Leaf Classification from Boundary Analysis

AMSC 663
Midterm Report
Fall 2007

Anne Jorstad
jorstad (at) math.umd.edu

Advisor: David Jacobs, Department of Computer Science
djacobs (at) umiacs.umd.edu

1 Abstract

In this project, I develop a system for leaf classification based on local analysis of boundary curves. The wavelet transform is used to generate a coefficient vector at each boundary point over several scales. The point vectors of many leaves are clustered, and the distribution of the points of each leaf over the clusters is then compared. Leaves with similar distributions are assumed to have similar local boundary characteristics. This system is meant to complement an already-implemented leaf classification system whose decisions rely entirely on global shape information.

2 Background

There is an ongoing project between members of the University of Maryland, Columbia University, and National Museum of Natural History Smithsonian Institution to create an electronic field guide for plants [1]. The ultimate goal of this project is to develop a system where a user in the field can take a picture of an unknown plant, feed it to the system carried on a portable computer, and have the system classify the species and display sample images of the closest matches in real time. A basic system has been implemented for the leaves of woody plants of the Baltimore-Washington, DC area, and manages to provide reasonable guesses of species classification for most examples. This database contains about 245 species in approximately 8000 images, and the classification is done by comparing the overall shape of each leaf to each specimen in the system. In my project, I extend the classification algorithm to examine local variation in leaf boundaries. Leaf edges can be smooth, serrated or lobed, and although these characteristics do affect the overall shape description of the leaves, the specific amount of variation in leaf edges has not been studied for this system. I develop a software module to classify leaves based exclusively on local information extracted from the leaf edges. We hypothesize that this will provide information independent from the current classification scheme, so that when combined with the current system, the classifier will be able to perform with higher accuracy.

3 Method

The current system examines the global shape of each leaf by calculating a distribution of Inner-Distances [3] across each leaf. This method works well for global characteristics, but does not take into account local boundary features. I use the wavelet transform to construct a complementary set of local information for each leaf boundary.

The boundary of each leaf is given as input, as a set of approximately 2000 discrete (x,y) points. The vectors are converted to complex one-dimensional (x+*i*y) points, in order to take advantage of 1-D algorithms. The discrete wavelet transform is then applied to the boundary vector. The wavelet transform converts a vector of *n* points into two vectors of *n*/2 points: a vector of approximation coefficients, which provide the best approximation for the original vector given only half as many points, and a second vector of detail coefficients, which provide the rest of the information to reconstruct the original vector. Because the goal of this project is to make decisions based on local information, it is the detail coefficients that will be used to classify each leaf. Reapplying the wavelet transform to the approximation coefficients generates the coefficients at the next coarser scale. For my experiments, I began by considering the detail coefficients at the first 5 scales.

For analysis, each of the original 2000 points should be represented by an *n*-dimensional coefficient vector, where *n* is the number of scales considered. Each application of the wavelet transform yields a coarser scale with half as many points as the previous finer scale. This means that after *n* transforms, there are 2^{-n} as many points as the original vector. In order to generate a full *n*-D vector for each point, 2^n wavelet transformation hierarchies must be calculated, starting at 2^n successive points in the vector, and applying periodic boundary conditions. The wavelet coefficients are calculated with the discrete wavelet transform in Matlab using a Daubechies-2 wavelet basis.

At this point in the algorithm, the input leaves have been converted to approximately 2000 *n*-dimensional vectors. It is desired that two leaves which are the same up to a rotation should be recognized as identical, so in order to enforce this rotation invariance, one more processing step must be applied to the coefficients before they can be classified. As we are now considering each boundary point as a separate entity, we are really attempting to classify the smoothness of each point. The coarsest scale of coefficient is rotated to lie strictly on the x-axis, and all other coefficients of that point vector are rotated by this same angle. This transformation provides a uniform orientation to all points, imposing the desired rotation invariance, and effectively reduces the degrees of freedom by one, to *n*–1.

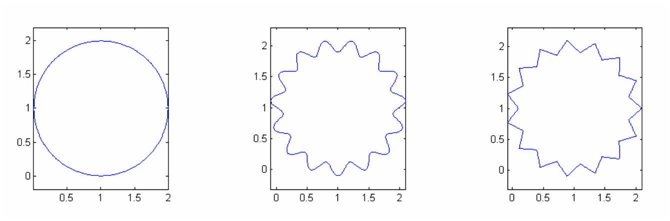
These vectors must now be classified. The basic idea is to compare distributions between coefficient vectors, where vectors with similar distributions are assumed to represent similar leaves. The most straightforward way to look at the coefficient distributions is to consider each scale separately. The coefficients for each scale can be sorted into equally spaced intervals by size, and the resulting histograms are then normalized to create a distribution. These distributions were compared across leaves, but through many examples it became clear that looking at each scale individually does not provide enough information to make any reliable classification decisions.

