Extracting 3D Parametric Curves from 2D Images of Helical Objects

Chris G. Willcocks, Philip T. G. Jackson, Carl J. Nelson, and Boguslaw Obara*

***Abstract*—Helical objects occur in medicine, biology, cosmetics, nanotechnology, and engineering. Extracting a 3D parametric curve from a 2D image of a helical object has many practical applications, in particular being able to extract metrics such as tortuosity, frequency, and pitch. We present a method that is able to straighten the image object and derive a robust 3D helical curve from peaks in the object boundary. The algorithm has a small number of stable parameters that require little tuning, and the curve is validated against both synthetic and real-world data. The results show that the extracted 3D curve comes within close Hausdorff distance to the ground truth, and has near identical tortuosity for helical objects with a circular profile. Parameter insensitivity and robustness against high levels of image noise are demonstrated thoroughly and quantitatively.**

***Index Terms*—Helical Curves, Shape Analysis, Feature Extraction, Geometry, Modeling, Skeletonization.**

I. Introduction

HELICAL curves occur in many natural and artificial structures. Evolution has exploited such geometrical shapes to provide living organisms, at various scales, with different functions. Similarly, human technology has used helical structures, from micro to macro scale, in a wide range of applications. Some examples of such helical designs in a variety of domains are:

- At a micro scale:
  - Biology and medicine (e.g. *Spirulina* [1], *Spirochaetes* [2], sperm [3], bacterial macrofibers, microtubules, keratin, DNA, dynamin [4]).
  - Nanotechnology (e.g. helical nanostructures, such as: nanosprings and graphitic carbon microtubules [5]).
- At a macro scale:
  - Medicine (e.g. umbilical cord).
  - Biology (e.g. climbing plants, twining vines [6], twisted trees, seashells, Arabidopsis root [7]).
  - Cosmetics industry (e.g. hair [8]).
  - Engineering (e.g. screws, coils, springs, synthetic fiber ropes [9]).

Extracting a representative 3D helical curve of an object has applications in quality control on manufacturing lines, making rapid decisions in high-throughput experimentation in plant sciences, modeling hair in cosmetics, and in the geometric modeling and understanding of microscopic structures in medicine, biology and nanotechnology. A representative 3D helical curve acts as an important descriptor for a complex helical-shaped object, describing its 3D geometric characteristics including tortuosity, local pitch, and local radius. However, most imaging techniques allow only a 2D view of the object, often with significant noise. Many 3D imaging techniques exist but these are usually slow, expensive, difficult to apply correctly and often only applicable to objects of a certain scale. Motivated by the desire to satisfy such use cases without the need for 3D imaging, we present an automatic method in three stages: (1) extract the main structural curve, (2) straighten the image and (3) fit a 3D parametric curve to the straightened image and map it back to the object's original shape.

Our approach works by exploiting the fact that most helical objects show some 'zig-zag' structure in their 2D projection, with the same pitch and radius as the true 3D helix. We detect the peaks on the object's external boundary and construct a helical 3D cubic spline that passes through them, looping above and below the image plane in-between them. By fitting a curve through these guide points we can extract a curve that accurately captures the overall curvature of the helix, including local variations in radius and pitch, neither of which would be possible if we only estimated global radius and pitch parameters.

Throughout our method we use only well-established techniques such as binary morphology, curve fitting and signal smoothing. Our contribution is in the composition of these simple steps into a novel pipeline, and in our definition of the control points that guide the helix. We show the main stages of our pipeline in Figure 1, and summarize it in Table I at the end of the Method section, attributing the steps to their sources where appropriate.

II. Related Work

A. 3D Reconstruction Approaches

There is a large body of research concerned with reconstructing 3D models from 2D data in the fields of computer graphics, geometry modeling, and image processing.

The two most similar approaches to ours, which reconstruct 3D helical curves from 2D data, are by [11] and [8]. The method by [11] involves fitting a piece-wise 3D helix curve to a 2D polyline curve such that its orthogonal projection matches the input. In contrast, [8] focuses on providing a compact characterization of hair geometry using generalized helicoids. They observe coherency amongst orientation and curvature in local neighborhoods of hair strands, and present a method to generate, fit and interpolate hair patterns. In this approach, they fit piece-wise generalized helicoids to unparametrised 3D polyline hair strands, and validate their results using a database of hair samples where each hair is represented by a polyline.

* The authors are with the School of Engineering and Computing Sciences, Durham University, South Road, DH1 3LE, Durham, UK.
* E-mail address: boguslaw.obara@durham.ac.uk

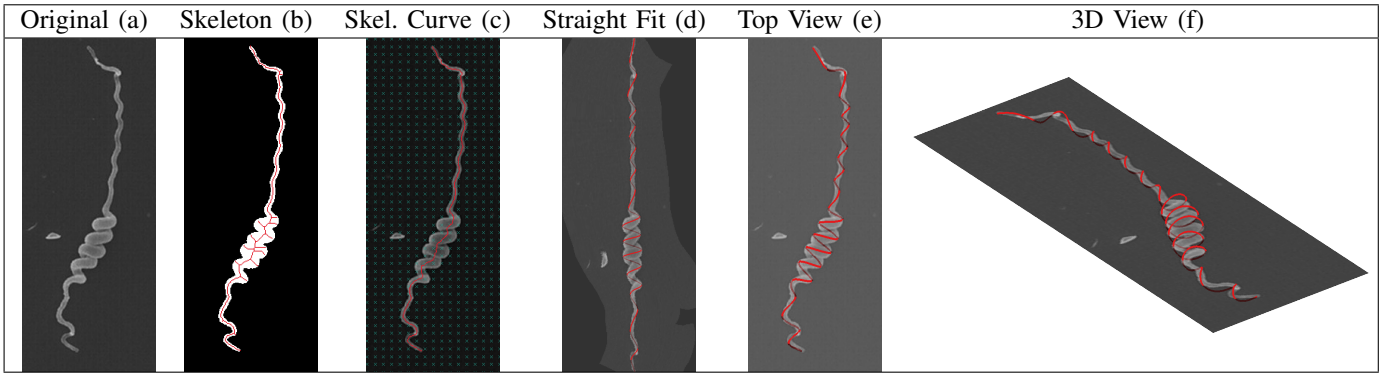


Figure 1: Stages of our pipeline to segment (a-b), straighten (c-d) and extract the 3D helix of a Leptospira image [10] (d-f).

While both [11] and [8]'s methods are equally valid, they require uniformly sampled polylines as input which greatly limits their range of applications in real-world datasets such as encountered in biology, medicine or engineering. Recently, [12] extract a single 3D helix from a noisy and irregularly sampled polyline, but they still do not propose a way to extract such a polyline from raw images which are frequently noisy and self-occluding. The problem of self-occlusion has been examined by [13], who extract a 3D shape from a 2D outline self-occluding sketch. Their approach firsts extracts a 2D skeleton on the sketching plane, where the 3D skeleton is derived and used to generate the 3D object, and they approximate the depth at the self-intersecting points in the 2D skeleton. The idea of using the skeleton to capture topological information of the image gives excellent results, therefore we briefly examine the image skeletonization literature in more detail.

B. Image Skeletonization

Image skeletonization is a large field covering many different methods and techniques. The first approaches iteratively apply morphological thinning by removing boundary pixels that do not affect the local connectedness of a shape until a single pixel wide 'skeleton' remains [14]. These approaches are generally very sensitive to boundary noise, requiring a pruning post-process to remove spurious branches [15]. Other approaches prune edges from the Voronoi diagram of a shape's boundary [16], [17]. Smoother and more robust approaches evolve a wave front over a distance transform from maximal distance points in the object boundary giving a smooth skeleton that is insensitive to boundary noise [18]. These approaches all operate on the shape boundary, e.g. with a binary representation or surface mesh, and then compute a distance image or use a gradient vector field [19]. The results in the literature show that the binary shape boundary provides enough information to describe the central curve of the shape [20]. The difference in our case is that the path of a helical curve passes through prominent features (peaks) of the shape boundary, instead of always being centered as with the topological skeleton. While the topological skeleton can act as a guide for the main curve, we seek to additionally fit our curve through peaks in the helical structure.

C. Curve Fitting and Representation

Fitting curves through data points is a well studied area, where a lot of work has focused on the representation of the underlying curves to provide compact or intuitive parameters and properties. Relevant representations include polyhelices [21], which are a sequence of helical segments defined by curvature, torsion and length, fitting through ordered points. Similarly [22] approximates an input Bezier spline with N-piecewise helix elements, which is used to compactly represent and reconstruct fiberous datasets such as hair, muscular fibers or magnetic field lines of a star. An alternative approach involves fitting helices to a curve within a maximal Hausdorff distance. This is shown by [23] who use an iterative bisecting algorithm to compute bi-helical arcs from one endpoint to produce near optimal bi-helical splines, which smoothly connect 2 helices approximating the curves to the desired accuracy. Instead of trying to fit exactly to datapoints, [24] present a method for computing a planar B-spline curve to approximate a target shape defined by a point cloud. They formulate the curve fitting problem as a nonlinear least squares problem which uses a curvature-based approximation to the true Hessian of the objective function. Furthermore, [25] introduce a weighting for least-squares fitting in combination with a local arc-length approximation of the input to estimate curvature and torsion of planar and spatial curves.

D. Semi-automatic Approaches

Besides curve fitting and skeletonization approaches, there is a large body of research that relies on user interaction to reconstruct 3D curves and objects from 2D images. In particular sketch-based approaches are a popular choice for 3D reconstruction given their simplicity and humans' natural 2D drawing ability. An interesting method was proposed by [26], who let the user draw the curve from the current view perspective, then draw the shadow on a floor plane, which is used to generate the 3D curve used in modeling applications. Similarly, [27] let the user sketch an initial 2D curve, but they compose it into bounding boxes by zero-crossings of the principal axis. They use the bounding boxes to infer properties of a generated 3D helix, however this leads to some poor quality approximation. A more complex solution is considered by [28], who interpret drawings of 2D line segments by minimizing the

entropy of angle distribution between line segments in a 3D wireframe using a genetic algorithm. While these approaches have good results, their applications are limited by their need for user-generated input data, such as a sketch image or polyline.

