Adaptive Graphics Interface

Marek Hołyński
Massachusetts Institute of Technology
Center for Advanced Visual Studies
40 Massachusetts Av., Boston, MA 02139

Robert W. Garneau
Suffolk University
Computer Engineering Department
41 Temple St., Boston, MA 02114

Abstract

The Adaptive Graphics Analyzer (AGA), presented in this paper, is an example of a graphics interface that learns about users' representational preferences and includes them into image generation algorithms. In order to incorporate learning and decision-making capabilities into graphics, the system extracts concrete and measurable statistics from a given image in terms of visual variables. The structure of the AGA is based on variable-value calculus that allows direct adoption of machine learning techniques. The AGA displays a series of images on a graphics terminal and requests the user to assign a preference value to each image. This evaluation and generation process repeats until the optimal sets of image variable values are found.

Keywords: Intelligent graphics system, user interface, knowledge acquisition.

I. Introduction

Graphical presentations should be adjusted to programming needs and on-the-spot applications in the same way an artist tailors his portfolio to the customer's requirements. The manner in which an architect perceives a picture of a building could be dramatically different from the way an engineer views the same structure. Image adjustments, based on properties of data, the user's objectives, design principles and perception rules, require implementation of techniques that help to organize and manipulate specific graphics knowledge.

In order to adjust the representation of data to preferences, needs and knowledge of people who are going to use it, we must equip graphics systems with an adaptive graphics interface that learns about these preferences and utilizes them as display rules. The Adaptive Graphics Analyzer (AGA), presented in this paper, is an example of such an interface, created for discovering the programmers' display rules and including them incrementally into image generation algorithms.

The simplest way to incorporate learning and decision-making capabilities into graphics systems is to borrow it from another field of computer science – artificial intelligence. There is no need to develop a separate "graphics intelligence", since many of the AI techniques can be integrated with graphics in a coherent manner and provide acceptable and possibly exceptional results.

II. Previous Work

Artificial intelligence and computer graphics have been close associates for some time. Early computer vision research, which used AI techniques extensively, addressed problems related to image analysis and synthesis. A bona fide linking of AI techniques and computer graphics took place in relational data base management systems. The data base, AI, graphics tie is explicitly mentioned by the builders of the automatic authoring system [FEINER, NAGY and VAN DAM 82]. In the concluding remarks of the paper, the authors discussed beginning efforts for the determination of layouts automatically, using domain-specific knowledge about the design rules for display formats. In [GARRETT and FOLEY 82] an augmented data base management system is described that removes the need for a procedural graphics generation module in interfaces to application programs. The manipulation of data representing images is done by the DBMS whenever there is a data update by the application or as a result of user queries.

A system that combines intelligent interfacing with image enhancement is the Steamer simulation program [STEVENS, ROBERTS and STEAD 83]. In this interactive program a combination of iconic images are used in conjunction with a knowledge base to produce configuration changing simulations of actual steam plant operation.

One question that continues to come up in any graphics interface is the experience level problem. How does one accommodate both the inexperienced user and the expert without bewildering the former and boring the latter? [JACOB 83] attempts to answer that question using high-level abstraction for describing the semantics of the user interface with two levels depending on the experience of the user. A somewhat similar technique is used by [TIBBERT and BERGERON 84] in XGRAPH, a system that combines rule-based knowledge base with a graphics program for specifying computer configurations interactively. Here, the authors use automaton type control (state changes) for knowledge base interaction to introduce a "guide" mode when necessary.

Knowledge-based object synthesis from primitive elements for both two and three- dimensional objects is first demonstrated by [FRIEDELL 84] and later used by [HOLYNSKI, GARNEAU and LEWIS 86] in two dimen-

sional layout for the determination of effective visual representation of images. The effectiveness of visual representation is further demonstrated by [MACKINLAY 86]. He uses a presentation tool, a rule-based system that automatically designs effective graphical presentations of relational information in the form of bar charts, scatter plots and connected graphs.

