공학석사학위논문

Color Image Quantization using Interactive Genetic Algorithm

대화형 유전 알고리즘을 이용한 이미지 색 양자화

2015 년 2 월

서울대학교 대학원

전기 · 컴퓨터공학부

조승현

공학석사학위논문

Color Image Quantization using Interactive Genetic Algorithm

대화형 유전 알고리즘을 이용한 이미지 색 양자화

2015 년 2 월

서울대학교 대학원

전기 · 컴퓨터공학부

조승현

Color Image Quantization using Interactive Genetic Algorithm

대화형 유전 알고리즘을 이용한 이미지 색 양자화

지도교수 문병로

이 논문을 공학석사학위논문으로 제출함

2014 년 12 월

서울대학교 대학원

전기 · 컴퓨터공학부

조승현

조승현의 석사학위논문을 인준함

2014 년 12 월

위 원 장	박 근 수	(인)
부위원장	문 병 로	(인)
위 원	Srinivasa Rao Satti	(인)

Abstract

Color Image Quantization using Interactive Genetic Algorithm

Seunghyun Jo
Department of Electrical Eng. & Computer Science
College of Engineering
The Graduate School
Seoul National University

Color image quantization is the problem of converting the given image to the image with limited number of colors used, maximizing the similarity. Previous studies measured the similarity as the smaller color difference by pixels, which is hard to evaluate the context of the image. In this thesis, a novel approach of considering the context is shown. Interactive genetic algorithm is applied to reflect the human knowledge in generating the quantized images. The experiment of the system has been done with 12 subjects and four test images, and the result shows that the quantized images generated with the interactive system are evaluated to be more suitable than those with the traditional optimization methods.

Keywords: Color image quantization, Interactive genetic algorithm

Student Number: 2013-20885

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Color image quantization is one of the most widely used image processing techniques, where the number of colors used in the image is to be reduced to a specific value.[2] The aim of the task is to minimize the distortion from the original image. In previous studies, this has been measured using the distance between the original color and the transformed color of each corresponding pixels. This concept is straightforward and can easily be calculated, thus has been used as a general measure. However, to find the color tuple which minimizes the difference is known to be NP-hard.[3] Therefore, a number of studies have been done using several optimization techniques.

The previous measure, however, lacks the ability of measuring the distortion of overall context of the image. Since the measure compares the difference of individual pixels, it does not contain the difference between the original and transformed images itself. This means that there might be the case where the pixel difference is minimized, but the quantized image is far from the optimal solution. Instead of using this measure, we can design another method of measuring the difference as a whole. While the comparison by pixels is easy to be formulated, However, it is hard to set the measurement criteria of the context of images that is not easily defined as a numerical value.

Interactive genetic algorithm (IGA) provides the solution to this type of problem. IGA is a branch of genetic algorithm that uses the human input for the computation of fitness values. It is used in the field such as music and graphics where the well-defined measure does not exist. In a typical experiment of IGA, a subject is asked to score the given objects according to his/her own criteria within the predefined range, and the system evolves the solution according to the input. Due to this intrinsic subjectiveness to IGA, it might be difficult to maintain the consistency of the experimental result. However, it is considered a strong tool which extends the capability of traditional genetic algorithm.

In this thesis, a novel approach using IGA for color image quantization is proposed. Given the image to be quantized, the system generates initial set of chromosomes which contains the information of selected colors. Randomly selecting the chromosomes and asking the subject to select the best solution gives the initial fitness evaluation criteria. After that, the solutions are developed by the iteration of evolutionary process and subjective user input. After a number of iterations, the converged solution is finally made. The generated solution shows the difference according to the subjects.

The rest of the paper is organized as follows: In Chapter 2, a background of color image quantization and genetic algorithms is presented. Chapter 3 is dedicated to the proposed system for finding the solution is described. Chapter 4 shows the experimental results and discussion. Finally, the conclusion is given in Chapter 5.

