Low Complexity Networks and Edge Enhancement for Monocular Depth Estimation

A Thesis submitted

in partial fulfillment for the degree of

Doctor of Philosophy

by

Sandip Paul

Department of Avionics

Indian Institute of Space Science and Technology

Thiruvananthapuram, India

July 2024

Certificate

This is to certify that the Thesis titled ***Low Complexity Networks and Edge Enhancement for Monocular Depth Estimation*** submitted by **Sandip Paul**, to the Indian Institute of Space Science and Technology, Thiruvananthapuram, in partial fulfillment for the award of the degree of **Doctor of Philosophy** is a bonafide record of the original work carried out by him under my supervision. The contents of this Thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Deepak Mishra
(Research Supervisor)
Professor,
Department of Avionics,
IIST

Dr. M. Senthil Kumar
(Research Supervisor)
OUTS SCI
GroupDir. QUTG,
SAC, ISRO

Dr. N. Selvaganesan
Professor and Head,
Department of Avionics,
IIST

Place: IIST, Thiruvananthapuram
Date: July 2024

Declaration

I declare that this Thesis titled ***Low Complexity Networks and Edge Enhancement for Monocular Depth Estimation*** submitted in partial fulfillment for the award of the degree of **Doctor of Philosophy** is a record of the original work carried out by me under the supervision of **Dr. Deepak Mishra** and **Dr. M. Senthil Kumar** has not formed the basis for the award of any degree, diploma, associateship, fellowship, or other titles in this or any other Institution or University of higher learning. In keeping with the ethical practice of reporting scientific information, due acknowledgments have been made wherever the findings of others have been cited.

Place: IIST, Thiruvananthapuram **Sandip Paul**
Date: July 2024 SC16D007

This Thesis is dedicated to . . .

To my parents
Late Sushil Ranjan Paul and Anita Paul

Acknowledgements

This thesis is the result of mentoring, support, and encouragement of many associates at the Space Applications Centre (SAC) and the Indian Institute of Space Science and Technology (IIST). I take this opportunity to extend my sincere gratitude and appreciation to all those who have contributed to this research work. First and foremost, I thank my research supervisors **Dr. Deepak Mishra** and **Dr. M. Senthil Kumar** for their continuous inspiration, guidance, encouragement, and helpful advice. Their persistence and continuous motivation shaped my knowledge, skill, and thinking ability towards this research. I am grateful to **Dr. Deepak Mishra** for enrolling me as his student and help in selecting the coursework for my thesis, for academic support, and for defining the problem statement. Further, he made continuous efforts to update my knowledge with up-to-date literature and papers. All the papers published were the result of his excellent mentoring. I wholeheartedly thank **Dr. M. Senthil Kumar** for the continuous support and advice extended to me at all needed times.

I thank the members of my doctoral committee for their periodic assessments and valuable suggestions for my research: Dr. B.S. Manoj (Professor, IIST), Dr. N. Selvaganesan (Professor, IIST), Dr. Gorthi R.K.S.S. Manyam (Professor-IIT-Tirupati), Dr.Priyadarshnam (Professor-IIST), Dr. Sheeba Rani J. (Associate Professor, IIST) and Dr. Soma Biswas (Professor-IISC). I also express my thanks and wishes to all my colleagues at IIST for their positive moral support and company.

I am grateful to the Professors and Lab-in-charges at IIST for providing an excellent environment for academic learning and filling the knowledge gap. I am grateful to the Director (IIST), Dean (Academics), and Dean(R&D) for providing me with an opportunity to carry out research at IIST. I am incredibly thankful to the Director (SAC), for enabling me to pursue a part-time PhD. I also express my sincere gratitude to Deputy Director (SEDA), and Group Director (SEG) for their support.

A heartfelt thanks go out to my wife and daughter for their sacrifice and unconditional support throughout my research.

Sandip Paul

Abstract

Estimating depth from a 2D image is valuable for robotics, navigation, object recognition, medical diagnosis, and 3D measurements. Mobile platforms often have limitations in size, weight, and power. Cost-effective monocular depth estimation with a single camera and low computational requirements are suited for such applications. Most research on depth recovery lacks work on depth range, dynamic targets, and practical setups. The thesis investigates various depth recovery methods to understand such gaps.

The analysis of experiments using depth from focus (DOD) found that blur detectors can estimate sparse depth maps, but the depth estimate is only accurate for close-range targets due to the shallow depth of field. Additionally, shadows can create false depth and artifacts not previously reported in the literature. This method is not useful for featureless surfaces or moving targets.

A coded aperture using two spectral filters (DFCCA) was developed, resulting in a larger 32-pixel disparity map compared to the 14 pixels reported in the literature. However, the accuracy was again only high for near-targets. The analysis showed that performance is sensitive to spectral leakage, signal-to-noise ratio (SNR), and reduced image resolution. The multi-coded aperture (DMCA) relies on non-overlapping Point Spread Function (PSF) signatures. Here, the depth estimate was found to be accurate for surfaces with dense textures but not suitable for moving targets due to the requirement of two images.

Most depth-recovery methods estimate relative depths. A new calibration method has been developed to recover absolute depth from monocular images. This approach uses a unique blur target with ground truth. The method is not affected by contrast variations, magnification artifacts, or spectral sensitivity. The blur ranges up to a radius of 1.2σ, as demonstrated, but is limited by the inherent camera optic blur.

Modern Deep Neural Networks (DNNs) estimate depth from single images. The key components are network architecture and network training loss functions. Regression loss for robust training is less researched. Here, new loss functions based on edge and Structural Similarity (SSIM) functions were proposed. These improved the log_{10} error and the $\delta 1$ accuracy (85%) emphasizing the training robustness.

Available Deep Learning Networks are computationally intensive for use in mobile platforms. A low-complexity multi-scale network architecture (NDWTN) is designed.

NDWTN has wavelets, attention, dense convolution, residual convolution, batch norm, and efficient activation layers to estimate the full-depth map. This network outperforms previously known DWT-based and UNET++ models in all six performance metrics and provides the best RMSE score among present state-of-the-art models. NDWTN trains faster within 17 minutes per epoch and reaches an accuracy of > 92% for 10 epochs. These are suitable for conservative mobile systems.

Contents

List of Figures **xvii**

List of Tables **xxiii**

Abbreviations **xxv**

Nomenclature **xxvii**

1 Introduction **1**

1.1 Depth estimate from stereo vision 2

1.2 Depth estimate from single cameras 4

1.3 Challenges for Mobile Platforms 5

1.4 Motivation and Research Objective 6

1.5 Organization of the Thesis 7

2 Depth Estimation and Methods-A Survey **12**

2.1 Introduction 12

2.1.1 Camera 12

2.1.2 Modeling the camera 13

2.2 Market survey of depth cameras 15

2.2.1 Stereo vision 15

2.2.2 LiDAR 17

2.2.3 Structured light 18

2.3 Monocular Depth Estimation Methods 18

2.3.1 Depth from Motion 19

2.3.2 Classical Methods 19

2.3.3 Coded aperture methods 20

2.3.4 Blur Cues 21

2.3.5 Light field method . . . 21
2.4 Learning-based methods . . . 23
2.4.1 Convolutional neural networks (CNNs) . . . 24
2.5 Datasets . . . 25
2.5.1 Selection of Database . . . 26
2.5.2 Data splitting and augmentation . . . 26
2.6 Performance Analysis . . . 26
2.6.1 Quantitative Measures . . . 27
2.7 Applications . . . 29
2.7.1 AR / VR and Entertainment: . . . 29
2.7.2 Security: . . . 30
2.7.3 Robotics: . . . 30
2.7.4 Artificial Intelligence (AI): . . . 30
2.8 Challenges . . . 31
2.9 Summary . . . 32

3 Depth from Defocus **34**
3.1 Introduction . . . 34
3.2 Aim . . . 35
3.3 Implementation Method . . . 36
3.4 Experiments . . . 39
3.5 Results and Analysis . . . 40
3.6 Summary . . . 41

4 Coded Aperture for Depth Estimation **45**
4.1 AIM . . . 45
4.2 Color-coded aperture . . . 45
4.2.1 Implementation of DFCCA . . . 46
4.2.2 Experiments . . . 47
4.2.3 Results and Observations . . . 48
4.2.4 Discussions . . . 50
4.3 Multi-coded Aperture . . . 52
4.3.1 Implementation Method . . . 52
4.3.2 Experiments . . . 55
4.3.3 Results . . . 56
4.3.4 Observations . . . 57

4.4 Summary . . . 57

5 Calibration and Absolute Depth **59**
5.1 Introduction . . . 59
5.2 AIM . . . 61
5.3 Implementation Method . . . 61
5.4 Calibration . . . 63
5.4.1 Choice of PSF . . . 63
5.4.2 Target and Result . . . 65
5.4.3 Experiments . . . 65
5.4.4 Ground truth . . . 69
5.5 Verification . . . 70
5.6 Summary . . . 72

6 Loss Functions for Edge Enhancement **74**
6.1 Introduction . . . 74
6.2 AIM . . . 74
6.3 Network architecture . . . 75
6.3.1 Training a Network . . . 76
6.4 Loss function . . . 77
6.5 Formulation-Multi-Loss Function . . . 80
6.5.1 Optimization . . . 85
6.5.2 Regularization . . . 85
6.6 Ablation Studies . . . 85
6.7 Experiments and Results . . . 86
6.8 Results and Observation . . . 87
6.9 Summary . . . 90

7 Nested Wavelet-Net for Depth Estimation **93**
7.1 Introduction . . . 93
7.2 AIM . . . 94
7.3 Wavelets . . . 95
7.4 Deep network architecture . . . 97
7.4.1 Convolution Layer . . . 98
7.4.2 Pooling Layer . . . 98
7.4.3 Activation functions . . . 98

7.4.4 Regularization . . . 99
7.4.5 Proposed Network Architecture . . . 99
7.5 Loss Function . . . 106
7.6 Experiments . . . 107
7.6.1 Network Models . . . 108
7.6.2 Ablation Studies and Results . . . 109
7.7 Observations . . . 112
7.8 Summary . . . 117

8 Conclusions and Future work **118**
8.1 Research Contributions . . . 120
8.2 Future Research Scopes . . . 121

Bibliography **121**

List of Publications **139**

9 Appendix A: Camera parameters **140**
9.1 Computation methods . . . 140
9.2 Canon EOS 400D Camera . . . 142
9.2.1 Specifications of Camera . . . 142
9.3 CM-030 GE / CB-030 GE . . . 144
9.3.1 Main Features . . . 144
9.3.2 Specifications . . . 145

10 Appendix B: Depth Perception by Humans **147**
10.1 Human Eye . . . 147
10.2 Depth Perception modes . . . 148
10.2.1 Binocular cues . . . 148
10.2.2 Monocular cues . . . 150
10.2.3 Oculomotor cues . . . 153
10.2.4 Camera . . . 154

11 Appendix C: Datasets **156**
11.1 Datasets using cameras . . . 156
11.2 Synthetic and simulated datasets . . . 161
11.3 Database from the Web . . . 164

11.4 Tools for Database 166

List of Figures

1.1 Stereo method and epipolar geometry 3

2.1 Overview of blur estimation approach 14
2.2 Variation of blur diameter (COC) with distance (Equation 2.1). It is minimal at focus. 14
2.3 RealSense™ D400 series stereo depth camera. The module contains two IR cameras, an IR source, and an RGB camera. 16
2.4 Left, right images and the derived disparity image 16
2.5 Relationship between depth and disparity f_0, B, pix are fixed for a camera setup and are kept here as unity for simplicity. 16
2.6 LiDAR Time Of Flight depth camera. 17
2.7 Structured light depth camera. 18
2.8 Dual aperture: (a) Schematic of the optical system (b) Inset image shows the color shift of a ceiling lamp (c) Modified camera with dual color apertures 21
2.9 Spatial and angular information of light rays in 4D representation 22
2.10 Plenoptic camera: Simulated multi-view of microlens array. 23

3.1 Schematic of the optical system. Here, f_o is the focal length, d, d_f is the distance of the lens to the objects, and v, v_f is the distance of the lens from the images. 36
3.2 Plot of blur (COC) concerning object distance (Equation 3.1). Blur is at its minimum when a distant object is focused on the image plane (0.75 m). . . 38
3.3 Zoomed corner points of the checkerboard. The points are erroneously identified due to image blur 40
3.4 Far and near-focused images for input 41
3.5 (a) Estimated color-coded depth map obtained with defocus. Small depths are red, while large depths are blue. (b) 3D image 42

3.6 Outdoor far and near focused images for input 42
3.7 (a) Depth map: near objects are red-coded and far objects are blue; (b) 3D image: negative distances are away from the camera focus 42

4.1 (a) Schematic of the optical system (b) Dual color aperture pair (c) Inset-stereo image pair from ceiling lamp 46
4.2 Theoretical plot of (a) Defocus with object distance (equation 4.2) (b) object distance versus separation (equation4.3) 47
4.3 RGB image and histogram of RGB bands 49
4.4 Histogram of RGB bands of the image after G band subtraction from R and B 49
4.5 (a) Disparity map of B & R band (b) 3D depth map of B & R band 50
4.6 (a) Images of R and B band after G band subtraction (b) Disparity map of R & B band after G band subtraction (c) 3D depth map of R & G band . . . 51
4.7 Image prior: A 54
4.8 Image with dense features 55
4.9 Synthetic depthmap 55
4.10 Aperture pairs for depth estimation 56
4.11 Defocused image pair with different apertures 56
4.12 All in focus images after deconvolution 57
4.13 Estimated depth map from different aperture pairs 57

5.1 Proposed method of blur estimation using a special target 64
5.2 PSF kernels 64
5.3 The proposed synthetic target of 400x3300 pixels contains 10 blocks of 400x330 pixels. Each block is a checkerboard pattern blurred incrementally with a PSF filter having scales from 1 to 10. The figure shows only 2 of the 10 blocks for clarity. 66
5.4 The synthetic blur map with a PSF scale of 1 to 10 obtained using equation 5.15. A block with the same PSF creates steps in the map. 66
5.5 The edge thickness from higher PSF scales changes the pattern for scales of 4 and above. 66
5.6 Blur range finalization. 67
5.7 Results with pill box PSF. 67
5.8 Range limits with Gaussian PSF. 68
5.9 Results with Gaussian PSF. 68

5.10 Ground truth, checkerboard pattern from multiple distances stitched together. Blur is higher for large distances. 69
5.11 Variation of k with depth 69
5.12 Three checkerboard pattern images, A, B, and C, taken with various slants to get continuous depth 71
5.13 Depth Map of (a) an image with a slanted pattern and (b) a synthetic target. The color code brown indicates the nearest depth. 71
5.14 Graph of distance w.r.t. σ for the 4 verification images. 72

6.1 Network architecture [1]. 76
6.2 Plot of Loss Functions 79
6.3 Plots of MAE, MSE, and BerHu functions and characteristics. 81
6.4 A visual comparison of predicted depths with proposed edge loss functions (a) SSIM', (b) BerHu, (c) SSIM' and BerHu, (d) Sobel, (e) Laplacian, (f) LOG, (g) DOG 89
6.5 (a) NYU Dataset original image (b) Ground truth (c) Estimated depth map from [1] 89
6.6 Predicted depth map with proposed loss functions (a) SSIM sharpened, (b) BerHu, (c) SSIM sharpened, and BerHu 90
6.7 Performance of depth estimation (a) false color ground truth, (b) Laplacian loss function, (c) Sobel loss function 90
6.8 Edge loss functions and predicted depth-map (a) false color ground truth, (b) LOG, (c) DOG 90
6.9 (a) NYU Dataset Original Image (b) Predicted depth map (c) Error map of ground truth and predicted 91

7.1 DWT decomposition and low-resolution coefficient maps. Here, the scale is 2, however, the downsampling operation requires a scale of 1. 97
7.2 CNN- basic elements. 98
7.3 Proposed Network Architecture (NDWTN) 101
7.4 Structure of down-sampling block. 101
7.5 Down-sampling block details: The stack of convolution operators and the sequence of Batch-Norm and activation layers are customized. 102
7.6 Residual convolution block. The stack of convolution operators and the sequence of Batch–Norm and activation layers are customized. 103

7.7 The upsampling block provides IWT and convolution operations. Information from the skip path 'a' is also concatenated. 103
7.8 Up–sampling block details: The stack of convolution operators and the sequence of Batch–Norm and activation layers are customized. The convolution stack can be replaced with a residual block. 104
7.9 Output block with Sigmoid activation layer 104
7.10 Skip layer with attention. This layer takes two inputs from encoder blocks of different scales, 'g' from the higher scale or input to the encoder and 'x' from the lower scale or output of the encoder, and feeds the decoder block with attention vectors 'a'. 104
7.11 Proposed Network with Attention Gates (NADWT) 105
7.12 Depth map prediction after training, a visual comparison. **A**: input image, **B**: ground truth, **C**:UNET++, **1**: NDWT (3C, 3R, 3Bs) + Bs, **2**: NADWT (3C, 3LR, 1Bs), **3**: NADWT (3C, 3LR, 1Bs) + Bs, **4**: NADWT (3C, 3Bs, 3R) + Bs, **5**: NADWT (3C, 3R, 3Bs) + Bs, **6**: NRDWT (3C, 3R, 3Bs) + Bs, **7**: NRDWT (3C, 3Bs, 3R) + Bs, **8**: NARDWT (3C, 3LR, 3Bs) + Bs, **9**: NARDWT (3C, 3R, 3Bs) + Bs, **10**: NARDWT (3C, 3Bs, 3LR) + Bs, **11**: NARDWT (3C, 3Bs, 3LR), **12**: NARDWT (3C, 3LR), **13**: NARDWT (4C, 4Bs, 4LR) + 1Bs). 111
7.13 Model loss performance. The best is DWT + Attention, followed by Residual + Attention architecture. 112
7.14 Model Evaluation Accuracy Performance 112
7.15 Model Training Loss Performance 113
7.16 Model Training Accuracy Performance. 113
7.17 Model Validation Loss Performance 114
7.18 Model validation accuracy Performance 115
7.19 Training with the KITTI V2 dataset. Results show the visual performance of DWT (UNet with DWT), ADWT (UNet with DWT and attention), and NDWT. 116

10.1 Basic components of an eye for human vision. 148
10.2 CIE spectral luminous efficiency function (a) photopic vision $V(\lambda)$ and (b) scotopic vision $V'(\lambda)$). 148
10.3 Binocular field of view limitation. 149

10.4 Stereopsis cue leads to disparity image in the brain by creating mirror image in L and R eye. Small distances give large disparities of β depth information. 149
10.5 Shadow stereopsis cue provides depth perception.(Copyright Akiyoshi Kitaoka 2005) 150
10.6 Convergence requires the eye to angle inward. Small distances give large angles. 150
10.7 Texture gradient as a depth cue. The flowers in front have more details than those in the background.(https://pxhere.com/id/photo/912200) 151
10.8 The objects are identical; the occluded object is behind. 152
10.9 Convergence of parallel lines creates a depth cue. 152
10.10 Light and shade create depth perspective. 153
10.11 The mountains at a distance have a bluish tinge due to atmospheric light scattering, giving rise to the color cue.(https://pixy.org/4801691/) 153
10.12 Basic digital camera schematic. Most are available as systems-on-chip. . . 155

List of Tables

2.1 Comparison between depth sensing technologies 18
2.2 Methods of depth recovery and reported performances 22

3.1 Comparative results of depth estimation methods 35
3.2 Comparison of ground truth and computed distance 39
3.3 Computed distance for GigE Camera 40

4.1 Aperture design details. 48
4.2 Disparity range. 49

6.1 Performance of weights for B+S+SSIM' model 86
6.2 Comparison of different encoders 86
6.3 Performance comparison of trained models 88

7.1 Trained Models and parameters. 106
7.2 Evaluation with different wavelets. 114
7.3 Model performances. 115

10.1 Properties of eye and camera: similarity and difference. 155

11.1 Various Datasets for Training 156
11.2 Various Datasets for Training 161
11.3 Various Datasets for Training 164
11.4 Tools for database development 167

Abbreviations

2D	Two dimensional
3D	Three dimensional
ADAM	Adaptive Moment Estimation
AI	Artificial intelligence
AMR	Autonomous Mobile Robots
AR	Augmented Reality
BerHu	Reversed Huber loss
CCD	Charged Coupled Device
CD	Chamfer Distance
CIE	International Commission on Illumination
cm	centimeter
CNN	Convolution Neural Network
COC	Circle of confusion
CRF	Continuous Random Fields
CV	Computer Vision
DFCCA	Depth From Color-Coded Aperture
DFT	Discrete Fourier transform
DMCA	Depth From Multi–Coded Aperture
DOD	Depth from defocus
DL	Deep learning
DLN	Deep learning Network
DLT	Direct Linear Transform
DNN	Deep Neural Network
DOG	Difference of Gaussian
DWT	Discrete wavelet transforms
ELU	Exponential Linear Unit
FCN	Fully convolutional network

FFT	Fourier Transform
FOV	Field of view
GPU	Graphic processing unit
HDR	High-Dynamic Range
IFFT	Inverse Fourier Transform
IWT	Inverse DWT
LIDAR	LIght Detection And Ranging
LOG	Laplacian of Gaussian
LR	Leaky ReLU
MAE	Mean Absolute Error
ML	Machine Learning
MSE	Mean Square Error
NDWTN	Nested Discrete Waveform Transform Net
nm	nanometer
PRNU	Photo Response Non-Uniformity
PSF	Point Spread Function
RADAR	RAdio Detection And Ranging
ReLU	Rectified Linear Unit
RGB	Red, Green, and Blue
RGBD	Red, Green, Blue, and Depth
RMSE	Root Mean Square Error
SFR	Spatial Frequency Response
SNR	Signal to Noise Ratio
SOTA	State Of The Art
SSIM	Structural Similarity Index
TOF	Time of flight
VR	Virtual Reality

