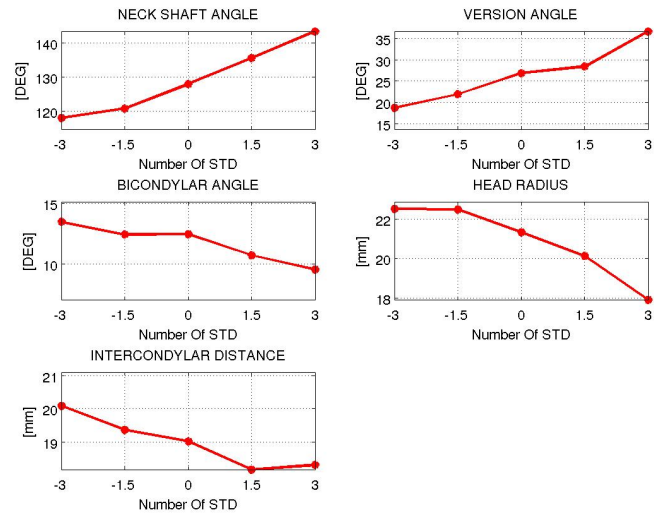
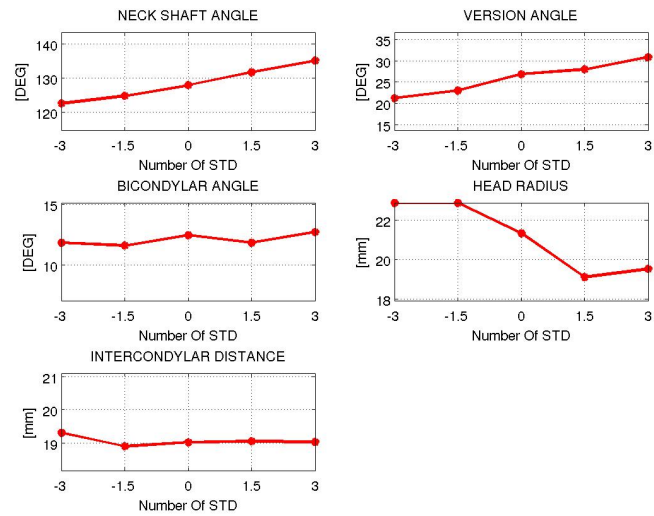


Figure 5.23: Effects of the MA model PCs on femoral shape (3 of 3)

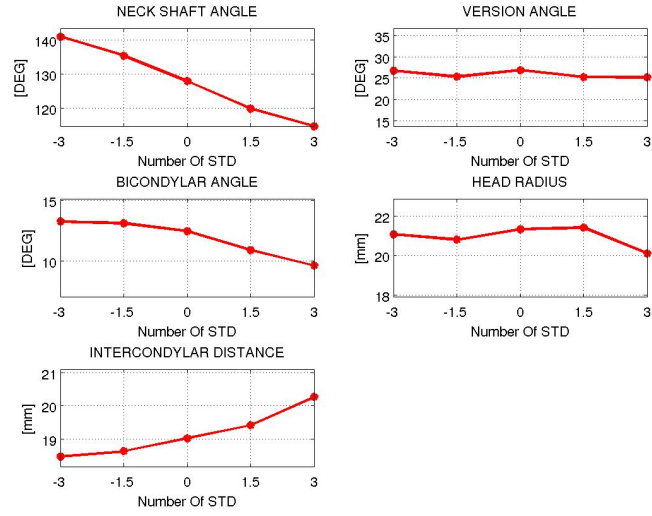


(a) 1st Component

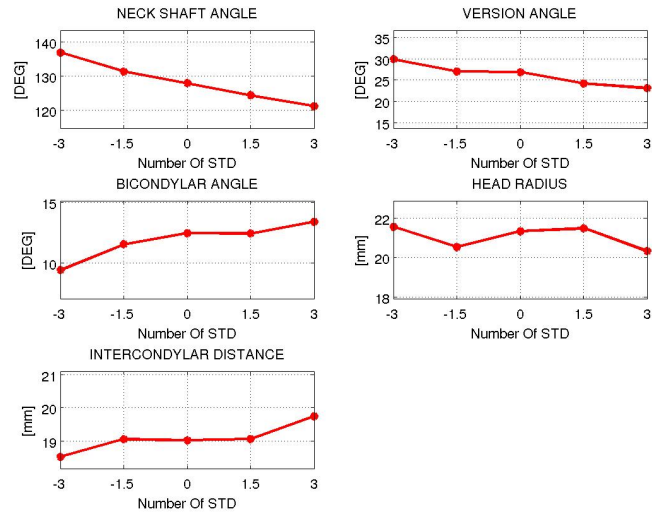


(b) 2nd Component

Figure 5.24: Effects of the principal components (1-2) from the MA model on the morphological measures

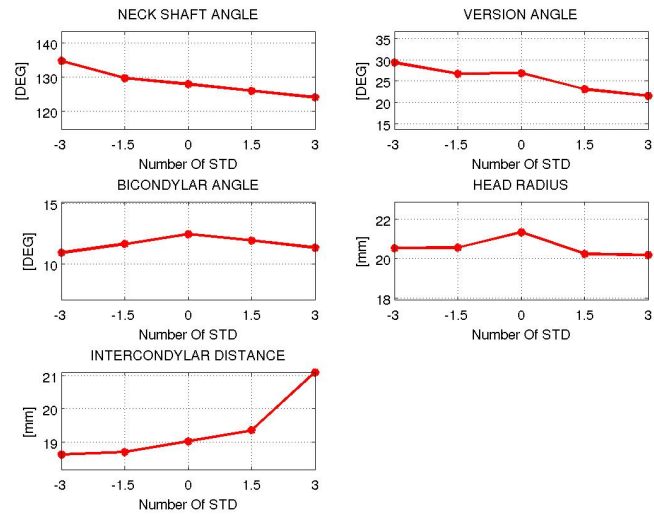


(a) 3rd Component

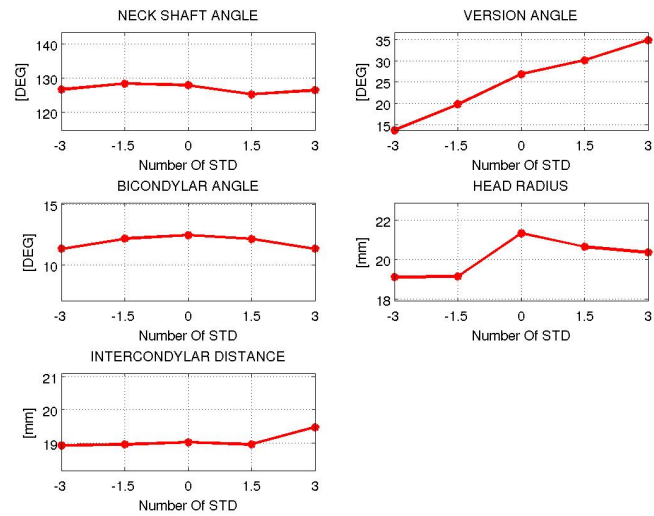


(b) 4th Component

Figure 5.25: Effects of the principal components (3-4) from the MA model on the morphological measures

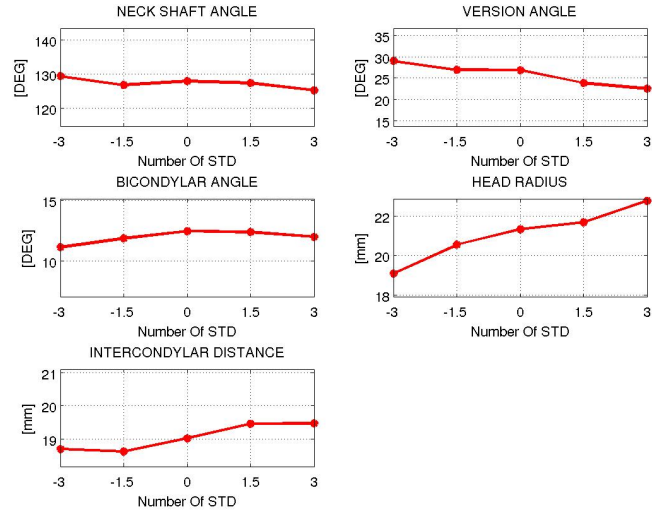


(a) 5th Component

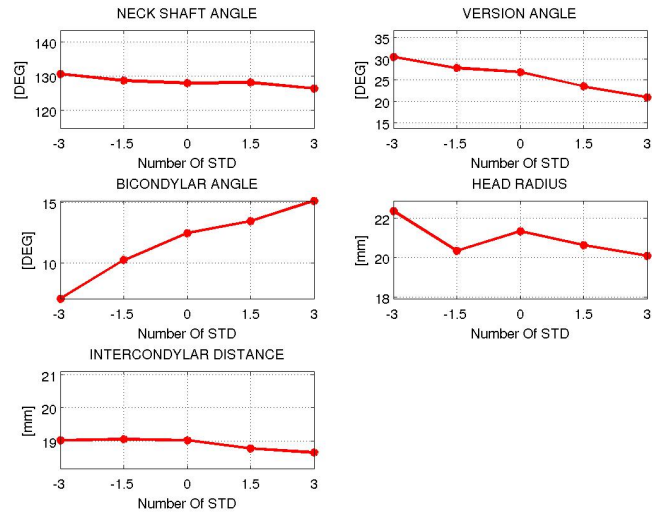


(b) 6th Component

Figure 5.26: Effects of the principal components (5-6) from the MA model on the morphological measures



(a) 7th Component



(b) 8th Component

Figure 5.27: Effects of the principal components (7-8) from the MA model on the morphological measures

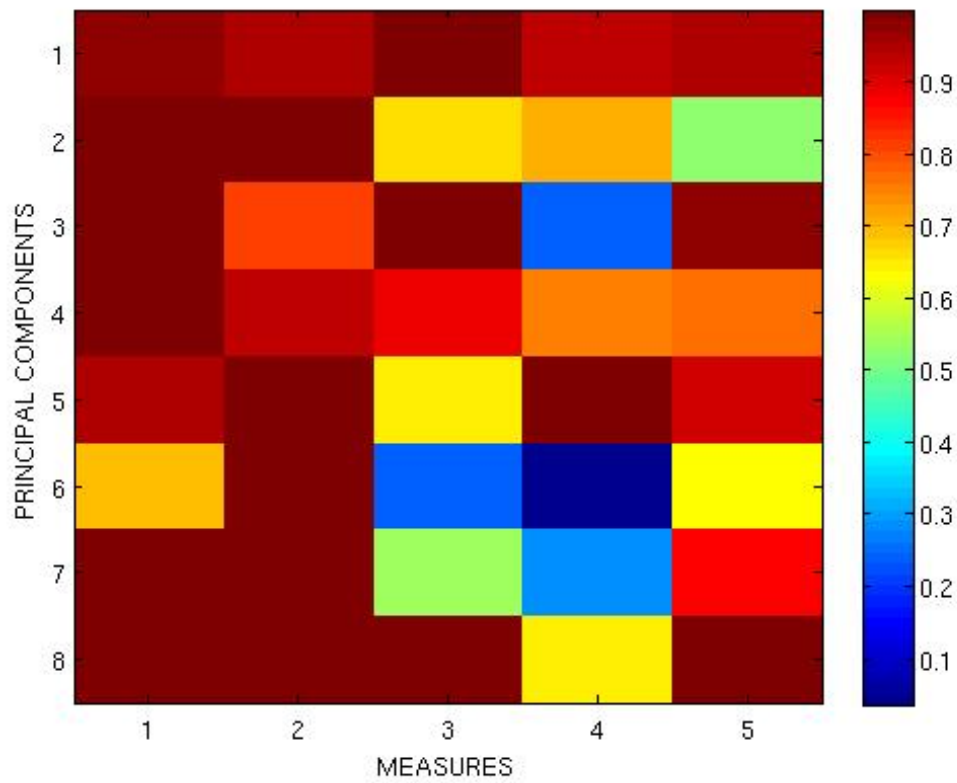


Figure 5.28: Heat Map representing the absolute value of the correlation coefficients between the measure obtained from the two models. Rows correspond to the PCs and columns to the different measures (1-NSA, 2-VA, 3-BA, 4-HR, 5-ID)

Component	Effect of the Component
1	Change in NSA. Medial Condyle gets more or less prominent. Shortening of the bone. Greater and lesser trochanter are also affected
2	Change on neck length. Change in version angle
3	NSA changes. Shaft changes its curvature in the frontal plane
4	Shaft changes its curvature
5	Change in version. Changes in shaft bending in sagittal plane. Medial condyle experiences some changes
6	Change in version. Lesser trochanter becomes less or more evident
7	Neck changes in length. Medial condyle experiences some changes
8	Change in intercondylar distance, head and neck shape

Table 5.2: Effects of the various components of the MA model on the femoral shape

therefore their MA would be unidimensional.

The axis of a cylindrical-symmetrical shape does not depend on the thickness of the shape, thus measures such as the NSA that rely on axes computed on cylindrical-symmetrical parts of the femur would be the same regardless of the inclusion of the radius information.

On the contrary, measures such as HR do not correlate between the two models; this is due to the fact that parts of the shape like the head of the femur are made up of a single sphere and the radius of this sphere actually determines the HR. This explains why the 6th components, for instance, exhibits a good correlation for the first two measures (NSA and NA) but not for HR.

As for the BA measure, it actually depends on both the position of the centres of the spheres of the MA and their radius. If the radius of one of the spheres corresponding to one of the condyles changes independently from

the radius of the other condyle then the BA would be affected even if the position of the centre of the sphere has not changed.

Therefore, the BA is affected both by the MA properties and the radii information, thus explaining the relatively poor correlation coefficients observed in the 3rd column in Fig. 5.28.

Given these considerations the two models provide two different ways of describing shape using different sets of directions.

In order to complete the MA model, the radius information needed to be analysed separately. The final model was considered to be the union of the two models, thus considering all the PCs of the two separate SSMs.

The results of the Radius Model are presented in Section 5.3.3.

5.3.3 Radius Model

Considering Fig. 5.29, it is clear that the first PC obtained by examining only the radii is by far the most important within the model; this component alone accounts for more than 50% of the overall variability contained in the data-set. By including other components up to the 8th the amount of variability does not change by a great amount, however when examining the effect of components 9 $\rightarrow$ 17, i.e. those components that should be retained to explain 95% of the variability, it is clear that they do not account for real changes in shape, but are actually modelling some noise due to the non perfect matching between MAT points. Thus just the first 8 components are going to be considered.

The effects of those components that were included in the model are shown in Figs. 5.30 to 5.32.

The 1st component is related to a change in thickness of the femur that involves every single structure; since the femurs included in the models are all fully-developed this component can actually distinguish between female and male bones, as generally males have thicker bones and muscle attachments are more prominent. However, since there was no information on the sex of

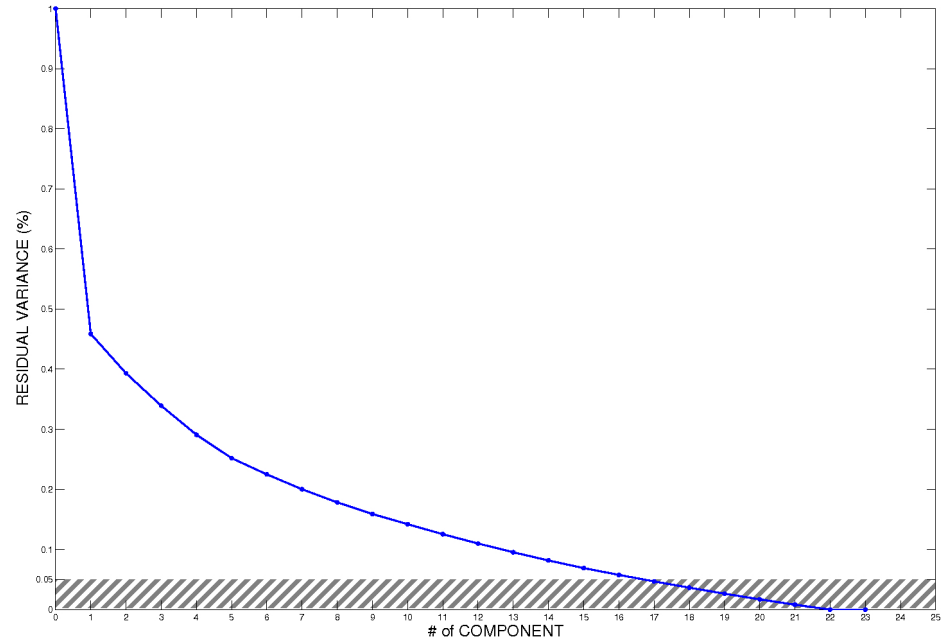


Figure 5.29: Scree Graph of the radius model

the specimens it was not possible to validate this hypothesis.

The third component is very interesting as it affects the dimension of the femoral head alone, whilst there is no other change in other structures of the bone.

Overall, by considering the radius information independently from the position of the centres of the spheres it is possible to decouple information about thickness from that involving shape.

However, as was pointed out earlier, there are some geometrical parameters that traditionally have been used to describe femoral morphology that rely both on the radius of the spheres and their position.

Both models are therefore meaningful and the choice of which one to use

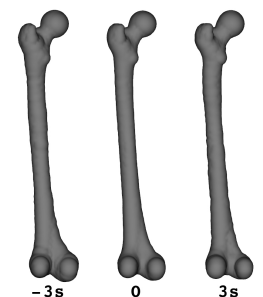
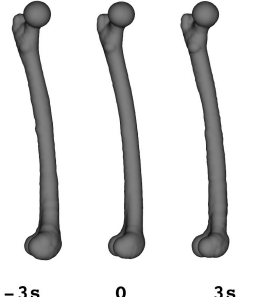
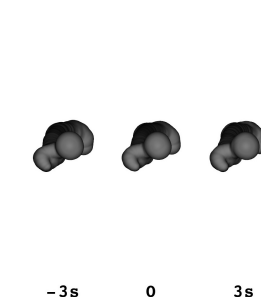
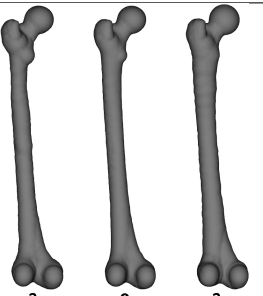
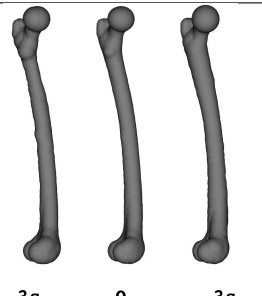
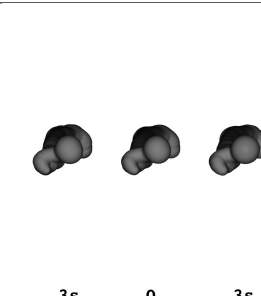
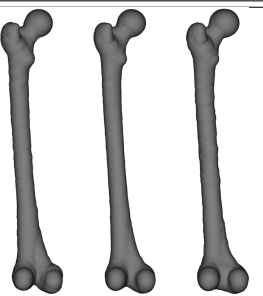
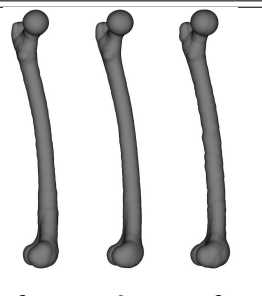
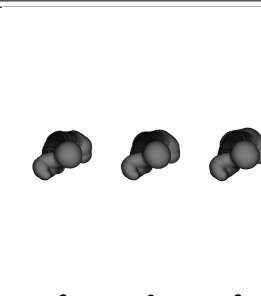
PC	CORONAL VIEW	SAGITTAL VIEW	TRANSVERSE VIEW
1	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
2	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
3	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s

Figure 5.30: Effects of the radius model PCs on femoral shape (1 of 3)

PC	CORONAL VIEW			SAGITTAL VIEW			TRANSVERSE VIEW		
4									
	-3s	0	3s	-3s	0	3s	-3s	0	3s
5									
	-3s	0	3s	-3s	0	3s	-3s	0	3s
6									
	-3s	0	3s	-3s	0	3s	-3s	0	3s

Figure 5.31: Effects of the radius model PCs on femoral shape (2 of 3)

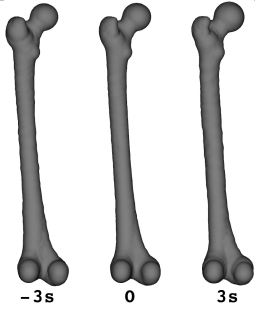
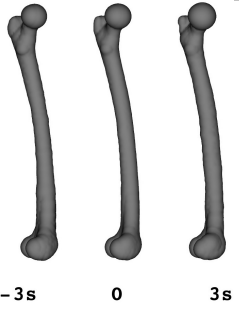
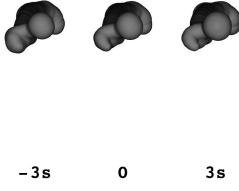
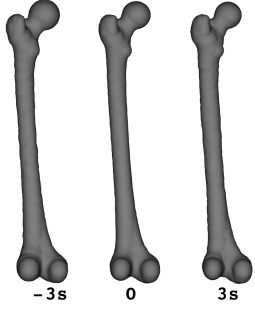
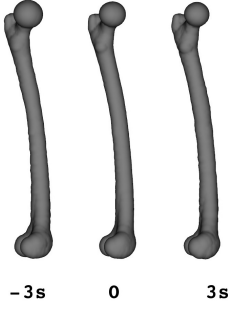
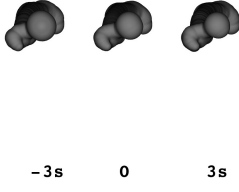
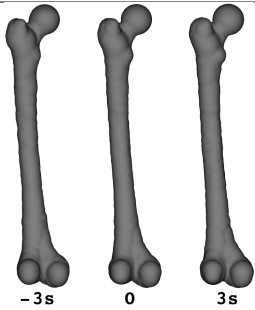
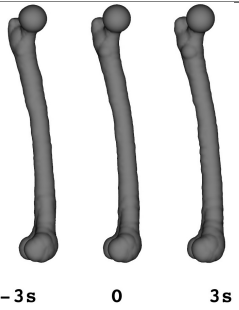
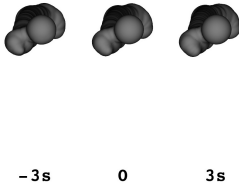
PC	CORONAL VIEW	SAGITTAL VIEW	TRANSVERSE VIEW
7	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
8	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
9	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s

Figure 5.32: Effects of the radius model PCs on femoral shape (3 of 3)

should depend on the final application.

5.4 Rotated Models

One of the main effects of MAT representation consists of the reduction of the dimensionality of the final model. Comparing the femoral shapes obtained from the point cloud based model to those given by the MAT model, it is evident that the former one is far more detailed.

However, the loss of some information in the MAT model should not be seen as a drawback of this representation; whenever modelling reality, there is always an intrinsic simplification of the real object or process modelled and it is this very simplification often allows a better understanding of the problem.

In Chapter 4, by applying an orthomax rotation to a SSM it was possible to obtain localised effects that could have a natural application in the description of pathological deformities. However, the high dimensionality of the vectors concatenating the coordinates of the points sampling the shape prevents the orthomax rotation to reach a structure very close to the perfect simple structure introduced by Bernaard & Jennrich (2003).

Since the rotation would ideally take into account each of these coordinates only once throughout the set of PCs, then inevitably, due to the high dimensionality there will be many points on the surface that would be influenced by the same rotated PC. Although this number would be considerably smaller compared to the original model, still the rotated model just provides a coarse subdivision of the shape having one component that relates to VA, one that relates to NSA and so on.

Due to the nature of these rotated models, it was decided not to measure the different shape parameters introduced in Chapter 2. This is due to the fact that there are different components that can be related to a change in the same parameters. By basing the evaluation of the model on these parameters

the main advantage of these rotated models could be missed. Each of the parameters can in fact be affected by morphological changes experienced by different structures. The rotated models would be able to describe these different mechanisms as will be shown in the remainder of this chapter.

5.4.1 Rotated MAT Model

Once the MAT model is computed it is possible to apply an orthomax rotation to it and evaluate the behaviour of the model. A visual evaluation of each component of the model is given in Figs. 5.33 to 5.35.

The effect of the rotation on a model with a lower dimensionality leads to a stronger localisations of the effects of the components.

The rotated model based on the point cloud representation was indeed able to describe those aspects that traditionally have been identified by doctors as important, such as the angular position of the neck axis relative to the condyles.

However, by rotating the MAT model something different is achieved: not only do the components relate to the same important aspects of shape, but they also differentiate between the different mechanisms that can lead to a change in one of these parameters.

Practically speaking, the rotated MAT model is able to identify those very structures the femur is composed of and also take into account all the possible different deformations these structures experience in the healthy population.

The effect of each rotated component on the shape is summarised in Tab. 5.3.

