Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (2008)

Distinguishing Between Sketched Scribble Look Alikes

Katie Dahmen and Tracy Hammond
Sketch Recognition Laboratory, Department of Computer Science, Texas A&M University
Mail Stop 3112, College Station, TX 77839
979 845 7143, kdahmen@cs.tamu.edu; 979 862 4284, hammond@cs.tamu.edu

Abstract

In hand-sketched drawings, nearly identical strokes may have different meanings to a user. For instance, a scribble could signify either that a shape should be filled in or that it should be deleted. This work describes a method for determining user intention in drawing scribbles in the context of a pen-based computer sketch. Our study shows that given two strokes, a circle and a scribble, two features (bounding ratio and density) can quickly and effectively determine a user's intention.

Introduction

Sketch recognition with a Tablet PC is a potentially useful and natural way for humans to interact with computers for many applications. Our goal is to create free sketch recognition systems that recognize shapes based on what they look like rather than forcing users to expend effort to learn a number of particular stylistic gestures (as in Rubine 1991, Long et al. 2001). This can be a difficult task when the same shape has different meanings given a subtle context which is obvious to a human but not necessarily to a machine.

Sheet music is a classical example where a single shape, specifically a scribble on the circle of a half note, defines that the half note should be deleted or filled in (i.e. turned into a quarter note). Figure 1 shows user examples of how similar the two different intentions can look to the human eye. The Music Notepad (Forsberg et al. 1998) handles this by having separate gestures for specifying quarter notes and deletion. However, in our attempt to create a free-sketch recognition system, we would like to be able to distinguish between the two intentions through context as a human would, without requiring any human-learned interaction.

Implementation

We conducted a user study to collect data of scribbles paired with circles as context. We had eleven users of varying ages and occupations, about half of whom were computer science graduate students having some familiarity with sketch recognition and Tablet PCs. We asked them to draw ten samples of a circle stroke with a scribble stroke to fill it in and then to draw ten samples of a circle stroke with a scribble stroke to delete it. We obtained 102 usable fill samples and 95 usable deletion samples. With this data, we calculated values of features of the strokes and looked at different combinations of features to try to distinguish between types of strokes. We looked at these features:

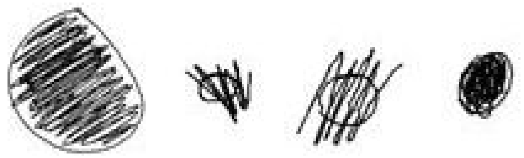


Figure 1: Sample scribble images.

- Density: the length of the scribble stroke divided by the area of its bounding box
- Speed: the average speed of the scribble stroke
- Bounding ratio: the smaller ratio of the circle width divided by the scribble width or the circle height divided by the scribble height
- Number of intersections: how many times the scribble stroke intersects the circle stroke.

Results

Our best results were found combining bounding ratio and density or number of intersections and density. The recognition rate for the data we have so far collected is 97.46% using bounding ratio with density (Figure 2(a)) and 98.98% using density with number of intersections (Figure 2(b)). All of the deletion samples are correctly classified, and the only incorrectly classified samples were in the fill set. However, these rates are calculated based on what users drew, and all four of the misclassified samples we found appear ambiguous or look more like delete than fill samples according to our human judgement (Figure 3).

Using the number of intersections, while possibly a more intuitive choice for identifying intention to fill versus to delete, has a couple of drawbacks when compared against the bounding ratio. One is that it is more time intensive to calculate, especially on very long strokes, such as when the users thoroughly color in their shapes. One such fill example had a scribble stroke 5259 segments long, which is a large quantity if every segment must be checked against every segment of another shape to test for intersections in a real-time setting. The intersection calculation method we used was a brute force method that takes $O(n^2)$ time, and while a faster method likely exists, we have not come across one that takes near $O(n)$ time. Calculating the bounding ratio, on the other hand, does take linear time given two strokes to compare. The other drawback to using number of intersections is that we worry that some people may just slightly overlap the boundary line with their scribble stroke when trying to fill the shape, so that there are many unintended intersections between the strokes.

A shortcoming of using the bounding ratio is that it is unlikely to work well for nonconvex shapes. The bounding boxes for fill and delete scribbles associated with identical star shapes could have exactly the same area.

Additionally, neither the bounding ratio nor the number of intersections can be calculated without the context of a boundary stroke, in this case a circle. In the context of a complex drawing with many strokes, it might be useful to identify a new scribble stroke as a fill or a delete gesture without looking at any other strokes. The comparison of density to speed is not nearly as clear cut in the smaller values of density and speed, but very dense strokes could be flagged as intended as a fill gesture, and very fast strokes could be flagged as deletion (Figure 2(c)).

