Sketch Recognition Using Vector Graphics

Ammar Hattab
Computer Engineering Department
Brown University
Providence, USA

James Hays
Computer Science Department
Brown University
Providence, USA

***Abstract*—this paper explores the use of vector graphics for sketch recognition, with the goal of enhancing the classification accuracy on a dataset of 20,000 sketches built in a previous paper [Eitz et all. 2012]. This paper explores three different ways that vector graphics could be used for sketch analysis and recognition, the first is to extract usual low level features like HOG but using vector graphics, then the use of global features, and at last the use of curve matching algorithms to match different sketches. The paper proposed a new method that combines local sketch features with pairwise sketch matching to achieve the best detection accuracy**

Keywords—Sketch Recongition; Vector Graphics; SVG; HOG Features; Shape Matching;

I. INTRODUCTION

Vector graphics stores the graphics data as drawing instructions rather than array of pixels which would contain many empty pixels taking unnecessary space, by saving that space vector graphics allows more efficient computations, at the same time in the process of converting the vector graphics to bitmaps (rasterization); the original information of strokes, points, and the order of drawing is lost, and there is no way to map the output features back to the original strokes, so keeping the data in vector graphics preserves all the original information provided by the user, this gives us the opportunity to exploit it for a better recognition algorithm with a higher accuracy.

Although the great use of vector graphics in all kind of applications, there are a few studies that explored their direct use for recognition, [Sciascio et al. 2004] built a retrieval system for SVG documents based on shape similarity which could be measured by geometrically matching the query shape to the dataset shapes, and retrieving the most similar ones that have the minimum matching distance, which could be the sequence edit distance [Jiang et all. 2007], Euclidean distance between anchor points [Rayashi et all. 2008], or Earth Mover's Distance (minimum transportation cost) [Hayashi et all. 2014], some other studies extracted global shape features like moments invariants [Mascio et all. 2010] and used them for matching. More recently a non-shape similarity measure based on style (color, shading and texture) was developed for vector graphics retrieval [Garces et all. 2014], while several of these methods provided impressive results; non of them were applied to hand sketches, nor they evaluated the accuracy of the retrieval on a large database.

In the other hand, there are too many features and methods that are built using bitmap images, which dominate the majority of computer vision research, most low level features like SIFT, HOG or SURF were designed for bitmap images, and there is no direct way to compute them using vector graphics without rasterizing it first, this forced previous research on sketches [Eitz et all. 2012] to convert data originally obtained in vector graphics format to bitmap images.

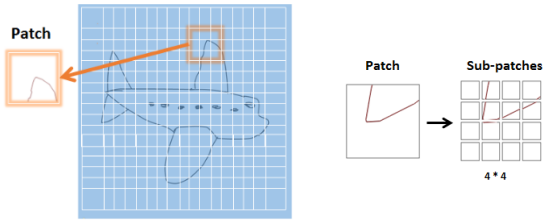
The goal of this study is to provide a simple way to compute low level features like HOG from vector graphics data, and to explore the different methods for using vector graphics in hand sketch recognition, and to evaluate them on a large sketches dataset, possibly enhancing the previous recognition accuracy.

II. EXTRACTING HOG FEATURES USING VECTOR GRAPHICS

Histogram of oriented gradients (HOG) are widely used low level features for object detection, originally designed for bitmap images [Dalal et all. 2005], they starts by extracting image gradients, then creating a histogram of gradients orientations in a localized patches of the image. Extracting HOG features using vector graphics requires a different operations but produces similar results

A. Extracting patches

The first step is to extract local patches, while in bitmap images this operation is trivial, in vector graphics its not, where we should find the intersection points of each stroke line (of the sketches strokes) using the line equation with every local patch square (within the sketch bounds) and then generate new lines using the intersection points.



Note here that the intersections are found independently for 4 X 4 = 16 small sub-patches of each patch.

