*Sensors and Materials*, Vol. 33, No. 4 (2021) 1343–1352
MYU Tokyo



S & M 2538

# Rapid Local Image Style Transfer Method Based on Residual Convolutional Neural Network

Liming Huang,[1] Ping Wang,[2*] Cheng-Fu Yang,[3,4**] and Hsien-Wei Tseng[2]

[1]College of Mathematics and Information Engineering, Longyan University, Fujian 364012, China
[2]College of Artificial Intelligence, Yango University, Mawei District, Fujian 350015, China
[3]Department of Chemical and Materials Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan
[4]Department of Aeronautical Engineering, Chaoyang University of Technology, Taichung 413, Taiwan

(Received October 21, 2020; accepted February 2, 2021)

***Keywords:*** image style transfer, residual neural network, semantic segmentation, DeepLab2, convolutional neural network

The technology of image style transfer can learn the style of a target image in a fully automated or semi-automated way, which is often very difficult to achieve by manual methods, thus saving much time and improving production efficiency. With the rapid spread of commercial software applications such as beauty selfie apps and short entertainment videos such as TikTok, local image style transfer and its generation speed of images are becoming increasingly important, particularly when these recreational products have features especially valued by users. We propose an algorithm that involves semantic segmentations and residual networks and uses VGG16 for feature extraction to improve the efficiency of local image style transfer and its generation speed, and our experiments prove that the proposed method is more useful than other common methods. The investigated technology can be applied in many specific areas, such as the beauty camera of smart phones, computer-generated imagery in advertisements and movies, computed tomography images, nuclear magnetic resonance imaging of cancer diagnosis under harsh conditions, and virtual simulation in industry design.

## 1. Introduction

Images can generally be divided into realistic and nonrealistic ones. The former are true reflections of the real world while the latter are commonly mixed with computational results to create new images with various artistic styles, a process also called nonrealistic rendering. Image style transfer is an important part of nonrealistic rendering and it often tries to extract the artistic style of art works and draw a picture with learned artistic knowledge, and this technology can blur the boundaries between the real world and the virtual one. Machine learning (ML) is the study of computer algorithms that improve automatically through experience. In this study, the proposed neural style transfer algorithm builds a model based on sample training in order to make predictions or decisions without being explicitly programmed to do so using ML techniques. The technology of image style transfer can have important

*Corresponding author: e-mail: pwang@ygu.edu.cn
**Corresponding author: e-mail: cfyang@nuk.edu.tw
https://doi.org/10.18494/SAM.2021.3172

ISSN 0914-4935 

applications in the future, and its development may be rapid and extensive because it can be applied in the culture, recreation, media, animation, and movie industries. The technology investigated in this study can be applied in many specific areas, such as the beauty camera of smart phones, computer-generated imagery in advertisements and movies, computed tomography images, nuclear magnetic resonance imaging of cancer diagnosis under harsh conditions, and virtual simulation in industry design.[1,2]

Image style transfer is a process of modifying the style of an image while still preserving its original content. The technology needs to learn the target image's style in a fully automated or semi-automated way, which can reduce the processing time and improve production efficiency. However, it is difficult and undesirable to achieve image style transfer by manual methods because they are expensive and time-consuming. The technology of neural style transfer is a very important optimization method and it can be used to blend two images, a content image and a style reference image, so that the output image can look like the content image. Because of the rapid progress in the effectiveness of graphics processing units (GPUs) in recent years, it may be possible for convolutional neural computation to be used in industry. Also, the applications of ML and deep learning technologies have led to the rapid development of image style transfer.[3] Today, image style transfer is required to meet the needs of fast conversion and the clipping of specific target areas, which are difficult to execute by the traditional image style transfer methods. Therefore, we proposed a specific local image style transfer method whose speed is improved with training. We also confirmed that the proposed method satisfies the requirements of many application fields, such as smartphone apps and computer-generated imagery (CGI) effect objects.

