Visual Embedding: A Model for Visualization

Çağatay Demiralp Stanford University

Carlos E. Scheidegger AT & T Research

Gordon L. Kindlmann University of Chicago

David H. Laidlaw Brown University

Jeffrey Heer Stanford University

Abstract: We propose *visual embedding* as a model for automatically generating and evaluating visualizations. A visual embedding is a function from data points to a space of visual primitives that measurably preserves structures in the data (domain) within the mapped perceptual space (range). Visual embedding can serve as both a generative and an evaluative model. We demonstrate its use with three examples: coloring of neural tracts, scatter plots with icons, and evaluation of alternative diffusion tensor glyphs. We discuss several techniques for generating visual embedding functions, including probabilistic graphical models for embedding within discrete visual spaces. We also describe two complementary approaches--crowdsourcing and *visual product spaces*--for building visual spaces with associated perceptual distance measures. Finally, we present future research directions for further developing the visual embedding model.

Keywords: Visual embedding, visualization, model, theory, crowdsourcing, perception, perceptual distance, visual space, visual product, probabilistic model.

Visual Embedding: A Model for Visualization

Automating the design of effective visualizations is an unsolved problem. Though numerous guidelines and heuristics have been proposed, a formal framework for design and evaluation remains elusive. Instead, user studies conducted a posteriori remain the primary tool for assessing the effectiveness of visualizations. Using theoretical models presents another, albeit less explored, approach to the problem [15]. 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 have a significant effect on the success of visualizations. Given our current knowledge, the problem of visualization design is underconstrained. As a result, 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 which maps from a domain of data points to a range of visual primitives (Figure 1). We claim a visualization is "good" if the embedded visual elements preserve structures present in the data domain. Our model is motivated by the fact that understanding patterned structures in data is one of the primary goals of visual analysis. The proposed basic framework can be used to generate and evaluate visualizations based on both the underlying data and–through the choice of preserved structure–desired perceptual tasks.

In this article, we first review previous model-based visualization work. We then describe our visual embedding model and provide examples of how it can be applied to visualization design and evaluation.

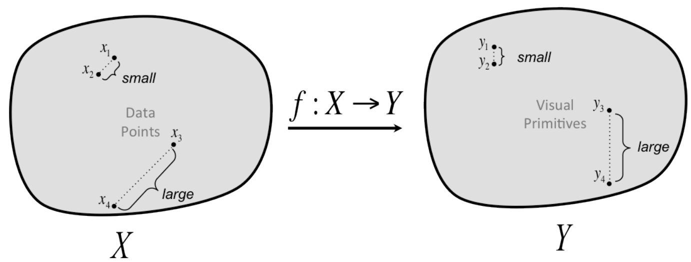


Figure 1: Visual embedding.

Related Work

In prior work, researchers have proposed both general and specific models of visualization. Due to space limitations, we confine our discussion to a small but representative subset.

The seminal work of Mackinlay [11] is one of the most influential systems for the automatic generation of visualizations. Following Bertin's aphorism of *graphics as a language for the eye* [2], Mackinlay formulates visualizations as sentences in a graphical language and argues that good visualizations are those that meet his criteria of *expressiveness* and *effectiveness*. A visualization meets the expressiveness criteria if it faithfully presents the data, without implying false inferences. Effectiveness concerns how accurately the chosen visual encoding variables are decoded by viewers, and is informed by prior studies in graphical perception (e.g., those of Cleveland and McGill [4]). Mackinlay's automatic presentation tool (APT) employs a composition algebra over a basis set of graphical primitives derived from Bertin's encodings to generate alternative visualizations. The system then selects the visualization that best satisfies formal expressiveness and effectiveness criteria.

APT does not explicitly take user tasks or interaction into account. To this end, Roth et al. [13] extend Mackinlay's work with new types of interactive presentations. Similarly, Casner [3] builds on APT by incorporating user tasks to guide visualization generation.

Some of these ideas are now used to support visualization recommendation within Tableau, a commercial visualization tool.

