Noname manuscript No.
(will be inserted by the editor)

Style-Based Exploration of Illustration Datasets

Elena Garces · Aseem Agarwala · Aaron Hertzmann · Diego Gutierrez

Received: date / Accepted: date

Abstract Searching by style in illustration data sets is a particular problem in Information Retrieval which has received little attention so far. One of its main problems is that the perception of style is highly subjective, which makes labeling styles a very difficult task. Despite being difficult to predict computationally, certain properties such as colorfulness, line style or shading can be successfully captured by existing style metrics. However, there is little knowledge about how we distinguish between different styles and how these metrics can be used to guide users in style-based interactions. In this paper, we propose several contributions towards a better comprehension of illustration style and its usefulness for data exploration and retrieval. First, we provide new insights about how we perceive style in illustration. Second, we evaluate a handmade style clustering of clip art pieces with an existing style metric to analyze how this metric aligns with expert knowledge. Finally, we propose a method for efficient navigation and exploration of large clip art data sets which takes into account both semantic labeling of the data and its style. Our approach combines hierarchical clustering with dimensionality reduction techniques, and strategic sampling to obtain intuitive visualizations and useful visualizations.

Keywords illustration · style · exploration · visualization

Elena Garces, Diego Gutierrez
IIS Department - University of Zaragoza, Spain
E-mail: {egarces,diegog}@unizar.es

Aseem Agawala
Google Research, Seattle, US
E-mail: aseem@agarwala.org

Aaron Hertzmann
Adobe Research, San Francisco, US
E-mail: hertzman@adobe.com

1 Introduction

The amount of visual information available online has increased dramatically during the last years. In particular, clip art collections contain massive amounts of images which are usually classified by content. While a semantic classification is undoubtedly needed for searching tasks, a style-based exploration might result extremely helpful in certain situations, for example, to create visually coherent presentations, web content, or any kind of graphic design. Occasionally, we find these images labeled by the designer, although the number of images in each group is usually very small and they cover a specific subject e.g. animals, food,... Having unambiguous style labels would be of great help in situations where we need to explore hundreds of images from multiple collections. The problem is that the subjectivity in the perception of style makes finding this labeling a very difficult task. The *style metric* of Garces et al. [8] contributed to quantify certain properties of style, such as line properties or shading, however there is little knowledge about how we distinguish between different styles and how these metrics can be used to guide users in style-based interactions.

In this work, we aim to obtain an efficient method to explore large collection of clip art data sets by style. First, with the purpose of better understanding the perception of illustration style and its correlation with the metric, we perform a user study in which we gain new insights on how do people identify similar styles; then, we evaluate the metric using a hand labeled data set. Finally, we propose an exploratory interface which allows users to combine images from multiple collections and to identify the most common styles at first glance. To do so, we leverage the existing metric to automatically discover the implicit style hierarchies existing in such data sets by means of hierarchical clustering and dimensionality reduction techniques.

The paper is structured as follows: in Section 2 we present the related work; in Section 3 we briefly describe the style metric of Garces et al. [8] and present our study about how do people identify similar styles. In Section 4 we perform the analysis of a labeled data set. The exploratory interface and results are described in Sections 5 and 6.

2 Related Work

Style Analysis. The analysis of artistic style has received much less attention than stylistic rendering. A common way to algorithmically capture the style elements is to learn a *generative* model; that is, a model that learns to create new examples of a style from scratch, such as generating new typefaces [32, 3], and learning 3D hatching styles from examples [13]. Another type of approaches *transfer styles* from examples, such as transferring painting styles [10], photographic styles [2], or curve styles of 2D shapes [17].

Willats and Durand [34] provide an overview of the elements of pictorial style in 2D illustrations. Recently, a computational set of low level fea-

tures which capture style has been developed for illustration [8] and fonts [24]. Several methods provide style similarity metrics for 3D shapes [9,18,19], clothes [35] or infographics [28]. These works provide a distance metric which allow content retrieval based on style, and that can be used in conjunction with any existing algorithm which requires a similarity measure. In particular, in this work we leverage the work of Garces et al. [8], to obtain aesthetically coherent clusters which are exploited to provide meaningfully visualizations of clip art datasets.

Exploration and Visualization of Data Collections. The huge amount of datasets available online has pointed out the necessity of new tools to explore its content in intuitive ways. In particular, the problem of exploring and browsing 3D shapes is currently a hot topic of study. Kleiman et al. [14] introduce the idea of dynamic maps to provide smooth navigation between the elements of 3D datasets. Averkiou et al. [1] propose a combined approach between exploration and synthesis of 3D shapes. A related approach to ours is the work of Huang et al. [11], which provide a method for organizing heterogeneous 3D shape collections based on different distance measures between shapes. They additionally study several tree-based hierarchies to present the data. However, none of these approaches explore the dimensionality of style.

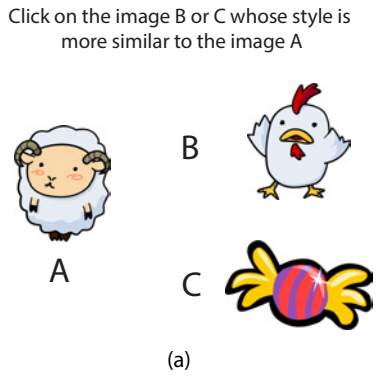
Artistic visualizations of illustration collections have been proposed in the context of packing layouts [26], although unlike us they do not take into account semantic labeling, and just focus on optimal arrangements for fixed layouts. Modeling and navigation through color spaces was introduced by Shapira et al. [30], which fit a Gaussian Mixture Model to the pixel colors of the image. Visualizing high dimensional spaces in two dimensions has been done before for sketches [5], color palettes [23] and 3D models [31].

3 Analysis of Style in Illustration

In this section we first briefly review the style similarity metric of Garces et al. [8], and then we present our study of how users identify style in illustration.

3.1 Style Similarity Metric

Thanks to the *style similarity* metric learned by Garces et al. [8], we can compute a real-valued measure of the similarity in style of two input pieces of clip art, independent of semantics or content. In the original work, the authors demonstrate the usefulness of the metric in search-by-style operations: given a particular query clip art, the search results appear sorted by style similarity. The metric was computed by combining crowdsourcing and machine learning. First, they modeled each clip art image as a high-dimensional feature vector which captures four aspects of style: *color*, *shading*, *texture* and *stroke* [4]. Then, through Amazon Mechanical Turk (MTurk), a set of users were presented with clip art triplets, and were asked the question: "Is clip art A more



(a)

Attribute	Reason	ID
Contour	No contour	1
	Thick line	2
	Thin line	3
	Irregular/broken line	4
Color	Black&White	5
	Monochromatic or very few colors	6
	Colourful	7
	Similar color saturation	8
Shading	Smooth	9
	Sharp transitions between colors	10
Shape	3D	11
	Flat or two-dimensional appearance	12
	Simple form: few detail	13
	Complex form: lot of detail	14
None	Other reasons	15

(b)

Fig. 1 (a) Example of question used to capture style similarity data. Each test (HIT) on Amazon Mechanical Turk contained fifty questions of this kind. (b) List of the available reasons offered to people to choose from.

similar in style to clip art B or C?". Figure 1 (a) shows an example of question of such a type. Last, using the feature vector and the relative comparisons collected via MTurk, the style similarity metric, $\mathsf{d_s}$, was learned by computing the Euclidean distance metric [16,29] (please refer to the original paper for extended details).

