Pol. J. Appl. Sci., **2018**, *4*, 131-136

HIGHLIGHTING A CLOSE-TO-ONE-SPECIFIC-COLOR SURFACE BASED ON THRESHOLDING COLOR-CHANNEL-WISE DISTANCES TO A PIXEL PALETTE

Vadim Romanuke

Faculty of Navigation and Naval Weapons
Polish Naval Academy, Gdynia, Poland

E-mail: romanukevadimv@gmail.com

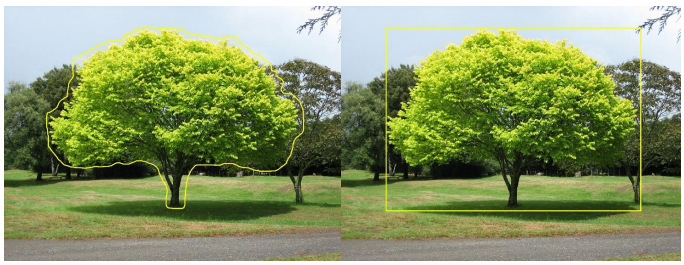
Abstract: A conception of highlighting close-to-one-specific-color surfaces (sea water, a green forest, asphalt on a highway, etc.) is presented. The goal is to develop an efficient algorithmic routine for highlighting such surfaces. Once a bank of diverse images having the color of interest is chosen, a pixel palette is formed. The pixel palette is of one-pixel samples, each of which has a unique color close to the color of interest. Then, for every single pixel sample in the palette, color-channel-wise distances between the image and palette are computed and normalized. The normalized distances are compared to a threshold for every sample. If the distances are less than the threshold (for all the color channels simultaneously), the corresponding pixels of the image are highlighted. With the before-formed pixel palettes, when the threshold is fine-tuned, the developed routine is fully automatic. It requires only a type of highlighting task (whether it is the sky, clouds, asphalt, calm river water, or any other similar surfaces of medium complexity) to be input to load the corresponding pixel palette and threshold. Similar to chroma keying, the suggested method can nonetheless work with complex colors, so it is more robust.

Key words: color image, surface highlighting, pixel palette, color-channel-wise distance, threshold

DOI: 10.34668/PJAS.2018.4.4.02

Introduction

The problem of highlighting objects in a color image is typical in processing images and representing them in a more convenient view for human analysis or decision making [1, 2]. This problem differs from an object detection problem in that an object meant to be highlighted should be contoured precisely (Fig. 1). In the simplest case, an object to be highlighted has close-to-one specific color. This can be, for example, sea water, a green forest, asphalt on a highway, etc. Then the problem may be called a surface highlighting [3, 4].



A green tree which is highlighted (contour only) A green tree which is detected

Fig. 1: A difference of the object highlighting problem from the object detection problem.

Despite the seeming simplicity for one-colored surfaces, an algorithm for highlighting cannot be built just by matching colors. Firstly, the surface color may slightly differ from image to image. For instance, the color of a sea depends on sunlight, depth, wind speed, and some other factors. Secondly, if a few objects have roughly the same color, then they may be confused with the surface and thus be wrongfully highlighted. Keeping the exemplification, a boat of the sea color is either masked in the surface or pops out in the horizon appearing after highlighting as a strange wave or something.

Hence, surface highlighting is intended to be robust. Robustness is meant here mainly as independence of natural and artificial light (including daylight opposed to dawn and evening lights), the presence of objects or surfaces having the same color (including casting shadows), contortion, and warping [5]. This is addressed by acquiring a variety of color representation [6].

Known approach analysis

Surface highlighting is commonly similar to image segmentation [2, 4, 7] including highlighting as a partial case. Both of them have a goal of simplifying and/or changing the representation of an image into something that is more meaningful and easier to analyze/perceive. Thresholding is the simplest method of image segmentation, which works best for grayscale images [4, 7]. Color images can also be thresholded, where a separate threshold for each of the RGB components of the image is designated, and then these three are combined with logical conjunction. This reflects the way a camera works and how the data is stored in a computer, but it does not correspond to the way that people recognize color. Therefore, the HSL and HSV color models are more

often used although they ignore much of the complexity of color appearance [1,8]. Moreover, since hue is a circular quantity, it requires circular thresholding, and thus the HSL and HSV color models may require the use of circular statistics [9]. Eventually, they appear too complicated for the partial case of image segmentation.

