Using Light Fields to Enable Deep Monocular Depth Estimation

Niall Hunt, 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 Computer Science

Supervisors: Dr. Martin Alain and Prof. Aljosa Smolic

August 2021

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.

Niall Hunt

August 4, 2021

Permission to Lend and/or Copy

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

Niall Hunt

August 4, 2021

Using Light Fields to Enable Deep Monocular Depth Estimation

Niall Hunt, Master of Computer Science

University of Dublin, Trinity College, 2021

Supervisors: Dr. Martin Alain and Prof. Aljosa Smolic

Knowing the depth of a scene is a vital part of many computer vision applications, ranging from autonomous driving to augmented reality. Monocular depth estimation is the technique of predicting the depth of a scene given a single image. This ill-posed problem is increasingly being tackled with end-to-end neural networks thanks to advancements in machine learning. Deep learning techniques require large datasets with accurate ground truth depth maps for training. There are many features within an image that are used to decipher depth, these include: texture, object size and defocused blur. The large datasets that are provided for monocular depth estimation by nature do not include all of these features.

This project studies the state of the art in monocular depth estimation and their ability to generalise to an unseen dataset that includes images with features across all-in-focus images and images with a defocus blur. This dataset consists of various four-dimensional light fields which unlike traditional two-dimensional images, capture both the spatial intensity and angular direction of light rays. We use these light fields to create three distinct datasets (all-in-focus, front focus and back focus), using a shift-and-sum algorithm. These datasets are used to study and evaluate the performance of the chosen monocular depth estimation techniques.

In this dissertation we present the findings of the study, showing that despite the advancements made in monocular depth estimation, there is still a large gap in performance in comparison to other depth estimation techniques and that the performance of the state of the art cannot be predicted across the three datasets.

The code that accompanies this work is available at:
`https://github.com/NiallEHunt/MonocularDepth-Using-LightFields`

Acknowledgments

I would like to thank Dr. Martin Alain for his support and guidance over the past year. Without his time and effort spent answering questions and providing helpful suggestions this work would not have been possible.

I would also like to thank Prof. Aljosa Smolic and Prof. Rachel McDonnell for their feedback and guidance after my presentation.

Most importantly, I would like to thank my parents and friends for supporting me and helping me through the last five years of my education. Without them, I would not have made it this far.

Niall Hunt

University of Dublin, Trinity College
August 2021

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Knowing the depth of a scene is an important task in computer vision. There are many current techniques for estimating a depth map from a single image. In this dissertation we explore the usage of four-dimensional light fields to evaluate the performance of existing depth estimation techniques and suggest future avenues of research into improving these techniques with the use of light fields.

1.1 Motivation and goals

Measuring the distance of an object relative to a camera is a vital part of many computer vision applications. Understanding the three-dimensional world through computer vision allows applications to interact with and understand their environment. This can be used in many exciting ways. Some of these include: three-dimensional modelling [16], foreground-background segmentation [7], robot navigation [8], autonomous driving [41] and augmented reality [26].

Traditional methods of depth estimation often include stereoscopic imagery or Lidar (Light detection and ranging) technology. While these methods are accurate, they are not suitable or practical for every application. In many instances the depth information of a scene is needed for an application where only a single camera is available. For example, an augmented reality phone application only has access to the phone's camera. For these situations, we use monocular depth estimation methods.

Monocular depth estimation uses a single input image rather than stereoscopic images to estimate the depth map of the captured scene. Monocular methods are primarily split into methods that use all-in-focus images and methods that utilize depth cues from defocused images. In this paper we refer to images with a defocused blur outside of the focal plane as defocused images. See figure 1.1 for examples of an all in focus image and defocused images.

(a) All-in-focus image (b) Front focused image (c) Back focused image

Figure 1.1: The centre view of the table light field from [18]. (a) shows the all-in-focus view (b) shows the defocused view with the focus plane near the camera (c) shows the defocused view with the focus plane far from the camera.

All-in-focus depth estimation models tend to rely on perspective, size, texture and shading cues to estimate the depth in an image. These factors can vary greatly depending on the scene and the camera used to take the image. Thus, all-in-focus methods tend not to generalise well to unseen data. Geiger et al. (2013) [11] provide the KITTI dataset which is widely used for training all-in-focus monocular depth estimation models. The images from this dataset are captured from the roof of a car. Models trained solely on this dataset may not perform well on images captured by a phone camera, for example, due to the differing perspective and image quality.

On the other hand, depth from defocus estimation models use defocus clues to estimate the depth in an image, often relying on the strength of the defocus blur to determine the depth of the scene. A study has not been done on the effectiveness of these methods on all in focus input images and so we cannot say if they generalise

well to all-in-focus images or not. Intuitively, we may think they will not perform well on all-in-focus images as their learned features relate to the blur seen in the image and all in focus images do not contain the same blur. Conversely, all-in-focus models may not perform well on defocused images as their learned features rely on geometric shapes, sizes and textures which lose detail in defocused areas of an image.

The aim of this project is to evaluate the performance of the existing state of the art on unseen all-in-focus images as well as unseen images with a defocused blur. We aim to study the ability of the state of the art to generalise to unseen datasets and to compare their results to the performance of light field depth estimation methods.

1.2 Project overview

For this project we evaluated four state of the art monocular depth estimation models on a range of input images. We used the HCI light field dataset provided by Honauer et al. (2016) [18] to generate all-in-focus and defocused images to test the chosen models. We used a shift-and-sum algorithm to generate the defocused images from the light fields. Figure 1.1 shows the generated all-in-focus and defocused images for the table scene in the HCI dataset.

Using the benchmark set of light fields from the HCI dataset (see figures A.1, A.2 and A.3 for the full benchmark dataset) we created three distinct test sets: all-in-focus, front-focus and back-focus. These test sets consist of the generated all-in-focus and defocused images for the benchmark light fields. The front-focus images have the focus plane near the camera while the back-focus images have the focus plane far from the camera. In both cases objects outside of the focus plane have a defocus blur, the intensity of which is dependent on its distance from the focus plane.

The three generated test sets were used as inputs into the four monocular depth estimation models chosen for this study. The evaluation methodology and code accompanying the HCI dataset [22] were used to evaluate the accuracy of these estimated depth maps. We compare the results of each model across the three test datasets. We found that the performance of the models across all-in-focus and

defocused images could not be predicted and that the monocular depth estimation models performed significantly worse than the light field depth estimation models.

1.3 Structure of dissertation

The rest of this dissertation is structured as follows. Chapter 2 provides a summary of the background research that was carried out as a part of this project. Here a summary of depth estimation, namely traditional and monocular depth estimation, is provided as well as a summary of light field imagery. Chapter 3 covers the study of current deep monocular depth estimation techniques and their evaluation using a light field dataset. Finally, chapter 4 summarises the conclusions of this project. Here a summary of the project's findings are discussed and suggestions for future work, including possible improvements to deep monocular depth estimation using light fields, are provided.

Chapter 2

Background Research

The goal of this project is to evaluate the performance of deep monocular depth estimation using light fields. This chapter summarises the background research that was carried out as part of this project. Section 2.1 covers the depth estimation techniques, while section 2.2 provides a summary of light field imagery.

2.1 Depth estimation

The goal of depth estimation is to generate the depth map of a scene from a certain viewpoint. This depth map is a representation of the spatial structure of the scene and encodes the z-axis as a depth value (see figure 2.1b). This spatial information is vital for many computer vision applications. There are many techniques for depth estimation that have evolved over the years alongside technological advancements. These techniques are split into traditional depth estimation and monocular depth estimation.

2.1.1 Traditional depth estimation

Traditional depth estimation methods are generally broken into binocular or multi-view methods and sensor based methods.

(a) Original RGB scene (b) Depth map (c) Disparity map

Figure 2.1: The centre view of the table light field from [18]. (a) shows the original RGB image (b) shows the ground truth depth map, the lighter the pixel the larger its depth value (i.e. it is further from the camera) (c) shows the ground truth disparity map, the lighter pixels have a greater disparity between images.

Binocular depth estimation

Multi-view methods use multiple cameras to capture a scene and use triangulation to estimate the depth. These methods primarily use two cameras, hence binocular depth estimation. Binocular depth estimation mimics one of the processes through which humans perceive depth [33].

This method captures two images of a scene separated by a distance, similar to human vision in which images are captured by two eyes. These two images form a stereo pair with a left image and right image. The displacement of an object between the left and right image is called the disparity. You can see this disparity by simply looking at your finger held in front of you while alternately closing an eye (figure 2.2 shows an example of this). The disparity is calculated for every pixel to form a disparity map [34] (see figure 2.1c).

To calculate the disparity, first the location of matching pixels from each image need to be found. This process is called stereo matching. Careful calibration and image rectification are required before calculating the disparity. These steps project the images onto a common plane known as an epipolar plane, thus allowing the vertical parallax between the two images to be omitted. Stereo matching can then be constrained to searching a single horizontal line [37]. Further processing

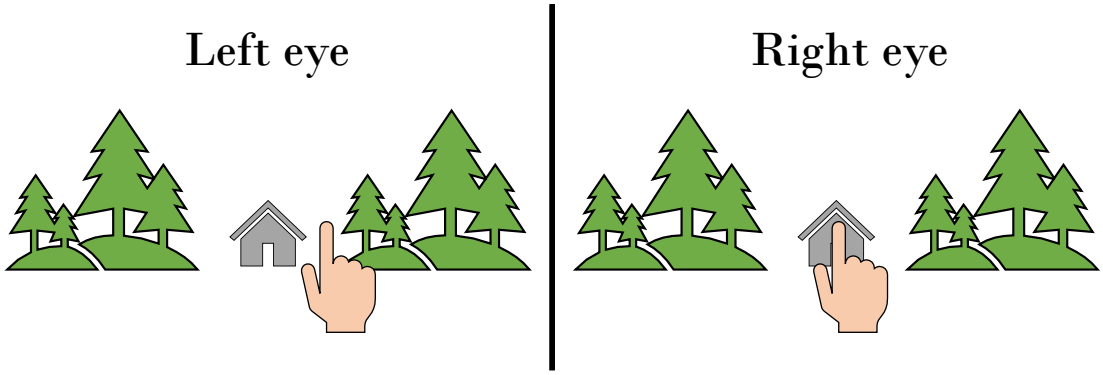


Figure 2.2: An example of the disparity between the stereo pair of images captured by human eyes.

is done to make the images equivalent to that taken from a pinhole camera.

We can use the relationship between depth and disparity, seen in figure 2.3, to obtain the depth value. Through similar triangles we can obtain the depth value shown in equation 2.1. The derivation for this equation is included in the appendix in section A.1.

$$z = \frac{fb}{d} \tag{2.1}$$

Sensor-based depth estimation

Sensor based depth estimation methods use specifically designed sensors to measure the distance to an object. A common example of this methodology is Lidar (light detection and ranging). Lidar is a method of determining depth using light in the form of a pulsed laser [31]. The time taken for the light to reflect off of an object and back to the sensor is recorded using a time-of-flight camera. The distance traveled by the beam is calculated using the speed of light and the time taken for the light to return to the sensor. This type of Lidar provides an accurate depth measurement for the singular point at which it was aimed. To create a depth map of a scene, the depth is measured point by point. The beam scans the scene measuring the depth at each point and the output is stored as a depth map.

Lidar sensors are currently used in many applications. They provide accurate depth information for autonomous vehicles [9] and three-dimensional mapping [25].