By looking at Fig.5.33, it can be seen that the behaviour of the first component is not consistent with a Gaussian distribution, i.e. the femur assumes a non realistic shape. This can be due to the different information represented in the shape vector; the radius and the coordinates of the centres represent qualitatively different information. Using the correlation matrix in the computation of the model would introduce a normalisation of the

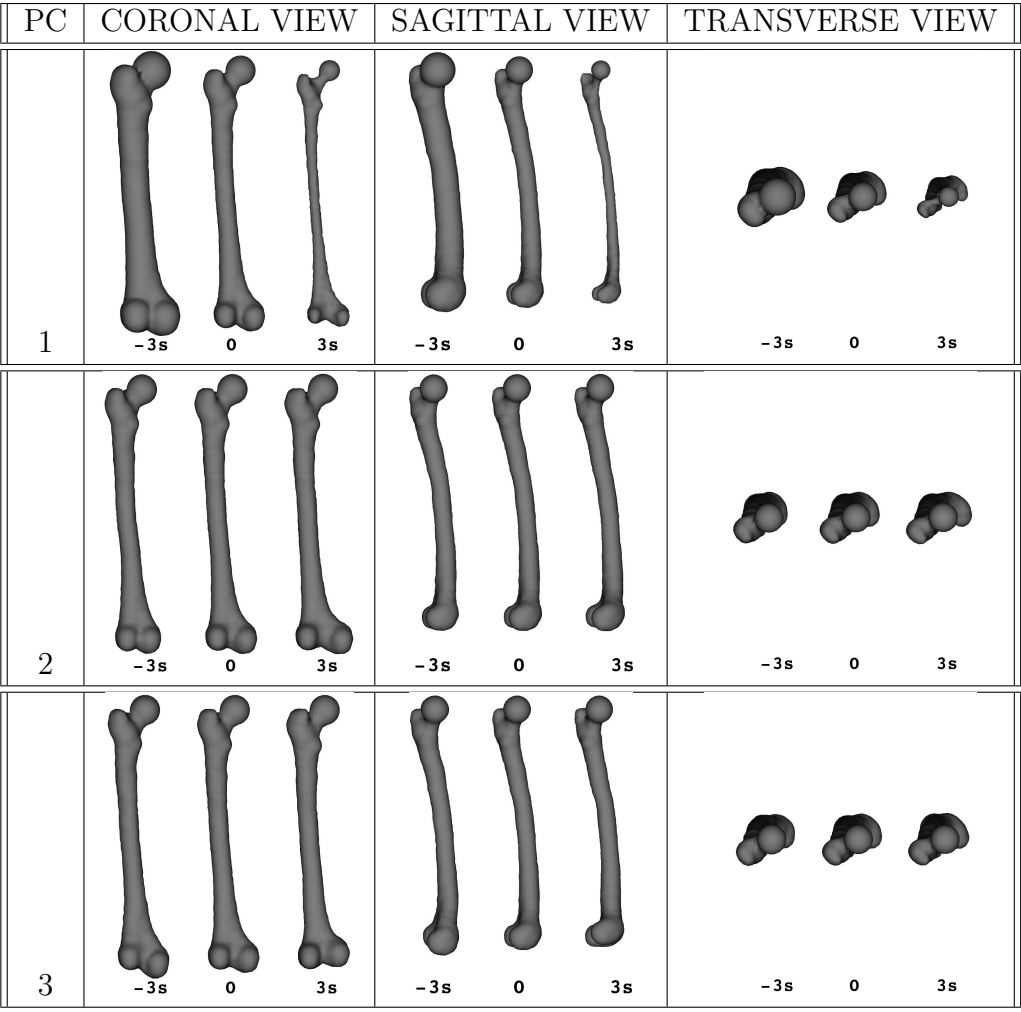


Figure 5.33: Effects of the rotated MAT model PCs on femoral shape (1 of 3)

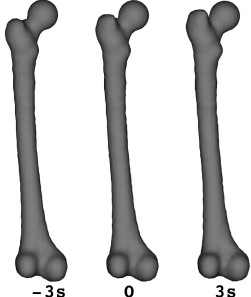
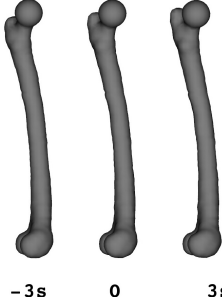
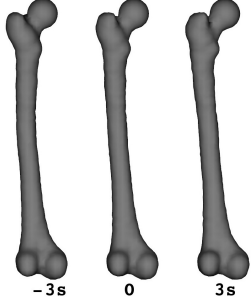
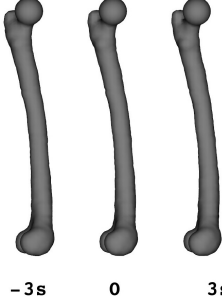
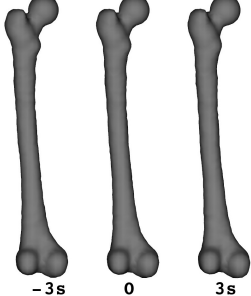
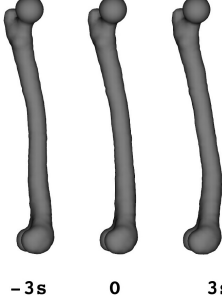
PC	CORONAL VIEW	SAGITTAL VIEW	TRANSVERSE VIEW
4	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
5	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
6	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s

Figure 5.34: Effects of the rotated MAT model PCs on femoral shape (2 of 3)

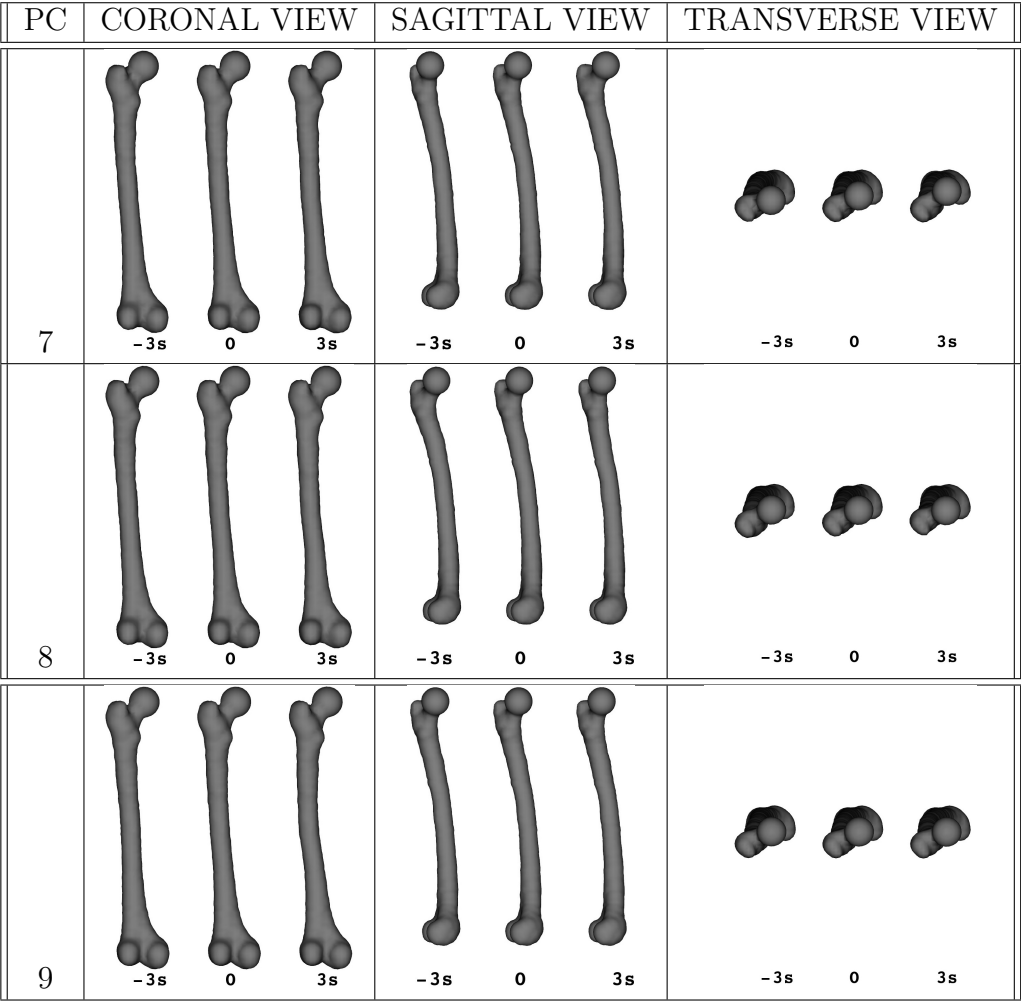


Figure 5.35: Effects of the rotated MAT model PCs on femoral shape (3 of 3)

Component	Effect of the Component
1	Increased thickness of the whole bone
2	Medial Condyle moves in the Medial/Lateral direction and varies in size
3	Medial Condyle moves in the Inferior/Superior direction with no change in size
4	The greater trochanter changes in size and moves in the Inferior/Superior direction
5	Proximal portion of the shaft moves in the coronal plane
6	Distal portion of shaft moves in sagittal plane
7	Version Angle changes due to change in femoral neck and head
8	Proximal Shaft shifts in sagittal plane
9	Middle and proximal section of shaft move in coronal plane

Table 5.3: Effects of the rotated MAT components on the femoral shape

variables, thus possibly eliminating this behaviour.

Each of these components can be easily related to the muscular groups that were defined in Chapter 2 and if the muscles attaching to a particularly structure have different orientations there are separate components for each of these directions (Tab. 5.4), as for instance with the muscles inserting or affecting the medial condyle or the proximal shaft.

The possibility of relating each component to the action of groups of muscles is particularly relevant for describing and identifying pathological deformations of the femur, since many of these are due to unbalanced muscular forces. This can be due to direct muscular damage, nerve dis-function or damage of the motor cortex. Regardless of the origin of the deformation, all these conditions have an effect on the bone that is always mediated by the muscles. Therefore a model that is able to link the different muscular groups to femoral shape might have some potential for diagnostic purposes.

Since no real-case scenarios were available for testing, it was not possible

STRUCTURE	MUSCLE	COMPONENT
GREATER TROCHANTER	M. gluteus medius	PC4
	M. gluteus minimus	
	M. piriformis	
	M. vastus lateralis	
	M. obturatorius internus	
	Ms. gemelli	
	M. quadratus femoralis	
LESSER TROCHANTER	M. iliacus	PC5
	M. psoas major	
	M. psoas minor	
SHAFT (proximal third)	M. gluteus maximus	PC5 and PC8
	M. pectineus	
	M. adductor minimus	
	M. adductor brevis	
SHAFT (remaining thirds)	M. adductor magnus	PC6 and PC9
	M.adductor longus	
	M. vastus medialis	
	M. vastus lateralis	
	M. vastus intermedius	
MEDIAL CONDYLE	M. biceps femoris	PC2
	M. adductor magnus	
	M. triceps surae	
	M. plantaris	
LATERAL CONDYLE	M. gracilis	PC3
	M. triceps surae	
	M. tensor fasciae latae	
		-

Table 5.4: Regions of the femur and principal muscles that are attached or in contact with them

to evaluate how this model relates to pathology; however, the extreme localisation of the effect renders this model very promising for the description of conditions that by their very nature affect well-defined regions or structures of the femur.

It should be noticed that in Tab. 5.4 the first PC is not listed. Since its effect consists in a change in thickness of the whole bone (see Tab. 5.3), this component might actually be related to body weight, a factor that is likely to influence femoral shape, given its load-bearing function.

5.4.2 Rotated MA Model

The orthomax rotation can also be applied to the MA and the radius information only. The first component of the rotate MAT model accounts for a change in thickness of the femur, therefore, by neglecting this information in the representation, it would be able to describe other aspects of the shape that are not included in the rotated MAT model.

The effects of each rotated MA component are depicted in Figs. 5.36 and 5.37.

The main effect of each component on the shape is described briefly in Tab. 5.5.

By excluding the information about the radius, some component and related changes in shape are now taken into account explicitly by the model. The most obvious one consists of Component 3 that describes a change in shape that can be linked to the action of the horizontally- oriented hip muscles. These muscles have a relative small cross-sectional area (Glitsch & Baumann, 1997) and it is well-known that this is proportional to the force they can exert. Thus, the effect on the shape of these muscles was not taken into account in the rotated MAT model as it was negligible relative to other effects. Nevertheless, by discarding the information about the radius of the MAT points, a component of the model is now able to represent the horizontally-oriented hip muscles.

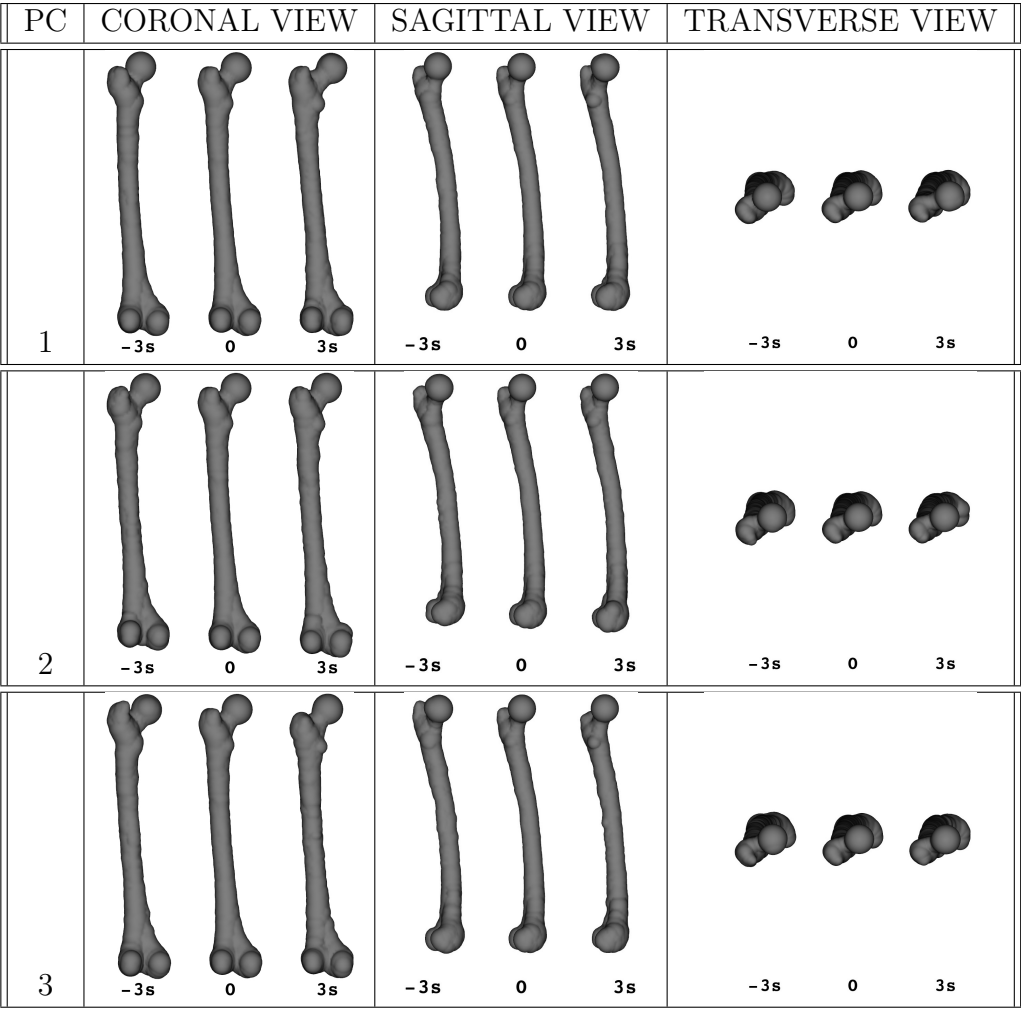


Figure 5.36: Effects of the rotated MA model PCs on femoral shape (1 of 3)

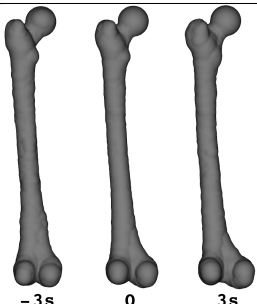
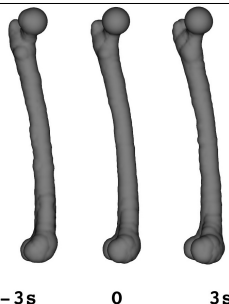
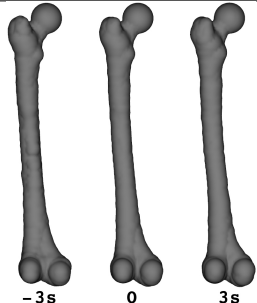
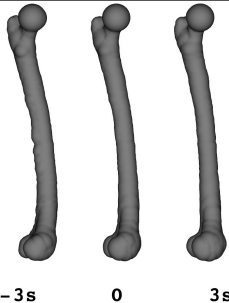
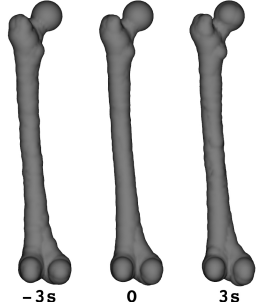
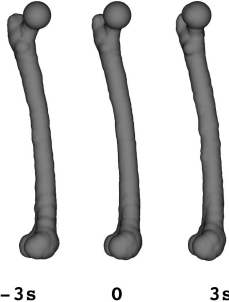
PC	CORONAL VIEW	SAGITTAL VIEW	TRANSVERSE VIEW
4	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
5	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
6	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s

Figure 5.37: Effects of the rotated MA model PCs on femoral shape (2 of 3)

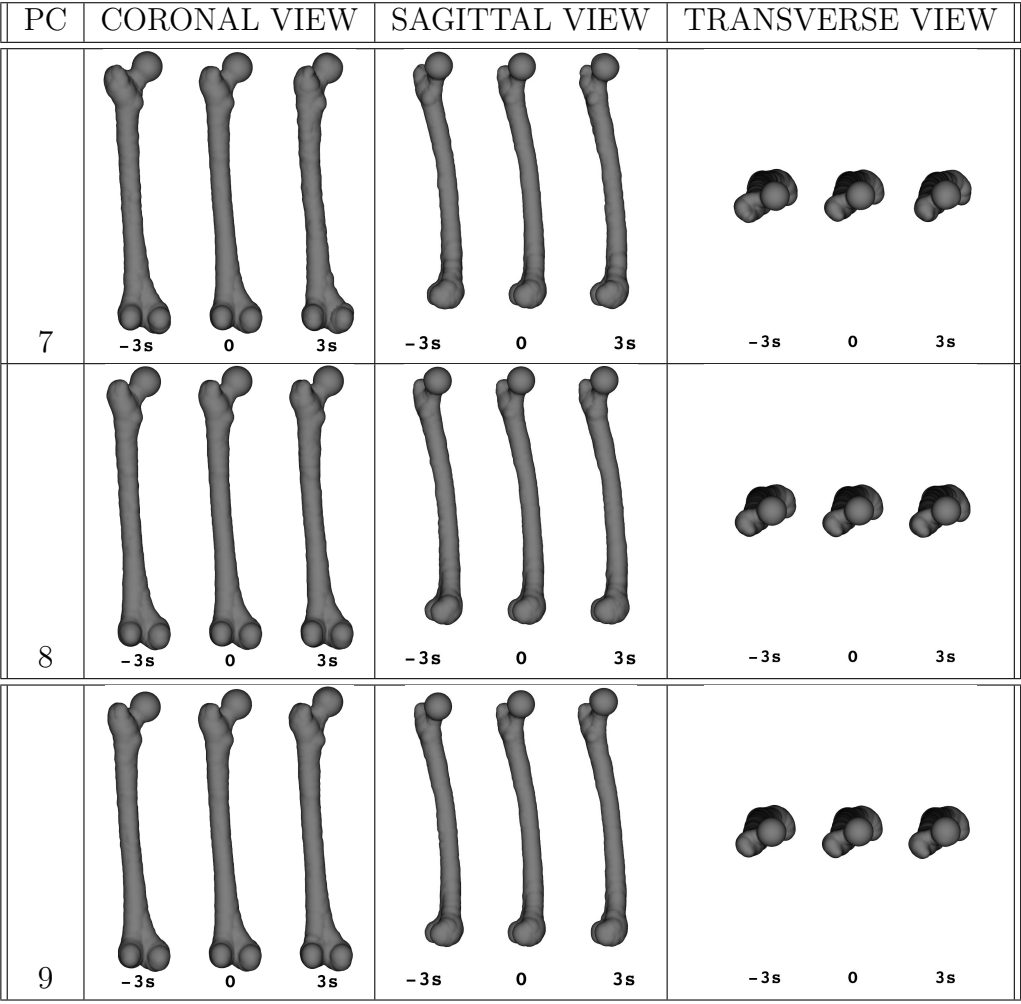


Figure 5.38: Effects of the rotated MA model PCs on femoral shape (3 of 3)

Component	Effect of the Component
1	Change in NSA due to movement of proximal shaft
2	Condyles move in the coronal plane, shortening the bone
3	The greater trochanter changes in size and moves in the Inferior/Superior direction
4	Medial condyle moves in the coronal plane in the Medial/Lateral direction
5	Distal part of the shaft moves in the sagittal plane, effectively changing the curvature of the femur
6	Change in version and length of the femoral neck
7	Movement of the greater trochanter in the coronal plane in the Medial/Lateral direction
8	Both condyles move medially/laterally
9	Change in NSA due to changes in shape of femoral neck and head

Table 5.5: Effects of the rotated MA components on the femoral shape

5.4.3 Rotated Radius Model

The orthomax rotation can be also applied to the model computed on the radius information only.

The effects on the shape are shown in Figs. 5.39 and 5.40 and their brief description is given in Tab. 5.6.

As expected, with the exception of the 3rd component, this model describes changes in shape which are specific to certain regions and structures.

It is worth noticing that this model actually identifies structure that were not explicitly described by any of the previously analysed models. These structures are the femoral neck and the femoral head, this should come as no surprise, as the main difference between these parts of the femur is in fact their radius.

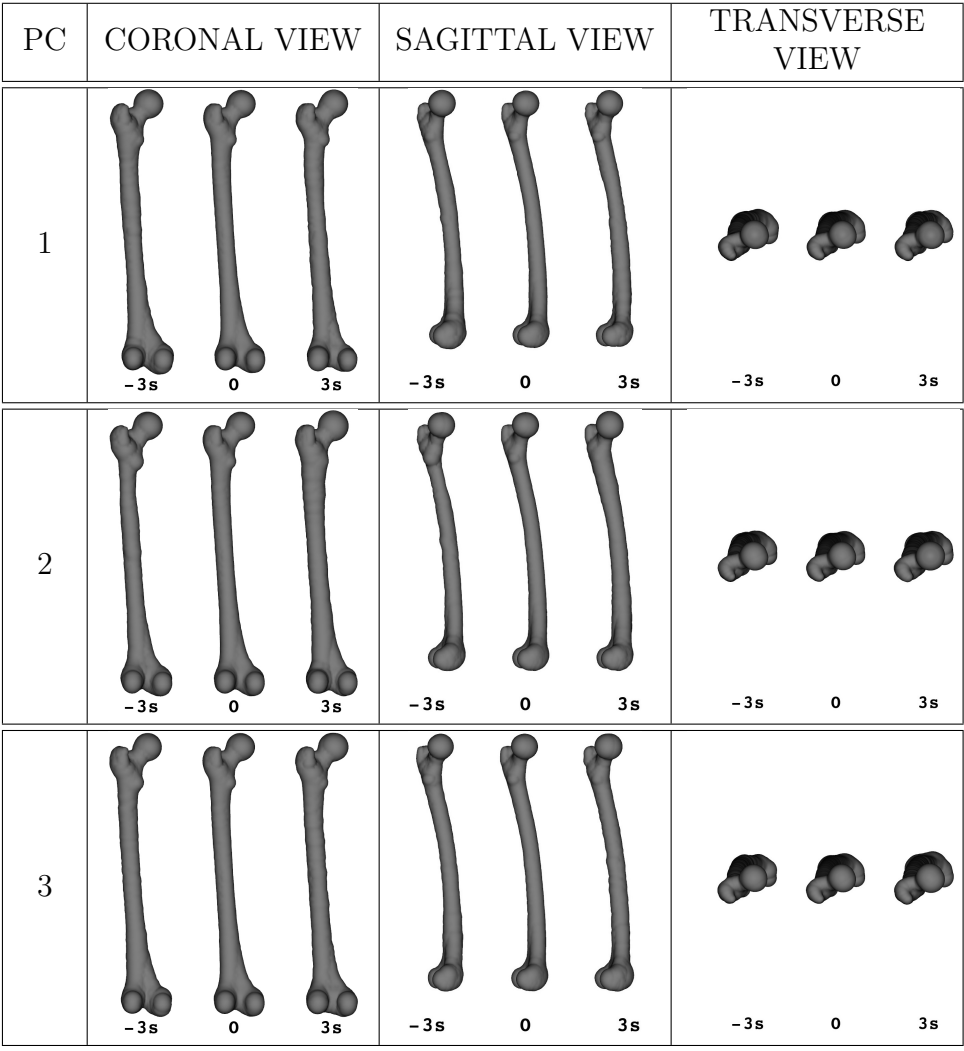


Figure 5.39: Effects of the rotated radius model PCs on femoral shape (1 of 3)

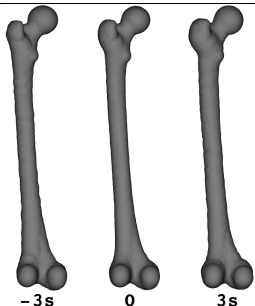
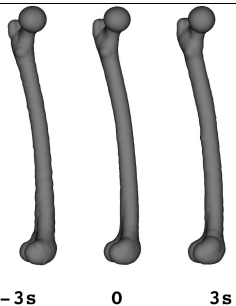
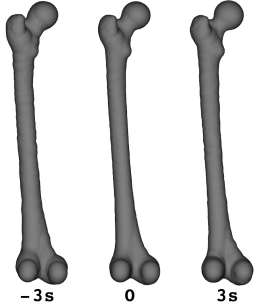
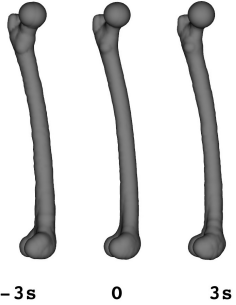
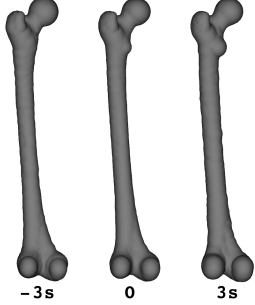
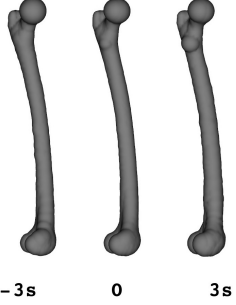
PC	CORONAL VIEW	SAGITTAL VIEW	TRANSVERSE VIEW
4	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
5	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s
6	 -3s 0 3s	 -3s 0 3s	 -3s 0 3s

Figure 5.40: Effects of the rotated radius model PCs on femoral shape (2 of 3)

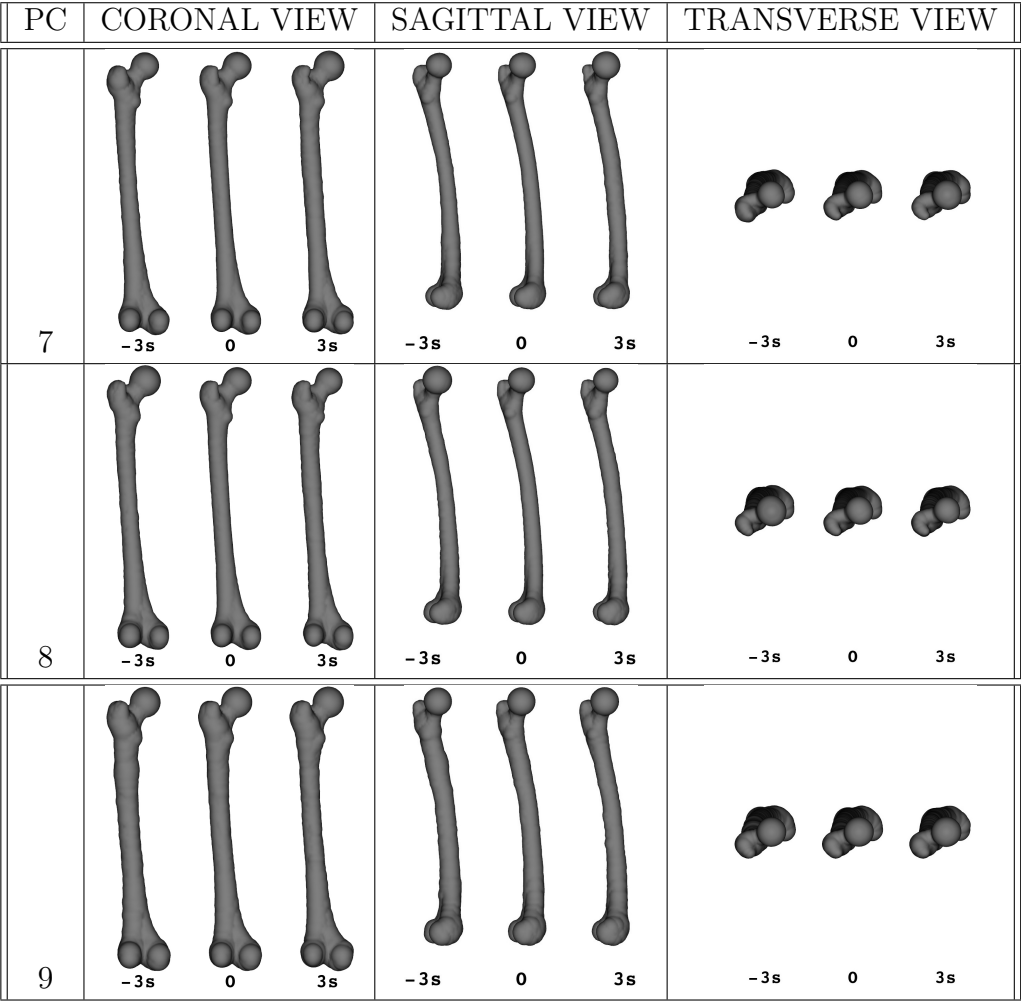


Figure 5.41: Effects of the rotated radius model PCs on femoral shape (3 of 3)

Component	Effect of the Component
1	Change in medial condyle size
2	Thinning of proximal shaft
3	Changes in condylar fossa shape
4	Greater trochanter increases in size while femoral head becomes slightly smaller
5	Changes in thickness of the femoral neck
6	Size of the lesser trochanter
7	Radius of femoral head and corresponding change in medial condyle
8	Changes in radius of femoral head

Table 5.6: Effects of the rotated radius-components on the femoral shape

5.5 Discussion

The introduction of the MAT has the main merit of increasing the clinical relevance of the model. By direct comparison of the model obtained from the MAT and the one obtained from the point cloud representation, there are some differences between the two mainly due to the fact that the point cloud model has to be drawn from a very high dimensional space and therefore is intrinsically subject to more noise.

