Editor: Theresa-Marie Rhyne

Visual Embedding

A Model for Visualization

Çağatay Demiralp
Stanford University

Carlos E. Scheidegger
AT&T Labs Research

Gordon L. Kindlmann
University of Chicago

David H. Laidlaw
Brown University

Jeffrey Heer
University of Washington

Automating the design of effective visualizations is an unsolved problem. Although researchers have proposed numerous guidelines and heuristics, a formal framework for design and evaluation is still elusive. Instead, conducting a posteriori user studies is still the primary tool for assessing a visualization's effectiveness. Using theoretical models presents another, albeit less explored, approach (for background on such models, see the sidebar). We believe that the generative potential of model-based visualizations can accelerate design and complement the summative nature of user studies.

Developing a theory of visualization that is both descriptive and generative is difficult. The space of visualizations is large, and the use of visualization spans many issues in human perception and cognition. Additional factors, such as interaction techniques, can significantly affect a visualization's success. Given our current knowledge, visualization design is an underconstrained problem. So, there is value in developing simpler, constrained models, each addressing certain aspects of visualization while ignoring others, like spotlights on a theater stage.

In this context, we introduce *visual embedding* as a model for visualization construction. We define a visualization as a function that maps from a domain of data points to a range of visual primitives (see Figure 1). We claim a visualization is "good" if the embedded visual elements preserve structures present in the data domain. A function meeting this criterion constitutes a visual embedding of the data points.

Our model is motivated by the fact that understanding patterned structures in data is a primary goal of visual analysis. The proposed basic framework can be used to generate and evaluate visualizations on the basis of both the underlying data and—through the choice of preserved structure—desired perceptual tasks. Our model is a generalization of earlier work on structure-preserving colorings.[1]

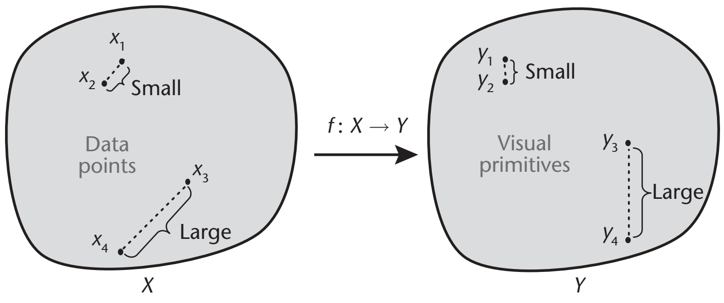


Figure 1. Visual embedding is a function that preserves structures in the data (domain) within the embedded perceptual space (range).

Representing Structures in Data

Structure is "the arrangement of and relations between the parts or elements of something complex" (http://oxforddictionaries.com/definition/english/structure). How, then, can we express structures in data? Unfortunately, a user might not explicitly know about important structures in the data, let alone be able to express or quantify them.

On the other hand, users often can hypothesize

Related Work in Visualization Models

Researchers have proposed general and specific models of visualization. Owing to space limitations, we confine our discussion to a small but representative subset.

Jock Mackinlay introduced one of the most influential systems for automatically generating visualizations.[1] Following Jacques Bertin's aphorism of graphics as a language for the eye,[2] Mackinlay formulated visualizations as sentences in a graphical language. He argued that good visualizations will meet the criteria of expressiveness and effectiveness. A visualization meets the expressiveness criterion if it faithfully presents the data, without implying false inferences. Effectiveness concerns how accurately viewers can decode the chosen visual-encoding variables; it's informed by prior studies in graphical perception (for example, by William Cleveland and Robert McGill[3]). Mackinlay's APT (A Presentation Tool) employed a composition algebra over a basis set of graphical primitives derived from Bertin's encodings to generate alternative visualizations. The system then selected the visualization that best satisfied formal expressiveness and effectiveness criteria.

APT didn't explicitly take into account user tasks or interaction. To this end, Steven Roth and his colleagues extended Mackinlay's work with new types of interactive presentations.[4] Similarly, Stephen Casner built on APT by incorporating user tasks to guide visualization generation.[5] Some of these ideas now support visualization recommendation in Tableau (www.tableausoftware.com), a commercial visualization tool.

