# Neural Style Transfer for Light Fields

**Dónal Egan, MASt, BA.**

## A Dissertation

Presented to the University of Dublin, Trinity College

in partial fulfilment of the requirements for the degree of

**Master of Science in Computer Science (Augmented and Virtual Reality)**

Supervisors: Dr. Martin Alain and Prof. Aljosa Smolic

September 2020

# Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Dónal Egan

September 7, 2020

# Permission to Lend and/or Copy

I, the undersigned, agree that Trinity College Library may lend or copy this thesis upon request.

Dónal Egan

September 7, 2020

# Acknowledgments

I would like to thank Dr. Martin Alain for all the time and help that he has given me over the past year, answering all of my questions and making many helpful suggestions to guide me.

I would also like to thank Prof. Aljosa Smolic for his guidance throughout this project.

Most importantly, thank you to my parents who always support me in everything I do.

Dónal Egan

*University of Dublin, Trinity College*

*September 2020*

# Neural Style Transfer for Light Fields

Dónal Egan, Master of Science in Computer Science

University of Dublin, Trinity College, 2020

Supervisors: Dr. Martin Alain and Prof. Aljosa Smolic

Style transfer involves combining the style of one image with the content of another to form a new image. Unlike traditional two-dimensional images which only capture the spatial intensity of light rays, four-dimensional light fields also capture the angular direction of the light rays. Stylizing a light field requires us to not only render convincing style transfer for each sub-aperture image, but also to preserve the angular structure of the light field. The naïve approach to stylizing a light field is to simply stylize each sub-aperture image independently. Unsurprisingly, doing so will destroy the light field's angular structure. We present our new method for light field style transfer which significantly outperforms this naïve approach. It uses our new initialisation method and angular loss function for the image-optimisation process to preserve the angular structure of the light field. We also present an architecture for a depth-aware approach to light field style transfer which uses a depth loss function to preserve the angular structure of the light field during the stylization process.

# Contents

# List of Figures

# Chapter 1

# Introduction

The goal of this project was to apply style transfer to four-dimensional light fields. In this dissertation we present our new approach to light field style transfer which significantly outperforms naïve baseline methods.

## 1.1 Overview of project

Style transfer involves combining the content of one image (for example, a photograph) with the style of another (for example, a painting) to form a new image. Doing so requires us to be able to define, separate and extract the content and style of an image. *Neural style transfer* uses the outputs from the hidden layers of deep convolutional neural networks pre-trained for object recognition to construct content and style representations of an image. These content and style representations are used to define content and style loss functions. Using these loss functions, a new image can be synthesised to match the content of one image and the style of another. This is done via an image-optimisation process. The new image is

initialised to be white noise. It is then iteratively updated using gradient descent so as to simultaneously minimise the content loss between it and the content image and the style loss between it and the style image. Impressive results using neural style transfer have already been achieved, for example, in [9, 18, 21].

The goal of this project is to apply neural style transfer to light fields. Unlike traditional two-dimensional images which capture the spatial intensity of light rays, light fields are four-dimensional and also capture the angular direction of the light rays. A light field can be visualised as a two-dimensional array of two-dimensional images. Each image of the array, called a *sub-aperture image*, captures the scene from a different view point. Applying style transfer to a light field not only requires us to render aesthetically pleasing style transfer for each sub-aperture image of the image array, but also to preserve the angular consistency of the light field. In other words, the stylization should be consistent across all of the sub-aperture images. Inconsistencies in the angular structure following the stylization process cause a flickering effect when the stylized light field is displayed.

Unsurprisingly, naïvely stylizing each sub-aperture image independently does not preserve the angular structure of the light field. This results from the image-optimisation process converging to different local minima for each sub-aperture image. In this project we present our new method for light field style transfer which significantly outperforms the naïve approach. Our new method preserves the angular consistency of the light field through the introduction of a new initialisation method for the image-optimisation process and a new angular loss function. The basic idea is to propagate the style outwards from the central view point of the light field while always ensuring that angular consistency with previously stylized view points is preserved. The new initialisation method ensures that points common to

multiple light field view points are initialised with the desired appearance. This is achieved by warping previously stylized view points according to the optical flow between them and the next view points to be stylized. The angular loss function, also constructed using the optical flow between view points, penalises inconsistencies between stylized view points.

We evaluate our new light field style transfer method according to three criteria. First, we subjectively evaluate the aesthetic quality of the stylized light fields. Second, we use epipolar plane images to examine the angular consistency of the stylized light fields. Third, we use light field depth estimation to examine the depth structure of the stylized light fields. We show that for all three criteria our method significantly outperforms the naïve approach of stylizing each sub-aperture image independently.

We also present an architecture for a depth-aware approach to light field style transfer. This approach differs from the above method in that the entire light field is processed in one go by a transformation network. This transformation network is a fully-convolutional neural network. It takes a light field as input and outputs the stylized light field. The angular structure of the light field is preserved during the stylization process through the introduction of a depth loss function. This depth loss function is used to preserve the depth structure of the original light field during the stylization process. It is defined using a pre-trained light field depth estimation network which is used estimate the depth maps of the original and stylized light fields. While we have not yet achieved results with this depth-aware method to match the quality of the results from our previously described method, the preliminary results achieved indicate the potential of our depth-aware approach to achieve high quality light field style transfer.

## 1.2 Structure of dissertation

The rest of this work is structured as follows: Chapter 2 provides a summary of the background research carried out for the project. It provides a summary of light field imaging and a summary of existing style transfer techniques for two-dimensional images and for videos. Chapter 3 describes the different light field style transfer methods that we developed and tested, i.e. it describes our contribution to the field. We also provide implementation details for our methods. Chapter 4 provides results and an evaluation of the methods described in chapter 3. Finally, in chapter 5 we conclude the project. We provide a summary of our contribution, discuss the limitations of our work and provide suggestions for future work in the field of light field style transfer.

# Chapter 2

# Background Research

The goal of this project is to apply style transfer to light fields. This chapter provides a summary of the background research carried out as part of the project. Section 2.1 provides a summary of light field imaging, while section 2.2 provides a summary of existing style transfer techniques for traditional two-dimensional images and videos.

## 2.1 Light fields

Light field imaging was introduced to computer graphics in the papers *Light Field Rendering* by Levoy and Hanrahan [20] and *The Lumigraph* by Gortler et al. [10]. Unlike traditional two-dimensional images which only capture the spatial intensity of light rays, four-dimensional light fields also capture the angular direction of the light rays.

Light fields are based on the five-dimensional plenoptic function $L(x, y, z, \phi, \theta)$ which captures all possible light rays at every point in space (the $x, y$ and $z$ di-

mensions) and in every direction (the $\phi$ and $\theta$ dimensions) [2]. However, by only considering a region of space free of occluders (the *free space assumption*) one of the five dimensions becomes redundant. This follows from the fact that the radiance of a light ray remains constant along a straight line. Hence, we are left with a simplified four-dimensional plenoptic function which we call a light field.

### 2.1.1 Light field representations

#### Two-plane parameterisation

A common way to represent the four-dimensional light field is using the two-plane parameterisation. Here, a ray of light $L(s, t, u, v)$ is parameterised by its intersection with two parallel planes, namely the $st$-plane and the $uv$-plane (figure 2.1). We refer to $s$ and $t$ as the *angular dimensions* and $u$ and $v$ as the *spatial dimensions*.

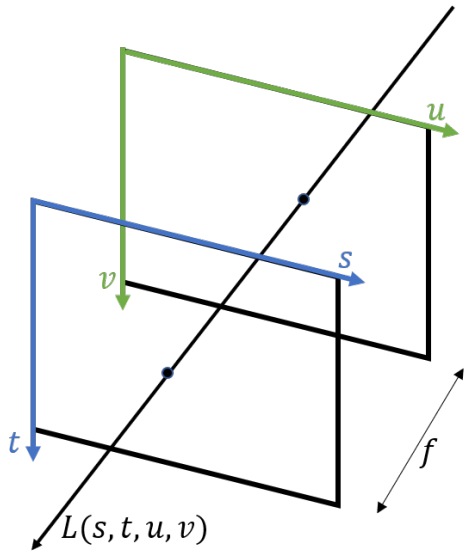
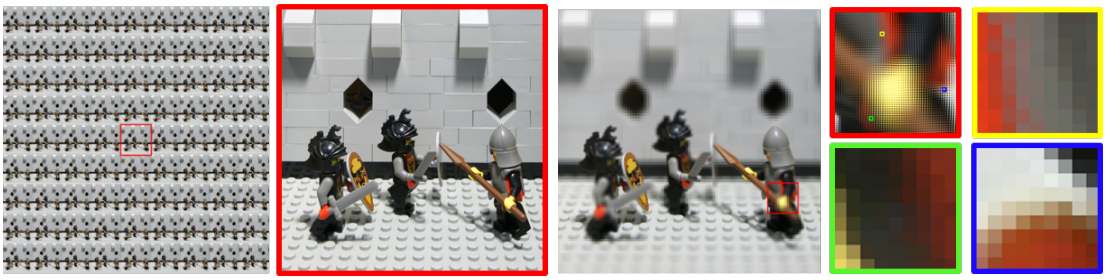



Figure 2.1: Two-plane light field parameterisation. The ray of light $L(s, t, u, v)$ first intersects the $uv$-plane and then the $st$-plane. The distance between the two planes is $f$.

Using this parameterisation, we can consider the $st$-plane as a set of cameras with their focal plane on the $uv$-plane. It follows that we can visualise the four-dimensional light field as a two-dimensional array of two-dimensional images. There are two possibilities for this: an $st$-array of $uv$-images or a $uv$-array of $st$-images (figure 2.2).



(a) (b)

Figure 2.2: (a) An $st$-array of $uv$-images: Each *sub-aperture image* captures the scene from a different view point. (b) A $uv$-array of $st$-images: Each sub-image of the array shows a point $(u^*, v^*)$ as seen from the different view points (green, yellow and blue boxes are examples).

When viewing the light field as an $st$-array of $uv$-images (figure 2.2a), each two-dimensional image of the array captures the scene from a different view point. Each image is called a *sub-aperture image* and is obtained by fixing the angular coordinates $s$ and $t$ for some values $s^*$ and $t^*$. The resolution of the $st$-plane, called the *angular resolution*, determines the number of sub-aperture images in the captured light field, while the resolution of the $uv$-plane, called the *spatial resolution*, determines the quality of each sub-aperture image. In general, the angular resolution of the light field will be significantly lower than the spatial resolution. For example, the light field shown in figure 2.2 has an angular resolution of $9 \times 9$ and a spatial resolution of $1024 \times 1024$.

Alternatively, we can view the light field as a $uv$-array of $st$-images (figure

2.2b). Here, a sub-image in the array is obtained by fixing the spatial coordinates $u$ and $v$ for some values $u^*$ and $v^*$. The sub-image then shows the point $(u^*, v^*)$ as seen from the different light field view points.

We adopt the two-plane parameterisation and visualise a light field as an $st$-array of $uv$-images (figure 2.2a) for the rest of this project.

### Epipolar plane images

Another useful way to visualise light fields is using *epipolar plane images* (EPIs). An epipolar plane image is a two-dimensional slice of a light field obtained by fixing one angular dimension and one spatial dimension. The horizontal EPI $E_{t^*v^*}(s, u)$ is obtained by fixing the angular dimension $t$ and the spatial dimension $v$ for some values $t^*$ and $v^*$, respectively. The vertical EPI $E_{s^*u^*}(t, v)$ is obtained in a similar fashion.

An epipolar plane image appears as a series of lines of varying slopes. This is illustrated in figure 2.3. The slopes of the lines reflect the depth of the scene captured by the light field. This follows from the result that depth is inversely proportional to disparity. For example, consider the horizontal EPI $E_{t^*v^*}(s, u)$ and a point in the captured light field with depth $Z$. As the angular coordinate $s$ is varied, the spatial coordinate $u$ varies according to the equation

$$\Delta u = \frac{f}{Z}\Delta s \tag{2.1}$$

where $f$ is the distance between the $st$ and $uv$ planes.

