## An attempt at creating font transitions

S.K. Venkatesan

**Abstract**

In this short note we discuss the abstract topological aspects of fonts and how a simple homotopy deformation can be created to transition a glyph in one font to that of another font. We demonstrate interesting transitions for some fonts from sans-serif to serif using a JavaScript implementation of a simple algorithm. We also discuss other cases which involve deformations that map non-homeomorphic configurations that involve catastrophic bifurcations.

### 1 Introduction

Donald Knuth in his METAFONT project [1] wanted to initiate a framework for creating fonts through computer programs. An essay in Douglas Hofstadter's book *Metamagical Themas: Questing for the essence of mind and pattern* [2] challenges this ambitious initiative and discusses it in great detail. A simple question is, can a computer program automatically create hybrid fonts, for example, a new font that is 30% like Helvetica and 70% like Times? We answer this question using a simple algorithm and demonstrate its ramifications for some fonts.

### 2 Algebraic topology of font glyphs and their deformations

Topology is usually defined as the study of clay geometry, explained through an example of a teacup being topologically equivalent to a ring (the handle of the teacup). The crucial idea is that you can remold the clay without breaking it anywhere through any continuous deformation, unlike in Euclidean geometry where lengths are also preserved.

Font glyphs are, for the most part, connected compact two-dimensional manifolds with simply-connected one-dimensional boundary curves. Although most glyphs are simply-connected, there are also some glyphs such as lowercase "i" and "j" that have two disconnected pieces of manifold, due to the dot that is disjoint from the main shape. In any case, each connected piece is a connected compact two-dimensional manifold with a boundary.

Typically, in the English uppercase alphabet, C,E,F,G,H,I,J,K,L,M,N,S,T,U,V,W,X,Y,Z and in the lowercase, c,f,h,k,l,m,n,r,s,t,u,v,w,x,y,z are glyphs that are two-dimensional manifolds with a simply-connected single boundary which can be homeomorphically mapped to a two-dimensional disc with one-dimensional circle as its boundary (we could have just as well used a square instead of circle, as they are topologically equivalent). In this case, it is not difficult to prove that any two such glyphs can deformed from one into another by a simple homotopy deformation. In any case, we will be constructing an algorithm that does this, so we prove it by example.

In other glyphs there are holes in the manifolds; as in the case of O, Q, A, etc. In some cases, as in B, there are two holes. So the glyphs can have more than one boundary as well. But in all cases the boundary is a simply-connected one-dimensional curve. As a special trivial case of the Poincaré conjecture,[1] we have that any simply-connected one-dimensional curve is homeomorphically equivalent to a circle. In fact, there is only one one-dimensional manifold (without boundary), the circle.

### 3 Description of the algorithm used for font deformation

We shall here briefly describe the first algorithm used to map the glyph of one font into another.

#### 3.1 Algorithm 1

**Step 1** We center a font glyph into a standard font box of 1em height.

**Step 2** We identify the boundary curves of the font.

**Step 3** We divide the curve into equal segments along the path of the curve and create a fixed number of an array of points.

**Step 4** We reorder the points of the array so that the one that is nearest to the origin appears first.

**Step 5** We do steps 1–4 for both the font glyphs.

**Step 6** We now transition from one set of arrays to another using a linear path in time, to show the transition.

#### 3.2 Algorithm 2

**Steps 1–4** Follow as in Algorithm 1

**Step 5** We first segment the sans-serif font glyph.

**Step 6** We now segment the serif font glyph into finer (typically 10 times more) segments than the sans-serif font glyph.

**Step 7** We now look for an orthonormal projection (due to symmetry requirements we may have to use x-projections and y-projections as well) from sans-serif (as sans-serif is more regular-shaped) font glyph onto serif font to locate the one-to-one correspondence.

**Step 8** Once the points have been paired we now transition from one set of arrays to another using a linear path in time, to show the transition.

[1] It is no longer a conjecture, as Grigori Perelman has proved the conjecture for the hitherto unproven three-dimensional case [5].



doi.org/10.47397/tb/42-1/tb130venkatesan-transfont

### 3.3 Some hints on filling the area inside the curve

The curve that has the longest length needs to be filled inside, while the rest of the curves should not be filled inside (except for the case of the dot for "i" and "j"). In order to identify which piece of curve maps into which, we use the length and the centroid of the curves as the important parameters to guide in our choice.

