Title:

Curves in the wild

Authors:
John K. Johnstone, jkj@uab.edu, UAB

Keywords:
Point cloud dataset, Curve dataset, Segmentation, Thresholding, Benchmarking dataset.

DOI: 10.14733/cadconfP.2023.260-264

Introduction:

Although there is a rich assortment of public 3D datasets of polygon meshes and surface point clouds (e.g., AIM@Shape [1], ModelNet40 [11], Princeton Shape Benchmark [12], Thingi10k [10], and so on), there are few public 2D datasets of smooth curves or their point clouds. As a modest contribution to curve datasets, this paper considers the construction of a dataset of 2D point clouds of curves derived from leaves. We are interested in datasets derived from real organic shapes, and we are particularly interested in developing a curve dataset with interesting tangent space structure (e.g., bitangents), to test shape modeling algorithms (e.g., we were originally motivated to study this problem through our study of computing bitangents [5]), and leaves offer this rich structure.

The curve dataset will be segmented from leaf images. A second goal of this study is the development of the simplest possible robust segmentation algorithm for extracting point clouds from leaf images, and other similar bimodal images. Standard approaches can fail, despite the simplicity of these images.

Leaf images and leaf clouds:

Consider a leaf image of a post oak, from the Cleared Leaf Image Database [2] (Figure 1). We are interested in building a smooth curve from such a leaf shape, as test data for curve algorithms, especially algorithms that benefit from the interesting tangent space of these organic shapes. Since it is well understood how to build a smooth curve from a point cloud, the problem reduces to building a good point cloud of these leaf boundaries.

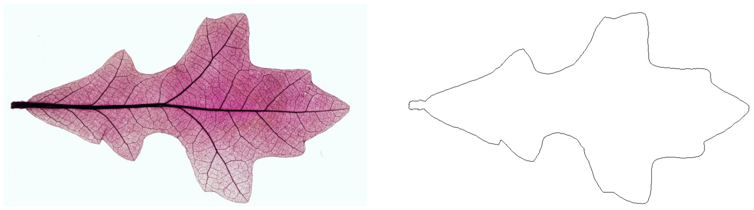

Fig. 1: Left: Image of post oak leaf. Right: A point cloud extracted from this image.

Figure 1 shows the point cloud extracted from this image by the algorithm presented in this paper. This appears to be a simple variant of the image segmentation problem, but standard algorithms in

OpenCV [7], a popular computer vision library, fail on this image (Figure 2). The solution is to address noise and scale in these images in creative ways, which is the focus of this paper.

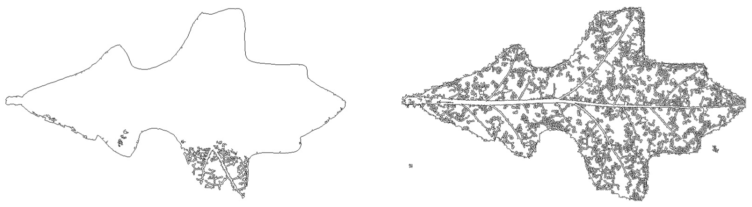

Fig. 2: Left: A point cloud extracted from the post oak image using the Otsu thresholding algorithm [8] in OpenCV. Right: A point cloud extracted from the post oak image using the adaptive thresholding algorithm in OpenCV.

Figure 3 illustrates another image/cloud pair developed by our algorithm, from an image of a red oak. Consider the gray-level histogram of this image, also in the figure. The image is noisily bimodal: an image with a foreground object of a noisily uniform intensity against a background of a noisily uniform intensity. Noise smears the grayscale values of the foreground and background into two noisily Gaussian hills, due to variations in pigment and lighting. The correct threshold for threshold segmentation lies in the valley between the two hills, but the computation of this threshold is slippery, as illustrated by the failure of two classical thresholding algorithms at finding a good threshold. We have found two insights to finding a good threshold, using two classic approaches to combating noise: Gaussian blurring and scale space.

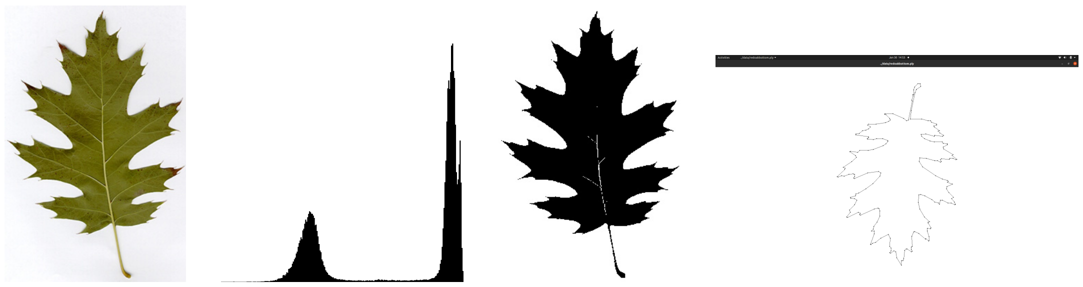

Fig. 3: Left: Image of red oak leaf.
Left Middle: Gray-level histogram of this image.
Right Middle: Bitmap extracted from this image by thresholding with our threshold.
Right: Point cloud extracted from this image.
(The point cloud is flipped upside down from the original image.)

