Generalizing Semantic Lenses for Large Element-based Plots

Hurter, Christophe; Ersoy, Ozan; Telea, Alexandru

Published in:
Proceedings ASCI/IPA/SIKS

Generalizing Semantic Lenses for Large Element-based Plots

Christophe Hurter
DGAC-DSNA Toulouse, France

Ozan Ersoy Alexandru Telea
Johann Bernoulli Institute, University of Groningen, the Netherlands

Abstract

Given a spatial embedding of multivariate relational data, we propose a semantic lens which selects a specific spatial and attribute-related data range. The lens keeps the selected data in focus unchanged and continuously deforms the data out of the selection range in order to maintain the context around the focus. Specific deformations include distance-based repulsion of scatter plot points, deforming straight-line node-link graph drawings, and as varying the simplification degree of bundled edge graph layouts. Our technique is simple to implement and provides real-time performance on large datasets.

1 Introduction

In this paper, we present MoleView, a framework for interactive exploration of large *element-based plots*, *i.e.* sets of discrete data elements, each with several data and/or position (layout) attributes, visualized in a single view, rather than linked views. We focus on high-density views where overdraw of data elements is typically present. Examples thereof are node-link layouts, (multidimensional) scatter plots, and images. Our contributions are as follows. First, we extend the well-know semantic lens with a range-based attribute filter to select a 'data layer' at a user-defined point, *i.e.* a set of data elements falling within the lens' position and attribute filter values. Instead of hiding the elements in the lens which fail passing the attribute filter, we use a dynamic re-layouting technique to smoothly push these away from the lens, or pull them back, hence the name of our technique. This allows users to see or brush over what is hidden 'under' the front-most elements. Second, we extend our data-driven deformation idea to explore bundled graphs. Given a bundled and unbundled version of the same graph, we use the MoleView to control the bundling strength *and* which edges get bundled at a certain location. In this way, users can explore bundled graphs (*e.g.* dig into a bundle to extract edges of interest based on attribute value) or, conversely, interactively simplify a given layout by bundling uninteresting edges. Finally, we extend the semantic lens concept for the task of exploring a dataset by the smooth animated interpolation between two completely different layouts of the same data, using as example the exploration of two-dimensional scalar images. This reduces the need for using linked views. Our technique has just a few parameters which are simple to control by end users, can be efficiently implemented to provide real-time interaction, and can be easily added to existing Infovis applications.

In Section 2, we present related work. Section 3 describes the principle of the MoleView technique and its three different modes (elements, bundles, and dual-layout), and illustrates our technique on several datasets. Section 4 discusses the presented technique. Finally, Section 5 concludes the paper.

2 Related work

Related work in Infovis falls within several areas, as follows.

Magic lenses: The Magic Lens modifies a screen region based on a user-selected operator [2, 1]. Tangible magic lenses extended the idea to 'slice' through, or zoom in, layered 2D or 3D datasets by interactively moving a 3D tracked physical planar object (the lens) [18, 13, 26].

Semantic lenses, focus & context, and deformation: The dust-and-magnet technique de-clutters scattered plots by placing data-attribute-driven 'magnets' in the view which attract data points based on the points' attributes [27]. Niels *et al.* visualize vessel movements using a blending technique which groups close trajectories into smooth shaded shapes [22]. Overdraw is eliminated as data is shown as a continuous shaded map. A simple semantic lens emphasizes trajectories, *e.g.* slow ships, via shading and blending values. However, spatial deformation is not used to declutter trajectories, since position data is too important to be altered.

Deformation techniques locally change a given spatial layout to give more space to important elements than to less important ones. Many variations of the original fisheye view exist [6]. For data tables, the table lens locally distorts the Cartesian cell layout to emphasize specific table rows [15]. For node-link layouts, techniques include local edge deformations (EdgeLens variations [25]), and selective edge hiding based on attributes at a focus point. The local edge lens and bring-neighbors lens [21] remove edges between nodes in a focus zone (lens) and pull connected nodes in the lens, respectively. Edge plucking enables users to explicitly drag edges away to clarify cluttered zones [24, 23], albeit with a certain amount of manual effort. Link sliding and 'bring & go' techniques [14] assist the exploration of node-link diagrams by constraining the focus point along a given path in a snap-to-edge manner and moving connected nodes close to the focus. Fisheye techniques have also been proposed for trees [21, 7].