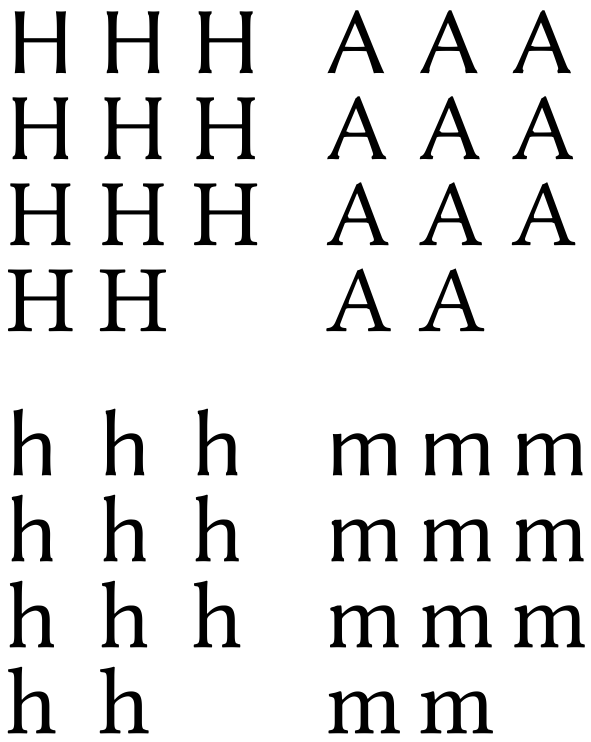
To identify that the dot in "i" and "j" needs to be filled, we need to find out if the other curves are inside the main curve or not, a non-trivial exercise. In fact, it took more than a century to prove the Jordan curve theorem [3], i.e., that a one-dimensional closed curve embedded in a two-dimensional Euclidean plane divides the plane into two disconnected pieces (the inside and outside), but for approximately convex shapes this identification is not that difficult.

## 4 The Demo site and JavaScript algorithm

The demo site is here:
`cqrl.in/dev/font-transition-js.html`

The JavaScript code, released under GPLv3, is here: `github.com/Sukii/font-transition`

## 5 Transition from Linux Biolinum Regular to Linux Libertine Regular



Now let us typeset some text with this transition font to see if it is reasonable:

100% sans-serif:

Quick Brown Fox Jumped High Over The Lazy Dog

70% sans-serif, 30% serif:

Quick Brown Fox Jumped High Over The Lazy Dog

30% sans-serif, 70% serif:

Quick Brown Fox Jumped High Over The Lazy Dog

100% serif:

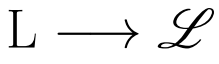
Quick Brown Fox Jumped High Over The Lazy Dog

## 6 The pessimism of Hofstadter and the confidence of Knuth

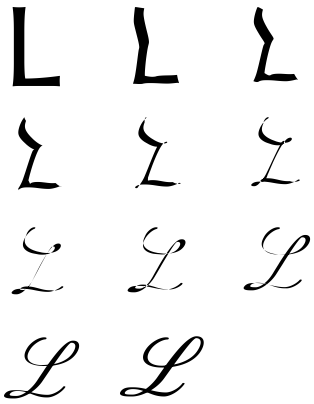
It is true that there are limits to what is feasible; for example, Einstein showed, using the principle of relativity, that the speed of light limits the speed of particles, as it would require infinite energy to cross that barrier. However, we also know that quantum tunnels break through such impossible barriers, as in the case of the Hawking tunneling effect, by which we are able to visualize black holes. More simply, X-rays are not visible to human eyes but they can be discerned through other means such as photographic plates. So limits of human perception and achievements can be extended by other means, as we gain better perception of the problems.

We have shown in this article, both theoretically and through demonstration of font transitions that it is possible in some effective way to achieve what Donald Knuth envisaged. In this article, we restricted ourselves to transition from one font to another only in the case where both are topologically equivalent (i.e., homeomorphic), but it is possible to go beyond that into other complicated designs such as script fonts and gothic (Fraktur, etc.) fonts, i.e., create a transition effect from a sans-serif font glyph to a script or gothic font glyph. This cannot be achieved by a simple homotopy deformation. It requires creation of bifurcation effects and other catastrophes studied by Rene Thom [4], V.I. Arnold [5], and many others.

