

KNEE FLUOROSCOPY

Samantha Ly

Bachelor of Engineering
Majoring in Mechatronics



Department of Engineering
Macquarie University

6 November 2017

Supervisor: Dr Danè Turner



ACKNOWLEDGMENTS

I would like to acknowledge a number of people who have helped me throughout this thesis and my undergraduate degree. Firstly, my supervisor, Dr Danè Turner who has guided and inspired me throughout this project. Secondly, the Department of Engineering at Macquarie University, who have helped me throughout my undergraduate degree and allow me to come this far.



STATEMENT OF CANDIDATE

I, Samantha Ly, declare that this report, submitted as part of the requirement for the award of Bachelor of Engineering in the Department of Engineering, Macquarie University, is entirely my own work unless otherwise referenced or acknowledged. This document has not been submitted for qualification or assessment at any academic institution.

Student's Name: Samantha Ly

Student's Signature: 

Date: 06/11/2017



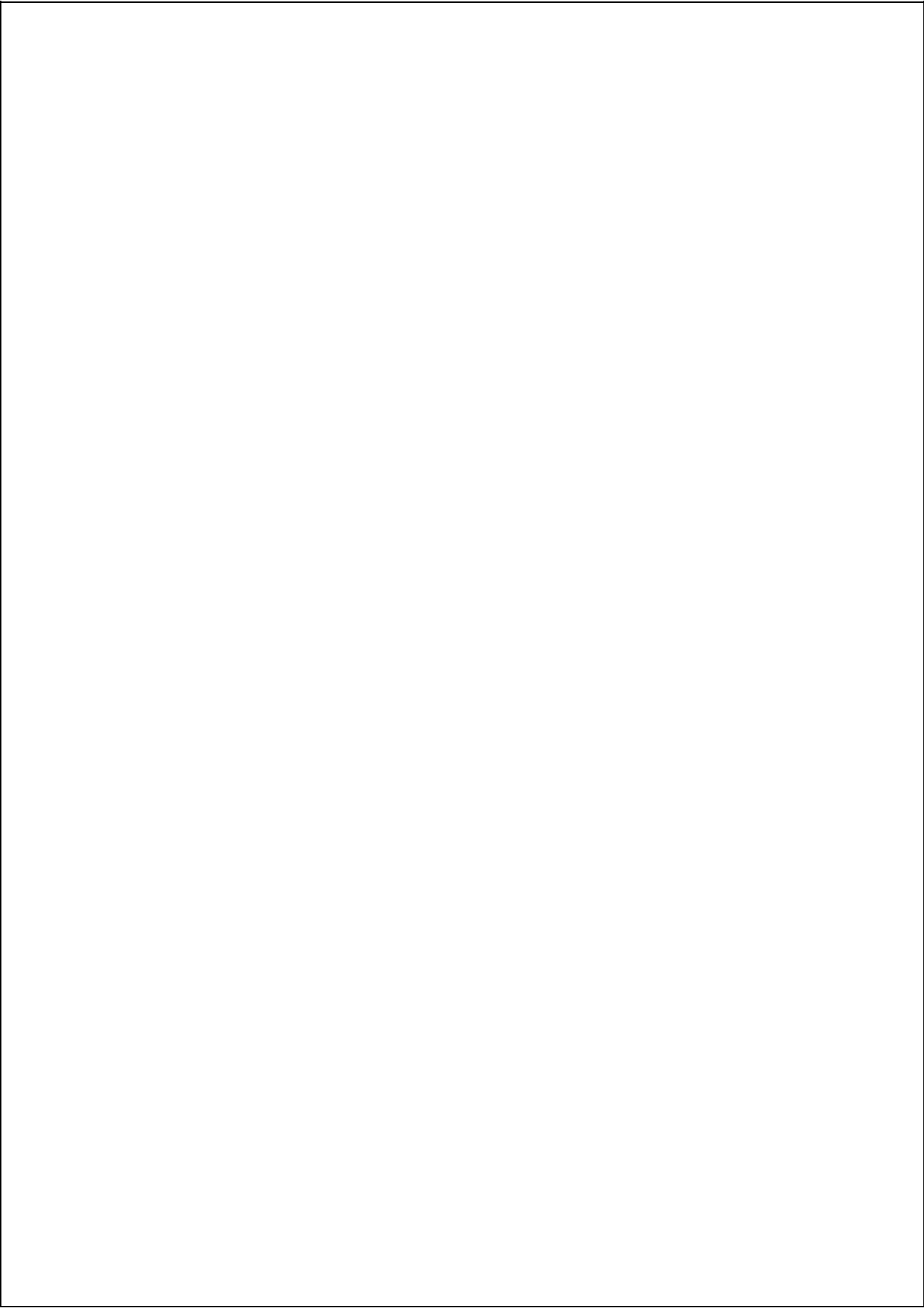
ABSTRACT

Selecting an appropriate method when describing the motion of the knee in 3D space can be challenging as there is no universal or standard definition currently utilised in biomechanical studies. The Joint Coordinate System (JCS), first proposed by Grood and Suntay, is widely used by various authors to define the relative position between two rigid bodies as the definitions when establishing the system is easily comprehended. Throughout this thesis, a program will be developed in Matlab to enable to the automation of the joint coordinate system on the femur and tibia by defining the anatomical landmarks. The JCS will be redefined and used as a basis to implement a slightly different coordinate system. This automated program will allow simplicity when manipulating the orientation and will enable the data to be presented in different clinical systems.



Contents

Acknowledgments	iii
Abstract	vii
Table of Contents	ix
List of Figures	xi
List of Tables	xiii
Abbreviations	xv
1 Introduction	1
2 Background and Related Work	3
2.1 The six degrees of freedom	3
2.2 The joint coordinate system	5
2.3 Applying JCS to the knee	7
3 Methodology	9
3.1 Image segmentation	9
3.2 Automated knee coordinate system using Matlab	12
3.2.1 Coordinate system for the femur	12
3.2.2 Coordinate system for the tibia	16
4 Results and Analysis	25
4.1 The femur's coordinate system	25
4.2 The tibia's coordinate system	31
5 Conclusions and Future Work	41
5.1 Conclusions	41
5.2 Future Work	42
Bibliography	43



List of Figures

2.1	The six degrees of freedom of the knee	3
2.2	A generalised joint coordinate system between two rigid bodies	5
2.3	The tibia and femur's Cartesian coordinate system	7
3.1	Original image of the femur to below knee	10
3.2	Image of the femur to below knee after filters have been applied	10
3.3	STL model of the femur	11
3.4	STL model of the tibia	11
3.5	User selection of the femoral head	12
3.6	User selection of the distal end of the femur	14
3.7	User selection of the two distal ends of the tibia	17
3.8	User selection of the distal end of tibia	19
3.9	User selection of the left side of the proximal end of the tibia	20
3.10	User removing unbrushed area of tibia model	22
3.11	User selection on the superior view of the tibia	23
4.1	Centre of the femoral head calculated by Matlab	26
4.2	Lateral and medial epicondyles calculated by Matlab	27
4.3	Calculated coordinate system for the femur	29
4.4	Redefined coordinate system for the femur	30
4.5	The most medial and distal points and the centre of ankle	32
4.6	The two intercondylar eminences with their midpoint	34
4.7	Edited circle on the square selection of the superior view	35
4.8	Inaccurate determination of the centre of the plateaus	37
4.9	Centre of each plateau	38
4.10	Superior view of the line connecting the plateaus and the z axis	38
4.11	Redefined coordinate system for the tibia	39
4.12	x axis of the tibia	40



List of Tables

4.1	Results from finding the centre of the femoral head	25
4.2	Coordinates of the epicondyles and most distal point	27
4.3	Results of the femoral axes as arrays	29
4.4	Results for defining the centre of the ankle	31
4.5	Results from finding the midpoint of the intercondylar eminences	33
4.6	Results from finding the centre of the plateau	36
4.7	Results of the axes as an array	40



Abbreviations

JCS	Joint Coordinate System
MRI	Magnetic Resonance Imaging
STL	Stereolithography



Chapter 1

Introduction

Joints are an essential element of the body as they allow movement of the body and are utilised to sustain weight. One in three people encounter joint pain or discomfort and an immense number of these are related to the knee. Therefore, understanding the biology of the knee and the kinematics of this joint is cardinal in the diagnosis of knee disorders and how treatments should be undertaken. To apprehend the three-dimensional motion behind the knee, three rotational and three translational components are utilised. These six degrees of freedom enable the establishment of the joint coordinate system (JCS) to comprehensively describe the relative motion between the tibia and femur.

This thesis will research and examine the effects of different coordinate systems on knee fluoroscopy results. To describe the kinematics between the femur and tibia in three-dimensions, JCS will be applied. Previous students have begun the development of an automated knee coordinate system using Matlab and the aim of this thesis is to refine and further develop this program to automatically determine the Cartesian coordinate system of the femur and tibia and thus, automate JCS on the knee. Automating the joint coordinate system is important as it is currently undertaken manually and therefore, requires more time to be defined on the knee. Additionally, by automating JCS, the results will be more reliable when acquiring the bony landmarks as it is more consistent than defining the system manually. Furthermore, defining a coordinate system to the knee is fundamental when describing fluoroscopy data.

Chapter 2

Background and Related Work

2.1 The six degrees of freedom

In order to comprehend three-dimensional knee kinematics, they must be broken down into six motion components. These elements can be described as measurements of three translational and three rotational components. These include flexion and extension, adduction and abduction, internal and external rotation, medial and lateral shift, compression and distraction and anterior and posterior drawer.