Donald House and his colleagues' automatic visualization system integrated user preferences.[6] Genetic algorithms refined a population of visualizations in response to user ratings. In contrast to this empirical approach, Daniel Pineo and Colin Ware used a computational model of the retina and primary visual cortex to automatically evaluate and optimize visualizations.[7] Jarke van Wijk argued for first modeling a perceptual domain (for example, luminance or shape perception) and then optimizing for some perceptual goal according to that model.[8] Visual embedding can be viewed as a reusable template within van Wijk's discussion on perceptually optimal visualizations.

If we chose a motto for visual embedding, it would be "visualization as a perceptual painting of structure in data." In this sense, visual embedding's perceptual-structure preservation criterion closes the cycle, explicitly linking Mackinlay's expressiveness and effectiveness criteria while providing a recipe to achieve both (see the main article).

References

1. J.D. Mackinlay, "Automating the Design of Graphical Presentations of Relational Information," *ACM Trans. Graphics*, vol. 5, no. 2, 1986, pp. 110–141.
2. J. Bertin, *Semiology of Graphics*, Univ. of Wisconsin Press, 1983.
3. W. Cleveland and R. McGill, "Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods," *J. Am. Statistical Assoc.*, vol. 79, no. 387, 1984, pp. 531–554.
4. S.F. Roth et al., "Interactive Graphic Design Using Automatic Presentation Knowledge," *Proc. SIGCHI Conf. Human Factors in Computing Systems* (CHI 94), ACM, 1994, pp. 112–117.
5. S.M. Casner, "Task-Analytic Approach to the Automated Design of Graphic Presentations," *ACM Trans. Graphics*, vol. 10, no. 2, 1991, pp. 111–151.
6. D.H. House, A. Bair, and C. Ware, "An Approach to the Perceptual Optimization of Complex Visualizations," *IEEE Trans. Visualization and Computer Graphics*, vol. 12, no. 4, 2006, pp. 509–521.
7. D. Pineo and C. Ware, "Data Visualization Optimization via Computational Modeling of Perception," *IEEE Trans. Visualization and Computer Graphics*, vol. 18, no. 2, 2012, pp. 309–320.
8. J.J. van Wijk, "Model-Based Visualization: Computing Perceptually Optimal Visualizations," *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, Springer, 2009, pp. 343–350.

some notion of distance between data points. Using pairwise distances is one simple and general way to implicitly express structures in spaces. For example, if a function transforms a 2D or 3D Euclidean space while preserving pairwise Euclidean distances, the shape and size of objects in the space will stay the same. Similarly, if a function existed from a sphere to a plane that preserved all pairwise distances on the sphere, we'd have world maps without distortion (angles and areas would be simultaneously preserved). Structure can be operationalized in terms of these atomic pairwise relations; in this context, the visualization function should picture what these pairwise relations amount to.

Ideally, distance in the data space should reflect users' understanding of the similarity between data points as it relates to their current task. This lets them hint at the type of structures they're interested in seeing. For instance, if a user is interested in symmetries, he or she should provide a measure that quantifies these relationships. In fact, structural criteria such as symmetry and continuity often serve as design choices in creating visualizations. In contrast, distance in the visual space should convey the perceptual distances between visual primitives.

Of course, there can be many other ways of expressing structures in data. However the structure is expressed, the corresponding perceptual range

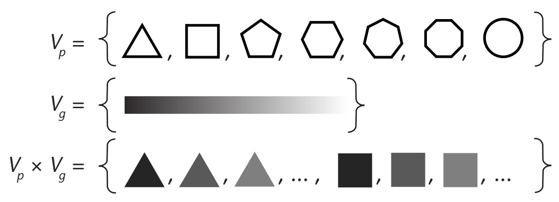


Figure 2. A visual product space. To create these spaces, we use the Cartesian product of existing visual spaces (in this case, V_p and V_g).

should be able to accurately convey that structure. One advantage of using pairwise distances is that their application to visual spaces is conceptually straightforward. We can encode them as perceptual differences of color, shape, texture, spatial distance, size, and so on. The following discussion uses pairwise distances to express structures in data.