Edge bundling techniques trade off clutter for overdraw by geometrically grouping spatially close edges in a graph [8, 5, 10, 12]. However, overdraw, or edge congestion, makes selection and brushing difficult [24]. The 'digging lens' partially addresses this by thinning overlapping bundles at a focus point [19].

Within the large body of work on lens techniques, our contribution is as follows:

1. *position and data*: we generalize semantic lenses to combine position and data attributes;

2. *lens shape:* we generalize the lens from simple shapes to

arbitrary user-painted 2D shapes;

3. *animation:* we use smooth animation to continuously deform elements within the lens;
4. *dual layout:* we generalize the deformation to interpolate between two different layouts of a given dataset;
5. *element types:* we treat any elements with position and data values uniformly (points, pixels, graph edges, bundles).

3 MoleView principle

As input, we consider a dataset $D = \{s_i\}$ of elements s_i with 2D layout positions $L = \{p_i = (x_i, y_i) \in \mathbf{R}^2\}$ and attributes $v_i = \{v_{ij} \in \mathbf{R}\}$. Examples are scatterplots, where s_i are data points; images, where s_i are pixels with color values; and node-link graph drawings, where s_i are nodes, edge control points, entire edges, or edge bundles. Positions p_i may overlap or not.

Exploring a 2D rendering of D starts by defining a so-called *focus* zone $Z \subset \mathbf{R}^2$. MoleView supports tasks involving the understanding of the spatial structure and data attribute distribution of elements $s_i \in D$ within focus (*i.e.* $p_i \in Z$), as follows. First, we select the elements $D^Z \subset D$ which are spatially within Z. Secondly, we filter D^Z to a subset D^{sel} of elements within the attribute region-of-interest (ROI) A. Third, we apply a smooth spatial deformation $\Delta : \mathbf{R}^2 \times \mathbf{R}^+ \rightarrow \mathbf{R}^2$ from the original layout $L^{filt} = \{p_i \in \mathbf{R}^2 | s_i \in D^{filt}\}$ of the so-called filtered elements $D^{filt} = D^Z \setminus D^{sel}$ to yield a new layout $L^{filt}_{new} = \Delta(L,t)$. The time parameter $t > 0$ morphs in both directions between L^{filt} and L^{filt}_{new} as the lens is activated, respectively deactivated. Suitable choices of Δ allow us to perform decluttering, selective fisheye-like exploration (Sec. 3.1), bundled graph exploration (Sec. 3.2), and also correlating data elements across layouts (Sec. 3.3).

3.1 Element-based exploration

Consider a simple dataset D whose elements s_i have just position p_i and an attribute value v_i. The user first defines a so-called *control set* $P \subset \mathbf{R}^2$ by direct interaction, *i.e.* brushing in the visualization using the mouse. In the simplest case, P is one or a few points, like in [25]. Next, we define the focus Z as a distance field $D_Z(P) : \mathbf{R}^2 \rightarrow \mathbf{R}^2$, as follows. First, we compute the distance transform $DT_P : \mathbf{R}^2 \rightarrow \mathbf{R}_+$ [4]

$$DT_P(x \in \mathbf{R}^2) = \min_{y \in p} \|x - y\| \tag{1}$$

Given DT_P, Z is simply the level set of DT_P at a user-specified distance $\delta > 0$. The size of the Z, δ, is controlled via the mouse wheel with a modifier key (Control). Hence, all elements *spatially* within Z are

$$D^Z = \{s_i \in D | DT_P(p_i) \leq \delta\} \tag{2}$$

Computing D^Z is simple for any data element shapes: We render a shape and apply the test in Eqn. 2 at each rendered pixel.

Next, we select the elements $D^{sel} \subset D^Z$ which are within focus *and* also within the attribute ROI, using *e.g.* range selection

$$D^{sel} = \{s_i \in D^Z | v_i \in [v_{min}, v_{max}]\} \tag{3}$$

Other attribute tests can be substituted immdiately. The range $[v_{min}, v_{max}]$ is controlled by the mouse wheel. The MoleView comes now into action: We keep points D^{sel} at their locations p_i and advect the points $p_i \in D^{filt}$ in the gradient field $-\nabla DT_P$ with a speed $\|\mathbf{v}\|$ which decreases as points get close to the lens border and further from the control set P. In detail

$$\mathbf{v} : \mathbf{R}^2 \rightarrow \mathbf{R}^2, \quad \mathbf{v}(x) = -\nabla DT_P(x)\ \lambda\left(\frac{DT_P(x)}{\delta}\right) \tag{4}$$

