Web-based Graphic Design Framework to Support Users by Intuitively Reusing and Sharing Abstract Appearance Graphs

Nodoka Yamamoto
Faculty of Environment and Information Studies
Keio University
Fujisawa, Kanagawa, Japan
e-mail: mild.summer.y@gmail.com

Shuichi Kurabayashi
Faculty of Environment and Information Studies
Keio University
Fujisawa, Kanagawa, Japan
e-mail: kurabaya@sfc.keio.ac.jp

***Abstract*— We propose a novel Web-based graphic design system that utilizes data-driven techniques to share and reuse the practitioners' knowledge of graphic design. A unique feature of this system is a user interaction model that utilizes an image being edited as a query for the knowledge base in order to recommend colors suited to it. The system stores images using a highly abstract graph structure, which we refer to as an Abstract Appearance Graph (AAG). The content being drawn at any given time is also modeled as an AAG. We have thus implemented a vector graphics design environment as an interactive graph database retrieval process to reuse existing designs. Our current prototype supports the extraction and reuse of the appearance properties of AAGs: the gradient of colors, their size, and positional relations. We implemented a cloud-based prototype system to create and share AAGs by using modern HTML5 technology in order to show the feasibility of our approach. We also performed experimental studies with impressional (*kansei*) graphic design knowledge. The experimental result shows that our system effectively supports graphic design by bringing to bear existing knowledge on new designs and rendering graphic design more flexible.**

Keywords- Graphic design; Color scheme; Image processing; Recommendation system;

I. INTRODUCTION

There is no end to the evolution of art and design. With the remarkable development of information and software technologies in recent times, it has become easier than ever before for anyone to create and share art. At the same time, however, this implies that anyone has come to be assessed and required artistic/aesthetic quality. It is evident that art and beauty are significant for a society [1]. Most universally, people recognize arts and aesthetic works as valuable things. Therefore, in today's society, it is necessary to constantly investigate ways to create high-quality works of art and design for anyone with advanced technology.

In the fields of art and design, a good sense in an artist or designer is accomplished only after having acquired considerable knowledge. In many cases, works of art and design consist of imitations of famous works. Artists and designers need to be versed in numerous technical details and styles used in existing works in order to create novel works of art and design by referring to them through a process of trial-and-error. Conventional works of art and design are sources of new works, so we can say that sharing more works contributes to more growth of art and design.

In light of the above, our aim in this study is to support graphic design by helping users effectively share and reuse existing works, which constitute design knowledge. In order to reuse conventional works for inspiration, artists and designers are required to abstractly comprehend the original author's intention through his/her works, and this accordingly requires technique. Nowadays, we have access to massive amounts of graphic design resources that are shared around the world. However, little research has been devoted to utilizing the large quantity of graphics-related resources available to us for the benefit of graphic design. While these innumerable resources have contributed to the worldwide growth of graphic design, the resources available to people at large are quite limited at present. Hence, a systematic framework that directly extracts the essence of graphic designs is necessary, not only for the growth of graphic design but also that of art.

Our concept is enhancement of graphic designer's creativity by systematizing the memory retrieval and the reuse of existing graphic design works. Our system supports graphic designers by suggesting design techniques and knowledge during his/her work. Therefore, in order to suggest technique of the graphic design for user's work interactively, we propose database construction methods, which analyze existing works for extracting technique and knowledge, and database retrieval methods.

In this paper, we propose a systematic way for graphic designers to accomplish the following: inputting and storing existing graphic designs, understanding and extracting typical features from stored graphics, and reusing the extracted features to create new graphic designs. To realize these objectives, we develop a new vector-based drawing tool on the Web specifically intended to facilitate the sharing and reusing of design knowledge possessed by designers. This knowledge consists of abstract and reusable features of existing graphic designs stored in an embedded database system. The system generates a query by capturing the status of a drawing in progress, which is submitted to the knowledge base to detect the most appropriate information related to the current drawing.

The most important feature of our method is a new data structure used to represent the flow of colors. It represents both the spatial distances in an image and the color distances

Copyright (c) IARIA, 2015. ISBN: 978-1-61208-412-1

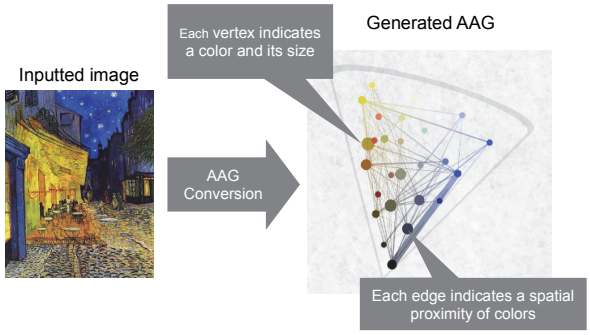

Figure 1. An example of an Abstract Appearance Graph

measured according to a color metric in existing images. We refer to this data structure as an Abstract Appearance Graph (AAG). Representing the structure of colors, an AAG can be visualized in an intuitive manner, where it models color construction on a bitmap image by analyzing the relationship among all pixels using color distance and spatial distance. In general, the impression of colors is heavily affected not only by their discrete features, but also by their inter-relationships. This is because the visual stimulation of human senses is caused by relative mechanisms. Therefore, we assume that utilizing such graph structures is appropriate to comprehend the color design of graphics.

