Elliptical Fourier Analysis of Tumoroid Morphology using High Resolution Episcopic Microscopy

Marcellus G. Augustine

Abstract

Aims: To develop a method to extract and reconstruct the 2D tumour morphology from a High Resolution Episcopic Microscopy (HREM) dataset.

Methods: A HREM tumoroid dataset was segmented to classify the tumour and background. A slice was chosen from this dataset at random. Using the Canny method, the outline was extracted and equiangularly spaced radii emanating from the centroid were created, which intersected the boundary to produce sample points along the outline. Elliptical Fourier analysis (EFA) was performed on these outline points, and the outline shape was reconstructed. The angular step (AngStp) between radii was increased and decreased by 10% and the resulting reconstructions compared. These reconstructions were done using 3, 50 and the maximum number of harmonics to compare how the shape changes with the number of harmonics used. The number of reconstruction points (P_n) was varied to: 50, 200, 500 and 1000 to see how this affected the shape.

Results: Changing AngStp had almost no effect on the shape. Increasing the harmonics used resulted in the shape incorporating more corners and straight edges, and with $P_n \geq 200$, the outline shape was almost identical.

Conclusion: Overall, EFA resulted in an accurate representation of the outline shape, with the major difference being size, whilst outline differences were subtler.

Introduction

Verbal descriptions of shape use adjectives such as: ‘wide’, ‘round’, ‘symmetrical’, but are ultimately limited by the extent of the vocabulary. Some shapes are so complicated that verbal descriptions are simply insufficient. Thus, the quantification of shape is a superior approach that bypasses these verbal limitations and enables the capture of all information pertaining to shape. The term shape is defined herein as the geometrical information that is retained once location, size and rotational effects are filtered out (1). Shape is one of the characteristics, alongside state, size, orientation, surface, interior and substance, that together define form (2). The precise quantification of a form must be considered as the first step towards identifying the biological processes responsible for alterations to that form (2). Therefore, precise methods of quantifying tumour shape are required for a thorough understanding of tumour form, and the processes underlying its progression.

Aims and Objectives

This project aims to develop a method to extract the 2D tumoroid morphology (TM) from a HREM image, which can in future be extended to extract the 3D TM. This will be an input for an *in-silico* hybrid, multiscale model (FEB3) to investigate the effect of tumour morphology on the tumour’s invasive properties. Such investigations could lead to a deeper understanding of tumour invasion and metastasis, potentially underpinning the development of new treatments. Furthermore, it could lead to treatment strategies based on tumour morphology.

Literature Review

Several methods of extracting an object’s morphology exist. Each requires the fulfilment of certain criteria. Ideally, the method should result in size, shape and structural information being recoverable (2). In addition, the transformations: uniform scaling, reflection, rotation and translation must be invariant (2).

Multivariate Morphometrics (MM)

MM analyses shapes using distances, angles (2) and ratios (3). Whilst MM can result in accurate quantitative representations of geometrically regular shapes, it is not suited for highly irregular shapes (2), which many biological shapes are. This method also precludes outline reconstruction.

Geometric Morphometrics (GM)

GM represents different methods, which normalize against rotation, translation and scale (3) meaning that if these transformations were applied onto the same shapes, they would still be classified as the same. These techniques are generally used for comparing shapes (4), (5), (6), not the extraction and reconstruction of a lone shape.

Fourier Analysis (FA)

FA involves decomposing the outline, using the Fourier series, into a sum of cosines and sines. These waves, termed harmonics, are superimposed to form the outline. Fourier coefficients or Fourier descriptors (FDs) are the coefficients of each sine and cosine term in the series and provide a close correspondence to the outline shape (2).

FA can be used with no landmarks or if the entire boundary is of interest (2). It is more informative than using single measures (e.g. length), as the entire shape can be reconstructed on demand. Lastly, analysis is simple due to the computing power available (2).

When multiple outline points exist at a radius, data loss occurs since only one of these points can be utilised (2), affecting the reconstructed image. FA cannot handle outlines involving multiple intersections (3). Finally, like all Fourier-based methods, it requires periodic functions (closed curves meet this requirement).

Fourier Radius Variation (FRV)

With a closed outline, the radius, r, specifying the distance from the centroid to a boundary point can be expressed as a periodic function of the angle θ (7). This method allows outline reconstruction (7).

FRV cannot handle outlines with multiple intersections of a radius with the outline (7). This could occur with outline concavities and convexities. With more complicated outlines, there may not be a single centre enabling the radius function to be single-valued (8).

Fourier Tangent Angle (FTA)

The outline is represented by the equation (1):

$$\phi(t)^* = \phi(t) - \phi(0) - t$$

$$where:$$

$$t = cumulative\ distance\ along\ curve$$

$$\phi(t) = \text{tangent vector angle at t}$$

$$\phi(0) = tangent\ vector\ angle\ at\ starting\ point$$

FDs are calculated by the least squares method (1) and can be used to reconstruct the shape. FTA works with complex contours, and with unequally-spaced, sparse boundary points (1).

Most biological shapes are closed, however reconstructions with few harmonics often are not closed (8), (1). Employing pseudo-landmarks along the outline avoids this (1). The FTA coefficients have increased noise sensitivity compared to other Fourier methods (8). Lastly, there is a poor convergence of the FTA FDs due to the jagged tangent angles of digital images' contours (1), which could lead to reconstruction inaccuracies.

Elliptical Fourier Analysis (EFA)

EFA represents the outline using semi-landmarks. The whole perimeter is collected, contrasting with other methods which use sparse landmarks, disregarding the inter-landmark information (3). The

harmonics are ellipses. The n^{th} harmonic's centre moves about the perimeter of the $(n-1)^{th}$ ellipse, thus tracing the shape (3). The FDs enable outline reconstruction, and even complex outlines involving intersections can be analysed (1).

EFA can be used, unlike other Fourier methods, to analyse contours which aren't single-valued functions (2). EFA is also useful when describing global form characteristics (2). It bypasses the constraint of needing equally spaced outline points along the outline (2). Therefore, more morphologies can be characterised using EFA than other methods.

EFA struggles with sharp corners and straight edges (3). Moreover, EFA results in increased numerical complexity as it circumvents many of the normal constraints of Fourier methods. However, with the available computing power, the analysis is simple (2).

Zahn-Roskies' Shape Function Fourier Analysis (ZRSF-FA)

ZRSF-FA is an angular function composed of the coordinates of points along a closed border that is expanded as a Fourier series (2). It avoids having to specify a centre from which a series of radii emanate (9). The outline shape is represented by the series of angular turns needed to move along the outline in equidistant steps and return to the initial point (9).

The ZRSF can recognise the same shape despite translation, rotation and scale changes (2). The FDs calculated contain all the outline information, enabling accurate reconstruction. Also, this method can be used with any boundary, regardless of complexity (9).

However, the series often needs to be truncated because of practical considerations, meaning the reconstructed curves may not close (2). Also, ZRSF-FA coefficients are more sensitive to noise (2), raising doubts about the reconstruction accuracy. Lastly, equally spaced landmarks along the outline are essential (9), and these can be difficult to compute.