The function $\lambda : [0,1] \rightarrow [0,1], \lambda(0) = 1, \lambda(1) = 0$ decelerates points as they get close to the lens border. In practice, exponential decaying profiles give smooth animation results. The advection implicitly yields a deformation $\Delta(t)$ of the filtered points' layout. Advection of points in D^{filt} starts at mouse clicking and stops when the mouse button is released, thereby creating a smooth animation. The user can move the lens (control set P) by moving the mouse, so points smoothly enter into, or exit from, the focus. When the lens is deactivated (mouse button release), we change $\mathbf{v}$ to an attraction field $\mathbf{V}$, defined at the current location of the displaced points p_i^{disp} as

$$\mathbf{V}(p_i^{disp}) = p_i - p_i^{disp} \tag{5}$$

where p_i are the unmoved point positions. This smoothly pulls back the displaced points *i.e.* reverse the lens effect. For additional cues, we linearly interpolate the transparency of the displaced elements p_i between a low value α_{min} at $DT_P = 0$ and a high value $\alpha_{max} = 1$ at $DT_P = \delta$, *i.e.* on the border of Z.

Trail dataset example: Our dataset is a set of 17275 trajectories (trails) whose end points indicate French airport locations. Trails are flight routes between airports. Each trail is a sequence of geographical locations with altitude data. Altitude is visualized by color mapping.

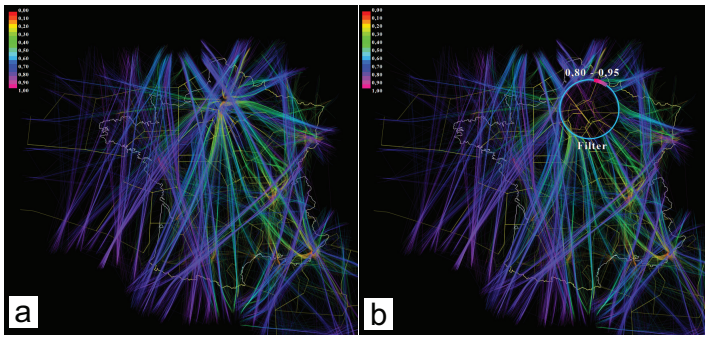

Figure 2: Flight trails dataset (a); element-based lens (b)

Rendering all trails with altitude-colored edges generates high occlusion (Fig. 2 a). We desire to study flights with a given altitude (variation) over a given spatial region, *e.g.* high-altitude flights, or take-off and/or touch-down flight portions [11]. For this, we select a circular focus by moving the lens to some location. Next, we tune the focus size and altitude ROI using the mouse wheel. The attribute ROI $[v_{min}, v_{max}]$ is shown by the colored bar on the lens's periphery, which moves around the center as the mouse wheel turns. Trails continuously move in

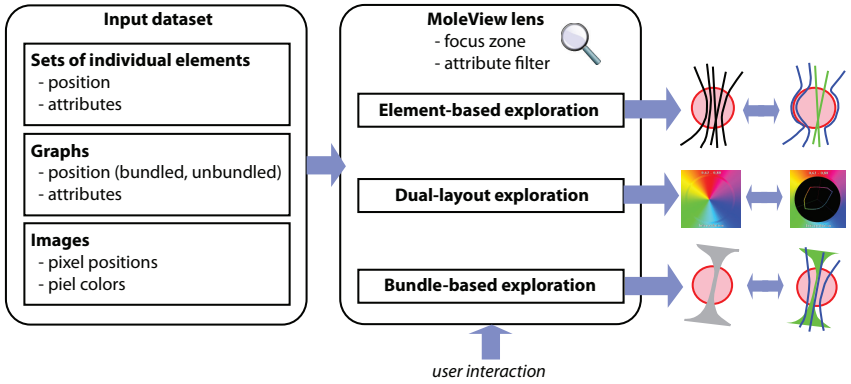

Figure 1: MoleView interactive exploration pipeline

or out of the lens as parameters are changed (see the video at [1]). Trails selected by the lens stay unmoved, hence are easier to spot. The overall effect reminds of a mole pushing earth (data elements) around as it digs, hence the name for our technique.

Image data example: Consider now an image dataset D. Elements of D are pixels with grayscale or color attributes. Figure 3 a-c show the lens applied to an ultrahigh-resolution angiography image of the human eye. The attribute ROI retains bright pixels (important blood vessels) and pushes darker pixels away from the focus, revealing blood vessels in context. Images (d-f) show the lens applied to a color-coded image of the Lisbon night traffic. Green shows slow moving vehicles. We now set the attribute ROI on hue to retain the green range, so we reveal slow motion traffic but also keep the spatial map context.