B. Histogram of Orientations

Then for each sub-patch we have to find a histogram of orientations of lines in the sub-patch using 4 orientation bins (0 – 180), and this is done by computing the angle of each line using the two line points, then the length of the line is added to

the bin corresponding to the angle with interpolation for the adjacent bins, then the HOG feature vector for each patch is formed by concatenating the histogram bins of the 16 sub-patches, forming a 64 features vector.

C. *Completing the pipeline*

To provide comparable results, we followed the same classification pipeline used in the original sketches paper [Eitz et all. 2012], where we applied k-means clustering to cluster the patches into visual words (k=500), and then building the bag-of-words features [Sivic et all. 2003] for each sketch by quantizing the sketch patches using a soft kernel assignment to the different visual words (clusters), and we saved the resulting bag-of-words features for all sketches

D. *HOG Result*

To measure the accuracy we used 3-folds cross validation, where the data is divided into three parts (two for training and one for testing), and in each fold we trained a one-vs-all SVM classifier (using C=3.2 and Gamma= 17.8) and measure its accuracy, the total accuracy is an average of the 3-folds accuracy.

We got an accuracy of 45.8% using the features built using vector graphics, and 52% using the features built using bitmap images, the two accuracies are similar, and we could reduce the difference by applying spatial interpolation in the vector graphics case (applied in bitmap images case).

The result shows that we could get HOG features with similar accuracy (and fewer computations) when using vector graphics, this implementation would allow other authors to extract low level features from vector graphics in a more natural way (without the need to rasterize it) and also to map the resulting HOG features to original stroke data, allowing for a wider range of applications (like synthesis applications), or the possibility to attach other features from the same strokes.

III. Adding Global Features

The second method we explored is to add global features, we started by computing strokes lengths and points counts from different sketches, then started adding other features, while there several global features that could be used [Yang et all. 2008] we decided to use moments invariants, as the image moments encompass many other global features which could be extracted from the moments (like the centroid, area, orientation, skewness, flatness…etc), and they could be computed easily using one formula.

$$M_{ij} = \sum_x \sum_y x^i y^j I(x,y)$$

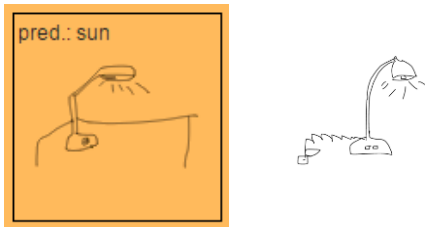
We chose the 7 moments invariants of Hu [Hu 1962] which are invariants to changes in translation, scale or orientation) and computed them for each sketch.

We tried two ways to combine the global features with the previous bag-of-words local features, at first we tried to normalize the features and concatenate them onto one features vector and used one SVM classifier on top of that, the second way is to use two separate SVM classifiers, one for the global features and one for the local features, then in testing we multiplied the probabilities output of the two SVM classifiers to get the final result.

We repeated the 3-folds cross validation procedure, and in both cases we got small or no effect at all when adding the global features, which suggests that the global features are weak compared to the local features, which seems to be strong enough to prevent any further enhancement by the global features.

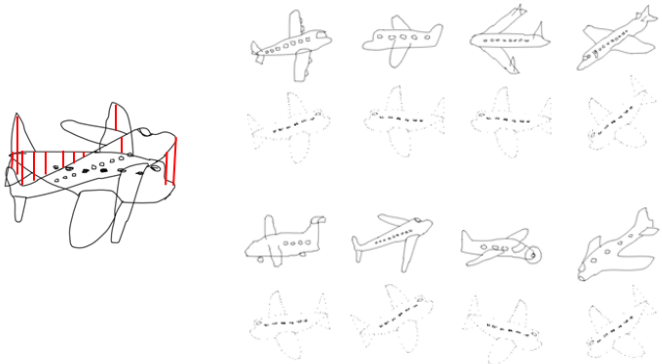
IV. Shape Matching

The third method we applied is using shape matching, because the problem with features is that they do not care about the spatial arrangement or the geometry of the shape, for example the following table lamp sketch is wrongly classified as "Sun" because it has a local appearance of the sun, while if we match it closely with a table lamp sketch the result will be closer to the table lamp than the sun.



