Automatic Measurement of Cartilage Thickness in the Human Knee

Simon Westfechtel, B.Sc.

November 15, 2021

1 Introduction

This paper seeks to discuss different methods to automate the measurement of cartilage thickness in the human knee using MRI scans. The methods are based on previous work by Wolfgang Wirth and Felix Eckstein. [WE08]
All computational methods make use of segmented MRI scans of the human knee, from the OAI dataset, and are written in Python. More precisely, there were two distinct sets of segmentations used: One set of manually segmented images, containing 507 samples, and one set of automatically segmented images, containing 24,783 samples. The set of automatic segmentations contained samples from the entire OAI dataset, i.e. over all time periods. Manual segmentations are stored as MHD and automatic segmentations as NIFTI files, respectively, which can be read and converted into Numpy arrays using the SimpleITK library. These arrays map each point in the three-dimensional space of the scan to an integer encoding of the segmentation, meaning for example points belonging to the femoral cartilage are assigned a value of 3. This makes isolating and extracting the cartilage volumes straightforward. Four different methods have been developed to determine mean cartilage thickness, plus other statistical measures. Thickness was measured for different subregions of the cartilage plate, which were determined as referenced in [WE08]. For a more detailed description of the procedure, refer to appendix A.

2 Mean cartilage thickness using meshes

This is a three-dimensional approach using meshes and normal vectors to determine the mean thickness of a cartilage volume. The main idea is building an upper and a lower mesh, and calculating the average distance between the two, by ray tracing along the normal vectors of the lower mesh against the upper mesh. Other methods, like a K-D-tree nearest neighbour search, are also possible, but have not been implemented. The mesh building is relatively trivial for the tibial cartilage due to its physical shape. Some formal definitions are necessary:

Definition 2.1 (Point Cloud) *Each cartilage is represented by a set of vectors V. The vectors $v \in V$ are defined as a triplet (x, y, z), with x, y and z denoting a unique position in the three-dimensional Euclidean space, such that $\forall u, v \in V : u \neq v$.*

Definition 2.2 (Mesh) *A mesh is a Delaunay-triangulated surface $DT(P)$ of a point cloud P, such that no point $p \in P$ lies inside the circumcircle of any triangle in $DT(P)$. [Wik21a]*

The point cloud representation of the tibial cartilage is obtained by filtering the aforementioned array representation of the MRI scan for the integer encoding specific to the tibial cartilage. The vectors forming the upper and lower surface of the cartilage, respectively, are defined as follows: Let P be the point cloud representing the tibial cartilage. Furthermore, let P_u, P_l be the point clouds

Algorithm 1 Point Cloud Representation of the Cartilage

```
1: procedure CartilageExtraction(A,C)
2:     $A \leftarrow$ image array
3:     $C \leftarrow$ color code $\in \{3, 4\}$
4:     $P \leftarrow \{\emptyset\}$
5:     for each $(x, y, z) \in A$ do
6:         if $A[(x, y, z)] = C$ then
7:             $P \leftarrow P \cup (x, y, z)$
8:
       return $P$
```

representing the upper and lower surface of the tibial cartilage. Let $f : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}; X \times Y \mapsto \mathbf{Z}$ be a function where X and Y are scalars representing x and y values in the three-dimensional Euclidean space, and $\mathbf{Z}$ be the vector of all z values in the Euclidean space for the position (x, y). Then P_u and P_l are defined as:

$$P_u := \{\ (x, y, z)_i \in P \mid z = max(f(x, y))\ \} \tag{1}$$

$$P_l := \{\ (x, y, z)_i \in P \mid z = min(f(x, y))\ \} \tag{2}$$

This essentially means grouping P by x and y coordinates and for each unique coordinate pair (x_i, y_i) in the Euclidean space, choosing the vector with the maximum z coordinate for the upper point cloud P_u, and the vector with the minimum z coordinate for the lower point cloud P_l. The result is two point clouds consisting of vectors (x, y, z), as seen in figure 1 (red points make up the upper cloud, green the lower). These are then converted into polygon meshes making use of the Delaunay triangulation. For the lower surface mesh, its surface normals are calculated. A surface normal vector is defined as the vector perpendicular to the tangent plane of the surface at a point P [Wik21b]. Finally, mean cartilage thickness, i.e. the mean distance between the two meshes, is calculated by tracing along the normal vectors of the lower mesh against the upper mesh. To this end, each surface normal is iteratively extended until it hits the upper mesh, and the Euclidean norm of the vector, i.e. its length, and therefore the distance between the two meshes at a specific point, is computed. This is done for all surface normals, and the average, standard deviation of the calculated distances as well as the mean of the minimum (maximum) 1% of measurements. Due to the orthogonal nature of the normals, these traces can take into account the topography of the cartilage, allowing for more accurate distance measurements than going in the same static direction for every point $p \in P_l$. For other methods not making use of surface normals, like the previously mentioned nearest neighbour search, the Delaunay triangulation may be omitted.