E. Image-based Approaches

Extracting curves from grayscale image data is much more challenging than from binary image or polyline data, given that real-world image datasets often have weak and broken boundaries and/or severely noisy, i.e. blurred or distorted edges, while still representing shapes with complex topology and large curvature variations. One of the most significant contributions was given by [29], who present a method for detecting curvilinear structures in images using scale-space analysis on the profile of asymmetrical bar-shaped lines. Their method is robust, however requires some parameter to be set on the line widths capturing a range of scale-space features. More noisy images can be improved in a pre-processing stage, for example using a technique by [30] who calculate the eigenvalues of the Hessian matrix at every pixel at multiple scale-spaces to enhance the vessel features.

There are approaches that also try and fit curves to edges in the image, however they rely on more than one view of the data to achieve good results. The work by [31] relies on edge detection and template matching to extract data points on road lines, then use colinearity between multiple-images to determine 3D data points before fitting a least-squares B-spline in the 2D and 3D space. Similarly [32] propose an approach to reconstruct a 3D curve from two different views of the data, however they deform an elastic curve in order to adapt to the projections of the two data views.

In diffusion MRI, [33] introduce 'co-helicity' to estimate local trace, tangent, curvature and torsion fields of curves in 3D images. This is an extension of 'co-circularity' in the 2D case [34] and has applications in streamline fibre tracking, which has been demonstrated in 3D brain datasets. While such approaches produce high-quality curves, they rely on input datasets of the same dimension.

There is a large body of local regression literature [35] which deals with the problem of separating a signal from noise, with excellent results in cleaning and reproducing a good curve from a large amount of variation. In particular, the popular LOESS approach uses a quadratic model to do the local fitting [36], which can be further improved by a robust weighting scheme to reduce the influence of outliers.

F. Proposed Approach

In our approach, we detect the object's centerline from its binary silhouette and interpret undulations in the centerline as corresponding to helical revolutions. We infer depth information by assuming the helix has a circular profile, and output a 3D helical cubic spline whose diameter and pitch adapt locally to the object's width and pitch. The 3D curve passes through the peaks on the object's boundary, transitioning above or below the image plane as it does so.

In conclusion, while there exist a number of 3D curve estimating approaches, there does not appear to exist a fully automatic solution for producing a 3D curve from a single 2D image. Other approaches require additional help in the form of 2D polylines, multiple image views from different angles, or user interaction. While better results are probably possible with more input data, our proposed method produces reliable results that satisfy a number of validation criteria from a single segmented image, using only standard image processing operations.

III. Method

A. Overview

The proposed method involves three main conceptual stages, discussed in more detail in subsections III-B, III-C, and III-D respectively.

1) Segment the object and extract its main structural curve.
2) Straighten the image by its structural curve.
3) Derive a 3D parametric curve from the straightened image and map it back to the original object.

The first stage extracts the structural 'spine' of the object (red line in Figure 1c) which describes the main object shape, but does not capture finer features of the helix. The second stage straightens the image by straightening the spine and warping the image such that every image point maintains the same position relative to the closest points on the spine. This produces an image of the straightened object that still has the helical structure of the target object (image in Figure 1d).

The third stage generates a helical curve by computing four 3D control points per helical revolution directly from the straightened image, and obtaining a parametric spline curve that interpolates cubically between them. The underlying assumption used to infer 3D geometry from a 2D image is that the cross-section of the helix is circular, therefore the maximum z-axis displacement from the image plane equals the radius of the helix. The control points we extract are the local extrema of the helix curve along its x and z axes (with the y-axis oriented along the object and the x-axis across it). Once the control points are calculated we perform the straightening transformation on them in reverse such that the 3D spline is mapped correctly back onto the original image (Figure 1e-f). The resulting curve, including its frequency, pitch, and tortuosity, are validated against synthetic and real-world data of both macro and microscopic scale.

B. Segmentation of the Main Structural Curve

Because helical objects often exhibit a macro level curvature in addition to their helical winding (see Table VII), we first straighten the image at the macro level (see Figure 2). This begins with extracting the object's macro-level spine, via a series of operations on a binary image of the object (Algorithm 1 and Figure 2a-d). In all our results, we obtain this segmentation using Otsu's intensity thresholding method [37]. We then select the largest connected component and fill any holes using a morphological algorithm [38] to ensure that we have a single binary representation of the object, separated from background noise, to process. The morphological skeleton is then calculated (Algorithm 1, line 4), which is well-known

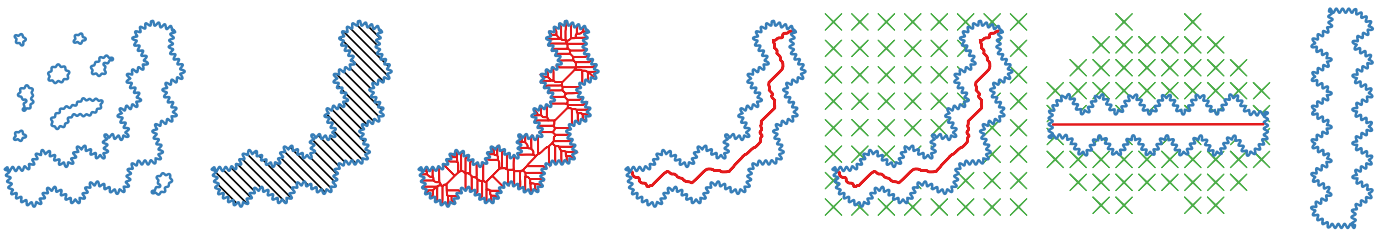


Figure 2: To straighten the image, we capture the main curve by finding the longest path between extrema in the binary skeleton (Algorithm 1), and calculate a local weighted mean transform from the original and deformed control points (Algorithm 2). Original image (a), largest thresholded object (b), skeleton (c), main curve (d), control points (e), and straightening (f-g).

to contain spurious branches [39] (Figure 2c), however we do not need a pruning heuristic as we only need to capture the main path through the skeleton. We take this main curve to be the diameter path through the skeleton, in other words the longest of shortest paths between two nodes. We find this path using the algorithm proved in [40] by first choosing an arbitrary seed endpoint, s_0, then finding s_1, the most distant endpoint to s_0 by quasi-Euclidean geodesic distance in the binary image [38] (Algorithm 1, lines 5 to 7). We then find s_2 as the most distant node to s_1, and use Dijkstra's algorithm with 8-connectivity to find the shortest path between s_1 and s_2. This results in a single path with no spurious branches that describes the spine of the complete object (Figure 2d).

Algorithm 1: Extract a curve $\mathbf{c}$ and binary image B from a greyscale input image of a helical object I.

Data: I
Result: $\mathbf{c}, B$
1 $B = I >$ threshold(I); ▷ Otsu's thresholding [37]
2 $B =$ fill(B); ▷ Fill holes [38]
3 $B =$ largest component(B);
4 $S =$ skel(B); ▷ Morphological skeleton [39]
5 $s_0 =$ index$_1$ of$(S = 1)$; ▷ Initial seed (any set pixel in S)
6 $D =$ dist(S, s_0) ; ▷ Geodesic distance from s_0 [38]
7 $s_1 =$ index of max(D); ▷ Seed point 1 (curve start)
8 $D =$ dist(S, s_1);
9 $s_2 =$ index of max(D); ▷ Seed point 2 (curve end)
10 $\mathbf{c} =$ dijkstra(S, s_1, s_2); ▷ Shortest path (8-connectivity)

C. Image Straightening

To straighten the image, we based our method on the unfolding approach presented in [41], originally developed to straighten 3D vertex meshes. The procedure begins by transforming the centerline curve (defined by a discrete chain of 2D points) into a horizontal line. Adapting the authors' work (Algorithm 2, line 6), the straightened curve is given by:

$$\mathbf{c}'_j.x = \mathbf{c}'_{j-1}.x + \|\mathbf{c}_j - \mathbf{c}_{j-1}\| \quad (1)$$

$$\mathbf{c}'.y = 0 \quad (2)$$

where $\mathbf{c}_j$ and $\mathbf{c}'_j$ are the j'th points of the original and straightened curves, and $\mathbf{c}'_j.x$ means "the x-component of the vector $\mathbf{c}'_j$". We now need to warp the image such that image points maintain their position relative to their nearest centerline points. To prevent the appearance of holes in the transformed image, we require an inverse geometric transform, which maps each pixel $\mathbf{p}'$ in the transformed image to a location $\mathbf{p}$ in the original image. In [41], a forward transformation is defined on mesh vertices, which are pulled along with the centerline to create a straightened object mesh. This transformation keeps the vertex in the same place relative to its nearest centerline point's frame of reference as the centerline is straightened. The transformed position of vertex $\mathbf{v}$ is given by:

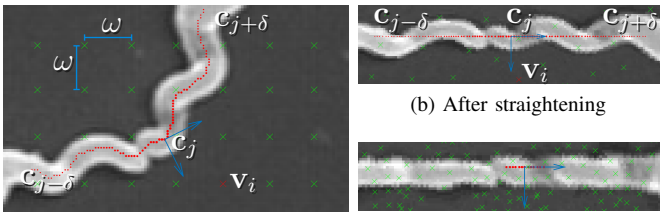
$$\mathbf{v}'_i = \mathbf{c}'_j + M_j^{-1}(\mathbf{v}_i - \mathbf{c}_j) \quad (3)$$

where $\mathbf{v}_i$ and $\mathbf{v}'_i$ are the original and transformed vertex points, $\mathbf{c}_j$ is the closest centerline point to $\mathbf{v}_i$ and M_j is a transformation matrix whose columns are the basis vectors of $\mathbf{c}_j$'s frame of reference. To form this basis, we simply take the centerline's tangent and normal vectors (by rotating the tangent 90°).

