MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

An example-based approach to style translation for line drawings

W. T. Freeman, J. B. Tenenbaum, E. Pasztor

TR99-11 December 1999

Abstract

We present an example-based system for translating line drawings into different styles. The system is given a training set of many different lines, each drawn by an artist in various styles, which is used to translate new lines made by a user into a particular desired style with a it K-nearest neighbor algorithm. This algorithm fits each input line as a linear combination of the several training lines in the same style which are most similar to it. The fit line can then be rendered in different styles because the training set contains versions of each training line in each style. Our example-based approach has a number of advantages over conventional parameteric approaches to translating style. It can handle styles which are difficult to describe parametrically, and its repertoire can be easily extended by the user at any time. Moreover, given appropriate representations, it can be generalized to modify the style of other kinds of graphics objects, such as the font of a letter or the movement style of an animated character.

This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of Mitsubishi Electric Research Laboratories, Inc.; an acknowledgment of the authors and individual contributions to the work; and all applicable portions of the copyright notice. Copying, reproduction, or republishing for any other purpose shall require a license with payment of fee to Mitsubishi Electric Research Laboratories, Inc. All rights reserved.

Copyright © Mitsubishi Electric Research Laboratories, Inc., 1999
201 Broadway, Cambridge, Massachusetts 02139

MERL – A MITSUBISHI ELECTRIC RESEARCH LABORATORY
http://www.merl.com

An example-based approach to style translation for line drawings

William T. Freeman
MERL, a Mitsubishi Electric Res. Lab.
201 Broadway
Cambridge, MA 02139
freeman@merl.com

Joshua B. Tenenbaum
MIT Dept. of Brain and Cognitive Sciences
Massachusetts Institute of Technology
Cambridge, MA 02139
jbt@psyche.mit.edu

Egon Pasztor
MERL, a Mitsubishi Electric Res. Lab.
201 Broadway
Cambridge, MA 02139
pasztor@merl.com

TR-99-11 February 1999

Abstract

We present an example-based system for translating line drawings into different styles. The system is given a training set of many different lines, each drawn by an artist in various styles, which is used to translate new lines made by a user into a particular desired style with a *K-nearest neighbor* algorithm. This algorithm fits each input line as a linear combination of the several training lines in the same style which are most similar to it. The fit line can then be rendered in different styles because the training set contains versions of each training line in each style.

By describing input lines as linear combinations of training set lines, this procedure is expressive enough to fit a broad range of input drawings. By restricting these linear combinations to contain only the most similar training set lines, this procedure is constrained enough to preserve the distinctive stylistic features of translated lines. We represent input lines by splines with nonuniformly spaced control points, which emphasizes these stylistic features.

Our example-based approach has a number of advantages over conventional parameteric approaches to translating style. It can handle styles which are difficult to describe parametrically, and its repertoire can be easily extended by the user at any time. Moreover, given appropriate representations, it can be generalized to modify the style of other kinds of graphics objects, such as the font of a letter or the movement style of an animated character.

This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of Mitsubishi Electric Information Technology Center America; an acknowledgment of the authors and individual contributions to the work; and all applicable portions of the copyright notice. Copying, reproduction, or republishing for any other purpose shall require a license with payment of fee to Mitsubishi Electric Information Technology Center America. All rights reserved.

Copyright © Mitsubishi Electric Information Technology Center America, 1999
201 Broadway, Cambridge, Massachusetts 02139

An example-based approach to style translation for line drawings

William T. Freeman
MERL, a Mitsubishi Electric Res. Lab.
201 Broadway
Cambridge, MA 02139
freeman@merl.com

Joshua B. Tenenbaum
MIT Dept. of Brain and Cognitive Sciences
Massachusetts Institute of Technology
Cambridge, MA 02139
jbt@psyche.mit.edu

Egon Pasztor
MERL, a Mitsubishi Electric Res. Lab.
201 Broadway
Cambridge, MA 02139
pasztor@merl.com

February 25, 1999

Abstract

We present an example-based system for translating line drawings into different styles. The system is given a training set of many different lines, each drawn by an artist in various styles, which is used to translate new lines made by a user into a particular desired style with a *K-nearest neighbor* algorithm. This algorithm fits each input line as a linear combination of the several training lines in the same style which are most similar to it. The fit line can then be rendered in different styles because the training set contains versions of each training line in each style.

