Flow Selection: A Time-Based Selection and Operation Technique for Sketching Tools

Gabe Johnson
Carnegie Mellon University
johnsogg@cmu.edu

Mark D Gross
Carnegie Mellon University
mdgross@cmu.edu

Ellen Yi-Luen Do
Georgia Institute of Technology
ellendo@gatech.edu

ABSTRACT

Flow selection is a time-based modeless selection and operation technique for freehand drawing and sketch tools. We offer flow selection as a modeless technique to address the observation that modal selection requires too much cognitive effort and causes breakdowns in creative flow. Flow selection provides input to a new class of operations by assigning increasing, fractional selection strengths to objects over time. We discuss the current prototype system and possible applications for this novel technique for interacting with sketches.

Categories and Subject Descriptors

J.6 [**Compter-Aided Engineering**]: Computer-aided design (CAD); J.5 [**Arts and Humanities]:** Fine arts; H.5.2 [**User Interfaces**]: Input devices and strategies

General Terms

Design, Human Factors.

Keywords

Sketch, mode, time-based selection, pen, stylus, flow selection, modeless interaction

1. INTRODUCTION

In this paper we present *flow selection*, a time-based selection and operation technique for freehand drawing and sketch tools. Flow selection differs from traditional techniques in three ways. First, flow selection is triggered without requiring the user to deliberately enter a selection mode on a toolbar or via keyboard commands. Second, selections made using this technique are 'fuzzy' in that the points along the selected region have variable selection strength, and respond variably to subsequent operations. Third, flow selection capitalizes on the passage of time to gradually expand the outer boundary of the selection region and the selection strength of those objects.

When editing a two-dimensional freehand drawing using traditional sketching interfaces, users must choose between making a rough drawing using an application that simply captures pixels on a canvas (e.g., Microsoft Paint) or creating a precisely prescribed rendering using splines and geometric structures (e.g., Adobe Illustrator). Some programs clean up the user's roughly sketched strokes by smoothing them out or snapping objects into place. Flow selection allows a user to pick regions of strokes that they want to fix and operate on them, such as repositioning the end of a line or smoothing a rough section.

Existing methods for editing curves vary in complexity. The simplest and most common method of editing a curve is to erase it and try again, or to draw a new stroke to replace an existing one. Even the smallest mistake forces the user to completely start over. In vector-based drawing applications, curves (such as splines) are often defined indirectly by a series of control points, which the user moves to modify the curve. Mistakes in this case are not as difficult to correct, but editing curves by moving control points is tedious and sometimes awkward work and this technique is unobvious to novice users.

In contrast, flow selection empowers even novice users to quickly repair small mistakes in their drawn strokes and move on to their next stroke. This technique is similar to pressure-sensitive interaction [4] but does not require the use of specialized hardware. Flow selection involves a selection phase and an operation phase, allowing the user to effortlessly switch from one to the other. Figure 1a shows a cartoon drawing of a face. To select a region of a drawn stroke, the user simply holds the stylus relatively still near the region to be edited. Soon, color begins to

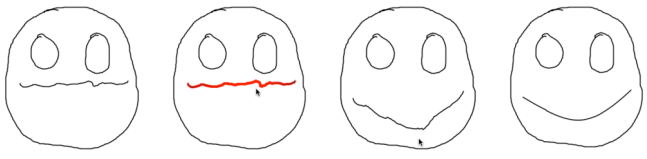

Figure 1: Using flow selection to move a drawn stroke. In (a), the user has just drawn the mouth and immediately selects it from near the middle (b), yanking downward to create a smile (c), and (d) removing the jagged parts with a smoothing operation.

AVI '06, May 23-26, 2006, Venezia, Italy. Copyright 2006 ACM 1-59593-353-0/06/0005...$5.00.

slowly flow outwards from the epicenter, the nearest point on the stroke (Figure 1b). After waiting until the desired region has become selected, the user moves the stylus to begin an operation. The most common operation is reshaping the selected region of a stroke. In this case, points closer to the epicenter move at nearly the same rate as the stylus; points in the selection far from the epicenter move less (Figure 1c). If the user wants to smooth a rough line, the user may once again hold the stylus still. This time, instead of selecting, the selected region begins to smooth out. Smoothing ends when the user lifts the pen. Alternately, a user who wants to reshape the selection again can do so by simply moving the stylus. After the pen is lifted, the user may continue drawing.

