Multimed Tools Appl
DOI 10.1007/s11042-016-4175-7

Automatic photomosaic algorithm through adaptive tiling and block matching

Hae-Yeoun Lee[1]

Received: 22 February 2016 / Revised: 18 October 2016 / Accepted: 18 November 2016

Abstract Mosaic refers to a technique that forms images by gathering several small materials in various colors. Because of the recent developments in digital imaging technology, photomosaic technologies that form images using photos are utilized. In this paper, we present an automatic computer algorithm that forms photomosaic images through adaptive tiling and block matching. A photo database stores photos called tiles for mosaic and their feature values. A photomosaic image is generated through separating the input image into blocks of a pre-set size, performing adaptive tiling by comparing the similarity between adjacent blocks, conducting block matching to find a similar tile in the photo database, and adjusting the intensity to enhance tile similarity within each image block. Furthermore, the quality of the photomosaic image is improved by minimizing tile redundancy in adjacent blocks and limiting the number of tiles used. In our experiment, the performance of the proposed algorithm is compared with that of Andrea mosaic, Mosaically, and Mozaika software packages. The results indicate that the proposed algorithm is excellent in quantitative and qualitative analyses.

Keywords Photomosaic · Adaptive tiling · Block matching · Euclidian distance

1 Introduction

As examples of decorative art, mosaics form figures or images by attaching small pieces or tiles in various colors to the basic draft [13, 24]. The human visual system perceives salient objects in an image [10]. Therefore, by viewing mosaic image from a short distance, we can see the small pieces that compose it, and by viewing it from a long distance, we can see one large form created by the pieces. Mosaic work has been mostly created by artists; however, it is currently considered an example of an unrealistic rendering technique by computer graphic

✉ Hae-Yeoun Lee
haeyeoun.lee@kumoh.ac.kr; hytoiy@gmail.com

[1] Department of Computer Software Engineering, Kumoh National Institute of Technology, 1 Yangho, Gumi, Gyeongbuk, Republic of Korea

researchers. The images created by this mosaic technique have enhanced the interest and artistic property of images.

In the past, many materials needed to be collected for a long time in order to create a mosaic. However, this problem has been resolved considerably using the wide variety of low-priced digital devices that have been launched recently. Thanks to the appearance of software packages that can be easily employed by general users, they are much more interested in mosaic management. Thus, unrealistic rendering techniques have recently become the target of embodiment for many graphic designers, rather than the technology of some researchers, and mosaics of diverse types have been introduced [8]. Figures 1(a) and (b) show the results of using the mosaic filter provided by Photoshop (Adobe Co.). Figure 1(c) shows the mosaic created with the Voronoi Diagram proposed by Haeberli [8], and Fig. 1(d) shows the mosaic proposed by Blashi and Gallo [4]. Figure 1(e) shows Hausner's [9] mosaic formed using the centroid Voronoi Diagram and edge technique. Figure 1(f) shows the photomosaic proposed by Silvers [21].

This paper studies the photomosaic algorithm that uses photos among diverse mosaic methods. "Photomosaic" refers to forming one large image by collecting many small photos. Because of the recent development of digital imaging technology, photomosaic technologies are actively used in magazines, posters, music videos, and TV advertisements [2, 3]. Because the works of mosaic images are mostly composed by specialized graphic designers manually, it is difficult for general users to create photomosaic images. Even specialized graphic designers can require a significant time to compose mosaic images because the work is executed manually. Although software packages that automate photomosaic algorithms have been developed, no program has been commercialized yet (to the best of our knowledge), and their function is not satisfactory.

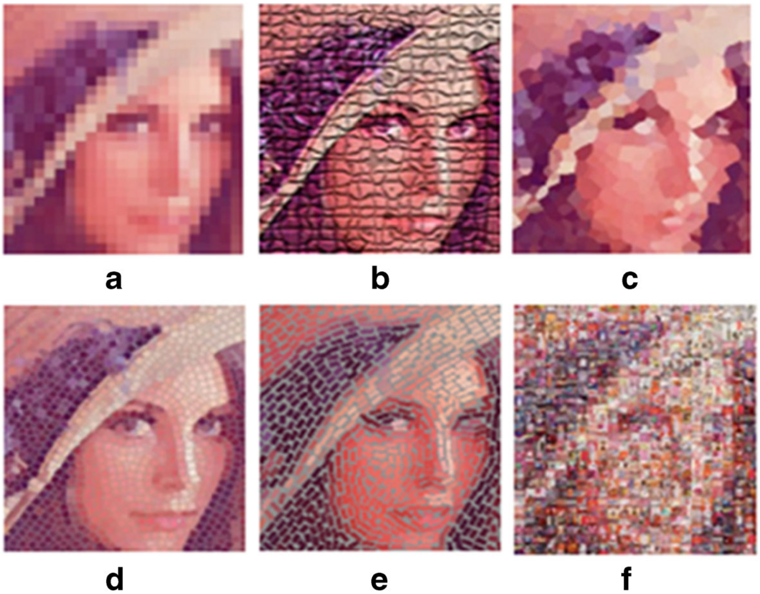


Fig. 1 Mosaics made by diverse technologies: **a** and **b** mosaic filter by Photoshop, **c** mosaic created with Voronoi Diagram, **d** mosaic proposed by Blashi and Gallo, **e**mosaic formed using centroid Voronoi Diagram and edge technique, and **f** photomosaic proposed by Silvers

This paper proposes a computer algorithm that forms automatic photomosaic images. For the high quality of photomosaic images, novel techniques such as adaptive tiling, block matching with tile redundancy minimization, and tile intensity adjustment methods are developed. The proposed algorithm is composed of two steps: one creates the photo database, and the other forms the photomosaic image. In the step that creates the photo database, the feature value of the photo (called a tile) used in the mosaic is extracted and stored. In the step that forms the photomosaic image, an input image is divided into block units in order to extract its features, and the similarity between the adjacent blocks is compared in order to perform adaptive tiling. Then, the blocks created with adaptive tiling are compared with the tiles from the photo database in order to perform block matching to find a similar tile. The tile value of light and darkness matched last is replaced by that of the relevant block in order to enhance the image similarity. Moreover, the quality of the mosaic image is improved by applying a technique that minimizes tile redundancy between adjacent blocks. The proposed algorithm is compared with the Andrea mosaic, Mosaically, and Mozaika software packages to show the performance excellence in quantitative and qualitative analyses.