Three graphics systems that use artificial intelligence techniques extensively are the Peridot [MYERS and BUXTON 86], Predikt [OXMAN and GERO 87], and Graflisp [HOLYNSKI, GARDNER and OSTROVSKY 86]. Peridot automatically creates the code for user interfaces using pop-up and pull-down menus with elaborate feedback mechanisms and system guesses to save the designer time in making corrections to the knowledge base. Predikt uses a stand-alone rule-based expert system shell with forward and backward-chaining to provide mappings between semantics and graphical syntax. Graflisp describes relations among picture elements and concepts represented by these elements in the form of semantic networks and uses inference reasoning for image creation.

III. Determining Graphics Constraints with Machine Learning

In order for the system to learn about programming constraints for graphics, it must have the ability to extract concrete and measurable statistics from a given visual image. The system must be able to assign specific image variable values to each picture generated and realize which values or set of values most influence its appearance.

There has been a great deal of preliminary research done to select image variables. [LEWIS and HOLYNSKI 85] did early work in determining the appropriate image variables to construct a "filter" for selecting potentially acceptable images from the population of possible images. They explored three different methods: fractals, structural randomness of primitive elements (the technique used in this research), and grid systems to establish and test image variables. They found that visual form could in fact be quantified using variables such as complexity, regularity, color and order (symmetry).

In more recent experiments we generated the stimuli as abstract display matrices comprised of controlled combinations of thirty-six primitive elements. The stimuli illustrated the four previously used variables along with three additional variables: balance (image equilibrium), variety (number of different primitive elements that make up a single image) and busyness (amount of display image that contains primitive elements). Two hundred subjects from three different student groups evaluated the stimuli. Standard regression analysis was used to discover which variables were appropriate predictors for user preference. Using an interval scale, we found that complexity, regularity, symmetry and busyness had a positive significant effect of approximately the same magnitude on user preference [HOLYNSKI, GARNEAU and LEWIS 86].

These findings were consistent with earlier findings in terms of relationship between regularity and complexity with preference. Although useful, conventional statistical packages do not immediately show how one variable interacts or depends on another. In order to more carefully examine the interrelationship among all the variables, we used a rule acquisition program that was developed at the University of Illinois. It allowed us to disclose structural descriptions of object classes, i.e., descriptions that involve not only object attributes but also relations among object components and subcomponents. The rule acquisition program more specifically supported the statistical evaluation and generated rules in the form accepted by display algorithms.

These preliminary results led to the adoption of more image variables, more levels within each variable and the inclusion of a knowledge-repair mechanism that resulted in the construction of the Adaptive Graphics Analyzer (AGA). Using these preliminary results a set of default values have been selected to initialize the AGA. This default set gives the analyzer a broad base of variable values so that the system will create initial images that appeal to a general audience and will help the user to quickly select his preference set. AGA interacts with the user and his preference selections to continually adjust its image variable values and generate an image or set of images that are most suitable to the particular user's preferences, specifications and requirements.

IV. Formalism for Preference Extraction

The structure of the Adaptive Graphics Analyzer, the visual form variables that it uses, and their relation to image generation and evaluation can be revealed in a formal way using an extension of variable-valued logic calculus* [MICHALSKI and CHILAUSKY 80].

In the AGA image stimuli are generated in sequences of ten to fifteen images called an *Image Group*. The first sequence of image stimuli, called the *Default Image Group* G_0, is generated based on initial image attribute information stored in a file. After the user assigns a preference to each image stimuli in this group, a second sequence of image stimuli called the *User Selected Image Group* G_1 can be generated which allow the user to construct his own preferred image stimuli. New *User Derived Image Groups* G_i are created using these two initial groups of image stimuli as a foundation.

The i^{th} Group (or i^{th} sequence of image stimuli) can be represented as an indexed list:

$$G_i \equiv < p_j, p_{j+1}, ..., p_k, ..., p_l > \qquad k \in [j, l]$$

where each image stimuli p_k is described by a *Set of Visual Form Variables* V_k:

$$p_k \equiv F(V_k)$$

and

$$V_k \equiv < f_1, f_2, ..., f_m, ..., f_n > \qquad m \in [1, n]$$

* The formal notation adopted here is based on Michalski's variable-valued logic calculus system VL2.

which can be considered to be *Graphic Image Generation Functions* mapping image stimuli into *Visual Form Variable Values* (and vice versa).