The simplification step introduced to deal with redundancy of the information within the MA representation can be used to obtain different models with different levels of complexity; by deciding to include more and more spheres it is possible to retain a greater level of information, thus leading to more accurate models. However, this decreases the strength of the statistical significance of the model.

The introduction of the MAT allowed for the decoupling of 2 kinds of information that could be then examined separately, thus leading to the MA and radius models. Using this model it is possible to discard the information about the thickness of the bone and just focus on the relative position of the

different structures of the bones. Whenever the information about thickness cannot be ignored, for instance when considering the BA, the model should include the radius information as well.

By introducing the rotation of the principal component also in the MAT and MA and radius model, pathologies can be analysed from a different point of view. Due to the decoupled information, it would be possible to differentiate pathologies from those that affect the radius of the spheres of the MA or those that actually affect the positioning of their centres.

Both the rotated MAT model and the rotated MA model, efficiently capture localised changes in shape. The rotated MA model in particular can be easily related to the different muscular groups that are attached to the femoral bone. By neglecting the radius information the effect of smaller muscular groups becomes evident, thus allowing the effect on shape linked to pathologies of these muscular groups to be described.

Although the mechanical behaviour of the femur is too complex to be reduced to the simple interaction between muscular groups and their effect on the inserting/originating structure, the results shown in Table 5.4 could provide a starting point for a more rigorous mechanical analysis.

By rotating the radius model, regions of the femur such as the neck and the femoral head can be identified automatically and changes in neck and head thickness are taken into account by separate components.

Overall, the introduction of the rotation of the components together with the MAT transformation allows a natural description of femoral anatomy in which no *a priori* information about the different structures comprising the bone is employed.

The localised effects of each component would make these models useful for describing different kinds of shape abnormalities.

It would be interesting to verify if the different kind of changes in shape described by the separate MA and radius model could be explained in different ways, possibly differentiating between biological and mechanical factors.

However this goes beyond the purpose of this work and might be addressed by further studies.

Chapter 6

Reconstruction of Femoral Morphology

In Chapter 2 a brief overview of femoral deformities was introduced; when the deformities are limited to rotational abnormalities or differences in thickness, the whole femoral shape can be used to evaluate the parameters of one of the models considered in the previous chapters. This would be just a projection onto the reduced-shape space and can allow the quantification of the amount of deformation (especially when a rotated model is considered). However, when the shape considered cannot be directly compared to the normal shape, such as in fractures or tumours, this approach can no longer be used.

In this chapter those parts of simulated fractured femurs, considered to be normally unaffected by the fracture were used to evaluate the parameters of the PCs of a SSM. By using these parameters it was possible to reconstruct the whole shape, thus enabling the comparison of the fractured bone to what is its predicted original shape, according to a statistical model.

Reconstruction of the complete three-dimensional shape from partial information can be also useful in forensic, for instance when trying to estimate the height of the individual based on fragment of a burnt bone (Grevin *et al.*, 1998) or in order to reduce radiation dosage, by reconstructing the anatomy

starting from bi-planar images (Mittona *et al.*, 2006).

The approach followed here can be applied, with no modifications, to the reconstruction of normal humeral anatomy; the accurate reconstruction of proximal humeral anatomy is particularly relevant whenever prosthetic replacements need to be employed (Robertson *et al.*, 2000).

Since the reconstruction produces the whole three-dimensional shape, it is possible to define a complete coordinate system, thus allowing a better classification of proximal humeral fractures.

Applications of this technique in paleontology would enable to reconstruct an intact bone from a fragment, thus simplifying the process of skeletal reconstruction from fossil remains (Carpenter, 1984). Further improvement can be obtained in association with a predictor of adjoining bones, as discussed by Yang (2008).

The evaluation of the predicting tool is performed in terms of the measures introduced in Chapter 2, as they are more relevant in a fracture-treatment scenario. It is worth noticing that among these measures, the angular ones would be the most important, since they are those that can be more directly influenced by surgical intervention.

Since the fractures were only simulated, a direct comparison between the prediction and the real shape was possible in order to evaluate the performances of the predicting tool.

6.1 Introduction

Traumatic events and bone tumours can significantly change the appearance of a bone up to the extent that characteristic anatomical features can no longer be recognised.

Surgery may be needed in order to restore normal anatomy and function, however it can be difficult to assess the shape that should be attained. Examination of the contralateral limb can be a possible approach for long bones

such as the femur (Stephen *et al.*, 2002), but may not be always possible and also the skeleton is usually not symmetrical (Cuk *et al.*, 2001). A predictive tool to reconstruct the real shape based on partial information can therefore make a better reference for surgical intervention.

It is the hypothesis of this work that the models developed previously (Chapters 4 and 5) can be used to plan surgical reconstruction of femoral fractures.

6.2 Methods

Due to the lack of real fractured bones, the predictive tool was tested on simulated fractures, thus using partial information from whole shapes. A proximal fracture scenario was simulated by reconstructing the shape just by considering part of the proximal shaft of the femur. Since there were too few examples in the data set to allow for both a training set and a test set, a leave-one-out approach was employed; 23 different model were computed using 22 shapes in the data-set. For each of the models one shape was not used for the creation for the model; a manual selection of MA points was performed on the shaft region. The extent of the selection was not fixed but left arbitrary on purpose, so to recreate conditions as close as possible to the real-life scenario in which the amount of shape that can be used cannot be controlled.

In Chapters 4 and 5 different models were introduced, however, since the aim is to predict the overall shape from portions of it, not every model is suitable.

Orthogonal rotations of the PCs were introduced in order to describe local effects, typical of pathologies. However, this very characteristic makes the resulting model unsuitable for the predictive task. This is evidenced by considering the following: each model that was computed could represent a shape in a reduced space defined by the eigenvectors computed by PCA. In

this space each instance of shape will be represented by a vector of coordinates $\mathbf{b}$, called the b-values of the shape.

$$\mathbf{X} = \mathbf{b}\Phi + \epsilon \quad (6.1)$$

In Eq. 6.1 the vector ϵ represents an error vector that is neglected for the purposes of modelling. The vector $\mathbf{X}$ allows for the reconstruction of the shape, in the case of a cloud of points this vector is computed by considering the concatenation of the coordinates of the points (or spheres in the case of the MA) sampling the shape and then subtracting the mean shape.

Starting from a complete shape it is possible to evaluate the values of vector $\mathbf{b}$ by solving a linear system. However, if a partial shape is considered, the model representation still holds but there are some of the features representing the shape that are unknown. Since it is possible to swap equations in the matrix representation without changing the solution, we can rewrite the system as in Eq. 6.2.

$$\begin{bmatrix} X_U^1 \\ \vdots \\ X_U^l \\ X_K^1 \\ \vdots \\ X_K^m \end{bmatrix} = \begin{bmatrix} \Phi_U^{11} & \Phi_U^{12} & \dots & \Phi_U^{1k} \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_U^{l1} & \Phi_U^{l2} & \dots & \Phi_U^{lk} \\ \Phi_K^{11} & \Phi_K^{12} & \dots & \Phi_K^{1k} \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_K^{m1} & \Phi_K^{m2} & \dots & \Phi_K^{mk} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \end{bmatrix} \quad (6.2)$$

where the both the vector $\mathbf{x}$ and the matrix Φ in Eq. 6.1 have been rewritten so that those elements that are relative to unknown features, with subscript U, are separated from those that correspond to known features, with subscript K.

By considering the linear system relative to the known features, it is possible to find an estimate $\tilde{\mathbf{b}}$ using the pseudo-inverse of Φ_K .

$$\mathbf{X}_K = \Phi_K \mathbf{b} \Rightarrow \tilde{\mathbf{b}} = (\Phi_K^T \Phi_K)^{-1} \Phi_K^T \mathbf{X}_K \quad (6.3)$$

This value of $\tilde{\mathbf{b}}$ is guaranteed to minimise the least square error between the real $\mathbf{X}_K$ and the predicted value $\tilde{\mathbf{X}}_K = \Phi_K^T \tilde{\mathbf{b}}$. Since the values of the vector $\mathbf{X}_U$ are unknown it would be possible to compute an estimate of them simply by considering Eq. 6.4.

$$\tilde{\mathbf{X}}_U = \Phi_U (\Phi_K^T \Phi_K)^{-1} \Phi_K^T \mathbf{X}_K \quad (6.4)$$

By considering Eq. 6.4 it can be seen that the values of this estimation for the unknown features relies on the properties of the matrix Φ .

The non-normalised version of Φ can be found by multiplying it from the right to the matrix Σ , which is a diagonal matrix containing the standard deviations σ_i associated with each column of Φ .

$$\mathbf{Q} = \Phi \Sigma \quad (6.5)$$

This matrix can be still subdivided into two matrixes that are relative to the unknown and known parts of the vector to reconstruct $\mathbf{X}$, therefore another set of parameters can be found by solving the system in Eq. 6.6.

$$\mathbf{X}_K = \mathbf{Q}_K \mathbf{c} \Rightarrow \tilde{\mathbf{c}} = (\mathbf{Q}_K^T \mathbf{Q}_K)^{-1} \mathbf{Q}_K^T \mathbf{X}_K \quad (6.6)$$

where $\mathbf{Q}_K = \Phi_K \Sigma$. Blanz & Vetter (2002) proved in their work that the vector $\tilde{\mathbf{c}}$ minimised the squared residual and also the maximum prior probability defined in Eq. 6.7, where $\mathbf{s}_i$ represents each of the eigenvectors forming the model and under the assumption of a Gaussian distribution in every direction.

$$p(\mathbf{x}) = \prod_{i=1}^m \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{1}{2\sigma_i^2} \langle \mathbf{s}_i, \mathbf{x} \rangle} \quad (6.7)$$

Their result considered a non-normalised version of the matrix Φ , as each

column of this matrix was multiplied by the square root of the corresponding eigenvalue, i.e. the associated standard deviation σ_i .

However, by using some basic algebraic properties of matrices it is possible to prove that the results will be equivalent if the normalised version of the matrix is used instead (Eq. 6.8).

$$\begin{aligned}
 \tilde{\mathbf{c}} &= ((\Phi_{\mathbf{K}}\Sigma)^T(\Phi_{\mathbf{K}}\Sigma))^{-1}\mathbf{Q}_{\mathbf{K}}^T\mathbf{X}_{\mathbf{K}} = \\
 &= (\Sigma^T\Phi_{\mathbf{K}}^T\Phi_{\mathbf{K}}\Sigma)^{-1}\Sigma^T\Phi_{\mathbf{K}}^T\mathbf{X}_{\mathbf{K}} = \\
 &= \Sigma^{-1}(\Phi_{\mathbf{K}}^T\Phi_{\mathbf{K}})^{-1}\underbrace{(\Sigma^T)^{-1}\Sigma^T}_{\mathbf{I}}\Phi_{\mathbf{K}}^T\mathbf{X}_{\mathbf{K}} = \\
 &= \Sigma^{-1}(\Phi_{\mathbf{K}}^T\Phi_{\mathbf{K}})^{-1}\Phi_{\mathbf{K}}^T\mathbf{X}_{\mathbf{K}}
 \end{aligned} \tag{6.8}$$

By substituting for $\tilde{\mathbf{b}}$ from Eq. 6.3:

$$\tilde{\mathbf{c}} = \Sigma^{-1}\tilde{\mathbf{b}} \tag{6.9}$$

So the reconstruction of $\mathbf{X}$ would have the same value regardless of the normalisation of Φ as can be seen in Eq. 6.10.

$$\mathbf{X} = \mathbf{Q}\tilde{\mathbf{c}} = \Phi\underbrace{\Sigma\Sigma^{-1}}_{\mathbf{I}}\tilde{\mathbf{b}} = \Phi\tilde{\mathbf{b}} \tag{6.10}$$

The orthomax rotations introduced to describe pathological deformities have an important effect on matrix Φ . Bernaard & Jennrich (2003) showed that orthomax rotations render the matrix Φ close to a *perfect simple structure* (PSS). A matrix is said to have a PSS if it has at most one nonzero element on each row. If this structure is not achievable, then the rotation will result in an approximate PSS for the matrix.

In the light of this, one might think that by employing a rotated model

for prediction purposes, since the matrix Φ is in a (nearly) PSS, then the probability that the sub-matrix $\Phi_K^T \Phi_K$ is singular would increase. However, by considering Eq. 6.11 it is clear that the invertibility of the matrix $\Phi_K^T \Phi_K$ is unaffected by rotations (or any geometric similarity), since the determinant of the transformation matrix is equal to one.

$$|(\Phi_K \mathbf{R})^T (\Phi_K \mathbf{R})| = \quad (6.11)$$

$$= |\mathbf{R}^T \Phi_K^T \Phi_K \mathbf{R}| = \quad (6.12)$$

$$|\mathbf{R}| |\Phi_K^T \Phi_K| |\mathbf{R}| = 1 \cdot |\Phi_K^T \Phi_K| \cdot 1 = |\Phi_K^T \Phi_K| \quad (6.13)$$

where $|\mathbf{A}|$ denotes the determinant of matrix $\mathbf{A}$.

The prediction of the femoral shape would then be the same regardless of the rotation. Therefore, the standard PCA model was used, since the rotated model is derived from it.

The last choice to be made is relative to which representation to use; in the previous chapters two different representations were employed: the MAT and the cloud of points. However, considering the kind of task to be performed, the MAT model has some advantages; since the aim is to reconstruct the overall shape starting from the geometry of the proximal shaft, registration of this partial shape to the whole shape needs to be performed. The femoral shaft is an elongated structure with cylindrical symmetry, so the matching between points lying on the outer surface might be difficult to be performed. However the MAT of a nearly-cylindrical shape is roughly linear thus the matching between features would be easier.

An interactive GUI, based on Python and VTK, was developed to allow the manual selection of MA points on the proximal shaft in order to reconstruct the whole shape. The measuring tool described in Chapter 2 was then employed both on the real shape and the reconstructed one to evaluate the goodness of the prediction.

In order to evaluate the properties of the predictive tool, the box-plot of the predictions across all the shapes in the data-set can be considered.

A box-plot is a non-parametric representation that summarises the properties of a probability distribution (Hoaglin *et al.*, 1983). Even though box-plots are non-parametric, the properties of this representation can be better evaluated by considering how it represents a well-known distribution such as the normal distribution.

A box-plot is composed of a box, a median line and the “whiskers”. The parameters needed to define a box-plot are the following:

- Median, which is the numeric value separating the higher half of a population, or a probability distribution, from the lower half.
- First Quartile (Q1), which is the numeric value that cuts off lowest 25% of data
- Third Quartile (Q3), which is the numeric value that cuts off highest 25% of data

The distance between Q3 and Q1 is called the Inter Quartile Range (IQR), and it is used to define the limit of the whiskers. The limits of the left and right whiskers are placed at $-1.5 \times \text{IQR} + Q1$ and $+1.5 \times \text{IQR} + Q3$, respectively.

All the data points that do not fall within the whiskers are considered to be outliers, and usually are represented as red crosses on the box-plot. The box-plot representation of a normal distribution is given as an example in Fig. 6.1.

The approach described here can be considered as an estimate of many parameters (the coordinates of the MA and the corresponding radii). Traditionally, the mean squared error computed from the real parameters and their estimate values is considered to test the performances of the estimator.

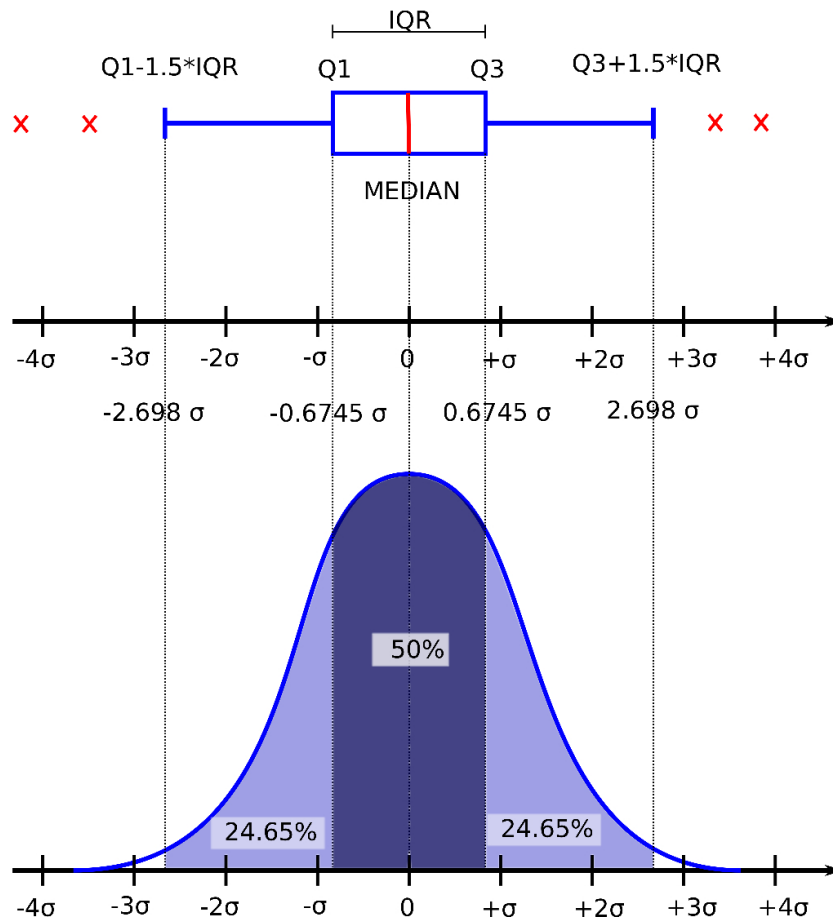


Figure 6.1: Box-plot relative to a normal distribution: the Median, Inter Quartile Range are defined in terms of standard deviations (σ) and the outliers are represented as red crosses

However, such an approach would not be particularly appropriate in this scenario, as in surgical procedures the important parameters are those angular measures that were introduced in Chapter 2. Hence, comparing the predicted shape and the real shape in terms of these angles seems to be the most relevant approach. Moreover, since the angular measures are the result of the interaction of many MAT points and refer to well defined properties of femoral shape, it is possible to investigate whether certain properties of shape are easier to predict compared to other ones.

Even though the parameters that are considered are no longer those employed for the representation of shape, the angular measures are nothing but a function of the original parameters. Therefore, the properties of the estimator can still be evaluated by considering the box-plots of the errors between the values of these measures on the real and reconstructed shape.

The two main properties of a predictor are the variance of the errors and the bias. A predictor is said to be biased if it consistently over or underestimates a parameter.

However, a biased predictor is not necessarily worse than an unbiased one. The variance of the predictions is in fact a more important property of an estimator. It is easy to understand that a predictor that always overestimates the real parameter exactly by a certain value b , should be preferred to a non-biased predictor, which experiences a larger amount of variation. These two possible behaviours are shown in Fig. 6.2 alongside the relative box-plot representations.

The properties of the predictor can be easily visualised considering the box-plot of the prediction errors. The median of the error represents the bias of the predictor, so that a median positioned exactly at zero would indicate an unbiased predictor, and the width of the box gives an indication of the variation within the predictions. Ideally, the best predictor would be unbiased (median at zero) and with zero variance (degenerate box, with width equal to zero). For obvious reasons, this ideal condition will never

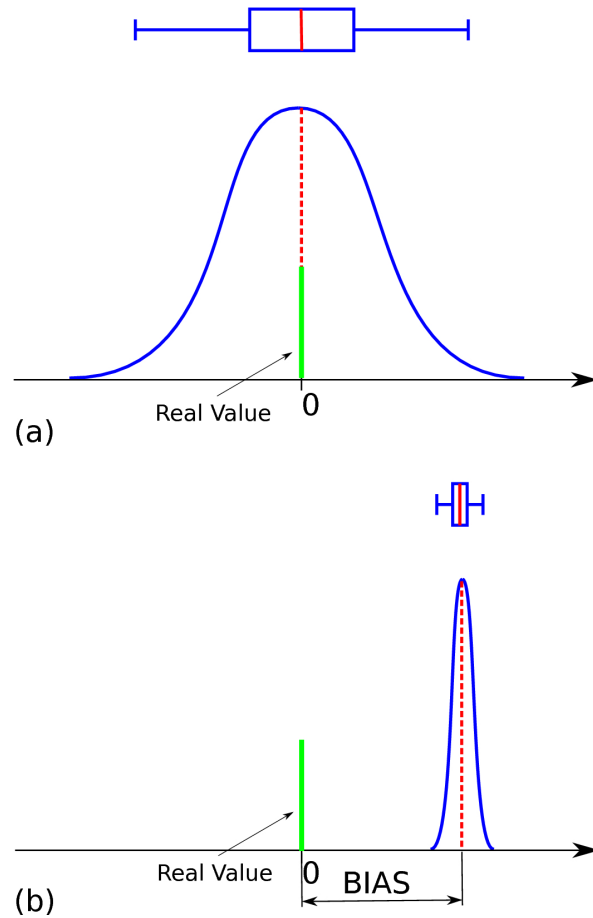


Figure 6.2: (a) Unbiased predictor, the distribution of the errors has median centred on zero. However, this predictor has worse performances than a biased predictor with a significantly lower variance (b).

be verified, so the predictions can be considered to be better the closer the predictor approaches the ideal estimator.

6.3 Results

To assess the predictive abilities of the MAT model, 23 different models were computed using 22 out of the 23 femoral shape available. The shape that was left out was used as the test shape, the MAT of this shape was computed and the corresponding MA was rendered using VTK and Python (since the MAT has 4 dimensions, only its three-dimensional counterpart, i.e. the MA, can be visualised).

The choice of representing the shape through its MA and not the actual reconstruction is motivated by the fact that in a real case scenario the feature matching is performed on the MAs and not on the real surface.

The user was then be able to perform a manual selection of a rectangular parallelepiped which contained medial axis points in the proximal shaft area. The computed medial axis was automatically oriented so that its first principal direction will point upwards, thus enabling an easier selection of points on the shaft (Fig. 6.3).

Once the MA points were selected, the corresponding MAT points formed the vector $\mathbf{X_K}$ previously defined and the estimated coordinates in the PC space could be computed.

The overall reconstructed shape $\tilde{\mathbf{X}}$ was then computed and rendered alongside the original shape, as shown in Fig. 6.4.

An example of one of the 24 reconstructions performed is given in Fig. 6.5, where the original shape, the reconstruction and the mean-shape are shown in the different anatomical orientations.

Although, this visual comparison could help to spot completely wrong predictions, a more quantitative comparison was performed. This analysis relied on the geometrical measures defined in Chapter 2.