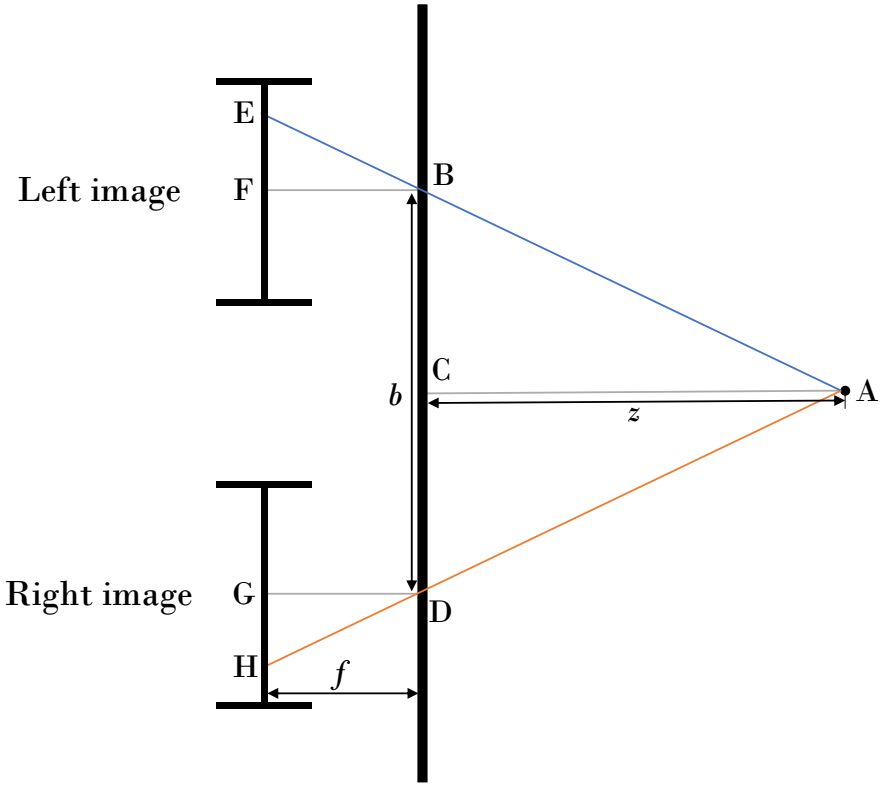


Figure 2.3: This figure shows the relationship between disparity and depth for stereoscopic imagery. The figure assumes the left and right images are co-planar and are taken by pinhole cameras. A is an object that we are capturing. B and D are the entry points to the two pinhole cameras through which the light rays are captured.

While these sensors provide accurate depth maps, they are prohibitively expensive for many applications.

2.1.2 Monocular depth estimation

The goal of monocular depth estimation is to generate a depth map for a scene from a single RGB image. While traditional methods use complicated setups with carefully calibrated cameras or expensive Lidar, monocular depth estimation aims to provide depth estimation for simpler and cheaper single camera setups.

Early work in this field used traditional computer vision techniques to identify

key features in images. Saxena et al. (2006) [35] used a discriminatively trained Markov Random Field model on multi-scale local and global image features. The authors created a dataset using a laser range finder and a camera. They were able to reproduce fairly accurate depth maps despite some errors and inaccuracies in their dataset.

Since [35], there have been many advancements in monocular depth estimation. Newer computer vision and deep learning techniques and larger, higher quality datasets have led to significant improvements in the state-of-the-art. Supervised, semi-supervised and unsupervised machine learning techniques have been utilised in monocular depth estimation [42]. Generally supervised models perform very well when testing on the dataset used to train them but do not generalize well to unseen data, while semi-supervised and unsupervised models are better at generalising but do not achieve the same accuracy. There are two main types of images used for training deep monocular depth estimation models: all-in-focus images and defocused images (see figure 1.1 for examples of all-in-focus and defocused images). This study evaluates the performance of three supervised models and one self-supervised model (outlined below) on unseen all-in-focus and defocused images.

The chosen models for this study are: **DenseDepth**[3], **GDN-Pytorch**[36], **Defocus-Net**[27] and **Monodepth2**[13]. DenseDepth uses an encoder-decoder network and transfer learning to predict depth maps for a given image. GDN-Pytorch uses a "guided" depth-to-depth network to restore accurate boundaries in the predicted depth maps from the color-to-depth network. Defocus-Net takes a focal stack (a stack of images of the same scene with different focus distances) and first generates a defocus map, this defocus map is used as input to their depth estimation network. Monodepth2 uses a self-supervised network that is trained with stereo imagery or consecutive temporal frames from a monocular video. The architectures of the chosen models are studied in detail in chapter 3.

2.2 Light fields

The term light field was first introduced by Gershun (1936) [12]. A light field is a description of the amount of light flowing in every direction through every point in space. The five-dimensional plenoptic function $L(x, y, z, \theta, \phi)$ gives the space of

all possible light rays [1]. The five dimensions of the function consider the point in space (x, y, z) as well as the direction (θ, ϕ) of the rays.

The five-dimensional plenoptic function can be reduced to a four-dimensional function, given the radiance of a ray remains constant along its length. If the space considered is free of occluders, one of the five dimensions becomes redundant. This four-dimensional light field was referred to as the *photic field* by Parry Moon (1981) [28]. However, it is more commonly known as the *4D light field* or the *Lumigraph* after its introduction into computer graphics by Levoy and Hanrahan (1996) [24] (4D Light Field) and Gortler et al. (1996) [14] (Lumigraph). These four-dimensional light fields capture not just the spatial intensity of the light rays like traditional photography but they also capture the angular direction of the light rays.

2.2.1 Light field representation

Two-plane parameterisation

The most common way to parameterise a light field is known as the two-plane parameterisation. This parameterisation was proposed by Levoy and Hanrahan (1996) and a similar technique was used by Gortler et al. (1996). In this parameterisation, a light ray, $L(u, v, s, t)$, is parameterised by its intersection with two parallel planes (see figure 2.4).

Using this parameterisation, we can create a representation of the 4D light field composed of a 2D array of 2D images [24, 14] (see figure 2.5). By placing a camera on the uv plane with its focal plane on the st plane, we can take pictures that capture a 2D slice of the light field at a fixed uv value. This method captures a uv array of st images, where each image is from a different viewpoint. The resolution of the uv plane is known as the *angular resolution* and specifies the number of images taken to capture the light field. The resolution of the st plane is known as the *spatial resolution* and corresponds with the dimensions of the captured images. In the example shown in figure 2.5 the angular resolution is 9 x 9 while the spatial resolution is 512 x 512.

The light field can also be visualised as a st array of uv images. In this visualisation each image represents the rays leaving a singular point on the st plane

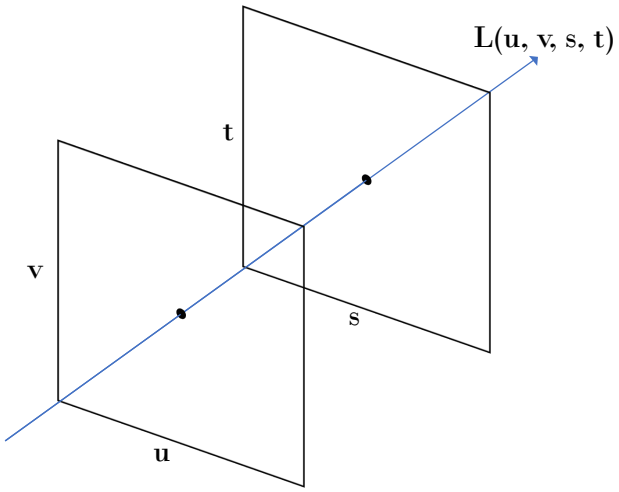


Figure 2.4: Two-plane parameterisation for 4D light fields. Here the ray of light $L(u, v, s, t)$ intersects the uv plane and then the st plane.

and arriving at all points on the uv plane. In other words the spatial coordinates s and t are fixed and the images show the point (s, t) from all points on the uv plane. This can be seen in figure 2.6, which is taken from [24].

Epipolar plane images

Light fields can be represented using epipolar plane images (EPIs). Bolles et al. (1987) first created EPIs for motion analysis and three-dimensional representation of scenes [4]. The proposed technique captured many closely spaced images along a linear camera path. If the camera is aimed perpendicularly to its trajectory, the epipolar lines correspond with the horizontal scan lines. All points in an epipolar plane are projected onto an epipolar line in one image and the corresponding epipolar line in the second image. With the given constraints, any point along a horizontal scan line will appear along that same scan line in the next image in the sequence. Using this knowledge, an EPI can be created by combining corresponding epipolar lines from every image in the sequence (i.e. stack the corresponding horizontal scan lines from each image in the sequence).

This technique can be extended to light fields. Each row and column of a 4D

Figure 2.5: A uv array of st images. Each image is captured from a different viewpoint (i.e. a different u and v value of the uv plane). The image on the right shows the st image at $u = 5$ and $v = 5$.

light field can be seen as a sequence of closely spaced images along a linear camera path. This allows an EPI to be computed by fixing an angular dimension and a spatial dimension [15]. This gives a series of epipolar lines which can be combined into an EPI for those angular and spatial dimensions. This can be seen in figure 2.7.

The resultant epipolar plane image (see figure 2.7 for an example) shows a series of lines. Each line corresponds to a feature point in the scene. The slope of the line is related to the disparity of the feature point between images. This relation can be exploited to robustly predict the depth of scene features (disparity is related to depth as discussed in section 2.1.1). For this reason, EPI representations of light fields are often used in light field depth estimation techniques.

2.2.2 Light field depth estimation

To predict the depth of a scene using light fields traditional stereoscopic techniques can be used. With this technique the disparity between two images within a 4D light field would be estimated and subsequently converted to a depth estimate,

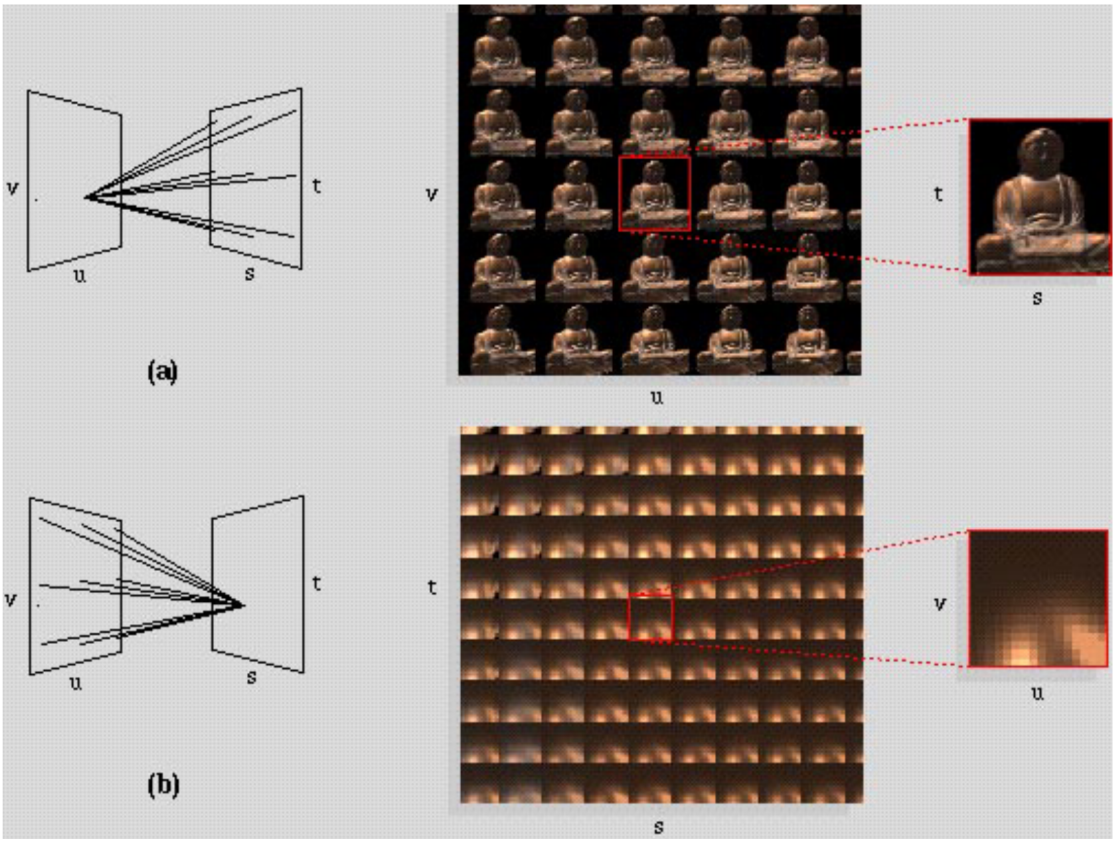


Figure 2.6: This image shows two visualisations of a light field. (a) shows the *uv* array of *st* images, where each image captures all light rays arriving to a point on the *uv* plane from the *st* plane. (b) shows the *st* array of *uv* images, where each image captures all rays leaving a point on the *st* plane from all points on the *uv* plane. Figure is from [24]

however, such a technique would fail to take advantage of the full structure of the light field. This technique can be extended to multi-view depth estimation where each image within the light field is used [21], thus utilizing the full data captured within a light field.

