浙江大学

生物智能与算法 课程报告
（2018-2019 春）

题　　目　神经网络组 图像的风格化绘制

姓　　名　胡一夫

学　　号　21821237

学科、 专业　计算机科学与技术

任 课 教 师　袁昕

EMAIL　if@zju.edu.cn

1 选题介绍

1.1 论文出处

题目：Image Style Transfer Using Convolutional Neural Networks
发表会议：2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

1.2 论文简介

本文构建了一个基于深度卷积神经网络的智能系统，绘制出具有艺术价值的图像。对于任意图片，该系统将使用神经元表征对其风格和内容进行分离。通过将不同艺术作品的风格特征融合到普通照片的内容特征，该方法可以对普通照片进行各式各样的艺术化的生成。

文章的方法中使用了已训练好的 VGG19 网络。VGG19 本来是用于物体识别的，但可认为其中编码了高级的“content”信息。而之前有研究显示在神经网络上定义的 Gram 矩阵可以用于提取图像的 texture。把两者结合，即可实现内容和风格的融合。在 VGG19 的结构中选择 content layer 和 style layer（论文分别使用 conv4_2 和 conv1_1、conv2_1、conv3_1、conv4_1、conv5_1 的集合）之后，计算内容图的 content 激活值以及风格图的 style 激活值，并使用这些激活值作为特征，构建损失函数。最后采用反向传播算法，优化损失函数，不断迭代，最后得到结果。

2 图像风格化绘制方法介绍

2.1 输入

风格图、内容图、初始图。一个已经训练好的用于 object recognition 的网络（VGG19）。 风格图大小任意。内容图和初始图大小需要一致（实际可以使用内容图作为初始图作为迭代初始值）。

2.2 Gram 矩阵

在神经网络任意层的输出上，都可以计算出 Gram 矩阵。从下述的计算方法可以看出，Gram 是一个对称矩阵。文章认为，这个矩阵可以用于表示图片的风格特征。

$$G_{ij}^{l} = \sum_{k} F_{ik}^{l} F_{jk}^{l}$$

式中，F 是神经网络 l 层的输出。

以本文采用的风格层中的 conv5_1 层为例。假设风格图的大小是 h*w*3 的。那么当它到达第五组卷积层时，大小为(h/16)*(w/16)*512。该组卷积层拥有 512 个过滤器(每个过滤器大小是 3*3*512)，因此风格图在本层的原始激活值是一个(h/16)*(w/16)*512 的矩阵。这个矩阵的转置乘以自己，变成对称阵。这个对称阵是 512*512 的。它就是 Gram 矩阵，即风格特征。

可以看出，无论风格图的大小如何，在给定层，它提取出的特征的维度都是固定的。因此，输入的风格图的大小是任意的。

2.3 损失函数定义

损失函数定义如下（其中 α 和 β 是可配置常数，分别表示内容和风格的比重）：

$$\mathcal{L}_{total} = \alpha \mathcal{L}_{content} + \beta \mathcal{L}_{style}$$

其中的 content 损失函数项定义如下（l 表示某一层 content layer；F 是迭代图在本层的输出；P 是内容图在本层的输出）：

$$\mathcal{L}_{\text{content}}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} \left(F_{ij}^{l} - P_{ij}^{l}\right)^2$$

而 style 损失函数项定义如下（L 即为 style layer 的集合）：

$$\mathcal{L}_{\text{style}}(\vec{a}, \vec{x}) = \sum_{l=0}^{L} w_l E_l$$

对于每层 style layer，计算的 loss 如下（G 是迭代图在本层的 Gram 矩阵；A 是风格图在本层的 Gram 矩阵；N_l 和 M_l 分别是本层的过滤器个数以及每个过滤器产生的 feature map 中的元素个数）：

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} \left(G_{ij}^l - A_{ij}^l\right)^2$$

3 方法复现

对论文的复现结果展示如下。图片依次为：内容图、风格图、迭代 1、2、3、10、20、100、300、500、1000、2000、4000、6000 次后的图片，共 14 张。