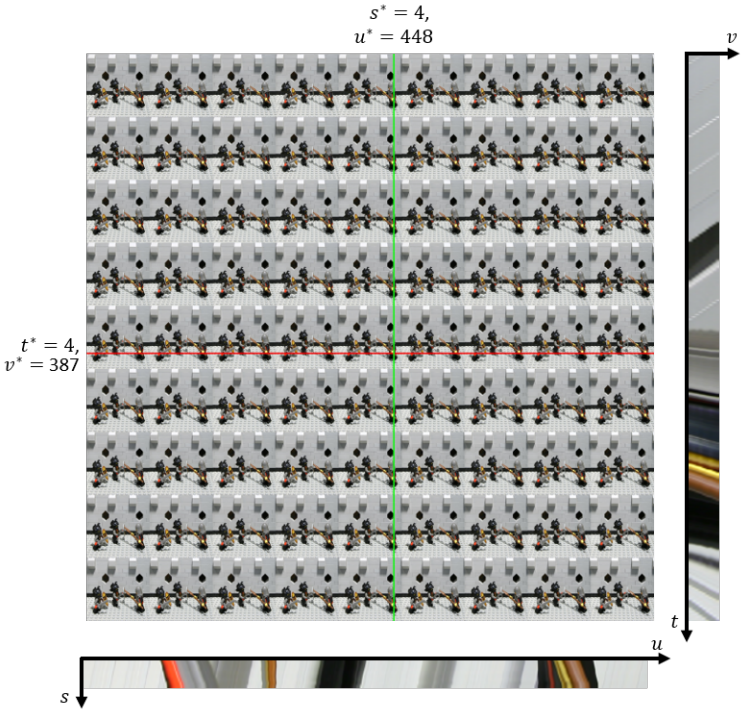



Figure 2.3: Two examples of epipolar plane images (EPIs). The horizontal EPI $E_{t^*v^*}(s, u)$ is obtained by fixing $t$ and $v$, while the vertical EPI $E_{s^*u^*}(t, v)$ is obtained by fixed $s$ and $u$

Epipolar plane images are a useful tool for visualising the angular structure of a light field. Later on, we will them to visualize the angular structure of stylized light fields and to evaluate how well different style transfer methods preserve the angular consistency of the original light field.

### 2.1.2 Light field depth estimation

Depth estimation is an important light field application. A light field image implicitly captures information about the depth structure of the scene. By rearranging equation 2.1, the depth $Z$ can be obtained by estimating the disparity $\Delta u$.

Traditional stereo-matching methods can be used to estimate the disparity $\Delta u$

from two light field sub-aperture images. However, such methods do not take full advantage of the light field structure. Multi-view stereo matching approaches use all of the sub-aperture images to estimate disparity (for example, see [16]).

Other light field depth estimation techniques use epipolar plane images. For example, in [27] the authors estimate depth by estimating the slopes of the lines in the epipolar plane images.

Deep learning approaches have also been proposed for light field depth estimation [11, 12, 24]. Later on we will use EPINET [24]. It is a fully-convolutional neural network that uses the light field epipolar geometry for depth estimation. The network architecture consists of two parts - a multi-stream input network and a merging network (figure 2.4). The multi-stream network independently processes four stacks of sub-aperture images with different angular directions, encoding the epipolar geometry of each. The merging network processes the combined outputs from the four streams to produce the estimated depth map.

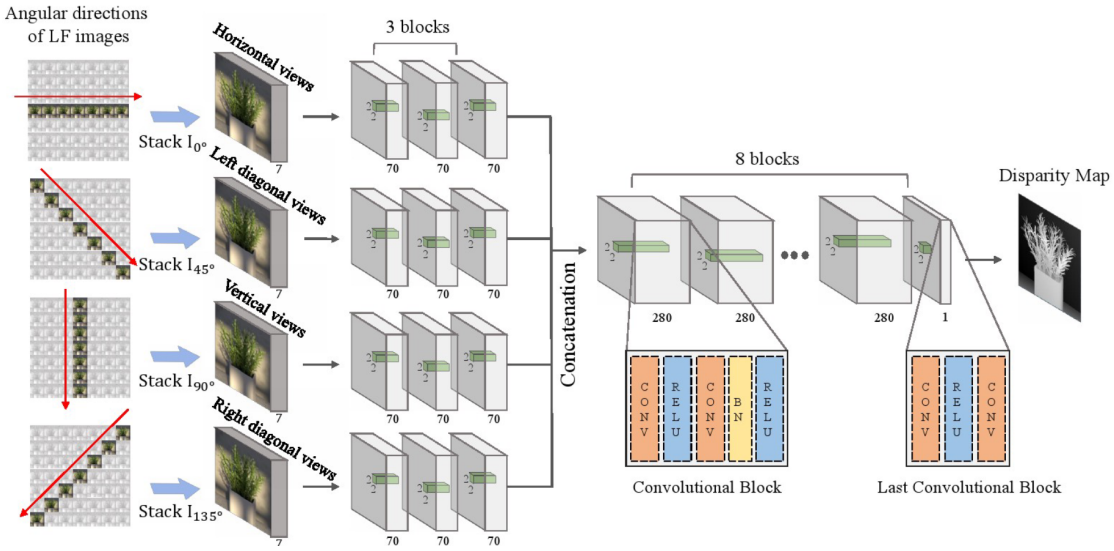



Figure 2.4: EPINET is a fully-convolutional newural network used for light field depth estimation. Figure taken from [24].

### 2.1.3 Light field datasets

For this project we use light fields from two different sources, namely the HCI 4D Light Field Dataset [13] and the (New) Stanford Light Field Archive [1]. The HCI 4D Light Field Dataset is a synthetic dataset containing twenty-four synthetic light fields along with their ground truth depth maps. The (New) Stanford Light Field Archive contains real light fields captured using using either a camera array or a gantry.

## 2.2 Style transfer

Originally developed for traditional two-dimensional images, style transfer involves combining the content of one image (for example, a photograph) with the style of another (for example, a painting) to form a new image (figure 2.5). If the content and the style of an image are represented by the functions $c()$ and $s()$, respectively, then the problem of style transfer is as follows:

*Given a content image* $\mathbf{p}$ *and a style image* $\mathbf{a}$*, construct a new image* $\mathbf{x}$ *such that*

$$c(\mathbf{x}) = c(\mathbf{p}) \text{ and } s(\mathbf{x}) = s(\mathbf{a}).$$

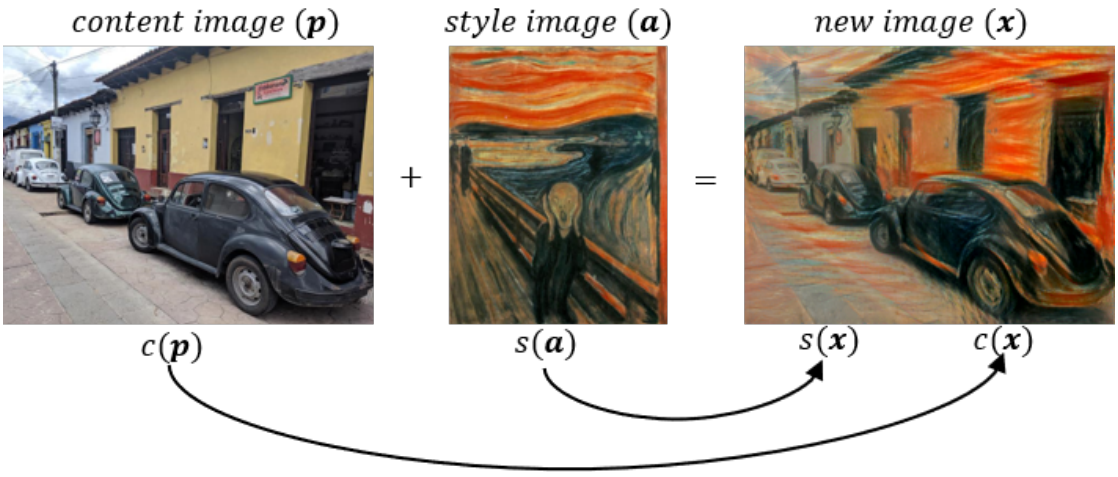



Figure 2.5: Style transfer - the new image $\mathbf{x}$ has the content of $\mathbf{p}$ and the style of $\mathbf{a}$.

To achieve this we need to be able to define, separate and extract the content and style of an image. Defining the style of an image is an arbitrary task. Moreover, it is arguable whether or not the content and style of an image are separable at all. It follows that style transfer is not a well-defined problem with a single correct solution. This makes evaluating style transfer a difficult task.

Style transfer methods divide into two categories - those which do not use deep learning and those which do use deep learning. Methods in the latter category are collectively referred to as *neural style transfer*. Neural style transfer uses convolutional neural networks pre-trained for object recognition to extract the content and style of an input image. While non-deep learning approaches can produce impressive results (see [7, 15], for example), the focus of this project is on neural style transfer. The following sections summarise some of the major breakthroughs in neural style transfer.

### 2.2.1 Image-optimisation approach

Neural style transfer was introduced by Gatys et al. in their paper *Image Style Transfer Using Convolutional Neural Networks* [9]. They showed that the outputs from the hidden layers of a convolutional neural network pre-trained for object recognition could be used to construct content and style representations of an input image.

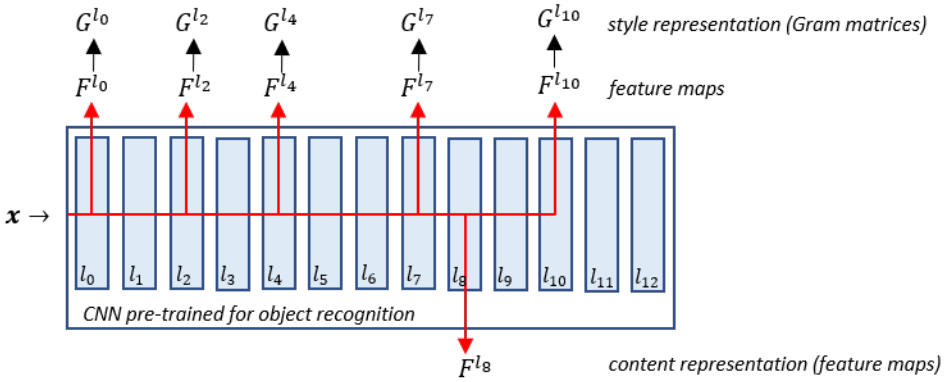



Figure 2.6: At each layer of the CNN, an input image **x** is encoded as a set of feature maps. These feature maps can be used to construct content and style representations of the image.

As an input image passes through the network, it is encoded at each layer as a set of feature maps (figure 2.6). If layer $l$ of the network has $N_l$ filters, then an input image $\mathbf{x}$ will be encoded at layer $l$ as a set of $N_l$ distinct feature maps, each of size $M_l$, say. Thus, the feature response of the network at layer $l$ to an input image $\mathbf{x}$ can be represented by a matrix $F^l \in \mathbb{R}^{N_l \times M_l}$ where $F^l_{i,j}$ is the activation of the $i^{th}$ filter at position $j$ in layer $l$.

### Content representation

Earlier network layers extract features which are more concerned with the specific pixel values. In contrast, the features extracted by the deeper network layers capture the high-level content of the input image in terms of the objects present and their arrangement in the image. Gatys et al. use the feature responses from one of these deeper layers to represent the content of an image. Thus, the content of an image is represented by a matrix $F^l \in \mathbb{R}^{N_l \times M_l}$ for some layer $l$ of the network.

### Style representation

The outputs from several network layers are used to construct a style representation for an input image. This style representation uses correlations between feature responses. The correlations are given by the *Gram* matrices $G^l \in \mathbb{R}^{N_l \times N_l}$ where $G^l_{i,j}$ is the inner product of the feature maps $i$ and $j$ for layer $l$ of the network, that is

$$G^l_{i,j} = \sum_k F^l_{i,k} F^l_{j,k}. \tag{2.2}$$

Thus, the style of an image is represented by a set $\{G^l \in \mathbb{R}^{N_l \times N_l} : l \in L\}$ of Gram matrices for some set $L$ of network layers.

