BULGARIAN ACADEMY OF SCIENCES
PROBLEMS OF ENGINEERING CYBERNETICS AND ROBOTICS • 2021 • Vol. **75**, pp. 35-42
p-ISSN: 2738-7356; e-ISSN: 2738-7364
https://doi.org/10.7546/PECR.75.21.04

# GIMP Plug-in for Tiles Mosaicing

*Petar Tomov, Dimitar Parvanov, Gergana Mateeva*

*Institute of Information and Communication Technologies*
*at the Bulgarian Academy of Sciences*
*Acad. Georgi Bonchev Str., block 2, office 514, 1113 Sofia, Bulgaria*
*E-mails: petar.tomov@iict.bas.bg, dimitar.parvanov@iict.bas.bg,*
*gergana.mateeva@iict.bas.bg*

**Abstract:** As web applications have become one of the main internet services, they really need well-composed images. Except web application there are many other different field that can benefit of images improvement. Among the variety of image processing software tools, the GNU Image Manipulation Program (GIMP) is the second most popular bitmap image processing tool after Photoshop. It has a well-organized plugin-based software architecture. GIMP project file organizes the image in separate layers, which makes it perfect for different manipulations. By using layers, the full-color raster image can be vectorized and the color reduced. Such a transformation is very useful in the case of planning a Venetian mosaic, which is constructed by square tiles. In this research, a Python plugin is proposed, which transforms a full-color image into a mosaic of square tiles, accompanied by a reduction of colors. The task thus set is combinatorial in nature. A genetic algorithm has been chosen to search for optimal or suboptimal solutions.

**Keywords:** *GIMP, Vectorization, Color Reduction, Optimization, Mosaicing*.

## 1. Introduction

Color is an important attribute in different fields including web applications via user experience, modeling, aesthetic design, etc. That is why many authors deals with the problems of understanding and carefully design images with respect to their shapes and colors. For example, in web applications it is important to use

accepted different standards for colors to formatting texts, lists, tables, backgrounds etc. [1]. Others focus on a visualization method to identify the relationships between color terms and typical color values used in images on the Internet to investigate their distribution in the WCS color palette [2]. The most recent article deal with a novel feature representation method for color images with adaptive structural pyramid pooling [3]. A special attention should be done on decorating glass-ceramic floor [4]. Ceramic or glass tiles with regular (Fig. 1) or irregular shapes are used in the process of Venetian mosaic creation [5].

Fig. 1. Ceramics square tiles

In the general case, the number of available colors for the tiles is limited. The size of the tiles is relatively bigger than the pixels in a digital image, which does not give an easy way for achieving mosaic with high resolution. The limit of available colors and the size of the tiles are difficulties that can be challenged to an algorithmic solution and proposition of a practical tool for the real art industries.

Modern information technologies provide an opportunity for mathematical reasoning in decision-making [6] for the manufacture of various things, as is the case with the construction of mosaics [7]. Because of its open-source nature, GIMP is the perfect candidate where image processing for vectorization and color reduction can be done. GIMP's plugin architecture allows an easy extension of

the available software functionality by an efficient usage of the available application programming interface (API).

The goal of this optimization process is for the approximate image to be as close as possible to the original image. The general term for such image processing is a simplification [8]. The same visual information is presented with fewer colors and higher granularity [9]. This article proposes a GIMP plugin, which is doing a tile-based mosaic image with the application of genetic algorithms as an optimization technique.

## 2. Genetic algorithm for vectorization and color reduction

Genetic algorithms are naturally inspired optimization metaheuristics for global optimization. The main idea is the organization of optimal solution searching as a list of candidates called population. In most genetic algorithm implementations the initial population is randomly generated, but it is not mandatory. In the case of mosaic construction, the initial population can be generated by another optimization algorithm. Evolution and recombination in genetic algorithms are organized into three common operators - selection, crossover, and mutation.

At each cycle of evolution (epoch) from the current generation in the population, the next generation is created as process of recombination. This process starts by selecting individuals to mate. The idea behind the selection operator is better-fitted individuals to recombine and to produce offspring with the hope that the offspring will be better than its parents. The crossover operator mixes the elements from the parents (generally two parents, but it is not mandatory and more than two can be used). Crossover does exploration in the variables space and it is done by single-point cut, two-points cut, or uniform swap in most of the implementations. The produce offspring after crossover is put under mutation operator. The mutation operator does exploitation in the variables space and it searches in local surroundings of the individuals. The mutation is done by a small change of some randomly choose elements in the individual. If some of the best-found individuals survive between the generations it is called an elitist rule. Each new individual is put under fitness value evaluation. The fitness value is calculated from the target function under optimization.