The rest of this paper is composed as follows. Technology related to the mosaic technique is summarized in Section 2, and the technology to create photomosaic images by adaptive tiling and block matching is proposed in Section 3. The results of the experiment are presented in Section 4, and the conclusion is summarized in Section 5.

2 Related works

Although the photomosaic technique was introduced by Silvers [21] and automatic programs have been studied by some developers, the technique's speed or quality has not yet been improved satisfactorily (to the best of our knowledge). Furthermore, those photomosaic images that are currently commercialized are mostly created manually.

Klein et al. [12] proposed a video mosaic algorithm. For a moving image, matching between blocks is performed by comparing block similarity using the color average value feature in the YIQ domain and three-dimensional (3D) wavelet disassembly. Figure 2 shows the results of color matching in the YIQ domain, and Fig. 3 shows the mosaic formed by combining an input and matched image through overlay in the color adjustment process. If the image is combined by overlay, as shown in Fig. 3, the overall influence is natural compared with simple color matching, but the quality of each tile image is rapidly lowered and the image content is perverted. Although there is some difference based on computer performance, a total of 3 h and 47 min is required to produce a moving image that lasts 10 s. Thus, this method cannot produce a detailed image for the tiles that compose a photomosaic image—it simply delivers the mosaic effect. To the best of our knowledge, there is no technology that can improve quality in consideration of eliminating redundant tile images and tiles of diverse sizes.

By proposing a fast photomosaic algorithm, Blasi and Petralia [5] improved speed compared with the existing photomosaic algorithm. Color matching was applied after dividing the input image into 3 × 3 blocks for the RGB domain. In particular, Blasi and Petralia attempted to improve performance by Antipole clustering. Although no specific algorithm was presented and no experiment was conducted to compare their results with existing studies, Blasi and Petralia required 5.98 s to form a photomosaic of size 320 × 240, and 32.487 s to form a mosaic image of size 1024 × 768 after considering the differences in computer performance. Furthermore, discerning the image of the tiles used in the produced result, as shown in Fig. 4,

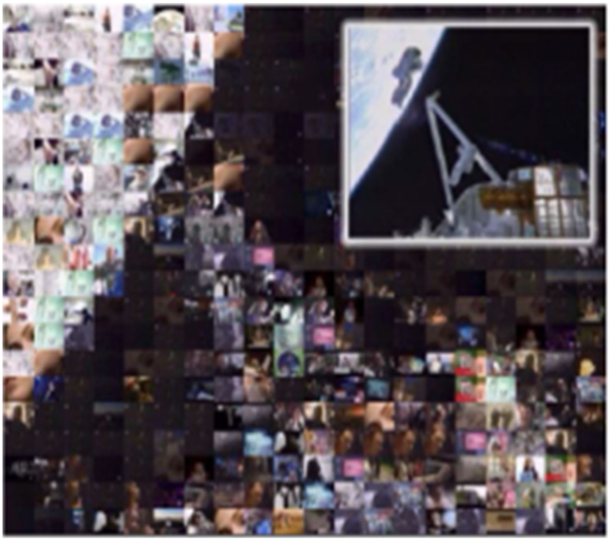


Fig. 2 Results after color matching

is difficult, and the photomosaic quality is unsatisfactory because no color adjustment was applied to the tiles.

Seo and Kang [20] proposed an algorithm for composing photomosaic images using the social network context. After dividing the input image into blocks of a fixed size, a similar photo was searched and its intensity averaged mathematically with that of the input image in order to form the mosaic image. Kang et al. [11] presented a stackable mosaic generation algorithm for mobile devices. Because the photomosaic algorithm with a large database requires high computational costs, they customized several tile images by rotation based on the content of the input image, and these tiles were averaged mathematically with the input image in order to form the mosaic images. Markus et al. [16] presented a mosaic image


Fig. 3 Results after color correction

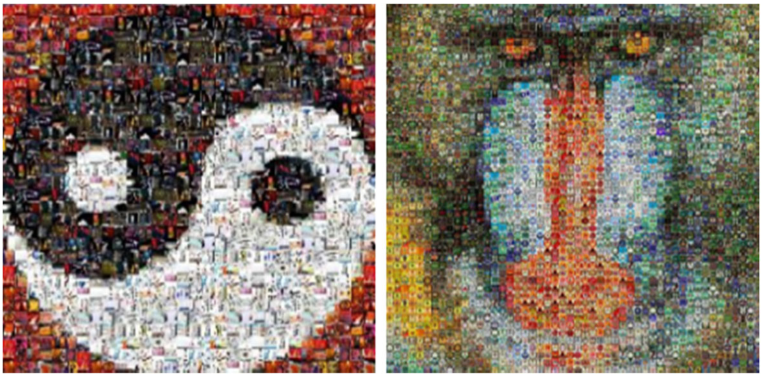


Fig. 4 Photomosaic image by fast photomosaic [5]

generation algorithm using ASCII art. Photomosaic images with a fixed block size were generated using a decision tree with enhanced performance speed.

After dividing the input image into blocks, Lama et al. [15] generated a mosaic image by filling blocks with the same color. During mosaicking, they presented a method for hiding secret images into the mosaic images. Lai and Tsai [14] presented a method for generating a secret-fragment-visible mosaic image where a secret image is divided into small blocks, and the blocks are used as photos for the mosaics. There have several studies for using photomosaic images for secret communication [6, 22].

Plant et al. [19] addressed a method for measuring the effectiveness of color-based image retrieval systems in forming photomosaic images manually. Because the relevance of the retrieved images and the reflection-in-action for creative design were difficult to evaluate in previous methods, they presented an average pixel-to-pixel distance for a mosaic test.

3 Proposed photomosaic algorithm

The proposed photomosaic algorithm is composed of two steps: one creates the photo database, and the other forms the photomosaic image. The photo database is composed of the photos (tiles) used to form the mosaic image. In the step for creating the photo database, a tile is divided into 4 × 4 blocks in order to calculate the average RGB value in each area and store it as the feature value. Namely, 48 feature values represent the features of the intensity value of the relevant tile. In the step for composing the photo mosaic, the photo database is used as the input image for forming the mosaic image, and the step is composed of the processes for extracting the feature, conducting adaptive tiling, performing block matching, and adjusting the intensity value.