**Loss functions**

These content and style representations are used to define content and style loss functions. Both losses are squared error losses. Let $\mathbf{p}$ be the content image, $\mathbf{a}$ be the style image and $\mathbf{x}$ be the new image that we wish to generate. Let $P^l$ and $F^l$ be the content representations of $\mathbf{p}$ and $\mathbf{x}$, i.e. their feature representations in some layer $l$ of the network. The *content loss* is defined as

$$\mathcal{L}_{content}(\mathbf{p}, \mathbf{x}) = \frac{1}{2} \sum_{i,j} (F^l_{i,j} - P^l_{i,j})^2. \tag{2.3}$$

Let $\{A^l : l \in L\}$ and $\{G^l : l \in L\}$ be the style representations of $\mathbf{a}$ and $\mathbf{x}$. The *style loss* is defined as

$$\mathcal{L}_{style}(\mathbf{a}, \mathbf{x}) = \sum_{l \in L} w_l E_l \tag{2.4}$$

where

$$E_l = \frac{1}{N_l^2 M_l^2} \sum_{i,j} (G^l_{ij} - A^l_{ij})^2. \tag{2.5}$$

and where the weights $w_l$ regulate the contribution of each layer to the total style loss (typical values are $w_l = \frac{1}{|L|}$). The content and style losses are combined to form the *total loss* function

$$\mathcal{L}_{total}(\mathbf{p}, \mathbf{a}, \mathbf{x}) = \alpha \mathcal{L}_{content}(\mathbf{p}, \mathbf{x}) + \beta \mathcal{L}_{style}(\mathbf{a}, \mathbf{x}). \tag{2.6}$$

where the weights $\alpha$ and $\beta$ regulate how much emphasis is placed on reconstructing the content or the style during the stylization process.

### Image optimisation

To transfer the style of an image **a** (for example, an artwork) onto the content of another image **p** (for example, a photograph), a new image **x** that simultaneously matches the content representation of **p** and the style representation of **a** is synthesised. This is achieved by iteratively updating the new image **x**, initialised to be white noise, so as to minimise the total loss $\mathcal{L}_{total}(\mathbf{p}, \mathbf{a}, \mathbf{x})$ (equation 2.6). The image **x** is updated via gradient descent using the partial derivatives $\frac{\partial \mathcal{L}_{total}}{\partial \mathbf{x}}$ which are calculated using back-propagation.

It is important to note that no neural network is being trained as part of this process. Rather, a pre-trained network is being used and the image **x** is being iteratively updated so that it produces the same feature responses as the content image **p** at a certain layer of the network and the same feature correlations as the style image **a** at certain layers of the network (figure 2.7).

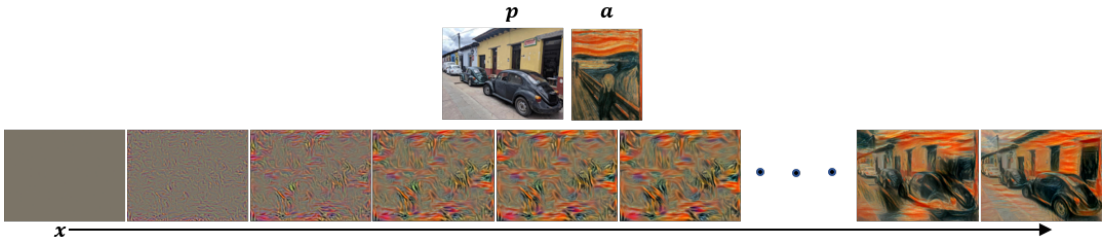



Figure 2.7: Initialised to be white noise, the new image **x** is iteratively updated using gradient descent until it simultaneously matches the content of **p** and the style of **a**.

Gatys et al.'s approach produces aesthetically pleasing results (figure 2.5). The main advantage of their method is its flexibility - there are no restrictions on the content and style images used as inputs. The main disadvantage is that it is very slow. Many forward and backward passes through the network are required to

stylize a single image.

### 2.2.2 Model-optimisation approach

As mentioned, the main disadvantage of the image-optimisation approach to style transfer is the large computational cost. In [18], Johnson et al. address this problem by training a feed-forward image-transformation network to approximate the solution to the image-optimization problem. A content image is then stylised by a single forward-pass through the image-transformation network.

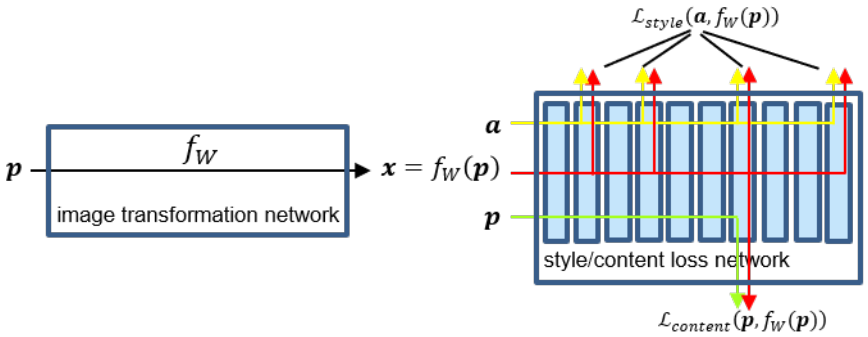



Figure 2.8: The system consists of two networks. The image transformation $f_W$ network stylizes a content image $\mathbf{p}$ with a single forward pass. The loss network is a CNN pre-trained for object recognition and is used to calculate the content and style losses during the training of $f_W$.

Johnson et al.'s system consists of two networks - an image transformation network and a loss network (figure 2.8). As with the the image-optimisation approach, the loss network is a convolutional neural network pre-trained for object recognition. It is used to calculate the content and style losses when training the image transformation network.

The image transformation network $f_W$ is a fully convolutional neural network. It transforms an input image $\mathbf{p}$ into an output input image $\mathbf{x} = f_W(\mathbf{p})$. It is param-

eterised by weights $W$. Training is carried out on the image transformation network using gradient descent so as to minimise the expected loss $\mathcal{L}_{total}\left(\mathbf{p}, \mathbf{a}, f_W(\mathbf{P})\right)$ for an arbitrary input image $\mathbf{p}$ and the chosen style image $\mathbf{a}$, i.e. to find network weights $W^*$ such that

$$W^* = \arg\min_W \mathbb{E}_{\mathbf{p}}\left[\mathcal{L}_{total}(\mathbf{p}, \mathbf{a}, f_W(\mathbf{p}))\right] \tag{2.7}$$

where the loss function $\mathcal{L}_{total}$ is as defined in equation 2.6. Thus, during training, a content image $\mathbf{p}$ is passed through the image-transformation network. The output $f_W(\mathbf{p})$ is then passed through the loss network and its content and style representations are extracted. The content loss $\mathcal{L}_{content}(\mathbf{p}, f_W(\mathbf{p}))$ (equation 2.3) and the style loss $\mathcal{L}_{style}(\mathbf{a}, f_W(\mathbf{p}))$ (equation 2.4) are calculated. Using these losses, the weights $W$ are updated accordingly using gradient descent. Once training is complete, an arbitrary content image can by stylized with a single pass through the image-transformation network.

This model-optimisation approach produces results of a similar quality to the image-optimisation approach. The main advantage is the significantly improved speed with only a single forward pass through the image transformation network required to stylize an image. The disadvantage of the model-optimisation approach is the reduced flexibility arising from the requirement to train an image-transformation network for each style.

### 2.2.3 Depth-aware style transfer

Depth-aware style transfer is an extension of the model-optimization approach described above and was introduced by Lui and Lai in [21]. It aims to preserve the

depth structure of the input content image in the stylized output image through the introduction of a depth loss function. The authors claim that this produces more desirable style transfer.

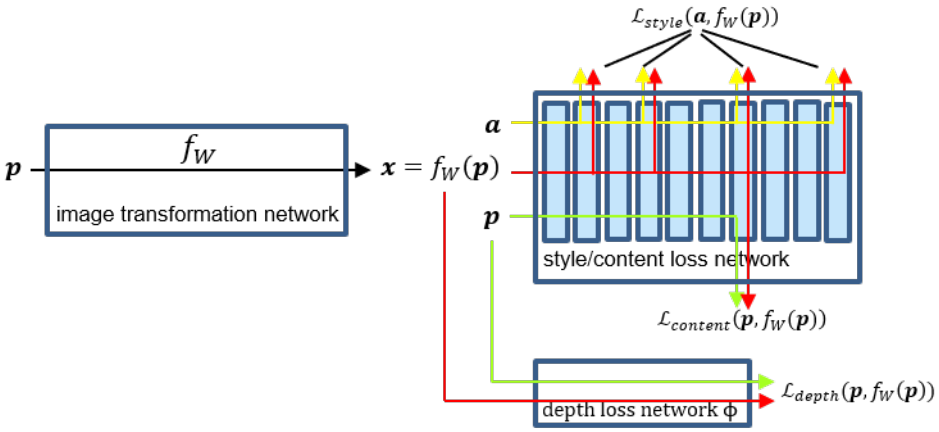



Figure 2.9: A pre-trained depth estimation network is added to the architecture and used to define a depth loss function.

Figure 2.9 illustrates the system architecture. It is the same as that described in the previous section except for the addition of a pre-trained depth estimation network $\phi$. This depth estimation network takes an image $\mathbf{x}$ as input and outputs an estimate $\phi(\mathbf{x})$ of its depth map. It is used to define the depth loss function which is the pixel-wise mean squared error between the estimated depth maps of the original content image $\mathbf{p}$ and the stylized image $f_W(\mathbf{p})$, i.e.

$$\mathcal{L}_{depth}(\mathbf{p}, f_W(\mathbf{p})) = \frac{1}{HW} \sum_{i,j} (\phi(\mathbf{p})_{ij} - \phi(f_W(\mathbf{p}))_{ij})^2 \tag{2.8}$$

where the summation is pixel-wise. Thus, the total loss function to be minimised

when training the image transformation network is now

$$\begin{aligned}\mathcal{L}_{total}(\mathbf{p}, \mathbf{a}, f_W(\mathbf{p})) = \alpha\mathcal{L}_{content}(\mathbf{p}, f_W(\mathbf{p})) + \beta\mathcal{L}_{style}(\mathbf{a}, f_W(\mathbf{p})) \\ + \gamma\mathcal{L}_{depth}(\mathbf{p}, f_W(\mathbf{p}))\end{aligned} \tag{2.9}$$

for some weights $\alpha, \beta$ and $\gamma$.

Figure 2.10 illustrates example style transfer outputs without the depth loss (i.e. the approach of section 2.2.2) and with the depth loss (i.e. the approach of this section). Using the depth loss clearly results in the spatial structure of the content image being better preserved during the stylization process. Arguably, this results in more aesthetically pleasing style transfer.

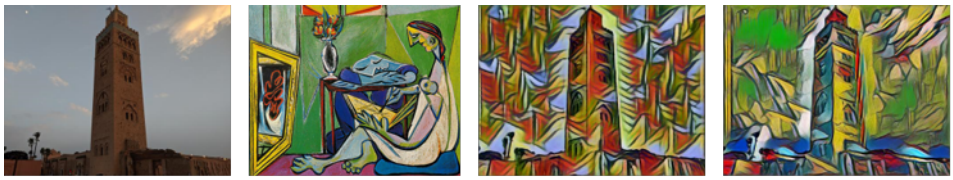



Figure 2.10: Comparing the impact of the depth loss. The spatial structure of the content image is better preserved by the stylization process when the depth loss function is used.

### 2.2.4 Style transfer for videos

Ruder et al. look at style transfer for videos in [22, 23]. Video style transfer involves transferring the style from a single style image (for example, a painting) to an entire video sequence. If the stylized video is to be aesthetically pleasing, the style transfer must be smooth and consistent between consecutive video frames.

Stylizing each video frame independently using the image-optimisation approach of section 2.2.1 leads to flickering and inconsistencies between the stylized

video frames (see middle row of figure 2.11). Even two video frames which appear very similar are stylized very differently. This occurs as the optimisation process converges to very different local minima.