Nomenclature

d	Distance of the object to the lens
f_o	focal length
g	defocused image
h	PSF or blur
I	sharp image
λ_i	hyper-parameter or weights
Y_{pred}	Predicted depth value/s
Y	Ground truth value/s

Chapter 1

Introduction

Depth is the dimension to a point away from the observer. Humans can perceive depth and judge the spatial position in a 3D world, which aids in many biological, social, and logical activities. Present-day machines also need vision for multiple activities to aid humans through 3D measurement, automatic inspection, security, medical imaging, navigation, robotics, meta-verses, virtual reality, and many more. Machines cannot perceive depth and hence require complex hardware and processing methods to visualize and make precision decisions. Here, the combined technique of hardware and algorithms provides the Cartesian plane representation of points from the 3D scene. Depth estimation is one of the fundamental computer vision (CV) tasks that is accomplished through machine learning (ML), artificial intelligence (AI), deep learning (DL), etc. Back in 1966, CV initiated as *"The Summer Vision Project", http://people.csail.mit.edu/brooks /idocs/AIM-100.pdf* to identify objects. Author [2] identified a method of image defocus to estimate depth. Over the years, research advances in biological neuroscience guided machine learning and shaped CV achievements [3]. Advanced neuromorphic-based algorithms enabled artificial intelligence (AI) through learnable neurons and further inspired neural networks by the year 2000. Neural networks are efficient AI algorithms that can classify complex objects like cars with highly variable attributes and environments and are based on biologically inspired convolutional layers and pooling layers. Deep learning is based on large neural networks. Neural networks learn a task through training before deployment. The training involves an organized dataset of multiple images and ground truth pairs related to the task. The network learns features by optimizing a regression loss function. A gradient descent algorithm with backpropagation between the image and the ground truth is used for the learning. The loss function and the dataset are therefore crucial to the accuracy, overall training time, and generalization of the outcomes.The datasets are large and need multiple iterations to minimize the training loss to an acceptable level. These algorithms therefore need multi-

core parallel processors like GPUs with tens of gigabytes of memory and run for a long duration. These systems are complex, and at present the thrust is towards efficient deep learning techniques with multi-scale capability and preferably low-complexity systems that can be implemented on popular mobile platforms with limited resources. textitThe debate over neural network complexity: Does bigger mean better?, VentureBeat, March 28, 2023, https://venturebeat.com/author /victordey/. This also puts a focus on efficient low-power capabilities for depth estimations. Even so, the present depth estimate capability of CV has unlimited daily applications for 3D measurement, automatic inspection, security, medical imaging, navigation, robotics, and many more. In a camera, light reflected from a scene is converted into electrical signals. The camera is an electro-optical system that records the shapes, colors, and movements of objects in a defined field of view (FOV). This camera largely mimics the human eye and interacts with CV algorithms aided by natural light or some arrangement of scene illumination. The camera captures a 2D scene from a real 3D scene by focusing the light through an arrangement of lenses into multiple aligned rows and columns of pixels in the image plane. An aperture in front of the image plane controls the energy passing to the pixels. Each of the spatially arranged pixels captures the contrast and color of the projected image. The image resolution also defines the accuracy of depth estimations and smaller pixel dimensions are better. The camera requires an optimum setting of the aperture to collect adequate light and a good focus to obtain a sharp image. These camera parameters can be fixed, manually adjustable, or automatic. The depth estimation task then converges to the measurement of the distance, in pixel dimensions, relative to the camera axes. Different configurations of this camera and algorithm define the processing of captured images for depth extraction. The popular arrangements are through stereo imaging (multi-views of a scene) or from a monocular (single) image. These are discussed in subsequent sections.

1.1 Depth estimate from stereo vision

A common method for camera-based depth estimation is based on stereopsis (stereo vision). Here, a pair of fixed cameras, kept apart at a known distance, capture overlapping images with two different angles [4]. In Fig.1.1, the projection of a point P at a distance Z is captured by two overlapping views.

P - Point in real world
P_L - Left image corresponding point
P_R - Right image corresponding point

X_L - Horizontal pixel distance of PL

X_R - Horizontal pixel distance of PR

B - Baseline distance between the center of left and right cameras

f_0 - Focal length of the cameras

pix - Physical size of a pixel in camera sensors

d - Distance between point P and camera centers

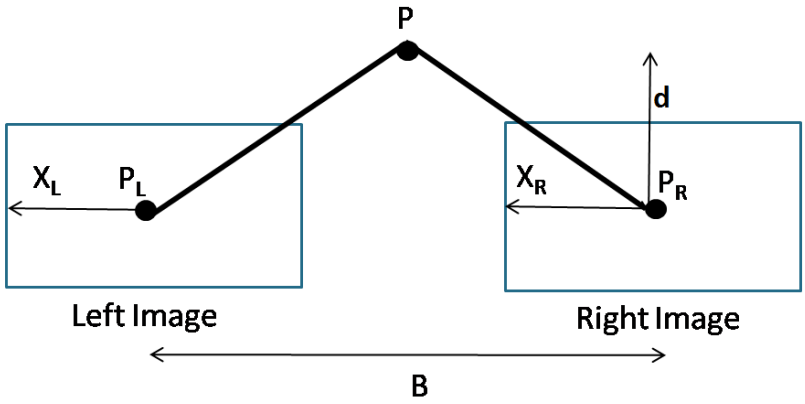


Figure 1.1: Stereo method and epipolar geometry

Traditional methods use this multi-view geometry to estimate depth, assuming the cameras are parallel to the point. Then a disparity map can be generated by the difference between the projected positions (X_L, X_R in pixels) in the left and right images.

$$Disparity(D_p) = X_L - X_R \tag{1.1}$$

The positions in one image are estimated by matching robust corresponding points in the other image. This is usually done by searching and similarity matching with a block of pixels. The camera parameters F_0, pix, and the baseline are known, so by triangulation, a depth map is obtained.

$$d = \frac{f_0 B}{pix D_p} \tag{1.2}$$

In the real world, the problem is more complex due to physical and practical aspects, such as:

- The accuracy of stereo methods is limited by the baseline. A large baseline provides large disparity and accuracy; however, this cannot be made impractically large.

- Further, the disparity follows a logarithmic relation with distance (d). Faraway objects fail to provide disparity, as the disparity is very small. Most commercial stereo cameras offer a depth range of < 10m.
- Each of the similar cameras' has manufacturing differences that mandate calibration for precise matching of image scale, translation, and gain. This calls for additional computing algorithms for image rectification. In practice, the camera may not be horizontal or parallel. Hence, matching the corresponding points requires searching the full 2D images instead of a horizontal axis. These add to computing costs.
- Reflective surfaces and non-textured regions degrade the features for robust matching. Also, repeating patterns can confuse the matching algorithm. These reduce the depth accuracy.
- The other inherent problems are poor dynamic range of cameras and blurred images of objects beyond focus (see section 2.1.2).

Over the years, significant research advancements have replaced traditional methods with deep learning-based approaches. These improve stereo accuracy, multi-scale feature maps, global structure, and the minimization of regression loss [5]. However, these methods are still limited by the baseline range.

1.2 Depth estimate from single cameras

Alternate options for improved depth range (> 10m) are single active cameras. These have an inbuilt, strong light source to illuminate the scene with known patterns. The reflected surface patterns require depth estimation. The power of the light source is scaled for a higher range and viewing angle. Presently, such cameras are expensive, bulky, consume more power, and have poor resolution *(Texas Instruments Technical White Paper-SLOA190B)*. Monocular depth estimation overcomes most of the limitations of stereo cameras and active cameras by using only one passive camera to reduce hardware technique complexity. These are simple, low-power, have the advantages of contrast robustness, have no magnification-based issues, and need simple calibration. These have immense potential applications in 3D reconstruction [6], medical imaging [7], robotics and autonomous driving [8, 9], augmented reality, and many more. Presently, there are many research papers on monocular depth estimation. The site *https://paperswithcode.com/task/monocular-depth-estimation* gives the recent trends and benchmarks. However, this method lacks reliable depth cues like parallax as in stereo vision, which poses a challenging and ill-posed problem. The monocular method suffers from scale ambiguity where the depths estimated are

relative and need information on the real scale through calibration. The method also suffers from structural ambiguity, like occlusions and weak texture patterns. Further, any 2D image can be the projection of many different real-world scenes. Initially, researchers used camera focus or aperture parameters to capture multiple images and estimate depth from the defocus blur filters [2]. Such methods were not practical, as customized setups were needed to capture multiple images while other camera parameters remained constant. The depth map was sparse with such methods. Later, [10] used cues like texture, haze, and global features and implemented supervised learning to train an MRF model to estimate depth from a single image. This called for meticulously crafted computer vision (CV) algorithms to solve this nonlinear problem. The author [11] applied Convolutional Neural Networks (CNN) to successfully address the ill-posed problem. They train a model to learn the structure of the scene, use a second network to refine it using local information, and directly regress the depth map. Recent advances use deep neural networks with improved datasets for training, evaluation, and benchmarking. These networks train models with carefully designed loss functions and relevant datasets of color images and ground truth to regress depth estimates. These networks use scale-aware losses [12] to overcome scale ambiguity and complement image features with a set of auxiliary variables to avoid structural ambiguity [13]. The projection problem is solved through various visual depth cues like perspective, shadows, relative size, obstruction, position, color, horizon, etc. [14]. Here, the prediction accuracy depends on network architecture and regression loss functions. Training with a large collection of well-defined and labeled datasets is also crucial for generalization and prediction quality. Present state-of-the-art methods use resource-intensive and complex networks to directly regress the depth map.

1.3 Challenges for Mobile Platforms

Recent advancements in semiconductor technology have increased computational power at a lower cost and are pushing near-real-time depth inferences to mobile platforms and edge computations. These mobile platforms have resource constraints like size, weight, and power (SWaP), giving rise to new challenges. Active systems are avoided as they are costly, bulky and power-hungry. The stereo methods need two cameras and increase the computation complexity for L-R correspondence matching, contrast normalization, and magnification adjustments. These increase setup complexity and are also avoided. Single-camera-based depth estimation is cheap and suited for mobile platforms. However, the accuracy depends on the optics used and the setup for acquiring images. The methods rely

on changing the camera parameters and multiple image acquisitions. These are not suitable for use with moving targets. Monocular depth estimation methods use a single image along with deep learning to provide the best solution. However, the computational complexity is high, requires elaborate training datasets and suitable loss functions. Therefore, mobile platforms demand real-time inferences with efficient software, lightweight-efficient networks, and good training.

1.4 Motivation and Research Objective

Most research papers focus on novel depth estimation methods with laboratory experiments or simulations through large computation capabilities. The literature lacks aspects like depth range, dynamic targets in the actual environment, and practical setups. These gaps need study with elaborate experiments to identify further scope for improvement. Most depth estimation methods provide relative depth. A good calibration method is hence the need of the hour for retrieving absolute depths. Dedicated literature was not found for such calibration. A study using new calibration targets is proposed to retrieve absolute depth. Further, it was highlighted that mobile platforms demand accurate estimations with efficient networks. This calls for investigating network systems like regression loss functions and network architecture for reductions in computation complexity. Again, most of the literature deals with standard loss functions for depth estimation. A study is conducted with alternate loss functions to improve the depth map details. Most light networks, like UNet and MobileNet, are utilized for classification and segmentation, with very few for depth estimation. The thesis work also develops an alternative light network for depth estimation.

The thesis has three major contributions in the research domains related to monocular depth estimation from a single image.

- The thesis aims to develop a framework for absolute depth estimation. This need arises from the relative depth maps obtained from present monocular depth estimation methods, whereas most applications need absolute depths. Proposed new calibration procedures to bridge this gap.

- Subsequently, extensive work on monocular depth estimation with deep learning is carried out. The regression loss functions for efficient deep learning are investigated and analyzed. New efficient multi-loss functions using SSIM and sharpening functions are proposed to improve depth map estimation accuracy.

- Monocular depth estimating methods are complex and computationally intensive, which limits their usage on mobile platforms.To tackle this issue, a new lightweight network architecture is proposed to estimate monocular depth. This network is multi-scaled, uses wavelets, and uses attention to train faster and improve accuracy. The performance competed with present-day state-of-the art algorithms.

1.5 Organization of the Thesis

The format of the thesis is:

- Chapter-2 describes the camera and discusses the basic camera model with a thin lens. A market survey on depth cameras highlights the various methods used and the technological capabilities. Stereo vision methods are analyzed and compared with active mono cameras. It is concluded that stereo methods have limitations in range, while active camera methods are expensive and power-hungry. This leaves the monocular method as a potential candidate for an efficient and cheap solution for depth estimation. A literature survey on monocular cameras is complete. Traditional methods of monocular depth estimation using defocus, aperture, coded aperture, blur cue, and light field are summarized. Modern AI-based learning methods are also surveyed. These require large, relevant datasets for training and evaluating a network model. The selection of the database and its usage for the thesis work are explained. Training a network model involves minimizing the loss function. The metrics for evaluation are discussed, and popular metrics for performance evaluation are finalized as RMSE, REL, Log10, and threshold. Further, various applications of monocular depth are listed. The challenges for the work are summarized.

- Chapter-3 discusses classical methods of depth estimation from defocus by changing focus. The method relies on the fact that when an object is focused, the corresponding image pixels have maximum sharpness. The out-of-focus pixels are blurred and are represented by the point spread function (PSF) with distinct σ values. These are related to the distance based on the thin-lens equation. The method uses the camera's intrinsic and geometric properties to estimate depth. In the experimentation, two images at different focal points are acquired, and the blur sizes (σ) are related to the respective focus settings. The images are initially smoothed with a Gaussian filter before being operated on by a Laplacian filter. The Laplacian of the Gaussian operation on the images gives the maximum contrast for focused pixels. The two

smoothed images are compared with the Laplacian average, and the magnitude map is estimated. The depth map is recovered through the relationship between magnitude and the thin lens equation. As the depth estimates are relative, a calibration is used to recover the absolute depth. A worst-case depth error of 4% is observed concerning ground truth. Though the method is simple, it has limited applications due to its short depth, still scenes, hardware setup, and aperture diameter. The method cannot deduce depth from pattern-less zones, which results in sparse depth maps. These non-recoverable depth zones are an ill-posed problem.

- Chapter-4 presents the work on aperture-based depth estimations. Multiple methods using shaped aperture masks and colored masks are possible. The color-coded aperture-based technique is based on two off-axis apertures with different spectral filters. This results in two spatially displaced images, leading to a disparity map. The disparity map for a known focal length and aperture diameter is related to the object distance by the thin lens equation. The work analyzes the critical requirements of filters, impacts on SNR, and image resolution. As in stereo, large depths are inaccurate due to inherent out-of-focus blur and reduced disparity. The characteristics of stereo methods are relevant here too. The range of depth that can be estimated is limited by the distance between the cameras and the diameter of the aperture. Weak scene patterns cause errors and are an ill-poised problem, like in DOD. Deep learning methods can improve on this problem, but the range of improvement is difficult.

  The second work with a multi-coded aperture is with synthetic images, as hardware modification needs professional skills. The work uses two pairs of custom-shaped apertures, which are usually used for image capture. One pair has asymmetric aperture patterns. Here, the aperture model is convolved with the image to modify the defocus point spread function (PSF) in the frequency domain. The two pairs of images so obtained are fed into a generalized Wiener deconvolution algorithm. Each pair of images provides phase information to reduce the ambiguity of depth in front of a focused object or behind a focused object. A known image PSF prior recovers the image with an unknown blur. If the PSF matches the unknown blur, a focused, sharp image is obtained. Here, the PSF is regressed for all samples of depth to recover an all-focused image. The final values of the depth estimate generate the depth map. Here, the image prior is assumed to be similar for all images, which is not the true case. Again, the finite blur of the aperture /lens in the images limits the depth accuracy. The depth estimates in homogeneous regions lack frequency content and

are ill-proposed. As two images are required with the same viewpoint and focus, this limits their use.

- Chapter-5 presents a study on a unique calibration target to provide absolute depth using real data. In this work, the blur variation in a single image is utilized to estimate a full-depth map. This efficient method involves Gaussian filtering of the image. Sharp features are more affected by filters than blurred features. The ratio of this blurred image to the original gives the magnitude of the gradient change. This magnitude of ratio represents image blur, which is dependent on object distance. This magnitude, therefore, represents the relative depth estimates for an image. An alpha-matte interpolation method gives a full-depth map. The proposed target is used to recover absolute depth estimates from the relative depths. The calibration uses two targets. One is made from multiple ground-truth targets of known depths captured with an experimental setup. The second is made from multiple synthetic targets with known PSF scales corresponding to the experimental depths. A ratio of the first and second derivatives of both targets estimates the depth. These depths of information are related and hence give the calibration coefficients. The method used has advantages like immunity to spectral sensitivity, contrast variations, and magnification issues. Additionally, the target also characterizes the blur range in the image. Verification of the methods showed a good correlation. This will be useful for calibrating relative depth using any depth estimation method.

- Chapter-6 includes a study, experiments, performance, and summary of training loss functions. Most classical depth estimation methods are ill-posed due to weak patterns and are solved with deep learning. Deep learning involves defining a network, training the network model, and using a good loss function to directly regress the depth map through gradient descent. As loss functions define accuracy, efficiency, and outcome, the work studied various popular loss functions used for training for depth estimates. The work proposes tailor-made edge loss functions with SSIM with sharpened images, Laplacian loss, Sobel loss, LOG loss, and DOG loss. The combined loss includes these losses along with L1 losses, reverse Huber losses, and gradient losses. The evaluation of these proposed losses was done with a ready-made network, an established transfer learning method, and a popular NYU dataset to identify the improvements. Analysis is based on the quality of depth maps and on popular standard quality metrics for easy comparison. The scores achieved fare better with the available state-of-the-art performances in this domain. The work concludes that edge

functions improve the estimation of depth maps. Further, derivatives and differentiation are better than smoothing-type loss functions.

- Chapter-7 studies wavelet-based network layers to reduce network complexity. Monocular depth estimation from a single image is ill-posed and is best solved by an encoder-decoder type of network. The present survey revealed that most networks are computationally expensive, and lighter networks are largely unexplored. This work proposes an alternative light network to reduce the computational complexity of the present system based on DenseNet, ResNet, etc. Here, wavelet-based layers replace the down-sampling and up-sampling without any loss of information. The wavelet coefficient is learned through training. The proposed nested network is NDWTN, and 13 variants are evaluated. These networks are multi-scale, have dense skip and attention functions, and have dense convolution blocks to aid learning of local features. The study includes optimization of convolution layers, residual convolution layers, batch normalization layers, activation types, and attention features. All the networks are trained on a standard NYU dataset and evaluated using standard quality metrics. All results are analyzed with UNet, UNET++, and DenseNet models. Ablation studies are completed, and training parameters as well as network architectures are optimized. The proposed network models successfully predict depth from an image after training from scratch with less than 400 image pairs. The performance of the proposed models is superior to that of published UNet, DWT-type UNet, and UNET++ models. The network has the best RMSE metrics among all the present state-of-the-art models. The proposed networks also trained faster and yielded improved scores. The NDWTN is a good fit for depth estimation on mobile platforms.

- Chapter-8 concludes the work of the thesis. This thesis studies classical monocular methods of depth estimation using computer vision (CV) algorithms. Experimental studies were completed with defocus and aperture-based methods. All these methods provide relative depth maps and were all ill-poised at weak patterns of the scene. A new method of calibration of relative depth maps has been successfully implemented to retrieve the absolute depth map. Deep convolutional neural networks (CNN) were studied to address the ill-posed problem of monocular depth estimation. Here, training, architecture, algorithm loss functions, and dataset are crucial to the depth estimation quality and accuracy. New loss functions are developed to enhance the edges and thus improve the details of depth maps. The performance score of the network with such loss functions was higher. A new light-weight, efficient network using

wavelet layers was also developed for faster training, using smaller datasets, and having low computational complexity. These networks met their goals with dense layers, attention, and skip paths. The RMSE performance was the best among the present state-of-the-art models. The research contributions are summarized. The thesis presents new directions of research for monocular depth estimations.

Chapter 2

Depth Estimation and Methods-A Survey

2.1 Introduction

Humans have two eyes and depth perception is based on binocular, monocular, and oculomotor cues [15]. The subconscious depth processing is aided by prior knowledge of the surroundings and the objects. Thus, humans can also judge depth with just one eye using depth cues. In machine systems, cameras are used to mimic the eye. A summary of human depth perception cues and a comparison with cameras are given in Appendix B. Here, computer vision (CV) mimics human sense through stereo imaging or monocular depth cues. Here, the challenges are deriving accurate depth information with a feasible cost in near real-time with lower power, lower computation, and cheaper input systems. An additional challenge for AI learning methods is the need for large datasets to train. Presently, depth is derived with low-cost RGB cameras using complex CV algorithms like stereo-vision, aperture variation, focus or defocus variation, defocus variation, etc. An overview of the camera model used, available depth-sensing cameras and methods, performance metrics, and applications are discussed below.