图 1 内容图

图 2 风格图

图 3 迭代次数为 1

图 4 迭代次数为 2

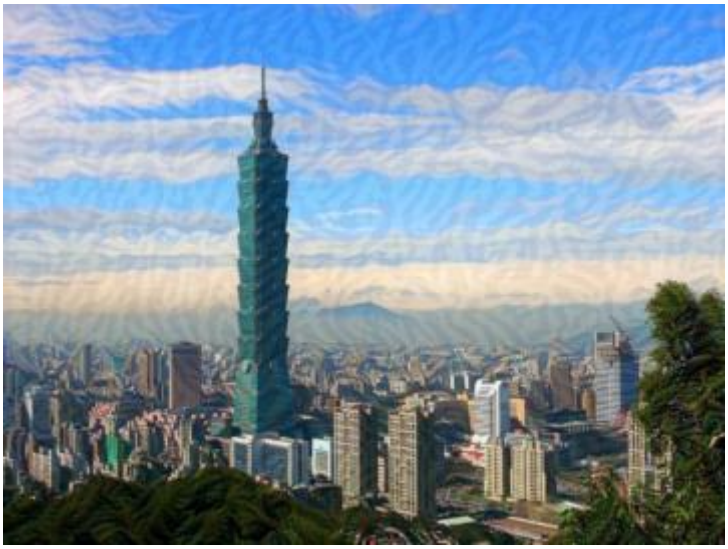

图 5 迭代次数为 3

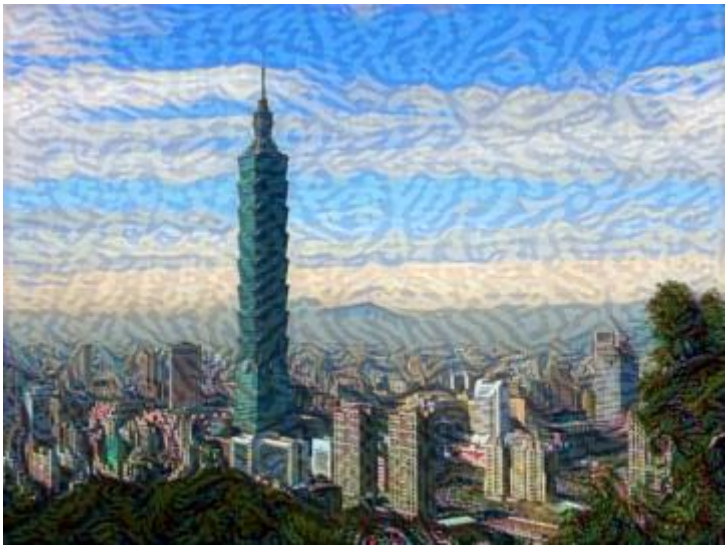

图 6　迭代次数为 10

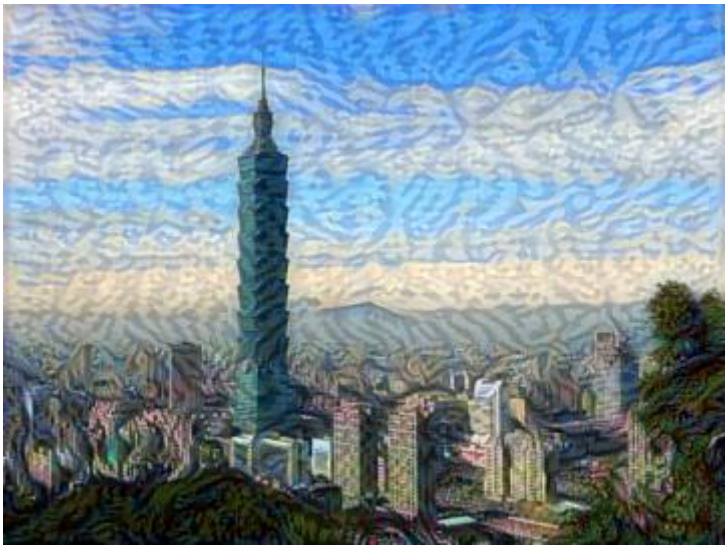

图 7 迭代次数为 20

图 8 迭代次数为 100

图 9 迭代次数为 300

图 10 迭代次数为 500