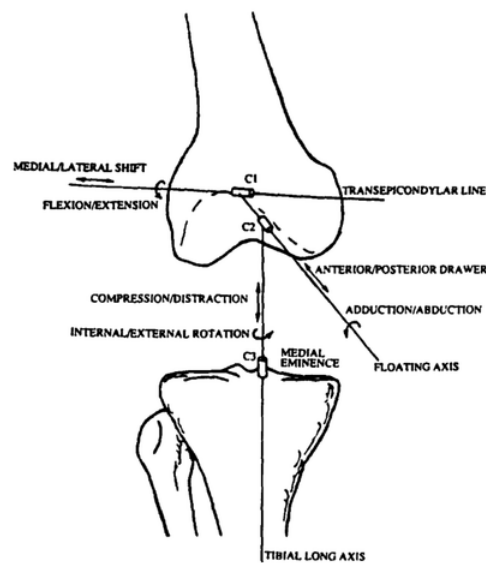


Figure 2.1: The six degrees of freedom of the knee

The rotational element of flexion and extension takes place about the transepicondylar line in the mediolateral direction and the translational component that describes this axis is the medial and lateral shift. The components that describe the floating axis which travels through the knee in the anterior posterior direction is defined by abduction and adduction rotation and anterior and posterior drawer [3]. The tibial long axis defines the axis along the tibia in the inferior superior direction and the components that describe this axis are internal and external rotation and compression and distraction. This is demonstrated in figure 2.1. These definitions of the six degrees of freedom in the knee allows the joint coordinate system to be established in order to thoroughly express the relative motion between the tibia and femur.

2.2 The joint coordinate system

Grood and Suntay first established JCS in 1983 to explicate the three-dimensional motion of the knee. Before the development of JCS, the majority of research into joint motion was conveyed only in relative rotational motion between the bodies [2]. The joint motion described by JCS is defined by determining the relative position between two rigid bodies such as the femur and tibia. First, JCS will be described in geometric terms, and then when the coordinate system is executed to the knee, the correspondent equations will be presented. Figure 2.2 demonstrates how the geometry of body A and body B is specified by a Cartesian coordinate system, with the origins positioned at O_A and O_B .

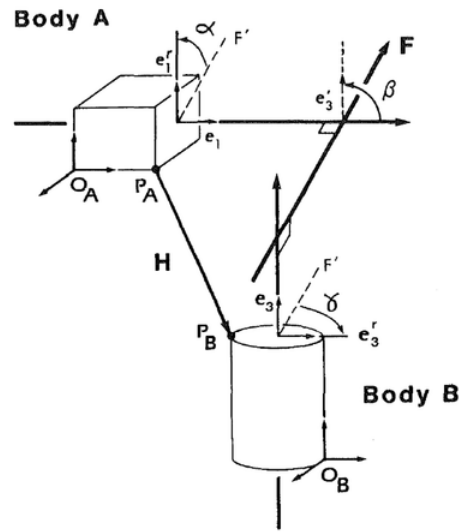


Figure 2.2: A generalised joint coordinate system between two rigid bodies

First, the angular position and the equivalent rotational motion between the two bodies will be evaluated. When determining the angular position in three dimensional motion, three independent angles are distinguished, therefore, three spatial axes for rotational motion must be established. Alternatively, the three planes could be defined as being orthogonal to the rotational axes. These rotation axes that compose the joint coordinate system can be seen in figure 2.2. e_1 , e_2 and e_3 specify the non-perpendicular unit base vectors which describes the axes [4]. The axes that are embedded each of the bodies are called body fixed axes, in which can be expressed by relative motion. The direction of these axes can be denoted by the unit base vectors e_1 for body A and in body B, e_3 . The movement of the fixed axes embedded in the two bodies are dependent on the two bodies, thus, their spatial relationship adjusts with motion. The final axis, F , is also referred to

as the floating axis as this axis is not fixed in either body A or body B but moves relative to both. This axis is implemented in such a way that it is the common perpendicular to \mathbf{e}_1 and \mathbf{e}_3 in the two bodies. Hence, its direction can be calculated by taking the cross product of the two unit base vectors, \mathbf{e}_1 and \mathbf{e}_3 as demonstrated in equation 2.1 [1].

$$\mathbf{e}_2 = \frac{\mathbf{e}_1 \times \mathbf{e}_3}{|\mathbf{e}_1 \times \mathbf{e}_3|} \quad (2.1)$$

α and γ represents the magnitude of the rotation that is structured between the floating axis and the reference line of the bodies and they determine the relative rotation of spin taken about the bodies axis while the other body remains fixed. \mathbf{e}_1' and \mathbf{e}_3' denote the unit vectors that represent the sense and reference lines of each body. The orientations of \mathbf{e}_1' and \mathbf{e}_3' are not definite and can be selected according to convenience during the application. β describes the relative rotation that is located at the floating axis. This describes the angle between the two fixed body axes and can be calculated using equation 2.2 [1].

$$\cos\beta = \mathbf{e}_1 \cdot \mathbf{e}_3 \quad (2.2)$$

The angular coordinates of α , β and γ allows a conventional geometric explanation of Euler angles. The alignment of the floating axis depends on \mathbf{e}_2 and will consistently be parallel to the line of nodes and is concomitant with it when all of the three axes intersect.

The joint's translations are elucidated by the relative positions of P_A and P_B , the reference points of each body as shown in figure 2.2. \mathbf{H} defines the vector that distinguishes the relative position of the reference point for each body and connects body A and body B at these points. The translation vector's elements are defined by the direction of the axes and the elements of it's magnitude is dependent on the position of P_A and P_B in each body. This characteristic is evident in all methods of identifying translations and dictates the necessity for a knowledgeable rationale when selecting P_A and P_B in the bodies [1]. This method of defining JCS is also utilised in [5].

2.3 Applying JCS to the knee

When applying JCS to the knee or to different joints of the body, it is imperative that the following things are established:

- The fixed coordinate system in the femur and tibia that is used to define its structure
- The fixed body axes and the reference axis of JCS employed to elucidate the relative motion between the tibia and femur
- The placement of the translation reference point in the two bodies

It is expedient to define the Cartesian system for both the femur and tibia such that the axes correlate with joint coordinate system's reference axis and the fixed body axes to establish the origin of the Cartesian system. The anatomical landmarks in the tibia and femur that is perceptible on bi-planar X-rays are the fundamentals when establishing Grood and Suntay's joint coordinate system [1]. This technique of identifying the bony landmarks to apply JCS is evident in [6] and [7]. However, these papers apply the coordinate system to other joints in the human body such as the ankle or elbow.

The Cartesian coordinate system in the femur is defined as X , Y and Z whereas the coordinate system in femur will be denoted as x , y and z . The femoral base vectors are represented by \mathbf{I} , \mathbf{J} and \mathbf{K} with respect the axes of the Cartesian coordinate system, whereas the tibial base vectors are denoted by the vectors \mathbf{i} , \mathbf{j} and \mathbf{k} .

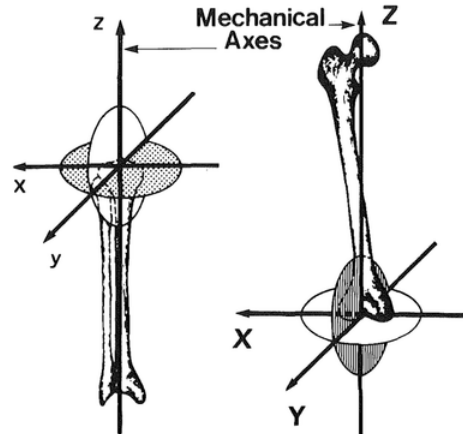


Figure 2.3: The tibia and femur's Cartesian coordinate system

When applying JCS to the knee, the rotation of the mechanical z axis is described as internal and external rotation. Thus, $\mathbf{e}_3 = \mathbf{k}$. \mathbf{e}_3' defines the reference direction and is

positioned anteriorly and is relative to the y axis of the tibia. Therefore, $\mathbf{e}_3' = \mathbf{j}$. The fixed body axis in the tibia is denoted as z and is defined such that it passes through the midpoint of the lateral and medial intercondylar eminences on the proximal end of the tibia and along the centre of the ankle on the distal end. The tibial y axis is defined by taking the cross product of the mechanical fixed axis with a line connecting the centre of the plateaus on the superior view. When the z and y axis have been established, the x axis can be determined by completing the right hand coordinate system [1].

The femoral fixed body axis is chosen in a way such that the rotations around this axis is relative to flexion and extension. The Z axis, or femoral mechanical axis is determined by connecting the centre of the femoral head to the most distal point on the posterior surface of the femur at the midpoint of the lateral and medial condyles. The femoral mechanical axis is within the frontal plane. It is aligned such that the most posterior points on the medial and lateral condyles are of equal distance from the plane. The Y axis, or the femoral anterior is determined by taking the cross product of the Z axis and the line that passes through the medial and lateral condyles on the posterior surface. By establishing the femoral mechanical axis and the femoral frontal plane, the flexion axis in the femur can be oriented. The direction of this axis can be determined as the cross product of the Z axis and the unit base vectors of the femoral anterior-posterior axis [1]. The fixed body axis in the femur, \mathbf{e}_1 , correlates to the femoral X axis that has a base vector \mathbf{I} . The reference axis is appointed in the anterior direction, thus, $\mathbf{e}_1^r = \mathbf{J}$.

Flexion and extension rotation can be defined about the fixed axis in the femur whilst internal and external rotation is about the fixed axis in the tibia and the rotational motion about the floating axis can be described by adduction and abduction which is demonstrated in figure 2.1 [3].