House et al. [8] integrate user preferences in their automatic visualization system. They use genetic algorithms to refine a "population" of visualizations in response to user ratings. In contrast to this empirical approach, Pineo and Ware [12] propose using a computational model of the retina and primary visual cortex to automatically evaluate and optimize visualizations. van Wijk [15] argues for first modeling a perceptual domain (e.g., luminance or shape perception) and then optimizing for some perceptual goal according to that model.

If we were to choose a motto for visual embedding, it would be *visualization as a perceptual painting of structure in data.* In this sense, Mackinlay's expressiveness and effectiveness criteria together loosely correspond to our model's perceptual structure preservation criterion. Visual embedding can also be seen a reusable template within van Wijk's discussion on perceptually optimal visualizations.

Visual Embedding

Mathematically, we can model a visualization as a function from data points to a space of visual primitives. We propose that a "good" visualization is *a function that preserves structures in the data (domain) within the embedded perceptual space (range).* We call a visualization function that meets this criterion a *visual embedding* of the data points. Our proposed model is a generalization of earlier work on structure-preserving colorings [5].

Representing Structures in Data

According to its dictionary definition, structure is "the arrangement of and relations between the parts or elements of something complex." How, then, can we express structures in data? The problem is that a user might not explicitly know about important structures in the data, let alone express or quantify them. On the other hand, one often can hypothesize some notion of distance between data points. Using pairwise distances is one simple and general way of implicitly expressing structures in spaces. For example, if a function transforms a two- or three-dimensional Euclidean space while preserving pairwise Euclidean distances, the shape and size of objects in the space will stay the same. Similarly, if there were a function from a sphere to a plane that preserved all pairwise distances on the sphere, we would have world maps without distortion (i.e., angles and areas would be simultaneously preserved). Structure can be operationalized in terms of these atomic pairwise relations; in this context, it is the visualization function's job to picture what these pairwise relations amount to.

Distance in data space ideally should reflect a user's understanding of the similarity between data points as it relates to her current task. This allows the user to hint at the type of structures she is interested in seeing. For instance, if the user is interested in symmetries then she should provide a measure that quantifies these relationships. In fact, structural criteria such as symmetry and continuity are often used as design choices in creating visualizations. Distance in the visual space, on the other hand, 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 must be capable of depicting that structure. One advantage of using pairwise distances is that application to visual spaces is conceptually straightforward. They can be encoded as perceptual differences of color, shape, texture, spatial distance, size, etc. Our following discussion assumes the use of pairwise distances as our method of expressing structure in the data domain.

Estimating Perceptual Distances with Crowdsourcing

To assess structural preservation, we require perceptual distance measures for a given visual embedding space. Except for a few perceptually uniform color spaces, however, we don't have these measures for most visual spaces. Online crowdsourcing can help us estimate perceptual distances in these cases [7]

A visual space is said to be perceptually uniform if a perturbation to any element in the visual space 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, whereas RGB and CIEXYZ are not. 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 may cause a disproportionate change in perceived colors.

Crowdsourcing is one means of collecting large and diverse perceptual data samples. For example, Heer and Bostock [7] successfully replicated prior graphical perception results using crowdsourced experiments on Amazon's Mechanical Turk. Note that 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 within an example below (Section 4.2) by crowdsourcing the perceptual distances for a simple discrete space of polygons (V_p in Figure 2).

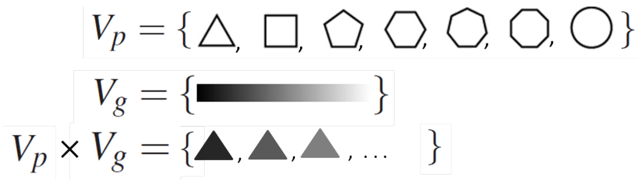


Figure 2: Example of a visual product space.

New Visual Spaces from Old: Visual Product Spaces

Formulating visualizations as structure-preserving functions raises the possibility of transferring other related concepts from mathematics. 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 new space a *visual product space* (Figure 2).