The approximate process for performing the step of composing the photomosaic image is shown in Fig. 5. (Step 1) The input image is divided into pre-set block sizes in order to extract the features. (Step 2) The similarity between adjacent blocks is compared and combined for adaptive tiling. (Step 3) The similarity of the blocks made by adaptive tiling is compared with that of the tiles in the photo database in order to perform block matching that finds a similar tile. Furthermore, the process for minimizing tile redundancy between adjacent blocks and reducing the use of repetitious tiles is performed. (Step 4) The value of light and darkness of the matched tile is replaced with that of the relevant block in order to adjust the intensity value

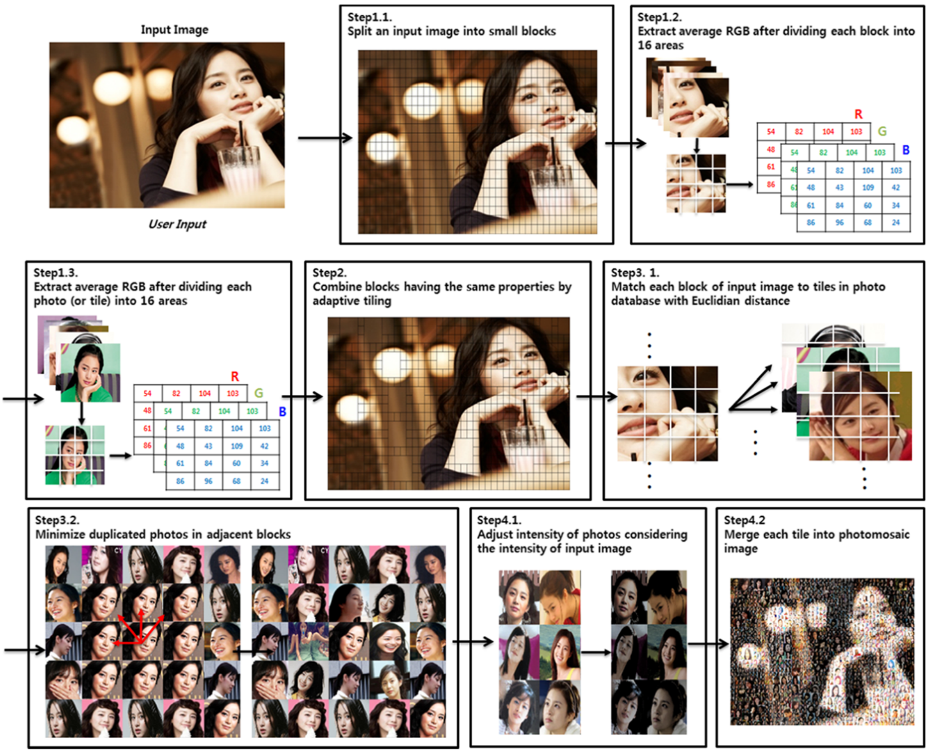


Fig. 5 Process for forming photomosaic through adaptive tiling and block matching

that heightens image similarity and create the photomosaic image. Each process is explained in detail in the following subsections.

3.1 (step 1) image feature extraction

(Step 1.1) The first process for composing photomosaic images is to divide the original (input) image into small blocks in order to calculate the feature of each block. The features thus calculated are utilized in the processes for adaptive tiling and block matching.

(Step 1.2) As shown in Fig. 6, the input image is divided into blocks of pre-set size; each block is then divided into 16 areas (4 × 4) to calculate the average RGB value of each area for use as the feature of each block. Thus, 48 feature values can be extracted for each block. (Step 1.3) The process for calculating such feature value is similar to that for extracting the feature of the tile (or photo) included in the photo database (the block size was set to 64 pixel units in our experiment). The number of feature values obtained is greater than that obtained by Blasi and Petralia's method [5].

3.2 (step 2) adaptive tiling

If tiles of diverse size, instead of those of the same size, are included in the photomosaic image, the details of the photo used as tile can be expressed well, improving the quality of the photomosaic image. Thus, if the feature value is similar between adjacent blocks for the block

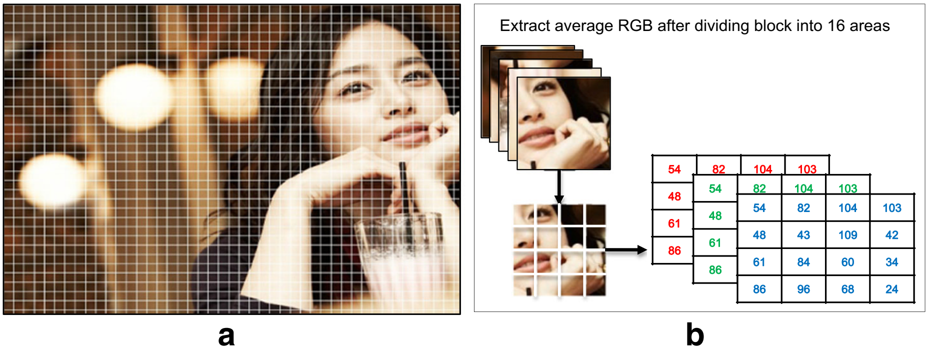


Fig. 6 Image: **a** divided into 64 × 64 pixel blocks, and **b** average value extracted by dividing block into 16 segments

made in the adaptive tiling process (Step 1), the block size can be enlarged by combining said blocks.

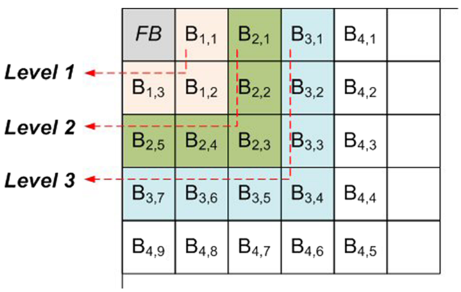
If the blocks are divided as shown in Fig. 7, block *FB* represents the fixed block used as the target of comparison for combining the adjacent blocks, and block $B_{i,j}$ represents the *j*th block that pertains to level *I* on the basis of block *FB*.

(Step 2.1) By searching in the direction from top left to bottom right, the fixed block *FB* that is not combined is selected. In order to determine whether a block has been combined, a two-dimensional (2D) array whose size is equal to the number of blocks in the X and Y-axes is used.

