Non-Photorealistic Animation and Rendering
Pierre Bénard and Holger Winnemoeller (Editors)

Patternista: Learning Element Style Compatibility and Spatial Composition for Ring-based Layout Decoration

H. Q. Phan[1] J. Lu[2] P. Asente[2] A. B. Chan[3] H. Fu[1]

[1]School of Creative Media, City University of Hong Kong, Hong Kong
[2]Adobe Systems, USA
[3]Department of Computer Science, City University of Hong Kong, Hong Kong

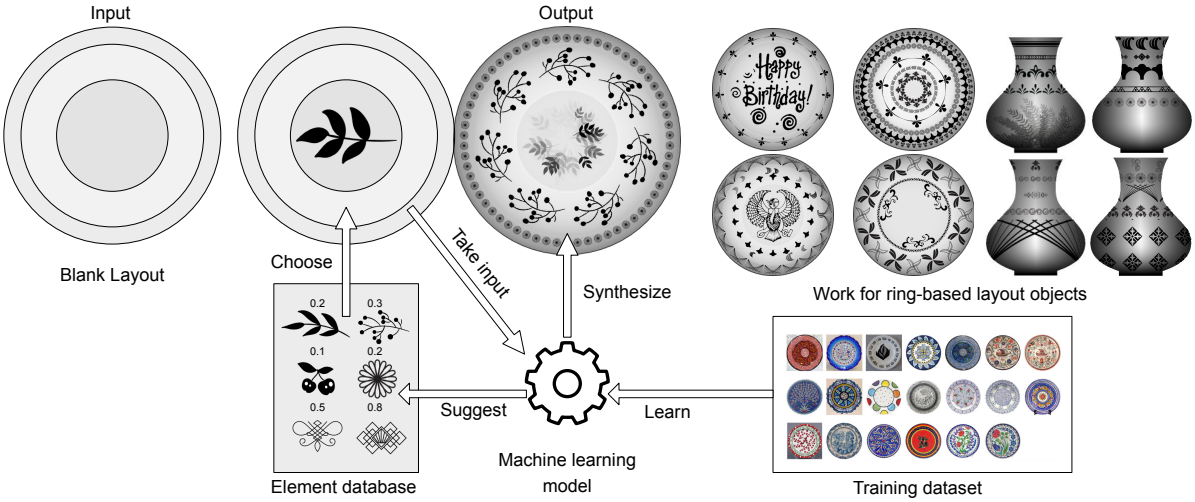

Figure 1: *An overview of our system. Given a blank layout containing a few concentric regions, the user selects an element from the element database and drags it into one of the regions. Trained on a separate dataset of ring-based objects such as plates, our machine learning model takes into account the user input and recommends the user with a list of ranked elements for the empty regions. Our system then synthesizes the final design by arranging the recommended elements according to the predicted composition. Our framework is applicable to a wide range of objects such as vases, pots and plates.*

Abstract

Creating aesthetically pleasing decorations for daily objects is a task that requires deep understanding of multiple aspects of object decoration, including color, composition and element compatibility. A designer needs a unique aesthetic style to create artworks that stand out. Although specific subproblems have been studied before, the overall problem of design recommendation and synthesis is still relatively unexplored. In this paper, we propose a flexible data-driven framework to jointly consider two aspects of this design problem: style compatibility and spatial composition. We introduce a ring-based layout model capable of capturing decorative compositions for objects like plates, vases and pots. Our layout representation allows the use of the hidden Markov models (HMM's) technique to make intelligent design suggestions for each region of a target object in a sequential fashion. We conducted both quantitative and qualitative experiments to evaluate the framework and obtained favorable results.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation

1. Introduction

Decorating everyday objects is not only a designer's work but also a hobby of many people. A hand-decorated object such as a greeting card, a living-room wall or a flower vase always makes the surrounding environment more stylistic and pleasing to the owner's taste. It is, however, not easy for an inexperienced person to deco-

DOI: 10.2312/exp.20161066

Figure 2: *Ceramic objects from various countries.*

rate an object since the task requires numerous design skills such as composition, coloring, creating elements and mixing elements. Even a professional designer can have to invest a great amount of time to come up with a satisfactory decoration [Owe00]. Nowadays, with the availability of collections of decorative elements on the Internet, designers are, to some degree, freed from creating every element from scratch and can focus more on the composition. Nevertheless, the element composition task requires the designer to have a strong personal style in order to create unique and aesthetically pleasing decorations. The main objective of our work is to build an automatic design recommendation system that, given some user constraints such as the element style and the arrangement type for one region, suggests stylistically compatible elements and arrangement types for other regions.

Several aspects of our problem have appeared in computer graphics and vision literature, for instance, the problems of finding compatible elements [LKS15, GAGH14], pattern coloring [HZMH14, KYKL14, LRFH13] and pattern composition [LA15, YBY*13]. There is, however, little work integrating these aspects together to build a complete computer-aided design system dedicated to object decoration. The well-known model *factor graphs* [Bis06] has been used for design recommendation [LRFH13, YBY*13]. This model is, however not directly applicable to our problem domain as decorative objects always have highly flexible layouts while factor graphs can only have fixed structures.

To allow a more flexible solution to design recommendation, we treat object decoration as a sequential process, in which each step is dependent only on the previous step. We limit the scope of our work to those decorative patterns that have ring-based layouts. Consider the case of ceramic design. Figure 2 shows ceramic objects from different cultures, all of which are decorated in a sequential fashion (region by region). For the case of the plate, the surface is divided into 2D ring-like regions. Each region contains a certain type of elements repeated according to some arrangement rule. Similarly, for vases, the surface is divided into 3D rings, wrapping around the object. We demonstrate our design suggestion system mainly in the context of ceramic plates.

This sequential design assumption is the key of our method, as it greatly simplifies our problem and allows the use of robust machine learning models designed for this type of data. To start, we introduced a *ring-based layout* (Figure 3), which consists of a sequence of rings. Each ring covers a region on the surface of an object and contains a single element. We also consider regions with multiple elements by allowing rings to overlap. Next, we designed a hidden Markov model (HMM) that explicitly takes composition and style compatibility into account. Although it is possible to incorporate color into our model, we decided not to consider this factor because color compatibility has been studied before [LRFH13]. In our model, each time step in the HMM is equivalent to a ring in the layout. The hidden states encode decorative styles and compatibility between elements. We trained our model on a dataset of annotated plate images, and used it to generate novel decorations automatically with no or little user intervention. We conducted both quantitative and qualitative experiments to assess the performance of our model and obtain promising results. Figure 1 summarizes our pipeline. We will discuss it in detail in Section 3.1.

2. Related Works

We briefly review two research fields directly related to our work, style compatibility analysis and pattern synthesis.