Solving image construction with tiles by genetic algorithms starts with a proper encoding of the individuals into the population. The problem is discrete and the variable space is finite. There is a fixed number of tiles to be used for an image approximation. Each element in the chromosome is represented by the index of the tile color to be used. Chromosomes are with a fixed lengths corresponding to the total number of tiles to be used for the image approximation. The list of colors is drawn in order columns by columns, rows by rows. Selection is done by taking two better-fitted individuals as parents from three randomly

selected individuals. Elitism is indirectly implemented in the chosen selection operation. The third individual is replaced by the produced offspring. Uniform crossover is done by randomly taking elements from the parents but by keeping their positions. The mutation is done by a random change of color, from the list of available colors, in the randomly chosen element. Evaluation of the fitness function is done by calculation of original image and approximated image closeness.

## 3. Technical implementation

GIMP supports two common ways for plugin development - C-based and Python-based. The main advantage of the C programming language for GIMP plugins development is the speed of calculations. C is a programming language that is very close to the assembler and to the machine code. Also, it is a compiler type of language. C has two common disadvantages for plugins creation. In the first place, the process of compiling and deploying C plugins is much more complicated than Python plugins. The second problem is the fact that the support of different platforms (hardware and OS) requires precompilation of the plugin for each platform. For Python plugins in GIMP tradeoff is between the lower performance and the easier distribution and deployment.

The usage of the Python subset in GIMP comes with some difficulties in the process of testing and debugging. There is no compilation in Python. Also, there is only runtime error handling. Due to the specifics of Python, best testing practices should be used during the software development process [10]. The plugin organizes the image processing in a group of layers. The topmost layer contains the original 24-bits color image. The layer below it contains an image in which pixels are used as a color palette. The color palette is important because it gives the list of colors available as real ceramic tiles. Getting all pixels from a bitmap image can be very time-consuming that is why it is rational for the color platte layer to be as smaller as possible. It is enough for each color to be represented only by a single pixel. During calculation, the plugin creates two more layers - for the approximated image and for tiles usage statistics.

The most important plugin argument is the desired number of tiles for the approximating image. Images are two-dimensional whether measured in pixels or in tiles. The number of desired tiles is transformed to a number of tiles by the x-axis and the number of tiles by the y-axis. This calculation is done according to single side square tile size in pixels. If the original image does not match the area covered with the estimated number of tiles the original image is resized (the resize is plugin parameter controlled).

When population initialization is done by searching closeness of tiles list of colors the average color of the area under the tile in the original image is

calculated. Having the average color, the list of tiles colors is checked for the closest value, which after that is used in chromosome initialization. Only one population is used in the implementation of the algorithm. The offspring emerge as a new individual, replacing one of the old individuals. Classic uniform crossover is realized by taking randomly selected elements from both parents. The mutation is realized by replacing a randomly selected element with a random element from the list of tile colors.

The evaluation of the fitness that the offspring has is performed through the built-in capabilities of GIMP. The DIFFERENCE_MODE mode is set on the layer containing the original picture. After that, the elements of the evaluated chromosome are drawn in the approximation layer. Of all the layers in the GIMP project file, only the layer with the original picture and the layer with the approximated picture are switched-on as visible. The resulting image is a difference between the original image and the approximated image. The areas with close pixels are drawn in black. The areas with distant pixels are drawn in some colorful shades. The average color of the resulting image is taken as chromosome fitness value.

## 4. Experiments and results

The experiments are done with GIMP 2.10.22 (Fig. 2) under macOS High Sierra 10.13.6. The population size is chosen empirically to be 37. According to the literature for the parameters of genetic algorithms, the crossover is chosen to be 0.95, and the mutation is chosen to be 0.01.