On the other hand, a surface could be highlighted by the means of more complex methods. In this way, algorithms for highlighting use clustering pixels [10]. The classic clustering algorithm is an iterative technique, and it is guaranteed to converge, but it may not return the optimal solution. The quality of the solution depends on the initial set of clusters. Compared to thresholding, clustering algorithms are slower due to the multiple iterations.

Histogram-based methods are very efficient compared to other image segmentation methods because they typically require only one pass through the pixels [11]. In this technique, a histogram is computed from all the pixels in the image. Then the peaks and valleys in the histogram are used to locate the clusters in the image, wherein either color or intensity is used as the measure. Despite the speed of this method, its disadvantage is that it may be difficult to identify significant peaks and valleys in the image.

Methods based on solving partial differential equations [12], including fast marching method [13], and minimizing a specific energy functional [14] can be fast and efficient. However, numerical approaches for approximating the solution often require the sampling strategy to be adjusted. Thus, those methods may slow down.

Deep neural networks are the most complex models to image segmentation [15]. If they are properly trained, their performance is close to perfect, especially when only a single close-to-one-specific-color surface is highlighted (the background is highlighted as well). Nevertheless, deep learning is very time-consuming: a deep neural network is trained slower than any of the listed methods, and it performs slowly consuming roughly the same computational resources.

So, thresholding is a compromise approach, which is non-iterative, allowing an appropriate solution to be rapidly found by parallelization. An obvious problem of thresholding is that it works normally when a good background to foreground contrast ratio exists. This problem trebles for color images. Factual convergence of thresholding algorithms trying to execute automatic thresholding varies dramatically [7]. Hence, the surface highlighting problem requires an algorithmic simplification, which is possible by working even with RGB models where GPU computations are easily applied nowadays [16,17].

Goal

Because of an apparent lack in algorithmizing the surface highlighting problem, an efficient algorithmic routine for close-to-one-specific-color surfaces will be developed. To achieve this goal, four tasks need to be performed:

1. To state mathematically the main principles of highlighting in color images using a standard three-dimensional matrix representation of a color image.
2. To select a surface highlighting task for benchmarking and test the stated principles on it.
3. To graph the routine for surface highlighting with pointing out weak places in it.
4. To discuss the routine and conclude by claiming future advances in highlighting surfaces.

Highlighting close-to-one-specific-color surfaces

A color image is typically represented as a three-dimensional matrix

$$\mathbf{M} = \left[m_{ijk}\right]_{h\times w\times 3} \tag{1}$$

where h is a number of height pixels, w is a number of width pixels, m_{ij1}, m_{ij2}, m_{ij3} are intensities of red, green, blue, respectively. Matrix (1) can be associated and referred to as just the image it represents and vice versa. Ranges of values in matrix (1) vary depending on a maximal number of colors (say, it can be 256, 4096, 65536, or even more colors). A specific color of interest can be thought of as a vector $\mathbf{P} = \left[p_{11k}\right]_{1\times 1\times 3}$, which is actually a color of the pixel. Vector $\mathbf{P}$ is factually represented as a 1-by-1-by-3 matrix. The difference between each pixel in image (1) and $\mathbf{P}$ is expressed as a distance common for the most of applications (though this is a color-channel-wise distance):

$$d_{ijk} = |m_{ijk} - p_{11k}| (i = \overline{1,h}, j = \overline{1,w}, k = \overline{1,2}). \tag{2}$$

Due to that ranges of values for various images may differ, distance (2) is normalized:

$$d_{ijk}^{\langle 1\rangle} = \frac{d_{ijk}}{\max\limits_{l=\overline{1,h}} \max\limits_{q=\overline{1,w}} \max\limits_{z=\overline{1,3}} d_{lqz}} (i = \overline{1,h}, j = \overline{1,w}, k = \overline{1,3}). \tag{3}$$

If

$$d_{ijk}^{\langle 1\rangle} < d_0 \quad \forall k = \overline{1,3}, \tag{4}$$

where d_0 is a threshold, then pixel i, j is highlighted.

However, $\mathbf{P}$ is not robust. It should be formed by a variety of vectors representing colors similar to the color of interest.