By describing input lines as linear combinations of training set lines, this procedure is expressive enough to fit a broad range of input drawings. By restricting these linear combinations to contain only the most similar training set lines, this procedure is constrained enough to preserve the distinctive stylistic features of translated lines. We represent input lines by splines with nonuniformly spaced control points, which emphasizes these stylistic features.

Our example-based approach has a number of advantages over conventional parameteric approaches to translating style. It can handle styles which are difficult to describe parametrically, and its repertoire can be easily extended by the user at any time. Moreover, given appropriate representations, it can be generalized to modify the style of other kinds of graphics objects, such as the font of a letter or the movement style of an animated character.

1 Introduction

In many computer graphics applications, the user wants independent control over the style and content of the output. For example, the user of a drawing program may want to apply a new style to selected lines of the drawing, while keeping the "content", or the overall shape of those lines, constant. Present systems typically provide such style control through parametric manipulations. For example, t he thickness of the line can be adjusted, or the Fourier spectrum of the x-y coordinates of the line can be modified to yield a different line character ([1]).

But even advanced parametric manipulations represent a small subset of the possible stylistic variations that a designer might consider. Figure 1 shows a set of line strokes each made in a range of different line styles. This range of styles would be very difficult to describe parametrically. Yet we would like to build a

system which supports content-preserving translation across such different styles of lines.

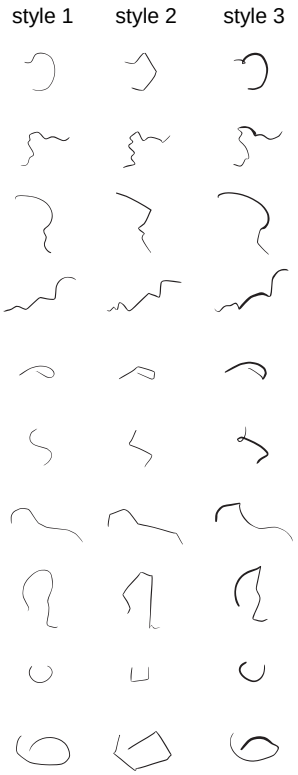


Figure 1: Training data. An artist drew the same lines in several different line styles, which we call generic, jaggy, and brushy. 123 lines were in the training set, in each style. By computer these were scaled in size over 6 scales, ranging from 0.3 to 2.0, over 4 angles, and flipped left/right. The outer product of all such manipulations yielded a training set of 5904 line elements.

We propose an example-based approach to this problem. The system designer collects examples of many different lines, each drawn in the various styles that the system is to contain (the "training data"). An artist (not a programmer) decides what it means to render a particular line over a range of styles. The system refers to this training data in order to translate lines made by a user into a particular desired style.

2 The translation algorithm

Here is the crucial problem we need to solve: the user draws some line in style 1, which we can assume is included in the training set. How should we use the training data to translate that line to style 2, the desired output style?

Consider first a very simple algorithm, the *nearest neighbor* algorithm. From all the lines in the training set for style 1, we look for the one closest to the input line. (We measure distance as a simple Euclidean distance between a vector representation of each line, described below in the section on representation). Call the closest training example the ith line. The training set includes line i in each of the styles, and so we simply output the training example of line i in style 2.

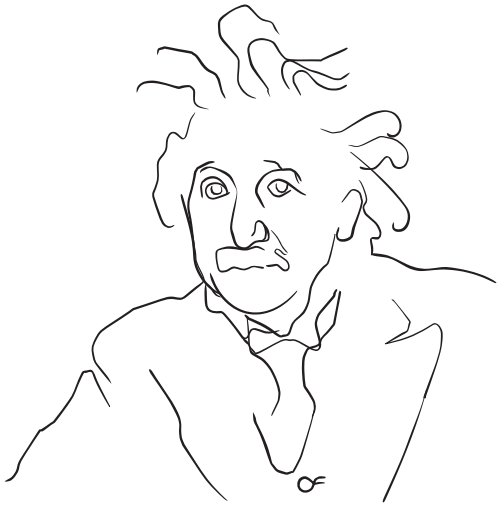


Figure 2: Line drawing traced from photograph of Einstein.