2.1.1 Camera

Machines enhance human performance. Machines mimic the human sense and perceive depth using Computer Vision and cameras. The camera is an optical instrument that captures mostly 2D images. Some advanced cameras can also capture 3D images. The camera has evolved from a pinhole camera to an optics-based lens camera. The image recorder, or photosensitive zone, has also evolved from photosensitive chemical plates, films, analog vacuum tubes, and solid-state CCDs to modern active-pixel digital sensors. The basic camera consists of a light-sealed box. The light enters the box through a lens system, a

small controllable hole (aperture) that regulates light input, and a mechanical or electronic shutter to integrate photons. The lens focuses the light onto the digital sensor to capture the image. The camera focuses on objects of interest by moving the lens along the optical axis to sharpen the object's features. The aperture consists of a ring of overlapping plates (the aperture ring), which adjusts the light intensity by changing the opening. The digital sensor has a cluster of photosensitive pixels. The dimension of the pixel defines the spatial resolving power, contrast resolution, and dynamic range characteristics. Present sensors have more than 64 megapixels. Color digital sensors use red, green, and blue filters to capture 2D RGB images. Modern sensors use pixel-size lattice filters arranged in a Bayes pattern mosaic to mimic human spectral sensitivity. This mosaic is a repeated 2×2-pixel set that has a pair of diagonal green pixels, a red pixel, and a blue pixel. The electronics or image processor uses a demosaicing algorithm to interpolate the RGB information. Cameras that rely on natural or artificial light to acquire images are passive systems. Active camera systems have in-built illumination sources and can work in both illuminated and dark environments. These active systems are complex, power-hungry, and expensive compared to simple passive systems.

2.1.2 Modeling the camera

A basic camera model is a paraxial system with a thin lens governed by lens law. The model's extrinsic parameters consist of the location, rotation, and translation in the world, while the intrinsic parameters are focal length, scale factors, optical center, and skew. Fig. 2.1 shows the optical schematic, where d, d_f is the distance to the objects from the center of the lens, which has a focal length of f_0 and a stop number of N (optics aperture/ f_0). The object is a cluster of points representing some 2D geometric shape. The lens and image plane are rigid, so while some object points focus on the image plane, other points from nearby objects are defocused. The defocused point on the image plane generates a blur or a circle of confusion (COC) with a diameter c. This c is proportional to the distance d from the lens as

$$c = \frac{|d - d_f|}{d} \frac{f_0^2}{N(d_f - f_0)} \tag{2.1}$$

Thus, in an image, various focused and defocused points are created, corresponding to scene object distances. Knowing c, a 2D depth map related to spatial blur variations of an image, and corresponding object distances can be derived. The dependency of c on object distance reduces after the hyperfocal distance (Fig. 2.2). The assumptions for this model are:

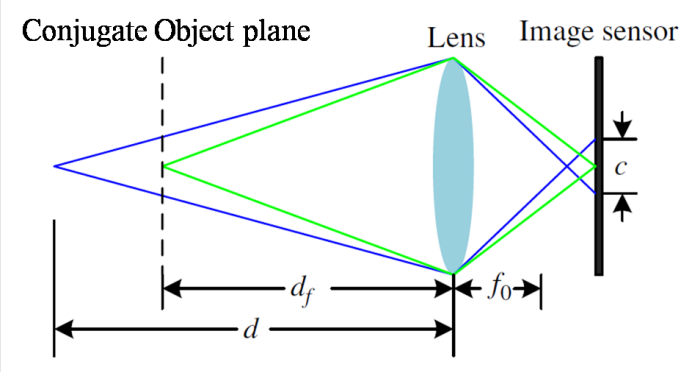


Figure 2.1: Overview of blur estimation approach

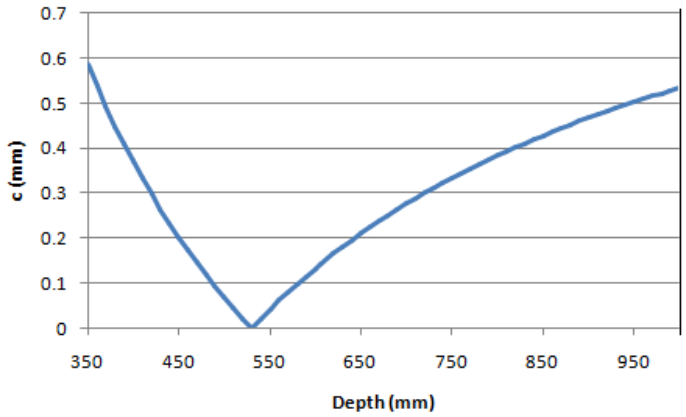


Figure 2.2: Variation of blur diameter (COC) with distance (Equation 2.1). It is minimal at focus.

1. The camera focuses on some object. Features are larger than the blur radius in the image plane to define edge contrast.

2. Blurred edges originally had sharp features with step edges. These edges are not recoverable at occlusion, boundaries, or for translucent objects.

3. A single-disc PSF represents the blur in the Fourier domain.

4. In an image, the blur diameter changes spatially.

5. The lens aperture does not over- or under-expose the image, thus conserving its features.

6. Objects in the scene are not transparent or as reflective as a mirror.

7. The model is lossless.

An image consists of many focused and defocused points that correspond to various object distances in the scene. Hence, the construction of a 2D depth map is possible with the knowledge of spatial blur diameters in the image.

2.2 Market survey of depth cameras

Passive cameras record images by projecting a real 3D scene within a field of view (FOV) onto a 2D grid of sensors. This grid includes spatially distributed sensors for color channels, viz. Red, Green, and Blue (RGB). These color sensors are clubbed into synthetic 3-channel grids. Each sensor in the channels, called $pixel$ records the contrast (reflected light) of visible objects within the FOV. The pixel contrast value is digitized into ranges from 0 to 255, such that a random black pixel is (0,0,0) and a white pixel can be (255,255,255) for the 3 channels. In this projection, the depth information of the scene is lost, resulting in a 2D RGB image. Adding a separate channel with a pixel-wise depth map to the recorded grid will provide the 3D space coordinates. This 4-channel record is the RGBD image. A camera usually does not provide depth by itself. An arrangement of the 2D camera, algorithm, computing, and calibration is used to get the depth information. Algorithms can estimate depth using one or more RGB images using triangulation. Most single-camera-based systems have active cameras. These have different methods of using active light to get depth. Some popular depth cameras available on the market are discussed below:

2.2.1 Stereo vision

This consists of two or more cameras to mimic human binocular vision and exploit the stereo disparity. The cameras used may be color (RGB) or monochrome (infrared) based. The depth computing algorithm runs inside the unit. A stereo vision algorithm like [4] is one method where images from two or more cameras are used to predict depth. Here, the local correspondences within the images are solved to derive a depth map. Stereo vision mandates matching of scale, translation, and gain among the cameras in use. Since these images need sufficient details and texture, good lighting or outdoor imaging is required. The infrared (IR) types usually have an IR source built-in for indoor imaging. These cameras have a small depth range of 3m with 2% accuracy. Fig. 2.3 shows one such product. 3D information can be estimated from two or more cameras [4] using stereo-vision. Stereo imaging uses the triangulation principle and epipolar geometry to derive depth. Here, two side-by-side cameras provide two overlapping views as shown in Fig.1.1. These two im-

Figure 2.3: RealSense™ D400 series stereo depth camera. The module contains two IR cameras, an IR source, and an RGB camera.

ages are rectified and the problem of correspondence pixel matching is solved to get the disparity map. The disparity is proportional to the baseline. A smaller baseline leads to a smaller disparity among the images. For a fixed baseline, disparity follows a logarithmic relation with d and close objects create a larger disparity. Fig.2.4 shows the disparity map for stereo images. The depth map is proportional to the baseline and inversely proportional

(a) Left image **(b)** Right Image **(c)** Disparity map

Figure 2.4: Left, right images and the derived disparity image

to disparity (Fig.2.5). The stereo technique has found good applications in mobile phones

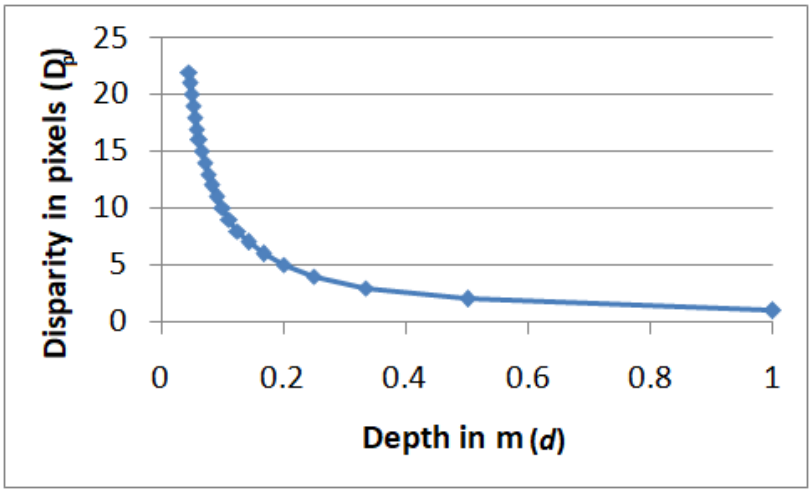
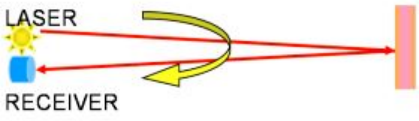


Figure 2.5: Relationship between depth and disparity f_0, B, pix are fixed for a camera setup and are kept here as unity for simplicity.

using two cameras or with custom sensors with twin pixels. Stereo vision does not provide real 3D, but 2D images with depth information, i.e., one cannot see images from different angles. Also, stereo methods have poor accuracy for distant objects. CNN has been used to predict depth from stereo image sets. These are trained to predict image alignment loss, left-right consistency, and the disparity map. In stereo imaging, it is assumed that the cameras are leveled, the image planes are flat, and they have the same focal length. The challenges in stereo depth lie in solving correspondence matching, which is affected by scene texture, occlusion, shadows, surface reflectivity, repetitive patterns, light mirroring, and transparent surfaces. Other issues are matching image scale and gain, image translation, precision calibration, and rigid mechanical structure [16].

2.2.2 LiDAR

LiDAR, or Time of flight (TOF), cameras do not mimic human vision and are termed active cameras as the internal light source is essential for the operation and cannot rely on external sources. These cameras rely on the echoes of transmitted light from surfaces. The total time taken by light to reach the target and echo back is the TOF (Fig. 2.6). The distance is then related to the light travel velocity and the travel time. These cameras are usually monochrome with high power sources and photon-counting sensors. The depth range and accuracy are better, and models with a 10m range and 0.001% accuracy are available.



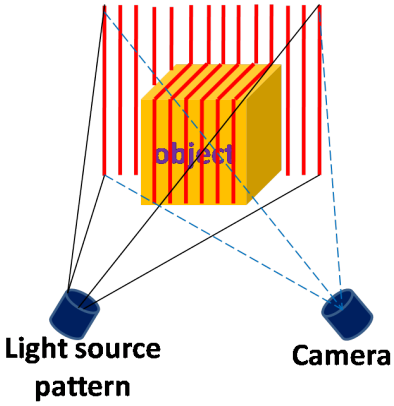
(a) The LiDAR schematic shows the round-trip delay for echoes. The delay time is related to depth.

(b) Helios2 TOF depth camera. The module contains an IR sensor and a Laser IR source.

Figure 2.6: LiDAR Time Of Flight depth camera.

2.2.3 Structured light

This is also an active camera and needs an internal light source. In structured light-based depth sensing, a light pattern (stripes, dots) is projected on the target and imaged by a camera (Fig. 2.7). The distortions of the patterns allow computation of the depths. Structured light methods achieve high spatial resolutions using a conventional camera, which is a major advantage. Microsoft's Kinect camera was popular for gaming applications. The typical depth range and accuracy of such cameras are up to 8m and 0.001% respectively.



(a) Imaging with light patterns. The distortions give depth information.

(b) Orbbec 3D Astra Series depth camera. The module contains an IR sensor and a patterned IR source.

Figure 2.7: Structured light depth camera.

A comparison between depth sensing technologies is summarized in Table.2.1.

Table 2.1: Comparison between depth sensing technologies

	Stereo vision	**Structured light**	**LiDAR**
Principle	Disparity of 2D images	Projected pattern distortion	Travel time of echos
Computation	High	Medium	Low
Cost	Low	High	Medium
Depth Accuracy	cm	um cm	mm cm
Depth Range	Limited	Scalable	Scalable
Light environment	Good light	Low light	Low light
Response	Medium	Slow	Fast
Power Consumption	Low	Medium	High

2.3 Monocular Depth Estimation Methods

In the robotic world, LiDAR and RGBD cameras (like Microsoft Kinect) provide depth maps. These are active cameras with a light source to illuminate the target and image it.

Presently, the capability of high-resolution LiDAR cameras is limited to a range of less than 10 m. Hence, dense pixel-wise depth maps from these types of cameras are rare. Further, the cost and high power of these cameras drive the use of passive, low-power, simple, and high-resolution cameras in many robotic applications with depth estimation algorithms. A trend for single image depth estimation (SIDE) methods is on the rise. These methods are described below.

2.3.1 Depth from Motion

Earlier depth recovery methods used geometry-based methods using multiple images. Structure from motion (SfM) [17] recovers the structure from four non-coplanar points through three orthographic projections. Sequential images with different time stamps also provide depth cues. Here, the depth estimation accuracy is dependent on correspondence, feature matching, and quality image sequences. SfM suffers from moving objects and from monocular scale ambiguity in between the frames.

2.3.2 Classical Methods

Early efforts focused on depth rather than automatic focus. The method automatically focused on a point in the scene using an image quality algorithm and varying lens positions. The method could provide depth for a single point. Ref. [2] first used the gradient of focus to estimate absolute depth from two images of a scene captured with different apertures. This gave depth to the image features. As shown in Fig. 2.2, the distance d is given as

$$d = \frac{f_o v_f}{v_f - f_o - \sigma N} \tag{2.2}$$

Here, the authors model v_f as the distance between the lens and the image plane and σ as the spatial constants of the point spread function (PSF) of the blur circles in the image. Knowing σ, the absolute depth is derived. As the blur changes with the aperture, they take the ratio of the Laplacian of the images along with linear regression to get σ of both images. Research on depth from defocus and depth from focus methods analyzes the sharpness or blur of an object either in the spatial domain or frequency domain. Later, they used multiple images with camera parameter variations to derive the depth map instead of using a single image with a known sharp reference image [18]. In most cases, a thin lens model and blur with a Gaussian PSF are assumed. In a scene, the blur or COC, is dependent on the distances of the objects, as some get defocused. In the spatial domain, depth was estimated by

varying the focal length and aperture size ([2, 19, 20, 21, 22]. Ref. [21] used spatial-domain convolution Transforms to arbitrarily blur the images. Ref. [22] added texture cues to get full-depth maps instead of sparse-depth maps. Author [19] proposed depth from focus by convolution of the sharpest image with a sequence of calibrated filters and minimizing the error with the other image. The filter, which provided the minimum error, gave the depth using a lookup table. The methods did not suffer from correspondence issues as with stereo cameras but had limitations like magnification variation (due to focus change) and contrast variation (due to aperture change). In the frequency domain, the blur is quantified with the point spread function (PSF). PSF is represented with a 2D Gaussian function $g(\sigma, r)$, where r is the radius. The increase in blur due to defocus decreases the high-frequency energy. This is more prominent in image features with sharp edges. Comparing the edge sharpness between focused and defocused images relates to the COC, which is again related to the real object distance. The images are usually preprocessed with a Laplacian operator, and the ratio of Laplacian magnitudes, contrast ratio, and PSF response is widely used to recover depth. Author [23] varied the defocus of 2-–3 images using either aperture or focal length. They estimate depth by convolution of the sharp image with a PSF of an appropriate blur filter with a defined σ, to produce the blurred image. The depth is then related to the calculated value of σ. They improve the previous method by applying a mask based on the thresholding of the Laplacian values and a rotationally symmetric PSF. The depth estimation methods have inherent errors due to noise, edge discontinuity, contrast variations, etc. Researchers implemented various interpolation, approximation, segmentation, and matting methods to improve the depth map. Here, the accuracy depends on at least one near the ideal sharp image. This factor is limited by camera quality, camera manufacturing tolerances, and the accuracy of adjustments. These methods require two or more images of the same scene.

2.3.3 Coded aperture methods

The camera aperture can be replaced with single or multiple-shaped apertures or optical phase masks [24, 25]. The aperture can also be changed to multiple color-coded apertures [26, 27, 28, 29] to provide multiple images on the sensor. The aperture mask hardware is inserted inside the camera (Fig.2.8). The image convolves with the aperture to introduce a spatial offset among RGB color channels. The direction and quantity of the offset are a function of the distance of an object from the plane of focus. Thus, aperture centers separated by r_c from the optical axis lead to a shift of Δy or Δx in the image plane depending

on object distance d. In terms of pixel size pix the shift is given as:

$$\Delta(x,y) = \frac{2r_c f_0(d_f - d)}{pix(d_f - f_0)d} \tag{2.3}$$

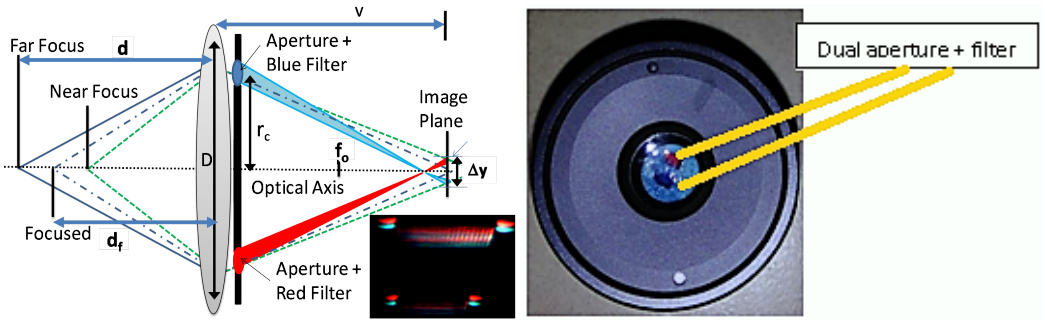


Figure 2.8: Dual aperture: (a) Schematic of the optical system (b) Inset image shows the color shift of a ceiling lamp (c) Modified camera with dual color apertures

2.3.4 Blur Cues

The blur present in an image also provides a depth cue [30, 31]. This method is immune to contrast, magnification, and spectral sensitivity as a single image is used. The first and second derivatives of the sharp features in an image provide the location and magnitudes for a relative depth map. In all the above methods, noise, shadow, and discontinuity of edges lead to errors. These methods work well for near-range objects but become inaccurate for distant objects due to nonlinear dependencies. Table.2.2 summarizes the accuracy achieved by the researchers and their methods.

2.3.5 Light field method

The light field is the total of light rays through any position and direction in 3D space and is defined by [32] using a 5D plenoptic function as $L(x,y,z,\theta,\phi) \in R^5$ having two angles (θ,ϕ) and three coordinates (x,y,z). Authors [33] optimized this function for 4D representation as $L(u,v,s,t) \in R^4$. Here, (u,v) is the first plane and (s,t) is the second plane (Fig.2.9). A 4D light field camera, also called a plenoptic camera, can simultaneously record spatial and angular information of light rays with multiple cameras (fly's eye) or with a single-shot camera having a microlens array [34, 35] or an optical mask [36] between the main lens and image sensor. In single-shot cameras, the microlens array or optical mask

Table 2.2: Methods of depth recovery and reported performances

Method	Performance	Issues
Stereo	Very Accurate	Uses triangulation and needs two cameras or two views. Suffers from corresponding and occlusion problems due to different viewing positions. Complex solutions and higher computational cost.
Depth from motion	Not Available	Uses motion cues from multiple images and optical flow. Higher computational cost.
Depth from shading	<10% error	Uses only one image, Depth cues from textures, defocus, etc. are used along with MRF techniques. Requires controlled illumination, which is difficult in the natural outdoors.
Depth from defocus	<2% error	One camera can be used to take near and far images. A defocus cue is used for depth computation. Very sensitive to changes in image intensity.
Depth from focus	<3% error	The focus of a single camera is varied and a large number of images are taken. One main issue is that focus change also changes magnification, which Requires computational speed.
Depth from aperture	<3% error	Depth is computed by varying the aperture of a camera. Magnification is constant. However, the contrast varies.

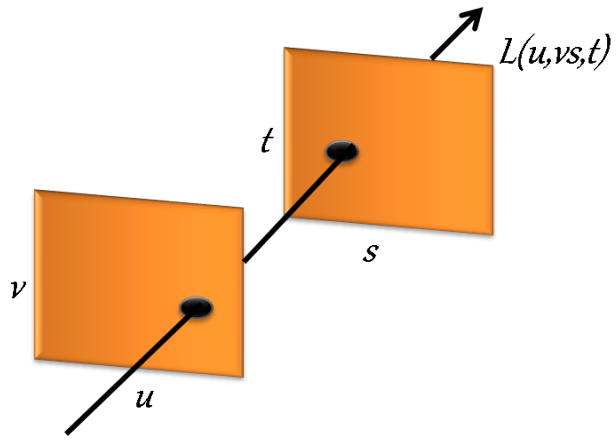


Figure 2.9: Spatial and angular information of light rays in 4D representation

separates a fraction of incident rays originating from objects into bundles or sub-apertures having the same origin and position but with different angular directions (Fig.2.10). There are now as many points of view as the number of pixels behind each microlens, leading to a focal stack of images covering a range of focusing distances. Depth is estimated using multi-stereo CV methods using pairs of refocused images from the focal stack. Also, the amount of focusing required for the rays crossing the focal stack can be used to predict depth. The depth range is defined by the dimensions of the lens array, the mask, and the sensor array. The Lytro Light Field plenoptic camera was the first commercial camera to exploit this method.

