

Transmogrification: Casual Manipulation of Visualizations

John Brosz¹, Miguel A. Nacenta², Richard Pusch¹, Sheelagh Carpendale¹ Christophe Hurter³

¹University of Calgary, Canada
{jdlbrosz,sheelagh}@ucalgary.ca
rapusch@alumni.ucalgary.ca

²School of Computer Science
University of St Andrews, UK
mans@st-andrews.ac.uk

³University of Toulouse
ENAC, France
christophe.hurter@enac.fr

ABSTRACT

A transmogrifier is a novel interface that enables quick, on-the-fly graphic transformations. A region of a graphic can be specified by a shape and transformed into a destination shape with real-time, visual feedback. Both origin and destination shapes can be circles, quadrilaterals or arbitrary shapes defined through touch. Transmogrifiers are flexible, fast and simple to create and invite use in casual InfoVis scenarios, opening the door to alternative ways of exploring and displaying existing visualizations (e.g., rectifying routes or rivers in maps), and enabling free-form prototyping of new visualizations (e.g., lenses).

Author Keywords

Visualization; image deformation; multi-touch.

ACM Classification Keywords

H.5.2 Information interfaces and presentation: User Interfaces - Graphical user interfaces.

INTRODUCTION

We live in an information world and, thanks partly to the web, this information is increasingly visual. Visual representations of data appear often at work and in our social lives; for example, in newspapers and other websites we regularly encounter graphical evolutions of economic indicators for the last few years, or maps that display the most expensive neighborhoods in the city. These data visualizations are necessarily limited to specific representations, usually constrained by the format (e.g., paper, raster images on websites), and by the representation choices of the designer.

Naturally, no representation can be perfect for all purposes, for all people, or for all data. Tversky [53] has discussed how all maps (also a type of visual data representation) omit some information and may scale, simplify, reduce dimensionality, regularize, or even use inconsistent scale and perspective in order to better fulfill their purpose (which can be practical, aesthetic, or even political). Similar constraints apply to other types of static and dynamic visualizations. Even dynamic information visualizations cannot foresee all the needs

and preferences of viewers [28]. Although InfoVis programming languages (e.g., [5]) and end-user software systems (e.g., Tableau¹) are sufficiently expressive to create a great range of purpose-adapted representations, producing visualizations with these tools takes significant time and effort. Often this is not even an option since, despite the current push for open data, most often the data behind a representation is not directly available.

In this work we aim at allowing viewers to easily explore alternative representations. To do this we created the Transmogrification Canvas (TC), a tool that allows quick, real-time, flexible transformations of the representation of 2D visuals. Transmogrifiers—the core technique of the TC—are novel interface elements that adapt graphical deformation techniques to enable the transformation of an arbitrary original shape into another shape.

Using Transmogrifiers people can create custom transformations of existing graphics that are adapted to the person's specific visualization purpose, preference, and cognitive abilities. The specially crafted interface works in real-time on any existing visual material, and without the need for programming. The TC is designed to allow people to sketch, transform, and compare visualizations from one or multiple sources in a way that fits real-life situations, where people discussing a graph would like to make a point through a visual representation while still keeping pace with the conversation.

The contribution of this paper is three-fold:

- The concept of transmogrification: an application of graphical deformation to enable cartographic and visualization manipulations.
- A novel integrated set of interaction and animation techniques that enable quick and flexible transmogrification in real-time.
- A desktop multi-touch tool that integrates these techniques and animations into a coherent interface.

AN EXAMPLE SCENARIO

Maria likes to go on long bike rides. She carries equipment that measures her position and altitude through GPS, as well as her heart rate. After processing her data, her software creates the map and graphs in Fig. 1.A. To investigate the data further, Maria starts the Transmogrifier Canvas, which captures the map and graphs from the desktop. She creates a route trace on the map with her finger and then transforms it into a rectangle. This rectification represents her ride as

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
UIST '13, October 8–11, 2013, St. Andrews, United Kingdom.

Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2268-3/13/10...\$15.00.
<http://dx.doi.org/10.1145/2501988.2502046>

¹<http://www.tableausoftware.com/>

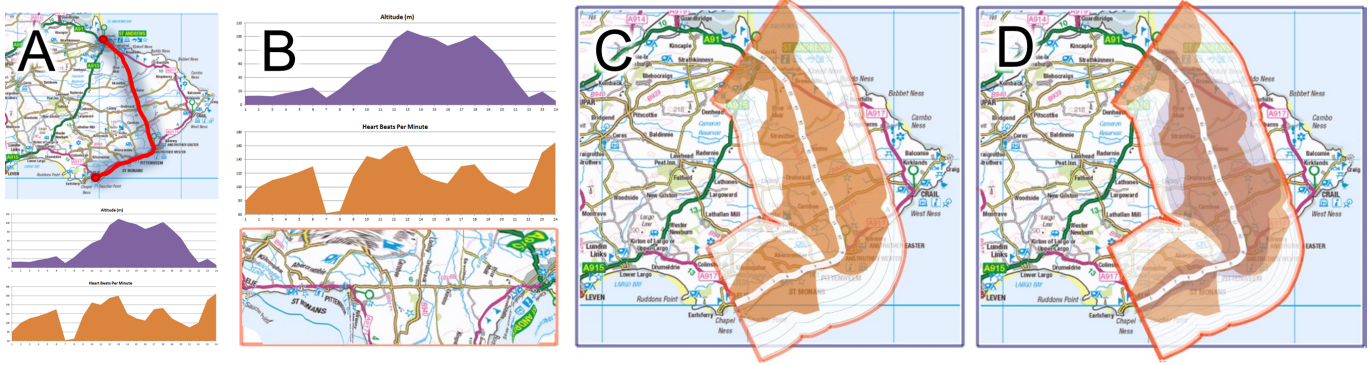


Figure 1. A) Cycling data: a map with a route and area graphs with average altitude (purple) and heart-rate (orange) at each kilometer. B) The rectified route map aligned under the linear graphs enables comparison of the measured variables to the map features. C) The heart-rate graph wrapped around the route in spatial context. D) Same as C, but with multiple variables. Map used contains Ordnance Survey data © Crown copyright and database right 2013.

a straight path from start to finish and she then resizes and moves it to lie under the graphs of altitude and heart-rate (see Fig. 1.B). This shows her how the different towns and intersections correspond to the different stages of her ride. She can quickly identify the name of the town where she stopped for lunch by looking at the correspondence between the rectified map and the graphed area of low heart rate.

To get a further overview of where she put in the most effort, she applies a transformation to plot her heart rate upon the map itself. Figure 1.C represents the effort (heart rate) at each point in the route, encoded in the width and amount of orange beside the route. Now she can easily identify that she worked hardest along the inland road. Finally, she applies the same transformation of the altitude graph to the map and overlays this with the transmogrified heart rate to see how one influenced the other (Fig. 1.D).