Hence, a bank of diverse images having the color of interest is chosen. Based on this image bank, a pixel palette is formed. This palette is a four-dimensional matrix/array:

$$\mathbf{M}_S = \left[p_{11ks}\right]_{1\times 1\times 3\times S}. \tag{5}$$

Pixel palette (5) is of S one-pixel samples, each of which has a unique color close to the color of interest. All pixels in (5) are enough different. Then condition (4) for highlighting is re-stated with more general distance (2) and normalized distance (3):

$$d_{ijks} = |m_{ijk} - p_{11ks}| (i = \overline{1,h}, j = \overline{1,w}, k = \overline{1,3}, s = \overline{1,S}), \tag{6}$$

$$d^{\langle 1\rangle}_{ijks} = \frac{d_{ijks}}{\max\limits_{l=\overline{1,h}} \max\limits_{q=\overline{1,w}} \max\limits_{z=\overline{1,3}} \max\limits_{u=\overline{1,S}} d_{lqzu}} \quad (i = \overline{1,h}, j = \overline{1,w}, k = \overline{1,3}, s = \overline{1,S}). \tag{7}$$

Thus, if

$$d^{\langle 1\rangle}_{ijks} < d_0 \quad \forall k = \overline{1,3} \quad \text{for (at least) some} \quad s \in \overline{1,S}, \tag{8}$$

then pixel i, j is highlighted. Obviously, a threshold in condition (8) should be lower than that in one-sample condition (4).

A greater number of samples in palette (5) correspond to a more complicated texture of the surface to be highlighted. For instance, the texture of a blue sky (without clouds) is very simple and is represented with a few tones of the light blue, whereas the texture of sea water (especially with waves) has a far wider range of colors. Therefore, number for the cloudless sky may not exceed 5 to 9, but the pixel palette for sea water should consists of at least a few thousands of pixels.

A highway cover highlighting task

Among others, the task of highlighting highway asphalt is not the simplest one. The texture of asphalt is similar to that of sea water, but the heterogeneousness of the asphalt surface is seen only by zooming in. This is why the highway cover highlighting task is of medium complexity.

A bank of 10 highway images from which pixel palette (5) is going to be formed is shown in Fig. 2 (the images are freely available on the Internet). Each image has quite a monotonous highway view, from which a few tiny parts are extracted and averaged to a pixel for its palette (5). Namely, if

$$\mathbf{T}_{ns} = \left[t^{(ns)}_{ijk}\right]_{h_{ns}\times w\times 3} \tag{9}$$

is an $h_{ns} \times w_{ns}$ region off an image (the n-th tiny part for the s-th pixel sample), where $n = \overline{1,N}$ by $N_s \in \mathbb{N}$, then the s-th pixel sample is

$$p_{11ks} = \frac{1}{N_s \cdot h_{ns} \cdot w_{ns}} \cdot \sum_{n=1}^{N_s} \sum_{i=1}^{h_{ns}} \sum_{j=1}^{w_{ns}} t^{(ns)}_{ijk} (k = \overline{1,3}, s = \overline{1,S}). \tag{10}$$

Pixel palette $\mathbf{P}_{30} = \left[p_{11ks}\right]_{1\times 1\times 3\times 30}$ for this task is shown in Fig. 3. Generally, the number of tiny parts clipped out of an image can vary. The number of pixel samples made off an image can vary as well. Smaller images may "produce" fewer pixel samples, for which integer N_s is likely to be small too (e. g., a one tiny part per image can be). A bigger image will give several pixel samples, for each of which the number of tiny parts (the respective integer N_s) is likely to be greater (up to 10 and even more).

Fig. 2: A basis of 10 highway images (the original aspect ratio has been maintained) for forming a pixel palette of the highway cover (dry asphalt).

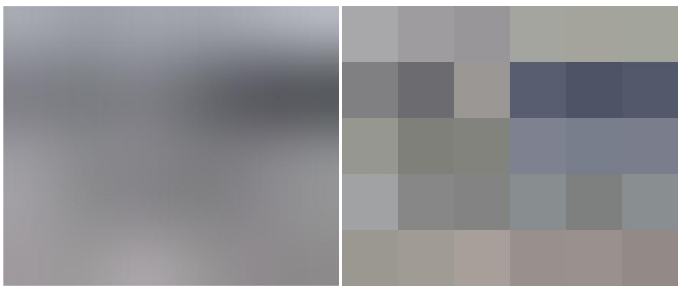


Fig. 3: A pixel palette of the highway cover (dry asphalt) based on initial images in Fig. 2 (every image in Fig. 2 has "produced" three pixel samples for this palette); the left-side view is just a result of file format conversion that may misleadingly appear as a "continuous" palette, contrary to the right-side real 5×6 palette (consisting of 30 pixel samples).