Future Work

One concern we would like to address is how to handle scribbles in broader contexts. One application we are working on is a pen-drawn sheet music recognizer wherein the user can create quarter notes by filling in a half note or delete a note by scribbling it out. This will require identification of not just whether the user meant to fill in a shape or delete it, but also what they meant to fill or delete. It may be difficult to determine whether the user meant to delete a single note, the tail of the note, or the staff behind the note.

Contributions

Our contribution in the area of sketch recognition is to provide a method for recognizing whether a scribble stroke is intended to fill or delete a convex shape. This leads toward a more natural interface for sketching and editing sketches and is a step towards the more complex problem of identifying scribble intention in more complex contexts.

Acknowledgements

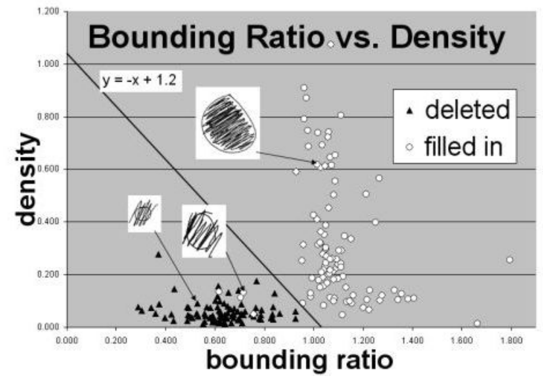
This work is funded in part by NSF IIS grant 0744150: Developing Perception-based Geometric Primitive-shape and Constraint Recognizers to Empower Instructors to Build Sketch Systems in the Classroom.

References

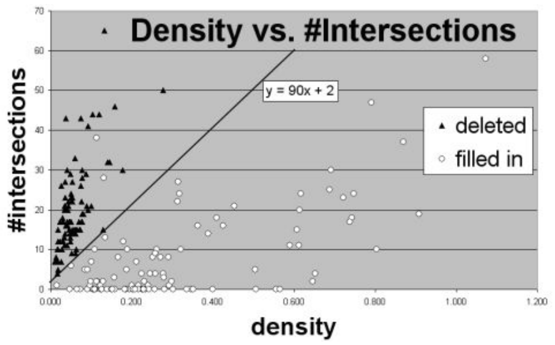
Forsberg, A.; Dieterich, M.; and Zeleznik R. 1998. The music notepad. In Proc. of the 11th annual ACM symposium on User interface software and technology, 203 - 210. San Francisco, California, United States.

Long, Jr., A.C., Landay, J.A. and Rowe, L.A. 2001. "Those Look Similar!" Issues in Automating Gesture Design Advice. *In Proc. of the 2001 Workshop on Perceptive User Interfaces*, 1-5. Orlando, Florida: ACM Press.

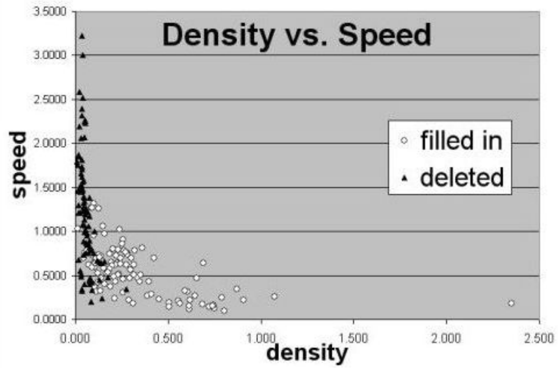
Rubine, D. 1991. Specifying Gestures by Example. *ACM SIGGRAPH Computer Graphics* 25(4): 329 - 337.



(a) Bounding ratio and density together.



(b) Density and #intersections together.



(c) Scribble density and speed together.

Figure 2: Graphs comparing combinations of feature values.

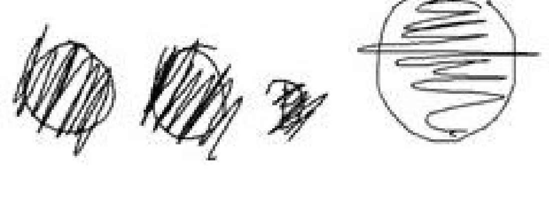


Figure 3: "Misclassified" examples - users saved these as "fill" gestures and they were classified as delete using bounding ratio and density.