As mentioned previously in section 2.2.1, there is a relationship between the slopes in an epipolar plane image and the depth of the corresponding feature point. Wanner and Goldluecke (2012) [40] use the slopes of the lines in an EPI to predict the depth of the scene captured by the light field. Deep learning techniques have subsequently been used to improve upon these methods, many using EPIs as input

to their models.

Chen et al. (2021) [5] propose an attention-based four-branch LF depth estimation network. The four branch inputs are EPIs at various angles. Weights are shared between the feature extractors for each branch. The features are passed through two steps called the intra-branch feature fusion step and the inter-branch feature fusion step, where skip connections are used. The full architecture for this model is shown in figure 2.8. The proposed model (AttMLFNett) at the time of writing is ranked first in the HCI benchmark [22]. We use this model later in the study for comparison with the monocular depth estimation models.

2.2.3 Light field image refocusing

Using the information captured in a light field, it is possible to create a digitally refocused image of a scene [30]. Ng et al. (2005) derive the equation (equation 2.2) for this technique. This technique can be conceptualised as a linear combination of shifted light field views. Hence, this technique is known as shift-and-sum algorithm. An example of this can be seen in figure 1.1, the refocused images in this figure were created using the shift-and-sum algorithm.

$$\bar{E}(s', t') = L(s' + \frac{u_o - s'}{\beta}, t' + \frac{v_o - t'}{\beta}, s', t') \tag{2.2}$$

The shift-and-sum algorithm is used to create a single refocus image where the refocus plane is parallel to the light field planes (also known as frontoparallel). Other techniques allow for non-frontoparallel refocus planes [20, 2]. The creation of a refocused image of a scene is a useful application in and of itself, however, it can also be used in conjunction with other image processing techniques to enable many new applications.

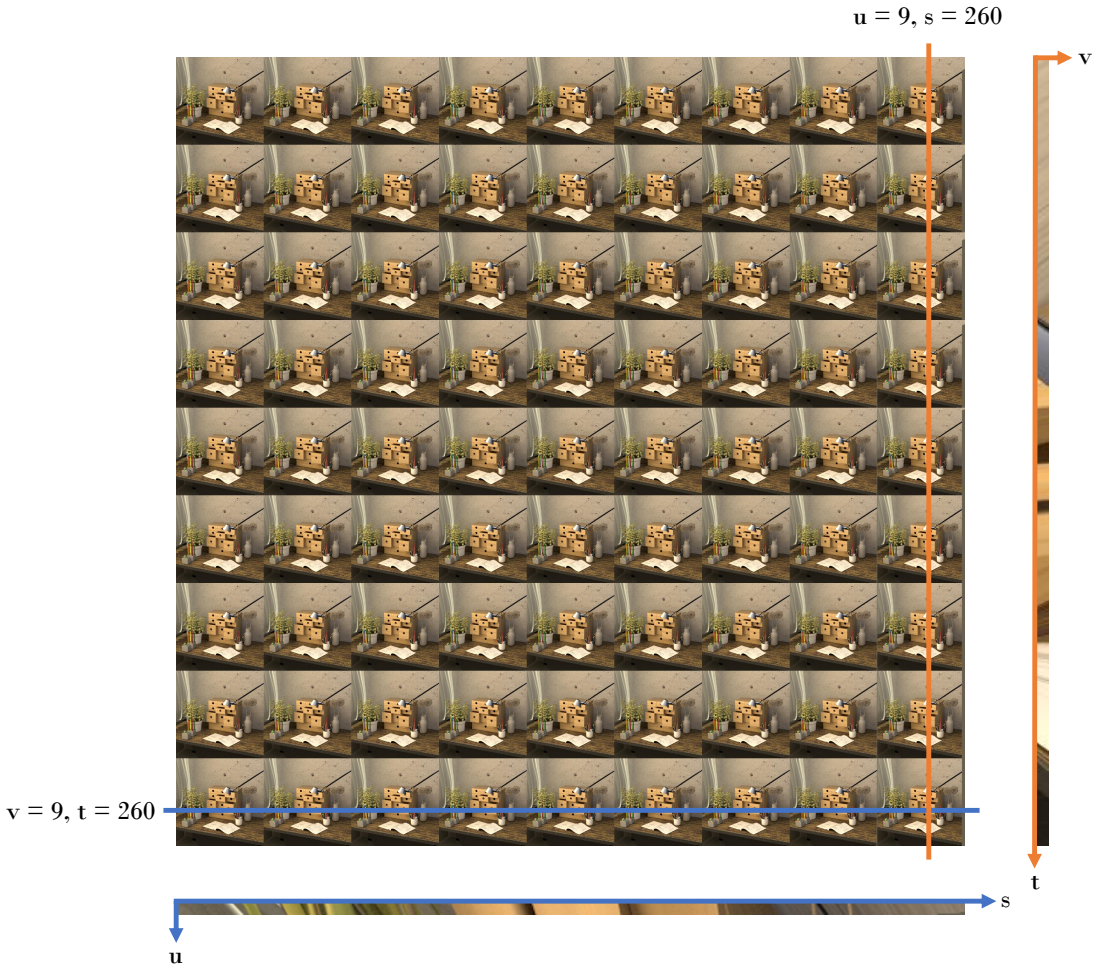


Figure 2.7: Shows a light field and two generated epipolar plane images. The vertical and horizontal EPIs are generated from the corresponding lines through the light field.

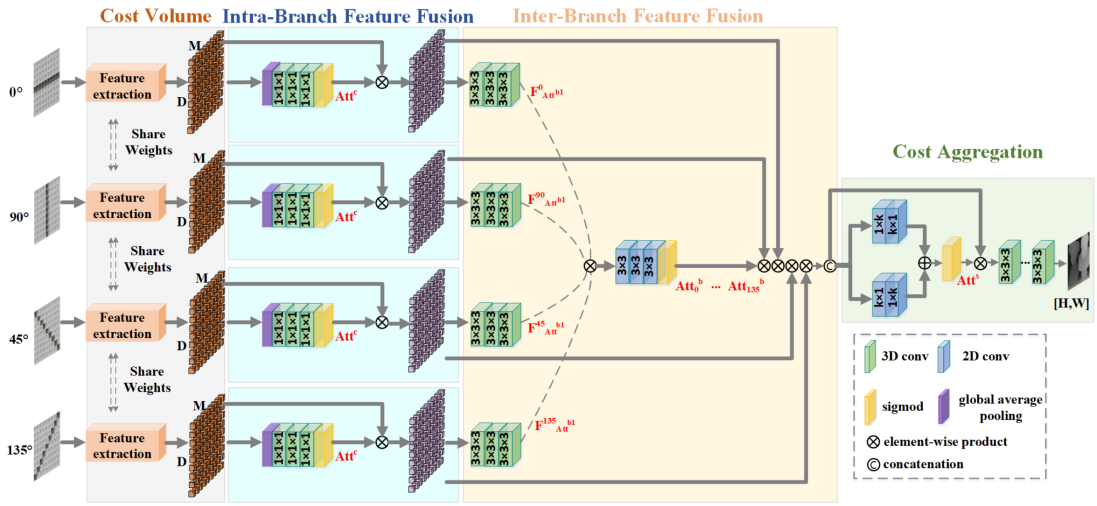


Figure 2.8: The proposed architecture for **AttMLFNett**. Figure from [5]

Chapter 3

Study of Monocular Depth Estimation using Light Fields

The goal of this project is to evaluate the performance of the state of the art in monocular depth estimation and their ability to generalise to an unseen dataset. This dataset should encompass all-in-focus images and defocused images (images with a blur in regions that are not in focus, see figure 1.1 for an example) to further test the generalisation of the models.

The study uses the models outlined in section 3.2, namely: **DenseDepth**, **GDN-Pytorch**, **Monodepth2** and **Defocus-Net**. These models are tested using the **HCI** dataset and evaluation methodology.

In this chapter the architectures of the chosen models are studied and the evaluation methodology of this study is presented along with the evaluation metrics used. Furthermore, the experimental results are compared across the four aforementioned models.

3.1 Overview

The experiment consisted of testing the models on three distinct datasets (all-in-focus, front focus and back focus) and comparing their results. The models were chosen due to the performance reported in their respective papers as well as the availability of their codebases. All models chosen were published and freely avail-

able on GitHub at the time of writing and all (excluding Defocus-Net) provided pre-trained models. When training Defocus-Net for use in this project, we were unable to achieve the same results reported in [27].

The three datasets used in this study were generated from the HCI benchmark dataset. The generated datasets were used as input for each model covered by the study. In some cases, a model required an input image of a different resolution to our datasets (512 x 512). In these cases, we resized the images using bilinear interpolation before depth prediction. The predicted depth maps were converted to disparity maps and evaluated. The results are discussed in section 3.5. Figure 3.1 shows an overview of the experiment setup.

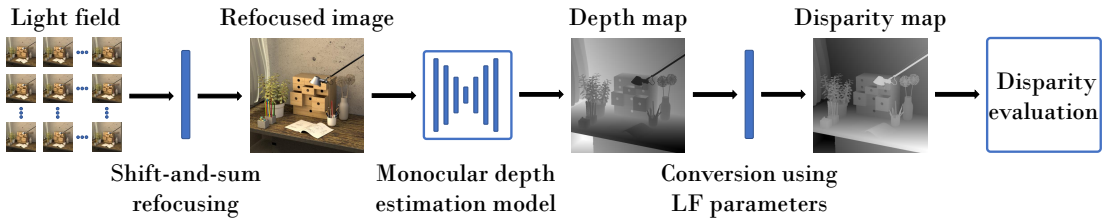


Figure 3.1: An overview of the workflow for the study. It shows the end-to-end processing and evaluation that was completed for each model.