In the example presented in this paper, the AGA uses seven groups of image stimuli:

$$G_0 \equiv < p_1, ..., p_{12} > \quad G_1 \equiv < p_{13}, ..., p_{36} >$$
$$G_2 \equiv < p_{37}, ..., p_{51} > \quad G_3 \equiv < p_{52}, ..., p_{61} >$$
$$G_4 \equiv < p_{62}, ..., p_{76} > \quad G_5 \equiv < p_{77}, ..., p_{80} >$$
$$G_6 \equiv < p_{81}, p_{82} >$$

with the following visual form variable set for all k∈[1, 82]:

$$V_{aga} \equiv < f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9 >$$
$$\equiv < Balance, GridSize, Busyness, Complexity, Regularity, ColorVariety, ShapeVariety, Symmetry, Color >$$

Visual Form Variable f_m of image stimuli p_k with value α is called a *selector* and represents a relational statement that selects the value α of the Visual Form Variable f_m for image stimuli k.

$$[f_m(p_k) = \alpha]$$

For example, if f_5 is *Regularity*, and α is low, then the selector $[f_5(p_8) = low]$ states that the *Regularity* of the image stimuli number 8 in the image generation group G_0 is low.

Each Visual Form Variable is only permitted to have values from a finite value set called the *Domain* of that Visual Form Variable $D(f_m)$. The Domain of the Visual Form Variable is dependent on the constraints of the image generation process. Domains of the Visual Form Variables used by the AGA are as follows:

$$D(f_1), D(f_4), D(f_5), D(f_8) \equiv \{low, medium, high\}$$
$$D(f_2) \equiv \{small, medium, large\}$$
$$D(f_3), D(f_6), D(f_7) \equiv \{low, medlow, medhigh, high\}$$
$$D(f_9) \equiv \{c_1, c_2, c_3, ...c_{30}\}^{**}$$

A complete description of a single image stimuli p_k, called a *Form Variable Value Set* I_k, is an expression which assigns values to all the Visual Form Variables of p_k and is written as a conjunction of selectors:

$$I_k \equiv [f_1(p_k) = \alpha] \wedge [f_2(p_k) = \beta] \wedge ... \wedge [f_n(p_k) = \omega]$$

For example, Default Variable Value Set 8 used by the AGA to generate image number 8 was defined as,

$$I_8 \equiv [f_1(p_8) = med] \wedge [f_2(p_8) = small] \wedge [f_3(p_8) = medh] \wedge$$
$$[f_4(p_8) = med] \wedge [f_5(p_8) = med] \wedge [f_6(p_8) = medh] \wedge$$
$$[f_7(p_8) = medl] \wedge [f_8(p_8) = low] \wedge [f_9(p_8) = (c_4, c_5, c_6)]$$

which can be simplified as a vector of Visual Form Variable values:

$$I_8 = (med, small, medhigh, med, med, medhigh, medlow, low, red, magenta, yellow)$$

** Color, depending on the Color Variety variable can be made up of one to four of the 30 available colors.

Each image stimuli description I_k can be viewed as a single point in the image space **I** which can be constructed as a cross product of Domains:

$$\mathbf{I} = D(f_1) * D(f_2) * ... * D(f_m) * ... * D(f_n)$$

where $D(f_m)$ is the domain of Visual Form Variable f_m. This image space contains all possible images. The image space of the AGA is the set of all possible Form Variable Value Sets each of which representing one image stimuli.

