A FUNCTION APPROXIMATION METHOD FOR IMAGES WITH GRADING REGIONS

KENTARO MIYAMOTO
Graduate School of Systems and Information Engineering
University of Tsukuba, Tennodai 1-1-1, Tsukuba, Ibaraki, 305-8573, Japan
kenta@wslab.risk.tsukuba.ac.jp

TETSUO KAMINA
CREST Program, Japan Science & Technology Agency
Tennodai 1-1-1, Tsukuba, Ibaraki, 305-8573, Japan
kamina@acm.org

TETSUO SUGIYAMA∗, KEISUKE KAMEYAMA†, KAZUO TORAICHI‡ and YASUHIRO OHMIYA§
Center for Tsukuba Advanced Research Alliance, University of Tsukuba
Tennodai 1-1-1, Tsukuba, Ibaraki, 305-8573, Japan
∗sugi@wslab.risk.tsukuba.ac.jp
†kame@emerald.risk.tsukuba.ac.jp
‡toraichi@risk.tsukuba.ac.jp
§yohmiya@wslab.risk.tsukuba.ac.jp

Received 21 August 2006
Revised 25 September 2007
Accepted 18 October 2008

In the last decade, many function-approximation methods for illustration images have been proposed. However, these methods cannot process more complex full color images effectively, even when some simple regularity of color transitions such as gradations appear in images. In this paper, we propose a rigorous function-approximation method that is applicable for gradation images. This method performs image segmentation that recognizes a gradation pattern as one region. This segmentation is performed by recursively approximating the tones of colors using multiple regression analysis of 2-variable functions. We compare the proposed method and the conventional methods with regard to the quality of enlarged images and the efficiency of image description and reproduction. The experimental results show that our approach is higher in quality, smaller in data size, and faster in computational costs than the conventional methods.

Keywords: Image coding; image processing; function-approximation; fluency information theory.

1. Introduction

Quality-maintained resolution conversion technique is essential for displaying and printing digital images on various devices. One way for performing it is to use
“vector format” for describing the images. In this approach, contours of “color regions,” namely a set of contiguous pixels of the same color, are function-approximated. In vector format, images consist of a set of drawing information such as coordinates of points or parameters of lines; therefore, it guarantees quality-maintained affine transform. It is suitable for describing contour-outlined images such as fonts, drawings of figures, and illustrations.

Vector format images may be created by using some drawing software; however, in many cases we must deal with images in raster format as well (e.g., importing printed illustrations into computers by using scanners or digital cameras). Therefore, automatic function-approximation techniques that convert raster illustrations into function-approximated format can be beneficial in desktop publishing (DTP) applications. For example, the Fluency function-approximation method\textsuperscript{1–6} that function-approximate contours of equi-colored regions has been proposed.

In the so-called full-color images, on the other hand, the notion of “contiguous pixels of the same color” is no longer important, because they have too many colors. The common approach for resolution conversion of such images is to interpolate the tones of colors. In the last decade, many interpolation methods have been proposed.\textsuperscript{7–11} Even though some quality deterioration is inevitable for these methods, interpolation is the most common approach for resolution conversion of full color images. This is because the contour approximation methods detect vast number of tiny color regions thus requiring much encoding/decoding cost and huge file size. Furthermore, since contours are segmented by tiny intervals, approximating functions cannot be selected properly, thus approximation error makes the images visually deteriorated. Therefore, function-approximation of contours so far, has not been considered applicable for full color images.

However, we can see that some full-color images contain some regularity of its tones that may be represented by simple functions. One example of such regularity is known as \textit{gradations}. So far, almost no attempts toward automatic detection and function approximation of gradation images have been made. If gradation regions can be detected and rigorously function-approximated, quality-maintained affine transform can be performed on such images thus expanding the horizon of high quality DTP applications.

In this paper, we propose a rigorous function-approximation method for images with gradation regions. This method proceeds as follows:

(1) Image segmentation that divides a image into multiple regions. In this stage, a gradation pattern that can be function-approximated by using multiple regression analysis of 2-variable functions is recognized as one region.

(2) Function-approximation of contours of regions that are segmented by the previous stage, by using the conventional method.