To apply this transformation in reverse to each pixel in the warped image, we first create a regular grid of control points in the original image (Figure 3a: green crosses), and transform them as described above (Figure 3b). We then use the original and transformed control point pairs to estimate a continuous inverse transformation function, using the local weighted mean (LWM) approximation scheme [42] (this is a technique from the image registration community implemented as a MATLAB library function). As in [41], the transformation of each control point is smoothed by considering a neighbourhood of δ centerline points either side of $\mathbf{c}_j$, whose influence is inversely proportional to their distance to $\mathbf{v}_i$ (see Algorithm 2, lines 8, 11, and 12).

The straightening process introduces three parameters: the transform smoothing neighbourhood size δ, the control point spacing ω and the LWM neighbourhood size o (where each control point uses the nearest o neighbours to estimate a local quadratic transformation function). In our experiments, discussed in the validation section, we find these parameters to be stable and can be set as constants $\delta = 50$, $\omega = 20$, and $o = 30$ in most cases. Setting ω to be too small results in over-straightening (Figure 3c) which removes the finer helical structure from the object.

Prior to helical extraction, we rotate the horizontally straightened binary image by 90-degrees such that we can work more



(a) Before straightening (b) After straightening (c) Over-straightening example

Figure 3: A pictoral representation of the control point transformation, as described in Algorithm 2, lines 1 - 12. A selected control point $\mathbf{v}_i$ is highlighted in red, and only its nearest centerline point $\mathbf{c}_j$ (along with its δ neighbours) are shown. The centerline points (red dots) are arranged into a straight line, 'pulling' the control points (green crosses) with them. The weights of the centerline points (Algorithm 2, lines 11-12) on $\mathbf{v}_i$ are shown by the size of the dots.

Algorithm 2: Straighten the $n \times m$ input image B by a curve $\mathbf{c}$, giving a straightened image B', subject to straightening parameters δ, ω and o. Here we denote by $\mathcal{V}$ the set of regularly spaced control points.

Data: B, $\mathbf{c}$, δ, ω, o
Result: B'
1 $\mathcal{V}$ = grid points over B with $\lfloor \frac{n}{\omega} \rfloor$ rows and $\lfloor \frac{m}{\omega} \rfloor$ cols;
2 $\mathbf{c}'_1.x = \mathbf{c}_1.x$;
3 $\mathbf{c}'_1.y = 0$;
4 q = number of centerline points;
5 **for** $j = 2..q$ **do**
6 $\quad \mathbf{c}'_j.x = \mathbf{c}'_{j-1}.x + \|\mathbf{c}_j - \mathbf{c}_{j-1}\|$; ▷ Straighten $\mathbf{c}$ to $\mathbf{c}'$
7 $\quad \mathbf{c}'_j.y = 0$;
8 $D_{i,k} = \|\mathbf{v}_i - \mathbf{c}_k\|$; ▷ Distance matrix between vertices $\mathbf{v}_i \in \mathcal{V}$ and centerline points
9 **foreach** $\mathbf{v}_i \in \mathcal{V}$ **do** ▷ Transform $\mathbf{v}_i$
10 $\quad$ let $\mathbf{c}_j$ be the nearest centerline point to $\mathbf{v}_i$;
11 $\quad a = \sum_{k=j-\delta}^{k=j+\delta} D_{i,k}^{-1}$; ▷ Weight normalization
12 $\quad \mathbf{v}'_i = \sum_{k=j-\delta}^{k=j+\delta} \frac{D_{i,k}^{-1}}{a}(\mathbf{c}'_k + M_k^{-1}(\mathbf{v}_i - \mathbf{c}_k))$;
13 B' =local weighted mean$(B, \mathcal{V}, \mathcal{V}', o)$; ▷ Transform [42]

intuitively vertically for the next section, and we then crop the result such that the object tightly fits the image borders.

D. Helical Extraction

From an $n \times m$ binary input image B of a straightened vertical helical object, which is cropped to fit the object's bounding box, we define a new centerline as a 1D array:

$$c[i] = \frac{\sum_{j=1}^{m} j \cdot B_{ij}}{\sum_{j=1}^{m} B_{ij}}, \quad i \in [1, n] \tag{4}$$

where $c[i]$ is the mean x-coordinate of the foreground pixels along the i'th row in the binary image. This array describes a centerline path which stays equidistant to the left and right boundaries of the straightened object (see Figure 4b), and is distinct from the macro-level centerline of the previous sub-section, which was used to straighten the object whilst preserving its helical structure. The object's helical structure will be derived primarily by detecting oscillation in this new centerline. This is preferable to measuring undulations in the object's boundary, since such boundary features will be present even in axially symmetric objects with no helical structure.

The extracted centerline is sensitive to the noise on the object boundary, therefore we smooth it using robust local regression [36] (Figure 4c) subject to a span parameter σ, yielding a smoothed 1D signal $r[i]$. We then find the positive and negative peaks (i.e. local maxima and minima in $r[i]$, see Figure 4d) subject to an optional minimum peak distance parameter d. These points will become the $z = 0$ control points, lying in the plane of the image. Setting $d > 0$ is useful to improve the peak detection in extremely noisy images, when prior knowledge about the helices' pitch can be exploited. For example, in microscale imaging techniques, we can often approximate how many peaks to expect within a given distance and set d to be a large enough minimum distance to prevent over-detection. In practice, we can typically rely solely on the smoothing parameter σ, and leave $d = 0$, where σ is set to be a value such as 0.1 to represent a span of 10% of the signal.

Depending on application, we sometimes wish the 3D helix to have the same width as the object, but the x-axis amplitude of the centerline is often much less than the width of the object. In these cases we can 'push' the $z = 0$ points along the x-axis until they reach the object boundary (Algorithm 3, lines 8 to 10 and Figure 4e).

Finally we calculate the other half of the control points; these are roughly midpoints between the $z = 0$ points and have alternating positive and negative z coordinates. Each midpoint's y coordinate $\mathbf{p}.y$ is halfway between those of its adjacent $z = 0$ peaks (line 14), however the x component is set to be the corresponding position along the robust centerline (i.e. $\mathbf{p}.x = r[\mathbf{p}.y]$), ensuring that it is centered in the object (line 15). In keeping with our assumption that the helix has a circular profile, we approximate the local z amplitude of the midpoint as the local width amplitude, approximated by half the x-distance of the locally adjacent points, with sign alternating from one midpoint to the next. In this way, our z amplitude adapts to the local width of the object, just as the x amplitude does. Bringing this all together, if $\mathbf{p}_i$ is a midpoint and its adjacent points are $z = 0$ points then:

$$\mathbf{p}_i.y = \frac{1}{2}(\mathbf{p}_{i-1}.y + \mathbf{p}_{i+1}.y) \tag{5}$$

$$\mathbf{p}_i.x = r[\mathbf{p}_i.y] \tag{6}$$

$$\mathbf{p}_i.z = \frac{1}{2}(\mathbf{p}_{i+1}.x - \mathbf{p}_{i-1}.x) \tag{7}$$

A diagrammatic explanation of the control points and how they relate to the centerline curve is shown in Figure 5.

There are some helical objects that are very thin, such as the Spring and Pseudomonas fluorescens in Table VIIIb and f. In such thin cases, pushing the peaks to the object boundary will not improve the curve and may even cause overfitting if the curve is pushed to noise adjoined to the boundary. Therefore we leave 'pushing' to be optional, but enable it by default given the rarity of such cases (Algorithm 3, line 6).

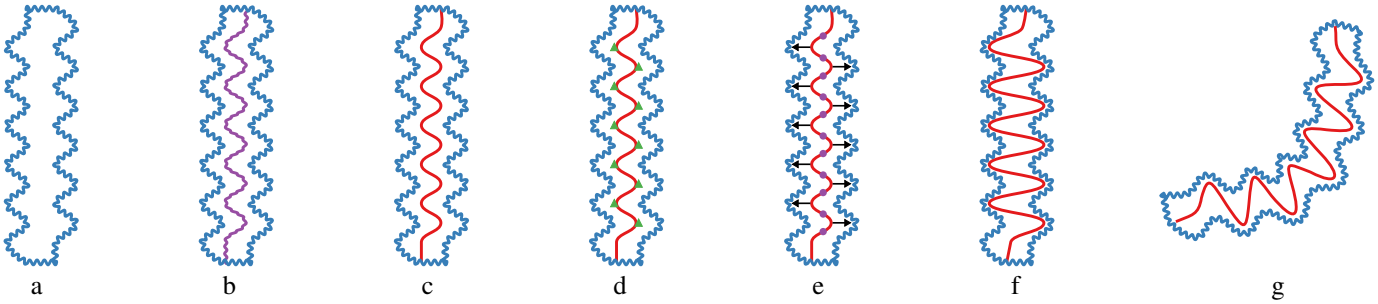


Figure 4: To extract the helical curve from a straightened image (a), we extract a 1D signal called the 'centerline' from the width profile (b), and fit a smooth curve using robust local regression (c). We then extend the peaks and find the midpoints (d-e) before fitting our final curve (f), which is projected back onto the original image (g). (Algorithm 3).

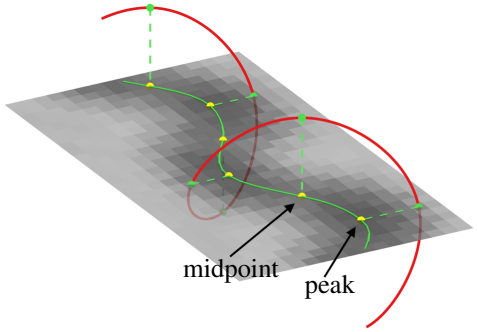


Figure 5: A small section of the helical curve for the Spirulina object (Table VIIb), showing how the control points (green dots) are related to the centerline peaks and midpoints (yellow dots). The helical curve itself (red) passes through all control points.