Figure 1 shows an AAG model, where a vertex represents a color, the size of the vertex denotes the color ratio in the original image, and an edge represents the spatial proximity between any two colors in the original image. Thus, the AAG represents the color status of image as a set of color vertices and edges connecting them.

In our proposed method, when a user creates a new graphic, the system recommends a relevant AAG that reflects the distribution of colors in the user's current work through a real-time analysis of its color structure. This smart graphic design environment also helps a user choose appropriate colors by reviewing his/her previous work, motifs, as well as the stored graphic design knowledge of all users. Our Web-based AAG sharing/reusing method is applicable to many fields, such as paint tools, Web design tools, and fashion outfitters because our method offers a smart method to determine sets of colors and distances between colors.

The remainder of this paper is structured as follows: Section 2 contains a summary of related research, and Section 3 is devoted to a structured overview of our system. We define the related fundamental data structure and algorithms in Section 4, and describe the implementation of our prototype system in Section 5. Section 6 contains an evaluation of the usability of our system, and we offer our conclusions in Section 7.

II. Related Works

With the recent advances in data processing, it has become popular to study design support systems, such as color scheme evaluation systems [2], color scheme support systems [3] [4], and automatic design support systems [5]. These systems exploit the relationships between colors and their evocative features by using the Color Image Scale [6] or fuzzy rules including it. The Color Image Scale [6] seeks to distill the emotional effects of color combinations, and consists of over 1,000 three-color combinations of 130 basic colors matched with key image words designed to reflect any mood or lifestyle. The Color Image Scale forms the basis of numerous studies and projects nowadays. Researchers studying color scheme support systems [3] [4] proposed a method to recommend harmonious color schemes from an input color and a keyword by utilizing fuzzy color schemes. These approaches are highly likely to correspond with direct user demand because they are based on textbook theories of design. However, because there is no "right" answer in the field of graphic design, supporting graphic design through specific rules cannot satisfy the designer's need for expansive and innovative work. From a broader perspective, a more open and flexible design support system is necessary to improve the design process.

Adobe Color [7] is a well-known service for creating and sharing color themes comprising five colors. Several color theme extraction methods have been proposed [8] [9] [10] by investigating sample models created by users. In order to reuse design knowledge, past research [11] [12] [13] has proposed methods that reflect the color scheme or color tone of an existing image in a target image. Furthermore, many community services have been developed to share art/design works, for example Dribbble [14], Behance [15], and Pixiv [16]. Indeed, there is considerable demand for means of sharing intricate knowledge in graphic design.

Based on the foregoing, we propose in this paper a systematic and direct method of supporting graphic design through a highly extensible and flexible social framework. An AAG, the data structure of graphic design in our system, is created from existing graphic design works selected by users as knowledge of color schemes. Therefore, the system permits all users to extend and benefit from the rule base as a knowledge sharing tool, as in [7] [14] [15] [16]. Prevalent research on design support, such as [3] [4] [5], does not have this feature. Furthermore, because an AAG consists of relationship among colors, the system can interactively support users' design work by recommending "flows of colors:" feasible colors for the work at hand. Thus, our system can construct a graphic design framework exhibiting extensibility and immediacy, and offers new possibilities for the development of graphic design.

III. System Architecture

The objective of our system is to recommend an appropriate color scheme for a user in real time by analyzing the color structure of the drawing in question in an on-the-fly manner. As shown in Figure 2, our design supporting mechanism consists of two phases: a storing design knowledge phase, and a real-time design support phase.

In the storing graphic design knowledge phase, our system constructs a database that stores design knowledge by converting existing graphic design objects, such as bitmap images, into simpler graph structures of color schemes. This color graph structure is the Abstract Appearance Graph. The AAG comprises the appearance properties (for example,

Copyright (c) IARIA, 2015. ISBN: 978-1-61208-412-1

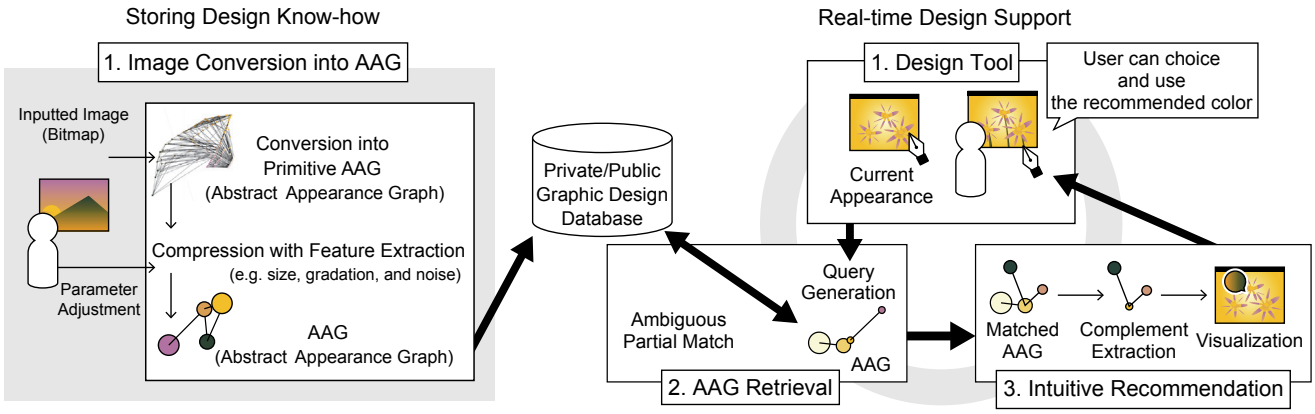

Figure 2. System architecture of the graphic design framework using AAG.

color and texture) and the positional relationships of colors. The system provides the user with an image conversion function, because of which the graphic design database of the system can be freely extended. When the system generates an AAG from an existing graphic design image, it analyzes the proximity and the contiguity between colors to detect the design elements of the image.