Eigenshape Analysis (EA)

Eigenshapes are eigenvectors which represent outline deformation patterns. EA's premise is it is more efficient to treat the ZRSF's angular terms as variables, than obtaining the FDs from them (9). EA is the

derivation of a set of empirical orthogonal shape functions (9), and represents the outline as geometrically equivalent shape functions (9).

The difference between EA and Fourier methods is that the former empirically determines the optimal orthogonal functions for the data, whilst Fourier methods choose them from a limited set of periodic functions (9).

The analysis is more efficient than the ZRSF-FA, as when the eigenvectors are extracted from both, more are required to describe shape variation when using ZRSF-FA, than EA (9), meaning there are fewer variables to analyse. Moreover, EA recognises the outline shape with superior clarity than EFA (9) but with increasing harmonics, this could change as more detail could be captured. Another advantage is the reconstructions are rougher and more asymmetric (9) meaning that more fine detail is captured (some of these are not due to noise), whilst Fourier reconstructions generally ignore these (except at higher harmonics). EA can also deal with the complex outlines which FRV cannot.

EA requires equally spaced semi-landmarks, which can be difficult to obtain. Traditionally, 100 semi-landmarks are used; this number provides an accurate representation of the shape and reduces the contribution of foreign material on the sample (9).

Methods

MATLAB R2018a was utilised to write scripts to detect the tumoroid edge, obtain the sample points, and perform EFA (see Appendix).

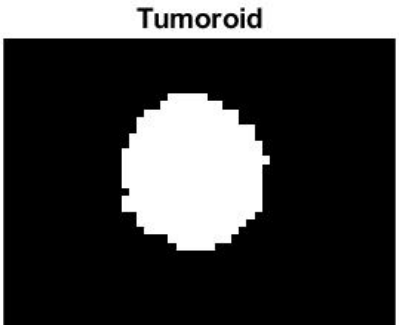

Figure 1: Tumoroid slice reconstructed.

Edge Detection

Firstly, the HREM dataset was automatically segmented into two classes: tumoroid and background, using ImageJ's machine learning plugin: Trainable Wecka Segmentation v3.2.20.

A segmented slice was then selected at random and cropped to produce a single tumour cell cluster (see Figure 1).

Connected component (CC) analysis was performed on the slice and the centroid of the largest CC was obtained. Next, the Canny method was employed to detect the edge of this CC, resulting in a binary image array where the only 1s were those representing boundary points (see Figure 2).

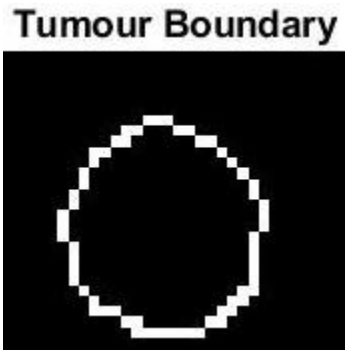

Figure 2: Tumour boundary outline.

Obtaining Sample Points

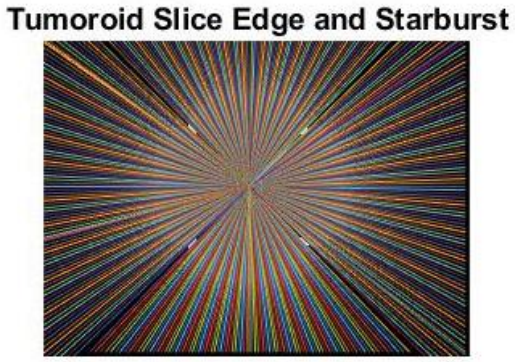

Figure 3: Representation of the radii from the centroid, intersecting the outline to provide the sample points.

Using the image size and the centroid, several radii were drawn from the centroid, to the image array edge, to create a 'starburst' (see Figure 3). Each radius was equiangularly spaced ($AngStp = \pi/180$ radians), from 0 to 2π. The intensity values of 1 along each radius were obtained, alongside their corresponding spatial coordinates. These corresponded to points where the radii intersected the tumoroid boundary, and were sampled for the EFA.

Elliptical Fourier Analysis

EFA was performed using the Elliptical Fourier shape descriptors add-on (10). The forward transform function requires the arguments: shape outline (the sample points), the number of harmonics, n, and whether one wants to normalise for the size and orientation (10). To see the difference caused by varying n, the forward transform was performed with the maximum number of harmonics, N, as defined by the Nyquist frequency requirements ($N < \frac{P_S}{2}$, where $P_S = number\ of\ sampled\ points$). When $n < N$, the harmonic's wavelength is less than twice the distance between sample points, so the harmonics are undetectable and aliasing occurs as they are incorporated into lower frequency harmonics (2).

Once the shape spectrum for the outline was generated using the forward transform, the reverse transform function was performed to reconstruct the outline. This required the FDs, n and P_n as input arguments (10). This function was used with: N harmonics to compare the effects of varying AngStp, constant P_n but varying n to see how the outline changes with different harmonics and constant n but varying P_n to see the effect of different numbers of reconstruction points on the reconstructed shape.

Results

Effect of Angular Step

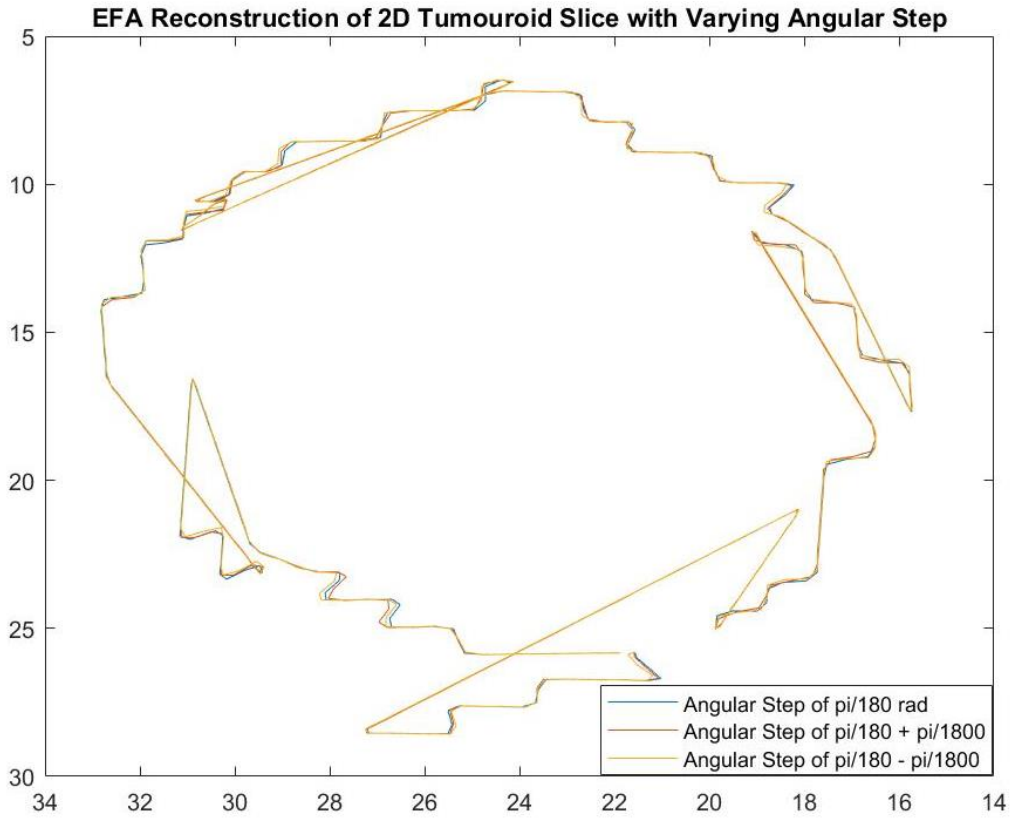

Figure 4: The difference in outline shape reconstruction when AngStp is changed.