Estimating Perceptual Distances with Crowdsourcing

To assess structural preservation, we require perceptual-distance measures for a given visual-embedding space. However, except for a few perceptually uniform color spaces, we don't have these measures for most visual spaces. In these cases, online crowdsourcing can help us estimate perceptual distances.[2]

A visual space is perceptually uniform if a perturbation to any element in it results in a proportional change in a viewer's percept. For example, perceptual experiments find that linear mappings for 2D position or 1D length are perceptually linear. By design, the CIELAB color space is approximately perceptually uniform; RGB and CIEXYZ aren't. The Euclidean distance between two color points in CIELAB is approximately proportional to the empirically reported perceptual difference between the colors. Conversely, a small change to RGB or CIEXYZ triplet values might cause a disproportionate change in perceived colors.

Crowdsourcing is one way to collect large, diverse perceptual data samples. For example, Jeffrey Heer and Michael Bostock replicated prior graphical-perception results using crowdsourced experiments on Amazon's Mechanical Turk.[2] CIELAB was also, in a sense, crowdsourced: it was created by fitting an appearance model to observers' color-scale judgments. We demonstrate the viability of this approach later.

New Visual Spaces from Old: Visual Product Spaces

Formulating visualizations as structure-preserving functions raises the possibility of transferring other related mathematical concepts. Product spaces (or sets) provide one example: we can generate a new visual space using the Cartesian product of existing visual spaces. We call this space a *visual product space* (see Figure 2).

Generally, the product of two perceptually uniform visual spaces won't be uniform. On the other hand, when we have two topological spaces endowed with metrics, constructing a metric for the product space is straightforward. One challenge is to discover whether cases exist that have an analogous procedure for constructing *visual product metrics*. This issue resonates strongly with research on interactions between perceptual dimensions (for example, integral versus separable visual encodings[3]). Searching the literature for separable cases might be a promising starting point.

Under our model, constructing a good visualization function is fundamentally an optimization problem. The nature of embedding spaces often determines the available techniques. The spaces can be Euclidean (for example, most color spaces, including RGB, CIELAB, and CIELUV), continuous but non-Euclidean (for example, parametric shape spaces and texture spaces), or discrete (for example, finite sets of icons, shapes, glyphs, and fonts). Some of the many techniques for embedding a domain in Euclidean space are principal component analysis, multidimensional scaling, isometric feature mapping, and locally linear embedding.[4]

Although embedding in the Euclidean space is computationally well studied, embedding in non-Euclidean spaces (continuous or discrete) is not. We can formulate the latter problem as a combinatorial optimization; graphical models[5] are one way to formulate and solve these problems.

A graphical model depicts a joint probability distribution of random variables. While a graphical model's nodes represent random variables, its edges represent their conditional dependencies. How might we use a graphical model for visual embedding? We can define a random variable (node) for each data point, assigning that point to a visual primitive (for example, color, icon, or shape) in the visual-embedding space. Similarly, we can use edges to express pairwise distances as conditional dependencies that we intend to preserve perceptually in the embedding space. Then, we can define the visual-embedding problem as finding the mode of the joint distribution defined by this graphical model, which we can compute using efficient inference algorithms.[5] Later, we give a simple example of how to use graphical models for visual embedding.

Directed and undirected graphical models have great potential for expressing and synthesizing visualizations. We can also extend them to con-

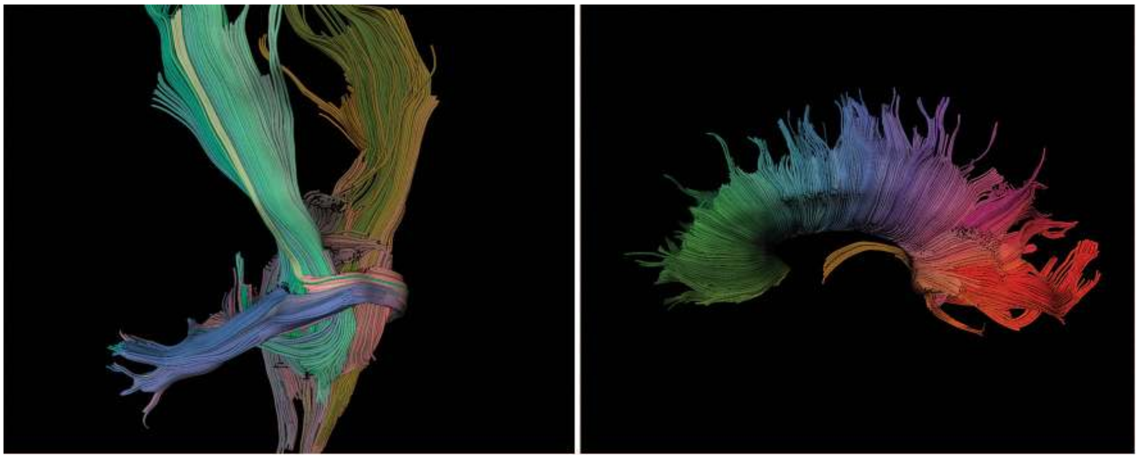

(a) (b)

