View Synthesis in Light Field Volume Rendering Using Convolutional Neural Networks

Seán Martin, B.A. (Mod)

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,

Graphics and Vision Technologies

Supervised by

Dr. Michael Manzke

August 2018

Declaration

I declare that the work described in this dissertation is, except where otherwise stated, entirely my own work, and has not been submitted as an exercise for a degree at this or any other university.

Signed: ______________________ Date: ______________________

Seán Martin

Permission to Lend and/or Copy

Acknowledgements

I extend a warm thanks to my supervisor, Michael Manzke, for providing valuable insight into my chosen topic, fantastic suggestions for areas of investigation, and great help in general.

Additionally, I am extremely grateful to Seán Bruton and David Ganter for their assistance with the Inviwo visualisation framework, and insightful discussions about current research in my area.

Furthermore, I greatly appreciate the support of my family and friends throughout this year, and those that came before.

Lastly, my fellow classmates are bright individuals and well engaged with the course, pushing me forward through co-operation and competition, which I am extremely thankful for.

Seán Martin

University of Dublin, Trinity College
August 2018

View Synthesis in Light Field Volume Rendering Using Convolutional Neural Networks

Seán Martin, Master of Science in Computer Science
University of Dublin, Trinity College, 2018

Supervisor: Dr. Michael Manzke

In this dissertation, view synthesis for light field volume rendering using deep learning is investigated. Given a single volume rendered central view along with the camera parameters and an estimated depth map from the rendering pipeline, a structured grid of 8×8 views of the volume are synthesised. Firstly, this involves estimating a per pixel depth map during volume ray casting. This depth map is converted to a disparity map using the camera parameters for the purposes of image warping. Preprocessing is performed on the disparity map and reference view, such as image warping and normalisation, in order to be fed to a Convolutional Neural Network. Deep learning is applied with the prospect of accounting for inaccuracies in the depth map, non-Lambertian effects, and occlusions to improve the visual coherency of synthesised views over pure disparity based image warping. Multiple residual Convolutional Neural Network architectures are experimented with, and in particular, the results of three dimensional convolutions are compared with two dimensional convolutions. The experiments performed reveal that the networks all learnt to improve the visual quality of views that are far away from the central reference view, but failed to improve views that were close to the reference. In addition, convolutional neural networks are fast enough for view synthesis at interactive rates, and geometrical image warping is the bottleneck for light field synthesis in real-time. Overall, the methods presented have limitations, but show promising results for future research and expansion. Source code for this dissertation is available online at `https://github.com/flabby99/LF_View_Synthesis` and a video demonstration is available at `https://youtu.be/AQ4ec7Bgn1s`.

Contents

List of Figures

List of Tables

Acronyms

2D	two dimensional.
3D	three dimensional.
4D	four dimensional.
CNN	Convolutional Neural Network.
CPU	Central Processing Unit.
EDSR	Enhanced Deep Super-Resolution.
EPI	Epipolar-Plane Image.
GIF	Graphics Interchange Format.
GPU	Graphics Processing Unit.
HDF5	Hierarchical Data Format, version 5.
LPIPS	Learned Perceptual Image Patch Similarity.
MRI	Magnetic Resonance Imaging.
PNG	Portable Network Graphics.
PSNR	Peak Signal to Noise Ratio.
RGB	Red Green Blue.
SD	Standard Deviation.
SSIM	Structural Similarity.
TF	transfer function.

1 Introduction

1.1 Motivation

The motivation for this research derives from the large body of related research available, but not specifically aimed towards advancing view synthesis for light field volume rendering. Particularly, recent research by Srinivasan et al. [Sri+17] showed impressive results with using only one image from a camera to synthesise a light field. Building upon this research to apply it to volume rendered light fields and draw out further insights is an exciting prospect. Volume rendering a light field offers the opportunity to achieve a new level of convincing immersive experiences. Although direct volume rendering by ray tracing is possible in real time for a single viewpoint, this is infeasible for a full light field. As such, performing fast and high quality view synthesis in light field volume rendering has the potential to disrupt visualisation techniques. For example, medical practitioners could view the result of Magnetic Resonance Imaging (MRI) scans in virtual reality in real-time, without the drawback of current virtual reality devices, such as a single focal plane. These impressive visualisations would allow for deep understanding of a patient's anatomy before surgery, and open up new avenues for medical training. With this in mind, the challenging problem of view synthesis for volume rendered light fields was determined as the focus of this dissertation.

In addition, applying deep learning to this complex problem drives the motivation of this research. In computer vision, deep Convolutional Neural Networks have quickly become state of the art for many tasks, including image classification, depth estimation and object segmentation. For view synthesis in light field volume rendering, taking advantage of the volumetric information available and combining it with rendered views for view synthesis is a complicated task, and it seems appropriate to apply deep learning to this. Recently, the Convolutional Neural Network (CNN) architecture has been extensively applied to resolution enhancement of light fields of natural images to great success, with the deep learning approaches Wu et al. [Wu+17a] and Kalantari, Wang, and Ramamoorthi [KWR16] constituting state of the art.

1.2 Overview

Light field technology is an exciting emergent subject, allowing for extremely rich capture of visual information. Unlike conventional images, which record a two dimensional (2D) projection of light rays, a light field describes the distribution of light rays in free space [Wu+17a], described by a four dimensional (4D) plenoptic function. Capturing an entire light field is infeasible, so a light field is generally sampled by capturing a large uniform grid of images of a scene. [LH96]. However, capturing this 4D data leads to an inherent resolution trade off between dimensions. As such, the angular resolution, or number of images in the grid, is often low. Because of this, view synthesis from a limited number of reference views is of great interest and benefit to any light field application.

In this dissertation, view synthesis for volume rendered light fields was considered. The general idea was to experiment with combining research on view synthesis for light field cameras, view synthesis in volume rendering, and deep learning techniques. In summary, this involved creating the following pipeline. Firstly, depth heuristics were used to estimate a reasonable depth map during volume ray casting, based on the work of Zellmann, Aumüller, and Lang [ZAL12]. This depth map was converted to a disparity map in pixel values using the camera parameters from ray casting. Using the disparity map, a fast image warping shifted information from the single reference view to all novel view locations. Finally, multiple different CNN architectures were experimented with to improve the visual consistency of all synthesised views in a single pass.

The results of this research suggest that deep learning can definitely improve view synthesis in light field volume rendering. The learnt residual light field does increase the visual consistency of synthesised views, especially for those views at a large distance from the reference view. Unfortunately though, the method of Srinivasan et al. [Sri+17] does not apply particularly well to volumes. The drawback of the three dimensional (3D) convolutions they use can be effectively removed by remapping the network to use 2D convolutions. However, their method is aimed towards images of flowers captured by light field cameras. The captured flower is always near the centre of the view, so the most important information in the scene is never outside the camera frustum of the reference view. In volume rendering, important information is often outside the central view, and no CNN architecture was able to learn to fill in the unseen information.

Additionally, the method is not faster than directly volume rendering the light field, both taking close to one second for a light field of 64 images with 512×512 pixels each. Promisingly, the bottleneck in terms of time performance was not using CNNs, but instead the geometrical image warping procedure is too slow, especially since only limited Graphics Processing Unit (GPU) memory is available, so it has to be performed on the Central Processing Unit (CPU). Any experiments performed to have a CNN learn to perform image warping, rather than improve warped images, did function faster than direct volume rendering, but the results were inaccurate. In this case, the CNN roughly learnt how to shift information using a disparity map, although the colour consistency between views was weak. If this was to be used, a custom loss function penalising colour inconsistency would be required.

1.3 Project Goals

1. Effectively combine and expand on ideas from multiple existing fields for this new research area. In particular, approaches from 2D image processing, light field angular resolution enhancement, deep learning and view synthesis in volume rendering are of interest.
2. Investigate applicability of deep learning in this domain, using volumetric data and rendered images.
3. Achieve a reasonable level of quantitatively measured view synthesis quality.
4. Experiment with the feasibility of view synthesis with deep learning for light field volume rendering at interactive rates.
5. Integrate the view synthesis pipeline into the volume rendering framework Inviwo [Sun+15].

2 Background Research

2.1 Summary of Primary Research Areas

Listed below are the main areas of research, with a short explanation on why the subject is important. Each of these research topics is later expanded with particularly relevant literature on that matter.

- **Light field representation:** Representing and sampling the 4D plenoptic function describing a light field is of utmost importance in any light field application. This is commonly performed by capturing a grid of images of a scene at a fixed exposure, which is most appropriate in this case.
- **View synthesis in volume rendering:** In volume rendering, view synthesis has been studied to increase application frame rate or reduce bandwidth in remote rendering applications. Primarily, depth heuristics during volume rendering will be studied in this domain.
- **Light field angular resolution enhancement:** A good deal of research has been performed into view synthesis, or angular resolution enhancement, for light field cameras. Depth based approaches to this problem are particularly suitable, since we have volumetric information available. Nonetheless, methods which take advantage of the light field structure will also be studied.
- **Deep learning applied to light fields:** Other problems in the light field domain aside from view synthesis have also seen success with deep learning. The ideas behind these deep learning architectures are novel and conducive to comprehensive study of light field learning techniques.
- **Image quality evaluation:** Regardless of the approach, evaluation of the synthesised output is critical. Quantitative evaluation is generally performed by comparing synthesised views to a ground truth by a metric.

2.2 Light Field Representation

The plenoptic function described by Adelson [Ade+91] is a seven dimensional function recording the intensity of light rays travelling in every direction (θ, ϕ) through every location in space (x, y, z) for every wavelength λ, at every time t. As a simplification, the plenoptic function is assumed to be mono-chromatic and time invariant, reducing it to a five dimensional function $L(x, y, z, \theta, \phi)$. In 1996, two very important papers were published on light fields, "Light Field Rendering" by Levoy and Hanrahan [LH96] and "The Lumigraph" by Gortler et al. [Gor+96]. They both realised that in space free of any occluders, the five dimensional plenoptic function contained a redundant dimension. In this case, the radiance along a ray remains constant along a straight line, so the z co-ordinate is unnecessary, and the plenoptic function reduces to the four dimensional function $L(x, y, \theta, \phi)$. This four dimensional plenoptic function is termed the four dimensional light field, or simply the light field; the radiance along rays in empty space.

The most popular light field representation was introduced by Levoy and Hanrahan [LH96], parametrising a ray by its intersection with two planes, a *uv* plane and an *st* plane. As such the 4D light field would map rays passing through a point (u, v) on one plane and (s, t) on another plane to a radiance value:

$$L : \mathbb{R}^4 \to \mathbb{R}, \quad (u, v, s, t) \mapsto L(u, v, s, t)$$

With this parametrisation, a light field could be effectively sampled using arrays of rendered or digital images by considering the camera to sit on the *uv* camera plane, and capture images on the *st* focal plane. This representation was ideal for generating new views from sample images, by interpreting the input images as 2D slices of the 4D light field [IMG00]. It was later shown by Lin and Shum [LS04] that if the disparity between neighbouring views in a light field sampling is less than one pixel, novel views can be generated without ghosting effects by the use of linear interpolation. Since sampling rates are rarely this high, to achieve such a densely sampled light field, view synthesis, or angular resolution enhancement, is a necessity.