2.1. Style Compatibility and Similarity

Artistic style analysis has been studied by both computer scientists and psychologists [GB14, HGR10, GFRF10]. Wavelets is one of the conventional tools for artist identification [JJHB*08]. More recently, Hughes et al. applied sparse coding to quantify artistic style [HGR10]. Shamir et al. used a large set of image features to classify painting according to its school of art [SMO*10]. In the field of computer graphics, researchers are interested more in measuring similarity in style between photographs or decorative elements. Garces et al. extracted features using *metric learning* to measure similarity in style between pieces of clip-art [GAGH14]. Bell and Bala applied deep learning to learn the similarity between different product designs [BB15]. O'Donovan et al. [OAH15] trained an energy-based model to characterize stylistic layouts. Although these works demonstrated promising performance, few of them attempted to perform style interpolation, which could be useful when the size of training data is small. In fact, our goal is to design a model that is flexible enough to work with a dataset of any size, ranging from a few dozens to millions.

2.2. Pattern Synthesis

Pattern synthesis is another topic that is closely related to our work. The classic work of Wong et al. [WZS98] proposed an algorithmic approach for generating floral pattern. Lu et al. introduced a technique to synthesize a coherent pattern along a user-provided line [LBW*14]. Although the technique in [LBW*14] is non-learning, it can achieve stylistic results through the use of examples that belong to a certain style category. Wu et al. [WWY14] introduced an energy function and an Expectaion-Maximization-based method to synthesize art patterns and textures by repeating an input example with user constraints. This field of pattern synthesis is different from ours as we are concerned more about the semantic and compatibility of the examples rather than the synthesis process itself.

Lin et al. introduced a learning model capable of synthesizing stylistic coloring by training on a certain set of color patterns [LRFH13]. The work of Yeh et al. [YBY*13] used a set of factor graphs to capture sub-structures within larger patterns such facades or decorative frames. This work is similar to ours as its objective is also to learn and to synthesize structural patterns. Their patterns, however, have tiled layout rather than ring layout, as in our case. Another difference is, they do not explicitly consider style compatibility between regions.

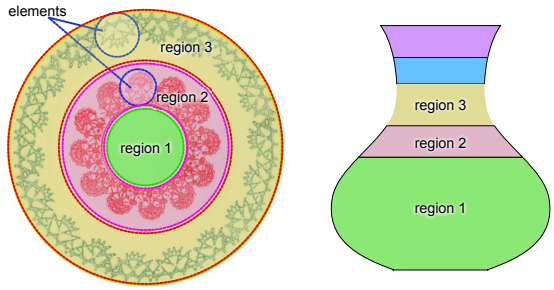

Figure 3: *In a ring layout, a surface is divided into sequential rings, each of which covers a certain decorative region.*

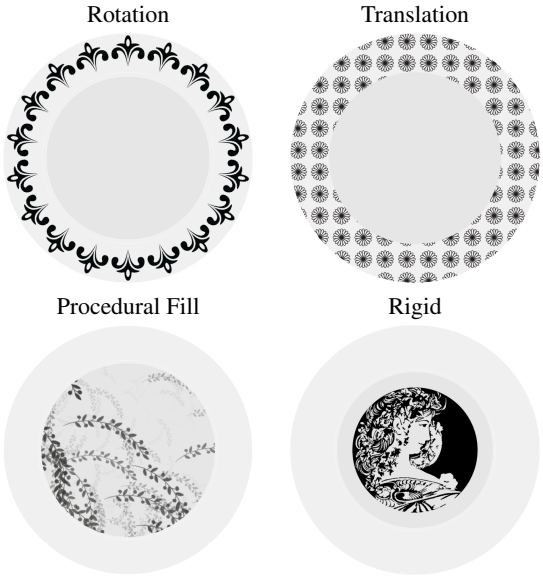

Figure 4: *There are 5 types of arrangements: (1) Rotation, (2) Translation, (3) Procedural Fill, (4) Rigid, and (5) Empty.*

3. Methodology

In this section, we explain how the data were collected, what features were used and our learning model.

3.1. System Overview

Our system aims to simultaneously suggest both *stylistic* elements and their *composition* to the user. Figure 1 illustrates the functioning pipeline of our system. In the initial step, the user is presented with a blank plate to decorate. In the second step, the user places an initial decorative element onto the surface. In the next step, our system queries a database of decorative elements and presents the most relevant ones to the user. In addition, the system also recommends an aesthetically pleasing way to arrange these elements. To achieve these goals, we train a machine learning model on a separate database of images containing decorated objects, collected from the Internet, and then use it to predict composition and style of the entire decoration. Style is represented in our model as a 9-dimensional feature vector. Composition is realized with three factors: the density of elements in a region, the shape of each element and their arrangement type. Details about the arrangement types will be explained in Section 3.2. Given any factor or a combination of these factors as the constraint, we predict the missing information and synthesize a complete design with or without any user intervention. If the user desires to have more control over the the synthesis, we introduce *incremental design*, a design process that makes use of our recommendation engine but still allows the user to jump in at any step (See Section 3.6.1).

Figure 5: *Our dataset consists of plate images that adhere to our ring layout.*

3.2. Ring Layout

Figure 3 visualizes our ring layout. Each region enclosed by the colored lines (red, pink and green) indicates a decorative region in the layout. A decorative region contains a single element type circled in blue and is associated with an arrangement type. By observing real-world plate designs, we derive 5 types of arrangement, which are common among the designs: Rotation, Translation, Procedural Fill, Rigid and Empty. These types of arrangement are illustrated in Figure 4. While Translation and Rotation are familiar concepts, we define Procedural Fill as a type of arrangement that uses a simple element to fill an arbitrary region by following a certain procedure. In [WZS98], the authors discussed similar arrangement types, among which the *glide reflection* can be thought as a particular case of Procedural Fill. Rigid is a type of arrangement that only has a single instance of the element and can scale it to fit in a region.

3.3. Data Collection and Annotation

We collected a dataset of 184 plate images from search engines like Google Images and Flickr. Figure 5 shows some examples from the dataset. The plates selected for training have at least two separate regions that follow the ring layout explained in Section 3. All images were cropped and resized to have a fixed size of 800x800 pixels. Since we do not consider color in our model, we converted all images to grayscale and binarized them to match the format of our element database. The images were binarized by a global threshold, which was determined by first sampling from local patches along strong edges (identified by the Laplacian of Gaussian (LOG) filter) and then choosing the threshold to be the mean edge intensity plus the standard deviation. The images were then manually annotated with the following information: the inner and outer ring boundaries, the scale of one typical element, the Euclidean distance between two neighboring instances of an element, and the type of arrangement used in each ring. For each ring, we only annotated one element.

We also prepared a database of binary decorative elements (called the Element DB), which was partially collected from the Internet and partially self-designed. The elements fall into 3 aspect ratios: 1x1, 2x1 and 3x1, suitable for different situations in the syn-

thesis step. Figure 6 shows a subset of the elements in the database.

3.4. Features and Extraction Methods

In this section, we explain the features used in our model. Our feature set consists of 2 groups of features: Element Style features and Composition features. The Element Style features were extracted from annotated elements while the Composition features were computed for each region in the layout. All features were normalized before any learning/inference step.