The AAG describes the proximity and contiguity between colors using vertices and edges. We can detect the spatial and semantic distance between colors by counting the number of hops between two vertices. The details of the AAG generation process are described in Section IV. It is important to render data structures simpler for accurate retrieval calculation because a pure bitmap image includes manifold colors in many cases. The graphic design storage module then stores the generated AAGs.

In the real-time design support phase shown in the right side of Figure 2, the system implements an interactive collaboration between a design tool and an AAG retrieval engine. The design tool provides functions for editing and creating graphic designs containing color schemes. It is important to mention that this design tool contains a function to convert current drawing content into an AAG data structure. The retrieval engine calculates the correlation between the working appearance of the design tool and the AAG objects stored in the graphic design knowledge storage module in order to obtain design elements relevant to the appearance of the drawing at hand. Following this, the retrieval engine performs an ambiguous partial match to extract reusable elements from the AAG obtained. The visualization interface then intuitively displays the obtained graphic design elements that complement the current drawing.

IV. Data Structure and Algorithms

As described in the previous section, AAG is a core data structure of our system. An AAG is represented as a graph $G = (V, E)$, where V is a set of vertices that denote simplex color elements in the image and E are edges that denote spatial proximity between colors (vertices) in the image. Each vertex has RGB color components and a size (a ratio to the image), and each edge has an angle of spatial proximity. In this section, we describe the algorithm to create an AAG object from an image and the retrieval function that scours the AAG database for design elements complementary to a drawing being edited using the design tool.

A. Conversion from Image to AAG

The AAG is constructed through the following two processes: gradient abstraction process and integration of similar vertices and edges. We detail the following conversion method on the assumption that the image to be converted is a bitmap image. A vector image data can then be easily converted into an AAG because its structure is already simple.

Process-1) Gradient Abstraction Algorithm: We designed a bottom-up abstraction method — in other words, a redundancy and noise reduction method — to convert a bitmap image into an AAG. The system aims to generate highly abstract data that includes texture information, such as gradient and noise, by analyzing relationships between adjacent colors. Our method uses the following algorithm to discover and abstract the gradient parts of an AAG. This is a recursive algorithm that repeats the analysis of three vertices.

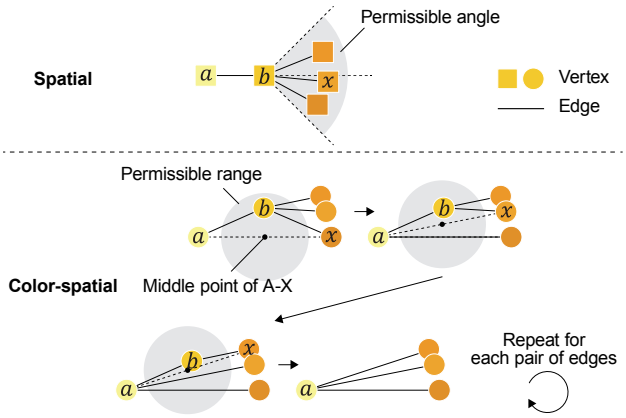

Figure 3. Schematic view of the gradient abstraction algorithm of the AAG.

Step 1) Convert an image into a color graph. Each vertex has a color and a size, which represents the pixel count. Each edge has an angle, a hop count (the initial value is 1), and gradient length (the initial value is 0). The initial angle of an edge is in units of 45 degrees because of contiguity on pixels, and edges consisting of the same colors at different angles are regarded as different edges.

Step 2) This step is the core phase of this algorithm. The system removes a redundant edge and introduces a directly connected edge instead. As shown in Figure 3, when we have three vertices (a, b, x) and two edges $\{a, b\}$ and $\{b, x\}$, the system deletes $\{b, x\}$ (and the opposite edge) and creates $\{a, x\}$ (and the opposite edge) if they meet the following conditions:

- We define the threshold for the difference between the magnitude of the angles formed by $\{a, b\}$ and $\{b, x\}$ as 45 degrees. This threshold is used to detect pixels in the same gradient direction in the bitmap image.
- In Euclidean color space, the distance between b and the midpoint of ax does not exceed the (constant rate of the) length of ax. This calculation uses RGB color components because the graphics software usually generates linear gradients in RGB space.

The newly created edge $\{a, x\}$ has the following properties: the angle is an average of the previous edges $\{a, b\}$ and $\{b, x\}$, and the number of hops is a summation of $\{a, b\}$ and $\{b, x\}$ because the new edge $\{a, x\}$ maintains the neighborhood relationships of pixels corresponding to the vertex b. The gradient length increases by the summation of the gradient lengths of the two edges because the previous edges are grouped into the gradient.

Following this, the system removes vertex b and the corresponding edges $\{a, b\}$ and $\{b, a\}$ if and only if vertex b is isolated from all vertices, except vertex *a*, by the above process.

Step 3) Repeat Step 2 until no edge meets the conditions.