Chapter 2

Preliminaries

2.1 Color Image Quantization Problem

Color image quantization is the process of finding the color palette which describes the given image best. The set of colors are selected to represent the image, and the mapping from the color space to the color set is computed.[6] The aim of the quantization is to convert the original image to the one with the selected colors only with the minimum difference between original color and the converted color.

We can formulate the problem in another way. Color image quantization is to divide the set S of colors in image into K disjoint subsets S_k with $1 <= k <= K$, where K represents the number of quantized colors, and map each set to its representative color z_k.[4] Now the problem is to find the best partition of the color set. Here, the error function is defined as follows:

$$E\left(S_1, S_2, \cdots, S_k\right) = \sum_{1<=k<=K} \sum_{c_i \in S_k} \langle c_i, z_k \rangle \cdot p\left(c_i\right)$$

where $\langle c_i, z_k \rangle$ denotes the distance between the color c_i and z_k, and $p(c_i)$ denotes the number of pixels with the color c_i. The distance is measured with the Euclidean distance in a color space.

Finding the set of colors which minimizes the error is one of the well-known NP-hard problems.[3] Therefore, like the other NP-hard problems, several approaches using various optimization methods have been researched. Linde saw this problem as a partitioning problem, and applied K-means clustering to solve this problem.[8] The computational efficiency of this work made it used as a postprocessing method in further researches. Brucker proposed another clustering approach, which is a hybrid of evolutionary algorithm and K-means clustering.[4]

Another approach is done by Omran[9] using particle swarm optimization (PSO). This algorithm is modeled after the movement of bird flocks to find suboptimal solutions. In this problem, K centroids, each represents color, are selected randomly and refined with PSO.

Quantization with differential evolution (DE) is also researched.[10] DE is another stochastic optimization method where candidate solutions are moved in the search space to find an improvement, according to the parameters. In this research, self-adaptiveness is applied, which means that the system automatically adjusts the parameters.

2.2 Genetic Algorithm

Genetic algorithm is one of the metaheuristic optimization methods which is modeled after natural selection. There is a population of candidate solutions called chromosomes, and the operations are done to these chromosomes to generate new solutions called offspring. Through the iteration of evolutionary operations and the evolution of the population, the solutions are developed and have higher fitness values. After a number of generations, the convergence of the population occurs, and the solution with the highest fitness value is presented as the final result.

A typical structure of GA is shown in Algorithm 1. Here, the population of n solutions is generated initially, and a number of genetic operations are done within this pool of chromosomes. The number of offspring k is chosen between 1 and n, the size of the population. The algorithm using the small value of k is called steady-state genetic algorithm, and that with the larger value of k is called generational genetic algorithm.

Algorithm 1 A typical procedure of genetic algorithm

create a population of size n
repeat
  for $i := 1$ **to** k **do**
    choose $parent_1$ and $parent_2$ from the population
    $offspring_i := crossover(parent_1, parent_2)$
    $offspring_i := mutation(offspring_i)$
  end for
  replace $offspring_1, \cdots, offspring_k$ with k solutions in the population
until stopping condition
return the best individual in the population

As mentioned above, the individual which forms the population are called chromosomes. Each of the chromosomes represents one solution which solves the problem. Choosing the proper representation of chromosomes is essential to obtain the solution with high fitness. One of the mainly used representation methods is binary encoding, where the chromosomes are represented with digits 0 and 1, and other methods such as real number, multi-dimensional string are also used.

Each chromosome has fitness value, which shows how well the chromosome fits to the problem. Thus, the aim of the genetic algorithm is to find the chromosome with high fitness value. Objective function maps a chromosome to its fitness value, and defining good objective function is also necessary for efficient search in the solution space.

In each evolutionary step, offspring is generated from the selected chromosomes from the population. Following the natural selection, the chromosomes with the higher fitness are more likely to be selected as parents. One of the methods to implement this feature is roulette-wheel selection, where the population of selection reflects the fitness of the chromosomes.

After selecting the parents to produce offspring, the genetic operations are done. Crossover operation produces the offspring from the parents by inheriting part of the chromosomes from each parent. By this operation, the offspring receives the features from the parents, and are likely to preserve the good schema to the next generation. Mutation operation is done rarely to supplement the variety of the population. Modification of only the part of the chromosome can make totally different solution, which helps exploration of the solution space.