The control points $\mathbf{p}$ of the helical curve are the union of the midpoints with the positive and negative peaks, sorted along the y-axis (Algorithm 3, line 17). These control points are then transformed back onto the original (non-straightened) image by applying the relevant inverse transforms in reverse order; specifically we invert the cropping and 90-degree rotation before applying the inverse of the local weighted means transform (Figure 4g). With the extrema and midpoints correctly located on the original image, we can now compute a piecewise cubic spline that passes through these points by choosing interpolation nodes under a centripetal parameterization [43] (implemented in the MATLAB library function `cscvn`). This yields a smooth parametric curve which is locally adapted to fit the helical object's shape. A summary of our pipeline, techniques attributed to their original sources where appropriate, is provided in Table I.

Having obtained a parametric 3D helical curve, we can now estimate a number of useful high-level properties that may be required for application specific reasons. These include (but are not limited to) radius, pitch, length and tortuosity. Length can be estimated to arbitrary precision by summing the distance between densely sampled points on the curve; tortuosity is then defined as the curve's total length divided by the Euclidean distance between its endpoints. Radius and pitch can be calculated prior to un-straightening the curve, and are defined not just globally but per control point, since the object described by the image need not necessarily have a fixed radius or pitch. Pitch can be obtained as the y-axis

Table I: An ordered list of the steps in our pipeline, with reference to their original source where applicable.

Step	Source
Otsu thresholding	Standard technique [37]
Morphological thinning	Standard technique [39]
Longest path selection	Proposed in [40]
Centerline straightening and control point transformation	Proposed in [41]
Continuous transform estimation and image warping	Local weighted mean approximation scheme [42]
Helical centerline extraction	Novel
Centerline smoothing	Robust local regression [36]
Centerline peak detection	Detection of local maxima
Control point calculation	Novel
Spline fitting	Centripetal parameterization [43]

Algorithm 3: Extract the curve points $\mathbf{p}$ from an $n \times m$ straightened binary image B, with smoothing σ, minimum peak distance d, and an edge condition $push$ = true.

Data: $B, \sigma, d, push$
Result: $\mathbf{p}$

1 **for** $i = 1..n$ **do**
2 $\quad c[i] = \sum_{j=1}^{m} j \cdot B_{ij} / \sum_{j=1}^{m} B_{ij}$; ▷ Get centerline
3 r = robust local regression(c, σ) ; ▷ Smooth centerline
4 $\{\mathbf{t}_i^+\}$ = local maxima(r, d);
5 $\{\mathbf{t}_i^-\}$ = local minima(r, d) ; ▷ Ordered by y-coordinate
6 **if** *push* **then**
7 $\quad$ **foreach** $\mathbf{t}_i^+$ **do**
8 $\quad\quad \mathbf{t}_i^+.x$ = max $j : B_{ij} = 1$; ▷ Extend all peaks
9 $\quad$ **foreach** $\mathbf{t}_i^-$ **do**
10 $\quad\quad \mathbf{t}_i^-.x$ = min $j : B_{ij} = 1$;
11 $\{\mathbf{t}_i\}$ = sort $(\{\mathbf{t}_i^+\} \cup \{\mathbf{t}_i^-\})$; ▷ Maintain y-axis ordering
12 $\mathbf{t}_i.z = 0 \ \forall\, \mathbf{t}_i$;
13 **foreach** $i \in 1\,..\,(|\{\mathbf{t}_i\}| - 1)$ **do**
14 $\quad \mathbf{m}_i.y = \frac{1}{2}(\mathbf{t}_i.y + \mathbf{t}_{i+1}.y)$; ▷ Midpoint height
15 $\quad \mathbf{m}_i.x = r[\mathbf{m}_i.y]$; ▷ Midpoint width
16 $\quad \mathbf{m}_i.z = \frac{1}{2}(\mathbf{t}_{i+1}.x - \mathbf{t}_i.x)$; ▷ Midpoint depth
17 $\{\mathbf{p}_i\} = \text{sort}(\{\mathbf{t}_i\} \cup \{\mathbf{m}_i\})$; ▷ Merge and sort in y

displacement between two control points $\mathbf{p}_i$ and $\mathbf{p}_{i+4}$, since our algorithm extracts four control points per helix period. Radius, finally, is the distance of a control point to the line where $z = 0$ and $y = \text{mean}(r)$.

IV. Results

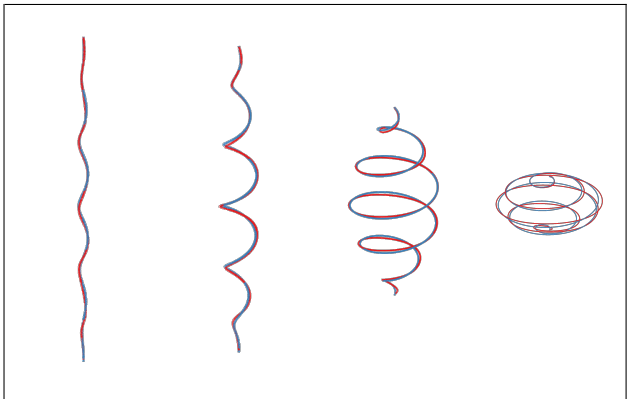
In this section, we provide quantitative and qualitative validation against both synthetic data and real-world data. In particular, we measure how well our generated curve fits to the helical object in extreme data scenarios, and how sensitive the algorithm is to its input parameters.

A. Synthetic Validation: Comparison to Analytical Helix Curve

Beginning with an analytical 3D helix equation specifying a helix as a path through 3D space, we generate our synthetic data. The 3D helix path is projected orthographically onto a plane to capture a single 2D image which forms our input test data. Using synthetic data allows us to compare our extracted tortuosity τ with the ground truth τ^*, and allows us to compute the Hausdorff distance h between the analytical 3D curve and the extracted 3D curve. In our initial experiment, we use the equation of a spherical helix using a fixed number of twists, but vary the bulge amount b (Table II). This is a worse-case for inferring the depth within a circular helical profile, as it yields a large change of both increasing and decreasing amplitudes within a fixed distance between the ends of the curve and a small number of twists.

In Table II, we can see that the extracted 3D curve in red gives a good approximation of the analytical 3D curve shown in blue, however there is a slight noticeable deviation from the curve in the extreme bulge shown on the right. To examine this relation, we run 250 experiments of increasing bulge amount and plot the results in Figure 6. Despite some initial noise from the discrete 3D rasterization, the results show that we can obtain an excellent approximation of the ground truth tortuosity τ^*, and we obtain a good fitting as measured by the Hausdorff distance h between a dense sampling of points on both curves. It is worth noting that this experiment for capturing the Hausdorff distance is extreme; in the real-world data we have collected we rarely find such local variation in amplitude above 30 pixels. The largest local change in amplitude we encountered is shown in the central section of the Leptospira image in Figure 1, whereby the local helical amplitude significantly deviates from its mean value, however our method captures this variation well and therefore we can expect a good fitting in most scenarios given the correctly chosen input parameters.

Table II: The analytical helix curve (blue) is compared to the generated curve (red) from a 2D rasterized image view of the analytical curve. We show four different bulge amounts b, where τ is the resulting tortuosity compared to the ground truth tortuosity τ^* of the analytical curve, and h is the Hausdorff distance between the two curves. More results in Figure 6.



b	2	8	32	128
τ	1.040	1.470	4.275	16.406
τ^*	1.039	1.482	4.304	16.603
h	1.052	1.629	2.076	7.7876