By this approach, quality-maintained resolution conversion of gradation regions can be performed without generating numerous tiny color regions. The experimental results show that our approach is higher in quality, smaller in data size, and faster in encoding/decoding costs than the conventional methods.
This paper is structured as follows. In Sec. 2, we explain an overview of contour approximation technique for illustration images as a preliminary of this research. In particular, our approach is based on the Fluency function-approximation method. There we also point out the motivation of this research. In Sec. 3, we propose a new function approximation method for gradation images. In particular, this section explains our image segmentation algorithm that recognizes a gradation pattern as one region and the method to approximate the gradation tones in detail. In Sec. 4, we show the experimental results and compare our approach with the conventional method of contour approximation and interpolation. Finally, Sec. 5 concludes this paper.

2. Image Description and Function Approximation Format

2.1. Image description in general

In general, an image consists of a set of regions. If each region occupies a relatively large space in the image, function approximation of contours of regions becomes an effective method for image description, because it supports quality-maintained affine transform with relatively small data size. How the image is divided into multiple regions depends on how the regions are modeled. The simplest model of regions may be regarding a region as a set of contiguous pixels of the same color. Using such a simple model, the application of the function approximation method is limited to simple illustration images that include small variation of colors like cartoons. In the full-color images, therefore, the common approach for image description is using the raster format that regards an image as just a set of pixels. However, if we use another model of regions, we may also apply the function approximation method to more complex images.

In this section, we overview the function approximation method in general, and explain the conventional method of function approximation of contours of “color regions” as the preliminary of this research.

2.2. Overview of contour approximation

The JSD diagram\textsuperscript{12} in Fig. 1 illustrates the function-approximation method in general. In this diagram, a rectangle represents one process. A rectangle dangling from the upper rectangle represents a subprocess of the upper process. If multiple subprocesses exist, the execution begins from the left-most subprocess, followed by the right neighbor process. The symbol “*” that appears on the right-upper side of rectangles denotes the Kleene’s closure.

At first, this approximation method divides the input raster image into multiple regions. In the conventional methods, the contiguous pixels painted in the same color are regarded as forming one region so nothing particular is performed on this subprocess. For all the regions, it then runs a contour tracing algorithm.\textsuperscript{6,13} The traced contours are segmented at corner points\textsuperscript{14} and approximated by using
piecewise approximating functions. The accuracy of this approximation strongly affects the fidelity of the contour approximations to the original image. Finally, colors of all the regions should be described in some ways; however, since colors in one region are all identical in the conventional methods, nothing particular is performed in this subprocess (just storing the color for each region).

In the recent years, there was much work on improving the quality of function-approximation. In particular, there is a novel work on contour tracing proposed by Sugiyama et al.\textsuperscript{6} and the Fluency function-approximation method. Our approach is based on these works. In the following subsections, we explain each of them.

2.3. Contour tracing

The contours of binary images such as characters can be effectively traced by using Freeman’s chain-code.\textsuperscript{13} If it is applied to multicolor illustrations, however, a problem arises; contours shared by different regions are traced twice and each traced contour may be approximated by different functions. As a result, gaps will appear between the different approximating functions, thus approximated images are visually deteriorated. Many function-approximation methods actually suffer from this problem.\textsuperscript{15, 16}

To solve this problem, Sugiyama et al. proposed a contour tracing algorithm that can effectively trace multicolor illustrations.\textsuperscript{6} Instead of tracing pixels, it traces the corners of the pixels, assuming each pixel to be a square tile, and the shared corner is traced only once. Therefore, in this method the aforementioned problem does not arise.

2.4. Fluency function approximation of contours

For historical reasons, Bezier curves are commonly used for function-approximation.\textsuperscript{17} On the other hand, Toraichi et al., proposed the Fluency functions for approximation. Fluency functions are piecewise polynomials that are characterized by a parameter that denotes the times of the approximating function’s continuous differentiability that represents its smoothness. By adaptively selecting the
class of functions w.r.t. the shape of contours, high quality contour approximated images are reproduced with small amount of data.