3.2 How do people identify similar styles?

While the *style metric* $\mathsf{d_s}$works reasonably well for style-based image retrieval operations, the authors do not offer any insights about what attributes people consider more important when judging if two styles are similar, and thus, the metric could be skipping relevant properties. In this work, we directly asked people what they look at when comparing two pieces by style.

We followed the same MTurk-based methodology as the original work to obtain relative comparisons (see Figure 1 (a)), and additionally, we gathered information on the reasons to select one result over another in the performed comparisons. Thus, during the test, each participant would occasionally be asked to choose one or several items out of a proposed set of reasons for picking a result [27]. This questionnaire appeared randomly with a probability of 1 in 10 in each test composed by 50 relative comparisons. Figure 1 (b) shows the complete list of proposed reasons grouped by style attribute. In total, 294 people took part on the experiments, where 83 had none artistic experience, 190 had some experience, and 21 were professional artists. We collected 2654 questionnaires of this kind, and grouped the responses by user. We count the number of times a user selects a reason and normalize that number by the total amount of answers per user.

The analysis of the answers is shown in Figure 2. It can be seen that the dominant high level attributes that people notice are color and shape. Among

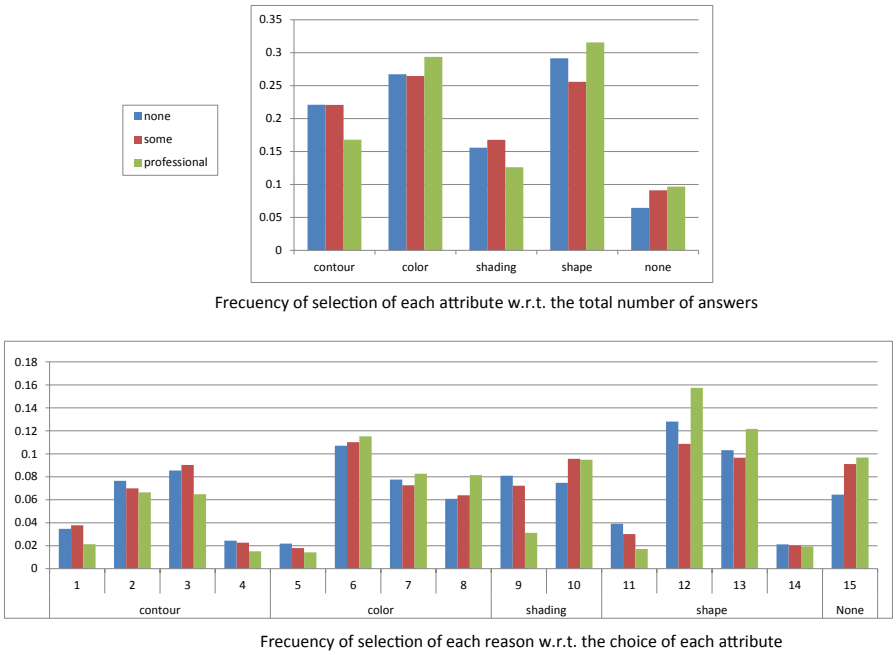


Fig. 2 Summary of the results from the MTurk questionnaire. Top, cummulative number of selections per attribute, normalized with respect to the total number of answers. Bottom, frequency of occurrence of each reason normalized with respect to the total number of answers per attribute. In both plots, users are separated by artistic experience.

color criteria, the most frequent reason is the presence of a dominant single color, over other reasons like color saturation. Many users identified *shape* as a very important attribute, which is interesting because the style metric $\mathsf{d_s}$do not include *shape* in the computed properties. Given that distinguishing between a 3D and a 2D shape is a high level vision task [7], this finding opens new avenues of future work and a big insight about how our visual system discards other style inputs such as shading to favor shape instead. Finally, we did not find any correlation between the artistic background of the users and the particular choices they made, suggesting that we may not need training to perceive style in this kind of artworks. The reason might be that the style of these pieces of art is very well defined and thus, our visual system finds it easy to discern styles with such a high level of stylization.

4 Analyzing a Labeled Dataset

In order to analyze how the style metric $\mathsf{d_s}$aligns with expert knowledge, we use the data set labelled by style from the Microsoft Office library (now discontinued) which, to our knowledge, is the only free clip art data set labeled according to style. We collected a total of 3024 clip art images of 220 differ-

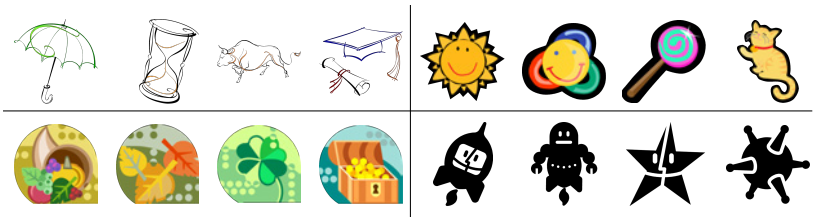


Fig. 3 Example of labeled styles from the MS data set. Top-left style is defined by sketchy strokes without fill. Top-right and bottom-left are more colorful styles, the first with black stroke and the second without it. Bottom-right is defined by rounded shapes with black fill and no colors.

ent styles, where the average number of images per style, or style-cluster, is sixteen. We can see in Figure 3 a few examples of the initial classification as labeled in the data set, which shows that in principle the labeling makes a good job at discerning styles, with style variations within the same cluster being minimal. An alternative option to use the metric $\mathsf{d_s}$, could have been to use this labeled data set of 3024 images to train a new metric [36,12,20]. However, this subset of data only represent a 1.5% of the total amount of data used to train $\mathsf{d_s}$, which was 200k images. Thus, it is expected that $\mathsf{d_s}$will generalize better for any other set of images and styles.