After offspring are generated, these are inserted to the population, and the same number of chromosomes is removed. The replacement policy should also be carefully selected since simply replacing the chromosomes with the least fitness values can cause premature convergence.

When the stopping condition is met, the process of genetic algorithm terminates, and it returns the solution with the highest fitness value. In this situation, most of the population shows the similar features, which is called convergence. It is typically seen that the longer time to reach the convergence produces the solution with higher fitness value. Therefore, adjusting the factor to find the efficient condition is important.

2.3 Hybrid Genetic Algorithm

Genetic algorithm has an advantage of finding the optimal solution in the search space compared with other heuristic methods. However, when it comes to the situation where the near-optimal solution is found, genetic algorithm lacks the ability to find the local optimum efficiently. To overcome this issue, additional operation of local optimization after crossover and mutation operation can be done to find local optimum, which is called hybrid genetic algorithm. A typical procedure of hybrid genetic algorithm is shown in Algorithm 2.

When applying local optimization to GA, we can see that the local optimization operation takes most of the time, more than genetic operations. However, it is widely known that hybrid GA shows more capability of finding solutions with higher fitness compared to traditional GA, since local optimization complements the weakness of genetic algorithm by fine tuning the solution near local optimum.

Algorithm 2 A typical procedure of hybrid genetic algorithm

create a population of size n

repeat

  for $i := 1$ **to** k **do**

    choose $parent_1$ and $parent_2$ from the population

    $offspring_i := crossover(parent_1, parent_2)$

    $offspring_i := mutation(offspring_i)$

    $offspring_i := local\_optimization(offspring_i)$

  end for

  replace $offspring_1, \cdots, offspring_k$ with k solutions in the population

until stopping condition

return the best individual in the population

2.4 Interactive Genetic Algorithm

Interactive genetic algorithm (IGA) is a branch of genetic algorithm in which the fitness evaluation process involves the user input process to find the fitness value of solutions. It is the part of interactive evolutionary computation (IEC), which is considered a technology which contains the human intuition in the target system.[11] It sometimes has a broader definition of the system having an interactive human-computer interface, but here we consider the "interactiveness" as the involvedness in the fitness evaluation, which is adopted in the original, narrow definition of IEC.

The system of IGA has initialization step, iteration of selection, crossover, and replacement, and termination like typical genetic algorithm framework. The fitness of chromosomes is evaluated based on the user input. However, since the human evaluation step requires much more resource than other computational steps, only in some cases the user input is performed, and the fitness of other chromosomes are approximated with the user input. A typical method is assigning the fitness value according to the chromosomes with user evaluation and the distance to the target.

Since it could be applied to the problems which are computationally difficult, it has wide variety of applications including graphic art, music, database retrieval, and so on. Takagi used IEC to generate the image filter which fits the user preference[12], and Bergen worked on the evolution of vector image using interactive genetic algorithm.[1]

In addition to these studies on applications, non-applicational research on IEC has also been done actively. One of the main issues is getting rich information from the subjects efficiently, with fewer steps to reduce human fatigue.[7] Also, work on the framework for interactive evolutionary algorithm is done, which is to be used for the tool of interactive evolutionary computation for general purpose.[5]

Chapter 3

The Proposed System

3.1 Formulation of the Problem

The specification of the problem used in this experiment is shown as follows. The images used in the experiments are from the standard test images typically used for the test of image processing and image compression algorithms, and are shown in Figure 3.1. The size of the images is 512x512 pixels, and the number of colors in these images is to be reduced to 16, which is the same conditions as the previous studies.[9, 10]

3.2 The Framework of the System

The framework of the system follows the typical structure of GA, with modules for subjective user input, which is described on Figure 3.2. Two main parts of the system are evaluation module and GA module. In the initial step and the steps between GA iterations, the subject is asked to select the fittest image among four given images. GA process is done

