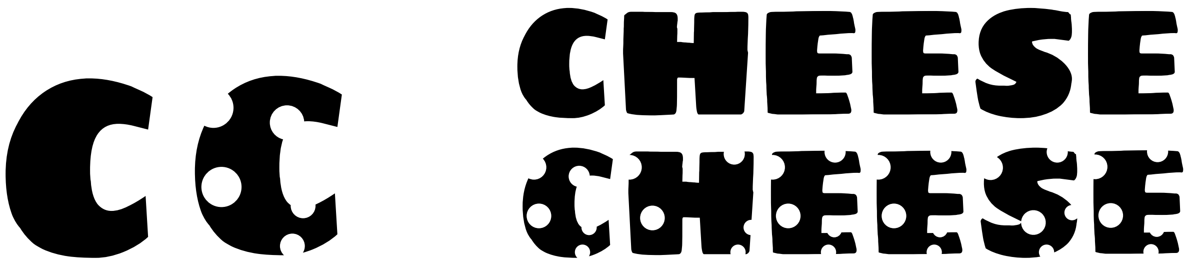
# Vector Based Glyph Style Transfer

Praveen Kumar Dhanuka*
Adobe
dhanuka@adobe.com

Nirmal Kumawat*
Adobe
kumawat@adobe.com

Nipun Jindal*
Adobe
njindal@adobe.com




(a) (b)

**Figure 1: (a) source glyph and modified source glyph ‘C’, (b) geometric style transfer with our approach to all glyphs of ’CHEESE’.**

## ABSTRACT

In this work, we solve the problem of real-time transfer of geometric style from a single glyph to the entire glyph set of a vector font. In our solution, a single glyph is defined as one or more closed Bézier paths which is further broken down in primitives to define a set of segments. The modification to these segments is percolated to the entire glyph set by comparing the set of segments across glyphs using techniques like the order and direction of segments and the spatial placement of segments. Once the target segments in other glyphs is identified the transformation from style glyph is applied to the target glyph.

Furthermore, we establish user-controlled policies for percolation of style like mapping line segment modification to curve segments. This extension to the algorithm enables the user to create multiple variations of a glyph.

## CCS CONCEPTS

• **Computing methodologies → Computer graphics**.

## KEYWORDS

Digital Typography, Font, Glyph, Vector Graphics

**ACM Reference Format:**
Praveen Kumar Dhanuka, Nirmal Kumawat, and Nipun Jindal. 2019. Vector Based Glyph Style Transfer. In *Proceedings of SIGGRAPH ’19 Posters.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3306214.3338600

*All authors contributed equally


*SIGGRAPH ’19 Posters, July 28 - August 01, 2019, Los Angeles, CA, USA*

ACM ISBN 978-1-4503-6314-3/19/07.
https://doi.org/10.1145/3306214.3338600

## 1 INTRODUCTION AND MOTIVATION

Conventional systems for glyph style transfer use neural networks such as a generative adversarial network to propagate a style from an image of one glyph to images of other glyphs [Azadi et al. 2017]. These conventional systems for glyph style transfer require multiple training examples of modified glyphs. In addition to requiring multiple training examples, conventional systems that use neural networks require training images having a raster format because of the spatial relationship of pixels in raster images. Thus, conventional systems cannot use vector graphics to transfer glyph style. Since fonts typically need to be generated at high resolution for most practical applications, these conventional techniques have limited value in many scenarios. Furthermore, generative adversarial network systems are a computationally expensive means of glyph style transfer. There are few other methods [Fišer et al. 2016], [Yang et al. 2016] which provides a raster based compute heavy solution to solve this problem. We show a highly performant vector based glyph style transfer which maps geometric style from a single observed glyph style to the complete glyph set. The proposed methodology works on end-to-end on vector glyphs as vector graphics and hence, it preserves the actual smoothness and curves of modified vector based glyphs.

## 2 TECHNICAL APPROACH

A computing device applies a style transfer system which receives an outline of a stylized vector-based glyph and an outline of an original vector-based glyph as input to the algorithm.