We use light fields to enable this evaluation. Light fields allow us to shift the focal plane of a scene to create new refocused images. Thus, allowing us to generate an arbitrary number of refocused datasets of the same scenes, where the only differences are the varying focal lengths. Using a shift-and-sum algorithm along with the light field parameters we can specify the focal plane as well as the amount of defocus blur. This allows us to study the generalisation and performance of monocular depth estimation methods across all-in-focus and defocused images.

As discussed in section 2.2.2, light field depth estimation techniques perform very well given the volume of information captured in a light field. We use a state-of-the-art light field depth estimation model in our comparisons to highlight the gap in performance between monocular estimation methods and light field methods. The chosen model is known as **AttMLFNett** and is outlined in section 2.2.2.

3.2 Chosen monocular depth models

3.2.1 Depth from all-in-focus images

GDN-Pytorch Song and Kim (2019) [36] propose a depth estimation method consisting of two networks: a "depth-to-depth" network referred to as the guided network and a "color-to-depth" network. The ground truth depth maps are used to train the guided network with the purpose of restoring the depth map. This encodes the geometrical structure of the depth-to-depth relationship in the latent space of the guided network. This guided network is then used to efficiently guide the learning process of the color-to-depth network. The generated depth maps from the color-to-depth map are fed into the guided network along with the ground truth depth map to calculate a latent loss which is used to train the model. Figure 3.2 shows the architecture diagram proposed.

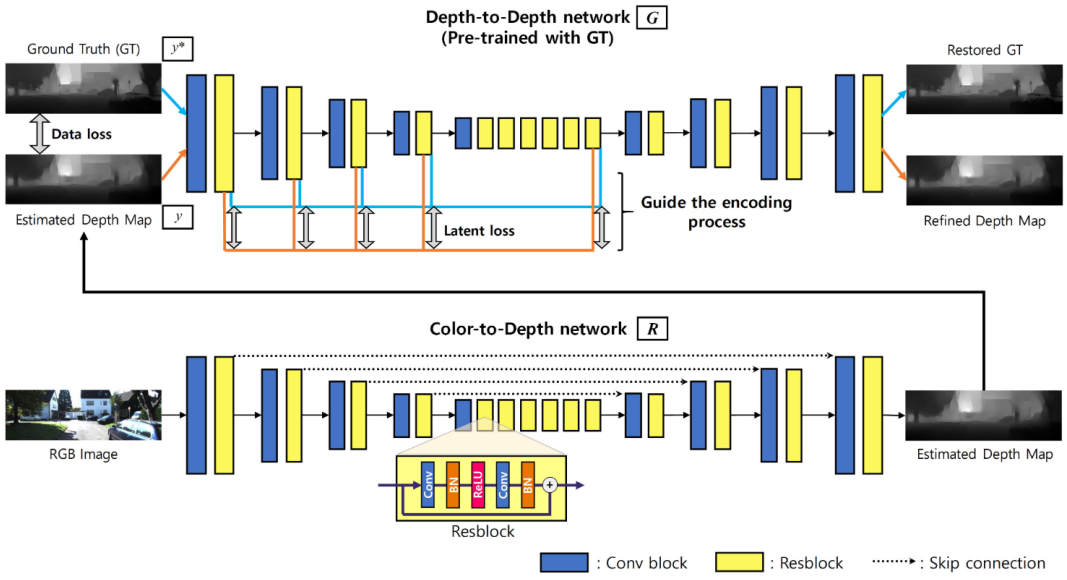


Figure 3.2: The proposed architecture for **GDN-Pytorch**. This shows the guided depth-to-depth model and the color-to-depth model. Figure from [36].

The proposed method uses a loss function comprised of four components when training the color-to-depth network, this is shown in equation 3.1. The generated depth map, y, is compared to the ground truth, y^*, to obtain the data loss L_d.

The latent loss, L_l is obtained using the features $G(y)$ and $G(y^*)$ extracted from the latent space of the already trained guided network when y and y^* are given as input. The gradient losses, L_{gd} and L_{gl} are obtained by applying a gradient operation to the outputs of each network. This loss function ensures the geometric structure encoded in the guided network helps to guide the training of the color-to-depth network. This helps to sharpen the depth boundaries in the generated depth maps.

$$L = L_d(y, y^*) + \alpha L_l(G(y), G(y^*)) + \beta L_{gd}(y, y^*) + \gamma L_{gl}(G(y), G(y^*)) \quad (3.1)$$

This model was trained and evaluated on the KITTI dataset [11]. It produced high quality depth maps and the paper's results can be seen in table 3.1. Throughout the rest of this paper the model proposed by [36] is referred to as ***GDN-Pytorch***.

DenseDepth Alhashim and Wonka (2018) [3] propose a simple encoder-decoder model that produces high quality depth maps with the help of transfer learning. The proposed model uses a DenseNet-169 [19] network pretrained on ImageNet [6] for the *encoder* and a series of up-sampling layers with skip connections for the *decoder*. Figure 3.3 shows the proposed architecture diagram. The model is trained with a data augmentation step to reduce over-fitting and to improve generalisation.

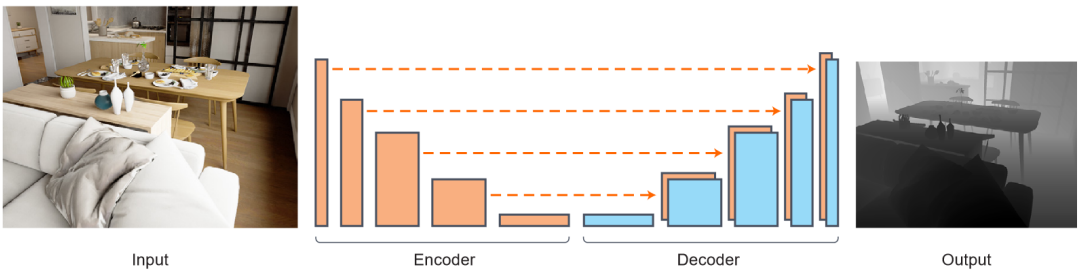


Figure 3.3: The proposed architecture for **DenseDepth**. This shows the simple encoder-decoder architecture with skip connections. Figure from [3].

The proposed loss function is comprised of three losses: L_{depth}, L_{grad} and L_{SSIM}. L_{depth} is the point-wise L1 loss between the ground truth depth map y and the

predicted depth map $\hat{y}$. L_{grad} is the L1 loss defined over the image gradient g of the depth image. L_{SSIM} uses the Structural Similarity (SSIM) [39] term which is commonly used for image reconstruction tasks. The full loss function can be seen in equation 3.2.

$$L(y, \hat{y}) = \lambda L_{depth}(y, \hat{y}) + L_{grad}(y, \hat{y}) + L_{SSIM}(y, \hat{y}) \tag{3.2}$$

The results on the KITTI dataset are shown in table 3.1. Throughout the rest of this paper the model proposed by [3] is referred to as ***DenseDepth***.

Monodepth2 Godard et al. (2019) [13] proposes a self-supervised network for monocular depth estimation. The model consists of two networks a depth network and a pose network. The depth network is an encoder-decoder architecture with skip connections. Similarly to **DenseDepth** [3], the encoder is pretrained using ImageNet [6], however, a ResNet18 [17] network is used. The pose estimation network is a modified ResNet18 network that accepts a pair of images and predicts a single 6-DoF relative pose. Figure 3.4 shows the proposed architecture diagram.

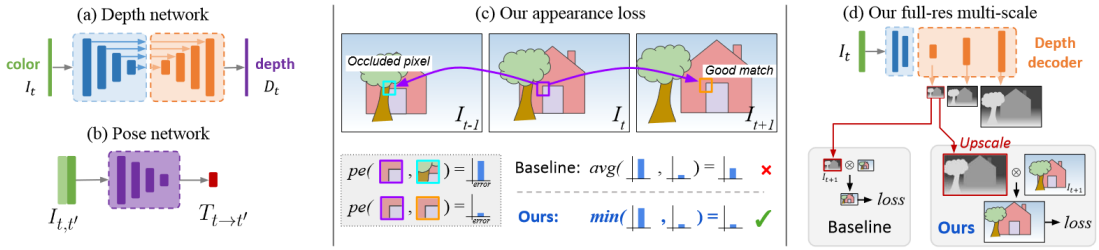


Figure 3.4: The proposed architecture for **Monodepth2**. Figure from [13]

With the proposed method the model can be trained using monocular images, stereo images or a combination of both. For self-supervised monocular training, consecutive temporal frames from a monocular video are used as the training signal. The pose network estimates the camera pose between frames, which is used to constrain the depth estimation at training time. The problem is formulated as the minimisation of a photometric reprojection error. The depth map is predicted that minimises this photometric reprojection error. Data augmentations are performed before training to improve generalisation.

The paper proposes improvements to the loss calculations used by similar self-supervised approaches. The authors use a per-pixel *minimum* reprojection loss, L_p, instead of the *average* reprojection loss. This ensures pixels that appear in the target image but not the source (due to egomotion or occlusions) do not negatively affect the predicted depth map. The authors propose a method of filtering out pixels that do not move between frames (due to objects moving at the same velocity as the camera or when the camera is stationary). This mask, μ, is applied to L_p. Edge-aware smoothness L_s is utilized to discourage shrinking of the estimated depth. This leads to the final loss function seen in equation 3.3.

$$L = \mu L_p + \lambda L_s \tag{3.3}$$

This method produces results similar to that of fully-supervised models on the KITTI dataset. These can be seen in table 3.1 below. Throughout the rest of this paper the model proposed by [13] is referred to as ***Monodepth2***.

Measure	Monodepth2 [13]	DenseDepth [3]	GDN-Pytorch [36]
$\delta < 1.25 \uparrow$	0.876	0.886	**0.893**
$\delta < 1.25^2 \uparrow$	0.958	0.965	**0.976**
$\delta < 1.25^3 \uparrow$	0.980	0.986	**0.992**
Abs Rel $\downarrow$	0.106	**0.093**	0.108
Sq Rel $\downarrow$	0.806	0.589	**0.278**
RMSE $\downarrow$	4.630	4.170	**2.545**
RMSE log $\downarrow$	0.193	0.171	**0.160**

Table 3.1: A comparison of the results reported by [36], [3] and [13] when tested on the KITTI dataset. The best results for each measure is highlighted in **bold**.

3.2.2 Depth from defocused images

Defocus-Net Unlike the aforementioned monocular depth estimation models, Maximov et al. (2020) [27] uses defocus cues to predict depth from a single image. The proposed method utilises two networks to predict the depth of a scene given multiple images taken at different focus distances. This combination of images with varying focal distances is known as a focal stack. The proposed method takes

a focal stack and first predicts a defocus map for the scene using a "DefocusNet" network. This defocus map is then used as the input to the "DepthNet" network which predicts the depth map. Figure 3.5 shows the proposed model architecture diagram.