As a general rule, the product of two perceptually uniform visual spaces will not itself be uniform. On the other hand, we know that when we have two topological spaces endowed with metrics, constructing a metric for the product space is straightforward. One challenge is to discover if there are cases where an analogous procedure exists for constructing *visual product metrics*. This issue strongly resonates with research on interactions between perceptual dimensions (e.g., integral vs. separable visual encodings). Looking into separable cases reported in the literature may be a promising starting point.

Constructing the Visualization Function

Construction of a "good" visualization function under our proposed model is fundamentally an optimization problem. The nature of embedding spaces often determines the techniques available to us. Embedding spaces can be Euclidean (e.g., most color spaces, including RGB, CIELAB, CIELUV, etc.), continuous but non-Euclidean (e.g., parametric shape spaces and texture spaces), and discrete (e.g., finite sets of icons, shapes, glyphs and fonts). Principal component analysis (PCA), multidimensional scaling (MDS), isometric feature mapping (ISOMAP), and local linear embedding (LLE) are just few examples of the many techniques used to embed a domain in Euclidean space [6].

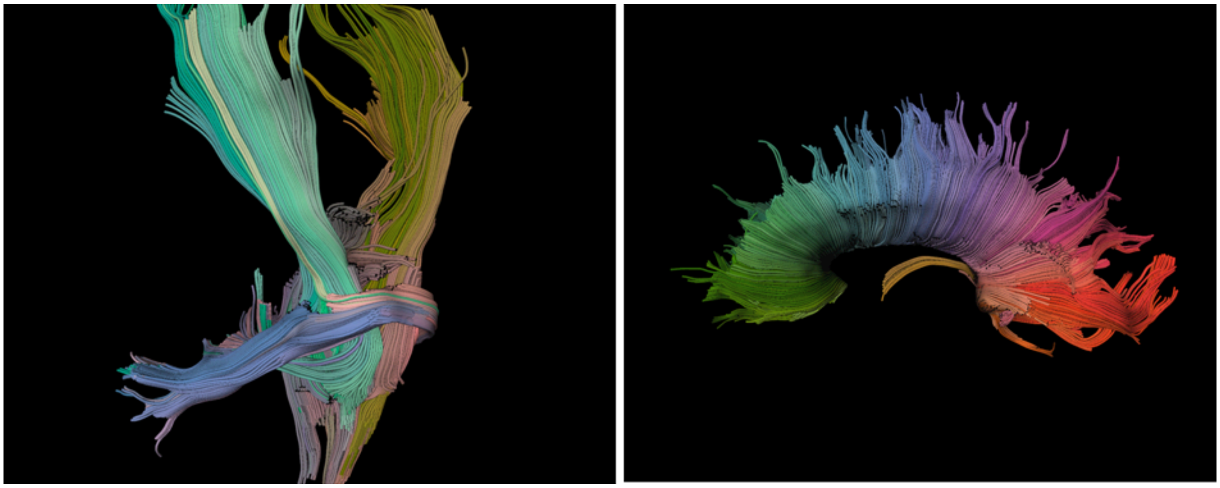
Although embedding in the Euclidean space is computationally well studied, embedding in non-Euclidean spaces (continuous or discrete) is not. The latter problem can be formulated as a combinatorial optimization; graphical models [10] are one way to formulate and solve these problems.

A graphical model depicts a joint probability distribution of random variables. Nodes and edges of a graphical model represent random variables and their conditional dependencies, respectively. How might we use a graphical model for visual embedding? Consider: we can define a random variable (node) for each data point, assigning the data point to a visual primitive (e.g., color, icon, shape, etc.) 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 the visual embedding problem can be defined as finding the mode of the joint distribution defined by this graphical model, which can be computed using efficient inference algorithms [10]. In Section 4.2, we give a simple example of how graphical models can be used for visual embedding.

Directed and undirected graphical models have great potential for expressing and synthesizing visualizations. They can also be extended to construct embeddings in continuous visual spaces. Using graphical models also opens up an opportunity to model conditional distributions of visual embeddings. One 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.