To provide useful classification information, each *n*-dimensional vector must be considered as a whole. A clustering routine was implemented, as sorting coefficients based on data-appropriate cluster centers provides much more relevant information than simply sorting into equally spaced intervals with arbitrary boundaries. The K-Means clustering scheme was chosen to perform the clustering. In this algorithm, k cluster centers are chosen at random from the data set, and each point in the full data set is assigned to its closest cluster center. The cluster centers are then redefined to be the mean value of all data points associated with each cluster. This process is iterated until the cluster centers stop changing locations. The data set here consists of every boundary point for every leaf in the input data. For the full system, this is approximately 8000 leaves * 2000 points per leaf. One important artifact of K-Means clustering is that the final cluster centers are highly dependent on the initially chosen points, so to provide reliable information, it must be ensured that the distances between distributions of leaves remain similar over several different sets of final cluster centers. The number of clusters, k, was chosen to be 36. Empirically, this value should be between about 25 and 50, as fewer than this and all the distributions look about the same, and much more and the exact boundaries between the clusters start to play too large of a role. The value of 36 has been used as the number of clusters for a variety of published experiments [6].

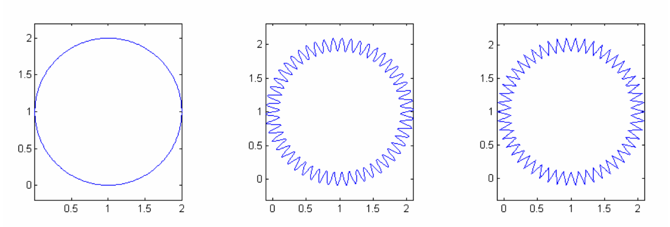
For each individual leaf, a distribution of its coefficients can then be found over the 36 cluster centers. This distribution is compared to the distributions of all other leaves, and the closest match is then returned as the classification decision. In practice, the top ten or twenty matches are returned in ascending order. The chi-squared distance was used to compare distributions.

4 Validation

To ensure that the output of the above algorithm is appropriate, after each segment was tested separately, a very simple input was processed. Three test cases, a circle, a circle plus a sine curve, and a set of straight lines joined into a circular star, were constructed (see fig. 1).



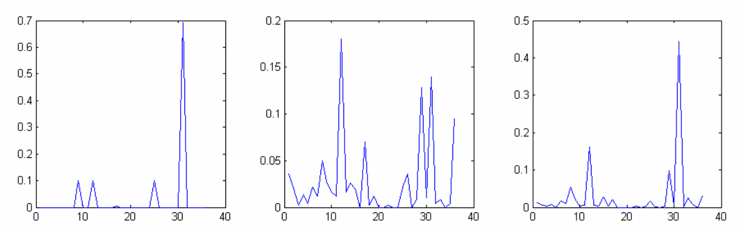
(a)



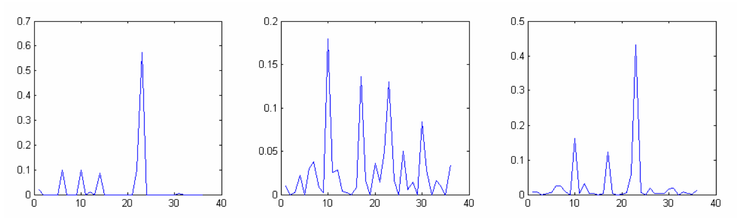
(b)

Figure 1: Simple validation curves. (a) Visual to understand the properties of the curves, with 15 peaks in each circle, (b) the actual curves tested, with 50 peaks in each circle.

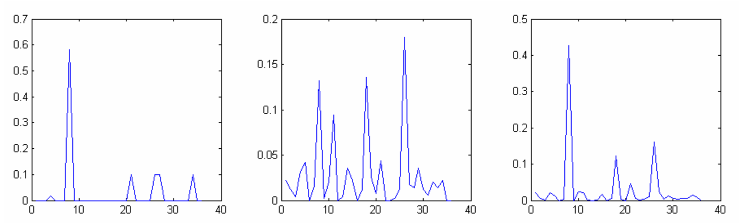
Running the algorithm many times, it is expected that the cluster centers change, but the general layout of the distributions should not. This can be tested by ensuring that the distances between distributions over several runs remain similar. From these tests, we observe that the actual locations of the cluster center peaks do change, but the general distribution of peaks remains similar (see fig. 2). Comparing the final distances between the three curves over several runs, we see that these relative distances remain close (see fig. 3). This is the desired result.