RELATED WORK

While motivated by Casual Infovis [40], our interaction techniques can be applied to non-casual scenarios and audiences. In this section we review related work in four subsections: the closest related tools, visualization and cartographic techniques, graphical deformation algorithms, and animation.

Related Tools

Most closely related to our work is Savva et al.'s ReVision [44], which uses machine learning techniques to analyze images of charts, extract chart type and infer the data, and then create alternative chart designs and styles. This approach supports one of Transmogrifiers goals (re-representation of an existing visual), but it is not real-time, it still requires significant knowledge for re-representation, and does not allow arbitrary transformations of shape. Also, while ReVision is 96% accurate classifying the infographic type, it is much less accurate in extracting the data; for instance, it was only $\sim 64\%$ successful at extracting data from the charts. In exchange, if successful in extracting data, ReVision would usually be more accurate, and its results avoid some of the problems that we discuss in the Limitations section.

SketchVis [7] and NapkinVis [10] both create visualizations through sketches on interactive surfaces. These share with

transmogrifiers the goal of quickly generating data visualizations without the need for programming. However, they are different in that they require access to the underlying data, do not operate on existing visualizations, and are focused on a few possible visualization types. Additionally, other tools exist that leverage multi-touch for drawing (e.g., [49]).

Related Visualization and Cartographic Techniques

The use of graphic deformation underlies many infovis techniques. A common type are local magnification techniques, which can be divided into two main categories (after Cockburn et al. [13]): overview+detail and focus+context. With overview+detail the magnified area is presented spatially separate from the context; focus+context techniques place the magnified region within the context by compressing some amount of the context area. Alternatives exist where transparency is used instead of compression [37], where context regions are hidden in folds [47], or where the lens automatically adapts its shape to match objects of interest [38].

Agrawala and Stolte [1] deform maps to reproduce characteristics of hand-drawn route maps, such as portraying the important turns and landmarks for a route without requiring an accurate representation of space. For this same goal of automatic creation of route maps Kopf et al. [27] make use of nonlinear optimization to non-uniformly adjust the position, scale, and orientation of an automatically determined subset of relevant roads. Bottger et al. [6] use image deformation to globally deform street-level map images to match the layout of an abstracted metro map/schematic.

Most of the techniques above can be simulated, reproduced, or approximated through the interactive use of transmogrifiers. Transmogrification provides a way of thinking about deformations as well a generalized interface to reproduce and prototype these deformations and many more.

Finally, there are related techniques based on spatial transformation and selection that are outside the scope of transmogrifiers. For example, magic lenses [4, 22] present information from a different or processed layer of spatial data in situ, within the spatial context, and Histomages [12] enable 2D selection operations in the transformed color histogram space.

Deformation Algorithms

Transmogrification is based on the ability to deform graphics. Computer graphics research has a long tradition of research in image deformation, which is often referred to as *tweening* (deformation has often been used to generate in-between frames for animations). Within tweening we review morphing and image deformation algorithms. Several of the following algorithms serve as base and are adapted to work with the constraints of transmogrifiers (see also *Implementation*).

Morphing. Morphing algorithms provide transitions where one object changes into another. Typically a set of correspondences are used to create a series of images that smoothly transition from an image to another. Although morphing was first achieved by simply cross-dissolving images [59], current approaches [48, 29] make use of image deformation style techniques, including one-to-one morphing. Reilly & Inkpen [42] and Haurert and Sering [19] provide examples of cartographic morphing—respectively deforming Cartesian maps into schematic maps and clarifying road knots.

Image Deformation. Image deformations encompass a variety of techniques designed to allow images to be manipulated as if they were flexible objects. These are applied to alter images, interpolate between keyframes, or even animate entire sequences. Usually deformations are enabled through placing and altering the positions of handles. There are many varieties of handles utilized for this task including points [24, 58, 25], lines [3], skeletons [8, 21], bones [46], lattices [60], sketches [15, 57], and cages [30, 56, 32]. Alternative interactions exist, for instance Schiwietz et al. [45] assign physical properties and create deformations by applying force vectors.

Many of these techniques, such as that of Igarashi et al. [24], break the image into small polygons and frame the problem as that of minimizing distortion. Other techniques, such as Lipman et al.'s [30], make use of different coordinate calculations with various shape preserving qualities to transform pixel coordinates. While minimization provides a physical basis for such manipulations, it also limits their scope. For transmogrification we can override these physically-plausible restrictions and instead seize the opportunity to alter imagery in a more abstract fashion.

Optimization-based deformation techniques work by moving pieces of the image to new locations and then attempting to make the rest of the image fit, providing a globally coherent reorganization of the image. These techniques are less useful for transmogrification, where the final shape is determined interactively rather than algorithmically.

In our exploration of algorithms we found that skeleton-type handles that allow for simple selection of visual elements are best suited for our purpose when combined with multi-touch interaction. Consequently transmogrifiers expand upon existing skeletal deformation techniques [8, 21], as discussed in the *Implementation* section below.

Animation

We provide animations to help viewers find correspondences between transformed and untransformed images. Tversky et al. [54] surveyed research in the cognitive value of animation

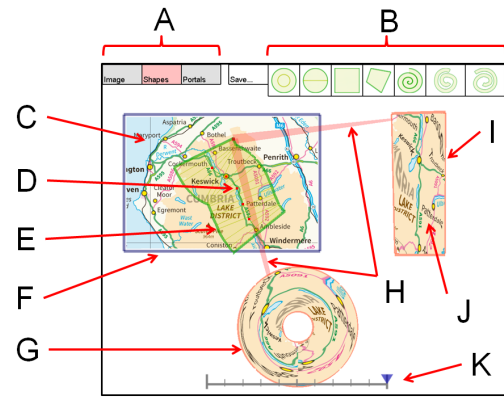


Figure 2. The basic interface of the Transmogrifier Canvas, containing two transmogrifiers. A) menu, B) shapes palette, C) imported image, D) control point, E) source shape (single-touch trace), F) imported image edge, G) destination shape (ring), H) transmogrifier lines, I) destination shape (rectangle), J) transmogrified image, K) animation control.

and found that it can be beneficial if it helps to portray extra information. This is the case for transmogrifiers, where animation is used to depict correspondences between 2D spaces that are not explicit otherwise. Other researchers have found benefits in animation for different domains such as tree visualization [2], common statistical graphs [20], and hierarchies [43]. These and other uses and studies of animation in information visualization (e.g., [16, 14, 23, 22]) inspire and inform the animations available in our tool.

TRANSMOGRIFICATION

Transmogrification is a technique designed for casual information exploration that allows people to easily take a personally selected part of the visual data and transform its geometry interactively. Transmogrification is instantiated by transmogrifiers, which are interface elements that implement adaptations of existing skeleton-based image deformation algorithms [8, 21], can be animated in different ways, and are controlled with multi-touch interaction techniques.