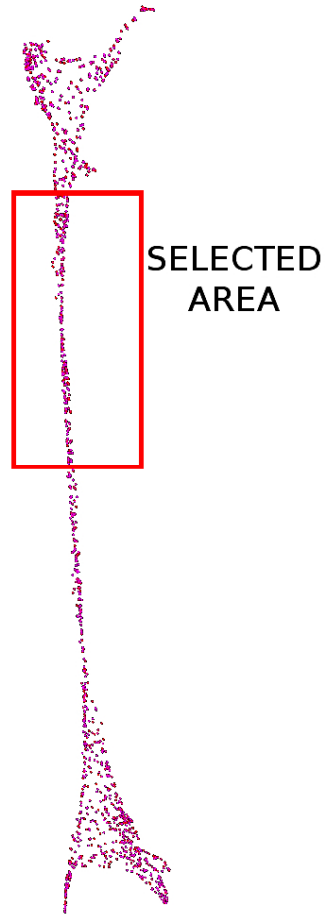


Figure 6.3: The MA of the test shape is rendered and points corresponding to the proximal shaft are manually selected, the selection would include all the points that are included in the parallelepiped defined by the selection rectangle and the perpendicular to the viewing plane



Figure 6.4: Rendering of the reconstructed shape (green) alongside the original one (red)

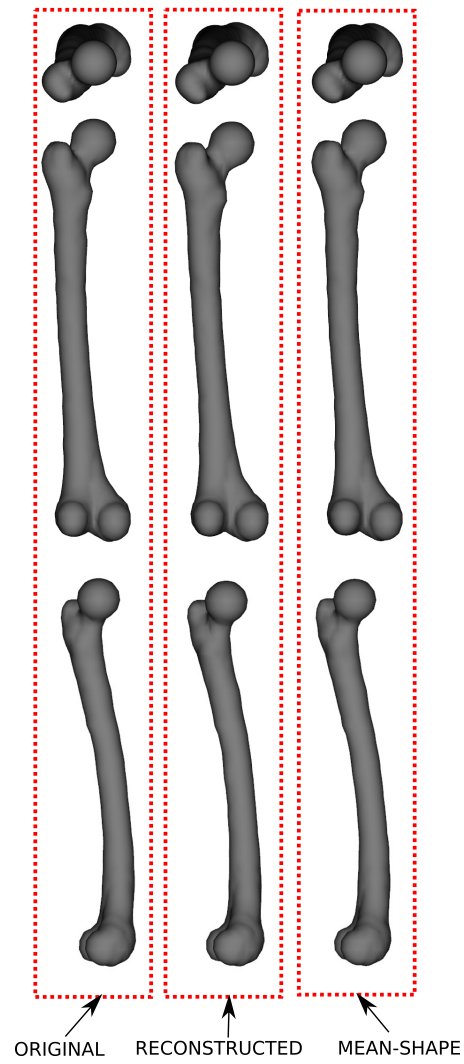


Figure 6.5: Comparison between the original shape, the prediction and the mean-shape. From top to bottom: transverse, coronal and sagittal views

The measures evaluated on the real shape were then compared to those extracted from the prediction and the mean shape. The box-plots of the differences are shown in Figs. 6.6 to 6.10.

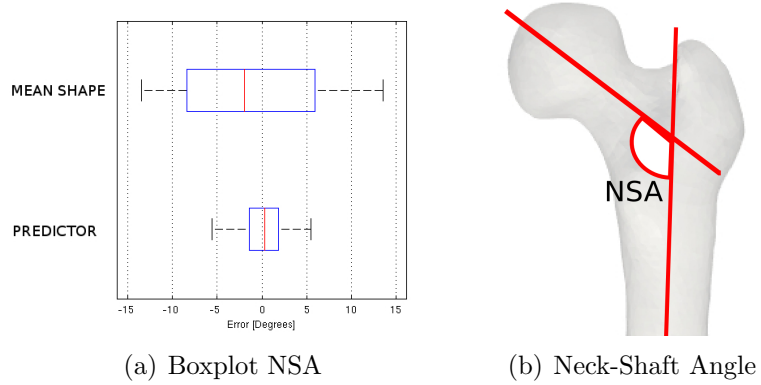


Figure 6.6: Box-plots of the signed-errors of the predictions relative to the NSA. The mean-shape and the prediction based on the SSM are compared.

It is worth noticing that, box-plots being non-parametric, they make no assumption on the statistical distribution of the data.

Since the error between the real shape and the prediction is considered, the box-plot of this value would ideally show the characteristics of the ideal predictor described earlier. Again the smaller the IQR of the errors relative to a predictor are, the better its performances would be.

By considering the different box-plots Figs. 6.6-6.10 it can be said that the predicting tool developed is consistently better than considering the mean shape.

The higher amount of variation for the angular measures, especially the NSA, can be explained by considering that the range of variation of these parameters is naturally higher, thus the errors are also expected to be higher.

From the values in Tab. 6.1 the model seems to give better predictions for all of the variables, although for length parameters the difference between the two predictors considered is not as high as with angular ones. Bearing

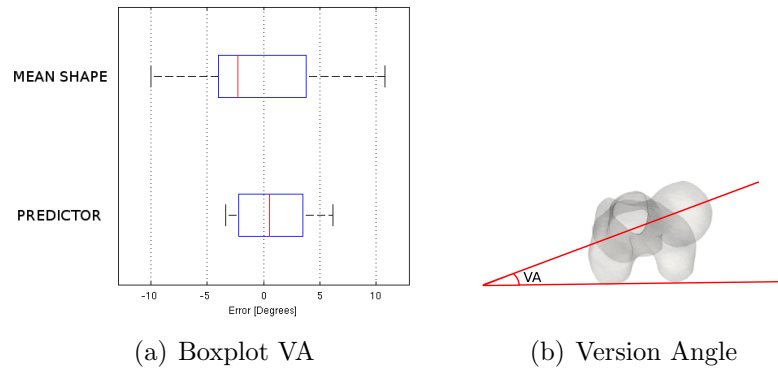


Figure 6.7: Box-plots of the signed-errors of the predictions relative to the VA. The mean-shape and the prediction based on the SSM are compared.

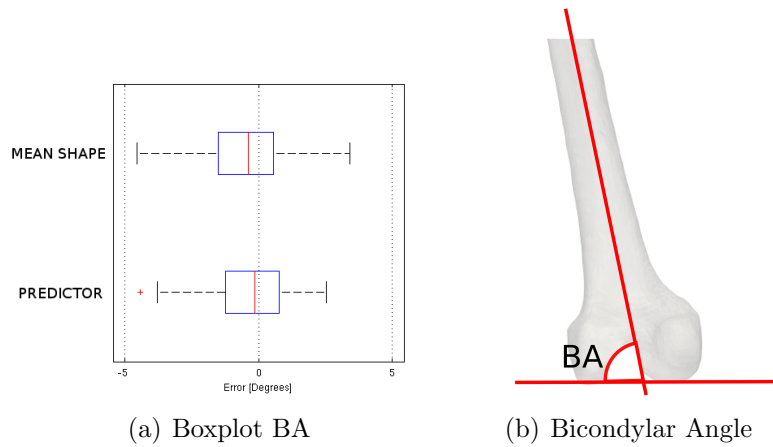


Figure 6.8: Box-plots of the signed-errors of the predictions relative to the BA. The mean-shape and the prediction based on the SSM are compared.

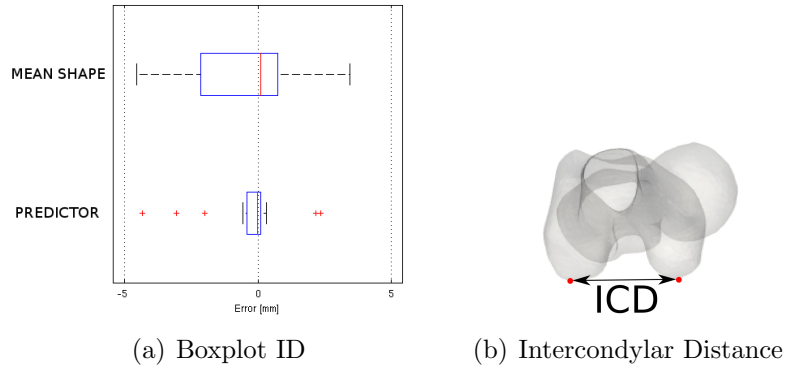


Figure 6.9: Box-plots of the signed-errors of the predictions relative to the ICD. The mean-shape and the prediction based on the SSM are compared.

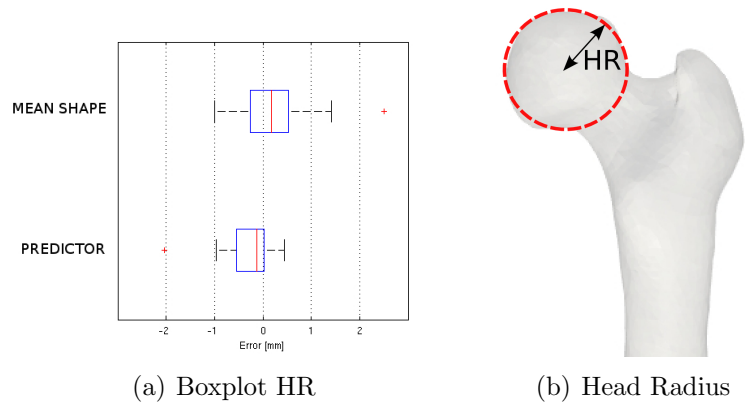


Figure 6.10: Box-plots of the signed-errors of the predictions relative to the HR. The mean-shape and the prediction based on the SSM are compared.

Parameter	Prediction Mean-shape			Prediction Model		
	Mean	Max	Min	Mean	Max	Min
NSA [Deg]	-0.952	13.5242	-13.4468	0.3097	5.5280	-5.5649
VA [Deg]	-0.4036	10.7716	-10.0138	0.7656	6.1905	-3.3925
BA [Deg]	-0.4271	3.4378	-4.5474	-0.3532	2.5601	-4.4342
ICD [mm]	-0.6858	3.4568	-4.5249	-0.2535	2.3369	-4.3203
HR [mm]	0.2011	2.4990	-0.9908	-0.2546	0.4411	-2.0371

Table 6.1: Predictions errors of the mean shape and the model-based prediction

in mind the proposed application however, this is hardly a problem, as the angular parameters are those that surgery aims to restore being the only ones that can be directly influenced by the procedure.

The performance of the SSM-based predictor are also accurate enough to justify its employment in pre-surgical assessment. Villanueva-Martínez *et al.* (2009) discussed the importance of VA and NSA in surgical total hip resurfacing procedures. They describe a surgical technique employing guide-wires that can achieve an average deviation from the ideal NSA of 3.3° (max 7°) and an average deviation for VA of 3° (max 8°). These values are a clear improvement compared to other techniques, where the average deviation is 6.5° (max 12°) and 6° (max 15°) for NSA and VA respectively.

These results give an idea of what should be considered an acceptable deviation in the values of these angular measures.

By comparing these findings to the results of the predictions in Tab. 6.1, it can be said that the predicting tool employed give performances that are in line with those of the guide-wires technique in terms of maximum deviation observed (Villanueva-Martínez *et al.*, 2009).

Assuming that the proposed technique is employed to determine the desired angle to be achieved during a surgical procedure, by applying a technique similar to the one described by Villanueva, the deviation from the real value would be small enough and hopefully prevent future complications.

6.4 Discussion

The issue of reconstructing the whole 3D geometry of healthy human femur has been addressed. Among all the models that were considered in the previous chapters, the MAT model was deemed to be the best candidate as it makes the feature matching easier and has the stronger statistical validity.

It was proved that using the rotated SSM, instead of the original one, will not change the reconstruction, contrary to what one might have initially thought. Thus, since the rotated model is nothing but a further manipulation of an initial PCA model, this was the model on which the prediction was based.

Results showed that the SSM has good prediction abilities that are better than simply considering the average femoral shape of the healthy population. Although another predictor based on the mean shape might have been considered (i.e. the one that simply replaces the unknown features with those of the mean shape), such a predictor would have not guaranteed the construction of a well-shaped bone and was not considered as a consequence.

Since the number of known features used for the reconstruction was not fixed, the results prove that the tool could be used in a real-life scenario thus enabling for better surgical planning based on standard images.

The predicted values for NSA and VA always fall in an acceptable range, at least from a surgical point of view (Villanueva-Martínez *et al.*, 2009), thus allowing for a possible application of the tool in surgical-planning.

The results presented by Blanz & Vetter (2002) are general enough to allow the reconstruction of a full three-dimensional shape starting from any known projection $\mathbf{L}$ of the shape.

This would allow for the reconstruction of the femoral shape from planar images. Nevertheless, since local 3D images in the proximity of fractures help the assessment of the condition of soft tissues involved in the fracture, this kind of approach was not considered although it may be useful for other applications.

It should be noted that the leave-one-out approach gives an opportunity to test the representativity of the data-set; since the reconstructions obtained from each of the 23 models tested resulted in accurate predictions, we can consider the shapes comprising the data-set to represent the general femoral anatomy.

Chapter 7

Discussion

7.1 Thesis conclusions

This thesis focuses on the modelling of femoral shape in the healthy human population. Since the pertinence of a model depends on the particular task aimed for, the choice of the modelling technique to be employed should be dictated by the final application.

In the case of tissue or organs, the main reasons for describing shape can either consist in a simple anatomical description, allowing for a better understanding of particular relationships between function and shape, or be addressed towards the much wider realm of pathological anatomy.

The classical SSM approach introduced in Chapter 4 mainly responds to a descriptive need; the complexity of the three-dimensional shape of the femur is captured by a reduced set of 9 parameters. These parameters offer a compact way of representing the shape and group together those aspects of shape that are statistically related.

For instance, by considering the first PC in this model, it is evident that proximal aspects of femoral anatomy are correlated to distal ones. This component is related to femoral alignment, which is a key factor in musculoskeletal research, as well as being useful in the diagnosis of arthritic conditions

affecting the knee joint (Cooke *et al.*, 2007). However, the actual relationship between the different aspects of femoral anatomy can be seriously disrupted in case of pathologies leading to bony deformities.

Building SSMs including also pathological bones would not be possible, as the data-set would not form a properly defined population and the amount of variability to account for would be too large. Considering just particular groups of pathologies could in theory be a viable option, but it would require a considerable amount of time and might still suffer from the large amount of variability observed.

By employing orthomax rotations, it was possible to redistribute the variability observed in the healthy population in a way that allows the description of pathology in a better way. Leung & Bosch (2007) used the orthomax rotation to describe local wall-motion abnormalities in the left ventricle, which are related to pathological conditions. By applying an orthomax rotation to the femoral SSM, it is possible to achieve similar results for the thigh bone.

Prior to the rotation, the first component of the model described changes in proximal and distal anatomy which synergetically resulted in an aligned bone; the orthomax rotation decouples different aspects of the shape, allowing the description of a non aligned bone using fewer, and therefore easier to understand, parameters.

Moreover, qualitatively different variations are taken into account by different parameters thus allowing each of these rotated components to be related to different muscular groups. However, the femur is subjected to complex mechanical conditions and the actual correspondence to muscular groups should be further investigated.

By means of an orthomax-rotated model, it is possible not only to identify whether an important aspect of shape (usually captured by a clinically relevant parameter) is abnormal, but also to pinpoint those regions or structures responsible for the abnormal parameter. This is a clear improvement, since some of these clinical parameters are affected by aspects of shape that are

not spatially close, and the identification of the exact location that originates the abnormality enables a better treatment.

Furthermore, this model does not subdivide the femur into its substructures (greater and lesser trochanter, condyles etc.) beforehand, but instead their definition originates from the modelling process itself. In this view, the rotated model leads to an analytical description of femoral anatomy, identifying its relevant structures and describing them with separate parameters. In theory, a similar description could be achieved by considering different SSMs describing each femoral feature separately; however such an approach relies on an accurate and unambiguous definition of the substructures, which cannot always be guaranteed.

Another effect of the rotation is to produce a model which is more understandable, compared to the standard PCA approach. The effect of each component translates into a defined change in one of the parameters (such as NSA and VA) that are routinely employed in orthopaedics. For those parameters that, due to their definition, depend on spatially distant features of the femur (like VA, which depends on the relative position of the femoral neck axis and the condyles), the rotated model offers different components accounting for the different ways a change in the measure can occur.

The representation introduced by the SSM can be difficult to interpret by surgeons and physicians, therefore a semi-automatic tool, developed in Python (van Rossum, 1995), that computes a set of commonly used geometric parameters describing femoral shape was developed and used to “translate” the effects of each model using terminology commonly employed in orthopaedic practice. This tool relies only on one of the many possible definitions of each parameter; however, since the main aim was to label each component, using only one definition did not introduce any limitation.

One of the problems often encountered when computing a SSM consists of the small cardinality of the data-set available, especially in the medical field. This can result in a low statistical strength of the model, since each

shape is artificially mapped in a high dimensional shape. The introduction of a different representation for the shape (Chapter 5), based on the MAT, reduces the dimensionality of shape space and increases the statistical significance of the model. Moreover, this representation decouples the information about thickness (represented by the radius of each MAT point) from the rest of the shape. In this way it is possible to neglect thickness variation and focus on other aspects of shape. This is somehow similar to the scaling process introduced in Chapter 3; by neglecting those changes in shape that are most evident, the amount of variability observed decreases and more subtle variations can be detected.

By employing an orthomax rotation on the various models based on the MAT representation, it was possible to obtain a localised description of morphological shape that was more evident compared to the one given by the rotated SSM based on the point cloud description. Not only changes in shape affecting the different structures of the femur, such as the greater trochanter, can be accounted for by different components, but it is also possible to relate these effects to particular muscle groups. This effect is particularly evident when considering the MA and radius model separately.

The rotated radius model also allows particular features of femoral anatomy to be detected that were not explicit in other models considered in this thesis. Since some of the components of this model describe the changes in diameter of both femoral neck and head, it can be said that the model identifies these structures as important femoral features. This was achieved only after separating the radius information from the one relative to position of the MAT points, thus allowing for the detection of aspects of shapes that would otherwise have been missed among changes with higher variability.

The MAT representation can also be used to predict the overall femoral shape, starting from partial data (Chapter 6). Employing just a fragment of the bone to reconstruct its shape can be of some utility in planning reconstructive surgery, after disruptive events such as bone fracture or tumour

surgical removal. Due to their nature, these events tend to preserve some parts of the shape which might contain some information about the overall anatomy.

The feasibility of this approach was tested by assuming a proximal fracture of a femur and using part of the proximal shaft to compute a reconstruction based on the SSM. The MAT representation was chosen as it enables feature matching in a less problematic way, especially when dealing with near-cylindrical structures, such as the femoral shaft.

The performance of this predicting tool was tested against the predictions offered by the mean femoral shape. Using the contralateral limb would have been more representative of a clinical situation, but since this was not possible due to the nature of the data-set, it was decided to use the only other predictor that would ensure a perfectly formed bone.

The results show that the accuracy of the prediction is good enough to be employed in surgical planning, as the range of variation is considered to be small enough compared to the acceptable values experienced in total hip resurfacing operations.

7.2 Further Considerations

This work has introduced a variety of novel techniques in the modelling of femoral anatomy.

By applying orthomax rotations to standard SSMs, the effects of the components of the model become more understandable since they are limited to specific structures of the femur. Despite this approach has previously been used for describing pathological related abnormalities, this work represents the first attempt in using rotations on three-dimensional shapes.

Even though the rotation does not change the general properties of the model, since the probability distribution of the shape (assumed to be a multi-variate Gaussian) is not affected, it offers a representation of shape in terms

of variables which are more relevant to clinical practice.

These variables can be easily related to those standard geometrical measures used by clinicians to quantify morphology, some of which were described in Chapter 2; however they offer a more complete description of shape. A clear definition of pathology based on these parameter is not possible; the availability of different methods and techniques to evaluate these measures implies that the range of normality needs to be varied according to the particular method considered. On the contrary the rotated SSM allows the detection of anatomical abnormalities in statistical terms and, since the representation is derived from the shape itself, does not rely on an arbitrarily chosen values representing normality.

A simple anomaly in the value of VA might not be sufficient to define the optimal treatment, since the location and cause of this anomaly is actually the important issue (Iguchi *et al.*, 2010). The rotated model introduces variables clearly related to individual geometrical parameters; however these parameters are not represented by single variables, as there are different modalities by which these parameters can experience a change from their normal range of values.

The rotation that was applied to the initial PCA model preserves the orthogonality of the original components. By removing the constraint of orthogonality, thus considering oblique “rotations”, it might be possible to obtain a simpler structure for the model (Jennrich & Sampson, 1966). However, this approach did not lead to any significant difference from the orthogonal rotations and the behaviour of this model was not considered.

This thesis also tested the feasibility of using a SSM based on medial representation, leading to the first SSM of three-dimensional shapes based on the MAT representation.

Although Styner (2001) uses the medial representation to add subsidiary information to a standard SSM, the approach described in Chapter 5 is solely based on the MAT. This allows the decoupling of radius information, thus

excluding thickness from the analysis.

By neglecting thickness it is possible to focus on finer aspects of shape that would be otherwise be missed, due to the predominance of thickness over other aspects of shape (Section 5.3.3).

This is something akin to what is achieved by scaling the shapes before model computation; size is the most evident difference between a set of human femurs and by normalising the data set relative to size, it is possible to detect other aspects of shape.

Since the MAT representation groups different kind of information that might not be compared directly (radii and positions of the centres), PCA based on the correlation matrix might be a better approach for the computation of the models.

All the models considered in this thesis do not take size into account, this was an arbitrary choice that simplifies feature matching between MAT representations (Section 3.3).

By applying orthomax rotations to the SSM based on the MAT representation, it is possible to relate morphological changes to the action of main muscular groups. Even though the effects of muscles on bone shape has already been described previously, this model infers this relationship directly from shapes and not by studying the effects of these forces on the bone, either through *in vivo* observations (Glauber & Vizkelety, 1966) or through clinical or cadaveric studies (Brookes & Wardle, 1962). The correspondence with the different muscular groups was performed solely on the basis of the location of the insertion points of the particular muscles; by considering the change of orientation during main activities, such as walking or running, it could be possible to refine this correspondence, opening the possibility to simulate the effect of muscle impairment on femoral anatomy.

By using the MAT model, it is also possible to reconstruct the complete femoral shape from partial information. Although the use of the point cloud representation is technically feasible, the MAT representation simplifies fea-

ture matching and therefore was the one of choice. The nature of the MAT could also simplify the reconstruction of the three-dimensional shape from planar images, although this would require to investigate the relationship between the MAT of three-dimensional objects and the medial representation of their projections.

When re-establishing the geometry of bones and joints, the usual practice consists of taking the contralateral limb (Bejui-Hugues *et al.*, 2007). However, this kind of information might not always be available, for instance in the (rare) cases of simultaneous bilateral fractures reported in literature (Schwartzmann *et al.*, 2001; Schröder & Marti, 2001; Tompkins *et al.*, 1990).

Moreover, the symmetry of the limbs is still debated (Cuk *et al.*, 2001), and being able to reconstruct the anatomy before the traumatic event starting from the shape of fragments, can possibly lead to better surgical outcomes.

In order to determine the best approach to follow in treating a fractured bone, surgeons often use classification methods which rely on the geometrical parameters introduced in Chapter 2. Medical images are usually limited to the region of the lesion, thus excluding important landmark for the evaluation of some parameters. The reconstruction will recreate this landmarks, enabling a consistent definition of the parameters.

In this thesis, it is assumed that there is a need to reconstruct the original shape of the femur. However, when considering a femoral fracture, it may be argued that the original anatomy could be somehow responsible to some extent for the occurrence of the traumatic event in the first place.

Nevertheless, since in clinical practice the possible scenarios are the most diverse, knowing the original shape may provide additional information that can assist clinicians in planning the best treatment or monitoring the recovery process after surgery. The representation of the shape which is introduced by the SSM can also allow for better classification of fractures, that can assist the surgeon in deciding the optimal treatment.

The approach introduced in this thesis is not limited to femoral fractures;

provided a sufficiently large dataset of intact bone is available, a SSM of the humerus can be computed, thus allowing for the application of this method to the treatment of proximal humeral fractures. This would be particularly helpful, as this type of fracture is quite frequent, especially in the elderly population, and the asymmetry between the left and right limb would be more evident, due to the predominant use of one arm compared to the other (Cuk *et al.*, 2001).

Reconstruction of shape from partial information can also play an important role in paleontology, archeology and forensic science (Carpenter, 1984; Grevin *et al.*, 1998). In all these fields there is often the need to recreate the original shape; the accuracy of the reconstruction does depend on the experience and skills of the operator. By formalising the available knowledge on a given shape in the form of a SSM, it is possible to obtain improved results.

7.3 Future Work

The main aim of this thesis was to explore different modelling approaches to femoral anatomy and define the scope of their application.

Re-computation of the models starting from a larger data-set would be one of the main improvements on this work, as obviously it would increase their power. Of course, the model can be constantly upgraded whenever a new suitable (i.e. healthy) instance of shape is available for processing.

A rigorous analysis of the behaviour of the rotated model in describing real pathological deformities would be needed to validate the model, and possibly introduce this technique in a diagnostic context. The correlation between the rotated components and the main muscular groups should be further investigated; the complex mechanical behaviour of the femur will require an in-depth analysis of the effects of muscles on the developing femur.

The effects of the simplification parameters introduced when dealing with

the MAT should also be tested to evaluate how the model is affected. The MAT naturally introduced a hierarchy within the different features of shape. In this thesis it was decided to use a degree of simplification that would preserve as many features as possible but keeping the computational time for reconstruction purposes reasonably small. Hopefully, with the increasing power of rendering hardware and available computational power, it would be possible to employ more accurate descriptions.

The models were all based on the surface representation of shape; modelling shape as a solid body can allow to understand volume changes in a better way and also enable to test mechanical hypothesis via finite element modelling. Since the PCs represent a vector field defined on points, by modelling the entire volume it would be possible to employ standard vector field operators (such as divergence and curl), thus allowing a proper definition and quantification of rotations, translations and changes in volume.

Regarding the predicting tool, the simulated scenario offered in Chapter 6 can be considered close enough to reality to predict good performances also in real life. However, tests on real bones should be performed. A large number of fracture humeri is available as well as a set of healthy bones, thus allowing for extensive testing of the predicting capabilities of the models.

Appendix A

Muscles Originating or Inserting on the Femur

The shape of the femur in the adult population is the result of many factors acting during development. The loading conditions (both static and dynamic), to which the femur is exposed, are among the main causes that influence bone shape since the early stages of development (Villemure & Stokes, 2009). It is therefore worth considering the main muscle groups that act on the bone itself, as together with body weight they define its loading conditions.

These muscles allow for a series of movements of the femur relatively to other bones, which are essential for locomotion. These movements are defined in terms of the anatomical planes. There are 3 anatomical planes and their position is depicted in Fig. A.1.

The main movements at the hip joint are:

- *Extension/Flexion*, in which the femur moves in the sagittal plane. (Fig. A.2 A)
- *Adduction/Abduction*, where the first movement brings the femur closer to the sagittal plane, whilst the second one is the opposite movement

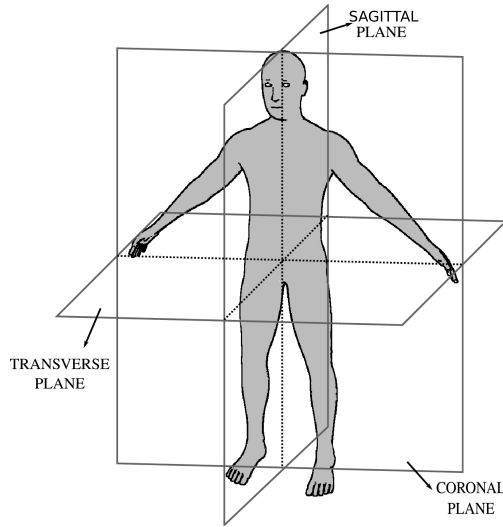


Figure A.1: Anatomical Planes of the human body

(Fig. A.2 B).