图 11 迭代次数为 1000

图 12 迭代次数为 2000

图 13 迭代次数为 4000

图 14 迭代次数为 6000

4 模型仿真

4.1 输入图片变化过程

以下展示了内容图和它迭代 10、200、1000、3000、5000、8000 后的变化，以及风格图。共 8 张图。

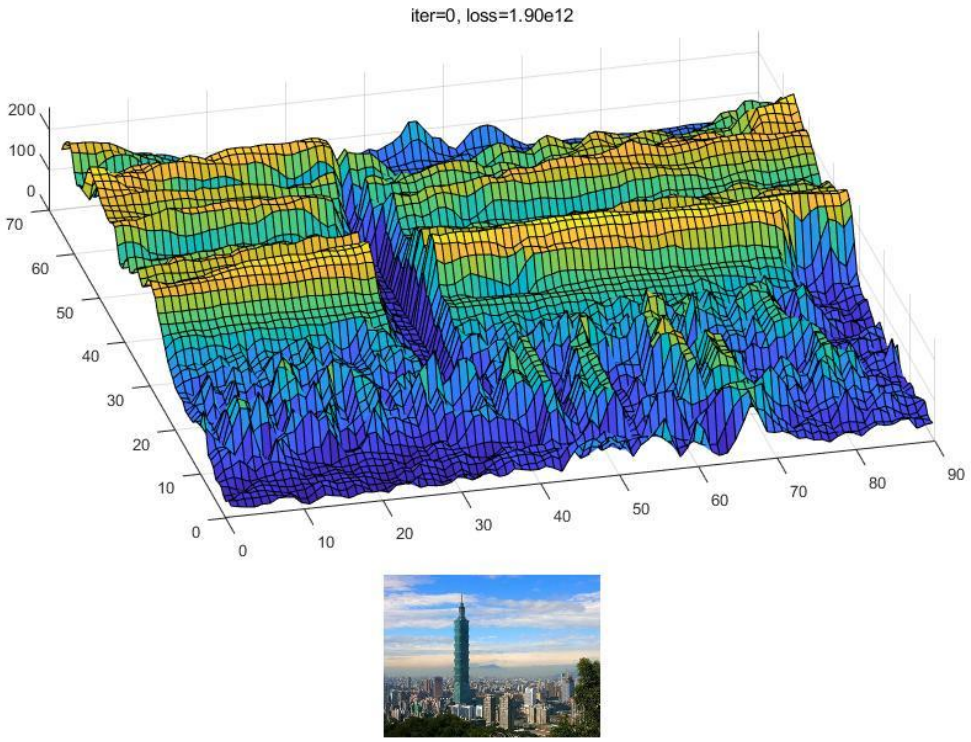

图 15 内容图仿真

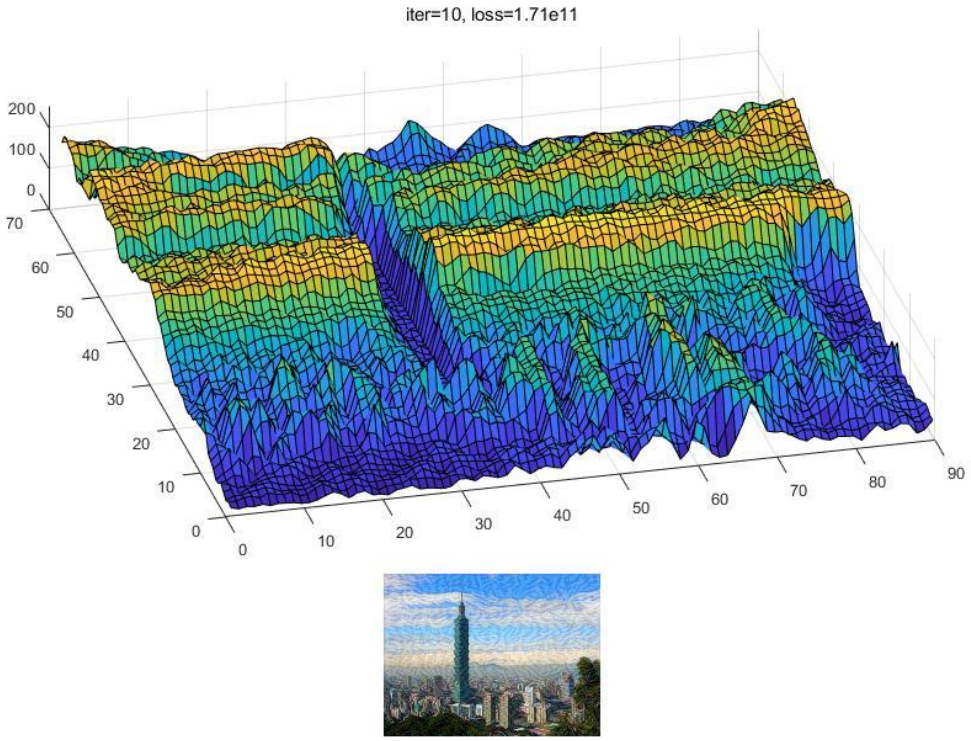

图 16 迭代 10 次仿真图