(a) first run

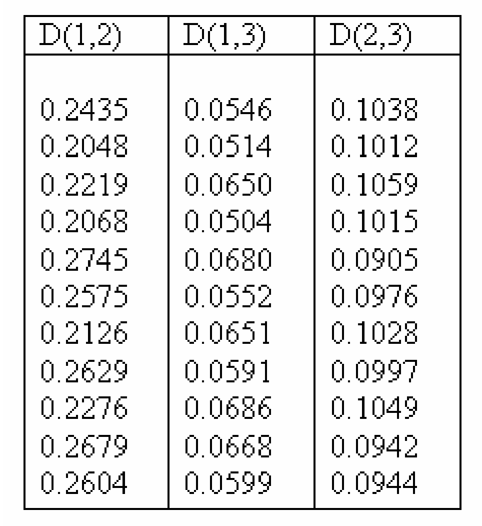


(b) second run



(c) third run

Figure 2: Distribution over cluster centers for validation curves.
Left plots: circle; center plots: circle+sine curve; right plots: circular star.



D(1,2)	D(1,3)	D(2,3)
0.2435	0.0546	0.1038
0.2048	0.0514	0.1012
0.2219	0.0650	0.1059
0.2068	0.0504	0.1015
0.2745	0.0680	0.0905
0.2575	0.0552	0.0976
0.2126	0.0651	0.1028
0.2629	0.0591	0.0997
0.2276	0.0686	0.1049
0.2679	0.0668	0.0942
0.2604	0.0599	0.0944

Figure 3: Distances between (1) circle, (2) circle+sine curve,
(3) circular star, over many trial runs.

5 Test Results

Considering elements of the leaf database, the first test was constructed to compare three very different leaves (see Fig. 4). These leaves are easily distinguished by the already implemented system, as their global structure is quite distinct, but it is relevant to know how my algorithm which only considers very local boundary information handles these cases.

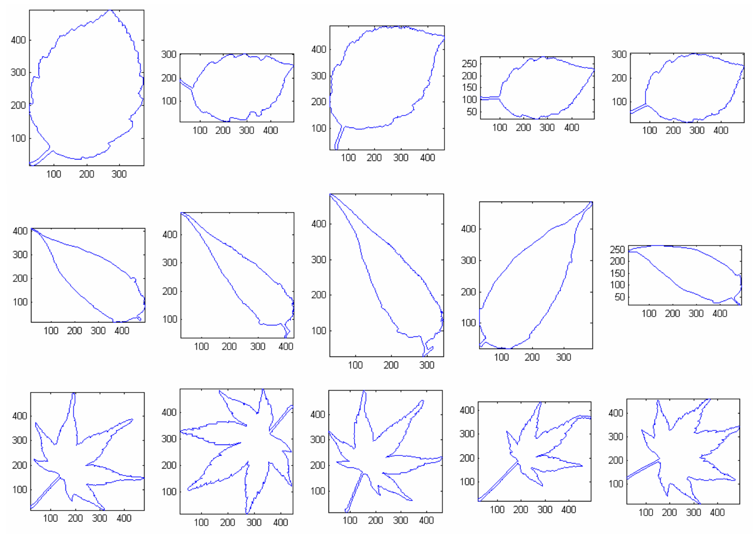
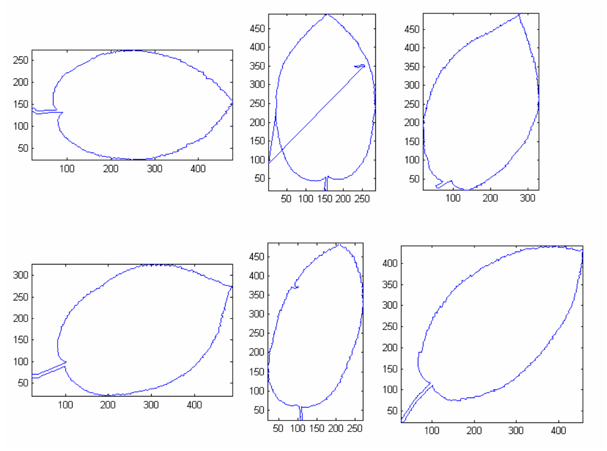


Figure 4: Very different trial leaves.

To test these leaves, cluster centers constructed from the points on all fifteen leaf boundaries were determined, and then the distribution of each leaf over these centers was calculated. The distribution of each leaf was compared to the distribution of each of the outer fourteen leaves, and the smallest chi-squared distance determined the closest match. This is different from the eventual program, which will have all cluster centers pre-calculated, and the distribution of each new leaf will be found over these given clusters.

My implementation was able to correctly identify each of these three species 60% of the time, which is sufficient. This just tells us that the classification scheme is not unreasonable. The original system can do much better, as expected, because the differences between these leaves are at the global scale.