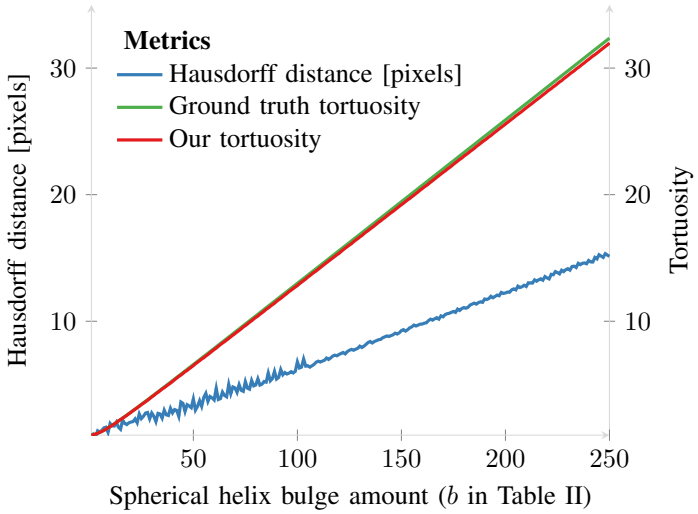


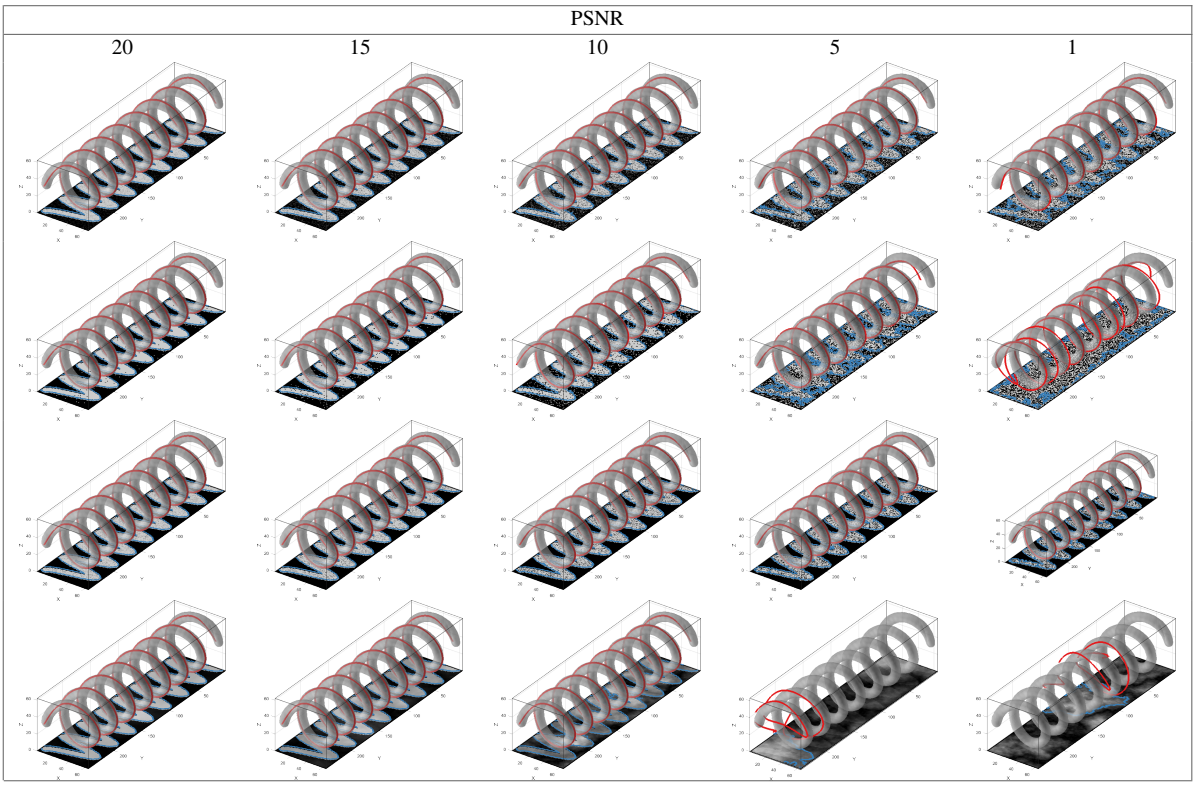
Figure 6: Comparing the Hausdorff distance between an analytical helix curve of increasing bulge amount b and our 3D construction from a single 2D view of the analytical curve.

B. Noise Experiment

As before, we create a synthetic 3D spring-like object that is projected onto a 2D plane. This time we create a thicker line, so we can apply more noise while still presenting the algorithm a fair chance to succeed. This 2D image is corrupted with multiple different types of noise with a decreasing peak signal-to-noise ratio (PSNR). We measure the algorithm's robustness to Gaussian, salt & pepper and speckle noise types, as well as its robustness to structured multi-frequency noise. The 3D curve is extracted from the corrupted 2D image, and it is compared with the analytical 3D curve from the helix equation. We measure the Hausdorff distance (Figure 7) as well as the difference in tortuosity (Figure 8), where a selection of the results are shown in Table III.

The results in Figures 7 and 8 show that our method is accurate for multiple types of noise, heavily corrupting the object to a PSNR of about 11. The method is also shown to retain precision across the noise types, however there is a small amount of constant error where the peaks are pushed to the object boundary instead of following the center of the spring cross-section. These results can be improved in specific

Table III: A section of results from our noise experiments in Figure 7 and 8: a synthetic 3D helix is generated with a thickness radius of 5 and projected onto a 2D plane. The resulting 2D image is corrupted with different levels and types of noise. We then apply our segmentation algorithm (Algorithm 1) and show the segmentation boundary in blue. The extracted curve is shown in red (Algorithm 3).



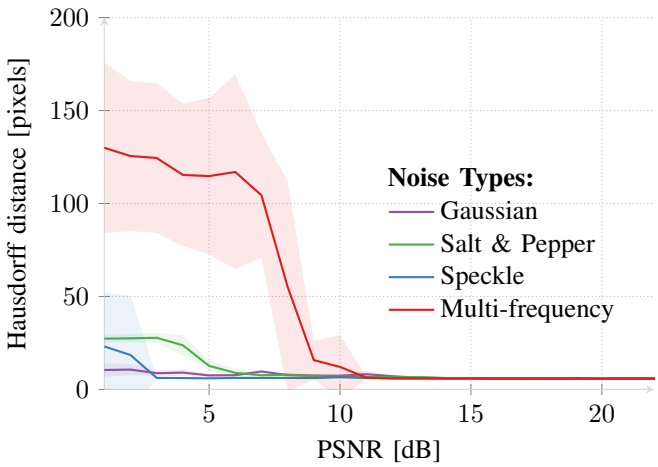


Figure 7: The Hausdorff distance from the ground truth synthetic helix with a thickness of 5 corrupted with different types and levels of noise (shown in Table III) using the same parameters $\sigma = 0.15$ and $d = 0$. The standard deviation is shown in the error bars (transparent shaded regions).

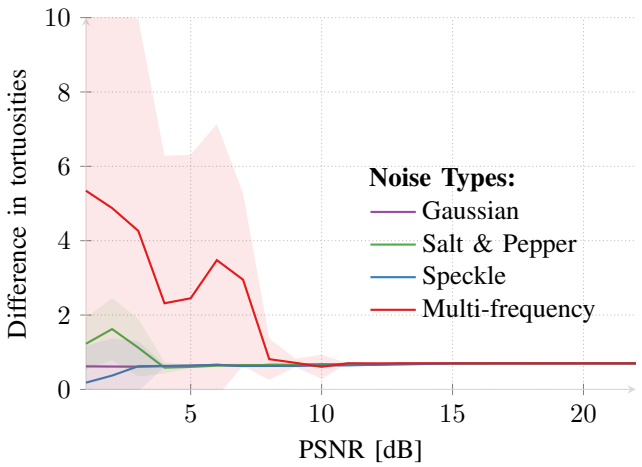


Figure 8: The difference between the analytical helix tortuosity and the tortuosity of the helix extracted from a 2D image corrupted with different types and levels of noise (shown in Table III) using the same parameters $\sigma = 0.15$ and $d = 0$.

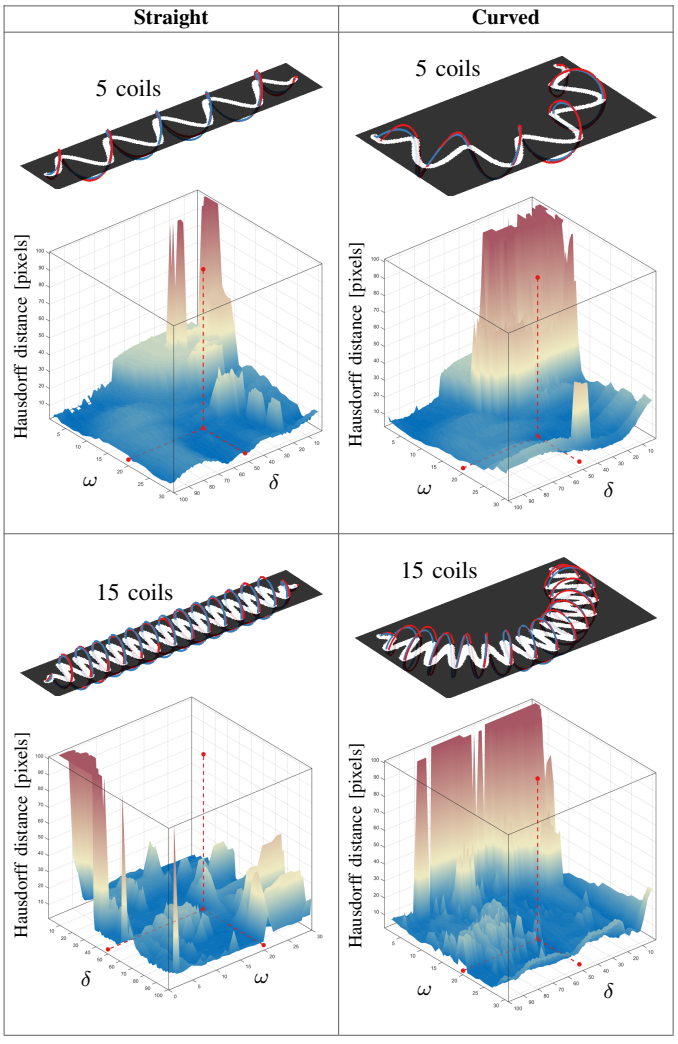
cases by disabling or even interpolating the amount of 'peak-pushing', however such behaviour is not always desirable in cases such as the hair in Table VIII, and we leave this optional (Algorithm 3 line 6).

C. *Straightening experiment*

The straightening process (Algorithm 2) introduces three parameters: ω, δ, and o. We measure the algorithm's stability to these parameters in a large synthetic experiment; we generate four synthetic helices which are (1) straight or curved, and (2) dense or sparse in terms of their number of coils. We then vary

$\omega = [1, 256]$, $\delta = [1, 256]$, $o = [6, 85]$ and plot the Hausdorff distance for each of the four synthetic objects, which results in four 3D parameter spaces accordingly. We found that there is a large, low, flat region in the results indicating that our method is stable to these parameters. We choose $\omega = 20$, $\delta = 50$, and $o = 30$ as default parameters at the center of this region, which typically require little-to-no tuning. We plot cross-sections of this parameter space at fixed $o = 30$ for the four test cases in Table IV, alongside images of the synthetic helix, analytical curve (blue), and extracted curve (red) for the proposed default parameters. To see how o affects the results, we also show a plot of $o = [6, 85]$ with $\omega = 20$ and $\delta = 50$, showing a local minimum in the range $o = [20, 40]$ (Figure 9).

Table IV: We generate synthetic straight and croissant-shaped helices with a varying number of coils. The Hausdorff distance between the analytical ground truth (blue curve) and the extracted helix (red curve) are plotted as a surface; varying the straightening parameters ω and δ from 1 .. 256 in 262,144 experiments for incremental values $o = 6, 7, \ldots 85$. Choosing $o = 30$ gives low and smooth Hausdorff distances (Figure 9). We show a cropped relevant region of the results with large, low, flat regions which suggest $\omega = 20$, $\delta = 50$ (dashed red lines) as default parameter values that work in most cases; results using these parameters are shown alongside the Hausdorff plots.



