

iGraph: A Graph-Based Technique for Visual Analytics of Image and Text Collections

Yi Gu^a, Chaoli Wang^a, Jun Ma^b, Robert J. Nemiroff^c, and David L. Kao^d

^aDepartment of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556;

^bDepartment of Computer Science, Michigan Technological University, Houghton, MI 49931;

^cDepartment of Physics, Michigan Technological University, Houghton, MI 49931;

^dNASA Ames Research Center, Moffett Field, CA 94035

ABSTRACT

In our daily lives, images and texts are among the most commonly found data which we need to handle. We present iGraph, a graph-based approach for visual analytics of large image and text collections. Given such a collection, we compute the similarity between images, the distance between texts, and the connection between image and text to construct iGraph, a compound graph representation which encodes the underlying relationships among these images and texts. To enable effective visual navigation and comprehension of iGraph with tens of thousands of nodes and hundreds of millions of edges, we present a progressive solution that offers collection overview, node comparison, and visual recommendation. Our solution not only allows users to explore the entire collection with representative images and keywords, but also supports detailed comparison for understanding and intuitive guidance for navigation. For performance speedup, multiple GPUs and CPUs are utilized for processing and visualization in parallel. We experiment with two image and text collections and leverage a cluster driving a display wall of nearly 50 million pixels. We show the effectiveness of our approach by demonstrating experimental results and conducting a user study.

Keywords: large image and text collection, graph layout, progressive drawing, node comparison, visual recommendation

1. INTRODUCTION

With the booming of digital cameras, image archiving and photo sharing websites, browsing and searching through large online image collections has become a notable trend. Consequently, viewing images separately as individuals is no longer enough. In many cases, we now need the capability to explore these images together as collections to enable effective understanding of large image data. Another notable trend is that images are now often tagged with names, keywords, hyperlinks and so on. Therefore, solutions that can nicely integrate images and texts together to improve collective visual comprehension by users are highly desirable.

In this work, we develop iGraph, a visual representation and interaction framework to address the increasing needs of browsing and understanding large image and text collections. These needs include the following. First, when relationships among images and texts are extracted and built in the general form of a graph, effective navigation through such a large graph representation becomes critically important. A good solution must allow collection overview and detailed exploration. This demands a flexible graph layout that dynamically and smoothly displays relevant information content at various levels of detail. Second, visual guidance should be given so that users can easily explore the collection with meaningful directions. Besides interactive filtering, the capability to compare nodes of interest for deep comprehension is necessary. Third, automatic recommendation that provides

Further author information: (Send correspondence to Chaoli Wang)

Yi Gu: E-mail: ygu5@nd.edu

Chaoli Wang: E-mail: chaoli.wang@nd.edu

Jun Ma: E-mail: junm@mtu.edu

Robert J. Nemiroff: E-mail: nemiroff@mtu.edu

David L. Kao: E-mail: david.l.kao@nasa.gov

the suggestions for further exploration is also desirable. Such a capability allows users to browse through the graph in a progressive manner.

We experiment with two well-known collections: the APOD collection and the MIR Flickr collection. The Astronomy Picture of the Day (APOD)¹ is an online astronomy image collection maintained by NASA and Michigan Technological University. Everyday APOD features a picture of our universe, along with a brief explanation written by a professional astronomer. Since its debut in June 1995, APOD has archived thousands of handpicked pictures, which makes it the largest collection of annotated astronomy images on the Internet. The MIR Flickr collection² is offered by the LIACS Medialab at Leiden University. The collection was introduced by the ACM MIR Committee in 2008 as an ACM sponsored image retrieval evaluation. We use the MIRFLICKR-25000 collection which consists of 25,000 annotated images downloaded from the social photography site Flickr through its public API.

We extract image and text information from each collection, analyze their similarities to build interconnections, and map extracted data and their relationships to a new compound graph representation. Our iGraph consists of tens of thousands of nodes and hundreds of millions of edges. To enable effective exploration, we incorporate progressive graph drawing in conjunction with animated transition and interactive filtering. Rather than building a fixed hierarchical organization of images and texts, our notion of *progressive graph drawing and refinement* allows more flexible and desirable navigation. Node comparison is enabled by visually arranging selected nodes and their most related ones for detailed analysis. We provide various means for image and keyword input so that users can conveniently select nodes of interest for purposeful comparisons. To provide effective guidance, automatic visual recommendation is realized by providing the suggestions for future exploration based on the analysis of image popularity, text frequency, and user exploration history. The performance of iGraph is ensured with the utilization of multiple GPUs and CPUs in the processing and visualization, along with a display wall consisting of 24 monitors. We also conduct a user study to evaluate the effectiveness of iGraph.

2. RELATED WORK

2.1 Image and Text Collections

Organizing image collections into various visual forms has been a well-studied topic. Examples include PhotoMesa with quantum treemaps and bubblemaps for zoomable image browsing,³ PhotoTOC for automatic clustering for personal photograph browsing,⁴ spiral and concentric rings for focus+context visualization,⁵ MoireGraphs with the radial graph layout for visual exploration,⁶ semantic image browser (SIB) with the multidimensional scaling layout for image overview,⁷ and dynamic image browsing with Voronoi diagrams.⁸ Companies like Google and Microsoft also developed products such as Image Swirl and Bing Visual Search for organizing large image collections. Besides images, text visualization has received lots of attention recently. Notable examples include Wordle,⁹ TIARA,¹⁰ and context preserving dynamic word cloud.¹¹ However, considering both images and texts in the visualization has not been thoroughly investigated.

2.2 Graph Layouts

Several survey papers discussed graph layout methods and performance comparisons.^{12–14} Recent trends on graph drawing are on large-scale complex graphs, such as heterogeneous, time-varying, and online graphs. For example, Handlak et al.¹⁵ introduced the concept of in situ visualization that tightly integrates existing visualization techniques for large dynamic networks. Feng et al.¹⁶ presented an algorithm for coherent time-varying graph drawing with multi-focus+context interaction. Zinsmaier et al.¹⁷ focused on dense edges in large graph drawing and proposed a technique that allows straight-line graph drawings to be rendered interactively with adjustable level of detail. Researchers have also investigated GPU-accelerated solutions for drawing large-scale graphs and graph drawing on large displays. For example, Frishman and Tal¹⁸ presented a multi-level graph layout on the GPU. Alper et al.¹⁹ presented the stereoscopic highlighting technique for 2D graph visualization and ran their experiment in the Allosphere Virtual Reality environment featuring a large spherical display with a 5m radius.

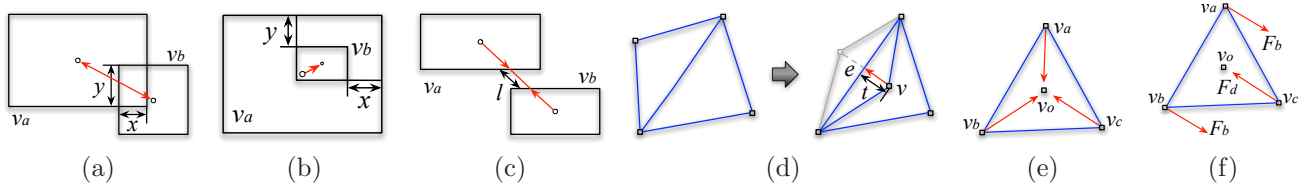


Figure 1. Four forces for constrained layout adjustment: (a) bidirectional repulsive force, (b) unidirectional repulsive force, (c) spring force, and (d) attractive force. Two forces for density adjustment: (e) density force and (f) bounding force.