(a) Airplane

(b) Baboon

(c) Lena

(d) Pepper

Figure 3.1: Standard test images used in the experiment

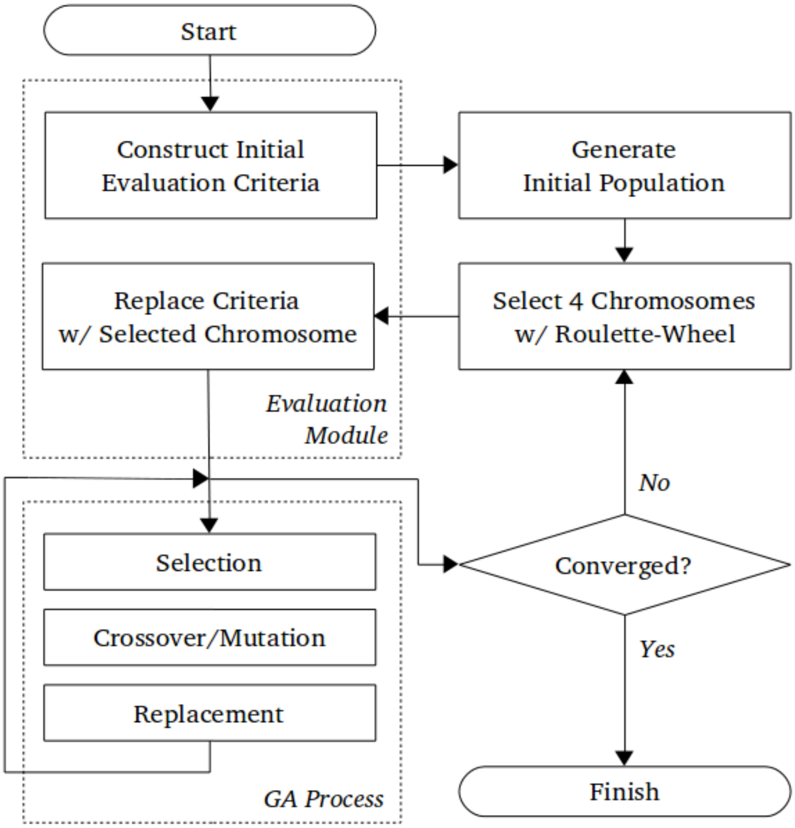

Figure 3.2: The framework of IGA system

after the initial user input. In each step of the iteration, four images are selected and shown to the subject to be selected, and the fitness measurement criteria are updated according to the selection, which is to be explained in detail in the later section. After that, the standard procedure of selection, crossover, mutation, and replacement are done. For the efficiency, user input is processed only on some of the iteration.

The detailed specification of the system is as follows. The number of chromosomes is 100. Initially, the subject selects the images 10 times, which forms the initial evaluation criteria. For every iteration of the evolutionary step, the selection is done once, and the evolutionary process is done 100 times. In each of the process, two chromosomes are selected. Roulette-wheel selection method is used when selecting the parents, and the selection pressure is set to 100, that is, the probability of the fittest chromosome is 100 times larger than that of the least fit chromosome. After the selection, the offspring is generated with crossover and mutation operation, and is replaced with the randomly selected chromosome in the population with uniform probability of selection. The probability of the mutation is 0.01, which means that the mutation is done approximately once every 100 times. When the average fitness is larger than 99% of the maximum fitness, it is considered that the convergence condition is met, and the iteration terminates. Finally, the chromosome with the highest fitness value is returned.

3.3 The Structure of Chromosomes

Since there are several ways to model the problem, a number of encoding methods can be applied. When the problem is considered partitioning problem, the mapping from the original color to the converted color is to be modeled. Otherwise, we can set the data of converted image itself as a chromosome. These methods are straightforward and they correspond to the concept of the problem. However, they have the issue of complexity since they need huge amount of memory to represent one solution. The method of color mapping needs the space of the number of elements in the color space, and the method of encoding result image needs the size proportional to the size of the image. Since the size of the chromosome directly affects the performance of GA system, the encoding method which results in large chromosome data should be avoided.