Mesh building for the femoral cartilage is not as trivial. Due to its convex shape, obtaining the upper and lower point clouds P_u and P_l can not be achieved

Algorithm 2 Upper and Lower Point Clouds (Tibia)

```
1: procedure PointClouds(P)
2:     $P_u \leftarrow max(P.groupby((x, y)))$
3:     $P_l \leftarrow min(P.groupby((x, y)))$
4:     return $P_u, P_l$
```

Algorithm 3 Mean Cartilage Thickness (Tibia)

```
1: procedure TibialThickness($P_u$, $P_l$)
2:     $D \leftarrow \{\emptyset\}$
3:     $M_u \leftarrow delaunay(P_u)$
4:     $M_l \leftarrow delaunay(P_l)$
5:     $N_l \leftarrow normals(M_l)$
6:     for each $n \in N_u$ do
7:         $v \leftarrow raytrace(n, M_u)$
8:         $d \leftarrow ||v||$
9:         $D \leftarrow D \cup d$
10:    $\overline{D} \leftarrow \frac{\sum_{i=0} d_i \in D}{|D|}$
11:    return $\overline{D}$
```

by simply always taking the maximum (minimum) z value for each coordinate pair (x, y). For an illustration of the problem, refer to figure **<TODO> Abbildung einfuegen, die das Problem veranschaulicht $< / >$**. Using the same approach as for the tibia results in point clouds where some areas are left bare, as illustrated in figure 3. One solution, used in this approach, is splitting the volume into parts, and rotating the curved sections, such that it is once again possible to choose points according to the z coordinate. For a detailed description of how this splitting is achieved, refer to appendix B. This allows for building an upper and lower point cloud and corresponding Delaunay mesh for each part, calculating the mean distance in the same manner as before (i.e. ray tracing), and finally combining the results.

3 Mean cartilage thickness using ray tracing from a central point

This is a three-dimensional approach using ray tracing along normal vectors to determine the mean thickness of a cartilage volume. This is another proposed solution to the previously discussed issue with the shape of the femoral cartilage volume. Instead of using meshes, this approach takes a central point underneath or above the cartilage and utilizes ray tracing from that point against the volume to discover intersection points. The central point is determined by calculating the halfway points between minimum and maximum x, y and z coordinates of the cartilage volume, respectively.

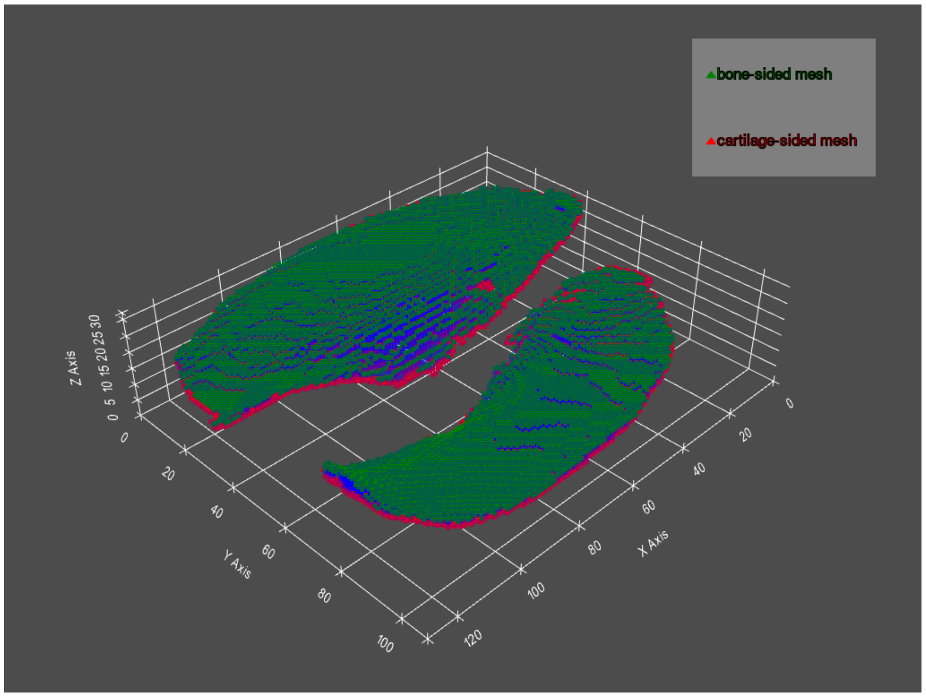

Figure 1: Point clouds of the tibial cartilage. The figure is flipped, i.e. the camera is focused on the bone side of the cartilage.