2.3 Our Contributions

Our iGraph strives for flexible and desirable navigation as images and texts are not organized hierarchically. Rather than focusing on images alone, by integrating images and texts together in a compound graph, we allow users to explore the entire collection by navigating through images and texts of interest. Dynamic visualization of the large iGraph is achieved by progressive drawing in conjunction with interactive filtering. Node comparison and visual recommendation are enabled for user-guided detail comprehension and graph refinement. We leverage a cluster to speed up both preprocessing and runtime visualization. We also demonstrate the performance of iGraph by delivering the visualization results to a large display wall.

3. IGRAPH DEFINITION AND CONSTRUCTION

We define iGraph as a compound graph that consists of two kinds of nodes, i.e., I-node and K-node, and three kinds of edges, i.e., I-I edge, I-K edge, and K-K edge, where “I” stands for image and “K” stands for keyword. An I-node (K-node) represents an image (keyword) extracted from the collection. An I-I edge (K-K edge) is formed between two I-nodes (K-nodes) and the edge weight indicates the similarity between these two I-nodes (K-nodes). Finally, an I-K edge is formed between an I-node and an K-node only if there is a connection between them.

3.1 I-node and K-node

In iGraph, each I-node corresponds to an image. Each K-node corresponds to a keyword related to more than one image (otherwise, we treat it as a rare keyword and exclude it from iGraph). Keywords are given as tags in the MIR Flickr collection. For the APOD collection, we extract meta-tagged keywords from the HTML header and other keywords from the paragraph of explanation accompanying each image.

3.2 I-I edge, K-K edge, and I-K edge

An I-I edge exists between any two I-nodes and the edge weight is the similarity between the two images. Different images in the collection come with different dimensions, types and formats. For simplicity, we convert all images to the same type and format (i.e., portable pixmap format, PPM), and scale them down to a fixed resolution (256×256) for similarity analysis. Such a resolution strikes a good balance between maintaining image content and achieving comparison efficiency. We consider three aspects of images, namely, *grayscale content*, *power spectrum*, and *color histogram*, to calculate the distance between two images.²⁰ The overall distance between two images is a weighted sum of the three partial distances.

An K-K edge exists between any two K-nodes and the edge weight is the similarity between the two keywords. To analyze the similarity between two keywords, we utilize the Google similarity distance (GSD).²¹ Given that a keyword K_a appears in a webpage, its GSD to another keyword K_b is calculated as follows:

$$GSD(K_a, K_b) = \frac{\log \max\{f_a, f_b\} - \log c_{ab}}{\log MAX - \log \min\{f_a, f_b\}}, \quad (1)$$

where f_a and f_b are the frequencies of K_a and K_b , respectively, c_{ab} is their co-occurrence frequency, and MAX is the multiplication of the largest two frequencies in all keywords. As we can see, if K_a and K_b always appear simultaneously, their distance is zero. When the frequencies of K_a and K_b are constant, their distance gets

smaller if they share more webpages. When the number of sharing webpages is constant, their distance gets larger if the number of non-sharing webpages increases.

An I-K edge exists between an I-node and an K-node if and only if the image has the keyword in its tags in the MIR Flickr collection or the image and keyword exist in the same webpage of the APOD collection.

4. PROGRESSIVE DRAWING

4.1 Initial Backbone iGraph

Direct drawing the entire iGraph consisting of over tens of thousands of nodes incurs a heavy computational cost and usually produces a poor visualization, making the subsequent exploration very difficult. We therefore advocate the visual information seeking mantra: “*overview first, zoom and filter, then details-on-demand*”²² in the drawing. Specifically, we first extract the backbone of iGraph by identifying representative I-nodes and important K-nodes from the graph for overview. We apply *affinity propagation*²³ to the image collection to identify representative I-nodes. Unlike k-means and k-medoids clustering algorithms, affinity propagation simultaneously considers all data points as potential exemplars and automatically determines the number of clusters. For K-nodes, we rank their corresponding keywords based on the frequency to determine their importance.

For an I-I edge (K-K edge), the edge weight is defined as the similarity of the two incident I-nodes (K-nodes). For any I-K edge, we define the edge weight as 1 (0) if there is a (no) connection between the I-node and the K-node. We modify a classical force-directed graph layout algorithm, the Fruchterman-Reingold (FR) algorithm,²⁴ to draw the initial iGraph. Specifically, we design two forces, the repulsive force F_r and attractive force F_a . Given two nodes with their graph distance d and similarity σ , the two forces are calculated as $F_r = k_r/d$ and $F_a = k_a \times d^2 \times \sigma^2$, where k_r and k_a are the constant factors for the repulsive and attractive forces, respectively. We add σ^2 in the attractive force to ensure that more similar nodes are closer to each other. This is not included in the repulsive force, because we want to ensure that there is always a distance between any two nodes.

4.2 Dynamic Layout Adjustment

As a user explores iGraph, we dynamically refine it around areas of interest to allow navigation through the graph in a progressive manner. The graph keeps updating as the new images or keywords of focus are selected. Note that each I-node or K-node displayed in the graph occupies a certain rectangular area. We map the size of an I-node to the importance of the corresponding image. For K-nodes, we use the same font size, while their frequency is mapped to color saturation (bright red to dark red). The most challenging issue for dynamic layout adjustment is to reduce their overlap or occlusion. In particular, we do not allow an K-node to have any overlap with other nodes, while two I-nodes could overlap each other but we want to reduce such an overlap as much as possible. A good layout adjustment algorithm should also maintain a good balance between preserving the structural information of the graph and revealing the dynamics.

To this end, we generate an initial layout for the backbone iGraph. To achieve stable graph update, good screen utilization, and node overlapping reduction, we introduce four adjustment steps: *constrained layout adjustment*, *density adjustment*, *K-node overlapping adjustment*, and *occluded I-node removal*. For constrained layout adjustment, we apply a triangulation scheme²⁵ to all the nodes in the initial graph and use the resulting mesh to perform the adjustment. Similar to the algorithm described by Cui et al.,¹¹ we consider four kinds of forces to reposition the nodes to reduce their overlap while maintaining the topology of iGraph. These forces are:

- *Bidirectional repulsive force*: This force pushes away two nodes v_a and v_b from each other and is effective only if v_a and v_b overlap each other. It is defined as $F_1(v_a, v_b) = k_1 \times \min(x, y)$, where k_1 is a given weight and x and y are the width and height of the overlapping (see Figure 1 (a)).
- *Unidirectional repulsive force*: This force pushes away a node v_b from a node v_a and is effective only if v_b is inside v_a . It is defined as $F_2(v_a, v_b) = k_2 \times \min(x, y)$, where k_2 is a given weight and x and y are the width and height of the gap region (see Figure 1 (b)).