Transmogrifiers operate on any chosen area of an image—that is, they can start from any arbitrarily shaped region—and transform its contents to fit a chosen resulting shape. These shapes can be regular geometrical shapes or free-form shapes created interactively to fit a particular feature of the original representation. For example, a square area can be transformed into a circular area, and a curved free-form area can be transformed into a rectangular area. We call the shape that determines the area to be transformed the *source*, and the intended shape of the transformation the *destination*. Fig. 2 shows two transmogrifiers sharing the same origin.

Interface

Interaction with transmogrifiers is designed to support two basic goals: enabling a flexible set of transformations that adapt to the underlying data, and allowing people to generate a large number of different explorations in a short time. To enable a large set of manipulations, source and destination can be chosen from a set of predefined shapes (rectangle, circle, quadrilateral, spiral, ring) available through a tool palette (see Fig. 2.B), created on the fly through single- and

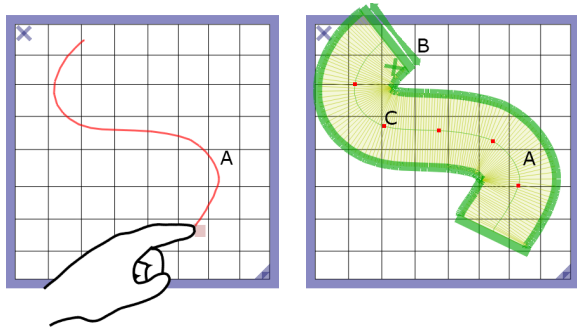


Figure 3. A single-point trace shape is created by dragging on free space. A) Spine, B) thickness control, C) sampling handles (red points).

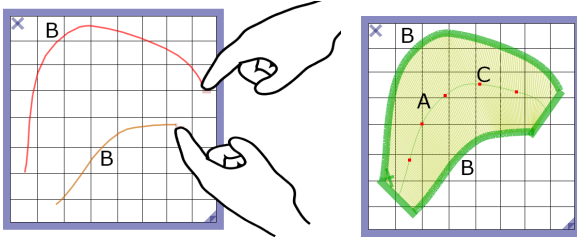


Figure 4. A double-point trace shape is created by dragging two fingers on free space. A) Spine, B) borders, C) sampling handles (red points).

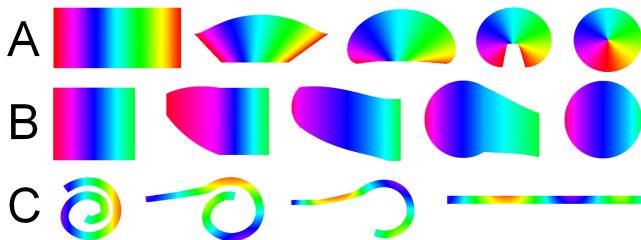


Figure 5. Implemented animations shown with coloured texture with the source shape (left), the destination (right), and tweened shapes (middle). A) Ring quadrature, B) path distance delay, and C) ribbon.

multi-point finger traces, or defined through bezier curves. The multiple combinations of source and destination shapes support different usages.

A transmogripher is created by dragging a finger from an existing shape into another. The shape under the start of the drag gesture becomes the source and the shape under the finger at touch release becomes the destination. A red transparent tapered line starts thick on the source and narrows to the destination, indicating the presence of a transmogripher (Fig. 2.H).

Shapes can be translated, scaled and rotated with multi-touch interaction techniques (e.g.,[35]). Most shapes have control points that enable further modifications, such as the quadrilateral, where corners can be adjusted, or the ring, where the size of the interior circumference can be enlarged.

One of the main features of transmogriphers that enables data-centric flexible analysis are *trace-shapes*. A trace-shape is created by touch-tracing a line on any region of the image space. This allows generation of arbitrary source and destination shapes that can follow the shape of the original graphical data on the screen. For example, with this multi-touch technique it is easy to create a source shape that follows the contour of elements in the image such as coastlines in a map. There are two types of trace-shape: single- and double-point.

In single-point trace-shapes, the analyst generates a line around which a shape is grown with constant width (Fig. 3). We call this central line the *spine*, which is related to Hsu et al.’s reference backbone [21]. The width of the shape is constant, but it can be modified through a control that appears at the start side of the shape (Fig. 3.B). Trace-shapes also provide handles along the trace to modify the local sampling densities, which allow compression or expansion of parts of the image longitudinally along the spine.

The double-point trace-shape is more flexible than the single-point because it enables dynamic variation of thickness along the trace, which allows a larger range of possible shapes. In exchange, it is harder to control because the additional degrees of freedom may make difficult to attain accurate traces [35]. However, while tracing with double-points one’s fingers do not occlude the middle content being traced (Fig. 4).

For traces that require more precision, we implemented a bezier curve drawing technique that works similarly to the single-point trace-shape. Bezier-generated shapes can also be translated, scaled or rotated with the usual gestures. Alternatively, the TC provides a “magnet” mode where traces are algorithmically aligned with nearby edges using our variant of Snakes [26], with the input points being aligned with the first-order gradient of a blurred internal image copy.

Animations

Transmogriphers are usually non-linear transformations which can be potentially hard to understand. To facilitate seeing the relationship between the original and transmogriphed space we incorporate animation, which has previously been shown to be successful in supporting comprehension during transformation in infovis (e.g., [43, 14, 20, 2]).

When a transmogripher is created, a default animated transformation from origin to destination is shown. The viewer can replay this animation at her desired speed by dragging a slider that appears at the bottom of the interface (Fig. 2.K).

A virtually infinite variety of combinations of origin and destination shapes implies that animations that are good for one transformation might not be good for another. To address this we implemented three different animation types, selectable through a menu in the TC interface. We chose these according to two criteria derived from previous work and our own experimentation: we try to maximize the number of pixels moving in the same direction (consistency), and we try to avoid twists and folds during the animation (clarity).

The animations are: *ring quadrature*, *path distance delay*, and *ribbon* (Fig. 5). Ring quadrature provides transitions from shapes in Cartesian coordinates to shapes in polar coordinates; path distance delay interpolates between shapes progressively from one end of the shape to the other; ribbon unwinds long, curving shapes. Future work should address the feasibility of automatic animation selection.

The Transmogriphication Canvas

Transmogriphers exist within the Transmogriphication Canvas. Although the general interface is designed to take advantage

of the multi-touch paradigm, during the design of the application we decided that—for some functions—menus, buttons, and sliders are preferable to invocation gestures that are sometimes difficult to discover and remember [36]. The following subsections describe the functionality of TC that goes beyond the individual transmogrippers, and how it interacts with the wider computing context.

General Interface

TC has a basic menu/palette that appears on the top left corner of the window (2.A). Large elements enable touch interaction, although indirect input is also possible for systems with mouse input. The shapes sub-menu in TC offers the standard shapes; however it can also save custom-made transmogrippers, which become part of the palette (Fig. 2.B). Transmogrippers can be turned into source raster images. The menu also contains options to select the animation type.

