Applied Intelligence 16, 129–138, 2002

Towards Creative Evolutionary Systems with Interactive Genetic Algorithm

SUNG-BAE CHO
Department of Computer Science, Yonsei University, 134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea
sbcho@csai.yonsei.ac.kr

Abstract. Evolutionary computation has shown a great potential to work out several real-world problems in the point of optimization, but it is still quite far from realizing a system of matching the human performance. Especially, in creative applications such as architecture, art, music, and design, it is difficult to evaluate the fitness because the measure depends mainly on the human mind. To overcome this shortcoming, this paper presents a novel technique, called interactive genetic algorithm (IGA), which performs optimization with human evaluation and the user can obtain what he has in mind through repeated interaction with. To show the usefulness of the IGA to develop effective human-oriented evolutionary systems, we have applied it to the problems of fashion design and emotion-based image retrieval. Experiments with several human subjects indicate that the IGA approach is promising to develop creative evolutionary systems.

Keywords: creative evolutionary systems, interactive genetic algorithm, human-computer interface, fashion design, emotion-based retrieval

1. Introduction

Evolution is a remarkable problem-solving machine [1]. First proposed by John Holland in 1975, genetic algorithm (GA) as one of the computational implementations is an attractive class of computational models that mimic natural evolution to solve problems in a wide variety of domains.

The basis of GA is that a population of problem solutions is maintained in the form of chromosomes, which are strings encoding problem solutions [2]. Strings can be binary or have many possible alternatives at each position. The strings are converted into problem solutions, which are then evaluated according to an objective scoring function. Often it is not possible to exhaustively test all aspects of a solution, and noise may be present on the objective function, so the assigned fitness is an estimate of the true fitness of a chromosome. It is important that this is a good estimate, otherwise the selective pressure that favors truly high scoring chromosomes can be lost in the noise caused by poor fitness estimates.

Following fitness evaluation, a new population of chromosomes is generated by applying a set of genetic operators to the original population. These are basically random copying and altering of individuals from the original population with the probability of copying of any individual from one generation to the next being proportional to its fitness.

By the way, most of the conventional applications of GA lack of the capability to utilize human intuition and emotion appropriately in creative applications such as architecture, art, music, and design. There is no clear measure to give the evaluation of fitness other than the one in the human mind. Interactive GA (IGA) is a technique that performs optimization with the human evaluation. A human can obtain what he has in mind through repeated interaction with the method, when the fitness function cannot be explicitly defined. This allows to develop effective human-oriented evolutionary systems, since this obtains from human the fitness value for the problem at hand, and produces better designs or images for the next generation.

The rest of this paper is organized as follows. Section 2 introduces the basic idea of the IGA and some possible applications briefly. In order to justify the usefulness of the IGA, typical examples of applying to fashion design and emotion-based image retrieval are presented in Sections 3 and 4, respectively. Each section includes a thorough experiment with subjective tests to confirm the potential of the IGA for creative evolutionary systems.

2. Interactive Genetic Algorithm

GA applies some of natural evolution mechanisms like crossover, mutation, and survival of the fittest to optimization and machine learning. GA provides very efficient search method working on population, and has been applied to many problems of optimization and classification [1].

The procedure of a simple genetic algorithm is as follows.

$t = 0$;
InitializePopulation P(t);
Evaluate P(t);
while not done **do**
  $t = t + 1$;
  P' = SelectParents P(t);
  Recombine P'(t);
  Mutate P'(t);
  Evaluate P'(t);
  P = Survive P, P'(t);
end_while

Each chromosome is encoded by a bit string, and crossover operation swaps some part of the bit string of parents. It emulates crossover of genes in the real world in that descendants inherit characteristics from both parents. Mutation operation inverts some bits in the bit string at very low rate. In real world we can see that some mutants come out rarely. Each individual in the population gets higher fitness as it goes generation by generation.

IGA is the same as GA except the way of assigning the fitness value. In IGA user gives fitness to each individual instead of fitness function. In this way IGA can 'interact' with the user, and also can perceive user's emotion or preference in the course of evolution. For this reason IGA can be used to solve problems that cannot be easily solved by GA, such as design and art [3–5].