- *Internal/External Rotation*, which is a rotation towards or away from the centre of the body (Fig. A.3).

The main muscles, inserting on or originating from the femur, can be classified based on their origin, function or other criteria, such as their innervation. In this thesis a topological classification is employed (Schuenke *et al.*, 2005).

Considering the hip and gluteus muscles, they can be divided into 2 distinct groups (Fig. A.4):

- The inner hip muscles
- The outer hip muscles

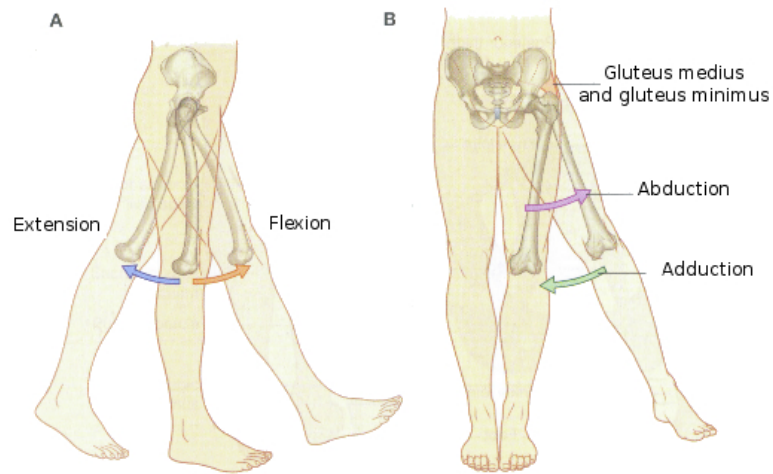


Figure A.2: Movement at the hip joint. Extension/Flexion (A) and Abduction/Adduction (B) (Gray)

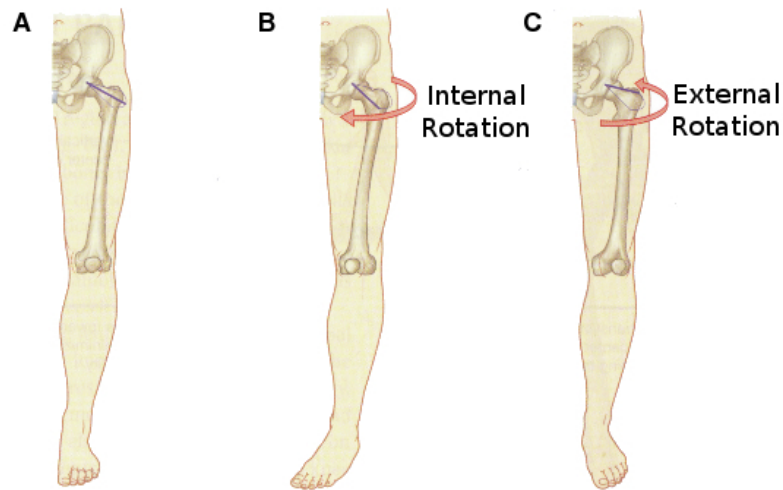
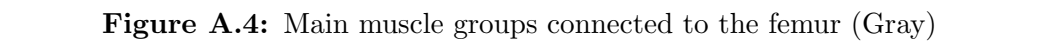


Figure A.3: Movement at the hip joint. (A) Rest Position (B) Internal Rotation (C) External Rotation (Gray)



A.1 The hip and gluteus muscles: the inner hip muscles

This group includes three different muscles that are sometimes addressed with a common name. These 3 muscles are:

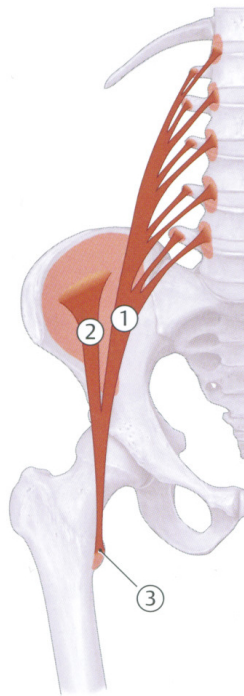


Figure A.5: Schematic of the Inner Hip Muscles (Numbered according to the text) (Schuenke *et al.*, 2005)

1. *M. psoas major*
2. *M. psoas minor*
3. *M. iliacus*

Since these three muscles share the same insertion point on the lesser trochanter of the femur (Fig. A.5), they are usually named *Iliopsoas Muscle*. The function of this muscle on the hip joint is to flex and externally rotate the joint.

Being the most powerful flexor of the hip joint it has an important role for standing, walking and running. Since walking influences the development of femoral shape it is expected that this muscle would play a relevant role in determining the adult shape of the femur.

A.2 The hip and gluteus muscles: the outer hip muscles

This group of muscles can be subdivided further according to the orientation of the muscles.

Among the *vertically-oriented* muscles there are:

1. *M. gluteus maximus*, which inserts on the posterior region of the proximal part of the femoral shaft. The main action of the muscle is to extend and externally rotate the hip. The upper fibres of the muscle insert on the iliotibial tract, a tendinous structure inserting on the proximal tibia. This structure provides stability to the hip joint and redistributes the stresses along the shaft due to the eccentric loading of the femur.
2. *M. gluteus medius*, which attaches to the lateral surface of the greater trochanter and, as a whole muscle, abducts the hip and stabilises the hip joint in the coronal plane. The anterior and posterior part take part in flexion/extension and internal/external rotation, respectively.
3. *M. gluteus minimus*, that inserts on the antero-lateral surface of the greater trochanter of the femur and whose main action is to abduct the hip and stabilise the pelvis in the coronal plane.

Muscles Originating or Inserting on the Femur

4. *M. tensor fascia latae*, which does not insert directly onto the femur, but is still affecting its loading conditions as it is connected to the iliotibial tract.
5. *M. piriformis*, which inserts on the apex of the greater trochanter and externally rotates, abducts and extends the hip joint.

A simplified representation of these muscles including their insertions and origins is given in Fig. A.6.



Figure A.6: Schematic of the vertically-oriented Outer Hip Muscles (Numbered according to the text) (Schuenke *et al.*, 2005)

Another group of muscles, that is included in the outer hip group, has a horizontal orientation. The muscles in this group are:

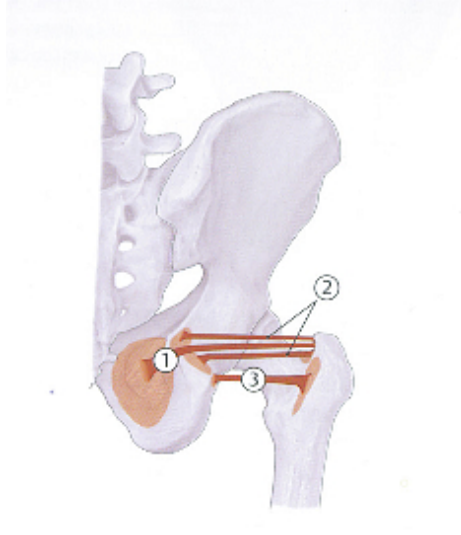


Figure A.7: Schematic of the horizontally-oriented Outer Hip Muscles (Numbered according to the text) (Schuenke *et al.*, 2005)

1. *M. obturatorius internus*, which is involved in the external rotation, adduction and extension of the hip joint. Its insertion point on the femur lies on the medial face of the greater trochanter.
2. *Ms. gemelli*, which insert together with the obturatorius on the medial face of the greater trochanter. Their action is the same as the obturatorius.
3. *M. quadratus femoralis*, that inserts posteriorly in the region between the 2 trochanters. It is involved in the rotation and adduction of the hip joint.

A schematic representation of these muscles is given in Fig. A.7.

A.3 The Anterior Thigh Muscles: the adductor group

The majority of these muscles insert on the posterior region of the shaft of the femur. They all are involved in the adduction of the hip joint, although some of them are also involved in other movements such as flexion (to different degrees) or external rotation.

The schematic representation of the origin and insertion points of these muscles is given in Fig. A.8.

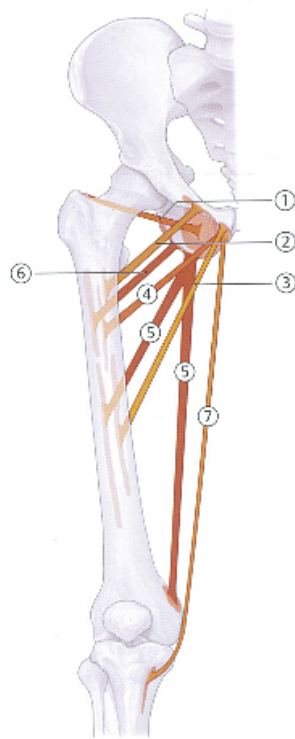


Figure A.8: Schematic of the Adductor Group, (Numbered according to the text)
(Schuenke *et al.*, 2005)

The muscles in this group are:

1. *M. obturatorius externus*, which inserts on the greater trochanter and besides being involved in the adduction of the hip joint, stabilises the pelvis in the sagittal plane.
2. *M. pectineus*, which inserts on the posterior face on the proximal femur on a region called pectineal line.
- 3-6 *Ms. adductor longus, brevis, magnus and minimus*, they all insert on the posterior face of the femur at different levels. The superficial part of the adductor magnus has a tendineous insertion on the medial epicondyle of the femur. As their names suggest all these muscles are responsible for the adduction of the hip joint and they also provide stability to the hip joint in the coronal and sagittal planes.
7. *M. gracilis*, this muscle is involved in the adduction and flexion of the hip joint but inserts on the tibia.

A possible classification of the thigh muscles considers the anterior and posterior muscles independently. The posterior and anterior muscles can be further sub-classified based on their function.

A.4 The Anterior Thigh Muscles: the extensor group

This group of muscles is responsible for the extension of the knee and, to a lesser extent, the flexion of the hip joint.

The muscles comprising this group are depicted in Fig. A.9 and are:

1. *M. sartorius*, which is involved in the flexion of the hip and the knee. It also abducts and externally rotates the hip joint. Its insertion point is not located on the femur but on the proximal tibia.
2. *M. Femoral Quadriceps*, which can be further divided in 5 muscles

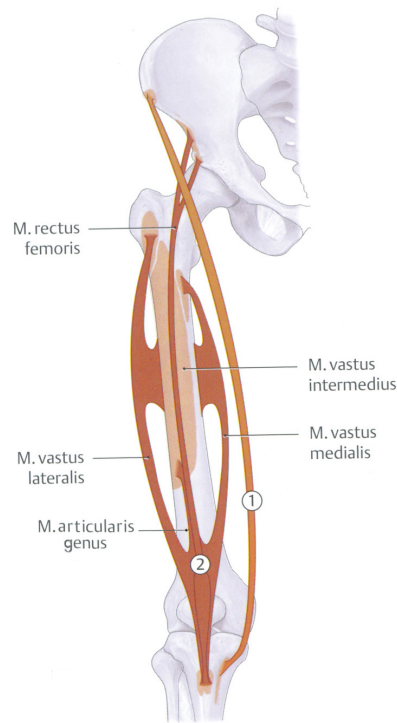


Figure A.9: Schematic of the Hip Extensor Muscles (Numbered according to the text) (Schuenke *et al.*, 2005)

- *M. rectus femoris*
- *M. vastus medialis*
- *M. vastus lateralis*
- *M. vastus intermedius*
- *M. articularis genus*

Among these muscles the vastus lateralis and medialis insert on the posterior region of the shaft, whilst the vastus intermedius inserts on the anterior part of the shaft.

Whenever the quadriceps is weak or paralysed, the centre of gravity of the

whole body is shifted forward as the stability in standing position is ensured by the hyper-extension of the knee obtained by the flexion effect of gravity on the knee joint.

A.5 The Posterior Thigh Muscles: the flexor group

The muscles in this group have an antagonistic action compared to those just introduced, thus flexing the knee joint and either externally or internally rotating it. Not all the muscles in this group originate or insert on the femur, however they will all be introduced for completeness.

The muscles in this group are (Fig. A.10):

1. *M. biceps femoris*, which has two originating points, one being located on the middle third of the femur. This muscle is directly involved in the stabilisation of the pelvis and it also assists hip joint extension.
- 2,3 *M. semimebranosus and semitendinosus*, which are not directly connected to the femur and have the main effect of extending the hip and flexing the knee joint.
4. *M. popliteus*, which originates from the lateral epicondyle and is involved in the flexion and internal rotation of the knee.

A.6 The Leg Muscles: the posterior compartment

This group of muscles is the only other one that includes muscles connected to the femur. The two muscles in the group are:



Figure A.10: Schematic of the posterior compartment of the thigh muscles (Numbered according to the text) (Schuenke *et al.*, 2005)

1. *M. triceps surae*, which is composed of two muscles, only one of which is directly connected to the femur.
 - *M. soleus*
 - *M. gastrocnemius*, which inserts on the two epicondyles of the femur and flexes the knee joint.
2. *M. plantaris*, which originates from the femur but has a negligible action since it has a very small cross section.

The schematic of the posterior compartment is shown in Fig. A.11.



Figure A.11: Schematic of the posterior compartment of the leg muscles (Numbered according to the text) (Schuenke *et al.*, 2005)

Appendix B

Code

B.1 Reconstruction of Shape from MAT

The following code was used to visualise the MATs in a `vtkRenderWindow` and to save the resulting shape into a `vtk` file for further visualisation.

```
1 #include <vtkPolyDataWriter.h>
2 #include <vtkRenderer.h>
3 #include <vtkRenderWindow.h>
4 #include <vtkActor.h>
5 #include <vtkPolyDataMapper.h>
6 #include <vtkSphereSource.h>
7 #include <vtkSphere.h>
8 #include <vtkImplicitBoolean.h>
9 #include <vtkSampleFunction.h>
10 #include <vtkMarchingContourFilter.h>
11 #include <vtkPolyDataMapper.h>
12
13 void main(int argc, char **argv)
14 {
15     ifstream istream(argv[1]);    // Getting stream from file name from [...]
16     [...] arguments
17     int n_spheres=atoi(argv[2]); // Getting number of MAT points
18
19     // Boolean implicit function that will be set to union
20     vtkImplicitBoolean *union = vtkImplicitBoolean::New();
```

```
21  union->SetOperationTypeToUnion();
22
23  // Rendering commands
24  vtkRenderer *ren = vtkRenderer::New();
25  vtkRenderWindow *renWin = vtkRenderWindow::New();
26
27  ren->SetBackground(1.0, 1.0, 1.0);
28  double x,y,z,radius;
29
30  // Initialise maximum/minimum coordinates to very low/high values
31
32  double xm,xM,ym,yM,zm,zM,RM;
33
34  xm=100000;
35  ym=100000;
36  zm=100000;
37  xM=-100000;
38  yM=-100000;
39  zM=-100000;
40  RM=0;
41
42  // Reading coordinates of MAT points and find max-min coordinates
43
44  for (int ispheres = 0; ispheres < n_spheres; ispheres++)
45  {
46      // Initialise sphere function LM
47
48      vtkSphere *LM = vtkSphere::New();
49
50      istream >> x;
51      istream >> y;
52      istream >> z;
53      istream >> radius;
54
55      LM->SetCenter(x,y,z);
56      LM->SetRadius(radius);
57
58
59      if (x>xM)
60          xM=x;
61      if (x<xm)
62          xm=x;
63      if (y>yM)
64          yM=y;
65      if (y<ym)
66          ym=y;
```

```
67     if (z>zM)
68         zM=z;
69     if (z<zm)
70         zm=z;
71     if (radius>RM)
72         RM=radius;
73
74     // Add function to the boolean function
75
76     union->AddFunction(LM);
77
78     PntMapper->SetInput(LM->GetOutput());
79
80     vtkActor *PntActor = vtkActor::New();
81     PntActor->SetMapper(PntMapper);
82     PntActor->GetProperty()->SetColor(0.92, 0.27, 0.34);
83     PntActor->GetProperty()->SetOpacity(opacity);
84
85     ren->AddActor(PntActor);
86
87     LM->Delete();
88     PntMapper->Delete();
89     PntActor->Delete();*/
90 }
91
92
93 // Sample the implicit boolean function to get poly data
94 vtkSampleFunction *pb = vtkSampleFunction::New();
95
96 pb->SetImplicitFunction(union);
97 pb->SetModelBounds(xm-RM,xM+RM,ym-RM,yM+RM,zm-RM,zM+RM);
98 pb->SetSampleDimensions(20,20,100);
99 pb->ComputeNormalsOff();
100
101
102 // Initializing the marching contour filter to get the poly data
103 vtkMarchingContourFilter *mcf = vtkMarchingContourFilter::New();
104
105 mcf->SetInputConnection(pb->GetOutputPort());
106 mcf->SetValue(0,0.0);
107
108 vtkPolyDataMapper *pdm = vtkPolyDataMapper::New();
109
110 pdm->SetInputConnection(mcf->GetOutputPort());
111 pdm->ScalarVisibilityOff();
112
```

```
113
114 // Saving the reconstructed shape into vtk format
115 vtkPolyDataWriter *pdw = vtkPolyDataWriter::New();
116
117 pdw->SetFileName(argv[3]);
118
119 pdw->SetInputConnection(mcf->GetOutputPort());
120
121 pdw->Write();
122 }
```

B.2 Medial Axis Computation

B.2.1 Computing Normal to Points

```
1 // normals.cpp : Read a vtk file and gives as an output a list of points [...]
  [...] of the surface and their normals
2 // Author: Angelo Tardugno, Imperial College London
3
4 #include "stdafx.h"
5 #include <CGAL/basic.h>
6 #include <list>
7
8 // standard includes
9 #include <iostream>
10 #include <fstream>
11 #include <cassert>
12 #include <stdio.h>
13 #include <string.h>
14 #include <vtkPoints.h>
15 #include <vtkPolyData.h>
16 #include <vtkPolyDataReader.h>
17 #include <vtkPolyDataWriter.h>
18 #include <vtkSTLWriter.h>
19 #include <vtkSTLReader.h>
20 #include "vtkPolyDataNormals.h"
21 #include <vtkDataArray.h>
22 #include <vtkPointData.h>
23 #include <vtkIdList.h>
24
25 using namespace std;
26
```



```
27 int main(int argc, char **argv)
28 {
29
30     // Reading vtk file
31     vtkPolyDataReader *readerS = vtkPolyDataReader::New();
32     readerS->SetFileName(argv[1]);
33     readerS->Modified();
34
35     vtkPointSet *pshape;
36     vtkPoints *shape_pnts = vtkPoints::New();
37     pshape = readerS->GetOutput();
38     pshape->Register(pshape);
39     readerS->Delete();
40     shape_pnts = pshape->GetPoints();
41
42     vtkPolyData *pdata = vtkPolyData::New();
43     pdata=readerS->GetOutput();
44
45     pdata->Modified();
46     pdata->Update();
47
48     // Calculating normals to shape
49     vtkPolyDataNormals *surfNormals = vtkPolyDataNormals::New();
50
51     surfNormals->SetFlipNormals(1);
52     surfNormals->SetAutoOrientNormals(1);
53     surfNormals->SetInput(pdata);
54     surfNormals->SetFeatureAngle(30.0);
55     surfNormals->SplittingOff();
56     surfNormals->ConsistencyOn();
57     surfNormals->Update();
58
59     const double PI=3.14159265;
60     int ptId=1;
61     double *norm;
62     ofstream file;
63     file.open(argv[2]);
64
65     file << pdata->GetNumberOfPoints() << endl;
66
67     for (int i=0;i<pdata->GetNumberOfPoints();i++)
68     {
69         double *point;
70         point=pdata->GetPoint(i);
71         // Writing point to file
72     }
```

Code

```
73     file << point[0] << " " << point[1] << " " << point[2] << " " ;
74     norm = surfNormals->GetOutput()->GetPointData()->GetNormals()->[...]
       [...] GetTuple(i);
75     // Writing normal to file
76     file << norm[0]<< " " << norm[1] << " " << norm[2] << " " ;
77
78     file.close();
79
80     return 0;
81 }
```

B.2.2 Compute Medial Axis Transform

```

1 // medial.cpp : Read output file from normals.cpp and computes the [...]
  [...] medial axis using Dey and Zhao's algorithm
2 // Author: Angelo Tardugno, Imperial College London
3 // Last Revision: 3/3/2008
4
5 #include "stdafx.h"
6 #include <CGAL/basic.h>
7 #include <list>
8 #include <set>
9
10 // standard includes [...]
11
12 //CGAL includes
13 #include <CGAL/Triangulation_hierarchy_3.h>
14 #include <CGAL/Exact_predicates_inexact_constructions_kernel.h>
15 #include <CGAL/Delaunay_triangulation_3.h>
16 #include <CGAL/MP_Float.h>
17 #include <CGAL/Lazy_exact_nt.h>
18 #include <CGAL/Quotient.h>
19 #include <CGAL/Triangulation_vertex_base_with_info_3.h>
20 #include <CGAL/Triangulation_ds_cell_base_3.h>
21 #include <CGAL/Triangulation_data_structure_3.h>
22 #include <CGAL/circulator.h>
23 #include <iterator>
24
25 typedef CGAL::Lazy_exact_nt<CGAL::Quotient<CGAL::MP_Float>> NT;
26
27 using namespace std;
28
29 typedef CGAL::Exact_predicates_inexact_constructions_kernel K;
30 typedef CGAL::Triangulation_vertex_base_with_info_3<int,K> tvb;
31 typedef CGAL::Triangulation_ds_cell_base_3<> tcb;
32 typedef CGAL::Triangulation_data_structure_3<tvb,tcb> tds;
33 typedef CGAL::Delaunay_triangulation_3<K,tds> Triangulation;
34 typedef CGAL::Direction_3<K> Direction_3;
35 typedef CGAL::Vector_3<K> Vector_3;
36
37
38 typedef Triangulation::Cell_handle Cell_handle;
39 typedef Triangulation::Vertex_handle Vertex_handle;
40 typedef Triangulation::Locate_type Locate_type;
41 typedef Triangulation::Point Point;
42 typedef Triangulation::Edge_iterator Edge_iterator;
43 typedef Triangulation::Cell_iterator Cell_iterator;

```

```
44 typedef Triangulation::Edge Edge;
45 typedef Triangulation::Cell_circulator Cell_circulator;
46
47 typedef CGAL::Container_from_circulator<Cell_circulator> cont;
48
49 typedef tds::Vertex_handle VH;
50
51 int main(int argc, char **argv)
52 {
53
54     ifstream file;
55     file.open(argv[1]);
56
57     ofstream medial;
58     medial.open(argv[2]);
59
60     const double PI = 3.14159265;
61
62     // Reading thresholds to be used for computation of medial axis
63     double threshold=atof(argv[3]);
64     double rthreshold=atof(argv[4]);
65
66     int n_points;
67
68     file >> n_points;
69
70     // Reading points from file
71
72     double *points = new double[3*n_points];
73     double *normals = new double[3*n_points];
74     double *cr = new double[n_points];
75
76     for (int i=0;i<n_points;i++)
77     {
78         file >> points[3*i];
79         file >> points[3*i+1];
80         file >> points[3*i+2];
81         file >> normals[3*i];
82         file >> normals[3*i+1];
83         file >> normals[3*i+2];
84         file >> cr[i];
85         double modNormal = 0;
86         // Normalising normal vectors
87         modNormal = sqrt(normals[3*i]*normals[3*i]+normals[3*i+1]*normals[3*...
            [...] i+1]+normals[3*i+2]*normals[3*i+2]);
88         normals[3*i]/=modNormal;
```

```
89     normals[3*i+1]/=modNormal;
90     normals[3*i+2]/=modNormal;
91 }
92
93
94 // Find the maximum x,y and z to build the bounding box
95
96 double mx,my,mz;
97 mx=abs(points[0]);
98 my=abs(points[1]);
99 mz=abs(points[2]);
100
101 for (int i=1;i<n_points;i++)
102 {
103     if (abs(points[3*i])>mx) mx=abs(points[3*i]);
104     if (abs(points[3*i+1])>my) my=abs(points[3*i+1]);
105     if (abs(points[3*i+2])>mz) mz=abs(points[3*i+2]);
106 }
107
108 // Computing Triangulations of points in data-set
109
110 Triangulation T;
111 VH vh;
112
113 //Populating the list
114 for (int i=0;i<n_points;i++)
115 {
116     vh = T.insert(Point(points[3*i], points[3*i+1], points[3*i+2]));
117
118     vh->info()=i;
119 }
120
121
122 std::list<Cell_handle> i_cells;
123 std::list<Vertex_handle> i_vertices;
124 std::set<Vertex_handle> vert_set;
125
126 int v_ind1, v_ind2;
127
128 // Iterate through triangulation vertices, consider the edges and [...]
129     [...] compute normal
130
131 for (Triangulation::Vertex_iterator vi = T.vertices_begin(); vi!=T.[...]
132     [...] vertices_end();vi++) // Iterate throughout the vertices of[...]
133     [...] Triangulation
134 {
```

```

132     i_cells.clear();
133     vert_set.clear();
134
135     Triangulation::Vertex v = *vi;
136
137     T.incident_cells(vi, std::back_inserter(i_cells));           [...]
138     [...] // Find incident cells to a given vertex vi
139     Cell_circulator cc, cc2;
140     Cell_circulator cc_end;
141     Point pt;
142     double xx, yy, zz;
143
144     // Iterate over the incident cells
145     for (std::list<Cell_handle>::iterator ci = i_cells.begin(); ci!= [...]
146         [...] i_cells.end(); ci++)
147     {
148         // Extract the index of the vertex within one of the incident [...]
149         [...] cells
150         v_ind1=(*ci)->index(vi);
151         for (int i=1; i<4; i++)
152         {
153             if (!(vert_set.count((*ci)->vertex((v_ind1+i)%4))))           [...]
154             [...] // Check if I've already visited one[...]
155             [...] particular vertex
156             {
157                 vert_set.insert((*ci)->vertex((v_ind1+i)%4));
158                 Edge e(*ci, v_ind1, ((v_ind1+i)%4));
159
160                 cc=T.incident_cells(e);                                     [...]
161                 [...] // [...]
162                 [...] Initialise circulators
163                 cc_end=cc;
164
165                 int n_cp=0;
166
167                 // Compute direction of edge e from vertex v_ind1 -> ([...]
168                 [...] v_ind1+1)%4
169
170                 Direction_3 d1, dn;
171
172                 double vx, vy, vz;
173                 vx = ((*ci)->vertex((v_ind1+i)%4)->point().x()) - ((*ci)[...]
174                     [...] ->vertex(v_ind1)->point().x());
175                 vy = ((*ci)->vertex((v_ind1+i)%4)->point().y()) - ((*ci)[...]
176                     [...] ->vertex(v_ind1)->point().y());