Figure 2 shows a test image we wish to translate to a different style. We had an artist draw a training set 123 lines, each in 5 different styles. Some elements of this training set, T, are shown in Fig. 1. Figure 3 (a) shows the nearest neighbor fit of the input drawing in a "generic" style of the training set. (In this and all subsequent examples, each line of the drawing is fit and translated separately.) Figure 3 (b) shows the fit drawing translated into a "jaggy" style using the nearest neighbor algorithm. This algorithm *translates* style well (i.e. the output lines (b) look like jaggy versions of the nearest neighbor fits (a)), because the nearest neighbor fit to each input line is represented explicitly in the artist-designed training set in both styles. But it doesn't *fit* the content of the input drawing well (compare (a) or (b) with Fig. 2), because the algorithm is only able to use lines that are already in the training set. Thus nearest neighbor is too limited to support content-preserving style translation.

To be of practical use, an algorithm must have the expressiveness to represent lines not explicitly put into the training data. Consider a second algorithm, the *linear combination* algorithm. In order to preserve the content of a broader range of a lines, this algorithm fits the input line with the best linear combination of all the training lines in style 1. Define the matrix A_s to have as its columns each of the training lines of style s. If the input line is the vector $\vec{y}$, then we find the coefficients $\vec{x}$ yielding the least squares solution to $\vec{y} = A_1\vec{x}$: $\vec{x} = \text{pinv}(A_1 * \vec{y}$. To render this input line in

(a)

(b)

Figure 3: (a) Nearest neighbor fit of original image to style 1. The fit does not describe the original image very well. (b) Nearest neighbor algorithm translation to style 2. The style translation, being a look-up into artist-drawn line strokes, is very good.

style two, we simply output $A_2\vec{x}$, a linear combination of lines in style 2 weighted by the same coefficients used to describe the input line in style 1.

Even with a small training set, the linear combination algorithm can fit an input line $\vec{y}$ quite well. Fig. 4 (a) shows the best linear fit to our test image for style 1 of the training set T. Rendering $A_2\vec{x}_i$ using the coefficients $\vec{x}_i$ found for each line of the test image results in Fig. 4 (b). While the fit in (a) is quite acceptable, style translation in (b) is poor. In general, linear combinations of a large set of the style-translated training lines will not "hold their style", because characteristic features such as inflections or loops are scrambled when averaged together across many lines.

These simple experiments demonstrate a tension between "goodness-of-fit" and "goodness-of-translation" that confronts any example-based approach to content-preserving style translation. The nearest neighbor algorithm focuses on one extreme, acheiving excellent style translation on a very limited set of lines which in general cannot fit the content of the input drawing. The linear combination algorithm goes to the other extreme, acheiving an excellent fit to the content of input lines but in a form that cannot be generically translated across style.

Our solution to this dilemma is based on the observation that linear combinations of *sufficiently similar* lines often do "hold their style". This is particularly true if the vector representation of lines pays attention to their salient features, as discussed below in the section on representation. Then, if we only take linear combinations of similar lines, we may be able to both fit the input image, as well as translate those fit lines well to the desired output style.

This intuition is the basis the algorithm we present here, the *K-nearest neighbor* algorithm for style translation. For an input line $\vec{y}$ in style 1, we find the K examples in the training set of style 1 that are most similar (in Euclidean distance) to that line. Let these K examples be the columns of a matrix, A_1^y. We then find the least squares solution $\vec{x}$ to $\vec{y} = A_1^y\vec{x}$, which describes the linear combination of the K training examples that best explains the input line. We translate this fit into style 2 by taking the same linear combination of the style-translated training examples. That is, we generate a matrix A_2^y by replacing each column of A_1^y with the style-translated version of that example found in the training set. Then $A_2^y\vec{x}$ describes the original input line $\vec{y}$ translated into style 2.