Chapter 3

Methodology

3.1 Image segmentation

Amira is a software that allows for the visualisation of 3D data and has the capabilities to process and analyse it. Using Amira, a patient's magnetic resonance imaging (MRI) files were stacked together in order to attain 3D representations of the pelvis, femur to below the knee, upper knee to mid shank and the mid shank to the ankles. Each of these sections contained an additional four files, in phase, out of phase, fat and water. A comparison of these four files was completed to determine which file would be the most suitable for image segmentation in order for it to be completed with ease. For each of the components of the leg, the in phase file was selected as these files were of higher quality.

Images can be complemented and highlighted by applying various filters such as smoothing, contrast or noise reduction. After experimenting with numerous filters, the following presented the best results:

1. Equalise at contrast 3
2. Edge-preserving smoothing – uses a diffusion filter preserving edge
3. Non-local means noise reduction – very effective on noisy data whilst still preserving the edges
4. Unsharp masking – sharpens the details and aids in reinforcing the contrast at the edges [8]

Once the filtering was completed, figure 3.2, a threshold of 150 was applied to exterior whilst each voxel with an equal or greater value was assigned to bone thus, allowing the differentiation between the exterior and bone. Using thresholding significantly decreases the requirement for manual interaction, however, it was still necessary to review and polish each slice to ensure an accurate representation. When the segmentation of all four files were completed, they could be combined and saved as two stereolithography (STL) files to attain 3D models of the femur and tibia as shown in figures 3.3 and 3.4.

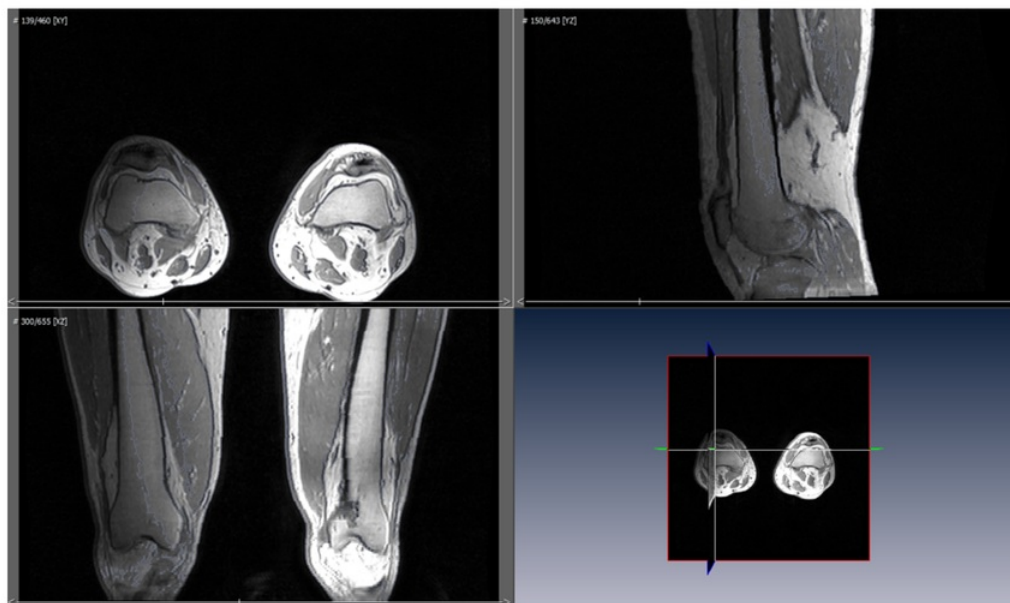


Figure 3.1: Original image of the femur to below knee

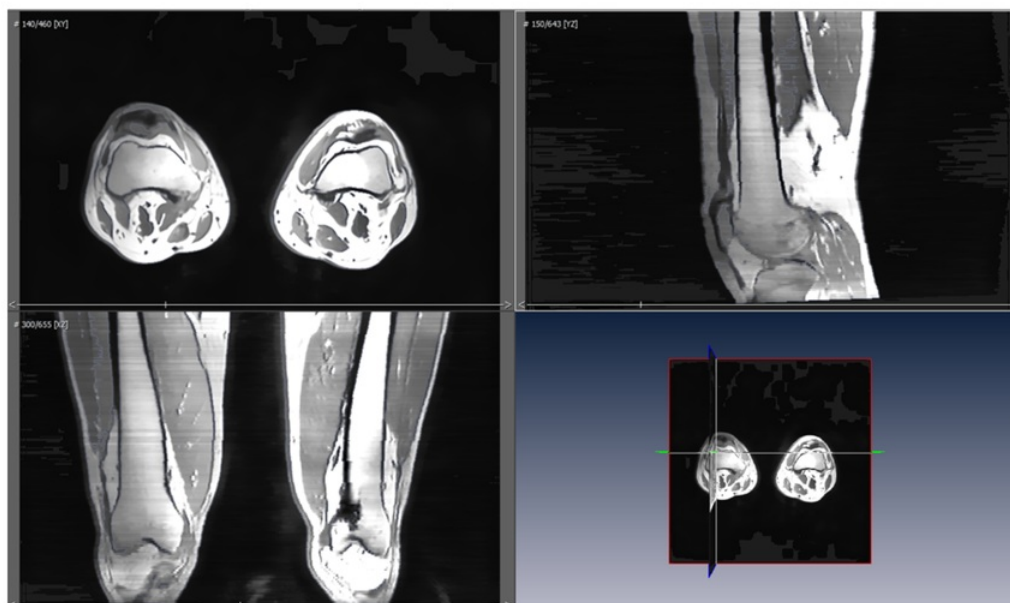


Figure 3.2: Image of the femur to below knee after filters have been applied



Figure 3.3: STL model of the femur



Figure 3.4: STL model of the tibia

3.2 Automated knee coordinate system using Matlab

In order to define the joint coordinate system in the knee, a fixed Cartesian coordinate system in the femur and tibia must be established as explained in section 2.3. Previous students had begun the development of a Matlab program to allow the automatic generation of the coordinate system. Using this program, the centre of the femoral head in the femur and the medial and lateral malleoli in the tibia could be determined. This program will be refined to automate the calculation of these coordinates in order to minimise manual interaction from the user and further developed to obtain all the other points needed to apply the joint coordinate system to the knee. The STL files attained from image segmentation in Amira were utilised to provide the 3D model of the femur and tibia.

3.2.1 Coordinate system for the femur

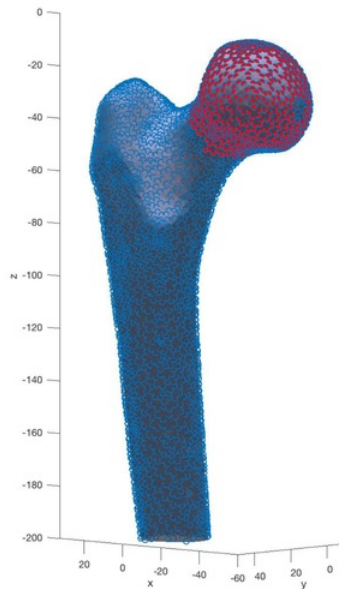


Figure 3.5: User selection of the femoral head

The mechanical axis, Z , of the femur is established by connecting the centre of the femoral head to the midpoint of the lateral and medial epicondyles at the most distal

point on the posterior surface. Ensuring the correct file name for the femur is entered where specified, when the Matlab program is run, a figure window will appear with a 3D model of the femur. The user can then use the brush tool to select the faces that depicts the curvature of the femoral head whilst avoiding the fovea as shown in figure 3.5. The selected area of the femur represents the faces that will be employed in the judgement section to find the centre of the femoral head. Ensuring that all tools are deselected, the user can right click the selected area and create a variable. Once the variable has been created, the program can be run by pressing enter in the command window.

When the user selects the faces on the model using the brush, each point that is selected is the centre point of a face on the model in which a normal is taken from. Each normal that is taken from each face is compared to one another. The shortest possible distance between the two normals is determined by locating the two unique points in which the two lines are the closest, a method developed by Dan Sunday [9]. Once this distance is calculated the midpoints are stored and averaged, which determines the centre point of the femoral head. Using this point as the centre, a sphere is then fitted to the femoral head. When the program has finished running, the coordinates of the centre of the femoral head, the number of faces selected and the radius of the fitted sphere will be displayed in the command window. Along with these results, a full femur will appear in the figure window with the centre of the femoral head depicted by a red dot.