In this system, chromosomes consist of 16 integer values, each representing the color used in the quantized image. The original image is transformed into the quantized image by selecting the closest color on each pixel. The transformed image becomes the phenotype of the chromosome and is shown to the evaluator to measure the fitness. This encoding method has the advantage of smaller chromosome size compared to the methods mentioned above. However, this lacks the capability of representing every quantized image since it is not the case where every quantized image is made with selecting the closest color. Nevertheless, reduction of search space resulting in the efficient solution search was the main reason of selecting this scheme.

Each integer value in the chromosome represents one color in the RGB color space. One color value is considered to have three integral values ranging from 0 to 255, each representing the factor of red, green, and blue. Therefore, the range of the value in the chromosome is $[0, 255^3]$. In a chromosome, the elements are sorted in order of distance from 0. When generating the chromosome, colors are extracted from the original image, which is also for the reduction of solution space.

3.4 Interactive Step and Evolutionary Step

As explained above, the initial part of the system is the construction of initial evaluation criteria with the user input. Prior to the first iteration of the GA process, the evaluator is asked to select the fittest image among four randomly generated quantized images. This is repeated until enough chromosomes for fitness evaluation are selected. After this initial selection, the evolutionary process begins.

Firstly, initial population is generated and the fitness values of the chromosomes are measured. After that, the main iteration part begins, which starts with the user evaluation. Four candidate images are selected randomly from the population, and one of them are selected by the evaluator. The selected solution is used as the new pivot for fitness calculation, thus performs the role of the attractor. Right after the selection, the fitness values are evaluated again according to the new fitness measure, and the sequence of selection, crossover, mutation, and replacement is performed.

One-point crossover method is used in crossover operation. That is, one integer from 0 to 16, the length of the chromosome, is chosen randomly, which is used as the cut point. The offspring is generated with the gene of parents, with the value filled with the first parent prior to the cut point and with the second parent after the cut point. Since the value in the chromosome is sorted by the distance from the origin, it is likely that the offspring preserves the feature of the parent chromosomes.

Mutation is done with the small amount of probability. When it occurs, one of the integer values of the chromosome is replaced with another value representing the randomly selected color from the original image. Since the quantized image is made with the selection of the closest color in the chromosome, the modification of one color can cause huge difference of the result image.

3.5 Measurement of Fitness

The fitness of the chromosomes is measured by how similar the chromosome is to the selected solutions which form evaluation set. Initially, the subject is asked to select one chromosome out of four candidates, and repeating this makes the initial evaluation set. The fitness of chromosome is calculated by getting the weighted average of the inverse of the distance between this and the selected chromosomes. Each time the new chromosome is selected, it is added to the evaluation set, and the weights of previous elements are damped. In short, the fitness is calculated with the following expression:

$$\text{fitness}(x) = \frac{\sum_{i=0}^{n} (\gamma^{t_i}/(dist(x, c_i) + 1))}{\sum_{i=0}^{n} \gamma^{t_i}}$$

where n is the number of elements in the evaluation set, γ is a user-defined positive constant less than 1, t_i is the difference between the period of the addition of the evaluation element and the current iteration, $dist(x, c_i)$ is Euclidean distance between the results generated with x and i-th evaluation element.

3.6 Local Optimization

To strengthen the ability to find the local optimum, the local optimization operation is applied. This is done with adjusting the single value of colors by a small amount until no improvement is found. The local optimization function is shown in Algorithm 3.

The fitness calculation is done six times at each of the iteration, and at most $6 * 16 * 5 = 480$ times in total. Since fitness calculation takes much

Algorithm 3 Local optimization function

Require: $[c_1, \cdots, c_n]$: the chromosome with n integers representing colors

$range := 16$

while no improvement is found **do**

  for $i := 1$ **to** n **do**

    calculate fitness with c_i, incremented and decremented by $range$

    $c_i :=$ the adjusted color with maximum fitness

  end for

  $range := range/2$

end while

return the optimized chromosome

time, local optimization consumes most of the time like the other hybrid genetic algorithm.

Chapter 4

Experiment

4.1 Experimental Setup

The experiment was done with 12 subjects. Two types of experiments are performed to see the capability of the system. The first experiment is to generate the quantized image using the system, and the second experiment is to compare the result with the other quantized image.