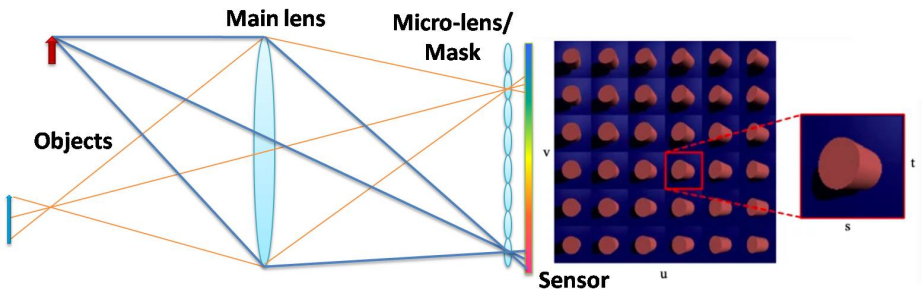


Figure 2.10: Plenoptic camera: Simulated multi-view of microlens array.

2.4 Learning-based methods

Machine Learning (ML) is a sub-field of artificial intelligence (AI) that learns from data, optimizes features, and performs tasks without using explicitly programmed algorithms. Deep learning (DL) is a sub-field of machine learning and uses complex algorithms to mimic human brain logic. Such algorithms process unstructured data, extract underlying patterns, and predict relevant outputs with minimal human intervention. Neural networks form the algorithms. Depth from monocular cues needs resolving ambiguous solutions (structure, scale, and projection) due to non-unique predicted depths. Here, many different 3D scenes can be solutions for one 2D image. CV has advanced from classical methods that relied on multiple images to extract depth to deep learning depth from a single image. Estimating a dense depth map from an RGB image requires predicting metric depth values at pixel level and involves a deep understanding of the detected features and their relations in 3D. Monocular methods of recovering depth are challenging as multiple solutions are possible due to a lack of deterministic cues, as in stereo. Here, multiple cues like perspective, size, obstruction, position, color, and horizon are needed. Machine learning networks are suited to solve such ill-posed inverse problems after training. A Markov Random Field (MRF) learning algorithm to capture monocular cues was proposed by [10]. He estimated a depth map from a monocular RGB image with supervised learning and regression loss using multiple filters on small image patches. The first application of neural networks to monocular images was by [11]. They estimated the global and local structures with two networks in a sequence to build a depth map using both NYU and KITTI datasets and supervised training. [37] proposed a fully convolutional network (FCN) architecture with residual learning for depth estimation. The network architecture is defined with custom building blocks like convolution layers, activation layers, pooling functions, and expansion layers [1, 38, 39]. Researchers have incorporated parameters related to se-

mantic segmentation, surface normal, and fundamental image decomposition into their loss functions to improve the robustness of training and learning of 3D relations. The monocular depth estimation methods evolved from unguided to RGB guidance. Classical methods of depth estimation for stereo and monocular cues have poor performance for large depths due to blur. Deep learning learns multiple cues to improve upon this aspect. Ref. [40, 41] used a deep neural network to estimate a full-depth map from a single out-of-focus image. Other researchers [42, 43, 44] recover depth information from a single image after learning the blur cue. Ref. [45] trained algorithm to learn optimal PSFs to estimate kilometer range depths. Author [46] used multi-scale CNN and Continuous Random Fields (CRF), while [47] proposed attention-guided networks to improve the estimation accuracy. Transfer learning was implemented by [1] to reduce the training time. The training strategies of SOTA CNN are classified as supervised or self-supervised [37, 38, 42, 43, 44, 48]. The supervised training methods [11] require labeled depth images for the network training to converge. Self-supervision eliminates the need for labeled data and exploits 3D geometry from stereo monocular videos or image pairs [49, 50] for image reconstruction or derive camera pose to estimate depth. New research studied attention mechanisms and transformers to preserve depth image details [51]. Authors have exploited innovative photometric loss [52], symmetry loss [53], and left-right disparity consistency loss [50]. Accurate depth images have also been reported with photometric stereo using a pair of surface orientation maps (surface normals) [54, 55, 56, 57] and by fusing photometric stereo with light field information [58].

2.4.1 Convolutional neural networks (CNNs)

DL has CNN (convolutional neural net) as one of its architectures. CNN consists of Convolution layers, Pooling layers, and activation layers. The Convolution layer uses filter kernels to extract features. This results in weight sharing, whereas in DL, the inputs are multiplied with different weights without weight sharing. The kernel learns through backpropagation. The weights and intended tasks solve a non-convex optimization problem. The error between the network output and the labeled data (ground truth) is back-propagated to update the trainable kernel weights. The Pooling layer selects the maximum value within a cluster of features and downsamples the feature map. This results in highlighting important features for intuitive classification. Additional layers of enlargement and deconvolution upsample the downsampled feature map and provide predictions from the classes. Increasing layers in a network improves accuracy. However, large and dense networks suffer from the

vanishing gradient problem as the information passes through input and output layers. Repeated multiplications in back-propagation reduce the gradients, which update the weights and biases. The fading away of gradients leading to loss of training is called the vanishing gradient problem. DenseNets connect every layer directly with each other and reuse the feature maps by concatenating the input and output feature maps of each layer instead of summing them. ResNet solves the vanishing gradient problem with skip layers. Image classification networks like ResNet, VGG, DenseNet, etc. have only an encoder. Semantic segmentation and depth estimation networks like SegNet, UNET, etc. use encoder-decoder architectures. The encoder has a sequence of convolutions with increasing filter banks and downsamples the feature map spatially. The decoder reverses the encoder structure with transposed convolutions to upsample the spatial resolutions. In an encoder-decoder network, the encoder can be DenseNet, ResNet, etc. as the backbone. Depth estimation tasks assign each pixel a depth value with an encoder-decoder CNN network architecture. The encoder extracts significant features from the input image to provide a down-sampled feature map. The decoder upsamples the compressed feature map to produce pixel-wise depth predictions from the classes. Pretrained image classification networks form the backbone of most architectures. [1] uses DenseNet as an encoder, while [37] uses ResNet as an encoder for depth map predictions. Networks can also be customized [1, 38, 39]. The training can be either supervised or unsupervised [37, 38, 42, 43, 44]. This thesis uses convolutional neural networks with an encoder-decoder architecture. The encoder is based on DenseNet.

2.5 Datasets

Learning to estimate depth needs datasets with images and corresponding ground truth data. Since the training, evaluation, and testing of the learning network are iterative, large numbers of low-correlation images are required to validate the algorithm. Generating datasets is a tedious task as it is mostly manual. Standard data sets are also available from researchers, organizations, and universities. These datasets are acquired with cameras with dedicated setups and manually updated to provide the required annotations and ground truth. Many synthetic and simulated datasets are also available for training and benchmarking. These synthetic datasets are more accurate and provide pixel-wise depth. A good number of databases have been created from information available on the Web. These are listed in Abnexure-C.

2.5.1 Selection of Database

The thesis deals with networks for depth estimation. Hence, the development of the dataset is not considered. Synthetic datasets have higher accuracy but have poor correlation with real images due to their perfect light, color, and feature quality. Among real camera data, the most popular is the KITTI dataset [59] for outdoors and the NYU dataset [60] for indoors, which is used in many of the art papers. Popular datasets provide a comparison of performance benchmarks, both qualitatively and visually, with researchers working in the same domain. This work also uses the KITTI and NYU datasets. The KITTI dataset has semi-synthetic depth maps to match 64 scans of depth cloud points to 512 rows of RGB images. The dataset gives 86k images for training, 7k images for validation, and 1k test images. The NYU dataset also has semi-synthetic depth data to remove missing values due to noise from specular, low-albedo surfaces and shadows. Hence, pixel-level depth information is not available in both datasets.

2.5.2 Data splitting and augmentation

Most researchers split the dataset for training, validation, and testing, as proposed by [11]. This work also follows the same Monocular depth prediction requires an encoder-decoder architecture. The training requires pixel-level ground-truth information. Small differences between synthetic ground truth and real data may create prediction errors. The estimated depth is relative and needs some scale factor for true depth. The NYU dataset is used in the experiments to provide a generic comparison. Further, this dataset trained the same network earlier, so any improvement from the experiments will be an achievement. This work also follows the same practices in data augmentation to improve the generalization of outcomes. These included random mirroring and random color channel swapping of RGB images. Ref. [1] has also provided a separate 50K dataset, derived from the NYU Depth V2 dataset, after due correction with in-painting and filling in missing values. The study also uses this dataset for training.

2.6 Performance Analysis

The performance of depth algorithms is judged by improvement w.r.t. reference ground truth and achieved benchmarks by researchers. Absolute performance is computed w.r.t. actual measurements. However, in most cases, the work compares the achieved performances with those of other algorithms. This is so, as the algorithms aim to improve some

parameters instead of all possible ones, which is a daunting task. In such cases, there is a need to quantify the output results for the specific improvements and rank them. The ranking is possible with standard, widely accepted measures. In most algorithms, the absolute distance is not important, and the estimated depth maps have relative distances. The commonly used evaluation metrics used by most researchers are accuracy under a threshold, root mean square error, log root mean square error, absolute relative distance, and squared relative distance.

2.6.1 Quantitative Measures

One widely used measure in machine learning is the root mean square error (RMSE). It is mostly used for evaluating the prediction performance. This measure gives the average deviation between the predicted values and the actual values. It uses the Euclidean distance to measure the residual between the predicted value and the true value or reference value for each data point. The measure computes the mean of the norm of the residuals and then the square root of the result. RMSE requires true measurements and is hence suitable for supervised applications. Since RMSE is sensitive to the scale of the data, standardized data is necessary. Mathematically, the predicted value (Y_{pred}), actual value (Y), and N data points are represented as

$$RMSE = \sqrt{\frac{1}{N}\sum_{i \in N}(Y_i - Y_{pred_i})^2} \tag{2.4}$$

The Mean Absolute Relative Error (REL) evaluates the accuracy relative to the actual size of the measurement, independent of the scale of the true value. This is the mean of the ratio of the absolute difference error to the predicted value. This is given as

$$REL = \frac{1}{N}\sum_{i \in N}\frac{|Y_i - Y_{pred_i}|}{Y_{pred_i}} \tag{2.5}$$

The RMS $log - 10$ error is the mean of the absolute residuals of the logarithmic (base 10) predicted value and actual value. The logarithmic error (log_{10}) is

$$log_{10} = \frac{1}{N}\sum_{i \in N}|log_{10}(Y_i) - log_{10}(Y_{pred_i})| \tag{2.6}$$

A measure based on canonical distances proposed by [61] is popular. The prediction error is assumed to be linear; hence, the neighboring depths Y_i and Y_i+1 have a fixed ratio $\frac{Y_i+1}{Y_i}$

and are independent of the image scale. A point in the image set is classified as positive if the ground truth depth pixel is close to the scaled image depth pixel (estimated image). The ratio is near 1 when max($\frac{Y_{pred_i}}{Y_i}$,$\frac{Y_i}{Y_{pred_i}}$) < δ_{POS}. This pixel is used as a positive for the corresponding classes and a negative for all other classes. If the values are distant, the ratio is low, and max($\frac{Y_{pred_i}}{Y_i}$,$\frac{Y_i}{Y_{pred_i}}$) > δ_{NEG}. The linear relationship of scaling to canonical depth fails and this pixel is used as a negative. Here, training samples are positive if the error is lower than δ_{POS}=1.25 and negative if the error exceeds δ_{NEG}=2.5. The samples having errors between δ_{POS} and δ_{NEG} are ignored. Good performance indicators are high values of δ below the defined threshold and small values of RMS, log_{10} and REL. Authors [62] felt that the present monocular depth prediction benchmark relies on metrics for 2D, which leads to over-fitting and thereby is less helpful for 3D application performances. They infer that current metrics disregard the 3D domain to provide depth accuracy in the 2D plane. Further, some datasets are noisy (dark holes) and add unwanted bias during training and evaluation. They advocated the addition of 3D structural awareness by assessing the quality of the 3D geometry. They propose a 3D point Cloud F-score benchmark using the Dice coefficient. This coefficient is a harmonic combination of recall and precision, where recall is the proportion of true values in the input data that the algorithm correctly identifies and precision is the proportion of correct predictions. For a distance threshold t, the recall is

$$R(t) = \frac{1}{|G|} \sum_{g \in G} [e_{g \to R} < t] \tag{2.7}$$

Where e is the distance to a reconstructed point r from a ground truth point g and is $e_{g \to R} = min_{r \in R} \parallel g - r \parallel$. The precision is

$$P(t) = \frac{1}{|R|} \sum_{r \in R} [e_{r \to G} < t] \tag{2.8}$$

Where e is the distance to ground truth G from a reconstructed point $r \in R$ such that $e_{r \to G} = min_{g \in G} \parallel r - g \parallel$. The F-score is then

$$F(t) = 2.\frac{P(d).R(d)}{P(d) + R(d)} \tag{2.9}$$

They add the Jaccard Index for the 3D reconstruction benchmark to measure the volumetric quality as 3D Point Cloud Intersection Over Union (IoU), which is calculated with two input sets of non-voxelized point clouds. This is expressed in terms of recall and precision

as

$$I_oU(d) = \frac{P(d).R(d)}{P(d) + R(d) - P(d).R(d)} \tag{2.10}$$

Evaluation of the boundaries of the depth map, as proposed by [63] using average Chamfer distance (CD) in pixels w.r.t. the predicted boundaries and the ground truth boundaries. CD measures the distance between two point sets in 3D models and evaluates how well these points match each other. All these distances are averaged to produce a single scalar value. The chamfer distance between point sets, S and S_pred is given as

$$d_{CD}(S, S_{pred}) = \sum_{x \in S} min_{y \in S_{pred}} \parallel x - y \parallel_2^2 + \sum_{y \in S_{pred}} min_{x \in S} \parallel x - y \parallel_2^2 \tag{2.11}$$

Each point $x \in S$ finds its nearest neighbor in S_{pred} and vice versa. Distances from these pair-wise points are averaged to produce the shape-level distance. The thesis evaluates the trained models and compares their performances using a common platform that is based on several prior works [1, 11, 61, 64, 65]. These are

- Linear Root Mean Squared Error(RMS)
- Absolute relative error (REL)
- Logarithmic RMS error (log_{10})
- Threshold ($Y_i\%$ s.t. max($\frac{Y_{pred_i}}{Y_i}$,$\frac{Y_i}{Y_{pred_i}}$) = $\delta < th$ for th = 1.25, 1.25^2, 1.25^3)

Presently, the trend toward 3D object detection and autonomous driving is with single-image methods. Improvements in computational capability (GPUs, TPUs, memory), extensive datasets, and software tools and libraries like Matlab, Python, TensorFlow, Keras, PyTorch, CV, Numpy, etc. have enriched depth estimation methods through complex, dense CNN models and faster implementation. Advances like Transfer learning reduce resources and time by enabling pre-trained models to be trained with smaller datasets.

2.7 Applications

2.7.1 AR / VR and Entertainment:

Depth is essential for Augmented Reality and Virtual Reality tasks to sense a real 3D environment and reconstruct a virtual world. Google's Project Tango derives depth for virtual 3D reconstruction. Depth knowledge is also required for human-machine interactions to

accurately respond to user movement in games, gesture-based activities, highlight objects of importance, etc.

2.7.2 Security:

Face identification is a feature that unlocks applications based on the 3D reconstruction of a user's face. 2D-based face detection algorithms can be fooled by impersonators, as they cannot differentiate between a real 3D face and a 2D photo. Algorithms exist to identify a face and track it through a network of security cameras.

2.7.3 Robotics:

Depth information is required for localization, mapping, navigation, and avoiding collisions by avoiding obstacles. Advanced driver assistance systems (ADAS) safety, self-driven vehicles, and navigation systems for unmanned vehicles rely on 3D images to maneuver by gauging the environment from depth. Autonomous mobile robots (AMR) are used for the delivery of goods, pick and place, farming, law and order, cleaning, maintenance, etc. Big warehouses use 3D measurement for logistics, transportation, sorting, placing items, and inventory management. Similarly, environmental perception improves robotic navigation and machine efficiency. Industrial robots rely on depth to determine the position of objects and to pick them up.

2.7.4 Artificial Intelligence (AI):

Humans have the capability of dividing the whole picture into the sky, cars, buildings, and other multiple meaningful parts. In computer vision (CV), this is called semantic segmentation and is a powerful tool for image analysis. The depth complements the image analysis. Depth cameras are used in many embedded vision applications for automation. These have also penetrated non-engineering fields, and the applications are expanding. These are:

- Segment Registration of 3D Map
- 3D Shape Compression
- 3D Video Conferencing
- Digital reconstruction of Holographic Images

- Genetic Programming
- Measures with Image Focus
- Iterative Reconstruction
- Modular Stereo Vision
- Stereoscopic Vision
- Watermarking in Stereo Imaging
- Human body tracking or motion tracking for safety, security, etc.
- 3D scanning for medical, engineering, analysis, modeling, etc.
- 3D map reconstruction for 3D print, SLAM, remote visualization, navigation, etc.
- Counting people, features, facial anti-spoofing systems, etc.
- Traffic monitoring
- License plate recognition for safety, law, and order, etc.
- Precision gripping tasks (medical, high-tech industry)
- Remote patient monitoring
- Automated measurement methods
- Automated Optical Inspection system in manufacturing defect screening, etc.

2.8 Challenges

Depth cameras mimic human eyes by identifying shapes, colors, and movements. Stereo cameras have a low depth range and suffer from correspondence problems wherein points from the left-right view are matched. These cameras also have false positives, and supplementary cameras like TOF are used to enhance the accuracy. TOF cameras are expensive, consume more power, and presently have poor resolution. Further, the laser power of these cameras poses safety issues and is regulated by guidelines. It may be noted that both cameras perform poorly on glass surfaces or mirrors. This is due to the spectral bands used for imaging. UV bands can detect glass, but such cameras are rare. Single RGB/IR

camera-based depth cameras do not exist on the market but can be configured by adding a computation system to the camera. These cameras can evolve as simple and low-power depth cameras and have the advantages of contrast robustness, no magnification-based issues, and simple calibration. However, monocular passive cameras need crafted algorithms to compute depth, which calls for CV methods. In CV, solving the depth problem with a single image is nonlinear with multiple solutions. The problem is ambiguous, unlike the well-defined correspondence problem in stereo-vision. Modern machine learning methods successfully involve Convolutional Neural Networks (CNN) to address this ill-posed problem. Such networks train models with carefully designed loss functions and relevant large data sets consisting of color images and ground truth. The trained models estimate depth from the image with good accuracy. Training is crucial, as the CV algorithms will be productive if the learning features are well-defined and labeled. The other factors for prediction accuracy are the network architecture and the algorithmic loss functions. Presently, the computation is complex and resource-intensive. Light-weight innovative networks and good training can bring these to mobile phone platforms, which is also the present trend.

2.9 Summary

The human brain interprets visual patterns in scenes to recognize depth using physiological and psychological cues. Physiological depth cues are binocular and require both eyes for sensing. These binocular depth cues arise from retinal disparity, wherein some variation is registered by human eyes for the same scene. Monocular cues are psychological and can be sensed with one eye. Monocular cues are based on static objects or targets in motion. Many static cues, like occlusion, parallel lines, size, texture, color, and motion, aid in human depth estimation. Humans use all the above available cues and their learned prior knowledge to sense depth. In the machine world, the RGB camera mimics the human eye. The model of the passive camera is dependent on object depth, focal length, aperture, and image plane distance. These cameras lose depth information due to a 'circle of confusion' or blur for out-of-focus objects. Active cameras like TOF and structured light use a light source to illuminate the target and image it. These cameras suffer from limited depth range, cost, and higher power. A single camera is commonly available, is cheaper, and consumes low power. Computer vision (CV) uses these cameras to mimic human senses through stereo imaging or monocular depth cues. Adding depth information to recorded 2D RGB images gives back 3D RGBD images. Monocular vision-based depth estimation is challenging due to 'ill-poses'. Many methods of depth recovery are possible, each with

its advantages and disadvantages. Monocular methods are based on single images and are best solved with artificial intelligence (AI). CNN and DLN fall under AI. Networks consist of multiple different layers and blocks to estimate outputs. Training the network model involves minimizing the loss function through gradient descent, forward, and backward propagation. Loss functions define accuracy, efficiency, and outcome. Transfer learning reduces resources by using pre-trained models for a new task. Network training calls for the optimization of algorithms and learning rates for computational efficiency. Regularization methods overcome overfitting, underfitting, and vanishing activations. Many standard datasets are available. The NYU dataset is the most popular for indoor scenes. However, depth measurements are sparse and affected by surface quality, reflectivity, etc. In-painting methods are used to create semi-synthetic datasets. Evaluation of depth accuracy is based on standard performance metrics consisting of RMS, REL, logarithmic RMS, and threshold. New metrics for 3D cloud points are also discussed.

Chapter 3

Depth from Defocus

3.1 Introduction

Classical depth from defocus (DOD) uses defocus cues in the image by exploiting the limited depth of focus in a camera. Researchers have estimated depth with filters that measure derivatives concerning the aperture size and their ratio, the ratio of the difference in deblurred images and their sum, the use of orthogonal filters that characterize the relative defocus by modeling the defocus blur as a heat diffusion process, probabilistic models to capture monocular cues w.r.t. the relation between different parts of the image, sparse depth maps from defocus gradients at edge locations and matting, etc. The comparative results of popular depth estimation methods is summarized in Table 3.1.

