Proximity Visualization via Common Fate Luminance Changes

Lior Wolf Chen Goldberg Yehezkel Yeshurun
The Blavatnik School of Computer Science
Tel Aviv University

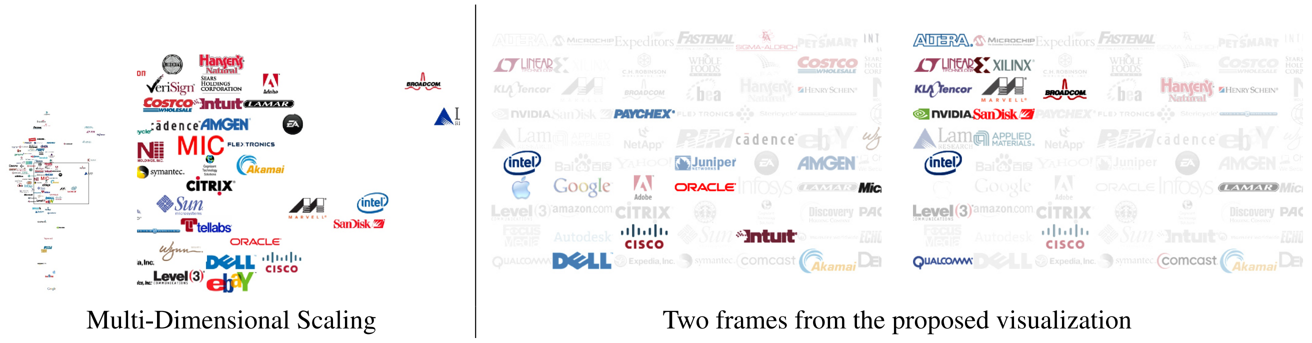
Multi-Dimensional Scaling
Two frames from the proposed visualization

Figure 1: A comparison between two visualizations of the NASDAQ-100 list. Proximities between stocks were computed based on their 2007 daily prices. (left) The conventional 2D embedding visualization, modified slightly to eliminate occlusions. One panel is the entire space and the other is a zoomed in view. (right) two frames out of the proposed animated visualization. The proposed design depicts several prominent connections that are not depicted by the conventional visualization. For example, SanDisk and Intel's prices were correlated in 2007, as were the prices of Intel and Oracle. However, the prices of Oracle and SanDisk were much less correlated. This can be easily observed in the proposed design, but not in the conventional visualization. The interactive animation is available in the accompanying supplementary material `http://www.cs.tau.ac.il/~wolf/demos/commonfate.zip`.

Abstract

We introduce a new design for visualizing high dimensional data. Points in high dimensional space are often illustrated through 2D or 3D embeddings, which are often cluttered and may-be unaccessible to non-scientific audience. In our design, instead of representing data points as 2D/3D vectors, each data point is represented by a smoothly varying function of time. These smooth functions are used to control the luminance of iconic representations of the data points, where the value of each such function determines the luminance of one icon that is associated with the underlying data point.

The resulting display has a high capacity and is intuitively clear to the viewer. Perceptually, it is based on a fundamental psychophysical principle, namely the Gestalt law of Common Fate. Algorithmically, we present two methods designed to meet the demands of the visualization: a new non-negative matrix factorization method, and a partition-based grid embedding technique.

1 Introduction

Perhaps the earliest record of a dynamic screen-like display is the one of the biblical Chosen (Exodus 28) that was a sacred garment in which 12 precious stones were set in a grid. The Hebrew high priest would communicate with the divine by interpreting the patterns of glowing gems. We propose a visualization design of a similar structure. Icons are placed on a grid, and their individual luminance values change over time. In the proposed design, the patterns of luminance-change convey proximities between the objects represented by the icons. In particular, icons that are lit and unlit together in a synchronized fashion are perceived as related. Thus, correlative information is conveyed to the user intuitively and effortlessly.

Problem formulation In this work we visualize proximity information between tens or hundreds of data points. The proximity can arise from Euclidean distances in the case of vectorial data, from the analysis of a given graph structure, or based on other means of measuring correlations. In all of these cases, the input to our method is the same – the number of data points n, iconic representations of the n points, and a symmetric $n \times n$ affinity matrix A containing positive values. A high affinity value between points i and j (a large $A(i,j)$ entry) indicates that data points i and j are similar. Low values indicate distant, dissimilar points.

In this work we concentrate on visualizations for the non-professional audience, and we adhere to several design requirements. First the data points are required to be effortlessly identifiable. Therefore, we are interested in visualizations that display icons (e.g., text, logos or thumbnails) and not glyphs (e.g., dots), as in most scatter-plots. Second, we wish for the display to be intuitively clear to the user, with no need for elaborate user training. This limits us to visual encoding which is cognitively natural. Third, we wish to have a clean uncluttered display that is visually attractive.

2 Related work

Many times the proximity we wish to visualize is estimated based on high-dimensional data. The visualization of such data is often treated as a Dimensionality-Reduction problem, where each data point i is represented by a vector $x_i \in \Re^m$, for an arbitrary m. The vectors x_i, $1 \le i \le n$ are chosen such that they are correlated, up to some error that the method minimizes, according to the pairwise correlations in A. One such method is multidimensional scaling (MDS) [Shepard 1962], which has been used extensively for visualization purposes (e.g., the ThemeScape system [Wise et al. 1995]). In such applications, one recovers $m = 2$ or $m = 3$ dimensions, which are used to determine the screen-location of each data glyph or icon.