Examples

We give three examples to demonstrate our model.



(a) (b)

Figure 3: Neural tracts, (a) the internal capsule and (b) the corpus callosum, colored using visual embedding in a perceptually uniform color space.

Coloring Neural Tracts

Our first example concerns the coloring of neural fiber pathways estimated from a diffusion imaging brain dataset. Given a set of tracts, we first compute distances (or dissimilarities) between pairs of pathways, using a simple measure that quantifies the similarity of trajectories that two given neural pathways follow. We then construct the visualization function by embedding the distances in CIELAB color space using MDS. Figure 3 shows the obtained colorings. Notice that spatial variations in tracts are reflected by perceptual variations in color.

Scatter Plots with Icons

The second example demonstrates embedding in a discrete visual space using a toy problem. We would like to assign polygonal icons from V_p (Figure 2) to a given set of 2D points so that the spatial proximity of the points is redundantly encoded via the perceptual proximity of the assigned polygons. Though simple, this setup is not unrealistic; redundant visual encoding is common in visualization. Alternatively we could use icons to convey attributes of other dimensions of the data points.

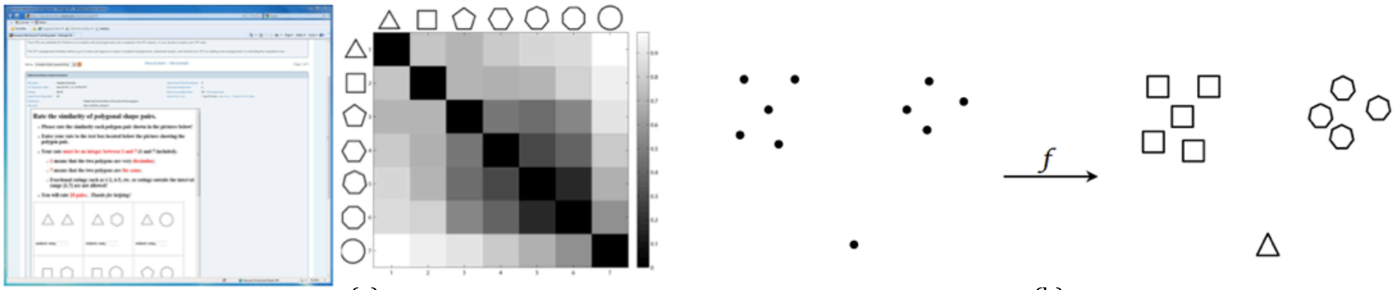
Unlike the previous coloring example, here we lack a perceptual model for estimating perceived distance. Consequently, we first obtain a crowdsourced estimate of the perceptual distances between the elements of V_p using Amazon's Mechanical Turk service. Users were shown all possible pairs, including identical ones. We use errant ratings of identical polygon pairs to filter "spammers." After this initial filtering, we normalize the ratings within each user and then average ratings across users. Finally, we normalize the averaged ratings and accumulate the results in a distance matrix. Figure 4a shows the task interface and resulting perceptual distance matrix.

We then pose 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 clustering of the data points, as desired.

Evaluating Tensor Glyphs

Our proposed model can also be used to evaluate existing visualizations; given suitable data and perceptual metrics, we can assess the structure-preserving qualities of competing visualization techniques.

To demonstrate this point, we compare superquadrics and cuboids, two alternative glyphs used in visualizing second order diffusion tensors (Figure 5a). We take the diagonal tensor **D**=[2.1 0 0; 0 2 0; 0 0 1] and rotate it around its smallest eigenvector (0,0,1) with five incremental degrees, while computing the change in the tensor value with the Euclidean distance between the reference tensor and the rotated tensor. We approximate 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. Note that we average the optical flow distances over 9 different viewpoints uniformly sampled on a circumscribed sphere under fixed lighting and rendering conditions.