A fundamental notion in computer vision and light fields is that of an Epipolar-Plane Image (EPI). As aforementioned, an image can be considered a 2D slice of the 4D light field by fixing constant camera co-ordinates (u^*, v^*) and letting (s, t) vary. However, the light field can be sliced in other manners, such as choosing a horizontal line by fixing t^* on the focus plane, and fixing v^* on the camera plane to produce an EPI:

$$E : \mathbb{R}^2 \to \mathbb{R}, \quad (u, s) \mapsto L(u, v^*, s, t^*)$$

EPIs are primarily of interest due to their structure. A line of constant slope in an EPI corresponds to a point in 3D space, where the slope of the line is related to the depth of that point. This means, for Lambertian objects, the intensity of the light field should not change along a line of constant slope [WG14].

2.3 Light Field Angular Resolution Enhancement

There are two main paradigms for synthesising views for light field cameras. One estimates some form of geometry in the scene, commonly depth, and bases the view synthesis on this geometry. The other focuses on the structure of light fields, using expected properties of EPIs for view synthesis, or transforming the problem to other domains with well defined behaviour. For instance, transforming the problem into the Fourier domain [Shi+14] or the Shearlet domain [VBG18]. Research on both sets will be described in turn.

2.3.1 Depth Based Approaches

Wanner and Goldluecke [WG14] presented among the first techniques for increasing angular resolution of light fields in a global manner, and it remains a state of the art non-learning approach. They require disparity maps for each input view for their resolution enhancement approach. Disparity maps can be obtained in any manner, but they propose to use EPI analysis along horizontal and vertical slices, which are integrated into a consistent disparity map. The view synthesis problem is formulated as a continuous inverse problem, and optimised by minimising an energy using convex relaxation techniques. However, as explained in previous work by Wanner and Goldluecke [WG12], backwards warping using depth maps requires an expensive pre-computation step to create the backwards warps. Furthermore, the convex optimisation is slow and scales with the number of input views and pixels.

In an effort to apply CNNs to synthesise a greater number of views from a sparse set of input views than in previous methods such as Yoon et al. [Yoo+15], Kalantari, Wang, and Ramamoorthi [KWR16] proposed a novel learning based approach. Using the four corner sub-aperture views of a light field Lytro camera, they synthesise novel intermediate views. To make learning more tractable, they introduced a disparity estimation component and a colour prediction component which are modelled by two sequential CNNs. Interestingly, instead of training the disparity CNN to directly reduce error between estimated disparity and ground truth disparity, both the disparity and colour prediction CNNs were trained to minimise the error between the estimated and ground truth novel view images. They surmise that this provides two benefits; firstly, that ground truth disparity maps are not required. Secondly, the network learns to estimate disparity maps specifically for view synthesis, which may be more appropriate than learning general disparity mapping. However, the method is based on warping the four corner images, so if none of the warped images contain valid information, this approach will fail to generate high quality results. This usually occurs with occlusions or non-Lambertian surfaces. Additionally, the system does not run at interactive rates, taking roughly 12.3 seconds to generate a single novel view from four input images of 541×376 resolution. This paper provides much inspiration for this method, but we will avoid synthesising the light field view by view to improve the time performance.

Srinivasan et al. [Sri+17] tackled the interesting, but severely ill-posed problem of synthesising a 4D light field from a single image sample. They aim to take advantage of redundancies in the light field and prior knowledge of scene content to infer the full light field. In effort to make the problem more manageable, they factorise light field synthesis into depth estimation for each ray in the light field with a CNN, then warping the original image based on the depth estimation to form a dense grid of 64 novel views. However, this depth-based image warping assumes that all surfaces in the scene are Lambertian. As such, they apply a 3D CNN to the stacked novel views in order to predict occluded rays and non-Lambertian effects, and correct for them. That is, their method accounts for specular highlights, rather than assuming that all surfaces exhibit diffuse reflection. With respect to volume rendering, this is very relevant, as surfaces are often anisotropically shaded. In contrast to most approaches, they produce all novel views at once instead of synthesising each novel view separately. This induces good performance, synthesising a 4D light in under one second on a single NVIDIA Titan X GPU. Although the output results are comparable to [KWR16], which takes four views as input, the method was trained on a large dataset of plants and flowers and experiments suggest it may not generalise well to other scenes. However, because of the speed of this approach, and the single sample input view required, our approach will be largely based on this method.

2.3.2 Structure Based Approaches

One of the earliest applications of the CNN framework to the domain of light field images was by Yoon et al. [Yoo+15]. To take advantage of the light field structure, they split the light field sub-aperture images into overlapping vertical pairs, horizontal pairs, and squares of four images. They employed three separate spatial resolution enhancement networks for the different sets of images, to produce a $\times 2$ spatial upsampling of the images. Three further angular resolution enhancement networks were applied to the upscaled image sets to produce a synthesised centre view for the three different sets of images, resulting in a $\times 2$ angular upsampling. Their work produced promising results, but using three separate networks for spatial resolution enhancement is rather redundant. Furthermore, the “method can only increase the [angular] resolution by a factor of two, and is not able to synthesise views at arbitrary locations” [KWR16]. This resolution increase is too low to directly be of use in this case, as too many direct volume rendered views would be required.

To address the redundancy issue in [Yoo+15], Yoon et al. [Yoo+17] revisited their method two years later. The primary modification involved performing $\times 2$ spatial upscaling of all the sub-aperture images using one CNN as opposed to three networks. As in their previous work, the resulting upscaled images were then passed through three networks for view synthesis. In addition to being more computationally efficient, the single CNN was able to learn a global view of spatial resolution enhancement, resulting in fewer artefacts in the reconstructed views. This suggests that using a global CNN for synthesis rather than multiple CNNs will produce better results, which will be adopted in this paper.

Wu et al. [Wu+17b] presented an approach to take advantage of the EPI structure in light field image data. An EPI is more undersampled in the angular domain than in the spatial domain. To balance this, the authors remove the high frequency components of an EPI using a blur. The blurred EPI is then upsampled to a higher angular resolution using bicubic interpolation. Next, a CNN is applied to restore the detail of the EPI in the angular domain. Finally, a deblur operation is applied to the EPI to retrieve the high frequency spatial information, based on the blur previously used. To reconstruct a full light field, multiple EPIs are reconstructed. This is somewhat ill explained in the paper, but the source code reveals that they interpolate over row EPIs, then column EPIs, to produce a higher resolution light field output. This output is then interpolated across rows again in a second iteration to improve the synthesis and produce the final result, which is very accurate. Performance timing is not reported in the paper, but testing the source code on a computer with a NVIDIA GeForce GTX 1080 GPU took almost sixteen minutes to produce a full 14×14 angular resolution light field. It is worth noting, that although this is too slow to be used in this case, it is actually faster than the state of the art method of Kalantari, Wang, and Ramamoorthi [KWR16].

2.4 Deep Learning on Light Fields

As discussed, CNNs have been applied with great success to light field angular resolution enhancement for light field cameras. Multi-view CNNs were chiefly studied, as although we have volumetric information available, multi-view CNNs often outperform volumetric CNNs. This indicates that current deep learning architectures are unable to fully exploit the power of 3D representations [Qi+16]. Research on deep learning towards other light field problems is also of benefit. This is principally because an increasingly popular approach is to train networks end to end, implying that the network learns all aspects of problem at hand. For example, in view synthesis, this avoids using computer vision techniques, such as appearance flow, image inpainting, and depth image based rendering to model certain parts of the network.

To help advance deep learning approaches in the light field domain, Wang et al. [Wan+16] proposed and compared several novel CNN architectures to train on light field images. In particular, they show how to map a 4D light field into an existing VGG network, which takes a 2D image input. Since light field datasets tend to be smaller than 2D image datasets, this provides huge benefit, as the weights of a pre-trained model can be updated rather than training from scratch. Additionally, the authors point out that although a 4D filter is intuitive to use on a 4D light field, the number of parameters quickly explode, and the benefit of pre-trained models is lost. This conflicts with the arguments presented by Srinivasan et al. [Sri+17] that the 3D CNN they employ, which uses a 4D filter, is effective. As such, the original 3D CNN they presented will also be remapped into multiple 2D architectures to experiment with removing the drawback of 3D CNNs. In order to achieve this, we will employ a subset of the methods from [Wan+16] to map a 4D light field into a 2D CNN. Their paper is aimed towards material recognition, and as such, the methods they presented will be discussed below in terms of relevance to view synthesis. Given a set of input images, they employed the following strategies.

1. Pooling all images together after a 5×5 convolution. This loses too much information to be effective for view synthesis, therefore it is not used in this dissertation.

2. Directly stacking the images over the colour channels before performing a convolution. This is more effective for view synthesis, as very little information is lost, and is used in Shin et al. [Shi+18]. This will be experimented with in this project.

3. Remapping the multiple input images into one large 2D image by replacing each pixel in the image by a block of pixels from the different input images. This is

very similar to the raw micro-lens array captured by a Lytro Illum camera, and will be tested in this dissertation.

4. Interleave two different types of 2D filters on a remapped image, spatial and angular filters. The idea is to replicate a 4D filter, but mitigate the drawbacks, and is a very interesting architecture. However, the spatial filter is not particularly useful for view synthesis, since it is intended to pool spatial information.

2.5 View Synthesis for Volume Rendering

Volume rendering techniques are typically used to simulate an optical model of light transfer through a 3D volume by a discrete approximation of the volume rendering integral [Eng+06]. A discrete uniform sampling of a full 3D scalar field is typically available for visualisation, such as that produced by an MRI scan. Rendering this scan would involve sampling the volume, interpolating the discrete MRI data to estimate a scalar value at each of these sample positions, and then applying a transfer function (TF) to map these scalar values to 3D colour values and scalar opacity values. To iteratively approximate the volume rendering integral, the sampled colour and opacity information is composited along viewing rays through the volume. If the rays are considered to travel from the viewer into the volume, then front-to-back compositing is used. Given a ray with colour C and opacity α, and new sample data along the ray with colour C' and opacity α', then these values are front-to-back composited as

$$C \leftarrow C + (1 - \alpha) \cdot C' \tag{2.1}$$

$$\alpha \leftarrow \alpha + (1 - \alpha) \cdot \alpha' \tag{2.2}$$

As a result of this alpha compositing, there is not always a single opaque surface displayed by volume rendering, which is in direct contrast to more traditional surface rendering.

Accelerating volume rendering has long been an active research area. Most approaches focus on the re-use of information, since volume rendering is expensive to perform and rendered images do not tend change dramatically between viewpoints. This ranges from re-using lighting information, to directly warping rendered images to produce images from novel view points. Most depth based image warping techniques assume that the scene is reducible to a single opaque surface with a single depth value. However, as aforementioned, this assumption does not hold for volume rendering, and errors will occur in image warping as a result. The solution we propose to this is to approximate depth values, use these for warping, then apply a CNN to the resulting images to correct for inaccuracies.