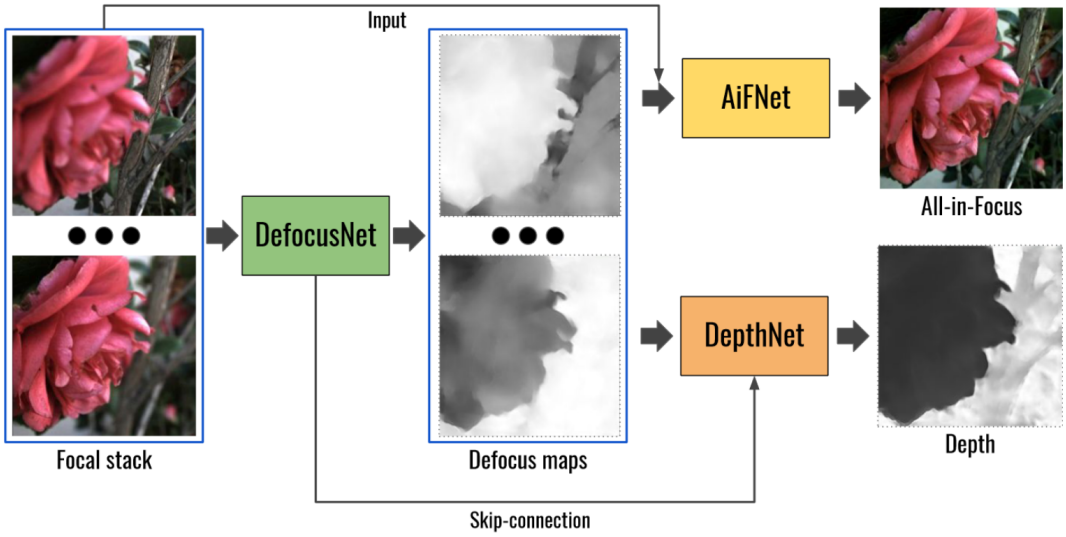


Figure 3.5: The proposed architecture for **Defocus-Net**. Figure from [27]

A defocus map is a measure of the amount of defocus per pixel. Examples can be seen in figure 3.5 as the output of the "DefocusNet". In order to compute this map, a measure for sharpness must be established. To do this the Circle of Confusion (CoC) measure is used, which measures the diameter of the focal point of a camera and is calculated as shown in equation 3.4, where f is the focal length, S_1 is the focus distance, S_2 is the distance to the object and N is the f-number. Figure 3.6 shows an illustration of the lens system on the left and on the right shows the evolution of CoC values as the distance to the object (S_2) increases. Each coloured line of the CoC plot represents a different focus distance. This plot shows the relationship between CoC value and object depth which the "DepthNet" exploits when predicting the depth map. The figure shows that the CoC and depth relationship is only an effective measure in a short range as after a certain depth the CoC values do not change.

$$c = \frac{|S_2 - S_1|}{S_2} \frac{f^2}{N(S_1 - f)} \tag{3.4}$$

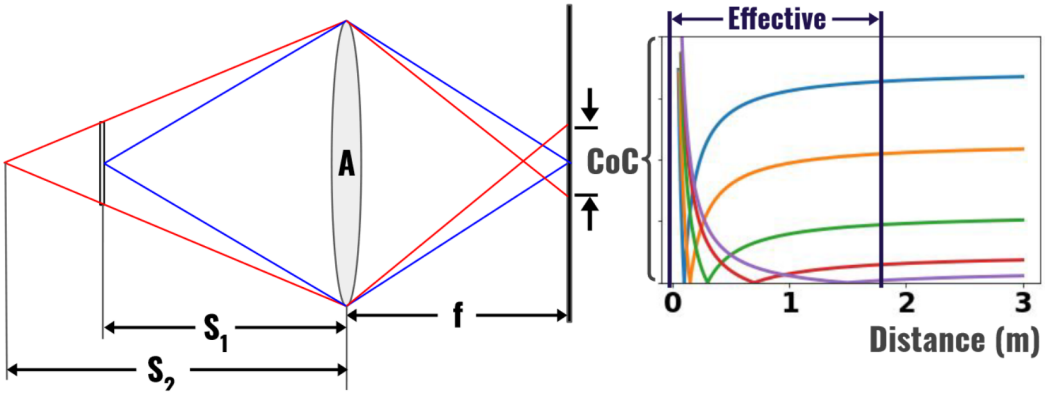


Figure 3.6: The left side of this figure illustrates the lens system. The right side shows the Circle of Confusion plot. Each coloured line represents a different focus distance (S_1). Figure from [27].

Maximov et al. (2020) create defocus maps by calculating clipped and normalised CoC values for every pixel in an image, giving a range between 0 (sharp) and 1 (blurry). This map is generated by the "DefocusNet" for every image in the focal stack. Figure 3.7 shows the architecture diagram for the "DefocusNet" network. The network is an encoder-decoder network with skip connections and layer-wise global pooling. The pooling allows the encoders to have local and global features for each step of the convolution. The output is used as the input to the "DepthNet" network. Both networks are encoder-decoder networks with skip connections between the encoder of the "DefocusNet" and the decoder of the "DepthNet". They are trained together end-to-end with the ability to use variable sized focal stacks. They show that the best results occur when training on the full focal stack of 5 images at different focal lengths.

The paper shows promising results on their own synthetic dataset and when testing on the **NYU Depth Dataset v2** [29] after artificial blurring. Throughout the rest of this paper the model proposed by [27] is referred to as ***Defocus-Net***.

These models were chosen for this study due to their varied architectures and reported results. These models are the state-of-the-art in monocular depth estima-

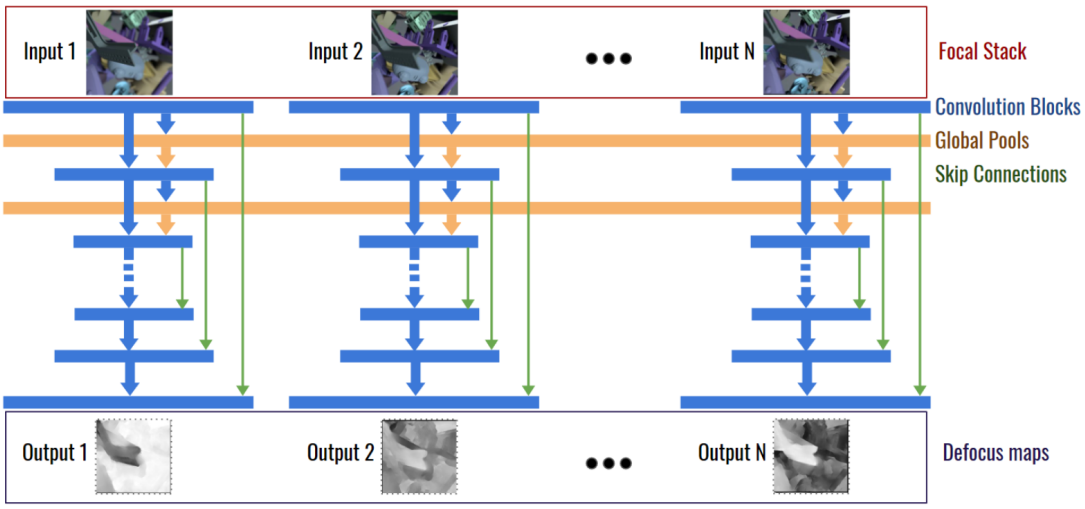


Figure 3.7: The proposed architecture of the defocus network contained within the ***Defocus-Net*** model. The predicted defocus maps are used as input to the depth network. Figure from [27].

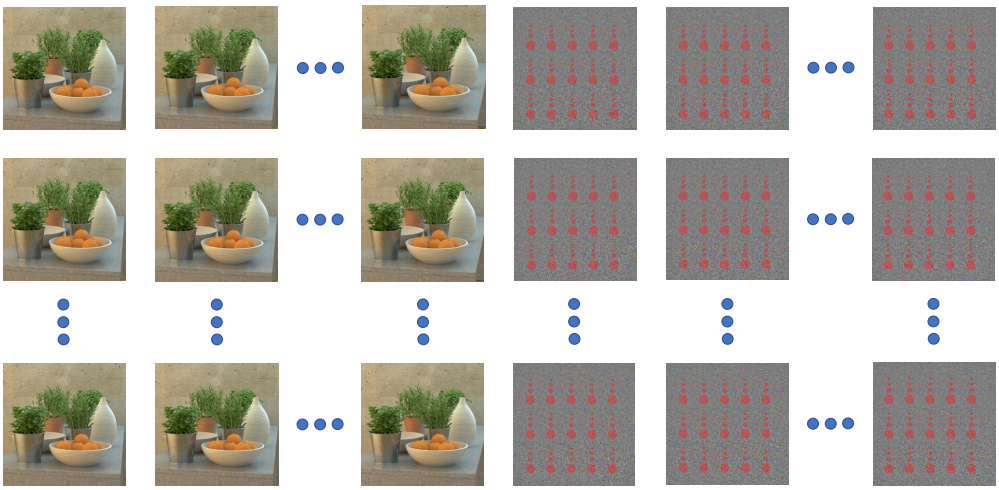
tion and cover a wide range of deep learning techniques in their implementations. The chosen models cover techniques such as transfer learning and guided networks as well as covering all-in-focus and depth from defocus techniques. They all perform comparatively despite the difference in architecture (as shown in table 3.1). For this reason we believe the chosen models are representative of the state of the art and can provide insights into the possible benefits of the chosen architectures.

3.3 HCI dataset

We selected the HCI dataset provided by [18] for this study. This paper provides the light field dataset along with an evaluation methodology for depth estimation. This dataset is split into photo-realistic and stratified scenes (an example of each can be seen in figure 3.8), each scene having an angular resolution of 9 x 9 and a spatial resolution of 512 x 512. The full benchmark dataset can be seen in the appendix in section A.2. The benchmark is designed to test the performance of the depth estimation across several lifelike scenes as well as specific aspects of depth estimation. The stratified scenes are not designed to be realistic, instead they aim

to isolate particular challenges faced by many depth estimation techniques.

- **Backgammon** - thin gaps and occlusions.
- **Pyramids** - slanted, convex and concave surfaces.
- **Dots** - noise and small objects.
- **Stripes** - texture and contrast at occlusions.



(a) Herbs scene (photo-realistic) (b) Dots scene (stratified)

Figure 3.8: Example light fields provided as part of the HCI benchmark dataset. (a) Shows an example of a photo-realistic scene. (b) Shows an example of a stratified scene.

3.4 Evaluation

In order to utilize the code provided by [18] to evaluate depth estimation performance, the predicted depth maps first had to be converted to disparity maps. As discussed in section 2.1.1, there is a relationship between depth and disparity. We can use the disparity of an object between images to estimate the depth of that object, given certain camera parameters. Conversely, given the camera parameters and the depth we can calculate the disparity (equation 3.5). The light fields

provided in the HCI dataset contain the required camera parameters to convert the predicted depth maps to disparity maps.

$$d = \frac{fb}{z} \tag{3.5}$$

After converting the predicted depth maps to disparity maps, the evaluation code could be used. The results were taken and compared. The evaluation code computes many metrics and visualisations to show the performance of each algorithm. The authors of [18] and [22] created a website [32] to display the results of various algorithms, this is used to compare the monocular depth estimation techniques to the light field techniques.

The evaluation computes three general metrics (these are outlined below) along with scene specific and feature specific metrics such as Bumpiness. This study is limited to the general metrics BadPix and MSE100 as we are comparing multiple models across multiple datasets and so are not interested in the specific characteristics and performances of each model across each scene, rather their performances across multiple datasets.

- **BadPix** - A function that gives the percentage of pixels at the given mask with an absolute difference between ground truth and estimated disparity map that is greater than a given threshold (the code calculates this at 0.01, 0.03 and 0.07).
- **MSE100** - A function that gives the mean squared error over all pixels at the given mask, multiplied by 100.
- **Q25** - A function that gives the maximum absolute error of the best 25% of pixels, multiplied by 100.

In this study we compare and contrast the performance of each model using these general metrics. Mean squared error multiplied by 100 (MSE100) is used as the primary metric in this study as it is a commonly used and recognised metric that provides a good general performance measure.