To test leaves that the original system does not distinguish well, which was the point of this project, I started with two leaf examples that are of the same genus, but different species (see fig. 5). The edges of the first species are slightly smoother than that of the second, but otherwise the leaves have essentially the same global structure.


(a) Amelanchier Arborea

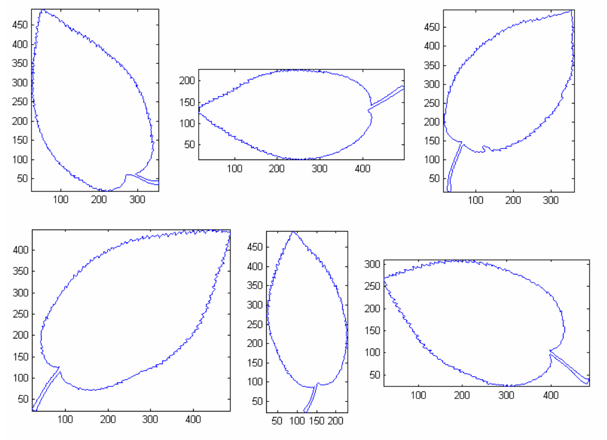

(b) Amelanchier Canadensis

Figure 5: Two very similar leaf species, where the boundary of (a) is slightly smoother than the boundary of (b).

The first full run I calculated using five scales of coefficients, and the results obtained were no better than chance (3/6 of the first were matched correctly, 4/6 of the second). This implied that my original hypothesis of about five scales being useful was incorrect. The code was run again using one, two, three, four, and five scales, with several runs at each scale to ensure consistency, and the results are shown in figure 6.

# of scales	*Leaf 1*	*Leaf 2*
1	33%	50%
2	83%	67%
3	100%	100%
4	67%	87%
5	50%	67%

Figure 6: Similar leaf results over several runs.

It is immediately seen that for this example, using three scales is the clear winner, with 100% correct classification results. This one example is not enough to declare that my implementation will do a good job of classifying all leaves with similar global shape but distinct shape at the very local level, but it is an encouraging starting result.

6 Future Work and Timeline

To better understand how my classification scheme is performing, I need to run tests on larger data sets, containing several examples of several different types of similar leaves. I need to experiment with the variables of the system, including number of scales, number of cluster centers, and type of wavelet. I would also like to update my system to use stationary (undecimated) wavelets, because these will construct more accurate *n*-dimensional vectors at each boundary point. I hope to be finished with this by the end of January.

Once the wavelet system is working satisfactorily, I will put that aside for the time and extend the current Inner-Distance algorithm over a hierarchy of scales. The current system makes all its calculations based on 64 evenly spaced points around each leaf boundary. I will add on the order of 16 points between each of these points, and perform local Inner-Distance calculations, between these points and the those of only the neighboring original points. This will provide more local edge information. I hope to complete this work in February and March.

Once, I am satisfied that the wavelet system and the local Inner-Distance system are both performing as desired, I will then combine them with the already implemented Inner-Distance algorithm. This will involve determining a weighting between the three distances, so that a better overall classification scheme is obtained. This combined classification then needs to be tested. This work may not be completed until the summer.

I will begin writing my results in April, and have the final report prepared in May.

References

[1] Gaurav Agarwal, Haibin Ling, David Jacobs, Sameer Shirdhonkar, W. John Kress, Rusty Russell, Peter Belhumeur, Nandan Dixit, Steve Feiner, Dhruv Mahajan, Kalyan Sunkavalli, Ravi Ramamoorthi, Sean White. “First Steps Toward an Electronic Field Guide for Plants”. Taxon, vol. 55, no. 3, Aug. 2006.

[2] Cene C.-H. Chuang, C.-C. Jay Kuo. “Wavelet Descriptor of Planar Curves: Theory and Applications”. IEEE Transactions of Image Processing, Vol. 5, No. 1, January 1996.

[3] Haibin Ling, David W. Jacobs. “Using the Inner-Distance for Classification of Articulated Shapes”. IEEE Conference on Computer Vision and Pattern Recognition, vol. II, June 2005.

[4] Ingrid Daubeshies. “Ten Lectures on Wavelets”. Society for Industrial and Applied Mathematicians, Philadelphia, PA, 1992.

[5] Pedro F. Felzenszwalb, Jushua D. Schwartz. “Hierarchical Matching of Deformable Shapes”. IEEE Conference on Computer Vision and Pattern Recognition, 2007.

[6] Jitendra Malik, Serge Belongie, Thomas Leung, Jainbo Shi. “Contour and Texture Analysis for Image Segmentation”. International Journal of Computer Vision, vol. 34, no. 1, July 2001.

[7] Stephane Mallat. “A Wavelet Tour of Signal Processing”. Academic Press, Chestnut Hill, Massachusetts, 1999.