4.1 Ranking Evaluation

In order to evaluate the metric using the labeled data set, and, since the style metric was originally designed for searching purposes, we decided to evaluate the quality of the *ranking* returned after a search operation. That is, for each image of the dataset, which we call the query, we let the metric compute the distances to the rest of the clip art images and rank them from low to high values. At the top of the resulting ranking we expect to find elements with the same style label as the query (true positives) and almost no elements with different labels (false positives); as we traverse the ranked list further down, we expect an increasing number of false positives. To do this, we employ several ranking metrics, which have been adapted to handle labeled data [21]:

AUC Measures the Area Under the ROC Curve. For binary classification problems, the ROC curve measures the amount of false positives against true positives. In our case, this value is computed counting the number of items returned. This metric is position independent, so, an incorrect item at the bottom of the list counts as much as an incorrect item at the beginning.

Precision-at-k (Prec@k) Measures the fraction of relevant results out of the first k returned. This measure is specially relevant when only the first few results matter, as in web browsing or clip art search applications.

Mean Average Precision (MAP) Precision-at-k score of a ranking, averaged over all positions k of relevant items.

	BASELINE	LEARNED WEIGHTS
	[Garces 2014]	
AUC	0.936	0.944
MAP	0.365	0.399
MRR	0.765	0.779
Prec@k=3	0.605	0.628
Prec@k=10	0.470	0.499
NDCG@k=3	0.614	0.638
NDCG@k=10	0.518	0.545

Fig. 4 Ranking measures between the unweighted features (Baseline) and the style similarity metric [8]

Mean Reciprocal Rank (MRR) Inverse position of the first relevant item in the ranking.

Normalized Discounted Cumulative Gain (NDCG) Extension of the MRR metric. In this case all of the top k items are scored at a decaying discount factor.

Figure 4 shows the results of these ranking metrics on the initial feature set of Garces el al. [8] with uniform weights (*baseline*), and with the *style metric* $\mathsf{d_s}$. A value of one means a perfect score. As expected, the results are always better for the learned weights and improve when the ordering of the result does not matter, as is the case of AUC and MRR. We can also observe that increasing the value of k for Prec@k and NDCG@k decrease the quality of the results, introducing more false positives. In general, these values are relatively low except for AUC, which does not take into account the relative ordering of the elements and thus, performs quite well. As we will see in the following, the huge similarity between the styles of the hand-made labels has heavily penalized these metrics.

Figure 5 shows some of the ranking results where the first item on the left is the query image and the other nine items are the most similar images according to the *style metric*. The images that do not belong to the same cluster as the query (as originally labeled) are highlighted in red. At first glance, we could say that the metric is performing poorly, however, a closer visual inspection of these elements reveals that the styles of the retrieved images are quite coherent. This indicates that many of the images initially labeled as different styles are actually very close in the feature space learned by the metric; and thus, they are *perceived* as similar styles. Although the *intra-cluster* distance is very small, which is a desired property, the *inter-cluster* distance is also sometimes very small. A disjoint labeling like this one do not capture the overall styles available in the data set and thus, would hinder style-based navigation task. In the following section, we propose a exploratory interface based on clustering which will allow to overcome this limitation by showing the most frequent styles available in the data set.

Fig. 5 Ranking results. The first image on the left is the query image, the remaining images are sorted by lower to high distance to it. We have used the style metric of Garces et al. [8] to obtain the distances.

5 Style-Based Exploration of Illustration Datasets

As we have seen in the previous section, we may find several occasions where images of the same style do not necessarily have the same style label, either because they come from different collections/artists or because the labeling is not accurate. Therefore, one of the problems that we face when trying to find elements in these collections is the prior lack of awareness of the available styles in the data set. This is specially problematic when we need to select more than one images stylistically coherent i.e. we may find that the requested images are not depicted in the desired style.

Suppose a user needs to find in certain data set several images that match in style. Each image has to belong to a different semantic category, e.g., a dog, a cat and a tree. We denote as $\mathcal{L}$ the total set of categories or semantic *labels* requested by the user, where in this example $\mathcal{L} = \{dog, cat, tree\}$. In a typical workflow, the user would start querying the data set sequentially by category watching that the style of all the images matches. We can make this task easier by sorting the retrieved results by style similarity to the previously selected items using the metric $\mathsf{d_s}$. In this second scenario, the more delicate step is to select the first image, since its style will define the order in which the remaining queries are sorted. The problem is that if the style chosen in the first place is unusual in the data set, the user may end up disappointed for not having enough stylistically similar images of all the requested categories. We propose a solution to this problem in which the user knows in advance the amount of feasible visual styles which are available in the data set. Our

approach combines unsupervised clustering techniques along with adaptive dimensionality reduction and mapping to sample and visualize the images of these huge collections in an efficient and practical interface.

Given a collection of semantically labeled images, which was previously filtered and labeled by the set of semantic categories $\mathcal{L}$ specified by the user, we first perform hierarchical clustering over the filtered collection. Since visualizing the resulting tree may be impractical, we propose two complementary visualizations. On the one hand, we visualize only the top level nodes of the resulting hierarchy in the form of a taxonomy of styles (Section 5.1). This gives us the most frequent styles in the set. On the other hand, we use dimensionality reduction techniques with adaptive relocation to visualize in two dimensions the high dimensional space of the style features, allowing the user to navigate through such space (Section 5.2).

5.1 Creating a Style Taxonomy

In order to facilitate data exploration and obtain meaningful taxonomies of the style elements, we have chosen an approach based on hierarchical clustering. Following a bottom up approach, each element initially starts on its own cluster, and, on successive iterations, clusters are merged according to a chosen criteria. In this work, we have chosen Ward's criteria [33], for which, at each step of the algorithm, the pair of clusters with the minimum variance within-cluster are merged. A typical problem with clustering is to choose the appropriate distance metric that captures the underlying relationships between the elements. In our case, we rely on the style metric $\mathsf{d_s}$, which comprises users' knowledge for style discrimination, and has been extensively evaluated.

From this step we obtain a hierarchical tree, which has two types of nodes: intermediate nodes and leaf nodes. Leaf nodes are the lower level elements of the hierarchy and are represented by their corresponding images. Intermediate nodes define *branches* of style and may contain leaf nodes and other intermediate nodes. To select the representative image of an intermediate node, we choose the leaf-node image with the closer distance to its centroid in the metric space of $\mathsf{d_s}$.

Representative Styles In huge data sets it may not be possible to show all the hierarchy in one view. Therefore, we need to develop a strategy to sample the hierarchy and select a representative number of intermediate nodes so that we show as much information as possible about the available styles. Our solution consists of pruning the branches that do not satisfy the following conditions: 1) the total number of images of the branch is above a certain number κ; 2) the number of images of each semantic category is greater than a value τ . The first criteria balances the taxonomy so that all the nodes contain the same number of images, the second criteria allows to discriminate feasible paths. If a certain node does not meet these two criteria, we remove the node and its branch. We then take the resulting leaf nodes of the pruned

hierarchy as representative styles. By changing the values of the thresholds we control the size of the resulting hierarchy and guarantee to capture with a few nodes the most frequent styles of the data set.