Other Dimensionality Reduction methods have also been adapted for visualization purposes. Recently, Venna and Samuel [Venna and Kaski 2007] have compared classical MDS (closely related to PCA), ISOMAP [Balasubramanian et al. 2002], Curvilinear Component Analysis [Pierre Demartines and Jeanny Herault 1997], Locally Linear Embedding [Roweis and Saul 2000] and Laplacian Eigenmaps [Belkin and Niyogi 2001]. The results show that all embedding methods face difficulties when reducing the dimensionality to as low as two dimensions, and that Curvilinear Component Analysis (CCA) seems to behave better than the other methods. CCA is a weighted version of MDS, which takes into account only distances that are smaller than a threshold.

The capacity of Dimensionality Reduction based scatter plots is limited by the low-dimensionality. Potentially, more dimensions can be added by color-coding or other means, but such combinations seem to be cognitively taxing for the viewers [Treisman and Gormican 1988]. Also, if we choose to display icons and not dots, the icons would many times overlap, creating a cluttered inapprehensive and unattractive visualization. This can be partly avoided at the expense of accuracy by adjusting the locations of the icons.

An alternative methods for the display of multi-dimensional data is the Parallel Coordinates method [Inselberg and Dimsdale 1990]. In this method each coordinate is typically associated with one vertical line, and each data point is drawn as a polyline with vertices on the parallel axes that are positioned according to the coordinates of the point. In Parallel coordinates the variables (less than 10 is most examples) are identifiable and the individual data points are typically not identifiable without interaction. In contrast, our method displays the affinity between the data points and does not assume the existence of an underlying vector representation; Individual data points are represented by large identifiable icons. Parallel coordinates tend to become cluttered when displaying large datasets, and the power of the method is mostly in understanding global patterns. The method has been extended to reduce clutter [Ellis and Dix 2006; Fua et al. 1999], and to create a focus+context display [Novotny and Hauser 2006].

Another successful Focus+context method for displaying large tabular datasets is the Table Lens technique [Rao and Card 1994]. In this method the sizes of the rows and columns of a tabular display change dynamically to allow focus on individual data points (rows) and variables (columns). The Table Lens method is able to allow the exploration of thousands of data points and tens of variables. The problem it solves, however, is quite different from ours. We focus on the display of affinities that might be handed directly, arise from graph structures, from very sparse matrices, or from continuous time data. In all of these cases, a tabular representation may be inappropriate.

An alternative method for displaying affinity information directly (and not tabular information), is the Generalized Association Plots (GAP) method. In this method the correlation matrix of the data is iteratively updated to reflect the correlations between its columns. This iterative process produces correlation matrices of gradually decreasing ranks that can be used to generate a hierarchical clustering of the data points. In addition, it produces a series of elliptical 2D embeddings. Each such 2D embedding induces a 1D embedding of the data points (ordering of the rows and columns of the correlation matrix). This form of visualization can handle extremely large datasets, and does not focus on identifying individual data points.

As a dynamic technique, GAP can be animated and each data point can move along a path governed by its location in the series of elliptical embeddings. Since the correlation matrix is iteratively simplified, the clustering of the data becomes more and more clear as the process progresses. On the other hand, information regarding proximity to points in other clusters, that is not depicted in the 2D embedding of the first iteration, will probably not emerge. In this limited sense the GAP method suffers from the capacity problem of other 2D methods.

Use of animation for visualization Animation is extremely suitable for expressing causality. The work of Michotte [Michotte 1963] shows that with appropriate timing of events, a causal relationship is perceived. It can also express communication by animating messages moving from message sources to message destination objects [Stasko 1990]. Most naturally, animation can be used to visualize dynamic and serial processes. The classical movie "Sorting Out Sorting" uses animation to illustrate a number of computer sorting algorithms [Baecker 1998].

Recently, Heer and Robertson [Heer and Robertson 2007] have studied the use of animated transitions between statistical graphs. The transitions were carefully designed to match the shift between the various aspects of the displayed data, and were shown to increase its apprehension.

More related to our work, Limoges et al. [Limoges et al. 1989] evaluated the use of motion in visualizing properties of data points. They enhanced a conventional scatter plot representation by allowing the points to oscillate sinusoidally, either horizontally or vertically about a center point. The type and amount of motion (frequency, phase and amplitude) were used as visualization properties. The results show that data mapped to phase is perceived as effectively as more conventional properties, such as point size or gray value.

Ware and Bobrow [Ware and Bobrow 2005] show that motion based highlighting is effective for visualizing local neighborhoods in graphs that contain many nodes. Moreover, they showed that motion highlighting and static highlighting can be combined to allow efficient visualization of set intersections. Can an entire graph be visualized by highlighting the neighborhood of each node sequentially? In Section 5 we report from our experience that such a visualization is hard to apprehend.

Using synchronized motion patterns is one of the many manifestations of the well known Gestalt principle of Common Fate. According to this principle, stimulus elements are likely to be perceived as a unit if they move together. However, there is some psychophysical evidence substantiating the view that luminance-based common fate is also very prominent [Sekuler and Bennett 2001]. In our work, the position of points does not change in order to communicate correlations. Instead, information is encoded by synchronization of light patterns, which is related to the concept of phase. Note, however, that the capacity of our method is much higher since the patterns we display evolve over time.

3 The proposed design

In our visualization a fixed grid of icons changes over time by associating a dynamic luminance value to each individual icon. There are two computational questions that need addressing. The first is how to determine the luminance of each icon as a function of time. The second question is how to distribute the icons in space.

The two questions are related. In the datasets that we consider there are more data points than the number of icons that the display window can accommodate at once. This requires us to pan the display over the grid as the animation progresses. Therefore, ideally, all icons that are lit at once are contained in a region of the grid which is small enough to display.

