Changing Video Game Graphic Styles Using Neural Algorithms

Byungho Yoo
Department of Computer Science and Engineering
Sejong University, Seoul, South Korea
bingo0080@gmail.com

Kyung-Joong Kim
Department of Computer Science and Engineering
Sejong University, Seoul, South Korea
kimkj@sejong.ac.kr

***Abstract*—Recently, procedural content generation (PCG) has attracted positive attentions from gamers and applied for various content types such as maps, items and so on. Deep neural networks have been reported that they have potential to learn styles of artistic images. In this study, we propose to apply convolutional neural networks to change artistic styles of video game graphics. It's expected to change original games into different styles (modern, old-fashioned, scientific, and so on) given the input images. We applied the neural styling algorithm to the game images from Hedgewars, an open-source turn-based strategy game. Our results show that styles of video games can be changed from an input styling image.**

Keywords—Procedural content generation, Convolutional neural network, Image conversion, Deep learning application

I. Introduction

Deep learning has been successfully applied to some game AI applications. Most of them focused on building AI players from reinforcement or supervised learning. In recent studies, they used the game screen as inputs to the AI player and the deep neural networks process the raw pixel information to select proper actions. Google's recent success on Go demonstrated the potential of deep reinforcement learning to solve very complex games.

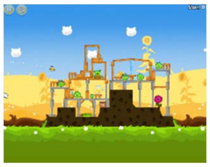
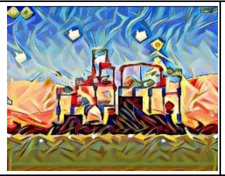
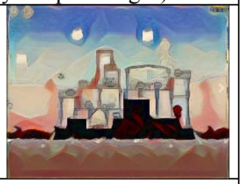
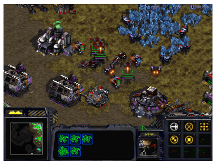
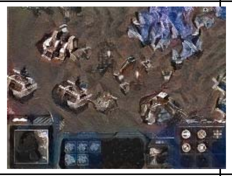
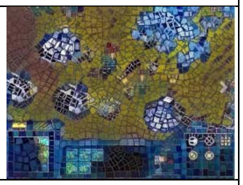
It has been known that the convolutional neural network (CNN) is not just powerful on image recognition but also useful for image styling and super resolution. Recently, Champandard applied deep learning to scale up and style pixel art for MineCraft textures [1].

In this study, we applied a neural styling algorithm to all game images from a single open-source game. Our research question is to see what makes difficult to use the neural styling for all aspects of games. Because game images have unique property separated from other graphics, artistic, or photo images, it's important to design a system suitable for the game graphics styling.

II. A Neural Algorithm of Artistic Style

Gatys *et al.*, used convolutional neural network to separate contents and styles of artistic images [2]. They demonstrated that the neural network can combine contents of arbitrary images with styles of well-known artists' works. They used Vincent van Gogh's The Starry Night for style representation and created photographs with the style. Because the source code was open to the public, there are web services to support the styling of small number of images [3].

Fig. 1 Applying neural styling algorithms to commercial game screen shots (it just converted a single screen image using https://dreamscopeapp.com/)

Screen shot	Outputs from Neural Styling Algorithm (with names of style input images)	
		
Angry Birds	Clayton Kashuba	Picasso Blue
		
StarCraft	Winter Solstice	Blue Mosaic

III. Proposed Systems and Experiments

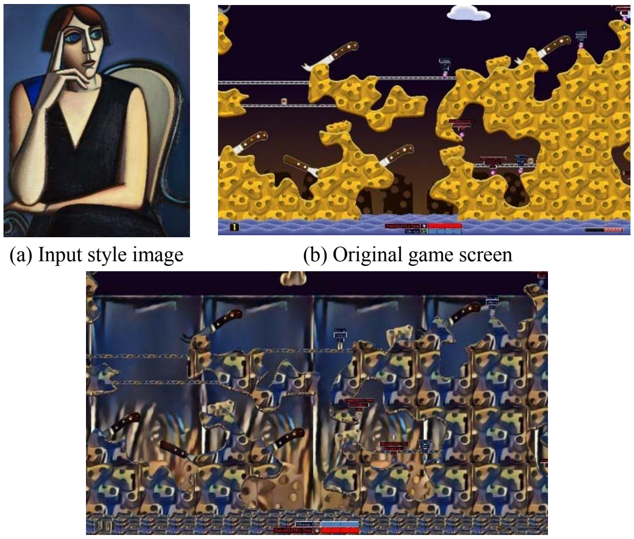
It needs high-end graphic cards to get high quality outcomes because it uses GPGPU. The higher outcome image resolution is necessary, the more memory space is required. For example, 7 GB of graphic card memory size is recommended to apply neural styling for an 1024 by 1024 pixels image. Even 800 x 800 images, it needs 5GB memory for the neural styling. Most of graphic card with GPUs support 2GB ~ 4GB memory. When the resolution exceeds the memory requirement, it's rescaled to affordable image size and processed by the neural styling algorithm. The outcome images are rescaled to the original input size. Because recent state-of-the art graphic cards provide up to 12GB memory size, it's possible to convert them without resizing with special hardware support. Table 1 summarizes the specification of our hardware systems. In our system, images larger than 800 by 800 or smaller than 64 by 64 are rescaled.