One of the novel feature of the Fluency function-approximation method is its effectiveness of automatic approximation. In approximation using Bezier curves, control points that decide the shape of curves are not located on the pixel sequences extracted by the contour tracing process, so it requires complex calculation and accurate approximation is difficult. In the Fluency function-approximation, on the contrary, free curves are drawn by convolution of the sampling function with the contour pixel coordinates. Therefore, control points are represented as values of sampling points that are located on the pixel sequences, so it does not require complex calculation and accurate approximation is achieved. Fluency function approximation method is now in practical use in the field of DTP.

2.5. Motivation of this research

Even though the function approximation of contours effectively describes simple illustrations, it is not suitable for processing more complex images that have numerous colors. If it is applied to such images, numerous color regions are detected, thus data size becomes very large and their quality is visually deteriorated due to approximation error.

For example, the contour approximation method cannot properly process gradation images. A result of applying the Fluency function approximation method to the gradation image shown in Fig. 2 is shown in Fig. 3. The scaled image of Fig. 3 is shown in Fig. 4. Table 1 summarizes the number of extracted contours, the number of extracted regions, and the encoding cost. It is clear that even for such a simple image, it requires a lot of computational time and memory capacities, because a large number of color regions and contours are extracted.

The source of this problem is the region segmentation subprocess explained in Fig. 1. There, only single color is allowed in a region. Even when the values of pixels are different to a small extent, they are recognized as different regions. This problem may be solved by treating a contiguous region having a regular transition of colors as one region, with a description of the transition by some approximating functions. A region with color gradation can be a candidate of such a case. In

Fig. 2. Input image: (BMP, 24bit/pixel, 528×278, 430KB).
this case, a new image segmentation algorithm that corresponds to the box labeled “Divide the input image into multiple regions” in Fig. 1 is required. Furthermore, when we develop a new image segmentation algorithm, it also becomes necessary to devise a way to reproduce the color transition in the regions that corresponds to the box labeled “For all the regions, describe color of region.” The purpose of this work, therefore, is to develop the function-approximation method in Fig. 1 to process gradation images.

3. Function Approximation Method for Gradation Images

To tackle the aforementioned problems, we propose a new function approximation method for gradation images. In particular, we refine the region segmentation subprocess in Fig. 1 to register a gradation pattern as one region.
3.1. Outline of the region segmentation

The control flow of the proposed region segmentation is shown in Fig. 5.

At first, regions that are apparently recognized distinct are segmented in the preprocessing step named “Global Segmentation”. This is effective for saving the processing cost. Secondly, colors in regions are approximated by the most suitable 2-variable functions among planes, spheres, and quadratic surfaces using multiple regression analysis. This process is called “Region Approximation”. Next, an approximation error is evaluated by calculating the mean square error between the values of approximating functions and the real pixel data. If the error is smaller than the preset threshold, the region segmentation is finished. Otherwise, the process repeats the “Region Segmentation using 2-variable functions” until the approximation error becomes smaller than the threshold.

3.2. Notations

At first, we define the terms used in the following algorithms. A pixel in an image $X$ is written $p = (i, j)$, and the value of $p$ is written $I(p) = [I_1(p), I_2(p), I_3(p)]^T$. Each $I_n(p)$ represents each RGB value, respectively. The number of pixels in $X$ is written $|X|$. Similarly, the number of pixels in region $R$ is written $|R|$. A set of 8-neighbors of pixel $p$ is written $\{p_c; c \in S_1\}$, where $S_1 = \{1, \ldots, 8\}$ (Fig. 6). Label number is written as $\lambda$.

3.3. Global segmentation

Global Segmentation is based on the labeling algorithm. The idea of this algorithm is that if the distance between the neighboring pixels is smaller than the preset threshold, we consider that these pixels are in the same region. This policy will
accommodate pixels with small change in color, to a single region. The algorithm is shown in Fig. 7 with ML-like pseudo code. Below, we explain it in detail:

Line 2: Search an unlabeled pixel $p$ according to the trace mode shown in Fig. 8.

Lines 3–4: If no unlabeled pixels are found, the algorithm stops.

Line 5: Attach a label $\lambda$ to $p$.

Line 6: Jump to the subprocess $\text{globalSegmentationInner}$ explained below.