EFA requires points sampled along the shape's outline. This was achieved through obtaining the points of intersection of several equiangularly spaced radii with the shape's outline. To ascertain the effect of changing AngStp, the outline shape was reconstructed (using N harmonics) with

$$AngStp = \begin{cases} \frac{\pi}{180} - \frac{\pi}{1800} \\ \frac{\pi}{180} \\ \frac{\pi}{180} + \frac{\pi}{1800} \end{cases}$$

and these reconstructions were superimposed onto one plot (see Figure 4).

A visual examination shows that these reconstructions are extremely similar. To get an idea of how different the reconstructions are, the residual for the AngStp of $\pi/180$ and $\pi/180 + 10\%$ was calculated as $0.0941\mu m$, and with AngStp equalling $\pi/180$ and $\pi/180 - 10\%$ it was calculated as $0.1047\mu m$.

Effect of Varying Number of Harmonics

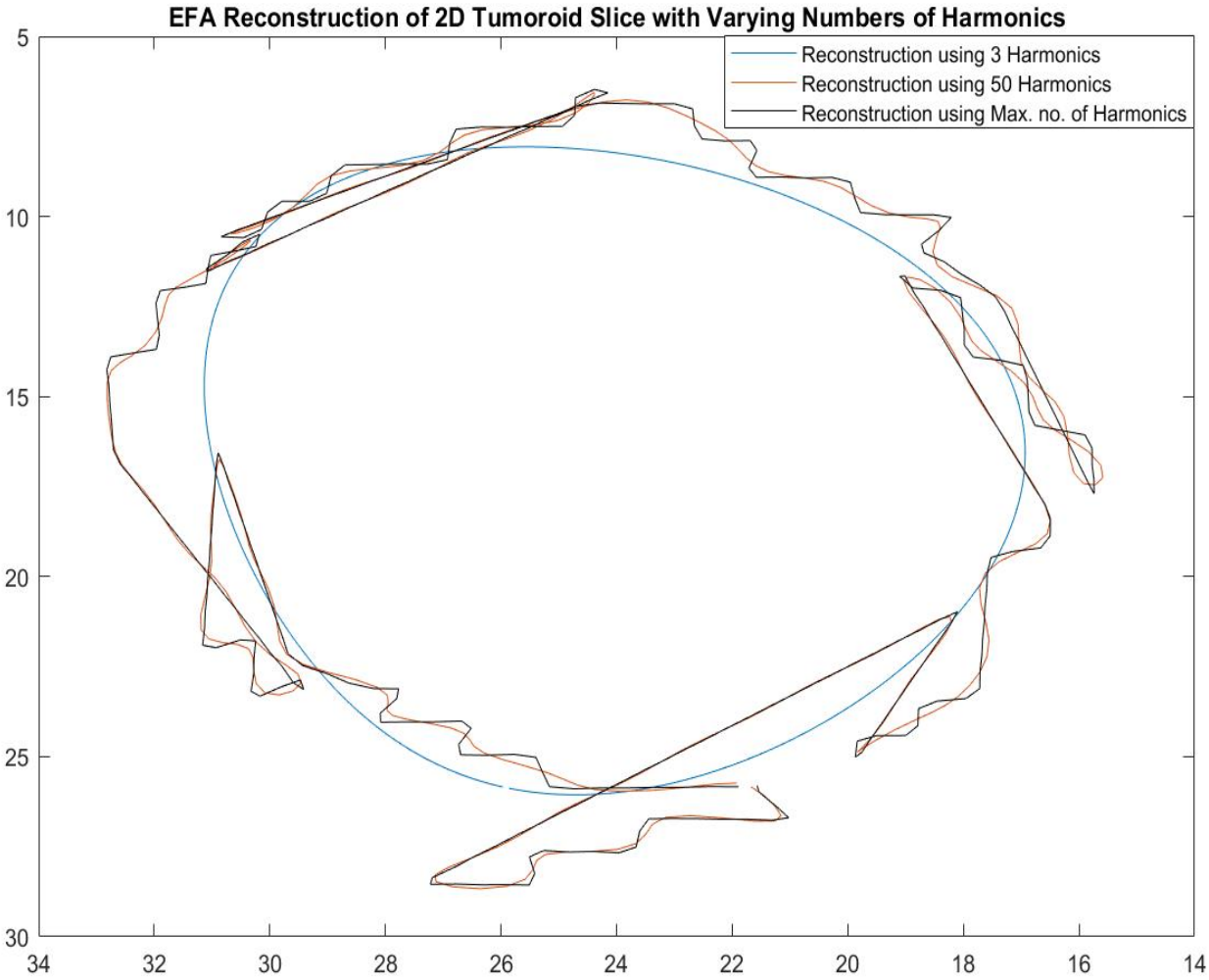

Figure 5: Comparison of how the tumoroid shape reconstruction changes when 3, 50 and the maximum number of harmonics are used.

As n increases, so too does the shape information captured by the analysis. To ascertain how the reconstruction varied with different n values, the reconstructions with $n = \begin{cases} 3 \\ 50 \\ N \end{cases}$ were superimposed (see Figure 5).

Effect of Varying Number of Reconstruction Points

The reconstructions with $P_n = \begin{cases} 50 \\ 200 \\ P_S \\ 500 \\ 1000 \end{cases}$ were superimposed in one plot (see Figure 6).

