Identification of Artistic Styles Using a Local Statistical Metric

Peter Wayner
Cornell University and Xerox Palo Alto Research Center
Dept. of Computer Science
Ithaca, New York 14853
wayner@cs.cornell.edu

Abstract: This paper describes an algorithm for identifying the artist who created a picture. The algorithm relies upon computing the distribution of long and short lines in the image and comparing this distribution. The results of testing the implementation on the daily comics is presented. Some of the related work in both computer science and art history which provides a conceptual background for the algorithm is also discussed.

AI topic: Machine Vision and Pattern Recognition
Domain Area: Art
Language/ Tool: Pascal
Status: Experimental
Effort: Three months, one person.
Impact: May help us expand the canonical questions of Artificial Intelligence. Provides an example of a question which humans might consider requires "skill" or "training" but can be solved robustly by a computer.

Humans can rely upon loosely defined rules to answer questions. The mountaineer might point out that the shapes of the peaks is a good way to distinguish the range. The Alps are craggy while the Adirondacks are smoother. The art historian knows that Georges Seurat is known for using small dots of paint to create the image while Al Hirschfeld uses long pen lines to define his forms. Examining the brush strokes is one clue to the artist.

The algorithm in this paper is one example of algorithms which can be designed to answer questions about global characteristics. This particular example computes line lengths throughout a drawing and compares the averages against a pre-computed set from model samples. The distribution of line lengths is a statistical characterization which can be used to distinguish between artists. The current implementation is limited to black and white binary images, but the principles of it should be easy to extend to include color and other forms of data. The algorithm is also easy to implement on a parallel machine.

The Relationship with Other Research

This work is quite thematically similar to a wide range of other topics in vision, artificial intelligence and English literature. For instance, much of the early work on the characterization of the texture of an image is largely statistical. Two good surveys of texture articles were written by David Haralick [3] and Van Gool, Dewael and Oosterlinck [11]. The concrete results of the texture algorithms are often surprising. In separate articles, Haralick and Sahumagam summarize their use of texture sensitive algorithms to identify objects. The type of land features in Landsat photographs was successfully categorized as either coastal forest, woodlands, annual grasslands, urban areas, large irrigated fields, small irrigated fields or water in [4,5]. Another algorithm could identify the rock types based on the texture.[6]

This research also has some thematic similarities with the research often known as stylistics done on using statistical profiles to identifying the author of a text. The books by James Oakman [10] and Stanley Fish [11] contain good summaries of the scope of other stylistic research and discuss important problems with it. One of the classic works in the area is a book by Frederick Mosteller and David Wallace which attempts to use the frequency of word usage to identify the author of the the twelve unsigned Federalist papers. They found, for instance, that Alexander Hamilton utilized the word "enough" in other papers he wrote while James Madison didn't. The actual conclusion (Madison wrote all twelve) was based on a larger set of words with similar discriminating capabilities.

Many of the classical artificial intelligence literature has been devoted to statistical classifiers. (ID3 etc.) These algorithms take a sample set of pre-classified items and a multi-dimensional space of measurements and partition the measurement space into regions corresponding to each classification. These are often used in visual
applications like classifying cells by measurements like length.

**What is style?**

Roughly speaking, the algorithm classifies the "style" of an illustration and matches it to a set of known images. Defining just what "style" is a difficult problem because "style" is not an immediately quantifiable concept. The definitions of words like "length" or "weight" are essentially algorithms for computing a value. They are much like constructive proofs. "Style," though, is a nebulous quantity which is impossible to define rigorously.

In this paper, style will be treated as another word for a collection of rules of representation which govern the way the drawing captures an image of the world. The paintings of Vincent Van Gogh could be characterized by a garish color scheme and large, swirling brushstrokes. Pablo Picasso’s early cubist inventions disregarded the rules of perspective and created his own set of rules for that period. Art historians would readily cite these differences as the rules they used to distinguish the styles from each other. This paper describes a simple metric which captures some aspect of the rules of representation—in this case a function of average line length and curvature used by the artist. This metric is obviously quite limited, but the results show that even such a limited statistic can be very accurate.

**The Details of the Algorithm**

This algorithm characterizes the "style" of binary images as a histogram of line lengths in the image. The histograms of different images can be averaged to produce models of the average distribution for a class of images. The histograms can also be compared with other histograms and models to determine their similarity using a distance function. The algorithm can identify a sample by comparing it to the models and selecting the model with the smallest distance function value, i.e. the closest one in space.

This algorithm defines the length of a line through a "on" pixel (x,y) as the longest straight line segment through the point which passes strictly through "on" pixels. The distribution of the length of these straight line is collected in a histogram of line length versus number of occurrences. More precisely, the algorithm computes the histogram \( H(l) \) of \( m(x,y) \) for all pixels (x,y) in the image that are "on" where

\[
m(x,y) = \max \, D(x,y, \delta_i) \text{ for all } i.
\]

\[
D(x,y, \delta_i) = \text{the length of the longest straight line segment which can be drawn through } (x,y) \text{ at angle } \delta_i \text{ and completely runs through "on" pixels.}
\]

\[
H(l) = \text{the number of pixels } (x,y) \text{ for which } m(x,y) = l.
\]

For example, if an image is strictly made up of 3 5x1 pixel horizontal lines running the length of the image, the histogram will show that 100% of the "on" pixels had lines of 5 pixels long running through them. If the image was made up of circular dots of radius r, then the histogram would show 100% of the on pixels had lines running through them of length approximately r. (The discrete nature of digitized images introduces some error here.)