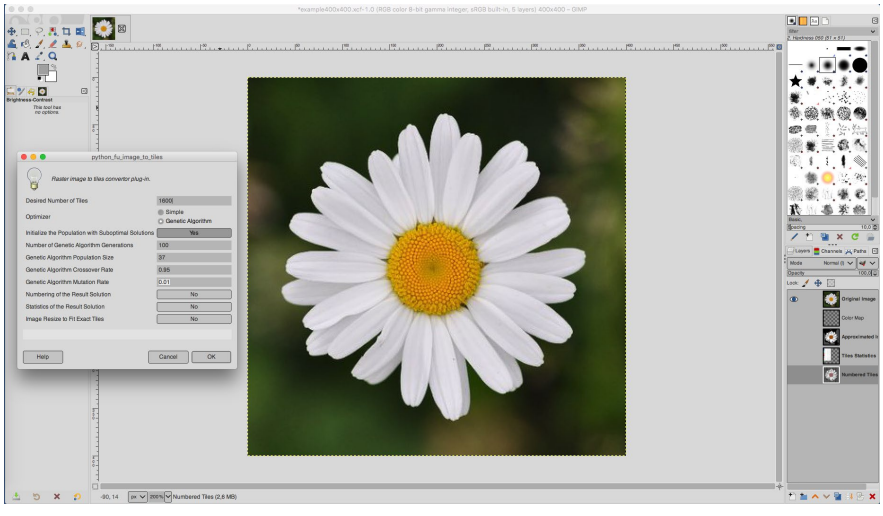



Fig. 2. Activation of the GIMP plugin

The original image (Fig. 3a) has 400 x 400 pixels as dimensions. Each pixel is represented as 24 bits colors (8 bits for each channel – red, green, blue). The color palette of the approximated image consists of 16 index colors (aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, yellow).

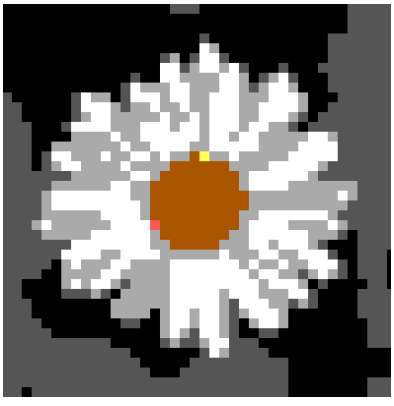
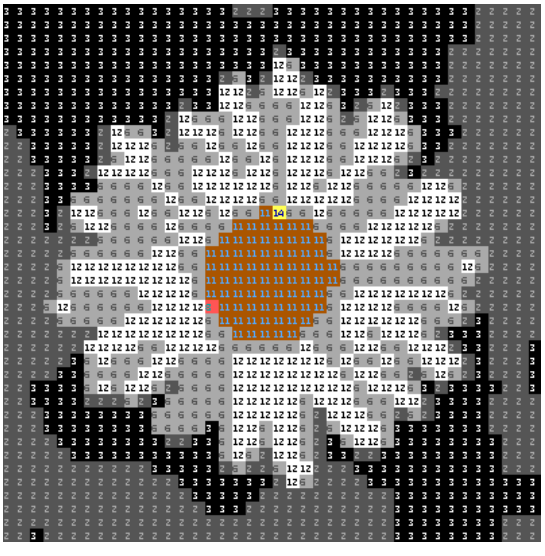
a)  b) 

Fig. 3. The original image (a) and the approximated image (b)

The experiment is performed with 1600 tiles, which resulted in an image 40 x 40 tiles (Fig. 3b). The information about the number of used colors (Fig. 4a) and the quantities of tiles can be very useful (Fig. 4b).

a) 

b)

| | | |
|---|---|---|
| 1 | 0 | (85, 85, 255, 255) |
| 2 | 474 | (85, 85, 85, 255) |
| 3 | 468 | (0, 0, 0, 255) |
| 4 | 0 | (85, 255, 255, 255) |
| 5 | 0 | (0, 170, 170, 255) |
| 6 | 296 | (170, 170, 170, 255) |
| 7 | 0 | (0, 170, 0, 255) |
| 8 | 1 | (255, 85, 85, 255) |
| 9 | 0 | (170, 0, 170, 255) |
| 10 | 0 | (170, 0, 0, 255) |
| 11 | 73 | (170, 85, 0, 255) |
| 12 | 287 | (255, 255, 255, 255) |
| 13 | 0 | (0, 0, 170, 255) |
| 14 | 1 | (255, 255, 85, 255) |
| 15 | 0 | (255, 85, 255, 255) |
| 16 | 0 | (85, 255, 85, 255) |

Fig. 4. Numbering of used tiles (a) and tiles statistics (b)

Optionally each tile has a number of the color used for it. Such numbering is very important during real construction works because projecting the approximated image on the working wall can lead to color deviations.