Process-2) Integration of Similar Vertices and Edges: Using the above gradient abstraction method, the system reduces the level of detail in the AAG, by integrating smaller vertices into a larger vertex. The system calculates the distance between all vertices in the AAG and, the calculated distance is smaller than the predefined threshold if and only if the system integrates a smaller vertex into a larger one. The detailed process is as follows:

Step 4) The system measures the distance between every pair of vertices in the AAG. If and only if the distance is smaller than a threshold, the system integrates the smaller vertex into the larger vertex, which is a neighbor of the target vertex. This new vertex has a larger size than either of the original vertices because the size of the smaller vertex is added to that of the larger one. Due to this size control principle, a user can know the size of the color represented by the vertex in the original image. The vertices of our AAGs have two properties besides a color. The first is a size property calculated by summing up the pixel areas of same (or similar) colors. The second property is a range property regarding the area to which a specific vertex relates. The system calculates this range property by choosing the larger value from a range property of the larger vertex, and the distance between the smaller vertex and the larger one. The edges connecting the original smaller vertex are then shifted to the new, larger vertex.

Step 5) Every pair of edges connecting each vertex, whose angles are closer than a threshold, are integrated. When two edges are integrated, their properties are determined according to Step 2 of the gradient abstraction method.

B. AAG Retrieval

The AAG data structure plays two roles in this system. The first role is referred to as the "working AAG," which appears on the image drawn by the user. The second role is referred to as the "model AAG," which is obtained from existing images and is used as a source of design knowledge. Design support is achieved by recommending highly relevant sub-graphs from the model AAG, where the colored vertices in the working AAG are used as the starting point. To obtain sub-graph recommendations, the system calculates the relevance between vertices in the following manner:

Step 1) Selecting vertices in working AAG: The system selects a set of vertices from the working AAG, denoted as $V \in W$, where V are the selected vertices specified using a color and W is the working AAG. The system calculates the distance between each vertex in the model AAG, denoted as M, and each vertex in V. Each vertex, which is denoted as v, contains RGB color information. Moreover, each vertex in M has a range property. We implement the distance of vertices equation as in (1):

$$dist(v_w, v_m) := x \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$$
$$x := \sqrt{(r_w - r_m)^2 + (g_w - g_m)^2 + (b_w - b_m)^2} - range_m \quad (1)$$

where v_w denotes the vertex in V and v_m denotes the vertex in M, r_w, g_w, and b_w are the components of v_w, and r_m, g_m, b_m, and $range_m$ are the components of v_m. This is a prototype implementation, and we plan to apply the CIEDE2000 delta equation [17] to improve the precision of distance calculations.

Step 2) Selecting the most relevant model AAG: The system selects the most appropriate AAG by calculating the relevance scores between the model AAGs and the current working AAG using (2):

$$score := \sum_{i=1}^{m} (t - d_i) \;\middle|\; d_i \leq t\,, \qquad d_i \leftarrow MIN\big(dist(i, \forall x \in M)\big) \quad (2)$$

where t denotes the threshold, i denotes the *i*-th vertex in the current working AAG W, and $\forall x \in M$ denotes each vertex in the model AAG M.

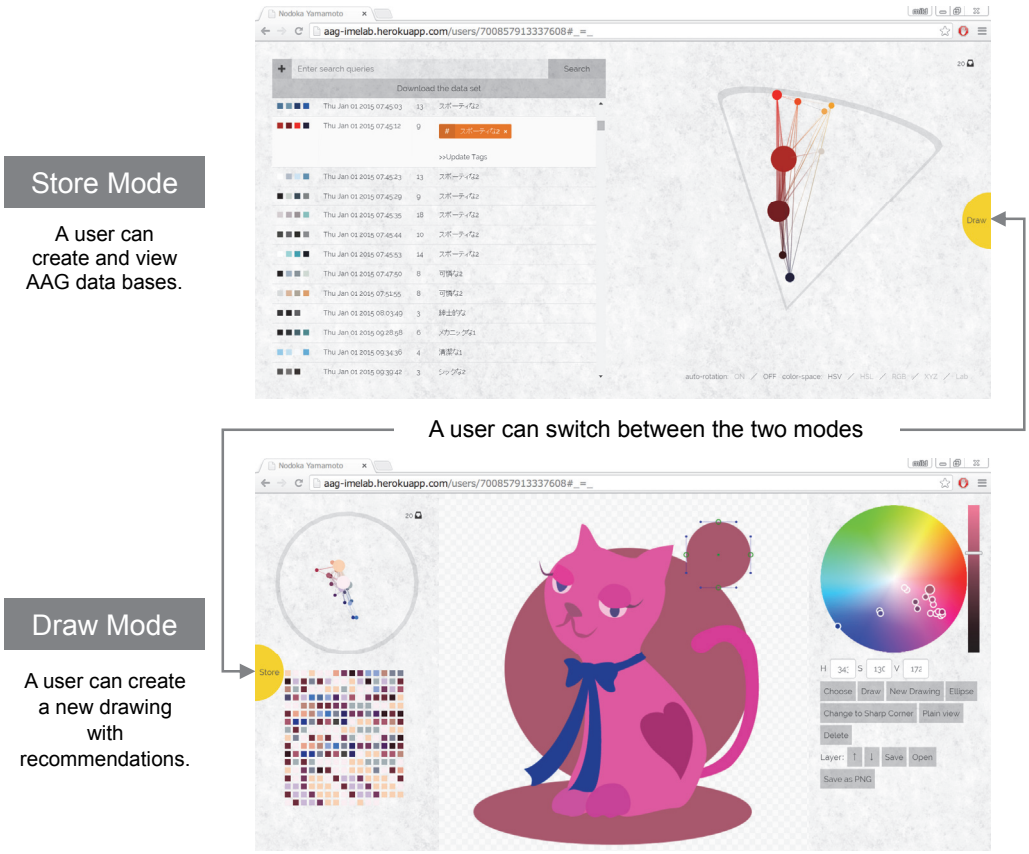

Figure 4. Prototype implementation system of the graphic design environment with HTML5 technology.