Multi-transmogripper Functionality

Beyond pure transmogripping functionality, the TC provides a workspace to combine and compose different transmogrippers for more power. For example, transmogrippers can not only share the same canvas space, but TC allows transmogrippers to share the same origin shape, which is useful to compare alternative transformations of the same data.

Because a transmogripper can be turned into a source image with the press of a button, we can chain multiple transmogripping operations, creating a 'transmogripping pipeline' reminiscent of Lark [51]. This operation, combined with the

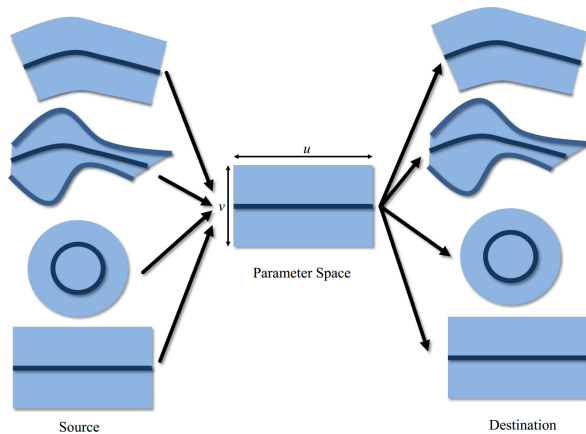


Figure 6. Transmogripping parameterizes a variety of source shapes (left) into parameter space (middle) and then re-maps it to any destination shape (right). Spine curves are shown as dark blue. From top to bottom the shapes are single-trace, double-trace, circle, and rectangle.

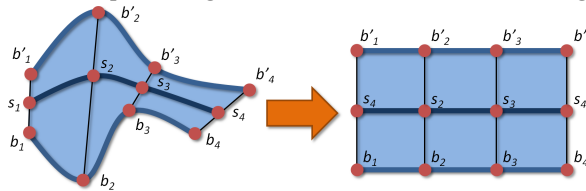


Figure 7. To render a transmogripper, we calculate the image's texture coordinates along the source's spine, s , and boundary curves, b' & b (left). These coordinates are used to render the transmogripper shape with texture coordinates placed along the equivalent destination curve.

ability of a transmogripper to operate on adjacent images enables a powerful *image transformation algebra* that we will return to in the Examples and Discussion sections.

The TC and its Computing Context

TC is designed to integrate in people's workflow through multiple features. First, images to transmogrippy can be imported into the canvas through a *load* option that accesses the user's operating system image files. Additionally, transmogripped images can be saved back to the hard-drive or copied into the system clipboard for use in other applications.

The TC can also be a sophisticated visualization and exploration tool when accessing web or other program's visualizations. By assigning a system-wide keystroke (e.g., Shift+PrtScr) TC can take a snapshot of the current state of the desktop and import it into a screen-sized TC window.

Lastly, TC has a multi-display mode in which the canvas can operate, in real-time, on objects that are present in other screens. This is a powerful feature, since it enables real-time interaction with transmogripped visualizations or even videos, which are updated in real time. This is possible without the need for interfacing our software with each specific visualization. Additionally, this feature allows viewers to configure personalized transmogripped information dashboards of information from other screens, a powerful generalization of the WinCuts concept [50]. To better understand the real-time desktop features, please refer to the Video Figure.

IMPLEMENTATION

This section describes our implementation to enable other researchers to reproduce our work; additionally, we report on how we adapted existing techniques, highlighting our contributions and providing a rationale for our choices.

Spines and Parameterization

The key element of the implementation of transmogrippers is the deformation algorithm that maps the source shape into the destination shape. For this we extend the skeleton-based approach of Hsu et al. [21]. Our algorithm requires that we define each shape through three curves: two boundary curves at the edges of the shape, and a spine somewhere between them (see Fig. 6). Based on those curves, the algorithm creates a parameterization of the source space that it uses to render it onto the destination. The destination space is a shape defined by a different set of spline and boundary lines (see Fig. 7).

With an equivalent parameterization in the source and the destination, we can then map the source area between the source's spine and boundaries to the shape defined by the destination's spine and boundaries. The parameterization of the shape's area is based on a linear interpolation between the spine, $f_s(u)$, and the boundary curves, $f_b(u)$ and $f_{b'}(u)$, where u is the parameter that expresses the distance of a point along the curves, and v the parameter that describes the relative position of a point between the curves:

$$f(u, v) = \begin{cases} (1 - 2v)f_b(u) + 2vf_s(u) & v \leq \frac{1}{2}; \\ (2 - 2v)f_s(u) + (2v - 1)f_{b'}(u) & \text{else} \end{cases}$$

where $v \in [0, 1]$.

Parameterization of standard shapes is easy to achieve (e.g., we parameterize the circle using polar coordinates); for trace-shapes we use piecewise linear parameterization of the touch traces. Once this is established, the rest of the process uses the texture coordinates to reshape the source area to fit the destination area. This is done by calculating the image’s texture coordinates along the source’s spine and boundary curves. These coordinates are used on the destination shape curves’ positions, rendering textured quad strips between the spine and each boundary (Figure 7).

We adopted this algorithm for four reasons: a) it is sufficiently flexible to provide a wide variety of transformations; b) it preserves the continuity of the source and destination shapes; c) the visual presence of spines and borders facilitates the understanding of transformations (as opposed to, e.g., formulae); and d) it lends itself to trace-based interaction and, specifically, multitouch input (see next subsection).

Our implementation described thus far differs from Hsu et al.’s Skeletal Strokes [21] because the latter are constrained to rectangles as source shapes and only introduce arbitrary shapes as destination shape. We can use arbitrary shapes for both source and destination; in the following sections we show multiple examples that need this flexibility. Additionally, Hsu et al. do not offer an interface for the flexible manipulation of their parameters. In contrast, transmogrification is an interactive technique at heart. The following sections describe adaptations of our algorithm that derive directly from the interactive nature of transmogrifiers.

Defining the Spine and Boundary Curves

As seen above, the main input for the algorithm is the curves that define the source and destination shapes. These are defined in different ways depending on the type of shape. For example, in a quadrilateral we choose two opposite sides as boundaries, and the median line as spine; in a circle the boundary lines are the circumference and the circle’s center, and the spine is a circle at half the radius. For trace-shapes the definition of the curves is built into the interaction and requires some extra considerations.

Single-point trace shapes. In these shapes the touch-trace generated by the analyst determines the spine, but the raw input has to be processed before it can be used due to the noisy touch input and irregular sampling provided by the input device. We interpolate the input linearly to subpixel resolution and then filter and decimate this signal [11]. The sampling of the spine curve is regularized according to arc-length along the spine. The viewer can alter the sampling by manipulating the handles described in the Interaction section.