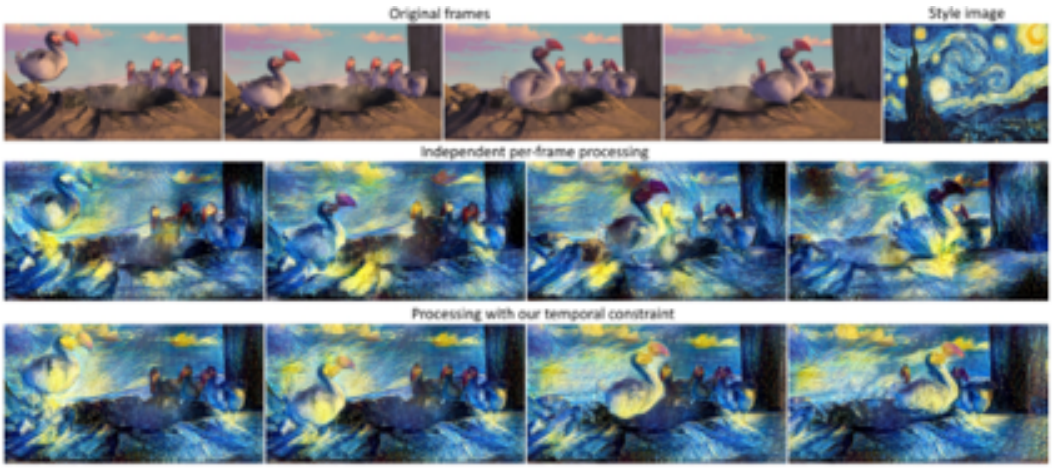



Figure 2.11: Video style transfer: Top row: original video frames and style image. Middle row: Independent stylization of video frames. Bottom row: Stylized frames using solution of Ruder et al. (Image taken from [23])

Ruder et al. solve this problem by building on the image-optimisation approach of section 2.2.1 through the introduction of a new initialisation method for the optimisation process and a new temporal loss function. Their method significantly improves on the naïve approach of stylizing each video frame independently (see bottom row of figure 2.11).

### A new initialisation

Let $\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_n$ be the original video frames, $\mathbf{a}$ be the style image and $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_n$ be the stylized video frames that are to be generated. For the first frame, $\mathbf{x}_0$ is still initialised to be white noise. For any subsequent frame, $\mathbf{x}_i$ is initialised to be $\omega^i_{i-1}(\mathbf{x}_{i-1})$ where $\omega^i_{i-1}$ is the function that warps a given image according to

the optical flow between the original video frames $\mathbf{p}_{i-1}$ and $\mathbf{p}_i$. This means that any points common to both frames $i-1$ and $i$ are initialised with the desired appearance in $\mathbf{x}_i$.

### A temporal loss function

A temporal consistency loss function penalises deviations between two consecutive stylized video frames. For frame $i$, where $i > 0$, the temporal consistency loss is defined to be the pixel-wise sum:

$$\mathcal{L}_{temporal}(\mathbf{x}_i, \mathbf{x}_{i-1}) = \sum \mathbf{c}_{i-1}^{i} \cdot \left(\mathbf{x}_i - \omega_{i-1}^{i}(\mathbf{x}_{i-1})\right)^2 \tag{2.10}$$

where $\mathbf{c}_{i-1}^{i}$ are per-pixel weights defined to be 0 for disoccluded regions and motion boundaries between the original video frames $\mathbf{p}_{i-1}$ and $\mathbf{p}_i$ and 1 elsewhere. The use of the per-pixel weights $\mathbf{c}_{i-1}^{i}$ means that disoccluded regions and motion boundaries are excluded from the penalty and can be rebuilt during the optimization process, while the appearance of the rest of the frame is preserved.

Video frames are stylized in sequential order. The total loss function being minimised during the image-optimisation process for frame $i$ is

$$\begin{aligned}\mathcal{L}_{total}(\mathbf{p}_i, \mathbf{a}, \mathbf{x}_i) = \alpha\mathcal{L}_{content}(\mathbf{p}_i, \mathbf{x}_i) + \beta\mathcal{L}_{style}(\mathbf{a}, \mathbf{x}_i) \\ + \gamma\mathcal{L}_{temporal}(\mathbf{x}_i, \mathbf{x}_{i-1}).\end{aligned} \tag{2.11}$$

for some weights $\alpha$, $\beta$ and $\gamma$ and where $\mathcal{L}_{content}$ and $\mathcal{L}_{style}$ are as defined in equations 2.3 and 2.4, respectively.

It is worth noting that Ruder et al.'s contributions (a new initialisation method

and a temporal loss function) can also be incorporated into the model-optimisation approach to style-transfer discussed in section 2.2.2.

# Chapter 3

# Style transfer for light fields

The aim of this project is to apply style transfer to light fields and to evaluate the results. Applying style transfer to a light field requires us to render aesthetically pleasing style transfer for each of the light field's sub-aperture images while also preserving the light field's angular structure. We already know that the methods described in section 2.2 can be used to render satisfactory style transfer for a single sub-aperture image. Thus, the main challenge for us is to preserve the angular consistency of the light field during the stylization process.

This chapter describes the new light field style transfer methods that we developed. The following notation is used throughout the chapter:

1. We denote the original light field that we wish to stylize by $lf = \{\mathbf{vp}_i : i = 1, \ldots, n\}$ where $n$ is the total number of light field view points and $\mathbf{vp}_i$ is the $i^{th}$ view point. These view points are our content images.

2. We denote the style image by $\mathbf{a}$.

3. We denote the stylized light field that we wish to generate by $lf' = \{\mathbf{vp}'_i :$

$i = 1, \ldots, n\}$ where $\mathbf{vp}'_i$ is the $i^{th}$ view point of this stylized light field.

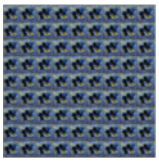
4. We denote by $\omega_i^j$ the function that warps a given image according to the optical flow between the original light field view points $\mathbf{vp}_i$ and $\mathbf{vp}_j$.

5. We denote by $\mathbf{c}_i^j$ the per-pixel weights defined between the original light field view points $\mathbf{vp}_i$ and $\mathbf{vp}_j$ and defined to be 0 for disoccluded regions between the two view points and 1 elsewhere.

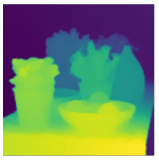
6. The content, style and temporal loss functions, denoted by $\mathcal{L}_{content}$, $\mathcal{L}_{style}$ and $\mathcal{L}_{temporal}$ are as defined in equations 2.3, 2.4 and 2.10, respectively.

$lf = \{\boldsymbol{vp}_i | i = 1,2,3, \ldots, n\}$ ⟵ original light field to be stylized (content images)

$\boldsymbol{a}$ ⟵ style image



$lf' = \{\boldsymbol{vp}'_i | i = 1,2,3, \ldots, n\}$ ⟵ stylized light field to be generated



$\omega_i^j(\,)$ ⟵ warps a given image according to the optical flow between the original viewpoints $\boldsymbol{vp}_i$ and $\boldsymbol{vp}_j$.

$\boldsymbol{c}_i^j$ ⟵ per pixel angular loss weights defined to be 0 for disoccluded regions between viewpoints $\boldsymbol{vp}_i$ and $\boldsymbol{vp}_j$ and 1 elsewhere.

Figure 3.1: Important notation used throughout this chapter.

## 3.1 The naïve approach

The first light field style transfer approach tried was to stylize each light field view point independently using the image-optimization approach described in section 2.2.1. This method acts as a baseline for the rest of our work. The following algorithm is used to stylize the light field $lf$.

For $i = 1, 2, 3, \ldots, n$:

1. Initialize $\mathbf{vp}'_i$ to be white noise;

2. Iteratively update $\mathbf{vp}'_i$ using gradient descent so as to minimise the loss function

$$\mathcal{L}_{total}(\mathbf{vp}_i, \mathbf{a}, \mathbf{vp}'_i) = \alpha\mathcal{L}_{content}(\mathbf{vp}_i, \mathbf{vp}'_i) + \beta\mathcal{L}_{style}(\mathbf{a}, \mathbf{vp}'_i). \tag{3.1}$$

Immediately, we can spot some short-comings with the loss function in equation 3.1. It only incorporates the current view point being stylized and it completely ignores the angular structure of the light field. Unsurprisingly, this approach does not preserve the light field's angular structure. This is illustrated in figure 3.2 which displays two neighbouring view points of the stylized light field. Both view points exhibit aesthetically pleasing style transfer when considered as independent images. However, despite the original view points being very similar, the stylized view points are very different. This occurs as the optimization process converges to different local minima. It leads to flickering and inconsistencies when the stylized light field is displayed.

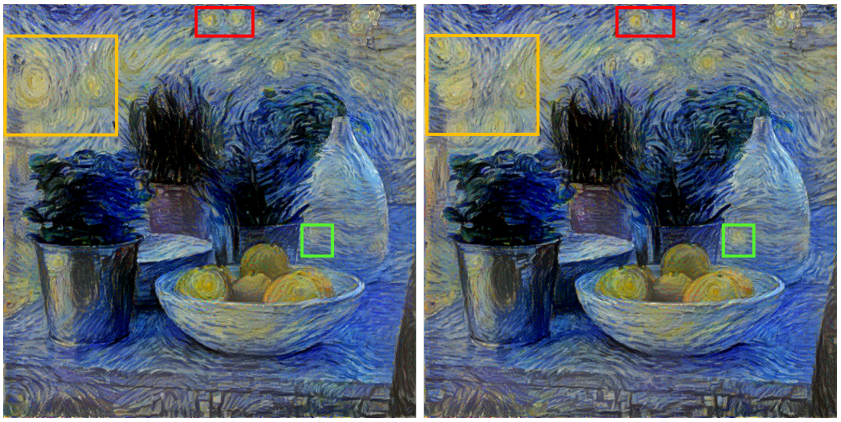


Figure 3.2: Two neighbouring view points are stylized very differently. This occurs as the optimization process converges to different local minima. The coloured boxes highlight some of the inconsistencies

Thus, this naïve approach to light field style transfer does not produce very good results. Despite rendering aesthetically pleasing style transfer for each view point, the angular structure of the light field is completely lost.

## 3.2 Video style transfer for light fields

The next approach tried was to apply the video style transfer method described in section 2.2.4 to light fields. A pseudo-video can be constructed from the light field view points by scanning them in, for example, a snake-like pattern or a spiral pattern (figure 3.3). Video style transfer can then be applied to this pseudo-video.

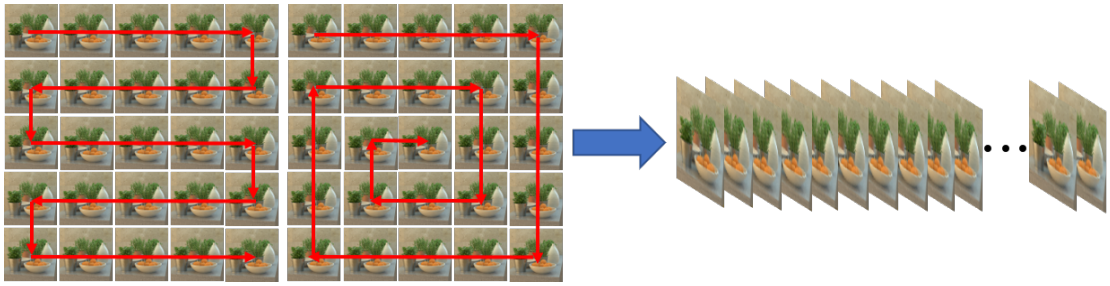


Figure 3.3: A pseudo-video can be constructed from the light field by scanning the view points in, for example, a snake-like pattern or a spiral pattern. Video style transfer can then be applied to this pseudo-video.

The following algorithm is used to stylize the light field $lf$ using video style transfer.