The K-nearest neighbor algorithm literally interpolates between the two extreme approaches presented above. When $K = 1$, it reduces to simple nearest neighbor. When K equals the size of the training set, it reduces to a simple linear combination. As we show below, intermediate values of K find an acceptable balance between goodness-of-fit and goodness-of-translation, yielding compelling style translations of an input drawing. In practice, it is easy to find a value of K that achieves this balance by starting at $K = 1$ and increasing K until a satisfactory fit is obtained. For the training set sizes we use here, we have typically set K between 3 and 8. In general, the optimal values of K will increase with the size of the training set.

The remaining sections present the vector representation of lines that we use, results of the K-nearest neighbor algorithm applied to the Einstein image, and conclusions and future work.

3 Line Representation

The K-nearest neighbor algorithm requires a vector representation of lines, and in particular, one that encourages a linear combination of similar lines in a certain style to maintain the stylistic quality of those component lines. We first fit an open uniform cubic b-spline to a set of finely-sampled points along each line. This allowed us to adjust the amount of data used to represent each line by fitting to fewer or more numerous control points. Because an open uniform cubic b-spline can be described as a sequence of connected cubic bezier curves, this representation can be easily translated into Postscript format, readable by many drawing tools. The K-nearest neighbor algorithm receives as input the x and y values of 20 spline control points for each line, along with a 3rd number at each control point describing the local line thickness.

The shape of a spline fit to data can be tuned by appropriately selecting the parametric value t assigned to each data point [2]. To best preserve the shape of each curve, we distributed t in a nonuniform fashion, such that greater intervals of t were clustered around areas of higher curvature. (Denoting arclength by s, the function relating dt/ds to local curvature had the form of a saturating (sigmoidal) nonlinearity.) Splines fit with t distributed in this manner tend to have control points clustered around bends and corners, and thus the salient features of a line are represented more accurately than would be possible with an even control point spacing.

Ideally, to maintain style, stylistic features such as bends or loops should be put in register from one line to the next, and then averaged. If they are mis-registered, their average will blur and distort each feature. Adaptively placed control points will tend to devote the same number of control positions to an uneventful straight segment, independent of its length. Thus the stylistic features will be more nearly in register with one another between two perceptually similar lines. Figure 5 illustrates this. The average of the two similiar lines, using evenly spaced control points, loses a stylistic feature, the sharp corner. Averaging with more control points devoted to the corner region maintains the stylistic characteristic of the line.

4 Results

Figure 6 shows the results of K-nearest neighbor style translation for $K = 6$. Panel (a) shows the fit of the Einstein image using a linear combination of the closest 6 lines in the style 1 training set to each line of the original image. Taking a linear combination of several nearby examples allows us to *fit* the input data well. Panels (b) and (c) show the Einstein drawing translated into "jaggy" and "brushy" styles, respectively. Combining only nearby examples allows the re-synthesized output lines to maintain their styles well.

Figure 7 shows a line-drawing by Picasso and its translation into "brushy" style, using the same training set and value of K.

Some insight into the limitations of the algorithm can be obtained by examining what appear to be "mistakes" in Figures 6 and 7. Because the algorithm does not know anything about the shapes of fingers, noses, or other real-world objects depicted in these drawings, it may inadvertently distort the contours of these objects in ways which are stylistically consistent but physically unrealistic. Because each line is modeled independently, relational properties between lines such as parallelism or perpendicularity may not be preserved. Such "mistakes" would be expected to occur with any low-level approach to style translation that treats a drawing as just a set of lines with no additional structure, and are not a distinctive problem with our example-based approach.

5 Conclusions and future work

We have described a system for example-based style translation in the domain of line drawings. The system is based on a simple and fast K-nearest neighbor algorithm, which is expressive enough to fit new input data, but constrained enough to translate that fit to different styles. Successful style translation depended on finding an appropriate representation for the input lines, which allowed linear combinations of similar lines to hold their style.