3.5 Results

In this section we discuss the quantitative and qualitative results found during our study. First we compare each model across all scenes. Using this comparison we select a single model to compare results across all-in-focus, front focus and back focus images in more detail. Additional results are shown for each model across all scenes and all datasets in the appendix in section A.3.

3.5.1 All-in-focus generalisation

Table 3.2 below shows the MSE100 for every model across every scene in the all-in-focus dataset. These results show the models generally performed poorly across all scences. For example, in the photo-realistic cotton scene DenseDepth has a MSE100 of 41.52. This error is roughly four times lower than the next best model, Defocus-Net, with an error of 163.26. Despite its relatively high performance in comparison to the other models in this study, the results shown by DenseDepth across all scenes fall short of the purported results by Alhashim and Wonka (2018) [3].

Scene	DenseDepth	GDN-Pytorch	defocus-net	monodepth2	*AttMLFNett*
backgammon	**61.27**	65.64	188.57	121.83	*3.86*
boxes	**64.72**	152.39	271.83	326.55	*3.84*
cotton	**41.52**	201.26	163.26	329.89	*0.06*
dino	**60.33**	259.07	275.53	233.84	0.05
pyramids	96.11	127	**26.24**	59.62	*0.003*
sideboard	109.14	436.17	**67.04**	169.71	*0.4*
stripes	15.39	18.9	21.91	**14.75**	*0.81*

Table 3.2: A comparison of the mean squared error over all pixels multiplied by 100, on the all-in-focus benchmark set. The lower the error the better. The best result for each scene is marked in **bold**. Also shown are the results for a light field depth estimation algorithm.

This holds true for every model. The results gathered in this study fall short of the performances reported by each model's respective paper. Table 3.3 below shows the BadPix(0.07) percentage for every model and every scene. This metric

gives us a percentage of pixels that are "bad" (i.e. a pixel where the absolute difference between its value and the ground truth values is more than 0.07). This table shows that even the best result has more than 50% bad pixels, meaning less than half of the pixels are within 0.07 of the desired ground truth value. Removing this one outlier we can see no other result has less than 80% bad pixels.

Scene	DenseDepth	GDN-Pytorch	defocus-net	monodepth2	*AttMLFNett*
backgammon	**81.72**	87.66	88.37	88.08	*3.23*
boxes	**94.96**	98.04	99.25	98.14	*11.138*
cotton	96.31	97.52	99.13	**95.24**	*0.195*
dino	**94.17**	99.57	99.95	97.14	*0.44*
pyramids	**93.89**	96.38	96.14	95.89	*0.174*
sideboard	96.92	99.63	**94.34**	96.67	*2.691*
stripes	90.56	83.41	**55.11**	90.31	*2.932*

Table 3.3: A comparison of the BadPix(0.07), on the all-in-focus benchmark set. The lower the percentage the better. The best result for each scene is marked in **bold**. Also shown are the results for a light field depth estimation algorithm.

Furthermore, looking to the qualitative results shown in figure 3.9 and the full results shown in the appendix in figure A.4, we show the poor performance of each model on the all-in-focus dataset. Visually DenseDepth is the most accurate model, however, it too struggles with the stratified scenes.

We argue that the quantitative and qualitative results discussed above indicate poor generalisation abilities for every model. Additionally, we argue the gap in performance seen between this study and the respective papers can be attributed to overfitting to the training dataset.

3.5.2 Generalisation across all-in-focus and defocus

From tables 3.2 and 3.3, DenseDepth performs significantly better than the other models on the photo-realistic scenes, however, it is outperformed in most of the stratified scenes. The qualitative results shown in figure 3.9 reinforce this, as we can see the DenseDepth disparity maps are visually more similar to the ground truth. The full qualitative comparison using all scenes is shown in the appendix

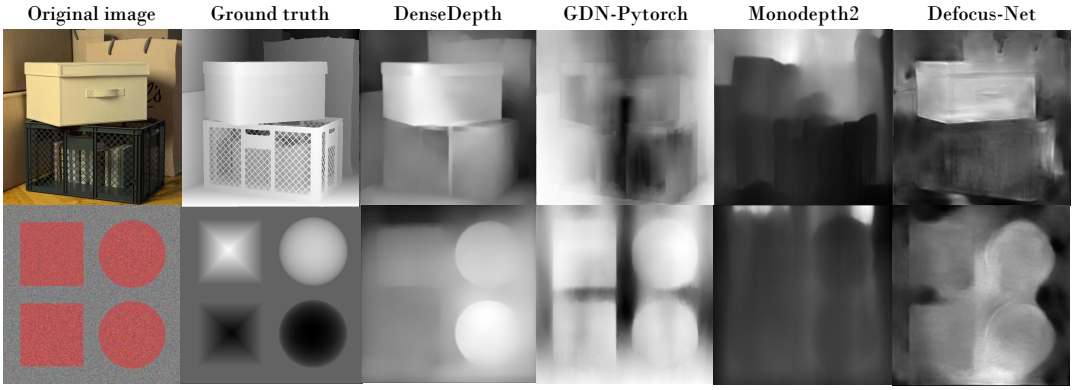


Figure 3.9: Qualitative comparison of the disparity maps computed for every model across the boxes scene (photo-realistic) and the pyramids scene (stratified). This comparison is for the all-in-focus dataset.

in section A.3.1. In the full comparison we can see DenseDepth continues to outperform the other models on all the photo-realistic scenes.

When comparing the results of each model across all three datasets (table 3.4 shows the best model for each scene and each dataset) we found DenseDepth generally outperformed all models including the depth from defocus model, Defocus-Net. This is surprising as intuitively we would reason that Defocus-Net should outperform the other models on the two defocused datasets. For this reason, we focus on comparing DenseDepth's performance across all three datasets below.

Scene	All in focus	Front focus	Back focus
backgammon	DD	DD	DD
boxes	DD	*DN*	*DN*
cotton	DD	DD	DD
dino	DD	DD	DD
pyramids	*DN*	DD	**MD**
sideboard	*DN*	*DN*	*DN*
stripes	**MD**	DD	**MD**

Table 3.4: A table showing the best performing model across the all-in-focus, front focused and back focused benchmark sets using the mean squared error over all pixels multiplied by 100. DD = DenseDepth, *DN* = Defocus-Net and **MD** = Monodepth2.

Table 3.5 below shows the MSE100 for DenseDepth across the all-in-focus, front focus and back focused datasets. This table shows, rather counter-intuitively, that DenseDepth performs better in the defocused images in all but one scene. We could argue that despite being trained and tested on all-in-focus images, the results show DenseDepth has an inherent understanding of depth from defocus cues. However, a closer look at the qualitative results shown in figure 3.10, indicates unusual behaviour in the defocused images. The white vase on the right side of the image, for example, disappears in the disparity map of the front focused image while it is clearly defined in the all-in-focus and back focused images. Similarly, the white bowl on the left side of the image is not seen in the disparity map of the front focused image. In both cases the object in question is blurred due to falling out of the focal plane. As expected DenseDepth is unable to account for this defocus blur. This contradicts the argument that DenseDepth has some inherent ability to perceive depth from defocus clues.

Scene	All in focus	Front focus	Back focus
backgammon	61.27	36.01	**31.42**
boxes	64.72	**55.88**	61.22
cotton	**41.52**	44.34	55.97
dino	60.33	**37.76**	55.9
pyramids	96.11	104.49	**70.17**
sideboard	109.14	**86.59**	176.38
stripes	15.39	16.48	**15.21**

Table 3.5: A comparison of DenseDepth across the all-in-focus, front focused and back focused benchmark sets using the mean squared error over all pixels multiplied by 100. The lower the error the better. The best result for each scene is marked in **bold**.

We argue that the poor performance of each model and the inability to predict the performance across all-in-focus and defocused images prevents us from asserting that a model has an inherent ability to understand depth from defocus clues. This is true even for Defocus-Net, which was trained on images with a defocus blur. We argue that despite the apparent ability of these models to generalise across all three datasets, we are unable to assert this to be true due to the unpredictability and overall poor results of each model. Furthermore, we argue that the apparent ability of these models to perceive depth from defocus despite being trained solely

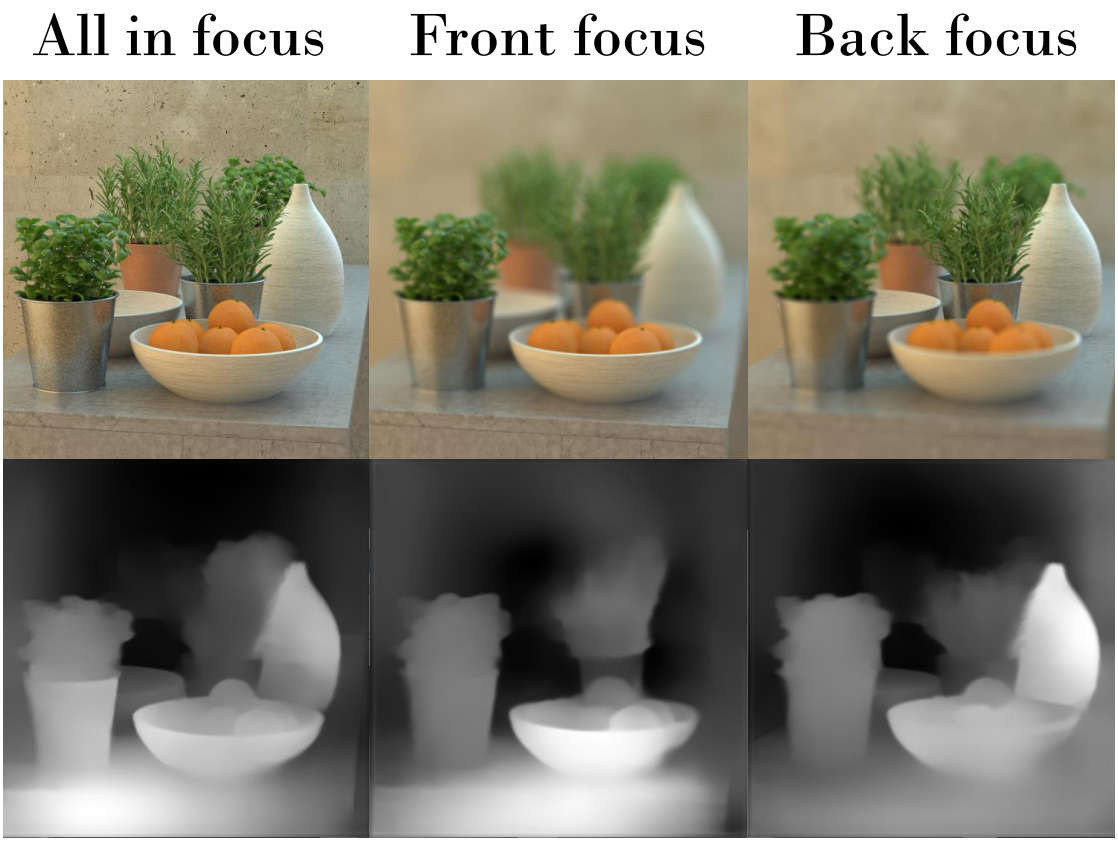


Figure 3.10: Shows the disparity maps converted from the depth maps produced by DenseDepth. This shows the comparison of the results across all-in-focus images and refocused images

on all-in-focus images, highlight a lack of knowledge about the features extracted and utilized by these deep learning techniques.