The generated figures clearly show that the proposed GIMP plugin is very efficient and can be a useful tool for workers in art industries as interior design and handmade tapestry.

## 5. Conclusion

In this research, a genetic algorithm is proposed as state of the art in soft computing [11], realized as a GIMP plugin. The advantages of this algorithm is the ability to make a vectorization of full-color images to square tiles and reduction of their colors. The experimental results are very promising and the proposed technical solution can lead to industrial effective usage.

As further research, with addition of human-computer interaction [12] the achieved results can be improved to a better extend. The hierarchical approach [13] in image processing or Kalman filter [14] are interesting directions in which further research can be done.

## Acknowledgments

This research is funded by Velbazhd Software LLC and it is partially supported by the Bulgarian Ministry of Education and Science (contract D01–205/23.11.2018) under the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICTinSES)", approved by DCM # 577/17.08.2018.

## References

1. Borissova, D.: Web Programming Basics. Publisher: Za Bukvite-O pismenah, pages 255 (2014).
2. Umezu, N., Takahashi, E.: Visualizing color term differences based on images from the web. Journal of Computational Design and Engineering, 4(1), 37–45, (2017), https://doi.org/10.1016/j.jcde.2016.08.002.
3. Song, T., Xin, L., Gao, Ch., Zhang, T., Huang, Y.: Quaternionic extended local binary pattern with adaptive structural pyramid pooling for color image representation. Pattern Recognition 115, 107891, (2021), https://doi.org/10.1016/j.patcog.2021.107891.
4. Yao, R., Liao, S., Dai, Ch., Yang, Y., Zheng, F.: Dual functions of novel glass–ceramic floor tile design and preparation. Ceramics International, 40(6), 8667–8675, (2014), https://doi.org/10.1016/j.ceramint.2014.01.084.
5. Ghosh, D., Kaabouch, N.: A survey on image mosaicing techniques. Journal of Visual Communication and Image Representation 34, 1–11, (2016), https://doi.org/10.1016/j.jvcir.2015.10.014.

6. Cvetkova, P., Pandulis, A., Borissova, D.: Application of information technologies to support mathematically reasoned decisions. In: Knowledge Society and 21st Century Humanism Proceedings, pp. 488–496, Academic Publishing House Za Bukvite - O Pismeneh, Sofia (2020).
7. Pandey, A., Pati, U. C.: Image mosaicing: A deeper insight. Image and Vision Computing 89, 236–257, (2019), https://doi.org/10.1016/j.imavis.2019.07.002.
8. Balabanov, T., Barova, M., Keremedchiev, D.: Image construction with 2D ellipses by genetic algorithms optimization. In: 11th Annual Meeting of the Bulgarian Section of SIAM Proceedings, pp. 10–11. Fastumprint, Sofia (2016).
9. Chen, Zh., Chi, Zh., Zinglersen, K. B., Tian, Y., Wang, K., Hui, F., Cheng, X.: A new image mosaic of Greenland using Landsat-8 OLI images. Science Bulletin 65(7), 522–524, (2020), https://doi.org/10.1016/j.scib.2020.01.014.
10. Dimitrov, W.: Software testing. Avangard, Sofia (2017).
11. Angelova, V.: Investigations in the Area of Soft Computing Targeted State of the Art Report. Cybernetics and Information Technologies 9(1), 18–24 (2009).
12. Bakanova, N., Bakanov, A., Atanasova, T.: Modelling human-computer interactions based on cognitive styles within collective decision-making. Advances in Science, Technology and Engineering Systems Journal 6(1), 631–635 (2021).
13. Tashev, T., Hristov, H.: Hierarchical Interconnection Modeling in Computer Systems. In: International Scientific Conference Communication, Electronic and Computer Systems Proceedings, pp. 215–220, Sofia (2000).
14. Alexandrov, A.: Ad-hoc Kalman Filter Based Fusion Algorithm for Real-Time Wireless Sensor Data Integration. In: 11th International Conference on Flexible Query Answering Systems Proceedings, pp. 151–159. Springer, Cham (2015).