Step 3) Selecting a connected vertex in the selected model AAG: As the final step in design support, the system finds the connected vertex and places the partial graph of the selected model AAG into the current working AAG. The system retrieves the target vertices from the working AAG according to the relevance score. If the distance between a vertex in the model AAG and the working AAG is below the threshold t, the system selects it as a candidate. Thus, the system obtains the candidate vertices, which are denoted by $C \in M$. The system then obtains the vertices that are neighbors of each vertex in C. Finally, the system displays these neighbor vertices on the current drawing of the design tool as recommendations.

V. IMPLEMENTATION

In this section, we describe a prototype implementation system of our study.

A. Prototype System

We developed a prototype system using the modern Web technologies HTML5 Canvas, WebGL with Three.js, and d3.js, as shown in Figure 4. The interface of the prototype system comprises two modes: a store mode for accumulating the AAG database, which contains an AAG conversion module, and a draw mode for creating a new graphic by utilizing the AAG database, which contains a graphic design tool module, an AAG retrieval module, and an intuitive proposal module. A user can switch between the both modes at will.

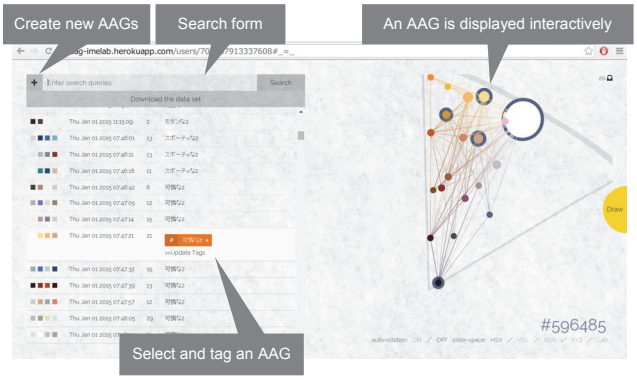

Figure 5. Store mode of the prototype system: a list and 3D visualization of AAG.

1) Store Mode

In the store mode, a user can load existing images to construct his/her own AAG database. A list of AAG objects created by the user is displayed on the left side of Figure 5,

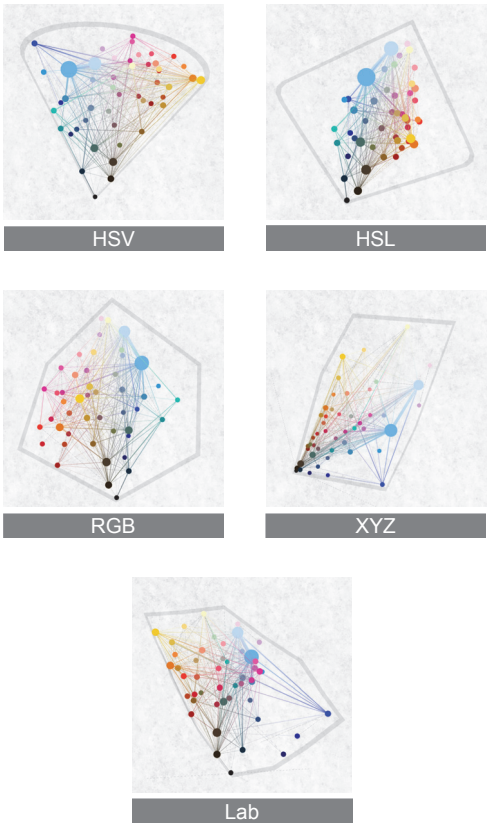

Figure 6. Five kinds of visualization color space in the prototype system.

and a 3D color space visualizing the relationships among colors in an AAG is displayed on its right side. The user can select bitmap images as new graphic design knowledge by clicking the "+" button. The system converts the input images into AAG objects, following which the AAG is represented in 3D color space developed using WebGL.

In this 3D visualization, the coordinates are equivalent to a color (color components). Each vertex of the AAG is plotted at its color coordinates regardless of its absolute position on the original image, and the relationships between colors are represented as edges connecting vertices. The dashed lines represent normal edges, and graded relations between two colors are indicated as gradient lines with a thickness corresponding to the gradient property. A user can interactively control the viewpoint. When a user clicks a color vertex, the connected vertices of that color are highlighted. Further, the user can choose the kind of the visualization color space from HSV, HSL, RGB, XYZ, and Lab, as shown in Figure 6. Hence, the user can easily grasp the color structure of the AAG.