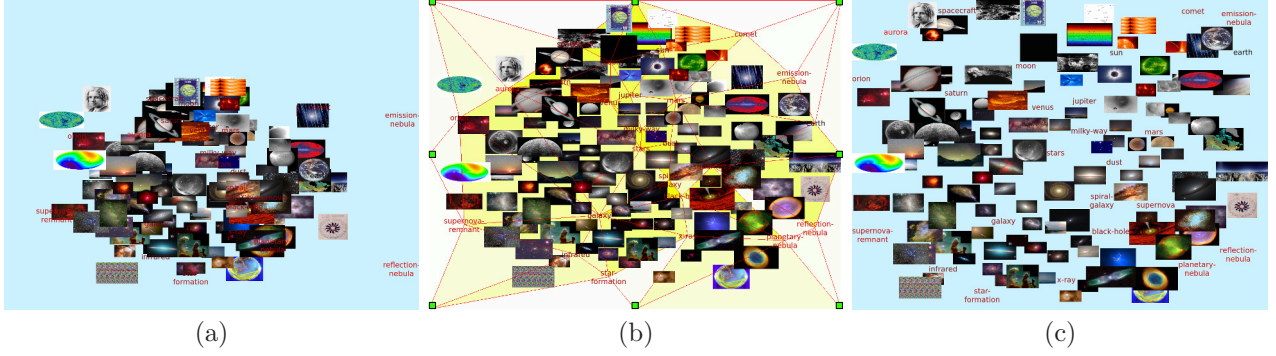


Figure 2. The process of dynamic iGraph layout adjustment. (a) The result of the modified FR algorithm. (b) The triangle mesh based on K-nodes after constrained layout adjustment. The colors of the triangles indicate their density values (higher saturation, higher density). Eight bounding nodes are marked in green. (c) The layouts after K-node overlapping adjustment and occluded I-node removal.

- *Spring force*: This force balances the graph by offsetting the two repulsive forces introduced. Given two nodes v_a and v_b , the spring force is defined as $F_3(v_a, v_b) = k_3 \times l$, where k_3 is a given weight and l is the length of the line segment connecting the centers of v_a and v_b that lies outside of their boundaries (see Figure 1 (c)).
- *Attractive force*: This force maintains the topology of the triangle mesh we construct for the graph. During layout adjustment, a triangle may be flipped (see Figure 1 (d)). Our goal is to maintain stable update of the graph by introducing an attractive force to flip the triangle back. The attractive force is define as $F_4(v) = k_4 \times t$, where k_4 is a given weight and t is the distance from node v to edge e . We also consider virtual triangle edges connecting extreme nodes in the graph to the *bounding nodes* (i.e., four corners and the midpoints of four sides of the drawing area, see Figure 2 (b)). This is to ensure that all graph nodes do not go out of bound.

For density adjustment, we apply the same triangulation scheme to the K-nodes only. For each triangle in the resulting mesh, we calculate its density value ρ as $\rho = \sum_i A_i / T$, where A_i is the size of the image whose center is in the triangle and T is the size of the triangle. Then we introduce the following two forces:

- *Density force*: The density force F_d keeps the triangle area proportional to its density. As shown in Figure 1 (e), for each node, F_d pulls it to the center of the triangle. The force is calculate as $F_d = k_d \times \rho$, where k_d is a given constant.
- *Bounding force*: If we only apply the density force, the eight bounding nodes will be pulled toward to the drawing center. However, since the bounding nodes are fixed, all the rest of points will be pulled to the drawing center. To balance the effect that F_d works on the bounding nodes, we introduce the bounding force F_b . In Figure 1 (f), assume v_c is the bounding node and v_a and v_b are not. F_d on v_c is the density force, F_b on v_a and v_b is the bounding force which has the same magnitude as F_d but in the opposite direction.

For K-node overlapping adjustment, we reduce the overlapping with any K-node by adjusting the positions of the nodes which overlaps with an K-node. Figure 2 show an example of our dynamic layout adjustment results. As we can see, constrained layout adjustment nicely reduces node overlap while maintaining the graph topology. Density adjustment is able to pull nodes further apart and further reduce their overlap. Finally, K-node overlapping adjustment dictates that no K-node should overlap any other node. This is to make sure that all keywords are easily readable.

For occluded I-node removal, we calculate for each I-node, the percentage of its pixels overlapped with any other I-node. When the largest percentage of all I-nodes is larger than a given threshold, we simply remove the corresponding I-node from the graph, update the percentages for the remaining I-nodes and repeat this removal process until the largest percentage is smaller than the threshold.

4.3 Graph Transition

To preserve the mental map (i.e., the abstract structural information a user forms by looking at the graph layout) during graph exploration, we provide animated transition from one layout to another by linearly interpolating node positions over the duration of animation. Besides the compound graph, we also provide users with the option to observe the image or keyword subgraph only in a less cluttered view through animated transition.

5. FILTERING, COMPARISON AND RECOMMENDATION

5.1 Interactive Filtering

We provide interactive filtering to users for sifting through the graph to quickly narrow down to images or keywords of interest. For images, users can scan through the entire list of images to identify the ones they would like to add into iGraph. They can also type a keyword to retrieve related images for selection. For keywords, users can scan through the entire list of keywords to identify the ones or type a keyword prefix to find matched ones for selection. For keywords that are already in the graph, we highlight them in red while the rest in the list are in black. For images, we draw them with lower opacity in the list if they are already in the graph. In addition, users can also dynamically adjust the number of images or keywords they would like to display in the current view.

5.2 Node Comparison

Our node comparison allows users to compare nodes of interest for detail comprehension. Similar to the work of PivotPaths,²⁶ we allow users to select or input two to four nodes for comparison. These nodes will be moved to fixed positions around the drawing center. For each selected I-node (K-node), we retrieve n most similar I-nodes (K-nodes) and m related K-nodes (I-nodes) as its group, where n and m are user-defined parameters. Then, we remove the nodes from the previous layout which are not in any group being compared. After that, we apply the modified FR algorithm while fixing the positions of selected nodes, fade in the nodes that are not in the previous layout, and dynamically adjust the layout while the selected nodes are given the freedom to move. The reason is that the layout will be very cluttered in most cases if we always fix the positions for the selected nodes. Meanwhile, the mental map would still preserved since the selected nodes will not change their positions dramatically compared to their previous fixed positions during the last step of the adjustment. We assign different colors to the selected nodes as well as the boundaries of all the rest of nodes to indicate their group memberships.

5.3 Visual Recommendation

To allow exploring iGraph with no prior knowledge, it would be desirable for our system to show only the most relevant portions of the graph to them, while suggesting directions for potential exploration. When a node is selected, we will apply collaborative filtering to recommend related nodes. These recommendations are highlighted in a focus+context manner. User interaction history is saved as the input to collaborative filtering.

Used by Amazon.com, the collaborative filtering²⁷ recommends items to customers based on the item they are currently viewing. We keep an *item-user matrix* \mathbf{N} where the rows are the items (images and keywords in our scenario) and the columns are the users. This item-user matrix records how many times an item has been selected by a user. Different from the collaborative filtering which recommends the items based on all the items that the current user has selected, we only recommend based on item I_j that the user just selected. An array A is created, where each entry A_i records the number of users that selected both items I_i and I_j ($i \neq j$). Then we sort A in the decreasing order and recommend the first n items while n is a user-defined number. We refer to this solution as the *user-centric* approach. Initially, when we either do not have many users or do not have much user exploration history, we mostly make recommendations based on the similarity defined in iGraph. That is, we recommend nodes that are most similar to the node selected. We refer to this solution as the *item-centric* approach. With an increasing number of users using iGraph resulting in richer user exploration history, we are able to gradually shift from the item-centric approach to the user-centric approach and effectively increase the number of items recommended.