```

```

167         vz = ((*ci)->vertex((v_ind1+i)%4)->point().z()) - ((*ci)[...]
168             [...] ->vertex(v_ind1)->point().z());
169
170         double norm_v;
171
172         norm_v=sqrt(vx*vx+vy*vy+vz*vz);
173
174         vx /= norm_v;
175         vy /= norm_v;
176         vz /= norm_v;
177
178         // Retrieving information about vertex vi normals
179
180         int index = vi->info();
181
182         // Vector_3 vn(normals[3*index], normals[3*index+1], [...]
183             [...] normals[3*index+2]);
184
185         double dotprod;
186
187         dotprod = vx*normals[3*index]+vy*normals[3*index+1]+vz*[...]
188             [...] normals[3*index+2];
189
190         // *** Ratio Condition ***
191
192         bool rc = (norm_v/cr[index]) > rthreshold;
193         bool rc2 = norm_v < 25;
194         check << norm_v << " " << (dotprod > cos((threshold[...]
195             [...] /180)*PI)) << " " << (rc && rc2) << " ";
196         check << ((*ci)->vertex((v_ind1+i)%4)->point().x()) << [...]
197             [...] " ";
198         check << ((*ci)->vertex((v_ind1+i)%4)->point().y()) << [...]
199             [...] " ";
200         check << ((*ci)->vertex((v_ind1+i)%4)->point().z()) << [...]
201             [...] " ";
202         check << ((*ci)->vertex(v_ind1)->point().x()) << " ";
203         check << ((*ci)->vertex(v_ind1)->point().y()) << " ";
204         check << ((*ci)->vertex(v_ind1)->point().z()) << endl;
205
206         // *** Angle Condition ***
207
208         if ((dotprod > cos((threshold/180)*PI)) // || (rc && [...]
209             [...] dotprod>0.5 && rc2))

```

```

205
206     {
207
208         cout << "Inserting edge " << endl;
209
210         do
211         {
212             if (!(T.is_infinite(cc)))
213             {
214                 ++n_cp;
215             }
216         } while ((++cc) != (cc_end));
217
218
219         cc=T.incident_cells(e);
220         cc_end=cc;
221
222         do
223         {
224             if (!(T.is_infinite(cc)))
225             {
226
227                 pt=T.dual(cc);
228
229                 xx = pt.x();
230                 yy = pt.y();
231                 zz = pt.z();
232
233                 //computing angle between extremes of edges[...]
234                  [...] and each vertex of voronoi diagram
235
236                 double vv1[3],vv2[3];
237                 double lvv1,lvv2;
238
239                 vv1[0]=(*ci)->vertex((v_ind1+i)%4)->point().[...]
240                 [...] x() - xx;
241                 vv1[1]=(*ci)->vertex((v_ind1+i)%4)->point().[...]
242                 [...] y() - yy;
243                 vv1[2]=(*ci)->vertex((v_ind1+i)%4)->point().[...]
244                 [...] z() - zz;
245
246                 lvv1=sqrt(vv1[0]*vv1[0]+vv1[1]*vv1[1]+vv1[...]
247                 [...] [2]*vv1[2]);
248
249                 vv2[0]=(*ci)->vertex(v_ind1)->point().x() - [...]
250                 [...] xx;

```



```
245         vv2[1]=(*ci)->vertex(v_ind1)->point().y() - [...]
           [...] yy;
246         vv2[2]=(*ci)->vertex(v_ind1)->point().z() - [...]
           [...] zz;
247
248         lvv2=sqrt(vv2[0]*vv2[0]+vv2[1]*vv2[1]+vv2[...]
           [...] [2]*vv2[2]);
249
250         double angle;
251
252         angle = acos( (vv1[0]*vv2[0] + vv1[1]*vv2[1][...]
           [...] + vv1[2]*vv2[2])/(lvv1*lvv2) )*180/[...]
           [...] PI;
253
254         medial << xx << " " << yy << " " << zz << [...]
           [...] " " << lvv1 << " " << lvv2 << " " <<[...]
           [...] angle << endl;
255     }
256
257     } while (++cc!=cc_end);
258
259     }
260 }
261 }
262
263 }
264
265
266
267 }
268
269
270 // Closing file
271 medial.close();
272 check.close();
273
274 return 0;
275 }
```

B.2.3 MAT Simplification

```
1 #
2 def pow_distance(a,b):
3     dist = pow(a[0]-b[0],2)+pow(a[1]-b[1],2)+pow(a[2]-b[2],2)
4     pd = dist - pow(a[3]+b[3],2) # Different distance
5     return pd
6
7 def findRadius(centre , listOfSpheres):
8     Rad=[]
9     xc=centre[0]
10    yc=centre[1]
11    zc=centre[2]
12    Rc=centre[3]
13
14    # Find radius of minimum including spheres with centre in (xc,yc,zc)
15    for i in range(0,len(listOfSpheres)):
16        x=listOfSpheres[i][0]
17        y=listOfSpheres[i][1]
18        z=listOfSpheres[i][2]
19        R=listOfSpheres[i][3]
20
21        dist=sqrt(pow((x-xc),2)+pow((y-yc),2)+pow((z-zc),2))+R
22
23        Rad=Rad+[dist]
24
25
26    ind= Rad.index(max(Rad))
27
28    return max(Rad)
29
30 def findBoundingSphere(lOs):
31     # Initialise guess sphere to first sphere in the list
32     guess=array(lOs[0])
33
34     maxIterations = 1000;
35     gs=100.0
36     mgs=0.01
37     numIt=0
38     # Use maximum radius of list of spheres as initial radius
39     guess[3]=findRadius(guess , lOs)
40
41     # Test other values on a sphere of radius gs, check on direction +-x,[...]
42     [...] +-y +-z
43     while ((numIt<maxIterations) and (gs>mgs)):
```

```

44     tr=[0,0,0,0,0,0]
45
46     tp=[[ ],[ ],[ ],[ ],[ ],[ ]]
47
48     tp[0]=(guess)+array([gs,0,0,0])
49     tp[1]=(guess)+array([-1*gs,0,0,0])
50     tp[2]=(guess)+array([0,gs,0,0])
51     tp[3]=(guess)+array([0,-1*gs,0,0])
52     tp[4]=(guess)+array([0,0,gs,0])
53     tp[5]=(guess)+array([0,0,-1*gs,0])
54
55     tr[0]=findRadius(tp[0],lOs)
56     tr[1]=findRadius(tp[1],lOs)
57     tr[2]=findRadius(tp[2],lOs)
58     tr[3]=findRadius(tp[3],lOs)
59     tr[4]=findRadius(tp[4],lOs)
60     tr[5]=findRadius(tp[5],lOs)
61
62     # Test other values on a sphere of radius gs, check on direction [...]
63     [...] at 45 deg with axes
64     if (min(tr)<guess[3]):
65         guess=tp[tr.index(min(tr))]
66         guess[3]=min(tr)
67     else:
68         tr=[0,0,0,0,0,0,0,0]
69         tp=[[ ],[ ],[ ],[ ],[ ],[ ],[ ],[ ]]
70
71         cf=(1/sqrt(3))*gs
72         tp[0]=guess+array([cf,cf,cf,0])
73         tp[1]=guess+array([cf,cf,-cf,0])
74         tp[2]=guess+array([cf,-cf,cf,0])
75         tp[3]=guess+array([cf,-cf,-cf,0])
76         tp[4]=guess+array([-cf,cf,cf,0])
77         tp[5]=guess+array([-cf,cf,-cf,0])
78         tp[6]=guess+array([-cf,-cf,cf,0])
79         tp[7]=guess+array([-cf,-cf,-cf,0])
80
81         tr[0]=findRadius(tp[0],lOs)
82         tr[1]=findRadius(tp[1],lOs)
83         tr[2]=findRadius(tp[2],lOs)
84         tr[3]=findRadius(tp[3],lOs)
85         tr[4]=findRadius(tp[4],lOs)
86         tr[5]=findRadius(tp[5],lOs)
87         tr[6]=findRadius(tp[6],lOs)
88         tr[7]=findRadius(tp[7],lOs)

```

```
89         if (min(tr)<guess[3]):
90             guess=tp[tr.index(min(tr))]
91             guess[3]=min(tr)
92         else:
93             # If no decrease, radius of ball is decreased
94             gs=gs/2.0
95
96         numIt=numIt+1
97
98     return guess
99
100 def simplifyMA():
101
102     indMA = listBoxM2S.curselection()
103
104     for ind in indMA:
105         file2s = listOfM2S[int(ind)]+'.mat'
106
107         # Threshold for sphericity
108         sthreshold=0.9
109
110         finalSetOfSpheres=[]
111
112         f = open(file2s,"r")
113
114         out = open("simplified.log","w")
115
116         c=0
117         # Counting number of spheres
118         for lines in f.readlines():
119             c=c+1
120
121         f.close()
122
123         f = open(file2s,"r")
124
125         # Building array containig data
126         data = zeros((c,4),typecode='float')
127
128         for i in range(0,c):
129             lin = f.readline().split()
130             data[i,0]=float(lin[0])
131             data[i,1]=float(lin[1])
132             data[i,2]=float(lin[2])
133             data[i,3]=float(lin[3])
134
```

```
135
136     # Building distance matrix
137     print "Number of points:", c
138
139     n_items = int(((c*c-c)/2.0))
140
141     distance = zeros((n_items),typecode='float')
142     maxnumb=1
143
144     index = 0
145     for i in range(0,c):
146         for j in range(0,i):
147             distance[index]=pow_distance(data[i],data[j])
148             index=index+1
149
150     print len(distance)
151
152     tree = treecluster(distancematrix=distance,method='m')
153
154     tree.scale()
155
156     print "Tree:", tree
157     print "Lenght tree:", len(tree)
158
159     length = len(tree)-1
160
161
162     # Iterate through all nodes in tree
163
164     flag = zeros(len(tree))
165
166     for i in range(0,len(tree)):
167         j=len(tree)-i-1
168
169         if (tree[j].distance<0):
170             if (flag[j]==0):
171                 print "Examining node ", j
172
173                 list=[tree[j].left]+[tree[j].right]
174                 print list
175
176                 listVisitedNodes=[j] # Starting visited list with [...]
177                     [...] current node
178
179                 # Consider only clusters with negative distance (all [...]
180                     [...] clusters are intersecting)
```

```

179         while (min(list)<0):
180             temp=list
181             in2r=[]
182             for k in range(0,len(list)):
183                 if (list[k]<0):
184                     in2r=in2r+[k]
185                     listVisitedNodes=listVisitedNodes+[abs(list[k...
186                                     [...] ])-1] # Adding index to list of [...]
187                                     [...] visited nodes
188                     temp=temp+[tree[abs(list[k])-1].left]+[tree[ [...]
189                                     [...] abs(list[k])-1].right]
190
191             list=temp
192             temp=[]
193             for k in range(0,len(list)):
194                 if not(k in in2r):
195                     temp=temp+[list[k]]
196             list=temp
197
198         print >> out, len(list)
199
200         cspheres=[]
201         for ii in range(0,len(list)):
202             cspheres=cspheres+[map(None,data[list[ii],:])]
203
204         bsphere=findBoundingSphere(cspheres)
205
206         listOfRadii=[]
207         for ii in range(0,len(cspheres)):
208             listOfRadii=listOfRadii+[cspheres[ii][3]]
209
210         sphericity = max(listOfRadii)/float(bsphere[3])
211
212         # Simplify spheres with sphericity > than threshold
213         if (sphericity>sthreshold):
214             for ki in range(0,len(listVisitedNodes)):
215                 # Setting flag of all nodes visited so they're [...]
216                 [...] not visited again
217                 flag[listVisitedNodes[ki]]=1
218
219                 # Adding the circumscribing sphere in the set
220                 finalSetOfSpheres=finalSetOfSpheres+[bsphere]
221                 print len(cspheres), " have been simplified\n"
222
223         #list of spheres forming the simplified MA
224         for i in range(0,len(tree)):

```

```
221     j=len(tree)-i-1
222     print j
223     print "Node",tree[j].left
224     if (flag[j]==0):
225
226         print "Examining node ", j
227         list=[tree[j].left]+[tree[j].right]
228         while (min(list)<0):
229             temp=list
230             in2r=[]
231             for k in range(0,len(list)):
232
233                 if (list[k]<0) and (flag[abs(list[k])-1]==0):
234                     in2r=in2r+[k]
235                     temp=temp+[tree[abs(list[k])-1].left]+[tree[abs(...
                        [...] list[k])-1].right]
236
237                 if (list[k]<0) and (flag[abs(list[k])-1]==1):
238                     print "Tree Node has to be removed"
239                     in2r=in2r+[k]
240
241             list=temp
242             temp=[]
243
244             for k in range(0,len(list)):
245                 if not(k in in2r):
246                     temp=temp+[list[k]]
247             list=temp
248             if len(list)==0:
249                 break
250
251         for iq in range(0,len(list)):
252             finalSetOfSpheres=finalSetOfSpheres+[array(data[iq])]
253
254     outfile = open(listOfM2S[int(ind)]+'.mas','w')
255
256     setOfSpheres=(finalSetOfSpheres)
257
258     # retaining unique elements in set
259     checked = []
260     for e in setOfSpheres:
261         if e not in checked:
262             checked.append(e)
263
264     finalSetOfSpheres=checked
265
```

Code

```
266     for i in range(0,len(finalSetOfSpheres)):
267         outfile.write("%f %f %f %f\n" % (finalSetOfSpheres[i][0],[...
            [...] finalSetOfSpheres[i][1],finalSetOfSpheres[i][2],[...]
            [...] finalSetOfSpheres[i][3]))
268         print finalSetOfSpheres[i]
269     outfile.close()
```


B.3 Anatomical Measurements

```
1 import sys
2 import time
3 from cStringIO import StringIO
4 from vtk import *
5 from rpy import *
6 from scipy import *
7 import math as m
8 import scipy.linalg as la
9
10 print sys.argv[1]
11
12 f = open("eigenrotated.txt","r")
13
14 global p1
15 global p2
16 global nsL
17 global neckPlane
18 global neckPoint
19 global newpoints
20 global mpoints
21 global zT
22 global dirpn
23 global centreN1,centreN2
24 global T
25 global lito
26 global renWin
27 global iren
28
29 centreN1=[]
30 centreN2=[]
31 zT=5
32
33 def comRelease(a,b):
34     global clickCount
35     global nsL
36     clickCount +=1
37     global p1
38     global p2
39     global p3
40     global O
41     global dirUP
42     global centreN1,centreN2
43
```

```
44  # Action to perform after 1st click: Selection of medial inferior [...] point in frontal view
45  if (clickCount==1):
46      global nsL
47      s = a.GetEventPosition()
48      s = list(s)
49
50      startpt = []
51      rend.SetDisplayPoint(s[0], s[1], 0);
52      rend.DisplayToWorld();
53      p1 = []
54      p1 = rend.GetWorldPoint();
55      p1=list(p1)
56
57
58  # Action to perform after 2nd click: Selection of lateral superior [...] point in frontal view
59  if (clickCount==2):
60      global nsL
61      s2 = a.GetEventPosition()
62      s2 = list(s2)
63      startpt = []
64      rend.SetDisplayPoint(s2[0], s2[1], 0)
65      rend.DisplayToWorld();
66      p2 = []
67      p2 = rend.GetWorldPoint();
68      p2=list(p2)
69
70      # Render line passing through the first 2 selected points
71      nsL.SetPoint1(p1[0],maxY,p1[2])
72      nsL.SetPoint2(p2[0],maxY,p2[2])
73
74      nsLMapper = vtkPolyDataMapper()
75      nsLMapper.SetInput(nsL.GetOutput())
76
77      nsLActor = vtkActor()
78      nsLActor.SetMapper(nsLMapper)
79
80      nsLActor.GetProperty().SetColor(0,1,0)
81      nsLActor.GetProperty().SetLineWidth(4)
82      pA=[p1[0],maxY,p1[2]]
83      pB=[p2[0],maxY,p2[2]]
84      P12=[pB[0]-pA[0],pB[1]-pA[1],pB[2]-pA[2]]
85
86      # Computing new view direction as perpendicular to the line [...] passing through pA and pB
```

```

87         dirUP=cross (P12,[1,0,0])
88
89         dirUP=dirUP.tolist()
90
91         rend.AddActor(nsLActor)
92         cam.SetViewUp(dirUP[0],dirUP[1],dirUP[2])
93         cam.SetFocalPoint(pA[0],maxY,pA[2])
94         cam.SetPosition(pA[0]+P12[0]*100,pA[1]+P12[1]*100,pA[2]+P12[...
          [...] 2]*100)
95
96         # Turn perspective OFF
97         cam.ParallelProjectionOn()
98         rend.ResetCamera()
99
100        # Action to perform after 3rd click: Selection of point in [...]
          [...] transverse view
101        if (clickCount==3):
102            s2 = a.GetEventPosition()
103            s2 = list(s2)
104            startpt = []
105            rend.SetDisplayPoint(s2[0], s2[1], 0)
106            rend.DisplayToWorld();
107            p3 = []
108            p3 = rend.GetWorldPoint();
109            p3=list(p3)
110
111            pA=[p1[0],maxY,p1[2]]
112            pB=[p2[0],maxY,p2[2]]
113
114
115            P12=[pB[0]-pA[0],pB[1]-pA[1],pB[2]-pA[2]]
116            P12n=[i/sqrt(dot(P12,P12)) for i in P12]
117            dir3=[p3[0]-pB[0],p3[1]-pB[1],p3[2]-pB[2]]
118            comp3=dot(P12n,dir3)
119
120            pC=[pB[0]+dir3[0]-comp3*P12n[0],pB[1]+dir3[1]-comp3*P12n[1],pB[2]+[...]
          [...] dir3[2]-comp3*P12n[2]]
121
122            P13=[pC[0]-pA[0],pC[1]-pA[1],pC[2]-pA[2]]
123            P23=[P13[0]-P12[0],P13[1]-P12[1],P13[2]-P12[2]]
124
125            pD=[pA[0]+P23[0],pA[1]+P23[1],pA[2]+P23[2]]
126
127            # Render Selection rectangle
128
129            centrepA=vtkSphereSource()

```

```
130     centrepA.SetCenter(pA[0],pA[1],pA[2])
131     centrepA.SetRadius(3)
132     centrepAMap=vtkPolyDataMapper()
133     centrepAMap.SetInput(centrepA.GetOutput())
134     centrepAMap.ScalarVisibilityOff()
135     centrepAAc = vtkActor()
136     centrepAAc.SetMapper(centrepAMap)
137     centrepAAc.GetProperty().SetColor(0,0,1)
138     centrepAAc.PickableOff()
139     rend.AddActor(centrepAAc)
140
141     centrepB=vtkSphereSource()
142     centrepB.SetCenter(pB[0],pB[1],pB[2])
143     centrepB.SetRadius(3)
144     centrepBMap=vtkPolyDataMapper()
145     centrepBMap.SetInput(centrepB.GetOutput())
146     centrepBMap.ScalarVisibilityOff()
147     centrepBAc = vtkActor()
148     centrepBAc.SetMapper(centrepBMap)
149     centrepBAc.GetProperty().SetColor(0,0,1)
150     centrepBAc.PickableOff()
151     rend.AddActor(centrepBAc)
152
153     centrepC=vtkSphereSource()
154     centrepC.SetCenter(pC[0],pC[1],pC[2])
155     centrepC.SetRadius(3)
156     centrepCMap=vtkPolyDataMapper()
157     centrepCMap.SetInput(centrepC.GetOutput())
158     centrepCMap.ScalarVisibilityOff()
159     centrepCAc = vtkActor()
160     centrepCAc.SetMapper(centrepCMap)
161     centrepCAc.GetProperty().SetColor(0,0,1)
162     centrepCAc.PickableOff()
163     rend.AddActor(centrepCAc)
164
165
166     centrepD=vtkSphereSource()
167     centrepD.SetCenter(pD[0],pD[1],pD[2])
168     centrepD.SetRadius(3)
169     centrepDMap=vtkPolyDataMapper()
170     centrepDMap.SetInput(centrepD.GetOutput())
171     centrepDMap.ScalarVisibilityOff()
172     centrepDAc = vtkActor()
173     centrepDAc.SetMapper(centrepDMap)
174     centrepDAc.GetProperty().SetColor(0,0,1)
175     centrepDAc.PickableOff()
```

```
176         rend.AddActor(centrepDAc)
177
178
179         O=[(pA[0]+pB[0]+pC[0]+pD[0])/4,(pA[1]+pB[1]+pC[1]+pD[1])/4,(pA[...
            [...]2]+pB[2]+pC[2]+pD[2])/4]
180
181         dir1 =P12
182         norm1 = sqrt(dot(dir1,dir1))
183         dir1n = dir1/norm1
184         dir1n = dir1n.tolist()
185         dir1n=array(dir1n)
186
187
188         dir2=P13
189         norm2=sqrt(dot(dir2,dir2))
190         dir2n=dir2/norm2
191         dir2n=dir2n.tolist()
192         dir2n=array(dir2n)
193
194         dirp=cross(dir1n,dir2n)
195         normp = sqrt(dot(dirp,dirp))
196         dirpn=dirp/normp
197
198
199         Bline = vtkLineSource()
200         Bline.SetPoint1(pC[0],pC[1],pC[2])
201         Bline.SetPoint2(pB[0],pB[1],pB[2])
202
203         Bmapper = vtkPolyDataMapper()
204         Bmapper.SetInput(Bline.GetOutput())
205
206         Bactor = vtkActor()
207         Bactor.SetMapper(Bmapper)
208         Bactor.GetProperty().SetColor(0,1,0)
209         Bactor.GetProperty().SetLineWidth(4)
210
211         Cline = vtkLineSource()
212         Cline.SetPoint1(pC[0],pC[1],pC[2])
213         Cline.SetPoint2(pD[0],pD[1],pD[2])
214         Cmapper = vtkPolyDataMapper()
215         Cmapper.SetInput(Cline.GetOutput())
216
217         Cactor = vtkActor()
218         Cactor.SetMapper(Cmapper)
219         Cactor.GetProperty().SetColor(0,1,0)
220         Cactor.GetProperty().SetLineWidth(4)
```

```

221
222     Dline = vtkLineSource()
223     Dline.SetPoint1(pD[0],pD[1],pD[2])
224     Dline.SetPoint2(pA[0],pA[1],pA[2])
225     Dmapper = vtkPolyDataMapper()
226     Dmapper.SetInput(Dline.GetOutput())
227
228     Dactor = vtkActor()
229     Dactor.SetMapper(Dmapper)
230     Dactor.GetProperty().SetColor(0,1,0)
231     Cactor.GetProperty().SetLineWidth(4)
232     rend.AddActor(Bactor)
233     rend.AddActor(Cactor)
234     rend.AddActor(Dactor)
235
236     rend.Render()
237
238     # Select Points inside the rectangle and change rendered colour
239     pointsNeckx=[]
240     pointsNecky=[]
241     pointsNeckz=[]
242     pointsNeck=[]
243
244     pts = newpoints.tolist()
245     maxT=0;
246
247     for pt in pts:
248
249         vtest=[pt[0]-pA[0],pt[1]-pA[1],pt[2]-pA[2]]
250         if (dot(vtest,dir1n)>maxT):
251             maxT=dot(vtest,dir1n)
252         if (dot(vtest,dir2n)>maxT):
253             maxT=dot(vtest,dir2n)
254         cond1=(dot(vtest,dir1n)>0) and (dot(vtest,dir1n)<norm1)
255         cond2=(dot(vtest,dir2n)>0) and (dot(vtest,dir2n)<norm2)
256         cond3=(abs(dot(vtest,dirp))<zT)
257
258
259         if (cond1 and cond2 and cond3):
260             pointsNeckx+=pt[0]
261             pointsNecky+=pt[1]
262             pointsNeckz+=pt[2]
263             pointsNeck+=pt
264
265     A,err,centreN1,aA,aB = fitEllipse3D(pointsNeckx,pointsNecky,[...]
        [...] pointsNeckz,1)

```

```
266
267     centreN1S=vtkSphereSource()
268     centreN1S.SetCenter(centreN1[0],centreN1[1],centreN1[2])
269     centreN1S.SetRadius(1)
270     centreN1Map=vtkPolyDataMapper()
271     centreN1Map.SetInput(centreN1S.GetOutput())
272     centreN1Map.ScalarVisibilityOff()
273     centreN1Ac = vtkActor()
274     centreN1Ac.SetMapper(centreN1Map)
275     centreN1Ac.GetProperty().SetColor(1,0,0)
276     centreN1Ac.PickableOff()
277     rend.AddActor(centreN1Ac)
278
279
280     nsource=[]
281     nmappers=[]
282     nactors=[]
283     for ij in range(len(pointsNeckx)):
284         nsource+=vtkSphereSource()
285         nsource[ij].SetCenter(pointsNeck[ij][0],pointsNeck[ij][1],[...]
286             [...] pointsNeck[ij][2])
287
288         nmappers+=vtkPolyDataMapper()
289         nmappers[ij].SetInput(nsource[ij].GetOutput())
290         nmappers[ij].ScalarVisibilityOff()
291         nactors += [vtkActor()]
292         nactors[ij].SetMapper(nmappers[ij])
293         nactors[ij].GetProperty().SetColor(0,1,0)
294         nactors[ij].PickableOff()
295
296     cam.SetViewUp(dirISn[0],dirISn[1],dirISn[2])
297
298     cam.SetFocalPoint(0,-400,0)
299     cam.SetPosition(0,400,0)
300     cam.ParallelProjectionOn()
301     rend.ResetCamera()
302
303     rend.Render()
304
305     # Action to perform after 4th click: definition of line passing [...]
306     [...] through centre
307     # of the ellipse fitting the points within rectangle and new point. [...]
308     [...] This direction
309     # is used for refinement of capito-collar axis
```

```

309     if (clickCount==4):
310         s = a.GetEventPosition()
311         s = list(s)
312         startpt = []
313         rend.SetDisplayPoint(s[0], s[1], 0);
314         rend.DisplayToWorld();
315         p4 = []
316         p4 = rend.GetWorldPoint();
317         p4=list(p4)
318
319         centreN2=[p4[0],O[1],p4[2]]
320         neckAxis=[centreN2[0]-centreN1[0],centreN2[1]-centreN1[1],centreN2[...
321             [...] 2]-centreN1[2]]
322
323         nAL = vtkLineSource()
324         nAL.SetPoint1(centreN1[0],centreN1[1],centreN1[2])
325         nAL.SetPoint2(centreN1[0]+neckAxis[0],centreN1[1]+neckAxis[1],[...
326             [...] centreN1[2]+neckAxis[2])
327
328         nMap = vtkPolyDataMapper()
329         nMap.SetInput(nAL.GetOutput())
330         nActor = vtkActor()
331         nActor.SetMapper(nMap)
332         nActor.GetProperty().SetColor(0,1,0)
333
334         rend.AddActor(nActor)
335         normNeckA=sqrt(dot(neckAxis,neckAxis))
336         NA=float(normNeckA)
337         neckAxisn=[qi/normNeckA for qi in neckAxis]
338
339         centresOfNeck=[]
340         ecactors=[]
341         ecs=[]
342         ecmmap=[]
343         ecmmap=[]
344         pts = newpoints.tolist()
345
346         # Iterate the fitting process until no significant change in the [...]
347         [...] direction occurs
348         angleN = 0
349         niter=0
350         maxRad=0
351         while (angleN<0.99):
352             centresOfNeck=[]
353             niter+=1