Zellmann, Aumüller, and Lang [ZAL12] proposed to warp images received from a remote server based on an additional depth channel. This has the benefit of covering up latencies by warping the last frame received from the server until a new one is available. Unfortunately, there is no single meaningful depth value for volumes. To further complicate matters, the appearance of the volume rendered largely depends on the transfer function applied. Different depth values would be more appropriate depending on the chosen transfer function. For example, taking depth as the first non-transparent voxel along the ray would be accurate if the transfer function applied tended to reveal isosurfaces The authors present multiple depth heuristics for image warping, comparing them based on Peak Signal to Noise Ratio (PSNR) with the ground truth image, and the number of holes produced by warping. In general, they found that modifying the ray tracer to return depth at the voxel where the accumulated opacity along the ray reaches 80% was the best balance between speed and accuracy. This heuristic will be used in this paper, and some experiments will be performed with modifying this heuristic.

To hide the latency in a real time remote rendering setup in which volume renderings were sent to a mobile phone, Lochmann et al. [Loc+16] proposed fast view synthesis using a piecewise-analytic representation. This involves creating a view-dependent volume representation of a sample image and synthesising new views from that representation. The qualitative results are pleasing, but the approach assumes that the emission of light out of the medium is view independent. For example, this would cause errors on the silver lining of clouds. Furthermore, the volume rendering code must be modified. As suggested by the authors, a completely new avenue would be to learn to produce novel views from example image data, as we will adopt.

2.6 Evaluation Methods

"The ability to compare data items is perhaps the most fundamental operation underlying all of computing" [Zha+18]. However, effectively comparing visual patterns based on human perception remains an unsolved problem. PSNR and Structural Similarity (SSIM) are commonly used quantitative metrics in view synthesis papers. These fixed functions are effective indicators of image quality, but they do not accurately model the human visual system.

PSNR is measured on a scale of 0 to 100 dB, with higher values indicating lower Euclidean distance between the pixel values in the reference image and the synthesised image. This is fast to calculate and has clear meaning, but assumes pixel-wise independence of images. As such, it does not always closely match the human visual system.

A good deal of research has been performed to develop an image quality assessment metric which more closely matches human perception. Wang et al. [Wan+04] proposed a "framework for quality assessment based on the degradation of structural information", SSIM. SSIM is measured on a scale of 0 to 1, with greater values suggesting better similarity of the luminance, contrast and structure of two images. As such, it better able to detect artefacts such as heavy image blur than PSNR. Nonetheless, SSIM is still a fixed function, and does not tend to accurately model human perception.

More recently, the deep features of neural networks trained on various tasks, such as image classification, have been found to be an effective perceptual metric. The features of multiple deep architectures were used by Zhang et al. [Zha+18] to form the Learned Perceptual Image Patch Similarity (LPIPS) metric, measured on a scale of 0 to 1, with lower values indicating less loss between the perceptual similarity between two images. The results from LPIPS were compared by to traditional perceptual quality metrics, such as SSIM and PSNR. Using a large database of human perceptual similarity judgements, the authors found that deep features consistently outperformed these traditional metrics as a measure of perceptual quality.

3 Experimental Design

3.1 Overview of the Pipeline

The overall aim of this dissertation is to investigate performing view synthesis for light field volume rendering with deep learning at interactive rates. That is, given volume rendered sample views, along with any information saved from the rendering pipeline, synthesise a structured set of novel views. Based on background research, the following steps will be involved in this view synthesis pipeline.

1. Estimate a per pixel depth map during volume ray casting.
2. Convert the depth map to a disparity map using the camera parameters.
3. Applying preprocessing to the input disparity map and reference view, such as image warping and normalisation.
4. Using the information from the previous step, apply a CNN to perform angular resolution enhancement.

Deep learning is performed with the prospect of accounting for inaccuracies in the depth map, anisotropically shaded surfaces, and occlusions to improve the visual coherency of synthesised views over solely depth based image warping.

3.2 Experiments

As mentioned in the introduction, the goals of the project were to combine approaches from 2D image processing, light field angular resolution enhancement, and deep learning to perform view synthesis in volume rendering. Performance is compared both in terms of visual accuracy as measured by PSNR, SSIM. Additionally, some tests will be performed with the LPIPS loss metric, which uses deep features of neural networks

[Zha+18]. In order to fairly evaluate the process, a number of experiments were planned before detailed design and creation of the view synthesis pipeline:

1. Compare depth heuristics used in terms of PSNR and SSIM.
2. Evaluate different CNN architectures over the full validation set of one hundred light fields by PSNR and SSIM.
3. Analyse the image warping and the best performing CNN for each image grid location with the LPIPS loss metric, PSNR and SSIM.
4. Perform detailed timing tests in the Inviwo [Sun+15] visualisation framework, to see if synthesis can compare with direct volume rendering a light field.
5. Appraise the results of the best performing CNN on unseen volumes and transfer functions from the training set.
6. Qualitatively investigate image patches where warping performs effectively and ineffectively.

3.2.1 Test System Specifications

Every experiment would be performed on a computer with 16GB memory, an Intel i7-7700K @ 4.20GHz CPU, and a NVIDIA GeForce GTX 1080 GPU running on Ubuntu 16.04. For deep learning, the PyTorch library [Pas+17], version 0.40 would be used with Cuda 9.1, with cuDNN 7.1.2 and NVIDIA driver version 390.30.

3.3 Key Design Decisions

There were many choices to be made in this project at the design phase. Often, they would involve balancing speed and quality for a particular method. However, some decisions were more specific, such as the input and output required at each stage in the pipeline. The most important decisions to be made are listed below, and are addressed in detail following this.

- How many reference views are required for light field reconstruction?
- Which dataset to train and evaluate on, and how to sample it?
- What volume depth heuristics to use?

- How to perform image warping?
- Which CNN architectures to experiment with?

3.3.1 Number of Reference Views Required

Frequently, the four corner grid images are required for light field view synthesis. Occasionally, the four corner images and the central image are required. It would be very interesting to create a method that can take advantage of any number of reference views, as mentioned in future work of Kalantari, Wang, and Ramamoorthi [KWR16]. Designing a CNN to handle variable sized inputs is challenging, as the number of input channels would vary. Possible approaches to this would be to:

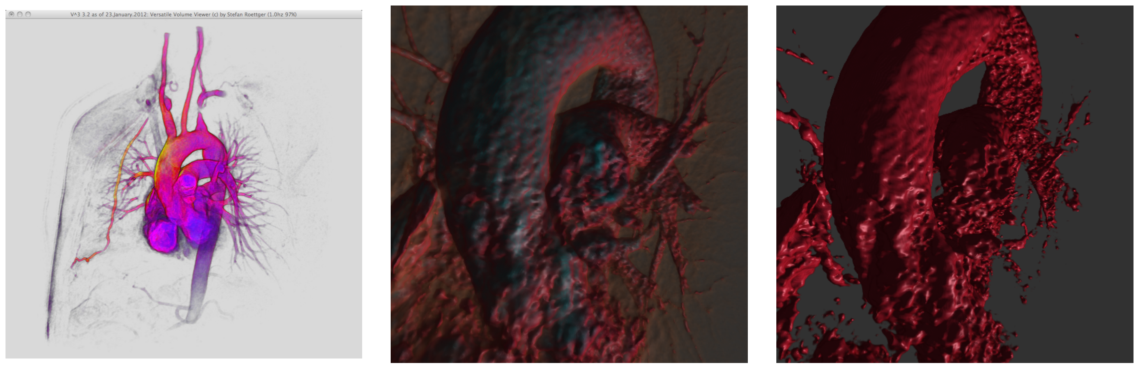
- Segment the input into smaller samples of fixed size, input those into a CNN and then combine the information.
- Apply preprocessing to convert the input information into a standard format, such as an estimated colour and depth map at each viewpoint via interpolation.
- Perform view synthesis with a per-pixel confidence mask for each reference view, similar to Zhou et al. [Zho+16]. Then, combine the synthesised views based on the confidence mask.

Unfortunately, all of the above methods take more time to perform than fixing an expected input pattern. Additionally, they would take more time to implement, pushing the project out of scope. For these reasons, an expected set of reference views are defined. A very exciting approach is to use only the central image in the light field sample to reconstruct the whole light field. Since volume rendering is slow, taking only the central image and an associated depth map would allow for much faster light field volume rendering. Thus, it was determined to require one direct volume rendered central view and a corresponding depth map as output from the volume rendering framework. That is, only one reference view would be available for view synthesis.

3.3.2 Dataset Selection

Choosing an appropriately complex dataset and associated transfer function for this project is of utmost concern. For example, suppose that a volume dataset was rendered with a transfer function that contained high frequencies which tended to reveal isosurfaces. Then the volume rendering algorithms could be "adjusted to detect these and set the depth value accordingly" [ZAL12]. This is not regularly the case, and to fairly evaluate handling non-Lambertian effects and occlusions, sufficiently complex volume datasets were required with adequate transfer functions. Additionally, the geometry revealed by the transfer function on the volume should not be too static in texture. This is in order to fairly test the depth map generation, as inaccurate depth maps can still produce correct warping on large regions with a static texture. To further elucidate this point by example, evaluating the contributions in this paper on a MRI of a head which is rendered to show only the surface of the head would be far less fruitful than a rendering which reveals the complex brain structure inside the head.

With the above information in mind, an MRI of a heart with visible aorta and arteries would be used for testing. This volume dataset has a resolution of $512 \times 512 \times 96$. The heart has a rough surface, and the aorta and arteries create intricate wavy structures which are difficult to reconstruct. The full dataset is available online [Roe18] and is also available for use in Inviwo, a visualisation framework [Sun+15]. See Figure 3.1 for examples of this dataset rendered. For training, all light field samples would be captured at 8×8 angular-resolution, and 512×512 spatial-resolution.



(A) Example rendering (B) Training transfer function (C) Simple transfer function

FIGURE 3.1: Demonstrating the training volume. Figure 3.1a was retrieved from the source repository and the other figures produced in Inviwo.

3.3.3 Volume Depth Heuristics

As opposed to surface rendering, a well-defined depth value does not exist in volume rendering. This is because in volume rendering, the 3D scalar data being rendered is accumulated to generate an image, so there is no well-defined surface. Since there is no single meaningful depth value for volumes, there are many possible heuristics for a per-pixel depth. For example, as in Zellmann, Aumüller, and Lang [ZAL12] a single pass ray marching depth heuristic would be to output the depth of the voxel where the gradient of the opacity along a ray is maximal. Alternatively, two passes could be implemented, the first pass to define an opacity threshold value and the second pass to output the depth at that threshold value. There is no single heuristic that performs best for all combinations of volumes and transfer functions. Ideally, the depth heuristic would be switched based on some evaluation of the volume and the transfer function, but this is out of scope for this project. Furthermore, two pass heuristics are too slow to be of use in this application. As such, the most consistently performing single pass method from Zellmann, Aumüller, and Lang [ZAL12] was chosen for experimentation, outputting depth at a fixed threshold value.