With flow selection users may alternate between drawing strokes to the canvas and selecting and operating on regions without explicitly changing modes. In this way, flow selection presents one solution to the 'mode problem' [6]. In contrast to the approaches to modeless interaction found in [5] and [1], flow selection is triggered at the start of a pen gesture, rather than afterwards.

2. SELECTION

After the user draws an ink stroke, the system normalizes the raw stroke's points to uniform spacing, moving and interpolating points to render equal the arc distances along the curve between each (x, y) point P[i] and P[i+1].

To support modeless interaction, pen strokes are interpreted as ink until a flow selection operation has been detected. Holding the pen down at nearly the same spot for a certain threshold unit of time triggers a flow selection. At that point, any ink that has been inadvertently drawn to the screen while initiating the selection becomes ephemeral ink [6], and begins to fade out over time so as to not distract the user.

Two functions are common to all select and operation process. An *effort function* determines the amount of time that it takes to select a particular object; a *strength function* determines how strongly that object is selected based on its effort value. While the user does not have direct control over the behavior of these functions in our prototype, other applications that implement flow selection may provide direct user control.

2.1 Effort Function

First, the flow selection algorithm discovers the stroke that contains the closest point to the pendown location. The point on the stroke that is nearest the pendown is referred to as the *epicenter* of the selection. Before visually echoing a selection, a function assigns each point on the epicenter's stroke an *effort* value E[i]. By definition, the epicenter has an effort of zero and each successive point further from the epicenter along the stroke has a non-decreasing effort value. The effort for each point indicates how much 'selection strength' is needed for an operation on the region to affect the point. Thus points nearest the epicenter are more easily affected by subsequent operations on the selected region.

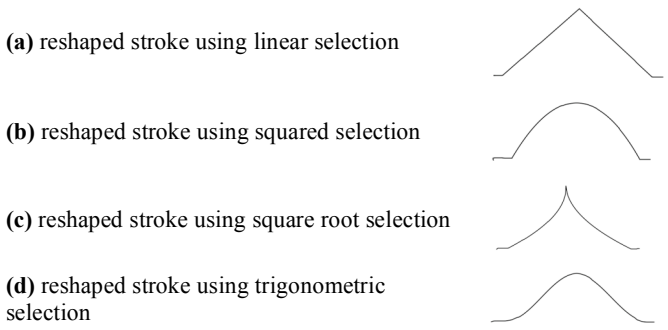

Figure 2: Examples of a stroke reshaped with four different strength functions. In all four examples, a straight horizontal stroke has been selected near the middle and pulled straight up. Note the discontinuities at the epicenter in (a) and (c), as well as discontinuities at the edge of the selected region in (a), (b), and (c).

The simplest effort function is the arc distance (distance along the curve) from the epicenter point:

$$E[i] = arc\_distance(\text{epicenter}, P[i])$$

As the curve is normalized, we may also use the equivalent:

$$E[i] = a \cdot |k|$$

where k is the difference in index values between i and the epicenter point index, and a is the normalized arc-distance between every pair of points.

Other less obvious but useful effort functions involve increased effort over curvy regions, or effort functions that look for corners and assign infinite effort to points beyond a corner, which can assist users in selecting straight regions more precisely. In our prototype, we can plug in and experiment with alternative selection effort functions.

2.2 Strength Function

After each point has been assigned an effort value, visual selection commences. Each point is assigned a selection *strength* S[i] between zero and one (inclusive) indicating no selection and full strength selection respectively. The selection strength can be used by subsequent operations on the selection region, which can affect points in the selection variably, depending on their selection strength.

For each point P[i] on the stroke, selection strength is a function of selection duration t (the length of time that the user has kept the stylus stationary) and that point's pre-computed selection effort E[i]:

$$S[i] = f(t, E[i])$$

An obvious strength function is one that selects the epicenter at strength=1 and linearly decreases selection strength as effort increases with distance along the stroke (figure 2a depicts this in use). For point i at selection duration t, this strength function can be written as:

$$S[i] = 1 - \frac{E[i]}{t \cdot C}$$

where C is a constant describing the effort increment for each time step. For example, after 3 time steps, if $C = 10$, then the selection strength of a point with effort 40 is

$$S = 1 - \frac{40}{3 \cdot 10} = \frac{-1}{3}$$

As the selection strength is negative, the point will not be included in the selection region until two successive time steps when the strength exceeds zero.