To illustrate the process, we applied it to a set of 3024 images of a wide variety of styles from the Microsoft Office online libraries. We show in Figure 6 the top level nodes of the resulting hierarchy after the pruning and the representative styles. We observe that the most dominant style has colorful images without contour -five representative nodes in the middle capture this style. On the contrary, the less frequent style is compound by black and white images- only one node capture this style. Depending on the amount of data that we want to visualize, we vary this threshold. In Figure 6, we want every representative style to have 10% of the total data ignoring semantic labeling, so $\kappa = 0.1 \cdot 3024$ and $\tau = \kappa$.

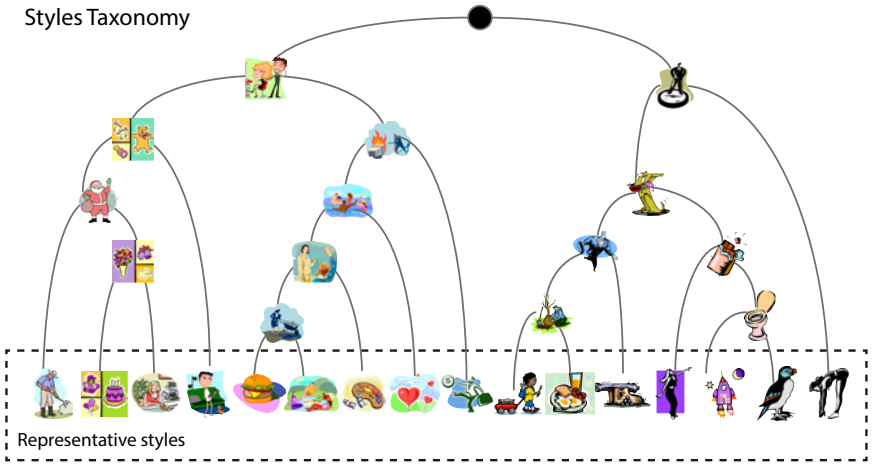


Fig. 6 Top level nodes of the style taxonomy with $\kappa = 0.1 \cdot 3024$. Each of the representative style branch contains around 10% of the total data. Note that black and white style (bottom-right node) is less frequent in the data set than colorful, stroke-less style (nodes five to nine in the middle).

5.2 2D Exploration: Adaptive arrangement

The style taxonomy resulting from the previous step makes it possible to visualize at high level the diversity of styles available in the data set. Each leaf node of the taxonomy represents a *style branch*, which the user can further explore to find the desired icon sets. Since each of these branches might contain too many images to visualize as is, we propose a two dimensional grid-based visualization where Euclidean distances between the images in the grid are

equivalent to distances in the perceptual metric space of $\mathsf{d_s}$. The optimal arrangement needs to satisfy the following conditions:

1) Fixed layout and number of slots. If the selected branch has more images than available slots in the grid, some images will need to remain hidden in the main view. According to this condition, interface designers can select the optimal size of the grid to fit their interface and data requirements.
2) Balanced number of categories. If the user requests more than one semantic category or label, the amount of *visible* images for each label should be approximately the same. The task of finding the requested set of images is easier if the user can visualize most of the images at first glance.
3) Perceptual distances. Images with similar style should remain close to each other in the two-dimensional arrangement. This kind of arrangement makes it easier the exploration of big sets of images which may contain multiple style variations.

With the above conditions, it is unsuitable to use existing methods of layout generation [6,22]. They do not provide support for fixed layouts, where there are more images than slots (condition 1), and do not take into account the balance of the labels (condition 2). Thus, our goal is to optimally place in a grid the set of images from a style branch according to the three previous conditions. We propose a greedy algorithm which keeps balanced the number of displayed labels, maintains the perceptual distances in style in the grid, and takes into account special situations where there are more images than positions in the grid, i.e., situations in which one position in the grid may contain multiple images.

Problem Formulation We consider a regular grid $G = \{X\}$ of size $M \times N$, the set of images of a branch $\mathcal{I}$ which we want to arrange in G, and a set of semantic labels $\mathcal{L}$. Each image $i \in \mathcal{I}$ has a label from $\mathcal{L}$ associated, and the function label(i) returns it. Each node $x \in G$ defines a tuple $x = (i, l, \mathbf{h}, s)$, where: $x^i \in \mathcal{I}$ is the image chosen to represent the node; $x^l \in \mathcal{L}$ is the semantic label of such image; $x^{\mathbf{h}}$ is the list of candidate images assigned to the node (which excludes x^i) and $x^{\mathbf{h}(k)}$ denote the kth image in such vector; x^s is the state of the node, where $x^s \in \{\textit{single}, \textit{multi}, \textit{empty}\}$. We define a function $o(x) = |x^{\mathbf{h}}|$ that counts the total number of candidate images per node, and a function $o_h(x, a) = |x^{\mathbf{h}_a}|$ that counts the number of candidate images per node x with a given label a. Each state x^s is defined as follows:

- $x^s = \text{single} \iff i \neq \emptyset, l \neq \emptyset, o(x) \geq 0$
- $x^s = \text{multi} \iff i = \emptyset, l = \emptyset, o(x) > 0$
- $x^s = \text{empty} \iff i \neq \emptyset, l \neq \emptyset, o(x) = 0$

where single and multi nodes differ in that single nodes have their representative image defined, and multi nodes do not. Note that *multi* nodes just define transitory states where there are multiple candidates for the node but a representative image has not been selected. A single node where $o(x) > 0$ contains *hidden* images that can be shown, for example, in auxiliary panels or pop-ups.

Initialization We initialize G with the output of SOM (Self-Organizing Map) algorithm [15] applied to the set of images $\mathcal{I}$ with a grid size $M \times N$. This method finds a mapping between the n-dimensional feature space of the images and the 2D grid. In our case, we use the 169-dimensional style feature vector [8]. The SOM algorithm performs unsupervised training of a neural network to obtain a weak classification of the data. This method is suitable for grid structures and has been successfully used to display continuous color palettes [22]. As a result, we have a mapping of each image on the grid where the perceptual distances in the feature space are well preserved. However, this mapping has three problems: 1) it may contain unassigned nodes (x^{empty}); 2) the mapping of the labels might not meet the balance criteria, i.e. the amount of images of each label in *single* nodes is not the same; and 3) it may have multiple images assigned to the same position of the grid. When this third situation happens, we set the node to *multi* state, and leave the representative image and label empty. Our goal is to find the optimal arrangement of G, so that it meets this three requested conditions, or similarly, the following energy is minimal:

$$\min_{G} |EN| + \sum_{a,b \in \mathcal{L}, a \neq b} |\gamma(a) - \gamma(b)| + |MN| \tag{1}$$

where $\gamma(a) = |\{x | x \in G \wedge x^s = \text{single} \wedge x^l = a\}|$ is the number of *single* nodes in G with label a ; $EN = \{x | x \in G \wedge x^s = \text{empty}\}$ is the set of empty nodes, and $MN = \{x | x \in G \wedge x^s = \text{multi}\}$ is the set of *multi* nodes. Solving the optimal arrangement can be shown that is an NP-hard problem, therefore, we propose a greedy efficient solution which works sufficiently well for our purposes. In a first step, we turn all the multi nodes into single nodes ($x^{\text{multi}} \longrightarrow x^{\text{single}}$) by selecting the optimal representative image among its candidate list. In a second step, we turn the *empty* nodes into *single* nodes ($x^{\text{empty}} \longrightarrow x^{\text{single}}$) by taking *hidden* images from neighbor nodes in the grid.