A. *Iterative Closest Point (ICP)*

We used a simple matching procedure that proved effective where two sketches are first aligned using iterative closest point (ICP) procedure [Arun et all. 1987], then the Euclidean distances between the points of the two sketches are combined and the total is computed as the matching distance.



Since the matching takes quite a long time (depending on the training dataset size, for example to match 20,000 X 20,000 sketches, it would need 277 days to finish), we choose to do matching between the query sketch, and the top 10 categories returned by the SVM classifier trained on the bag-of-words features.

We should notice that on the sketches dataset, using the SVM classifier, 80% of the time, the correct sketch is within the top 10 categories.

B. *Matching Result*

We couldn't apply the same 3-folds validation procedure on all categories as it also takes a long time with matching, so we chose to test it on the hardest category "Monkey", and the accuracy increased from 7.4% to 19% when using matching, and we tested it also on "bottle opener" category, where the accuracy increased from 14.8% to 26%, so in conclusion the use of shape matching increase the accuracy but takes much longer time, where for a new sketch, the algorithm takes 1 minute to classify it, that could be made faster by using parallel computing, or a faster and better matching algorithm.

V. Training Data

We noticed that the 20,000 sketches dataset still contains several bad sketches that should be cleaned or at least not used in training, so we manually selected the best 25 sketches from each category (~ 30%), resulting in a total of 6250 sketches, we applied the 3-folds cross validation procedure on this min-sketches dataset, and we got an accuracy of 52% which is higher than the accuracy (~44%) reported by the sketches paper [Eitz et all. 2012] on the same data size.

Conclusion

Vector graphics provide more possibilities for analysis and recognition of sketches, as it provides more information about the sketch while requiring less size, which leads to a higher accuracy and more efficient computations, but their use for recognition requires carefully selecting a method that exploits the information provided by the vector graphics.

References

[1] Eitz, Mathias, James Hays, and Marc Alexa. "How do humans sketch objects?." ACM Trans. Graph. 31.4 (2012): 44.

[2] Di Sciascio, Eugenio, Francesco M. Donini, and Marina Mongiello. "A logic for SVG documents query and retrieval." Multimedia Tools and Applications 24.2 (2004): 125-153.

[3] Yang, Mingqiang, Kidiyo Kpalma, and Joseph Ronsin. "A survey of shape feature extraction techniques." Pattern recognition (2008): 43-90

[4] Jiang, Kaiyuan, et al. "Information Retrieval through SVG-based Vector Images Using an Original Method." e-Business Engineering, 2007. ICEBE 2007. IEEE International Conference on. IEEE, 2007.

[5] Rayashi, T., et al. "Retrieval of 2D vector images by matching weighted feature points." Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on. IEEE, 2008.

[6] Hayashi, Takahiro, Akihiko Sato, and Nobuaki Ishii. "Similarity Retrieval of Vector Images with Indirect Matching." International Journal of Semantic Computing 8.02 (2014): 169-183.

[7] Di Mascio, Tania, Daniele Frigioni, and Laura Tarantino. "VISTO: A new CBIR system for vector images." Information Systems 35.7 (2010): 709-734.

[8] Garces, Elena, et al. "A Similarity Measure for Illustration Style." To appear in ACM TOG 33: 4.

[9] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Vol. 1. IEEE, 2005.

[10] Sivic, Josef, and Andrew Zisserman. "Video Google: A text retrieval approach to object matching in videos." Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on. IEEE, 2003.

[11] Hu, Ming-Kuei. "Visual pattern recognition by moment invariants." Information Theory, IRE Transactions on 8.2 (1962): 179-187.

[12] Arun, K. Somani, Thomas S. Huang, and Steven D. Blostein. "Least-squares fitting of two 3-D point sets." Pattern Analysis and Machine Intelligence, IEEE Transactions on 5 (1987): 698-700.