Line 7: Recursively continue the same process with a new label $\lambda + 1$, until there are no labeled pixels.

Line 10: Obtain the pixels in a similar color with $p$ from $p$'s 8-neighbors by calculating the distance $d_1$ between $p$ and each of its 8-neighbors, by the equation

$$d_1 = \sqrt{\frac{\sum_{n=1}^{3}(I_n(p) - I_n(p_c))^2}{3 \times 255^2}}, \text{ for each } c \in S_1.$$ (1)

If $d_1$ is smaller than the preset threshold $\theta_1$, append $p_c$ to the set $l$.
Lines 11–13: Attach the label $\lambda$ for each pixels $p_i$ in $l$, and repeat the subprocess $\texttt{globalSegmentationInner}$ with pixel $p_i$.

By using this Global Segmentation, contiguous pixels with gradually changing colors is recognized as a single region.

### 3.4. Region approximation

Next, the regions recognized by the process of “Global segmentation” are function approximated by 2-variable functions. Each approximating function, denoted by $F(p)$, is obtained by using multiple regression analysis. Each $F(p)$ is defined by $F(p) = [F_1(p), F_2(p), F_3(p)]^T$, of which components are associated with each of RGB value $I_n(p)$, respectively. Each $F_n$ is a plane, a sphere, or a quadratic surface. A plane is expressed by

$$F_n(p) = ai + bj + c.$$  

A sphere is expressed by

$$F_n(p)^2 = ai^2 + bj^2 + ci + dj + eij + f.$$  

A quadratic surface is expressed by

$$F_n(p) = ai^2 + bj^2 + ci + dj + eij + f.$$  

Coefficient $a$ through $f$ are calculated by using multiple regression analysis. Then, the mean square error between the value of approximating functions and the real pixel data is calculated by

$$e_n = \frac{\sum_{p \in R}(I_n(p) - F_n(p))^2}{|R|}.$$  

The function with the smallest error $e_n$ is adopted as the region approximating function.
3.5. Judging the need of further segmentation

Sometimes, none of the approximating functions obtained by the above process cannot make the error $e_n$ small. In this case, the region should be further divided. For judging the need of further segmentation, we use the maximal value of $e_n$; since only one or two of RGB values vary to a large extent within a gradation pattern, we consider that the maximal value of $e_n$ is more effective than the minimal $e_n$ for using this judgment. If $\max e_n$ is larger than the preset threshold $\theta_2$, the following segmentation method is performed.

3.6. Region segmentation using 2-variable functions

Further region segmentation is based on the following labeling algorithm. For each region $R$ recognized by the algorithm explained in Sec. 3.2, we apply the algorithm shown in Fig. 9. The basic strategy of this algorithm is explained as follows. At first, a small subregion inside $R$ is function approximated. Then, this subregion is extended until the approximation error becomes unallowable. If a more suitable approximating function is found during the expansion, the old approximating function is replaced with the new one, and continues the expansion. The algorithm stops

```plaintext
1 regionSegmentation R =
2   let $F_n$ = multiRegression R m in
3   let $p = \text{pop } R$ in
4   let ref marked = [p] in
5   regionSegmentationInner p marked R $F_n$
6
7 regionSegmentationInner p marked R $F_n$ =
8   markAround p marked R $F_n$;
9   let $F'_n$ = multiRegression marked in
10  let $e'_n$ = meanSquareError marked $F'_n$ in
11  if $\sum (e_n - e'_n) \leq 0$ then
12    regionSegmentationInner p marked R $F'_n$
13  else
14    let $\lambda = \text{newLabel}()$ in
15    for each $p_i$ in marked
16      attach $p_i \lambda$;
17    if $R$ is not empty then
18      let $e_n = \text{meanSquareError } R F_n$ in
19      if $\max e_n \geq \theta_2$ then
20        regionSegmentation R
21
22 markAround p marked R $F_n$ =
23   let $l = \text{pixelsWithSmallColorDistance } p_1 R_n$ in
24   for each $p_i$ in l
25     (marked := append marked [p_i];
26      remove $p_i R_i$;
27     markAround $p_i$ marked R $F_n$)
28
Fig. 9. Region segmentation using 2-variable functions.
```
when there are no pixels that are not classified in a subregion in $R$. We explain the algorithm in detail (in Fig. 9, the mutable variable marked represents a set of pixels constructing a subregion):

Line 2: Obtain the new approximating function $F_n$ from a subregion of $R$. The subregion may be arbitrarily chosen. Its size $m$ (number of pixels) is preset.