For testing the highlighting method, images with noise or interference are suitable (Fig. 4). Fig. 5 shows those three test images, wherein the highway cover is highlighted by threshold $d_0 = 0.05$ (a 5% deviation analogue) under condition (8) for $s = \overline{1,30}$. More rigorous analysis, by

$d_0 = 0.03$, reveals some highway parts (Fig. 6) missed while averaging by (9) and (10). Either this may imply that the set of 10 initial images for forming pixel palette misses a few representative images (a non-complete basis of the surface).

Fig. 4: A set of three test images (the original aspect ratio has been maintained).

Fig. 5: The set of three test images of the highway whose asphalt is appropriately highlighted by $d_0 = 0.05$ (although there are some gaps).

Fig. 6: The set of three test images of the highway whose asphalt cover is non-uniformly highlighted by $d_0 = 0.03$ (an unacceptable result).

It is quite obvious that the quality of highlighting strongly depends on the threshold. For the tested task, it is sufficiently effective to set $d_0 = 0.05$ although it only concerned the definite set of bank images (Fig. 2). For cases of other tasks, the threshold value should be decreased if the surface dominant colors may be easily confused with some fragments in the same image.

In general, dichotomization can be used for setting (fine-tuning) the threshold appropriately [18,19]. First of all, the threshold is set in the middle between a minimally possible $d_0^{\langle min\rangle}$ and maximally possible $d_0^{\langle max\rangle}$ value:

$$d_0 = \frac{d_0^{\langle max\rangle} + d_0^{\langle min\rangle}}{2}, \quad \Delta d_0 = \frac{d_0^{\langle max\rangle} - d_0^{\langle min\rangle}}{2}. \tag{11}$$

Then, the next threshold d_0 is tried, either closer to $d_0^{\langle min\rangle}$ (to the left if the previous threshold shows excessive highlighting) or $d_0^{\langle max\rangle}$ (to the right if highlighting by the previous threshold is deficient). Hence,

$$\Delta d_0^{\langle obsol\rangle} \leftarrow \Delta d_0, \quad \Delta d_0 \leftarrow \frac{\Delta d_0^{\langle obsol\rangle}}{2}, \tag{12}$$

and either

$$d_0^{\langle obsol\rangle} \leftarrow d_0, \quad d_0 \leftarrow d_0^{\langle obsol\rangle} - \Delta d_0 \tag{13}$$

after excessive highlighting, or

$$d_0^{\langle obsol\rangle} \leftarrow d_0, \quad d_0 \leftarrow d_0^{\langle obsol\rangle} + \Delta d_0 \tag{14}$$

after deficiency in highlighting. Such a subroutine is repeated until Δd_0 becomes too insignificant (say, less than 0.0001). An alternative option is a maximal number of repetitions of (12) and either (13) or (14). Those two options can be aggregated by an operation of logical conjunction. It is worth noticing that formulae (12), (13), and (14) are intentionally given in the form convenient for explicit low-level programming (at the stage of implementation).

A routine for surface highlighting

In gathering a bank of images for pixel samples, only relevant images where surface-of-interest appears differently in every image must be selected. Then tiny parts of the surface are extracted, about 3 to 10 parts per image (or even a few tens of parts if the surface is big enough). The parts do not necessarily need to be the same size. They can be as square, as well as rectangular, or like long rectangular stripes. Suppose that, after averaging over those tiny parts (9) by formula (10), a total number of pixel samples is $S_0 = S$ (it is no less than the number of the relevant images selected before). Entries of the pixel palette

$$\mathbf{P}_{S_0} = \left[p_{11ks}\right]_{1\times 1\times 3\times S_0} \tag{15}$$

are sorted so that

$$\left\|p_{11(s)}\right\| \leqslant \left\|p_{11(s+1)}\right\| \quad \text{for} \quad s = \overline{1, S_0 - 1} \tag{16}$$

by the Euclidean norm

$$\left\|p_{11(s)}\right\| = \sqrt{\sum_{k=1}^{3} p_{11ks}^2} \quad \text{for} \quad s = \overline{1, S_0}. \tag{17}$$