3.3.4 Image warping

Once the disparity map in pixels is obtained, there are multiple approaches to image warping, falling into the categories of forward mapping and backward mapping. Forward mapping views the problem as mapping pixels from the reference view into the novel view, which results in holes. Backward mapping works in the inverse direction. For each pixel in the novel view, the most relevant information from the reference view is assigned to that pixel, so no holes are formed, but the reference view is usually oversampled. In general, balancing warping speed and quality is the aim. For example, disparity could be used to directly move pixels from the reference image to a novel view, or a triangular mesh could be built over the reference image and the mesh warped to a novel view. The latter will produce better results, but of course, is more expensive. In particular, the following areas are of primary concern for image warping:

1. **Handling non-integer shifts:** Warped data very rarely lies at integer pixel positions in novel views. Of course, these can be rounded to the nearest pixel. However, sub-pixel information could be handled by interpolating at non-integer positions in backward warping, or splatting in forward warping. Splatting involves adding contributions at a sub-pixel location to neighbouring pixels with a certain weight.

2. **Handling occlusions:** In forward warping, holes and cracks may appear in a novel view, since some information that is visible in a novel view may have been occluded in the reference view. Using multiple views will often cover up these holes. However, we only have one reference view available. As such, the missing holes can be filled by interpolated from neighbouring pixels. Alternatively, image inpainting could be applied by a CNN to fill in the holes.

3. **Handling unseen information:** The border pixels of a novel view might be blank as they contain scene information that was outside the camera frustum of

the reference view. There are multiple strategies to fill in this information. As one example, the holes could be filled in by stretching out the border information from the reference view.

It was decided that two image warping techniques would be experimented with. Firstly, a simple forward warping with

1. Non-integer shifts rounded to the nearest pixel.
2. Holes from occlusions left as black holes.
3. Unseen information left as black holes.

This would be performed in the hope that the CNN would be able to easily identify incorrect information, as it would be a black hole. Through experimentation, it quickly became clear that this image inpainting operation would be an ineffective approach. Secondly, a backward warping with

1. Non-integer shifts reading information from four nearest pixel locations by bilinear interpolation.
2. Occlusions producing no holes, as backward warping is performed.
3. Unseen information filled in by stretching the border information from the reference view.

This warping would be far more accurate, and faster. However, it is uncertain if a CNN would be able to identify and correct for inaccurate information in this case. This backward warping would be implemented on both the CPU to allow for multiprocessing and avoid using GPU memory. This is because the CNNs tested require between 6GB and 7GB memory on the GPU to complete a forward pass with an input image of size $512 \times 512 \times 3$ with 32 bit precision floating point numbers, and the NVIDIA GeForce GTX 1080 has 8GB of memory available.

3.3.5 Convolutional Neural Network Architecture

From background research on CNNs applied to light field angular resolution enhancement, it is clear that multiple valid approaches exist. Since volume rendering is being considered, scene geometry is available and the challenge is taking advantage of it. Therefore, approaches related to EPI reconstruction, or reconstruction in Fourier or

Shearlet domains are less applicable in this case. On the contrary, approaches which are related to estimating depth in the scene, and performing reconstruction using this depth are very relevant. A per-pixel depth can be estimated accurately during volume rendering, and existing methods which use depth information can be taken advantage of. One consideration is whether to take a global view of view synthesis or a local view. A local view would consist of taking a warped image, a disparity map and a set of novel position co-ordinates as inputs to a CNN and outputting a corrected novel view at that single position. A global view would consist of using all warped images and disparity maps as inputs to a CNN and outputting all corrected images in one step, similar to Srinivasan et al. [Sri+17]. The method of Kalantari, Wang, and Ramamoorthi [KWR16] is a mix of the two above approaches, taking in all warped images, disparity maps and a position, outputting an estimated view at the input position.

Based on these observations, for this project two primary architecture types would be experimented with. The first would be based on the method of Srinivasan et al. [Sri+17], using the 3D convolutions in that paper, but removing the depth estimation step. This second architecture type would have attempt to combat the drawbacks of the 3D convolutions applied by Srinivasan et al. [Sri+17]. A similar input would be required and output produced, but the information would be modified as in Wang et al. [Wan+16] so that 2D convolutions can be performed. The hope is that remapping the 3D convolutions to 2D convolutions would give similar accuracy, but faster performance. Additionally, pre-trained 2D models could be taken advantage of, and a larger set of high performance 2D models exist for inspiration. To break this down further, the following architectures were experimented with.

1. A residual 3D network taking a grid of warped images and a colour mapped disparity map as input [Sri+17].
2. A 3D network taking a central view and a colour mapped disparity map as input and directly synthesising a light field.
3. Two different residual 2D networks taking a set of warped images and a colour mapped disparity map as input, stacked over the colour channels. One is based on the ResNet18 architecture [He+16] and the other on the Enhanced Deep Super-Resolution (EDSR) network [Lim+17].
4. A residual 2D network taking a set of warped images, all remapped into one large image.

4 Implementation

4.1 Outline

Chapter 3, Experimental Design, provided information on the design of the view synthesis pipeline. In this chapter, the exact algorithms, frameworks and architectures created to fit this design will be described. To briefly reiterate an overview of the pipeline, During volume ray casting, a per-pixel depth map is estimated. This is then converted to a disparity map in pixel values using the camera parameters. Using the above disparity map, the input image is warped to desired novel view locations. All warped views are stacked together with the disparity map and a residual CNN is applied to improve visual consistency.

4.1.1 The Inviwo Visualisation Framework

Inviwo [Sun+15] is extensively used in this dissertation, and as such, a high level discussion of the framework is necessary. Inviwo is an open source extensible visualisation framework written in C++ and designed for researchers. Inviwo is based on data flow networks, with nodes in the network represented by processors with input and output ports that data travels through. Properties define the state of these processors, and properties can be shared among processors. For example, the camera location could be shared among a ray casting processor and a processor which estimates a bounding box for a volume. Inviwo can be extended by creating new processors in C++ and custom Python scripts can be used to drive the application. This work makes extensive use of Inviwo to capture a training set of volume rendered light fields, and to perform experiments. A ray tracing processor was developed based on the default Inviwo ray tracing that implements depth heuristics in the fragment shader. To demonstrate validity of the method, the entire view synthesis pipeline was integrated into Inviwo using Python.

4.2 Depth Heuristics for Volumes

Depth maps are useful for image warping, but there is no single view of depth for a volume. Using default Inviwo raycasting, the depth of the first non-transparent voxel along the ray is returned, which is very unsatisfactory for image warping purposes. This is because this depth tends to be corrupted by almost transparent volume information that is close to the camera. To retrieve more meaningful depths from Inviwo two methods were employed. One method involved extracting an isosurface from the volume using the marching cubes algorithm on the CPU. This isosurface was then surface rendered with the same camera parameters as the volume rendering, and a well-defined depth extracted from the depth buffer. The depth from isosurfaces performed well for warping, but the pre-calculation step of isosurfaces on the CPU is slow. Furthermore, any changes in the volume during runtime would require isosurfaces to be recalculated.

A superior method, and the one used as the final output in this project, is to estimate a depth during ray casting. For each pixel, a ray is cast and opacity is accumulated along the ray as is the ray is advanced through the volume and sampled at the Nyquist rate. Following the research of Zellmann, Aumüller, and Lang [ZAL12], the best single pass depth heuristic was used. That is, when a ray accumulates a fixed amount of opacity, the depth at that voxel is saved. After some deliberation, it was determined to expand the above method slightly. This was due to the depth map missing some information when a ray never accumulated the desired opacity. As such, a depth value was saved when a ray accumulated a low threshold opacity, and overwritten if it later accumulated the high threshold opacity. Since ray traversal is implemented as a single pass fragment shader loop in Inviwo, it is very fast to compute this depth heuristic. On average, computing this depth heuristic takes only 20ms longer than traditional ray casting. An implementation of this depth heuristic is demonstrated by pseudocode in Algorithm 4.1.

ALGORITHM 4.1: Ray casting with depth heuristic

```
input: float low threshold, float high threshold
output: vec3 colour, float opacity, float depth
begin
    initialise low depth and high depth to $-1$
    Determine volume entry position
    Compute ray direction

    while ray position in volume
        Get voxel data at current ray position
        Composite colour and opacity

        /* Save depths by threshold heuristic */
        if opacity > low threshold and low depth is $-1$
            low depth ← depth of current position
        else if opacity > high threshold and high depth is $-1$
            high depth ← depth of current position

        Perform early ray termination
        Advance position along ray
    end

    depth ← max(low_depth, high_depth)

    /* Rays that don't pass either threshold have maximum depth */
    if depth is $-1$
        depth ← 1.0

    return colour, opacity, depth
end
```

4.3 Converting Depth to Disparity

The volume rendering pipeline provides a depth buffer, or Z-buffer, which gives a depth value for each pixel location. The depth from the Z-buffer is converted to disparity using the intrinsic camera parameters obtained from Inviwo for warping purposes. The process to convert a value $Z_b \in [0, 1]$ from the depth buffer to a pixel disparity value is as follows. The depth buffer value Z_b is converted into normalised device co-ordinates, in the range $[-1, 1]$, as $Z_c = 2 \cdot Z_b - 1$. Then the perspective projection is inverted to give depth in eye space, Z_e, as

$$Z_e = \frac{2 \cdot Z_n \cdot Z_f}{Z_n + Z_f - Z_c \cdot (Z_f - Z_n)}$$

Where Z_n and Z_f are the depths of the camera's near and far planes in eye space, respectively. Note that Z_n should be set as high as possible to improve depth buffer accuracy, while Z_f has little effect on the accuracy. Given eye depth Z_e, it can be converted to a disparity value d_r in real units by the use of similar triangles [WMG13] as

$$d_r = \frac{B \cdot f}{Z_e} - \Delta x$$

Where B is the camera baseline, or distance between two neighbouring cameras in the grid, f is the focal length of the camera, and Δx is the distance between two neighbouring cameras' principle points. Again, using similar triangles, the disparity in real units can be converted to a disparity in pixels as

$$d_p = \frac{d_r W_p}{W_r}$$

Where d_p and d_r denote the disparity in pixels and real world units respectively, W_p is the image width in pixels, and W_r is the image sensor width in real units. If the image sensor width in real units is unknown, W_r can be computed from the camera field of view θ and focal length f as, $W_r = 2 \cdot f \cdot \tan(\frac{\theta}{2})$. In Inviwo, the image sensor size in real units is roughly 2.