## 2. Related Studies

Before neural network methods were adopted in image style transfer, it was usually performed using traditional algorithms such as morphological operations, the Sobel operator, region segmentation, and nonlinear filters.[4] Hertzmann proposed the snake relaxation algorithm based on stroke-based rendering (SBR) to stimulate the style of strokes in oil paintings, and he mainly used a multilayer pyramid and the Sobel operator in his technique.[5] DeCarlo and Santella were the first to incorporate region-based segmentation into image style transfer, and they used the down-sampling method to obtain the features in different scales with the Canny operator to refine the edges of target objects.[6] Gatys *et al.* were the first to incorporate the idea of deep learning into image style transfer, and they made a lot of improvements at the same time to overcome the disadvantages of the high complexity and long processing time of the traditional artificial model.[7] Johnson *et al.* proposed a feedforward network trained by a perceptual loss network, which was particularly targeted at image style transfer, and their method improved training speeds as compared with the method proposed by Gatys *et al.*[8] Convolutional neural networks (CNNs) have a highly impressive performance and great potential for advancing different state-of-the-art detections. A deep CNN (DCNN) is also applied in many sensing technologies for different detections, for example, fingerprint likeness detection. Chen *et al.* refined the work proposed by Gatys *et al.* and used the CNNMRF algorithm, which was a combination of the

Markov random field (MRF) and trained DCNNs, to preserve specific features of original images as much as possible. This method used the most similar region of the style images to match each input of the neural area, which reduced the effect of inaccurate feature transfer occurring in other methods.[9] For local image style transfer, image segmentation is difficult to avoid. Champandard proposed a method using a human–machine interaction that allowed users to directly and manually mark priority regions.[10] Long *et al.* first applied fully convolutional networks (FCNs) to image segmentation from end to end, which gave a direct classification of pixels based on abstract semantic representations.[11]

## 3. Our Image Style Transfer Procedures

The novelty of this paper is that we use the technology of semantic image segmentation based on DeepLab2 to segment the foreground and background of art works, and then we use residual network to accelerate the training. The fundamental theorem for image style transfer based on neural networks is to extract the feature maps of an original image. A style image and a style-transferred image are recognized as random white noisy images at the initial stage. The content loss value is defined as

$$L_{content}(\vec{p},\vec{x},l)=\frac{1}{2}\sum_{i,j}(F_{ij}^{l}-P_{ij}^{l})^2, \tag{1}$$

where $P$ is the content of the input image, $x$ is the image after automated style transfer, $l$ is the $l$th layer, $P_{ij}^{l}$ is the content feature of $P$ formed at the $l$th layer, $i$ is the $i$th channel of the convolution, $j$ is the $j$th position of the convolution, and $F_{ij}^{l}$ is the convolution of content feature $x$ formed at the $l$th layer. We can decrease the content loss value by changing the $x$ value. The style loss value is defined as

$$L_{style}\ (\vec{a},\vec{x})=\frac{1}{4N_l^2M_l^2}\sum_{i,j}\ (G_{i,j}^{l}-A_{i,j}^{i})^2, \tag{2}$$

where $a$ is the original style of the input image, $A_{i,j}^{i}$ is the convolution at one layer of the Gram matrix for the input image $a$, $G_{ij}^{l}$ is the convolution of content feature $x$ of the Gram matrix at the $l$th layer, and $M$ and $N$ are the height and width in the $l$th layer channel, respectively. The sum of the loss function, which is a combination of the content and style loss values, is used for calculating the corresponding derivatives to obtain the gradients of all feature maps and then to approach the final style-transferred image by iterative calculations.[12]

### 3.1 Image semantic segmentation

Image style transfer has big potential in commercial applications such as selfies. When traditional methods are used to handle images or videos, they usually have effects on a global scale. The final effects are often not satisfactory, for example, focus points such as human faces

or interesting foregrounds exist, which need to be dealt with independently; thus, it is necessary to segment images as foreground and background. The traditional method for deep-learning-based image segmentation often adopts FCNs, in which the maximum pooling is adopted for down-sampling because it cannot provide exactly classified edges. Although DeepLab2 used in this study is also an image segmentation method, it does not cause this problem.[9,13,14] The whole procedure of the image segmentation is described next.

(1) First, we input original content images into the ResNet-101 pretrained deep neural network. In order to mitigate the negative effects of down-sampling operations, dilated convolution is introduced, which can locate the targets more accurately during the operations of FCNs, then image edges can be confirmed more specifically.

(2) Atrous spatial pyramid pooling (ASPP) is introduced for image segmentation in multiscale states during the procedure mentioned in step (1). The multiple segmented images in different scales are input into the DCNN to calculate the gross rating maps from final feature infusions, which can improve the accuracy of multiscale image segmentation.

(3) Bilinear interpolation is used to recover the resolution of processed images to that of the original image.

(4) A dense convolutional random field (DenseCRF) is used for the classification of the segment image, with pixels being gradually refined to give them classified categories. Then the final segmented image edges are confirmed. Here, DenseCRF is superior to a CRF in FCNs, and the image segmentation effect can be seen in Fig. 1.