Step 1: Arrangement of Multi nodes ($x^{multi} \longrightarrow x^{single}$) The naïve solution to turn every *multi* node into *single* node is to randomly choose one of the candidate images in $x^{\mathbf{h}}$. This is an optimal option if we only have one label in $\mathcal{L}$, however if we have more than one label, we need to choose the image in a more principled way to guarantee the balance condition. Our strategy is the following: first, find the label with minor occurrence in *single* nodes; second, find a suitable *multi* node which contains that label in the list of candidates $x^{\mathbf{h}}$; and third, set as representative image any appropriate image from the candidate list $x^{\mathbf{h}}$ of the chosen node. In more detail, in each iteration, we find the set of minority labels in *single* nodes (line 3); then, sequentially for each of these labels, we find a suitable *multi* node which contains such label in $x^{\mathbf{h}}$ (line 5). If, for a certain label, we do not find a *multi* node which contains it, we remove such label from the candidate label set $\mathcal{L}_C$ (line 7). For finding the optimal *multi* node, we give priority to the nodes with smaller number of candidates (line 9). This criteria aims at maximizing the chances to find a suitable *multi* node for the next label at every step.

Algorithm 1: Step 1: Arrangement of Multi nodes

Data: $G = \{X\}$, $\mathcal{L}$
$\mathcal{L}_C \leftarrow \mathcal{L}$
while $\exists x^{multi} \in G'$ **do**
  $\{l_s\} \leftarrow \min_{l \in \mathcal{L}_c} \gamma(l)$
  foreach l *in* l_s **do**
    $S(l) = \{x | o_h(x, l) > 0 \wedge x^{\text{multi}}\}$
    if $S(l) = \emptyset$ **then**
      $\mathcal{L}_C \leftarrow \mathcal{L}_C - \{l\}$
    else
      /* Find optimal multi node in $S(l)$ */
      $x_j \leftarrow \min_{x \in S(l)} o(x)$
      /* Update node values: state, image, label and list of candidates */
      $x_j^s \leftarrow$ single
      $x_j^i \leftarrow x_j^{\mathbf{h}(k)} : \text{label}(x_j^{\mathbf{h}(k)}) = l$
      $x_j^l \leftarrow l$
      $x_j^{\mathbf{h}} \leftarrow x_j^{\mathbf{h}} - \{x_j^{\mathbf{h}(k)}\}$
    end
  end
end

Algorithm 2: Step 2: Arrangement of Empty nodes

Data: $G = \{X\}$, $\mathcal{L}$
$\mathcal{L}_C \leftarrow \mathcal{L}$
while $\exists x^{empty} \in G \wedge \mathcal{L}_C \neq \emptyset$ **do**
  $\{l_s\} \leftarrow \min_{l \in \mathcal{L}_c} \gamma(l)$
  foreach l *in* l_s **do**
    $M(l) = \{x | o_h(x, l) > 0\}$
    $S(l) = \{x | x^{\text{empty}} \wedge x \in 8\mathcal{N}_{x'} \wedge x' \in M(l)\}$
    if $S(l) = \emptyset$ **then**
      $\mathcal{L}_C \leftarrow \mathcal{L}_C - \{l\}$
    else
      /* First, find the optimal empty node x_j in the set $S(l)$ */
      $x_j \leftarrow \min_{x \in S(l)} o_h^{\mathcal{N}}(x) : o_h^{\mathcal{N}}(x) = \sum_{x' \in 8\mathcal{N}_x} o(x')$
      /* Second, take the representative image from the neighbor node with fewer available candidates. Then, update values */
      $x_j^i \leftarrow x_k^i : x_k = \min_{x \in 8\mathcal{N}_{x_j}} o(x)$
      $x_j^l \leftarrow l$
      $x_j^s \leftarrow$ single
    end
  end
end

Step 2: Arrangement of Empty nodes ($x^{empty} \longrightarrow x^{single}$) The process to turn empty nodes into multi nodes is similar as before. We aim to fill empty nodes with the less frequent labels l_s, so we start by sorting the labels by frequency of appearance (line 4). Since empty nodes do not have a candidate list, we need to find a suitable image from its neighborhood nodes' candidate lists. We start by computing the list of nodes $M(l)$ which contain the chosen label l in their candidate list (line 6). Then, we explore its 5×5 neighborhood $\mathcal{N}_x$ in the $M \times N$ grid, and create the candidate list $S(l)$ with the nodes within $\mathcal{N}_x$ which are empty (line 7). If $S(l)$ is empty, there are no hidden images[1] with the requested label, so we remove the label from $\mathcal{L}_C$. Otherwise, we select the empty node in $S(l)$ which has the fewer amount of candidate images in its neighborhood (line 11). To select the image to assign, we find the neighbor node with fewer candidates available and take any suitable image.

The whole process is illustrated in Figure 7 for a subset of data and a small grid layout of 5×6. The initialization step shows several empty slots and unequal label distribution. *Multi* nodes are marked with a cross in the top panels; the assignment of labels and representative images for these nodes is done randomly in the initialization. In the second step, the label and images of *multi* nodes are set to maximize the variety of visible labels (balance condition). Finally, in the last step, the holes are filled with hidden candidates. We can see in purple and yellow two examples of this rearrangement. In the final step, all the images are visible while keeping the style arrangement.

6 Results and Evaluation

In the following section, we test and evaluate every step of our approach separately. We first provide several examples of the representative styles for multiple variations of a data set. Then, we provide examples of 2D arrangements and comparisons with related work. Finally, we evaluate the usefulness of our approach with a user study.