The methods used are simpler and, therefore, better suited to limited computational resources. Here, cameras with shallow depth of focus acquire multiple images [4, 16, 19, 23, 25], and a defocus cue is used for building the depth map. Since the view remains the same, even with changes in focus or aperture, correspondence and gain issues as with stereo cameras are avoided. One can calibrate the camera and provide the camera blur or point spread function (PSF) as a prior, thus using a single image. Depth ambiguity due to objects placed before or after the focal plane is resolved by setting the focus to either the near field or the far field. A literature survey reveals that most DOD experiments cover short distances (indoors), except one where outdoor experiments cover distances up to 81 m. Table. 3.1 compares the popular methods.

Table 3.1: Comparative results of depth estimation methods

Method	Error	Features
Depth from defocus	<4% (Blayvas et. al.,J. Opt. Soc. Am. A, 2007)	• Robust to noise, occlusion and correspondence issues. • Single camera setup captures $\geq$2 defocused images. • Magnification is constant. However, the contrast varies. • Large aperture gives more defocus than small aperture. The optimal ratio of the apertures is 1.73. • Remove inherent blur and provide relative depth. • Very sensitive to changes in image intensity. • Degrade quality of image. • Error increases non-linearly with distance.
Depth from focus	<3% (Ens, J. E. ,1990, UBC)	• Robust to noise, occlusion and correspondence issues. • Needs >10 images with varying focus distance. • Images analyzed for best focus to estimate relative depth. • High computational cost, gives all-focused image. • Error increases non-linearly with distance.
Depth from coded aperture	<2% (Mina M. et. al., Ferdowsi University of Mashhad, Iran)	• Occluding patterns generate specific blurs of varying scales in spatial domain. • Deficiency of specific frequencies/phase as reference. • Blur depend on PSF amplitude and phase. • Sensitive to noise. Implementation is complex.
Depth from color coded aperture	<3% (Ivan et. al.,2016, Society for Imaging Science and Technology)	• Multiple colored apertures shift object on image plane. • This shift is a function of the distance. • Spectral filters code the shifted images for recovery. • Brightness constancy of the images need normalization. • Disparity map decreases with distance. • Single image, loss of spatial resolution • Method needs modification of cameras.
Plenoptic	<0.6%, (Hahne et al. Int J Comput Vis)	• Uses micro-lens array to project multiple images. • Analysis of best focus provides the depth. • Large loss of spatial resolution.

3.2 Aim

Theoretically, the depth estimation for long distances is poor due to the non-linear relationship between COC and distance. Therefore, there is a need to understand the performance of defocus methods for larger distances under daylight illumination to define problem areas for improvement.

Most classical DOD methods need more than 10 blured images to estimate depths through analysis and search of best focus. Additionally, these methods need a focused image for reference. The present work adapts a simpler approach which can estimate depth from only two arbitrary focused images without the need for a reference image. The reported error of 2.25% is also comparable to the alternative methods. The algorithms include intensity normalization, smoothing and Laplacian estimates which is computationally more efficient and faster then existing classical methods. The thesis proposes a median filtering step to denoise the images and a image segmentation step. These improvements allow estimation of full depth maps instead of sparse depth maps. Additionally, the MATLAB calibration tool was used to derive and calibrate the camera parameters using checkerboard images.

3.3 Implementation Method

The adapted method for this work uses two scenes [25], one focused on near objects and another on far objects, keeping other camera parameters constant. Here, a thin lens, a lossless paraxial system, and a planar scene are assumed, as shown in Fig. 3.1. The position of a point on an object is related to the position of its focused image by the lens law, where f_0 is the focal length, d_f is the distance from the focused object to the lens, and v_f is the distance of the focused image from the lens. When the camera is focused, other objects at various distances from the focused object are defocused and create blur circles (c) on the image plane. The depth from the blur circle is recovered by modeling the image as a third-order polynomial in the spatial domain for a fixed image plane v away from the actual focused image v_f.

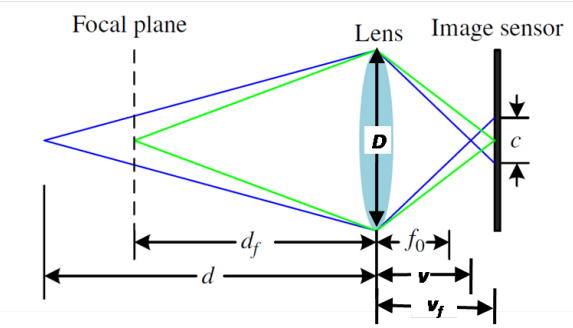


Figure 3.1: Schematic of the optical system. Here, f_o is the focal length, d, d_f is the distance of the lens to the objects, and v, v_f is the distance of the lens from the images.

The blur diameter is proportional to the distance between the objects. The radius R of

this blur is $c/2$, which for each defocused point in each image i, corresponding to the object point, is additionally dependent on the lens aperture diameter D. This is computed as:

$$R = \frac{Dv_f}{2}\left[\frac{1}{f_0} - \frac{1}{d} - \frac{1}{v_f}\right] \tag{3.1}$$

Here R can be either positive or negative depending on $v \geq v_f$ or $v < v_f$ depending on object distance d. The blur circle is modeled as a Gaussian point spread function (PSF), where the blur radius R is related to the spread σ of PSF and the object distance as:

$$\sigma = \frac{R}{\sqrt{2}} = \frac{Dv_f}{2\sqrt{2}}\left[\frac{1}{f_0} - \frac{1}{d} - \frac{1}{v_f}\right] \tag{3.2}$$

and for two images 1 and 2 with different focus (f_1, f_2) settings, give the ratio-metric relation:

$$\frac{\sigma_1}{\sigma_2} = \frac{R_1}{R_2} \tag{3.3}$$

Defocused image g is related to the focused image I as a convolution of PSF, h:

$$g_i = h_i * I_i \tag{3.4}$$

Fourier transforms G is the multiplication of optical transfer function H with image F. The S transform used by [25] is defined as:

$$S_H\{f(t)\}(\tau) \triangleq \sum_{k=0}^{N} \frac{(-1)^k}{k!} h_k \frac{d^k}{d\tau^k} f(\tau) \tag{3.5}$$

Where h_k is the k^{th} moment of $h(x, y)$. It is used as the basis for the deconvolution formula.

$$f_c(x, y) = f_b(x, y) - \frac{\sigma_h^2}{4}\nabla^2 f_b(x, y) \tag{3.6}$$

The above basis is used along with the laplacian operator and the standard deviation as the spread. Here f_c is replaced with I and f_b with g, giving:

$$I = g_1 - \frac{1}{4}\sigma_1^2\nabla^2 g_1 = g_2 - \frac{1}{4}\sigma_2^2\nabla^2 g_2 \tag{3.7}$$

In practice, the average of neighboring pixels reduces noise.

$$(\sigma_1^2 - \sigma_2^2)^2 = G^2 = 16\frac{\iint (g1 - g2)^2 dxdy}{\iint (\nabla^2)^2 dxdy} \tag{3.8}$$

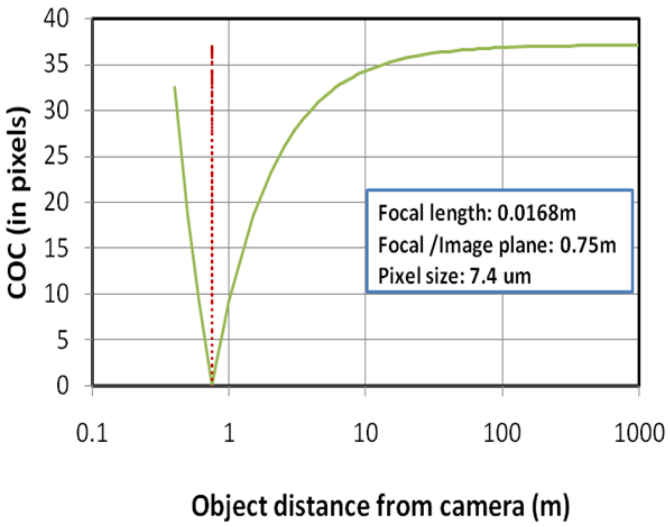


Figure 3.2: Plot of blur (COC) concerning object distance (Equation 3.1). Blur is at its minimum when a distant object is focused on the image plane (0.75 m).

Finally, the root is found in the integral and the blur circle:

$$\sigma_2 = \pm\sqrt{\left[\frac{G}{(\frac{R_1}{R_2})^2 - 1}\right]} \tag{3.9}$$

The distance u is computed from the relation of σ_2 and object distance from equation 3.2. The sign indicates the focus point before or after the image plane. Fig. 3.2 shows the theoretical plot of object distance versus blur diameter. It is seen that with increasing distance, the blur diameter decreases and then increases again. As the object distance reaches beyond a certain range, the blur reaches an upper limit. This distance and the distance with minimum defocus blur (at focus) define the camera range. As the interest is only in object depths further away from the focused objects, this ambiguity is avoided by keeping the objects of interest far away from the camera. In the plot, this distance is taken to be greater than 0.75 m and is marked by a red dotted line. The intensity levels of the captured image pattern also define the blur extent, which is presently not considered as the illumination variation for two outdoor scenes will be alike. In an image, the scaling is constant in terms of scene distance; hence, the ratio of the blur extents can be recovered by the blur-depth relation curve.

3.4 Experiments

The method for this work uses two scenes, one focused on near objects and another on far objects, keeping other camera parameters constant. The object's distances are greater than the focus distance to have one minimum and one maximum for all scales of blur (Fig. 3.2). The distance is computed from the relation of blur (equation 3.2). This DOD method is based on edge detection, wherein sharp contrasts provide the lowest blur. The experiments aim to understand underlying problem areas which are finally summarized.

Camera calibration provides intrinsic and extrinsic parameters. The camera parameters are required to obtain the depth (equation 3.1). A MATLAB calibration tool was used to derive the camera parameters and the depth. This tool requires the target camera to capture checkerboard images at various known distances. To verify the calibration accuracy, the experiment uses a SONY DSC-HX1 camera, as the camera parameters are known. The camera has an F-number of 2.8–5.2. The specified focal length ranges from 28 to 560 mm. A checkerboard pattern ($\approx$25 mm square) was used to get a set of 36 images from 700 to 3000 mm. The derived depth from the calibration tool was compared with the actual distance and tabulated in Table 3.2. The errors computed are due to the blurring

Table 3.2: Comparison of ground truth and computed distance

Sl. No.	Actual distance (mm)	Calibration tool depth(mm)	Error (%)
1	766.2	740	-1.47
2	1021.6	990	-1.13
3	1532.4	1490	-0.79
4	2298.6	2300	12.94
5	3064.8	3575	15.99

of the corners of the checkerboard patterns. The software uses the corner information for calibration. The acquired images show the ambiguity of several pixels, as shown in Fig. 3.3. The error increases with distance as the blur increases. The error is low for distances up to 1.5m. This camera has a small aperture, hence the blur is inherently higher.

After verification, a GigE Vision camera (Gev-CB-030GE) is calibrated. The camera comes with 12mm/F1.4 optics for imaging. The other parameters are not available as the optics were bought independently. The detector has 565 x 494 x 7.4 μm square pixels. The focus is set manually. The calibration tool is used with the GigE Vision camera images to get the camera parameters. The calibrated and actual distances are shown in Table 3.3. Here, the errors were within 3%. The camera parameters obtained were used in the depth

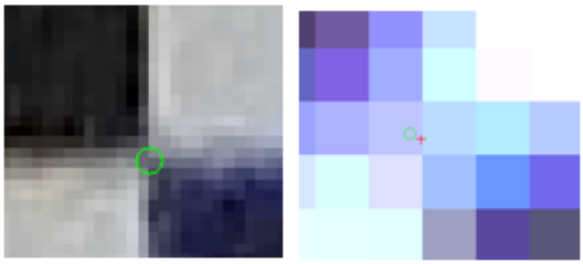


Figure 3.3: Zoomed corner points of the checkerboard. The points are erroneously identified due to image blur

recovery algorithm.

Table 3.3: Computed distance for GigE Camera

Sl. No.	Measured distance (mm)	Estimated distance(mm)	Error (%)
1	3600	3682.072	2.26
2	4800	4920.074	2.44
3	6000	6247.763	3.97
4	7200	7381.662	2.46
5	9600	9878.415	2.82

The depth recovery algorithm is developed with the MATLAB tool. Two images are obtained for near and far, as shown in Fig. 3.4. These images are denoised with a median filter. The algorithm computes the laplacian and the integral after blurring the images (equations 3.7, 3.8). The standard deviation is derived using equation 3.9, and the depth is obtained using equation 3.2. The sparse depth map is obtained at the feature edges in the image. The texture sections do not provide depth. To get a full-depth map, segmentation of alike objects was carried out by attributing the local average depth information to the segment. The segmentation is based on the mean shift algorithm.

3.5 Results and Analysis

The depth map and the estimated 3D image are shown in Fig. 3.5. The experiment was repeated for distant targets (Fig. 3.6). The results are shown in Fig. 3.7. The results show that depth can only be derived from (a) textures in the scene and (b) good scene contrast. This is expected as the PSF is obtained only at feature edges and is not recoverable in uniform areas. Low-contrast zones have higher noise and, hence, higher errors. Some scene features blend smoothly with the background and do not provide good depth estimation.

Figure 3.4: Far and near-focused images for input

The chimney shadow creates false depth. Since the depth estimation is based on relative defocus scales (equation 3.3), the method gives false depth for sharp black features on bright backgrounds. Such artifacts are not reported in any literature. The depth estimation method uses a ratio-metric relation of two differently focused images. The blur circle is inversely related to object distance 3.2) and increases rapidly for smaller distances. This results in good depth estimations for objects near the camera. However, for distant objects, the blur size increments at a slower rate leading to sub-pixel level changes. Hence the depth resolution is poor at distant objects. These are only detectable with large changes in distance. Further, increasing the focus distance v leads to focus at infinity, which hampers depth recovery. This is so, as both the images are perceived to be at near focus and the differences among the images are insignificant. On the other side, focusing very near to the camera increases blur rapidly but reduces object sharpness, which in turn reduces accuracy. Also, the depth range decreases as the change in blur becomes negligible within a short distance. In the experiments also, the far objects were blurred to a great extent, and depth recovery was not successful except at the skyline/ horizon. Here, as the object distance is inversely related to the F-number of optics (focal length/aperture diameter), this can be improved to some extent with a larger F-number of optics (> F 1.4). However, this has a bearing on size, weight, and cost.

3.6 Summary

In this chapter, DOD methods for depth estimation from the image were studied. The aim was to study the performance over large distances and understand the issues. A summary

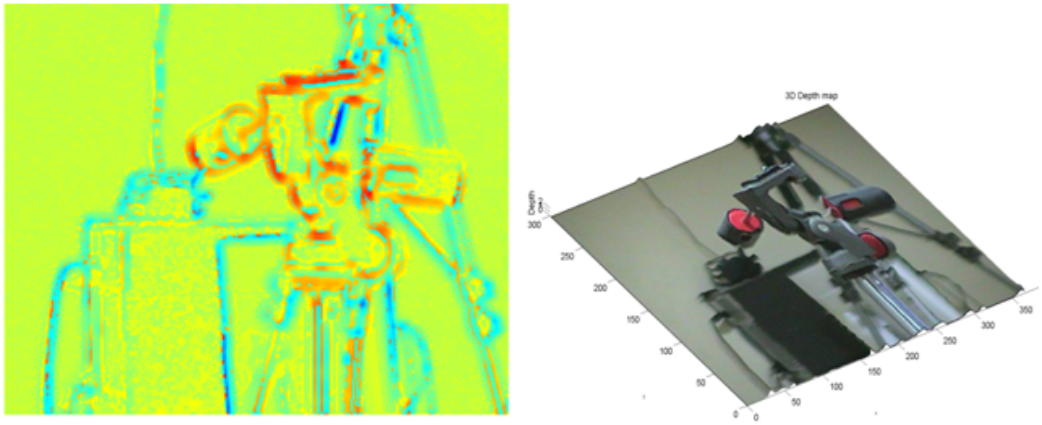


Figure 3.5: (a) Estimated color-coded depth map obtained with defocus. Small depths are red, while large depths are blue. (b) 3D image

Figure 3.6: Outdoor far and near focused images for input

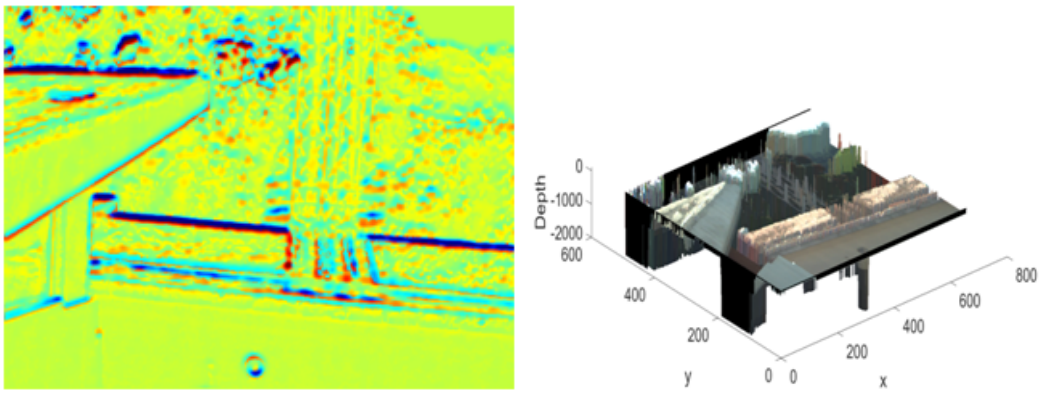


Figure 3.7: (a) Depth map: near objects are red-coded and far objects are blue; (b) 3D image: negative distances are away from the camera focus

of the conclusions is listed below.

1. Identified sensitivity to shadows, which introduce artifacts and errors. The blur created also reduces object sharpness, which in turn reduces accuracy. These are not reported in literature.

2. Low contrast leads to higher errors. DOD will work only during the day, as the cameras are passive.

3. The inherent blur created by apertures and optics limits far-range accuracy.

4. Depth resolution is poor for distant objects as the differences reach the sub-pixel range. Only large changes in distance are detectable. Inherent optical blur (MTF) submerge these finite small changes and limit detection of long depths. This method needs feature edges to detect blur scales. These features shrink with distance and eventually merge, creating uniform sections, which makes depth estimation erroneous. Further, the parametric value need to be tuned to capture large depths and hence is not arbitrary as thought. This mandates measurement iterations and large optics (larger F number).

5. The method provides higher resolution for short distances than for long distances. This is acceptable for most navigation, as near objects are a priority.

6. The DOD method is based on edge detection. These perform well for scenes with sharp and ample features. These methods may fail in featureless settings like a calm sea. Natural earth surfaces have powerful features along with shadows, while in the sea, waves create features. Depth can be derived in such cases. The featureless zones introduced an ill-posed problem. This was addressed by researchers using deep learning. [66].

7. In DOD, moving objects will create errors due to changes in the images, thus limiting the usage to only static environments. Alternatively, two cameras can be used. Here, one needs to address the correspondence and detector calibration aspects.

DOD avoids correspondence, scale and contrast issues faced by stereo methods. The method is strongly dependent on optics and focus positions. These change the mean image contrast, which increases estimation errors. Depth varies non-linearly with distance as shown in Figure 3.2. Here, the blur values reduce with distance. The depth resolution is influenced by the ratio of blur radius (R1, R2) detected from the two images (equation 3.9)

and large variations of blur are robustly detected. This method needs good feature edges to detect blur scales. Now, with increasing distance, the features in the scene become smaller and ultimately merge creating uniform zones which reduce blur values to sub-pixel range which is difficult to detect. The inherent lens blur further add to this error and limit robust detection of long distances. Further, this feature edge detection method leads to to only sparse depth maps. These aspects can be improved through better edge algorithms like Canny's method or the use of deep learning. DOD methods advocate shallow depth of field, which degrades image quality by blurring. These reduce the dynamic range of depth and lead to poor results for distant targets. Large optics (large F number) can improve upon these aspects but however compromises weight and size. It is also found during experiments, that the parametric values (near and far settings) need to be tuned to capture long depths and is not arbitrary as thought. This mandates iterative measurements for optimization of the depth of field thus making the real life situation complex. It may be noted here that for Depth from Defocus methods, moving objects will also create errors due to changes in the images with time, thus limiting the usage to only static environments. Depth from Defocus uses passive cameras and hence day light illumination is needed for outdoor applications. Artificial light consume high power and are generally avoided for long distances. DOD is useful for depth recovery from static images. If the objects move, depth recovery will not be possible for the object. The method needs a specialized setup to take two images at different focuses without any change in the scene environment. This method, hence, has limited use for predicting the depth of images in the wild. All the above points limit the usability of Depth from Defocus method for long distances.

Aperture-coded methods have the advantage of using a single image, which is an improvement over DOD.

Chapter 4

Coded Aperture for Depth Estimation

The coded aperture uses patterned, occluding spectral masks between the camera lens and the sensor. The camera model with such apertures creates specific blur patterns or spectral patterns in the image plane. A mathematical method of discriminating these patterns w.r.t. object distance and angle provides the all-in-focus image and related depth or light-field information. The aperture pattern is designed to maximize discrimination, and many types of apertures have been identified. The pattern detector usually matches a known PSF or blur.