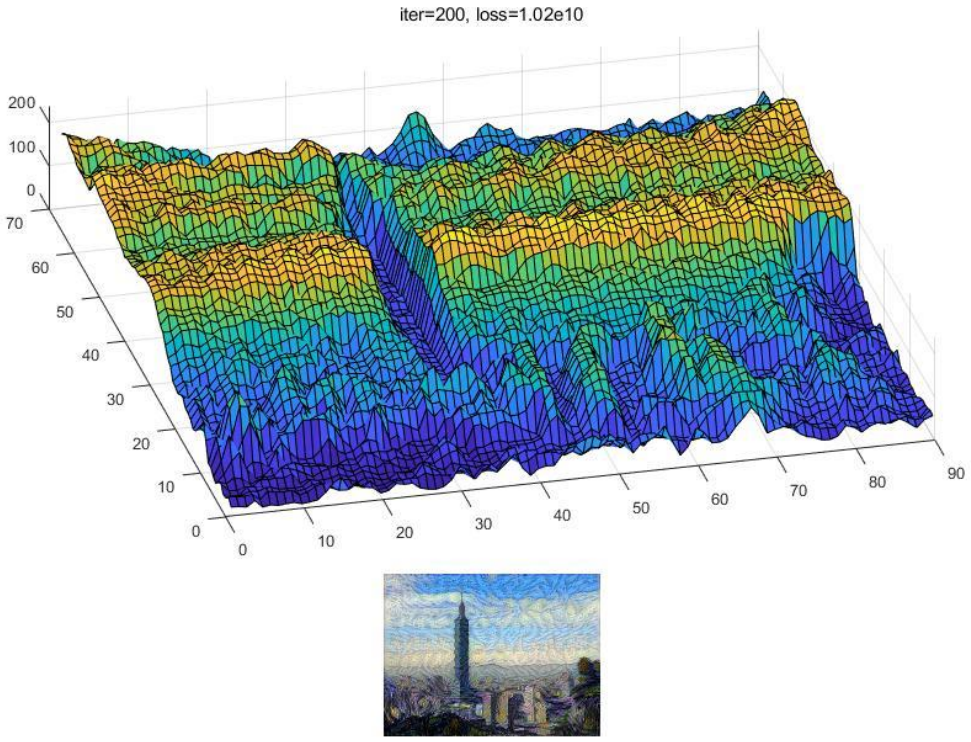

图 17 迭代 200 次仿真图

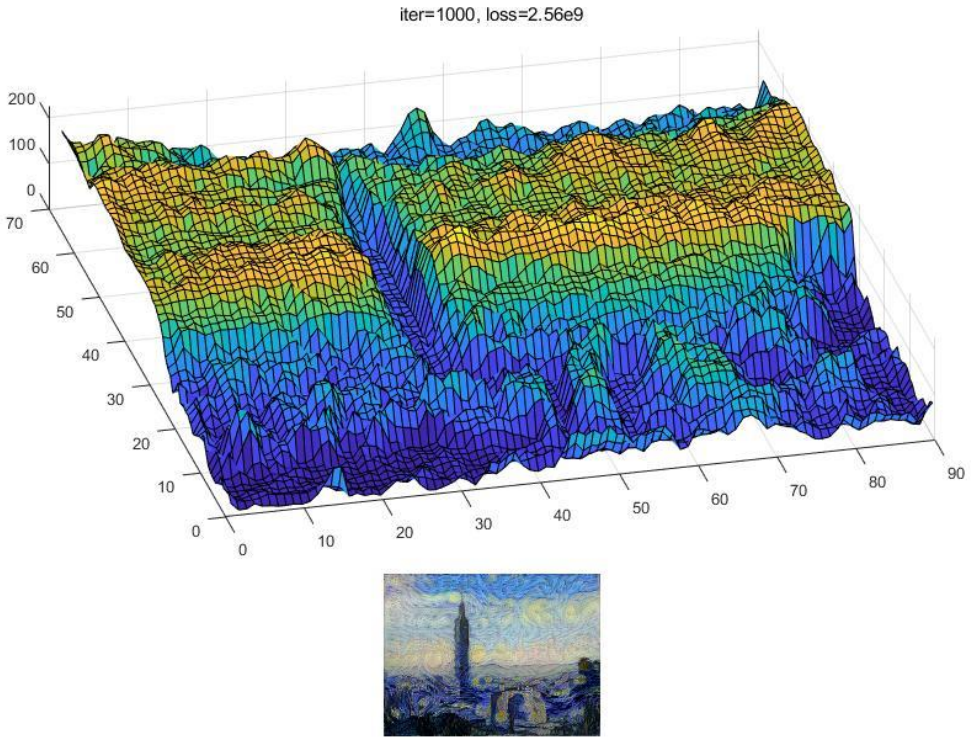

图 18 迭代 1000 次仿真图

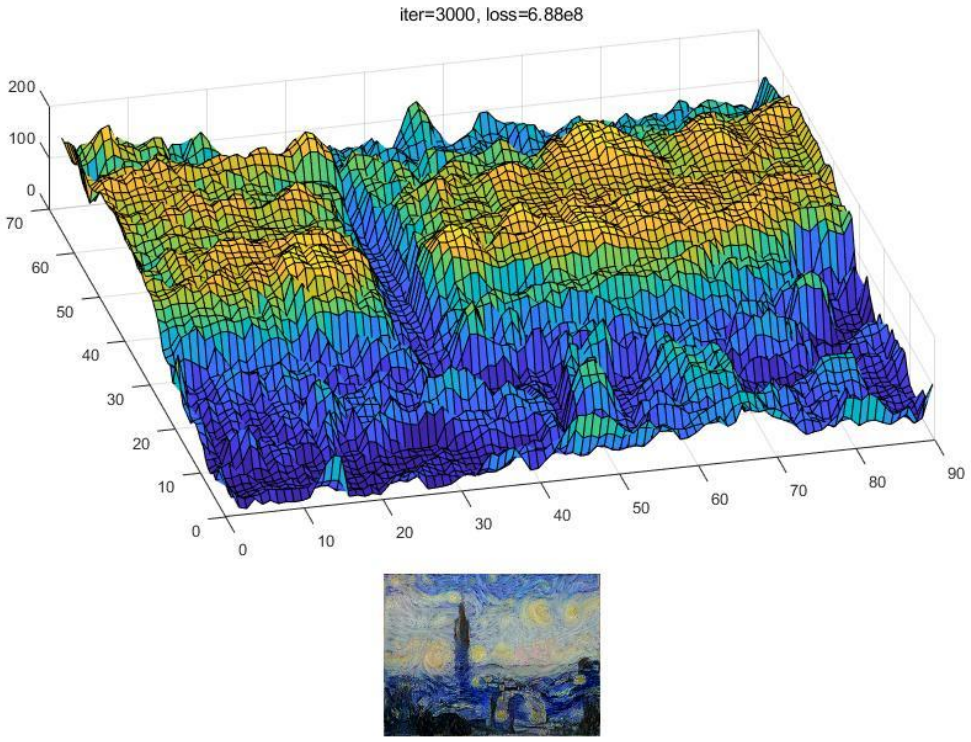

图 19 迭代 3000 次仿真图

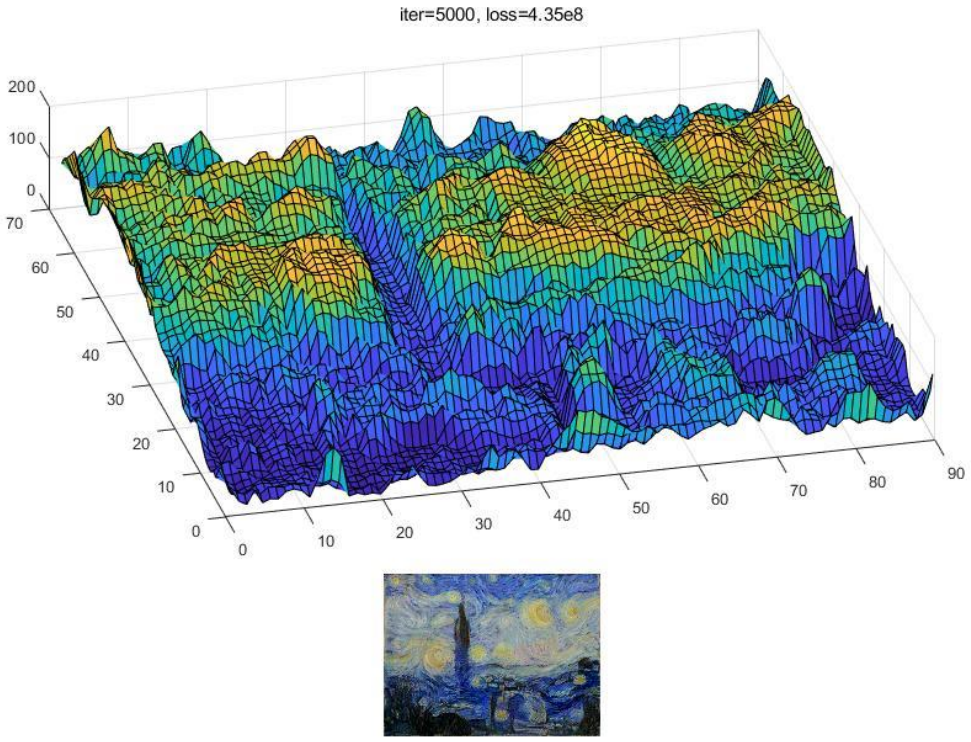

图 20 迭代 5000 次仿真图