3.4.1. Element Style Features

Although many features have been proposed for representing the style of images [HGR10, SMO*10], we find them unsuitable for our application for the following reasons. First, these features were developed for discriminative purposes such as classification and retrieval, thus they tend to have high number of dimensions – a few thousands. With a relatively small dataset, our model cannot use high-dimensional features for fear of over-fitting. Second, these features do not allow meaningful interpolation, which is important for generating novel designs. For the reasons above, we design our own feature set – a 9-dimensional vector composed of texture-based features (2D), curvature-based features (4D) and pixel-counting features (3D). Inspired by prior works on visual style recognition [LC09,GAGH14], we aim to use our feature set to capture different characteristics of decorative style such as simplicity/complexity (via texture-based features), hardness/smoothness (via curvature-based features) and boldness/thinness (via pixel-counting features). We demonstrate the performance of our features in Figure 6, in which we applied conventional clustering and *manifold learning* techniques to visualize the Element DB.

Texture-based features consist of a pair of real numbers $[\kappa, \lambda]$, where λ represents the variability and κ is the max kurtosis value – indicating how the peaks and the tails of a distribution differ from a normal distribution. To extract λ and κ, we first use a sliding window to extract low-level features from the image. The low-level feature being used is the Maximum-Response 8 (MR8) [VZ05]. κ is the maximum kurtosis value across all 8 dimensions of MR8. λ is the parameter of an exponential function, fitted to the sorted histogram of MR8 clustering means in the context of a Bag-of-Words model. The dictionary for the bag-of-words model was produced with K-means.

Curvature-based features consist of the first 4 statistical moments (mean, standard deviation, kurtosis and skewness) of the curvature values sampled along prominent edges in the element. To extract edges, we first apply the Canny edge detector [Can86] and then mask out edges lying further than some distance from the element center. This distance threshold is the radius of the circle bounding an element and was specified at the data annotation step (Section 3.3). Figure 7 (A, C) demonstrates the steps to extract curvature. (A) is the element image, (C) shows the edge map in which we sample curvature values at random points on the edges. Given a center point and 4 pairs of neighboring points (spaced 2, 3, 4, and 5 pixels from the center point) on the same edge, we estimate the curvature with the method described by Anoshkina et al. [ABS02]. We also segment out the edges (connected edge pixels) before estimating the curvature to avoid the noise arising when edges are too close. The connectivity between pixels is determined by a 3x3 cross-shaped neighborhood, which includes adjacent pixels along vertical and horizontal directions.

Pixel-counting features consist of the number of curves in the element, the average thickness of curves, and the ratio between the number of black and white pixels over the entire element image. To calculate the average thickness of curves we apply the exact Euclidean distance transform to find up to 200 peaks in the binary image. Next, we compute the ratios between the number of black and white pixels within 9×9 pixel windows centered at the peaks (yellow boxes in Figure 7 (B, D)) and average them across all windows. In Figure 6 (Left), we demonstrate the usefulness of our feature vector by carrying out a clustering task on our set of elements.

3.4.2. Composition Features

As mentioned in Section 3.1, the composition in a ring is governed by three factors: the element density, the shape of element and their arrangement type.

- The arrangement type is encoded by an integer $1:5$.
- Element shape is represented by a pair of real numbers, the width and height of the element relative to the ring radius (See Figure 3).
- Element density is measured by the smallest distance between any pair of instances of an element and was manually annotated during data preparation (See Section 3.3).

3.5. Learning and Inference

Our learning model is derived from a standard hidden Markov model (HMM) with discrete hidden states, where each time step is equivalent to a ring in our ring layout. In our context, the hidden states can be thought as the implicit *styles* behind a design. A sequence of hidden styles will decide the final look of a design. Another important concept in HMM is the emission densities, which translate the implicit styles into actual elements and compositions. We utilize custom emission densities that take into account both discrete and continuous observations. The last piece of information needed to describe an HMM is the *transition matrix* and the starting probability, which connect the hidden states. HMMs are conventionally used to model sequential data, which is typically time-series data [RJ93]. We, however, found that the model is also useful for modeling design processes that involve sequential steps. We call this type of design *sequential design*.

Figure 8 shows how HMM is used in our problem. At each time step/ring, we have a set of random variables:

- $\mathbf{z}_n$: The hidden state. It is a discrete variable and can take values $1:K$, with K being the number of hidden states.
- $\mathbf{g}_n$: The element style (9 dimensional vector, continuous). (See Section 3.4.1).
- $\mathbf{e}_n$: The element's width and height, relative to the ring's thickness (2 dimensional vector, continuous).
- u_n: The density of elements (scalar, continuous).
- s_n: The type of arrangement (discrete, one of 5 values: rotation, translation, procedural fill, rigid or empty).
- r_n: The ring's thickness, relative to the plate's radius (scalar, continuous).
- d_n: The distance from the middle of the ring to the center of the plate, relative to the plate's radius (scalar, continuous).

In our model the transition distribution $p(\mathbf{z}_n|\mathbf{z}_{n-1}, \mathbf{A})$ and the distribution of the initial state $p(\mathbf{z}_1|\mathbf{b})$ are the same as in the

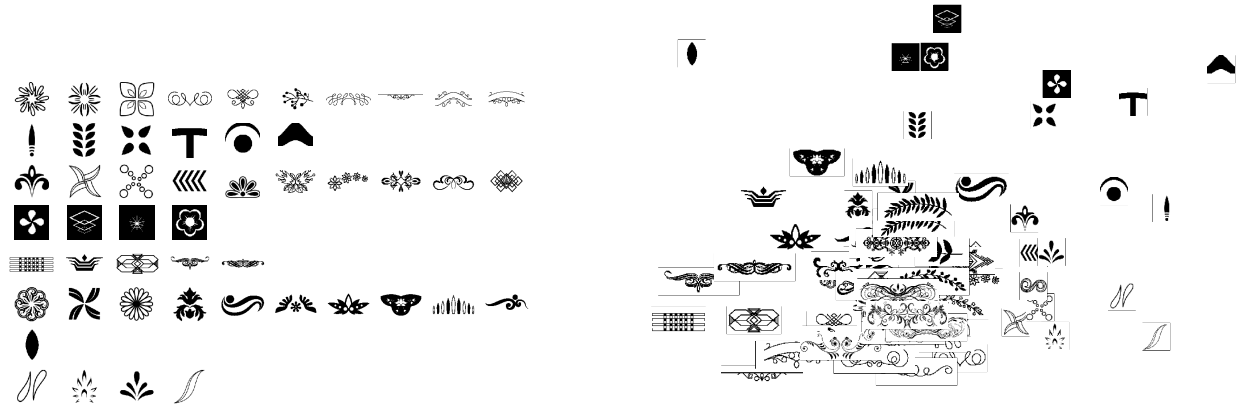

Figure 6: *(Left) Clustering performance with our feature set– each row is a cluster. (Right) We visualize the manifold of elements using MDS.*