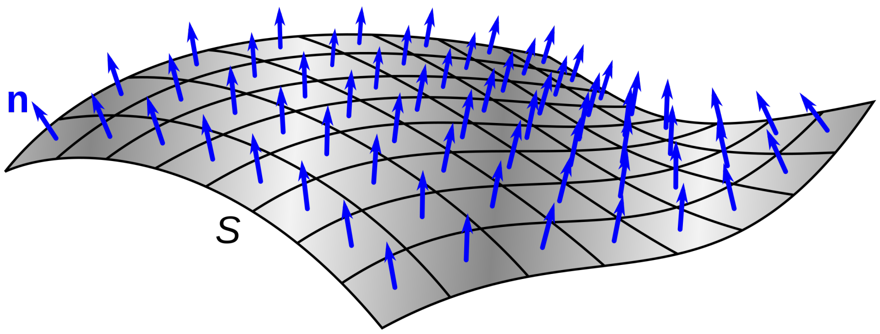

Figure 2: A curved surface showing the unit normal vectors (blue arrows) to the surface. [Che19]

A sphere is constructed around the central point, and each of its normal vectors gets extended until it hits the cartilage point cloud, or a maximum number of iterations is reached. If a point is hit, it is saved and the vector again gets extended until it doesn't hit any points anymore, i.e. it is extended

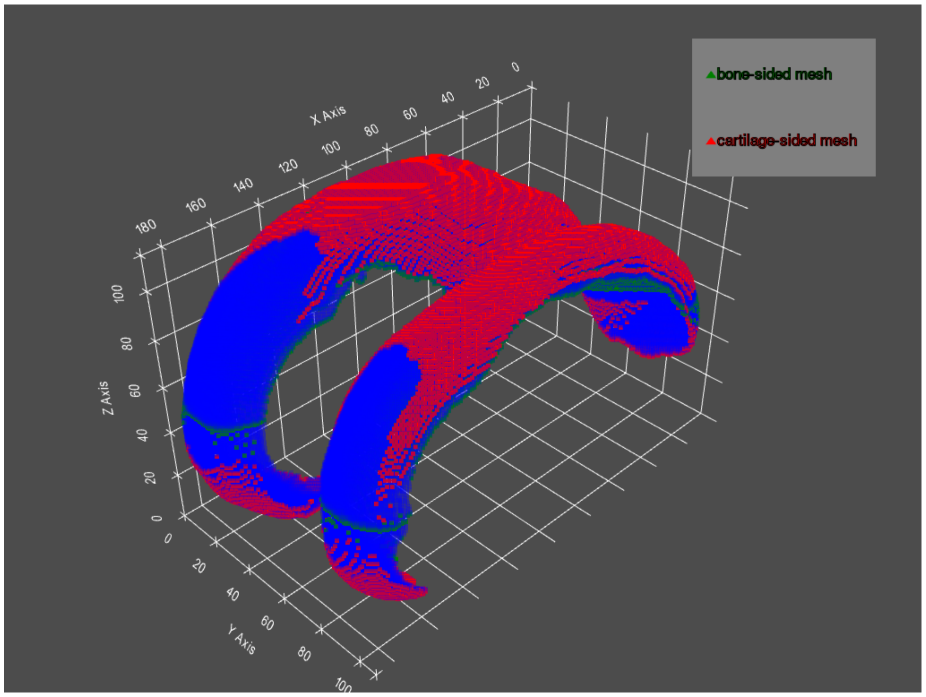

Figure 3: Point clouds of the femoral cartilage.

past beyond the cartilage. The distance between the first and last point hit is calculated and added to the result set. This way, it is possible to determine the average thickness of the cartilage. One issue with this approach is that it is computationally expensive, as intersection problems tend to be; in essence, there is a trade-off between accuracy and runtime: The more vectors are used for the ray tracing, the more accurate the result is going to be, but each added vector makes the computation more expensive. The resolution of the sphere was set to 30×30, resulting in $30 + 29 \cdot 28 = 842$ normal vectors. No optimization analysis has been performed as of yet.