In the first experiment, the subjects were asked to use the system with four given images to generate the result according to their own decision. To show that the system accurately finds reproducible solutions, there were repetitions for the same images. The similarities between the results from the repeated experiments are to be compared.

In the second experiment, the subjects were given three images. One of the images was the image generated by the subject, and another one was generated by another subject. The third image was from the previous studies.[9, 10] Among them, they were asked to select the most feasible

Table 4.1: The number of user inputs until convergence

Image	The number of user inputs												Average
Airplane	43	38	37	34	56	39	45	39	49	46	48	35	42.39
	39	42	34	37	52	42	40	45	52	41	47	37	
	44	40	36	35	51	43	47	42	45	42	42	42	
Baboon	34	30	37	36	43	46	36	41	42	36	45	47	39.03
	45	34	35	42	40	42	38	43	32	39	38	48	
	37	34	41	41	39	37	32	39	35	41	37	43	
Lena	42	44	39	38	33	27	35	40	39	35	38	29	36.58
	39	34	39	35	29	34	39	34	43	38	36	41	
	44	36	38	40	30	32	29	37	41	32	39	39	
Pepper	41	35	39	29	33	40	34	42	39	32	38	29	35.92
	33	37	42	35	36	32	39	35	37	39	42	31	
	33	34	43	32	31	38	30	34	40	37	33	39	

image according to his/her criteria. This was done with four test images, and this experiment was done on different day from the day of the first experiment to reduce the bias.

4.2 Generated Image

On average, 38.47 user inputs were needed to generate the result. Ten inputs were used for the construction of the initial evaluation set, and the rest were used to evolve solutions until convergence. Table 4.1 shows the number of user inputs in each of the trials. The result showed varying numbers with images and subjects.

Table 4.2: Comparison of MSE of results

Image	w/ the same subject	w/ different subjects
Airplane	39.752	52.263
Baboon	28.718	33.133
Lena	20.612	20.675
Pepper	25.656	27.559

The samples of the generated images are shown in Figure 4.1. For some image, the difference is quite large so that the different results are easy to distinguish, while the result from other images shows little difference each other.

Table 4.2 shows the comparison of the similarities between the results with the same subject and those with different subjects in terms of mean squared error (MSE). In this table, it is shown that the MSE between the results with the same subject shows less value than the others, and the degree of difference varies according to the type of test images.

4.3 Comparison of Results

The subjects are given three quantized images from the same test image. Three images are the image generated with the same subject, that with another subject, and the quantized image in Su's research[10]. These images are shown in random order, and they are asked to select the best images among them. The result is shown in Table 4.3.

In total, subjects selected the image that he/she had generated in 27 out of 48 total cases, which is 56% of the total cases. The interesting thing

(a) Airplane

(b) Baboon

(c) Lena

(d) Pepper

Figure 4.1: Sample experimental results

Table 4.3: The result of the selection

Image	w/ the same subject	w/ different subjects	Previous Work
Airplane	6	4	2
Baboon	8	2	2
Lena	6	2	4
Pepper	7	4	1
Total	27	12	9

is that, a number of subjects selected the image that the other subjects had generated. The images with the IGA system are chosen 39 out of 48 cases, which is over 80%. This shows that the quantized images generated with interactive system are more likely to be considered better quantization than the images generated with traditional optimization system where the objective is to minimize the pixel difference from the original image.

Chapter 5

Conclusion

Color image quantization has been widely studied due to its intuitiveness and applicability to image compression and other image processing-related algorithms. So far, the studies have focused on minimizing the difference between the original and converted images, and it lacks the capability of preserving the context recognized by human. In this research, a novel approach for applying this has been proposed. Since it is hard to measure quantitatively, the methodology of interactive genetic algorithm is used. With a number of user input, the system develops solutions to show the result that fits the user's own criteria.