A strength function that works well is trigonometric, with the selection strength at the end of the selection region tapering off to zero (figure 2d).

$$S[i] = \frac{\cos\left(\frac{E[i]\cdot\pi}{t\cdot C}\right)+1}{2},\ E[i] \le t\cdot C\ ;\ 0 \text{ otherwise}$$

A nice feature of this selection strength function is that its first derivative is zero at both the epicenter and the boundary of the selection region. Reshaping operations do not create discontinuities or corners near the selection epicenter and boundaries.

As with the selection effort function, our prototype allows us to plug in and experiment with alternative selection strength functions. In our prototype, the selection flows over a stroke at an approximate rate of forty pixels per second. It is not easy to create precise selections using this method. However, with the use of strength functions that maintain continuity with the surrounding non-selected regions (such as in figure 2d), the imprecision does not create a problem.

3. Operating on the Selected Region

In second phase of flow selection the user operates on a region of the drawing that has been selected with variable strengths. In our prototype, the user cycles through two successive operations, move and smooth, that both modify the selected region. After completing each operation the user can continue to operate on the selection or return to inking. Our choice to use reshaping and smoothing is only one possible implementation of the theme, which is for the user to be able to switch from one operation to another by simply holding the stylus in place or dragging.

Figure 3 shows a finite state diagram of the selection-operation sequence. When creating a drawing, the user switches between "idle" and "draw ink" (Op1) states, as the stylus goes up and down on the tablet. When the user holds the stylus steady the system initializes flow selection (Op2), and automatically expands the selection extent until the user either raises the pen (returning the system to inking state) or drags the stylus. When the user drags the stylus, the system begins the primary modification (Op3, in our example, "reshape"), as long as the user continues to drag. When the user holds the stylus steady again, the system initiates the second modification operation (Op4, in our example, "smooth"). Whenever the user lifts the pen the system returns to an idle state.

Note the difference in the state diagram between the two modification operations. The first modification continues as long as the user drags and halts when the user holds. The second modification is opposite: as soon as the user drags, the operation halts.

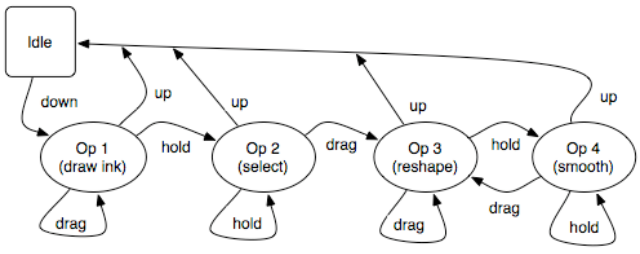

Figure 3. State diagram of drawing and flow selection and operation showing inking, selection, and two subsequent modification operations.

3.1 Reshape Function

When the user has selected a region and moves the mouse beyond a certain squelch radius (to compensate for shaky hands), the selection phase ends and the operating phase begins. Our prototype's default post-selection operation is a simple reshape command. For every (*dx, dy*) distance the user moves the stylus, each point moves this distance scaled by its selection strength. Points closer to the epicenter (whose selection strengths are greater) will move a greater distance than points toward the ends of the selection.

$$P[i] = \left(x[i] + S[i]\cdot dx, y[i] + S[i]\cdot dy\right)$$

After moving, the user may lift the pen, signaling that the process is done, and the pen returns to inking the canvas. Alternatively, the user may instead hold the pen steady, which after a while triggers a secondary operation. This secondary operation continues until the user lifts the stylus (the process is done) or moves it again to re-enter the primary operation.

3.2 Smoothing

Our prototype's default secondary operation is a smoothing function. While active, the smoothing function iteratively tweens the selected region to a natural cubic spline that is derived from the points along the selected region. Each point on the stroke is moved in the direction of a corresponding point along the spline. The amount each point is moved is a function of its selection strength. Initially, for a selected region of *n* points, the spline is constructed with *n-1* control points, using linear interpolation along the selection curve. When the selected region is close enough to the spline, a new spline is created with one fewer control points, and tweening continues. This smoothing process continues until the selected region becomes a straight line, or when the user exits the operation.