4 Mean cartilage thickness using two-dimensional function fitting

4.1 Determine thickness via function normals

This is a two-dimensional approach using a least-squares fit of single cartilage layers to determine the mean thickness of a cartilage volume. As the MRI scans consist of a number of slice exposures, it is possible to do the thickness calculation layer by layer. For each layer, a polynomial function gets fitted over

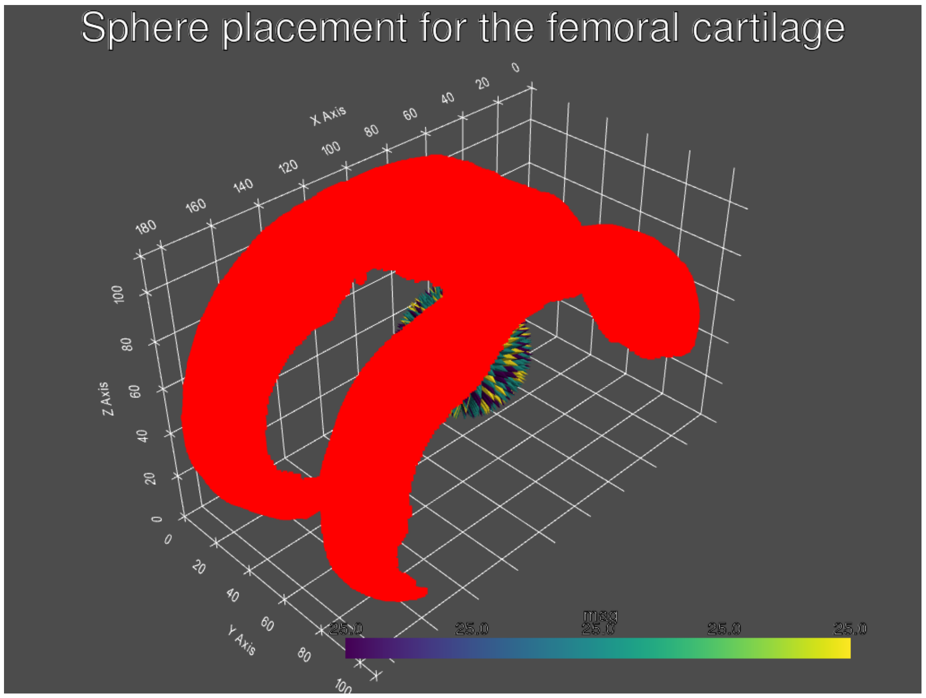

Figure 4: Ray tracing against the femoral cartilage using a central sphere

the data points, i.e. through the middle of the volume, and a variable number of normals is calculated along the fitted function. For each normal, the inner- and outermost intersection points with the cartilage are determined, and the distance between these to points is added to the result set, as illustrated in figure 6.

4.2 Determine thickness via integration/function values

This is another two-dimensional variant, where two functions are fitted over the data points, i.e. not through the middle of the volume but rather along the outlines of the volume, as illustrated in figure 7. The distance between the functions, i.e. the thickness of the cartilage, can then easily be calculated, for example by integrating both functions and taking the difference ($\int f(x)dx - \int g(x)dx$), or calculating the difference between the function values for every x value ($[\sum_{x_i=0}^{max(x)} f(x_i) - g(x_i)] \cdot \frac{1}{max(x)}$).

One issue with both of these approaches is that the function fitting may not work well for certain shapes, especially for layers where the number of data points is very sparse. The degree for the polynomial fit was chosen as four. This is because in practice, no layer is going to have more than two inflection points (layers of the femoral cartilage can generally be approximated