```



```

352     print "Iteration #" + str(niter)
353     # ellipses are fitted from 10 to 90% of the line previously [...]
354     # [...] defined
355     for tk in range(10,90,2):
356         ec=[centreN1[0]+tk*neckAxisn[0]*NA/100.0,centreN1[1]+tk*[...]
357             [...] neckAxisn[1]*NA/100.0,centreN1[2]+tk*neckAxisn[2]*[...]
358             [...] NA/100.0]
359         x2f=[]
360         y2f=[]
361         z2f=[]
362         pointsFlag=0
363
364         for pt in pts:
365             test=[pt[0]-ec[0],pt[1]-ec[1],pt[2]-ec[2]]
366             if (abs(dot(test,neckAxisn))<zT):
367                 pointsFlag=1
368
369                 x2f+=pt[0]
370                 y2f+=pt[1]
371                 z2f+=pt[2]
372
373             if (pointsFlag==1):
374                 A,err,centreE,aAxis,bAxis = fitEllipse3D(x2f,y2f,z2f[...]
375                     [...],1)
376                 meanRad=(aAxis+bAxis)/2.0
377                 if (meanRad>maxRad):
378                     maxRad=meanRad
379
380                 centresOfNeck+=centreE
381
382     centresOfNeck = matrix(centresOfNeck)
383
384     meanCentres = centresOfNeck.mean(0)
385
386     centresOfNeck -= meanCentres
387
388     neckline = r.prcomp(centresOfNeck)
389
390     eigenVectorsN = neckline['rotation']
391     lineNDirection = eigenVectorsN[:,0]
392

```

```
393         dirN = [eigenVectorsN[0,0],eigenVectorsN[1,0],eigenVectorsN[...]  
394                 [...][2,0]]  
395  
396         normN=sqrt(sum([i**2 for i in dirN]))  
397  
398         dirNn=[i/normN for i in dirN]  
399  
400         # The new direction is flipped if not pointing in the same [...]  
401         [...] direction  
402         # of the first approximation of the capito-collar axis  
403  
404         if (dot(dirNn,neckAxis)<0):  
405             dirNn=[-1*elem for elem in dirNn]  
406  
407         neckAxisn/=sqrt(dot(neckAxisn,neckAxisn))  
408         dirNn/=sqrt(dot(dirNn,dirNn))  
409  
410         angleN=dot(neckAxisn,dirNn)  
411  
412         neckAxis=array([i for i in dirNn])  
413         neckAxisn=[i/sqrt(dot(neckAxis,neckAxis)) for i in neckAxis.[...]  
414                     [...][tolist()]]  
415         neckAxisn=array(neckAxisn)  
416  
417         neckAxis/=sqrt(dot(neckAxis,neckAxis))  
418         dirpn=neckAxisn  
419         cNe=[meanCentres[0,0],meanCentres[0,1],meanCentres[0,2]]  
420         neckAxisL = vtkLineSource()  
421         neckAxisL.SetPoint1(cNe[0]+200*dirpn[0],cNe[1]+200*dirpn[1],cNe[...]  
422                             [...][2]+200*dirpn[2])  
423         neckAxisL.SetPoint2(cNe[0],cNe[1],cNe[2])  
424  
425         neckMapper = vtkPolyDataMapper()  
426         neckMapper.SetInput(neckAxisL.GetOutput())  
427         neckActor = vtkActor()  
428         neckActor.SetMapper(neckMapper)  
429         pdmw = vtkPolyDataWriter()  
430         pdmw.SetInput(neckAxisL.GetOutput())  
431         pdmw.SetFileName(sys.argv[1]+"_N.vtk")  
432         pdmw.Write()  
433  
434         neckActor.GetProperty().SetColor(0,0,0)  
435         neckActor.GetProperty().SetLineWidth(4)  
436         rend.AddActor(neckActor)
```

```
435
436     # Creating ans savin L-M direction
437     lineLM = vtkLineSource()
438     CLM=cNe
439     lineLM.SetPoint1(CLM[0]-normLM*dirLMn[0],CLM[1]-normLM*dirLMn[1],[...]
440                     [...] CLM[2]-normLM*dirLMn[2])
441     lineLM.SetPoint2(CLM[0]+normLM*dirLMn[0],CLM[1]+normLM*dirLMn[1],[...]
442                     [...] CLM[2]+normLM*dirLMn[2])
443
444     lineLMMapper=vtkPolyDataMapper()
445     lineLMMapper.SetInput(lineLM.GetOutput())
446
447     pdmw = vtkPolyDataWriter()
448     pdmw.SetInput(lineLM.GetOutput())
449     pdmw.SetFileName(sys.argv[1]+"_LM.vtk")
450     pdmw.Write()
451     actorLM = vtkActor()
452     actorLM.SetMapper(lineLMMapper)
453
454     actorLM.GetProperty().SetColor(0,0,1)
455     actorLM.GetProperty().SetLineWidth(4)
456
457     rend.AddActor(actorLM)
458
459     # Creating ans savin P-A direction
460     linePA = vtkLineSource()
461
462     linePA.SetPoint1(CLM[0]-20*dirPAn[0],CLM[1]-20*dirPAn[1],CLM[...]
463                     [...] [2]-20*dirPAn[2])
464     linePA.SetPoint2(CLM[0]+200*dirPAn[0],CLM[1]+200*dirPAn[1],CLM[...]
465                     [...] [2]+200*dirPAn[2])
466
467     pdmw = vtkPolyDataWriter()
468     pdmw.SetInput(linePA.GetOutput())
469     pdmw.SetFileName(sys.argv[1]+"_PA.vtk")
470     pdmw.Write()
471
472     linePAMapper=vtkPolyDataMapper()
473     linePAMapper.SetInput(linePA.GetOutput())
474
475     actorPA = vtkActor()
476     actorPA.SetMapper(linePAMapper)
477
478     actorPA.GetProperty().SetColor(1,0,1)
479     actorPA.GetProperty().SetLineWidth(4)
```

```
477
478     rend.AddActor(actorPA)
479
480     # Creating ans savin I-S direction
481     lineIS = vtkLineSource()
482
483     lineIS.SetPoint1(CLM[0]-20*dirISn[0],CLM[1]-20*dirISn[1],CLM[...]  
        [...] [2]-20*dirISn[2])
484     lineIS.SetPoint2(CLM[0]+200*dirISn[0],CLM[1]+200*dirISn[1],CLM[...]  
        [...] [2]+200*dirISn[2])
485
486     pdmw = vtkPolyDataWriter()
487     pdmw.SetInput(lineIS.GetOutput())
488     pdmw.SetFileName(sys.argv[1]+"_IS.vtk")
489     pdmw.Write()
490
491     lineISMapper=vtkPolyDataMapper()
492     lineISMapper.SetInput(lineIS.GetOutput())
493
494     actorIS = vtkActor()
495     actorIS.SetMapper(lineISMapper)
496
497     actorIS.GetProperty().SetColor(0,1,0)
498     actorIS.GetProperty().SetLineWidth(4)
499
500     rend.AddActor(actorIS)
501
502     cam.SetViewUp(dirPAn[0],dirPAn[1],dirPAn[2])
503
504     cam.SetFocalPoint(CLM[0],CLM[1],CLM[2])
505     cam.SetPosition(CLM[0]+200*dirPAn[0],CLM[1]+200*dirPAn[1],CLM[...]  
        [...] [2]+200*dirPAn[0])
506
507     cam.ParallelProjectionOn()
508     rend.ResetCamera()
509
510
511     rend.Render()
512
513     pts = newpoints.tolist()
514     maxT=0;
515
516     # Find most posterior points on condyles MC= medial condyle LC= [...]  
        [...] lateral condyle
517
518     minM=10000
```

```
519     minL=10000
520     MC=[]
521     LC=[]
522
523
524     for pt in pts:
525         if (pt[0]<0) and (pt[2]<minM):
526             minM=pt[2]
527             MC=[pt[0],pt[1],pt[2]]
528         if (pt[0]>0) and (pt[2]<minL):
529             minL=pt[2]
530             LC=[pt[0],pt[1],pt[2]]
531
532     dirCON=[LC[0]-MC[0],LC[1]-MC[1],LC[2]-MC[2]]
533
534     normCON=sqrt(dot(dirCON,dirCON))
535
536     dirCONn=[il/normCON for il in dirCON]
537
538
539     lineCO = vtkLineSource()
540
541     lineCO.SetPoint1(MC[0],MC[1],MC[2])
542     lineCO.SetPoint2(LC[0],LC[1],LC[2])
543
544     lineCOMapper=vtkPolyDataMapper()
545     lineCOMapper.SetInput(lineCO.GetOutput())
546
547     pdmw = vtkPolyDataWriter()
548     pdmw.SetInput(lineCO.GetOutput())
549     pdmw.SetFileName(sys.argv[1]+"_COND.vtk")
550     pdmw.Write()
551     actorCO = vtkActor()
552     actorCO.SetMapper(lineCOMapper)
553
554     actorCO.GetProperty().SetColor(1,0,1)
555     actorCO.GetProperty().SetLineWidth(4)
556
557     rend.AddActor(actorCO)
558
559     Nx = dot(neckAxisn,dirLMn)
560     Ny = dot(neckAxisn,dirPAn)
561     Nz = dot(neckAxisn,dirISn)
562
563     NSangle = (90-m.acos(dot(neckAxisn,dirISn))*180/pi)+90
564     Version = (m.atan(Ny/Nx)*180/pi)
```

```
565
566     NxCond = dot(dirCON,dirLMn)
567     NyCond = dot(dirCON,dirPAn)
568     NzCond = dot(dirCON,dirISn)
569
570
571     Bangle = m.atan(abs(NzCond/NxCond))*180/pi
572     CONDLenght = normCON
573
574     fileOut = open("results.txt","a")
575     fileOut.write(str(NSangle)+" "+str(Version)+" "+str(Bangle)+"[...]
576                 [...] "+str(CONDLenght)+" "+str(maxRad)+"\n")
577
578     w2iFilter=vtkWindowToImageFilter()
579     w2iFilter.SetInput(renWin)
580     jpegw=vtkJPEGWriter()
581     jpegw.SetInput(w2iFilter.GetOutput())
582     jpegw.SetFileName(sys.argv[1]+"C.jpg")
583     jpegw.Write()
584
585     cam.SetViewUp(dirPAn[0],dirPAn[1],dirPAn[2])
586
587     cam.SetFocalPoint(0,0,-400)
588     cam.SetPosition(0,0,400)
589
590     cam.ParallelProjectionOn()
591     rend.ResetCamera()
592
593     w2iFilter=vtkWindowToImageFilter()
594     w2iFilter.SetInput(renWin)
595     jpegw=vtkJPEGWriter()
596     jpegw.SetInput(w2iFilter.GetOutput())
597     jpegw.SetFileName(sys.argv[1]+"T.jpg")
598     jpegw.Write()
599
600     cam.SetViewUp(dirISn[0],dirISn[1],dirISn[2])
601
602     cam.SetFocalPoint(-400,0,0)
603     cam.SetPosition(400,0,0)
604
605     cam.ParallelProjectionOn()
606     rend.ResetCamera()
607
608     w2iFilter=vtkWindowToImageFilter()
609     w2iFilter.SetInput(renWin)
610     jpegw=vtkJPEGWriter()
```

```
610     jpegw.SetInput(w2iFilter.GetOutput())
611     jpegw.SetFileName(sys.argv[1]+"S.jpg")
612     jpegw.Write()
613
614
615     exit()
616
617
618
619
620 def fitEllipse3D(x,y,z,plot):
621     p2f=[]
622     numpoints=len(x)
623
624     for i in range(len(x)):
625         p2f+=[[x[i],y[i],z[i]]]
626
627     p2f=matrix(p2f)
628     centroid=p2f.mean(0)
629
630     p2fc=p2f-centroid
631
632     PC=r.prcomp(p2fc)
633
634     Tr=PC['rotation']
635
636
637     XY=p2fc*Tr[:,0:2]
638
639     njl=(XY.shape)[0]
640
641     centre = XY.mean(0)
642
643     D=[]
644
645     for i in range(numpoints):
646         D+=[[ (XY[i,0]-centre[0,0])**2, (XY[i,0]-centre[0,0])*(XY[i,1]-[...]
647             [...] centre[0,1]), (XY[i,1]-centre[0,1])**2, XY[i,0]-centre[0,0], [...]
648             [...] XY[i,1]-centre[0,1], 1]]
649
650     D=matrix(D)
651
652     S = D.transpose()*D
653
654     # Constrain Matrix
655     C=zeros((6,6))
```

```

654     C[0][2]=-2
655     C[1][1]=1
656     C[2][0]=-2
657
658     err=0
659     eigSol=la.eig(S,C)
660
661     geval=eigSol[0]
662     gevec=eigSol[1]
663     indice=0
664     for i in range(int(geval.shape[0])):
665         if (geval[i]<0) and (not(isinf(geval[i]))):
666             indice=i
667
668     A=gevec[:,indice].tolist()
669
670
671     xc = centroid[0,0]+Tr[0,0]*centre[0,0]+Tr[0,1]*centre[0,1]
672     yc = centroid[0,1]+Tr[1,0]*centre[0,0]+Tr[1,1]*centre[0,1]
673     zc = centroid[0,2]+Tr[2,0]*centre[0,0]+Tr[2,1]*centre[0,1]
674     centrellipse=[xc,yc,zc]
675     tilt=1*0.5*m.atan(real(A[1])/(real(A[2]-A[0])))
676     angolo=0
677
678     a=real(A[0])
679     b=real(A[1])
680     c=real(A[2])
681     d=real(A[3])
682     f=real(A[4])
683     g=real(A[5])
684
685     aAxis = sqrt((2*(a*f**2 + c*d**2 + g*b**2 - 2*b*d*f - a*c*g))/((b**2-a[...
        [...] *c)*((sqrt((a-c)**2+4*b**2)-(a+c))))))
686     bAxis = sqrt((2*(a*f**2 + c*d**2 + g*b**2 - 2*b*d*f - a*c*g))/((b**2-a[...
        [...] *c)*((-1*sqrt((a-c)**2+4*b**2)-(a+c))))))
687
688
689     if (plot==1):
690         ellipsePt=[]
691         ellipseMap=[]
692         ellipseAc=[]
693         nsp=0
694         ellipseCentre=vtkSphereSource()
695         ellipseCentre.SetCenter(centrellipse)
696         ellipseCentre.SetRadius(1)
697         ellipseCMap=vtkPolyDataMapper()

```



```
698     ellipseCMap.SetInput(ellipseCentre.GetOutput())
699     ellipseCMap.ScalarVisibilityOff()
700     ellipseC = vtkActor()
701     ellipseC.SetMapper(ellipseCMap)
702     ellipseC.GetProperty().SetColor(0,0,1)
703     ellipseC.PickableOff()
704     rend.AddActor(ellipseC)
705     rend.Render()
706
707
708     A=[a,b,c,d,f,g]
709     return A,err,centrellipse,aAxis,bAxis
710
711
712
713
714 # Main Program #
715
716 lines = f.readlines()
717 numlines = (len(lines))
718
719 eigvect=[]
720
721 # Reading points from files
722 for j in lines:
723     eigvect+=[[float(q) for q in j.split()]]
724
725 matRPC=matrix(eigvect)
726
727 condyles=matRPC[:,1]
728 trochanter=matRPC[:,2]
729
730 loadCondyles=reshape(condyles,(-1,3))
731 loadTrochanter=reshape(trochanter,(-1,3))
732
733 normC=[]
734 normT=[]
735
736 for i in range(list(loadCondyles.shape)[0]):
737     cL=loadCondyles[i,:].tolist()
738
739     normC +=[sqrt(dot(cL[0],cL[0]))]
740     cT=loadTrochanter[i,:].tolist()
741     normT+=[sqrt(dot(cT[0],cT[0]))]
742
743
```

```
744
745 trC = 0.5*max(normC)
746 trT = 0.5*max(normT)
747
748 listC = []
749 listT = []
750 for i in range(list(loadCondyles.shape)[0]):
751     if (normC[i]>trC):
752         listC += [i]
753
754     if (normT[i]>trT):
755         listT += [i]
756
757
758 global newpoints,mpoints
759 reader=vtkPolyDataReader()
760
761 reader.SetFileName(sys.argv[1]+"r")
762 reader.Update()
763
764 shape = vtkPolyData()
765 shape = reader.GetOutput()
766
767
768 points=[]
769
770 numP=shape.GetNumberOfPoints()
771
772 for i in range(numP):
773     points+=[list(shape.GetPoint(i))]
774
775
776 mpoints=matrix(points)
777
778
779 rend=vtkRenderer()
780 renWin=vtkRenderWindow()
781
782 renWin.AddRenderer(rend)
783
784
785 diffTot=1;
786 numIt1=0
787 maxIt=100
788 maxIt2=100
789
```

```

790 # Rotate the bone along long axis and tentative condyle axis
791 # until contact points are found
792
793 while (difftot > 0.1 and numit1 < maxit):
794     numit1 += 1
795     #STEP 1
796
797     angleA = -10 * pi / 180
798     angleB = 10 * pi / 180
799
800     psi = angleA
801     Rz = matrix ([[cos(psi), -sin(psi), 0], [sin(psi), cos(psi), 0], [0, 0, 1]])
802
803     sA = mpoints * Rz
804
805     psi = angleB
806     Rz = matrix ([[cos(psi), -sin(psi), 0], [sin(psi), cos(psi), 0], [0, 0, 1]])
807
808     sB = mpoints * Rz
809
810     sM = mpoints
811
812     valuesA = [i[0] for i in sA[listC, 1].tolist()]
813     valuesB = [i[0] for i in sB[listC, 1].tolist()]
814     valuesM = [i[0] for i in sM[listC, 1].tolist()]
815
816     minA = valuesA.index(min(valuesA))
817     minB = valuesB.index(min(valuesB))
818     minM = valuesM.index(min(valuesM))
819
820     A = sA[listC[minA], :]
821     B = sB[listC[minB], :]
822     M = sM[listC[minM], :]
823
824     distAM = sqrt((A[0, 0] - M[0, 0])**2 + (A[0, 1] - M[0, 1])**2 + (A[0, 2] - M[0, 2])**2)
825     distBM = sqrt((B[0, 0] - M[0, 0])**2 + (B[0, 1] - M[0, 1])**2 + (B[0, 2] - M[0, 2])**2)
826
827     thres = (distAM + distBM) / 2
828
829     if (distAM > thres):
830         B = M
831         angleB = 0
832         angleA = -10 * pi / 180
833
834     if (distBM > thres):
835         A = M

```

```
836         angleA=0
837         angleB=10*pi/180
838
839
840
841         difference=abs(B[0,1]-A[0,1])
842         numit2=0
843
844         # STEP 2
845         while (difference > 0.1 and numit2 < maxit2):
846             numit2+=1
847
848             angleM=(angleA+angleB)/2
849
850             psi=angleA
851             Rz=matrix([[cos(psi), -sin(psi), 0], [sin(psi), cos(psi), [...]
852                        [...] 0], [0, 0, 1]])
853
854             sA=mpoints*Rz
855
856             psi=angleB
857             Rz=matrix([[cos(psi), -sin(psi), 0], [sin(psi), cos(psi), [...]
858                        [...] 0], [0, 0, 1]])
859
860             sB=mpoints*Rz
861
862             psi=angleM
863             Rz=matrix([[cos(psi), -sin(psi), 0], [sin(psi), cos(psi), [...]
864                        [...] 0], [0, 0, 1]])
865
866             sM=mpoints*Rz
867
868             valuesA=[i[0] for i in sA[listC,1].tolist()]
869             valuesB=[i[0] for i in sB[listC,1].tolist()]
870             valuesM=[i[0] for i in sM[listC,1].tolist()]
871
872             minA=valuesA.index(min(valuesA))
873             minB=valuesB.index(min(valuesB))
874             minM=valuesM.index(min(valuesM))
875
876             A=sA[listC[minA],:]
877             B=sB[listC[minB],:]
878             M=sM[listC[minM],:]
879
```

```

878     distAM=sqrt((A[0,0]-M[0,0])**2+(A[0,1]-M[0,1])**2+(A[0,2]-M[0,2]))[...]
879     distBM=sqrt((B[0,0]-M[0,0])**2+(B[0,1]-M[0,1])**2+(B[0,2]-M[0,2]))[...]
880
881
882     if (distAM>thres):
883         B=M
884         angleB=angleM
885
886
887     if (distBM>thres):
888         A=M
889         angleA=angleM
890
891     difference=abs(B[0,1]-A[0,1])
892
893     vecML=A-B
894
895     vecML=[vecML[0,0],vecML[0,1],vecML[0,2]]
896     normML=sqrt(sum([c**2 for c in vecML]))
897
898     U=[com/normML for com in vecML]
899
900     newpoints=sA
901
902     # Rotation about the newly found axis (general rotation about axis)
903
904     psi=-5*pi/180
905     Ru=[[U[0]**2+(1-U[0]**2)*cos(psi), U[0]*U[1]*(1-cos(psi))-U[2]*sin(psi)[...]
906         [...]), U[0]*U[2]*(1-cos(psi))+U[1]*sin(psi)], [U[0]*U[1]*(1-cos(psi)[...]
907         [...])+U[2]*sin(psi), U[1]**2+(1-U[1]**2)*cos(psi), U[1]*U[2]*(1-cos[...
908         [...](psi))-U[0]*sin(psi)], [U[0]*U[2]*(1-cos(psi))-U[1]*sin(psi), U[...
909         [...][1]*U[2]*(1-cos(psi))+U[0]*sin(psi), U[2]**2+(1-U[2]**2)*cos([...]
910         [...]psi)]]
911     sC=newpoints*Ru
912
913     psi=5*pi/180
914     Ru=[[U[0]**2+(1-U[0]**2)*cos(psi), U[0]*U[1]*(1-cos(psi))-U[2]*sin(psi)[...]
915         [...]), U[0]*U[2]*(1-cos(psi))+U[1]*sin(psi)], [U[0]*U[1]*(1-cos(psi)[...]
916         [...])+U[2]*sin(psi), U[1]**2+(1-U[1]**2)*cos(psi), U[1]*U[2]*(1-cos[...
917         [...](psi))-U[0]*sin(psi)], [U[0]*U[2]*(1-cos(psi))-U[1]*sin(psi), U[...
918         [...][1]*U[2]*(1-cos(psi))+U[0]*sin(psi), U[2]**2+(1-U[2]**2)*cos([...]
919         [...]psi)]]
920     sD=newpoints*Ru

```

```

912
913     sM=newpoints
914
915     ## Check if points minimum in the AP direction is on troch or [...]
916     [...] condyles fC = 1 if condyles
917
918     valuesCC=[i [0] for i in sC[listC,1].tolist()]
919     valuesDC=[i [0] for i in sD[listC,1].tolist()]
920     valuesMC=[i [0] for i in sM[listC,1].tolist()]
921
922     valuesCT=[i [0] for i in sC[listT,1].tolist()]
923     valuesDT=[i [0] for i in sD[listT,1].tolist()]
924     valuesMT=[i [0] for i in sM[listT,1].tolist()]
925
926     fC=0
927     fD=0
928     fM=0
929
930     minCC=valuesCC.index(min(valuesCC))
931     minCT=valuesCT.index(min(valuesCT))
932     minDC=valuesDC.index(min(valuesDC))
933     minDT=valuesDT.index(min(valuesDT))
934     minMC=valuesMC.index(min(valuesMC))
935     minMT=valuesMT.index(min(valuesMT))
936
937     if (min(valuesCC)<min(valuesCT)):
938         fC=1
939         C=sC[listC[minCC],:]
940     else:
941         C=sC[listT[minCT],:]
942     if (min(valuesDC)<min(valuesDT)):
943         fD=1
944         D=sD[listC[minDC],:]
945     else:
946         D=sD[listT[minDT],:]
947     if (min(valuesMC)<min(valuesMT)):
948         fM=1
949         M=sM[listC[minMC],:]
950     else:
951         M=sM[listT[minMT],:]
952
953     if (fM==fD):
954         angleD=0
955         D=M
956         angleC=-5*pi/180