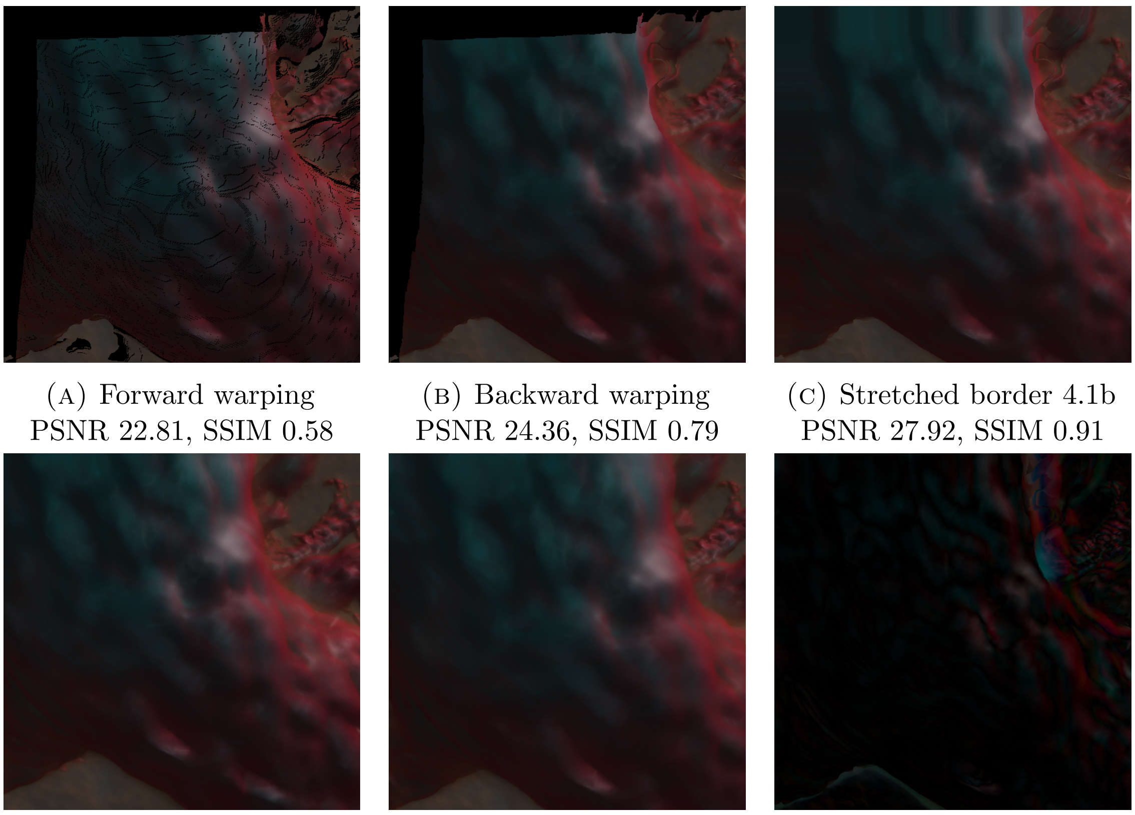
4.4 Disparity Based Image Warping

A disparity map $D : \mathbb{R}^2 \mapsto \mathbb{R}^2$ is used to relate pixel locations in a novel view to those in a reference view. Let $I_{(u_r,\ v_r)} : \mathbb{R}^2 \mapsto \mathbb{R}^3$ denote a reference Red Green Blue (RGB) colour image at grid position $(u_r,\ v_r)$ with an associated pixel valued disparity map D. Then a synthesised novel view $I'_{(u_n,\ v_n)}$ at grid position $(u_n,\ v_n)$ can be formulated as:

$$I'_{(u_n,\ v_n)}(x + (u_r - u_n)D(x,\ y),\ y + (v_r - v_n)D(x,\ y)) = I_{(u_r,\ v_r)}(x,\ y)$$

Keep in mind that the y-axis is assumed to be inverted from the regular cartesian co-ordinate system, since we use Portable Network Graphics (PNG) images, which store data left to right on the x-axis and top to bottom on the y-axis. The (u, v) grid co-ordinates are represented as matrix style indices, so position $(1, 2)$ would be in the first row, second column. As such, the co-ordinate systems of the images and the grid match.

To perform fast but accurate image warping using a disparity map, a form of backward warping with bilinear interpolation was implemented. The estimated disparity map for the central view was basically used an an estimate for all views. Pixels in the novel view that should read data from a location that falls outside the border of the reference view were set to read the closest border pixel in the reference view instead. Essentially, this would stretch the border of the reference view in the novel view, rather than produce holes. Since warped pixels rarely fell at integer positions, bilinear interpolation was applied to accumulate information from the four nearest pixels in the reference view. This resulted in fast warping with no holes, and good accuracy. A comparison between backwards mapping and forward mapping is presented in Figure 4.1.



(A) Forward warping
PSNR 22.81, SSIM 0.58

(B) Backward warping
PSNR 24.36, SSIM 0.79

(C) Stretched border 4.1b
PSNR 27.92, SSIM 0.91

(D) Central reference view

(E) Ground truth

(F) Difference 4.1e and 4.1c

FIGURE 4.1: Demonstrating different warping methods to synthesise the top left novel view at grid position $(0, 0)$ from a reference view 4.1d at position $(4, 4)$. Figure 4.1c represents the final backward warping used. For comparison, the reference view has 24.43 PSNR and 0.87 SSIM with the ground truth novel view.

4.5 Deep Learning

4.5.1 Proposed Learning Based Algorithm

From a reference view and disparity map for this view, a function f should generate a structured set of 8×8 novel views. f is modelled as the composition of three functions

$$f = p \circ s \circ w$$

Where w is a disparity based warping function, s stacks together information from w to be input into p, and p is an occlusion prediction function modelled as a residual CNN. Namely, given a reference view V and associated disparity map D

$$f(V, D) = W' = W'_1, W'_2, \ldots, W'_{64} \quad \text{where 64 is the number of output views.}$$

The function w normalises the reference view V to have all values in the range $[0, 1]$. Then, the reference view is warped to novel positions using the disparity map D. That is, w returns n warped views and the input disparity map:

$$w : (V, D) \to W_1, W_2, \ldots, W_{64}, D$$

Note that each warped view W_i is an RGB image, so it has three channels. To match these number of channels, s converts the single channel disparity map D to an RGB image using a fixed colour map with values in the range $[0, 1]$.

$$s : (W_1, W_2, \ldots, W_{64}, D) \to (W_1, W_2, \ldots, W_{64}, D')$$

Finally, p performs a CNN on the output of s, producing 64 residual images with values in the range $[-1, 1]$. This range is chosen so that the residual can remove or add colour over the full range of input values. These residual images $(R_1, R_2, \ldots, R_{64})$ are added to the warped images $(W_1, W_2, \ldots, W_{64})$. Since the residuals have values in the range $[-1, 1]$, but the warped images are in the range $[0, 1]$, the result of the addition then gets every value clipped to the range $[0, 1]$.

$$p : (W_1, W_2, \ldots, W_{64}, D') \to (W'_1, W'_2, \ldots, W'_{64})$$

4.5.2 Data Collection

Using Inviwo [Sun+15], a synthetic light field dataset is captured. The light field capturing geometry used is a “2D array of outward looking (non-sheared) perspective views with fixed field of view” [LH96] To capture the synthetic dataset, a Python script is created to move the camera in Inviwo along a regular equidistant grid. The cameras are shifted along the grid rather than rotated in order to keep the optical axes parallel. This removes any need to rectify captured images to a common plane. Each light field sample has an angular resolution of 8×8, with 512×512 spatial resolution. That is, each light field sample consists of 64 sub-aperture images, each with 512×512 pixels. See Figure 4.2 for the central sub-aperture image of five captured light fields.

A Python script was used to randomly move the camera around the scene in the region between two spheres, so as not to be too close or far away from the origin. To avoid the camera pointing to empty space, it was set to always look to a random position within a small distance from the origin. Sampling was performed uniformly, rather than focusing on particular sections of the heart. At these random positions, for each light field sub-aperture image a colour PNG is saved, and depth as a NumPy array of 32 bit precision floating point numbers is saved. To increase the diversity of the data captured, a plane with a normal aligned with the camera view direction is used to clip the volume for half the captured examples. This plane clipping can reveal detailed structures inside the volume. Furthermore, it demonstrates the validity of depth calculation during ray casting, as the only possible effect clipping on this is increased speed from a reduction in voxels. However, if isosurfaces were used to estimate depth, they would have to be recalculated each time the volume is clipped by a plane.

Overall, 2000 training light fields are captured, and 100 validation light fields. After data collection, the colour and depth images, along with camera metadata, are processed into a Hierarchical Data Format, version 5 (HDF5) file [The97]. During this processing, the camera data is used to convert the depth into disparity, and disparity is saved instead of depth. The ZLIB filter is applied for compression purposes, reducing the overall size of the dataset by close to 30%. Additionally, the file is saved with single writer, multiple reader enabled to allow for data reading with multiple cores.

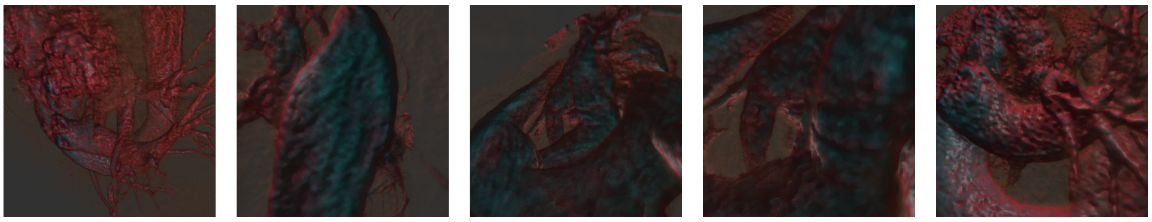


FIGURE 4.2: Sample training light field central sub-aperture views.

4.5.3 Convolutional Neural Network Architectures

As demonstrated in Wang et al. [Wan+16], there is strong evidence that 3D convolutions, which use 4D filters, can be mapped into 2D architectures. To experiment with this, four primary architectures were implemented. Note that any network's Rectified Linear Unit activations have been replaced by Exponential Linear Unit activations. The first network tested was the 3D occlusion prediction network from Srinivasan et al. [Sri+17]. All estimated sub-aperture views are stacked along one dimension and a 3D CNN is employed so that each filter is processing global information from each view.

The second network tested was a modified version of ResNet18, [He+16]. For convenience, this will be called StackedResNet. The input to StackedResNet is all warped images, and a colour mapped disparity map which are stacked over the colour channels, a 195 channel input. The first layer of ResNet18 is replaced, as it is intended to pool spatial information, and the input is instead convolved into 64 features to gather angular information. All spatial pooling is removed from ResNet18. The final fully connected layer of ResNet18 is replaced by a convolutional layer with a tanh activation function.

The third network, labelled StackedEDSR, was based on the Enhanced Deep Super-Resolution (EDSR) network by He et al. [He+16]. The body of the network remains as in the original paper. Additionally, a pre-processing step to subtract the mean of each colour channel is maintained as 1×1 filter over the colour channels. However, the first convolutional layer is modified to map 195 colour channels, instead of 3 colour channels, to 256 features. StackedEDSR also removes the final spatial upscaling performed by EDSR, and applies tanh activation after the last layer.