Lines 3–4: Search a pixel $p$ in the subregion whose label is unchanged from the globalSegmentation algorithm explained in Fig. 7, according to the tracemode shown in Fig. 8. The pixel $p$ is popped from $R$ and marked.

Line 5: Jump to the subprocess regionSegmentationInner.

Line 8: Mark all the pixels whose value is near the pixel values of 8-neighbors of $F_n$ w.r.t. the preset threshold $\theta_3$. Details are explained as follows (In Fig. 9, see lines 26 to 32). Note that all the marked pixels are popped from $R$.

Line 23: Obtain the pixels whose value is near the pixel values of 8-neighbors of $F_n$ by calculating the distance $d_2$ between $F_n$ and $p$’s 8-neighbors, by the equation

$$d_2 = \sqrt{\max\{(I_n(p_c) - F_n(p_c))^2\}}, \quad \text{for each } c \in S_1. \quad (6)$$

If $d_2$ is smaller than the preset threshold $\theta_3$, append $p_c$ to the set $l$.

Lines 24–27: For all the pixels $p_i$ in $l$, mark it and remove it from $R$. Then, repeat the markAround process by setting the parameter $p$ with $p_i$. markAround.

Line 9: Find the new approximating function $F_n'$ by using all the marked pixels obtained by markAround.

Line 10: Calculate the new mean square error $e_n'$.

Lines 11–12: If $\sum(e_n - e_n') \leq 0$, replace $F(x)$ to $F(x)'$ and repeat the subprocess regionSegmentationInner.

Lines 14–16: Otherwise, attach a new label to all the marked pixels.

Lines 17–20: For all the remaining pixels in the region, further evaluate the error of mean square $e_n$. If it is smaller than the preset threshold $\theta_2$ (see Sec. 3.5), the algorithm stops. Otherwise, repeat the regionSegmentation algorithm on the remaining region.

By this algorithm, a gradation pattern is recognized as one region. Although grading tones are common in many full color images, more complex patterns may also appear there. In such cases (i.e., in the cases where approximation error in line 2 of Fig. 9 is too large), we do not try to encode this region by the proposed method (in Fig. 9, this judgment is omitted for simplicity).

3.7. Image description and reproduction

So far, an image is divided into multiple gradation patterns. Furthermore, what makes this algorithm novel is that we can simultaneously obtain the approximating
functions; by using $F(p)$ obtained by the aforementioned algorithms, we can reproduce the tones of colors within the regions.

To reproduce an image, we also record how to reproduce the contours of the gradation regions. For this purpose, we apply the contour approximation method to the gradation patterns; we apply the Sugiyama’s contour tracing algorithm, and function-approximate the extracted contours by using piecewise Fluency functions.

4. Experimental Results

In this section, we compare the quality of enlarged images when the conventional methods and the proposed method are used. We further compare their data size and processing costs.

The program for function approximation is written in Java 1.4. The operating system used for the experiment is Windows 2000, running on Intel Pentium III 700 MHz CPU, with 382 MB main memory. Recall that the proposed algorithms require four preset parameters: $\theta_1$, $\theta_2$, $\theta_3$, and $m$. For the experiment, we set each of them as 0.05, 20, 25, and 20, respectively.

4.1. Evaluating the quality of enlarged images

At first, we compare the proposed method and conventional methods in quality. In this experiment, to measure how the proposed method accurately reproduce the original image, a multi-resolution image in EPS format is prepared (Fig. 2). This EPS image is artificially created by using some drawing software and contains simple gradations. The experiment proceeds as follows:

(1) The EPS image is rasterized.

(2) To compare the conventional function-approximation methods and our approach, the rasterized image (BMP format) is re-function-approximated in each method. Each re-approximated image is then scaled to the same size. The original EPS image is also scaled. Finally, each re-approximated image is compared with the original EPS image in each scale, to see how the re-approximated image has deteriorated.