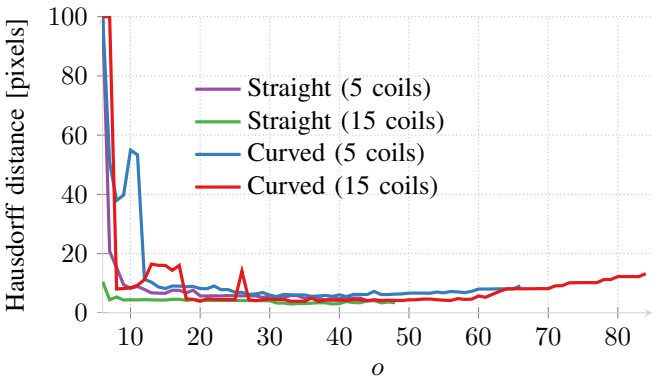


Figure 9: We plot the Hausdorff distance for $o = [6, 85]$ with ω and δ held constant at the suggested values $\omega = 20$ and $\delta = 50$ for the synthetic helices shown in Table IV. Generally for $o = [12..40]$ the Hausdorff distance is flat, low, and stable; however choosing too many or too few control points results in poor locality of the weighted geometric transform. We suggest $o = 30$ based on the flat region in this experiment.

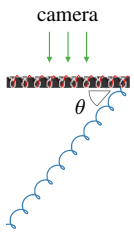
D. *Projection experiment*

We assess our algorithm in cases where the helix is viewed from increasingly oblique angles until its 2D projection is a complete circle (Table V). We show the extracted 3D curve (red) compared to the analytical curve (blue) and plot the torutosity compared to the ground truth in Figure 10. The results show that our method can correctly capture the number of coils at highly oblique angles, but the length decreases according to the viewing angle. The straightened curve can be stretched by diving its length by $\cos(\theta)$ if the viewing angle θ is known beforehand, e.g. from the imaging setup. This produces far more accurate tortuosity measurements, plotted in blue.

E. *Real-World Validation: Parameter Sensitivity*

The smoothing parameter σ is used to clean the noisy boundary of real-world images of helical objects. To measure its sensitivity, we run a hundred experiments on three different

Table V: Synthetic experiment to evaluate the extracted curve from different viewing angles. An analytical helix is rotated away from the viewing plane and projected onto a 2D image, of which we extract the 3D curve shown in red. The tortuosities are compared to the analytical helix and plotted in Figure 10.

70° Diagram	θ	Top-down orthographic projection
	0°	
	10°	
	20°	
	30°	
	40°	
	50°	
	60°	
	70°	
	80°	
	90°	

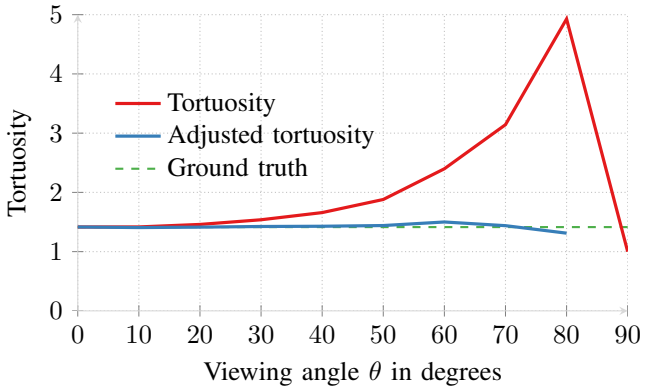


Figure 10: Extracted tortuosities from different viewing angles (shown in Table V). If θ is known beforehand, e.g. from the imaging setup, we adjust the straightened curve by dividing its length by $\cos(\theta)$. This stretches the extracted curve to approximately the true 3D length of the target object, correcting much of the distortion caused by non-perpendicular viewing angles and yielding a far more accurate tortuosity.

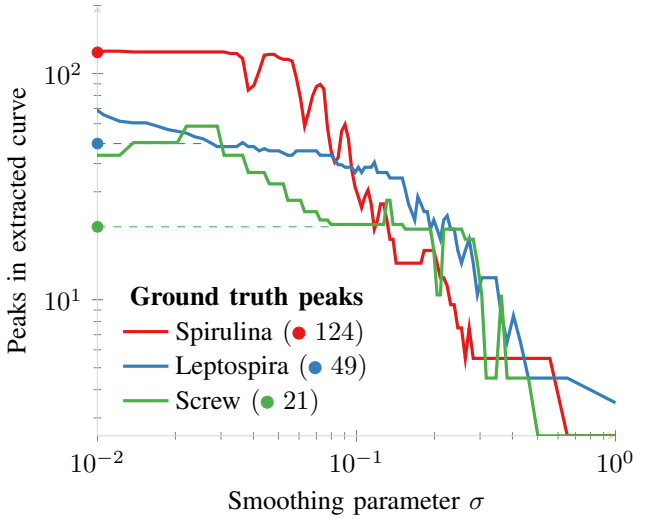


Figure 11: Parameter sensitivity for our main smoothing parameter $\sigma = 0.01$ to 1 in three different noisy helical objects, showing the number of peaks detected as σ varies accordingly.

macro and microscopic real-world images (Table VIIa & b, and Table VIIIa) plotting $\sigma = 0.01$ through to 1 and count the number of peaks in the generated 3D curve. We then compare these peaks to a ground truth, which we have manually counted for each image, and plot the results in Figure 11. In this experiment, all of the other parameters were set to their defaults as discussed in the previous sections.

In Figure 11 we see substantial noise in the results; in particular before or after the ground truth there is often a noticeable dip or peak in the results. The general decrease in the number of detected peaks is caused by peaks becoming indiscernible due to smoothing. The occasional increase meanwhile is caused by more subtle effects, where excessive robust local regression smoothing changes the number of first order derivative zero crossings in regions where the signal becomes almost constant. However we can see a clear phase of delay for each of the three sample images at the ground truth; this plateau indicates the stability of our algorithm over the range of σ.

F. Real-World Validation: Discussion

We ran our algorithm on a wide range of image types and show the results in Table VII. Additionally, we ran the algorithm without the unwrapping process on naturally straight images in Table VIII. All of the results were gathered with little tuning of σ and δ and otherwise default parameters were set as discussed in the previous sections; we show the values chosen for σ and d for each of the images in Table VI. Our algorithm achieves a good visual fitting of the helical curve for the image types; in particular it can robustly handle noisy images, as shown in Table VIId and Table VIIIf. Table VIIIe was the most noisy scenario, imaging a DNA fibre, where we required finer tuning of d compared to the other scenarios to correctly identify the peaks. With the default straightening parameters some minor underfitting occurs in cases such as the top end of Table VIIc; this can be improved by increasing the density of control points ω in the straightening phase. Our algorithm extracts 3D curves with the correct number of coils from images taken at oblique angles, however the central axis of the resulting 3D curve will always be parallel to the image plane whether or not the object itself is, as shown in Table VIIc, Table VIIIa & d.

Table VI: Parameters used for the extracted curve in the images in Table VII and Table VIII, and their resulting tortuosities τ. We also show the number of extracted peaks which in all cases are the same as the ground truth. We improve the fitting by setting d according to the number of rows n in the straightened image B, and use the default values $\delta = 50$ and $\omega = 20$.

	Table VII				Table VIII					
Image	a	b	c	d	a	b	c	d	e	f
σ	0.01	0.01	0.07	0.05	0.1	0.05	0.1	0.09	0.01	0.1
d	$n/20$	0	0	0	0	$n/5$	0	0	$n/12$	0
τ	3.19	2.87	3.49	2.74	5.86	5.55	1.27	19.23	15.28	1.36
peaks	49	124	19	13	21	8	7	29	19	7

G. Performance

The algorithm performance is shown in Table IX for each of the images in Tables VII and VIII. Our unoptimized MATLAB implementation performs reasonably quickly; the slowest part is the straightening process, in particular evaluating the local weighted mean transform for the nearest $o = 30$ control points.

V. Limitations

The limitations of our approach are intuitively simple and are mostly a product of the method's underlying assumptions:

1) The object is assumed to have a circular cross-section; the method will never produce an elliptical helix.
2) The helical object cannot wrap over itself or intersect with other helical objects at a macro-level; the skeleton of the segmented helical object must have a tree structure otherwise the straightening results are undefined.

Table VII: Different views of our result (middle columns) and the intermediate straightened image (right column) for Leptospira: (a) [10] and a busy scenario (d) [44]. Spirulina image types: (b) with a large number of curls [45] and (c) a skewed view [46].

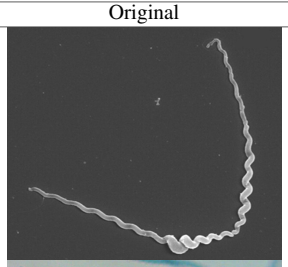
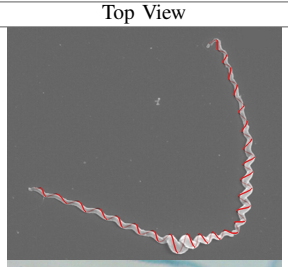
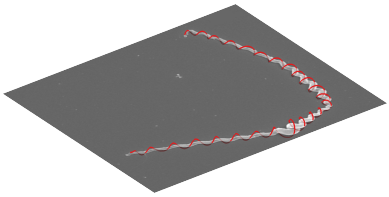
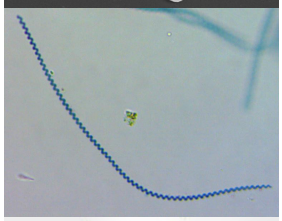
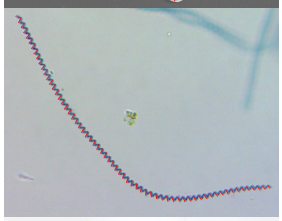
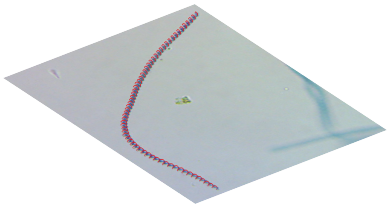
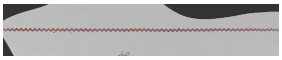
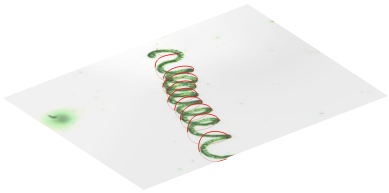
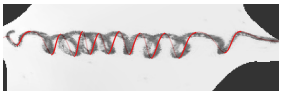
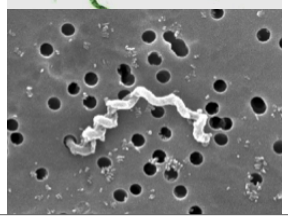
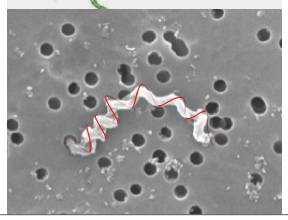
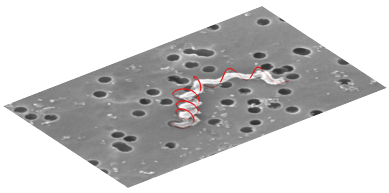
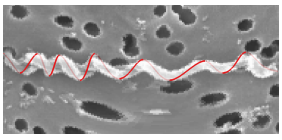
	Original	Top View	Side View	Straightened Image
a				
b				
c				
d				

Table VIII: Different output views of macro- and microscale straight image types, without the straightening process (Algorithm 2). Images of screws (a,d) [47] [48], springs (b), hair (c), and noisy images of: DNA (e) [49] and Pseudomonas fluorescens (f) [50].