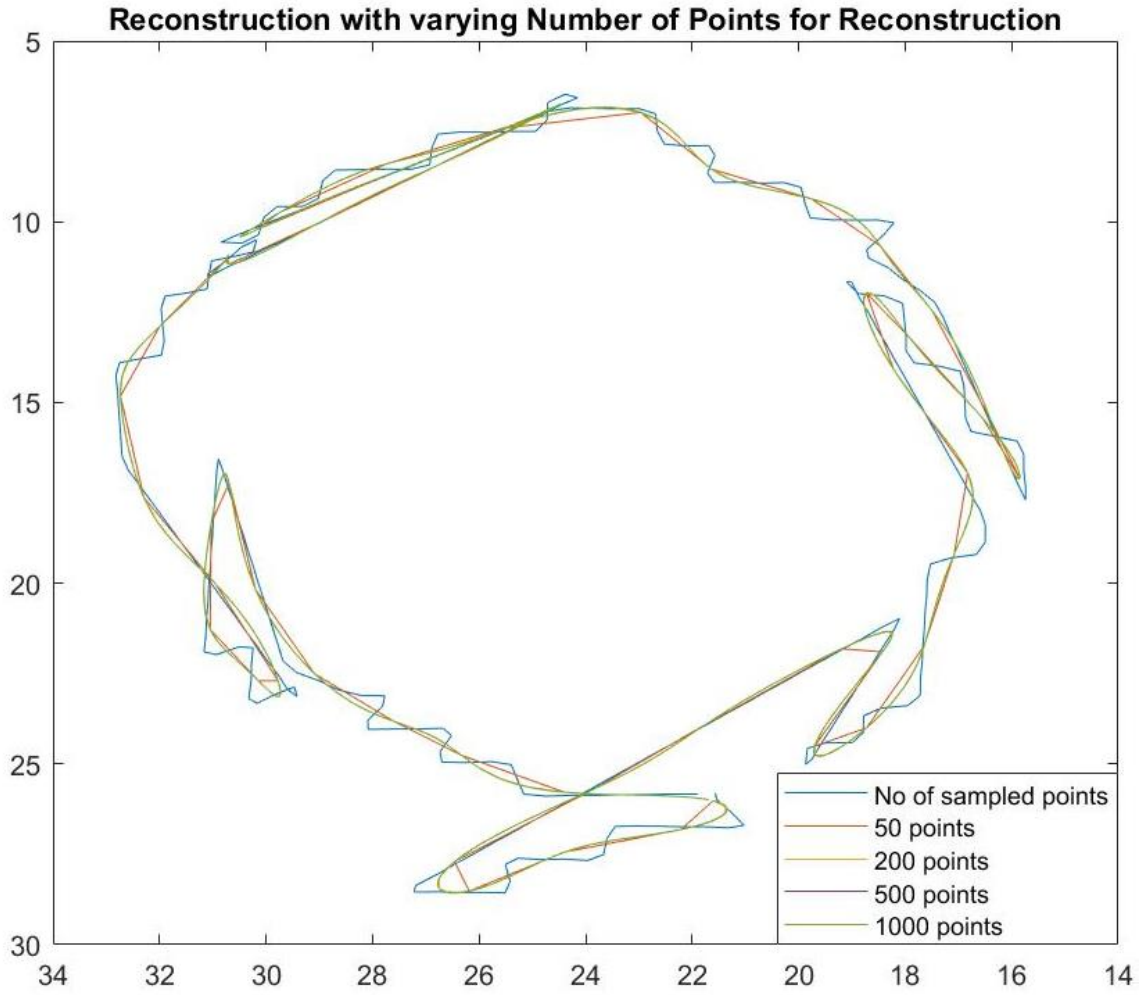

Figure 6: Comparison of outline shape reconstructions varying P_n.

Comparison of Sample with Reconstruction

To visually assess the similarity between the sampled outline and the reconstructed outline, the two were superimposed (see Figure 7). The residual between the object outline and the reconstructed shapes was: $4.0040\mu m$.

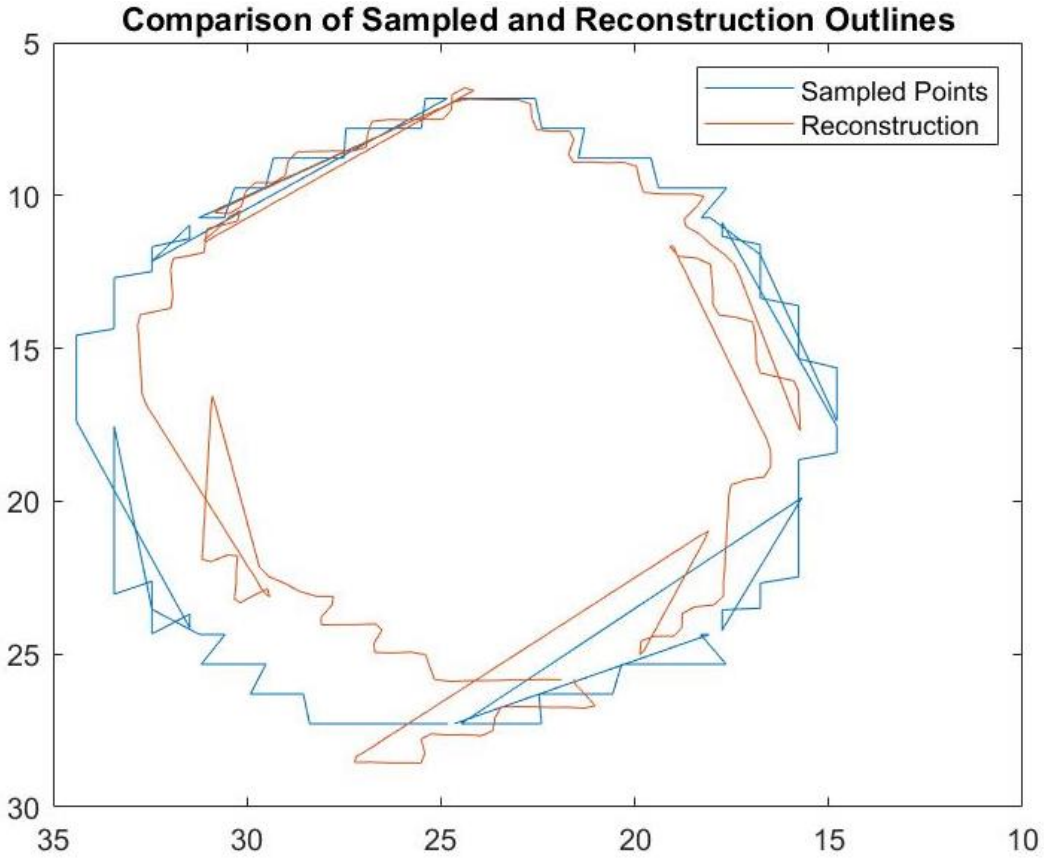

Figure 7: Comparison of the sampled and reconstructed outlines, using the maximum number of harmonics.

Discussion

Figure 4 clearly shows that visually there is almost no difference in the reconstructions when AngStp is varied by $+/-10\%\ of\ \pi/180$ radians. Only at miniscule portions of the outlines do the reconstructions vary. This means that EFA is not hugely affected by AngStp. Furthermore, when the AngStp was changed to +/-10% of $\pi/180$ radians, P_S fell from 318 (at $\pi/180$ rad) to 292 (at +10%) and 356 (at -10%). Thus, even with P_S changing, the reconstructions are still hugely similar. The residuals of $0.0941\mu m$ and $0.1047\mu m$, being small, enunciate this similarity. To prevent P_n influencing the

shape reconstructions, P_n was set to 318, which equalled to P_S, when $AngStp = \pi/180$ radians. As n varies, the level of detail varies, and so n was set to N. Since P_S varied with AngStp, P_n was set to the minimum value of P_S, resulting in $N = 145$ harmonics.

Figure 5 depicts how the reconstruction varied when different values of n, were used. At lower harmonics, the outline shape is more rounded. This is due to EFA using ellipses to reconstruct the outline, meaning that it is great at capturing curves, but is only able to capture corners and straight lines well at higher harmonics (3). When $n = 3$, the shape is a distorted ellipse. As n increases to 50, the tumoroid outline shape is more recognisable, but it contains far more rounded invaginations and projections than when $n = N$. When generating these shapes, P_n was kept constant at P_S (which was 318) to prevent this variable having an effect. Furthermore, for all these reconstructions, $AngStp = \pi/180$ radians.