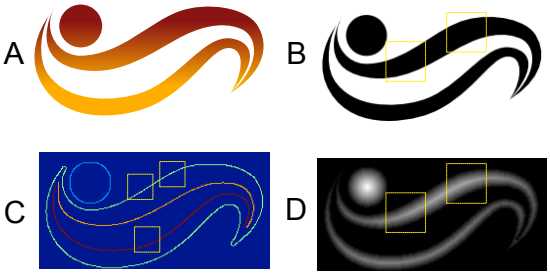

Figure 7: *Curvature-based and Pixel-counting-based feature extraction. (A) original image. (B) thresholded binary image. (C) edge map. (D) Euclidean distance transform map.*

standard model [Bis06]. Whereas, the continuous variables $\mathbf{c}_n = [\mathbf{g}_n, \mathbf{e}_n, r_n, d_n, u_n]$ are modeled as a Gaussian emission density:

$$p(\mathbf{c}_n|\mathbf{z}_n,\mu,\Sigma) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{c}_n|\mu_k,\Sigma_k)^{z_{nk}}, \quad (1)$$

and the discrete arrangement type s_n is modeled as a Categorical emission density:

$$p(s_n|\mathbf{z}_n,\mathbf{E}) = \prod_{k=1}^{K} \mathcal{C}(s_n|E_k)^{z_{nk}}, \quad (2)$$

where $z_{nk} = 1$ if the n-th state has value k. The parameters of our model include:

- $\mathbf{A} \in \mathbb{R}^{K\times K}$: A transition matrix.
- $\mathbf{b} \in \mathbb{R}^{K}$: The prior for the initial state in a sequence.
- $\mu_k \in \mathbb{R}^{14}, \Sigma_k \in \mathbb{R}^{14\times 14}, k \in 1:K$: Parameters of Gaussian emission probabilities. In practice, we used diagonal covariances, assuming the independence between continuous variables.
- $\mathbf{E}_k \in \mathbb{R}^{K\times 5}$: The emission probabilities for all possible values of s_n (5 types of arrangement).

The joint probability distribution of the model is:

$$p(\mathbf{X},\mathbf{Z}|\mathbf{b},\mathbf{A},\phi) = p(\mathbf{z}_1|\mathbf{b}) \left[\prod_{n=2}^{N} p(\mathbf{z}_n|\mathbf{z}_{n-1},\mathbf{A})\right] \prod_{m=1}^{N} p(\mathbf{c}_n,s_n|\mathbf{z}_n,\phi),$$

where $\mathbf{X} = \mathbf{x}_1,\ldots,\mathbf{x}_N$ is the training dataset.The parameters were estimated using Expectation-Maximization algorithm (EM) with the forward-backward algorithm.

Inference

Our use case is that some information will be given to a few rings in the layout and we aim to reconstruct the missing information. For example, in a design with 3 rings, the user may have specified their preferred elements for the first and the second rings but did not provide the desired element for the last ring, or the arrangement type or element density for any ring. Our strategy is to make use of as much observed information as possible and to ignore the missing information by using default emission densities—a Gaussian distribution with zero mean, identity covariance, or a uniform distribution. When some continuous variables are available, we use the marginal Gaussian distribution of those variables instead. In this way, we are able to approximate the conditional distribution $p(\mathbf{c}_n, s_n|\mathbf{z}_n, \phi)$, needed for the forward-backward [Bis06].

To recover the missing information, we first estimate the hidden states $\mathbf{z}_n$ for each ring and then use the emission densities to generate the missing information. Formally, we want to sample from the distribution $p(\mathbf{z}_{1:N}|\mathbf{x}_{1:N})$, where $\mathbf{x}_{1:N}$ is the input dataset with missing data and N is the number of rings in the input layout. We have:

$$\begin{aligned} p(\mathbf{z}_{1:N}|\mathbf{x}_{1:N}) &= p(\mathbf{z}_N|\mathbf{x}_{1:N}) \prod_{n=1}^{N-1} p(\mathbf{z}_n|\mathbf{z}_{n+1},\mathbf{x}_{1:N}) \\ &= p(\mathbf{z}_N|\mathbf{x}_{1:N}) \prod_{n=1}^{N-1} p(\mathbf{z}_n|\mathbf{z}_{n+1},\mathbf{x}_{1:n}). \end{aligned} \quad (3)$$

We notice that $p(\mathbf{z}_n|\mathbf{z}_{n+1},\mathbf{x}_{1:n})$ does not depend on $\mathbf{x}_{n+1:N}$ since $\mathbf{z}_{n+1}$ is given. We sample this distribution in a reverse order, from the last ring $\mathbf{z}_N$ to the first ring $\mathbf{z}_1$. The procedure is:

1. Sample $\hat{\mathbf{z}}_N$ from $\gamma_N = p(\mathbf{z}_N|\mathbf{x}_{1:N})$, where γ_N is estimated with the backward-forward algorithm.
2. Sample $\hat{\mathbf{z}}_n$ from

$$p(\mathbf{z}_n|\hat{\mathbf{z}}_{n+1},\mathbf{x}_{1:n}) \propto p(\hat{\mathbf{z}}_{n+1}|\mathbf{z}_n)p(\hat{\mathbf{z}}_n|\mathbf{x}_{1:n}),$$

where $p(\hat{\mathbf{z}}_{n+1}|\mathbf{z}_n)$ is taken from the transition matrix and $p(\hat{\mathbf{z}}_n|\mathbf{x}_{1:n})$ is from the forward algorithm.

Once we have obtained all the hidden states, the emission distribu-

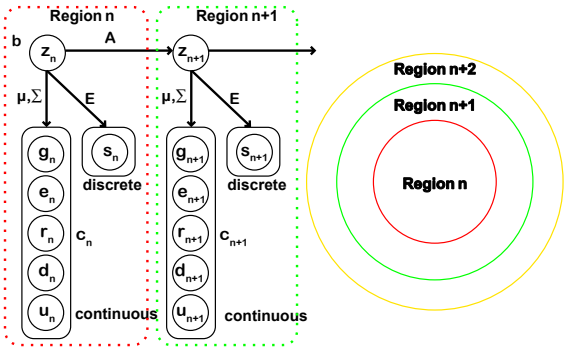

Figure 8: *Hidden Markov model.*

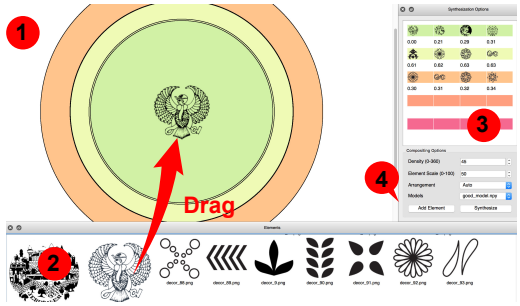

Figure 9: *User interface.*

tions (2) and (1) are used to generate the observations. This step concludes our sampling/inference process.