Figure 4 shows the user selecting and smoothing a jagged stroke. In this example, the final result is very nearly a straight line. However, the user may exit the smoothing process at any point before then when the shape of the curve is in the desired state.

4. Applications

We are currently early in our prototype phase of work and are still

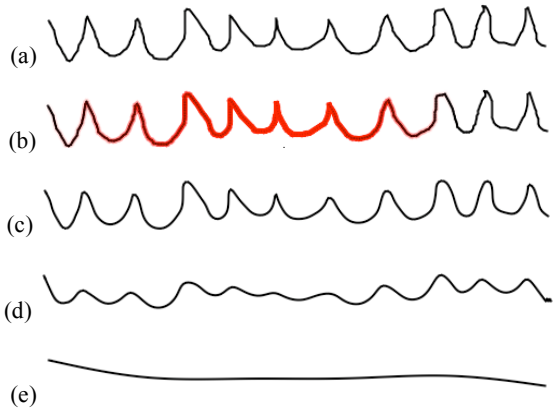

Figure 4: Using flow selection to smooth a shakily drawn cartoon sea. The original waves are shown in (a), partway through selecting the entire stroke in (b), with successive smoothed results shown in (c), (d) and (e).

exploring possible applications of this novel interaction technique.

We first developed flow selection as part of a sketch-based design application that enables novice users to create 3-D physical models of dinosaurs [2]. We wanted a simple means to edit the drawn shapes of the wood pieces that model the dinosaur bones. Although we had not set out to invent a novel interaction method, everyone who used or saw the dinosaur design application immediately became enthralled with the method of flow selection. We have not conducted a formal user evaluation of the technique, and we are currently considering appropriate metrics by which to evaluate it.

Flow selection can be useful to drawing and sketch applications used by artists and non-artists alike. People are likely to make frequent mistakes while performing tasks such as creating an artistic drawing or providing diagrammatic sketch input to a flowchart application. Rather than erasing their marks and trying again (which has an equal likelihood of resulting in a mistake), flow selection provides users a quick and intuitive method of making a correction. Flow selection's fluid, modeless interaction style can be appropriate in applications for artists, which can improve their creative output by helping to avoid the loss of concentration associated with frequently switching from drawing to selection mode.

5. Future Work

In the near future, we plan to create a more full-featured cartooning application that uses flow selection to help novices create simple colorful drawings. We plan to test this application in a controlled study in order to find out if our method empowers novice users' abilities to create artistic drawings.

Beyond the short term, we plan to develop an application that uses flow selection in 3D. Such an application will extend flow selection to work over curved surfaces and volumes.

Last, we imagine flow selection as a portable interaction technique. We would like to create a toolkit for flow selection that can be readily used in other applications.

6. ACKNOWLEDGMENTS

This work is supported in part by the Pennsylvania Infrastructure Technology Alliance (PITA), a partnership of Carnegie Mellon, Lehigh University, and the Commonwealth of Pennsylvania's Department of Community and Economic Development (DECD) and by the National Science Foundation under Grant ITR-0326054. The views and findings contained in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

7. REFERENCES

1. Hinckley, K., Baudisch, P., Ramos, G. and Guimbretiere, F. Design and analysis of delimiters for selection-action pen gesture phrases in Scriboli. in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Portland, Oregon, USA, April 02 - 07, 2005). CHI '05.*, ACM Press, New York, NY, 2005, 451-460.
2. Oh, Y., Johnson, G., Gross, M.D. and Do, E.Y.-L. The Designosaur and the Furniture Factory: simple software for fast fabrication. in Gero, J. ed. *Proceedings Second International Conference on Design Computing and Cognition*, 2006.
3. Qian, D., Gross, M., 1999. Collaborative Design with Netdraw. in *CAAD Futures 1999 Conference*. pp. 213–226.
4. Ramos, G., Boulos, M. and Balakrishnan, R. Pressure widgets. in:*CHI '04:Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM Press, New York, NY, 2004, 487-494.
5. Saund, E. and Lank, E. Stylus input and editing without prior selection of mode. in *UIST '03. Proceedings of the 16th Annual ACM Symposium on User interface Software and Technology (Vancouver, Canada, November 02 - 05, 2003).* ACM Press, New York, NY, 2003, 213-216.
6. Tesler, L. The Smalltalk Environment. *Byte, 6(8) August* (1981). 90-147.