Concurrent Editing of Vector Graphics using Similarity

Ankit Phogat
Adobe
phogat@adobe.com

Vineet Batra
Adobe
vbatra@adobe.com

Matthew Fisher
Adobe
matfishe@adobe.com

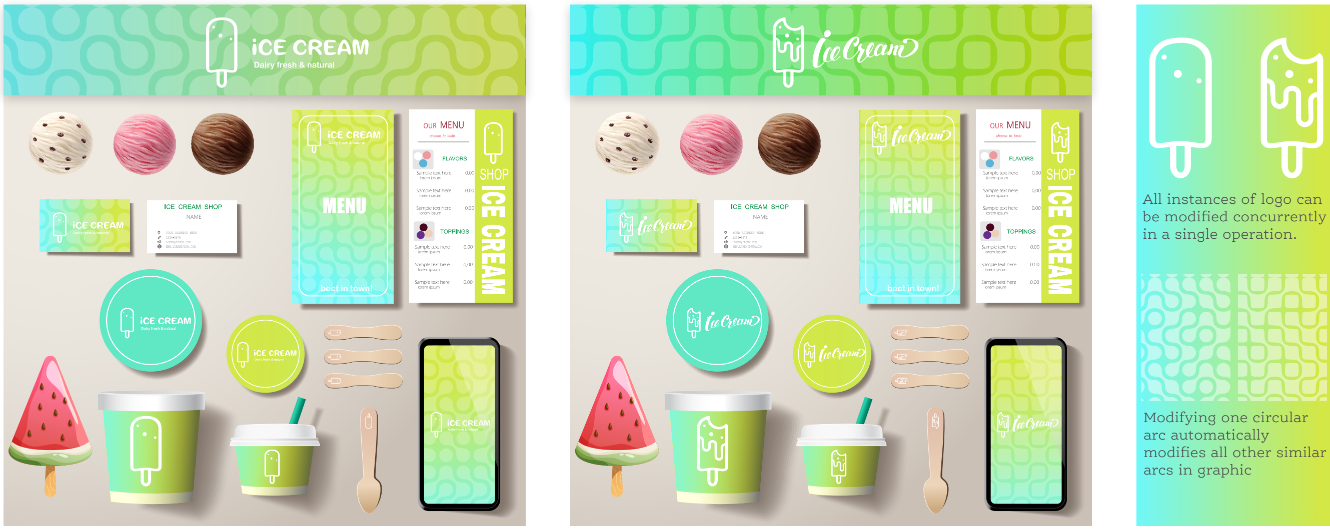

Figure 1: Proposed method allows easy and precise editing of all instances of an object (such as logos) in a global context, as well as all similar Bézier segments within an object in a local context.

ABSTRACT

We propose a novel paradigm for concurrent editing of vector geometry. Our method automatically establishes an editing context, which encompasses similar Bv́ezier geometry within a graphic object (local repetition), as well as multiple instances of entire objects (global repetition). We start with given *key* geometry, efficiently analyze entire vector graphic document to identify *similar* geometry, and then propagate modifications (both geometric and stylistic) from key geometry to its variants. To accomplish this, we orchestrate Procrustes algorithm in a one-to-many solve to determine all affine variants of key geometry. This solve also computes a per-variant transformation matrix which is used to propagate modifications. Our method is performant and efficient, identifying and concurrently modifying tens of thousands of objects in real time. In addition, it does not require any instrumentation from designers, and is thus applicable to all existing vector graphics documents (such as SVG and PDF files).

CCS CONCEPTS

• **Computing methodologies** → **Shape analysis**.

ACM ISBN 978-1-4503-7973-1/20/08.
https://doi.org/10.1145/3388770.3407400

KEYWORDS

Vector Graphics

ACM Reference Format:
Ankit Phogat, Vineet Batra, and Matthew Fisher. 2020. Concurrent Editing of Vector Graphics using Similarity. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Posters (SIGGRAPH '20 Posters), August 17, 2020.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3388770.3407400

1 INTRODUCTION

Vector graphics is a powerful medium for creative expression, as it allows for high degree of precision and arbitrary geometric complexity. The simplest (yet powerful) method of editing vector geometry is piece-wise editing of Bézier segments using control handles. However, when working with real world vector graphics, which typically comprises of hundreds or thousand of objects, this method becomes impractical. Recent advancements such as Linear Blend Skinning [Liu et al. 2014] attempt to abstract underlying complexity and allow semantic editing. While such methods are often easier, these lack precision and control. Moreover, such methods compel designers to change their workflows, and thus find low adoption. To overcome this, our method enables concurrent, *one-to-many* modification of thousands of paths without compromising on precision and control, or changing user workflows. Typically, vector graphic documents contain significant amount of repeating geometry. We leverage this fact to devise a one-to-many editing workflow, where modifications made to key object are propagated to its affine variations. These variations are automatically identified in real-time and do not require any intervention, so designer's editing workflow remains unchanged. Our method eliminates (error prone) repeated modifications, thereby improving accuracy.