We have selected a collection of images, which we name the *tree-dog-sky* data set, which contains 2609 images of three different categories: 568 images of *sky* category, 894 of *tree*, and 1147 of *dog*. After the branch pruning of Section 5.1 with parameters $\tau = 10$ and $\kappa = 100$, we obtain a total of 1129 images (376 *sky*, 321 *tree* and 432 *dog*) which yield fourteen representative styles (Figure 8 (a)). This step guarantees that for each of these representative styles we have at least one hundred images in total and ten images of each category. In the representative styles we can see that the data covers a great variety of styles: four nodes for black and white styles with different levels of sketchiness, two nodes for colorful styles without contour and seven additional nodes with different variations of shading and complexity. We have also done experiments by randomly removing images from one category. In Figure 8 (b-c)

[1] Note that candidate images in a single node become *hidden* images after step 1.

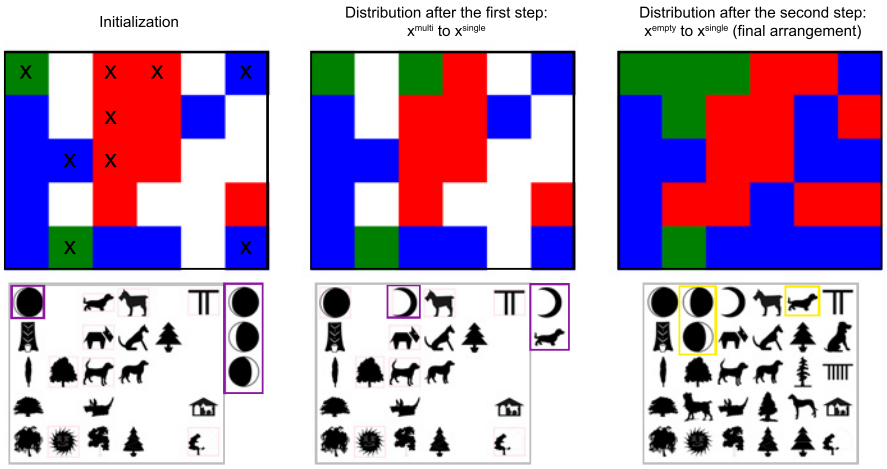


Fig. 7 Distribution of nodes and labels after each step of the algorithm: initialization (first column), step 1 (second column) and step 2 (third column). Top row: labels (x^l) for dog, sky and tree categories represented by red, green and blue colors respectively. Bottom row: images (x^i). The crosses in the initialization column define *multi* nodes where label and images are randomly selected. Bottom panels of columns one and two, contain an additional vertical panel on the right (squared in purple). The images within this panel belong to the corresponding purple node on the left panel, i.e. in this nodes the number of candidate images is greater than zero after both steps $o(x) > 0$. On the third column, these images have been redistributed to produce a fully occupied grid. The input is a set of 2141 images with three different labels: dog, sky, tree. The number of images for each label in the same order is: 1147, 100 and 894. Note that green label (sky) is the less frequent and yet it has significant representation in the final arrangement.

we can observe the resulting styles by reducing to 100 the number images of sky and tree categories and $\tau = 5$ and $\kappa = 50$. We can see that the representative styles change depending on the available data.

We have compared our adaptive 2D arrangement with the recent method of Fried et al. [6] and the raw output of Self-Organizing Maps (SOM) [15]. We have selected two of the style branches of Figure 8 (a) and mapped its content in a 2D grid (Figure 9) of fixed size 7×10. Since Isomatch does not handle grids of smaller size than the number of images, we have randomly sampled a subset of the data to visualize it. In our solution and SOM we mark the nodes with more than one images assigned with a red square. We can observe that our solution and SOM preserves better than Isomatch the style distances in the grid, see for example, that the moon images in the black style are located closed to each other in SOM while the arrangement of these images in isomatch looks random. The same happens with several other clusters of style marked in the figure. Our solution keeps this good behaviour of SOM while producing a fully occupied grid.

Evaluation We performed a user study to evaluate the impact of our approach on icon selection tasks. In this study, some users were provided with our style-

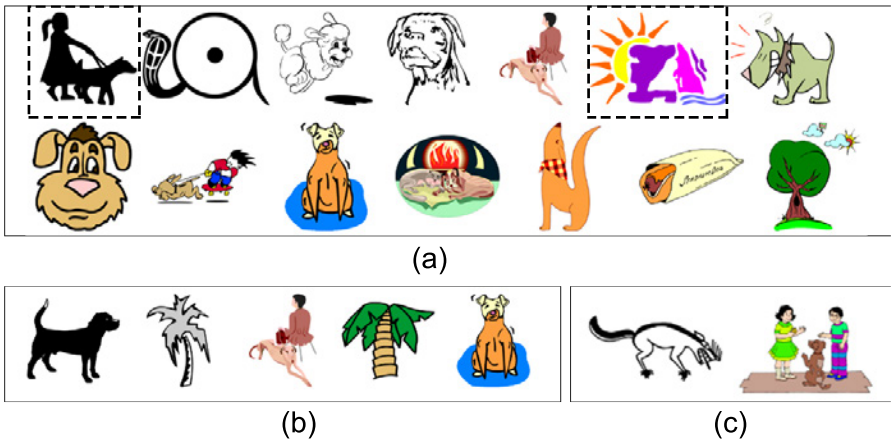


Fig. 8 Representative Styles of the *tree-dog-sky* data set. (a) Contains the full data set with $\tau = 10$ and $\kappa = 100$. In (b) we have reduced the amount of tree elements to 100, while in (c) we have reduced the tree category by the same amount, both with $\tau = 5$ and $\kappa = 50$. As we can see the representative styles change depending on the available data.

based exploratory interface, and others received a simpler version were the search results were just sorted by style similarity. Users were not aware of this difference, and they just interacted with one version of the interface.

The task was to find a set of icons matching in style for a kids' user interface. We gave them the list of requested categories: *face*, *fish*, *flower* and *bird* (with 937, 1014, 1022 and 1015 images), which they had to find on the data set and place on a canvas. Each time a category was set on the canvas, we automatically eliminated the images of such category on the retrieved results to ease the task in both interfaces. We asked each user to create four sets of icons of four different styles, and at every moment, they were able to see all the completed sets to avoid repeating styles. In each of the requested sets, one of the requested categories had considerably less images than the other three. To avoid bias, the users were not aware of this fact. We specifically selected this scenario to prove that our method can handle such difficult cases, when the data set does not contain all the images in all the styles. There is a screen capture of both interfaces and the instructions in Figure 10. Please, see also the accompanying video.