(3) To compare the interpolation methods and our approach, the rasterized image (BMP format) is re-function-approximated in the proposed method. The re-approximated image is then scaled, and the BMP image is scaled by the interpolation method. The original EPS image is also scaled to the same size. Finally, the re-approximated image and the interpolated image is compared with the original EPS image in each scale, to see how the re-approximated image and the interpolated image is deteriorated.

As the conventional function-approximation methods, we use the Fluency function-approximation method explained in Sec. 2 and the AutoTrace tool that converts a bit-map image into an EPS image. As the interpolation method, we use the bicubic algorithm that is implemented in most common DTP software.
For quantitative evaluation, we use the peak signal-to-noise ratio (PSNR), which is calculated by the equation

$$\text{PSNR} = 10 \log_{10} \frac{255^2 |X|}{\sum_{p \in X} (I_n(p) - I_n'(p))^2}.$$  \hspace{1cm} (7)

Table 2 lists the evaluation results. We then list the reconstructed images in Fig. 10 to compare each method in appearance.

The conventional contour approximation method cannot recognize gradation patterns; therefore, numerous small regions and contours are extracted, which makes approximation error visible on the scaled image. The low PSNR values also indicate the processed images contain much noise. On the other hand, the proposed method can recognize a gradation pattern as one region; therefore, only small number of contours is used for function approximation. This enables smooth function approximation. By Fig. 10, we can see that the interpolation approach also fails to reproduce smooth edges. Interestingly, although the difference between the interpolation approach and our approach is apparent in Fig. 10, the PSNR measurement does not reflect this fact. The reason we consider is that PSNR cannot distinguish human-sensible noise and human-insensible (allowable) noise. This result indicates that there is a room for argument whether PSNR is an adequate measure for evaluating quality of images or not. Our position is the highest PSNR value does not always indicate the highest quality of images, especially when it is applied to the contour outlined images.

### 4.2. Evaluating the efficiency of image description and reproduction

Then, we show how the application of proposed image segmentation enhances the encoding/decoding cost. Figure 11 shows the result of image segmentation algorithm explained in Sec. 3. This figure shows that the image segmentation algorithm can adequately recognize the regions bounded by borderlines that are recognized as “contours” by human eyes. The result of contour approximation after the segmentation is shown in Fig. 12. Compared with the results of function approximation of contours of color regions shown in Fig. 3 (in Sec. 2), the number of detected contours becomes much smaller. Table 3 reflects this fact; it lists encoding
Fig. 10. Comparison of the images scaled to 5 times.

(a) bicubic.  (b) B-spline.
(c) Fluency (only contours).  (d) AutoTrace.
(e) The original EPS.  (f) The proposed method.

We can see that, by reducing the number of extracted regions and contours, the encoding time is reduced to 0.049% in the proposed method, and the decoding time is also significantly reduced.
Fig. 11. Regions recognized by the proposed method. It shows that the image segmentation algorithm can adequately recognize the regions bounded by borderlines that are recognized as “contours” by human eyes.

Fig. 12. Result of contour approximation (Blue parts represent lines, red parts represent arcs, green parts represent quadratic curves, and black crosses represent joint points. By comparing it with Fig. 4, we can see that gradation regions are appropriately recognized).

Table 3. Processing time, total number of contours, and total number of regions (the same contour tracing and approximation algorithm is used in each method).

<table>
<thead>
<tr>
<th></th>
<th>Color regions</th>
<th>Gradation regions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coding time (sec)</td>
<td>58377.610 sec</td>
<td>0.265 sec</td>
</tr>
<tr>
<td>Contour tracing</td>
<td>6.724 sec</td>
<td>1.797 sec</td>
</tr>
<tr>
<td>Total time (sec)</td>
<td>58844.344 sec</td>
<td>28.765 sec</td>
</tr>
<tr>
<td>Decoding time (sec)</td>
<td>3.619 sec</td>
<td>0.125 sec</td>
</tr>
<tr>
<td>Total number of contours</td>
<td>6850</td>
<td>19</td>
</tr>
<tr>
<td>Total number of regions</td>
<td>2933</td>
<td>18</td>
</tr>
</tbody>
</table>

Finally, we compare the proposed method and the conventional contour approximation method in data size, which is shown in Table 4. By the table, we can see that the file size is reduced to 2.16% in the proposed method.