To generate the shape, we expand from the spine outwards, perpendicular to the spine’s tangent until boundary curves at the desired width are created. As noted by Hsu et al., this can result in wrinkling or folding on the inside of boundary curves (Figure 8). Figure 9 shows the solution proposed by Hsu et al. [21], which only works for small loops, and our solution, which avoids the area becoming unnaturally stretched by distributing points along the path to the loop’s centroid (for larger loops). While this was not a concern in their scenario,

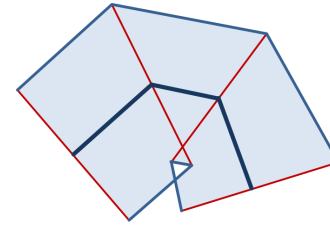


Figure 8. If only the tangent is used, growing the spine into the trace-shape produces overlaps (white triangle at bottom). Dark blue: trace (spine); red: path tangent; light blue: constructed bounding curves.

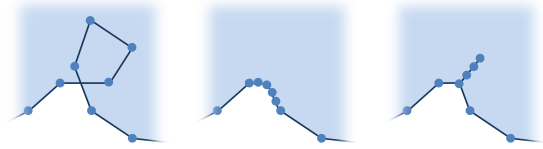


Figure 9. Repairing a self-intersection in a boundary curve (left). Center: moving sample points formerly within the loop onto the intersecting line segment. Right: To reduce stretching, we distribute loop points between the intersection and the loop’s centroid instead.

we need to consider it because we use trace shapes as source shapes. For larger loops with more than three points in the loop we distribute points along a path to the loop’s centroid.

Double-Point Trace. In a double-point trace-shape the touch-traces determine the boundary curves, and the spine is interpolated midway between them. Both boundaries are captured simultaneously, but care must be taken to match the sampling of both curves.

The piecewise curves resulting from each trace are linearly re-sampled based on input time so that each curve has the same number of points. Once both curves have an equal number of sample points the same interpolation and filtering process described for the single-point trace is applied. The parameterization for all three curves is based on the spine distance (as opposed to one of the boundaries).

Note that it is possible to use touch timestamps instead of distance to generate alternative sampling patterns. This opens a number of interesting possibilities that have not been considered in previous work. Moreover, our algorithmic modifications also support non one-to-one mappings, where source areas can appear more than once in the destination shape. In other words, transmogrifier shapes can fold upon or loop over themselves, making the same source pixels appear multiple times in the transmogrified image.

EXAMPLES

This section illustrates usage of transmogrification to achieve alternative visualizations in a range of scenarios.

Path Rectification and Deformation

Maps are typically represented as projections onto Cartesian planes. Distances and areas as measured in these representations usually revert to approximately Euclidean distances, which can be perceived reasonably accurately through direct sight. However, there are instances where we need to see distances beyond “straight line” distance. A straightforward example of this is J. H. Colton’s visualization of river’s length,

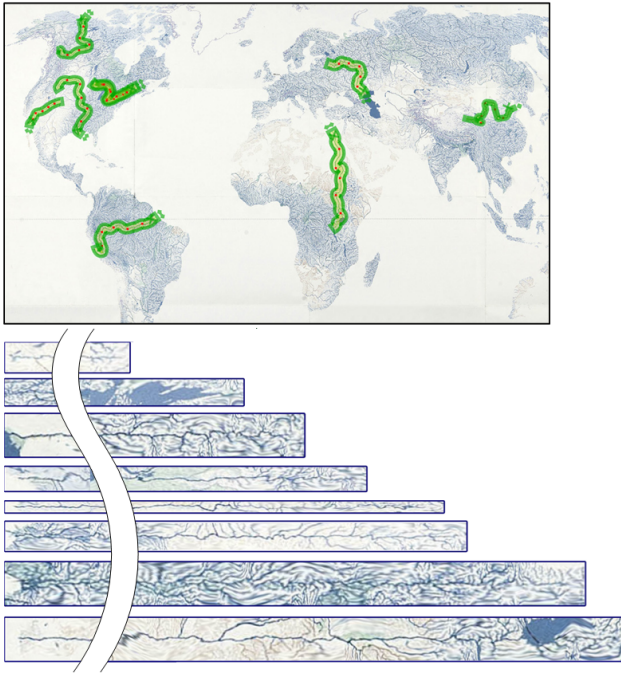


Figure 10. Straightening rivers. Top: rivers are traced on a map (in green), and transmogrified into a straight shape to present the river's length for visual comparison while retaining local context. River imagery is screen capture of National Geographic's World Rivers Water Map (<http://bit.ly/bVOMkq>). From top to bottom: the Colorado, St. Lawrence, Volga, MacKenzie, Yellow, Mississippi, Amazon and Nile. Mid section of the rivers omitted for space.

recently analyzed by Tufte [52], p. 77. Because it can be difficult to compare the lengths of rivers as displayed in their own locations, with different bends and orientations, Colton decided to “hang rivers to dry” in a visualization that, (a) is straightforward to understand, (b) provides a quick, visual answer to the question of distance, and (c) preserves local context and much of the graphical information of the rivers contained in their original graphical representation. What was a compelling but likely a painstakingly difficult visualization to produce for Colton is interactively trivial with a single-point trace transmogrifier (see Figure 10). All that is needed is to simply trace one's finger along the path of the river. This technique can enable quick visual calculations of distance to other elements such as roads, paths, and coastlines. Precision will depend on multiple factors such as the accuracy of the initial representation, whether the ‘magnet’ mode is on, the accuracy of the tracing, and the sampling algorithm.

When transmogrification is used to straighten roads, as in the opening scenario of the paper (Fig. 1.B), the resulting imagery is close in spirit to Matthew Paris's medieval pilgrimage maps [18]. Essentially, the resulting maps abstract away turns and provide a representation where the reader sees to the left or to the right what the travelers would see as their trip progresses.

The example in Fig. 11 uses spatial adjustments achieved through spine scaling for the route from Los Angeles to Las Vegas. The left part of the figure is an undistorted cutout of



Figure 11. Compressing the highway between Las Vegas and Los Angeles. Left: original route cut from surroundings. Right: route transmogrified to show more details around cities and less across the desert.

the map; the right part shows a transmogrification of the route where sampling is intentionally varied. It has been manipulated to provide more detail in the areas that contain more important information (the details such as curves and turns within each city), and less detail in the long straight roads crossing the Mojave Desert. This manipulation is reminiscent of the route maps that people draw when sketching directions, and of those generated by Agrawala and Stolte [1] and Kopf et al. [27]. However, here we only adjust scale along the spine of the transmogrifier; adjusting scale in the dimension perpendicular to the spine can be achieved through a two-finger trace. Adding semantic zoom capabilities requires further work.

Straightening trajectories can offer additional benefits beyond the above length comparisons or traveler-centric maps. For example, transmogrifiers can help represent spatial data that encodes other visual variables. Fig. 12 (left) is an actual route visualization of aircraft taking off from Lyon airport; it shows how the aircraft take different routes that reach high altitudes (in red to blue) at different stages. Optimizing altitude profiles is important because it affects fuel consumption. However, the layout and the turns in the trajectories make color-coded altitude profiles difficult to compare. Fig. 12 (right) shows an alternative representation that rectifies the routes. We discussed this visualization with an air traffic controller, who was able to confirm insights from the graph that are otherwise difficult to perceive in the original layout (e.g., current routes going to the East have less steep profiles, despite heading towards the Alps).