More than 100 publications have been made with the application of the IGA to the problems that require creative solutions, and they can be classified into three categories: artistic, engineering and edutainment applications [6]. For a good and quick overview of the classic examples, refer to the handbook of evolutionary computation, Section C2.9 by Banzhaf [7]. Table 1 summarizes the possible applications of the IGA. Here is an example of design that shows the advantage of using IGA instead of GA. We can apply GA for design by initializing the population of individuals encoded from design characteristics, setting and evolving the fitness as 'how good the design is.' However, there is no standard of 'goodness of design,' and it is almost impossible to organize the fitness function. IGA might be a solution in this case. IGA can reflect personal preference, because it perceives the fitness directly from human instead of computing some function [8].

Table 1. The summary of IGA applications.

Category	Examples
Artistic applications	Computer graphics, CG lighting, design, music
Engineering applications	Speech and image processing, hearing aid, virtual reality, database retrieval, data mining, control and robotics
Edutainment applications	Composition support, robot control, games

3. Fashion Design

Though the meaning of design has changed by time, the works that designers do has not changed much when they design clothes. They start with a sketch and flesh it out into an illustration. With clippings from magazines, journals and photographs of similar or rival designs from fashion shows, they make these concepts into some sample pieces. These samples are tested for quality, feel and aesthetics very rigorously. Recently, the computer has begun to aid these works.

There are many systems that aid fashion design works. AutoCAD from Autodesk, the famous computer-aided design system software, can be used well for fashion design aid system. The design work can be drastically reduced by using apparel design software or plug-ins, such as ApparelCAD, that is designed to work along with AutoCAD. Creative Designer System from Gerber is another solution for fashion design. It is an elaborate graphic workstation with software and some hardware, which can get image and color from sketches, photos, fabrics, or real materials, and edit them by selecting menus. Other general software, such as Photoshop and Illustrator from Adobe, can be used as well as other dedicated fashion design system. Virtual Reality (VR), which is achived by some computer system, can also reduce time and cost to make and test samples, and throw away one below some level. Laser scanner [9] or digital camera takes one's measurement. The captured body models can be transformed to virtual mannequins, and computers can make them walking just as real fashion models do. User can get finely fitted clothes both as 3D models and as 2D pattern block representation [10]. These fashion design aid systems work well, but they are usually for professionals only, and it is hard for non-professionals to use.

Some design-aid systems have been developed using Evolutionary Computations (EC). Fashion design aid systems using EC can be used by non-professional persons, because it evolves individuals according to interaction with the user. Nakanishi developed a fashion design aid system using genetic programming [5]. He encoded a number of different lines from a dress into chromosome. The system evolves each dress design according to the user's selection. But most of its productions were somewhat impractical, because encoded individuals did not contain realistic knowledge on the domain of fashion.

3.1. *Method*

Figure 1 shows the overview of the proposed fashion design aid system based on the IGA. There is a database of partial design elements, which are stored in 3D models. The system selects the models of each part and combines them into a number of individual designs. The population is displayed on screen and user gives a fitness value to each design. Then, the system reproduces the population proportional to the fitness value of each design, and applies crossover and mutation to make the next generation. The results are displayed again in the screen with 3D graphics. Iteration of these processes can produce the population of higher fitness value, namely better designs.

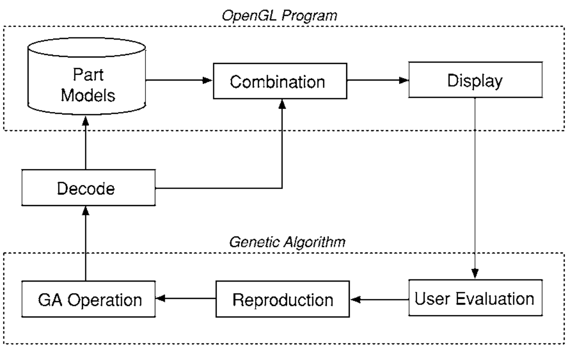

Figure 1. Fashion design system using IGA.

As mentioned earlier, previous design aid systems using evolutionary computation produce impractical designs because they do not consider domain-specific knowledge. To solve this problem, we have encoded the detail model based on the knowledge of fashion design. First, we have reclassified general detail factors into three parts: neck and body, arm and sleeve, skirt and waistline. Next, we have encoded them with extra 3 bits for each, which chooses the color of each part. A design is made from combining them, and with IGA some combination that produces the design preferred by user is found out, resulting in more realistic and reasonable design. Encoded detail parts are described as follows.