**Step 1:** Create a pseudo-video from the light field view points.

**Step 2:** Stylize the first frame of the pseudo-video. That is, initialise $\mathbf{vp}'_1$ to be white noise and iteratively update it using gradient descent so as to minimise the loss

$$\mathcal{L}_{total}(\mathbf{vp}_1, \mathbf{a}, \mathbf{vp}'_1) = \alpha\mathcal{L}_{content}(\mathbf{vp}_1, \mathbf{vp}'_1) + \beta\mathcal{L}_{style}(\mathbf{a}, \mathbf{vp}'_1).$$

**Step 3:** Stylize the remaining frames of the pseudo-video in sequential order. That is, for $i = 2, 3, \ldots, n$:

1. Initialize $\mathbf{vp}'_i$ to be $\omega^i_{i-1}(\mathbf{vp}'_{i-1})$.

2. Iteratively update $\mathbf{vp}'_i$ using gradient descent so as to minimise the loss

$$\begin{aligned}\mathcal{L}_{total}(\mathbf{vp}_i, \mathbf{a}, \mathbf{vp}'_i) &= \alpha\mathcal{L}_{content}(\mathbf{vp}_i, \mathbf{vp}'_i) + \beta\mathcal{L}_{style}(\mathbf{a}, \mathbf{vp}'_i) \\ &+ \gamma\mathcal{L}_{temporal}\left(\mathbf{vp}'_i, \mathbf{vp}'_{i-1}\right).\end{aligned}$$

We tried scanning the light field view points in both a snake-like pattern and a spiral pattern to generate the pseudo-video. Both led to improved results when compared with the naïve approach (section 3.1), with the snake-like pattern arguably performing better than the spiral pattern.

However, there are still notable problems with this method. While the initialisation and temporal loss ensure that the stylization of two consecutively processed view points is consistent, the long term consistency of the stylization wears off as the process moves further along the pseudo-video. For example, when using either the snake-like or spiral patterns illustrated in figure 3.3, there are inconsistencies between the stylization of the first view point on the second row and the first view point on the first row. This is despite them being neighbouring view points in the light field array.

These issues were partially alleviated using a multi-pass approach. Here, the stylization process passes over the light field several times with the view point scanning pattern alternating each pass, for example, between a snake-like pattern starting in the top-left corner and traversing the view points row-wise and a snake-like pattern starting in the bottom-right corner and traversing the view points column-wise. Following the first pass, each view point is initialised to be blend of its stylized output from the previous pass and non-disoccluded parts of the

previous view point on the current pass warped according to the optical flow (i.e. $\mathbf{c}^i_{i-1} \cdot \omega^i_{i-1}(\mathbf{vp}'_{i-1})$) as in [22]. While this multi-pass approach did improve performance, it led to significant increases in computational time.

## 3.3 A better approach

The previous two light field style transfer approaches (sections 3.1 and 3.2) were just style transfer methods designed for other purposes (i.e. images and videos) adapted for light field application. In this section we present our novel style transfer approach which is tailored specifically for light fields. The basic idea is to propagate the style outwards from the central view point while always ensuring that angular consistency is preserved (figure 3.4). We achieve this through the introduction of a new initialisation method for the image-optimisation process and a new angular loss function.

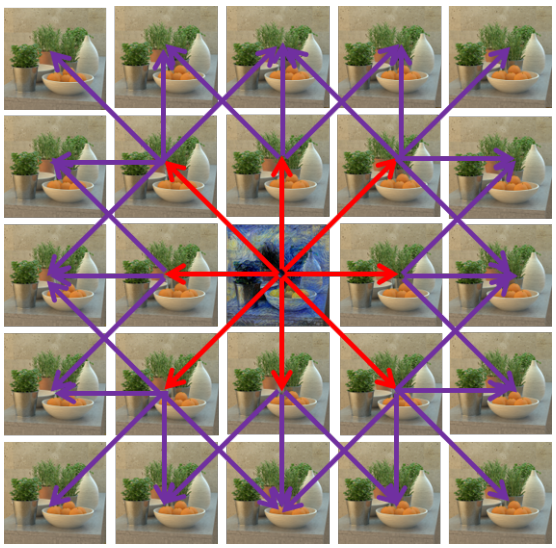


Figure 3.4: Starting with the central view point, the style transfer is propagated outwards while always ensuring that angular consistency with previously stylized view points is preserved.

In general terms, the basic algorithm is as follows:

**Step 1:** Stylize the central view point;

**Step 2:** Stylize the view points surrounding the central view point while preserving angular consistency with the central view point;

**Step n:** Stylize the next outer most view points while preserving angular consistency with neighbouring view points stylized at step $n-1$.

Step 1 is straight-forward - we stylize the central view point using the image-optimisation approach of section 2.2.1. For all other view points we extend this image-optimisation approach to incorporate our new initialisation method and angular loss function.

### A better initialisation

White noise initialisation is still used for the central view point. For all other view points $j$, we warp each previously stylised neighbouring view point $\mathbf{vp}'_i$ according to the optical flow between $\mathbf{vp}_i$ and $\mathbf{vp}_j$. We then initialise $\mathbf{vp}'_j$ to be a weighted sum of these warped view points. More formally, we initialise $\mathbf{vp}'_j$ according to

$$\mathbf{vp}'_j = \sum_{i \in P} w_i \omega_i^j(\mathbf{vp}'_i) \tag{3.2}$$

where $P$ be the set of neighbouring view points that have already been stylized. When choosing the weights $w_i$, greater weight is given to horizontal and vertical neighbours than to diagonal neighbours. Figure 3.5 illustrates the initialisation process.

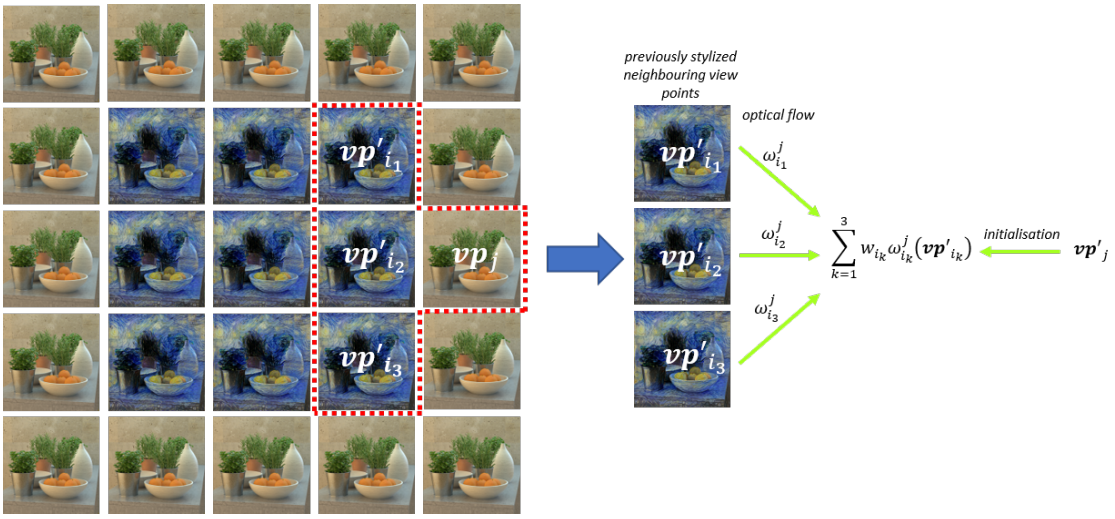



Figure 3.5: Initialisation of $\mathbf{vp}'_j$ for the image-optimisation process: Each previously stlylized neighbouring view point $\mathbf{vp}'_i$ is warped according to the optical flow between $\mathbf{vp}_i$ and $\mathbf{vp}_j$. Then $\mathbf{vp}'_j$ is initialised to be a weighted sum of these warped stylized view points.

### An angular loss

We define the *angular loss* for the $j^{th}$ view point to be:

$$\mathcal{L}_{angular}(\mathbf{vp}'_j) = \sum_{i \in P} \left( \sum_{pixels} \mathbf{c}_i^j \cdot \left( \omega_i^j(\mathbf{vp}'_i) - \mathbf{vp}'_j \right)^2 \right) \tag{3.3}$$

where the inner summation is pixel-wise and again $P$ is the set of neighbouring view points that have already been stylized. This angular loss penalises deviations between neighbouring stylized view points of the light field. The use of the per-pixel weights $\mathbf{c}_i^j$ means that disoccluded regions between the nieghbouring view points are excluded from the penaliser. Figure 3.6 illustrates the construction of the angular loss for a sample view point.

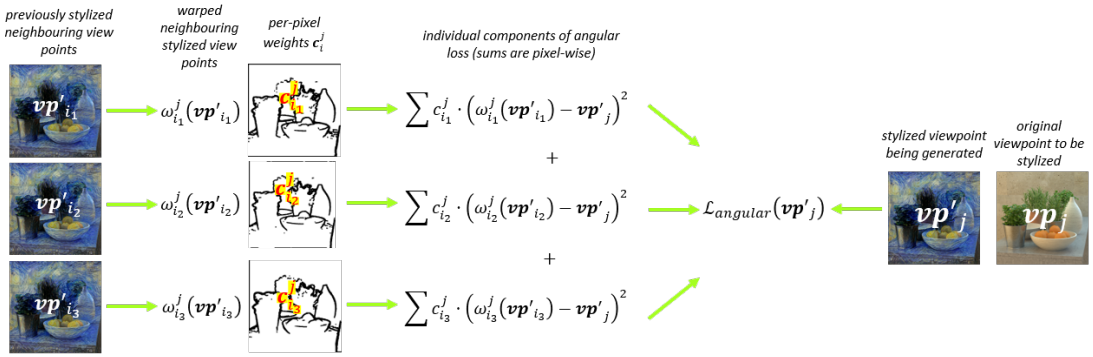



Figure 3.6: Angular loss for $\mathbf{vp}'_j$: For each previously stylized neighbour $\mathbf{vp}'_i$, we get the pixel-wise summed squared error between $\omega_i^j(\mathbf{vp}'_i)$ and $\mathbf{vp}'_j$. This pixel-wise sum is weighted by the per-pixel weights $\mathbf{c}_i^j$. This is repeated for each previously stylized neighbour and the results are summed to get the total angular loss $\mathcal{L}_{angular}(\mathbf{vp}'_j)$.

Thus, to stylize a view point $j$ (where $j$ is not the central view point), we first initialise $\mathbf{vp}'_j$ according to equation 3.2. We then use gradient descent to iteratively update $\mathbf{vp}'_j$ so as to minimise the total loss function

$$\begin{aligned}\mathcal{L}_{total}(\mathbf{vp}_j, \mathbf{a}, \mathbf{vp}'_j) = \alpha\mathcal{L}_{content}(\mathbf{vp}_j, \mathbf{vp}'_j) + \beta\mathcal{L}_{style}(\mathbf{a}, \mathbf{vp}'_j) \\ + \gamma\mathcal{L}_{angular}(\mathbf{vp}'_j)\end{aligned} \tag{3.4}$$

for some weights $\alpha$, $\beta$ and $\gamma$.

Together, our new initialisation method and angular loss function help to preserve the angular structure of the light field during the stylization process. We will see later that our method produces significantly better results than the naïve baseline method of section 3.1.

**Implementation details**

Similar to Gatys et al. [9], we use the VGG19 network [25] to construct the content and style representations of images. The VGG19 network is a deep convolutional neural network pre-trained for object recognition. Also similar to Gatys et al., we use the output from layer conv4_2 to construct the content representation of an input image and the output from layers conv1_1, conv2_1, conv3_1, conv4_1 and conv5_1 to construct the style representation of an input image.[1] This is in accordance with the definitions given in section 2.2.1 for the content and style representations of an image.

As used by Ruder et al. [22, 23] for their video style transfer, we use DeepMatching [28] and DeepFlow [29] to calculate the optical flow $\omega_i^j$ between the light field view points $\mathbf{vp}_i$ and $\mathbf{vp}_j$. DeepMatching is a matching algorithm for computing dense correspondences between two images. DeepFlow uses DeepMatching to calculate the optical flow between two images.