In order to facilitate the discovery of new, more preferred images, the system must perform *Transformation Operations* on the initial default image group and the user selected image group creating *User Derived Image Groups*. These transformations can be viewed as a mapping T from one group of image stimuli G', defined by Form Variable Value Sets $I'(I' \subset \mathbf{I})$, described by default visual form variables V_d, to another group of user derived image stimuli G'', defined by Form Variable Value Sets $I''(I'' \subset \mathbf{I})$, and described by default visual form variables V_d:

$$T_{\eta,\kappa,\lambda} < G', I', V_d > \Rightarrow < G'', I'', V_d >$$

where $T_\eta, T_\kappa, T_\lambda$ are different transformation functions.

In our example, the AGA uses four transformation functions, the *Default Image Generation Module* (T_d), the *Image Selection Module* (T_s), the *Averaging Evaluation Module* (T_a), and the *Value Set Evaluation Module* (T_v) which are described by [GARNEAU 88]. The operation of the AGA can be described as a series of mappings from the Default Image Group G_0, defined by default I_d, and the User Selected Image Group G_1, defined by user selected I_s, to the final User Derived Image Group G_6, with I_f, which should contain images that are attuned to individual preferences.

$$T_d < G_0, I_d, V > \Rightarrow < G_2, I_0, V >$$
$$T_s < G_1, I_s, V > \Rightarrow < G_2, I_1, V >$$
$$T_v < G_2, \{I_0, I_1\}, V > \Rightarrow < G_3, I_2, V >$$
$$T_a < G_3, I_2, V > \Rightarrow < G_4, I_3, V >$$
$$T_v < G_4, \{I_2, I_3\}, V > \Rightarrow < G_5, I_4, V >$$
$$T_v < G_5, \{I_2, I_3, I_4\}, V > \Rightarrow < G_6, I_5, V >$$
$$T_v < G_6, I_5, V > \Rightarrow < I_f, V >$$

V. Evaluation

The AGA displays a series of images on a graphics terminal and allows the user to assign a preference value to each image. Each image is a composite of up to four different shapes chosen from a reservoir of thirty-six primitive elements. Each primitive element has a specific value for regularity and complexity and was constructed so that the user will not associate any particular meaning to the composite image. By purposely using non-representational abstract patterns rather than more meaningful images we were able to determine relevant form variables and a methodology for measuring preference. The primitive shapes are stored as data points in files which are read in at system start-up and can be changed at any time to any desired shape.