3.2 Bundle-based exploration

We now consider the more specific case of a dataset D representing a bundled graph. Elements s_i are individual edge control points, entire edges, or entire bundles. As outlined in Sec. 2, bundling simplifies large graphs but also increases overdraw. This makes it hard to see which edges are part of a bundle, unless the bundling is data-driven, which is not the case in all examples we are aware of. For instance, hierarchical edge bundles (HEBs) used in software visualization have data attributes like the dependency type (call, uses, inherits, includes) or number of times and moment when a function gets called [9]. In reverse-engineering and quality assessment we need to understand how such attributes are distributed over the edges in a bundle.

Given a user-defined control set and focus (Sec. 3.1), we consider a bundled layout L^b and an unbundled layout L^u of a graph. We apply our lens (Sec. 3) by setting the original and deformed layouts L and L^{filt} to L^b and L^u respectively. The deformation Δ smoothly interpolates between L^b and L^u rather than moving points away from the focus as for element-based exploration.

$$\Delta(t, p_i) = \lambda(t)L^b + (1-\lambda(t))L^u \qquad (6)$$

When the lens is deactivated, moved elements go back smoothly to their positions in L^b by applying Eqn. 5 like for element-based exploration.

[1] www.cs.rug.nl/svcg/SoftVis/Moleview

Point-level exploration: Figure 4 left shows the bundle-based lens for the trails graph. Compared to Fig. 2, trails in the altitude ROI smoothly get *unbundled* rather than being pushed out of the lens. By swapping L^b and L^u in Eqn. 6 and using the lens on an unbundled graph, we can locally bundle selected elements while leaving all filtered elements unmoved, or locally bundle filtered elements leaving all selected ones unmoved. These scenarios allow different focus-and-context effects.

Edge-level exploration: Elements $s_i \in D$ are now whole edges rather than edge control points. We now apply the deformation (Eqn. 6) to *all* control points of edges in the lens rather than to points in the lens. In Fig. 5), complete flights through the Paris area are smoothly bundled, while other flights are kept unmoved. This is useful when the exploration focuses on an entire edge set passing through a region.

Bundle-level exploration: At the coarsest level, we consider a whole bundle as an element s_i. Fig. 6 we have a radial layout of a software system (nodes are software entities; edges are dependencies). Bundles are assigned different colors (a). Local unbundling reveals the structure at some focus point (b). We could now use color to show some other attribute since unbundling eliminates overdraw. We can also unbundle whole bundles under the lens (c). Finally, we can combine the local and whole-edge unbundling effects to achieve a two-stage unbundling effect (d). When animated, this gives additional cues as to the identities of the bundles brushed by the lens, but keeps clutter minimal within the lens area (see the referred video).

3.3 Dual-layout exploration

We now explore a dataset D via two completely different layouts. Figure 7 shows the dual-layout lens applied to two color-mapped scalar fields. Now, pixels are smoothly advected in a deformation field $\Delta(t)$ from their location L_C in their natural Cartesian layout (leftmost images) to their location in a HSV layout L_P with hue mapped to angle, saturation mapped to radius, and value (luminance) seen as an attribute. We apply our lens on all elements in the zone of interest, *i.e.* $D^{sel} = D^Z$. This yields effectively a histogram of the hues and saturations of the pixels in the lens.

The first field (a-d) shows the frequency of lightning on the Earth surface with a heat colormap. The pixel patterns in the HSV space in the lens in (b,c) are nearly identical, so these