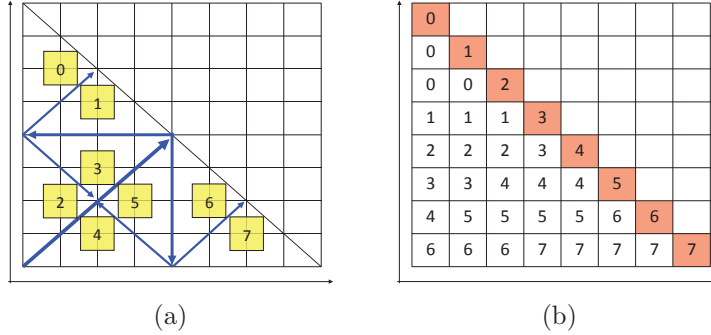


Figure 3. Two different approaches to partition a half matrix, assuming eight GPUs (0 to 7) are used. (a) The first approach partitions the matrix into halves recursively. (b) The second approach works with any number of partitions.

In terms of visualization, rather than visualizing recommended items in a separate view,²⁸ we add recommended nodes to the current node being explored and rearrange the iGraph layout. The end result is that more nodes will show up in the surrounding in a focus+context fashion, and in the meanwhile, we selectively remove some nodes that are less relevant from the layout so that the total number of nodes displayed is maintained as a constant. The criteria to remove less relevant nodes could be nodes which have longest paths from the current node being explored, or nodes which are least recently explored, etc.

We also utilize the image popularity and keyword frequency gathered from our data analysis stage for initial suggestions. That is, popular images will enjoy larger display sizes and frequent keywords will be highlighted with more saturated colors in iGraph, thus directing the user’s attention for more purposeful exploration. User interaction history will also be incorporated to adjust the image and keyword ranking as well as their corresponding visual properties accordingly.

6. PARALLEL ACCELERATION AND DISPLAY WALL

To improve the performance of our approach, we leverage a nine-node GPU cluster along with a display wall employed in the Immersive Visualization Studio (IVS) at Michigan Technological University. The GPU cluster consists of one front-end node and eight computing nodes. The display wall consists of 6×4 thin-bezel 46-inch Samsung monitors, each with 1920×1080 pixels. In total, the display wall can display nearly 50 million pixels simultaneously. These 24 monitors are driven by eight computing nodes for computation and visualization. Each node comes with a quad-core CPU, two NVIDIA GeForce GTX 680 graphics cards, and 32GB of main memory. The aggregated disk storage space is over tens of terabytes for the GPU cluster. We not only use this cluster for visualization, but also for performance improvement of preprocessing and layout generating.

6.1 GPU Parallel Preprocessing

For the computation of grayscale and spectrum image distance matrices, it could still take hours to complete when running on a single GPU. To further improve the performance, we use Open MPI and distribute the workload to several GPUs and perform the computation simultaneously.

There are two ways to evenly partition the symmetric distance matrix. As shown in Figure 3 (a), the first approach iteratively partitions the matrix into halves as indicated by blue lines, and the partition results are indicated with yellow rectangles. This approach achieves a perfectly balanced workload while the number of GPUs must be a power of two. Furthermore, since the triangle of each partition is stored as an array in CUDA and the number of images in a collection is an arbitrary number, it is challenging to combine the resulting triangles to the original half matrix in an efficient way.

Figure 3 (b) illustrates the second approach. Assuming that we need to partition the half matrix into p partitions, we first divide the matrix into p rows and p columns. Then for the p rectangles along the diagonal, each partition gets one. The number of remaining rectangles is $p \times (p - 1) / 2$, which means that each partition gets $(p - 1) / 2$ rectangles. If p is an odd number, then each partition gets the same number of rectangles. Otherwise,

data set	# images	# keywords	# I-I edges	# K-K edges	# I-K edges
APOD	4,560	5,831	10M	17M	137K
MIR Flickr	25,000	19,558	312M	191M	173K

Table 1. The sizes of APOD and MIR Flickr data sets (M: million, K: thousand).

data set	# images	similarity type	block config.	thread config.	loading GPU	computing GPU	speedup factor	saving GPU	# graph nodes	layout CPU	speedup factor
APOD	4,560	grayscale	16,384	512	0.214s	762.312s	-	0.008s	125	0.38s	-
		spectrum	16,384	1,024	0.213s	515.042s	-	0.007s	225	1.29s	-
		histogram	16,384	1,024	0.002s	3.259s	-	0.008s	525	6.96s	-
MIR Flickr	25,000	grayscale	512	512	12.715s	25.722s	-	0.312s	550	10.24s	-
			768	768	3.179s*	3.127s*	8.22	0.088s*	-	0.90s*	11.37
		spectrum	512	512	12.623s	13.380s	-	0.272s	1050	47.60s	-
			768	768	3.163s*	2.151s*	6.22	0.089s*	-	2.07s*	22.99
		histogram	1,024	1,024	0.0296s	0.117s	-	0.274s	2050	5.61s*	-

Table 2. Parameter values and timing results. The GPU’s loading and saving are copying the data from and back to the CPU. The CPU layout time includes the time for workload assignment, data copying, layout computation (150 iterations, this step dominates the timing), and communication of node information. A * indicates the timing with multiple processors (8 GPUs for distance computation, 24 CPUs for layout generation). Otherwise, the timing is on a single processor.

half of the partitions get one more rectangle than the rest. Although the number of images for each partition to load is not perfectly balanced, this approach still achieves a very balanced workload for a large p and it works well with any number of GPUs. In practice, p could be a large number (e.g., a multiple of the number of GPUs) and we can distribute the partitions to each GPU in a round robin manner. Meanwhile, because we store the computation results for each partitioned rectangle, it is easy to index and retrieve the similarity value for any pair of images. We therefore implement this second approach.

6.2 CPU Parallel Graph Layout

We use MPI for the parallel iGraph layout generation using multiple CPUs. The most time-consuming computations are initial backbone iGraph, constrained layout adjustment, and density adjustment. Thus we parallelize these three steps to achieve good performances. The input is the information of all I-nodes and K-nodes (i.e., their initial locations, widths and heights) while the output is the iGraph layout after applying these steps.

For initial backbone iGraph, each node needs to calculate the repulsive force with any other node. After we distribute the graph nodes evenly to the processors, they need to exchange position information in each iteration of the FR algorithm. The initial layout changes dramatically in early iterations and gets more and more stable in later iterations. To reduce the communication cost, we allow the processors to have more frequent communications in early iterations than later ones. For constrained layout adjustment, there are four forces. The bidirectional and unidirectional repulsive forces may occur between any two nodes. Therefore, after we distribute the graph nodes to different processors, they also need to exchange position information in each iteration. Our experience shows that the attractive and spring forces take much more time to compute than the two repulsive forces. As such, when the processors have similar numbers of attractive and spring forces, we consider that workload balance is achieved. Since the numbers of attractive and repulsive forces are related to the degrees of nodes, we sort node degrees in the decreasing order and allocate those graph nodes to different processors in a round robin manner. A processor would not receive any more nodes if the accumulated node degrees reaches the average total node degrees it should have. In this way, all the processors shall have a similar amount of accumulated node degrees. Similar to constrained layout adjustment, in density adjustment, we distribute K-nodes based on their degrees. In each iteration, after calculating the new positions of the K-nodes, the processors exchange position information, update the densities of the triangles, and repeat the process until certain criteria is met.