Figure 2: A poster demonstrating reuse (repetition) of graphic objects in different form factors.

2 OUR APPROACH

Given a key geometry, which may be a complete object (global context), or partial selection of Bézier segments within an object (local context), the first step is to identify all similar geometry (affine variations) present in the document. As vector geometry is an ordered sequence of geometric primitives (line, quadratic or cubic Béizer segments), similar objects must have an identical sequence of such primitives. We start with a set of all objects, each represented by a set of control points $c_i \in \mathbb{R}^2$, $c = (c_1, ..., c_m)^T \in \mathbb{R}^{2m}$, the key object is denoted by p, while all candidate objects are denoted by $q_1...q_{m-1}$ respectively.

2.1 Pre-filtering

We use a hash function for each Bézier spline, comprising of number of segments N, type of primitives and continuity between adjacent segments (i.e. C^0, C^1 or G^1). Hash value is calculated as

$$H = (N)(Line \parallel Curve)(C^0 \parallel C^1 \parallel G^1)$$

We use this hash function to bucketize geometry and reduce candidate search space to a single bucket.

2.2 Computing Similarity and Transform

Displacement about object centers and average distance from center yields translation and scale component of the affine transform respectively. To compute rotation, we first normalize key object and all candidate objects about their center and scale them to a unit bounding box while maintaining aspect ratio. Then, we construct a pair of sparse, block-diagonal matrices P and Q corresponding to key and candidate objects respectively. All candidate objects $(q_1...q_{m-1})$ are arranged diagonally in Q, whereas P contains copies of key object corresponding to each candidate object. Then,

$$H = P^T Q \quad (1)$$

Next, SVD of H is computed as

$$H = U\Sigma V^T \quad (2)$$

Here, U and V are orthogonal (as U_A and U_B are orthogonal and Σ is a diagonal matrix).

We claim that if $A = U_A \Sigma_A V_A^T$ and $B = U_B \Sigma_B V_B^T$, then SVD of

$$\begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix} = \begin{pmatrix} U_A & 0 \\ 0 & U_B \end{pmatrix} \begin{pmatrix} \Sigma_A & 0 \\ 0 & \Sigma_B \end{pmatrix} \begin{pmatrix} V_A^T & 0 \\ 0 & V_B^T \end{pmatrix} = U\Sigma V^T \quad (3)$$

Using 3, it is clear that SVD of H is also a block diagonal matrix and each block is SVD of pairwise cross-covariance of key object and candidate object. Next, we compute another block-diagonal matrix D_i such that every block has dimension 2x2

$$D_i = \begin{pmatrix} 1 & 0 \\ 0 & det(V_i U i^T) > 0?1 : -1 \end{pmatrix} \quad (4)$$

$$R_i = U_i D_i V_i \quad (5)$$

R_i is the rotation matrix corresponding to ith candidate object.

We combine scale, rotate and translate into a single matrix, and compute residual error between each pair of key and candidate object. If residual error is below certain threshold, the objects are deemed similar. These transformation matrices are used to propagate edits to all similar objects in a one-to-many fashion.

2.3 Local Edit

We apply our method for synchronized editing of repeating geometry within a single object, where modifications made to a selected Béizer segments are propagated to all similar segments within the object, which also facilitates geometric variations (Figure 3).

2.4 Global Edit

Global edit facilitates simultaneous editing of multiple instances of similar objects (e.g. due to copy and paste). All geometric and stylistic modifications (fill, stroke etc) are applied to all similar objects in one-to-many fashion (Figure 4).

3 RESULTS

The following figures show the initial object and modified results using proposed method.

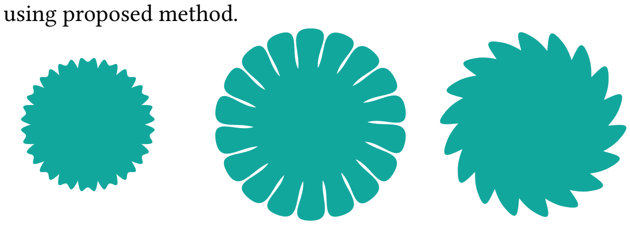

Figure 3: Variations of the flower generated by editing one petal, all other petals of flower are modified automatically.

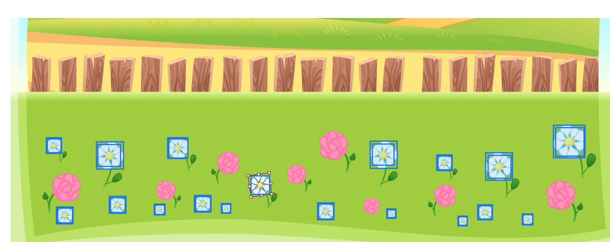

Figure 4: On editing one flower, all similar instances are modified automatically.

REFERENCES

Songrun Liu, Alec Jacobson, and Yotam Gingold. 2014. Skinning Cubic Bézier Splines and Catmull-Clark Subdivision Surfaces. *ACM Transactions on Graphics (TOG)* (2014).