The final network, which will be denoted AngularEDSR, is the exact same network as EDSR, bar removal of spatial upscaling at the last layer, and application of tanh activation at the final layer. In order to map the input into the three colour channel input required for EDSR, the angular remapping from [Wan+16] is employed. The involves creating a large three channel colour image. Consider a light field sample with 8×8 images having 512×512 pixels and three colour channels. This would be remapped into a large image having $(8 \cdot 512) \times (8 \cdot 512)$ pixels and three colour channels. In this remapped image, the uppermost 8×8 pixels would contain the upper-left pixel from each of the original 8×8 views.

The 3D convolution method did not have pre-trained weights available. Pre-trained weights were not used for the ResNet based model, since all pooling was removed. Pre-trained weights were tested for both EDSR based networks, but the spatial resolution enhancement problem proved too far removed from angular resolution enhancement.

4.5.4 Training Procedure

All CNN architectures are trained by minimising the per-pixel loss between the ground truth views and the synthesised views. Specifically, the mean squared error is used, which is easily differentiable, allowing for backpropagation. To increase training speeds and the amount of available data, four random spatial patches of size 128×128 are extracted from each light field at every training epoch. Additionally, training colour images have a random gamma applied as data augmentation. Multiple Python processes are launched to load data from the HDF5 file, with single writer, multiple reader enabled for this purpose. During training and validation, sample input images, residual outputs, final outputs and target images, and difference images are logged to inspect the effectiveness of training. To further visualise the training, the validation and training loss, learning rate, and a histogram of the convolutional layer weights are logged.

During initial training experiments, the learning rate was set to reduce by a factor when the validation loss plateaued. However, this proved ineffective as the model learnt very fast, but learning came to a halt early as well. To combat this, different learning rate scheduling strategies were tested. A cyclic learning rate [Smi17] and cosine annealing the learning rate with warm restarts [LH17] were tested. Both learnt slower, but achieved lower loss over time. Cosine annealing with warm restarts proved to give the best validation loss in small tests, and was used to train on the full set. Other hyper-parameters selected for training are shown in Table 4.1

Training takes roughly 14 hours using 2D CNN architectures with eight CPU cores used for data loading and image warping. Unfortunately, the 3D CNN would not work with multiprocessing for data loading. As such, training took much longer for the 3D CNN, taking three days.

Hyper-Parameter	Value
Initial Learning Rate	0.1
Learning Rate Schedule	Cosine annealing, warm restarts
Data Augmentation	Patch extraction, random gamma
Regularisation	L2
Gradient Clipping	Based on norm at 0.4
Optimisation	Stochastic gradient descent
Momentum	Nesterov momentum at 0.9
Batch Normalisation	Yes, bar EDSR network
Weight Decay	0.0001

TABLE 4.1: Hyper parameters for CNN training

5 Evaluation

5.1 Quantitative Evaluation of View Synthesis

In order to evaluate the different network architectures experimented with, they are compared in terms of PSNR and SSIM average and standard deviation between synthesised views and ground truth views over the validation set. Furthermore, the timings for these architectures are contrasted with the time taken to directly volume render a light field. Finally, the best performers are compared using the deep features of neural networks, as suggested by Zhang et al. [Zha+18].

Overall, the PSNR and SSIM values achieved are quite reasonable for each architecture tested. In Table 5.1, the Standard Deviation (SD) and mean of these metrics averaged over all light field sub-aperture images are presented for the full validation set of one hundred light fields. For certain, these experiments show that the 3D convolutions performed in Srinivasan et al. [Sri+17] can be effectively mapped into 2D CNNs. In particular, the EDSR [Lim+17] based networks both outperform the slower 3D convolutions.

	PSNR (dB)		SSIM	
CNN Architecture	**Mean**	**SD**	**Mean**	**SD**
No CNN	**35.02**	2.75	0.920	0.024
2D StackedResNet	34.10	2.64	0.919	0.024
3D	34.45	2.54	0.917	0.022
2D StackedEDSR	34.48	2.64	0.922	0.023
2D AngularEDSR	34.34	2.64	**0.923**	0.023

Table 5.1: Comparing quantitative results for view synthesis with different CNN architectures over the validation set consisting of one hundred synthesised light fields.

From Table 5.1, it may appear that none of the residual CNNs exhibit much performance difference from geometrical warping. However, investigating the results image by image reveals further insights. This is because the CNN residual modifies the reference view. Geometrical warping obtains maximum PSNR of 100 for the central view, and SSIM of 1.0. However, when a residual CNN is applied this generally falls to roughly 40 and 0.92, respectively. This is a huge loss in PSNR, and a large contribution to difference in averages. The per image difference is more interesting, and Figure 5.1 presents this for one sample validation light field with AngularEDSR applied. In particular, note that the difference is similar along rows and spikes for images in the same column as the reference view. The synthesised image for the bottom right sub-aperture view of this light field can be seen in Figure 5.2.

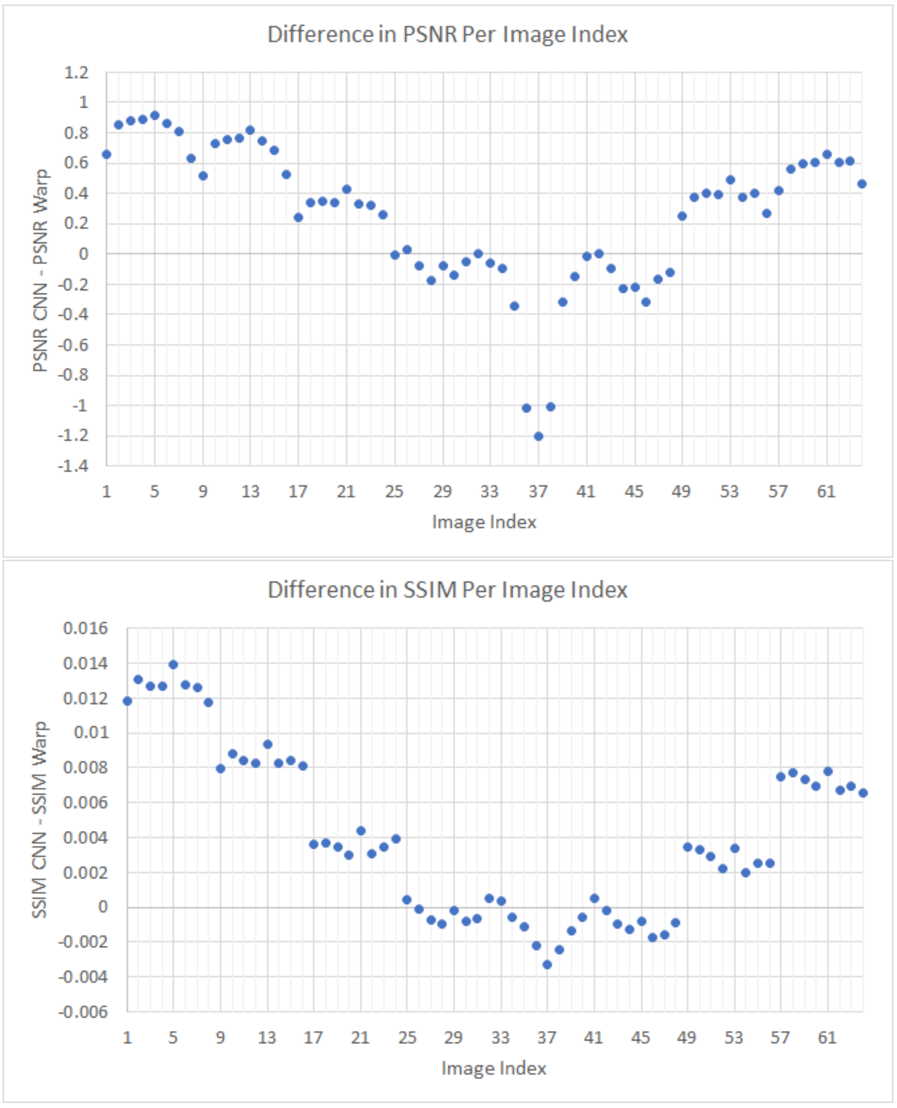
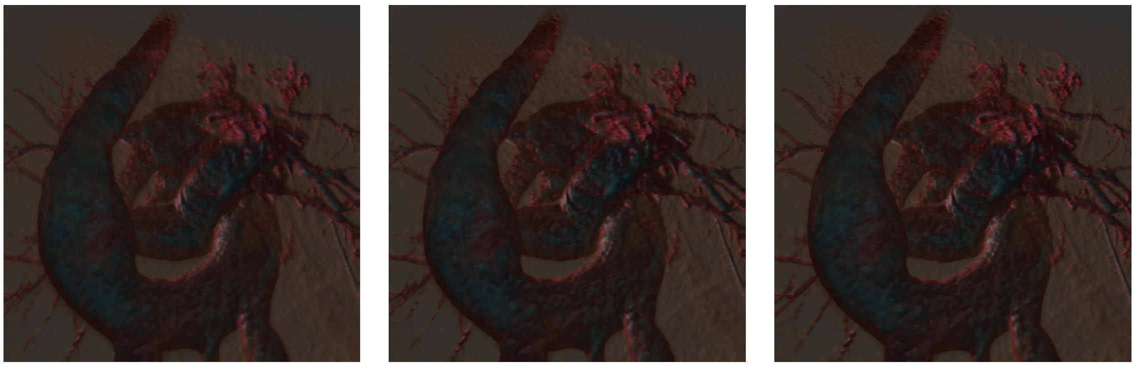


FIGURE 5.1: SSIM and PSNR increase / decrease on applying AngularEDSR on top of image warping. Results are shown per image location in an 8 × 8 grid, indexed from left to right, top to bottom. Note that index 37 is the location of the reference view. Furthermore, the loss in PSNR at index 37 is scaled to make the graph more readable.



(A) AngularEDSR
PSNR 36.05, SSIM 0.909

(B) Warping alone
PSNR 35.59, SSIM 0.903

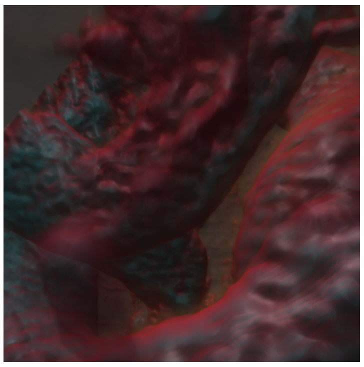
(C) Ground truth
PSNR 100.0, SSIM 1.000

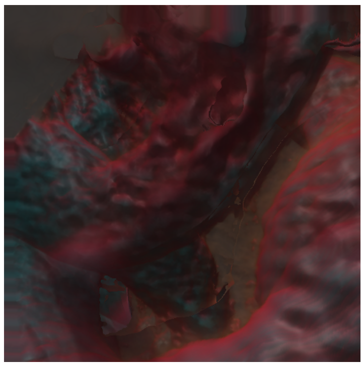
FIGURE 5.2: Comparison of the bottom right synthesised sub-aperture view in the light field which Figure 5.1 presents results for.