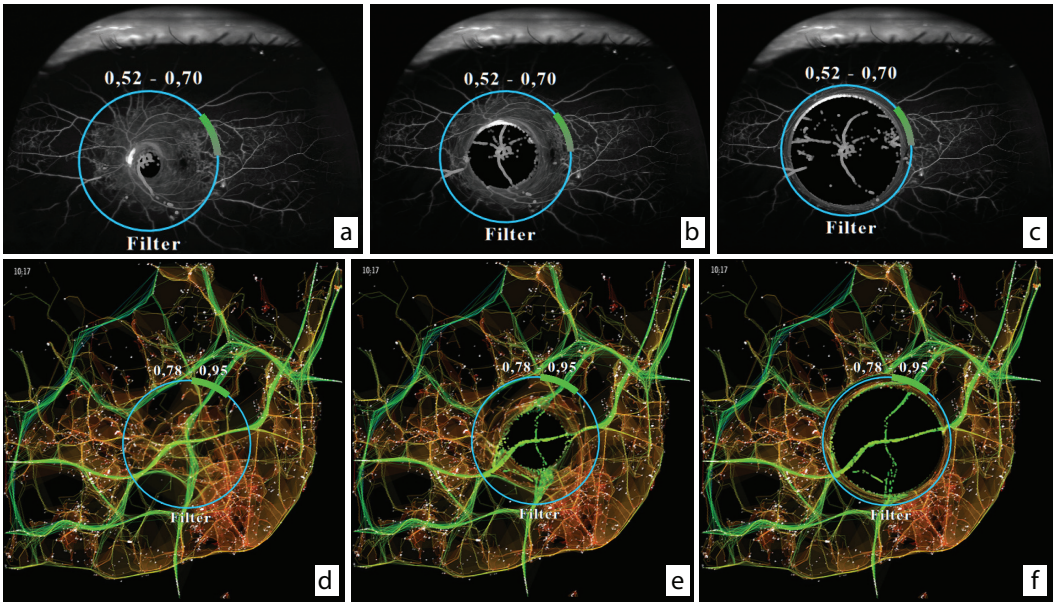

Figure 3: Element-based MoleView applied to grayscale angiography image (a-c) and color-mapped traffic speed image (d-f)

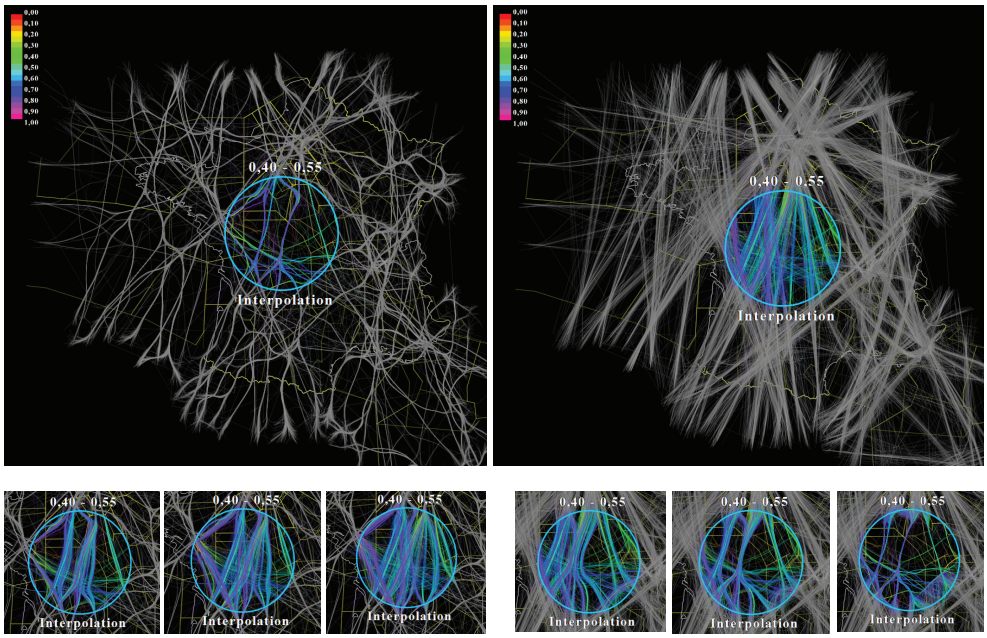

Figure 4: Bundle-based exploration (Sec. 3.2). Local unbundling (left). Local bundling (right)

zones have similar lightning distributions. The zone in lens in (d) is different – the green-blue 'tail' of the shape in the lens in (b,c) is missing. Hence, the zone (d) has no low lightning frequencies. The original image (a) does not show the above – the pixel color patterns in the three regions are quite similar.

The second scalar field (e-g) shows a 3D skeleton, or medial axis, of a cow model, computed with the voxel-based method in [16]. Skeleton voxels are colormapped by their so-called importance on a rainbow colormap. Less important skeleton points (blue) are for small-scale object features, *e.g.* the horns or hoof tips. Most important points (yellow..red) are for large object features, like the rump. The skeletonization method in [16] *conjectures*, but does not rigorously prove or disprove, that the importance of skeleton points varies smoothly over small, connected, skeleton areas. We use our dual-layout lens to check this hypothesis. In the head region (f), the lens shows a continuous blue-to-green curve, *i.e.* voxels in this region have, indeed, importances covering the low-to-medium range. In the rump regions (g,h) the lens shows, as expected, a broader color spectrum, since voxel importances here go from very low (blue) to highest in the model (red). However, these curves are not continuous, but *broken* in the yellow range. Hence, there are no voxels here with medium-high importance values, which questions the validity of the conjecture in [16]. Given that we worked with this skeletonization method and model for about a year in a different project, this was an unexpected result. Close examination revealed the answer: the model contains some small-scale incorrect importance discontinuities, almost impossible to no-