(a) (b)

Figure 4: (a) Perceptual distances estimated for a discrete polygonal shape space, V_p, using crowdsourcing: (left) The task interface at Amazon Mechanical Turk, (right) the estimated perceptual distance matrix. Darker colors indicate closer distances. (b) Planar data points are embedded in V_p. Note that the polygonal icon assignment reflects the spatial variation and clustering of the data points.

The trends in Figure 5b suggest that superquadrics represent the change in the data more faithfully (i.e., better preserve the structure) than cuboids. This supports the visualization design choice motivating superquadrics [9].

Conclusions

Good visualizations should facilitate exploration and understanding of patterned structures in data. Motivated by this, we proposed a visualization model based on structure-preserving functions into visual primitive spaces and discussed some tools to construct these functions. Although we focus on visualization, embedding spaces need not be restricted to visual stimuli: any perceptual channel or combinations thereof – such as color, texture, shape, icon, tactile, and audio features – might play the role of the embedding space. For example, one could in theory apply our formulation to construct sonifications for people with visual disabilities.

Our current examples are intended only as a proof-of-concept, including our approach for perceptual distance estimation via crowdsourcing. Visualizations live in context; estimated perceptual distances with crowdsourcing cannot capture all the perceptual interactions of every context. Also, running large-scale crowdsourcing studies can be difficult. In the example here, we have a small discrete space and thus it was feasible to present every pair of embedding space points to each of the study participants. Running a similar experiment with thousands of discrete visual primitives will require larger studies and more sophisticated analysis methods for estimating a distance matrix (e.g., [14]). Similarly, large-scale embedding can be slow, although there are many heuristics such as restriction of pair-wise distances to local neighborhoods and sampling that can ameliorate the problem.

Based on these challenges and insights derived from our examples, we envision several research directions going forward:

1. **Standard library of visual spaces:** We would like to see a "standard" library of visual spaces with associated perceptual measures made available to the larger visualization community. Such a library would be a practical resource for constructing useful defaults for visualizations. This effort will require consulting the perception literature on the interference of perceptual dimensions as well as running large-scale crowdsourcing studies. It is possible that metric learning might help for the latter [14].
2. **Probabilistic models of visualizations:** Implementation of visual embedding with graphical models provides an opportunity for exploring probabilistic models of visualization design spaces. This may prove to be fruitful because there are often several "optimum" visualizations. Using graphical models can also help in expressing high-level structures in data. Such models may also make it easier to incorporate aesthetic or subjective criteria into automatic visualization generation processes.
3. **Evaluating visualizations:** Given alternative visual encodings for the same type of data and task, it is natural to ask: which encoding is better? As our tensor glyph example demonstrates, visual embedding can provide a framework for evaluating visualizations. One of the challenges in this direction is to devise and validate appropriate image space measures (e.g., optical flow) to approximate perceptual distances.
4. **Tools:** Finally, we would like to develop tools that facilitate the construction of visualizations under our proposed model. Two challenges stand out. The first is to develop a visualization language allowing users to express and create visual embeddings without having to implement any optimization algorithm. This language should integrate libraries of visualization defaults for different data and task domains. It may also benefit from crowd programming ideas (e.g., [1]) to enable "coding" of user validation in visualization programming. The second challenge is to develop a visualization debugger in the spirit of the example (Section 4.3), allowing users to get "runtime" feedback about the quality of their visualizations. We envision visualization development environments in the future integrating such languages and debuggers.