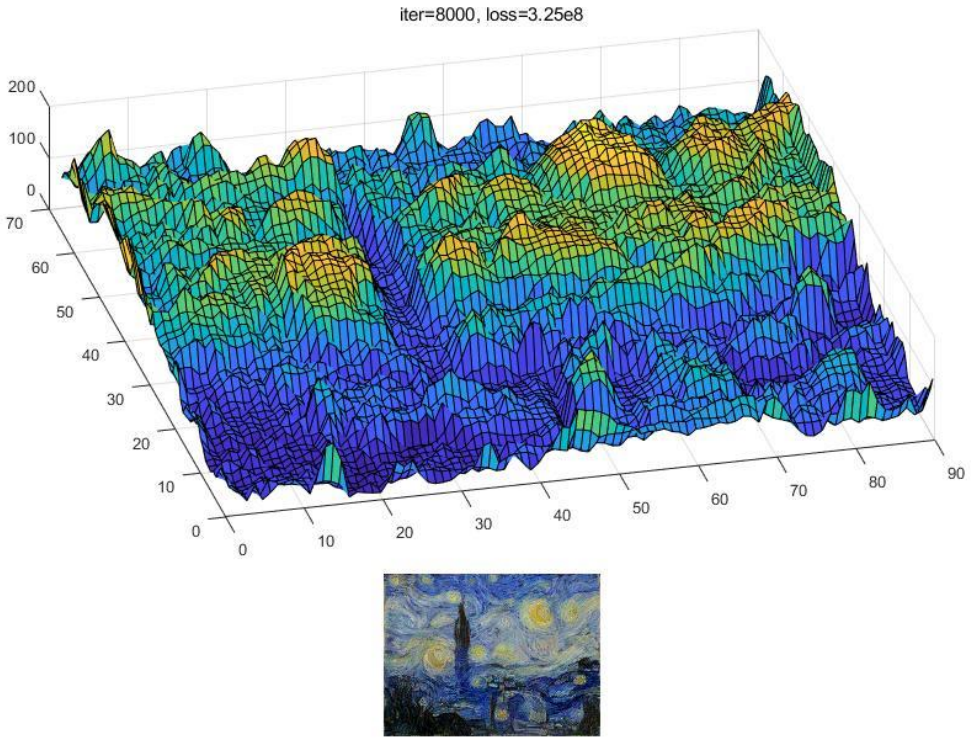

图 21 迭代 8000 次仿真图

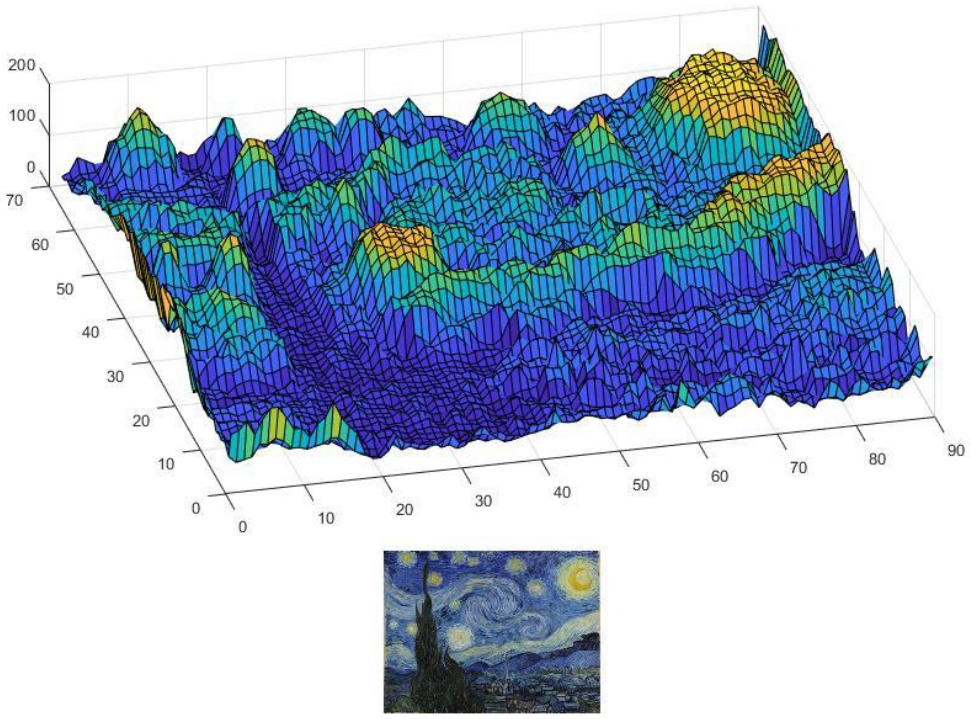

图 22 风格图仿真

4.2 损失函数曲面图

共挑选了两对输入来绘制损失函数曲面图。由于本文的损失函数由 content 和 style 两项损失线性加权而得，因此对于每对输入，均绘制了总损失曲面、内容损失曲面、风格损失曲面三个曲面。从曲面图中可以看出，损失函数和损失函数的两项都是凸函数，适用凸优化技巧，可迭代至全局最优解。