3.1 Embedding in time

In our design, every data point $i = 1..n$ is represented by a positive function $f_i(t)$ that captures its luminance over time. The correlations between these functions should adhere to the given affinity matrix A, which suggests that we minimize a discrepancy score such as

$$\sum_{i,j}(A(i,j) - f_i \cdot f_j)^2$$

where

$$f \cdot g := \int_t f(t)g(t)dt$$

In addition, to create a smooth animation, the luminance should change gradually. Therefore, we also minimize

$$\sum_i ||\partial f_i/\partial t||^2 := \sum_i \int_t (\partial f_i/\partial t)^2(t)dt$$

In the optimization process, we discretize the functions f_i and use vectors $v_i \in \Re^m$ instead. The continuous derivative operator becomes a matrix operator, given by the $m \times m$ matrix H that has a main diagonal of -1, all values of the adjacent upper diagonal equal 1, and the rest of the elements are zero. Furthermore, in order to play the animation in a loop, we modify the first element in the last row of H to be 1. This alteration connects the last frame to the first.

Let V be the matrix whose columns are v_1,v_2,...,v_n. Our optimization procedure minimizes the following energy function for some given parameter λ

$$E = ||A - V^\top V||_F^2 + \lambda ||HV||_F^2$$

Replacing the Frobenius norms with the equivalent traces and eliminating constants, we obtain:

$$E = tr(V^\top V V^\top V) - 2tr(V^\top VA) + \lambda tr(V^\top H^\top HV)$$

Differentiating:

$$\frac{\partial E}{\partial V} = 4VV^\top V - 4VA + 2\lambda H^\top HV$$

We optimize in an iterative manner until convergence, using the gradient projection method (e.g., [Nocedal and Wright 1999]). We initialize a random matrix V^0 and update it by the following rule:

$$V^{s+1} = \max(V^s - \eta \frac{\partial E}{\partial V}, 0) \quad ,$$

where η is the learning rate, and the maximization is performed element by element.

Note that without the smoothness term, our computational problem would have been similar to the problem of (Symmetric) Nonnegative Matrix Factorization (NMF) [Lee and Seung 2001; Li et al. 2007] . NMF solutions are known to be sparse. In our case, this means that most icons would be unlit most of the time, which is desirable since it allows the viewer to focus on a handful of lit icons at every frame.

3.2 Motion and embedding in space

The proposed visualization-technique is designed for hundreds of data-points. However, at any given time only 25-100 can be displayed, depending on the screen size. We solve this by panning a display window such that it is centered at each frame on the center of mass of all lit icons. In other words, let x_i, y_i ($i = 1..n$) be the grid locations of the icons associated with the n data points. The center of the window in time t is placed in coordinates $\bar{x}(t)$, and $\bar{y}(t)$, where

$$\bar{x}(t) = \frac{1}{\sum_{i=1}^n v_i(t)} \sum_{i=1}^n v_i(t) x_i$$

and similarly for $\bar{y}(t)$.

As mentioned above, the main requirement for the spatial grid embedding is that icons that are lit together would fit together in a region small enough to display at once. The computational problem is as follows: given a threshold θ, embed a set of vectors v_i, $i = 1..n$ in a 2D grid, such that for every pair i and j for which there exists a frame index t such that both $v_i(t) \geq \theta$ and $v_j(t) \geq \theta$, the maximal horizontal and vertical distances between the embedding of i and j is not larger than a specified amount.

We are not guaranteed that such an embedding exists. Furthermore, finding such an embedding is an NP-complete problem (by reduction to the Maximal Common Subgraph problem, details omitted). In a sense, this grid embedding challenge is a consequence of the unsuitability of 2D embeddings for the display of high-dimensional data. However, the constraints here are more relaxed since nearby pairs of points that have low affinities are still separated by the luminance modulation.

To heuristically solve the embedding problem we employ the following method, which is based on a quadtree partitioning of the space [Samet 1984]. First, we compute an $n \times n$ co-occurrence matrix O, in which O(i,j) is one if there is a frame index t such that both $v_i(t)$ and $v_j(t)$ are larger than the threshold $\theta = 0.1$, zero otherwise. Next, we embed all n points in $2D$ by applying MDS to the matrix O and obtaining coordinates $(\tilde{x_i}, \tilde{y_i})$, $i = 1..n$. We then partition the grid horizontally to two sub-grids, followed by a vertical partitions of each sub-grid. At each partitioning stage, we divide the points between the grid partitions. The splitting process repeats recursively until all the partitions contain a single point.

At each stage of the recursion, there is a rectangular $s_x \times s_y$ sub-grid in which we need to embed $s_x s_y$ icons. Assume that the next partitioning operation is along the horizontal axis, and that s_x is an even number. We sort all points that are assigned to the sub-grid by their $\tilde{x_i}$ value, and assign the $s_x s_y/2$ points with the lowest $\tilde{x_i}$ values to the left $s_x/2 \times s_y$ partition. The rest are assigned to the right partition of the same size. If s_x is an odd number, we compute the minimum, median, and maximal values of the x-axis embedding $\tilde{x_i}$ of all points that are associated with the grid. We then check if this median is to the left or to the right of the middle of the section between the min value and the max value. The middle grid column is assigned to the left or right half-grid accordingly.

3.3 Interactivity

The proposed design enables one to select data-points by clicking on the corresponding icons. Once there are selected points, the only frames that are shown are frames t for which for every selected data-point i, the value of $v_i(t)$ is larger than a threshold $\theta = 0.1$.

Some data points, which do not share strong links with the other points, may not be lit in all of the computed frames. Similarly, in the multiple selection case, the set of frames that are lit for all selected data-points may be empty. In such cases the selection frame turns from green to red and the selected icons are all lit up. In the case of an empty multiple selection, a text message appears to explain the situation.

Once a point is selected, a "deselect all" icon appears. Mouse-clicking on this icon returns the display to the normal display mode starting from the last frame that is shown in the selection mode. Note that other forms of selection are also possible based on the application. For example, one might wish to display all frames for which there is at least one icon lit out of the selected icons.