- Arm and sleeve part contains sleeve and cuffs detail. There are 12 models including sleeveless design, and encoded into 4 bits.
- Neck and body part includes neckline, collar, and body shape. We have collected and encoded 34 models into 6 bits.
- Skirt and waistline part includes waistline and skirt below it. We have collected and encoded 9 models into 4 bits.

Each part can take their own color out of 8 colors. Therefore, additional 9 bits are needed to complete chromosome encoding. Figure 2 describes how a chromosome is completely encoded. We can calculate the size of search space easily by computing all the combinations of design and their colors. The size of search space is 1,880,064 resulted from $34 \times 8 \times 12 \times 8 \times 9 \times 8$. The system searches the best designs out of 1,880,064 candidates according to user's feedback on preference and emotion.

To implement the system, each component of design parts should be created in 3D models. We have

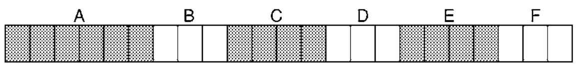

Figure 2. Chromosome encoding: Neck and body style (A), color of neck and body (B), arm and sleeve style (C), color of arm and sleeve (D), skirt and waistline style (E) and color of skirt and waistline (F).

used 3D Studio MAX R2.5 for this task. Each model is converted into OpenGL lists [11]. There are two alternatives we can make to show 3D models: VRML and OpenGL. The former is much easier to show 3D models, but it is too slow and has low modeling quality [12]. OpenGL is faster than VRML but it demands far more time and efforts to implement the system. Considering these pros and cons, we have used OpenGL library from SGI with GLUT library developed by Mark Kilgard and ported to Win32 by Nate Robins [13].

Converted models are inserted into a C program, written in Visual Studio 6.0 of Microsoft. The system can show combined 3D models through OpenGL operation, according to decoding from individual genotype. Figure 3 gives an example of combining models from genotype bit string. Figure 4 shows the user interface of

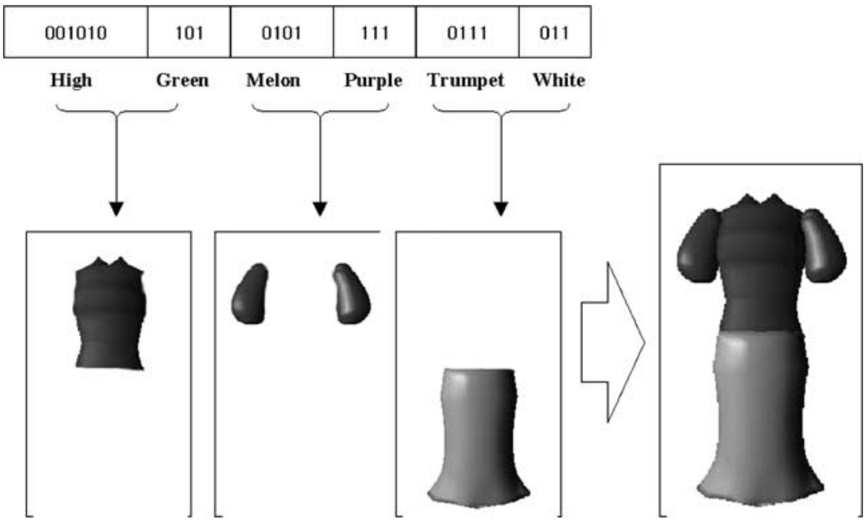

Figure 3. Decoding from an example genotype.

Figure 4. The user interface of fashion design system.

our system. The system shows current population composed of 8 individual models in one screen. There is a slider bar for each individual design to obtain user's feedback or preference. The rightmost part of screen shows the current status of evolution, and provides the controls 'to generate next population' and 'to restore previous population.' One can find out their favorite design from the large search space by interacting with the system.

3.2. Experimental Results

The system runs on a Pentium PC. The population is composed of 8 individuals. We have used one-point crossover of 0.5 and mutation of 0.05. As a strategy of evolution, we have preserved one elitist individual in each generation for the next generation. To evaluate the performance of this system, we have carried out convergence and subjective tests.