4.2.1 第一对输入结果图

总损失：

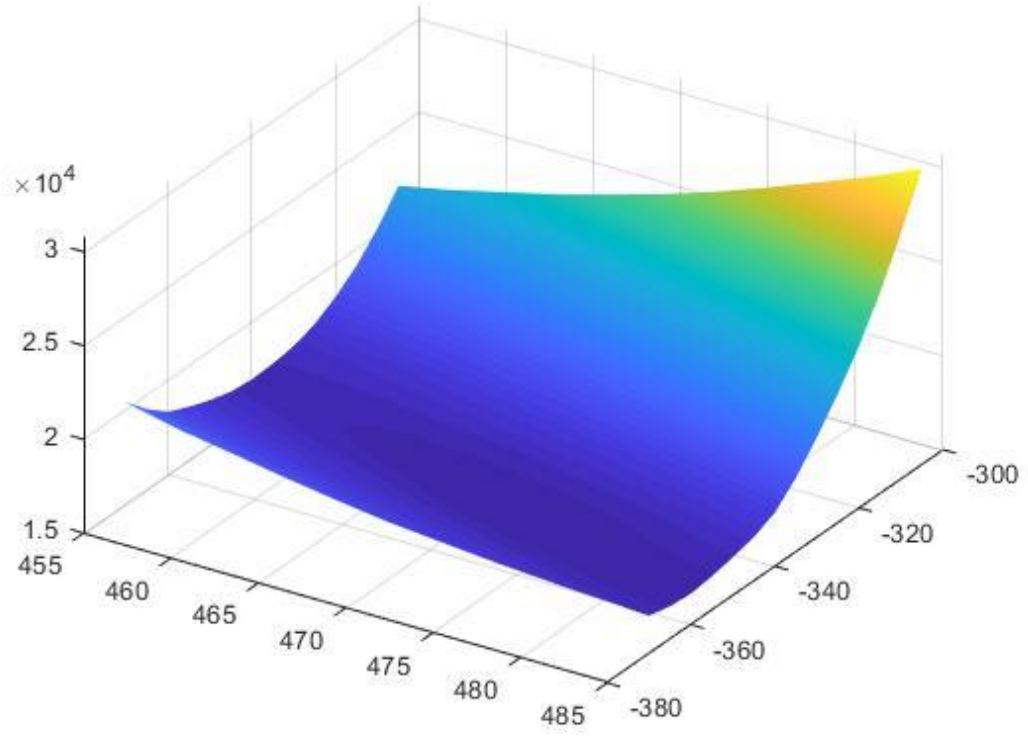

图 23 总损失函数关于第一对输入的曲面

内容损失：

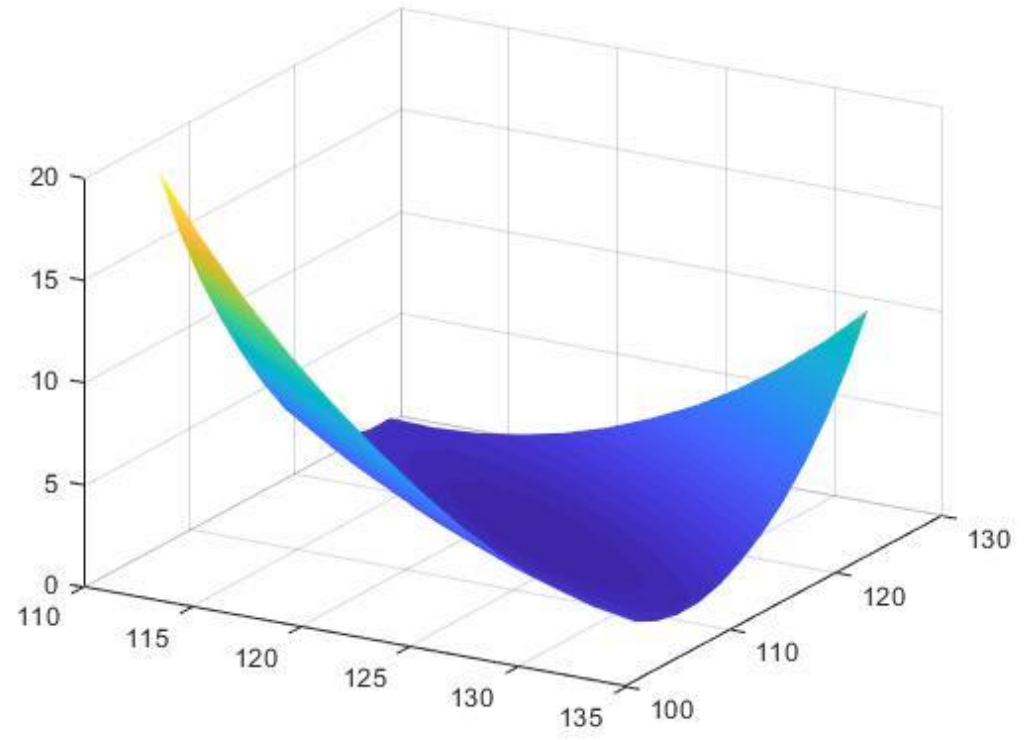

图 24 内容损失函数关于第一对输入的曲面

风格损失：

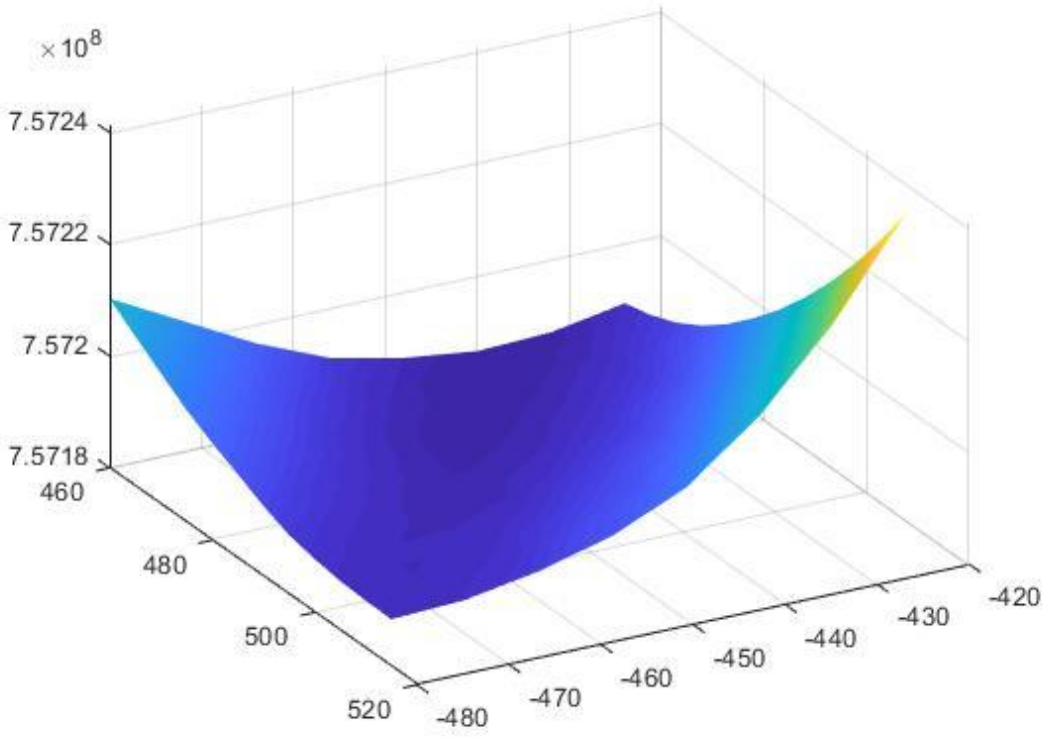