3.4 Limitations

The information-capacity of our visualization is not limited to 2D or 3D, but instead is limited by the length of the animation sequence and by its speed (how fast the luminance of the icons change). Long sequences convey more information, as do faster changing ones. Put differently, their effective dimension as an embedding space is higher. The utilization of the higher dimensionality is bounded, of course, by the memory and attentiveness of the viewer.

Our visualization method is limited in that when no user interaction is present, the user takes a passive observing role following a predetermined animation. In contrast, a 2D-projection type visualization may allow the viewers to freely foveate (interactive zooming may be required for large datasets). Depending on the application, the user may prefer to search around for the desired information, or to *effortlessly* observe an animation sequence.

	Parameter	value
m	Target dimension (Sec. 3.1)	n
	Video length (in frames)	$30m$
λ	Smoothness weight (Sec. 3.1)	3
η	Learning rate (Sec. 3.1)	0.01
ε	Convergence Threshold (Sec. 3.1)	10^{-5}
	Max number of iterations (Sec. 3.1)	1000
θ	Visibility threshold (Sec. 3.2)	0.1

Table 1: The set of fixed parameters used throughout the reported experiments and in the online facebook application.

Tversky et al. [Tversky et al. 2002] pointed to possible pitfalls of certain kinds of animations, analyzing animated visualization based on *congruence* between internal and external representations and on the *apprehensibility* of the resulting animation. In congruent visualizations the format and the contents of the graphics match those of the displayed concept; Apprehensive visualizations are those in which the graphical information is perceived appropriately and accurately enough.

We believe that although we use animation not to depict time based dynamics, our animation is congruent since what appears in our mind together is thought of as linked. As for apprehensibility, this property lacks mostly in high paced animations, where transitions are rapid. The smoothness in time of our underlying representation eliminates some of this danger. While the user cannot be expected to extract all the information that exists in the visualization, the capacity of the proposed visualization is much larger than the one of 2D embedding even after considering loses arising from partial apprehension.

4 Results

We implemented our visualization design using the ActionScript programming language, which is used to create Adobe Flash animation. The computational part is performed in Matlab and saved as XML files. This separation between computation and visualization is natural for our design and implies its suitability in the client-server setting. In fact, the facebook visualization below is already implemented as a "facebook application" that was shown on actual facebook user-pages.

We use the design to visualize four different datasets, described below. In each dataset, we embed the data in an m dimensional space of smooth vectors as described in Section 3.1, where m is set to be the number of data points. In the actual visualization we interpolate this data to $30m$ frames by employing linear interpolation between every two consecutive coordinates. We then display the resulting animation at 30 frames per second.

The parameters used to visualize all four datasets are summarized in Table 1. These parameters seem to be appropriate for a wide range of dataset sizes. For example, we have had users for our facebook visualization with as few as 5 friends and users with as many as 800.

For the graph based data (facebook) we use the adjacency graph as the affinity matrix. In the case of vectorial data (all other datasets), the data points are normalized to have a norm of one, and the proximity is computed by considering the dot product between every two data points. Many other alternatives exist in the *kernel methods* literature (e.g., [von Luxburg 2007]). For example a Guassian kernel can be employed which computes affinities as $A(i,j) = \exp(-||x_i - x_j||^2/2\sigma^2)$, and the Matrix Laplacian can be used instead of the original affinity matrix. Our choice to employ the linear dot-product kernel was primarily motivated by our wish to minimize the number of parameters we need to tune during the data representation stage, thus validating the robustness of our method.

Netflix DVD Ratings A dataset containing DVD ratings by over 480,000 users was downloaded from the Netflix Prize website (`www.netflixprize.com`). We selected the subset of the 512 DVDs that were released in 2005, which is the last year for which DVD ratings are included in the dataset. Similarities were computed based on the user-specified ratings only, i.e., from long vectors where each user corresponds to one coordinate. Reviews are specified on a scale of 1 to 5. A rating of 0 is used to indicated no review for a specific DVD by a specific viewer.

The resulting visualization is depicted in Figure 2 and is presented in full (as do the other visualizations) in the accompanying supplementary material available at `http://www.cs.tau.ac.il/~wolf/demos/commonfate.zip`. As can be seen, it clearly displays proximities between related DVDs, and by watching and interacting with the visualization the perception of genres and subgenres emerges. In contrast, MDS, due to its low capacity, places very distinct genres nearby. As a result, one cannot judge reliably whether two DVDs are similar by observing their spatial proximity.

Movie Stars We selected 50 actors and actresses and represented each one by a binary vector which depicts the movies he/she participated in according to The Internet Movie Database (IMDb). Each movie was represented by one coordinate in this representation. The dot product (sans-normalization) between every two such vectors counts the number of movies both actors participated in together. The result, captured in Figure 3, provides a clear view of which actors and actresses played side by side.

While in the MDS-based plots actors that are well separated spatially may have played in the same movie and vise versa, the high capacity of our visualization depicts the relations between the actors reliably.

NASDAQ The stocks in the January 2008 NASDAQ-100 list were selected. Each stock was represented by a piece-wise normalized vector of its price changes during 2007 as described in [Gavrilov et al. 2000]. Note that the resulting linear kernel has negative values. We replaced these values by zeros. A number of frames are shown in Figure 1. The visualizaton reveals expected relations, such as the ones between Google and Yahoo, and Sandisk and Intel, as well as some less expected correlations such as the ones between the stocks of Apple and Google, and Autodesk and Starbucks. These correlations exist in the original data and arise from similar price patterns throughout 2007.