Changing Visualization Layouts

There are multiple instances of visualizations that can benefit from dynamically adjusting their layout geometries; for example, circular or square layouts of trees might be preferable depending on whether the visualization is meant for a single person or a group of people on a tabletop [51]. Fig. 13 shows how an icicle plot is transmogrified into a sunburst tree, and the transmogrifier in Fig. 14 turns a circular chord visualization [33] into a rectangular layout.

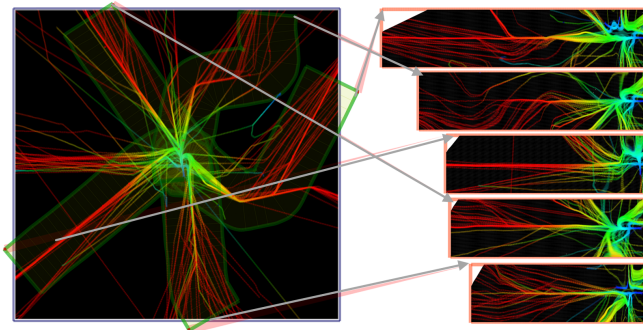


Figure 12. Left) One day of recorded aircraft positions taking off from Lyon airport. The color corresponds to the aircraft altitude (blue for low altitudes, red for high altitudes). Right) The rectified routes. Grey arrows added for clarity.

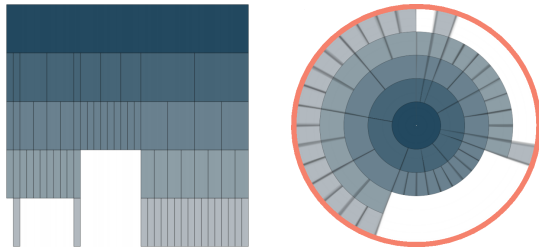


Figure 13. Transmogrification of an icicle plot to a sunburst tree.

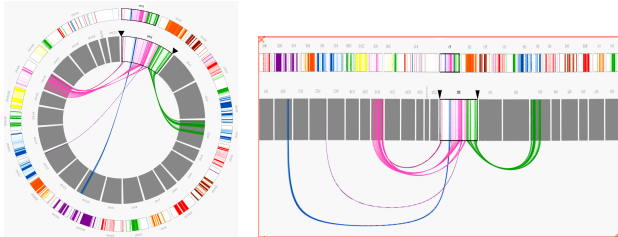


Figure 14. Transmogrification of a circular chord visualization (left) to a rectangular layout (right). Technique from [33].

A different example of visualization re-representation is the reverse of path rectification shown in the opening scenario (Fig. 1.C), where a variable is shown in its spatial context by warping to a path. More complicated geometry transformations can take advantage of the free-form shapes to, for example, adjust a stacked graph [55] by straightening the base line of a data region (Fig. 15).

The recursive features of the TC also enable composite manipulations, such as the elimination of part of the data in a circular graph by first transforming to a square geometry, clipping, and then transforming back to circular (Fig. 16). Note that similarly to removing data, we can add or merge data from multiple sources into a single graph.

DISCUSSION

Just as we often find solutions to problems by testing different alternatives in our minds, generating visual alternatives of visual data sets may be the key to new insights. Conversely, the ability to reverse distortions in representations should also facilitate analyzing data that would otherwise be partially obscured. Many existing techniques are currently being used by analysts and artists precisely for these purposes; for example,

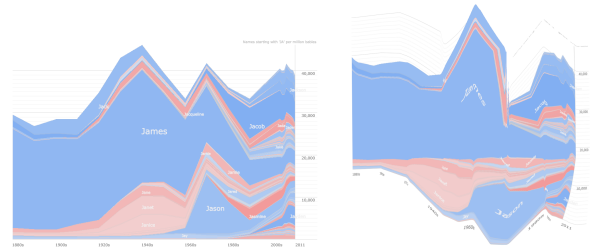


Figure 15. Transmogrification of a stackedbar graph can be used to straighten the base of one of the regions. Left: stacked graph; right: transmogrification. Technique from [55].

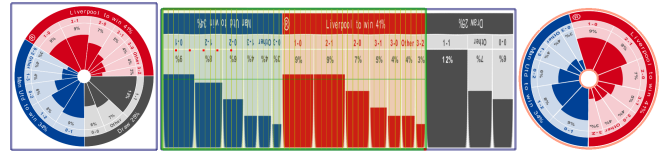


Figure 16. Transmogrification of a Nightingale chart into a bar chart, and then back to Nightingale with a subset of the data.

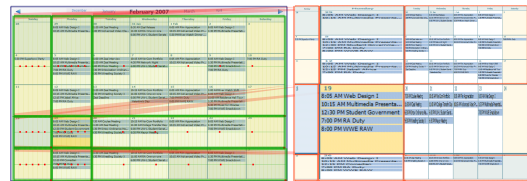


Figure 17. Use of several transmogrifiers to create a Table Lens [41].

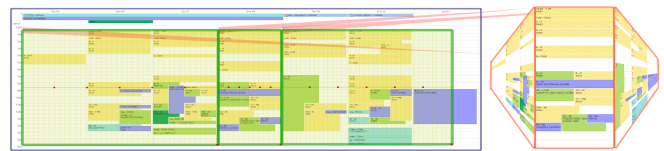


Figure 18. Use of three transmogrifiers to create a Perspective Wall [31].

collages and sketches (digital and paper) are often useful to reorganize data representations of images or data sets. However, it is difficult to find examples of techniques that allow us to quickly use non-linear and arbitrary transformations, especially in a casual setting.

With Transmogrification we extend existing knowledge and algorithms from computer graphics to serve the purpose of information assessment and exploration, particularly in initial stages where a quick look at an alternate view could confirm or refute an idea. We have found that multi-touch manipulation and tracing input lends itself particularly well to the control of graphical distortion techniques, where the large parametric space would be tedious and difficult to control through a single point (e.g., a mouse). Moreover, from our own experience with the system and during demonstrations, we speculate that the directness of touch interaction is particularly suited for deformation techniques because most of our experiences of deformation come from the physical world (e.g., stretching, folding of papers, cloth or other pliable or plastic materials [34]).

From the point of view of information visualization, we see transmogrification as the natural evolution of existing distortion-based techniques such as lenses [9], Melange [17], or Perspective Wall [31]. Recreations of a TableLens and a Perspective Wall with transmogrifiers are shown in

Figures 17 and 18. In fact, it is possible to think about transmogriegers as a generalization of lenses and portal techniques. However, its flexibility and the possibility to transform visuals in arbitrary ways enables a qualitatively different way to use them; instead of applying a technique to an image, we create an ad-hoc interaction technique where visual results are based on direct interaction with the image. This information-centric perspective represents an inversion of most current techniques in visual data exploration.