The weights $\mathbf{c}_i^j$ are calculated using a consistency check of the forward optical flow and the backward optical flow. Similar to Ruder et al. [22, 23], we use the consistency check provided in [26]. The check is based on the fact that in non-disoccluded regions the backward flow vector should point in the inverse direction as to the forward flow vector

## 3.4 Depth-aware style transfer for light fields

This section presents an alternative depth-aware approach to light field style transfer. It is inspired by the depth-aware style transfer for traditional two-dimensional

[1] conv$i$_$j$ is the $j^{th}$ convolutional layer of the $i^{th}$ convolutional block of the network.

images outlined in section 2.2.3. The angular consistency of the light field is preserved during the stylization process through the introduction of a depth loss function which preserves the depth structure of the original light field in the stylized light field.

Unlike our previous light field style transfer method (section 3.3) which uses the image-optimisation approach, our depth-aware light field style transfer method uses the model-optimisation approach, i.e. a transformation network is trained to approximate the solution to the image-optimisation problem. Also unlike our previous method which processes the light field one view point at a time, this depth-aware approach processes the entire light field in one go. It follows that we now have loss functions defined for an entire light field rather than for single view points.

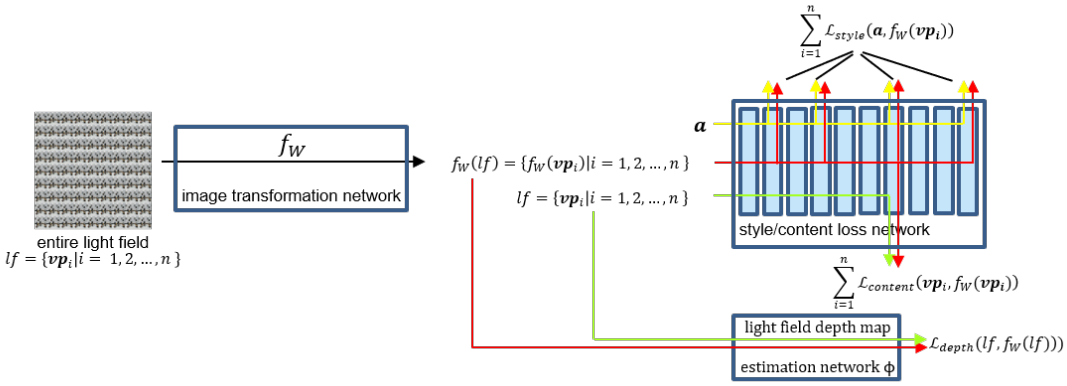



Figure 3.7: Network architecture for depth-aware light field style transfer: The system consists of 3 networks - a transformation network which reads in a light field and outputs the stylized light field and two pre-trained loss networks used for training the transformation network.

Figure 3.7 illustrates the system architecture. It consists of three networks

- a transformation network and two pre-trained loss networks. The transformation network $f_W$ takes a light field $lf$ as input and outputs the stylized light field $f_W(lf) = \{f_W(\mathbf{vp}_i)|i = 1, 2, \ldots, n\}$. It is parameterised by weights $W$. Once again, the content and style losses are calculated using a CNN pre-trained for object recognition. The content and style losses for the entire light field are obtained by summing the content and style losses of the individual view points. A depth loss is defined using a pre-trained light field depth estimation network $\phi$ which takes a light field $lf$ as input and outputs an estimate $\phi(lf)$ of its depth map. The depth loss is then the pixel-wise mean squared error between the estimated depth maps of the original and stylized light fields, i.e.

$$\mathcal{L}_{depth}(lf, f_W(lf)) = \frac{1}{HW} \sum_{i,j=1}^{H,W} \left(\phi(lf)_{i,j} - \phi(f_W(lf))_{i,j}\right)^2 \tag{3.5}$$

The total loss function for a light field $lf$ is therefore

$$\begin{aligned}\mathcal{L}_{total}\left(lf, \mathbf{a}, f_W(lf)\right) = \alpha \sum_{i=1}^{n} \mathcal{L}_{content}(\mathbf{vp}_i, f_W(\mathbf{vp}_i)) + \beta \sum_{i=1}^{n} \mathcal{L}_{style}(\mathbf{a}, f_W(\mathbf{vp}_i)) \\ + \gamma \mathcal{L}_{depth}\left(lf, f_W(lf)\right)\end{aligned} \tag{3.6}$$

for some weights $\alpha$, $\beta$ and $\gamma$. Training is carried out on the transformation network $f_W$ to find weights $W$ so as to minimise the expected loss $\mathcal{L}_{total}\left(lf, \mathbf{a}, f_W(lf)\right)$ for an arbitrary input light field $lf$ and the chosen style image $\mathbf{a}$. Thus, during training all of the light field view points are passed through the transformation network. Only after all of the view points have been processed, and the style, content and depth losses for the entire light field have been calculated, are the network weights

$W$ updated via gradient descent.

### Implementation details

Similar to Johnson et al. [18], we use the VGG16 network [25] to calculate the content and style losses. Layer conv2_2 is used to construct the image content representations, while layers conv1_1, conv2_2, conv3_3 and conv4_3 are used to construct the style representations.[2]

For the depth loss network we use EPINET [24] which, as mentioned in section 2.1.2, is a fully-convolutional neural network for light field depth estimation.

For the transformation network we use the same architecture as used by Johnson et al. [18] for their image transformation network for stylizing single images (section 2.2.2). It follows that the stylized light field $f_W(lf)$ is given by $f_W(lf) = \{f_W(\mathbf{vp}_i)|i = 1, 2, \ldots, n\}$. A possible area for further research would be to consider alternative architectures, possibly more suited to light field applications, for the transformation network.

[2] conv$i$_$j$ is the $j^{th}$ convolutional layer of the $i^{th}$ convolutional block of the network

# Chapter 4

# Results and Evaluation

In this chapter we evaluate the performance of our light field style transfer methods as described in the previous chapter.[1]

As mentioned in section 2.2, style transfer is not a well-defined problem and this makes evaluation difficult. When evaluating style transfer algorithms there is a three way trade-off between quality, speed and flexibility. However, in this project we only focused on the quality of the style transfer and so this is the focus of our evaluation. While loss functions (for example, for style and content) can be used as metrics to put a numerical value on style transfer quality, it is arguable that the most important criteria when evaluating style transfer quality is that it is aesthetically pleasing to look it. If this is so, then subjective evaluation makes the most sense when evaluating style transfer quality.

As mentioned previously, the main challenge with light field style transfer is to

[1] All implementations were done in Python and use the PyTorch library. All of our code is available in the GitHub repository associated with the project. See https://github.com/doegan32/Light-Field-Style-Transfer. Videos illustrating some of our results are also available in this GitHub repository. These videos are helpful for subjective evaluation of our results.

preserve the angular consistency of the light field. However, there are no readily available metrics to evaluate the angular consistency of an edited light field and so subjective evaluation is required here.

Given the difficulties in evaluating both style transfer and edited light fields, it follows that evaluating light field style transfer is not a straight-forward task.

## 4.1 Evaluation of light field style transfer using our new initialisation method and angular loss

In this section we evaluate our light field style transfer approach from section 3.3, i.e. propagating the style outwards from the central view point and using our new initialisation method and angular loss function to preserve angular consistency. We evaluate its performance against the naïve baseline approach from section 3.1, i.e. stylizing each view point independently. There are three parts to our evaluation. First, we do a subjective evaluation of the aesthetic quality of the style transfer (subsection 4.1.1). Then we use epipolar plane images to examine our approach's ability to preserve the angular structure of the original light field (subsection 4.1.2). Finally, we look at how well our approach preserves the depth structure of the original light field (subsection 4.1.3). We use three example light fields for the evaluation, namely the *herbs* and *table* light fields from the HCI 4D Light Field Dataset [13] and the *lego knights* light field from the (New) Stanford Light Field Archive [1].

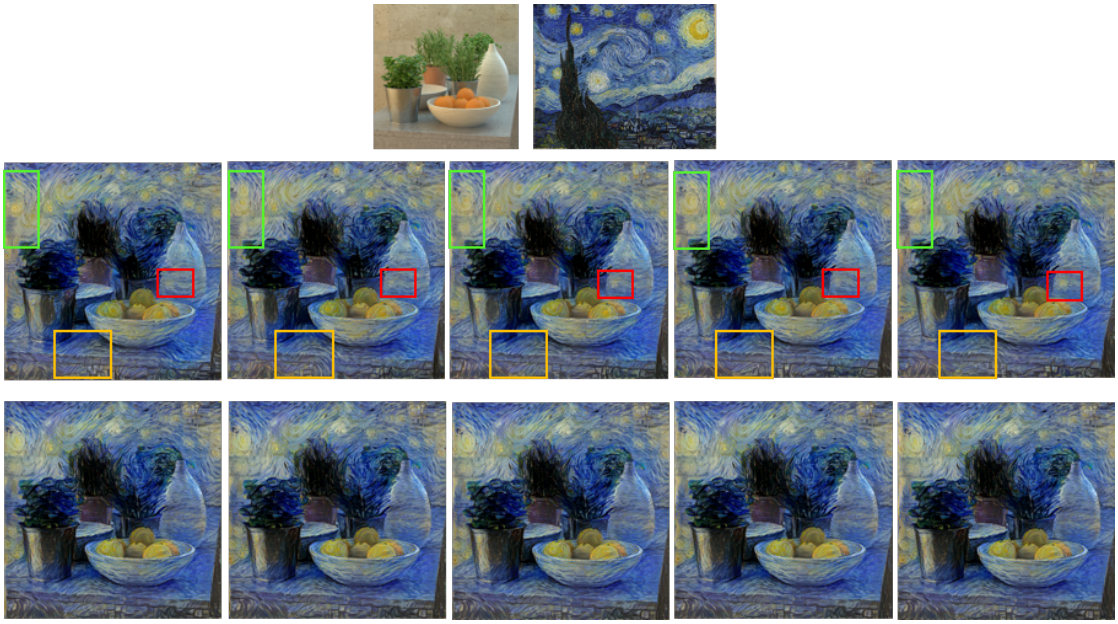
### 4.1.1 Subjective/aesthetic evaluation

Figure 4.1 displays subsets of the outputs (five neighbouring view points from the middle row of the light field array) from the baseline approach and our approach for each of the three sample light fields. For each light field, the top row was stylized using the baseline naïve approach, while the bottom row was stylized using our approach.

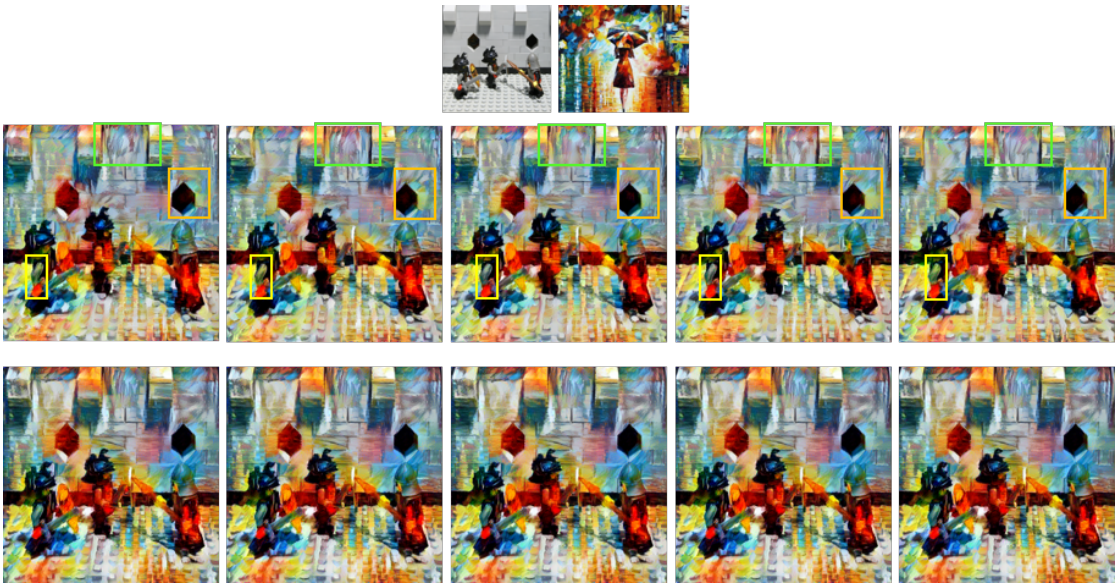
When considering the light field view points as independent images, both approaches produce results of a similar quality. However, our approach clearly leads to more consistent style transfer between the view points. The coloured boxes highlight just some of the inconsistencies in the outputs from the baseline approach. As mentioned in section 3.1, these inconsistencies arise as the image-optimisation process converges to different local minima.