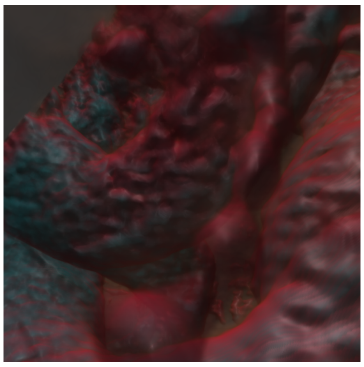
Lastly, evaluation was performed with the LPIPS metric [Zha+18]. Using this, the deep features of AlexNet [KSH12] are used to form a perceptual loss function. This revealed similar information to the plots shown in Figure 5.1, and tended to agree with these metrics. Generally, images far away from the central view exhibited lower loss according to LPIPS when a residual CNN was applied on top of image warping. However, images close to the central view showed the opposite behaviour, and had higher loss after incorporating the residual.

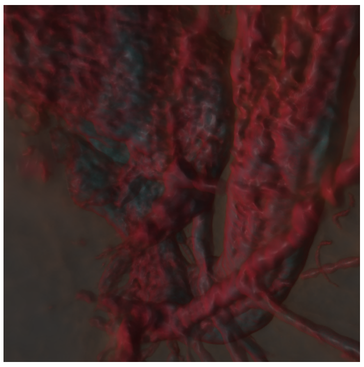
5.2 Qualitative Analysis of View Synthesis

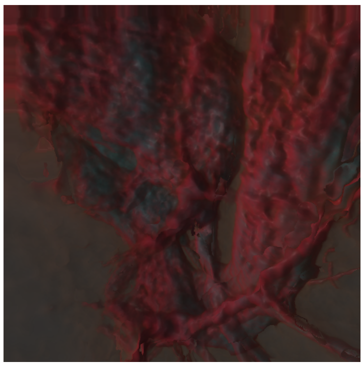
To investigate where the method performs effectively and inefficaciously, an example of a high, middling and low quality synthesised light field from the validation set is presented in Figure 5.3. Figure 5.3b is a poor reconstruction due to the opaque structure that should be present in the centre of the view. This structure is not picked up by the disparity map, resulting in a large crack appearing in the synthesised image. Figure 5.3e is a reasonably well synthesised view. The majority of the information is accurately shifted from the reference view, but some arteries lose their desired thickness and the image is not very sharp. Figure 5.3h is a very accurate synthesis. Some errors are seen around object borders, such as on the arch of the aorta, but overall it is hard to distinguish from the ground truth information. It is difficult to truly demonstrate effects such as the parallax between views on paper, so the ground truth and synthesised images in Figure 5.3 open an animated Graphics Interchange Format (GIF) when clicked. Additional qualitative results are presented in a supplementary video provided at `https://youtu.be/AQ4ec7Bgn1s`. This video demonstrates a comparison between synthesised light fields and ground truth light fields.

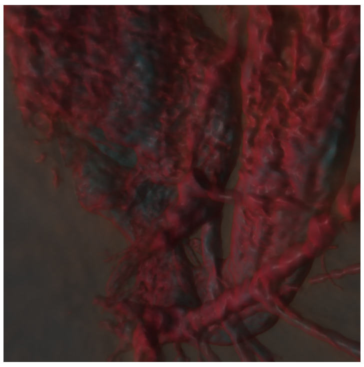

(a) Reference image

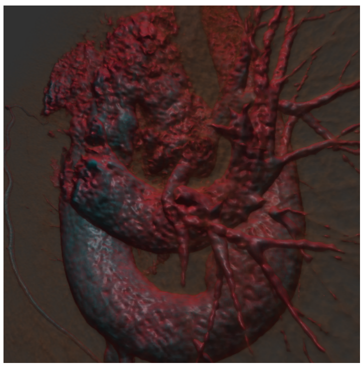

(b) Synthesised view

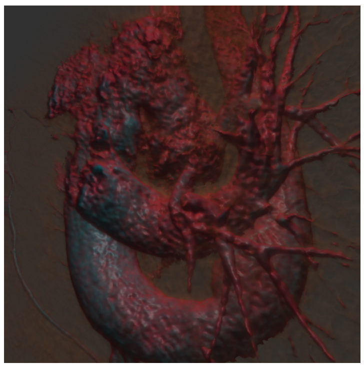

(c) Ground truth

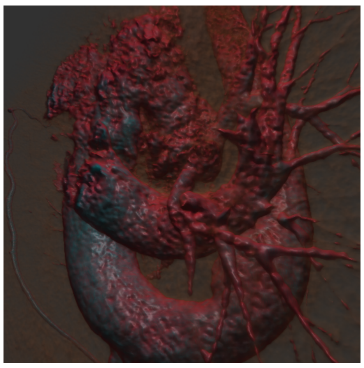

(d) Reference image

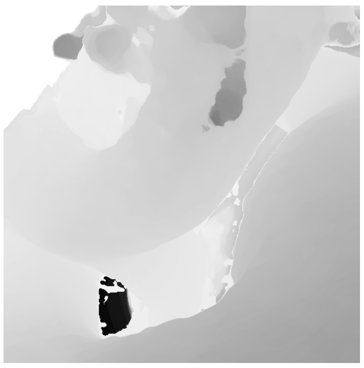

(e) Synthesised view

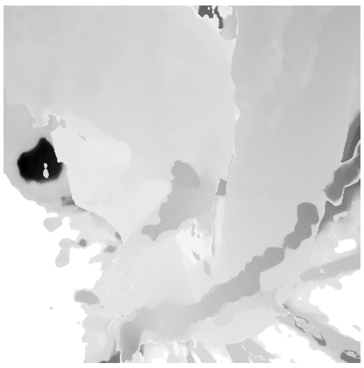

(f) Ground truth


(g) Reference image


(h) Synthesised view


(i) Ground truth


(j) Disparity map at (A)


(k) Disparity map at (D)

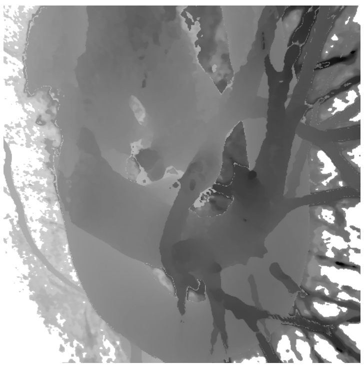

(l) Disparity map at (G)

Figure 5.3: Example synthesised upper-left images from the validation set. The first row has low performance, the second row has middling performance, and the third row has high performance. Disparity maps for the central reference views are presented in the final row. Clicking a ground truth or synthesised image opens an animated GIF in a browser.

5.3 Time Performance in Inviwo

Currently, the presented method is not fast enough to be useful for light field volume rendering at interactive rates. In Inviwo [Sun+15], synthesising and displaying a light field takes nearly 3.73 seconds if bilinear interpolation is used for backward warping. This time is broken down as follows. Rendering the reference view, and passing input and output information through Inviwo generally takes about 0.91 seconds. None of CNNs architectures tested exhibit slow enough performance to be the issue, completing a forward pass in at most 200 milliseconds. However, it is worth noting that all of 2D CNN architectures experimented with are faster than a 3D CNN architecture. As such, the time performance bottleneck is geometrical image warping, which takes approximately 2.77 seconds to warp a 512×512 image to a grid of 8×8 locations on the CPU. This is performed with bilinear interpolation, but removing the interpolation and using the value of the nearest neighbouring pixel still takes roughly 1.17 seconds on the CPU. The quality of the resulting images is similar in both cases, so bilinear interpolation could be removed without jeopardising quality significantly.

In the Inviwo [Sun+15] visualisation framework, directly rendering a light field directly takes close to 1.13 seconds. This is practically the same time as for a 2D CNN applied on top of image warping with nearest neighbours, disregarding the time to pass information through Inviwo. The positive point for the time of the CNN synthesis is that it has very little deviation. The time taken to directly volume render a light field depends heavily on the complexity of the scene. A CNN performs the same operations regardless of input complexity, which results in extremely steady performance. Additionally, the CNN performance is agnostic to the resolution of the volume data, and only depends on the spatial resolution of the reference image. As such, for very large complex volumes, this method would be applicable.

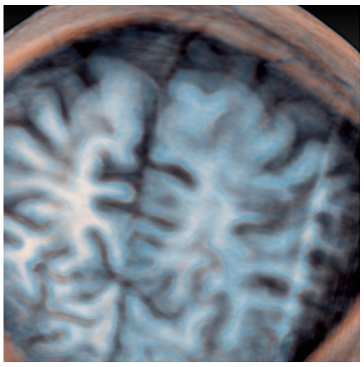
Because of this time drawback, a 3D CNN which directly took a reference view and associated depth map to perform view synthesis was tested. This took only 0.49 seconds on average, far faster than direct volume rendering. Unfortunately, the results were low quality, averaging 26.1 PSNR and 0.83 SSIM. In general, the CNN learnt decently well how to move information to new views, but the colour consistency between views was very low. This method could see use if a loss function was developed to penalise a lack of colour consistency between views.

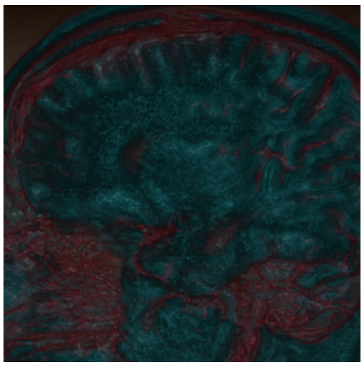
5.4 Performance on Unseen Data

Although the networks were all trained on one volume dataset with one transfer function, it is possible that the performance is maintained for different volumes and transfer functions (TFs). Additionally, the depth heuristic used during volume rendering seems reasonable, but there is no guarantee it would perform well with different volumes and TFs. Three experiments were performed with the AngularEDSR architecture, on ten sample light fields in each case. The new volume set chosen was a head MRI, available online [Erl18]. Results are presented in Table 5.2, and the central reference view and a sample warped view presented for each volume TF combination in Figure 5.4. The results suggest that the depth heuristic used and the image warping applied generalise well. Unsurprisingly, the AngularEDSR network fails to generalise to unseen volumes and transfer functions.

	Residual CNN		Warping Alone	
TF volume combination	**PSNR**	**SSIM**	**PSNR**	**SSIM**
Unseen TF, head MRI	34.46	0.955	36.50	0.956
Training TF, head MRI	40.18	0.949	41.43	0.949
Unseen TF, heart MRI	36.89	0.927	37.78	0.932

TABLE 5.2: Results on unseen transfer function and volume combinations during training.


(A) Unseen volume, Unseen TF


(B) Unseen volume Training TF

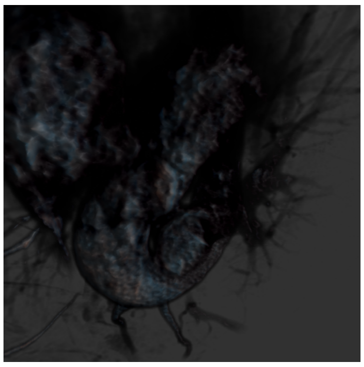

(C) Training volume Unseen TF

FIGURE 5.4: Sample reference views used for synthesis of light fields on unseen data.