### 3.2 Feature extraction

VGG16 is a CNN architecture first proposed by Simonyan and Zisserman. It is used in DCNNs for large-scale image recognition and won the ImageNet Large Scale Visual Recognition (ILSVR) competition in 2014.[15] VGG16 belongs to one of the feedforward neural networks, and it was chosen in this study for feature extraction because it has a powerful learning ability of representation knowledge and is widely adopted in computer vision.[16] As

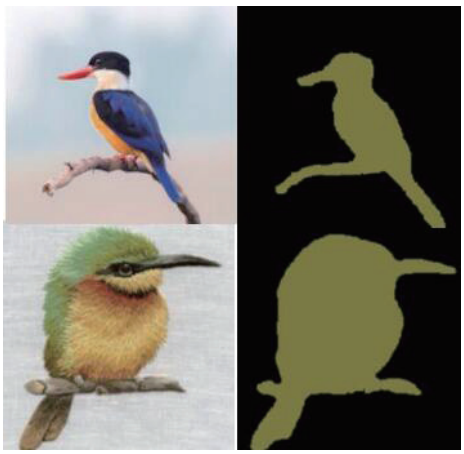

Fig. 1. (Color online) Example of semantic image segmentation.

compared with former neural networks such as AlexNet, the feature extraction ability of VGG16 is much larger than a neural network structure with fewer required training parameters; thus, it has a simplified topology structure and improved performance. The most general structure of the standard invisible layers of VGG16 consists of 13 convolutional layers, five pooling layers, and three fully connected layers, as shown in Fig. 2.

The original image and style image are input into the network and manipulated under convolutional operations with a kernel size of 3 × 3 to extract local features of the input image. The pooling layer is responsible for the image's compression and for halving the size of the output feature map, for which there are two methods: max pooling and average pooling.

### 3.3 Residual network

Gatys' ground-breaking method, which introduces loss functions based on VGG networks, uses gradient descent to obtain the final style-transferred image. This method requires many iterations, resulting in a time-consuming model training process and the use of excessive memory. Because of these problems, this method often fails to converge during training to generate the final style-transferred image. In response, we introduce a residual network to solve the problems of training in the DNN layers and try to reduce the training burden of CNN operations as much as possible.[17]

First, we assume the network structure to be $H(x) = F(x) + x$; then we can transform the equation $H(x) = x$ into a residual function for learning, which is in the form of $F(x) = H(x) - x$. In this equation, when $F(x) = 0$, the optimal solution can be found in the form $H(x) = x$, and then the residual structure can be simplified as

$$x_{l+1} = x_l + F(x_l, w_l). \tag{3}$$

Then, the features of unit $u$ in any deep layer can be expressed as

$$x_u = x_1 + \sum_{i=1}^{u-1} F(x_i, w_i), \tag{4}$$

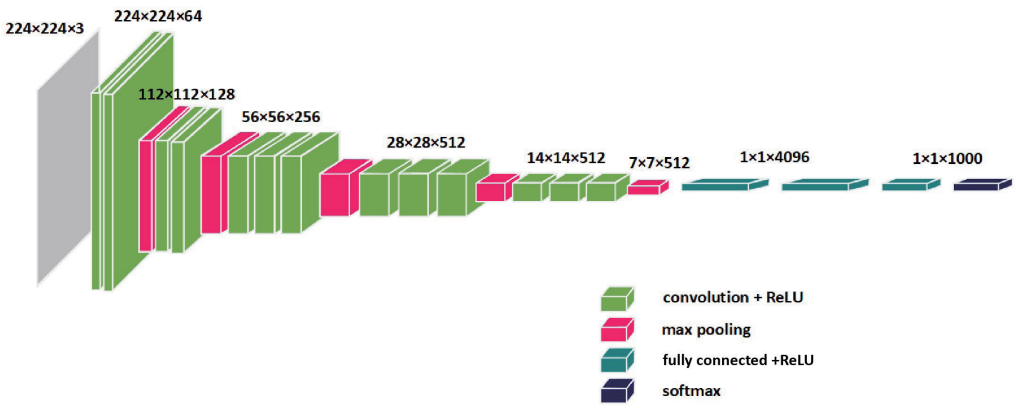


Fig. 2. (Color online) Structure of VGG16.

in which $l$ refers to any shallow layer, $x$ is the feature of any unit, $w$ is the weight value, and $y$ is the output after the residual operation. The interiors of the residual structure are connected by hopping and their calculation formulas can be described as below, in which $\sigma$ is a rectified linear unit (ReLU) of the activation function.

$$F = w_2 \cdot \sigma(w_1 x) \tag{5}$$

$$y = F(x, w_i) + x \tag{6}$$

Similarly, it is much easier to perform optimization in residual networks than in general neural networks because traditional CNNs need many multiplication operations in convergence calculations. However, $H(x)$ in residual networks is realized by multiple additions, resulting in greatly enhanced performance, which is a basis for fast image style transfer. The unit of the residual network can be seen in Fig. 3.