The experimental result shows that the quantized images generated by the IGA system are evaluated more feasible than those without interactive step. When selecting the best quantized image among several images given, the subjects chose the image that they had generated at the rate over the half, and the rate increases to 80% when also considering the image generated by the other subjects. This result is encouraging since it shows the effect of the application of user input when generating the quantized image. Moreover, this research and its experimental result have room for improvement. The quantitative research on the effect of interactiveness and the generalization of the quantization to the other images can be studied in the further researches.

In this thesis, the application of interactive genetic algorithm to the image quantization problem is suggested, where the approach and methodology can be generalized to other problems. Through this and the following researches, it is expected to find ways to utilize the human knowledge to image-related problems that are computationally difficult.

Bibliography

[1] Steven Bergen and Brian J Ross. Automatic and interactive evolution of vector graphics images with genetic algorithms. *The Visual Computer*, 28(1):35–45, 2012.

[2] J-P Braquelaire and Luc Brun. Comparison and optimization of methods of color image quantization. *Image Processing, IEEE Transactions on*, 6(7):1048–1052, 1997.

[3] Peter Brucker. On the complexity of clustering problems. In *Optimization and operations research*, pages 45–54. Springer, 1978.

[4] Bernd Freisleben and Andreas Schrader. An evolutionary approach to color image quantization. In *Evolutionary Computation, 1997., IEEE International Conference on*, pages 459–464. IEEE, 1997.

[5] Mario García-Valdez, Juan J Merelo, Leonardo Trujillo, Francisco Fernández-de Vega, José C Romero, and Alejandra Mancilla. Evospace-i: a framework for interactive evolutionary algorithms. In *Proceeding of the fifteenth annual conference companion on Genetic and evolutionary computation conference companion*, pages 1301–1308. ACM, 2013.

[6] Paul Heckbert. *Color image quantization for frame buffer display*, volume 16. ACM, 1982.

[7] Raffi Kamalian, Ying Zhang, Hideyuki Takagi, and Alice M Agogino. Reduced human fatigue interactive evolutionary computation for micromachine design. In *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, volume 9, pages 5666–5671. IEEE, 2005.

[8] Yoseph Linde, Andres Buzo, and Robert M Gray. An algorithm for vector quantizer design. *Communications, IEEE Transactions on*, 28(1):84–95, 1980.

[9] Mahamed G. Omran, Andries P. Engelbrecht, and Ayed Salman. A color image quantization algorithm based on particle swarm optimization. informatica 29:261–269, 2005.

[10] Qinghua Su and Zhongbo Hu. Color image quantization algorithm based on self-adaptive differential evolution. *Computational intelligence and neuroscience*, 2013:3, 2013.

[11] Hideyuki Takagi. Interactive evolutionary computation: Fusion of the capabilities of ec optimization and human evaluation. *Proceedings of the IEEE*, 89(9):1275–1296, 2001.

[12] Hideyuki Takagi and Norimasa Hayashida. Interactive ec-based signal processing. In *4th Asia-Pacific Conference on Simulated Evolution and Learning (SEAL2002), Singapore*, pages 375–379, 2002.

요약

이미지 색 양자화 문제는 주어진 이미지와 사용할 수 있는 색깔의 수가 주어졌을 때 원본 이미지와 비슷한 양자화 이미지를 생성하는 문제이다. 기존의 연구에서는 픽셀별 색깔의 차이를 최소화하는 것을 비슷함의 기준으로 정했는데, 이는 전체 이미지의 정보를 반영하지 못하는 문제가 있다. 이 논문에서는 이미지의 정보를 반영하여 양자화하는 방법을 제시한다. 대화형 유전 알고리즘을 이용함으로써 양자화 이미지를 만드는 과정에서 인간의 지식을 적용할 수 있는 시스템을 구현하였다. 실험은 12명의 피험자에 대해 네 개의 이미지를 이용하여 진행되었으며, 대화형 시스템을 통해 만들어진 양자화 이미지가 기존의 최적화 방법을 이용하여 만들어진 이미지보다 인간의 관점에서 더 적절한 이미지라는 결론을 얻을 수 있었다.

주요어: 이미지 색 양자화, 대화형 유전 알고리즘

학번: 2013-20885