(Step 2.2) The combining process is applied using the fixed block *FB*. If the error between the feature values of all block $B_{i,j}$ values that pertain to fixed block *FB* and level *I* for adaptive tiling is ±10 value, the content of the block included in level *I* is determined to be similar for performing the combination. If any block in level *I* is beyond the error range, the blocks in level *I* are not combined and the adaptive tiling process for fixed block *FB* stops. After the combining process, the process to search the next fixed block *FB* (Step 2.1) is repeated until all blocks are combined.

For instance, if the similarity is within ±10 value when comparing block *FB* with blocks $B_{1,1}$, $B_{1,2}$, and $B_{1,3}$ in level 1, the tile can be combined and the blocks in level 2, i.e., $B_{2,1}$, $B_{2,2}$, $B_{2,3}$, $B_{2,4}$, and $B_{2,5}$, are compared constantly for similarity. If any of the blocks in level 2 are beyond the error range, the blocks in level 2 are not combined. The size of block *FB* is determined by the size that is expanded twice to the X and Y-axes. When the level increases, the size of the combined block is expanded. However, the number of levels is limited because the quality of the mosaic image is reduced if the block is too large.

Fig. 7 Correlation between fixed block (*FB*) and peripheral blocks ($B_{i,j}$) for comparison



In order to compare block similarity, the value difference between the feature values of the block is calculated and used according to the following formula, where i is the block level, j is the block number in the level, and k represents the feature value.

$$\text{Difference} = \sum_{k=1}^{48} \left(FB(\mathrm{k}) - B_{i,j}(k)\right)^2$$

Figure 8 shows the expanded shape of the block after adaptive tiling. The block marked by "1" represents a fixed block, and the block marked by "0" shows the shape of the block expanded based on the fixed block. By utilizing the "0" and "1" properties of the block as explained above, the block marked by "0" in the adaptive tiling process represents a combined block and such block is not used as a fixed block in the process of adaptive tiling. For the block marked by "1", tile expansion is attempted and the process of block matching in the next step is facilitated.

3.3 (step 3) block matching with redundancy minimization

Block matching is the process for finding the best photo (or tile) in the photo database within each divided block of the input image. If the same photo is inserted into adjacent blocks, the visual quality is lowered. Thus, minimizing tile redundancy is a process for preventing the insertion of the same image in the adjacent block.

In the block matching process, performance speed is improved by calculating the Euclid distance between two dots based on measuring the similarity between blocks in order to eliminate/utilize root calculation. The image combined by block matching pertains to the image with the lowest similarity value among those obtained from the following formula, where $FB(k)$ represents the kth feature value of each divided block, and $DB_{\mathrm{i}}(k)$ represents the kth feature value of photo i in the photo database.

$$\text{Block}_{\text{match}h} = min\left\{ \sum_{k=1}^{48} \left(FB(k) - DB_{\mathrm{i}}(k)\right)^2 \right\}$$

When composing a photomosaic image, quality is lowered by the redundancy of the same tile and using a tile of fixed size. Therefore, if a tile with the best value already pertains to the peripheral block, the tile with the other best value is searched by combining the image for use in the process for minimizing tile redundancy. Furthermore, repetitious use is limited by measuring the times that the tile from the photo database is used.

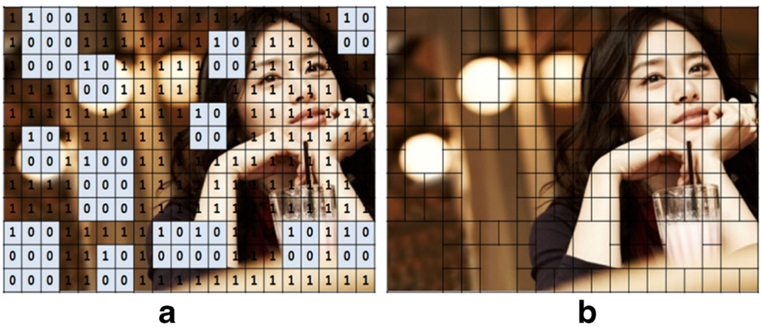


Fig. 8 Method for creating an adaptive tile: **a** marked blocks, and **b** blocks by adaptive tiling

(Step 3.1) For blocks, the block matching process starts at the top left area and ends at the bottom right area. Thus, when block matching is performed for random blocks, only those blocks from the top left are matched.

(Step 3.2) To test tile redundancy for the proposed algorithm, the peripheral block included in the range of 45–180 degrees is searched based on the block for matching, as shown in Fig. 9. The search depth is set to three. Given that the block for matching is $B_{i,j}$ and the peripheral block is $B_{k,l}$, the *max*(|i-k|, |j-l|) value, namely, the maximum value of the distance difference, represents the search depth. If the search depth is low, a similar photo is quite likely to appear; if the search depth is deep, performance time is lengthened because many blocks must be compared in the block matching process.

(Step 3.3) As adaptive tiling is performed, if the peripheral block for comparison is an expanded block that has already been combined, the information on the tile is allocated to fixed block *FB* that represents the used relevant block.

The results obtained by applying block matching and the technique for minimizing redundancy are shown in Fig. 10. The results obtained before eliminating redundancy are shown in the left image, and those obtained by minimizing redundancy are shown in the right, which imply the improved quality of the qualitative mosaic image in Fig. 10(b).

3.4 (step 4) adjustment of intensity value and image combination

If an image set to the optimum value in the photo database by block matching is used as is to compose the photomosaic image, the color sense is quite different from the input image, as shown in Fig. 11.

(Step 4.1) In the intensity value adjusting process, the quality is improved by adjusting the tile set in the photo database so that it can be similar to the intensity value of the input image block. To this end, the tile set and input image block are changed to the HSI domain, and the intensity value data of the tile set are replaced by the intensity value data of the input image block for conversion into the RGB domain image; this way, the intensity value of the tile set can be adjusted. (Step 4.2) Finally, the input image block is replaced as the adjusted tile set to get the photomosaic image. As shown in Fig. 11, the quality is improved at the boundary part compared

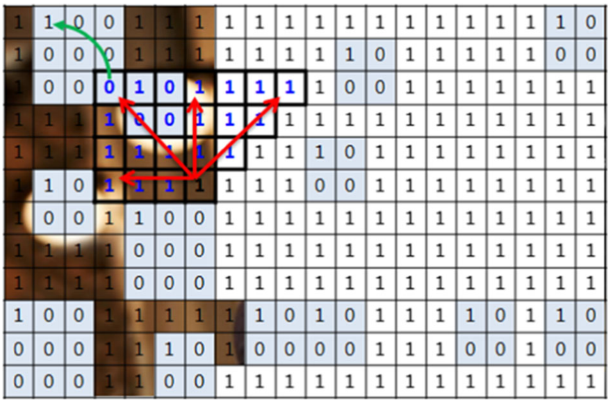


Fig. 9 Peripheral blocks for comparison in minimizing redundancy when depth is three (*blue*)

Fig. 10 Example of minimizing redundant image: **a** before minimization, and **b** after minimization

with that before adjustment. Furthermore, a natural image can be made through this process without overlapping the input image.