Figure 3. Coloring neural tracts: (a) the internal capsule and (b) the corpus callosum. We colored them using visual embedding in a perceptually uniform color space. Perceptual variations in color reflect the spatial variations in the tracts.

struct embeddings in continuous visual spaces. Using graphical models also presents an opportunity to model conditional distributions of visual embeddings. We can imagine a scenario in which a visualization tool presents a user with sampled visualizations drawn from a distribution over possible visualizations learned by the model.

Applying Our Model

The following three examples demonstrate our model.

Coloring Neural Tracts

We colored neural-fiber pathways estimated from a diffusion-imaging brain dataset. Given a set of tracts, we first computed distances (or dissimilarities) between pairs of pathways. To do this, we used a simple measure that quantified the similarity of two given neural pathways' trajectories. We then constructed the visualization function by embedding the distances in the CIELAB color space using multidimensional scaling. Figure 3 shows the obtained colorings; perceptual variations in color reflect the spatial variations in the tracts.

Scatterplots with Icons

Here, a toy problem demonstrates embedding in a discrete visual space. We wanted to assign polygonal icons from the discrete polygonal-shape space V_p (see Figure 2) to a given set of 2D points so that the points' spatial proximity was redundantly encoded via the assigned polygons' perceptual proximity. Though simple, this setup is realistic: redundant visual encoding is common in visualization. Alternatively, we could have used icons to convey attributes of other dimensions of the data points.

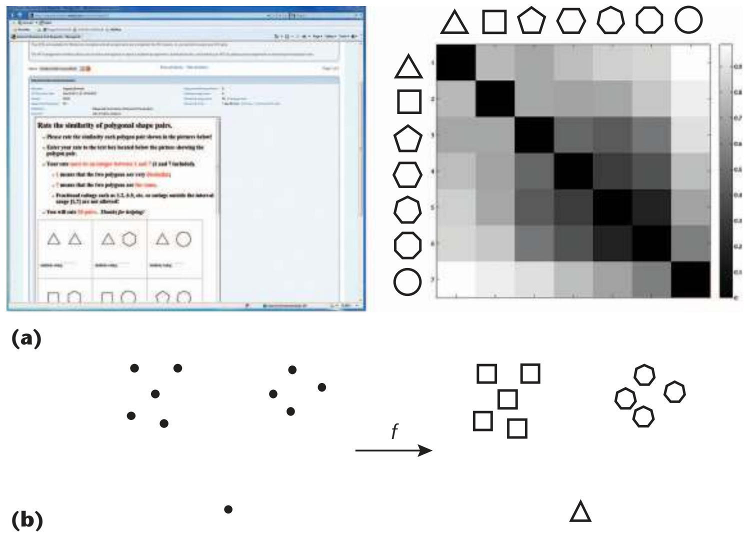
Unlike the coloring example, here we lacked a perceptual model for estimating perceived distance. So, we obtained a crowdsourced estimate of the perceptual distances between the elements of V_p, using Amazon's Mechanical Turk. The study participants saw all possible pairs, including identical ones. We used errant ratings of identical polygon pairs to filter "spammers." After this initial filtering, we normalized each participant's ratings and averaged the ratings across the users. Finally, we normalized the averaged ratings and accumulated the results in a distance matrix. Figure 4a shows the task interface and resulting perceptual-distance matrix.

We then posed the embedding problem as maximum a posteriori estimation in a Markov random field (an undirected graphical model) to find an embedding of a simple 2D point set in V_p. Figure 4b shows the result. The polygonal primitive assignment reflects the data points' clustering, as we desired.

Evaluating Tensor Glyphs

With our model, given suitable data and perceptual metrics, we can assess competing visualization techniques' structure-preserving qualities.