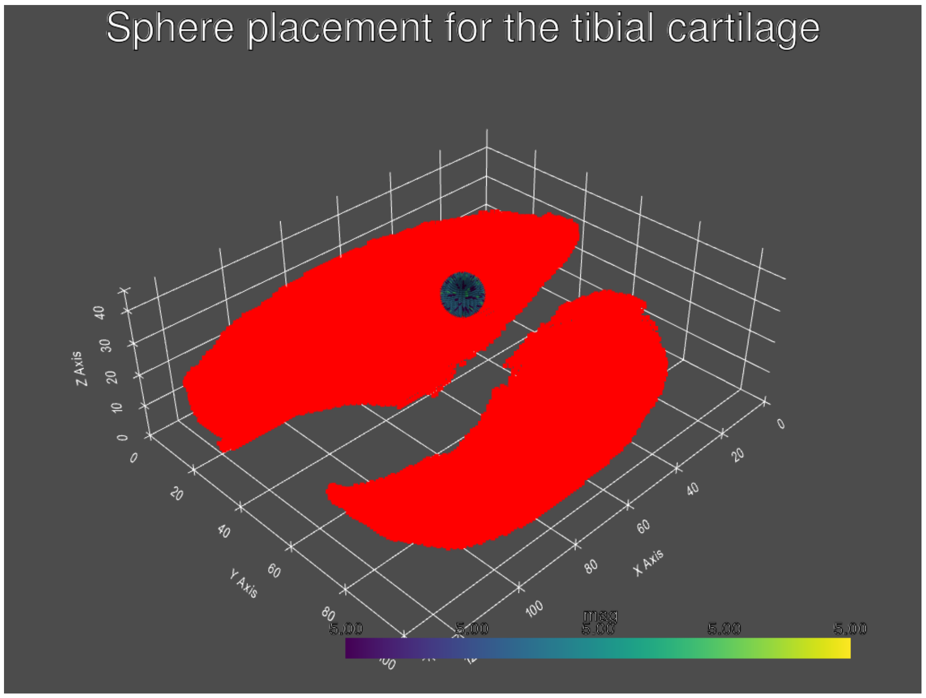

Figure 5: Ray tracing against the tibial cartilage using a central sphere

by a second-order polynom, while the layers of the tibial cartilage volume follow a third-order form.) and choosing a higher degree than needed is undesirable (overfitting, Runge's phenomenom, etc). While in theory, a degree of two and three, respectively, would be sufficient, it has been set to four to accomodate for edge cases where a lower-order fit might yield poor or no results, e.g. for the layers on the edges of the volume, and the higher flexibility that comes with a higher degree is beneficial.

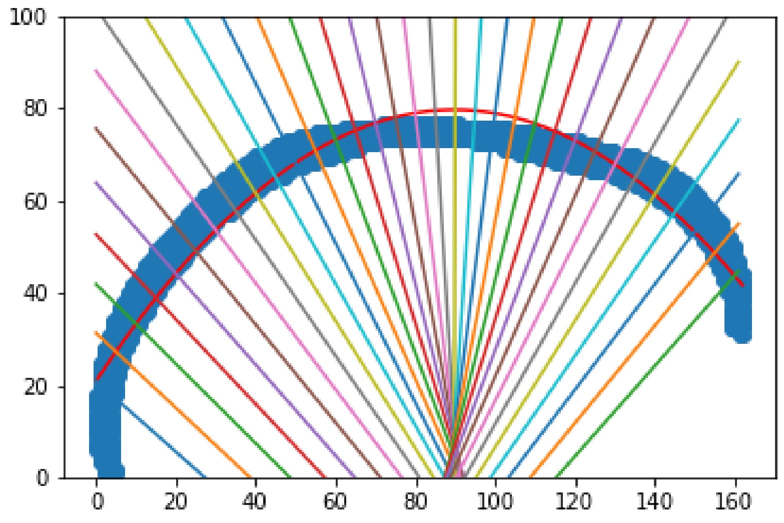

Figure 6: Normals along a two-dimensional least squares fit

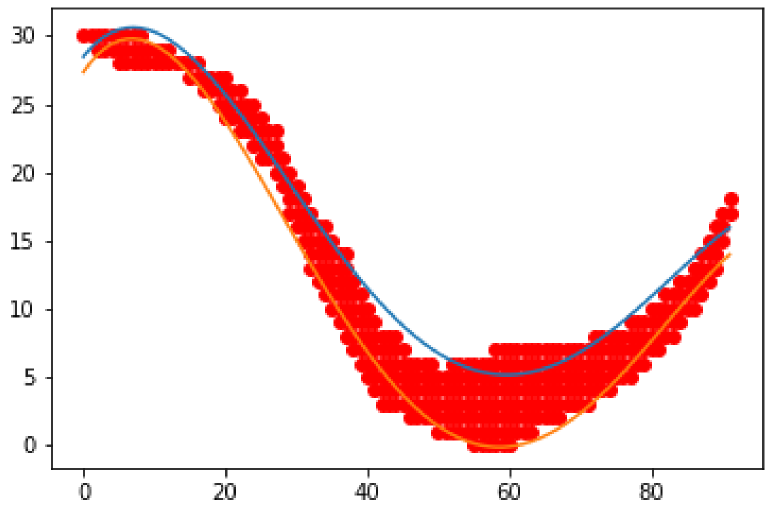

Figure 7: Two functions fit over the outlines of a tibial cartilage slice exposure

A Determination of the cartilage subregions

As proposed by Wirth & Eckstein, the two cartilage plates of the medial / lateral femoral condyle are divided into three subregions each (central, internal and external), while the two cartilage plates of the medial / lateral tibia are divided into five subregions each (central, internal, external, anterior, posterior). For nomenclature, refer to the table of acronyms below.

A.1 Determination of the femoral subregions

For each plate of the femur, each subregion should encompass approximately 33% of the plate volume. This was achieved in practice by splitting the volume into three parts along the y-axis; two splitting points $x1, x2$ are chosen arbitrarily, such that $y_{min} < y1 < y2 < y_{max}, y1 = \frac{y_{max}-y_{min}}{3}, y2 = 2 \cdot y1$, where y_{max}, y_{min} are the maximum / minimum y values, respectively. $y1$ and $y2$ are then iteratively moved along the y-axis until the constraint is satisfied, i.e. 33% of all data points lie left of $y1$, 33% lie between $y1$ and $y2$ and 33% lie right of $y2$.