Vivid photomosaic images are shown in Figs. 12, 13, and 14 through the process of adaptive tiling, block matching, and intensity value adjustment. As shown in the figures, quality improvements can be confirmed by adjusting the intensity value of the tile in diverse sizes. In particular, the overall outline and content of the photos used as tiles are better expressed by increasing the tile size, and thus quality is more satisfactory. (Figs. 12, 13, and 14 are included in Section 4 along with a detailed explanation of each figure).

Fig. 11 Example of replacing light and darkness tile value: **a** before replacement, and **b** after replacement

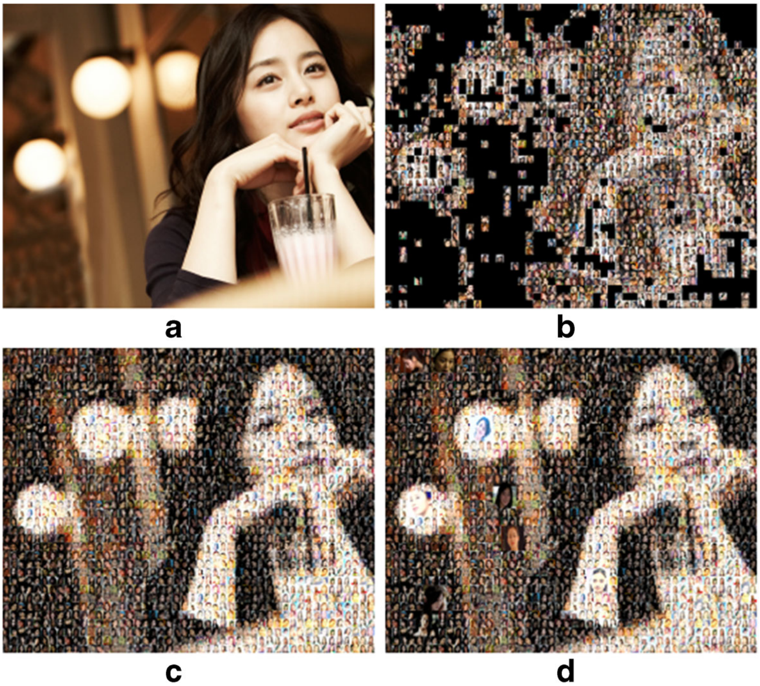


Fig. 12 Photomosaic results when number of redundant tiles is limited to five: **a** original image, **b** Andrea mosaic, **c** mosaic without adaptive tiling, and **d** mosaic with adaptive tiling

4 Experiment results

In order to rate the performance of the proposed photomosaic algorithm, we conducted a comparative analysis of the photomosaic results of our algorithm with those of Andrea mosaic [1], Mosaically [17], and Mozaika [18] software packages. Andrea mosaic software is free offline software with customizable features such as pixel width, number of tiles per row, tile spacing and duplication, and color changes. Mosaically is free online software with simple features such as tile size and colorization. Mozaika is offline shareware with customizable features such as pixel width, tile spacing and duplication, and color correction. The algorithms were tested under the same conditions in the environment of Intel(R) Core i5 650 processor, nVidia GeForce 310, and memory 3 GB DDR3 computers.

The proposed photomosaic algorithm was analyzed for algorithm division with or without adaptive tiling, and the number of tiles used was limited to 266. Table 1 lists the results of the ten images that were arranged for the experiment. Because the size of the photomosaic images composed by the proposed algorithm and the Andrea mosaic software is different, an intuitive comparison is difficult. We could confirm that although the memory use of the proposed algorithm is high, the CPU share is low and the performance is faster when adaptive tiling is not applied, compared with the Andrea mosaic. In addition, because the proposed algorithm with adaptive tiling determines the blocks to use for combining, it requires higher computation costs than the proposed algorithm without adaptive tiling. We believe that memory usage, CPU share, and performance time are not critical in comparison with the quality of the photomosaic