Figure 6 shows how the reconstructed shapes are affected by the value of P_n. When $P_n = 50$ points, the reconstruction is very rounded and clearly lacks the jagged shape detail present at greater P_n values. $P_S = 318$ points, and when $P_n \geq 200$ points, the outline shapes are almost the same with only extremely minor differences between them. 25 harmonics were used, corresponding to N when $P_S = 50$ (and so the lowest N for the points used), thus preventing aliasing errors by using $n > N$. Once again, $AngStp = \pi/180$ radians, ensuring that P_S remained constant. It was decided to make $P_n = P_S$, as this value clearly resulted in an outline shape that was able to capture the curved aspects, the corners and the straight edges of the original outline.

Figure 7 shows the comparison of the original outline shape with the reconstructed version, with $n = N = 158$ harmonics, and $P_n = P_S$. The predominant difference between the two is that the reconstructed outline shape is smaller than the original. In addition, there are certain portions where the reconstruction depicts undulations, whilst the original shows these as a straight line. The latter is only true for small portions of the outline, and if the size difference is ignored, there is generally a pretty good agreement between the two. The overlapping regions which can be seen in both outlines could be due to how the points were ordered when connecting them. When the sampled points are plotted as a scatter plot, the overlapping portions are to a far lesser extent than in the line plot version (compare Figure 7 with Figure 8). The residual for these boundaries was $4.0040\mu m$, which suggests a fairly large difference between the two. From visual examination, it is clear the major contributor to is the size difference between the original and the reconstruction. EFA is a relatively good method for analysing and reconstructing the 2D tumoroid shape, as the size difference owing to uniform scaling does not contribute to the shape (1), and the other differences are much subtler.

The larger n is, the more shape detail reconstructed, but at higher harmonics, the signal-to-noise ratio is smaller (7), and so some of the detail could correspond to noise and not to the actual shape detail from the image. Therefore, the Nyquist frequency was used to choose the greatest number of harmonics such that the reconstruction captures much of the original detail but is not hugely corrupted by noise. The reconstruction using N harmonics is the best fit to the shape (7).

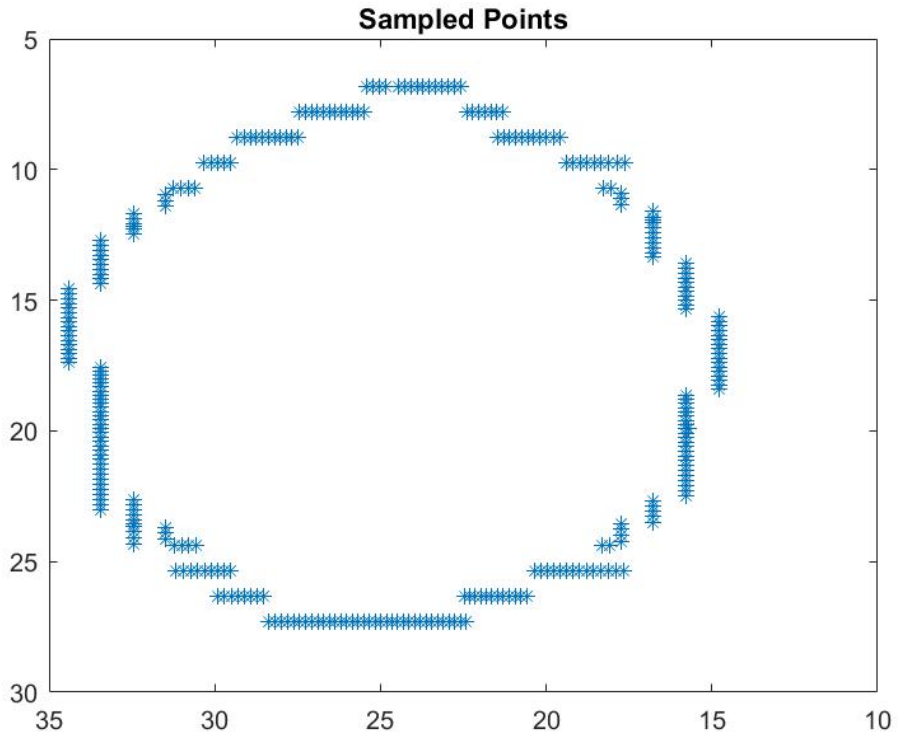

Figure 8: The sampled points plotted as a scatter plot.

Since there were no comparisons involved, the analysis was not normalised for size or orientation. Also, it was found that when these factors were normalised, the reconstruction was much smaller, whilst the outline shape is very similar to the non-normalised versions (see Figure 9).

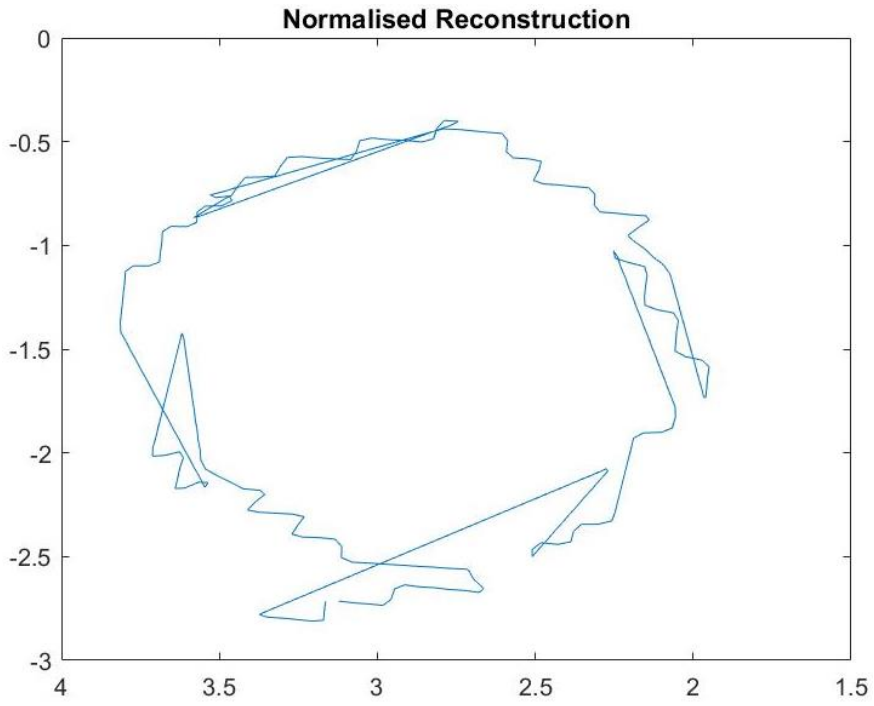

Figure 9: Reconstruction normalised for size and orientation.

With Fourier analysis, the radii should emanate from the centroid, as the positions of these radii remain invariant to rotation (2). Furthermore, using other centres can result in erroneous FDs which are not linked to the shape outline (2). Therefore, the centroid was used as the centre. As a result, the FDs obtained and hence the reconstruction is accurate or at least has avoided the pitfalls associated with using arbitrary centres.

Another problem faced was that some radii intersected the outline multiple times and so there were several sample points instead of just one, due to overlapping regions in the original shape. To get around this, it was decided that the first point of intersection should be sampled at the expense of the others. This means that some of the detail would be lost, and certain parts of the outline would not be captured and replicated accurately. These inaccuracies might explain the differences between the original and the reconstruction.