3.2.1. Convergence Test. It is difficult to show the convergence of IGA with quantitative analysis because it is operated based on human evaluation, very different from standard GA. To show the convergence of the method as an experimental result, we have requested 10 subjects to find cool-looking design and splendid design using the system. Each searching is limited to 10 generations. Figure 5 shows the changes of fitness on average and the best, when subjects are searching cool-looking design. Though the search space is extremely large, the experiments show the steady improvement of the result over generations. Figure 6 shows some images of converged solutions for cool-looking design.

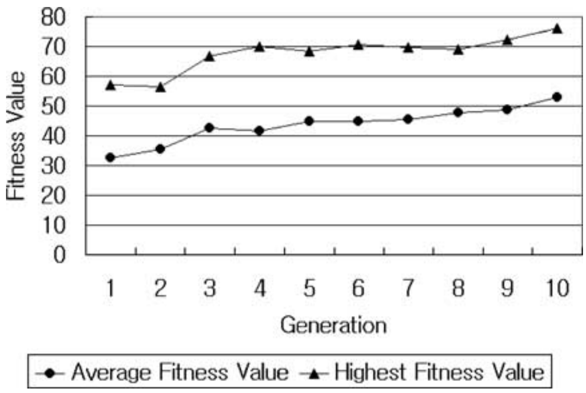

Figure 5. Fitness changes on searching for cool-looking design.

Figure 6. Some typical images of converged solutions for cool-looking design.

3.2.2. Subjective Test. This test shows how much the user is satisfied with the system performance. We have used the pair-test of Sheffé [14]. It prevents users from evaluating searched design too subjectively by requesting them to compare each design with some relatively objective ones, pair to pair.

First, we have randomly selected 500 sample designs from the entire search space, and requested 3 subjects to evaluate the sample designs with 2 categories, cool and splendid, into 5 degrees (from −2 to 2). From their average scores, 10 most cool-looking designs and another 10 most splendid ones are selected as standards of evaluation. Next, we have requested 10 subjects to find cool-looking designs and splendid designs using the system. Each searching is limited to 10 generations. After 10 generations, subjects must select the best one from the last population, and compare it with the standards we made before, pair to pair. This score has 7 degrees (from −3 to 3). Finally, the result is analyzed statistically. Figure 7 shows the degree of user's satisfaction at 95% and 99% of reliability. Subjects have given 2.17 to the design searched for cool-looking design and 1.74 for splendid design on average. It can be seen that searching for cool-looking design has gained higher satisfaction score, and shows narrower confidence interval.

We can assert that subjects are significantly satisfied with the system on searching for cool-looking designs and splendid designs. Therefore, applying IGA to the fashion design aid system for non-professionals is very promising.

4. Image Retrieval

The main stream of previous approaches to image retrieval is the keyword-based method that indexes image database with keywords manually and queries are

Figure 7. Mean of satisfaction degrees.

specified using these keywords. This method allows effective search with small image collections, but it costs much time and labor to construct index when it applies to large database, and the efficiency of search decreases when the viewpoint of index constructor is different from that of actual user. The important problem is that it is inherently difficult to describe some visual aspects.

The content-based image retrieval is proposed to solve these problems. In this approach image contents are modeled as a set of attributes extracted from the image, and queries are specified using these attributes. Generally, the procedure of feature extraction is automatic or semi-automatic. This method has an advantage of reducing time and labor to be needed to construct image database, but it is difficult to extract exact feature information. Features of image to be used for searching might be color, texture, shape, and relationship of objects within the image.

For the content-based image retrieval, for example, several working systems have already been developed: QBIC system of IBM [15], QVE of Hirata and Kato [16], Chabot of UC Berkeley [17], Photobook of MIT, and Image Surfer of Interpix software. It can be applied to digital library, medical management system, home shopping, and so on.

4.1. Method

The whole system is constructed as shown in Fig. 8. In the preprocessing step, at first, the wavelet transform is performed for every image in the database and stored are the overall average color and the indices and signs of the m magnitude wavelet coefficients in a search table. The system displays twelve images, obtains the fitness values of the images from human, and selects candidates based on the fitness. Genetic operation, vertical or horizontal crossover, is applied to the selected candidates. To find the next twelve images, the stored image information is evaluated by each criterion. Twelve images of the higher magnitude value are provided as a result of the search.