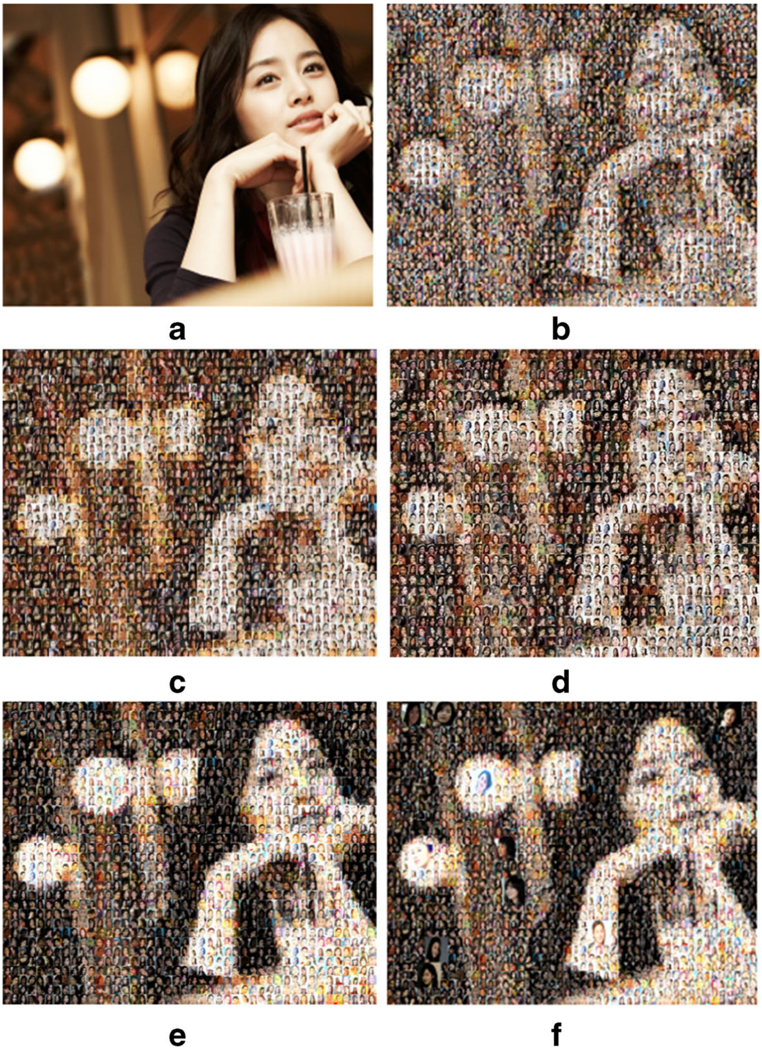


Fig. 13 Photomosaic results when number of redundant tiles is limited to ten for the proposed algorithm and Andrea mosaic software: **a** original image, **b** Andrea mosaic software, **c** Mosaically software, **d** Mozaika software, **e** mosaic without adaptive tiling, and **f** mosaic with adaptive tiling

image. When frameworks that employ many-core processors are applied for composing photomosaic images [25–27], such differences can be ignored. However, partial retardation of performance can be confirmed when adaptive tiling is introduced to improve the quality of the mosaic image, even when fewer tiles are used.

Figure 12 shows the resulting photomosaic when the number of redundant tiles is limited to five. Figures 13 and 14 show the resulting photomosaic when the number of redundant tiles is limited to ten for the proposed algorithm and Andrea mosaic software. The results of the Mosaically software, which has no option of controlling tile redundancy, were generated by setting 0% colorization. After the distance between the same images was set to six, the results of Mozaika were generated with normal rendering.

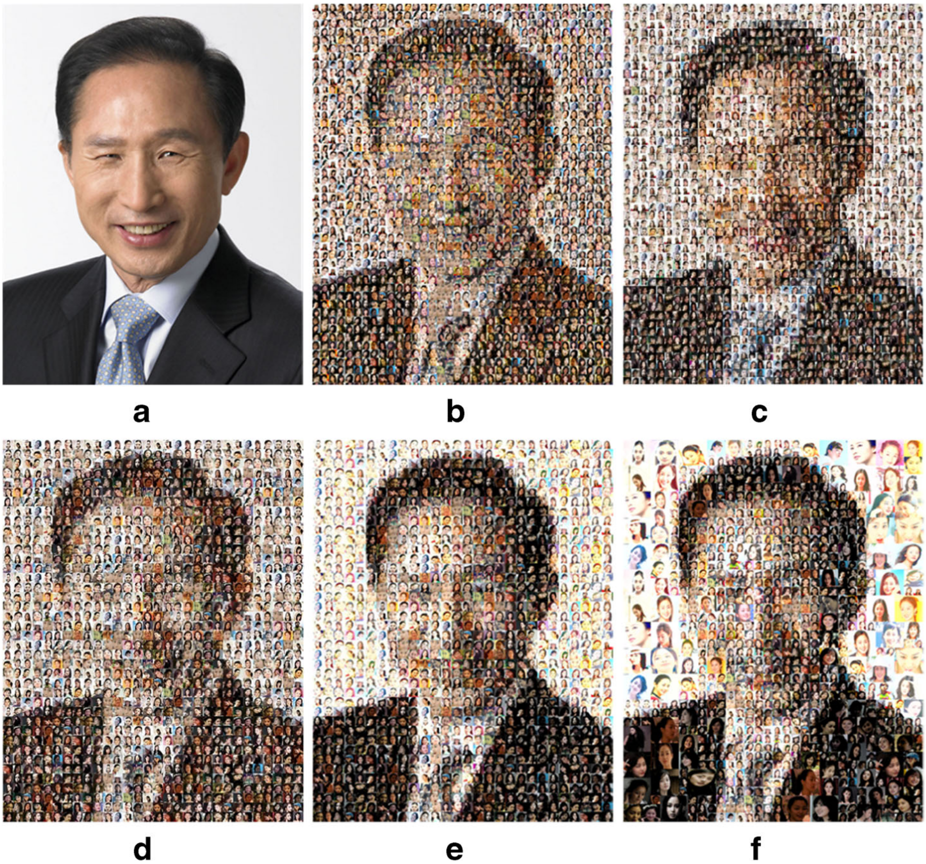


Fig. 14 Photomosaic results when number of redundant tiles is limited to ten for the proposed algorithm and Andrea mosaic software: **a** original image, **b** Andrea mosaic software, **c** Mosaically software, **d** Mozaika software, **e** mosaic without adaptive tiling, and **f** mosaic with adaptive tiling

The technology for avoiding tile redundancy between adjacent blocks is introduced in the proposed algorithm in order to minimize redundant tile use, thereby making it possible to employ the tiles from the photo database to the maximum extent. In the case of Andrea mosaic,

Table 1 Results of analyzing average algorithm performance for ten images

	Proposed algorithm without adaptive tiling	Proposed algorithm with adaptive tiling	Andrea mosaic software
Size (pixel)	3072 × 2304	3072 × 2304	4281 × 3425
Number of tile (pcs)	1728	1353 ~ 1728	2068
Tile size (pixel)	64 × 64	64 × 64	98 × 73
Memory capacity	400 MB	400 MB	100 MB
Permitted # of redundant images	5 or 10 pcs	5 or 10 pcs	5 or 10 pcs
# of images in use	266 pcs	266 pcs	266 pcs
Use rate of CPU	30%	30%	80–100%
Performance time	4.3–5.2 s	9.7–11.4 s	7–8 s