We have found that transmogriegers can be also useful to prototype and replicate many existing visualizations. In particular, the ability of the canvas to work as a transmogrieger overlay on top of existing interactive visualizations makes it a powerful tool to explore questions about visualizations and about the data itself; for example, would a graph show the same effect if straightened out? How would this look on a perspective-wall? Is there a clear relationship between house prices and distance to the coast?

Although it is too early to say, we think that the composability and bidirectionality of transmogriegers will enable a large set of unexplored avenues. A common problem-solving strategy in physics, math, and engineering is to transform the data into another domain or geometry (e.g., the frequency domain, complex vector space), solve an easier problem in the transformed domain, and transform the solution back to the original domain. Transmogrieger enables a similar process for spatial representations.

Finally, this paper is focused on the application of transmogrieger to information visualization. During the development of transmogriegers and the TC we have found many potential applications for collaborative environments (e.g., reproducing Cut-outs [39]), image editing (e.g., correcting perspective or drawing in perspective), and art and entertainment (e.g., creating caricatures and fabric patterns).

LIMITATIONS AND FUTURE WORK

Visualization tasks are sometimes focused on representation accuracy. The interactive nature of transmogriegers and their reliance on touch traces might make it difficult to achieve shape-accuracy, even with the ‘magnet’ mode. For example, in the hanging rivers example above, the length of a river will be affected by the person’s tracing accuracy which, in turn, is affected by the fat-finger problem. Although we see this more as a consequence of the sketch-like and approximate nature of transmogrieger than as a problem, there are additional input techniques (e.g., input offset) that could enable better precision when needed.

Our implementation has been refined to avoid deformation artifacts when possible, but deformation of raster images always implies the risk of making text and other symbols less readable. We built a parallel version of transmogriegers that addresses this problem through vector rendering. Given an initial point within the source shape we calculate its position in the destination shape. In this way, text or other primitives such as lines or glyphs can simply be drawn at their transformed position as shown in Figure 19. Future work is to automatically extract labels from source raster images (similarly

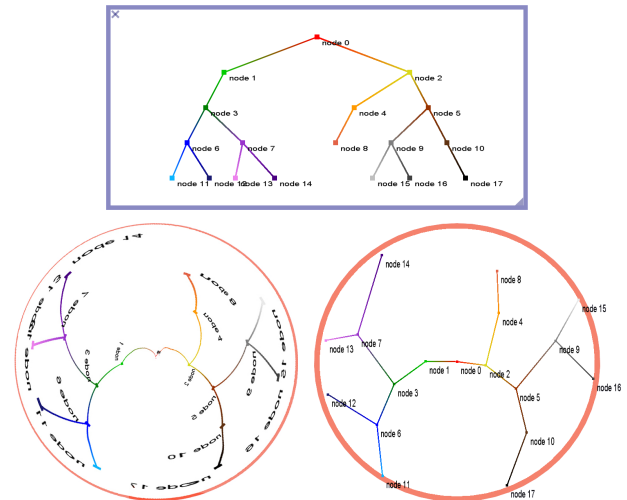


Figure 19. Transmogrieger of a rectangular tree source (top) to circular destinations. Left: raster-based result. Right: vector-based result.

to Savva et al. [44]) and alter transmogriegers to preserve text, or recognize and vectorize text and symbols to present in dual vector-raster transmogriegers.

CONCLUSION

In this paper we have presented transmogrieger, a visual exploration technique based on arbitrary transformations of 2D shapes. The technique is based on the transformation of arbitrarily-shaped source areas of the image into equally arbitrary destination regions. Through the adaptation of graphic transformations based on a spine, and using novel interaction techniques for the specification of shapes based on multi-touch traces, we enable people to design transformations based on the existing image and create real-time alternative visualizations quickly. We also presented examples that explore the application of transmogriegers to visual map exploration, distortion compensation, and layout transformation.

ACKNOWLEDGEMENTS

We thank the reviewers and the InnoVis group. Research support was provided by the Natural Sciences and Engineering Research Council of Canada, SurfNet, GRAND NCE, AITF, Smart Technologies Inc., and the School of Computer Science at the University of St Andrews.

REFERENCES

1. Agrawala, M., and Stolte, C. Rendering effective route maps: improving usability through generalization. In *Proc. SIGGRAPH*, ACM (2001).
2. Bederson, B., and Boltman, A. Does animation help users build mental maps of spatial information? In *IEEE InfoVis* (1999).
3. Beier, T., and Neely, S. Feature-based image metamorphosis. In *Proc. SIGGRAPH*, vol. 26, ACM (1992).
4. Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. Toolglass and magic lenses: The see-through interface. In *Proc. SIGGRAPH*, ACM (1993).
5. Bostock, M., Ogievetsky, V., and Heer, J. D3: Data-driven documents. *IEEE TVCG* 17, 12 (2011).
6. Bottger, J., Brandes, U., Deussen, O., and Ziezold, H. Map warping for the annotation of metro maps. *IEEE CG&A* 28, 5 (2008).