4.1 AIM

The study aimed to verify the performance of Coded aperture methods to understand the scope of further improvements. Theoretically, coded apertures reduce light collection and behave as multiple small lenses. Hence, some light from an object will be focused by at least one sub-aperture. This affects the overall signal-to-noise ratio (SNR), or contrast, and the sharpness or quality of the image. Therefore, there is a need to understand the performance of this method to define problem areas for improvement. This chapter studies two methods: (a) spectral coding and (b) spatial coding.

4.2 Color-coded aperture

The color-coded aperture-based technique [24, 26, 27, 28] is based on two or more off-axis apertures, which, when out of focus, form two or more images displaced spatially. Using different color filters for each aperture, these images are formed with different viewpoints in different spectral bands. The color-coded dual aperture system works on redundant scene information within the bands. Here, as the scheme is based on spatial displacement

and spectral correlation, objects with poor spectral correlation show more errors. In such a case, one may use a green filter instead of a blue one after considering spectral leakage. The aperture can also be modified to include a third aperture [28] with a green band. The correspondence between the images provides a depth disparity map for the given scene. This is similar to the depth disparity map for stereo imaging. Fig. 4.1(a) shows the schematic of this method. The optics with an in-house aperture disc are shown in Fig. 4.1(b).

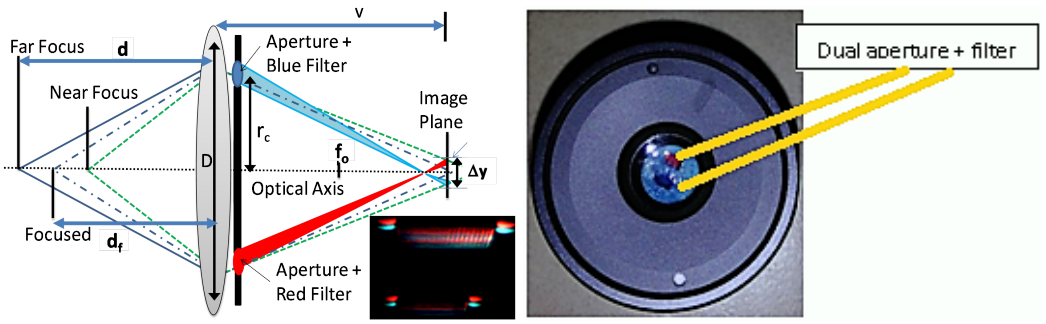


Figure 4.1: (a) Schematic of the optical system (b) Dual color aperture pair (c) Inset-stereo image pair from ceiling lamp

4.2.1 Implementation of DFCCA

The depth from a color-coded aperture (DFCCA) is modeled as a thin lens followed by dual offset apertures and an RGB detector. An object located at d_f and focused at image plane v or v_f is represented w.r.t. lens focal length f_0 as

$$v_f = \frac{f_0 d_f}{d_f - f_0} \tag{4.1}$$

However, for a defocused image, the image plane is not at distance v_f (Fig. 4.1). This leads to a blurred circle [28] at the image plane:

$$2R = \frac{f_0 D(d - d_f)}{d_f(d - f_0)} \tag{4.2}$$

where d is the defocused object distance. The image projection on the image plane depends on the location of the aperture. The aperture centers are separated by $2r_c$ from the optical axis $(c_x, c_y, 0)$. The shift of objects [28] Δy or Δx in the image plane w.r.t. object distance

d in terms of detector pixels size pix is related as

$$\triangle y = \frac{2r_c f_0(d_f - d)}{pix(d_f - f_0)d} \tag{4.3}$$

Fig. 4.2(a) shows the plot of blur circle with distance (equation 4.2). The blur increases to about 3 pixels. This reduces the sharpness of distant object edges. The camera is set to focus at 0.75m. Fig. 4.2(b) shows the theoretical plot of object distance versus separation. Initially, with increasing object distance along the z-axis, the separation (Δy or Δx) of object projection in the image plane decreases. A further increase in object distance leads to separation in the opposite direction.

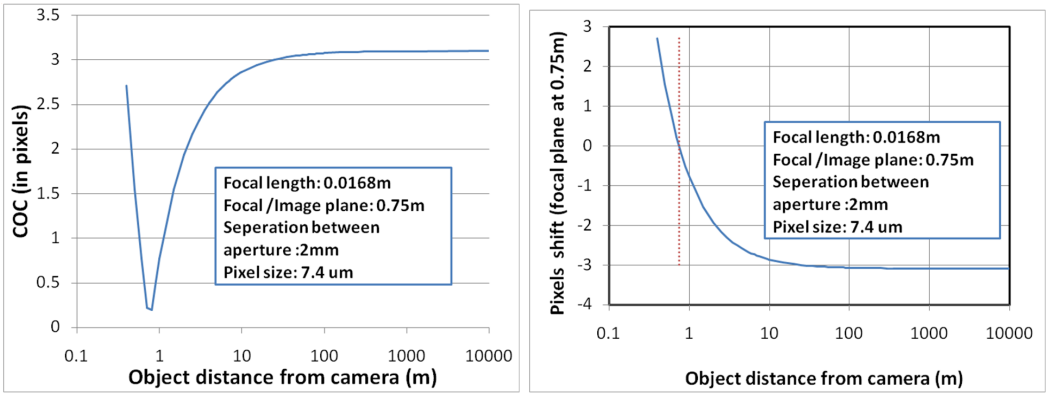


Figure 4.2: Theoretical plot of (a) Defocus with object distance (equation 4.2) (b) object distance versus separation (equation4.3)

4.2.2 Experiments

The depth range of coded apertures is influenced by the aperture size, spacing, and the sensor size. Better depth is achieved with a larger center-to-center distance between the apertures (equation 4.3). The effect is equivalent to a larger stereo baseline and is limited by sensor dimension and the lens diameter. This leads to a shift in image which is non-linear. The aperture size influences the image mean contrast and the blur (aperture PSF). Here smaller apertures are preferred. The aperture of the camera CB-030GE used was 6 mm. A margin of 1.5 mm is taken for assembly margin at the periphery. The balance 3 mm is used for the two apertures. Small apertures create less blur but are difficult to fabricate. Further, small aperture reduces the light collection which reduces image contrast calling for long exposure. An aperture size of 1 mm x 1.5 mm translated to a blur circle of 4.5

pixels in this work for a distance of 10 km. The center-to-center distance between the two apertures is then 2 mm. The details of aperture design are given in Table. 4.1.

Table 4.1: Aperture design details.

Design Parameters	**Values**
Aperture Diameter	6mm
Pixel dimension	7.4 μm square pixels
Sensor dimension in pixels	656 (h) x 494 (v)
Margin for assembly	1.5mm
Max blur due to aperture	<1 (Optical conjugate) to 5 pixels (10km)
Aperture size	1.5x 1 mm
Number of apertures	2, equally separated from the optical axis
Aperture pitch	2mm
Pixel shift	12.38 pixels at 10km
Color filters	Red and Blue

4.2.3 Results and Observations

Imaging with a double aperture was carried out by modifying the Gev-CB-030GE camera optics. A disc with two rectangular holes was inserted as a dual aperture. Red (R) and blue (B) filter paper were added to the openings (Fig. 4.1). In this experiment, available red and blue band filters were chosen to reduce intra-band spectral leakage. The aperture is placed between the lens and the sensor, in front of the original aperture. The original aperture is opened maximally.

The signal-to-noise ratio (SNR) in this method is inherently lower as the aperture is now smaller and the spectral band is narrower. This is compensated with longer exposure time. Focused images merge and do not have a spatial shift. Hence, defocused images and not-focused images provide depth. Color-coding the aperture simplifies depth extraction by digitally separating RGB colors. This color-coded method has lower computational complexity compared to binary-coded aperture methods, which do not use color coding.

The two spatially shifted images are obtained by digitally separating R, G, and B color images in MATLAB. Median filtering was used to remove impulse noise. A 5 x 5 block-based correlation algorithm generates the disparity map. Only a few researchers have published measured results for images taken inside the lab as most of them emphasized on new methods. This work compares the achievements with their results as in Table 4.2 with the SOTA range of disparity. Ours is for long-range outdoor images which has the highest disparity. The work also adds segmentation for full depth estimate. The mean shift algorithm

Table 4.2: Disparity range.

Methods	Disparity (Pixels)
Paramonov (2016) [27]	13
Lee (2013) [28]	14
Hahne et al. (2018)	8
Thesis (2017)	32

segregates the average depth for the features. Fig. 4.3 shows the histogram of RGB images. It is observed that the correlation between the B and G bands is better. However, the filters used are not ideal and have more spectral leakage in these bands. Fig. 4.4 shows the histogram of RGB images after the G band is subtracted from R and B images. It is observed that the intensity variations have reduced. The depth information is then computed through the stereo disparity algorithm after rectification and segmentation with a mean shift. Fig.

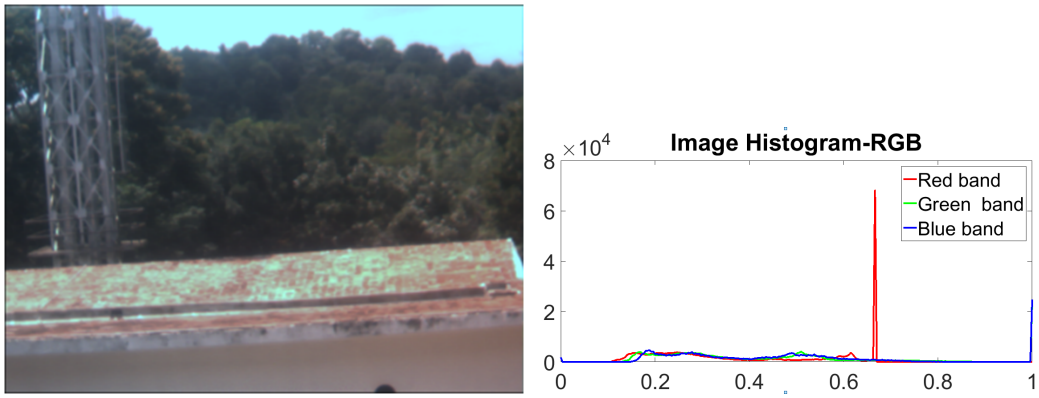


Figure 4.3: RGB image and histogram of RGB bands

Figure 4.4: Histogram of RGB bands of the image after G band subtraction from R and B

4.5(a-b) shows the disparity map and the 3D image, respectively, without green band subtraction. Here again, sharp features and high contrast are necessary for better disparity. A good correlation exists between the bright sky and vegetation, as shown in Fig. 4.5(a).

Smoother features at a similar distance do not show the same depth. Fig. 4.6(a) shows the red and blue band images after green band subtraction. The metal chimney structure is less prominent in the R band and is not fully detected in the depth map. Fig. 4.6(b-c) shows the disparity and depth map, respectively, with the green band subtracted. Smooth features at a similar distance show depth errors. This is more prominent on the building edges. The disparity map yields beneficial results, mostly due to lower spectral leakage. However, due to the large spectral gap, the correlation is low. This may be improved with a separate green band or with neighboring narrow spectral bands.

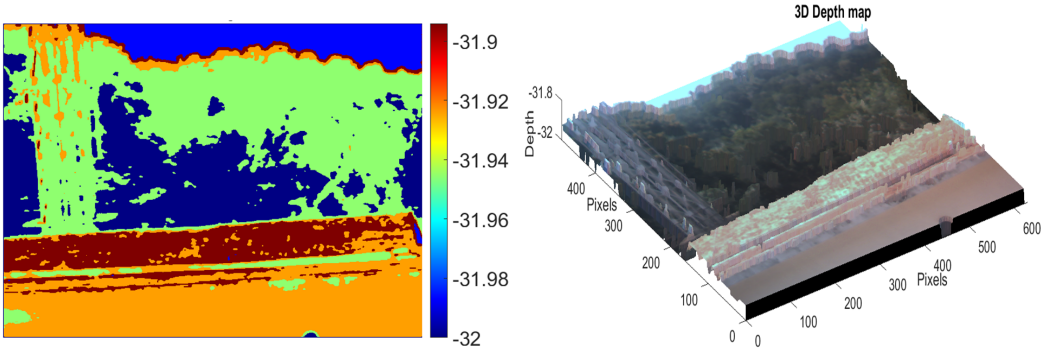


Figure 4.5: (a) Disparity map of B & R band (b) 3D depth map of B & R band

4.2.4 Discussions

In this chapter, DFCCA for depth estimation from images was studied. The main purpose was to understand the issues with such applications concerning the outdoor natural environment. Since it is a single camera and single image capture-based method, a simple calibration with known depths gives the ground truth. The experiments, however, skipped this calibration as the object distances were large and a calibrated distance meter (laser ranger, etc.) was not available. The theoretical performance is computed and found to be similar to DOD (compared in Fig. 4.2). DFCCA methods work with small optics. This is an advantage for mobile applications. The filter bands need to be optimized to cater to maximum redundancy. The disparity resolution is good for objects near the camera and degrades fast as the objects move away. Finally, the changes are at the sub-pixel level and are detectable only with large changes in distance. Accuracy can be improved with a larger separation between apertures (equation (4.3). However, this will lead to large optics. The effect is equivalent to a larger stereo baseline. The summaries of the conclusions are listed below. (1) DFCCA methods are based on edge detection. These perform well for scenes

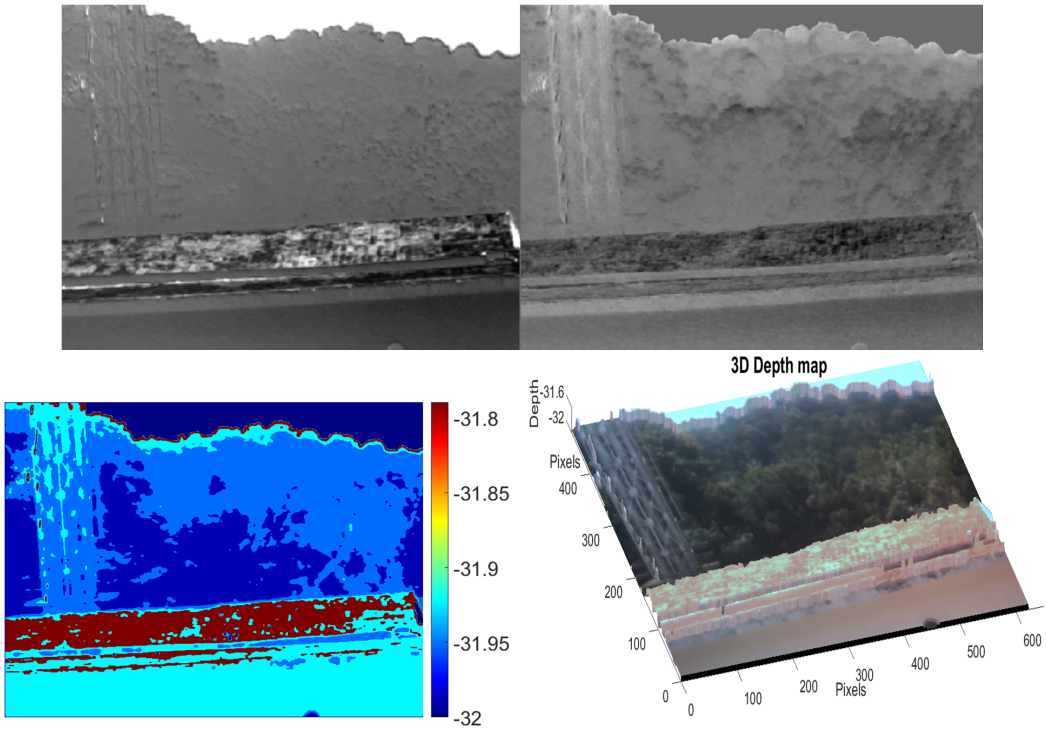


Figure 4.6: (a) Images of R and B band after G band subtraction (b) Disparity map of R & B band after G band subtraction (c) 3D depth map of R & G band

with sharp features. These methods cannot be used in featureless settings like a calm sea. Natural earth surfaces have powerful features along with shadows, while in the sea, waves create features. Depth can be derived in such cases.

1. DFCCA is used at the cost of image resolution.

2. DFCCA has the advantage of low computational complexity but inherently has a lower SNR and needs longer exposure time. This is not a major issue, as sensitive detectors can improve the system.

3. The choice of filters in the DFCCA method is critical, as redundant information is needed in both bands. Most natural scenes have energy in the red and green bands. Tuning filters within these ranges can provide better disparity maps.

4. The aperture shape also impacts depth recovery. [67] proposed aperture codes with higher discrimination scores.

4.3 Multi-coded Aperture

An image with a bigger aperture leads to higher defocus for distant objects, and this is exploited in the multi-coded aperture method (DMCA) to estimate depth. Further, aperture methods can significantly improve depth estimation by enhancing the relative defocus estimations [68]. Researchers used multiple coded apertures [67, 68] to recover all focused images using Wiener deconvolution. [68] proposed a pair of optimum complementary coded apertures w.r.t. noise for defocus deblurring and depth recovery without much loss of high frequencies. He used a robust prior based on natural images, a synthetic depth map as ground truth, recovered sharp images through modified Wiener deconvolution and estimated depth using the L2 norm using a range of PSFs. He uses an N×N binary aperture code C, where $C \in 0, 1$(N×N). Optimized apertures have features like (a) minimum number of apertures, (b) minimum size and distance of apertures, and (c) good decoding of images corresponding to each aperture with minimum spatial or spectral overlap. In the Fourier domain, the frequencies are designed to never overlap from each aperture by selective filtering.

Coded apertures are geometrically altered with occluding patterns to generate defined blur shapes. These provide zero-crossing frequencies (in power spectral domain) to enable depth estimation. Single coded apertures have zero crossings which degrade the image quality due to loss from occluding patterns. Using two different apertures overcomes this limit. These adequately compensate the losses by complimenting those frequencies lost due to the other aperture pattern. Since both share the same power spectra, the overall image quality is retained with wide frequency content. The geometrical shape, size and position of the aperture pair are tuned to obtain aperture shapes that have orthogonal magnitude and phase. Here, two off-centered circular Gaussian apertures were optimized by setting a radius ratio of 1.5. This also improves noise immunity and robustness to weak textured surfaces.