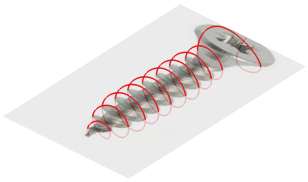
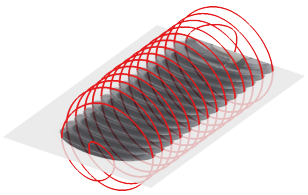
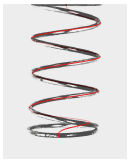
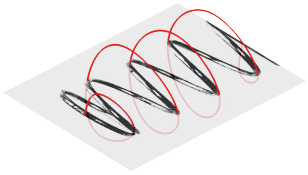
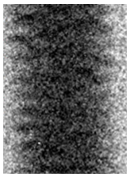
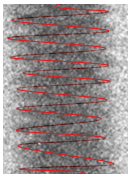
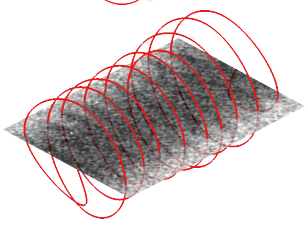
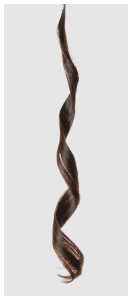
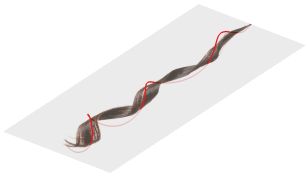
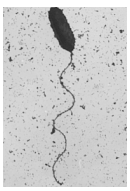
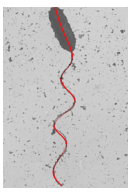
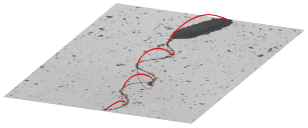
	Original	Top View	Side View		Original	Top View	Side View
a				d			
b				e			
c				f			

Table IX: Performance timings (seconds) for the segmentation of the structural curve (Algorithm 1), the straightening process (Algorithm 2), and the fitting of the final 3D curve (Algorithm 3) for each of the images in Table VII and Table VIII.

	Table VII				Table VIII					
Image	a	b	c	d	a	b	c	d	e	f
X Size	653	771	352	304	169	971	1,702	212	280	220
Y Size	539	590	264	191	269	999	4,039	380	380	280
Alg. 1	0.16	0.14	0.12	0.11	0.13	0.22	0.38	0.16	0.26	0.13
Alg. 2	15.4	24.7	1.69	0.85	-	-	-	-	-	-
Alg. 3	0.34	0.35	0.24	0.19	0.21	0.32	0.83	0.21	0.21	0.17
Total (s)	15.9	25.2	2.05	1.15	0.34	0.54	1.21	0.37	0.47	0.3

3) The object must first be segmented, and this segmentation must be unbroken and must accurately capture the undulations in the object's boundary. In all of our experiments we have found intensity thresholding by Otsu's method, followed by largest connected component selection, to be sufficient.
4) The object is assumed to lie parallel with the image plane, with its axis orthogonal to the viewing direction. If the object is inclined too steeply when imaged then its coils will appear to self-intersect; the resulting helix curve will still have the correct radius and number of coils, but the length and pitch will be underestimated since the boundary peaks will appear closer together now. If the viewing angle θ is known (see Table V), the straightened helix curve can be corrected by dividing its length by $\cos(\theta)$.

Limitation 1 occurs because it is extremely difficult to infer depth information from a single 2D image without prior knowledge of the imaging modality and lighting conditions.

Limitation 2 occurs because our macro-level spine extraction (Algorithm 1) works by finding the diameter path (i.e. longest shortest path) through the object's morphological skeleton; if such a path self-intersected then it would contain a loop, and so could not be a diameter path, as it could be made shorter by bypassing the loop.

To keep our approach generic across all 2D imaging modalities, we ignore the object's gray level information and work purely with its binary image, hence limitation 3. We consider it reasonable that some such pre-processing step be required in order to extract the object's geometry from the image. Segmentation is a very large field which has been intensely researched, therefore if segmentation is an issue then the less researched problem of helix extraction is likely to be harder still.

Finally, limitation 4 is a direct consequence of the post-straightening centerline detection. We estimate the centerline as staying equidistant to the object's left and right extrema, and express it as a 1D function of position along the object's long axis. Such a function cannot accurately model the centerline of an object such as Table VIIc, because it would need to be many-valued in some places. Replacing this function with a more flexible parametric curve of the form $(x(t), y(t))$ is a possible extension to our method, which may improve results on objects whose images self-intersect.

VI. Future Work

As a possible extension, it would be worth seeing if this work could be extended to handle images of multiple helical objects. This would be a matter of replacing the Otsu threshold step with a more powerful segmentation algorithm able to reliably identify multiple objects, and iterating our method over those objects.

For each sample image in Figure 11 we observed an interval of values for the smoothing parameter σ in which the correct number of peaks is detected. It may be possible to find this interval automatically by testing a range of values for σ and looking for the flat regions in Figure 11, thus removing the need for the user to tune σ. Furthermore, it may be possible to simplify and optimize the straightening process, thus eliminating some of the straightening parameters, by adapting the mean width profile (Equation 4) to sample the image along the profile which is normal to the macro curve.

The algorithm approach may fail to identify the parity of the depth of the extracted curve; for example in Table VIId, the top view of the 3D curve would look like a better fit if the sign of all z values were flipped. In our experiments, we attempted to automatically set the sign of the z value based on the mean of the intensity profile across alternating segments of the curve. We expected regions of the curve further away from the camera to be darker due to shadowing from foreground parts of the object, however due to shine and other image artifacts this is often not the case. For future work, it would be worth investigating how to improve the curve fitting in such cases using image intensity and gradient information, and for offsetting the skew in oblique angles.

VII. Conclusion

We have proposed a robust method to automatically extract a 3D helix curve from a single 2D image of a 3D helical object. We straighten the input image object at a large scale, before fitting a finer helical curve to peaks in the object boundary. Our approach has been shown to produce stable results in noisy real-world data without requiring extensive parameter tuning. The experimental results show that depth information can be inferred from the object's 2D projection, assuming that the helical object has a circular profile. We find that, for synthetic data, the extracted 3D curve is close to the ground truth in terms of Hausdorff distance and tortuosity. In our results, we show that helical objects in real-world 2D image data often exhibit curve structure at a macro-level in addition to the helical coils themselves. In the future, we would like to see a simpler algorithm for fitting 3D parametric curves to noisy 2D data without the straightening process, which would speed the algorithm up and remove the need for several parameters.

Availability and Implementation

The software has been implemented in MATLAB and is made available at: https://github.com/cwkx/extract-3d-curve

Acknowledgment

This work was supported by research grants from Dyson Ltd, UK (RF080296) and The Royal Society, UK (RF080232).

REFERENCES

[1] D. Wu, S. Wang, K. Liu, X. Yu, Y. He, and Z. Wang, "Rapid measurement of morphological features of spirulina microalgae filaments using microscopy and image processing algorithms," *Biosystems Engineering*, vol. 112, no. 1, pp. 35–41, 2012.

[2] C. Li, C. Wolgemuth, M. Marko, D. Morgan, and N. Charon, "Genetic analysis of spirochete flagellin proteins and their involvement in motility, filament assembly, and flagellar morphology," *Journal of Bacteriology*, vol. 190, no. 16, pp. 5607–5615, 2008.

[3] S. Vazquez and A. M. J. Flores-Alonso, "Confocal microscopy and image analysis indicates a region-specific relation between active caspases and cytoplasm in ejaculated and epididymal sperm," *PLoS ONE*, vol. 7, no. 4, p. e35477, 2012.

[4] A. Kruchten and M. McNiven, "Dynamin as a mover and pincher during cell migration and invasion," *Journal of Cell Science*, vol. 119, no. 9, pp. 1683–1690, 2006.

[5] P. Bandaru, C. Daraio, K. Yang, and A. Rao, "A plausible mechanism for the evolution of helical forms in nanostructure growth," *Journal of Applied Physics*, vol. 101, no. 9, p. 094307, 2007.

[6] A. Goriely and M. Tabor, "The mechanics and dynamics of tendril perversion in climbing plants," *Physical Review A*, vol. 250, pp. 311–318, 1998.

[7] F. Migliaccio, A. Fortunati, and P. Tassone, "Arabidopsis root growth movements and their symmetry: Progress and problems arising from recent work," *Plant Signaling & Behavior*, vol. 4, no. 3, pp. 183–190, 2007.