Our example-based approach has a number of advantages over conventional parameteric approaches:

First, it can handle any style that a user can draw examples of, regardless of how difficult it may be to describe parametrically. A standard parametric approach to style translation such as power spectrum modification ([1]) would have difficulty with the style translations presented in Figures 6 and 7, because crucial shape information encoded in the phase component of the frequency signal is ignored.

Second, the repertoire of an example-based system can be easily extended by the user at any time. While our system produces good results with manageable training set sizes (around 100 user-entered examples per style), these results can certainly be improved with more training data. To add a new style, a user only needs to draw examples of the basic set of training lines

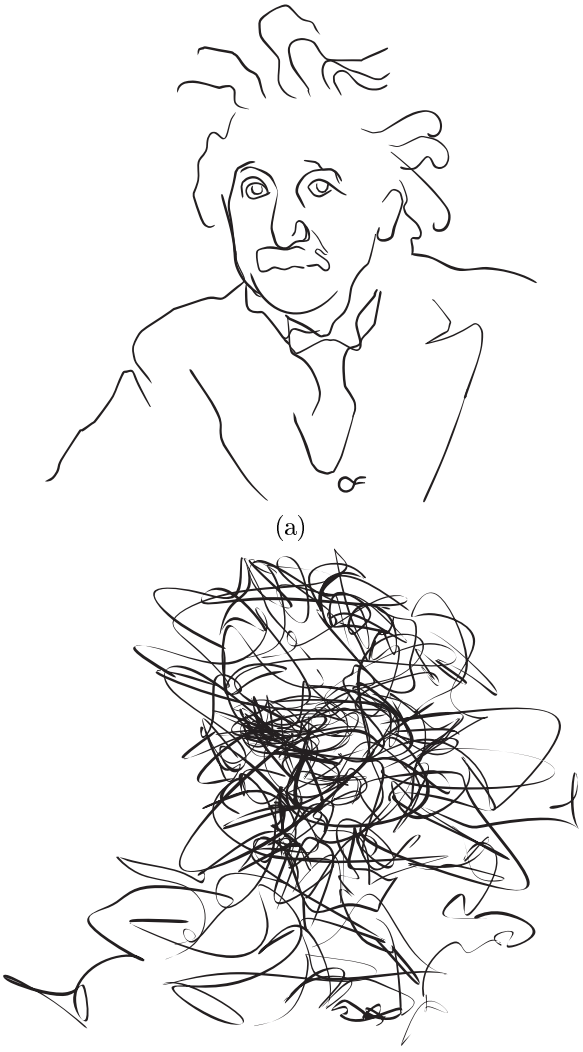
in that style. A user who works mainly with drawings of a certain class of objects (e.g. faces or mechanical parts) can supplement the default training set with contours from these objects, in order to encourage the system to more faithfully preserve their essential shape under style translations.

Finally, the example-based approach can be generalized to modify the style of other kinds of graphics objects, such as the font of a letter or the movement style of an animated character. The problem of having to compromise between quality of fit to a wide range of signals and quality of style translation is just as crucial in these domains, and the simplicity and generality of the K-nearest neighbor algorithm makes it a promising approach in these other domains. Of course, as in the domain of line drawing, successful example-based style translation in these other domains will also require a representation of the input signals which allows linear combinations of similar inputs to hold their style. This area is a focus of our ongoing work.

References

[1] A. Finkelstein and D. Salesin. Multiresolution curves. In *ACM SIGGRAPH*, pages 261–268, 1994. In *Computer Graphics* Proceedings, Annual Conference Series.

[2] Rogers90. *Mathematical elements for computer graphics.* McGraw-Hill Inc., 1990.



(a)

(b)

Figure 4: (a) Linear algorithm fit of original image to style 1. The optimum linear combination of all training lines yields a high fidelity fit. (b) Linear algorithm translation to style 2. Taking a linear combination of the 5904 content elements in style 2 yields an unintelligible result; the translation is very poor.