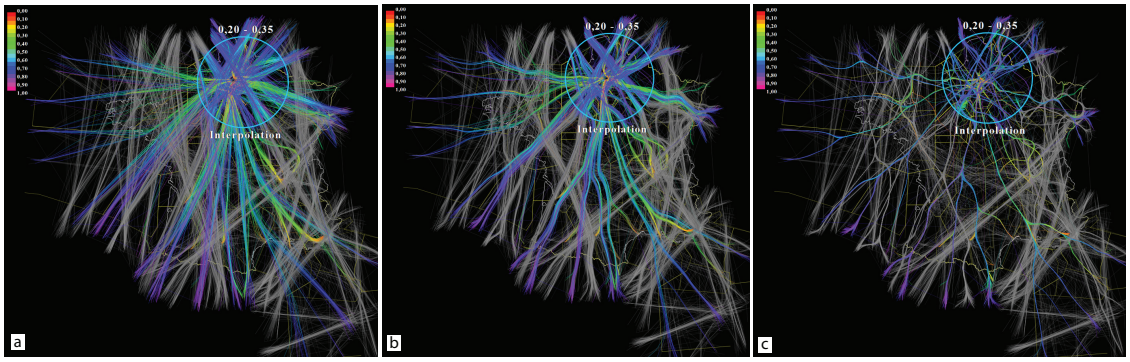

Figure 5: An unbundled dataset (a) is gradually bundled within the focus zone (b), finally yielding the bundled dataset (c)

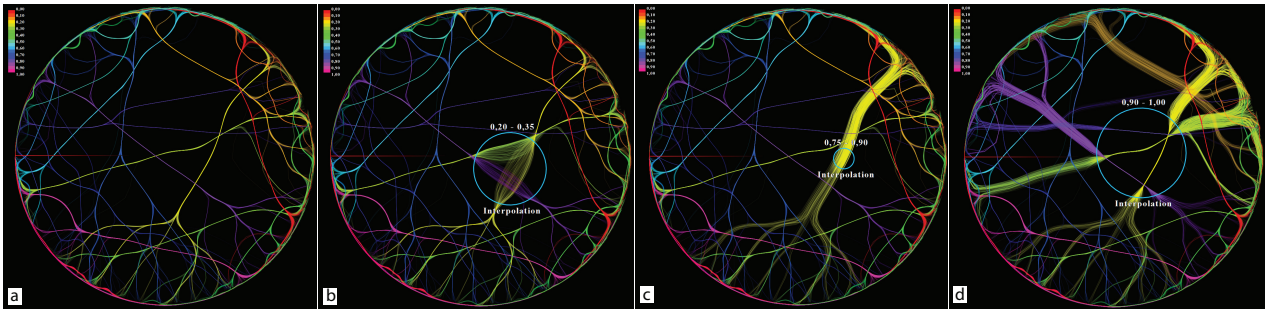

Figure 6: Bundle-level lens on a software graph. Original layout (a). Local (b), whole-edge (c), and combined unbundling (d)

tice in standard views (e), but salient when using the lens.

3.4 Implementation

MoleView can be efficiently implemented. First, we compute the distance transform DT_P of the control set P (Eqn. 1, Sec. 3.1) using the GPU-based method of Cao *et al.* [3]. This method also delivers the feature transform $FT_P : \mathbf{R}^2 \rightarrow \mathbf{R}^2$ of P defined as

$$FT_P(x \in \mathbf{R}^2) = argmin_{y \in P} \|x - y\| \qquad (7)$$

Since $|FT_P| = -\nabla DT_P$ [20], we can obtain in this way the gradient field needed in Eqn. 4 with no numerically sensitive differentiation. The method of Cao *et al.* is $O(N)$ for a N pixel image, roughly 4 milliseconds on a CUDA GT 330M laptop card for a 800^2 image. This delivers fluid real-time interaction.