Conclusion

The reconstructed shape was a relatively accurate representation of the original outline. However, there were subtle differences, including a decrease in size of portions of the reconstruction. Any outlines can be accurately analysed by EFA when enough harmonics are used (1), and if the outline is smooth enough between the sample points (7). Also, when EFA is coupled with the continuous wavelet transform (CWT), it can provide greater detail regarding corners or complex curvatures by using wavelets that focus on specific locations (3). In the future, this method will be extended to 3D tumoroids. More consideration will be given to dealing with overlapping regions to prevent the loss of data that occurred with the strategy employed here. Also, the potential combination of EFA and CWT (or other alternatives) shall be evaluated to obtain an optimal method which improves the level of detail captured and reconstructed.

Acknowledgements

I want to express my utter gratitude to my supervisors: Dr Claire Walsh for her continued assistance throughout this project, Dr Peter Wijeratne for his guidance, and Dominic Giles for his constant help and support.

Bibliography

1. *Geometric morphometrics in entomology: Basics and applications.* **Tatsuta, Haruki, Takahashi, Kazuo H and Sakamaki, Yositaka.** December 2017, Entomological Science, Vol. 21, pp. 164 - 184.

2. **Lestrel, Peter E.** *Fourier Descriptors and their Applications in Biology.* s.l. : Cambridge University Press, 1997. 9780511529870.

3. *Elliptical Fourier analysis: fundamentals, applications and value for forensic anthropology.* **Caple, Jodi, Byrd, John and Stephan, Carl N.** Fedruary 2017, International Journal of Legal Medicine, Vol. 131, pp. 1675-1690.

4. *Extensions of the Procrustes Method for the Optimal Superimposition of Landmarks.* **Rohlf, F. James and Slice, Dennis.** 1, s.l. : Taylor & Francis, Ltd, March 1990, Systematic Zoology, Vol. 39, pp. 40-59.

5. *Generalized Procrustes Analysis.* **Gower, J. C.** 1, March 1975, Psychometrika, Vol. 40, pp. 33-51.

6. *Advances in Geometric Morphometrics.* **Mitteroecker, Philipp and Gunz, Philipp.** 2, June 2009, Evolutionary Biology, Vol. 36, pp. 235-247. 0071-3260.

7. *Momocs: Outline Analysis Using R.* **Bonhomme, Vincent, et al.** 13, February 2014, Journal of Statistical Software, Vol. 56.

8. *A comparison of Fourier methods for the description of wing shape in mosquitoes (Dipthera: Culicidae).* **Rohlf, James F and Archie, James W.** 3, 1984, Systematic Zoology, Vol. 33, pp. 302-317.

9. **MacLeod, Norm.** PalaeoMath 101: Going Round the Bend: Eigenshape Analysis I. [ed.] A. J McGowan. *Palaeontology Newsletter.* July 2012. Vol. 80.

10. **Thomas, David.** File Exchange: Elliptcal Fourier shape descriptors. *MathWorks.* [Online] October 23, 2006. https://uk.mathworks.com/matlabcentral/fileexchange/12746-elliptical-fourier-shape-descriptors?s_tid=FX_rc2_behav.