```

```

957
958     if (fM==fC):
959         angleC=0
960         C=M
961         angleD=5*pi/180
962
963
964     difference=abs(D[0,1]-C[0,1])
965
966     while (difference > 0.1):
967
968         angleM=(angleC+angleD)/2
969
970         psi=angleC
971         Ru=[[U[0]**2+(1-U[0]**2)*cos(psi), U[0]*U[1]*(1-cos(psi))-U[2]*sin[...
          [...] (psi),U[0]*U[2]*(1-cos(psi))+U[1]*sin(psi)], [U[0]*U[1]*(1-[...
          [...] cos(psi))+U[2]*sin(psi), U[1]**2+(1-U[1]**2)*cos(psi),U[...
          [...] [1]*U[2]*(1-cos(psi))-U[0]*sin(psi)], [U[0]*U[2]*(1-cos(psi)[...
          [...] )]-U[1]*sin(psi), U[1]*U[2]*(1-cos(psi))+U[0]*sin(psi), U[...
          [...] [2]**2+(1-U[2]**2)*cos(psi)]]
972
973
974
975         sC=newpoints*Ru
976
977         psi=angleD
978         Ru=[[U[0]**2+(1-U[0]**2)*cos(psi), U[0]*U[1]*(1-cos(psi))-U[2]*sin[...
          [...] (psi),U[0]*U[2]*(1-cos(psi))+U[1]*sin(psi)], [U[0]*U[1]*(1-[...
          [...] cos(psi))+U[2]*sin(psi), U[1]**2+(1-U[1]**2)*cos(psi),U[...
          [...] [1]*U[2]*(1-cos(psi))-U[0]*sin(psi)], [U[0]*U[2]*(1-cos(psi)[...
          [...] )]-U[1]*sin(psi), U[1]*U[2]*(1-cos(psi))+U[0]*sin(psi), U[...
          [...] [2]**2+(1-U[2]**2)*cos(psi)]]
979
980
981         sD=newpoints*Ru
982
983         psi=angleM
984         Ru=[[U[0]**2+(1-U[0]**2)*cos(psi), U[0]*U[1]*(1-cos(psi))-U[2]*sin[...
          [...] (psi),U[0]*U[2]*(1-cos(psi))+U[1]*sin(psi)], [U[0]*U[1]*(1-[...
          [...] cos(psi))+U[2]*sin(psi), U[1]**2+(1-U[1]**2)*cos(psi),U[...
          [...] [1]*U[2]*(1-cos(psi))-U[0]*sin(psi)], [U[0]*U[2]*(1-cos(psi)[...
          [...] )]-U[1]*sin(psi), U[1]*U[2]*(1-cos(psi))+U[0]*sin(psi), U[...
          [...] [2]**2+(1-U[2]**2)*cos(psi)]]
985
986         sM=newpoints*Ru
987

```

```

988     valuesCC=[i [0] for i in sC[listC,1].tolist()]
989     valuesDC=[i [0] for i in sD[listC,1].tolist()]
990     valuesMC=[i [0] for i in sM[listC,1].tolist()]
991
992     valuesCT=[i [0] for i in sC[listT,1].tolist()]
993     valuesDT=[i [0] for i in sD[listT,1].tolist()]
994     valuesMT=[i [0] for i in sM[listT,1].tolist()]
995
996     fC=0
997     fD=0
998     fM=0
999
1000    minCC=valuesCC.index(min(valuesCC))
1001    minCT=valuesCT.index(min(valuesCT))
1002    minDC=valuesDC.index(min(valuesDC))
1003    minDT=valuesDT.index(min(valuesDT))
1004    minMC=valuesMC.index(min(valuesMC))
1005    minMT=valuesMT.index(min(valuesMT))
1006
1007
1008    if (min(valuesCC)<min(valuesCT)):
1009        fC=1
1010        C=sC[listC[minCC],:]
1011    else:
1012        C=sC[listT[minCT],:]
1013    if (min(valuesDC)<min(valuesDT)):
1014        fD=1
1015        D=sD[listC[minDC],:]
1016    else:
1017        D=sD[listT[minDT],:]
1018    if (min(valuesMC)<min(valuesMT)):
1019        fM=1
1020        M=sM[listC[minMC],:]
1021    else:
1022        M=sM[listT[minMT],:]
1023
1024    if (fM==fD):
1025        angleD=angleM
1026        D=M
1027
1028    if (fM==fC):
1029        angleC=angleM
1030        C=M
1031
1032
1033    difference=abs(D[0,1]-C[0,1])

```



```

1034
1035     newpoints=sC
1036     mpoints=newpoints
1037     diffTot=abs(B[0,1]-A[0,1])+abs(B[0,1]-C[0,1])
1038
1039 # Finding Long Axes Based on fitted ellipses
1040
1041 valuesZ=[i[0] for i in newpoints[:,2].tolist()]
1042 valuesY=[i[0] for i in newpoints[:,1].tolist()]
1043 minZ=min(valuesZ)
1044 maxZ=max(valuesZ)
1045 global minY,maxY
1046 minY=min(valuesY)
1047 maxY=max(valuesY)
1048 rangeZ=linspace(minZ,maxZ,100)
1049 listOfCentres=[]
1050 for t in range(1,len(rangeZ)-1):
1051     x=[]
1052     y=[]
1053     z=[]
1054     for j in range(numP):
1055         if (abs(newpoints[j,2]-rangeZ[t])<zT):
1056             x+=newpoints[j,0]
1057             y+=newpoints[j,1]
1058             z+=newpoints[j,2]
1059
1060     AA,error,ecentre,aAx,bAx = fitEllipse3D(x,y,z,0)
1061
1062     listOfCentres+=ecentre
1063
1064 # Fit line to centers of shaft
1065
1066 centresToFit = listOfCentres[20:80][:]
1067
1068 centresToFit = matrix(centresToFit)
1069
1070 meanCentresToFit = centresToFit.mean(0)
1071
1072 centresToFit -= meanCentresToFit
1073
1074 pcline = r.prcomp(centresToFit)
1075
1076 eigenVectors = pcline['rotation']
1077 lineDirection = eigenVectors[:,0]
1078
1079 dirIS = [eigenVectors[0,0],eigenVectors[1,0],eigenVectors[2,0]]

```

```

1080
1081 normIS=sqrt(sum([i**2 for i in dirIS]))
1082
1083 dirISn=[i/normIS for i in dirIS]
1084
1085 # Check Direction of IS axis
1086
1087 if (dirISn[2] < 0):
1088     dirISn=[-i for i in dirISn]
1089
1090
1091 # Long Axis
1092
1093 lengthZ=(maxZ-minZ)/1.8
1094 longAxis = vtkLineSource()
1095 longAxis.SetPoint1(meanCentresToFit[0,0]-lengthZ*lineDirection[0,],[...
    [...] meanCentresToFit[0,1]-lengthZ*lineDirection[1,],meanCentresToFit[...]
    [...] [0,2]-lengthZ*lineDirection[2,]])
1096 longAxis.SetPoint2(meanCentresToFit[0,0]+lengthZ*lineDirection[0,],[...
    [...] meanCentresToFit[0,1]+lengthZ*lineDirection[1,],meanCentresToFit[...]
    [...] [0,2]+lengthZ*lineDirection[2,]])
1097 longmap=vtkPolyDataMapper()
1098 longmap.SetInput(longAxis.GetOutput())
1099 longmap.ScalarVisibilityOff()
1100 longactor=vtkActor()
1101 longactor.SetMapper(longmap)
1102 longactor.GetProperty().SetColor(0,0,1)
1103
1104 rend.AddActor(longactor)
1105
1106 # Find final L->M axis
1107 dirLM = [B[0,0]-A[0,0],B[0,1]-A[0,1],B[0,2]-A[0,2]]
1108 normLM=sqrt(sum([i**2 for i in dirLM]))
1109 dirLMn=[i/normLM for i in dirLM]
1110
1111 # Check the direction of dirLMn
1112 if (dirLMn[0] < 0):
1113     dirLMn=[-i for i in dirLMn]
1114
1115 dirCond=dirLMn # This is the direction of the condylar line
1116
1117 dirPA = cross(dirISn,dirLMn).tolist()
1118
1119 normPA=sqrt(sum([i**2 for i in dirPA]))
1120
1121 dirPAN=[i/normPA for i in dirPA]

```

Code

```
1122
1123 #refine dirLMn
1124
1125 dirLM=cross(dirPAn,dirISn).tolist()
1126
1127 normLM2=sqrt(sum([i**2 for i in dirLM]))
1128
1129 dirLMn =[i/normLM2 for i in dirLM]
1130
1131 ## Rendering [...]
```

Code

References

- AHMAD, T., TAYLOR, C. J., LANITIS, A., & COOTES, T. F. 1997. Tracking and recognising hand gestures, using statistical shape models. *Image and Vision Computing*, **15**(5), 345–352.
- AMENTA, N., & BERN, M. 1999. Surface Reconstruction by Voronoi Filtering. *Discrete Computational Geometry*, **22**, 481–504.
- AMENTA, N., BERN, M., & EPPSTEIN, D. 1998. The crust and the Beta-skeleton: combinatorial curve reconstruction. *Graphical Models Image Processing*, **60**, 125–135.
- AMENTA, N., CHOI, S., & KOLLURI, R.K. 2001. The power crust, union of balls, and the medial axis transform. *Computational Geometry. Theory Appl.*, **19**, 127–153.
- ARGENSON, J.-N., RYEMBAULT, E., FLECHER, X., BRASSART, N., PARRATTE, S., & AUBANIAC, J.-M. 2005. Three-dimensional anatomy of the hip in osteoarthritis after developmental dysplasia. *J Bone Joint Surg [Br]*, **87-B**, 1192–1196.
- ATTALI, D., & LACHAUD, J.-O. 2001. Delaunay conforming iso-surface, skeleton extraction and noise removal. *Computational Geometry Theory Appl.*, **12**, 175–189.

References

- ATTALI, DOMINIC, & MONTANVERT, ANNICK. 1997. Computing and Simplifying 2D and 3D Continuous Skeletons. *Computer Vision and Image Understanding*, **67**(3), 261–273.
- BAJCSY, R., & SOLINA, F. 1987. Three Dimensional object representation Revisited. *Computer Society Press of the IEEE*.
- BAREQUET, G., & HAR-PELED, S. 2001. Efficiently approximating the Minimum-volume bounding box of a point set in three dimensions. *Journal of Algorithms*, **38**, 91–109.
- BEJUI-HUGUES, J., TAYOT, O., & GUYEN, O. 2007. Reconstructed hip joint architecture with a standard hip arthroplasty with a unique declined offset. *Interact Surg*, **2**, 220–222.
- BERNAARD, C. A., & JENNRICH, R.I. 2003. Orthomax rotation and perfect simple structure. *Psychometrika*, **68**(4), 585–588.
- BERNAARDS, CA, & JENNRICH, RI. 2005. Gradient Projection Algorithms and Software for Arbitrary Rotation Criteria in Factor Analysis. *Educational and Psychological Measurement*, **65**, 676–696.
- BESL, P.J., & MCKAY, ND. 1992. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **14**(2), 239–256.
- BLANZ, V., & VETTER, T. 2002. Reconstructing the complete 3d shape of faces from partial information. *Informationstechnik und Technische Informatik*, **44**(6), 1–8.
- BLUM, HENRY. 1967. A transformation for extracting new descriptors of form. *Models for the Perception of Speech and Visual Form*.
- BOTEV, Z. I., GROTHOWSKI, J. F., & KROESE, D. P. 2010. Kernel Density Estimation Via Diffusion. *Annals of Statistics*.

References

- BRANDT, J.W. 1994. Convergence and continuity Criteria for Discrete Approximation of the Continuous Planar Skeletons. *CVGIP: Image Understanding*, **59**, 116–124.
- BRIQUER, L LE, & GEE, J C. 1997. Design of a statistical model of brain shape. *Lecture Notes in Computer Science*, **1230**, 477–482.
- BROOKES, M, & WARDLE, EN. 1962. Muscle action and the shape of the femur. *The journal of bone and joint surgery*, **44B**(2).
- CARPENTER, K. 1984. Skeletal reconstruction and life restoration of Sauropelta (Ankylosauria: Nodosauridae) from the Cretaceous of North America. *J. Earth Sci.*
- COOKE, TD, SLED, EA, & SCUDAMORE, RA. 2007. Frontal plane knee alignment: a call for standardized measurement. *J Rheumatol*, **34**, 1796–801.
- COOTES, T.F., TWINING, C.J., & TAYLOR, C.J. 2004. Diffeomorphic Statistical Shape Models. *Proc. British Machine Vision Conference*, **1**, 447–456.
- CUK, T, LEBEN-SELJAK, P, & STEFANČIČ, M. 2001. Lateral Asymmetry of Human Long Bones. *Variability and Evolution*.
- DA SILVA, V J, YODA, J Y, & DE MELLO GONCALES, D. 2003. Anatomical Aspects Of the Proximal Femur of Adult Brazilians. *International Journal of Morphology*, **21**(4), 303–308.
- DAVIES, R. H., TWINING, C. J., ALLEN, P. DANIEL, COOTES, T. F., & TAYLOR, C. J. 2003. Shape Discrimination in the Hippocampus Using an MDL Model. *Lecture Notes in Computer Science*, **2732**, 38–50.

References

- DELIALIOGLUA, M.O., TASBASA, B.A., BAYRAKÇIA, K, DAGLARA, B, KURTB, M, AGARA, M, & GUNELA, U. 2006. Alternative reliable techniques in femoral torsion measurement. *J Pediatr Orthop B*, **15**, 28,33.
- DEY, TAMAL K., & ZHAO, WULUE. 2004. Approximating the Medial Axis from Voronoi Diagram with a Convergence Guarantee. *Algorithmica*, **38**, 179–200.
- EDWARDS, G. J., LANITIS, A., TAYLOR, C. J., & COOTES, T. F. 1998. Statistical models of face images improving specificity. *Image and Vision Computing*, **16**(3), 203–211.
- ETHIER, CR, & SIMMONS, CA. 2007. *Introductory Biomechanics*. Cambridge University Press.
- FITZGIBBON, A, PILU, M, & FISHER, RB. 1999. Direct Least Square Fitting of Ellipses. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, **21**(5).
- GIBLIN, P, & KIMIA, BB. 2004. A Formal Classification of 3D Medial Axis Points and Their Local Geometry. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, **26**(2).
- GLAUBER, A, & VIZKELETY, T. 1966. The influence of the iliopsoas muscle on femoral antetorsion. *Archives of Orthopaedic and Trauma Surgery*, **60**, 71–79.
- GLITSCH, U., & BAUMANN, W. 1997. The three-dimensional determination of internal loads in the lower extremity. *Journal of Biomechanics*, **30**(11-12), 1123 – 1131.
- GRAY. 2005. *Gray's Anatomy for Students*. Elsevier.

References

- GREVIN, G., BAILET, P., QUATREHOMME, G., & OLLIER, A. 1998. Anatomical reconstruction of fragments of burned human bones: a necessary means for forensic identification. *Forensic Science International*, **96**.
- HEMMER, MICHAEL. 2009. Algebraic Foundations. In: CGAL EDITORIAL BOARD (ed), *CGAL User and Reference Manual*, 3.5 edn.
- HOAGLIN, DAVID C., MOSTELLER, FREDERICK, & TUKEY., JOHN W. (eds). 1983. *Understanding robust and exploratory data analysis*. John Wiley and Sons Ltd.
- I, J. S. KIM, PARK, T.S., PARK, S.B., I, J.S. KIM I, & KIM, Y. KIM I S.I. 2000. Measurement of femoral neck anteversion in 3D. Part 2: 3D modelling method. *Med. Biol. Eng. Comput*, **38**, 610–616.
- IGUCHI, H., TANAKA, N., KOBAYASHI, M., NAGAYA, Y., GOTO, H., NOZAKI, M., MURAKAMI, S., HASEGAWA, S., TAWADA, K., YOSHIDA, Y., OTSUKA, T., & FETTO, J. 2010. What is anteversion, where is it located, how does it effect to the stem design and hip arthroplasty? *Journal of Bone and Joint Surgery - British Volume*, **92-B**.
- IZENMAN, A. J. 1991. Recent developments in nonparametric density estimation. *Journal of the American Statistical Association*, **86**(413), 205–224.
- JENNRICH, RI, & SAMPSON, PF. 1966. Rotation for simple loadings. *Psychometrika*, **31**(3).
- JONKERS, A CARRIERO I, & SHEFELBINE, SJ. 2010. Mechanobiological prediction of proximal femoral deformities in children with cerebral palsy. *Computer Methods in Biomechanics and Biomedical Engineering*.
- KANG, YAN, ENGELKE, KLAUS, FUCHS, CHRISTINA, & KALENDER, WILLI A. 2005. An anatomic coordinate system of the femoral neck for

References

- highly reproducible BMD measurements using 3D QCT. *Computerized Medical Imaging and Graphics*, **29**, 533–541.
- KELEMEN, A, SZEKELY, G, & GERIG, G. 1999. Elastic model-based segmentation of 3d neuroradiological data sets. *IEEE Transactions on Medical Imaging*.
- KERSNIČ, B., IGLIČ, A., KRALJ-IGLIČ, V., JACKLIČ, A., SRAKAR, F., PERNUŠ, F., & ANTOLIČ, V. 1996. Determination of the femoral and pelvic geometrical parameters that are important for the hip joint contact stress: Differences between female and male. *Pflügers Archiv European Journal of Physiology*, **43**.
- KINGSLEY, PAUL C., & OLMSTED, K. L. 1948. A STUDY TO DETERMINE THE ANGLE OF ANTEVERSION OF THE NECK OF THE FEMUR. *Journal of Bone and Joint Surgery*, **30**, 745–751.
- KNUTTGEN, HG. 1976. *Neuromuscular mechanisms for therapeutic and conditioning exercise*. University Park Press.
- KUHN, HAROLD W. 1955. The Hungarian Method for the assignment problem. *Naval Research Logistics Quarterly*, **2**, 83–97.
- LAMECKER, H., LANGE, T., & SEEBASS, M. 2002. A Statistical Shape Model for the Liver. *Proceedings of the 5th International Conference on Medical Image Computing and Computer-Assisted Intervention-Part II*, 421 – 427.
- LAMECKER, H., SEEBAB, M., HEGEA, H-C, & DEUFLHARDA, PETER. 2004. A 3D statistical shape model of the pelvic bone for segmentation. *Proceedings of SPIE*, 1341–1351.
- LEE, S, WOLBERG, G, & SHIN, SY. 1997. Scattered data interpolation with multilevel b-splines. *IEEE Transaction on Visualization and Computer Graphics*, **3**, 228–244.

References

- LEE, YEON SOO, OH, SEUNG HOON, SEON, JONG KEUN, SONG, EUN KYOO, & YOON, TAEK RIM. 2006. 3D femoral neck anteversion measurements based on the posterior femoral plane in ORTHODOC system. *Med Bio Eng Comput*, **44**, 895906.
- LEUNG, K.Y. ESTHER, & BOSCH, JOHAN G. 2007. Local Wall-Motion Classification in Echocardiograms Using Shape Models and Orthomax Rotations. *Lecture Notes in Computer Science*.
- LEYMARIE, FF, & KIMIA, BB. 2001. The Shock Scaffold for Representing 3D Shape. *Lecture Notes in Computer Science*, **2059**, 216–229.
- LORENSEN, WE, & CLINE, HE. 1987. Marching Cubes: a high resolution 3D surface reconstruction algorithm. *Computer Graphics*, **21**(4).
- LORENZ, CHRISTIAN, & KRAHNSTÖVER, NILS. 2000. Generation of Point-Based 3D Statistical Shape Models for Anatomical Objects. *Computer Vision and Image Understanding*, **17**, 175–191.
- MAHAISAVARIYA, BANCHONG, TONGDEE, KRISKRAI SITTHISERIPRATIPAND TRONGTUM, SLOTEN, ERIK L.J. BOHEZ JOS VANDER, & ORIS, PHILIP. 2002. Morphological study of the proximal femur: a new method of geometrical assessment using 3-dimensional reverse engineering. *Medical Engineering & Physics*, **24**, 617622.
- MATERIALISE. 2009. *Mimics*. <http://www.materialise.com/materialise/view/en/92458-Mimics.html>.
- MILLER, M.D. FREEMAN. 2005. *Cerebral Palsy*. Springer.
- MITTONA, D., DESCHNESB, S., LAPORTEA, S., GODBOUTB, B., BERTRANDA, S., DE GUISEB, J. A., & SKALLIA, W. 2006. 3D reconstruction of the pelvis from bi-planar radiography. *Computer Methods in Biomechanics and Biomedical Engineering*, **9**(1).

References

- MORELAND, JR, BASSETT, LW, & HANKER, GJ. 1987. Radiographic analysis of the axial alignment of the lower extremity. *The Journal of Bone and Joint Surgery*, **69**(5), 745–749.
- MORIERA, W, & WARNES, GR. 2004. *Rpy, a robust Python interface to the R Programming Language*.
- MURPHY, SB, SIMON, SR, KIJEWski, PK, WILKINSON, RH, & GRISCOM, NT. 1987. Femoral anteversion. *J Bone Joint Surg Am.*, **69**, 1169–1176.
- PARSONS, F.G. 1914. The characters of the English thigh bone. *J Anat Physiol*, **48**, 238–267.
- PEARSON, K, & BELL, J. 1919. A study of the long bone of the English thigh bone. *Drapers' Company Research Memiors, Biometric Series X ans XI, part I*.
- QUAN, W., MATUSZEWSKI, B. J., SHARK, LIK-KWAN, & AIT-BOUDAUD, D. 2009. Facial Expression Biometrics Using Statistical Shape Models. *EURASIP Journal on Advances in Signal Processing*.
- R DEVELOPMENT CORE TEAM. 2009. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.
- RANJAN, VISHWA, & FOURNIER, ALAIN. 1995. *Shape Transformations Using Union of Spheres*. Tech. rept. Vancouver, BC, Canada, Canada.
- ROBERTSON, DD, YUAN, J, & BIGLIANI, LU. 2000. Three-dimensional analysis of the proximal part of the humerus: relevance to arthroplasty. *J Bone and Joint Surg*, **82-A**(11), 1594–1602.
- RUECKERT, D., SONODA, L. I., HAYES, C., HILL, D. L. G., LEACH, M. O., & HAWKES, D. J. 1999. Nonrigid Registration Using Free-Form

References

- Deformations: Application to Breast MR Images. *IEEE TRANSACTIONS ON MEDICAL IMAGING*, **18**.
- RUECKERT, D., CHANDRASHEKARA, R., ALJABAR, P., BHATIA, K.K., BOARDMAN, J.P., SRINIVASAN, L., RUTHERFORD, M.A., DYET, L.E., & EDWARDS, A.D. 2006. Quantification of Growth and Motion Using Non-rigid Registration. *Lecture Notes in Computer Science*.
- SCHEUER, L, & BLACK, S. 2004. *The Juvenile Skeleton*. Elsevier.
- SCHRÖDER, J, & MARTI, RK. 2001. Simultaneous bilateral femoral neck fractures: case report. *Swiss Surg*, **7**(5).
- SCHROEDER, W. 2003. *The Visualization Toolkit, 3rd Edition*. Kitesre Inc.
- SCHUENKE, M, SCHUENKE, E., & SCHUMAKER, U. 2005. *Atlas of Anatomy, General Anatomy and Musculoskeletal System*. Thieme, Stuttgart - New York.
- SCHWARTSMANN, CR, OLIVEIRA, GK, OLIVEIRA, RK, CARPES, D, & WERLANG, PM. 2001. Simultaneous bilateral femoral neck fracture. *Acta Orthop Bras*, **9**(4).
- SHEFELBINE, SJ, TARDIEU, C, & CARTER., DR. 2002. Development of the femoral bicondylar angle in hominid bipedalism. *Bone*, **30**(5), 765–70.
- SHENTON, M. E., GERIGB, G., MCCARLEYA, R. W., SZEKELYC, G., & KIKINIS, R. 2002. Amygdalahippocampal shape differences in schizophrenia: the application of 3D shape models to volumetric MR data. *Psychiatry Research: Neuroimaging*, **115**(1-2), 15–35.
- SOLOMON, L, WARWICK, D, & NAYAGAM, S. 2005. *Apley's Concise System of Orthopaedics and Fractures*. Hodder Arnold.

References

- STEGMANN, MIKKEL B., SJOSTRAND, KARL, & LARSEN, RASMUS. 2006. Sparse Modeling of Landmark and Texture Variability using the Orthomax Criterion. *Progress in biomedical optics and imaging*, **7**.
- STEPHEN, D JG, KREDER, H J, SCHEMITSCH, E H, CONLAN, L B, WILD, L, & MCKEE, M D. 2002. Femoral Intramedullary Nailing: Comparison of Fracture-Table and Manual Traction : A Prospective, Randomized Study. *J Bone Joint Surg Am*, **84**, 1514–1521.
- STYNER, MA. 2001. *Combined Boundary-Medial Shape Description of Variable Biological Objects*. Tech. rept. UNC Chapel Hill.
- SUGANO, N., NOBLE, P. C., & KAMARIC, E. 1998a. A Comparison of Alternative Methods of Measuring Femoral Anteversion. *Journal of Computer Assisted Tomography*, **22**(4), 610–614.
- SUGANO, NOBUHIKO, NOBLE, PHILIP C., KAMARIC, EMIR, JOSEPH K. SALAMA, TAKAHIRO OCHI, & S.TULLOS, HUGH. 1998b. The morphology of the femur in developmental dysplasia of the hip. *Journal of Bone and Joint Surgery*, **80-B**(4), 711–719.
- TAM, ROGER C., & FOURNIER, ALAIN. 1998. Image interpolation using unions of spheres. *The Visual Computer*, 401–414.
- TAN, PN, STEINBACH, M, & KUMAR, V. 2006. *Introduction to Data Mining*. Addison-Wesley. Chap. 8.
- THE MATHWORKS. 2009. *Matlab*. Tech. rept. , Natick, MA.
- TOMPKINS, GS, HENDERSON, RC, & PETERSON, HD. 1990. Bilateral simultaneous fractures of the femoral neck: case report. *J Trauma*, **30**(11), 1415–1416.
- TURK, M, & PENTLAND, A. 1991. Eigenfaces for Recognition. *Journal of Cognitive Neuroscience*, **3**(1).

References

- VAN ROSSUM, G. 1995 (May). *Python tutorial*. Tech. rept. CS-R9526. Centrum voor Wiskunde en Informatica (CWI), Amsterdam.
- VILLANUEVA-MARTÍNEZ, M, RÍOS-LUNA, A, & VILLAMOR-PÉREZ, Á. 2009. A Simple Technique for Alignment in Total Hip Resurfacing Arthroplasty: Technical Note and Preliminary Report. *Hospital for Special Surgery Journal*, **5**, 129136.
- VILLEMURE, ISABELLE, & STOKES, IAN A.F. 2009. Growth plate mechanics and mechanobiology. A survey of present understanding. *Journal of Biomechanics*.
- WEINER, D.S., COOK, A.J., HOYT, W.A., & ORAVEC, C.E. 1978. Computed Tomography in the Measurement of the Femoral Neck. *Orthopedics*, **1**, 299–306.
- YANG, Y., BULL, AMJ, RUECKERT, D, & HILL, A. 2006. 3D Statistical Shape Modeling of Long Bones. *Lecture Notes in Computer Science*.
- YANG, YUHUI. 2008. *Shape Modelling of Bones, Application to the Primate Shoulder*. Ph.D. thesis, Imperial College London.