To establish the femur's mechanical axis, the centre of the femoral head must be connected to the midpoint of the lateral and medial epicondyles at the most distal point on the posterior surface. To find the midpoint of the two epicondyles, the user will use the brush tool to select the distal end of the femur as shown in figure 3.6. The program will analyse the selected data and extract the coordinates with the maximum and minimum values for x . The following code was used to do so.

```

1 % defining the selected data (testFaces) as X, Y, Z
2 XData = testFaces(:,1);
3 YData = testFaces(:,2);
4 ZData = testFaces(:,3);
5
6 % finds the maximum x value to get the lateral epicondyle
7 [~,i] = max(XData(:));
8 lateralEpicondyle = [X(i), Y(i), Z(i)];
9
10 % finds the minimum x value to get the medial epicondyle
11 [~,i] = min(XData(:));
12 medialEpicondyle = [X(i), Y(i), Z(i)];

```

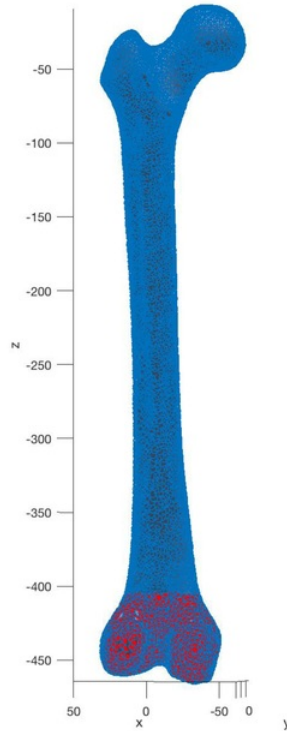


Figure 3.6: User selection of the distal end of the femur

When the program has finished running, the coordinates of the lateral and medial epicondyle will be displayed in the command window and a femur with these two points depicted as a red dot will appear in the figure window. The next step is to find the origin of the femur's coordinate system, which is the most distal point on the posterior surface. This point is also connected to the centre of the femoral head to define the mechanical Z axis. To determine the origin, the program gathers all the points midway between the lateral and medial epicondyles. For each face on the model, a vector \mathbf{A} is found in the direction from the centre point between the two epicondyles and the centre point of the face. The dot product is utilised to determine the projected distance of \mathbf{A} on \mathbf{B} where \mathbf{B} is the unit vector in the same direction as the two epicondyles. This is described by Equation 3.1. When the projected distance is zero, the points are midway between the lateral and medial epicondyle.

$$A \cdot B = |A||B|\cos\theta \quad (3.1)$$

Each midpoint is analysed and the distance between these points and the centre of femoral head is calculated. The greatest distance will be saved and thus, provides us with the most distal point as shown in the code below.

```

1  % gather all the points midway between the epicondyles
2
3  allMidwayPoints = zeros(10000,3);
4  totalMidwayPoints = 0;
5  midway = (medialEpicondyle + lateralEpicondyle)/2;
6  for i = 1:size(faceCentreVertices,1)
7      B = [midway(1) - medialEpicondyle(1), midway(2) - ...
            medialEpicondyle(2),midway(3) - medialEpicondyle(3)];
8      unitB = B/norm(B);
9      A = [midway(1) - faceCentreVertices(i,1), midway(2) - ...
            faceCentreVertices(i,2),midway(3) - faceCentreVertices(i,3)];
10
11     distanceMedial = abs(dot(A,unitB));
12     if(distanceMedial ≤ 1)
13         allMidwayPoints(totalMidwayPoints+1,:) = faceCentreVertices(i,:);
14         totalMidwayPoints = totalMidwayPoints+1;
15     end
16 end
17
18 % remove null midpoints
19
20 allMidwayPoints(all(allMidwayPoints==0,2),:)=[];
21
22 %go through each pt and find its distance to the femoral head. save ...
    the greatest distance
23 greatestDistance = 0;
24 femurCentrePoint = [0 0 0];
25 for i = 1:size(allMidwayPoints,1)
26     distance = sqrt((centreOfHead(1) - allMidwayPoints(i,1))^2 + ...
                     (centreOfHead(2) - allMidwayPoints(i,2))^2 + (centreOfHead(3) ...
                     - allMidwayPoints(i,3))^2);
27     if distance > greatestDistance
28         greatestDistance = distance;
29         femurCentrePoint = allMidwayPoints(i,:);
30     end
31 end

```

After all of these landmarks on the femur have been defined, the mechanical Z axis and the X axis can be established. Using the two coordinates, the program will plot the line that describes the axis. The Y axis is then determined by taking the cross product of the Z axis and the X axis. In order to compute the cross product using Matlab's in

built function, the two axes must be converted into arrays. This is demonstrated in the code below.

```

1 % plot mechanical z axis
2 pts = [centreOfHead; femurCentrePoint];
3 zAxis = plot3(pts(:,1), pts(:,2), pts(:,3), 'r', 'LineWidth', 2);
4 % turn it into an array
5 zar=[pts(:,1) pts(:,2) pts(:,3)];
6
7 % plot x axis
8 pts = [lateralEpicondyle; medialEpicondyle];
9 xAxis = plot3(pts(:,1), pts(:,2), pts(:,3), 'r', 'LineWidth', 2);
10 % turn it into an array
11 xar=[pts(:,1) pts(:,2) pts(:,3)];
12
13 % plot y axis
14 yAxis = cross(zar, xar);
15 plot3(yAxis(:,1), yAxis(:,2), yAxis(:,3), 'r', 'LineWidth', 2)

```

3.2.2 Coordinate system for the tibia

The fixed body axis in the tibia is denoted as z and is defined such that it passes through the midpoint of the lateral and medial intercondylar eminences on the proximal end of the tibia and along the centre of the ankle on the distal end. To compute the centre of the ankle, the coordinates of the most medial point of the medial malleolus on the tibia and the most distal point of the lateral malleolus on the fibula must be located. The midpoint of these two coordinates present the coordinates for the ankle's centre. First, the most medial point of the medial malleolus will be determined. Ensuring that the correct file name for the tibia is entered where necessary, run the program thus, enabling a figure window displaying the tibia to open. The user can then use the brush selection tool to select the two ends of the shaft of the tibia as shown in figure 3.7 and thus, create a variable with this highlighted area. Using these points that have been selected, a line of best fit will be created and the program can be run from the command window.

Whilst the program is being executed, all the distances down the central axis is determined through the use of the dot product and is stored. The program will then determine the maximum distance and the centre point of the face in which it corresponds to as shown on the next page. Once the program has finished running, the most medial point, axis normal and the axis point will be displayed in the command window. Along with this data, a new tibia will be displayed in the figure window displaying the axis normal, axis point and the most medial point of the medial malleolus depicted by a red point.

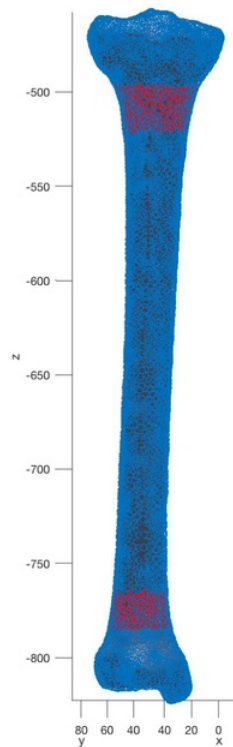


Figure 3.7: User selection of the two distal ends of the tibia

```

1 % find the most medial point
2
3 % define the direction of the distal end (-1 is up, +1 is down)
4 direction = +1;
5
6 distances = zeros(size(faceCentreVertices,1),1);
7 for i = 1:length(distances) % go through each face(point) in the ...
    entire fibula
8     %based on A.B = |A||B|cos(theta)
9     A = [faceCentreVertices(i,1) - axisPoint(1), ...
          faceCentreVertices(i,2) - axisPoint(2), faceCentreVertices(i,3) ...
          - axisPoint(3)];
10    distances(i) = dot(A,axisNormal)*-direction; %|A|cos(theta). ...
        diretion lets it know if above or below
11 end

```

```

12
13 %find location of the greatest distance
14 mostMedialPoints = faceCentreVertices(distances==max(distances),:);
15
16 %average the points incase there are two/three equally distal points
17 if(size(mostMedialPoints,1)==1)
18     mostMedialPoint = mostMedialPoints;
19 else
20     mostMedialPoint = sum(mostMedialPoints)/size(mostMedialPoints,1);
21 end

```

As a STL model of the fibula is currently unavailable, the tibia will be utilised to define the most distal point. In order to determine the most distal point of the lateral malleolus, run the program and similar to the previous section, a tibia will appear. The user can use the brush tool to select the distal end of the tibia, ensuring that the most medial point is also selected as shown in figure 3.8 and create a variable with the highlighted data. When the program is executed, the vector **B** will be found, a vector that is from the most distal point to the point on the axis that is closest to the most distal point. The program will find all the distances of all the points along **B** and save the greatest distance as shown in the code below. The max distance is determined in line 9 of the code and defines the most distal point. This is based on the Equation 3.1. A new tibia will appear in the figure window with the red dot representing the most distal point of the lateral malleolus of the tibia. Thus, the centre of the ankle can be determined by taking the midpoint of the most medial point of the medial malleolus and the most distal point of the lateral malleolus.

```

1 B = [N(1)-mostMedialPoint(1), N(2)-mostMedialPoint(2), ...
      N(3)-mostMedialPoint(3)]; % it is the vector going from O to N
2 unitB = B/norm(B);
3 % Find all the distances along B of points. Save the greatest distance
4 maxDistance = 0;
5
6 for i = 1:size(testFaces,1) % go through each face(point) in the ...
    entire fibula
7     %based on A.B = |A||B|cos(theta)
8     A = [mostMedialPoint(1) - testFaces(i,1), mostMedialPoint(2) - ...
          testFaces(i,2),mostMedialPoint(3) - testFaces(i,3)];
9     distance = abs(dot(A,unitB)); %|A|cos(theta). diretion lets it ...
        know if above or below
10
11     if distance > maxDistance
12         maxDistance = distance;
13         DistalPoint = testFaces(i,:);
14     end
15 end

```

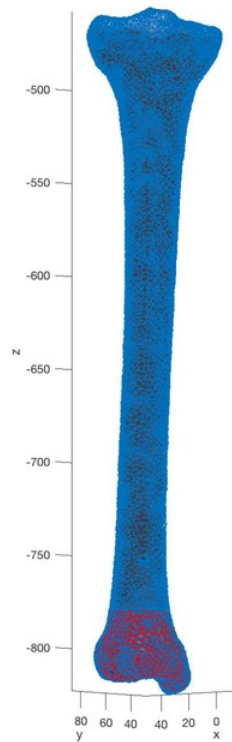



Figure 3.8: User selection of the distal end of tibia

The next step when establishing the tibia's fixed body axis is to determine the midpoint of the two intercondylar eminences. To determine the lateral intercondylar tubercle, the user will select the left side of the proximal end of the tibia, ensuring that the medial intercondylar tubercle is not selected as shown in figure 3.9. Using a similar method to find the most medial point of the medial malleolus, the lateral intercondylar tubercle was determined. As the program is being executed, all the distances down the central axis is determined through the use of the dot product and stored. The program will determine the maximum distance and the centre point of the face in which it corresponds to and thus, determines the coordinates for the lateral intercondylar tubercle which is displayed on a new tibia.