4.3.1 Implementation Method

The model of defocused image g is the convolution of a sharply focused image i, a blur function having a PSF of h and noise n. The n in this model is Gaussian white noise with a zero mean and a standard deviation of σ. The PSF is dependent on the aperture a which relates to the estimated depth.

$$g = i \otimes h(a) + n \tag{4.4}$$

As the defocus blur is due to the loss of certain frequencies, these are best analyzed in the Fourier domain.

$$G_i = I_o.H_i + \xi_i \tag{4.5}$$

Where, in the image ii, the discrete Fourier transforms (DFT) of i_o, h, and n are I_o, H_i, and ξ_i, respectively. The aim is to find a PSF that estimates a focused image $\hat{I}_0$ such that it minimizes the mean square error w.r.t. I_o as

$$\hat{I}_0 = G.H' \tag{4.6}$$

and the error is

$$\in = \mathbb{E}|I_0 - \hat{I}_0|^2 \tag{4.7}$$

Since the model with the Gaussian function has logarithmic energy, the above problem requires minimization of

$$E(d|KdE(d|H_i, \sigma) = [min_{I_o} \sum_i ||\hat{I}_0.H_i^d - G_i||^2 + ||C\hat{I}_o||^2 \tag{4.8}$$

where H_i^d is the Fourier transform of PSF at depth d of the i^{th} image and $||C\hat{I}_o||^2$ is the regularization term. Solving $\delta E/\delta \hat{I}_o = 0$ yields the Wiener deconvolution filter.

$$\hat{I}_0 = \frac{\sum_i G_i.\acute{H}_i^d}{\sum_i |H^d|^2 + |C|^2} \tag{4.9}$$

Where the complex conjugate of H is $\acute{H}$ and $|C|^2$ is the optimal matrix of inverse signal-to-noise ratios. The unknown I_o and noise ξ relate to the estimated distance d. Since ξ is a random matrix, the expectation of the L_2 distance between the ground truth I_o and $\hat{I}_o$ w.r.t. ξ gives

$$R(H, I_o, C) = E_\xi||\hat{I}_o - I_o||^2 = E_\xi||\frac{\xi.H - I_o.|C|^2}{|H|^2 + |C|^2}||^2 \tag{4.10}$$

The white noise ξ is represented as N($N(0; \sigma^2)$), so

$$R(H, I_o, C) = E_\xi||\frac{\sigma.\acute{H}}{|H|^2 + |C|^2}||^2 + E_\xi||\frac{I_o.|C|^2}{|H|^2 + |C|^2}||^2 \tag{4.11}$$

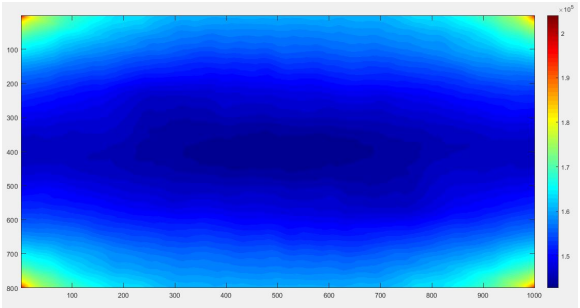


Figure 4.7: Image prior: A

All images have a certain distribution, and an I_o sampled from such images will provide a C that minimizes the expectation of R w.r.t. I_o as

$$|C|^2 = \sigma^2/A \tag{4.12}$$

Finally

$$\hat{I}_0 = \frac{\sum_i G_i.\acute{H}_i^d}{\sum_i |H^d|^2 + \frac{\sigma^2}{A}} \tag{4.13}$$

A is the average power spectrum obtained from several natural images. A variety of seven images were taken, and A was computed. Fig. 4.7 shows the 3D plot of A. The noise level ξ is determined from the camera, and hence this gives a parameter-free variant of the Wiener deconvolution algorithm. This generalized Wiener deconvolution algorithm recovers the all-focused image $\hat{I}_o$. The depth is estimated from this all-focused image. The $\hat{I}_o$ is computed for each sampled depth value $d \in D$, to reconstruct two focused images using a 2D inverse Fourier transform (IFFT). The residual W^d for each pixel is the difference between the reconstructed images and the observed images, which gives an error of d w.r.t. the actual depth $d*$.

$$W^d = \sum_{i=1;2} |IFFT(\hat{I}_o^d * H_i^d - G_i)| \tag{4.14}$$

The minimum of $W^d(x, y)$ gives d for the pixel (x, y), which equals the real depth. The estimated depth map Z is

$$Z(x, y) = argmin_{d \in D} W^d(x, y) \tag{4.15}$$

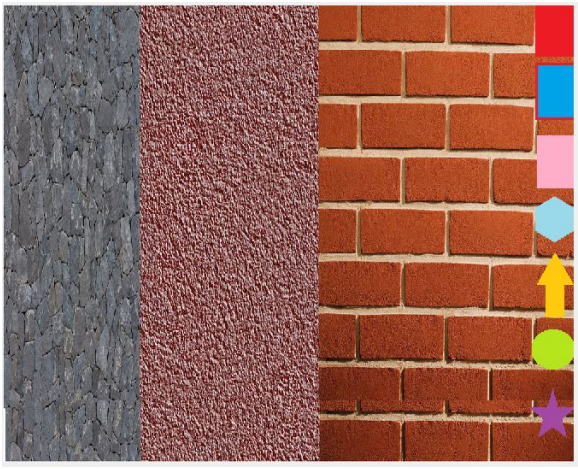
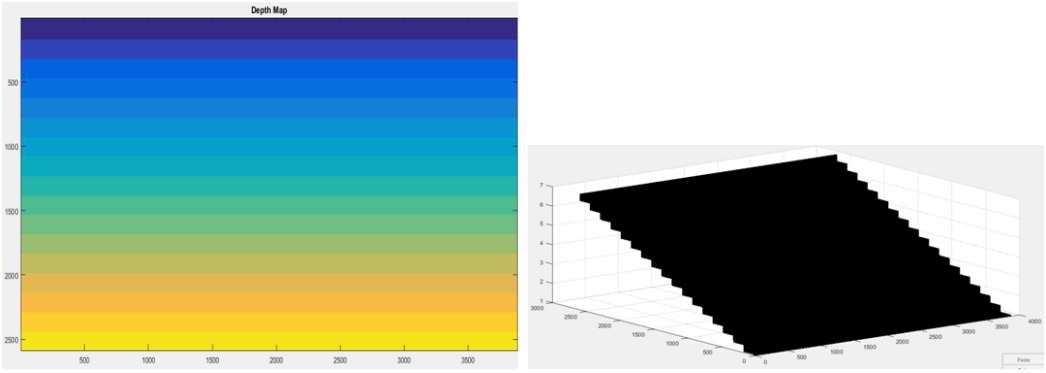


Figure 4.8: Image with dense features

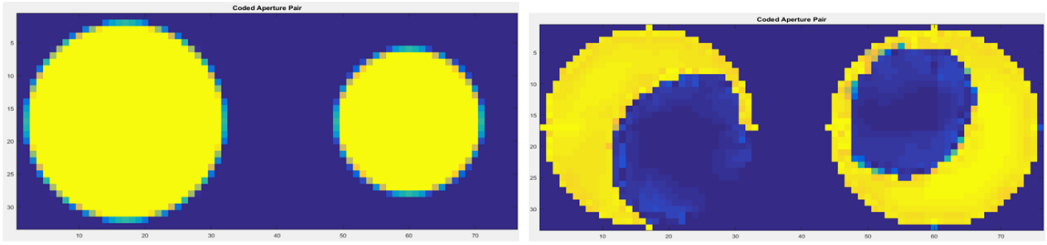


(a) 2D Depth Map **(b)** 3D view of depth

Figure 4.9: Synthetic depthmap

4.3.2 Experiments

This study experiments with his method to understand the performance of depth estimation. The method requires modification of the camera aperture, which requires skill. Hence, the experiments were done with synthesized images and apertures. The ground truth is generally arrived at by calibrating the camera blur (PSF) with distance. Here, synthetic data is generated for the study. A synthetically focused image is created, with some zones having a lot of features and some zones having a smooth texture, as in Fig. 4.8. An artificial depth map is generated in Matlab. This is a staircase depth map with about 17 levels. Blue is the nearest. This map is used as ground truth and is shown in Fig. 4.9. Two pairs of apertures are generated artificially. The first pair consists of circular apertures (Fig. 4.10a). The second pair is from [68], as in Fig. 4.10b. These pairs operate as filter kernels of



(a) Circular aperture pair **(b)** Zhou's optimized aperture pair

Figure 4.10: Aperture pairs for depth estimation

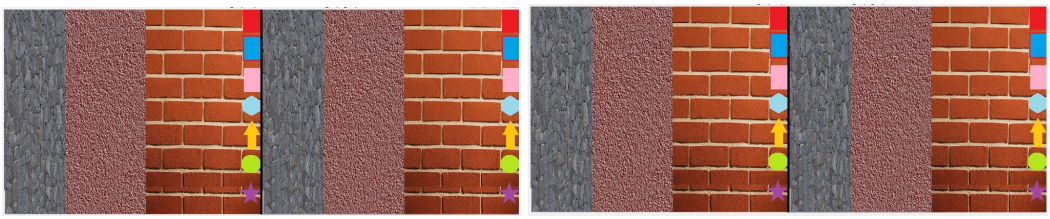
(a) Defocused images-circular apertures **(b)** Defocused images-Zhou's apertures

Figure 4.11: Defocused image pair with different apertures

dimension 33 by 33 pixels. The dimensions are claimed to produce less diffraction due to the smoother pattern. The proposed optimal ratio of the large to small circular aperture radius is 1:5. This is maintained in the work.

4.3.3 Results

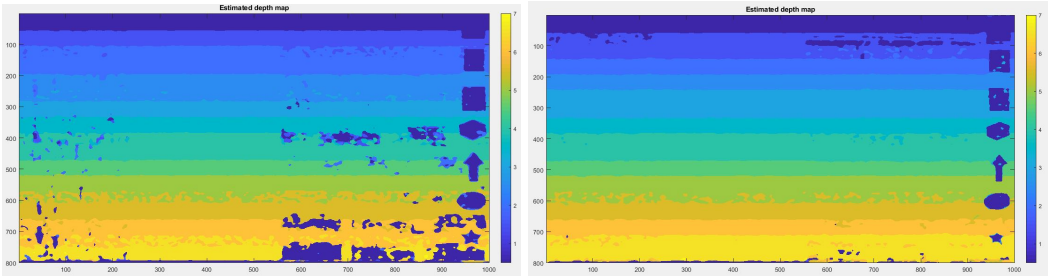
Using the aperture pairs, the depth map, and the synthetic image, defocused images were synthesized in the Fourier domain by applying the PSF and aperture kernels. In the spectral domain, the PSF was scaled as per the depth map. Both PSF and aperture kernel spectra were multiplied with image spectra. Finally, a noise sigma of 0.003 was added. The images were converted to the spatial domain by applying IIFT. The defocused image pair with circular aperture pair is shown in Fig. 4.11a. With different apertures, the contrast has changed. The defocused image pair based on the [68] aperture pair is shown in Fig. 4.11b. Here, the contrast is similar in both images. The convolution of these deblurred patches with different scales of blur kernels, along with the computation of the L_2 norm concerning the input-defocused patch, gives insight into the correct blur kernel. The blur scale for the least L_2 error is the final correct blur size. Both the circular and optimized aperture-based images show similar recovery. The recovered images after Wiener deconvolution are shown in Figs. 4.12a and 4.12b. Fitting a 3rd-order polynomial curve with a sequence of



(a) Circular aperture: original image (L), recovered image (R).

(b) Optimized aperture: original image (L), recovered image (R).

Figure 4.12: All in focus images after deconvolution



(a) Circular aperture-depth map

(b) Optimized aperture-depth map

Figure 4.13: Estimated depth map from different aperture pairs

residuals from the above-sampled depth values improves the depth resolution at a location (x,y). This interpolation gives a continuous depth estimate from the minima of the curve. These depth maps are shown in Figs. 4.13a and 4.13b.

4.3.4 Observations

It is seen that the depth estimates from images using optimized apertures have fewer errors. In all images, depth estimation fails at smooth surfaces, which were implemented through various shapes. The estimation is best in all images where textures are dense. The middle texture has dense features, and the estimation is with a minimum error. This reveals that this method is strongly influenced by features. The optimized aperture does perform better but reduces the light collection, as shown in Fig. 4.11b.

4.4 Summary

Two methods of aperture-based depth maps were studied. The color-coding method was sensitive to spectral leakage and image-to-image contrast. The filters used for the DFCCA method are therefore critical for preventing spectral leakage. DFCCA has the advantage

of low computational complexity but inherently reduces the energy per aperture due to the spectral band limitation of the filter used and also due to the smaller aperture. This low SNR can be overcome with a larger lens or with a longer exposure time. Alternatively, new sensitive detectors can improve the system. DFCCA also reduces the image resolution. Here, larger sensors can offset this disadvantage. DMCA uses specially shaped fixed aperture pairs used in sequence to capture images, while DFCCA uses a fixed aperture to capture shifted spectral images. The post-processing is slightly more complex than DFCCA. Moving objects for DMCA using a single camera is a disadvantage, as these will introduce errors. Using two cameras with different apertures can solve the problem but will call for correspondence and scale issues. DFCCA is sensitive to the light spectrum and will perform poorly at sunset or sunrise when the spectral content of light is redder. Both methods are based on edge detection and perform poorly for featureless scenes like a calm sea, wall, or road. However, most natural earth surfaces have powerful features. Both require camera aperture modifications and work only with an external light source. As these methods are based on the blurring of objects with distances, they provide higher resolution for short distances than for long distances. The implementation is complex, costly, and requires skill for both methods. These methods, therefore, have limited use for predicting the depth of images in the wild.

Blur cues and single images can also predict depth without any modification of the camera and are not sensitive to the daylight spectrum. This is studied in detail.

Chapter 5

Calibration and Absolute Depth

5.1 Introduction

Camera calibration is multi-fold and includes geometric calibration, radiometric calibration, and spatial frequency response. The geometric calibration estimates lens parameters and the relationship of the 2D/1D image to the corresponding 3D objects. These parameters are more useful in machine vision for measurements and locomotion. The two popular methods of geometric calibration are based on Tsai's method [69], which uses complex precision 3D objects, and Zhang's method [70], which uses simple planar grids. Zhang models the camera with lens distortions, which is more suitable for real cameras. The 3-D world points and their corresponding 2-D image points are obtained from multiple images and poses of a calibration target. The calibration target has feature points like the corners of the checker box pattern, which are detected in the images. The two camera parameters of interest are (1) intrinsic parameters like axis skew, principal point, focal length, and scale factor, and (2) extrinsic parameters like rotation and translation. The lens's radial and tangential distortions are also taken into account. The scaled image projected from world objects is related to:

$$s \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = K \begin{bmatrix} R & | & T \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{5.1}$$

Where s is the scale factor, x, y are the image positions (pixels), K represents the intrinsic parameters and lens distortions, R (rotation) and T (translation) are the extrinsic parameters, and the world coordinates are X_w, Y_w, Z_w. For co-planar grids, $Z_w = 0$. A direct linear transform (DLT) algorithm is used to solve the above equation using homography

projection h_ij parameters using known world coordinates and target size.

$$s \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_1 1 h_1 2 h_1 3 \\ h_2 1 h_2 2 h_2 3 \\ h_3 1 h_3 2 h_3 3 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix} \tag{5.2}$$

The re-position errors obtained are then minimized using the Levenberg-Marquardt optimization method. Many toolboxes are available that implement this method for camera calibration. Other calibration coefficients deal with camera response. Radiometric calibration represents pixel-wise measurements of incoming light. The response of pixels usually varies with neighboring pixels, giving rise to photo-response non-uniformity (PRNU). A calibrated, uniform light source with different intensities provides each pixel with various responses. These are curve-fitted (usually linear) to get a pixel-wise coefficient matrix. This correction avoids noise in features due to variations in surrounding pixel outputs. The response of the camera to spatial contrast variations (neighboring pixels) is the spatial frequency response (SFR). This response is dictated primarily by the lens and the sensor pixel size and is related to the image's sharpness. The SFR is measured by a target with varying widths of black and white bar pairs (or line pairs). The response to finer and finer line pairs trails off and finally is not distinguishable. All the above calibration methods used focused images and a camera model (Fig. 2.1). These parameters are usually calibrated by the manufacturer, and corrections are applied as part of processing inside the camera before image acquisition. Camera properties also include defocus blur (Fig. 2.2), which is dependent on aperture diameter and focus. Larger apertures increase this blur and have a shallower depth of focus. This property is exploited by DOD methods. For Lambertian surfaces, various non-focused random objects in the camera field of view cause spatially varying blur points in the image plane. These can reduce overall image sharpness. Mathematically, the blur is modeled as the convolution of a sharply focused image with a Gaussian filter kernel. Present calibration methods do not address out-of-focus blur size w.r.t. object distance. This aspect is important in this work as this spatial blur information forms a 2D depth map. Further, single image-based depth estimation lacks geometric reference to absolute distance and so provides only relative depth maps. Since the relationships are non-linear, camera-centric references need to be worked out. Such calibrations require custom calibration targets and methods.

5.2 AIM

A calibration method to retrieve the absolute distance from relative depths is studied. As most depth estimation methods, like DOD, coded aperture, etc., rely on blur in the image, the work is predominantly based on blur. This work derives a method of calibration for blurred image targets. The theoretical blur and the ground truth are utilized to recover the absolute depth. The coefficients computed once can be used for all images. Here, a relative depth map from a single image (target) is used as the input for calibration.

5.3 Implementation Method

There are many methods for relative depth estimation. The method from [31] to estimate relative depth from a single image is adapted for this work. Here, initially, an edge detector extracts an edge map from the image. The focused object points provide sharp and fine edges w.r.t. defocused points. The unit step function $u(x)$ at $x = 0$ represents the focused edge in the 1D domain as

$$f(x) = Au(x) + B, \tag{5.3}$$

Where A represents the edge intensity or amplitude, while the edge offset from the origin is B. A Gaussian blur filter, when applied to these edges, degrades the sharpness and introduces a gradient on the step function. In the Fourier domain, this filter has a point spread function (PSF) with a kernel $h(x, \sigma)$. The standard deviation σ of PSF changes the blur scale. The step with the gradient in image $g(x)$ is related to the original step in the sharp image $i(x)$ as

$$g(x) = i(x) \otimes h(x, \sigma) \tag{5.4}$$

After filtering, the sharp edges undergo higher blurring than the existing blurred edges in the image. The filter changes the edge gradient, which leads to a gradual change of intensity (logarithmic) among a bulk of neighboring pixels, creating a slope similar to one side of the Gaussian curve. Mathematically, in the Fourier domain, this gradient on the x-axis is:

$$\nabla G_1(x) = \nabla(I(x) \otimes h(x, \sigma_0)) = \nabla((Au(x) + B) \otimes h(x, \sigma_0)) \tag{5.5}$$

$$= \frac{A}{\sqrt{2\pi(\sigma^2 + \sigma_0^2)}} exp(-\frac{x^2}{2(\sigma^2 + \sigma_0^2)}), \tag{5.6}$$

Where A is the intensity, σ is the standard deviation of the original edge in the image, and σ_0 is the 0^{th} scale standard deviation of the blurring Gaussian kernel. The equation is simplified by reducing the dependency of the gradient ∇G on A. The ratio (R) of the original gradient and the post-filtered gradient gives:

$$\frac{|\nabla G(x)|}{|\nabla G_1(x)|} = \sqrt{\frac{\sigma^2 + \sigma_0^2}{\sigma^2}} exp(-(\frac{x^2}{2\sigma^2} - \frac{x^2}{2(\sigma^2 + \sigma_0^2)})) \tag{5.7}$$

The edge location is obtained with the derivative of dR/dx at $x = 0$. The R is the maximum at this location and is

$$R = \frac{|\nabla G(0)|}{|\nabla G_1(0)|} = \sqrt{\frac{\sigma^2 + \sigma_0^2}{\sigma^2}} \tag{5.8}$$

The equation is now independent of A and relates R with σ, the edge blurs in the original image. Rewriting equation 5.8 w.r.t. blur gives

$$\sigma = \frac{1}{\sqrt{R^2 - 1}} \sigma_0. \tag{5.9}$$

In the same way, if the edges are blurred with two scales of σ (σ_1 and σ_2), the original blur is related to these scales as

$$\sigma = \sqrt{\frac{\sigma_1^2 - R^2\sigma_2^2}{R^2 - 1}} \tag{5.10}$$

Computing the blur for both axes leads to a 2D blur map. The gradient magnitudes ∇G_x and ∇G_y are computed along the x and y axes from the edge map using 2D isotropic Gaussian kernels as

$$||\nabla G(x, y)|| = \sqrt{\nabla G_x^2 + \nabla G_y^2}, \tag{5.11}$$

The gradient magnitude is a function of blur and is high for sharp edges. The method requires an initial edge map obtained with an edge detector. Though there are many edge detectors, the Canny edge detector is widely used because it is very robust. σ changes with c and is related as $\sigma = kc$ (equation 2.1). Since other camera lens parameter settings like f_o, N, and d_f are known and remain the same for an image, knowing k can give the depth estimate. If the values are unknown for an image from the wild, this can be derived using available application software, like the camera calibration application in Matlab. The value of k should ideally be unity. However, during experimentation, it was found that this parameter is dependent on physical camera opening shape and size, fabrication imperfections, lens abrasions, and the finite size of sensor pixels, which creates a small, non-avoidable blurring of the image. These were not accounted for in the thin-lens

model used. Hence, calibration is necessary to get the coefficients of k, which will reliably provide the depth information. The depth information so obtained defines a sparse depth map, which has depth data only at feature edges. To get a full-depth map, some technique of segmentation based on holistic features using this depth information is required. Author [20] proposed an alpha-matte method to interpolate a full-depth map. This work deals with methods of calibration to get the best value of k and information interpolation to get a full-depth map.

5.4 Calibration

This work proposes new targets to calibrate the values of k and recover absolute depth using a single image. One target is made of multiple captured ground truth targets obtained through experiments. The captured data includes a known depth of images. A second synthetic target contains various patterns blurred by different scales of PSF kernels. To obtain the gradient magnitude and position of the blur, the first and second derivatives of both the original and blurred targets are computed. The ratio of the absolute magnitudes provides depth information. Fig.5.1 summarizes the proposed method. The method used has advantages like immunity to spectral sensitivity, contrast variations, and magnification issues. This method using a special target for depth calibration is unique and not found in related research domains. Further, the work contributes to the research community with a hybrid method that relates theoretical blur with real blur and absolute depth. The following sections provide the details of this work.