3.6. Decoration Synthesis

The final decoration is governed by a set of variables $\{\mathbf{g}_n, \mathbf{e}_n, s_n, u_n\}_{n=1}^{N}$, where N is the number of rings in the decoration and $\mathbf{g}_n, \mathbf{e}_n, s_n$ and u_n describe element style, element size, arrangement type and element density (see Section 3.5). Note that we do not consider r_n, and d_n as the ring layout is assumed to be given in our model. During the inference step, r_n, d_n are always observed and we only need to generate $\mathbf{g}_n, \mathbf{e}_n, s_n$ and u_n for the rings with missing information using the sampling method described in Section 3.5. The predicted $\mathbf{g}_n$ are then used as a query to retrieve the style-compatible elements from the element database. To do this, we first extract the style features, as described in Section 3.4, for each element in the element database. Let the feature vectors extracted from this database be $\{\mathbf{g}_i\}_{i=1}^{E}$, where E is the total number of elements in the element DB. Then, the elements selected for the synthesization will be the ones that are closest to the queries $\mathbf{g}_n$, measured with Euclidean distance.

To arrange the elements in each ring (indicated by s_n), we used simple methods to visually show 4 types of arrangement: Rotation, Translation, Procedural Fill and Rigid. Figure 4 illustrates these methods. For Rotation, we duplicate and rotate the elements around the center. For Translation, we distribute instances of an element on a grid. For Procedural Fill, we randomly distribute the instances in the designated region. The scales of the element instances are varied by a small amount (± 0.5) around the predicted scales ($\mathbf{e}_n$ in the model). The spacing between elements is determined by the *density* variable (u_n in the model). Note that our Procedural Fill visualization is only to indicate that a region should be filled by an actual procedural algorithm. When there were more than two neighboring rotational rings with similar densities, we adjusted the positions of the elements so that an element in one ring would interleave with two adjacent elements in the next ring. This minor adjustment is a common practice in plate design and is not encoded in our learning model.

3.6.1. User Interface

Figure 9 shows our user interface. The interface has 4 main areas, allowing the user to interact with the synthesis engine through 3 methods. Area (1) shows the layout of the design, which may come from a database of plate layouts or be randomly generated. Area (1) also shows the final results after receiving results from the synthesizer. Area (2) displays a list of available elements, which can be customized by the user. The user can drag an element into a ring in area (1) to indicate that they want that particular element to appear there. Area (3) shows a list of recommended elements, each row containing the elements suitable for a ring in the layout that bears the same color. Below each element is a label showing how likely that element could fit in the ring. This score was generated by comparing the actual feature vector extracted from the element and the one predicted by the model—the smaller, the better. Area (3) also reacts to new inputs; as soon as the user changes some elements in the layout, new recommendations will be immediately updated. Area (4) allows the user to input all the information about a ring, including the density, the element size, the type of arrangement and the element itself.

Incremental Design Our sequence-based learning and inference engine offers an interactive way for the designer to interact with the system during the design process. In Figure 10, we illustrate this idea. In Figure 10 (A), the user dragged the first element into the innermost ring. The system then generated an array of element suggestions with accompanied ranking. The first element in the first row of each sub-figure is the input and therefore always has 0 distances from the style vectors. Each row includes a list of elements bearing the same style, predicted to be suitable for a particular ring. As the first row provides suggestions for the first ring, the user might change the first element to those alternatives suggested by the system. In step (B), the user chose the third suggestions in the second row and triggered the system generating a new list of compatible elements. In the last step, the user chose the last element and had the system do the remaining tasks such as deciding element size, arrangement and density. The final result is shown in the rightmost figure. As the array now only contains the alternatives for each ring, the user will be able to refine the results just by clicking on one of these suggestions.

4. Experiments and Results

Since our system involves multiple aspects of object decoration such as composition and visual style, we conducted experiments to study each aspect separately (Section 4.2, 4.3) and then we show how the two aspects work together under our model in Section 4.4.

4.1. The Baseline Method

Since there is no previous work that is designed for our problem, we use simple distributions estimated directly from ring data as

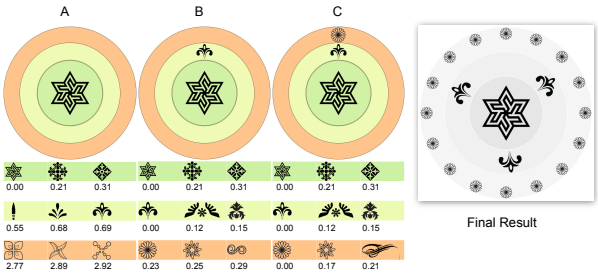

Figure 10: *A demonstration of incremental design.*

the baseline to compare with our method. Specifically, to model the style vector $\mathbf{g}_n$, we estimate a multivariate Gaussian distribution over a dataset $\mathcal{G}$ of $\mathbf{g}_n$ from all the rings in the dataset. Similarly, we estimate Gaussian distributions to model $\mathbf{e}_n$ and u_n. For the arrangement type, we use the multinomial distribution instead. Whenever required, these distributions can be used to generate a feature descriptor for an arbitrary region. The synthesis step is the same as for our method and is discussed in Section 3.6.

4.2. Composition Study

To demonstrate the capability of our system for spatial composition recommendation, we fixed the style descriptors for all the rings in a layout and let the model infer the element shape, the density and the arrangement type. From Figure 11, it is apparent that with our prediction, the elements have been arranged smoothly into the layout while the baseline results appear to be quite unsatisfactory. The first row shows a typical behavior of our model when dealing with adjacent rotational rings. That is, when a ring is too dense, the next ring usually becomes much sparser. In contrast, the baseline method was not aware of the neighboring rings and ended up leaving too much empty spaces in the outermost ring. One may notice that, when a complex element such as the woman portrait in the second row and the town in the first row are placed at the central ring, they should be rigid elements instead of a rotational element, as in the results of the baseline method. Moreover, when given a large empty space, our method figured out that it should repeat the element in a procedural way – see the first result of the last row in Figure 11. In the 2nd row, we also demonstrate the capability of interleaving elements by having two overlapping rings in the layout.

4.3. Style Study

We assess the performance of our model in predicting element style in two ways: Querying with different element styles and studying how the user perceives our synthesis results.

4.3.1. Query with Different Element Styles

In Figure 12, we demonstrate the capability of our system to recommend elements that resemble the style of the input. To produce the figure, we chose a fixed layout of 3 rings and specified one element with four different styles into the middle region. By considering the style of the input element and the layout parameters, our method predicts the style vectors for each of the empty regions. Our system recommended 4 elements for each region by measuring the Euclidean distance between the predicted style vectors and the ones in the Element DB. Each row of each example in Figure 12 is a set of 4 recommended elements; the colored rows are ordered to match the corresponding colored rings in the layout. The Euclidean distances are marked under each recommended element. The recommended elements in each row resemble the input style. For example, in (A) and (D), our system recommended elements with thin lines similar in style with the input. In contrast, the input elements in (C) and (B) have bold details and this implicit style was reflected in the recommendations. Although there is similarity in styles between the inputs and the recommendations, the elements themselves are not repeating, as repetitive elements would make the final designs look boring. The quality of the recommended elements is further examined through a user study in Section 4.3.2.