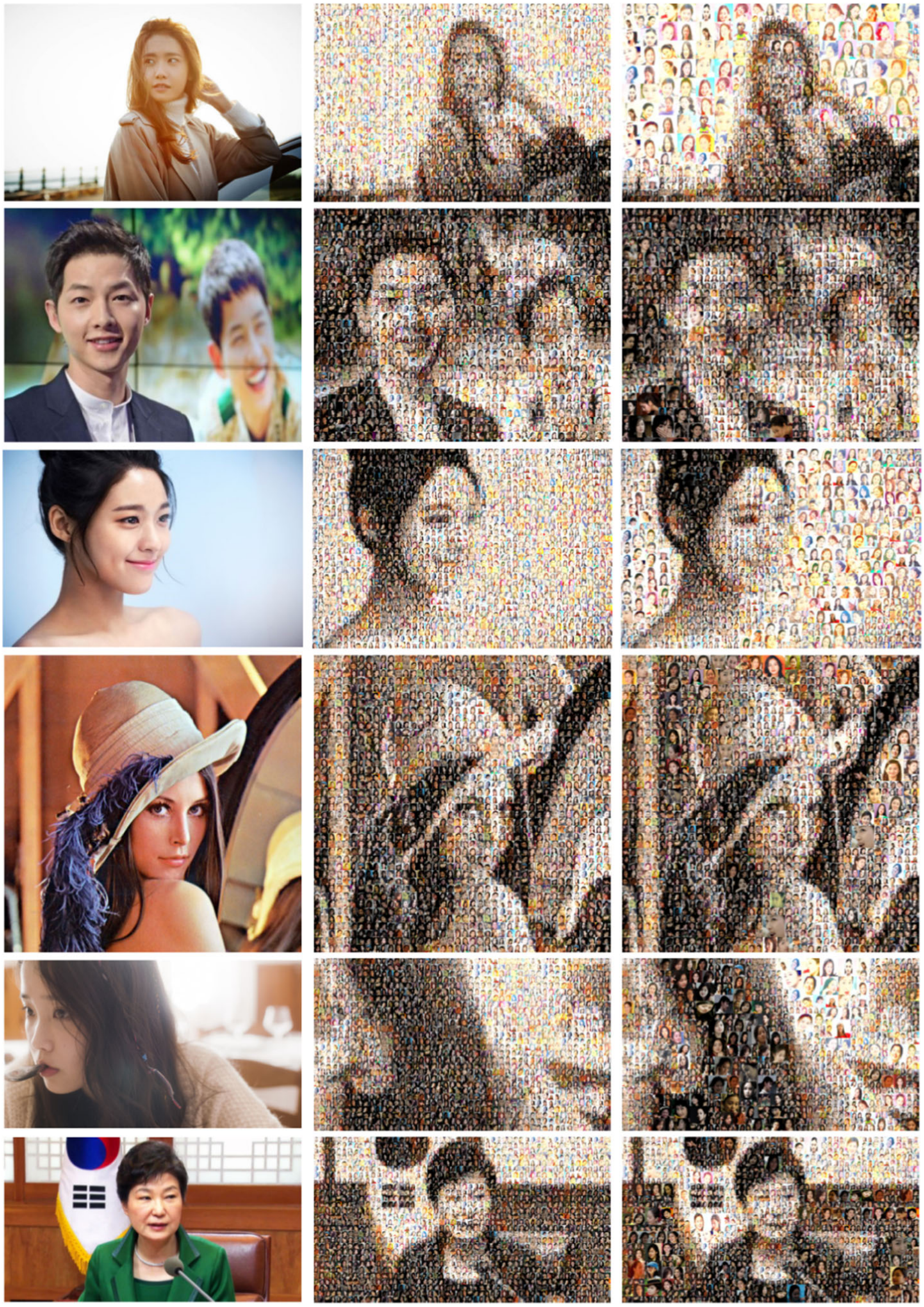


Fig. 15 Photomosaic results when number of redundant tiles is limited to ten for the proposed algorithm (labeled by column): **a** original images, **b** mosaics without adaptive tiling, and **c** mosaics with adaptive tiling

however, the number of redundant images is limited as indicated in Fig. 12(b) to show empty space, because not all the tiles of the input image are filled. When number of redundant tiles are limited to ten, the resulting normal photomosaic is as shown in Figs. 13(b) and 14(b).

Table 2 Average PSNR values of photomosaic images from each algorithm and software

	Andrea mosaic software	Mosaically software	Mozaika software	Proposed algorithm without adaptive tiling	Proposed algorithm with adaptive tiling
PSNR	11.03 dB	12.26 dB	10.76 dB	14.68 dB	13.46 dB

However, the quality of the photomosaic image continues to be unsatisfactory. In the case of the Mosaically software, the resulting photomosaic is not satisfactory in terms of color presentation. In the case of the Mozaika software, although the photos used as tiles appear to be clear, the details of the input image are not well presented in the photomosaic image. The photomosaic image composed by the proposed algorithm is of high quality in terms of the colors of the input image, achieved by adjusting the intensity value and performing adaptive tiling. When these photomosaic results are seen from a distance, the performance of the proposed algorithm is well presented.

The use of tiles of diverse size can increase the details of the photo used as tile and improve visual quality. In addition, because a photo is used without combining the intensity of the input image pixel by pixel, the content of the photos is well preserved. However, there is a possibility of decreasing the details of the content in the input image. As shown in Fig. 14(f), although the visual quality of the photo is enhanced, the ears and suit collar are confusing.

Figure 15 depicts six photomosaic results of the proposed algorithm with and without adaptive tiling. The results are visually pleasing and show the effectiveness of the proposed algorithm.

The quality of the photomosaic image cannot be easily rated quantitatively. However, since PSNR (dB) can be a measure to analyze the differences between input and mosaic images, average PSNR values for five photomosaic images from the proposed algorithm, Andrea mosaic software, Mosaically software, Mozaika software are calculated and summarized in Table 2. Since the proposed algorithm adjusts the intensity of photomosaic images, high perceptual quality is achieved.

For the qualitative test results in terms of visual quality, each of the five photomosaic images composed by the proposed algorithm with and without adaptive tiling, Andrea mosaic software, Mosaically software, Mozaika software including the photomosaic images from Figs. 13 and 14 were presented to 200 users. Considering the quality of photomosaic and tile images and the redundancy of tiles, they were asked to select the best. These users are university students, and include men and women. The analyzed results are presented in Table 3; as can be seen, preference for the proposed photomosaic algorithm is 66% (27% + 39%), which is much higher than preference for other software packages. When adaptive tiling is used, tile size is increased. Thus, the details of the photo used as tile and the outline are better expressed to further satisfy users with quality.

Table 3 Quality of photomosaic image as rated by 200 users

	Andrea mosaic software	Mosaically software	Mozaika software	Proposed algorithm without adaptive tiling	Proposed algorithm with adaptive tiling
Preference	18 users (9%)	23 users (11.5%)	27 users (13.5%)	54 users (27%)	78 users (39%)

5 Conclusions

Because of the development of digital imaging technology, photomosaic technology that composes images using photos is currently utilized. Because most photomosaic technologies are performed manually by specialized designers, considerable time is spent, and such images cannot be created easily by the general public because of technical difficulties.