4 Discussion

Animation is key element to MoleView's effectiveness: by continuously changing the position of the points affected by the lens, one can brush through a dataset and obtain smooth change of the visualization. Smoothness is also present when toggling the lens on and off, thereby creating a context-and-focus effect. The MoleView and the EdgeLens bubble variant [25] are related techniques. However, there are several differences. First, MoleView is not limited to decluttering edges in node-link diagrams. For this, the usage of a general advection field, rather drawing edges as Bézier curves as in EdgeLens, is essential. Secondly, select the attribute ROI allows exploration based on data *and* spatial position rather than spatial position only. Our control set (Sec. 3) is any subset $P \subset \mathbf{R}^2$, specified by direct painting. The lens shape, and its repulsion vector field computed using the shape's feature transform, yield very different advection patterns than when using a few discrete foci as in EdgeLens. FT_P yields a locally smooth field wherever P does not have strong curvature discontinuities [17]. The set of discontinuities of FT_P is a null set corresponding to the feature points of the branching points of the skeleton S_Z of the zone of interest Z [17]. There are no practical robustness or quality issues when advecting elements, as these move *away* from S_Z.

5 Conclusion

We have presented MoleView, a set of interactive lens techniques for large 2D spatial datasets. Three exploration modes are presented. The element-based mode repels filtered data points in a distance field, thus unearthing points which may be obscured due to overdraw. The bundle-based mode locally deforms a bundled layout into an unbundled one or conversely, for specific attribute values.The dual-layout mode smoothly interpolates between two different layouts allowing correlations between two data views. Several extensions are possible. The attribute filter can operate on a histogram of data values in the lens rather than the values themselves, allowing better outlier detection. Also, the dual-layoutmode can work beyond Cartesian RGB plots and HSV polar plots, *e.g.* to correlate completely different graph layouts for graph exploration or 2D plots which show pairs of dimensions in a multivariate dataset.

References

[1] E. Bier, M. Stone, and K. Pier. Enhanced illustration using MagicLens filters. *IEEE CG & A*, 17(6):62–70, 1997.

[2] E. Bier, M. Stone, K. Pier, W. Buxton, and T. DeRose. Toolglass and magic lenses: The see-through interface. In *Proc. ACM SIGGRAPH*, pages 137–145, 1993.

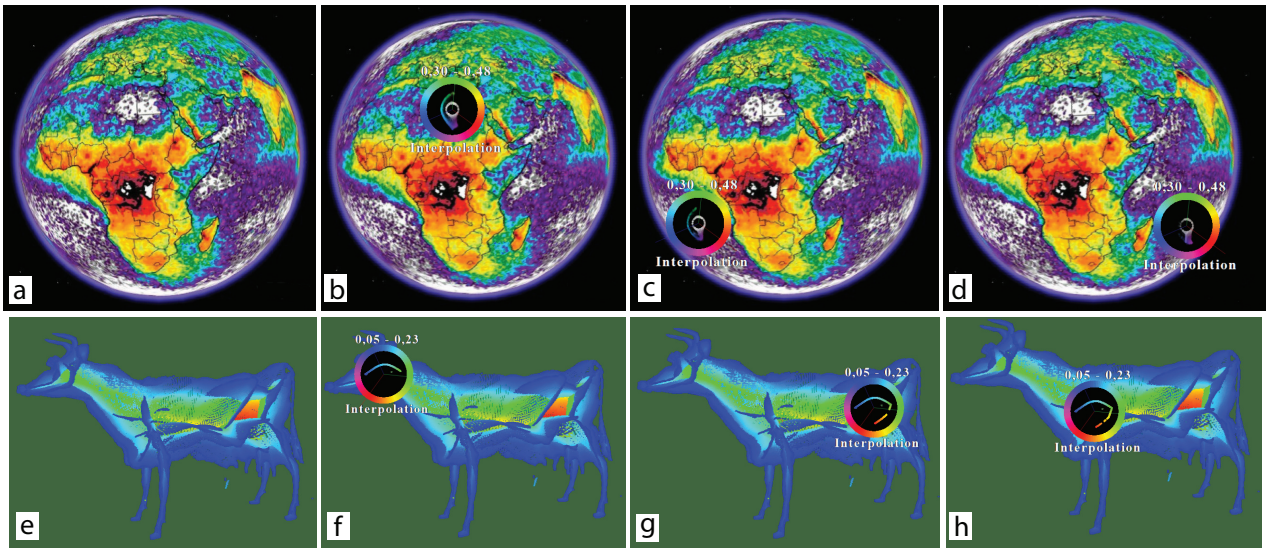

Figure 7: Dual-layout lens applied to two color-coded scalar field images. Top row: lightning frequency on the surface of the Earth (heat colormap). Bottom row: 3D skeleton color-coded by importance (rainbow colormap)

[3] T. Cao, K. Tang, A. Mohamed, and T. Tan. Parallel banding algorithm to compute exact distance transform with the GPU. In *Proc. ACM SIGGRAPH Symp. on Interactive 3D Graphics and Games*, pages 134–141, 2010.