### 3.4 Establishment of loss functions

Loss functions are set up for image derivation with different numbers of iterations, and the final style-transferred image is obtained by repeatedly processing pixels in the white noisy image. We assume that $x$ is the input image, $y$ is the output image, $l$ is the $l$th layer in the convolutional layers, $P$ is the intermediate product after calculation in the $l$th convolutional layer of the input image, $i$ is the number of the channel, $j$ is the position in the convolutional layer, and $F$ is the corresponding convolutional feature in the $l$th layer of $y$. Then the loss function of the content can be expressed as Eq. (7), which can be used to measure the similarity of the input and output images of the contents.

$$Loss_{content}(\vec{x}, \vec{y}, l) = \frac{1}{2}\sum_{i,j}(F_{ij}^{l} - P_{ij}^{l})^2 \tag{7}$$

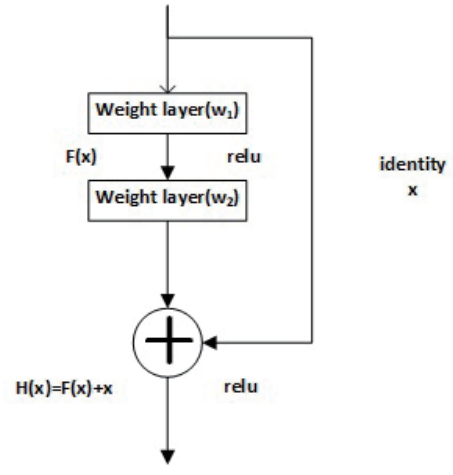


Fig. 3. Unit of residual network.

The fundamental calculation of the image style's core can be expressed as a Gram matrix in the convolution layers, which is a feature matrix for measuring the similarity of style. The loss function of style is given as Eq. (8), in which $M$ and $N$ refer to the height and width of the $l$th layer, respectively, $X$ is the Gram matrix of the input style image, $z$ is in the $l$th layer, and $Y$ can be inferred.

$$Loss_{style}(\vec{z},\vec{y},l) = \frac{1}{4N_1^2M_1^2}\sum_{i,j}(X_{ij}^l - Y_{ij}^l)^2 \tag{8}$$

Finally, the total loss function can be given as below, which is calculated using five convolutional layers:

$$Loss_{total} = \lambda_{content}Loss_{content} + \lambda_{style}Loss_{style}. \tag{9}$$

## 4. Experimental Procedure

We performed experiments to demonstrate the performance of the proposed method, in which we used Python 3.6.4 as the programming language, Ubuntu 16.04 as the operating system, an i5-8600k CPU with 64 GB memory, and a 1080-8G graphics card, with TensorFlow2.0 used as an open-source ML framework. Both the compute unified device architecture (CUDA) and the NVIDIA CUDA deep neural network (cuDNN) were used to increase the GPU performance. Microsoft Coco2014 was chosen as the data list in this study, where we randomly selected 50000 images with an adjusted size of 256 × 256 for training. The relevant parameters were an initial learning rate of 0.01, 500 iterations, and a batch size of 4. The Adam algorithm was used for optimization because it is an adaptive learning rate optimization algorithm and it can be designed specifically for training deep neural networks. One demonstration is presented in Fig. 4 for different styles of image transfer. The images in the rightmost column are rendered by our proposed method, where the people in the photos are friends of the authors, with different art works as the style images.