The converted AAG can be stored in the database. In the store mode, a user can tag his/her own AAGs with appropriate words, and can search AAGs using tags from all users' databases. When the "Download the data set" option is selected, the listed AAG objects are downloaded and stacked as dataset of the retrieval module in the draw mode. In order to take advantage of its extensibility, one of the

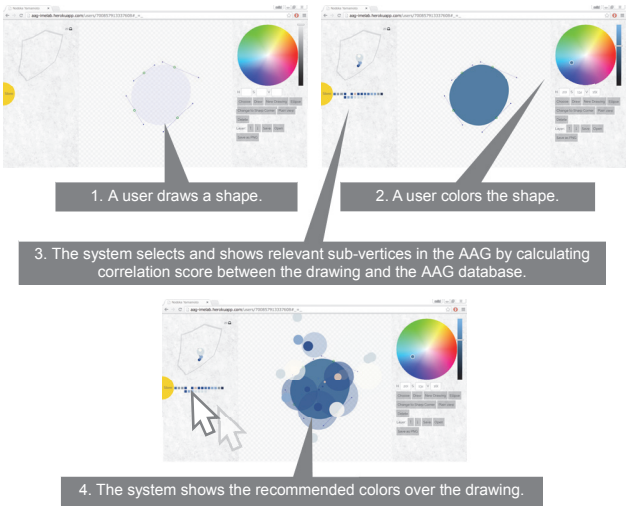

Figure 7. UI flow in the draw mode of the Web-based prototype system.

important features of our system, the dataset is not treated as a static entity. The system extracts the AAG dataset through queries.

2) Draw Mode

In the draw mode, as shown in Figure 7, a user can draw an image consisting of Bezier curves using a graphic design tool that we developed on HTML5 Canvas. When a user provides or changes the color of a shape, the system searches for complementary AAG objects in the dataset. The interface then displays the color swatches of neighboring vertices that correspond to the colors on the current drawing. When the user positions the mouse pointer over it, the color is intuitively overlaid and dynamically reflects the size, distance, and angle in the model AAG. A user can thus re-color a shape by dragging and dropping the color swatches to the current drawing space.

Moreover, it is an important merit of our Web-based prototype implementation that the system can record all users' operation histories, such as the process according to which a user created a work, and the recommendation adopted by him/her for a work.

B. Use Case

We now present a use case and a use flow of the prototype system. The use case involves a user, Sharaku, who wants to create a drawing like Ukiyo-e, a genre of Japanese antique woodblock prints and paintings.

Sharaku first needs to construct the graphic design knowledge base of Ukiyo-e in store mode. Figure 8 shows that Sharaku has 20 favorite pictures from the genre Ukiyo-e; he enters them with the tag "ukiyo-e," and the system converts the input images into AAG objects without pausing, and lists them as shown in Figure 9. From this, the graphic design knowledge base associated with "ukiyo-e" is created and is open to the public. The AAG dataset for the recommendation system is then downloaded and stacked when the user clicks the "Download the data set" button.

Copyright (c) IARIA, 2015. ISBN: 978-1-61208-412-1

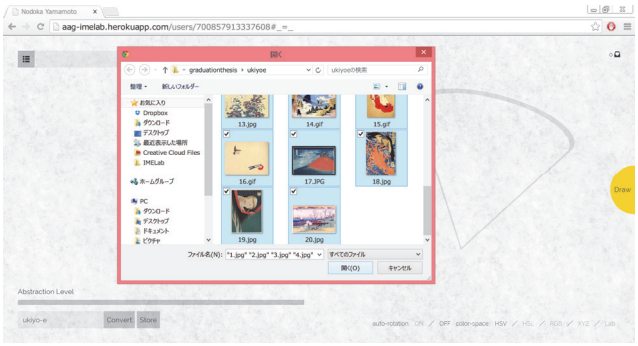

Figure 8. Use Case: Selecting Knowledge Images of *Ukiyo-e* in the Store Mode

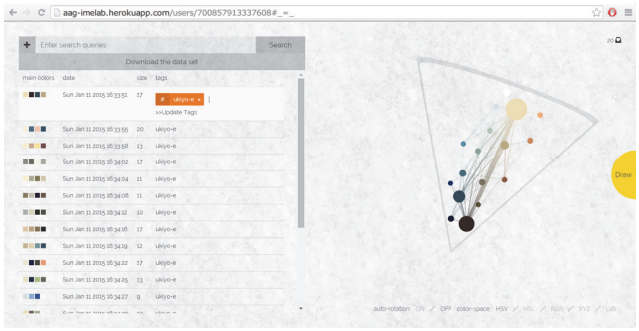

Figure 9. Use case: The list of created AAGs in store mode.

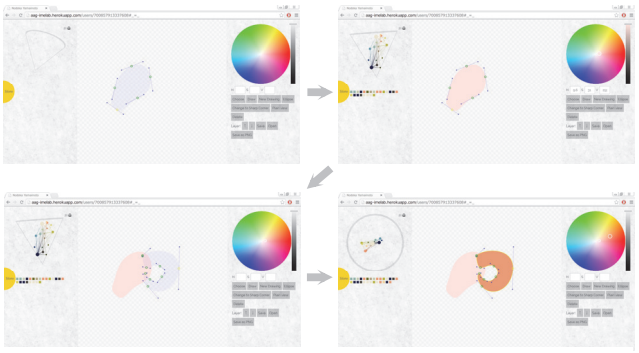

Figure 10. Use case: The flow of the drawing with recommendations in draw mode.

Figure 11. Use Case: The drawing of a sheep in the style of Ukiyo-e.