[4] L. Costa and R. Cesar. *Shape analysis and classification: Theory and practice*. CRC Press, 2000.

[5] W. Cui, H. Zhou, H. Qu, P. Wong, and X. Li. Geometry-based edge clustering for graph visualization. *IEEE TVCG*, 14(6):1277–1284, 2008.

[6] G. Furnas. Generalized fisheye views. In *Proc. ACM CHI*, pages 16–23, 1986.

[7] E. Gansner, Y. Koren, and S. North. Topological fisheye views for visualizing large graphs. In *Proc. InfoVis*, pages 175–182, 2004.

[8] D. Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE TVCG*, 12(5):741–748, 2006.

[9] D. Holten and J. J. van Wijk. Visual comparison of hierarchically organized data. *Comp. Graph. Forum*, 21(4):759–766, 2008.

[10] D. Holten and J. J. van Wijk. Force-directed edge bundling for graph visualization. *Comp. Graph. Forum*, 28(3):670–677, 2009.

[11] C. Hurter, B. Tissoires, and S. Conversy. FromDaDy: Spreading data across views to support iterative exploration of aircraft trajectories. *IEEE TVCG*, 15(6):1017–1024, 2009.

[12] A. Lambert, R. Bourqui, and D. Auber. Winding roads: Routing edges into bundles. *Comp. Graph. Forum*, 29(3):432–439, 2010.

[13] J. Looser, R. Grasset, and M. Billinghurst. A 3D flexible and tangible magic lens in augmented reality. In *Proc. ISMAR*, pages 254–262. IEEE, 2007.

[14] T. Moscovich, F. Chevalier, N. Henry, E. Pietriga, and J.-D. Fekete. Topology-aware navigation in large networks. In *Proc. ACM CHI*, pages 320–329, 2009.

[15] R. Rao and S. Card. The table lens: merging graphical and symbolic representations in an interactive focus+context visualization for tabular information. In *Proc. CHI*, pages 348–356, 1994.

[16] D. Reniers, J. J. van Wijk, and A. Telea. Computing multiscale skeletons of genus 0 objects using a global importance measure. *IEEE TVCG*, 14(2):355–368, 2008.

[17] K. Siddiqi and S. Pizer. *Medial Representations: Mathematics, Algorithms and Applications*. Springer, 1999.

[18] M. Spindler and R. Dachselt. Exploring information spaces by using tangible magic lenses in a tabletop environment. In *Proc. ACM CHI (EA)*, pages 243–248, 2010.

[19] A. Telea and O. Ersoy. Image-based edge bundles: Simplified visualization of large graphs. *Comp. Graph. Forum*, 29(3):543–551, 2010.

[20] A. Telea and J. J. van Wijk. An augmented fast marching method for computing skeletons and centerlines. In *Proc. VisSym*, pages 251–259, 2002.

[21] C. Tominski, J. Abello, F. van Ham, and H. Schumann. Fisheye tree views and lenses for graph visualization. In *Proc. Information Visualisation*, pages 202–210, 2006.

[22] N. Willems, H. van de Wetering, and J. J. van Wijk. Visualization of vessel movements. *Comp. Graph. Forum*, 28(3):959–966, 2009.

[23] N. Wong and S. Carpendale. Supporting interactive graph exploration with edge plucking. In *Proc. IEEE Visualization (interactive posters)*, 2005.

[24] N. Wong and S. Carpendale. Supporting interactive graph exploration using edge plucking. In *Proc. SPIE*, pages 235–246, 2007.

[25] N. Wong, S. Carpendale, and S. Greenberg. EdgeLens: An interactive method for managing edge congestion in graphs. In *Proc. IEEE InfoVis*, pages 167–175, 2003.

[26] Y. Yang, J. Chen, and M. Beheshti. Nonlinear perspective projections and magic lenses: 3D view deformation. *IEEE CG & A*, 25(1):567–582, 2005.

[27] J. Yi, R. Melton, J. Stasko, and J. Jacko. Dust & magnet: Multivariate information visualization using a magnet metaphor. *J. of Information Visualization*, 4(4):542–551, 2006.