The above loss formulas are linear combinations of the loss functions of the content and style, and the critical point is how to choose the optimum coefficients to maximize the performance. Through multiple experiments, we have found that if we increase the weighting of the data on the content of the handled image (enhance the coefficient of the loss function of content), the final style of the transferred image can retain more of the content and structure of the original image while also retaining most of the style information. From our experiments, we found that the best ratio coefficient between the content image and style image is $10^{-2}$, as confirmed in a related paper.[18] The transfer speed is a critical factor for large-scale commercial deployment such as in mobile communication applications. We thus compared the transfer speed for our proposed algorithm and other mainstream methods for images of 256 × 256, 512 × 512, and 1024 × 1024 pixels, and the results for the batch transfer speed are shown in Table 1. From the table, we can see that our method is faster than the methods investigated by Gatys *et al.*[12] and Chen *et al.*,[9] but slower than the method investigated by Johnson *et al.*[8] The reason for

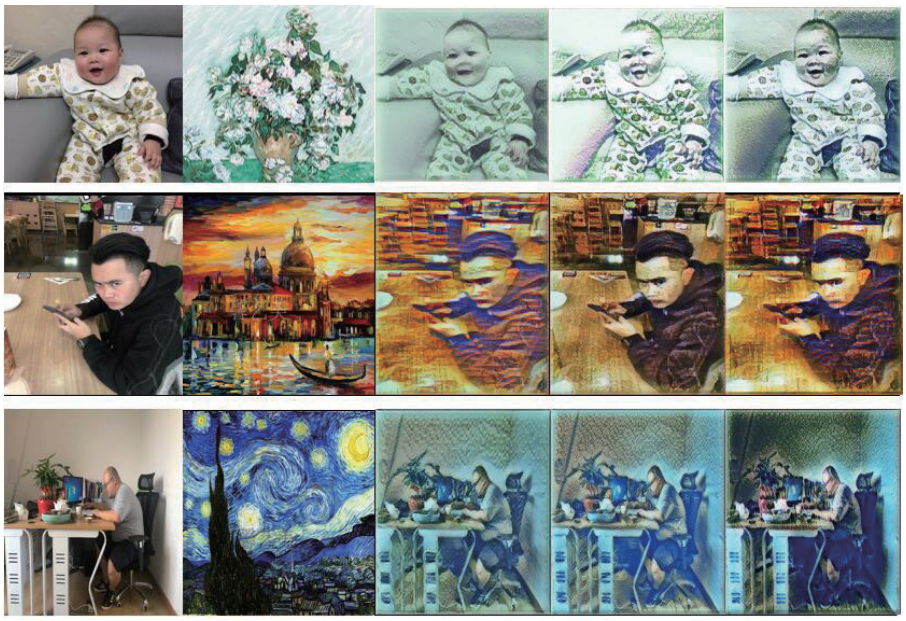

Fig. 4. (Color online) Different styles of image transfer.

Table 1
Comparison of batch transfer speeds.

| Method | Image size | | |
|---|---|---|---|
| | 1024 × 1024 (ms) | 512 × 512 (ms) | 256 × 256 (ms) |
| Gatys | 251.48 | 63.78 | 18.15 |
| Johnson | 20.11 | 5.82 | 1.83 |
| Chen | 30.21 | 8.57 | 2.35 |
| Ours | 26.50 | 7.69 | 2.92 |

the latter is that the style image in the method of Johnson *et al.* is pretrained, with some models generated in advance. However, this means that the method proposed by Johnson *et al.* can only deal with fixed images rather than arbitrary real-time images, where the latter can be easily handled by our method.

Our method of image transfer was used for different numbers of images of 25000, 50000, 100000, and 150000. As shown in Fig. 5, our method has a very good performance as compared with the other methods, consistent with the above results. Gatys *et al.* proposed a neural algorithm for artistic style that can separate and recombine the image content and style of natural images and produce new images with perceptual quality. However, the proposed algorithm needed a long time for its model's training. The model proposed by Johnson *et al.* combined feedforward CNNs using a per-pixel loss between the output and ground-truth images and a parallel work generated by defining and optimizing perceptual loss functions. Even though this model has the highest image transfer speed, it can only be used for fixed types of pictures. Chen *et al.* found that the responses at the final layer of DCNNs were not sufficiently

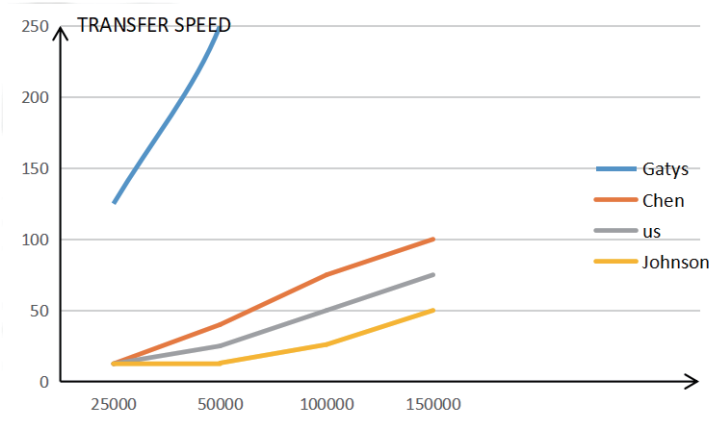


Fig. 5. (Color online) Transfer speed performance of image style transfer for different numbers of images.