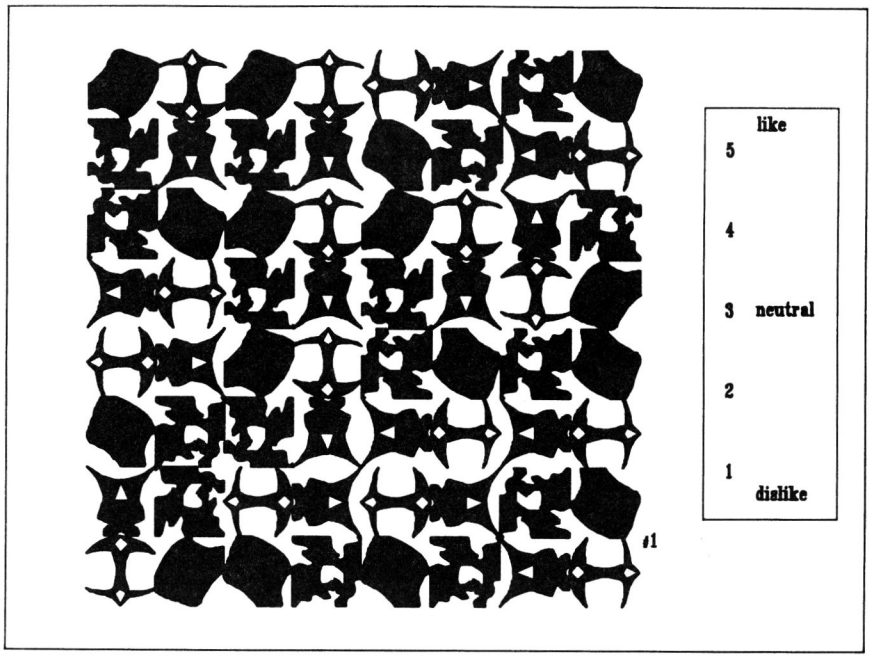

Figure 1: Image with Preference Menu

The system generates a total of 82 images in 7 image transformation groups. There were fifty-five research participants generating and evaluating over four thousand image stimuli. The appearance of each image was dependent on specific Visual Form Variable Values which are dynamically generated as images are evaluated. The first thirty-six images that make up groups G_0 and G_1 are generated based on a combination of Default Visual Form Variable Values [HOLYNSKI, GARNEAU, LEWIS 86] and user image variable selections.

The initial mapping functions (T_d, T_s) create image stimuli from the initial Visual Form Variable Values in G_0 and G_1. After image evaluation, the associated Visual Form Variables are dynamically mapped into a new set of Visual Form Variable Values which are used to generate the first group of user derived images G_2. These user derived images are displayed with a preference menu for immediate evaluation (Figure 1). Subsequent user derived image groups (G_3 through G_6) are evaluated and generated based on additional mapping functions (T_a, T_v).

Figure 2 shows the average preference assigned to images in each transformation group. The average preference increases from 2.987 in group two to 3.852 in group six. Extensive statistical analysis of these results is provided in [GARNEAU 88]. The default images (G_0) and the user selected images (G_1) are not listed. These two groups utilize a comparison technique rather than a menu to extract user preference and their inclusion would be misleading.

The averaging mapping function T_a used to generate group three (G_3) generated four random images per person which are shown in the graph as the white bar. The random images were produced by AGA to insure that the users were not making selections for the wrong reasons and also to introduce more image variety. Subjects had no idea that some of the images were produced with random form variable values. The average preference of the random images generated in group three is 2.232. From a purely statistically viewpoint, it could be expected that the random images would have the average preference value of 3, yet when they are compared to user derived images they are typically assigned lower than average values. This phenomenon was witness throughout the empirical testing. Random images when displayed with user derived images often evaluated to be sub-average in mean preference.

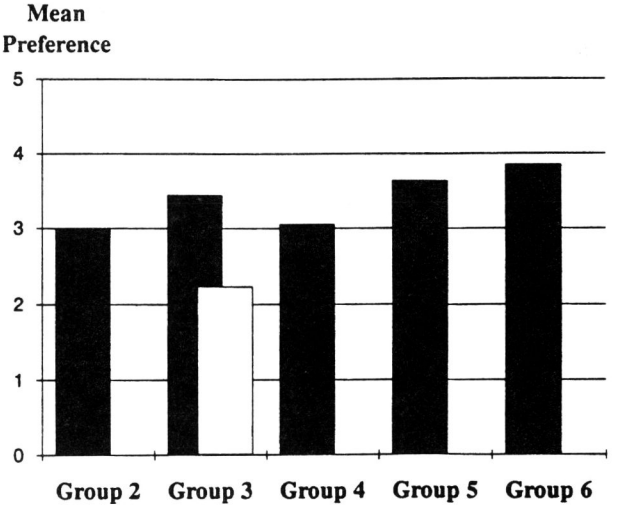

Figure 2: Mean Preferences Assigned to Images

VI. Further Research

We have constructed an experimental system which attempts to eliminate one of the most difficult tasks in computer graphics generation systems: the determination of application constraints, user's specific requirements, and preferences. Too many of todays graphic programmers are attuned to their own priorities and methods, and remain out of touch with the exact needs and requirements of the user of their products. Rather than have the programmer determine image qualities, there will one day be graphics systems that extract the appropriate presentation variables directly from the users. Once these variables have been found, they could be used in almost all types of computer graphics applications from computer aided design to simple presentation graphics in order to improve and select both functional and pleasing display techniques. Eventually it will lead, not only to improved user oriented computer graphics, but to a system that makes intelligent user friendly decisions in the most optimal way. It will display a given image or set of data based on the needs of an individual or group of individuals. It may be able to select the most popular way to display particular images or even animated images for specified products and specified populations of users.