A.2 Determination of the tibial subregions

For each plate, an ellipse around the center of the plate, aka the central region, should encompass approximately 20% of the plate volume, and four triangles surrounding the ellipse should be of variable size. This was achieved in practice by first determining the center of gravity of the plate through K-Means clustering and constructing an ellipse around this point; a radius r is chosen arbitrarily, and is lengthened iteratively until the constraint is satisfied, i.e. 20% of all data points lie within the ellipse. The four triangles surrounding the ellipse are determined by calculating the corners a, b, c, d of the plate, and data points are assigned to subregions according to their position relative to the vectors $\vec{ac}$ and $\vec{db}$, which is calculated via cross product.

```
Procedure to determine femoral subregions
Given:
y_axis := range of x-axis
plate := data points (x, y, z) making up a cartilage
 ↪  plate

Procedure:
y_min, y_max := min(y_axis), max(y_axis)
y_range := y_max - y_min
y1 := y_range / 3
y2 := 2 * y1
first_third := empty set
second_third := empty set

```

```
while len(first_third) / len(plate) is not .33 do
        first_third := {d ∈ plate | d.y < y1}
        if len(first_third) > .33
                y1 := y1 - 1
        else
                y1 := y1 + 1

while len(second_plate) / len(plate) is not .33 do
        second_plate := {d ∈ plate | y1 < d.y < y2}
        if len(second_plate) > .33
                y2 := y2 - 1
        else
                y2 := y2 + 1
```

```
Procedure to assign a femoral point to a subregion
Given:
plate := data points (x, y, z) making up a cartilage
 ↪  plate
y1, y2 := split points

for point in plate do
        if point.y < y1
                point.region = external/internal #
                 ↪  depending on whether point lies in
                 ↪  left or right plate
        if y1 < point.y < y2
                point.region = central
        else
                point.region = external/internal #
                 ↪  depending on whether point lies in
                 ↪  left or right plate
```

```
Procedure to determine tibial subregions
Given:
plate := data points (x, y, z) making up a cartilage
 ↪  plate

Procedure:
r := 20
c := KMeans(plate)
points_in_ellipse := empty set

while len(points_in_ellipse) / len(plate) is not .2 do
        points_in_ellipse := {d ∈ plate | dist(d, c) < r}
        if len(points_in_ellipse) > .2
                r = r / 2
        else
                r = r + .5

x_min := {min(d.x) | d ∈ plate}
x_max := {max(d.x) | d ∈ plate}
y_min := {min(d.y) | d ∈ plate}
y_max := {max(d.y) | d ∈ plate}

a := (x_min, y_min)
b := (x_max, y_min)
c := (x_max, y_max)
d := (x_min, y_max)
```

```
Procedure to assign a tibial point to a subregion
Given:
plate := data points (x, y, z) making up a cartilage
 ↪  plate
a, b, c, d := plate corners
points_in_ellipse := set of points lying within the
 ↪  central ellipse

Procedure:
for point in plate do
        if point is in points_in_ellipse
                point.region := central

        ac := c - a
        db := b - d
        pc := c - point
        pb := b - point

        if ac × pc > 0
                if db × pc > 0
                        point.region := internal
                else
                        point.region := posterior
        else
                if db × pc > 0
                        point.region := anterior
                else
                        point.region := external
```