Similar to finding the lateral intercondylar tubercle, the user will select the right side of the proximal end of the tibia, to find the medial intercondylar tubercle whilst ensuring

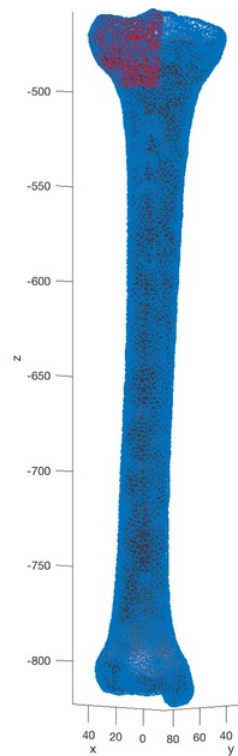


Figure 3.9: User selection of the left side of the proximal end of the tibia

that the lateral tubercle is not selected. Right click the selected data and select paste data into command line. All the coordinates of the selected data will be displayed in Matlab's command window. The medial intercondylar tubercle is defined as the coordinate with the largest value for z as it is the highest point of the selected data. When this coordinate is located, the user will have to manually enter it into the program as shown in line 2 of the code below. Now that the coordinates of the two intercondylar eminences have been attained, the midpoint can be determined as shown in line 12 of the code below and thus, the mechanical axis can be established.


```

1 %ENTER COORDINATES OF MEDIAL INTERCONDYLAR TUBERCLE HERE
2 medialTubercle = [10.4530232747396, 59.3087692260742, -458.721160888672];
3
4 %plot the two intercondylar eminences
5 scatter3(lateralTubercle(1), lateralTubercle(2), lateralTubercle(3), ...
   'r', 'filled');
6 scatter3(medialTubercle(1), medialTubercle(2), medialTubercle(3), ...
   'r', 'filled');
7
8 disp(['Medial intercondylar tubercle (of intercondylar eminence):', ...
   mat2str(medialTubercle)])
9
10 %midpoint of the two eminences
11
12 eminenceMidpoint = (medialTubercle + lateralTubercle)/2;
13 scatter3(eminenceMidpoint(1), eminenceMidpoint(2), ...
   eminenceMidpoint(3), 'g', 'filled');

```

The tibial y axis is defined by taking the cross product of the mechanical fixed axis with a line connecting the centre of the plateaus on the superior view. To locate the centre of the plateau on the tibia, a circle needs to be fit on both sides. The centre of these two circles dictates the points used for the line that will allow the determination of the y axis. Using the brush tool, the user can select the proximal end of the model of the tibia, right click a point and select remove unbrushed to obtain a model as shown in figure 3.10. By doing so, when the program is run, the faces through the whole tibia will not have to be tested and only the surface of the plateau will be tested thus, minimising the time required for the program to be executed. To select the relevant faces to be tested, go to the superior view of the tibia. Using the brush, the user will make a square selection on the medial side ensuring that three of the corners are on the perimeter and that the sides are equal in length as shown in figure 3.11.

Each point that is selected on the model is the centre point of a face in which a normal is taken from. The normals are then analysed and compared against one another and the shortest possible distance between two normals is determined similar to the methods used when calculating the centre of the femoral head in Section 3.2.1. When this distance has been calculated, the midpoints are stored and the mean is found, thus, determining the centre point of the selected area. The coordinates for the centre of the plateau on the medial side will be displayed in the command window and depicted by a red dot on the tibia model. This process is then repeated for the lateral side and thus, the two points that defines the line used to calculate the y axis is attained. When the z and y axis have been established, the x axis can be determined by completing the right-hand coordinate system [1].

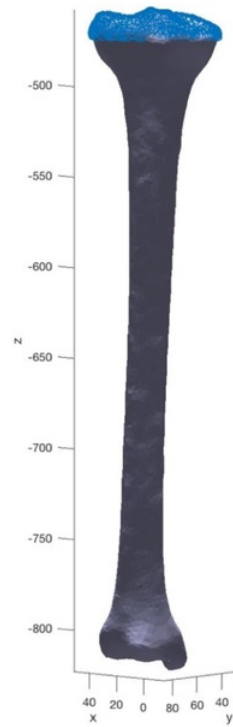


Figure 3.10: User removing unbrushed area of tibia model

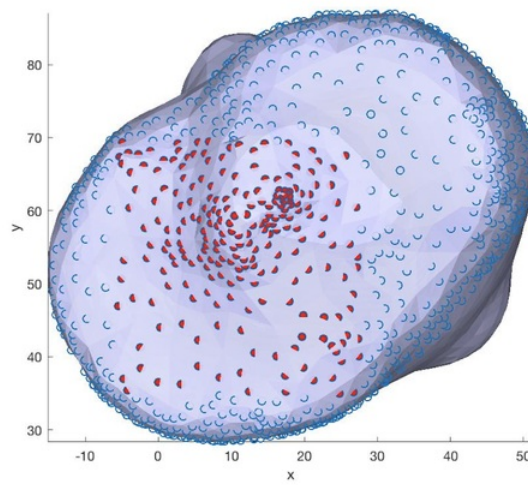


Figure 3.11: User selection on the superior view of the tibia

Chapter 4

Results and Analysis

4.1 The femur's coordinate system

The mechanical axis, Z , of the femur is determined by connecting the centre of the femoral head to the midpoint of the medial and lateral epicondyles at the most distal point of the posterior surface. When the user selects the femoral head using the brush tool, each point that is selected is the centre point of a face on the model in which a normal is taken from. Each normal that is taken from each face is compared to one another. The shortest possible distance between the two normals is determined by locating the two unique points in which the two lines are closest, a method developed by Dan Sunday [9]. Once this distance is calculated, the midpoints are stored and averaged, thus, determining the centre of the femoral head. A sphere is then fitted onto the femoral head, using the coordinates of the centre of the femoral head as the centre point, thus allowing us to determine the approximate radius of the femoral head. The results are shown in table 4.1 and figure 4.1.

Centre of femoral head $[x, y, z]$	$[-37.337, 18.086, -31.204]$
Number of faces selected	1162
Fitted sphere's radius	22
Average offset of each face	1.098

Table 4.1: Results from finding the centre of the femoral head

In order to attain more reliable results, using a circular brush to select the femoral head would be suitable as it allows for more accuracy when selecting the data. However, the graphical user interface currently used in Maltab only allows rectangular selections. MATLAB GUI Toolbox is a toolbox developed by MatPi and has many advanced features that can be utilised in Matlab such as “mbrush” which allows data to be selected using the lasso capability. However, when integrating this function into the program, numerous

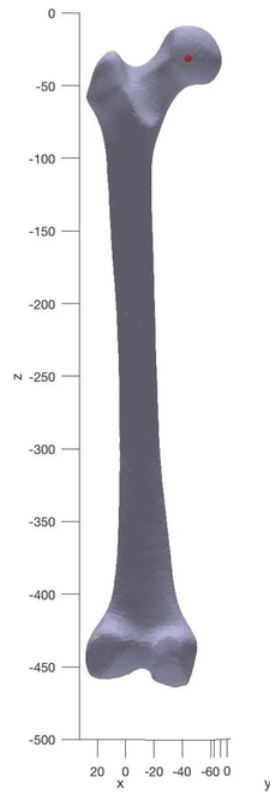


Figure 4.1: Centre of the femoral head calculated by Matlab

errors emerged and the results were no longer reliable. The results for the centre of the femoral head appeared on other areas of the femur such as the shaft. Thus, the previous method of using the standard graphical user interface provided by Matlab was utilised.

To complete the definition of the femur's mechanical Z axis the midpoint of the two epicondyles at the most distal point on the posterior surface must be specified. When the user selects the distal end of the femur, the program will determine the maximum and minimum x values and these coordinates denote the values for the lateral and medial epicondyles. The results are shown in table 4.2 and figure 4.2. Using the coordinates for the two epicondyles, the X axis can be established along with the origin of the femur's coordinate system. The program gathers all the points that are midway between the

lateral and medial epicondyles. Each of these midpoints are analysed and the distance between these points and the centre of the femoral head is determined. The greatest distance will be saved and provides us with the most distal point on the posterior surface of the femur thus, allowing the establishment of the Z axis.