This paper proposed a computer algorithm that composes automatic photomosaic images by adaptive tiling and block matching. The tile used for the mosaic is created and utilized by a photo database, and mosaic images of high quality are created in a step for composing photomosaic images by extracting image features and conducting adaptive tiling, block matching, and intensity value adjustment processes. Furthermore, the proposed algorithm was quantitatively and qualitatively compared with the disclosed Andrea mosaic, Mosaically, and Mozaika software packages to show that there are no serious problems in terms of performance time and memory use rate of the proposed algorithm. We showed that the quality of the mosaic images composed by the proposed algorithm is excellent.

The main contribution of this work is to generate high quality photomosaic images with the novel automatic computer algorithm. For this purpose, the adaptive tiling, block matching with tile redundancy minimization, and tile intensity adjustment methods were presented. Although the performance time of the proposed algorithm includes calculating processes that require significant time, such as adaptive tiling, block matching, and intensity adjustment, there are some parts that can be optimized using many-core processors [25–27]. Since the human visual system perceives salient objects in an image, salient features can be considered for block matching to increase the performance [10]. Also, computational complexity can be minimized by detecting salient regions and focusing on these regions for the quality of photomosaic images. Because the proposed algorithm uses photos from a photo database without geometric distortions, the quality of the mosaic images can be improved by applying feature-based matching and registration before mosaicking [7, 23]. Thus, there are still some aspects that should be improved for commercialization.

Acknowledgments This work was supported by Kumoh National Institute of Technology.

References

1. Andrea mosaic software (2016) https://andreamosaic.en.softonic.com/
2. Battiato S, Blasi GD, Farinella GM, Gallo G (2006) A survey of digital mosaic techniques. In: Proc. of Eurographics Italian chapter conference, pp 129–135
3. Battiato S, Blasi GD, Farinella GM, Gallo G (2007) Digital mosaic frameworks-an overview. Computer Graphics Forum 26(4):794–812
4. Blasi GD, Gallo G (2005) Artificial Mosaic. Vis Comput 21(6):373–383
5. Blasi GD, Petralia M (2005) Fast photomosaic. In: Proc. of international conference in central Europe on computer graphics, visualization and computer vision, pp 1–2
6. Chavan AS, Manjrekar AA (2015) Data embedding technique using secret fragment visible mosaic image for covered communication. In: Proc. of IEEE international conference on information processing, pp 16–19
7. Fu Z, Wang L (2014) Optimized design of automatic image mosaic. Multimedia Tools and Applications 72(1):503–514
8. Haeberli P (1990) Paint by numbers: Abstract image representation. In: Proc. of the annual conference on computer graphics and interactive techniques, pp 207–214

9. Hausner A (2001) Simulating decorative mosaics. In: Proc. of the annual conference on computer graphics and interactive techniques, pp 573–580
10. Jian M, Lam K-M, Dong J, Shen L (2015) Visual-patch-attention-aware saliency detection. IEEE Transactions on Cybernetics 45(8):1575–1586
11. Kang D, Seo S, Ryoo S, Yoon K (2013) A study on stackable mosaic generation for mobile devices. Multimedia Tools and Applications 63(1):145–159
12. Klein AW, Grant T, Finkelstein A, Cohen MF (2002) Video mosaics. In: Proc. of international symposium on non photorealistic rendering, pp 21–28
13. Kyprianidis JE, Collomosse J, Wang T, Isenberg T (2013) State of the "art": a taxonomy of artistic stylization techniques for images and video. IEEE Trans Vis Comput Graph 19(5):866–885
14. Lai I-J, Tsai W-H (2011) Secret-fragment-visible mosaic image–a new computer art and its application to information hiding. IEEE Transactions on Information Forensics and Security 6(3):936–945
15. Lama RK, Han S-J, Kwon G-R (2014) SVD based improved secret fragment visible mosaic image generation for information hiding. Multimedia Tools and Applications 73(2):873–886
16. Markus N, Fratarcangeli M, Pandzic IS, Ahlberg J (2015) Fast rendering of image mosaics and ASCII arts. Computer Graphics Forum 34(6):251–261
17. Mosaically software (2016) http://mosaically.com
18. Mozaika software (2016) http://www.mazaika.com
19. Plant W, Lumsden J, Nabney IT (2013) The mosaic test: measuring the effectiveness of colour-based image retrieval. Multimedia Tools and Applications 64(3):695–716
20. Seo S, Kang D (2015) A photomosaic image generation method using photo annotation in a social network environment. Multimedia Tools Appl :1–11
21. Silvers R, Hawley M (1997) Photomosaics. Henry Holt and Co., Inc., New York
22. Singhavi DG, Chatur PN (2015) A New method for creation of secret-fragment visible-mosaic image for secure communication. In: Proc. of IEEE international conference on innovations in information embedded and communication systems, pp 1–5
23. Wang Z, Chen Y, Zhu Z, Zhao W (2016) An automatic panoramic image mosaic method based on graph model. Multimedia Tools and Applications 75(5):2725–2740
24. Wikipedia, Mosaic (2016) http://en.wikipedia.org/wiki/Mosaic
25. Yan C, Zhang Y, Xu Z, Dai F, Zhang J, Dai Q, Wu F (2014a) Efficient parallel framework for HEVC motion estimation on many-core processors. IEEE Transactions on Circuits and Systems for Video Technology 24(12):2077–2089
26. Yan C, Zhang Y, Xu J, Dai F, Li L, Dai Q, Wu F (2014b) A highly parallel framework for HEVC coding unit partitioning tree decision on many-core processors. IEEE Signal Processing letters 21(5):573–576
27. Yan C, Zhang Y, Dai F, Dai Q (2014c) Parallel deblocking filter for HEVC on many-core processor. Electron Lett 50(5):367–368

Hae-Yeoun Lee received his MS and PhD degrees in computer science from Korea Advanced Institute of Science and Technology (KAIST) in 1997 and 2006, respectively. From 2001 to 2006, he was with Satrec Initiative, Korea. From 2006 to 2007, he was a postdoctoral researcher at Weill Medical College, Cornell University, United States. He is now with Kumoh National Institute of Technology, Korea. His major interests are image processing, digital watermarking, digital forensics, remote sensing, and digital rights management.