Facebook We have created a facebook application that allows users of the social networking website to visualize the connections between their friends. The service was tested by Computer Science students, their friends and a few random facebook users who stumbled across it, a total of 100 users. The service is currently down for the duration of the review period. The loading time of the application is dominated by the retrieval of the photographs of the facebook contacts, which is done in parallel to the computation of the smooth functions and the spatial embedding. Once the computation is done, the remaining images are loaded in the order in which they appear on screen, to allow for the quickest startup possible. The analysis takes a few seconds for users having 50 friends, and less than 2 minutes for users with $500-800$ friends. Due to the dictated structure of facebook applications, this computation is not done beforehand. However, in other (non-facebook) applications this may be possible. Also note that since solutions for nearby networks are expected to be similar, caching the results of previous sessions of the same user can reduce the computation time considerably assuming that the network of friends did not change drastically (this is not implemented yet).

In the facebook application, all the contacts of the user are rep-

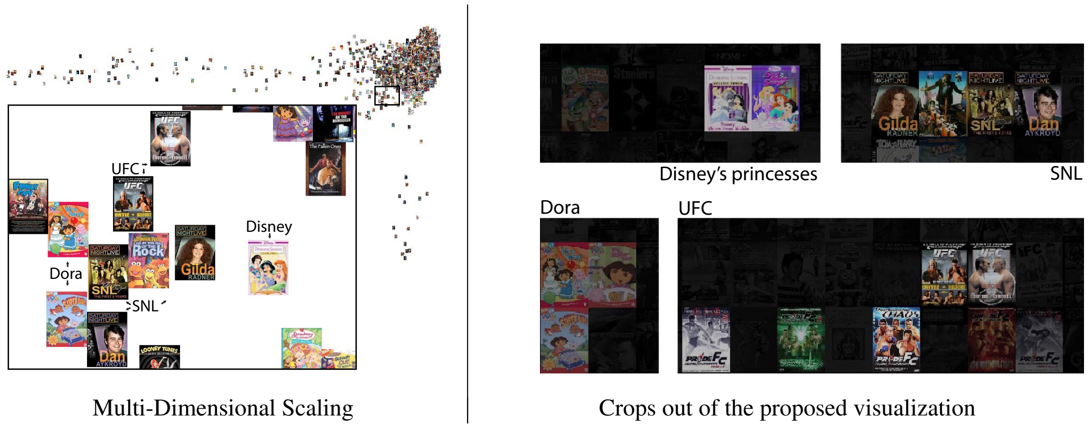
UFC
Disney
Dora
SNL
Multi-Dimensional Scaling
Disney's princesses
SNL
Dora
UFC
Crops out of the proposed visualization

Figure 2: A comparison between two visualizations of the netflix challenge 2005 DVDs. Proximities are based on the individual users' ratings only. The 2D embedding puts Disney's "Princess Stories", "Ultimate Fighting Championship", "Dora the explorer", and "Saturday Night Live" nearby. The capacity of our visualization (only crops are shown), enables full separation.

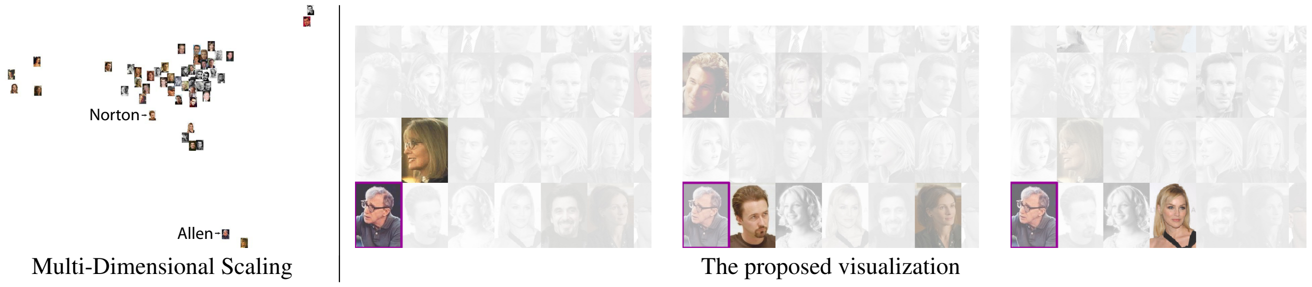

Figure 3: A dataset containing 50 actors and actresses, each associated with the list of movies he played in. Did Edward Norton and Woody Allen play in the same movie? Norton is in the middle of the MDS display. Allen is at the very bottom near Diane Keaton. The displayed animation frames were sampled when Allen's icon was selected.

resented as graph vertices. If person i and person j appear in the contact list of each other we create an edge between graph nodes i and j. As mentioned above, in this application, the resulting binary adjacency matrix is used as the affinity matrix. The resulting visualization for one person with 160 friends is a part of the attached flash-animation, and is captured in Figure 4.

We have received a considerable amount of feedback from the application users. It became evident that without preparation the purpose of the visualization is not clear to most viewers. We had to update the welcome splash screen and make it more appealing by adding an explanatory toy animation based on well known characters from "The Simpsons" television show (please refer to the supplementary material available at `http://www.cs.tau.ac.il/~wolf/demos/commonfate.zip`). The animation was accompanied with one line of text: "Cliques (groups of friends) appear lit together". All users we asked agreed that the structure of cliques among their friends is depicted correctly.

In the initial implementation, the selection could have led to a screen where all icons are unlit. Several users asked why some friends are never lit up. Similarly, two users reported dissatisfaction with the all unlit screen one would get in the initial implementation when selecting multiple people that do not share lit-up frames. To answer these concerns we added a better treatment of isolated graph nodes and the empty intersection problem as explained in Section 3.3 above.

Some users complained that the visualization is too slow, and some that it is slightly too fast (the facebook application did not contain the speed control at first). One user expressed the wish to have the ability to scroll by himself to other parts of the visualization. This feedback was made before the time slider and the data point selection box were added to the application.