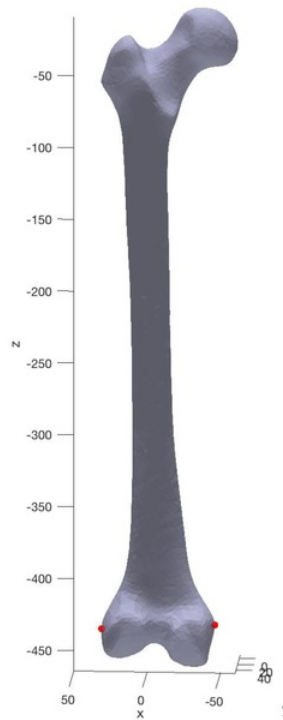


Figure 4.2: Lateral and medial epicondyles calculated by Matlab

	$[x, y, z]$
Lateral epicondyle	[32.337, 18.086, -31.204]
Medial epicondyle	[-44.572, 24.322, -436.189]
Most distal point on posterior surface	[-5.730, 10.450, -451.146]

Table 4.2: Coordinates of the epicondyles and most distal point

Now that the mechanical Z axis and X axis has been established, the Y axis can be defined as the cross product of the Z axis and X axis. This can be calculated using the “cross” function in Matlab. The code below shows the definition of the Z axis and X axis and how they are plotted. Both of the axes need to be defined as arrays, as shown in lines 4 and 9, in order to determine the cross product using Matlab’s function and this result is also plotted as shown in lines 12 and 13. The results are demonstrated in figure 4.3.

```

1 % mechanical z axis
2 pts = [midEpicondyle; centreOfHead];
3 zAxis = plot3(pts(:,1), pts(:,2), pts(:,3), 'r', 'LineWidth', 2);
4 zar = [pts(:,1) pts(:,2) pts(:,3)];
5
6 % x axis
7 pts = [medialEpicondyle; lateralEpicondyle];
8 xAxis = plot3(pts(:,1), pts(:,2), pts(:,3), 'r', 'LineWidth', 2);
9 xar = [pts(:,1) pts(:,2) pts(:,3)];
10
11 % y axis
12 yar = cross(zar, xar);
13 yAxis = plot3(yar(:,1), yar(:,2), yar(:,3), 'r', 'LineWidth', 2);

```

The Y axis is inconsistent with Grood and Suntay’s definition of the coordinate system as shown in figure 2.3. It is evident that the femur’s Z axis and X axis determined through Matlab does not intersect. This may be the basis for the error. Due to time restrictions, the origin for the femur’s coordinate system was redefined to be the midpoint of the lateral and medial epicondyles hence, the mechanical axis was also redefined as the connection of the centre of the femoral head and the midpoint of the two epicondyles. Thus, the joint coordinate system by Grood and Suntay is no longer being utilised, but provides a basis for the altered coordinate system being implemented. After redefining the origin, the results are shown in table 4.3 and figure 4.4. Matlab produces the results for the Y axis as an array however, the axis is not shown on the femur. Thus, manual calculations and online cross product calculators were utilised to ensure the validity of the results provided by Matlab. However, after numerous alterations, the Y axis was not visually attained on the femur. The results are not as desired and this may be due to the redefinition of the coordinate system as there is no previous research to confirm that this altered coordinate system can be applied to the femur in a reliable manner.

	$[x, y, z; x, y, z]$
z axis	$[-6.142, 25.771, -437.708; -37.345, 17.930, -30.900]$
x axis	$[-44.572, 24.322, -436.189; 32.288, 27.221, -439.227]$
y axis	$[-595.397, 16\ 830, 999.291; -7\ 034.4, -17\ 401, -1\ 595.5]$

Table 4.3: Results of the femoral axes as arrays

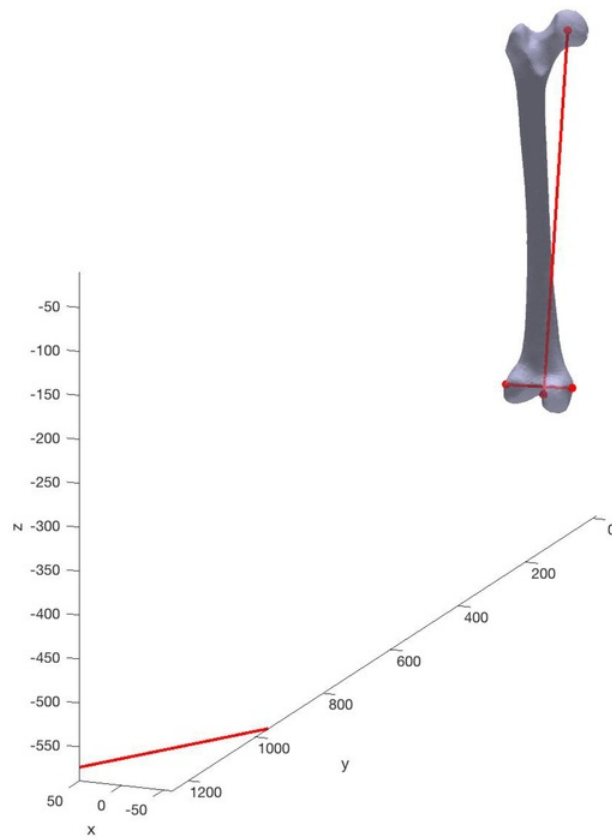


Figure 4.3: Calculated coordinate system for the femur

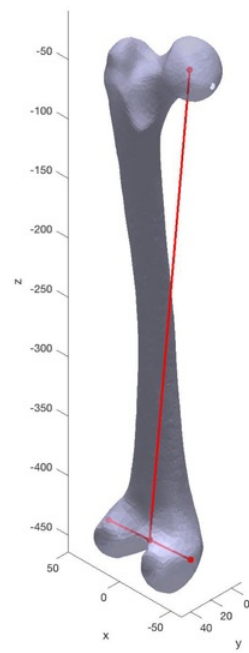


Figure 4.4: Redefined coordinate system for the femur

4.2 The tibia's coordinate system

The z axis is the fixed body axis in the tibia. This axis is defined such that it passes through the centre of the ankle and midway between the lateral and medial intercondylar eminences. The centre of the ankle is specified as the midpoint of the of the most medial point of the medial malleolus on the tibia and the most distal point of the lateral malleolus on the fibula. To determine the most medial point on the tibia, the two distal ends of the shaft is selected by the user. These selected points will create a line of best fit and all the distances down the central axis is determined. The maximum distance and the centre point of the face that it corresponds to is stored and thus computes the most medial point. Along with these coordinates, the results for the axis normal and axis point are displayed as shown in table 4.4.

When the program has been run several times, the results for the axis normal and axis point alter slightly whilst, the result for the most medial point remains quite constant. Although the determination of the most medial point relies on the definition of the axis normal, it remains moderately constant as it determines the maximum distance from the axis normal. Thus, if there was inconsistency during the user selection of the two distal ends of the tibia's shaft, the results would not be affected.

The most distal point of the lateral malleolus is found by selecting the distal end of the tibia. When the program is executed, vector **B** is found, a vector that is from the most distal point to the point on the axis that is closest to the most distal point. The program will obtain all the distances of all the points along **B** and the greatest distance will be stored, thus, defining the most distal point. Therefore, the centre of the ankle can be determined by averaging the most medial point and the most distal point. The results are shown in table 4.4 and in figure 4.5 where the green point depicts the most medial and distal points and the red point denotes the centre of the ankle.

	$[x, y, z]$
Axis normal	[0.010; -0.051; 0.999]
Axis point	[24.181, 57.491, -602.118]
Most medial point	[16.298, 47.825, -822.654]
Most distal point	[35.259, 83.690, -802.856]
Centre of ankle	[25.779, 65.758, -812.755]

Table 4.4: Results for defining the centre of the ankle

However, these results are not reliable as the most distal point of the lateral malleolus should be found on the fibula and not on the tibia. This was not carried out as a STL model of the fibula was unavailable thus, the most distal point on the tibia was found.

This will alter the definition of the mechanical axis on the tibia slightly, however, will not dramatically reorient the axis. This inaccuracy may alter the determination of the y axis as it is dictated by the cross product of the z and x axis. When a model of the fibula is attained, more reliable results of defining the coordinate system on the tibia can be achieved.

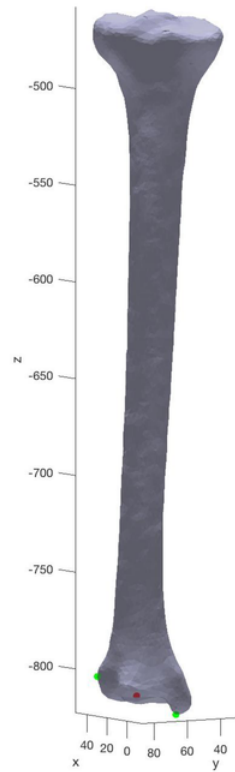


Figure 4.5: The most medial and distal points and the centre of ankle

The next landmark that needs to be defined when establishing the tibia's mechanical axis is the midpoint of the two intercondylar eminences. The lateral intercondylar tubercle is determined when the user selects the left side of the proximal end of the tibia, ensuring that the medial intercondylar tubercle is not selected. The code used to determine the lateral tubercle is similar to the code used to find the most medial point of the

medial malleolus. When code is being executed, all the distances down the central axis is determined through the use of the dot product and stored. The program will acquire the maximum distance and the centre point of the face in which it corresponds to, thus, determining the coordinate for the lateral intercondylar tubercle as shown in table 4.5.

To find the medial intercondylar tubercle, the same technique is applied, however the user will select the right side of the proximal end of the tibia whilst ensuring that the lateral intercondylar tubercle is not selected. The user will right click the selected data and select “paste data into command line”, allowing the coordinates of all the selected data to appear in Matlab’s command window. The coordinate with the highest value for z will identify the medial intercondylar tubercle as it is defined as the highest point of the selected data. The user will have to manually input the results of this coordinate into the program.

Initially, when the program was being written, the method of finding the medial intercondylar tubercle was similar to that used when finding the lateral intercondylar tubercle. Using the same code that was slightly altered for the medial side, the program would define the medial intercondylar tubercle as the same coordinates as the lateral intercondylar tubercle even though the lateral side was not selected. After numerous alterations of the program, the only results achieved were still the coordinates of the lateral intercondylar eminence. When looking at the tibia model, it is evident that the eminence on the lateral side is higher than the medial intercondylar eminence. Thus, applying the technique used to calculate the most medial point and the lateral intercondylar eminence may be redundant as it calculates the lowest and highest points on the full tibia and not just the area that is selected by the user. Due to time restraints, an automated method was not implemented, thus manual interaction by the user is necessary in this section.