图 25 风格损失函数关于第一对输入的曲面

4.2.2 第二对输入结果图

总损失：

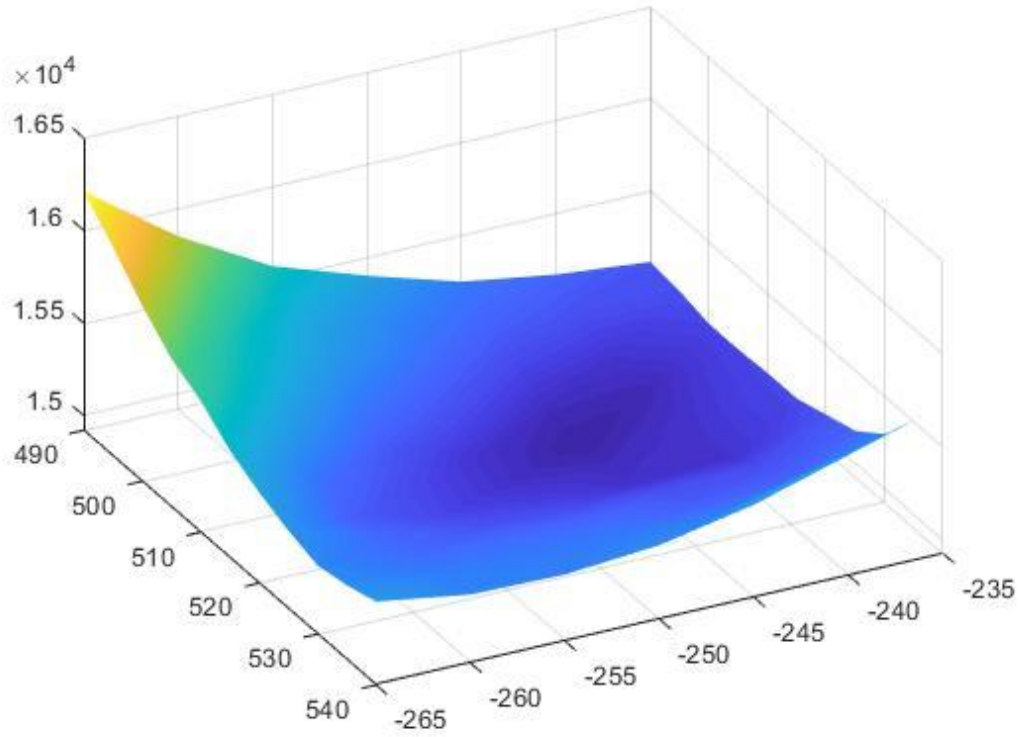

图 26 总损失函数关于第二对输入的曲面

内容损失：

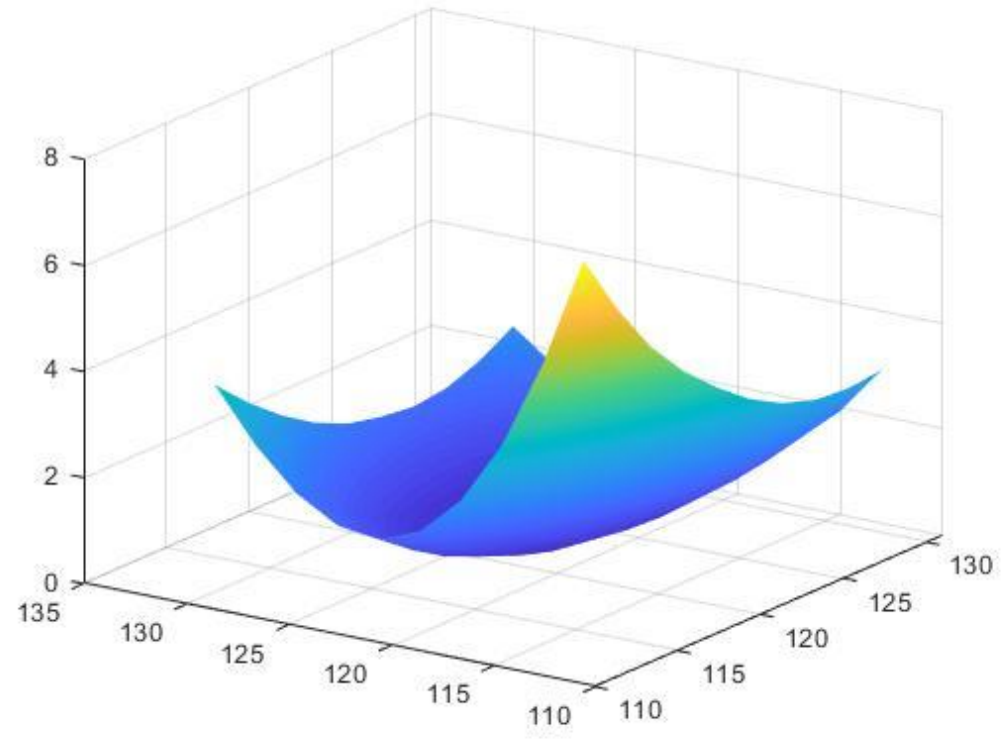

图 27 内容损失函数关于第二对输入的曲面

风格损失：

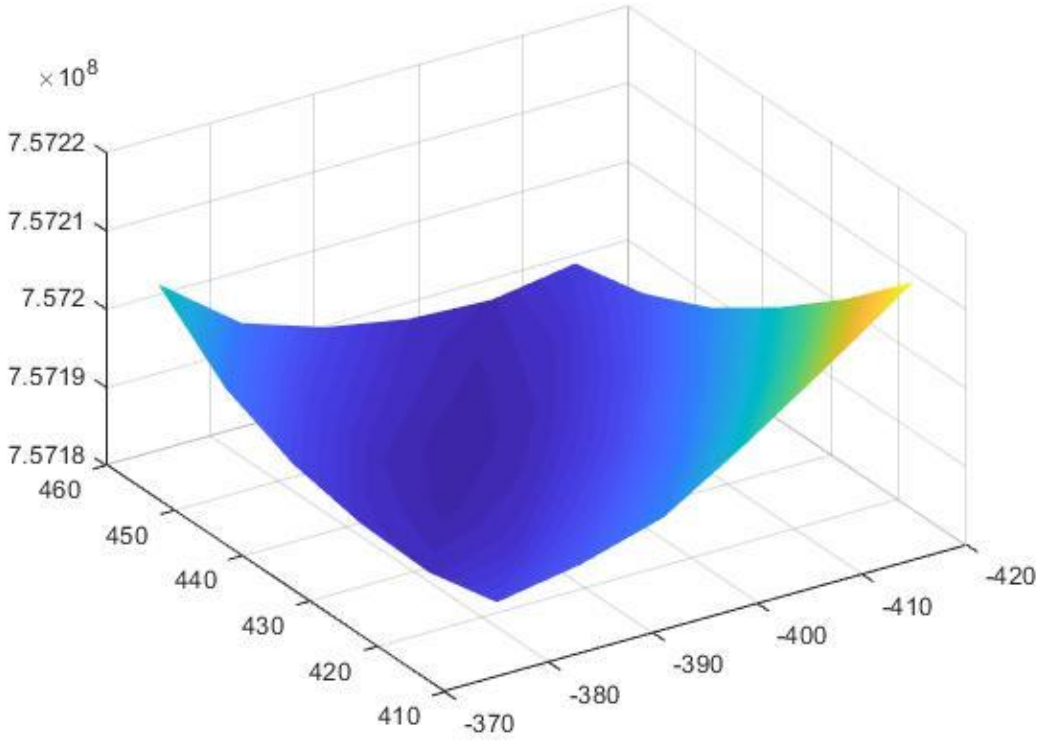

图 28 风格损失函数关于第二对输入的曲面

5 结果讨论

所谓的风格化绘制，实际上是一个纹理生成的过程，它并不具备认知画作表达的语义的能力。它根据每个像素点，以及这个点和它周围的点的相关关系（这个相关关系由神经网络的 feature map 捕捉），调整这个像素点的值，使之在一个局部范围内，具有与风格图的局部特征相似的纹理。可认为是在内容图和风格图之间进行插值。从损失函数的曲面中可以看出，整个优化过程是凸优化。