### 2.1 MODIFIED GEOMETRY RETRIEVAL

The style transfer system converts the glyph outline (Bézier paths) into individual segments such as Line and Curve segments and identifies modified and newly added segments of the stylized glyph by comparing against the original glyph segments on a segment-by-segment comparison basis.

## 2.2 STYLE PERCOLATION

Then the system identifies the segments in the target glyph on which the style transformation needs to be applied by various methodology. Each methodology identifies the segment in a unique way and helps in providing multiple variations of the same glyph. We use directions of stylized segments and identify similar direction segments in target glyph. The style transfer policy includes a transfer policy module which defines rules for transferring the styles across glyph set. Examples of style transfer policies may include segment-based mapping and region-based mapping.

*2.2.1 SEGMENT BASED MAPPING.* Segment based mapping algorithm compares, the segments to be replaced to the target glyph set's segments, the system can identify similar segments of the target glyph. Broadly speaking, the similar segment is identified based on having some feature (e.g. direction of the segment) in common with the segments to be replaced of the unmodified glyph. The system transfers the modification to the additional glyph by mapping the replacement segments to the similar segment. For example, the style of line segment will be applied to a similar direction line segment in target glyph. Similarly, style of curve segment will be applied to a similar direction curve segment in target glyph. In this way, the style transfer system applies the style feature of the modified glyph to the additional glyph based on the feature that is common between the similar segment and the segments to be replaced.

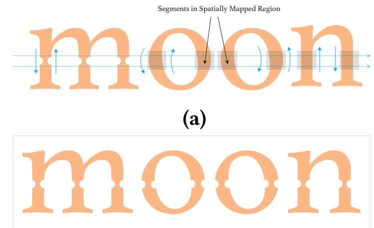
(a) (b) (c)

**Figure 2: Segment-based mapping: (a) source unmodified glyph, (b) source modified glyph, (c) resultant modified glyphs.**

*2.2.2 REGION BASED MAPPING.* This extends the capability of Segment Based Mapping where line segments can be mapped to curve segments or vice versa spatially. The policy can specify that the style should also be mapped to a specific region spatially to the target glyph set. Such controls and policy, gives the end user a free hand in creating variation performantly from the alogrithm.


(a) (b)

**Figure 3: Region-based mapping: (a) Unmodified input glyph. (b) Modified input glyph along with the region of modification.**




(a)

(b)

**Figure 4: Region-based mapping: (a) direction of segments and regions to apply the modification, (b) modification done on line segments are transformed to curve segments.**

## 3 MORE RESULTS

**Figure 5: Shows the original glyphs and generated stylized glyphs based on a user modified glyph (marked by orange box)**

## 4 LIMITATIONS AND FUTURE WORK

The proposed work could be improved in multiple ways. Firstly, it could be extended to handle the changes done across different closed paths of a glyph [Figure 6]. Secondly, an intuitive control needs to be explored to assist a user in defining the correspondence between segments of multiple glyphs and use these correspondence for transferring geometric styles.

**Figure 6: Transferring style applied on lower part of glyph 'B' to other glyphs may not produce the expected results with proposed methodology**

## REFERENCES


Samaneh Azadi, Matthew Fisher, Vladimir Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. 2017. Multi-Content GAN for Few-Shot Font Style Transfer. (12 2017).

Jakub Fišer, Ondřej Jamriška, Michal Lukáč, Eli Shechtman, Paul Asente, Jingwan Lu, and Daniel Sýkora. 2016. StyLit: Illumination-guided Example-based Stylization of 3D Renderings. *ACM Trans. Graph.* 35, 4, Article 92 (July 2016), 11 pages. https://doi.org/10.1145/2897824.2925948

Shuai Yang, Jiaying Liu, Zhouhui Lian, and Zongming Guo. 2016. Awesome Typography: Statistics-Based Text Effects Transfer. *CoRR* abs/1611.09026 (2016). arXiv:1611.09026 http://arxiv.org/abs/1611.09026