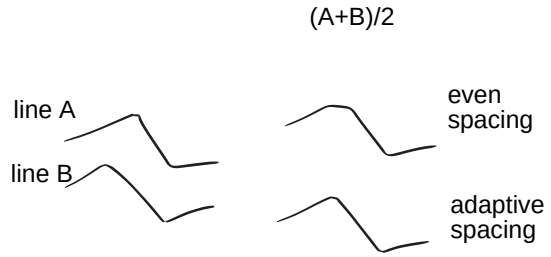


Figure 5: Left: two lines in the same style which are similar to each other. We would like to represent them in such a way that their vector average another line in the same perceived style. Top: Average using a spline with equally spaced control points. The features of the two lines are blurred. Bottom: The average using a spline with adaptively space control points. This tends to place the interesting feature points into similar dimensions.

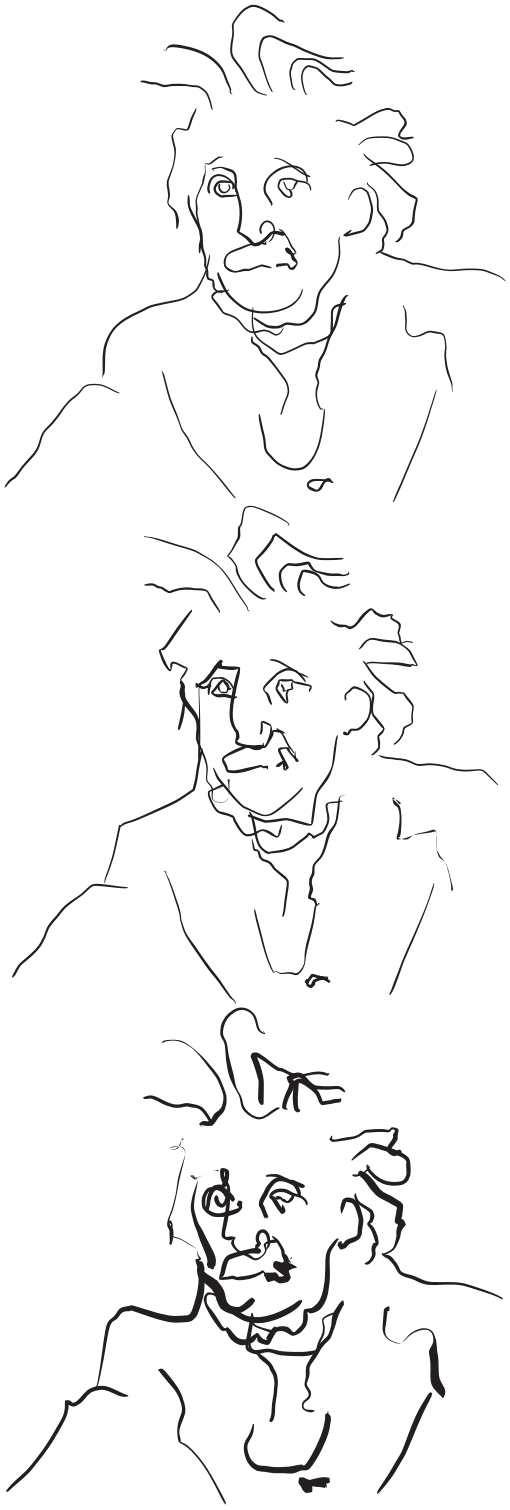


Figure 6: (a) 6-NN algorithm fit to image data in style 1. While not exactly the same as the original, the fit is quite faithful. (b) 6-NN algorithm image translation to style 2. In addition to fitting well, this algorithm translates well to the new style. (c) Image data translated to style 3, where the line quality of the training data in that style is still maintained.

(a)

(b)

Figure 7: Another 6-NN style translation example. (a) A tracing of the lines of Picasso's "Mother and Child". (b) Translation to style 3, using the 6 nearest neighbor algorithm, and fitting assuming the original was in style 1.