If

$$\frac{\left\|p_{11(s+1)}\right\| - \left\|p_{11(s)}\right\|}{\left\|p_{11(S_0)}\right\|} < \varepsilon \quad \text{for some} \quad \varepsilon > 0 \quad \text{and some} \quad \varepsilon \in \left\{\overline{1, S_0 - 1}\right\} \tag{18}$$

then these two pixel samples are very similar. They practically duplicate information for a highlighting tool, and one of them is deleted. Then number S_0 is correspondingly decreased. Condition (18) is controlled for all until (after re-indexation due to a decreased number of pixel samples after deletions) inequality

$$\frac{\left\|p_{11(s+1)}\right\| - \left\|p_{11(s)}\right\|}{\left\|p_{11(S_0)}\right\|} \geqslant \varepsilon \quad \forall s = \overline{1, S_0 - 1} \tag{19}$$

holds. In fact, inequality (19) is a requirement of that the pixel palette be formed of different enough entries.

Thus, the formation of pixel palette (5) is completed. Afterwards, distances (6) and (7) are found. Eventually, condition (8) is checked for highlighting. The described routine is graphed in Fig. 7.

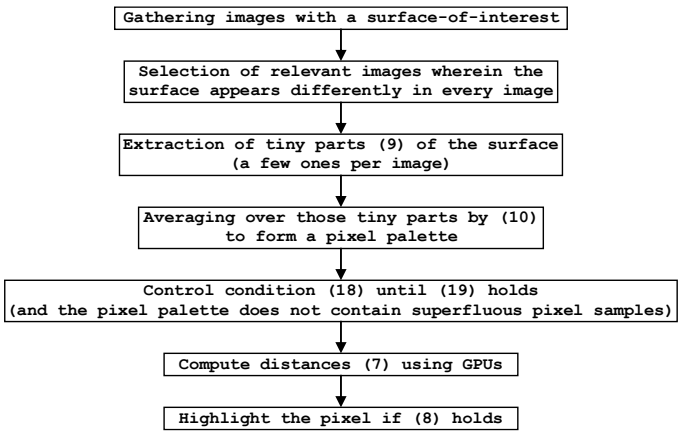


Fig. 7: A routine for highlighting a surface which is of a close-to-one specific color.

A deficiency of the routine is that the threshold cannot be set at a value from scratch. Fine-tuning of the threshold is commonly needed for every new task. Another weakness is that relevance of images in forming the pixel palette is subjective. Thus, sometimes the pixel palette may become sparse although condition (19) holds. On the other hand, the pixel palette may be outer incomplete, i.e. there exist such indices s^* (along with respective images or their parts), for which either inequalities

$$\left\|p_{11(S_0)}\right\| < \left\|p_{11(s^*)}\right\| \quad \text{and} \quad \frac{\left\|p_{11(s^*)}\right\| - \left\|p_{11(S_0)}\right\|}{\left\|p_{11(s^*)}\right\|} \geqslant \varepsilon \tag{20}$$

or inequalities

$$\left\|p_{11(s^*)}\right\| < \left\|p_{11(1)}\right\| \quad \text{and} \quad \frac{\left\|p_{11(1)}\right\| - \left\|p_{11(s^*)}\right\|}{\left\|p_{11(S_0)}\right\|} \geqslant \varepsilon \tag{21}$$

become true. In such situations, the palette is supplemented with the s^*-th pixel sample. However, the greatest weakness of the routine is that it works poorly with surfaces like forests, mountains, sea water with waves, etc.

Discussion

The developed routine based on thresholding color-channel-wise distances by (8) is intended for highlighting close-to-one-specific-color surfaces. This is reminiscent of working with the chroma key [20, 21], but the routine would be applicable for cases when the background "screen" is color-heterogeneous ("sloppy"). If the surface-of-interest is of a few dominant colors, but they are close by the color-channel-wise distance (3), the suggested method is efficiently applicable also by sufficiently filled pixel palette (5). The efficiency apparently increases if the surface colors are more distant (by the same distance) from the rest of the colors in the image (like the green/blue screen for applying the chroma key). For processing a bunch of one size images, the highlighting routine can be efficiently parallelized for GPU computations by assigning small groups of pixels to computational threads and thresholding in parallel. Thus, the operation speed will be satisfactory compared to other techniques of highlighting.