Although significantly better, there are still some minor issues with the outputs from our approach. These mainly relate to inconsistencies in the stylization of finer structures and occlusions such as the plant leaves in the *herbs* light field and the arm of the lamp in the *table* light field.[2]

[2]Inconsistencies within each output and differences between outputs are more easily identified in the videos on the project GitHub page.



(a) *Herbs* light field from [13].



(b) *Lego Knights* light field from [1].

(c) *Table* light field from [13].

Figure 4.1: For each light field the top row has been stylized using the baseline method, while the bottom row has been stylized using our method with our new initialisation and angular loss. The coloured boxes highlight some of the inconsistencies in the baseline method. Each row displays five neighbouring view points from the middle row of the light field array.

### 4.1.2 Evaluation using epipolar plane images

As mentioned in subsection 2.1.1, epipolar plane images (EPIs) are a useful tool for visualising the angular structure of a light field.

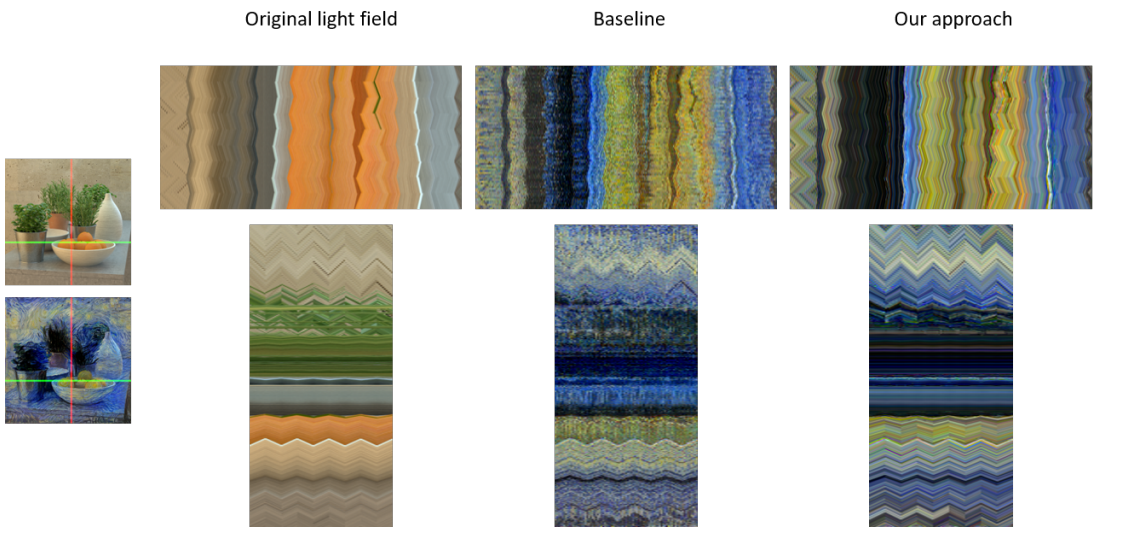
Figure 4.2 displays example epipolar plane images for each of the three sample light fields. For each light field, the top row shows horizontal EPIs and the bottom row shows vertical EPIs.[3] The EPIs on the left correspond to the original light field, the EPIs in the middle correspond to stylization using the baseline method and the EPIs on the right correspond to stylization using our approach.

For each of the three light fields, we can see that the EPIs for the baseline method are quite noisy. This noise corresponds to the inconsistencies or flickering that we observe when we cycle through the stylized view points. In comparison, the EPIs corresponding to our approach are quite similar in structure to those of the original light fields. This illustrates the fact that our approach is much better at preserving the angular structure of the original light field.

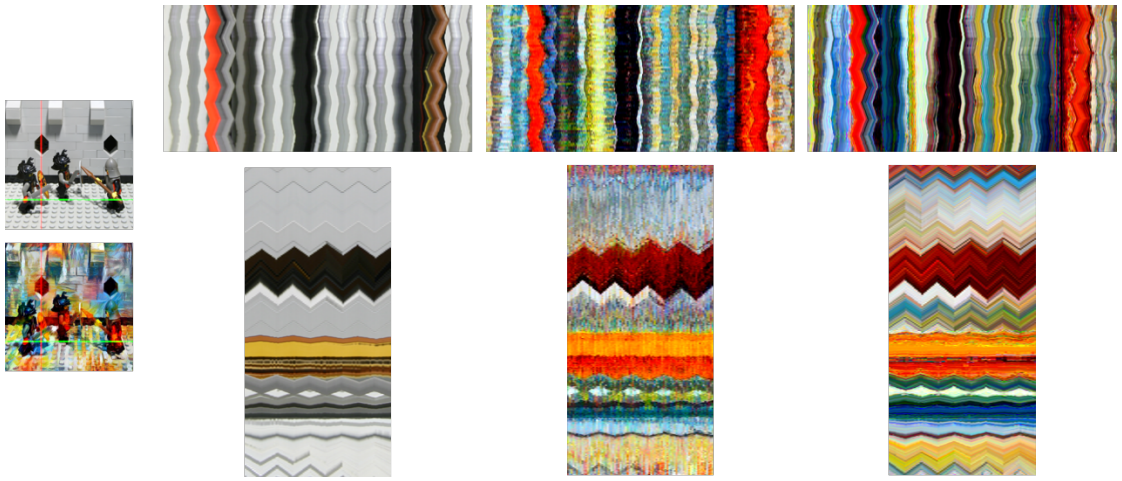
However, there are still some slight issues with our approach. As mentioned in the previous section, these mainly correspond to finer structures and occlusions in the light field scene. For example, in the vertical EPIs for the original *herbs* light field (bottom left of figure 4.2a) we can see an almost horizontal green line crossing over the other zig-zagged green lines. This structure corresponds to the leaves of the plant and it is almost completely lost in the vertical EPIs for our approach (bottom right of figure 4.2a). Similarly, in the vertical EPIs for the original *table* light field (bottom left of figure 4.2c) the black zig-zag pattern corresponds to the

[3]While the formal definition of horizontal (vertical) EPIs is to take rows (columns) from view points in a single row (column) of the light field array, here we stack together the EPIs for all rows (colomns) of the light field array as if they were scanned in a snake-like order.

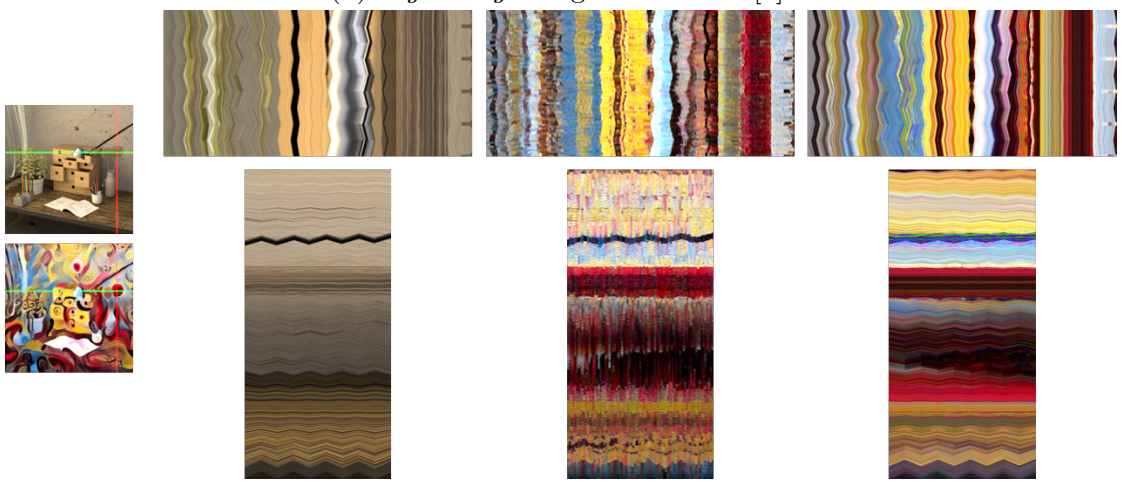
arm of the lamp. This structure is almost completely lost in the vertical EPIs for our approach (bottom right of figure 4.2c). Our approach was unable to properly preserve these fine occlusions.




(a) *Herbs* light field from [13].



(b) *Lego Knights* light field from [1].



(c) *Table* light field from [13].

Figure 4.2: Evaluating the angular structure of the stylized light fields. For each light field, the top row shows example stacked horizontal EPIs, while the bottom row shows example stacked vertical EPIs.

### 4.1.3 Evaluation using depth estimation

Figure 4.3 shows three estimated depth maps for each of the sample light fields. On the left are the estimated depth maps for the original light fields, in the middle are the estimated depth maps for the light fields stylized using the baseline method and on the right are the estimated depth maps for the light fields stylized using our approach. All depth maps were estimated using EPINET [24].

For each of the three sample light fields, the estimated depth map for the baseline method is extremely poor with the light fields' depth structure being completely destroyed. In comparison, our approach is significantly better at preserving the depth structure of the original light field. Given that EPINET uses the epipolar geometry of the light fields to estimate the depth maps, this illustrates that our approach is significantly better at preserving the epipolar geometry (and thus the angular structure) of the light fields. However, once again, we can still see some issues with our approach. For example, some detail is lost with fine occlusions and structures such as the leaves on the plants in the *herbs* light field and the arm of the lamp in the *table* light field.

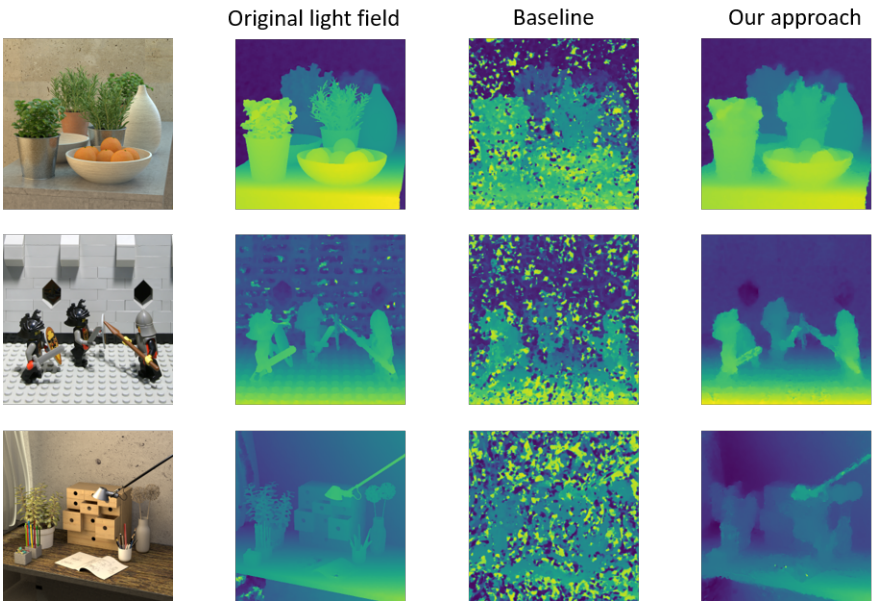



Figure 4.3: Light field depth map comparison. Top row: *Herbs* light field from [13]. Middle row: *Lego Knights* light field from [1]. Bottom row: *Table* light field from [13].

## 4.2 Evaluation of depth-aware light field style transfer

As mentioned in section 3.4, our depth-aware light field style transfer method requires us to train a transformation network which processes the entire light field in one go. This has a high computational cost and due to limitations in the hardware available to us we were unable to complete this. However, we did achieve some preliminary results testing the process on a single light field with reduced angular and spatial resolutions.