For this problem, a chromosome is represented by an array that consists of the indices of wavelet coefficients. Figure 9 shows the procedure that is used to construct a chromosome. The wavelet coefficients are obtained by decomposing an image using wavelet transform. The $r \times r$ matrix, T, obtained by Haar wavelet transform has the average color of the image in entry $T[0, 0]$ and the wavelet coefficients in the other entries of T. We can reconstruct the original image without loss using this information, but because we do not have to maintain all the information to search, we just extract the largest 50 coefficients in RGB channels and use them for constructing a chromosome in 3×50 array.

Jacobs' work shows that storing the 40–50 largest-magnitude coefficients in each color works best and truncating the coefficients appears to improve the discrimination power of the metric [18]. Therefore, we have stored only the sign information of coefficient values into the chromosome.

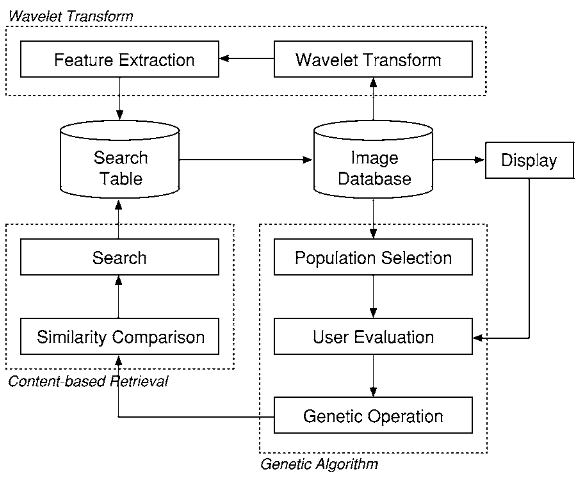

Figure 8. Image retrieval system using IGA.

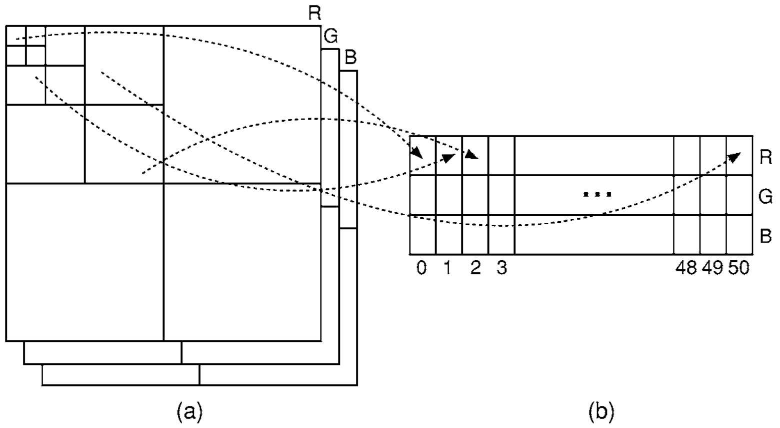

Figure 9. The procedure that is used to construct a chromosome. (a) Wavelet transformed 256 × 256 image. (b) The chromosome constructed using the wavelet coefficients of (a).

This system obtains the evaluation values from human. It provides a solution that reflects user preference. The size of population is 12. The strategy of selection is governed by expected frequency of each individual. We use one point crossover that proceeds in selecting a point and swapping a part of chromosome on the basis of it. In that case, two individuals having the highest fitness are selected and mated at random. In detail, horizontal and vertical crossovers are used. Those allow to exchange the color and shape features respectively. Mutation is not adopted explicitly, but the search process explained below works as a sort of mutation.

In this sense, the image retrieval is to find new twelve images that have the highest fitness after applying crossover operation. At present, we consider an image of high fitness as target image and an image in database as candidate image, and compare the similarity between them to determine a new image to be displayed. The similarity between potential target image and candidate image is calculated by the following equation.

$$\begin{aligned}\|Q, T\| = {} & w_{0,0}|Q[0,0] - T[0,0]| \\ & + \sum_{i,j} w_{i,j} - \sum \Big| Q[i,j] \\ & \geq \sum w_{i,j} \Big| Q[i,j] | Q[i,j] - T[i,j]| \qquad (1)\end{aligned}$$

$Q[0, 0]$ and $T[0, 0]$ mean overall average intensities of single color channels, and $Q[i, j]$ and $T[i, j]$ represent those color channels of wavelet decomposition of the query and target images. Here, $w_{0,0}$ and $w_{i,j}$ are determined empirically.