4.1 Evaluation of the spatial embedding method

In the four visualizations described above there are two display modes: a windowed display mode of 800×600 pixels and a full screen one (1280×1024 pixels). The sizes of the icons range from 100×40 to 100×140, allowing visible grids of various sizes in each mode. Table 2 depicts the percentage of displayed lit items out of the total number of lit items. For example, in the NASDAQ visualization, there are 3,000 frames, and a total of 41,161 icons (counted again each frame) that are lit above a threshold of 0.1. Out of these lit events 39,341 are displayed on screen during the visualization loop, producing a hit rate of 96 percents. In the full screen mode of the same visualization all icons appear on screen, and the hit rate is 100%. As can be seen from the table, the hit rate is quite high.

5 Alternative Visualizations

Our framework of transforming proximities into smooth functions and displaying those through animation can be extended to other methods and some of the design choices we make can be replaced.

5.1 Alternative embedding methods

The proposed embedding, which associates a smooth positive function with each data point obtained through the method proposed in Section 3.1, can be replaced with other methods. We have con-

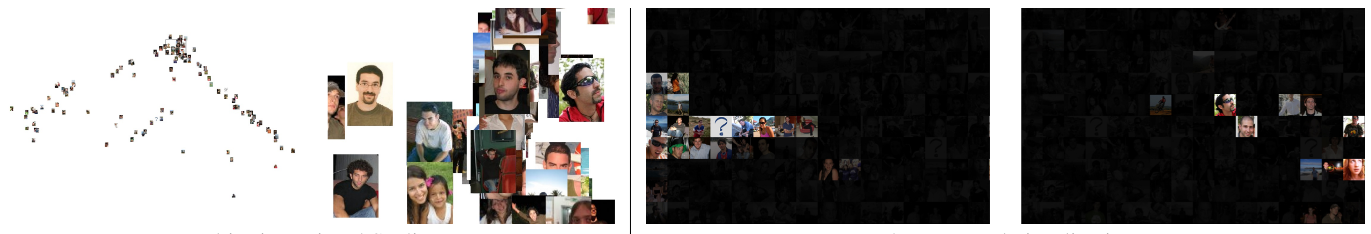

Figure 4: A comparison between MDS and our approach for visualizing the social network of one facebook user who has 160 contacts. Many other users have experimented with the application themselves. The number of friends per user in our experiments was as high as 800, and typically between 50 and 150.

Dataset	n	Windowed mode		Full screen mode	
		Grid size	Hit rate	Grid size	Hit rate
Netflix	512	11×6	77%	18×10	91%
Movie Stars	50	8×4	95%	10×5	100%
NASDAQ	100	9×10	96%	10×10	100%
Facebook	160	10×7	88%	16×10	100%

Table 2: The ratio of lit items displayed on screen out of all lit items in each visualization. The grid size columns depict the size of the visible area of the grid in each dataset for the windowed and the full-screen modes.

ducted experiments with several of these, and attach the results as part of our Multimedia Attachment. To access the visualizations that are based on alternative methods, please scroll on the dataset choice drop-down menu below the first four datasets.

The first such alternative is traversing through the data points, one at a time, and assigning a luminance value to all icons according to the proximities between the underlying data points. We have tried several methods to determine the traversing order, such as employing 1D MDS, and performing hierarchical clustering followed by Optimal Leaf Ordering [Bar-Yoseph et al. 2001]. In all cases the results seem to be too raw to be apprehended by the viewer. The structure of the data is not easy to understand without any automatic consolidation of the data taking place before it is displayed.

On the other hand of the spectrum, one can first cluster the data and then display the clusters one after the other. The order of the display can be determined by employing a 1D embedding method based on the proximities between the clusters. To create a smooth display, the clusters morph from one cluster to the next. This display is easy to grasp, however, it is limited by associating each data point to just one cluster, and has a very limited capacity.

To overcome this, one can attempt to employ a fuzzy clustering algorithm such as the fuzzy c-means algorithm [Bezdek 1981]. The clusters can then be sorted by their similarity and displayed sequentially, changing smoothly from one cluster to the next. The weighted cluster membership value is mapped to the luminance value of each icon.

The experiments conducted exposed several limitations of the fuzzy clustering based method. First, while the smoothness-based functional-embedding method of Section 3.1 is able to assign to each data point a varying amount of total luminance, fuzzy clustering methods assign a normalized value. Trying to relieve the normalization constraint seems to somewhat help. Second, the local nature of the fuzzy c-means algorithm seems to neglect some of the correlations between the data points.

Since the smooth functional embedding method is related to Non-negative Matrix Factorization (NMF), we also try to factorize the proximity matrix using a symmetric NMF algorithm [Li et al. 2007], and to sort the rows of the resulting factor matrix in order to produce a smooth animation. Three ordering methods were tried: a greedy method that starts with the first column, finds the most similar columns and continues iteratively, a 1D MDS method based on the similarities between the rows of the resulting factor matrix, and hierarchical clustering followed by optimal leaf ordering. Despite our efforts, the NMF based methods result in jittery animations, emphasizing the need to combine the smoothness requirement into the initial optimization stage.

As mentioned above, the resulting visualizations for all alternative methods are accessible in the Multimedia Attachment. Figure 5 provides a static view of the various alternatives. Each figure color codes (heatmap) the luminance for every datapoint (rows) for every frame (columns). It can be observed that the results obtained through the method proposed in Section 3.1 are not as cluttered as the Sequential display, are more continuous in time compared to the Symmetric NMF based method and the fuzzy c-means method, and unlike the k-means based method allow each data point to be highlighted more than once.

5.2 Alternative design choices

In the current implementation of the proposed visualization we choose to fix the grid locations of the icons associated with the individual data points. We believe that maintaining the spatial location is important when trying to relocate an icon we have seen previously. However, by relaxing this demand and only fixing the locations of icons that are displayed on screen at a given time we may be able to get better screen utilization, and deal with extremely large datasets. These adaptations are left for future implementations.

Throughout the paper we have focused on luminance based common fate. We have also experimented with other variables such as the color saturation (colorfulness), image blurriness or sharpness, and the perceived depth (see Figure 6). We found luminance changes to be much more appealing visually. Saturation and sharpness changes are hard to apprehend, and changes to the perceived depth create cluttered visualizations.

The common fate based luminance changes can be utilized in concert with other visualization methods. For example, it can be used to turn Parallel Coordinate plots into animated plots. On the other hand, other visualization techniques can be incorporated into the proposed grid visualization, for example, in order to display the source of proximity between data points that are selected. Note, however, that such modifications require information that is not contained in the affinity matrix, and may require adaptation for each type of data (graph/tabular based, dense/sparse, few/many variables).

6 Conclusions

The mapping of one variable to an image axis is a common visualization metaphor. Objects are often sorted, e.g., by size, and portrayed in order. When there are several classes of objects, it is

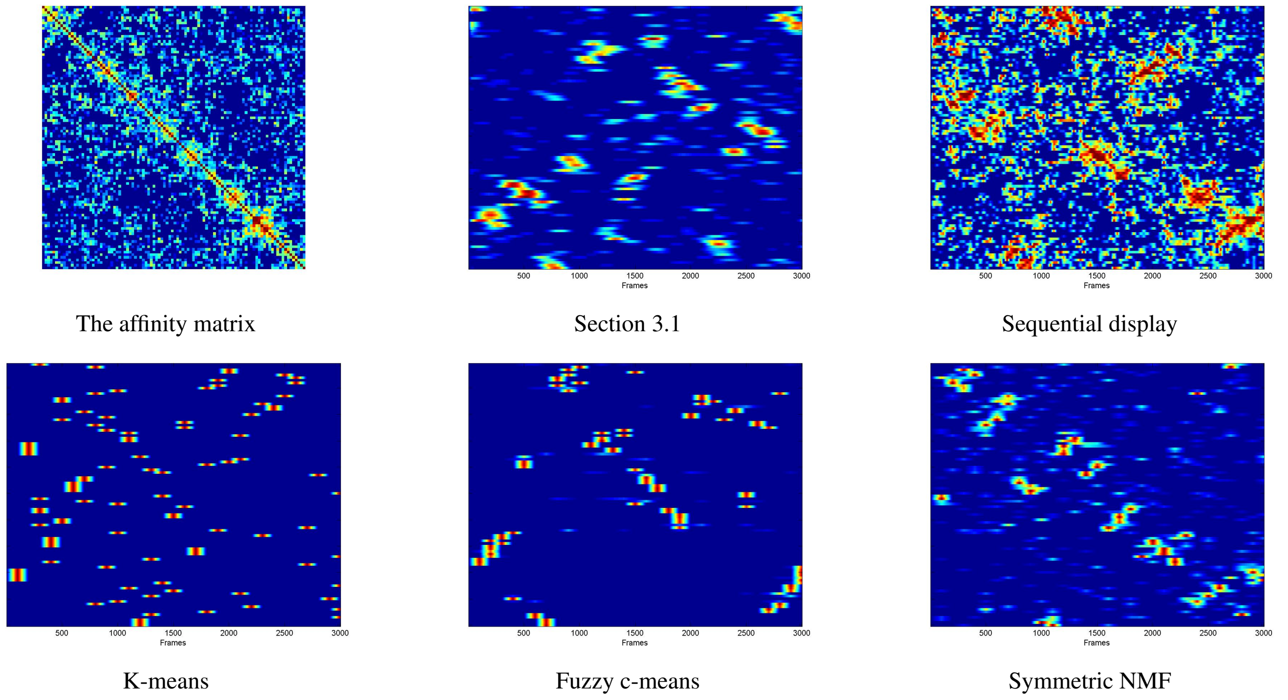

Figure 5: A comparison between various methods of embedding the data points in time on the NASDAQ dataset. The datapoints (rows of the images) are ordered, for display purposes, according to the 1D MDS projection. The method of Section 3.1 is the Smooth Non-negative Matrix Factorization method introduced in this paper. Sequencial display is a stretching in time of the original affinity matrix, where non-diagonal elements are further emphasized. The k-means method displays one cluster at a time, and fuzzy c-means works similarly. In Symmetic NMF, the matrix is factored and the rows of the factor matrix are rearranged to create an animation which is as smooth as possible.

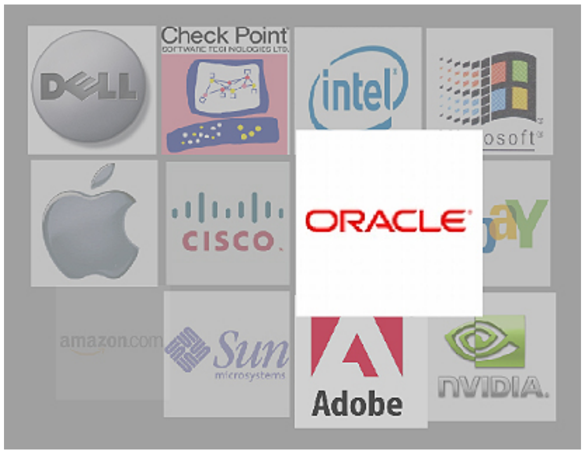

Figure 6: A screen capture of an alternative visualization in which the values associated with each data point at each frames were mapped to location in depth rather than to luminance. In this image a small subset containing 12 data points from the NASDAQ dataset is shown. Mapping values to depth information results in cluttered unattractive displays.

also common to group their iconic representation. It is therefore surprising that the graphical depiction of proximity in non-spatial multi-dimensional properties as positioning proximity is mostly a modern Western invention, that played little or no part of the rich visual languages developed since ancient times [Beniger and Robyn 1978]. Moreover, it seems that illustrating correlations as proximity in space is intuitively obvious only to certain educated sections of the population.

Looking for other alternatives for visualization, there is a clear psychophysical evidence, based on the Gestalt principle of Common Fate, that synchronized luminance modulation is a very effective perceptual tool [Sekuler and Bennett 2001]. This is best exemplified by our animation that shows the superiority of luminance common fate over spatial proximity in linking objects.

7 Acknowledgements

This research is supported by the Israel Science Foundation (grants No. 1440/06, 1214/06), the Colton Foundation, and The Ministry of Science and Technology Russia-Israel Scientific Research Cooperation.

References

BAECKER, R. 1998. Sorting out sorting: A case study of software visualization for teaching computer science. In *Software Visualization – Programming as a Multimedia Experience*, The MIT Press, 369–382.

BALASUBRAMANIAN, M., SCHWARTZ, E. L., TENENBAUM, J. B., DE SILVA, V., AND LANGFORD, J. C. 2002. The Isomap algorithm and topological stability. *Science 295*, 5552, 7.

BAR-YOSEPH, Z., GIFFORD, D. K., AND JAAKOLA, T. S. 2001. Fast optimal leaf-ordering for hierarchical clustering. *Bioinformatics 17*.

BELKIN, M., AND NIYOGI, P. 2001. Laplacian Eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems 14*, 585–591.

BENIGER, J. R., AND ROBYN, D. L. 1978. Quantitative graphics in statistics: A brief history. *The American Statistician 32*, 1, 1–11.

BEZDEK, J. 1981. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum, New York.

ELLIS, G., AND DIX, A. 2006. Enabling automatic clutter reduction in parallel coordinate plots. *IEEE Transactions on Visualization and Computer Graphics 12*, 5, 717–724.

FUA, Y.-H., WARD, M. O., AND RUNDENSTEINER, E. A. 1999. Hierarchical parallel coordinates for exploration of large datasets. In *VIS '99: Proceedings of the conference on Visualization '99*, 43–50.

GAVRILOV, M., ANGUELOV, D., INDYK, P., AND MOTWANI, R. 2000. Mining the stock market: Which measure is best? In *ACM Int. Conf. Knowledge Discovery and Data Mining*, 487–496.

HEER, J., AND ROBERTSON, G. 2007. Animated transitions in statistical data graphics. In *IEEE Information Visualization (InfoVis)*.

INSELBERG, A., AND DIMSDALE, B. 1990. Parallel coordinates: a tool for visualizing multidimensional geometry. In *IEEE Visualization*, 361–378.

LEE, D. D., AND SEUNG, H. S. 2001. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13*, MIT Press, 556–562.

LI, T., DING, C., AND JORDAN, M. I. 2007. Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization. In *ICDM*, IEEE Computer Society, 577–582.

LIMOGES, S., WARE, C., AND KNIGHT, W. 1989. Displaying correlations using position, motion, point size or point colour. In *Graphics Interface '89*, 262–265.

MICHOTTE, A. 1963. *The perception of causality*. Basic Books, New York.

NOCEDAL, J., AND WRIGHT, S. 1999. *Numerical Optimization,*. Springer Verlag.

NOVOTNY, M., AND HAUSER, H. 2006. Outlier-preserving focus+context visualization in parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics 12*, 5, 893–900.

PIERRE DEMARTINES AND JEANNY HERAULT. 1997. Curvilinear Component Analysis: A Self-Organizing Neural Network for Nonlinear Mapping of Data Sets. *IEEE Transactions on Neural Networks 8*, 1.

RAO, R., AND CARD, S. K. 1994. The table lens: merging graphical and symbolic representations in an interactive focus + context visualization for tabular information. In *CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems*, 318–322.

ROWEIS, S. T., AND SAUL, L. K. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science 290*, 5500, 2323–2326.

SAMET, H. 1984. The quadtree and related hierarchical data structures. *ACM Comput. Surv. 16*, 2, 187–260.

SEKULER, A., AND BENNETT, P. 2001. Generalized common fate: Grouping by common luminance changes. *Psychological Science 12*, 6, 437–444.

SHEPARD, R. 1962. The Analysis of Proximities: Multidimensional Scaling with an Unknown Distance Function (Part I). *Psychometrika 27*, 125–140.

STASKO, J. T. 1990. Tango: A framework and system for algorithm animation. *Computer 23*, 9.

TREISMAN, A., AND GORMICAN, S. 1988. Feature analysis in early vision: Evidence from search asymmetries. *Psychological Review 95*, 1, 15–48.

TVERSKY, B., MORRISON, J. B., AND BETRANCOURT, M. 2002. Animation: can it facilitate? *Internation Journal Human-Computer Studies 57*, 4, 247–262.

VENNA, J., AND KASKI, S. 2007. Comparison of visualization methods for an atlas of gene expression data sets. *Information Visualization 6*, 2, 139–154.

VON LUXBURG, U. 2007. A tutorial on spectral clustering. *Statistics and Computing 17*, 4, 395–416.

WARE, C., AND BOBROW, R. 2005. Supporting visual queries on medium-sized node-link diagrams. *Information Visualization 4*, 1, 49–58.

WISE, J., THOMAS, J., PENNOCK, K., LANTRIP, D., POTTIER, M., SCHUR, A., AND CROW, V. 1995. Visualizing the non-visual: spatial analysis and interaction with information from text documents. In *Proc. Info. Visualization*, 51–58.

WOLF, L., AND DONNER, Y. 2008. An experimental study of employing visual appearance as a phenotype. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on 0*, 1–7.

WOLF, L., AND DONNER, Y. 2008. Local regularization for multiclass classification facing significant intraclass variations. 748–759.