5.5 Depth Heuristic Comparison

To compare the depth heuristics, ten light field samples were captured with volume clipping, and ten without volume clipping. Five depth maps are recorded:

1. The depth at 0.8 opacity during ray casting is recorded.
2. The depth at 0.8 opacity, and if that is not reached, 0.3 opacity during ray casting is recorded. This is the strategy that was selected for use in the training set, as it achieves the highest SSIM.
3. The depth at 0.7 opacity, and if that is not reached, 0.35 opacity during ray casting is recorded.
4. An isosurface at a value of 80 is precomputed on CPU, and the depth of this surface is recorded.
5. The depth of the first non-transparent voxel hit is recorded.

For the clipped volume set, isosurfaces are not applicable since they would have to be recomputed each time the clipping changes. Each of these depth maps is used to warp the central light field sample image to all 64 grid locations. The average PSNR and SSIM over the ten synthesised light fields for each different depth map are presented in Table 5.3. The depth of the first non-transparent voxel hit is not expected to perform well, and is provided for reference.

	Clipped		Unclipped	
Depth map type	**PSNR (db)**	**SSIM**	**PSNR**	**SSIM**
One depth (0.8)	33.23	0.915	34.63	0.907
Two depths (0.8, 0.3)	34.00	**0.925**	35.95	**0.922**
Two depths (0.7, 0.35)	**36.47**	0.920	**35.97**	**0.922**
Isosurface depth (80)	n/a	n/a	35.01	0.909
First hit depth	27.38	0.818	27.96	0.802

TABLE 5.3: Comparing average quantitative results with different depth maps used for view synthesis on ten synthesised light fields.

6 Conclusion and Future Work

6.1 Conclusion

This research has promising results, but requires further investigation to remove some limitations. Certainly, deep learning has high potential for application to view synthesis in light field volume rendering. Learning a residual light field improves the visual consistency of the geometrically based warping function, especially for views far away from the reference view. Additionally, producing the entire light field in one step lends to a global sense of view synthesis, with better visual quality than producing views individually at faster speed. Regrettably, the method of Srinivasan et al. [Sri+17] experimented with does not function particularly well for volumes. This is because their method deals with pictures of flowers, with the flower near the centre of the image, so the most important information in the scene never goes out of sight. For volumes, this happens regularly, and the CNNs applied never learnt to fill in the missing information. However, the drawback of the 3D convolutions they use can be effectively removed by remapping the network to use 2D convolutions.

Light field synthesis is performed quickly compared to existing methods, but is still too slow to compete with direct volume rendering. However, in contrast to light field volume rendering, the time to completion is independent of the volume resolution, and only depends on the resolution of the sample volume rendered image. Encouragingly, a low percentage of this time is spent on the CNN, with geometrical image warping being the performance bottleneck. Although image warping is performed on the CPU due to GPU memory limitations, the GPU based warping from [Sri+17] is also a performance bottleneck. For images of size 192×192, their GPU accelerated warping takes 0.13 seconds, while our CPU warping with bilinear interpolation takes 0.17 seconds. In conclusion, our view synthesis results for light field volume rendering using CNNs are of high quality and deep learning can be effectively be applied to this problem, but the geometrical image warping procedure prevents synthesis at interactive rates and requires further research.

6.2 Future Work

There is a myriad of avenues to expand upon this body of work in the future. Of course, there is the clear expansion of developing a faster geometrical image warping procedure, or removing this step and directly train a CNN to perform image warping. Additionally, there is multiple more succinct points for further research, many of which are listed below.

- **Switching the depth heuristic:** At runtime, the transfer function and volume could be evaluated and the most appropriate depth heuristic chosen. For example, if the transfer function maps a value to a colour with full opacity, it would tend to reveal an isosurface at this value. The depth heuristic could then be chosen to mimic the depth at this isosurface.
- **Multiple depth maps:** Further to the above point, multiple depth heuristics could be used to form depth maps for the foreground, middleground and background layers. Then a CNN can be used to segment the input images into these layers, and the three layers warped using the corresponding depth maps. This extra information might allow the CNN to learn to better handle occlusions, and depth map inaccuracy. Alternatively, these depth maps could be input into a CNN and be combined to form one improved depth map.
- **Learning over multiple volumes and transfer functions:** In this project, training was performed on one particular volume for a specific transfer function to test validity of the method. Through experimentation, it was shown that this learning did not generalise well to different volumes and transfer functions However, it would be beneficial to learn a universal view synthesis for light field volume rendering.
- **Incorporating additional volume information:** The only information extracted from the volume dataset is a depth map and rendered view. It would be possible to develop a mixed method which learns on the actual volume data as well as the depth and rendered image. Unfortunately, the extra time this would take would be a difficulty. Additionally, learning the correct information from the volume would be challenging.
- **Experimenting with additional reference views:** Only one reference view was tested, and it would be interesting to experiment with multiple input views. It is uncertain if the extra information available would be worth the extra time spent on volume rendering.

Bibliography

[Ade+91] Edward H Adelson et al. "The plenoptic function and the elements of early vision". In: *Computational Models of Visual Processing.* MIT, 1991, pp. 3–20.

[Gor+96] Steven J. Gortler et al. "The Lumigraph". In: *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques.* SIGGRAPH '96. ACM, 1996, pp. 43–54.

[LH96] Marc Levoy and Pat Hanrahan. "Light Field Rendering". In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques.* SIGGRAPH '96. ACM, 1996, pp. 31–42.

[The97] The HDF Group. *Hierarchical Data Format, version 5.* http://www.hdfgroup.org/HDF5/. 1997-2018.

[IMG00] Aaron Isaksen, Leonard McMillan, and Steven J. Gortler. "Dynamically Reparameterized Light Fields". In: *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques.* SIGGRAPH '00. ACM, 2000, pp. 297–306.

[LS04] Zhouchen Lin and Heung-Yeung Shum. "A Geometric Analysis of Light Field Rendering". In: *International Journal of Computer Vision* 58.2 (2004), pp. 121–138.

[Wan+04] Zhou Wang et al. "Image Quality Assessment: From Error Visibility to Structural Similarity". In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612.

[Eng+06] Klaus Engel et al. *Real-time volume graphics.* AK Peters/CRC Press, 2006.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems.* 2012, pp. 1097–1105.

[WG12] Sven Wanner and Bastian Goldluecke.
"Spatial and angular variational super-resolution of 4D light fields".
In: *European Conference on Computer Vision.* Springer. 2012,
pp. 608–621.

[ZAL12] Stefan Zellmann, Martin Aumüller, and Ulrich Lang.
"Image-based remote real-time volume rendering: Decoupling rendering from view point updates".
In: *ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference.*
American Society of Mechanical Engineers. 2012, pp. 1385–1394.

[WMG13] Sven Wanner, Stephan Meister, and Bastian Goldluecke.
"Datasets and benchmarks for densely sampled 4d light fields."
In: *Vision, Modeling, and Visualization.*
Ed. by Matthias Hullin, Marc Stamminger, and Tino Weinkauf. 2013.

[Shi+14] Lixin Shi et al. "Light Field Reconstruction Using Sparsity in the Continuous Fourier Domain".
In: *ACM Transactions on Graphics* 34.1 (2014), pp. 1–13.

[WG14] Sven Wanner and Bastian Goldluecke. "Variational Light Field Analysis for Disparity Estimation and Super-Resolution". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.3 (2014), pp. 606–619.

[Sun+15] Erik Sundén et al.
"Inviwo - An extensible, multi-purpose visualization framework".
In: *IEEE Scientific Visualization Conference (SciVis).* 2015, pp. 163–164.

[Yoo+15] Youngjin Yoon et al. "Learning a Deep Convolutional Network for Light-Field Image Super-Resolution". In: *Proceedings of the IEEE International Conference on Computer Vision Workshops.* 2015,
pp. 24–32.

[He+16] Kaiming He et al. "Deep residual learning for image recognition".
In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2016, pp. 770–778.

[KWR16] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi.
"Learning-based View Synthesis for Light Field Cameras".
In: *ACM Transactions on Graphics* 35.6 (Nov. 2016), 193:1–193:10.

[Loc+16] Gerrit Lochmann et al. "Real-time Novel-view Synthesis for Volume Rendering Using a Piecewise-analytic Representation".
In: *Vision, Modeling and Visualization.*

Ed. by Matthias Hullin, Marc Stamminger, and Tino Weinkauf. The Eurographics Association, 2016.

[Qi+16] Charles R Qi et al. "Volumetric and multi-view cnns for object classification on 3d data". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 5648–5656.

[Wan+16] Ting-Chun Wang et al. "A 4D light-field dataset and CNN architectures for material recognition". In: *European Conference on Computer Vision*. Springer. 2016, pp. 121–138.

[Zho+16] Tinghui Zhou et al. "View synthesis by appearance flow". In: *European conference on computer vision*. Springer. 2016, pp. 286–301.

[Lim+17] Bee Lim et al. "Enhanced deep residual networks for single image super-resolution". In: *The IEEE conference on computer vision and pattern recognition (CVPR) workshops*. Vol. 1. 2. 2017, p. 4.

[LH17] Ilya Loshchilov and Frank Hutter. "SGDR: Stochastic gradient descent with warm restarts". In: *International Conference on Learning Representations*. 2017.

[Pas+17] Adam Paszke et al. "Automatic differentiation in pytorch". In: (2017).

[Smi17] Leslie N Smith. "Cyclical learning rates for training neural networks". In: *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*. IEEE. 2017, pp. 464–472.

[Sri+17] P. P. Srinivasan et al. "Srinivasan, Pratul P and Wang, Tongzhou and Sreelal, Ashwin and Ramamoorthi, Ravi and Ng, Ren". In: *IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2262–2270.

[Wu+17a] Gaochang Wu et al. "Light Field Image Processing: An Overview". In: *IEEE Journal of Selected Topics in Signal Processing* 11.7 (2017), pp. 926–954.

[Wu+17b] Gaochang Wu et al. "Light Field Reconstruction Using Deep Convolutional Network on EPI". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1638–1646.

[Yoo+17] Youngjin Yoon et al. "Light-Field Image Super-Resolution Using Convolutional Neural Network". In: *IEEE Signal Processing Letters* 24.6 (2017), pp. 848–852.

[Erl18] University of Erlangen. *Head Volume Dataset.* `http://schorsch.efi.fh-nuernberg.de/data/volume/MRI-Head.pvm.sav`. Accessed: 24/08/2018. 2018.

[Roe18] Stefan Roettger. *Heart Volume Dataset.* `http://schorsch.efi.fh-nuernberg.de/data/volume/Subclavia.pvm.sav`. Accessed: 15/08/2018. 2018.

[Shi+18] Changha Shin et al. "EPINET: A Fully-Convolutional Neural Network Using Epipolar Geometry for Depth from Light Field Images". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2018, pp. 4748–4757.

[VBG18] Suren Vagharshakyan, Robert Bregovic, and Atanas Gotchev. "Light Field Reconstruction Using Shearlet Transform". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.1 (2018), pp. 133–147.

[Zha+18] Richard Zhang et al. "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2018.