TABLE I. Hardware Sepcification used in Neural Styling

CPU	intel(R) Core(TM) I7-4790 3.60 Hz
RAM	16.0GB
GPU	Geforce GTX 980 6GB GDDR5

There are some parameters to be tuned for the styling. For example, content weight can control the loss of original content and important to adjust the level of change. It can balance between the style and content representation to produce combined outcomes.

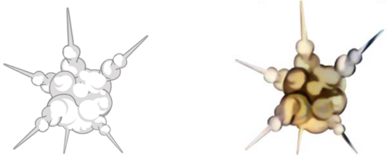
Fig. 2 Styling of game play screens using the input image and convolutional neural networks (two videos of the game playing is provided in the supplementary part.)



(a) Input style image (b) Original game screen

(c) Game screen processed by the neural styling algorithm

Fig. 3 Change of explosion effect by the neural styling



(a) Original (b) After styling

Hedgewars is an open-source turn-based strategy game (http://www.hedgewars.org/). When we select the target game for this experiment, we search for an open-source game with all the game images accessible in a widely acceptable file format such as JPEG or PNG. Some open-source games are excluded because parts of images are stored as resources not supported in neural styling.

In the game, the total size of all images files (formatted in JPEG and PNG) is 70MB. We applied the neural styling algorithm [4] to the images one by one and took about 12 hours. For the big size images, it included rescaling process and PNG files with alpha channel, it did reconstruction of the transparency channel after neural styling. PIL (Python Imaging Library) was used. Although the neural styling is promising, it's not yet realistic to do the conversion of all game images in real-time. Instead, the game designers prepare several sets of game images styled with different input images (modern, old-fashioned, scientific and so on) and allow users to select their preference when they play the game. If the styling is targeting only the small portion of all image files, it's realistic to apply them in real-time. For example, the conversion is applied only to the main character or small-sized items.

In this study, we just used a single input style image to convert all the game images. However, it's more desirable to use several styles to convert game images. For example, different styles can be used for background and foreground objects. Also, it's necessary to apply the styling selectively and some images are not suitable for the change. Images with letters and symbols can be damaged after the styling.

IV. CONCLUSIONS AND FUTURE WORKS

The outcomes show the potential of neural styling for video games. In this study, we just applied to a single game with one input style. There is still enough room for improvement for better use of the automatic styling in video games. We found that the neural styling has not yet fully supported all kinds of graphic images from games. We addressed issues to handle large size image files, real-time processing, alpha channel in PNG file format, and letter/symbols distortion. There are additional potential problems to be considered for wide use of this system. For example, they're as follows.

- **Content descriptions**: Assuming that certain game item description says it is a sword, but this algorithm transmutes the item design into something else which is not seen as a sword.
- **Paired contents**: Some contents are paired each other such as a key and locked door. Because they're designed to be connected, it's important to keep the contents in their original position.
- **Continuous motions**: Animation of characters are coming from lots of frame images. It's not fully tested that the neural styling can produce smooth transition of similar frames.

ACKNOWLEDGEMENTS

This work was supported by the National Research Foundation of Korea (NRF) grant (2013 R1A2A2A01016589), National program for Excellence in Software program (R7718-16-1005), Ministry of Culture, Sports and Tourism (MCST) and Korea Creative Content Agency (KOCCA) in the Culture Technology (CT) Research & Development Program 2016.

SUPPLEMENTARY VIDEOS

(Before) https://www.youtube.com/watch?v=odbl6llQFjs
(After) https://www.youtube.com/watch?v=XW1uMpm1cRc

REFERENCES

[1] nucl.ai. (2016). *Minecraft, ENHANCE! Neural Networks to Upscale & Stylize Pixel Art*. [online] Available at: https://nucl.ai/blog/enhance-pixel-art/ [Accessed 7 Jul. 2016].

[2] Gatys, L., Ecker, A. and Bethge, M. (2015). *A Neural Algorithm of Artistic Style*. [online] Arxiv.org. Available at: http://arxiv.org/abs/1508.06576 [Accessed 4 May 2016]

[3] Deepart.io. (2016). [online] Available at: https://deepart.io/hire/ [Accessed 7 Jul. 2016]

[4] GitHub. (2016). dmlc/mxnet. [online] Available at: https://github.com/dmlc/mxnet/tree/master/example/neural-style [Accessed 31 May 2016.]