3.5.3 Comparison to light field models

Looking back to table 3.2 we can see the state of the art light field depth estimation model, AttMLFNett, has a MSE100 of 0.06 on the cotton scene. DenseDepth has an MSE100 of 41.52 which is orders of magnitude worse than AttMLFNett. The lowest ranked algorithm shown on [32] (at the time of writing this is MVCMv0) has a MSE100 of 5.16 for the cotton scene. DenseDepth has a MSE100 roughly

8 times larger than the lowest ranked light field algorithm tracked using the HCI benchmark for the cotton scene. This large gap in performance is seen across every monocular depth model. It stands to reason that light field depth estimation techniques will outperform monocular depth estimation techniques due to large volume of disparity information inherently stored in a light field. This comparison highlights the large gap that remains between monocular methods and light field methods in terms of depth estimation performance, despite the recent advancements in the field.

Chapter 4

Conclusions and Future Work

4.1 Conclusions

The goal of this project was to evaluate the state-of-the-art monocular depth estimation techniques and study their ability to generalise to an unseen dataset. Furthermore, we studied the ability of these models to generalise across all-in-focus images and images with a defocus blur (a blur that appears outside of the focal plane of the image).

First we carried out a review of the existing state-of-the-art in monocular depth estimation (section 2.1). We selected and adapted four models to use in this study (chapter 3). We used the HCI light field dataset to generate three distinct benchmark datasets: all-in-focus, front focus and back focus. These were generated using a shift-and-sum algorithm on the provided light fields. The models were used to predict a depth map for every scene and every dataset. To utilize the evaluation methodologies provided with the HCI dataset, the predicted depth maps had to be converted to disparity maps. These disparity maps were evaluated and their results compared (section 3.5).

We conclude that the quantitative and qualitative results shown in our study highlight the inability of monocular depth estimation models to generalise well to an unseen dataset. Furthermore, we argue the unpredictability of results across the all-in-focus and defocused datasets for each model shows a lack of knowledge about the features extracted and used by these deep learning models. The uses of

monocular depth estimation mentioned in section 1.1 such as augmented reality applications, require the depth estimation models to perform well in a broad range of scenes and camera settings. Our study shows there are still improvements to be made in this field to provide a model that can generalise well to the varied settings such an application would use it in.

4.2 Limitations

The following list outlines some of the limitations of our study.

- While the HCI benchmark dataset is designed to effectively test depth and disparity estimation models with carefully constructed scenes, the overall benchmark dataset is small. Perhaps a larger dataset could provide a more accurate measurement across a wider range of scenery.
- Defocus-Net [27] provided their code and the dataset used in their research, however, no pre-trained model was provided. The provided training scripts needed to be modified before the model could be successfully trained. For these reasons we were unable to replicate the results shown in their paper.
- We struggled to find state-of-the-art depth from defocus models that provided code or pre-trained models. This led us to only include a single model from this category. Ideally more of these models would be included to provide a better comparison.

4.3 Future work

Improving upon the limitations of this study could prove beneficial in understanding the general performance of monocular depth estimation techniques. The depth maps generated by the state-of-the-art light field models are very accurate and so could be used as the ground truth. In this way we could utilize a larger light field dataset that does not provide ground truth depth maps. AttMLFNett [5], for example, could be used to generate the ground truth depth maps for the (New)

Stanford Light Field Archive [23]. This would allow the study to integrate more light field datasets to improve the accuracy of results.

Using light fields to train a monocular depth estimation model could prove an interesting avenue of research. The light fields could be used as training input to a model that has a light field refocusing "layer". This layer would create images with varying defocus blurs in an attempt to train the model to understand defocus clues alongside all-in-focus clues. Following from the discussion above, we could use light field datasets that do not have ground truth data if we include a light field depth estimation model. This would allow for a more diverse training set and possibly lead to better generalisation.

A new model architecture could be used that combines recent advancements in deep learning. The use of an attention network [38] could improve results as seen in Chen et al. (2021) [5]. Similarly, transfer learning could be used to improve results as seen in DenseDepth [3]. More recent advancements such as transformer networks [10] could also be used. Researching the benefits of these techniques in monocular depth estimation could lead to overall improvements in accuracy as well as an improved generalisation ability.

Combining these potential architectural improvements with the use of light fields in the training step is an interesting area of research left for future work.

Bibliography

[1] Edward H. Adelson and James R. Bergen. *The Plenoptic Function and the Elements of Early Vision*, pages 3–20. Computation Models of Visual Processing, M. Landy and J.A. Movshon. Vision and Modeling Group, Media Laboratory, Massachusetts Institute of Technology, Cambridge, Mass., 1991.

[2] Martin Alain, Weston Aenchbacher, and Aljosa Smolic. Interactive light field tilt-shift refocus with generalized shift-and-sum. *CoRR*, abs/1910.04699, 2019.

[3] Ibraheem Alhashim and Peter Wonka. High quality monocular depth estimation via transfer learning. *arXiv e-prints*, abs/1812.11941, 2018.

[4] Robert C. Bolles, H. Harlyn Baker, and David H. Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, 1(1):7–55, 1987.

[5] Jiaxin Chen, Shuo Zhang, and Youfang Lin. Attention-based multi-level fusion network for light field depth estimation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(2):1009–1017, May 2021.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[7] Helisa Dhamo, Keisuke Tateno, Iro Laina, Nassir Navab, and Federico Tombari. Peeking behind objects: Layered depth prediction from a single image. *Pattern Recognition Letters*, 125:333–340, July 2019.

[8] Sotirios Ch. Diamantas and Richard M. Crowder. Localisation and mapping using a laser range finder: A goal-seeking approach. In *2009 Fifth International Conference on Autonomic and Autonomous Systems*, pages 270–276, 2009.

[9] R. Domínguez, E. Onieva, J. Alonso, J. Villagra, and C. González. Lidar based perception solution for autonomous vehicles. In *2011 11th International Conference on Intelligent Systems Design and Applications*, pages 790–795, 2011.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[12] Andrey Gershun. *The light field*. Unify Technical Press, 1936. Translated by P. Moon and G. Timoshenko in Journal of Mathematics and Physics, Vol. XVIII, MIT, 1939, pp. 51–151.

[13] Clement Godard, Oisin Mac Aodha, Michael Firman, and Gabriel Brostow. Digging into self-supervised monocular depth estimation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3827–3837, 2019.

[14] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, page 43–54, New York, NY, USA, 1996. Association for Computing Machinery.

[15] Xianfeng Gu, Steven J. Gortler, and Michael F. Cohen. Polyhedral geometry and the two-plane parameterization. In Julie Dorsey and Philipp Slusallek, editors, *Rendering Techniques '97*, pages 1–12, Vienna, 1997. Springer Vienna.

[16] T. Hassner and R. Basri. Example based 3d reconstruction from single 2d images. In *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*, pages 15–15, 2006.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[18] Katrin Honauer, Ole Johannsen, Daniel Kondermann, and Bastian Goldluecke. A dataset and evaluation methodology for depth estimation on 4d light fields. In *Asian Conference on Computer Vision*. Springer, 2016.

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017.

[20] 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, page 297–306, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[21] Hae-Gon Jeon, Jaesik Park, Gyeongmin Choe, Jinsun Park, Yunsu Bok, Yu-Wing Tai, and In So 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.

[22] Ole Johannsen, Katrin Honauer, Bastian Goldluecke, Anna Alperovich, Federica Battisti, Yunsu Bok, Michele Brizzi, Marco Carli, Gyeongmin Choe, Maximilian Diebold, Marcel Gutsche, Hae-Gon Jeon, In So Kweon, Alessandro Neri, Jaesik Park, Jinsun Park, Hendrik Schilling, Hao Sheng, Lipeng Si, Michael Strecke, Antonin Sulc, Yu-Wing Tai, Qing Wang, Ting-Chun Wang, Sven Wanner, Zhang Xiong, Jingyi Yu, Shuo Zhang, and Hao Zhu. A taxonomy and evaluation of dense light field depth estimation algorithms. In *Conference on Computer Vision and Pattern Recognition - LF4CV Workshop*, 2017.

[23] Stanford Graphics Laboratory. The (new) stanford light field archive. `http://lightfield.stanford.edu/lfs.html`, 2008. Accessed on: Jul 6, 2021.

[24] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, page 31–42, New York, NY, USA, 1996. Association for Computing Machinery.

[25] Yi Lin, Juha Hyyppä, and Anttoni Jaakkola. Mini-uav-borne lidar for fine-scale mapping. *IEEE Geoscience and Remote Sensing Letters*, 8(3):426–430, 2011.

[26] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 39(4), 2020.

[27] Maxim Maximov, Kevin Galim, and Laura Leal-Taixe. Focus on defocus: Bridging the synthetic to real domain gap for depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[28] Parry Hiram Moon and Domina Eberle Spencer. *The photic field*. MIT Press, 1981.

[29] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[30] Ren Ng, Marc Levoy, Mathieu Brédif, Gene Duval, Mark Horowitz, and Pat Hanrahan. Light Field Photography with a Hand-held Plenoptic Camera. Research Report CSTR 2005-02, Stanford university, April 2005.

[31] National Oceanic and Atmospheric Administration. What is lidar? `https://oceanservice.noaa.gov/facts/lidar.html`, October 2012. Accessed on: Apr 30, 2021.

[32] University of Konstanz and HCI at Heidelberg University. 4d light field benchmark. `https://lightfield-analysis.uni-konstanz.de`, October 2016. Accessed on: Jun 23, 2021.

[33] Ning Qian. Binocular disparity and the perception of depth. *Neuron*, 18(3):359–368, 1997.

[34] N.R. Raajan, M. Ramkumar, B. Monisha, C. Jaiseeli, and S. Prasanna venkatesan. Disparity estimation from stereo images. *Procedia Engineering*, 38:462–472, 2012. INTERNATIONAL CONFERENCE ON MODELLING OPTIMIZATION AND COMPUTING.

[35] Ashutosh Saxena, Sung Chung, and Andrew Ng. Learning depth from single monocular images. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18. MIT Press, 2006.

[36] Minsoo Song and Wonjun Kim. Depth estimation from a single image using guided deep network. *IEEE Access*, 7:142595–142606, 2019.

[37] Nurollah Tatar and Hossein Arefi. Stereo rectification of pushbroom satellite images by robustly estimating the fundamental matrix. *International Journal of Remote Sensing*, 40(23):8879–8898, 2019.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[39] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

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

[41] Feng Xue, Guirong Zhuo, Ziyuan Huang, Wufei Fu, Zhuoyue Wu, and Marcelo H. Ang. Toward hierarchical self-supervised monocular absolute depth estimation for autonomous driving applications. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2330–2337, 2020.

[42] ChaoQiang Zhao, QiYu Sun, ChongZhen Zhang, Yang Tang, and Feng Qian. Monocular depth estimation based on deep learning: An overview. *Science China Technological Sciences*, 63(9):1612–1627, September 2020.

Appendix A

A.1 Triangulation derivation

Depth can be obtained using the relationship between the disparity of an object between a stereo pair of images and the objects distance to the image plane. The relationship can be seen in figure 2.3. A represents the object we are capturing. $Left\ image$ and $Right\ image$ represent a co-planar stereo pair of images. The light from A passes through the pinholes B and D onto the image screens at E and H. The disparity d is equal to $EF + GH$. f is the focal length of the camera. b is the distance between the centre of each camera which is known as the baseline. Triangles ABC and BEF are similar and triangles ACD and DGH are similar. From this we can prove the depth z is equal to $fb \div d$. This is shown in equations A.1 and A.2.

$$\begin{aligned} d &= EF + GH \\ &= BF(\frac{EF}{BF}) + DG(\frac{GH}{DG}) \\ &= f(\frac{EF}{BF}) + f(\frac{GH}{DG}) && , f = BF = DG \\ &= f(\frac{EF}{BF} + \frac{GH}{DG}) \\ &= f(\frac{BC + CD}{AC}) && , \text{from similar triangles} \\ &= f(\frac{b}{z}) && , BC + CD = b \text{ and } AC = z \end{aligned} \tag{A.1}$$

Rearranging equation A.1 gives us a formula for our depth z,

$$z = \frac{fb}{d} \tag{A.2}$$

A.2 All in focus benchmark dataset

(a) Boxes

(b) Cotton

(c) Dino

(d) Sideboard

Figure A.1: The centre views of the "training" subset of the HCI benchmark dataset [18]. A set of photo-realistic scenes to train light field models.