We compared superquadrics and cuboids, two alternative glyphs used in visualizing second-order diffusion tensors (see Figure 5a). We rotated the diagonal tensor $\mathbf{D} = [2.1\ 0\ 0;\ 0\ 2\ 0;\ 0\ 0\ 1]$ around its smallest eigenvector (0, 0, 1) with five incremental degrees. We computed how the tensor value changed as the Euclidean distance between the reference tensor and the rotated tensor changed. We approximated the perceptual change in the corresponding glyph visualizations with the sum of the magnitudes of the optical flow at each pixel in the image domain. We averaged the optical-flow distances over nine viewpoints uniformly sampled on a circumscribed sphere under fixed lighting and rendering conditions.


Rate the similarity of polygonal shape pairs.

- Please rate the similarity each polygon pair shown in the pictures below!
- Enter your rate to the text box located below the picture showing the polygon pair.
- Your rate must be an integer between 1 and 7 (1 and 7 included).
  - 1 means that the two polygons are very dissimilar.
  - 7 means that the two polygons are the same.
  - Fractional ratings such as 1.2, 4.5, etc. or ratings outside the interval range [1,7] are not allowed!
- You will rate 28 pairs. Thanks for helping!

Figure 4. Embedding in a discrete visual space. (a) We used crowdsourcing to estimate perceptual distances for a discrete polygonal-shape space, V_p (see Figure 2). On the left is the task interface for Amazon's Mechanical Turk; on the right is the estimated perceptual-distance matrix. Darker colors indicate closer distances. (b) We embedded the planar data points in V_p. The polygonal-icon assignment reflects the data points' spatial variation and clustering.

The trends in Figure 5b suggest that superquadrics represented the change in the data more faithfully (that is, preserved the structure better) than cuboids. This supports the visualization design choice motivating superquadrics.[6]

Discussion and Research Directions

Embedding spaces needn't be restricted to visual stimuli. They could be any perceptual channel or combinations thereof, such as color, texture, shape, icon, tactile, and audio features. For example, we could, in theory, apply our formulation to construct sonifications for people with visual disabilities.

Our current examples are only a proof of concept, including our approach for estimating perceptual distance through crowdsourcing. Visualizations live in context; crowdsourcing-based estimated perceptual distances can't capture all the perceptual interactions of every context. Also, running large-scale crowdsourcing studies can be difficult. Because we used a small discrete space, we could present every pair of embedding-space points to each study participant. Running a similar experiment with thousands of discrete visual primitives will require larger studies and more sophisticated analysis methods for estimating a distance matrix.[7] Similarly, large-scale embedding can be slow; however, many heuristics, such as restricting pairwise distances to local neighborhoods and sampling, can ameliorate the problem.

On the basis of these challenges and insights derived from our examples, we envision the following research directions.

A Standard Library of Visual Spaces

The visualization community could benefit from a standard library of visual spaces with associated perceptual measures. The library would be a practical resource for constructing useful defaults for visualizations. This goal will require consulting the literature on the interference of perceptual dimensions and running large-scale crowdsourcing studies. For the latter, metric learning might help.[7]

Probabilistic Models of Visualizations

Implementing visual embedding with graphical models provides an opportunity to explore probabilistic models of visualization design spaces. This might prove fruitful because several "optimum" visualizations often exist. Using graphical models can also help express high-level structures in data. Such models might also make it easier to incorporate aesthetic or subjective criteria into automatic visualization generation.

Evaluating Visualizations

To use visual embedding to evaluate visualizations, a primary challenge is to devise and validate appropriate image-space measures (for example, optical flow) to approximate perceptual distances.

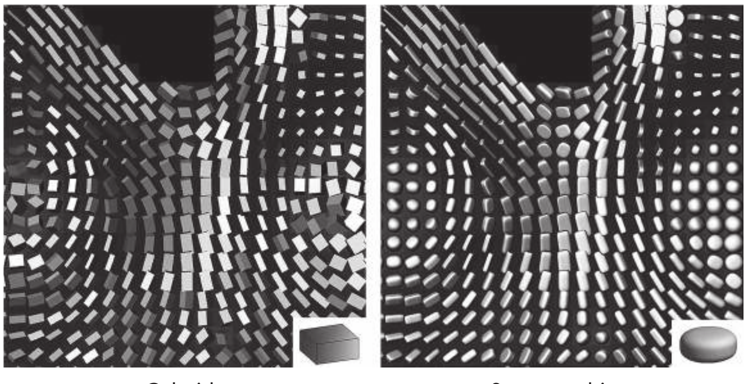
Tools

Finally, we want to develop tools that facilitate construction of visualizations under our model. Two challenges stand out. The first is to develop a visualization language that lets users express and create visual embeddings without implementing an optimization algorithm. This language should integrate libraries of visualization defaults for different data and task domains. It might also benefit from crowd-programming ideas[8] to enable automated support for running crowd-sourced evaluations.