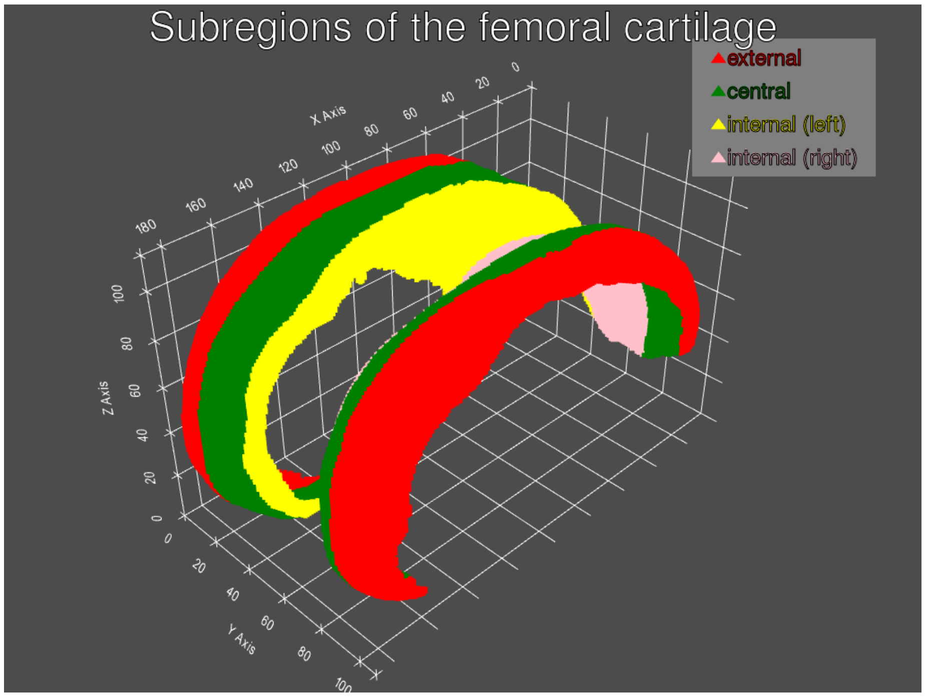

Figure 8: Subregions of the femoral cartilage

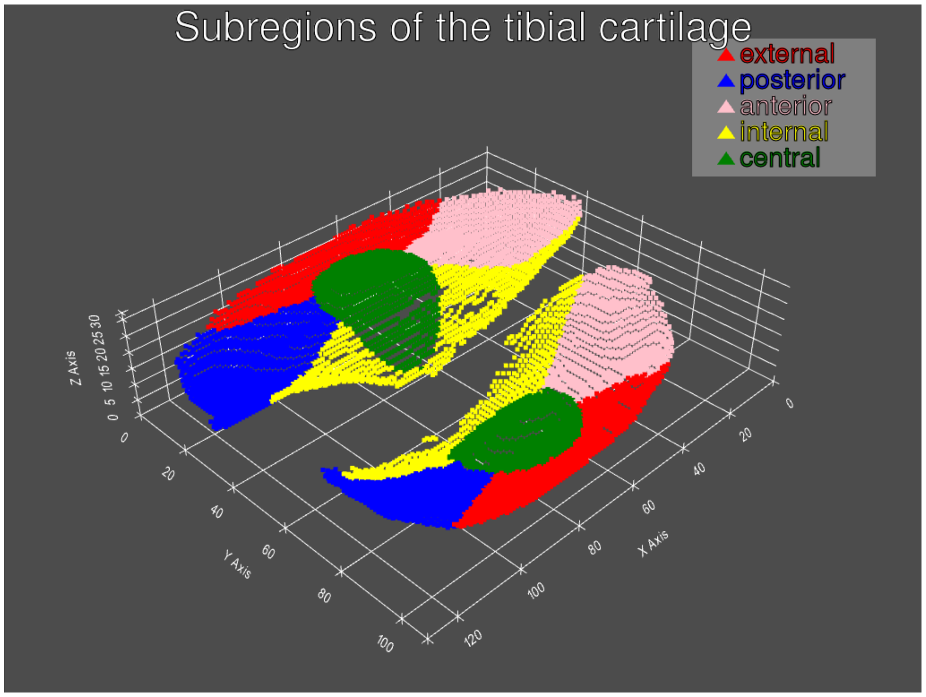

Figure 9: Subregions of the tibial cartilage

B Splitting of femoral cartilage volume for meshes

As described above, constructing meshes for the femoral cartilage is not possible without some pre-processing. The convex parts of the volume on either side need to be split off and rotated by 90° so that minimum and maximum points can reliably be extracted without missing a significant amount. The sections are isolated as follows:
The cartilage volume is a point cloud made up of vectors (x, y, z). Group the points by (x, y) and for each of these combinations, calculate the range of z. Then calculate the mean or median of z and take all combinations (x, y) for which the range of z is less than the previously calculated mean/median, i.e. those points where the minimum and maximum z values are not too far apart. Note that the convex parts of the cartilage are characterized by a large ranges for z. Having now isolated the central part of the volume, it is a trivial task to extract the convex parts: We can calculate the minimum and maximum x value of the central vectors, and everything left of the minimum gets assigned to the left convex part, and everything right of the maximum gets assigned to the right convex part.
With the sections isolated, the problematic convex parts can then be rotated by 90°. Now, again for every combination (x, y), the vector $(x, y, max(z))$ can be added to the upper mesh, and the vector $(x, y, min(z))$ to the lower ($min(z)/max(z)$ may be replaced by distance to a central point, depending on implementation).

```
Procedure to split a femoral cartilage volume into three
↪  parts
Given:
volume := vectors (x, y, z) making up the volume

Procedure:
z_range := volume.group_by((x, y)).z.max() -
↪  volume.group_by((x, y)).z.min()
z_med := median(z_range)
z_index := {d ∈ volume | d.z < z_med}
lower_bound := {min(d.x) | d ∈ z_index}
upper_bound := {max(d.x) | d ∈ z_index}
left_part := {d ∈ volume | d.x < lower_bound}
middle_part := {d ∈ volume | lower_bound < d.x <
↪  upper_bound}
right_part := {d ∈ volume | d.x > upper_bound}
left_part := rotate(90)
right_part := rotate(90)
```