Appendix

```
%% Elliptical Fourier Analysis on HREM Tumoroid Slice
%%%%% SCRIPT BY MARCELLUS AUGUSTINE %%%%%
%%%%% marcellus.augustine.17@ucl.ac.uk %%%%%

close all
clear
clc

%% Generate Stack
addpath('D:\Tumouroids\Weka\cropped for 3D\full cluster')

% Read image into MATLAB
Imstack(:,:)=imread('3Dcropp0008.tif');
```

```
Imstack = imbinarize(Imstack);

%% Show image
figure
Imstack = imcomplement(Imstack);
imshow(Imstack(:,:),[0 max(max(Imstack(:,:)))]);
hold on
title('Tumoroid')
hold off
%% Connected Component Analysis
CC_Imstack = bwconncomp(Imstack);

% Remove all but the largest connected component
numPixels = cellfun(@numel,CC_Imstack.PixelIdxList);
[biggest,idx] = max(numPixels);
sz1 = size(CC_Imstack.PixelIdxList);

for n = 1:sz1(2)
   if numPixels(n) ~= biggest
      Imstack(CC_Imstack.PixelIdxList{n}) = 0;
   end
end

% redefine the connected components (should only produce one):
CC_Imstack = bwconncomp(Imstack);

% Find the Centre of largest Connected Component
stats_padedg = regionprops(CC_Imstack,'Centroid');
centre = stats_padedg.Centroid;
%% Detect Boundary:
padedg = edge(Imstack, 'approxcanny', 0.5);

%% Display
%Display the edge using patch 3D add-on
figure
edgypatch = imshow(padedg);
title('Tumoroid Slice Edge and Starburst')

clear biggest
clear idx
clear numPixels
clear CC_Imstack
clear testim

%% Obtain Sample Points

% Define Angular step
Angstep = pi/180;

% Define Variables
 CentreX = centre(1,1);
 CentreY = centre(1,2);
 [Y_size,X_size] = size(Imstack);

clear centre
clear n
clear stats_padedg
```

```
clear sz
clear sz1

C_0till45 = zeros(46,2);
counter = 0;
for i = 0:Angstep:pi/4
   if (0 < i) && (i < pi/4)
      %Work out coordinates as the limits on image edge until which...
      ...intensity values are retrieved
      X = ((Y_size-CentreY)*tan(i)) + CentreX;
      %Retrieve intensity values and corresponding X and Y spatial...
      ...coordinates
      [impX,impY,impre] = improfile(padedg,[CentreX X],[CentreY Y_size]);
      %Check if there are any NaNs in impre
       Nany = isnan(impre);
      %Find the indexing in Nany where there are NaNs
      [nrow,~] = find(Nany);
      %Set the value of the NaNs to equal 0
      impre(nrow) = 0;
      %Obtain the indexing within the array impre where the intensity...
      ...values are 1
      [row, ~] = find(impre);
      %Obtain the coordinates at which the intensity values are 1
      Xg = impX(row);
      Yg = impY(row);
      %Store coordinates in a vector returned by B
      B = [Xg Yg];
      %If there are any points stored in B, work out their coordinates...
      ... and store them in C_0till45
      if numel(B)>= 2
      counter = counter + 1;
      C_0till45(counter,1) = B(1,1);
      C_0till45(counter,2) = B(1,2);
      end
   end
end
disp('Done 0 - 45')

C_at45 = zeros(1,2);
for i = pi/180:Angstep:pi/4
   if i == pi/4
      [impX,impY,impre] = improfile(padedg,[CentreX X_size],[CentreY Y_size]);
       Nany = isnan(impre);
      [nrow,~] = find(Nany);
      impre(nrow) = 0;
      [row, ~] = find(impre);
      Xg = impX(row);
      Yg = impY(row);
      B = [Xg Yg];
      if numel(B)>= 2
      C_at45(1,1) = B(1,1);
      C_at45(1,2) = B(1,2);
      end
   end
end
disp('Done at 45')
```

```
C_45till90 = zeros(46,2);
counter = 0;
for i = pi/4:Angstep:pi/2
   if (pi/4 <= i) && (i < pi/2)
      Y = ((X_size - CentreX)*tan(i - (pi/4))) + CentreY;
      [impX,impY,impre] = improfile(padedg,[CentreX X_size],[CentreY Y]);
      Nany = isnan(impre);
      [nrow,~] = find(Nany);
      impre(nrow) = 0;
      [row, ~] = find(impre);
      Xg = impX(row);
      Yg = impY(row);
      B = [Xg Yg];
      if numel(B)>= 2
      counter = counter + 1;
      C_45till90(counter,1) = B(1,1);
      C_45till90(counter,2) = B(1,2);
      end
   end
end
disp('Done 45 - 90')

C_at90 = zeros(1,2);
for i = pi/2:Angstep:3*pi/2
   if i == pi/2
      [impX,impY,impre] = improfile(padedg,[CentreX X_size],[CentreY CentreY]);
      Nany = isnan(impre);
      [nrow,~] = find(Nany);
      impre(nrow) = 0;
      [row, ~] = find(impre);
      Xg = impX(row);
      Yg = impY(row);
      B = [Xg Yg];
      if numel(B)>= 2
      C_at90(1,1) = B(1,1);
      C_at90(1,2) = B(1,2);
      end
   end
end
disp('Done at 90')

C_90till135 = zeros(46,2);
counter = 0;
for i = pi/2:Angstep:3*pi/4
   if (pi/2 < i) && (i < 3*pi/4)
      Y = CentreY - ((X_size - CentreX)*tan(i- (pi/2)));
      [impX,impY,impre] = improfile(padedg,[CentreX X_size],[CentreY Y]);
       Nany = isnan(impre);
      [nrow,~] = find(Nany);
      impre(nrow) = 0;
      [row, ~] = find(impre);
      Xg = impX(row);
      Yg = impY(row);
      B = [Xg Yg];
      if numel(B)>= 2
      counter = counter + 1;
      C_90till135(counter,1) = B(1,1);
```

```
        C_90till135(counter,2) = B(1,2);
        end
    end
end
disp('Done 90 - 135')

C_at135 = zeros(1,2);
for i = 3*pi/4:Angstep:pi
    if i == 3*pi/4
        [impX,impY,impre] = improfile(padedg,[CentreX X_size],[CentreY 0]);
        Nany = isnan(impre);
        [nrow,~] = find(Nany);
        impre(nrow) = 0;
        [row, ~] = find(impre);
        Xg = impX(row);
        Yg = impY(row);
        B = [Xg Yg];
        if numel(B)>= 2
        C_at135(1,1) = B(1,1);
        C_at135(1,2) = B(1,2);
        end
    end
end
disp('Done at 135')

C_135till180 = zeros(46,2);
counter = 0;
for i = 3*pi/4:Angstep:pi
    if (3*pi/4 < i) && (i < pi)
        X = (CentreY*tan(i - (3*pi/4))) + CentreX;
        [impX,impY,impre] = improfile(padedg,[CentreX X],[CentreY 0]);
        Nany = isnan(impre);
        [nrow,~] = find(Nany);
        impre(nrow) = 0;
        [row, ~] = find(impre);
        Xg = impX(row);
        Yg = impY(row);
        B = [Xg Yg];
        if numel(B)>= 2
        counter = counter + 1;
        C_135till180(counter,1) = B(1,1);
        C_135till180(counter,2) = B(1,2);
        end
    end
end
disp('Done 135 - 180')

C_at180 = zeros(1,2);
for i = pi:Angstep:5*pi/4
    if i == pi
        [impX,impY,impre] = improfile(padedg,[CentreX X],[CentreY 0]);
        Nany = isnan(impre);
        [nrow,~] = find(Nany);
        impre(nrow) = 0;
        [row, ~] = find(impre);
        Xg = impX(row);
        Yg = impY(row);
```

```
        B = [Xg Yg];
        if numel(B)>= 2
        C_at180(1,1) = B(1,1);
        C_at180(1,2) = B(1,2);
        end
    end
end
disp('Done at 180')

C_180till225 = zeros(46,2);
counter = 0;
for i = pi:Angstep:5*pi/4
    if (pi < i) && (i < 5*pi/4)
        X = CentreX - (CentreY*tan(i - pi));
        [impX,impY,impre] = improfile(padedg,[CentreX X],[CentreY 0]);
        Nany = isnan(impre);
        [nrow,~] = find(Nany);
        impre(nrow) = 0;
        [row, ~] = find(impre);
        Xg = impX(row);
        Yg = impY(row);
        B = [Xg Yg];
        if numel(B)>= 2
        counter = counter + 1;
        C_180till225(counter,1) = B(1,1);