Finding the threshold:

The threshold of interest lies in the valley between the two hills in the gray-level histogram, one (called the alpha hill) associated with the global maximum, and the other (called the beta hill) associated with a local maximum (but a global maximum after suppression of the alpha hill). One approach to find the alpha hill is to find the global maximum M of the histogram (intensity bin with most pixels) and walk out from the maximum while strictly dropping, defining a neighbourhood of M. However, due to noise, this is ineffective, since the alpha hill will not extend far enough. The first insight is that a better approach is to walk out from the max M while the binsize is larger than a 'zero' binsize z. Intuitively, any bin smaller than z would be empty but for noise. This reduces the problem to defining the 'zero' binsize z. Once z is defined, the alpha hill may be defined by the nonzero bins surrounding the global max, and the beta hill by the nonzero bins surrounding the secondary max.

The second insight is to find z using the blurred histogram, not the original histogram: in particular, z is defined by walking out from the global max of the blurred histogram, in both directions, while the binsize is monotonically dropping, and defining the zero binsize as the minimum binsize of this neighbourhood (necessarily one of the two boundaries of the neighbourhood). By blurring the histogram, the hill becomes truly monotonic. Once the zero binsize is defined, it is used to define the alpha hill in the original histogram, by walking out from the maximum while the binsize is greater than z. The beta hill is defined similarly after suppressing the beta hill. So the zero is defined by a monotonic walk in a blurred histogram, while the hill is defined by a zero-constrained walk in the original histogram.

We use a Gaussian kernel of width 9 to blur the histogram (viewed as a 1d signal) by convolution. The width of this Gaussian kernel is a parameter that can be manipulated by the algorithm, but this value has worked for the leaf images we have considered.

Once the alpha and beta hills are defined, the desired threshold is the midpoint intensity between the two hills. If the image is bimodal (foreground object against background), the two hills will not overlap, the threshold bin will be essentially empty (but not empty: see Figure 3), and indeed all of the bins outside the two hills should be essentially empty: this may be used to diagnose if an image is bimodal and the algorithm of this paper applies. All of the leaf images that we have considered (cleared leaves) are bimodal in this sense.

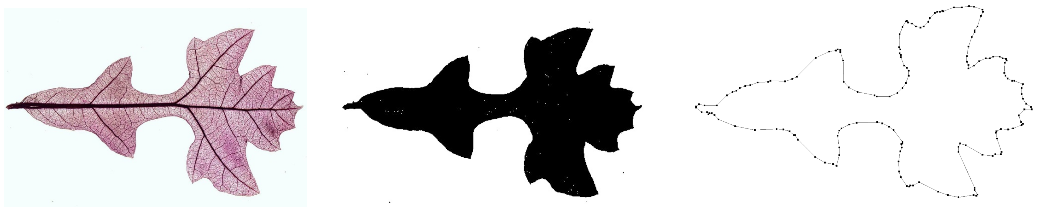

Fig. 4: Left: Image of post oak leaf (#11 in CLDB Quercus stellata group).
Middle: Bitmap extracted by thresholding.
Right: Point cloud extracted from this bitmap by contouring and decimation.
(This point cloud is flipped upside down from the original image.)

Once the threshold is defined, a bitmap is computed using binary thresholding (Figure 4), and this bitmap is contoured using Suzuki/Abe's contouring algorithm [9], yielding a collection of contours. Because of noise, many contours are found. For example, contouring yields 65 contours for the post oak leaf of Figure 4. But most of these contours are very short, and the longest contour is always the leaf. This longest contour is then decimated using Douglas-Peucker's algorithm [3], yielding a better point cloud for curve interpolation. For example, before decimation, the longest contour of the post oak leaf in Figure 4 has 976 points but 167 points after decimation, yielding the point cloud shown in the figure. The amount of decimation is a parameter of our algorithm, but we have found an epsilon of 1 to work well. Note that decimating the longest contour, and ignoring the short contours, further removes the noise inherent to the leaf segmentation problem.