6.3 Display Wall Visualization

To render iGraph on the display wall, we design four program components: the *master*, *computation*, *relay* and *slave* programs. The master program not only has a user interface to accept all the user interactions but also displays iGraph. It runs on a local computer which is located in the same room as the display wall. This computer captures user interactions, sends instructions to the computation program for generating the graph layout in parallel. After receiving the layout result from the computation program, the master program sends

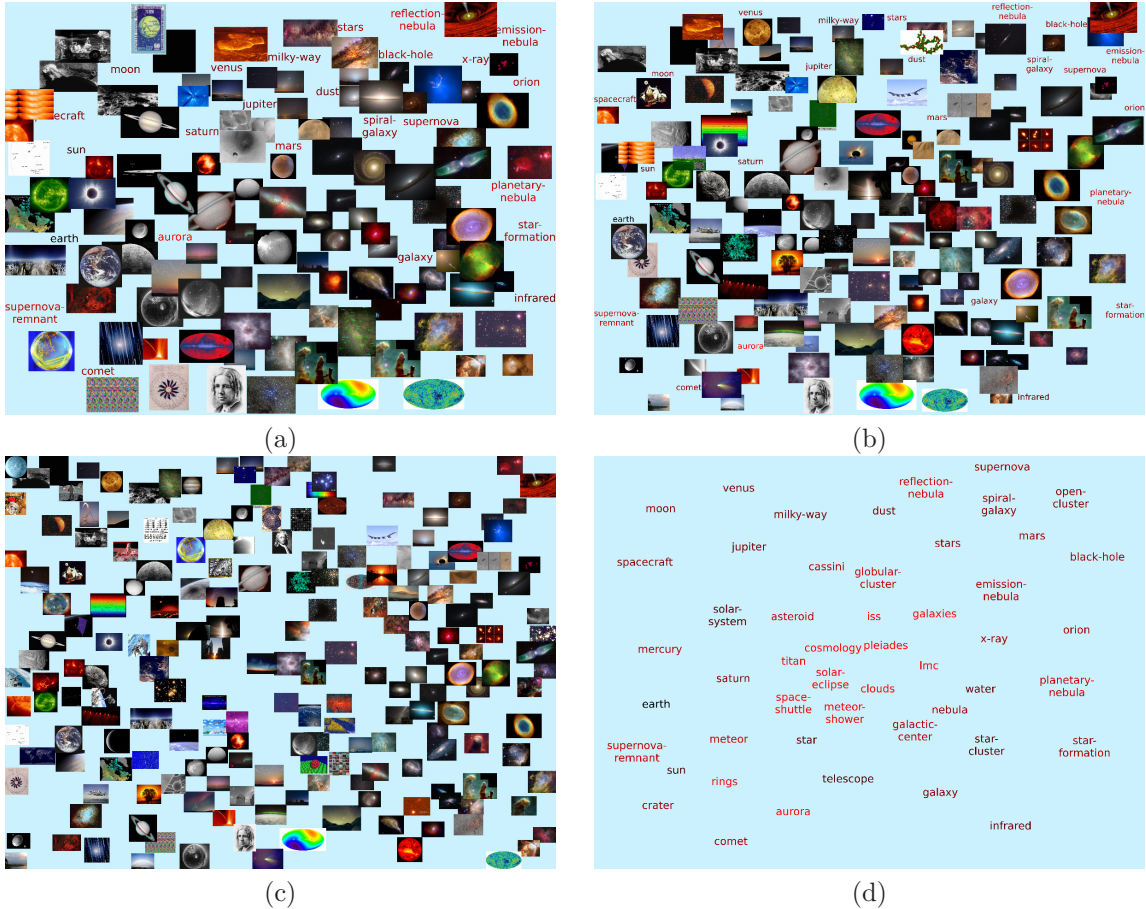


Figure 4. iGraphs with different numbers of I-nodes and K-nodes. For (a) and (b), the numbers of I-nodes and K-nodes are 110 and 25, 190 and 25, respectively. (c) and (d) show the single graphs with 270 I-nodes and 50 K-nodes, respectively.

data to the cluster’s front-end node. The data contains all the information for rendering, e.g., all the I-nodes and K-node positions, widths and heights. Meanwhile, this front-end node running the relay program receives the data and broadcasts it to the eight computing nodes of the cluster. The slave program is an OpenGL application running on these eight nodes. Each node receives the data sent from the relay program, decodes the data, and renders the visualization results to the three tiles that it is responsible for.

For the slave program, there are two problems that we need to address. The first is that the time gaps between neighboring receiving is less than that of decoding so that the data may change undesirably during decoding. The second is the synchronization of receiving, decoding and displaying because the data may change if two of them (i.e., receiving, decoding and displaying) happen simultaneously. To address the first problem, we create a large buffer, use one thread to receive the data and save it to the buffer so that the new coming data will not override the previous data. This also prevents receiving from interrupting decoding and rendering. Furthermore, to solve the synchronization of decoding and rendering, we create another thread to iteratively decode the data and evoke OpenGL to display. In our experiment, we did not encounter inconsistency among the eight computing nodes. Therefore, we do not synchronize the tasks among them.

7. RESULTS

7.1 Data Sets and Performance

In our experiments, we used two data sets, APOD and MIR Flickr. Their data sizes are shown in Table 1. We can see that the connections between I-nodes and K-nodes are less in MIR Flickr than in APOD. On average