4.2. *Experimental Results*

The system is programmed in Microsoft Visual C++ on a Pentium PC. The size of image database is 2000. In order to search more efficiently and quickly, a searching table is constructed by a batch job over the 256 × 256 JPEG images. It maintains signs and indices of wavelet coefficients. The crossover rate is 0.6. Initial population consists of twelve images selected randomly. As user gives images the fitness values based on how similar to what he wants, the system presents new images in the next generation using genetic algorithm. Figure 10 shows the user interface of the system.

This procedure is repeated until the user obtains the image that is most similar to what he has in mind. In case that the results of the next generation are not satisfactory, the system allows user to go back to the previous generation. Moreover, the user can increase or decrease the importance of the color to search image.

4.2.1. Usefulness of Wavelet Transform. At first, we test whether wavelet coefficients are appropriate for genetic representation or not. To do this, we classify the images into two classes: gloomy and the other. For each image, we calculate the similarity between images contained in the same class (within-class similarity) and the similarity between images contained in different classes (inter-class similarity), respectively. The similarity is evaluated through the same method that is used to compare the target image with the candidate image in the system. The higher the value is, the closer the images are. High similarity would mean that the two images are in near points on the wavelet space.

Figure 10. The user interface of image retrieval system.

Most of the gloomy images produce higher similarity value in the same class. The averages of within-class and inter-class similarities are 1154.692 and 1114.939, respectively. Table 2 shows that the mean of within-class similarities is larger than that of inter-class. In order to determine whether this difference is statistically significant or not, we have performed a paired t-test.

This is useful for testing whether the difference of mean values is significant or not. We hypothesize that the mean values of within-class and inter-class similarities are not different, and the t-test value is calculated. The result of t-test is 5.508 and our hypothesis is rejected at a 99.995%. This means that the images in the same class are closer on the wavelet space and we can assert that wavelet coefficients represent the mood of images well.

Table 2. Within-class and inter-class similarities for gloomy images.

	Within-class	Inter-class	Difference
Mean	1154.692	1114.939	39.752
Variance	27368.739	24196.524	3229.317
SD	165.434	155.552	56.827
t-test			5.508

4.2.2. Convergence Test. It is pretty hard to prove the convergence in the case of interactive genetic algorithm. Toward this goal we have attempted to show the change of fitness according to the generations and compare the results obtained. Figure 11 shows the average and best fitnesses for 10 generations in the case of searching the image of gloomy impression. We can see that the fitness is effectively increased by adopting user's evaluation, although the search space of chromosome encoded using wavelet coefficients is very large. Figure 12 shows some images of converged solutions for gloomy impression.

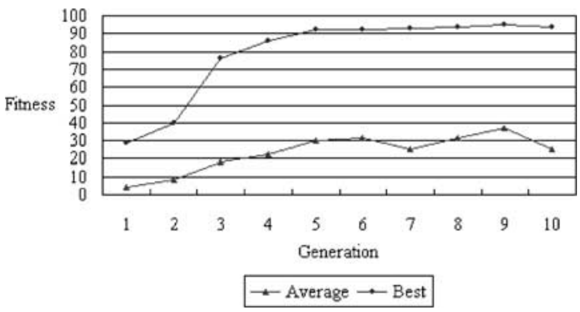

Figure 11. The average fitness and the best fitness in the case of searching the image of gloomy impression.

Figure 12. Some typical images of converged solutions for gloomy impression.

4.2.3. Subjective Test. A psychological test by Sheffé's method is conducted to see user's satisfaction. The Sheffé's method of paired comparison is a subjective test that requests subjects to give a score to the difference between a given pair of images [14]. It makes a psychological distance measure from the score. The significance of difference among evaluated images is tested by the analysis of variance. In order to test, we give 3 subjects two motifs that are gloomy and cheerful impressions respectively and they are requested to select eight images manually according to each motif in 500 images from the database. Next, we ask 10 subjects to search images with the same motifs using the proposed system. Here, the number of images to be compared is 9, among which 8 images are selected manually and one image searched using the system. Thus, the number of paired images used for this subjective test is ${}_9C_2 = 36$. We ask 10 subjects to give 7 steps score to the difference between a pair of images with considering the given motif. The evaluation of subjects is statistically tested, and the result indicates that the users are satisfied with the images searched by the proposed interface. Figure 13 shows the result of statistical test. The x-axis means a degree of satisfaction, and gray and black bars in this figure are confidence intervals of 95% and 99%, respectively.