Sharaku can now create a drawing in the Ukiyo-e style using the color scheme support of the system in draw mode. He shifts to draw mode with the ukiyo-e dataset and starts working.

Sharaku is thinking of drawing the face of a sheep in the Ukiyo-e style. When he draws a shape resembling the face of a sheep and colors it beige, the system retrieves an AAG with a color matching the face color and automatically shows color swatches, as shown in Figure 10. All Sharaku has to do is pick up the ones he likes. Of course, if there is no appropriate AAG for the work at hand, the system does not recommend colors.

When Sharaku moves the mouse cursor over the swatches, the colors are dynamically displayed around the shape. In this case, when the mouse pointer hovers over the navy swatch, a big navy circle appears on the given shape.

Using the recommendation in this case, Sharaku adopts dull orange as the color of the sheep's horns. He drags and drops the swatch on the horn. The system once again retrieves an AAG following this color assignment.

In this manner, Sharaku creates a drawing using real-time recommendation from the system, as shown in Figure 11.

VI. EVALUATION

In this section, we report an experiment to examine the effectiveness of our proposed system using our prototype implementation. The purpose of the experiment was to determine whether the AAG-based system adequately conveys graphic design knowledge from user to user and is sufficiently flexible for graphic design.

The experiment consisted of three phases: collecting images for design knowledge base, creating graphic works with the design knowledge base, and assessing the created works. We will explain these phases in order followed by the result of the experiment.

We assig(without/with recommendations).

ned seven subjects the task of collecting four groups of graphic design images from Internet severally. Each subject was provided with four impressions in order to create impressional (*kansei*) datasets for color recommendations. For example, a certain subject collected "clean," "modern", "energetic" and "antique". The four impression words were selected in order to not be similar to each other. We also told the subjects that the selected images should provide the relevant impression using colors because an AAG does not contain any information, such as shapes and structures, except colors. For example, images of one's favorite dishes are not an adequate dataset to represent the word "tasty" in this experiment, whereas images that appear "tasty" to many people according to their color distribution are considered fit. We thus created 28 AAG datasets from the collected graphic design images.

Following this, we assigned seven subjects the task of creating four pairs of "cat" or "dog" drawings conveying either impression, using the draw mode of our prototype system in two ways: with and without recommendations from the impressions dataset. That is, each subject had to create eight drawings. We distributed the tasks such that a subject who had collected images according to an impression

Copyright (c) IARIA, 2015. ISBN: 978-1-61208-412-1

Figure 12. Samples of drawings created by subjects in the evaluation experiment.

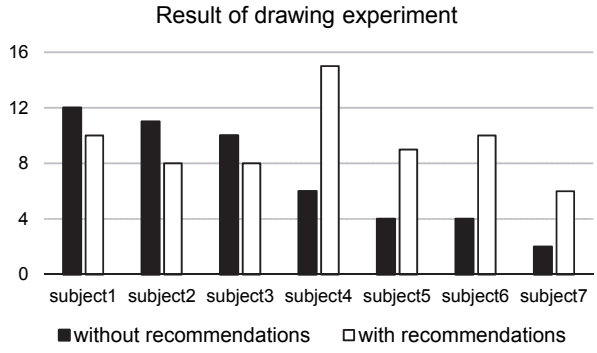

Figure 13. Seven subjects' scores in the drawing experiment.

Figure 14. Drawings created by Subject 1.

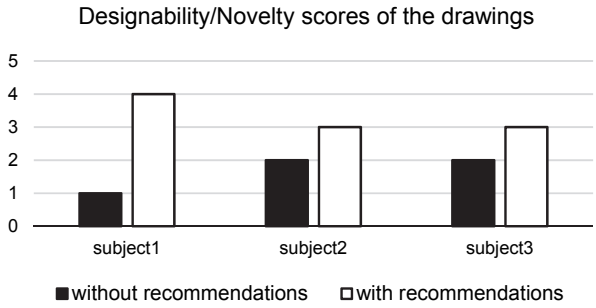

Figure 15. The result of novelty assessment in the drawing experiment.

in the previous phase was tasked to create images corresponding to a differ impression in this phase. Figure 12 shows sample images drawn by the subjects.

In the third phase of the experiment, we asked seven subjects to assess whether the created drawings conveyed the relevant impressions. Each assessor for an impression was the subject who had collected images according to the impression in the first phase of the experiment. For example, the subject who collected images corresponding to the impressions "lively," "fresh," "elegant," and "natural" in the first phase assessed drawings corresponding to these impressions created by another subject in the second phase. The manner of assessing drawings was as follows: the assessor assigned a decreasing order of preference to four drawings that conveyed each impression, of the eight drawings created for each impression. We then scored each subject who had created the drawings according to the evaluator's choice and the intended impression. We accorded points depending on the order of preference: four points when an image was accorded first choice, three points when it was chosen second, two points for third choice, and one point for fourth choice. Each subject who created the drawings had two scores: without/with recommendations.

Figure 13 shows the experimental result by assessing the impressions of the drawings. The vertical axis indicates the scores of the seven subjects who created the drawings according to the two methods (without/with recommendations).

The most important feature of the result is the difference in effect by the recommendations between the subjects who originally had high scores and those who originally had low scores. Our system notably supported subjects not having adequate drawing ability: subject4, subject5, subject6, and subject7. This means that the recommendations of our system conveyed design knowledge of the original images from one user to another. In contrast to results for these subjects, our system to some extent reduced the scores of subjects with high powers of representation: subject1, subject2, and subject3. This implies that our system is not universally sensitive to artistic capacity in supporting the creation of graphic designs.