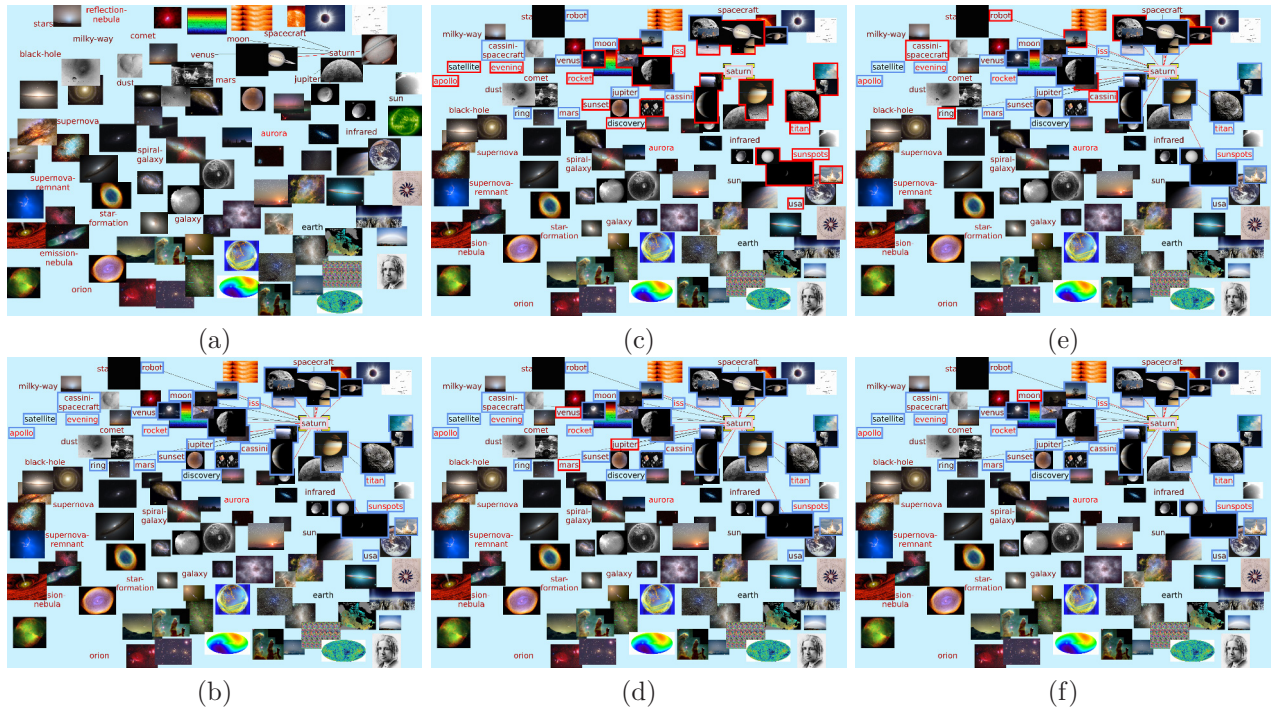


Figure 5. Visual recommendation of the APOD data set. (a) and (b) iGraphs before and after the recommendation (based on K-node “saturn”), respectively. The recommended nodes are highlighted with blue boundaries in (b). The nodes recommended by the user-centric and item-centric approaches are highlighted in (c) and (d), (e) and (f), respectively. Recommended nodes that do not and do show up in the previous iGraph are highlighted with red boundaries in (c) and (e), (d) and (f), respectively.

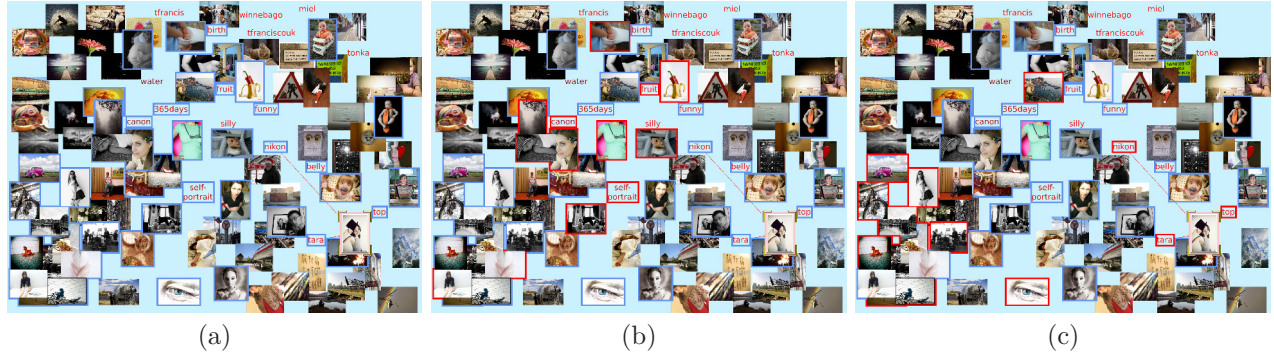


Figure 6. Visual recommendation of the MIR Flickr data set. (a) iGraph after recommendation based on I-node which shows a girl holding her hat (the image is shown in the bottom-right corner and highlighted with the yellow boundary). Recommended nodes are highlighted with blue boundaries. (b) and (c) highlight the nodes recommended by the user-centric and item-centric approaches, respectively. Recommended nodes that do not show up in the previous iGraph are highlighted with red boundaries.

there are only 6.9 K-nodes connected to each I-node in MIR Flickr while 31.1 in APOD. This results in two distinguished groups of I-nodes and K-nodes in the initial iGraph for the MIR Flickr data set. The configurations and timing results for computing image distance matrices are shown in Table 2. The single GPU computation time was reported using a desktop PC (not a cluster node) with an nVIDIA GeForce GTX 580 graphics card, while multiple GPUs computation time was reported using the GPU cluster (Section 6). At run time, all the tasks and interactions are interactive. In the following, we present iGraph results with screenshots captured from our program. For iGraph interaction and its running on the display wall, please refer to the accompanying video.



Figure 7. Node comparison of (a) the APOD data set and (b) the MIR Flickr data set. (a) Three K-nodes “emission nebula”, “reflection nebula”, and “planetary nebula” are chosen for comparison. The nodes with red left bottom corners, cyan right bottom corners, and purple upper boundaries are related to “emission nebula”, “reflection nebula” and “planetary nebula”, respectively. (b) One K-node and two I-nodes are chosen for comparison. The K-node is “self-portrait” and the two I-nodes are images of people. One is a picture of a girl wearing a T-shirt and the other is a picture of a pair of feet.



Figure 8. Photos showing iGraph of the MIR Flickr data set running on the 24-tile display wall (1000 images and 50 keywords are displayed).

7.2 Initial Backbone iGraph and Single Graph

The initial backbone iGraph chooses representative I-nodes and K-nodes to show the overall structure. The representative I-nodes are preselected using affinity propagation, and the representative K-nodes are those with the highest frequencies. Meanwhile, users can choose the number of nodes as shown in Figure 4, where the number of I-nodes increases from 110 to 270 and the number of K-nodes increases from 25 to 50. When the number of nodes increases, to maintain the screen occupancy, the sizes of nodes decrease. However, we set the minimal sizes for I-nodes and K-nodes respectively in order to make the images and keywords readable. Figure 4 (c) and (d) are the single graphs with only I-nodes and K-nodes, respectively. Single graphs are displayed when users want to observe only I-I edge or K-K edge relationships.

7.3 Visual Recommendation

The initial backbone iGraph only gives an overview while our recommendation system serves as a navigation tool to help users in their exploration. The example with the APOD data set is shown in Figure 5. From (c) we know that users who are interested in “saturn” are also interested in “titan”, “rocket”, and “satellite”. In (e), for the keywords with red boundaries, “cassini” and “cassini spacecraft” correspond to the Cassini orbiter which reaches Saturn and its moons in 2004. “ring” means the outer ring of Saturn. For the images with red boundaries, we can see that most of them are the images of Saturn except the one to the left of “iss”. This image is about sunset, but there is a star with label “Saturn” in it.

task	description	summarized question(s)	outcome
T1	Compare the compound graph with the single graphs	Was iGraph helpful in showing the relationship among images and keywords?	5 out of 5
T2	Select the numbers of images and keywords that the user is satisfied for the initial backbone graph	Were the images more useful than the keywords?	4 out of 5
T3	Go over all the images and keywords to get an impression of the APOD data set	Were the functions that allow the user to go through all images (keywords) helpful?	4 out of 5 (5 out of 5)
T4	Evaluate the visual recommendation function with a given example	Was the visual recommendation function helpful with respect to the given example?	5 out of 5
T5	Evaluate the visual recommendation function based on user preferences	Was the visual recommendation function helpful with respect to user preferences?	5 out of 5
T6	Evaluate the node comparison function with a given example	Was the node comparison function helpful with respect to the given example?	5 out of 5
T7	Evaluate the node comparison function based on user preferences	Was the node comparison function helpful with respect to user preferences?	5 out of 5

Table 3. Seven tasks in the user study and users’ response to summarized questions.

Another example with the MIR Flickr data set is shown in Figure 6. In (a), we show iGraph after recommendation based on an image of a girl holding her hat. From (b) we know that the users who are interested in the girl are also interested in “self-portrait”, “canon” and other self-portrait images. In (c), the three keywords “nikon”, “top”, and “tara” are related to the image of the girl.