There are a number of directions in which this research can expand and progress to. One can attempt to convert the primitive elements to three dimensional shapes and adjust the AGA to make the appropriate transformations in generating images. By using three dimensional images we will be able to investigate interesting visual concepts such as depth, viewpoint, lighting, texture and shadows and how they affect user preference. We will then be in a position to relate these ideas to presentation graphics, solid modeling for computer imagery, computer aided design and other scientific and engineering applications.

References

1. FEINER, STEVEN, NAGY, SANDOR and VAN DAM, ANDRIES - 1982, January - "An Experimental System for Creating and Presenting Interactive Graphical Documents", ACM Transactions on Graphics, Volume 1, Number 1, January 1982, pp. 59-77.

2. FRIEDELL, MARK - 1984, July - "Automatic Synthesis of Graphical Object Descriptions", Computer Graphics July 1984, Volume 18, Number 3, pages 53-62.

3. GARRETT, MICHAEL T. and FOLEY, JAMES D. - 1982, April - "Graphics Programming Using a Database System with Dependency Declarations", ACM Transactions on Graphics, Volume 1, Number 2, April 1982, pp. 109-128.

4. GARNEAU, ROBERT W. - 1988, May - "Interactive Adaptive Analysis of Graphic Displays", Ph.D. Thesis, Boston University, Dissertation Abstracts International, University Microfilms Inc., Publ. # 88-20,146.

5. HOLYNSKI, MAREK, GARDNER, BRIAN R. and OSTROVSKY RAFAIL - 1986, May - "Knowledge-Based Generation of Computer Images", Computer Graphics '86 Proceedings, National Computer Graphics Association, Anaheim, May 1986, pp. 624-633.

6. HOLYNSKI, MAREK, GARNEAU, ROBERT W. and LEWIS, ELAINE - 1986, August 25 - "Adaptive Graphics Interface for Selection of an Effective Visual Representation", Proceedings of Eurographics '86: The computer Interface pp. 195-206.

7. JACOB, ROBERT J.K. - 1983 April - "Using Formal Specifications in the Design of a Human-Computer Interface", Communications of the ACM, Working Toward Successful Human-Computer Interface, Volume 26, Number 4.

8. LEWIS, ELAINE and HOLYNSKI, MAREK - 1985, October - "Effective Visual Representation of Computer Generated Images", IEEE Proceedings, 5th Symposium on Small Computers in the Arts, IEEE Press, pp. 9-12.

9. MACKINLAY, JOCK - 1986, April - "Automating the Design of Graphical Presentations of Relational Information", ACM Transactions on Graphics, Volume 5, Number 2, pp. 110-141.

10. MICHALSKI, RICHARD S. and CHILAUSKY, RICHARD L., - 1980 - "Learning by Being Told and Learning from Examples: An Experimental Comparison of the Two Methods of Knowledge Acquisition on the Context of Developing an Expert System for Soybean Disease Diagnosis", International Journal of Policy Analysis and Information Systems, Vol.4, No.2.

11. MYERS, BRAD A. and BUXTON, WILLIAM - 1986, August 18 - "Creating Highly-Interactive and Graphical User Interfaces by Demonstration", Siggraph 86 Conference Proceedings, Dallas, Texas, Volume 20, Number 4.

12. OXMAN, RIVKA and GERO, JOHN S. - 1987, February - "Using an Expert System for Design Diagnosis and Design Synthesis", Expert Systems, The Computer Society of the IEEE, Volume 4, Number 1, pp. 4-15.

13. STEVENS, ALBERT, ROBERTS, BRUCE and STEAD, LARRY - 1983, March - "The Use of a Sophisticated Graphics Interface in Computer-Aided Instruction", Computer Society of the IEEE, Computer Graphics and Applications, pp. 25-30.

14. TIBBERT, LEE and BERGERON, R. DANIEL - 1984, January - "Graphics Programming for Knowledge-Guided Interaction", University of New Hampshire, Computer Science Tech. Report CS 84- 18.