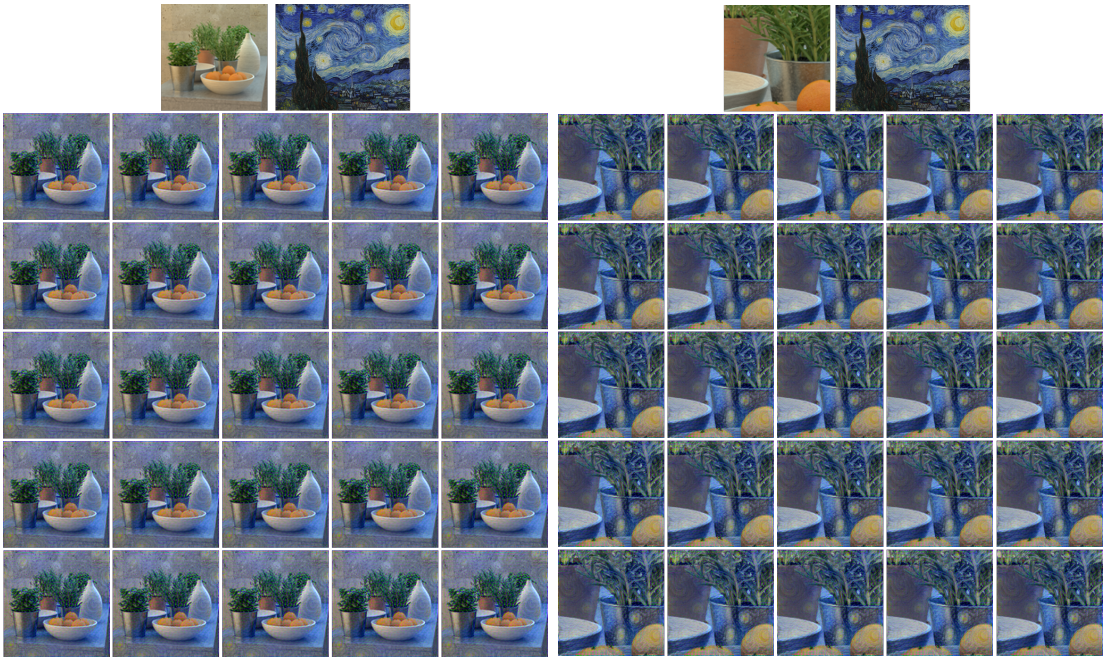
We reduced the angular resolution of the input light field to $5 \times 5$. This is the minimum angular resolution required by the EPINET network that we used to

define the depth loss function.

We first reduced the spatial resolution to $128 \times 128$ by downsampling. This signifcantly reduces the quality of the light field. Figure 4.4a shows a stylized downsampled light field. We can see that the result is not very stylized. Possible causes for this could be a bad choice of weights in the loss function, not enough training, or perhaps the reduced quality of the input caused the style and content loss network to fail. There are also inconsistencies in the stylization between the view points.[4] A possible cause for this could be a reduction in the quality of the depth estimation network's performance arising from the poor quality input.

We then tried reducing the spatial resolution to $128 \times 128$ by cropping the light field. Figure 4.4a shows a stylized cropped light field. Once again, the result is not very stylized. However, the stylization between the view points is more consistent than for the downsampling approach.

[4]These inconsistencies are not very obvious in figure 4.4a. They are more easily identified when dynamically cycling through the view points (see videos on the project GitHub page).



(a) (b)

Figure 4.4: For both experiments, the angular resolution of the light field was reduced to $5 \times 5$. For (a) the spatial resolution was reduced to $128 \times 128$ via downsampling. For (b) the spatial resolution was reduced to $128 \times 128$ by cropping the light field.

While we did not achieve results with our depth-aware light field style transfer to match the quality of the results of our earlier approach from section 3.3 (i.e. using the image-optimisation approach with our new initialisation and angular loss), the preliminary results achieved here for the cropped light field illustrate the potential for the depth-aware approach to light field style transfer. With better hardware, more training could be carried out on better quality light fields. This could produce significantly better results. Also, as mentioned in section 3.4, an area of further research could be to look into different network architectures, possibly more suited to light field applications, for the transformation network.

# Chapter 5

# Conclusions and Future Work

## 5.1 Conclusions

The goal of this project was to apply neural style transfer to light fields. We first carried out a review of light field imaging and of existing style transfer methods for traditional two-dimensional images and videos. We adapted these existing style transfer methods to be able to apply them to light fields. After observing the shortcomings of these methods when applied to light fields, i.e. their inability to preserve the angular structure of the light field, we built upon them to develop our new approach for light field style transfer. With this new approach, the centre view point of the light field is stylized first. The style is then propagated outwards while always ensuring that the angular consistency of the light field is preserved. This is achieved through the introduction of a new initialisation method and an angular loss function for the image-optimisation process.

We evaluated our method against the naïve baseline approach of stylizing each light field view point independently. We saw that our method significantly outper-

forms this baseline approach according to all criteria evaluated. Despite this, we also observed that our method still has room for improvement when dealing with the finer structures and occlusions of a light field scene.

We also presented an architecture for a depth-aware approach to light field style transfer. With this approach the entire light field is processed in one go rather than one view point at a time. A pre-trained light field depth-estimation network is used to define a depth loss function. This depth loss function is used to preserve the depth structure and hence the angular structure of the light field during the stylization process. While we have not yet achieved results with our depth-aware method to match the quality of the results from our first light field style transfer method, the preliminary results achieved suggest that the depth-aware approach has the potential to produce high quality light field style transfer.

## 5.2 Limitations and future work

The following list outlines some limitations of our work and some possible directions for future work.

1. Our work was focused solely on achieving high quality light field style transfer. We did not focus on the efficiency of our method. It is computationally very slow. Future work could look at ways to speed up the process.

2. All of our evaluation methods were subjective. We did not have any quantitative evaluation. Numerical metrics for quantitatively evaluating light field style transfer could be developed. For example, in [3, 6] the authors propose metrics for evaluating the angular consistency of light fields. Such metrics

could be incorporated into a quantitative evaluation of light field style transfer methods.

3. The implementation of our depth-aware light field style transfer is incomplete. With better quality hardware more training could be carried out on higher quality light fields. Also, as mentioned earlier, it would be worth looking at alternative architectures, possibly more suited to light field applications, for the transformation network.

4. If the depth-aware approach produces high quality light field style transfer, a subjective evaluation campaign could be organised to evaluate the results against those from our first light field style transfer method.

5. The light field style transfer approaches looked at in this project all look to stylize the sub-aperture images while simultaneously trying to preserve the light field's angular structure. An alternative approach, which may better preserve the angular structure, could be to look at stylizing the epipolar plane images instead. However, covolutional neural networks pre-trained for object recognition could most likely no longer be used here. An alternative would need to be found.

# Bibliography

[1] The (new) stanford light field archive. `http://lightfield.stanford.edu/lfs.html`. Accessed on: July 28, 2020.

[2] E. H. Adelson and J. R. Bergen. *The Plenoptic Function and the Elements of Early Vision*. Computational Models of Visual Processing, M. Landy and J. A. Movshon. MIT Press, Cambridge, 1991.

[3] A. Ak, S. Ling, and P. Le Callet. No-reference quality evaluation of light field content based on structural representation of the epipolar plane image. In *2020 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, pages 1–6, 2020.

[4] Yang Chen, Martin Alain, and Aljosa Smolic. Fast and accurate optical flow based depth map estimation from light fields. In *Irish Machine Vision and Image Processing Conference (Received the Best Paper Award)*, 2017.

[5] D. G. Dansereau, O. Pizarro, and S. B. Williams. Decoding, calibration and rectification for lenselet-based plenoptic cameras. *Computer Vision and Pattern Recognition (CVPR)*, pages 1027–1034, 2013.

[6] P. David, M. L. Pendu, and C. Guillemot. Angularly consistent light field

video interpolation. In *2020 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2020.

[7] Jakub Fišer, Ondřej Jamriška, David Simons, Eli Shechtman, Jingwan Lu, Paul Asente, Michal Lukáč, and Daniel Sýkora. Example-based synthesis of stylized facial animations. *ACM Transactions on Graphics*, 36(4), 2017.

[8] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style. *ArXiv*, abs/1508.06576, 2015.

[9] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pages 2414 – 2423, 2016.

[10] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. *SIGGRAPH: International Conference on Computer Graphics & Interactive Techniques*, page 43, 1996.

[11] S. Heber and T. Pock. Convolutional networks for shape from light field. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3746–3754, 2016.

[12] S. Heber, W. Yu, and T. Pock. Neural epi-volume networks for shape from light field. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2271–2279, 2017.

[13] Katrin Honauer, Ole Johannsen, Daniel Kondermann, and Bastian Gold-

luecke. A dataset and evaluation methodology for depth estimation on 4d light fields. In *Asian Conference on Computer Vision*. Springer, 2016.

[14] I. Ihrke, J. Restrepo, and L. Mignard-Debise. Principles of light field imaging: Briefly revisiting 25 years of research. *IEEE Signal Processing Magazine, Signal Processing Magazine, IEEE, IEEE Signal Process. Mag*, 33(5):59 – 69, 2016.

[15] Ondřej Jamriška, Šárka Sochorová, Ondřej Texler, Michal Lukáč, Jakub Fišer, Jingwan Lu, Eli Shechtman, and Daniel Sýkora. Stylizing video by example. *ACM Trans. Graph.*, 38(4), July 2019.

[16] H. Jeon, J. Park, G. Choe, J. Park, Y. Bok, Y. Tai, and I. S. Kweon. Accurate depth map estimation from a lenslet light field camera. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1547–1555, 2015.

[17] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural style transfer: A review. *IEEE Transactions on Visualization and Computer Graphics*, 2019.

[18] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *Computer Vision - ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II*, page 694, 2016.

[19] M. Levoy. Light fields and computational imaging. *Computer*, 39(8):46 – 55, 2006.

[20] Marc Levoy and Pat Hanrahan. Light field rendering. *Proceedings of the 23rd Annual Conference: Computer Graphics  Interactive Techniques*, page 31, 1996.

[21] Xiao-Chang Liu, Ming-Ming Cheng, Yu-Kun Lai, and Paul L. Rosin. Depth-aware neural style transfer. In *Proceedings of the Symposium on Non-Photorealistic Animation and Rendering*, NPAR '17, New York, NY, USA, 2017. Association for Computing Machinery.

[22] Manuel Ruder, Alexey Dosovitskiy, and Thomas Brox. Artistic style transfer for videos. *Pattern Recognition: 38th German Conference, GCPR 2016, Hannover, Germany, September 12-15, 2016, Proceedings*, page 26, 2016.

[23] Manuel Ruder, Alexey Dosovitskiy, and Thomas Brox. Artistic style transfer for videos and spherical images. *International Journal of Computer Vision*, 126(11):1199 – 1219, 2018.

[24] C. Shin, H. Jeon, Y. Yoon, I. S. Kweon, and S. J. Kim. Epinet: A fully-convolutional neural network using epipolar geometry for depth from light field images. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4748–4757, 2018.

[25] K. Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015.

[26] N. Sundaram, T. Brox, and K. Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *European Conference on Computer Vision (ECCV)*, Lecture Notes in Computer Science. Springer, Sept. 2010.

[27] S. Wanner and B. Goldluecke. Globally consistent depth labeling of 4d light fields. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 41–48, 2012.

[28] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. DeepFlow: Large displacement optical flow with deep matching. In *IEEE Intenational Conference on Computer Vision (ICCV)*, Sydney, Australia, December 2013.

[29] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. DeepFlow: Large displacement optical flow with deep matching. In *IEEE Intenational Conference on Computer Vision (ICCV)*, Sydney, Australia, December 2013.

[30] G. Wu, B. Masia, A. Jarabo, Y. Zhang, L. Wang, Q. Dai, T. Chai, and Y. Liu. Light field image processing: An overview. *IEEE Journal of Selected Topics in Signal Processing, Selected Topics in Signal Processing, IEEE Journal of, IEEE J. Sel. Top. Signal Process*, 11(7):926 – 954, 2017.