7.4 Node Comparison

While visual recommendation helps users explore similar nodes or nodes clicked by others, node comparison allows users to choose multiple nodes for detailed comparison. For example, a user interested in the three types of nebulas would choose keywords “emission nebula”, “reflection nebula”, and “planetary nebula”. The comparison result is shown in Figure 7 (a). The images provide the user with a brief impression of what these nebulas look like. The images of nebulas with red left bottom corners are “emission nebula”. Most of them are reddish because they are clouds in high temperature and emit radiation. The images of nebulas with cyan right bottom corners are “reflection nebula”. They are bluish and darker because they are efficient to scatter the blue light. The images of nebula that look like planes and have purple upper boundaries are “planetary nebula”. Meanwhile, the user could also get knowledge from the other keywords. For example, “white dwarf” and “layer” are linked with “planetary nebula” because when an aged red supergiant collapses, the outer layer is the “planetary nebula” while the inside core is the “white dwarf”.

In another example, we choose one keyword “self-portrait” and two images of people. The comparison result is shown in Figure 7 (b). The keyword “365 days” is marked by three different colors, which means that it is related to the K-node and two I-nodes chosen. “me”, “ofme”, and “self” are highlighted with both red and cyan boundaries which means that they are related to the keyword “self-portrait” and the image showing a pair of feet. Keywords “red shoes” and “square” with cyan boundaries indicate that there is a pair of shoes in the picture and the photo was taken at a square.

7.5 Running on Display Wall

Figure 8 shows three iGraph photos of the MIR Flickr data set rendered on the display wall. With the display wall, we are able to display thousands of images and keywords simultaneously for comfortable viewing. Currently, we are using this display wall for showing iGraph demos to visitors, including university alumni, visitors, and summer students. Initial feedback from several groups of visitor is fairly positive as they comment that running iGraph on this life-size tiled display is much more expressive and fun to watch compared with on a regular desktop display. The advantage of using the display wall is that it allows more than a dozen of people to comfortably view and discuss the results together in such a collaborative environment. Nevertheless, with the dramatic expanding of display area, it takes more effort for a viewer to correlate and compare images that are on the opposite sides of the display wall, especially for those images close to the wall’s boundary.

8. USER EVALUATION

We recruited five unpaid researchers in astronomy for a user study of iGraph on a PC: one professor, three PhD students and one master student. Four of them are researchers in physics studying astronomy and the remaining

one is a computer science PhD student with a master degree of physics focusing on astronomy. The user study was conducted in a lab using the same PC. The PC has a 27-inch monitor with 1920×1080 resolution, where the visualization result occupied an area of 1600×900 . Users were first introduced to iGraph about its goal and main functions. Then they were given the APOD data set for free exploration to get familiar with the system.

The iGraph survey includes seven tasks and five general questions. The seven tasks (T1 to T7) were designed to evaluate the effectiveness of iGraph. T1 and T2 asked the user to compare the compound graph with the single graphs and select the numbers of images and keywords that she was satisfied with for the initial backbone graph. T3 asked the user to go over all the images and keywords to get an impression of the APOD data set. T4 and T5 asked the user to evaluate the recommendation function with a given example and based on her preference, respectively. T6 and T7 asked the user to evaluate the comparison function with a given example and based on her preference, respectively. Every task has several survey questions and write-in questions except T2 which consists of only write-in questions. All the survey questions have the same set of choices on a 5-point Likert scale: 1: strongly disagree, 2: disagree, 3: neutral, 4: agree, and 5: strongly agree.

We summarized the questions of the seven tasks as shown in Table 3. For simplicity, if the user rated the question as 4 or 5, we consider that she agreed; if she rated it as 1 or 2, she disagreed; otherwise, she was neutral. We used the number of users that agreed to measure the effectiveness of iGraph functions.

Users could perform the tasks whenever they felt comfortable. Each study took about one hour to complete. Although each task could be performed within a few minutes, users frequently returned to the interface for further verification when writing their comments, which took most of the time. Users were informed that the tasks were not timed and their comments were of crucial importance. Among these tasks, two of them are related to the free explorations of the data set with recommendation and comparison functions. Each of these two tasks took them a little more time till they were ready to write comments. The outcome of the user study is summarized in Table 3.

Next we provide further explanation on these outcomes. For T1 and T2, we asked users to shift freely between the single graphs and compound graph. In addition, they were asked to change the numbers of images and keywords displayed on the screen in order to get an impression of the best ratio between images and keywords. In terms of T1, the feedback was positive. All users stated that iGraph was helpful to show the relationships among images and keywords in both compound and single graphs. Furthermore, a compound graph was considered better than a single graph. The animation of changing between compound and single graphs was helpful to gain an impression of node changes. In terms of T2, one PhD student preferred more keywords than images because the keywords were easier to understand and intuitive to query. The rest of users indicated that since images were more attractive, the ratio between the numbers of images and keywords should be within the range of $3 : 1$ to $10 : 1$.

For T3, we asked users to go through all the images and keywords to gain an impression of the APOD collection. All the images were shown in the order of their published dates and all the keywords were sorted based on their frequencies. Four users considered the functions that allow the user to go through all images were helpful to gain an overall impression of the data set and to select images of interest. The remaining user was neutral on this since she was not familiar with all images and viewing individual images did not make much sense to her. For better browsing of the collection, we can group images based on their similarities and show them one group at a time. Moreover, all users indicated that going through all keywords was helpful. Three agreed that iGraph could be helpful to identify popular and rare keywords while two were neutral on it. This was probably due to the equal weights of the keywords extracted from the explanation and keyword sections. Since the number of keywords extracted from the explanation section is larger than that from the keyword section, equal weights lead to the reduced importance of the keywords from the keyword section. Finally, all users agreed that color and opacity encoding was helpful to identify I-nodes and K-nodes already in iGraph so that they could select new nodes into iGraph for recommendation and comparison.

For T4, users were asked to type keyword "earth" for recommendation and see the images and keywords recommended using different approaches. For T5, users could select any I-node or K-node from iGraph for recommendation and repeat this process as long as they wanted to. All users agreed that the recommendation function was useful, especially for the item-centric approach because it recommended more relevant nodes than

the user-centric approach. However, we believe that the user-centric approach will recommend more relevant nodes when the number of users increases to a certain level. The functions that allow the user to add any image and keyword made the recommendation function more effective. They also believed that the highlighted boundaries were helpful to identify the recommended nodes.

For T6, users were asked to select an image related to “jupiter” and the keyword “moon” for comparison. For T7, users could select up to four nodes to compare, and then select any image or keyword in the collection for further exploration. All users agreed that the comparison function helped them find the nodes related to one or multiple selected nodes. Three of them could further dig out the underlying relationships of the selected nodes while two could not. This was because we linked both images and keywords to the selected nodes to show their relationships, which was not a direct approach. All of them agreed that the highlighted boundaries could help identify the belonging relationships of the nodes. Two of them thought that the animation could not demonstrate node changes. This was because when users selected multiple nodes for comparison, most of the nodes after comparison did not appear before. Thus, the animation only gave them the impression of layout change.

For the general questions, two users mentioned that iGraph could not only support researchers to find the underlying relationships of the APOD data set but also help teachers teach in class. Besides that, they stated that iGraph could be extended to handle other image-text collections (such as movie collections) and to assist in image search. Users also pointed out some possible improvements for iGraph. First, the techniques related to object identification could be helpful for measuring image similarity. Second, they suggested us to provide more sorting options while going through the APOD collections, e.g., sorting according to the image colors, alphabetical order of keywords, or user preference. Third, they also mentioned that we could provide more flexible query functions. Finally, one suggested that we should show the detailed information of a selected image.