We gathered a total of 64 icons sets, 28 created with our interface and 36 without it. For each interface, we measure the average time to complete each set and the number of delete operations. The average time for each set using our interface was 79 seconds with a confidence interval at 95% of ± 20, and 108 seconds without it , with a confidence interval of ± 27. The average number of images deleted was 0.6 with our interface and 2.8 without it, with confidence intervals at 95% of 1.1 and 1.2 respectively. We additionally captured several comments from the users which tested the basic version that complained about not having enough data of certain styles. From this experiments we can

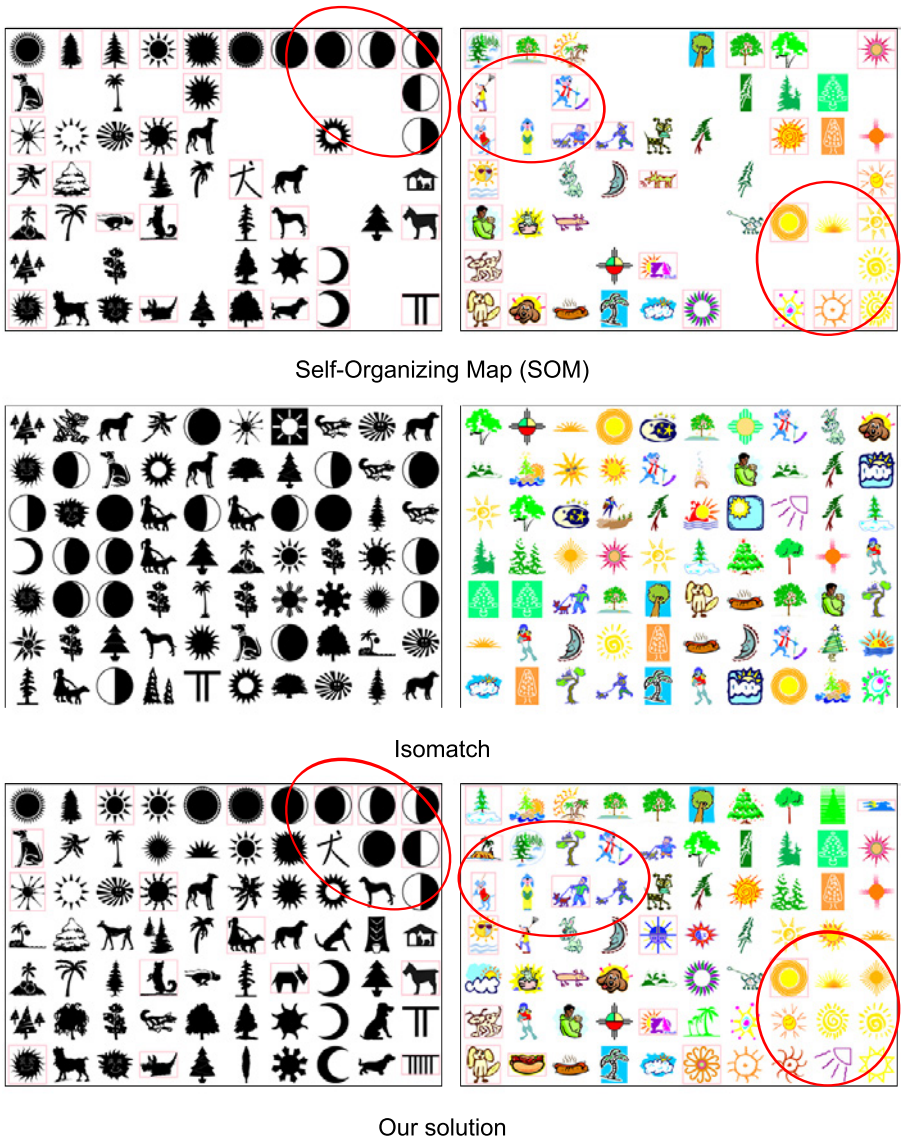


Fig. 9 2D arrangement for the two style branches marked with dotted lines in Figure 8 (a). Top: Self-Organizing Map [15]. Middle: Isomatch [6]. Bottom: Our solution. We observe clear style clusters in SOM (marked in red) which are also preserved in our solution. The arrangement of these images in Isomatch looks more chaotic.

conclude that using our approach, the task of finding optimal icons sets of multiple styles is easier and around 30 % faster than using basic exploration, providing more information and guidance.

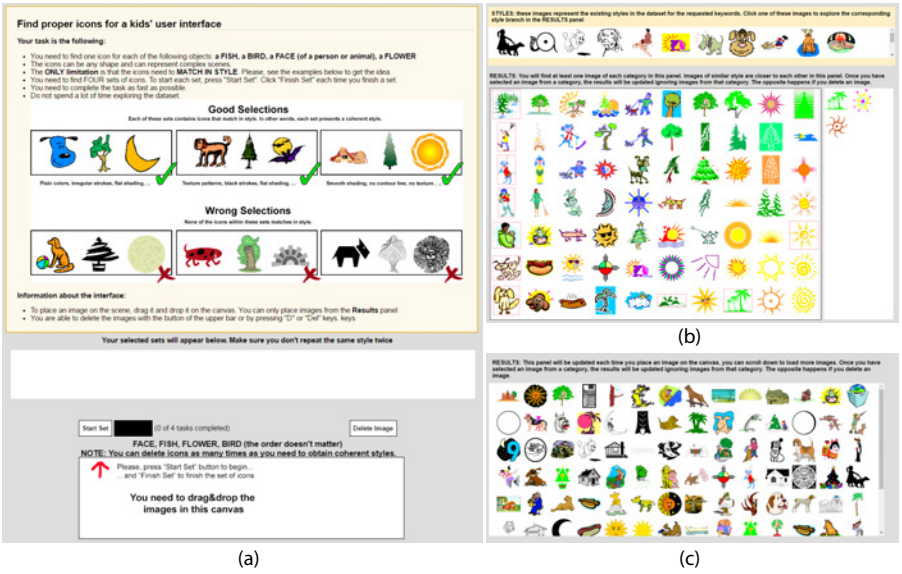


Fig. 10 (a) Instructions for all users and both interfaces. (b) Our proposed solution. On the top row we can check the available styles. At the bottom, the selected branch is arranged in 2D. In nodes with more than one image, the remaining images are shown on the right vertical panel. (c) Basic interface, where images of the data set are sorted randomly at the beginning, and then, sorted by style similarity to the images of the canvas.

7 Conclusions and Future Work

In this paper we have presented a method to explore and visualize big collections of clip art images according to its visual style. We have relied on an existing style similarity metric which we have evaluated with two novel approaches: by means of a user study, we have learnt that users perceive stroke, shape and color as the most dominant attributes to identify similar styles; by using a labeled data set, we have obtained objective error metrics which measures its quality. Using this metric, we have built a hierarchy which captures the underlying structure of styles existent in a given data set. A combined approach between dimensionality reduction techniques, and strategic sampling of certain nodes of the hierarchy allows to intuitively visualize the styles present in the data set and navigate through them. We have tested and confirmed the usefulness of our approach by means of a user study.