Instead of meandering into profundity, we shall now discuss a concrete example. Consider the following calligraphic character L:



generated using the `calrsfs` LaTeX package. Let us see what the transition from the sans-serif "L" in Linux Biolinum to this script L might look like:



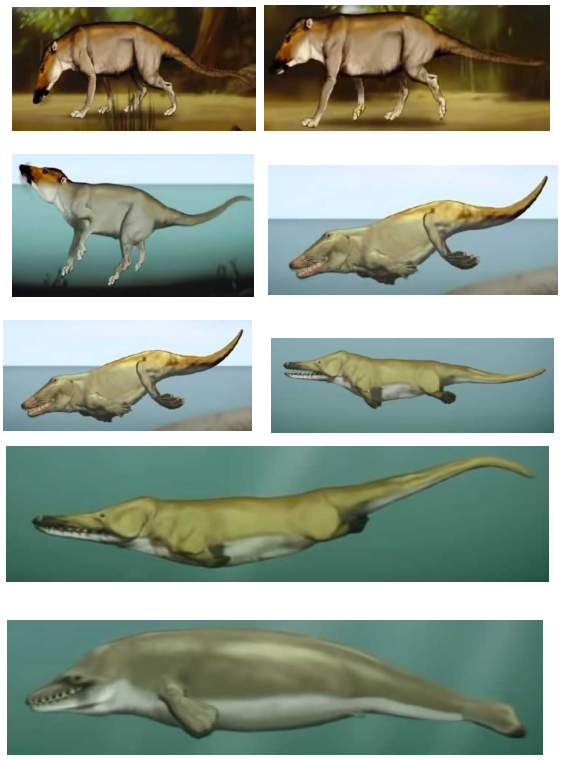
However, creating this transition was not as easy as the previous cases. The "script L" has three boundary curves while "sans-serif L" has only one boundary curve. But, if we carefully observe the pen stroke as it flows through the glyph, we can see it is just one stroke but self-intersecting at two places. This can be considered as a two-dimensional ribbon embedded in (2+1)-dimension (like a twisting and turning fly-over we see in big cities).

Unfortunately, the font files do not store this self-intersecting curve. Instead, the font files stores it as three boundary curves. In order to create the above transition we exported the glyph from the font file as three SVG curves, welded them together as a single curve that self-intersects at two junctions but flows unobstructed through the channels. Unfortunately, if we try to import this back into FontForge as a glyph and export it as a font, it doesn't work.

This describes the flow of the pen in (2+1)-dimension paper (the third dimension being time). To explain this I always mention this joke: There were once a big truck and a bus speeding along a highway at the same physical point on the road but didn't collide. How is it possible? Answer: The bus travelled in the morning, while the truck travelled at night.

## 7 Evolution of a whale from a mouse-deer

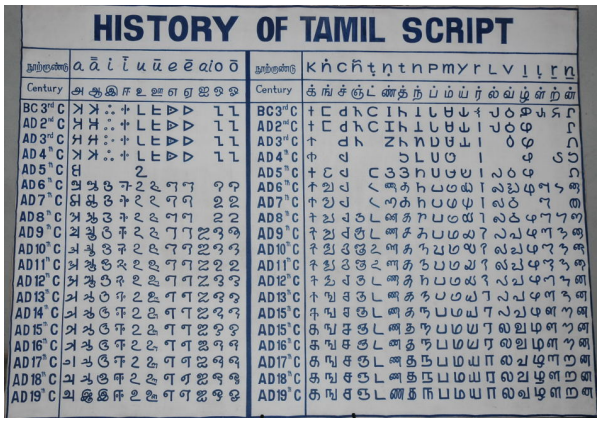
Here we explain the complexity of morphing of one glyph using the evolution of a land mammal (a mouse-deer) into a whale, giving a classical Darwinian angle to it:



In the above transition, one can observe that all animals retain approximate left-right symmetry in their external form. This left-right symmetry is imposed by the Darwinian pressure on the external form by the environment that punishes any left-right asymmetry as it decreases survival fitness due to impaired physical mobility. We shall see in the next section how such symmetry conditions can be imposed on glyphs to preserve approximate left-right symmetry in some characters in the transition states.