Once the coordinates of the lateral intercondylar tubercle and medial intercondylar tubercle have been located, the midpoint of these two eminences can be defined to obtain the z axis of the tibia. The results are shown in table 4.5 and figure 4.6 where the two red dots are the lateral and medial intercondylar tubercles and the green point is the midpoint of the two intercondylar eminences.

	$[x, y, z]$
Lateral intercondylar tubercle	[17.329, 62.147, -457.583]
Medial intercondylar tubercle	[10.453, 59.309, -458.721]
Midpoint of the two eminences	[13.891, 60.728, -458.152]

Table 4.5: Results from finding the midpoint of the intercondylar eminences

The y axis of the tibia is established by taking the cross product of the z axis and a line connecting the centre of the plateaus on the superior view. Using the brush tool,

the user can select the proximal end of the model of the tibia, right click a point and select “remove unbrushed”. By doing so, the execution when determining the centre of the plateaus will be much faster as instead of testing all the faces that run through the whole tibia, only the selected data on the surface of the plateaus will be tested. When the unbrushed selection is not removed, the program will take a couple of hours to execute, whereas when the unbrushed data is removed, the program will take no more than 10 seconds to execute.

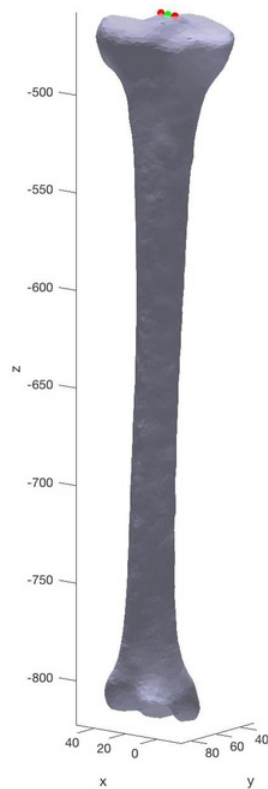


Figure 4.6: The two intercondylar eminences with their midpoint

In order to determine the centre of the plateaus, two circles need to be fitted on either side and the centre of these circles define the coordinates needed to establish this

line. On the Matlab forum, there a number of functions developed by different users that allow a circle to be fitted on 3D plots. However, downloading these functions and implementing them into the code was unsuccessful as errors would continuously occur or the tibia model and the window used to fit the circle would appear on separate figures. Thus, the rectangular brush selection was utilised as it can mimic an approximate circle if the selection is done carefully as shown in figure 4.7. The figure was edited to aid in the visualisation of how the circle would be fitted to a square selection. To fit a circle as accurately as possible using the rectangular brush selection, three of the corners of the rectangle should touch the perimeter of the tibia's surface and the length and width of the selection must be equal.

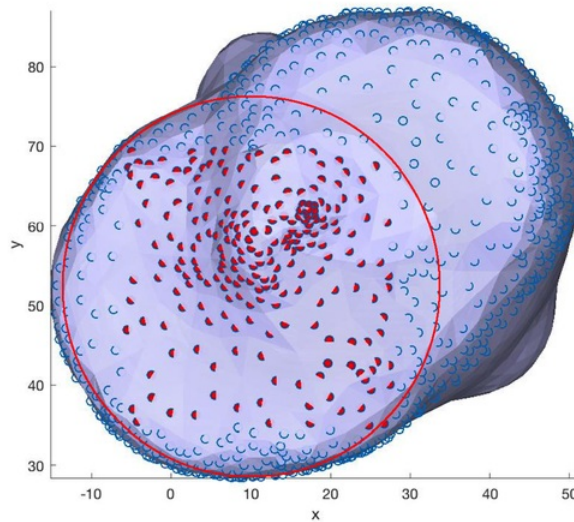


Figure 4.7: Edited circle on the square selection of the superior view

After the selection is made, when the program is run, each point that is selected on the model is the centre point in which a normal is taken from. The normals are then analysed and compared against one another and the shortest possible distance between the two normals is determined. When this distance has been calculated, the midpoints are stored and the mean is calculated, thus determining the centre point of the selected area. However, when the centre point is evaluated, it determines the centre of the entire selection and not just the surface of the superior view as shown in figure 4.8. To determine the centre of the plateau on the surface, the maximum z coordinates corresponding to the x and y coordinates found earlier are determined and substituted as shown in the code

below. The results are shown in table 4.6 and figure 4.9.

```

1 XData = testFaces(:,1);
2 YData = testFaces(:,2);
3 ZData = testFaces(:,3);
4
5 % find the max point
6 distances = zeros(size(faceCentreVertices,1),1);
7 for i = 1:length(distances)
8     A = [faceCentreVertices(i,1) - axisPoint(1), ...
          faceCentreVertices(i,2) - axisPoint(2), ...
          faceCentreVertices(i,3) - axisPoint(3)];
9     distances(i) = dot(A, axisNormal)*-direction;
10 end
11
12 MmaxPoints = faceCentreVertices(distances==max(distances),:);
13
14 if (size(MmaxPoints,1)==1)
15     MmaxPoint = MmaxPoints;
16 else
17     MmaxPoint = sum(MmaxPoints)/size(MmaxPoints, 1);
18 end
19
20 % substitute the max point of z for the z coordinate when plotting
21 scatter3(centreOfPlateauM(1), centreOfPlateauM(2), MmaxPoint(3), 'r', ...
          'filled')

```

Centre of plateau on the medial side $[x, y, z]$	[11.831, 54.556, -472.403]
Fitted circle radius for the medial side	76.6
Centre of plateau on the lateral side $[x, y, z]$	[22.890, 62.419, -457.583]
Fitted circle radius for the lateral side	120.1
Midpoint of the two centres $[x, y, z]$	[17.361, 58.488, -457.583]

Table 4.6: Results from finding the centre of the plateau

Now that the centre of the two plateaus have been established, the cross product of the line connecting the two centres and the mechanical fixed axis will define the y axis. However, when plotting the two on Matlab, it is evident that they do not intersect as shown in figure 4.10. This may be due to inaccuracies when determining the centre each plateau as a square selection was used instead of a circular selection. Due to time restrictions, the origin for the tibia's coordinate system was redefined to be the midpoint of the centre of each plateau. This coordinate is shown in table 4.6. Thus, the joint coordinate system by Grood and Suntay is no longer being utilised, but provides a basis for the coordinate system being implemented.

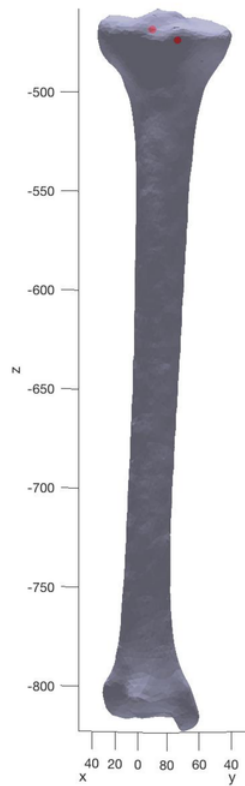


Figure 4.8: Inaccurate determination of the centre of the plateaus

Redefining the origin, the results as seen in figure 4.11 are obtained. The cross product of the z axis and the line connecting the centre of each plateau was determined using the “cross” function in Matlab. The code below shows the definition and plotting of the z axis and the line connecting the centre of each plateau. Each of them need to be defined as arrays such that the cross product can be determined by Matlab as shown in lines 4 and 9. Line 12 uses Matlab’s inbuilt function to determine the cross product of these two arrays and line 13 plots the results. However, whilst Matlab produces the results of the y axis as an array, the axis does not plot onto the tibia. Manual calculations and the use of online cross product calculators confirmed the validity of the results provided by Matlab however, through numerous alterations the y axis was not achieved on the plot.