7. Browne, J., Lee, B., Carpendale, S., Riche, N., and Sherwood, T. Data analysis on interactive whiteboards through sketch-based interaction. In *Proc. ITS*, ACM (2011).
8. Burtnyk, N., and Wein, M. Interactive skeleton techniques for enhancing motion dynamics in key frame animation. *Commun ACM* 19, 10 (1976).
9. Carpendale, S., and Montagnese, C. A framework for unifying presentation space. In *Proc. UIST*, ACM (2001).
10. Chao, W., Munzner, T., and van de Panne, M. Poster: Rapid pen-centric authoring of improvisational visualizations with napkinvis. In *Posters Compendium*, InfoVis (2010).
11. Cherlin, J. J., Samavati, F., Sousa, M. C., and Jorge, J. A. Sketch-based modeling with few strokes. In *Proc. SCCG*, ACM (2005).
12. Chevalier, F., Dragicevic, P., and Hurter, C. Histomages: fully synchronized views for image editing. In *Proc. UIST*, ACM (2012).
13. Cockburn, A., Karlson, A., and Bederson, B. B. A review of overview+detail, zooming, and focus+context interfaces. *ACM Comput. Surv.* 41, 1 (2009).
14. Dragicevic, P., Bezerianos, A., Javed, W., Elmqvist, N., and Fekete, J.-D. Temporal distortion for animated transitions. In *Proc. CHI*, ACM (2011).
15. Eitz, M., Sorkine, O., and Alexa, M. Sketch based image deformation. In *Proc. VMV* (2007).
16. Elmqvist, N., Dragicevic, P., and Fekete, J. Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *IEEE TVCG* 14, 6 (2008).
17. Elmqvist, N., Henry, N., Riche, Y., and Fekete, J.-D. Melange: space folding for multi-focus interaction. In *Proc. CHI*, ACM (2008).
18. Gaudio, M. Matthew paris and the cartography of the margins. *Gesta* 39, 1 (2000).
19. Haurert, J.-H., and Sering, L. Drawing road networks with focus regions. *IEEE TVCG* 17, 12 (2011).
20. Heer, J., and Robertson, G. Animated transitions in statistical data graphics. *IEEE TVCG* 13, 6 (2007).
21. Hsu, S. C., Lee, I. H. H., and Wiseman, N. E. Skeletal strokes. In *Proc. UIST*, ACM (1993).
22. Hurter, C., Telea, A., and Ersoy, O. Moleview: An attribute and structure-based semantic lens for large element-based plots. *IEEE TVCG* 17, 12 (2011).
23. Hurter, C., Tissoires, B., and Conversy, S. Fromdady: Spreading aircraft trajectories across views to support iterative queries. *IEEE TVCG* 15, 6 (2009).
24. Igarashi, T., Moscovich, T., and Hughes, J. F. As-rigid-as-possible shape manipulation. *ACM Trans. Graph.* 24, 3 (July 2005).
25. Jacobson, A., Baran, I., Popović, J., and Sorkine, O. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.* 30, 4 (2011).
26. Kass, M., Witkin, A., and Terzopoulos, D. Snakes: Active contour models. *IJCV* 1, 4 (1988).
27. Kopf, J., Agrawala, M., Barger, D., Salesin, D., and Cohen, M. Automatic generation of destination maps. *ACM Trans. Graph.* 29, 6 (2010).
28. Kostelnick, C. The visual rhetoric of data displays: The conundrum of clarity. *IEEE Trans. Prof. Commun.* 50, 4 (2007).
29. Lee, T.-Y., Lin, Y.-C., Sun, Y. N., and Lin, L. Fast feature-based metamorphosis and operator design. *Comp. Graph. Forum* 17, 3 (1998).
30. Lipman, Y., Levin, D., and Cohen-Or, D. Green coordinates. *ACM Trans. Graph.* 27, 3 (2008).
31. Mackinlay, J. D., Robertson, G. G., and Card, S. K. The perspective wall: Detail and context smoothly integrated. In *Proc. CHI* (1991).
32. Meng, W., Zhang, X., Dong, W., and Paul, J.-C. Multicage image deformation on gpu. In *Proc. VRCAI*, ACM (2011).
33. Meyer, M., Munzner, T., and Pfister, H. Mizbee: A multiscale synteny browser. *IEEE TVCG* 15, 6 (2009).
34. Mikulecky, K., Hancock, M., Brosz, J., and Carpendale, S. Exploring physical information cloth on a multitouch table. In *Proc. ITS*, ACM (2011).
35. Nacenta, M. A., Baudisch, P., Benko, H., and Wilson, A. Separability of spatial manipulations in multi-touch interfaces. In *Proc. GI*, Canadian Info. Proc. Society (2009).
36. Nacenta, M. A., Kamber, Y., Qiang, Y., and Kristensson, P. Memorability of pre-designed and user-defined gesture sets. In *Proc. CHI*, ACM (2013).
37. Pietriga, E., and Appert, C. Sigma lenses: focus-context transitions combining space, time and translucence. In *Proc. CHI*, ACM (2008).
38. Pindat, C., Pietriga, E., Chamuis, O., and Puech, C. Jellylens: Content-aware adaptive lenses. In *Proc. UIST*, ACM (2012).
39. Pinelle, D., Stach, T., Stuckel, D., Nacenta, M., Dyck, J., and Gutwin, C. Cutouts: A flexible interaction infrastructure for tabletop groupware. In *Ext. Abs. CSCW* (2006).
40. Pousman, Z., Stasko, J., and Mateas, M. Casual information visualization: Depictions of data in everyday life. *IEEE TVCG* 13, 6 (2007).
41. Rao, R., and Card, S. K. The table lens: merging graphical and symbolic representations in an interactive focus+context visualization for tabular information. In *Proc. CHI*, ACM (1994).
42. Reilly, D. F., and Inkpen, K. M. Map morphing: making sense of incongruent maps. In *Proc. GI*, Canadian Info. Proc. Society (2004).
43. Robertson, G., Cameron, K., Czerwinski, M., and Robbins, D. Animated visualization of multiple intersecting hierarchies. *Information Visualization* 1, 1 (Mar. 2002).
44. Savva, M., Kong, N., Chhajta, A., F.-F., L., Agrawala, M., and Heer, J. Revision: automated classification, analysis and redesign of chart images. In *Proc. UIST*, ACM (2011).
45. Schiwietz, T., Georgii, J., and Westermann, R. Freeform image. *IEEE CG&A* (2007).
46. Shen, Y., Ma, L., and Liu, H. An MLS-based cartoon deformation. *Visual Comput.* 26 (2010).
47. Slack, J., and Munzner, T. Composite rectilinear deformation for stretch and squish navigation. *IEEE TVCG* 12, 5 (2006).
48. Smythe, D. B. A two-pass mesh warping algorithm for object transformation and image interpolation. Tech. Rep. 1030, ILM Comp. Graph. Dept., Lucasfilm, California, USA, 1990.
49. Sun, Q., Fu, C.-W., and He, Y. An interactive multi-touch sketching interface for diffusion curves. In *Proc. CHI*, ACM (2011).
50. Tan, D. S., Meyers, B., and Czerwinski, M. Wincuts: manipulating arbitrary window regions for more effective use of screen space. In *Ext. Abs. CHI*, ACM (2004).
51. Tobiasz, M., Isenberg, P., and Carpendale, S. Lark: Coordinating co-located collaboration with information visualization. *IEEE TVCG* 15, 6 (2009).
52. Tufte, E. R. *Envisioning Information*. Graphic Press, 1990.
53. Tversky, B. Some ways that maps and diagrams communicate. In *Spatial Cognition II*, vol. 1849 of LNCS. Springer, 2000.
54. Tversky, B., Bauer Morrison, J., and Betancourt, M. Animation: can it facilitate? *Int. J. Hum-Comp Studies* 57, 4 (2002).
55. Wattenberg, M. Baby names, visualization, and social data analysis. In *IEEE InfoVis* (2005).
56. Weber, O., and Gotsman, C. Controllable conformal maps for shape deformation and interpolation. *ACM Trans. Graph.* 29, 4 (2010).
57. Weng, Y., Shi, X., Bao, H., and Zhang, J. Sketching MLS image deformations on the GPU. *Comp. Graph. Forum* 27, 7 (2008).
58. Weng, Y., Xu, W., Wu, Y., Zhou, K., and Guo, B. 2d shape deformation using nonlinear least squares optimization. *Visual Comput.* 22 (2006).
59. Wolberg, G. Image morphing: a survey. *Visual Comput.* 14 (1998).
60. Yang, W., and Feng, J. 2d shape manipulation via topology-aware rigid grid. *Computer Animation and Virtual Worlds* 20, 2-3 (2009).