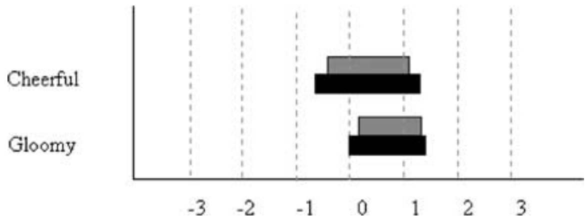

Figure 13. The intervals of satisfaction of images searched according to cheerful and gloomy impressions. Gray and black bars in this figure represent confidence intervals of 95% and 99%, respectively.

5. Concluding Remarks

This paper has presented an approach that implements creative evolutionary systems with human preference and emotion using interactive genetic algorithm. Several experiments show that our approach allows to design and search digital media not only explicitly expressed image, but also abstract image such as "cheerful impression," "gloomy impression," and so on. However, one of the biggest problems might be in human fatigue that hinders from applying the IGA to the real-world problems. We have to devise effective methodologies to reduce the human burden with better interface and acceleration methods of evolution.

Acknowledgments

This work was supported by Korea Research Foundation Grant (KRF-2000-005-C 00012). The author would like to thank Ms. Joo-Young Lee and Mr. Hee-Su Kim for their helps to implement the systems and perform some part of the experiments.

References

1. D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley: Reading, MA, 1989.
2. M. Srinivas and L.M. Patnaik, "Genetic algorithms: A survey," *IEEE Computer*, vol. 27, no. 6, pp. 17–26, 1994.
3. J.A. Biles, "GenJam: A genetic algorithm for generating jazz solos," in *Proc. Int. Computer Music Conf.*, Aarhus, Denmark, 1994, pp. 131–137.
4. C. Caldwell and V.S. Johnston, "Tracking a criminal suspect through face-space with a genetic algorithm," in *Proc. 4th Int. Conf. Genetic Algorithm*, 1991, Morgan Kaufmann: San Mateo, CA, pp. 416–421.
5. Y. Nakanishi, "Capturing preference into a function using interactions with a manual evolutionary design aid system," in *Genetic Programming, 1996*, Late-Breaking Papers, 1996, pp. 133–138.

6. H. Takagi, "Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation," *Proc. of the IEEE*, vol. 89, no. 9, pp. 1275–1296, 2001.
7. W. Banzhaf, "Interactive evolution," *Handbook of Evolutionary Computation*, 1997.
8. J.-Y. Lee and S.-B. Cho, "Interactive genetic algorithm for content-based image retrieval," in *Proc. Asia Fuzzy Systems Symposium*, 1998, pp. 479–484.
9. Cyberware Inc., `http://ghiberti.cyberware.com`.
10. S. Gray, "In virtual fashion," *IEEE Spectrum*, vol. 35, no. 2, pp. 19–25, 1998.
11. R.S. Wright and M. Sweet, *OpenGL Superbible*, Waite Group Press, 1996.
12. A.L. Ames, D.R. Nadeau, and J.L. Moreland, *VRML 2.0 Sourcebook*, John Wiley: New York, 1996.
13. M.J. Kilgard, *The OpenGL Utility Toolkit (GLUT) Programming Interface API Version 3*, Silicon Graphics, Inc., `http://reality.sgi.com/mjk_asd/spec3/spec3.html`.
14. H.A. David, *The Method of Paired Comparison*, Charles Griffin, 1969.
15. W. Niblack et al., "The QBIC project: Querying images by content using color, texture, and shape," in *Storage and Retrieval for Image and Video Databases*, SPIE, pp. 173–187, 1993.
16. K. Hirata and T. Kato, "Query by visual example: Content based image retrieval," *Advances in Database Technology, EDBT '92*, pp. 56–61, 1992.
17. V.E. Ogel and M. Stonebraker, "Chabot: Retrieval from a relational database of images," *IEEE Computer*, vol. 28, no. 9, pp. 40–48, 1995.
18. C.E. Jacobs, A. Findkelstein, and D.H. Salesin, "Fast multiresolution image querying," in *Proc. SIGGRAPH 95*, 1995.