9. CONCLUSIONS AND FUTURE WORK

We have presented iGraph, a new approach to visualize large image and text collections with essential capabilities including interactive filtering, node comparison, and visual recommendation. The general approach proposed for iGraph is applicable to many other large graph drawing, visualization, and exploration applications. We leverage multiple GPUs for preprocessing and multiple CPUs for runtime graph layout generation and visualization. Our user evaluation shows that iGraph is a promising approach for navigating and viewing large image and text collections. In the future, we will develop an online version of iGraph to benefit a wider user community.

ACKNOWLEDGMENTS

This research was supported in part by the U.S. National Science Foundation through grants IIS-1017935, IIS-1456763, and IIS-1455886. We thank the anonymous reviewers for their helpful comments. We also thank Mark Huiskes, Bart Thomee, and Michael Lew for making the MIR Flickr collection available. All the images at the APOD website are credited to the owners or institutions where they originated.

REFERENCES

- [1] Nemiroff, R. J. and Bonnell, J. T., “Astronomy picture of the day: <http://antwrp.gsfc.nasa.gov/apod/astropix.html>,” in [*Bulletin of the American Astronomical Society*], 1291 (1995).
- [2] Huiskes, M. J. and Lew, M. S., “The MIR Flickr retrieval evaluation,” in [*Proceedings of ACM International Conference on Multimedia Information Retrieval*], 39–43 (2008).
- [3] Bederson, B. B., “PhotoMesa: A zoomable image browser using quantum treemaps and bubblemaps,” in [*Proceedings of ACM Symposium on User Interface Software and Technology*], 71–90 (2001).
- [4] Platt, J. C., Czerwinski, M., and Field, B. A., “PhotoTOC: Automatic clustering for browsing personal photographs,” in [*Proceedings of IEEE Pacific Rim Conference on Multimedia*], 6–10 (2003).
- [5] Torres, R., Silva, C., Medeiros, C., and Rocha, H., “Visual structures for image browsing,” in [*Proceedings of International Conference on Information and Knowledge Management*], 49–55 (2003).
- [6] Jankun-Kelly, T. J. and Ma, K.-L., “MoireGraphs: Radial focus+context visualization and interaction for graphs with visual nodes,” in [*Proceedings of IEEE Information Visualization Symposium*], 59–66 (2003).

- [7] Yang, J., Fan, J., Hubball, D., Gao, Y., Luo, H., Ribarsky, W., and Ward, M., “Semantic image browser: Bridging information visualization with automated intelligent image analysis,” in [*Proceedings of IEEE Symposium on Visual Analytics Science and Technology*], 191–198 (2006).
- [8] Brivio, P., Tarini, M., and Cignoni, P., “Browsing large image datasets through Voronoi diagrams,” *IEEE Transactions on Visualization and Computer Graphics* **16**(6), 1261–1270 (2010).
- [9] Viégas, F. B., Wattenberg, M., and Feinberg, J., “Participatory visualization with Wordle,” *IEEE Transactions on Visualization and Computer Graphics* **15**(6), 1137–1144 (2009).
- [10] Wei, F., Liu, S., Song, Y., Pan, S., Zhou, M. X., Qian, W., Shi, L., Tan, L., and Zhang, Q., “TIARA: A visual exploratory text analytic system,” in [*Proceedings of ACM SIGKDD Conference*], 153–162 (2010).
- [11] Cui, W., Wu, Y., Liu, S., Wei, F., Zhou, M. X., and Qu, H., “Context preserving dynamic word cloud visualization,” in [*Proceedings of IEEE Pacific Visualization Symposium*], 121–128 (2010).
- [12] Díaz, J., Petit, J., and Serna, M., “A survey of graph layout problems,” *ACM Computing Surveys* **34**(3), 313–356 (2002).
- [13] Hachul, S. and Jünger, M., “An experimental comparison of fast algorithms for drawing general large graphs,” in [*Proceedings of International Symposium on Graph Drawing*], 235–250 (2006).
- [14] Herman, I., Melancon, G., and Marshall, M. S., “Graph visualization and navigation in information visualization: A survey,” *IEEE Transactions on Visualization and Computer Graphics* **6**(1), 24–43 (2000).
- [15] Hadlak, S., Schulz, H.-J., and Schumann, H., “In situ exploration of large dynamic networks,” *IEEE Transactions on Visualization and Computer Graphics* **17**(12), 2334–2343 (2011).
- [16] Feng, K.-C., Wang, C., Shen, H.-W., and Lee, T.-Y., “Coherent time-varying graph drawing with multi-focus+context interaction,” *IEEE Transactions on Visualization and Computer Graphics* **18**(8), 1330–1342 (2012).
- [17] Zinsmaier, M., Brandes, U., Deussen, O., and Strobel, H., “Interactive level-of-detail rendering of large graphs,” *IEEE Transactions on Visualization and Computer Graphics* **18**(12), 2486–2495 (2012).
- [18] Frishman, Y. and Tal, A., “Multi-level graph layout on the GPU,” *IEEE Transactions on Visualization and Computer Graphics* **13**(6), 1310–1317 (2007).
- [19] Alper, B., Höllerer, T., Kuchera-Morin, J., and Forbes, A., “Stereoscopic highlighting: 2D graph visualization on stereo displays,” *IEEE Transactions on Visualization and Computer Graphics* **17**(12), 2325–2333 (2011).
- [20] Wang, C., Reese, J. P., Zhang, H., Tao, J., and Nemiroff, R. J., “iMap: A stable layout for navigating large image collections with embedded search,” in [*Proceedings of IS&T/SPIE Conference on Visualization and Data Analysis*], (2013).
- [21] Cilibrasi, R. L. and Vitányi, P. M. B., “The Google similarity distance,” *IEEE Transactions on Knowledge and Data Engineering* **19**(3), 370–383 (2007).
- [22] Shneiderman, B., “The eyes have it: A task by data type and taxonomy for information visualizations,” in [*IEEE Symposium on Visual Languages*], 336–343 (1996).
- [23] Frey, B. J. and Dueck, D., “Clustering by passing messages between data points,” *Science* **315**(5814), 972–976 (2007).
- [24] Fruchterman, T. M. J. and Reingold, E. M., “Graph drawing by force-directed placement,” *Software - Practice and Experience* **21**(11), 1129–1164 (1991).
- [25] Shewchuk, J. R., “Triangle: Engineering a 2D quality mesh generator and delaunay triangulator,” in [*Proceedings of ACM Workshop on Applied Computational Geometry*], 203–222 (1996).
- [26] Dörk, M., Riche, N. H., Ramos, G., and Dumais, S., “PivotPaths: Strolling through faceted information spaces,” *IEEE Transactions on Visualization and Computer Graphics* **18**(12), 2709–2718 (2012).
- [27] Linden, G., Smith, B., and York, J., “Amazon.com recommendations: Item-to-item collaborative filtering,” *IEEE Internet Computing* **7**(1), 76–80 (2003).
- [28] Crnovrsanin, T., Liao, I., Wu, Y., and Ma, K.-L., “Visual recommendations for network navigation,” *Computer Graphics Forum* **30**(3), 1081–1090 (2011).