In order to determine the effects of our system on users possessing high drawing ability, we selected the drawings by subject1, subject2, and subject3, and asked five subjects to compare the designability and novelty of each two groups of the drawings. Figure 14 shows examples of drawings created by subject1, and Figure 15 shows the results of this assessment. The vertical axis of Figure 15 indicates the number of times the drawings without/with recommendations were assessed more highly than the others.

Although assessing designability and novelty is exceedingly difficult, the results show that our system helps users render graphic design works more flexible and designable using existing design knowledge. The novelty and universality of impressions are contrary properties. Thus, the results of this experiment imply the hybrid nature of our system because it simultaneously helped subjects 1, 2, and 3, who had good drawing skills, to draw higher-quality images, as well as subjects 4, 5, and 6, who were novices, to draw images of acceptable or adequate quality.

VII. Conclusion and Future Works

In this study, we proposed a framework to share and reuse graphic design knowledge on the Web. We

Copyright (c) IARIA, 2015. ISBN: 978-1-61208-412-1

implemented the system using a novel Web-based graphic design system that utilizes data-mining techniques to abstract design features from existing image data. A unique feature of this system is a Web-based user interaction model for drawing graphics, where a user's operations are enhanced by extracting and reusing color schemes in a dynamic manner. We performed experiments to examine the effectiveness of our system by publishing it as a modern HTML5 application on a cloud infrastructure. Experimental result shows that our system can support users by conveying graphic design knowledge and rendering graphic design works more flexible. In future research, we plan to improve the design abstraction method and the color retrieval function in order to provide more effective recommendations according to the needs of the user.

REFERENCES

[1] K. Sasaki, Invitation to Aesthetics, Tokyo: Chuokoron-Shinsha, 2004.

[2] M. Tokumaru, and K. Yamashita, "Color Coordinate Evaluating System Using Fuzzy Reasoning," Trans. IEICE, Japan, vol. J83-D2, no. 2, Feb. 2000, pp. 680–689, ISSN: 0915-1923.

[3] M. Tokumaru, N. Muranaka, and S. Imanishi, "Color Design Support System Considering Color Harmony," FUZZ-IEEE'02, May. 2002, pp. 378–383, doi: 10.1109/FUZZ.2002.1005020.

[4] M. Tokumaru, and N. Muranaka, "An Evolutionary Fuzzy Color Emotion Model for Coloring Support System," FUZZ-IEEE'08, Jun. 2008, pp. 408–413, doi: 10.1109/FUZZY.2008.4630400.

[5] A. Jahanian, et al. "Recommendation System for Automatic Design of Magazine Covers," 18th International Conference on Intelligent User Interfaces (IUI'13), Mar. 2013, pp. 95–106, doi: 10.1145/2449396.2449411.

[6] S. Kobayashi, Color Image Scale, Tokyo: Kodansha Intern, 1992.

[7] Adobe Systems Incorporated. *Adobe Color*. [Online]. Available at: https://color.adobe.com/ 2015.01.19.

[8] S. Lin, and P. Hanrahan, "Modeling How People Extract Color Themes from Images," ACM Human Factors in Computing Systems (CHI 2013), Apr. 2013, pp. 3101–3110, doi: 10.1145/2470654.2466424.

[9] J. Delon, A. Desolneux, J. L. Lisani, and A. B. Petro, "AUTOMATIC COLOR PALETTE," Inverse Problems and Imaging (IPI), vol. 1, no. 2, May. 2007, pp. 265–287, doi:10.3934/ipi.2007.1.265.

[10] P. Obrador, "Automatic color scheme picker for document templates based on image analysis and dual problem," SPIE's International Symposium Medical Imaging 2006 (SPIE 2006), vol. 6076, Feb. 2006, pp. 64–73, doi: 10.1117/12.647075.

[11] T. Kagawa, H. Nishino, and K. Utsumiya, "A color design assistant based on user's sensitivity," IEEE International Conference on Systems, Man & Cybernetics (SMC 2003), Oct. 2003, pp. 974–979, doi: 10.1109/ICSMC.2003.1243941.

[12] Y. Chang, S. Saito, K. Uchikawa, and M. Nakajima, "Example-Based Color Stylization of Images," ACM TAP, vol. 2, pp. 322–345, July. 2005, doi: 10.1145/1077399.1077408.

[13] B. Wang, Y. Yu, T. Wong, C. Chen, and Y. Xu, "Data-Driven Image Color Theme Enhancement," SIGGRAPH ASIA '10, Dec. 2010, no. 146, doi: 10.1145/1866158.1866172.

[14] Dribbble LLC. *Dribbble*. [Online]. Available at: https://dribbble.com 2015.01.19.

[15] Adobe Systems Incorporated. *Behance*. [Online]. Available at: https://www.behance.net 2015.01.19.

[16] pixiv Inc. *pixiv*. [Online]. Available at: http://pixiv.net/ 2015.01.19.

[17] M. R. Luo, G. Cui, and B. Rigg, "The development of the CIE 2000 colour-difference formula: CIEDE2000," Color Research & Application, vol. 26, issue. 5, Oct. 2001, pp. 340–350, doi: 10.1002/col.1049.

Copyright (c) IARIA, 2015. ISBN: 978-1-61208-412-1