4.3.2. A User Study for Style Compatibility

To understand how much a user appreciates the set of elements recommended by our system, we conducted a user study on the compatibility of the predicted elements. To carry out the study, we first prepared a set of 10 pairs of input elements picked randomly from the Element DB. Each element was given to an arbitrary ring of a layout that contains up to 5 rings. All other factors (compositions and layout parameters) were fixed. Thus, our model made use of the given information to make predictions while the baseline method simply sampled random results from the distributions in Section 4.1. The element style vectors for the remaining rings were then generated by the models and were used to retrieve 1 element for a ring from the Element DB. We asked 11 users with different backgrounds, such as college students, designers and high school teachers, to pick a pair of designs that they preferred. Figure 13 shows a typical test appeared in our user study. Each column contains a pair of results from one of the two methods and were generated automatically given the input. The common element appear in the designs is the input element. We used a pair of images instead of one because both methods depend on randomization, which may create biases in the results. The order of appearance of each method was counter-balanced so that each method will appear on the left and on the right exactly 5 times. The result shows that the users favored our method as they chose our results 80 times out of 110 times, equivalent to 73%. A one-sample t-test showed that user preference for our method was statistically significant ($M = 7.27, t(10) = 6.33, p < 0.0001$).

4.4. Joint Composition and Style Study

We generated random examples by individually sampling from the distributions described in Section 4.1 to obtain a complete descriptor for each ring, including element style, element shape, element density and arrangement type, and then used these information to synthesize the final results (see Section 3.6). To obtain the ring descriptors with our learning model, we used the feature vector of an arbitrary ring as the input and predicted the style and the composition for other rings. The same synthesis method was used for both cases. Figure 14 shows that our results are visually more natural than the random results. In all tests, our method was able to select additional elements that go well with the input element at the middle column. Consider the first row as an example. Stylewise, our model was aware that the input element had thin and curvy lines so it predicted the style vectors to be compatible with that style, which results in the central element and middle element.

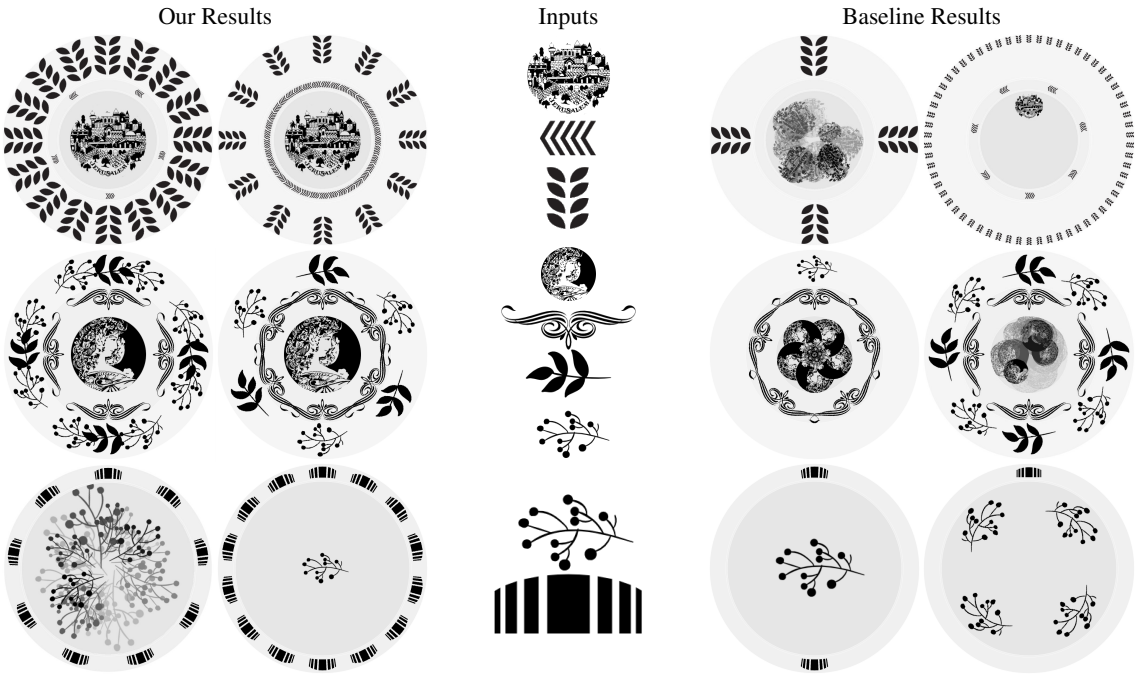

Figure 11: *The decorations were generated by fixing the input elements and letting the algorithms decide the compositions. Each row, from left to right, contains the synthesis results from our method, the input elements and the baseline results. The inputs were ordered inside-out.*

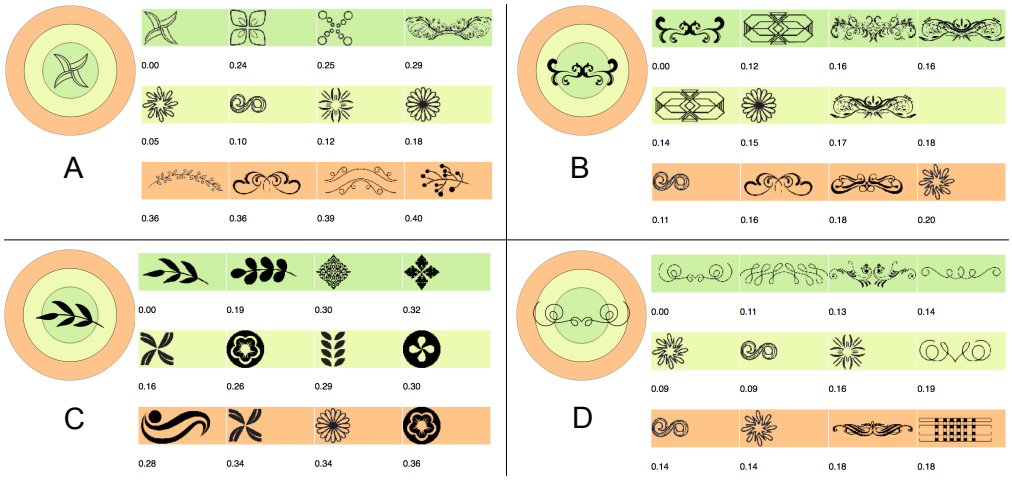

Figure 12: *Different input styles result in recommendations bearing dedicated motifs. The inputs are shown with the layout next to each recommendation array.*

Composition-wise, our method constantly predicted that the middle element should have low density—just 4 elements—which is reasonable because those predicted elements are quite long. In contrast, the elements predicted by the baseline method are quite arbitrary as there is no consistency in both style and composition between the regions. By looking at row 1, 2 and 3 at once, one might notice that our model always chooses the densities for the unknown regions so that the overall density appears neither too dense nor too sparse. When the input density is high, the adjacent regions are adjusted to be sparser. Similar behaviors can be seen in row 4 and 5.