Insects possess some kind of left-right symmetry, but not all life forms have left-right symmetry, e.g., amoeba, octopus, jellyfish or for that matter trees and plants. In the zoo of Latin alphabets, only some characters possess approximate left-right symmetry, like H, A, M, W, but they each have their own unique shape and some internal symmetry and aesthetics of their own.

One can also employ these techniques to study the evolution of scripts over time, e.g., the evolution of Tamil script, as shown here (`wikipedia.org/wiki/Tamil_script`):




## 8 Machine learning approaches for fonts

An interesting broad question is whether one can create entirely new fonts out of old ones, using approaches like genetic algorithms. Here we have shown how one can create intermediate transition states from two sans-serif/serif pairs. In theory, it is also possible to create transition states using more than two fonts as well, i.e.,

$$\text{Font}_{\text{new}} = f(\text{Font}_1, \text{Font}_2, \text{Font}_3, \ldots)$$

One interesting problem that has been studied well is the aspect of character recognition (OCR), i.e., recognition of a glyph as an avatar of a particular character of a Latin alphabet. This in a way recognizes in essence what a character is, but it is more in relation to what other characters are and does not capture the essence of what a character is. The question is whether machine learning is capable of doing this. Hofstadter [2] explains how this is a notoriously difficult problem as there many avatars like the Gothic black letters that are so different in form.

As we showed here for the case of "Script L", it is possible to iron out some of the twists and turns of the flow of the pen by studying them as regular manifolds in (2+1)-dimension, that when projected onto a two-dimensional paper produces self-intersecting curves. We shall endeavor to bring out some of the qualitative features of the alphabets and how they can be defined using some abstract topological concepts in mathematics in (2+1)-dimension in future work.

Recent techniques in one-shot machine learning techniques [7] also hold genuine promise. This uses a function called "triplet loss" that trains on three images: an anchor image, a positive image, and a negative image. The neural network adjusts the parameters so that the features for the anchor and positive image are near while that of the negative image is far off. However, as we mentioned there are internal approximate symmetries (like the left-right symmetry) that need to be discovered and preserved, a genuinely difficult problem.

Of course, not all gyphs produced by an algorithm through machine learning will be usable as glyphs by humans. Some have ugly overlaps, as can be easily discerned, but the interesting question is: Is there always a perfect transition path from sans-serif to serif that avoids those pitfalls? Felix Klein wanted to define concepts of beauty through aspects of geometrical symmetry in an object and recently Roger Penrose also has been discussing these aspects in physics as well. The interesting aspect of human art (thinking of fonts and glyphs as an art form) is that it contains within it seeds of that fundamental question of what it is to be human. This may be a philosophical question that lies outside the realm of science, but our endeavor is to find those aspects that lie within the capacity of a scientific enquiry by trying to transcend human limitations through machines.

## References

[1] Knuth, Donald E. (1982) The Concept of a Meta-Font, *Visible Language*, Vol. 16, No. 1, pp. 3–27. visiblelanguage.herokuapp.com/issue/61

[2] Hofstadter, D. (1996) *Metamagical Themas: Questing for the essence of mind and pattern*. Chapter 13: Comments on Donald Knuth's Article 'The Concept of a Meta-Font'. Basic Books.

[3] Alexander, J.W. (1920) A proof of Jordan's Theorem about a simple closed curve. *Annals of Mathematics*, Vol. 21, No. 3, pp. 180–184. doi.org/10.2307/2007256

[4] Thom, R. (1972) *Structural Stability and Morphogenesis*. W.A. Benjamin.

[5] Arnold, V.I. (1984) *Catastrophe Theory*. Springer-Verlag.

[6] Tao, T. (2006) Perelman's proof of the Poincaré conjecture: A nonlinear PDE perspective. arxiv.org/abs/math/0610903 See also: claymath.org/millennium-problems-poincar%C3%A9-conjecture/perelmans-solution

[7] Dickson, B. (2020) What is one-shot learning? bdtechtalks.com/2020/08/12/what-is-one-shot-learning

◇ S.K. Venkatesan
TNQ Technologies, Chennai
skvenkat (at) tnqsoftware dot co dot in