localized for accurate object segmentation, so they used the CNNMRF method to optimize and localize segment boundaries at a specific level of accuracy. Our proposed algorithm was constructed on the basis of the deep residual networks proposed by He *et al.*[19] The framework of the presented DCNNs can reduce the training and processing required for local image style transfer, and it is substantially deeper than those used previously. Also, we have proven that our proposed method is faster than that proposed by Gatys *et al.* and can compete with those proposed by Johnson *et al.* and Chen *et al.*

## 5. Conclusions

To overcome the problems of Gatys' ground-breaking method of many iterations, excessive time required for the model training process, and excessive memory use, in this study, we introduced a residual network to solve the problems of training in deep network layers and to reduce the training burden of CNN operations. Through multiple experiments, we found that if we increased the weighting on the content of the handled image, the final style of the transferred image retained more of the content and structure of the original image as well as much of the style information. Our method is faster than the methods investigated by Gatys *et al.* and Chen *et al.* but it is slower than the method investigated by Johnson *et al.* However, the method of Johnson *et al.* can only deal with fixed images rather than arbitrary real-time images, which are easily handled by our method.

## Acknowledgments

This work was supported by projects under Nos. MOST 109-2622-E-390-001-CC3 and MOST 109-2221-E-390-023 and the Young and Middle-aged Teacher Program of the Education Department of Fujian (JAT200601).

## References


1 J. E. Kyprianidis, J. Collomosse, T. H. Wang, and T. Isenberg: IEEE Trans. Visual Comput. Graphics **19** (2013) 866.
2 Y. C. Shih, S. Paris, C. Barnes, W. T. Freeman, and F. Durand: ACM Trans. Graphics **33** (2014) 1.
3 Y. C. Jing, Y. Z. Yang, Z. L. Feng, J. W. Ye, Y. Z. Yu, and M. L. Song: IEEE Trans. Visual Comput. Graphics **6** (2019) 1.
4 A. Hertzmann: Comput. Graphics Inter. July (2001) 47.
5 A. Hertzmann: 1998 SIGGRAPH '98: Proc. 25th Annu. Conf. Computer Graphics and Interactive Techniques (1998) 453–460.
6 D. DeCarlo and A. Santella: ACM Trans. Graphics **21** (2002) 3.
7 L. A. Gatys, A. S. Ecker, and M. Bethge: Comput. Sci. **111** (2015) 98.
8 J. Johnson, A. Alahi, and F. F. Li: 14th European Conf. Computer Vision (ECEV 2016) 694.
9 L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille: 2018 IEEE/CVF Conf. Computer Vision and Pattern Recognition (2018) 4013.
10 A. J. Champandard: Con. 2016 Artificial Intelligence in Creative Industries (2016) arXiv:1603.01768.
11 J. Long, E. Shelhamer, and T. Darrell: IEEE Conf. Computer Vision and Pattern Recognition (CVPR, 2015) 3431.
12 L. A. Gatys, A. S. Ecker, and M. Bethge: IEEE Conf. Computer Vision and Pattern Recognition (CVPR, 2016) 2414.
13 L. Geng, S. G. Zhang, J. Tong, and Z. T. Xiao: Comput. Assisted Surg. **24** (2019) 27.
14 C. O. Kiselman: Sci. China Math. **60** (2017) 1005.
15 K. Simonyan and A. Zisserman: 3rd Inte. Conf. Learning Representations (ICLR 2015) arXiv:1409.1556.
16 A. AlShibli and H. Mathkour: Sensors **19** (2019) 4207
17 M. Gregor, R. Pirník, and D. Nemec: Transp. Res. Procedia **40** (2019) 1327.
18 X. Huang and S. Belongie: IEEE Int. Conf. Computer Vision (ICCV, 2017) 1501.
19 K. He, X. Zhang, S. Ren, and J. Sun: IEEE Conf. Computer Vision and Pattern Recognition (CVPR, 2016) 770.