4.5. Quantitative Experiments

In this section, we quantitatively evaluate our model by performing 4 experiments, each experiment will test whether a variable or some variables can be a good cue to predicting other variables.

1. Given the style vector and the layout parameters for all the rings $(\mathbf{g}_n, r_n, d_n)$, predict the compositions $(\mathbf{e}_n, u_n, s_n)$.
2. Given the compositions and the layout for all the rings $(\mathbf{e}_n, u_n, s_n, r_n, d_n)$, predict the style of each ring $(\mathbf{g}_n)$.
3. Given the style, the element size and the layout parameters $(\mathbf{g}_n, \mathbf{e}_n, r_n, d_n)$ of an arbitrary ring, predict the styles and compositions for other rings.
4. Given the layout (r_n, d_n), predict the style and the compositions for all rings.

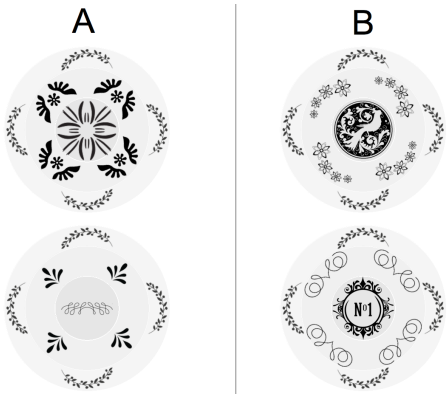

Figure 13: *A typical test image given to the user. Each column contains two designs generated by either our method or the baseline method.*

-	Test 1		Test 2		Test 3		Test 4	
-	A	B	A	B	A	B	A	B
e (MSE)	0.019	0.122	-	-	0.035	0.100	0.066	0.117
g (MSE)	-	-	0.011	0.061	0.021	0.058	0.029	0.061
u (MSE)	0.038	0.245	-	-	0.032	0.251	0.083	0.239
s (%)	92%	42%	-	-	92%	44%	71%	43%

Table 1: *Prediction results averaged over 40 random splits. A is our method and B is the baseline method. Element's size (***e***), style (***g***) and density (u) are measured with mean squared errors—the lower, the better—and arrangement type prediction was measured with accuracy—the higher, the better.*

To measure the prediction accuracy, we normalize $\mathbf{g}_n, \mathbf{e}_n, r_n, d_n, u_n$ to the range of (0, 1) and concatenate them together with s_n into a feature vector for each ring. We divided the data into training and testing sets with a ratio of $\{0.7\}$. We experimentally determined that 40 was an appropriate number of hidden states to use. The error rates for $\mathbf{g}_n, \mathbf{e}_n, u_n$ were measured with the Mean Squared Error (MSE) and the accuracy for s_n was measured in percent (%). We compare our prediction method with the baseline method mentioned in Section 4.1. As shown in Table 1, our method shows impressive performances for test (1) and (3) where it correctly predicted the arrangement type with 92% of the time. This performance has been demonstrated visually in Section 4.2. The MSE for u in for both (1) and (3) is 0.016, equivalent to an angle of 2.88 degrees, which is quite small. In Figure 15, we show how the MSE between style vectors ($\mathbf{g}_n$) is related to the visual dissimilarities between the actual elements. In test (4), as the given information is just the layout parameters r_n and d_n, we see a significant drop in the performance as only 71% of the arrangement types were correctly predicted. Along with the results in Test (3), this test implies that other factors such as the element style and the element shape are important cues to predicting the overall look of a design. Test (1) and (2) give us a hint that, given either the composition or the style of a design we can make much more accurate prediction about the other variable. This has been shown visually in Section 4.3 and 4.2.

5. Conclusion and Limitations

In this paper we presented a recommendation system for sequential design. The main contribution is twofold. First, we introduced a ring-based layout system, which is capable of describing a wide range of real-world objects such as plates, vases, pots etc. Second, we designed an HMM-based learning model to capture the rules of decorating these objects and successfully produced novel decorations that resemble the training data. Our quantitative experiments show that the quality of the decorations produced by our system is significantly better when compared to the baseline. Our work, however, suffers a number of limitations. First, our synthesis module does not consider multiple elements per region and the method to achieve Procedural Fill is relatively primitive. The lack of an adequate synthesis engine kept us from studying each of the arrangement types from the user perspective. Second, since the elements in the Element DB and in the training dataset can be different, there appears to be a misalignment between the feature vectors extracted from the two sets. Even though we have binarized the training element images before performing feature extraction, the predicted style vectors are sometimes different from the available style vectors, which could result in undesirable effects. One way to improve this is to increase the number of elements in the Element DB.

Future work: In Figure 1, we show some preliminary results for vase where we had used a ring layout similar to that in Figure 3 (Right). For demonstration purpose, we directly used a model trained on plate images for prediction. In the future, we would like to further investigate other ring-based objects such as vase, pot and decorative frames. Another promising direction is to incorporate symmetry into the model as this could be a strong cue for predicting other variables.

References

[ABS02] ANOSHKINA E. V., BELYAEV A. G., SEIDEL H.-P.: Asymptotic analysis of three-point approximations of vertex normals and curvatures. 4

[BB15] BELL S., BALA K.: Learning visual similarity for product design with convolutional neural networks. *ACM Transactions on Graphics (TOG) 34*, 4 (2015), 98. 2

[Bis06] BISHOP C. M.: *Pattern recognition and machine learning*. springer, 2006. 2, 5

[Can86] CANNY J.: A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 6 (1986), 679–698. 4

[GAGH14] GARCES E., AGARWALA A., GUTIERREZ D., HERTZMANN A.: A similarity measure for illustration style. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 93. 2, 4

[GB14] GERHARD H. E., BETHGE M.: Towards rigorous study of artistic style: a new psychophysical paradigm. *Art & Perception 2*, 1-2 (2014), 23–44. 2

[GFRF10] GRAHAM D. J., FRIEDENBERG J. D., ROCKMORE D. N., FIELD D. J.: Mapping the similarity space of paintings: image statistics and visual perception. *Visual Cognition 18*, 4 (2010), 559–573. 2

[HGR10] HUGHES J. M., GRAHAM D. J., ROCKMORE D. N.: Quantification of artistic style through sparse coding analysis in the drawings of pieter bruegel the elder. *Proceedings of the National Academy of Sciences 107*, 4 (2010), 1279–1283. 2, 4

[HZMH14] HUANG H.-Z., ZHANG S.-H., MARTIN R., HU S.-M.: Learning natural colors for image recoloring. In *Computer Graphics Forum* (2014), Wiley Online Library. 2