The main specificity of the developed routine is in forming pixel palette (5) and setting the same threshold at some value. If this palette is too big, the highlighting process will be retarded. For a shallow pixel palette, this process will be quicker but probably not effective (resembling the result in Fig. 5). A rational solution is to form primarily a big pixel palette (15) and narrow its entries by conditions (18), (19). For this, however, an additional substantiation of is required.

Despite the normalizations in (7), (18), and (19), both the threshold and are fine-tuned in a few stages for every new highlighting task. This is an issue as the fine-tuning takes its while, and thus the highlighting process is retarded. Besides, an automation mode for the fine-tuning is desirable.

Conclusion

Compared to chroma keying, the developed routine uses complex thresholding to highlight surfaces of medium complexity. With the before-formed pixel palettes, when the threshold is fine-tuned, the highlighting routine is fully automatic. It will only require a type of highlighting task (whether it is the sky, clouds, asphalt, calm river water, or something similar) to be input to load the corresponding pixel palette and threshold. Similar to chroma keying, the suggested method can nonetheless work with complex colors, so it is more robust. A promising solution for tasks with a more complicated texture of the surface to be highlighted, for which the suggested method works poorly, is to provide an adaptive formation of the bank of samples. In such cases, a bank of small regions (bigger than just a pixel) is formed from the image with the surface-of-interest, using the shortest color-channel-wise distance between the image and the same size small region montaged as a "big pixel" from a primary pixel palette.

Literature

[1] Rhyne T.M. *Applying Color Theory to Digital Media and Visualization.* CRC Press, Taylor & Francis Group, Boca Raton, 2017.

[2] Klette R. *Concise Computer Vision: An Introduction into Theory and Algorithms.* Springer, London, 2014.

[3] Rogowska J. *Handbook of Medical Image Processing and Analysis, 2nd edition*, chapter Overview and Fundamentals of Medical Image Segmentation., pages 73 – 90. Academic Press, San Diego, 2009.

[4] Davies E.R. *Computer Vision, 5th edition.* Academic Press, San Diego, 2018.

[5] Wong W.K. *Applications of Computer Vision in Fashion and Textiles.* Woodhead Publishing, Cambridge, 2018.

[6] Best J. *Colour Design, 2nd edition.* Woodhead Publishing, Cambridge, 2017.

[7] He H.J., Zheng C., Sun D.W. *Computer Vision Technology for Food Quality Evaluation, 2nd edition.*, chapter Image Segmentation Techniques, pages 45 – 63. Academic Press, San Diego, 2016.

[8] Keith J. *Video Demystified, 5th edition: A Handbook for the Digital Engineer (Learning Made Simple)*, chapter Color Spaces.

[9] Fisher N.I. *Statistical Analysis of Circular Data.* Cambridge University Press, Cambridge, 1993.

[10] Larsson C. *5G Networks*, chapter Clustering, page 123 – 141. Academic Press„ San Diego, 2018.

[11] Shapiro L.G., Stockman G.C. *Computer Vision.* Prentice-Hall, New Jersey, 2001.

[12] Caselles V., Kimmel R., Sapiro G. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1997.

[13] Sethian J.A. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.

[14] Chan T.F., Vese L. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001.

[15] Badrinarayanan V., Kendall A., Cipolla R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(16):2481–2495, 2017.

[16] Trobec R., Vajteršic M., Zinterhof P. *Parallel Computing. Numerics, Applications, and Trends.* Springer, London, UK, 2009.

[17] HajiRassouliha A., Taberner A.J., Nash M.P., Nielsen P.M.F. Suitability of recent hardware accelerators (DSPs, FPGAs, and GPUs) for computer vision and image processing algorithms. *Signal Processing: Image Communication*, 68:101–119, 2018.

[18] Romanuke V.V. Interval uncertainty reduction via division-by-2 dichotomization based on expert estimations for short-termed observations. *Journal of Uncertain Systems*, 12(1):3–21, 2018.

[19] Romanuke V.V. Hard and soft adjusting of a parameter with its known boundaries by the value based on the experts' estimations limited to the parameter. *Electrical, Control and Communication Engineering*, 10:23–28, 2016.

[20] Jackman J. *Bluescreen Compositing.* Focal Press, Oxford, 2007.

[21] Meyer T., Meyer C. *Creating Motion Graphics with After Effects: Essential and Advanced Techniques, 4th edition.* Focal Press, Oxford, 2008.

Received: 2018
Accepted: 2018