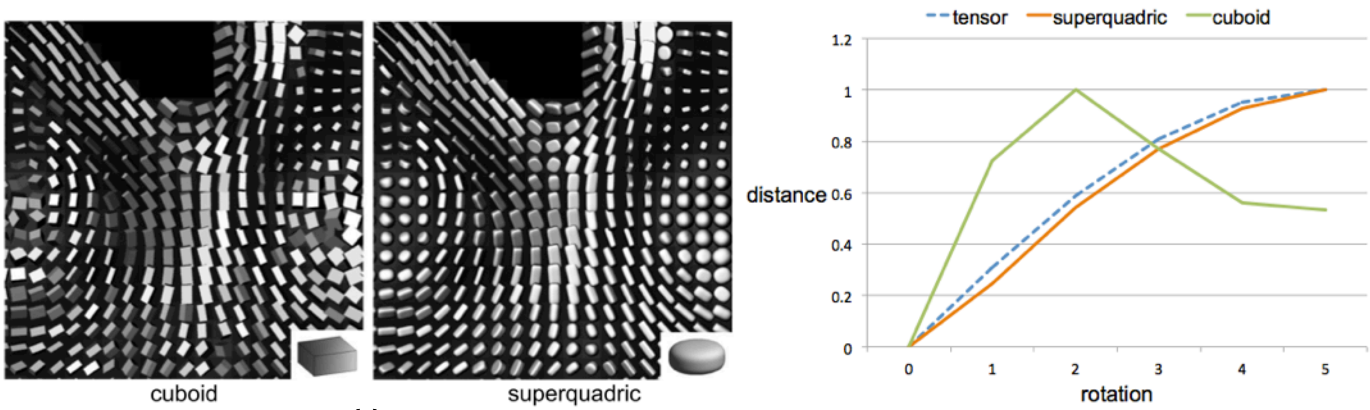


Figure 5: (a) Two different tensor glyphs, cuboid and superquadric, used for visualizing the same tensor field (adapted [9]). Respective inset glyphs represent the diagonal tensor **D**. (b) Changes in the size of the diagonal tensor **D** and its superquadric and cuboid representations with respect to rotations around the tensor's smallest eigenvector are shown. Note that the tensor size and the superquadric glyph appearance follow a similar trend while the cuboid glyph appearance has a trend different from the two. This suggests that superquadric glyphs better preserve the structure in the data in this experiment.

References

[1] D. W. Barowy, C. Curtsinger, E. D. Berger, and A. McGregor. Automan: a platform for integrating human-based and digital computation. In *Proceedings of OOPSLA*, pages 639–654, 2012.

[2] J. Bertin. *Semiology of graphics*. University of Wisconsin Press, 1983.

[3] S. M. Casner. Task-analytic approach to the automated design of graphic presentations. *ACM Trans. Graph.*, 10(2):111–151, Apr. 1991.

[4] W. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association*, pages 531–554, 1984.

[5] Ç. Demiralp, J. F. Hughes, and D. H. Laidlaw. Coloring 3D line fields using Boy's real projective plane immersion. *IEEE Trans. Vis. Comput. Graphics (Proc. IEEE Visualization)*, 15(6):1457–1464, 2009.

[6] T. Hastie, R. Tibshirani, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2013.

[7] J. Heer and M. Bostock. Crowdsourcing graphical perception: using mechanical turk to assess visualization design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 203–212, 2010.

[8] D. H. House, A. Bair, and C. Ware. An approach to the perceptual optimization of complex visualizations. *IEEE Trans. Vis. Comput. Graphics*, 12(4):509–521, 2006.

[9] G. Kindlmann. Superquadric tensor glyphs. In *Proceedings of IEEE TVCG/EG Symposium on Visualization*, pages 147–154, 2004.

[10] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[11] J. D. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Trans. Graph.*, 5(2):110–141, 1986.

[12] D. Pineo and C. Ware. Data visualization optimization via computational modeling of perception. *IEEE Transactions on Visualization and Computer Graphics*, 18(2):309–320, 2012.

[13] S. F. Roth, J. Kolojejchick, J. Mattis, and J. Goldstein. Interactive graphic design using automatic presentation knowledge. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, pages 112–117, New York, NY, USA, 1994. ACM.

[14] O. Tamuz, C. Liu, S. Belongie, O. Shamir, and A. Kalai. Adaptively learning the crowd kernel. In *Proc. International Conference on Machine Learning*, pages 673–680, 2011.

[15] J. J. van Wijk. Model-based visualization: Computing perceptually optimal visualizations. In *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*. Springer, 2009.