[JJHB*08] JOHNSON JR C. R., HENDRIKS E., BEREZHNOY I. J., BREVDO E., HUGHES S. M., DAUBECHIES I., LI J., POSTMA E., WANG J. Z.: Image processing for artist identification. *Signal Processing Magazine, IEEE 25*, 4 (2008), 37–48. 2

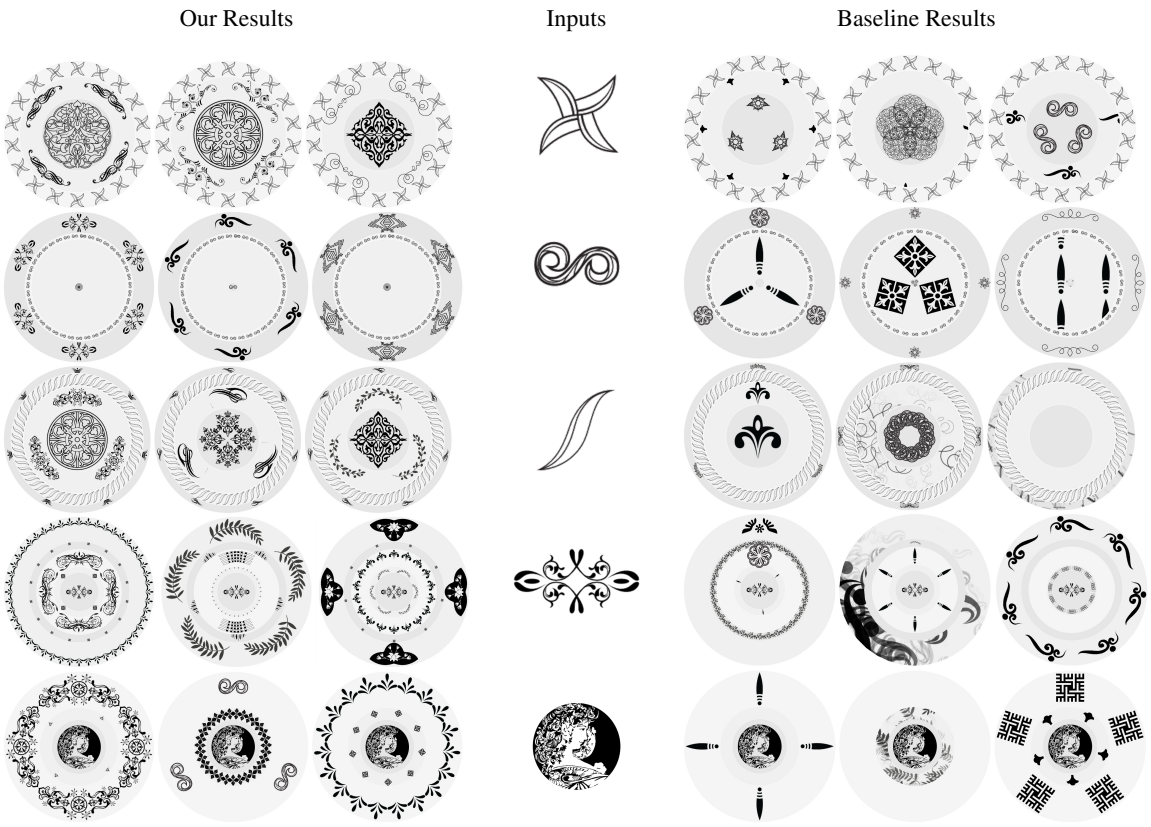

Figure 14: *Our results versus baseline results. The inputs are given at the second column. Each row contains the results produced with the same input element. The layouts were retrieved from the database.*

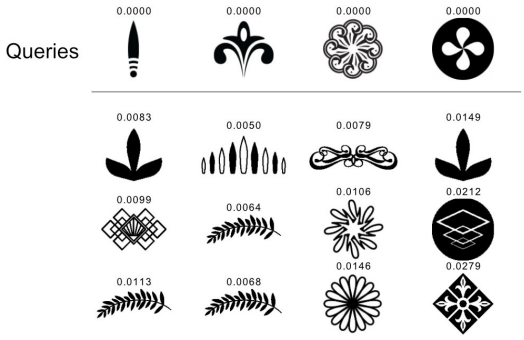

Figure 15: *Distance between element styles was measured as the Euclidean distance between our feature vectors.*

[KYKL14] KIM H.-R., YOO M.-J., KANG H., LEE I.-K.: Perceptually-based color assignment. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 309–318. 2

[LA15] LANG K., ALEXA M.: The markov pen: online synthesis of free-hand drawing styles. In *NPAR* (2015), pp. 203–215. 2

[LBW*14] LU J., BARNES C., WAN C., ASENTE P., MECH R., FINKELSTEIN A.: Decobrush: Drawing structured decorative patterns by example. *ACM Trans. Graph. 33*, 4 (July 2014), 90:1–90:9. 2

[LC09] LI C., CHEN T.: Aesthetic visual quality assessment of paintings. *Selected Topics in Signal Processing, IEEE Journal of 3*, 2 (2009), 236–252. 4

[LKS15] LUN Z., KALOGERAKIS E., SHEFFER A.: Elements of style: learning perceptual shape style similarity. *ACM Transactions on Graphics (TOG) 34*, 4 (2015), 84. 2

[LRFH13] LIN S., RITCHIE D., FISHER M., HANRAHAN P.: Probabilistic color-by-numbers: Suggesting pattern colorizations using factor graphs. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 37. 2

[OAH15] O'DONOVAN P., AGARWALA A., HERTZMANN A.: Designscape: Design with interactive layout suggestions. In *CHI* (2015), ACM, pp. 1221–1224. 2

[Owe00] OWEN C.: *Creative Ceramic Painting*. 2000. 2

[RJ93] RABINER L., JUANG B.-H.: *Fundamentals of speech recognition*. Prentice hall, 1993. 4

[SMO*10] SHAMIR L., MACURA T., ORLOV N., ECKLEY D. M., GOLDBERG I. G.: Impressionism, expressionism, surrealism: Automated recognition of painters and schools of art. *ACM Transactions on Applied Perception (TAP) 7*, 2 (2010), 8. 2, 4

[VZ05] VARMA M., ZISSERMAN A.: A statistical approach to texture classification from single images. *International Journal of Computer Vision 62*, 1-2 (2005), 61–81. 4

[WWY14] WU R., WANG W., YU Y.: Optimized synthesis of art patterns and layered textures. *Visualization and Computer Graphics, IEEE Transactions on 20*, 3 (2014), 436–446. 2

[WZS98] WONG M. T., ZONGKER D. E., SALESIN D. H.: Computer-generated floral ornament. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques* (1998), ACM, pp. 423–434. 2, 3

[YBY*13] YEH Y.-T., BREEDEN K., YANG L., FISHER M., HANRAHAN P.: Synthesis of tiled patterns using factor graphs. *ACM Transactions on Graphics (TOG)* (2013), 3. 2