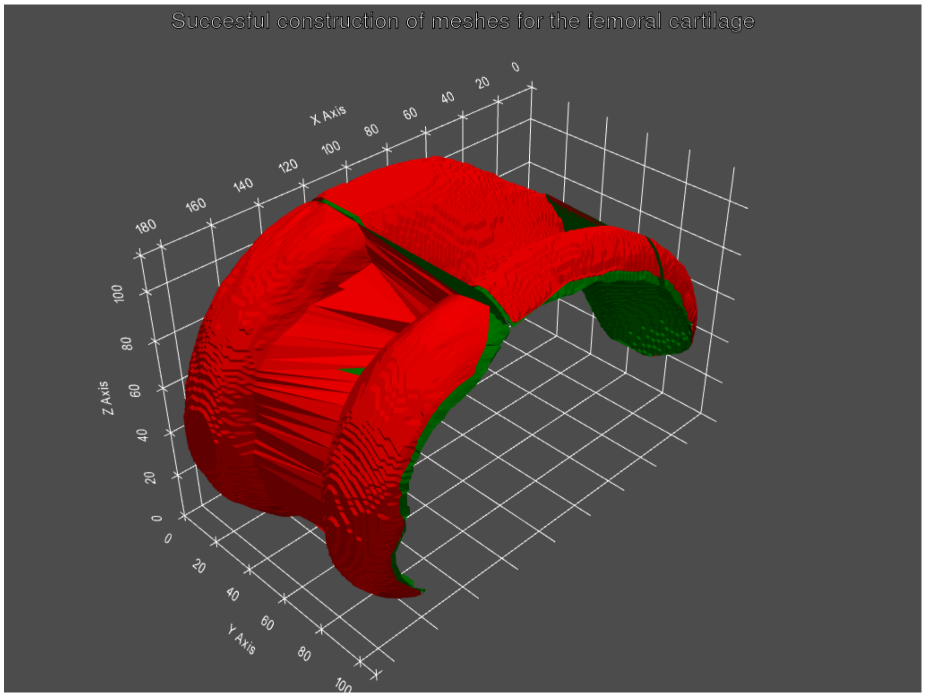

Figure 10: Successful construction of meshes for the femoral cartilage

C Acronyms

ecLF	Mean thickness of the external cartilage subregion of the central part of the lateral femoral condyle
ccLF	Mean thickness of the central cartilage subregion of the central part of the lateral femoral condyle
icLF	Mean thickness of the internal cartilage subregion of the central part of the lateral femoral condyle
icMF	Mean thickness of the internal cartilage subregion of the central part of the medial femoral condyle
ccMF	Mean thickness of the central cartilage subregion of the central part of the medial femoral condyle
ecMF	Mean thickness of the external cartilage subregion of the central part of the medial femoral condyle
aLF	Mean thickness of the anterior cartilage subregion of the lateral femoral condyle
aMF	Mean thickness of the anterior cartilage subregion of the mdeial femoral condyle
cLT	Mean thickness of the central cartilage subregion of the lateral tibia
aLT	Mean thickness of the anterior cartilage subregion of the lateral tibia
eLT	Mean thickness of the external cartilage subregion of the lateral tibia
pLT	Mean thickness of the posterior cartilage subregion of the lateral tibia
iLT	Mean thickness of the internal cartilage subregion of the lateral tibia
cMT	Mean thickness of the central cartilage subregion of the medial tibia
aMT	Mean thickness of the anterior cartilage subregion of the medial tibia
eMT	Mean thickness of the external cartilage subregion of the medial tibia
pMT	Mean thickness of the posterior cartilage subregion of the medial tibia
iMT	Mean thickness of the internal cartilage subregion of the medial tibia
x.aSD	Standard deviation of the thickness of the respective (x) cartilage subregion
x.aMav	Mean value of the maximum 1% measurements of cartilage thickness of the respective (x) cartilage subregion
x.aMiv	Mean value of the minimum 1% measurements of cartilage thickness of the respective (x) cartilage subregion

References

[Che19] Chetvorno. Normal vectors on a curved surface. `https://en.wikipedia.org/wiki/File:Normal_vectors_on_a_curved_surface.svg`, 2019. [Online, accessed 14-November-2021].

[WE08] Wolfgang Wirth and Felix Eckstein. A technique for regional analysis of femorotibial cartilage thickness based on quantitative magnetic resonance imaging. *IEEE transactions on medical imaging*, 27(6):737–744, 2008.

[Wik21a] Wikipedia contributors. Delaunay triangulation — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Delaunay_triangulation&oldid=1044718653`, 2021. [Online; accessed 14-November-2021].

[Wik21b] Wikipedia contributors. Normal (geometry) — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Normal_(geometry)&oldid=1054386602`, 2021. [Online; accessed 15-November-2021].