Another adjustment is important for some images: resizing. Very large images, as many of the leaf images are, have too much detail. The algorithm is improved by shrinking large images so that their smaller dimension is 400. For example, the leaf image of Figure 1 has shape (1328, 2388). If the algorithm is applied to this image, 9784 contours are generated by contouring its bitmap. If the algorithm is applied to the resized image (shrunk to shape (400, 719)) 49 contours are generated. Resizing the image is a form of smoothing that further reduces noise.

Results:

The segmentation algorithm of this paper has been implemented in C++ using OpenCV [7]. It generates point clouds in PLY format, a standard mesh format that has the flexibility to be used for point clouds and has good readers/writers (e.g., Sharp's hapPLY [4]). We have built leaf clouds from 168 leaf images of the Cleared Leaf Image Database [2]. Two more examples are shown in Figure 5. We have also found other leaf images outside the CLDB dataset. Space prevents a fuller treatment of these other datasets, and their statistics.

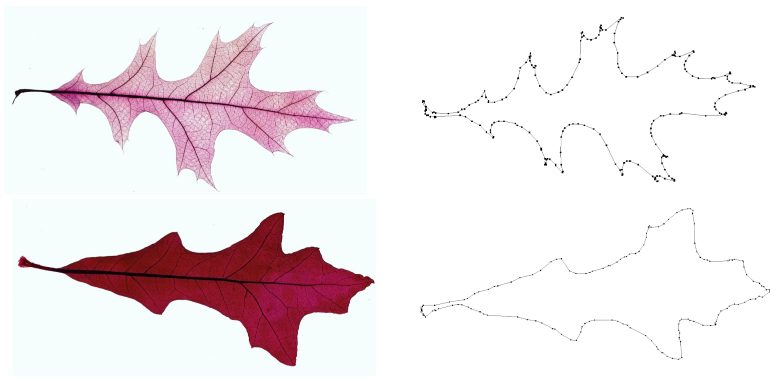

Fig. 5: Top row: Spotted oak leaf and its point cloud. Bottom row: Overcup oak leaf and its point cloud. (Point clouds are flipped upside down relative to image.)

Conclusions and future work:

We have built a dataset of leaf point clouds, which can then be used to build curve datasets for testing shape modeling algorithms on organic datasets with interesting tangent spaces (locally and globally, such as bitangents). The algorithm can be applied to any bimodal leaf image (e.g., cleared leaves), and has been successfully applied to 168 leaves of the Cleared Leaf Image Database. We hope that this resource will be useful to the research community as they compare and benchmark curve algorithms in shape modeling. It can be extended to build more curve datasets from similar images.

The success of the algorithm depends on handling noise carefully: noise (or unnecessary detail) in the image (by resizing), noise in the histogram (by blurring), noise in the 'zero' bin of a histogram (by computing this degenerate binsize by falling down monotonically in the blurred histogram), noise in the contouring algorithm (by eliminating short contours), and noise in the final leaf contour (by decimating).

A direction for future work is to apply this algorithm to other bimodal image classes, and to explore other leaf image datasets, such as the TensorFlow plant leaves dataset.

References:

[1] The Shape Repository of the Visualisation Visual Services (VVS): https://visionair.ge.imati.cnr.it/ontologies/shapes. (earlier, AIM@Shape)

[2] Cleared Leaf Image Database: https://www.clearedleavesdb.org.

[3] Douglas, D.; Peucker, T.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature, The Canadian Cartographer, 10:2, 1973, 112–122. https://doi.org/10.3138/FM57-6770-U75U-7727

[4] hapPLY: https://github.com/nmwsharp/happly.

[5] Johnstone, J.:A Parametric Solution to Common Tangents. Shape Modeling International (SMI), 2001, 240–249.

[6] ModelNet40: https://modelnet.cs.princeton.edu

[7] OpenCV: opencv.org.

[8] Otsu, N.: A Threshold Selection Method from Gray-Level Histograms. IEEE Trans Systems, Man, and Cybernetics, 9:1, 1979, 62–66. https://doi.org/10.1109/TSMC.1979.4310076

[9] Suzuki, S.; Abe, K.: Topological Structural Analysis of Digitized Binary Images by Border Following. CVGIP, 30:1, 1985, 32–46. https://doi.org/10.1016/0734-189X(85)90016-7

[10] Zhou, Q.; Jacobson, A.:Thingi10K: A Dataset of 10,000 3D-Printing Models. arXiv:1605.04797, 2016, https://ten-thousand-models.appspot.com

[11] Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J.: 3D ShapeNets: A Deep Representation for Volumetric Shapes, CVPR, 2015.

[12] Princeton Shape Benchmark: http://gfx.cs.princeton.edu/proj/shape/