[8] E. Piuze, P. Kry, and K. Siddiqi, "Generalized helicoids for modeling hair geometry," *Comput Graph Forum*, vol. 30, no. 2, pp. 247–256, 2011.

[9] S. R. Ghoreishi, P. Cartraud, P. Davies, and T. Messager, "Analytical modeling of synthetic fiber ropes subjected to axial loads. part i: A new continuum model for multilayered fibrous structures," *International Journal of Solids and Structures*, vol. 44, no. 9, pp. 2924–2942, 2007.

[10] L. Slamti, M. A. de Pedro, E. Guichet, and M. Picardeau, "Deciphering morphological determinants of the helix-shaped leptospira," *Journal of Bacteriology*, vol. 193, no. 22, pp. 6266–6275, November 2011.

[11] N. Cherin, F. Cordier, and M. Melkemi, "Modeling piecewise helix curves from 2d sketches," *Computer-Aided Design*, vol. 46, pp. 258–262, 2014.

[12] F. Cordier, M. Melkemi, and H. Seo, "Reconstruction of helices from their orthogonal projection," *Computer Aided Geometric Design*, vol. 46, pp. 1–15, 2016.

[13] F. Cordier and S. Hyewon, "Free-form sketching of self-occluding objects," *IEEE Comput Graph & Appl*, vol. 27, no. 1, pp. 50–59, 2007.

[14] P. Maragos and R. Schafer, "Morphological skeleton representation and coding of binary images," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 34, no. 5, pp. 1228–1244, 1986.

[15] X. Bai, L. Latecki, and W. Liu, "Skeleton pruning by contour partitioning with discrete curve evolution," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 449–462, 2007.

[16] R. Ogniewicz and O. Kubler, "Hierarchic voronoi skeletons," *Pattern Recognition*, vol. 28, no. 3, pp. 343–359, 1995.

[17] N. Amenta, S. Choi, and R. K. Kolluri, "The power crust, unions of balls, and the medial axis transform," *Computational Geometry*, vol. 19, no. 23, pp. 127–153, 2001.

[18] M. S. Hassouna and A. Farag, "Robust centerline extraction framework using level sets," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 20-25 June 2005, pp. 458–465.

[19] M. S. Hassouna, A. Farag *et al.*, "Variational curve skeletons using gradient vector flow," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 12, pp. 2257–2274, 2009.

[20] L. Lam, S.-W. Lee, and C. Suen, "Thinning methodologies-a comprehensive survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14, no. 9, pp. 869–885, Sep 1992.

[21] A. Goriely, S. Neukirch, and A. Hausrath, "Polyhelices through n points," *International Journal of Bioinformatics Research and Applications*, vol. 5, no. 2, pp. 118–132, 2009.

[22] A. Derouet-Jourdan, F. Bertails-Descoubes, and J. Thollot, "Floating tangents for approximating spatial curves with piecewise helices," *Computer Aided Geometric Design*, vol. 30, no. 5, pp. 490–520, 2013.

[23] S. Ghosh, "Geometric approximation of curves and singularities of secant maps: a differential geometric approach," Ph.D. dissertation, University of Groningen, 2010.

[24] W. Wang, H. Pottmann, and Y. Liu, "Fitting b-spline curves to point clouds by curvature-based squared distance minimization," *ACM Transactions on Graphics*, vol. 25, no. 2, pp. 214–238, 2006.

[25] T. Lewiner, J. D. G. Jr., H. Lopes, and M. Craizer, "Curvature and torsion estimators based on parametric curve fitting," *Computers and Graphics*, vol. 29, no. 5, pp. 641–655, 2005.

[26] J. Cohen, L. Markosian, R. Zeleznik, J. Hughes, and R. Barzel, "An interface for sketching 3d curves," in *Proceedings of the Symposium on Interactive 3D Graphics*, 26-29 April 1999, pp. 17–21.

[27] J. Wither, F. Bertails, and M. Cani, "Realistic hair from a sketch," in *IEEE Conference on Shape Modeling International*, 13-15 June 2007, pp. 33–42.

[28] K. Shoji, K. Kato, and F. Toyama, "3-d interpretation of single line drawings based on entropy minimization principle," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 8-14 December 2001, pp. 90–95.

[29] C. Steger, "An unbiased detector of curvilinear structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 2, pp. 113–125, 1998.

[30] A. Frangi, W. Niessen, K. Vincken, and M. Viergever, "Multiscale vessel enhancement filtering," in *Medical Image Computing and Computer-Assisted Interventation*, 1998, vol. 1496, pp. 130–137.

[31] R. Y. Li, C. Y. Hsieh, and Y. H. Tseng, "3d b-spline curve fitting with mms image features of road lines," 18-23 October 2009.

[32] C. Canero, P. Radeva, R. Toledo, J. Villanueva, and J. Mauri, "3d curve reconstruction by biplane snakes," in *International Conference on Pattern Recognition*, vol. 4, 3-7 September 2000, pp. 563–566.

[33] P. Savadjiev, J. S. Campbell, G. B. Pike, and K. Siddiqi, "3D curve inference for diffusion MRI regularization and fibre tractography," *Medical Image Analysis*, vol. 10, no. 5, pp. 799–813, October 2006.

[34] P. Parent and S. W. Zucker, "Trace inference, curvature consistency, and curve detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 8, pp. 823–839, August 1989.

[35] C. Loader, *Local Regression and Likelihood*, ser. Statistics and Computing. Springer New York, 2006.

[36] W. Cleveland, "Robust locally weighted regression and smoothing scatterplots," *Journal of the American Statistical Association*, vol. 74, no. 368, pp. 829–836, 1979.

[37] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans Syst, Man, Cybern, Syst*, vol. 9, no. 1, pp. 62–66, 1979.

[38] P. Soille, *Morphological Image Analysis: Principles and Applications*, ser. Engineering online library. Springer, 2004.

[39] L. Lam, S.-W. Lee, and C. Suen, "Thinning methodologies-a comprehensive survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14, no. 9, pp. 869–885, 1992.

[40] R. W. Bulterman, F. W. van der Sommen, G. Zwaan, T. Verhoeff, A. J. M. van Gasteren, and W. H. J. Feijen, "On computing a longest path in a tree," *Inf. Process. Lett.*, vol. 81, no. 2, pp. 93–96, Jan. 2002.

[41] J. Yao, A. Chowdhury, J. Aman, and R. Summers, "Reversible projection technique for colon unfolding," vol. 57, no. 12, 2010, pp. 2861–2869.

[42] A. Goshtasby, "Image registration by local approximation methods," *Image Vision Computing*, vol. 6, no. 4, pp. 255–261, 1988.

[43] E. T. Lee, "Choosing nodes in parametric curve interpolation," *Computer-Aided Design*, vol. 21, no. 6, pp. 363–370, 1989.

[44] J. Chen, J. Bergevin, R. Kiss, G. Walker, T. Battistoni, P. Lufburrow, H. Lam, and A. Vinther, "Case study: A novel bacterial contamination in cell culture production-leptospira licerasiae," *PDA J Pharm Sci Tech*, vol. 66, no. 6, pp. 580–591, 2012.

[45] (2015) Forensic lab supply, cyanophycota: Spirulina. Accessed: 2015-09-07. [Online]. Available: http://grauhall.com/catalog/product_info.php?products_id=1672

[46] (2015) Algatarifa spirulina. Accessed: 2015-09-07. [Online]. Available: http://www.algatarifa.com/Benvenuto.html

[47] (2015) Zinc plated countersink pozi screws. Accessed: 2015-09-07. [Online]. Available: http://www.ukpictureframingsupplies.co.uk/zinc-plated-countersink-pozi-screws-4-x-38-10mm-320-p.asp

[48] (2015) Spring plungers with grub screw. Accessed: 2015-09-07. [Online]. Available: http://www.cotel.co.uk/p/493/spring-plunger-with-grub-screw-and-plastic-nose-pin

[49] F. Gentile, M. Moretti, T. Limongi, A. Falqui, G. Bertoni, A. Scarpellini, S. Santoriello, L. Maragliano, R. P. Zaccaria, and E. di Fabrizio, "Direct imaging of dna fibers: The visage of double helix," *Nano Letters*, vol. 12, no. 12, pp. 6453–6458, 2012.

[50] (2007) Imaging bacteria, biopolymers, and colloidal particles. Accessed: 2015-09-07. [Online]. Available: https://www.wpi.edu/academics/che/BA/imagingbacteria.html

Chris G. Willcocks received a PhD in Computer Science at Durham University in 2013 where he specialized in real-time GPGPU rendering and deformation of large volumetric datasets. He researched object deformation at Newcastle University Game Lab before accepting a post-doctoral research position at Durham University in 2015.

His interdisciplinary research aims to enable elegant solutions to otherwise challenging or computationally expensive problems in the fields of computer graphics and bioimage informatics.

Philip T. G. Jackson received a BSc degree in Computer Science & Physics within the Natural Sciences Programme from Durham University followed by an MSc in Computer Science, and is currently reading for a PhD degree in Computer Science, also from Durham University, UK.

His research investigates the use of recurrent neural networks for multi-object localisation.

Carl J. Nelson received an MSci degree in Biology & Physics within the Natural Sciences Programme from Durham University and is currently reading for a PhD degree in Computing Science, also from Durham University, UK.

His interdisciplinary research focuses on developing advanced and novel quantification techniques for bioimaging. The tools are used to further the understanding of biological processes that can be gleaned through bioimaging techniques.

Boguslaw Obara received an MSc in Physics from the Jagiellonian University and PhD in Computer Science from the AGH University of Science and Technology, Krakow, Poland. He has been a researcher at the Polish Academy of Sciences (2001-2007), a Fulbright fellow (2006-2007) and a postdoctoral researcher at the University California, USA (2007-2009) and the University of Oxford, UK (2007-2009). He is currently a senior lecturer in Computer Science at the University of Durham, UK.

His interdisciplinary research focuses on advancing the state of the art in bioimage informatics technologies aimed at a better understanding of the complex biological processes, from nano to macro.