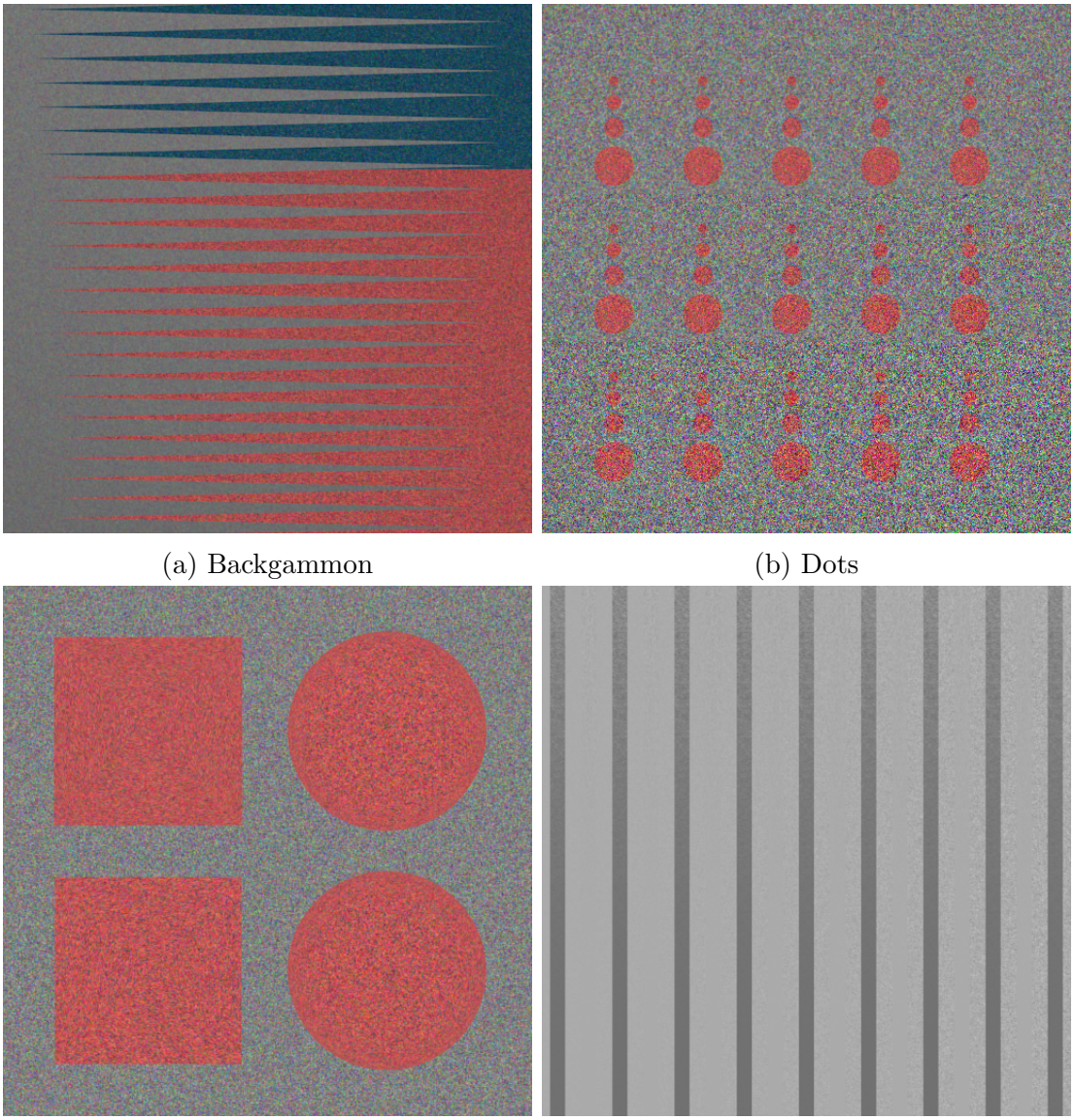
(a) Bedroom

(b) Bicycle

(c) Herbs

(d) Origami

Figure A.2: The centre views of the "test" subset of the HCI benchmark dataset [18]. A set of photo-realistic scenes to test light field models.



(a) Backgammon (b) Dots

(c) Pyramids (d) Stripes

Figure A.3: The centre views of the "stratified" subset of the HCI benchmark dataset [18]. This subset is designed to test specific aspects of depth estimation such as occlusions or slanted surfaces and so are not photo-realistic.

A.3 Additional results

A.3.1 Full qualitative results

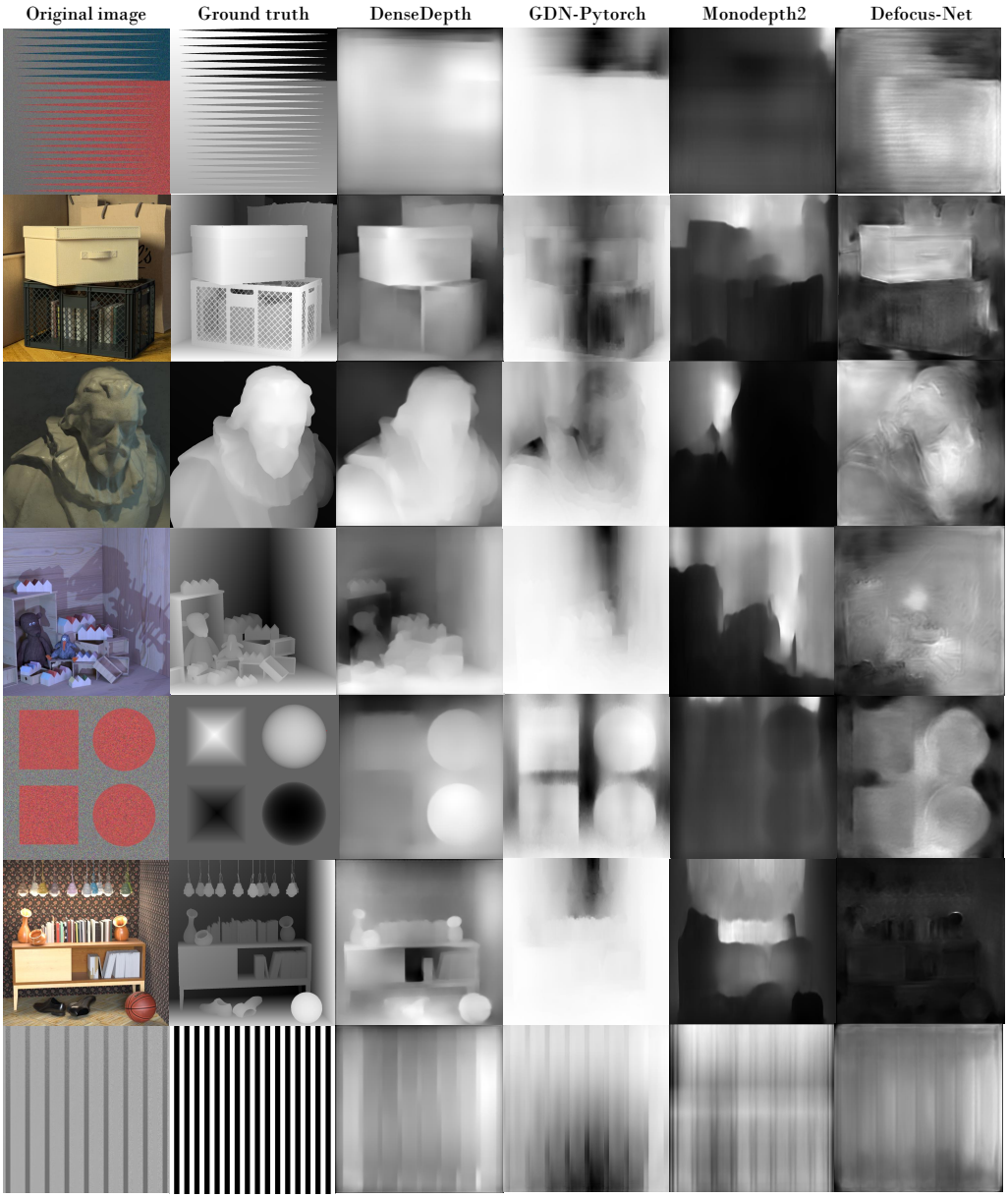


Figure A.4: Qualitative comparison of the disparity maps computed for every model and every scene. This comparison is for the all-in-focus dataset.

A.3.2 All-in-focus

Scene	DenseDepth	GDN-Pytorch	defocus-net	monodepth2	*AttMLFNett*
backgammon	**61.27**	65.64	188.57	121.83	*3.86*
boxes	**64.72**	152.39	271.83	326.55	*3.84*
cotton	**41.52**	201.26	163.26	329.89	*0.06*
dino	**60.33**	259.07	275.53	233.84	*0.05*
pyramids	96.11	127	**26.24**	59.62	*0.003*
sideboard	109.14	436.17	**67.04**	169.71	*0.4*
stripes	15.39	18.9	21.91	**14.75**	*0.81*

Table A.1: A comparison of the mean squared error over all pixels multiplied by 100, on the all-in-focus benchmark set. The lower the error the better. The best result for each scene is marked in **bold**. Also shown are the results for a light field depth estimation algorithm.

A.3.3 Front focus

Scene	DenseDepth	GDN-Pytorch	defocus-net	monodepth2	*AttMLFNett*
backgammon	**31.42**	74.59	99.38	146.72	*3.86*
boxes	61.22	164.06	**47.54**	316.31	*3.84*
cotton	**55.97**	223.24	189.21	334.98	*0.06*
dino	**55.90**	239.59	190.78	196.49	*0.05*
pyramids	**70.17**	151.78	73.67	73.91	*0.003*
sideboard	176.38	328.78	**87.02**	179.32	*0.4*
stripes	**15.21**	20.11	19.53	15.59	*0.81*

Table A.2: A comparison of the mean squared error over all pixels multiplied by 100, on the front focus benchmark set. The lower the error the better. The best result for each scene is marked in **bold**. Also shown are the results for a light field depth estimation algorithm.

A.3.4 Back focus

Scene	DenseDepth	GDN-Pytorch	defocus-net	monodepth2	*AttMLFNett*
backgammon	**36.01**	71.21	207.18	163.87	*3.86*
boxes	55.88	152.36	**54.12**	319.25	*3.84*
cotton	**44.34**	210.41	225.44	307.47	*0.06*
dino	**37.76**	203.09	203.21	211.64	*0.05*
pyramids	104.49	127.49	39.10	**33.56**	*0.003*
sideboard	86.59	471.61	**65.72**	168.06	*0.4*
stripes	16.48	20.68	19.31	**15.77**	*0.81*

Table A.3: A comparison of the mean squared error over all pixels multiplied by 100, on the back focus benchmark set. The lower the error the better. The best result for each scene is marked in **bold**. Also shown are the results for a light field depth estimation algorithm.