The main problem we have found to capture style is that visual style attributes are very correlated. For this reason, is highly difficult to identify clear categories. An interesting avenue of future work would be to describe the style with relative attributes [25], that is, it is easier to say that an image is very colorful than to classify an image as colorful or not. In this sense, we want to explore fuzzy clustering algorithms to see whether they can capture complex relationships between the visual elements.

Acknowledgements We would like to thank all reviewers for their thoughtful comments. We also thank Carlos Bobed for insightful comments and proofreading the paper. This work was partially supported by the European Social Fund, the Gobierno de Aragon, and the Ministerio de Economia y Competitividad (project LIGHTSLICE).

References

1. Averkiou, M., Kim, V., Zheng, Y., Mitra, N.J.: Shapesynth: Parameterizing model collections for coupled shape exploration and synthesis. Computer Graphics Forum (Proc. Eurographics) (2014)
2. Bae, S., Paris, S., Durand, F.: Two-scale tone management for photographic look. ACM Trans. Graphics **25**(3) (2006)
3. Campbell, N.D.F., Kautz, J.: Learning a manifold of fonts. ACM Trans. Graphics (Proc. SIGGRAPH) **33**(4) (2014)
4. Durand, F.: An Invitation to Discuss Computer Depiction. In: Proc. NPAR (2002)
5. Eitz, M., Hay, J., Alexa, M.: How Do Humans Sketch Objects ? ACM Trans. Graphics (Proc. SIGGRAPH) (2012)
6. Fried, O., DiVerdi, S., Halber, M., Sizikova, E., Finkelstein, A.: IsoMatch: Creating informative grid layouts. Computer Graphics Forum (Proc. Eurographics) **34**(2) (2015)
7. Frisby, J.P., Stone., J.V.: Seeing: The Computational Approach to Biological Vision. MIT Press (2010)
8. Garces, E., Agarwala, A., Gutierrez, D., Hertzmann, A.: A similarity measure for illustration style. ACM Trans. Graphics (Proc. SIGGRAPH) **33**(4) (2014)
9. Han, Z., Liu, Z., Han, J., Bu, S.: 3d shape creation by style transfer. The Visual Computer **31**(9) (2014)
10. Hertzmann, A., Jacobs, C.E., Oliver, N., Curless, B., Salesin, D.H.: Image Analogies. In: Proc. SIGGRAPH (2001)
11. Huang, S.S., Shamir, A., Shen, C.H., Zhang, H., Sheffer, A., Hu, S.M., Cohen-Or, D.: Qualitative organization of collections of shapes via quartet analysis. ACM Trans. Graphics (Proc. SIGGRAPH) **32**(4) (2013)
12. Jin, R., Wang, S., Zhou, Y.: Regularized distance metric learning:theory and algorithm. In: Proc. Neural Information Processing Systems (2009)
13. Kalogerakis, E., Nowrouzezahrai, D., Breslav, S., Hertzmann, A.: Learning Hatching for Pen-and-Ink Illustration of Surfaces. ACM Trans. Graphics **31** (2012)
14. Kleiman, Y., Fish, N., Lanir, J., Cohen-Or, D.: Dynamic maps for exploring and browsing shapes. In: Proc. Eurographics/ACMSIGGRAPH Symposium on Geometry Processing (2013)
15. Kohonen, T.: The self-organizing map. Proc. of the IEEE **78**(9) (1990)
16. Kulis, B.: Metric learning: A survey. Foundations and Trends in Machine Learning **5**(4) (2013)
17. Li, H., Zhang, H., Wang, Y., Cao, J., Shamir, A., Cohen-Or, D.: Curve Style Analysis in a Set of Shapes. Computer Graphics Forum **32**(6), 77–88 (2013)
18. Liu, T., Hertzmann, A., Li, W., Funkhouser, T.: Style compatibility for 3d furniture models. ACM Trans. Graphics (Proc. SIGGRAPH) **34**(4) (2015)
19. Lun, Z., Kalogerakis, E., Sheffer, A.: Elements of style: learning perceptual shape style similarity. ACM Trans. Graphics (Proc. SIGGRAPH) **34**(4) (2015)
20. Luo, Y., Liu, T., Tao, D., Xu, C.: Decomposition-based transfer distance metric learning for image classification. IEEE Transactions on Image Processing **23**(9) (2014)
21. McFee, B., Lanckriet, G.: Metric learning to rank. In: Proc. International Conference on Machine Learning (2010)
22. Nguyen, C.H., Ritschel, T., Seidel, H.P.: Data-driven color manifolds. ACM Trans. Graphics **34**(2) (2015)
23. O'Donovan, P., Agarwala, A., Hertzmann, A.: Collaborative filtering of color aesthetics. In: Proc. Computational Aesthetics (2014)
24. O'Donovan, P., Lībeks, J., Agarwala, A., Hertzmann, A.: Exploratory Font Selection Using Crowdsourced Attributes. ACM Trans. Graphics (Proc. SIGGRAPH) **33** (2014)

25. Parikh, D., Grauman, K.: Relative attributes. In: Proc. IEEE International Conference on Computer Vision (2011)
26. Reinert, B., Ritschel, T., Seidel, H.P.: Interactive by-example design of artistic packing layouts. ACM Trans. Graphics (Proc. SIGGRAPH Asia) **31**(6) (2013)
27. Rubinstein, M., Gutierrez, D., Sorkine, O., Shamir, A.: A comparative study of image retargeting. ACM Trans. Graphics (Proc. SIGGRAPH Asia) **29**(5) (2010)
28. Saleh, B., Dontcheva, M., Hertzmann, A., Liu, Z.: Learning style similarity for searching infographics. In: Proc. Graphics Interface Conference (2015)
29. Schultz, M., Joachims, T.: Learning a Distance Metric from Relative Comparisons. In: Proc. Neural Information Processing Systems (2003)
30. Shapira, L., Shamir, A., Cohen-Or, D.: Image Appearance Exploration by Model-Based Navigation. Computer Graphics Forum (Proc. Eurographics) **28**(2) (2009)
31. Sidi, O., van Kaick, O., Kleiman, Y., Zhang, H., Cohen-Or, D.: Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. ACM Trans. Graphics (Proc. SIGGRAPH Asia) **30**(6) (2011)
32. Tenenbaum, J.B., Freeman, W.T.: Separating Style and Content with Bilinear Models. Neural Computation **12**(6) (2000)
33. Ward, J.H.: Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association **58**(301) (1963)
34. Willats, J., Durand, F.: Defining pictorial style: Lessons from linguistics and computer graphics. Axiomathes **15**(3) (2005)
35. Yamaguchi, K., Kiapour, M.H., Ortiz, L.E., Berg, T.L.: Retrieving similar styles to parse clothing. IEEE Trans. on Pattern Analysis and Machine Intelligence **37**(5) (2015)
36. Yang, L.: Distance metric learning: A comprehensive survey. Tech. rep. (2006)