Therefore, the experimental results show that the proposed method enables high quality resolution conversion with small amount of data and encoding/decoding cost.
5. Conclusions

In this paper, a new function approximation method that is applicable for gradation images is proposed. In particular, a new image segmentation algorithm that can recognize a gradation pattern as one region is proposed, which solves the problems in the conventional contour approximation method. Simultaneously, through the image segmentation algorithm, 2-variable functions that approximate tones of colors are also obtained. The experimental results show that the proposed method enables high quality resolution conversion with small amount of data and processing costs.

This work is the first attempt for applying function-approximation method to gray scale or full color images that contain numerous tiny color regions. The experiments are performed on artificial images. The next attempt will be tailoring this method to process more “natural” gradations (e.g., gradations contained in the pictures taken by digital cameras), which remains in the future work.

Acknowledgments

This work was funded by the National Institute of Information and Communications Technology.

References


Kentaro Miyamoto received his ME from Graduate School of Systems and Information Engineering, University of Tsukuba in 2006. His major research interests were in signal processing and adaptive information processing methods. Currently, he works at Fujifilm cooperation.
Tetsuo Kamina received his BA from Division of Natural Sciences, International Christian University in 1999, and his MA and PhD from Graduate School of Arts and Sciences, the University of Tokyo in 2002 and 2005, respectively. He belonged to Tsukuba Advanced Research Alliance, University of Tsukuba as a postdoctoral researcher from April 2005 to March 2006. In 2006, he joined Information Media Center, Graduate School of Humanities and Sociology, the University of Tokyo. His research interests include design and implementation of object-oriented programming languages.

He is a member of ACM, IPSJ, and JSSST.

Tetsuo Sugiyama received his PhD in engineering from Tsukuba University, Japan, in 2005. In 2004, he established SoftEther corporation that develops VPN software. Since then, he has been worked around software development in network security field.

Keisuke Kameyama received the BE, ME, and Dr. Eng. degrees in electrical engineering (1989), precision machinery engineering (1991), and computer science (1999), respectively, all from Tokyo Institute of Technology, Tokyo, Japan. From 1992 to 2000 he worked as a Research Associate at Tokyo Institute of Technology. In 2000, he joined the Tsukuba Advanced Research Alliance (TARA), University of Tsukuba. Since 2007, he works as an Associate Professor at the Graduate School of Science and Information Engineering, University of Tsukuba. His research interests include pattern recognition, signal processing, and adaptive information processing methods.

Dr. Kameyama is a member of IEEE and IEICE.

Kazuo Toraichi received the PhD degree in electronics engineering from Hokkaido University, Japan, in 1971. From 1971 to 1980, he worked in the Electro Technical Laboratory of MITI. In 1980, he became an Associate Professor at the Institute of Information Sciences and Electronics, University of Tsukuba, Japan, and in 1997, he became a Multimedia Aspect Professor in the Center for Tsukuba Advanced Research Alliance, University of Tsukuba and Distinguished Professor. He is currently a
Emeritus Professor and Executive Research Director of Fluency Lab, Inc. His main fields of research are wisdom systems and Fluency information theory. He was the recipient of the new generation network project research sponsored by National Institute of Information and Communications Technology. In 2004, he received the fourth SINKOU award of the FUNAI Foundation for Information Technology for his research on the Fluency information theory and its applications to multimedia. In 2005, he was awarded the 30th INOUE HARUSHIGE prize. In 2008, he was awarded the Commendation for Science and Technology by the Minister of Education, Culture, Sports, Science and Technology.

Yasuhiro Ohmiya received the ME in risk engineering from the University of Tsukuba, Japan, in 2004. He has worked in Tsukuba Advanced Research Alliance, University of Tsukuba from 2004 as a researcher. His main fields of research are Fluency information theory based signal processing and its applications to multimedia systems.