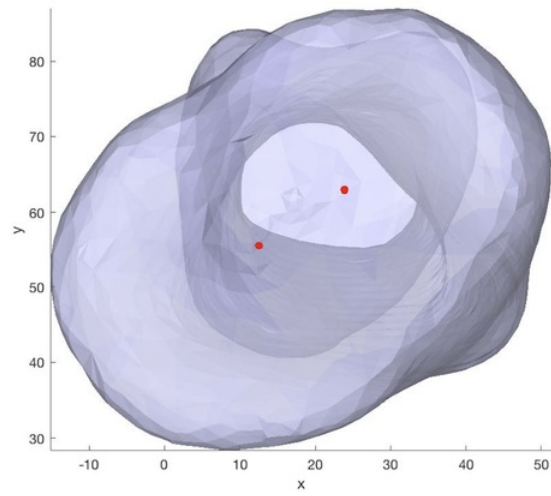


Figure 4.9: Centre of each plateau

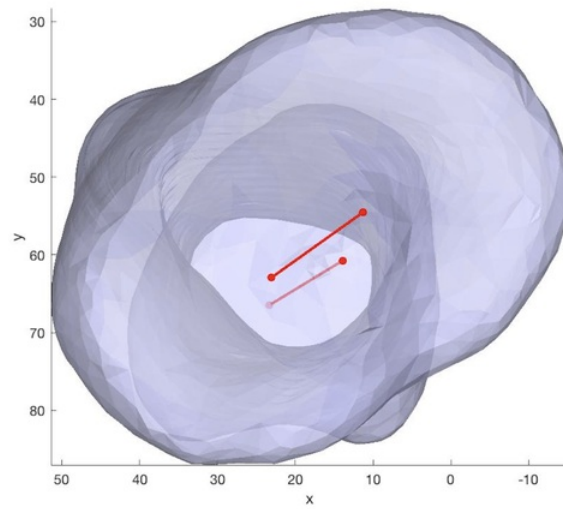


Figure 4.10: Superior view of the line connecting the plateaus and the z axis

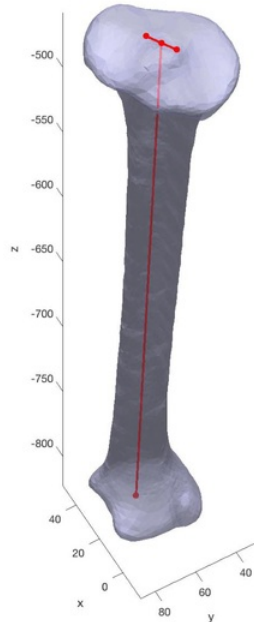


Figure 4.11: Redefined coordinate system for the tibia

```

1 % mechanical z axis
2 pts = [plateauMid; centreAnkle];
3 zAxis = plot3(pts(:,1), pts(:,2), pts(:,3), 'r', 'LineWidth', 2);
4 zar = [pts(:,1) pts(:,2) pts(:,3)];
5
6 % centre of plateau line
7 pts = [centreOfPlateauL; centreOfPlateauM];
8 plateau = plot3(pts(:,1), pts(:,2), pts(:,3), 'r', 'LineWidth', 2);
9 par = [pts(:,1) pts(:,2) pts(:,3)];
10
11 % y axis
12 yAxis = cross(par, zar);
13 plot3(yAxis(:,1), yAxis(:,2), yAxis(:,3), 'g', 'LineWidth', 2)
14
15 % x axis
16 xAxis = cross(yAxis, zar);
17 plot3(xAxis(:,1), xAxis(:,2), xAxis(:,3), 'g', 'LineWidth', 2)

```

Although a visual representation of the y axis was not accomplished, the array achieved through Matlab's calculations was utilised to determine the x axis of the tibia. The x axis is established by completing the right-hand coordinate system, therefore the cross product of the z axis and y axis will be taken. The results of all the axes are shown in table 4.7. However, similar to the results of the femur, when the axis is plotted onto the figure, it is quite far from the actual tibia model itself as shown in figure 4.12. The green line represents the x axis whereas the red denotes the z axis. However, since the results for the x axis is so much larger than the z axis, the red line is barely visible and appears as a point. It is clear that these are not the desired results. This may be due to the redefinition of the coordinate system as there is no previous research to confirm that this modified coordinate system can be applied to the tibia correctly.

	$[x, y, z; x, y, z]$
z axis	[16.535, 58.875, -457.583; 25.779, 65.758, -812.755]
Line connecting the centre of each plateau	[22.992, 63.061, -457.583; 10.679, 54.688, -457.583]
y axis	[-1 915.6, 2 817.1, 291.979; -14 359, -3 116.8, -707.591]
x axis	[-1.31e+06, -8.72e+05, -1.61e+05; 2.58e+06, -1.17e+07, -8.64+05]

Table 4.7: Results of the axes as an array

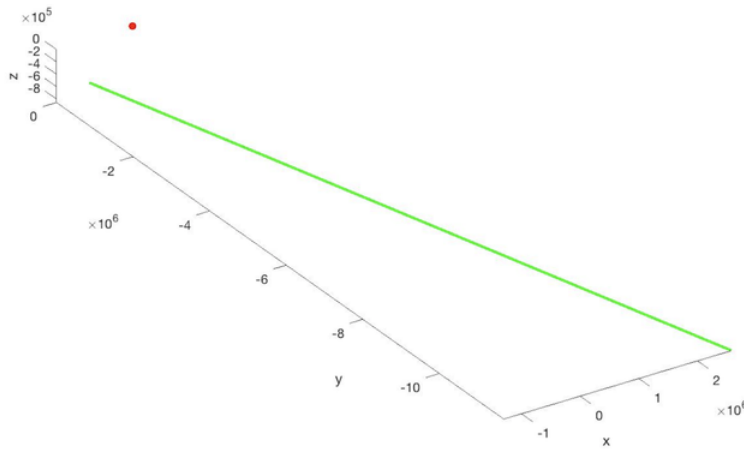


Figure 4.12: x axis of the tibia

Chapter 5

Conclusions and Future Work

5.1 Conclusions

The objective of this thesis was to develop an automated Cartesian coordinate system for the femur and tibia so that the joint coordinate system can be established at the knee. Automating JCS is important as it allows consistency and thus, more reliable results whilst, reducing the time spent as it will no longer have to be taken in a manual manner. This program allows simplicity when manipulating the orientation and enables the data to be presented in different clinical systems. However, throughout the course of this project, the joint coordinate system that was established by Grood and Suntay was redefined and used as a basis to implement a slightly different coordinate system.

A literature review was conducted in order to perceive the background of this thesis and how the joint coordinate system is established. The literature review enabled the research of three-dimensional knee kinematics and how JCS is implemented onto the knee and other various joints.

This thesis will provide the foundation to allow the successful implementation of the joint coordinate system on the knee by identifying anatomical landmarks in the tibia and femur and thus allows improvement in the description of fluoroscopy data as more accurate results will be achieved. Additionally, it will allow ease when researching and examining the effects of different coordinate systems on knee fluoroscopy results. The automation of JCS is not only limited to the knee, but can be applied to various other joints in the human body thus, allowing simplicity when describing and manipulating joint motion.

5.2 Future Work

It is recommended that the Matlab program developed in this thesis is further progressed to successfully establish the joint coordinate system for the knee. Currently, the brush tool used to select the femoral head is rectangular however, a circular brush will increase the accuracy when determining the centre of the femoral head. This does not alter the results significantly however, would be desirable to increase the accuracy.

To define the centre of the ankle, the most distal point and the most medial point on the two malleolus' are required. However, a STL model of the fibula was unavailable thus, the most distal point of the lateral malleolus was found on the tibia during the course of this thesis. This decreased the reliability of the results when finding the centre of the ankle and thus, the accuracy of the tibia's fixed axis. A STL model of the tibia should be developed and implemented into the program so that the most distal point can be found on the fibula.

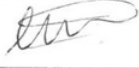



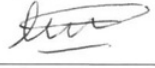

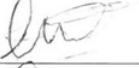

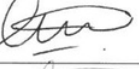

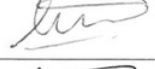
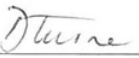
Currently, the determination of the medial intercondylar tubercle requires manual interaction from the user as the coordinates of this point is entered in the program. This should be automated by the program to minimise user interaction and allow ease of use.

When the determining the centre of each plateau in the tibia, a circle needs to be fitted on either side. However, the program currently uses a square selection to determine these centre points. The program should be further developed to enable the fitting of two circles on the superior view of the tibia. Although, the results will be quite similar, this is another step to increasing the accuracy of the automated coordinate system.

Bibliography

- [1] E. S. Grood and W. J. Suntay, “A joint coordinate system for the clinical description of three-dimensional motions: application to the knee,” *Journal of biomechanical engineering*, vol. 105, no. 2, pp. 136–144, 1983.
- [2] D. Dabirrahmani and M. Hogg, “Modification of the grood and suntay joint coordinate system equations for knee joint flexion,” *Medical engineering & physics*, vol. 39, pp. 113–116, 2017.
- [3] G. Pennock and K. Clark, “An anatomy-based coordinate system for the description of the kinematic displacements in the human knee,” *Journal of biomechanics*, vol. 23, no. 12, pp. 1209–1218, 1990.
- [4] G. Desroches, L. Chèze, and R. Dumas, “Expression of joint moment in the joint coordinate system,” *Journal of biomechanical engineering*, vol. 132, no. 11, p. 114503, 2010.
- [5] G. Cole, B. Nigg, J. Ronsky, and M. Yeadon, “Application of the joint coordinate system to three-dimensional joint attitude and movement representation: a standardization proposal,” *Journal of biomechanical engineering*, vol. 115, no. 4A, pp. 344–349, 1993.
- [6] G. Wu, S. Siegler, P. Allard, C. Kirtley, A. Leardini, D. Rosenbaum, M. Whittle, D. D D’Lima, L. Cristofolini, H. Witte *et al.*, “Isb recommendation on definitions of joint coordinate system of various joints for the reporting of human joint motion—part i: ankle, hip, and spine,” *Journal of biomechanics*, vol. 35, no. 4, pp. 543–548, 2002.
- [7] G. Wu, F. C. Van der Helm, H. D. Veeger, M. Makhsous, P. Van Roy, C. Anglin, J. Nagels, A. R. Karduna, K. McQuade, X. Wang *et al.*, “Isb recommendation on definitions of joint coordinate systems of various joints for the reporting of human joint motion—part ii: shoulder, elbow, wrist and hand,” *Journal of biomechanics*, vol. 38, no. 5, pp. 981–992, 2005.
- [8] *Amira 3D Software User’s Guide*, Zuse Institute Berlin, 2016.
- [9] D. Sunday, “Distance between 3d lines and segments,” 2012. [Online]. Available: http://geomalgorithms.com/a07-_distance.html

Consultation Meetings Attendance Form

Week	Date	Comments (if applicable)	Student's Signature	Supervisor's Signature
0	10/7/2017	Overview of thesis. Getting started		
1	4/8/2017	08/08/17 Checking progress. Talked about moving to jointTrack		
3	17/8/2017	08/08/17 Talked through what needs to be done. Automated Knee Sys, MATLAB		
5	31/8/2017	Went through matlab code.		
Mid Sem break	19/9/2017	checked matlab code		
Mid Sem Break	28/9/2017	Help on matlab		
11	26/10/2017	matlab	