        C_180till225(counter,2) = B(1,2);
        end
    end
end
disp('Done 180 - 225')

C_at225 = zeros(1,2);
for i = 5*pi/4:Angstep:3*pi/2
    if i == 5*pi/4
        [impX,impY,impre] = improfile(padedg,[CentreX 0],[CentreY 0]);
        Nany = isnan(impre);
        [nrow,~] = find(Nany);
        impre(nrow) = 0;
        [row, ~] = find(impre);
        Xg = impX(row);
        Yg = impY(row);
        B = [Xg Yg];
        if numel(B)>= 2
        C_at225(1,1) = B(1,1);
        C_at225(1,2) = B(1,2);
        end
    end
end
disp('Done at 225')

C_225till270 = zeros(46,2);
counter = 0;
for i = 5*pi/4:Angstep:3*pi/2
    if (5*pi/4 < i) && (i < 3*pi/2)
        Y = CentreY - (CentreX*tan(i - (5*pi/4)));
        [impX,impY,impre] = improfile(padedg,[CentreX 0],[CentreY Y]);
        Nany = isnan(impre);
```

```
        [nrow,~] = find(Nany);
        impre(nrow) = 0;
        [row, ~] = find(impre);
        Xg = impX(row);
        Yg = impY(row);
        B = [Xg Yg];
        if numel(B)>= 2
        counter = counter + 1;
        C_225till270(counter,1) = B(1,1);
        C_225till270(counter,2) = B(1,2);
        end
    end
end
disp('Done 225 - 270')

C_at270 = zeros(1,2);
for i = 3*pi/2:Angstep:7*pi/4
    if i == 3*pi/2
        [impX,impY,impre] = improfile(padedg,[CentreX 0],[CentreY CentreY]);
        Nany = isnan(impre);
        [nrow,~] = find(Nany);
        impre(nrow) = 0;
        [row, ~] = find(impre);
        Xg = impX(row);
        Yg = impY(row);
        B = [Xg Yg];
        if numel(B)>= 2
        C_at270(1,1) = B(1,1);
        C_at270(1,2) = B(1,2);
        end
    end
end
disp('Done at 270')


C_270till315 = zeros(46,2);
counter = 0;
for i = 3*pi/2:Angstep:7*pi/4
    if (3*pi/2 < i) && (i < 7*pi/4)
        Y = (CentreX*tan(i-(3*pi/2))) + CentreY;
        [impX,impY,impre] = improfile(padedg,[CentreX 0],[CentreY Y]);
        Nany = isnan(impre);
        [nrow,~] = find(Nany);
        impre(nrow) = 0;
        [row, ~] = find(impre);
        Xg = impX(row);
        Yg = impY(row);
        B = [Xg Yg];
        if numel(B)>= 2
        counter = counter + 1;
        C_270till315(counter,1) = B(1,1);
        C_270till315(counter,2) = B(1,2);
        end
    end
end
disp('Done 270 - 315')
```

```
C_at315 = zeros(1,2);
for i = 7*pi/4:Angstep:2*pi
   if i == 7*pi/4
      [impX,impY,impre] = improfile(padedg,[CentreX -X_size],[CentreY Y_size]);
      Nany = isnan(impre);
      [nrow,~] = find(Nany);
      impre(nrow) = 0;
      [row, ~] = find(impre);
      Xg = impX(row);
      Yg = impY(row);
      B = [Xg Yg];
      if numel(B)>= 2
      C_at315(1,1) = B(1,1);
      C_at315(1,2) = B(1,2);
      end
   end
end
disp('Done at 315')

C_315till360 = zeros(46,2);
counter = 0;
for i = 0:Angstep:pi/4
   if (0 < i) && (i < pi/4)
      j = i + 7*pi/4;
      X2 = ((Y_size-CentreY)*tan(i)) - CentreX;
      [impX,impY,impre] = improfile(padedg,[CentreX -X2],[CentreY Y_size]);
       Nany = isnan(impre);
      [nrow,~] = find(Nany);
      impre(nrow) = 0;
      [row, ~] = find(impre);
      Xg = impX(row);
      Yg = impY(row);
      B = [Xg Yg];
      if numel(B)>= 2
      counter = counter + 1;
      C_315till360(counter,1) = B(1,1);
      C_315till360(counter,2) = B(1,2);
      end
   end
end
disp('Done 315 - 360')

C_at360 = zeros(1,2);
for i = 2*pi:Angstep:9*pi/4
   if i == 2*pi
      [impX,impY,impre] = improfile(padedg,[CentreX CentreX],[CentreY Y_size]);
      Nany = isnan(impre);
      [nrow,~] = find(Nany);
      impre(nrow) = 0;
      [row,~] = find(impre);
      Xg = impX(row);
      Yg = impY(row);
      B = [Xg Yg];
      if numel(B)>= 2
      C_at360(1,1) = B(1,1);
      C_at360(1,2) = B(1,2);
      end
```

```
    end
end
disp('Done at 360')

% Remove all 0 values from radius vectors
C_1semiquad1 =  C_0till45(:,1);
C_1semiquad1 = C_1semiquad1(C_1semiquad1 ~= 0);
C_1semiquad2 =  C_0till45(:,2);
C_1semiquad2 = C_1semiquad2(C_1semiquad2 ~= 0);
clear C_0till45
C_2semiquad1 =  C_45till90(:,1);
C_2semiquad1 = C_2semiquad1(C_2semiquad1 ~= 0);
C_2semiquad2 =  C_45till90(:,2);
C_2semiquad2 = C_2semiquad2(C_2semiquad2 ~= 0);
clear C_45till90
C_3semiquad1 =  C_90till135(:,1);
C_3semiquad1 = C_3semiquad1(C_3semiquad1 ~= 0);
C_3semiquad2 =  C_90till135(:,2);
C_3semiquad2 = C_3semiquad2(C_3semiquad2 ~= 0);
clear C_90till135
C_4semiquad1 =  C_135till180(:,1);
C_4semiquad1 = C_4semiquad1(C_4semiquad1 ~= 0);
C_4semiquad2 =  C_135till180(:,2);
C_4semiquad2 = C_4semiquad2(C_4semiquad2 ~= 0);
clear C_135till180
C_5semiquad1 =  C_180till225(:,1);
C_5semiquad1 = C_5semiquad1(C_5semiquad1 ~= 0);
C_5semiquad2 =  C_180till225(:,2);
C_5semiquad2 = C_5semiquad2(C_5semiquad2 ~= 0);
clear C_180till225
C_6semiquad1 =  C_225till270(:,1);
C_6semiquad1 = C_6semiquad1(C_6semiquad1 ~= 0);
C_6semiquad2 =  C_225till270(:,2);
C_6semiquad2 = C_6semiquad2(C_6semiquad2 ~= 0);
clear C_225till270
C_7semiquad1 =  C_270till315(:,1);
C_7semiquad1 = C_7semiquad1(C_7semiquad1 ~= 0);
C_7semiquad2 =  C_270till315(:,2);
C_7semiquad2 = C_7semiquad2(C_7semiquad2 ~= 0);
clear C_270till315
C_8semiquad1 =  C_315till360(:,1);
C_8semiquad1 = C_8semiquad1(C_8semiquad1 ~= 0);
C_8semiquad2 =  C_315till360(:,2);
C_8semiquad2 = C_8semiquad2(C_8semiquad2 ~= 0);
clear C_316till360

% Concatenate all point data into one array:
C_0till45 = cat(2,C_1semiquad1,C_1semiquad2);
clear C_1semiquad1
clear C_1semiquad2
C_45till90 = cat(2,C_2semiquad1,C_2semiquad2);
clear C_2semiquad1
clear C_2semiquad2
C_90till135 = cat(2,C_3semiquad1,C_3semiquad2);
clear C_3semiquad1
clear C_3semiquad2
C_135till180 = cat(2,C_4semiquad1,C_4semiquad2);
```

```
clear C_4semiquad1
clear C_4semiquad2
C_180till225 = cat(2,C_5semiquad1,C_5semiquad2);
clear C_5semiquad1
clear C_5semiquad2
C_225till270 = cat(2,C_6semiquad1,C_6semiquad2);
clear C_6semiquad1
clear C_6semiquad2
C_270till315 = cat(2,C_7semiquad1,C_7semiquad2);
clear C_7semiquad1
clear C_7semiquad2
C_315till360 = cat(2,C_8semiquad1,C_8semiquad2);
clear C_8semiquad1
clear C_8semiquad2

C_points1 = cat(1,C_0till45,C_at45, C_45till90,C_90till135,...
   C_at135,C_135till180,C_at180,C_180till225,C_at225,C_225till270,...
   C_at270,C_270till315,C_at315,C_315till360,C_at360);

%% Elliptical Fourier Analysis and Reconstruction
addpath('C:\Program Files\MATLAB\Useful Matlab Functions\ellipitical fourier shape descriptors')

% Calculate Maximum Number of Harmonics based on Nyquist Frequency
NoOfPoints1 = numel(C_points1)/2;

if rem(NoOfPoints1,2) == 0
   MaxNoHarmonics1 = (NoOfPoints1 / 2) - 1;
else
   MaxNoHarmonics1 = ((NoOfPoints1 - 1)/2) - 1;
end

% Forward Elliptical Fourier Transform
fwd1 = fEfourier(C_points1,MaxNoHarmonics1,0,0);

% Reverse Elliptical Fourier Transform
bckwd1 = rEfourier(fwd1,MaxNoHarmonics,NoOfPoints1);

% Display Reconstruction
figure
plot(bckwd1(:,1),bckwd1(:,2))
hold on
title('EFA Reconstruction of 2D Tumoroid Slice')
view(180,90)
hold off
```