An image made up of T-shaped glyphs with vertical segments 1x6 pixels and horizontal crossbars 5x1 pixels long would show 60% of the points with values of m equal to 6 and 40% of the points having values of m equal to 4. The shared point has an measure of 6 because the vertical line is the longest of the two lines which can be drawn through it. Naturally, the images discussed here will have a much more complex distribution. These examples are illustrated in Figure 1.

The accuracy of the measure function, as implemented, depends upon the selection of angles it considers. The algorithm computes m by measuring the lengths of the on lines at a set of fixed angles \( \delta_1 \ldots \delta_n \). The maximum is found over this set of values of D. Adjusting the number of angles and their orientation controls the amount of computation and the sensitivity to lines oriented in particular directions. This implementation has only experimented with samples taken at evenly space intervals.

The speed of the algorithm is affected by the density of "on" pixels on a page. The algorithm takes \( O(cn^2) \) where c is a constant which is style dependent and essentially the average value of m. Implementations using specialized hardware are certainly possible.

This algorithm measures a mixture of curvature and length. The function m is only designed to measure straight-line distance, not the length of connected components. For this reason, it is sensitive to the distribution of straight lines in an image. It is also be affected by amount of curvature in a curved line. If (x,y) is some point in a highly curved, thin section of "on" pixels, then m(x,y) will...
be small because the longest straight line segment which can be drawn through "on" pixels will still be quite short.

The Results

The algorithm and the difference function were tested on a sample of seven different comic strips. The daily, black-and-white comic strips are ideal sources of data for this algorithm because they do not contain many of the extraneous experimental problems of control which could introduce side-effects. Comic strips are all produced in a uniform size with identical reproduction. This removes many of the potential sources of error which might enter the data because the different samples took different technological routes between the artist's hand and the computer's memory.

The test used six samples of each of seven strips: Calvin and Hobbes, Peanuts, Shoe, Dick Tracy, Hi and Lois, Snuffy Smith and Doonesbury. The strips were digitized at 200 pixels per inch and the final images were about 500x1500 pixels. The line lengths at each pixel were computed by measuring the length at sixty equally-spaced angles. The distribution of line lengths over each of the image was computed and an average distribution of line lengths over all six samples was constructed. The samples were identified by comparing the histogram with the seven average histograms. The histograms were compared by computing the sum of the squares of the difference at different lengths. i.e.

$$\sum_{i=1}^{n} (H_1[i] - H_2[i])^2$$

In all, the algorithm made only two mistakes out of the 42 trials. In both of these cases, the correct answer was the second choice. Graphs which show the average distributions $H$ of the various comic strips will be included in the second volume with slide pages.

While it is impossible to certain about the reasons for the error, it is possible to examine the strips and propose hypotheses. Both of the strips used an uncharacteristic amount of black silhouettes. These obviously added many extra long lines and affected the distribution in the histogram. A more robust algorithm might have the ability to isolate certain line segments.

Some Roots in Art History

The work of William Ivins [7], who was curator of the Metropolitan Museum of Art's print collection, provides some insight into why the algorithm works. He saw the history of prints as the history of the development of a visual syntax for communicating three-dimensional reality. Over the years, artists created and passed on different ways of cross-hatching, shading and stippling the engraving to produce the illusion of three-dimensional images. Ivins presents a number of examples of the same scene rendered by different artists from different times. The eye perceives the same objects, but the differences in the style are obvious. Each of the different artists had a different set of rules for representing objects.

Giovanni Morelli [8] was a connoisseur in the late nineteenth century. His book, Italian Painters; Critical Studies of their Works, analyzes the minute brush strokes of the painters of the time and notes that these are often enough to distinguish between them. Some of the later work of Max Friedlaender, [2] which is summarized in his book On Art and Connoisseurship however rejects this approach as too narrowly concentrating on details.

Conclusion and Further Directions

This paper describes a simple algorithm for measuring the difference in visual styles and reports the results of testing it on a selection of samples from the daily comics. The algorithm proved to be right more than 95% of the time. Extending the principles to work with color data must take information about the human perception of color as well as a general metric for measuring color difference. More rigorous experimentation is also necessary to help determine better and more robust algorithms. There are many different ways of categorizing the shapes and arrangements of lines in a picture and it is unclear what the main principles are that are effecting the classification.


Figure 1:

Figure 1a: The histogram of this picture will show that the longest line through every pixel is 5 pixels long. That is, a $H[5]=1$ and $H[i]=0$ for all $i<5$. The shaded sections represent the "on" pixels.

Figure 1b: Two different images with the same histograms. In both of these cases $H[5]=.4$ and $H[6]=.6$.

Figure 1c: An image with a histogram of $H[4]=2/7$ and $H[5]=5/7$. 

References:


