Interactive Visualization of Multidimensional Feature Spaces

Robert van Liere[1], Wim de Leeuw[1], and Florian Waas[2]

[1]CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands
{*robertl,wimc*}*@cwi.nl*

[2]Università di Bologna
Viale del Risorgimento 2
40130 Bologna
Italy
flw@acm.org

Abstract

Image similarity models characterize images as points in high-dimensional feature spaces. Each point is represented by a combination of distinct features, such as brightness, color histograms or texture characteristics of the image, etc. For the design and tuning of features, and thus the effectiveness of the image similarity model, it is important to understand the interrelations of individual features and the implications on the structure of the feature space.

In this paper, we discuss an interactive visualization tool for the exploration of multidimensional feature spaces. Our tool uses a graph as an intermediate representation of the points in the feature space. A mass spring algorithm is used to layout the graph in a 2D space in which arrangements of similar images are attracted to each other and dissimilar images are repelled.

The emphasis of the visualization tool is on interaction: users may influence the layout by interactively scaling dimensions of the feature space. In this way, the user can explore how a feature behaves in relation to other features.

1 Introduction

Visual information retrieval systems allow images to be retrieved from data repositories subject to a user defined query. Although the preferred mode of querying an image is semantic, queries are usually based on syntactic features of the image (such as color, texture and object shape). The discrepancy that results from using syntactic features to satisfy semantic queries causes a basic problem with the traditional query/response style of interaction. In addition, syntactic features are context sensitive in that a feature may successfully be used in one context, but can be inadequate in a different context. Hence, it depends on the image set which feature (or combination of features) is most useful for a search.

Image similarity models for visual information retrieval are a well studied subject. Such models usually represent an image as a point in a multidimensional feature space where *similarity* of two images is expressed by the distance between their points in the feature space. A larger similarity/dissimilarity corresponds to a smaller/larger distance of the points. Traditionally, image retrieval systems return a list of images, sorted by similarity to the query image. This list is then presented to the user as a list of thumbnail images. Unfortunately, such a presentation can be disorienting since relationships between the images of the answer set are largely ignored and only the similarity to the query image is assessed. More appropriate would be a presentation of the multidimensional space where the similarity relationships of all images in the vicinity of the query image, i.e. the answer set, are preserved and presented in a way that is easy and intuitive to grasp for the user.

In this paper, we represent the multidimensional feature space of images with a graph. A vertex in the graph represents a point in the space, while edges represent similarity relationships between images. The graph is displayed in such a way that vertices with strong similarities are attracted to each other and dissimilar vertices are repelled. The advantage of this presentation is that it gives a global overview of points in the feature space as well as similarity relations among points. In addition, the method allows a user to interactively scale each dimension of the feature space. By interactively scaling a dimension, the user can explore how a feature behaves in relation to other features.

The focus of our research is not to develop interfaces for end users of visual information retrieval systems. Rather, we are developing a framework in which feature developers can experiment with features on wide varieties of image sets. Our framework allows developers to gain insight into the weak and strong points of an individual feature, as well as insight into combinations of features. We believe that interactive interfaces, in which developers continuously control one or more dimensions of the feature space, are very intuitive for understanding the effect that features have on the underlying similarity model.

The remainder of this paper is organized as follows: After reviewing related work, in Section 3, we review the building blocks of our system. In Section 4 we present experimental results obtained with a real-world data set, detailing different scaling effects. We discuss possibilities and limitations of application of our system in Section 5 and present our conclusions in Section 6.

2 Related Work

User interfaces to visual information retrieval systems have gained much attention recently, see e.g. [1, 2]. Research is underway in defining new ways of representing the content of visual archives and the paths followed during a retrieval session. In retrieving visual information, high-level semantic concepts are often used together with perceptual features in a query.

Mass-spring algorithms are well known for graph layout, [3, 4, 5]. For example, Gross *et al.* developed a mass-spring based system that in which a similarity metric is quantified for objects in financial applications. This similarity metric drives the spring stiffness parameters of the mass-spring system which, in its equilibrium state, will reveal multidimensional relations and adjacency in terms of spatial neighborhoods.

Given a set of n objects in a K-dimensional space and a dissimilarity measure between objects, Multidimensional scaling (MDS) computes a configuration of points in a low-dimensional Euclidean space so that the Euclidean distances between two points match the original dissimilarities between the corresponding objects as precise as possible, [6]. The MDS procedure is realized by applying a least-squares technique to an objective function that penalizes the overall disparity between distances and dissimilarities. A minimum of the objective function yields the desired configuration. A number of commercial and research prototype image retrieval systems, which are based on MDS to display similarity, have been developed, including QBIC [7] and a research system built at Stanford Vision Laboratory [8]. QBIC displays the returned images as a list sorted by dissimilarity from the query. The Stanford system applies MDS to the dissimilarity matrix and places image thumbnails at the coordinates of the resulting two-dimensional projection.

3 Methods

3.1 Similarity Metrics

For any given image, a feature is expressed as a k-dimensional vector $f_i = \langle v_{i1}, v_{i2}, \ldots, v_{ik} \rangle$. The dimension of the vector may vary significantly from feature to feature. For example, the brightness of an image maybe noted as a single value, i.e., $k = 1$, whereas a color histogram may consist for instance of 128 values, i.e., $k = 128$.

Now, given a set of M features, we define the *feature vector* of an image as the composition of the single feature values:

$$F = \langle v_{11}, \ldots, v_{1k_1}, v_{21}, \ldots, v_{2k_2}, \ldots, v_{M1}, \ldots, v_{Mk_M} \rangle$$

Accordingly, the dimension of F is $K = \dim(F) = \sum_{i=0}^{M} \dim(f_i)$. For some $\lambda_i \in [0; 1]$, let S be a matrix of the shape

$$S = \begin{pmatrix} A_1 & & & \mathbf{0} \\ & A_2 & & \\ & & A_3 & \\ & & & \ddots \\ \mathbf{0} & & & A_M \end{pmatrix}$$

with matrices $A_i = \lambda_i \cdot I_{\dim(f_i)}$ where I_n denotes the identity matrix of order n.

Using S as a scaling matrix, we can compute the distance matrix D for a set of images $I_1, \ldots, I_n$ as

$$D = \begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \vdots & \vdots & & \vdots \\ d_{n1} & d_{n2} & \cdots & d_{nn} \end{pmatrix}$$

with $d_{ij} = \| S \, |(F_i - F_j)| \, \|$. F_i and F_j are the feature vectors of image I_i and I_j, respectively. The d_{ij} can be interpreted as the *scaled similarity metric* between image I_i and I_j.

Examples. To illustrate the concept of features, we survey some features widely used in the literature.

Color-based features. Color-based features operates solely based on the color information contained in the image. The most prominent representatives of which are color histograms where the images are usually first dithered and the incidence of the single colors is determined afterwards [9, 7, 10]. Color histograms have been frequently used in related work as reference technique to assess the performance of new features [11]. Also simpler features that reduce even to a single scalar value are conceivable like the information of how many different colors occur, the brightness, or the contrast of an image. Especially with cliparts, i.e. non-photographic images, these very simple features are often of high distinctive power. On the other hand, color histograms can be refined to reflect also some spatial information of the image by allowing for color transitions, i.e. these histograms do not record the number of pixels per color but the number of pairs of neighbored pixels that make up a certain color transition. For example with a color palette of 16 colors the transition histogram covers 256 color transitions. Depending on the particular feature, the quality can be enhanced by applying filters like despeckle or blur filters to the image as a pre-processing.

Texture-based features. Texture-based feature captures structures within the images. The most typical representatives are approximations with periodic functions like the Fourier transform. Here, the feature vector corresponds to the sequence of coefficients found. Other important members of this class include Gabor filters and wavelets. Texture-base features are particularly successful when applied to genres of images where color information is of lesser importance, e.g. air photography [12].

3.2 Layout

A graph is used as an intermediate representation of the points in the feature space. Vertices represent the points in the K-dimensional feature space while edges model the similarity relationship between points.

For the layout of the graph a mass-spring system is used. Edges are modeled as springs. A minimization algorithm computes an equilibrium configuration of the points with minimal energy. Unfortunately, to compute one step in the energy minimization algorithm, most spring mass systems are in $O(N^2)$, where N is the number of vertices. This makes such fully connected mass-spring systems non-scalable for interactive usage.

To overcome this scaling problem, we define an alternative mapping from the distance matrix to the graph. Instead of generating a fully connected graph, we generate a graph in which only *highly similar* vertices are connected. For this, we introduce a threshold distance T. When the graph is constructed only edges corresponding to a distance that is less than T are taken into account; i.e. vertices i and j have an edge if and only if $d_{ij} < T$.

The governing equations of the interactive mass spring layout model are captured as follows: denote the position of vertex i in visualization space as p_i and the position of point i in multidimensional space as P_i. The force applied by a vertex j onto a vertex i depends on the discrepancy between the $\|p_j - p_i\|$ and $\|P_j - P_i\|$. If the discrepancy of these two distances is large than the force will be large. The resulting total force applied to a vertex is the sum of all forces on the vertex. The mass spring algorithm will minimize the total discrepancy of the distances.

With $\mathbf{v}_{ij} = \frac{p_i - p_j}{\|p_i - p_j\|}$, the unit vector in the direction from p_i to p_j, we define the force between vertices i and j as

$$\mathbf{F}_{ij} = \begin{cases} w_{ij}(d_{ij} - \|p_i - p_j\|) \cdot \mathbf{v}_{ij} & \text{if } \|p_i - p_j\| < R \text{ or if } d_{ij} < T \\ 0 & \text{otherwise} \end{cases}$$

The visualization space paramter R can be set by the user.

This formulation allows an efficient algorithm to be implemented. The computation of F_i is sped up in two ways. First, a uniform grid of radius R around vertex i is used to quickly test and select only those vertices in the neighborhood of i. Second, edges are used to select only those vertices of a distance smaller than d_{ij} to i. In this way, instead of testing all vertices, only a limited number of vertices have to be tested, (see [13]).

3.3 GraphSplatting

GraphSplatting is a technique to transform the graph into a continuous field. It is based on the observation that the density of vertices is an important characteristic of the graph. Splatting projects each vertex of the graph onto a two-dimensional scalar field. Instead of showing the individual vertices and edges, the variations in density are shown. Vertices of the graph are represented in the field by a splatting function. Each vertex contributes to the field with a two-dimensional Gaussian shaped basis function. The resulting field is constructed by adding all the contributions. This field is called the *splat field*.

Figure 1 illustrates the mapping primitive. The figure shows a cross section of the Gaussian splatting function. The width of the Gaussian (σ in Figure 1) determines the 'smoothness' of the splat field. A large value of σ will result in smoothing out the details of the graph. Using a small value for σ will present more detail of the graph. In the limited case $\sigma = 0$ the original vertices will be represented as points. The user can interactively control the width of the splats with a global parameter.

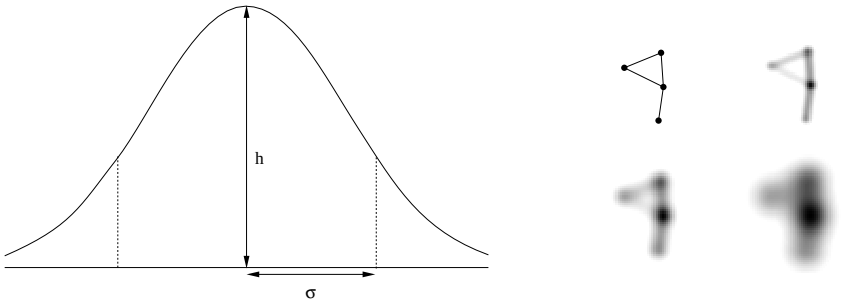

Figure 1: Visualization of a splat field. The left panel shows the mapping parameters for the base function; The right panel shows three splat fields of a graph with 4 vertices and 4 edges (upper left). The splat fields have different splat widths.

The height (h in Figure 1) of each splat can be used for mapping an attribute of the vertices. In this way, different properties of the graph can be highlighted. Vertices with a large attribute values will contribute more to the splat field that vertices with low attribute values.

GraphSplatting is designed to be used in combination with other graph rendering methods. A continuous representation is often useful for obtaining an overview of the data associated with the vertices of the graph. After zooming into a detail, it can be combined with other graph visualization methods.

4 Results

Test Image Set. We applied our methods to a synthetic test set of 3276 images. The test set consisted of 36 groups of images with distinct hue values. Each group had 91 textures of varying frequency and orientation. For each image, 6 feature vectors were computed: a 1 four-dimensional gabor feature vector for texture analysis and 5 distinct color-based features vectors. The color-based features vectors including a hue histogram, a hue histogram of the center region of the image, and 3 hue transition histograms. For transition histograms, the hue is first dithered to 16 bins; then the histogram of the 256 resulting combinations is recorded. As a pre-processing step, the images were segmented into 32, 128, and 256 tiles, and each tile was replaced by its dominant hue. The dimensionality of the feature space spanned by the 6 features vectors is 804.

Figure 6 shows a snapshot of the user interface. The upper panel shows the graph view: an arrangement of the graph in the visualization space. Small dots are used to represent vertices. Grey lines represent edges between points with distances below the threshold distance T. Edges also provide additional feedback on the state and progression of the layout algorithm. For example, very long edges will indicate that the layout algorithm has not reached an equilibrium. Some selected vertices are annotated with a thumbnail image. The lower panel shows the splat field, which is color encoded from white (low density values) to black (high density values). In Figure 6, the mass spring algorithm has reached an equilibrium. Users can drag vertices to other positions, after which the mass spring algorithm will com-

pute a new equilibrium. Animation is used to display each step of the mass spring system evolving to an equilibrium. In this way, the user can study how an arrangement evolves towards another.

The graph provides a 2D view in which the images are displayed according to their mutual dissimilarities and similar images are clustered. A problem with the graph view is the potential cluttering, making it difficult to estimate density of vertices in dense regions. The splat field provides a 2D view of a continuous density field. Colors are used to show which areas have a high density of vertices. In this way, the user can see in a glance which images are similar.

Scaling is illustrated in Figure 3. Each panel of the 3x3 matrix show the splat field of the graph layout in a equilibrium. Each row has scaled the hue histogram feature vector by incrementing the corresponding λ by 0.5. Similarly, each column has scaled the gabor feature vector with increasing λ settings.

The influence of scaling the hue histogram feature vector in combination with scaling the gabor feature vector can be analyzed from the matrix. For example, the arrangements shown in the first row are very different than the arrangements in the last row. In addition, the circular pattern of small clusterings seen in the last row, can already be discerned in the second row. This observation indicates that the hue histogram feature vector is dominating the gabor feature vector for this test set.

Corel Image Collection. We also experimented with images taken from the Corel Image Collection [14]. A set of 200 images were selected across different genres, yet, at the same time care has been taken that there is a small fraction of images per genre that would be commonly regarded as "similar". For example, images of similar objects like sailing boats, or image of objects which differ in lighting characteristics or camera positions only. The 6 feature vectors mentioned above were computed for each image.

The four panels in Figure 4 demonstrate the effects of different λ settings. These panels shows that different weightings of features lead to distinctly different clusterings of the graph. Scaled features are meaningful in their own way but provide significantly different separation of different regions or images.

In this case the gabor and hue histogram feature vector were scaled. The upper left panel shows the graph with λ setting for the gabor feature at 1.0 and the λ setting for the hue histogram feature at 0.0. In the upper right panel, the λ settings were 0.8 and 0.2. In the lower left panel, the λ settings were 0.2 and 0.8. In the lower right pane, the λ settings were 0.0 and 1.0. The threshold value was set so that each graph contained approximately 1100 edges.

Each panel show a very different structure of the underlying graph. The upper left panel shows a structure with 4 clusters of vertices. Clusters are connected with relatively few links. The lower right panel shows a structure with dense cluster.

To illustrate the scalability of the splat field, we have applied our methods to larger image sets. Figure 6 shows the splat fields of four image sets taken from the Corel Image Collection. The sets have 1000, 4788 and 10000 images respectively. The same set of features as above was computed for both sets. To generate the snapshots, the mass spring algorithm was used with the λ of the gabor feature vector set to 1.0. All other λ factors were set to 0.0.

The top row shows the graph view for each layout. Graph views with 1182, 33524 and 86521 edges are very cluttered and it is very difficult to determine which areas contain images that are similar. The bottom shows the splat field for each layout. Here, the structure of the graph is clearly shown. This is useful particularly in areas of high vertex density, i.e. those areas in which images have high similarity. Also, the structure of the graph is very similar for the four image sets.

5 Discussion

The framework discussed in this paper allows developers concerned with the design and tuning of features to experiment with the precision of particular features, feature distributions, similarity models, and the visualization of similar images. The interactive visualization tools are tailored towards the exploration and presentation of the underlying multidimensional spaces.

5.1 Visualization

The layout algorithm detailed in section 3.2 generates arrangements in which similar images are attracted and dissimilar images are repelled. Since related images are grouped together in order of increasing dissimilarity, the density of the images can be interpreted as a measure of image similarity. Images are displayed either as points (useful for density distributions), thumbnails (useful for visual similarity comparisons), or the complete image.

However, the interactive nature of the interface provides additional advantages regarding both the layout and modification of the points in the space by different feature weightings.

- Layout

  Besides the actual layout algorithm, the tool comes with a whole array of interactive elements including zooming, drag-and-drop of vertices, inspecting a vertex properties, highlighting of neighbored vertices and connecting edges etc. This enables users to disentangle dense areas and study neighborhoods of individual vertices.

  The animation of the layout algorithm gives an immediate impression of the strength of the links and components. Additionally, the user can also adjust the velocity of the convergence of the system to study these effects also in slow-motion.

- Scaling

  It is well-known that there is no *universal* concept of similarity but similarity always depends on both the properties of the query image and the images stored in the repository. For different sets of images, features differ in their effectiveness. In addition, the effectiveness may vary even from one region of the feature space to another [15]. Thus, weighting and scaling of features is a necessity.

  Studying these effects with statistical methods like cluster analysis etc. is often not satisfactory. Capturing the structure of the points in space in order to describe the effects of scaling is computational expensive and results are difficult to grasp. In contrast, scaling the influence of features interactively, helps to grasp these effects in an immediate and evident way.

Our experience has been that for the bulk of images the weighting is of little influence—these images appear very similar under many different features, particularly, since features often subsume other features. However, there are also areas where the influence of a few, sometimes even one single feature, is crucial. Interactive and animated scaling of single features makes it easy to explore and analyze the impact of individual features. Our layout provides for the visualization of areas which is facilitated by the threshold parameter. Modifying this parameter enables pruning of the similarity relationships in the graph to blot out regions that are of little interest. It does not matter exactly how distant dissimilar images are from a given image, as long as they are far in relation to similar ones.

A splat field is used to show the density of one single dimension of the feature space by mapping the feature value of a feature to a splat. Since splat fields are continuous representations of the discrete

graph, they allow for the visualization of very large graphs. Individual vertices will not be discerned, but the continuous field will contain density information that can be used to determine clusters of similar images.

5.2 Comparison with other MDS based systems

The use of a mass-spring system for the layout of a multidimensional space is an approximation to MDS. Recall that MDS uses a least-square technique to define an objective function that penalizes the overall disparity between distances and dissimilarities. The MDS objective function may be interpreted as the total energy of a fully connected mass-spring system, with a vertex for each image, and springs connecting each vertex. The relaxed length of a spring connecting two vertices is given be the dissimilarity between the corresponding pair of images. The actual length of the spring is the Euclidean distance between vertices. The equilibrium of this spring system corresponds to the minimum of the MDS objective function.

Our method differs from MDS when the graph is not fully connected. Constructing a graph from the distance matrix uses a distance threshold T. When the graph is constructed only edges corresponding to a distance that is less than T are taken into account; i.e. vertices i and j have an edge if and only if $d_{ij} < T$. For example, when $T = 0$, the graph will have no edges and if $T = \infty$, the graph will be fully connected.

In the case of a not completely connected graph, our mass-spring algorithm will result in groups of similar images. Mutual distances between images within a group can be interpreted as a measure of similarity. Distances between groups have no meaning, but the similarity between images in different groups is known to be larger than the distance threshold.

6 Conclusion

The framework discussed in this paper allows image feature developers to experiment with the precision of particular features, feature distributions, similarity models, and the visualization of similar images. The interactive visualization tools are tailored towards the exploration and presentation of the underlying multidimensional spaces.

We have demonstrated that our tools can be used to gain insight into the strengths and weaknesses of features and how they perform in combination with other features. We believe that interactive interfaces, in which users continuously control one or more dimensions of the feature space, are very intuitive for understanding the effect that features have on the underlying similarity model.

References

[1] S.K. Card, J.D. Mackinlay, and B. Shneiderman, editors. *Readings in Information Visualization*. Morgan Kaufmann Publishers, 1999.

[2] A. del Bimbo. *Visual Information Retrieval*. Morgan Kaufmann Publishers, 1999.

[3] G. di Battista, P. Eades, R.A. Tamassia, and J.G. Tollis. *Graph Drawing*. Prentice Hall, 1999.

[4] R.J. Hendley, N.S. Drew, A.M. Wood, and R. Beale. Narcissus: Visualizing information. In S.K. Card, J.D. Mackinlay, and B. Shneiderman, editors, *Readings in Information Visualization*, pages 503–511. Morgan Kaufmann Publishers, 1999.

[5] M.H. Gross, T.C. Springer, and J. Finger. Visualizing information on a sphere. In *Proceedings Symposium on Information Visualization*, pages 11–16. IEEE Computer Science Press, 1997.

[6] T.F. Cox and M.A.A. Cox. *Multidimensional Scaling*. Chapman & Hall, London, 1994.

[7] J. Ashley, M. Flickner, J.L. Hafner, D. Lee, W. Niblack, and D. Petkovic. The Query By Image Content (QBIC) System. In *ACM SIGMOD Conference on Management of Data*, page 475, 1995.

[8] Y. Rubner, C. Tomasi, and L.J. Guibas. A Metric for Distributions with Applications to Image Databases. In *IEEE International Conference on Computer Vision*, pages 59–66, Bombay, India, January 1998.

[9] V. Ogle and M. Stonebraker. Chabot: Retrieval From a Relational Database of Images. *IEEE Computer*, 28(9):40–48, September 1995.

[10] A. Pentland, R. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. *International Journal of Computer Vision*, 18(3):233–254, June 1996.

[11] J. Huang, S. R. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih. Image Indexing Using Color Correlograms. In *IEEE Computer Vision and Pattern Recognition*, pages 762–768, Puerto Rico, 1997.

[12] B.S. Manjunath and W.Y. Ma. Texture features for browsing and retrieval of large image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842, August 1996.

[13] R. van Liere and W. de Leeuw. Graphsplatting: visualizing graphs as continuous fields. Submitted for publication. Available at http:// www.cwi.nl/~robertl.

[14] Corel, http://www.corel.ca/products/clipartandphotos/photos/index.htm. *Corel Stock Photos*, 1999.

[15] S. Santini and R. Jain. Similarity is a Geometer. *Multimedia Tools and Applications*, 5(3):377–306, November 1997.

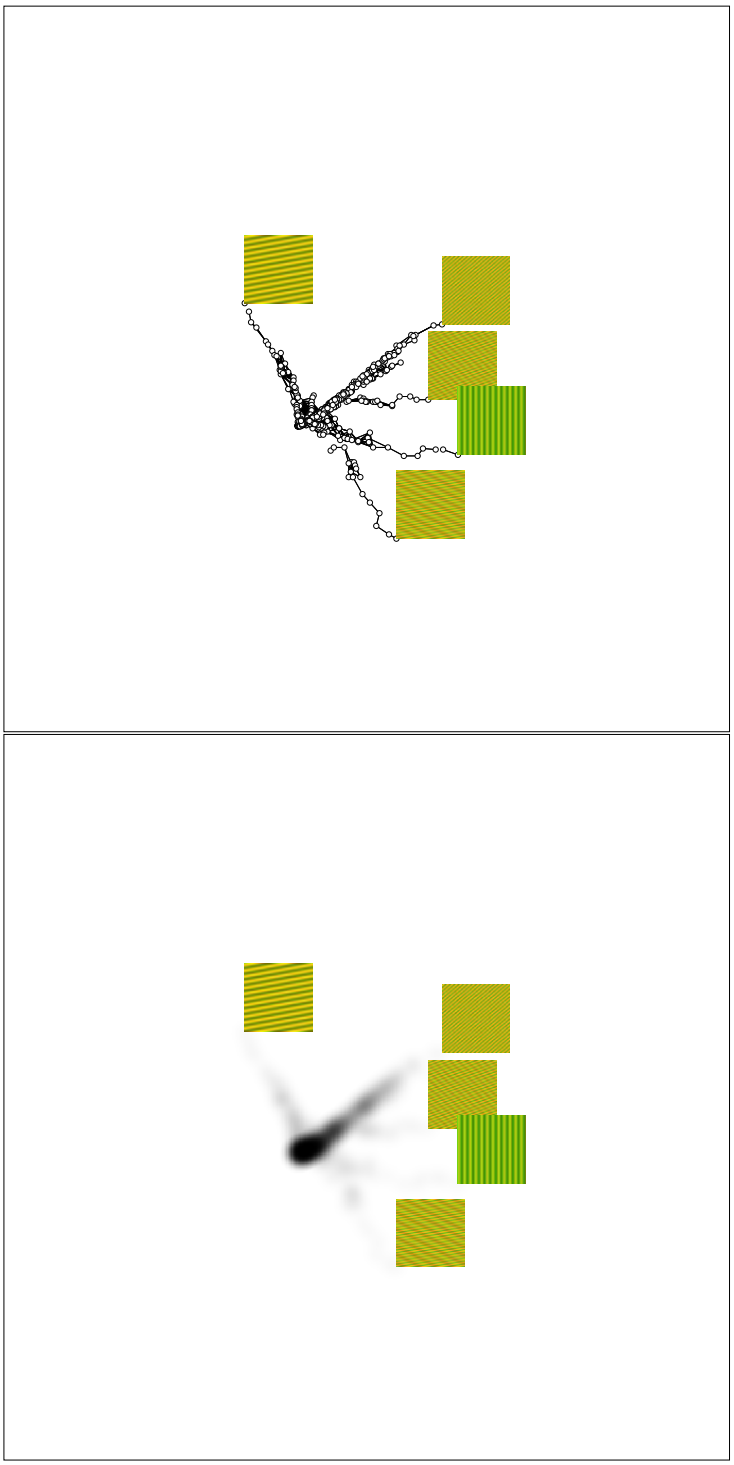

Figure 2: Two views of a graph arrangement for the test set. The upper panel shows graph view with vertices and edges. The lower panel shows the splat field. Some vertices are annotated with a thumbnail image.

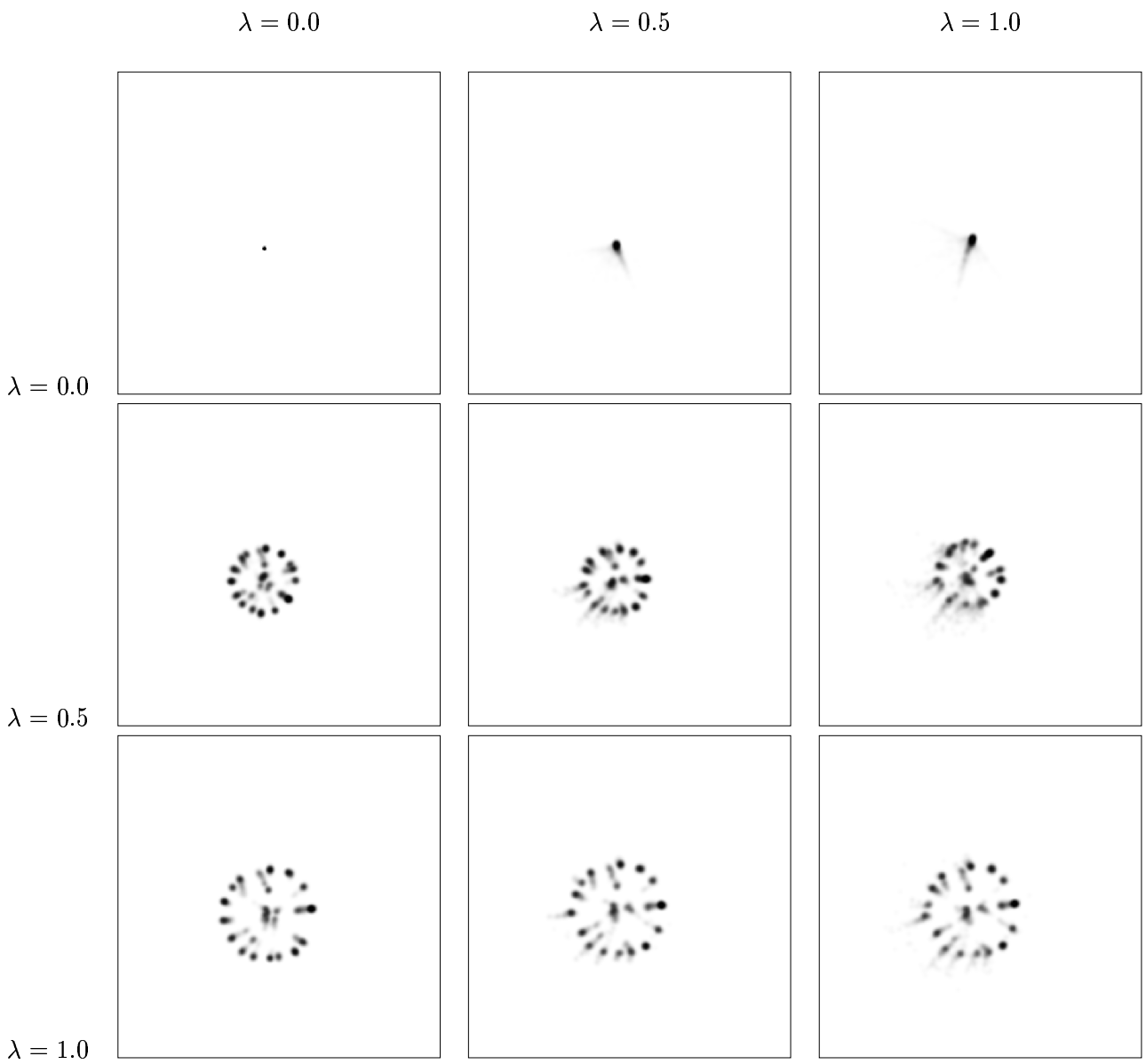

Figure 3: Nine splat fields with different scaling factors. Rows have increasing λ values for the hue histogram feature vector. Columns have increasing λ values for the gabor feature vector.

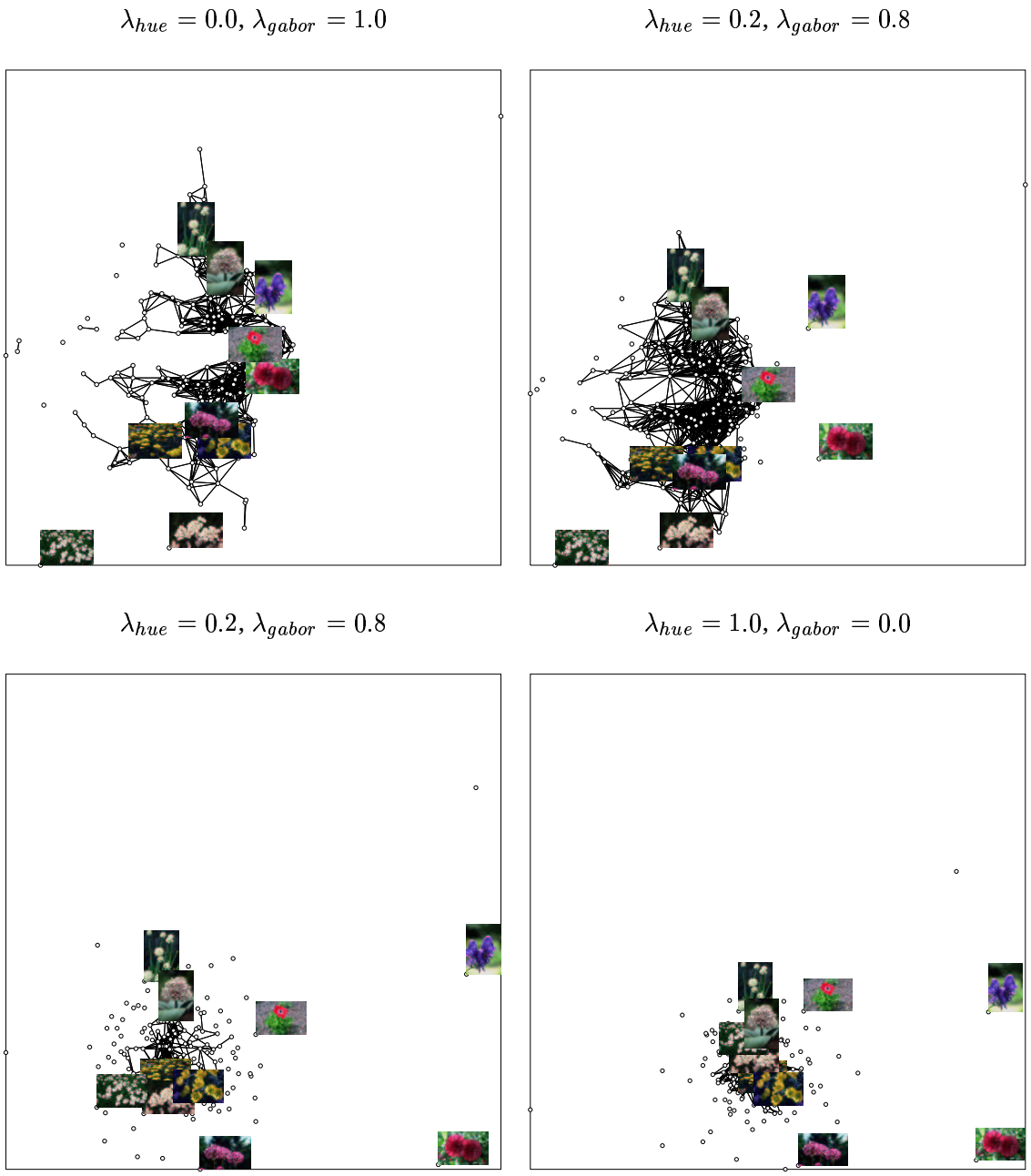

Figure 4: Four panels showing the graph with different scaling factors for the hue histogram and gabor feature vectors. Ten vertices are annotated with a thumbnail image.

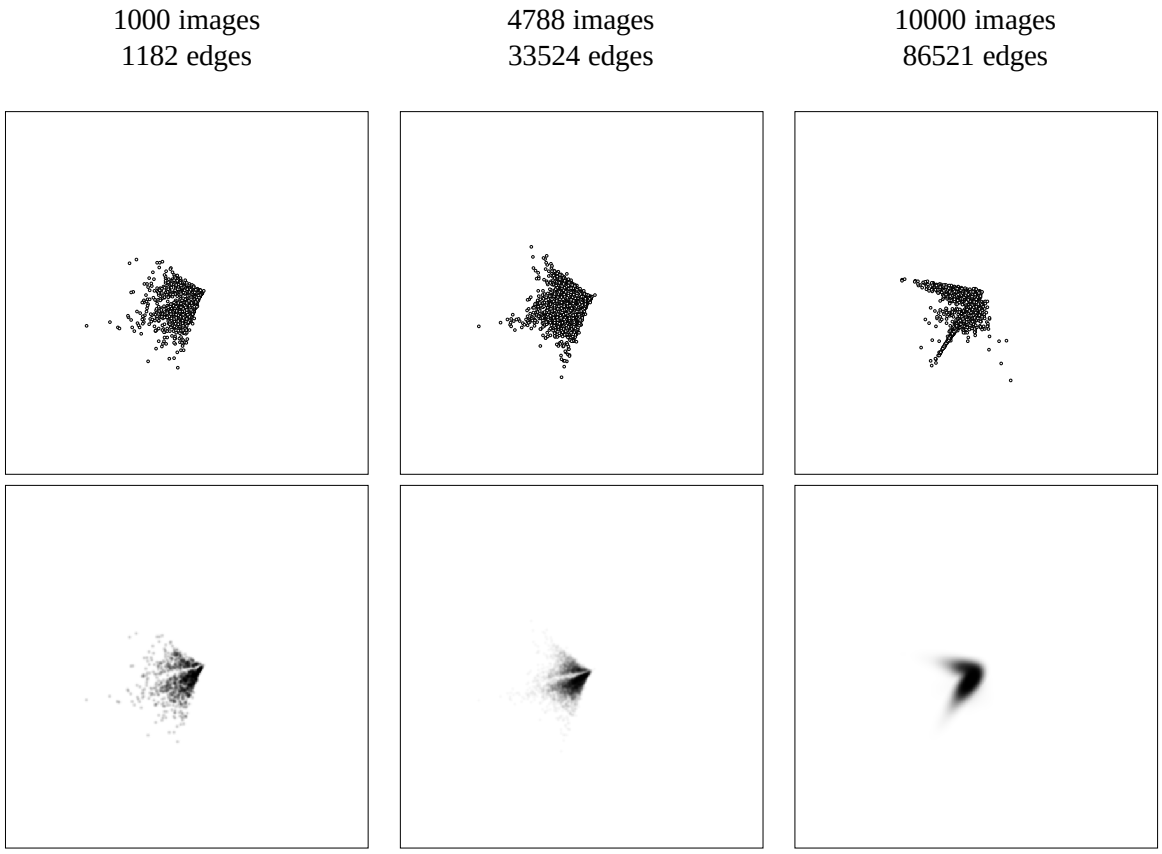

Figure 5: Scalability of splat fields. The top three panels show the graph views of three image sets. The bottom three panels show the splat fields. The gabor feature vector was scaled to 1.0.