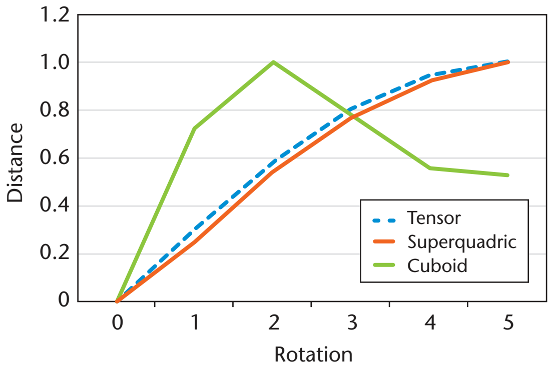
The second challenge is to develop a visualization debugger in the spirit of the tensor glyph example, letting users get runtime feedback about visualization quality. We envision future visualization development environments integrating such languages and debuggers.

References

1. Ç. Demiralp, J.F. Hughes, and D.H. Laidlaw, "Coloring 3D Line Fields Using Boy's Real Projective Plane Immersion," *IEEE Trans. Visualization and Computer Graphics*, vol. 15, no. 6, 2009, pp. 1457–1464.



(a)



(b)

Figure 5. Evaluating tensor glyphs. (a) A superquadric and a cuboid glyph, used for visualizing the same tensor field.[6] The insets represent the diagonal tensor D. (b) Changes in the size of D and its superquadric and cuboid representations with respect to rotations around the tensor's smallest eigenvector. The tensor size and the superquadric glyph's appearance follow a similar trend; the cuboid glyph's appearance differs. This suggests that superquadric glyphs better preserved the structure in the data.

2. J. Heer and M. Bostock, "Crowdsourcing Graphical Perception: Using Mechanical Turk to Assess Visualization Design," *Proc. SIGCHI Conf. Human Factors in Computing Systems* (CHI 10), ACM, 2010, pp. 203–212.
3. W.R. Garner and G.L. Felfoldy, "Integrality of Stimulus Dimensions in Various Types of Information Processing," *Cognitive Psychology*, vol. 1, no. 3, 1970, pp. 225–241.
4. T. Hastie, R. Tibshirani, and J.H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, 2013.
5. D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*, MIT Press, 2009.
6. G. Kindlmann, "Superquadric Tensor Glyphs," *Proc. 6th Joint Eurographics–IEEE TVCG Symp. Visualization* (VISSYM 04), Eurographics Assoc., 2004, pp. 147–154.
7. O. Tamuz et al., "Adaptively Learning the Crowd Kernel," *Proc. Int'l Conf. Machine Learning* (ICML 11), 2011, pp. 673–680.
8. D.W. Barowy et al., "AutoMan: A Platform for Integrating Human-Based and Digital Computation," *Proc. ACM Int'l Conf. Object-Oriented Programming Systems Languages and Applications* (OOPSLA 12), ACM, 2012, pp. 639–654.

Çağatay Demiralp *is a postdoctoral scholar in Stanford University's Visualization Group. Contact him at cagatay@cs.stanford.edu.*

Carlos E. Scheidegger *is a researcher in the Information Visualization group at AT&T Labs Research. Contact him at carlos.scheidegger@research.att.com.*

Gordon L. Kindlmann *is an assistant professor in the University of Chicago's Department of Computer Science. Contact him at glk@uchicago.edu.*

David H. Laidlaw *is a professor in Brown University's Department of Computer Science. Contact him at dhl@cs.brown.edu.*

Jeffrey Heer *is an associate professor in the University of Washington's Department of Computer Science & Engineering. Contact him at jheer@uw.edu.*

Contact department editor Theresa-Marie Rhyne at theresamarierhyne@gmail.com.