5.4.1 Choice of PSF

The PSF is a measure of camera and image quality, and a perfect PSF represents a 2D impulse function. However, imperfections in real-world fabricated systems lead to blurred edges even for focused images, which leads to a corresponding PSF with finite diameters. Sharp edges will have PSF with small diameters. Further, the PSF can be a flat-top 'pill box' (Fig. 5.2 a) or a Gaussian type (Fig. 5.2 b). The 2D Gaussian PSF is more popular as most natural systems resemble Gaussian probability. In this work, both PSFs were used. Since this work focuses more on the outcome of results, the section descriptions will be with a 'pill box' PSF, while the results will have the outcome of both PSFs. For simplicity,

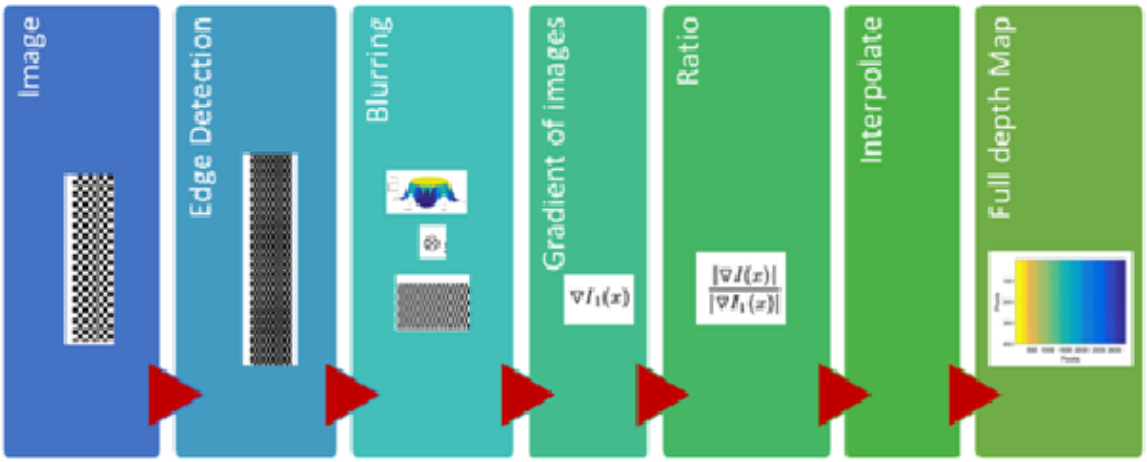
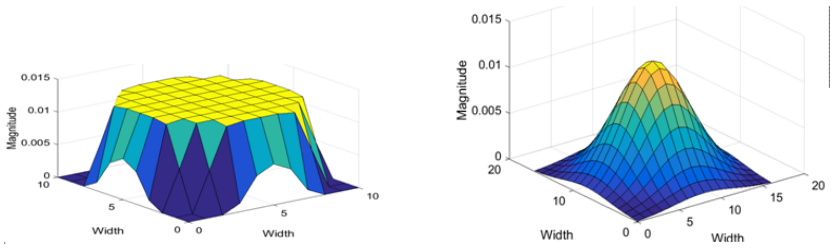


Figure 5.1: Proposed method of blur estimation using a special target



(a) Sample pill-box kernel **(b)** Sample Gaussian kernel

Figure 5.2: PSF kernels

the pill box PSF in 1D is

$$h_p(x) = \frac{1}{2r}[u(x+r) - u(x-r)] \tag{5.12}$$

The Gaussian PSF (1D) is

$$h_g(x) = \frac{1}{\sqrt{2\pi\sigma}} exp[-\frac{1}{2}(x-\bar{x})^2] \tag{5.13}$$

Image G is modeled as

$$G_k(x,y) = i(x,y) \otimes h_p(x,y,r_k) \tag{5.14}$$

and

$$G_k(x,y) = i(x,y) \otimes h_g(x,y,\sigma_k) \tag{5.15}$$

5.4.2 Target and Result

A synthetic target with 400x3300 pixels was developed. This target contains blocks of checkerboard patterns, each blurred with a different known scale of PSF. The scales are implemented with various PSF radii (r). The work used 10 blocks, and hence the target provides a range of 10. The size and the scale range are arbitrarily fixed and can be varied. The target Fig.5.1 is used as the input image, as in Fig.5.3, and the blur (σ) map is derived. This map, as shown in Fig.5.4, has a range of gradient magnitudes, with dark blue as the lowest. A scale of 10 was found to be outside the bounds of the blur detector. Therefore, the range was experimentally tuned before use. The checkerboard pattern images are captured at different known calibrated depths for ground truth. Here, the camera focus matches the pattern at the smallest depth and is kept constant for all measurements. The captured images have higher defocus at larger depths, which increases the blur. These images are cropped and stitched together to provide a single image with various real blur ranges w.r.t. known depths (Fig.5.10). These two targets provide synthetic blur and real blur for computation of k and relate the blurs to known depths.

5.4.3 Experiments

The synthetic blur map shown in Fig. 5.4 has a blur range of 10, with the lowest value indicated by dark blue. The colors seem to repeat for incremented scales on the map, indicating that the values are not monotonic. On analysis, it is found that on the scale of PSF at r = 7 and above, the blur is very high and the pattern edges merge with the

Figure 5.3: The proposed synthetic target of 400x3300 pixels contains 10 blocks of 400x330 pixels. Each block is a checkerboard pattern blurred incrementally with a PSF filter having scales from 1 to 10. The figure shows only 2 of the 10 blocks for clarity.

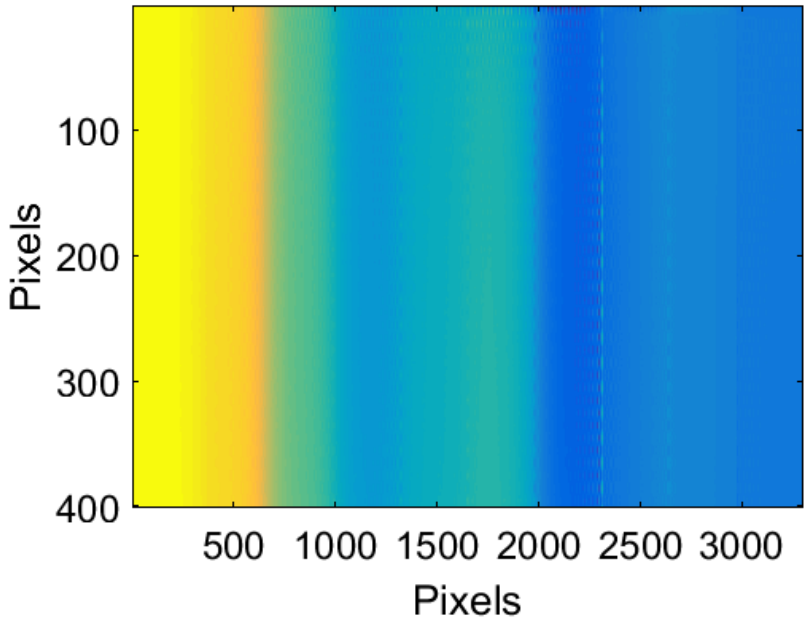


Figure 5.4: The synthetic blur map with a PSF scale of 1 to 10 obtained using equation 5.15. A block with the same PSF creates steps in the map.

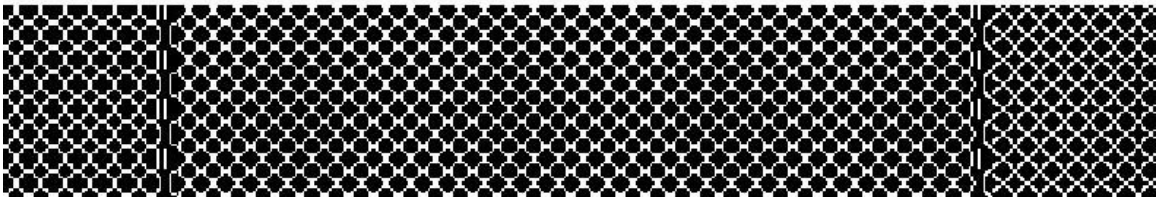
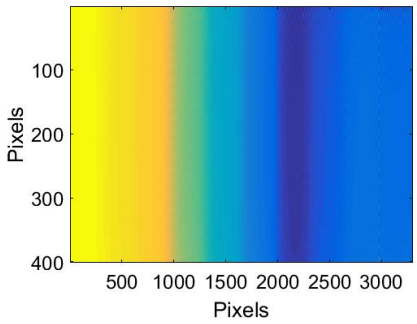
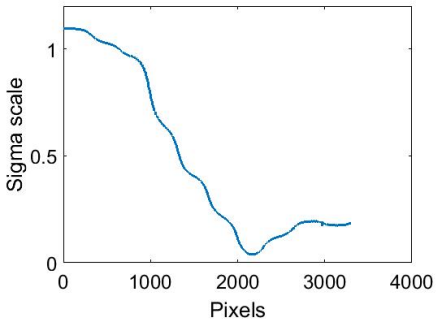


Figure 5.5: The edge thickness from higher PSF scales changes the pattern for scales of 4 and above.

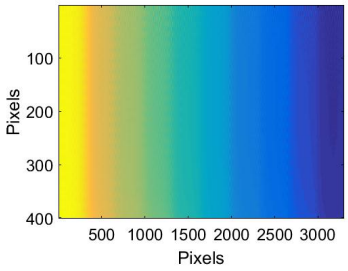


(a) Synthetic blur map with scale range up to 6 at 0.5 scale step. The pattern style reverses after a scale of 4.

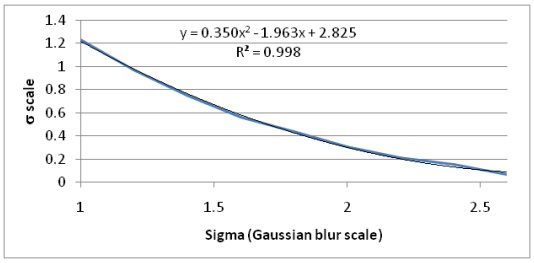


(b) Plot of a row (200) from the blur map. It can be seen that the useful scale range is >1 to 4.

Figure 5.6: Blur range finalization.

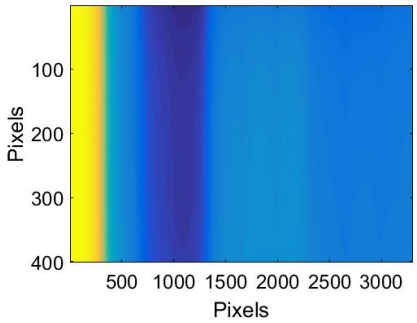


(a) Blur map with 10 PSF scales from 2 to 4.

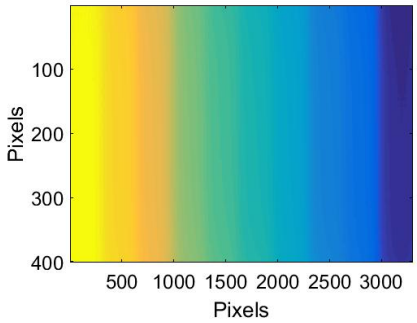


(b) Plot of synthetic blur and estimated blur after polynomial fit (row 200, mid value of each step taken).

Figure 5.7: Results with pill box PSF.

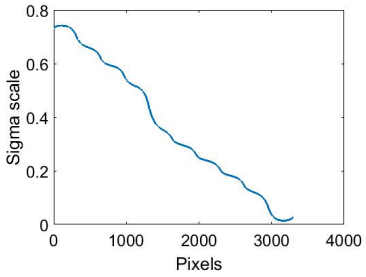


(a) Blur map, upper limit reached at σ <4.

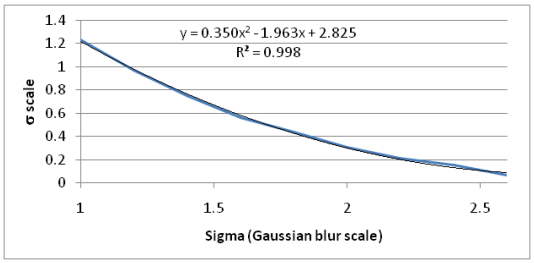


(b) PSF scale of 1 to 2.2 is not usable.

Figure 5.8: Range limits with Gaussian PSF.



(a) Plot of 200^{th} row of blur map.



(b) Plot of the mid values of each block and sigma variation and polynomial curve fitting (row 200).

Figure 5.9: Results with Gaussian PSF.

background. In addition, after r of 4, the blur level makes the black and white pattern reverse (Fig. 5.5). Fig. 5.6a shows the effect near pixel 2000. This is due to the edges becoming thicker than the square patterns. This scale hence sets the upper scale limit. A row plot of Fig. 5.6b indicates that up to r of 2, the slope gradient is slow and will provide poor depth accuracy. This sets the lower range. Hence, the usable range for r is 2 to 4. Fig. 5.7a is the blur map for this range with 10 smaller scales (r increments are < 1). Fig. 5.7b is the polynomial curve fit plot obtained from the mid values of each of these blocks at the 200^{th} row (arbitrarily chosen). The slope is similar to the Gaussian curve. The above experiments were with pill box PSF. The experiments were repeated with Gaussian PSF, and it is observed that the results are similar, with a useful range of r from 2 to 4. Figs 5.8 and 5.9 show the results with Gaussian PSF.

Figure 5.10: Ground truth, checkerboard pattern from multiple distances stitched together. Blur is higher for large distances.

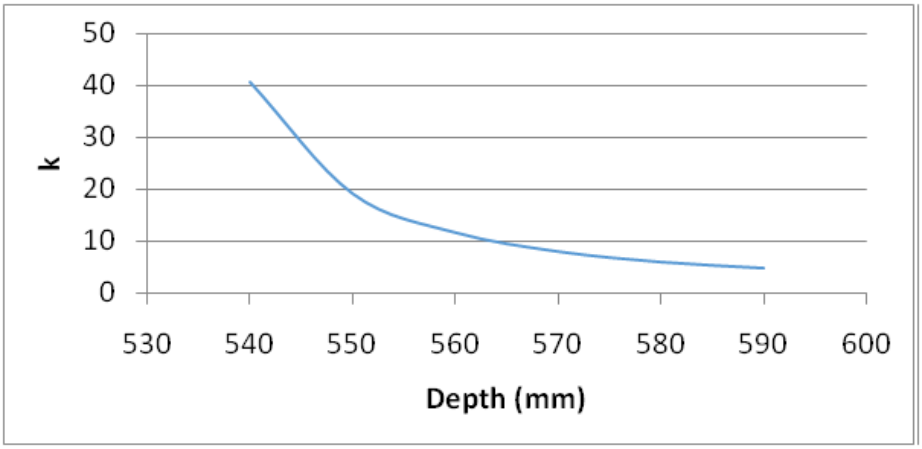


Figure 5.11: Variation of k with depth

5.4.4 Ground truth

A Canon 400D camera was used to acquire the images. The camera specifications are in Appendix (Chapter 9). The camera was set manually to 5.6 numerical aperture, focus at 530mm focus, and images were captured for multiple calibrated known depths. These images are appropriately cropped and stitched together to form the ground truth (Fig.5.10). Now, each row gives the range of real blurs corresponding to known depths. This blur is unique to a camera and the camera settings. To compute k, the relation $kc = \sigma$ is used from equation 2.1 (Fig.2.2).

Though , most manufacturers calibrate the optics before shipping, the calibration needs to be confirmed before any experiment. This was verified and Tables 3.2 and 3.3 show the experiments carried out to generate the ground truth. Further, calibration was carried out in the PSF domain, and Fig.5.6 to Fig.5.9 show the experiments completed to generate the ground truth depth range. It can be seen that the slope can be approximated to be uniform for a small range. Hence for the same camera, it is found that the curves need to be calibrated for images taken for different ranges. The 'k' values were the calibration coefficients. Fig.5.11 plots the values of k with depth. The experiments conclude that k is not a constant. The reasons are many. A theoretical thin lens model is assumed for deriving

the equations (section 2.1.2). This model follows Gaussian response for images, noise and defocus. Real cameras have a compound lens instead of a single lens and the collective focus does not match the theoretical values. Also, the aperture and the optics introduce an inherent finite blur and measured as the optical modulation transfer function. This defines the sharpness of the focused image. Further, engineering parameters like reflective index of the glass material, lens curvature (sphere, aspheric), type and number of lenses define the optical response. The above are also limited by manufacturing tolerances. The calibration method here uses a target made from various scales of blur obtained by filtering a synthetic checker box pattern with Gaussian or pill box filters. Using the calibration method, these blurs are verified with expected theoretical blurs (Fig.5.7 b and Fig.5.9 b). The polynomial fit in both cases show a good Gaussian match as R^2= 0.998. Thus the theoretical k=1 is valid.

A real blur scale target is also obtained by imaging checker box patterns at various depths (Fig.5.10). The ratio (k) of the theoretical and actual blurs showed mismatch which decreased with distance (Fig.5.11). Ideally these blur scales should match the estimated blurs of Fig.5.7 b and Fig.5.9 b. However, experiments point that there may be other factors which prevent the same. Hence, here the calibration is possible with various k coefficients. This was validated with multiple experiments (Fig.5.14). This is attributed to measurement errors which used manual setup and manual focus. Here, the best focus is subjective. Further the blur detector is sensitive to contrast, and variation of illumination during the experimentation period would also create blur scale errors.

5.5 Verification

A camera with a checkerboard pattern was set up for verification. The checkerboard pattern was horizontally slanted (about 38 degrees) to produce continuous variable-depth data. A measuring tape gives the distance from the lens center to the target pattern edges. Triangulation gave the distances from the lens center to the image pixels, as shown in Fig. 5.12. The camera's optical axis corresponds to the image center. Hence, the central horizontal row of the image provides the data for this study. This avoids radial errors on the vertical axis and at the corners. As in earlier experiments, the mid values of the step pattern (Fig. 5.13) provide the data for plotting along with ground-truth (target) depth. This graph then represents the measured blur w.r.t. distance. Three captured images (A, B, and C) with slight variations provided the measurements. Image A had a blur scale (radius) of 1.2 to 1.96. B had a range of 0.48 to 1.5, while C had 0.65 to 1.4. These blurs, along with the k

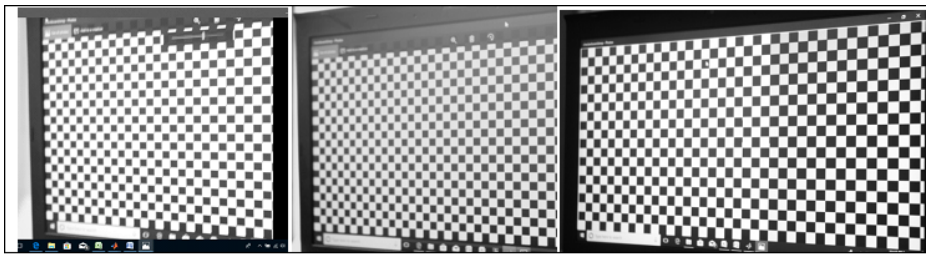


Figure 5.12: Three checkerboard pattern images, A, B, and C, taken with various slants to get continuous depth

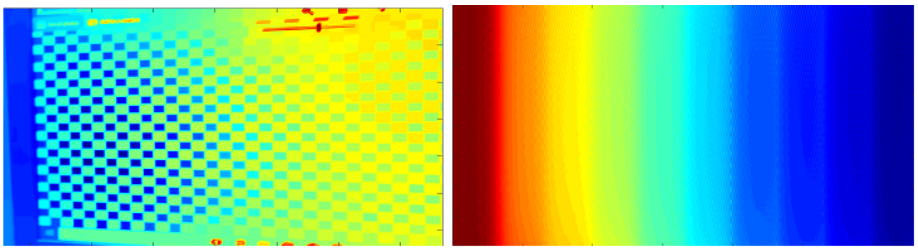


Figure 5.13: Depth Map of (a) an image with a slanted pattern and (b) a synthetic target. The color code brown indicates the nearest depth.

coefficients, give the estimated absolute depth. A comparison with the triangulated depth data showed an offset for data from A, B, and C. On analysis, it is found that the offset is due to the following:

(a) Measurements and iterations took many days at different times, and hence, the illumination varied during the experiments. The edge detector is sensitive to contrast variations.

(b) Manual setup and commercial measurement tapes posed a limit for repeatable depth accuracy. Further, radial distances from the lens create a semi-spherical object plane, whereas the 2D target was flat, which introduces errors at the image corners.

(c) The objects were focused manually, which is subjective, and a near focus is the best possible.

The offsets of images for A, B, and C were subtracted, and Fig.5.14 plots these blurred data with depth. Images for C and the target pattern used the same setup and were acquired during the same period. Therefore, the offsets of both match better, even though the blur range is different. The depth obtained using the k coefficients matched the ground truth (target) data, thus validating the proposed calibration method.

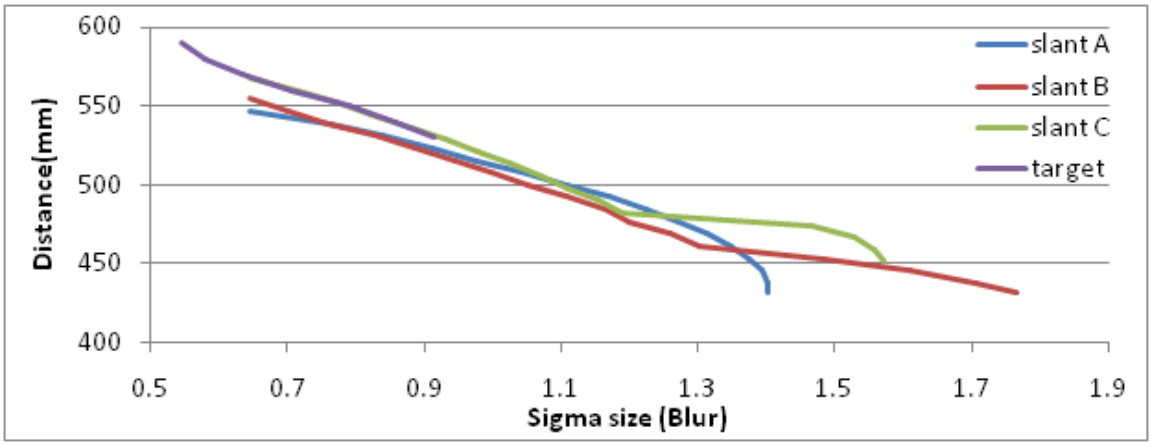


Figure 5.14: Graph of distance w.r.t. σ for the 4 verification images.

5.6 Summary

This work proposes a method with a unique blur target and ground truth to recover absolute depth from monocular images. In the target, different PSFs generate various scales of synthetic blur. This work studies these blurs and their contributions to blur detector range, object distance, illuminations, and camera focus. The proposed calibration also uses ground truth data. This consists of imaging a pattern for various known distances and stitching the images together. The theoretical blur from the synthetic target and the real blur from ground truth are related w.r.t. absolute distance to compute coefficients of k. These coefficients can estimate absolute depth from monocular images taken by the same camera. In the verification process, slanted images were calibrated using this method. A good correlation was obtained with the measured data. Offset correction was required to reduce errors due to manual setup and manual focusing. A setup with calibrated optical rails, smooth movements, and automatic measurements will reduce such errors. This method is effective for a blur radius of 1.2 in the existing setup. Beyond this range, the relation becomes nonlinear, with reduced edge detector performance and higher error in the interpolation of the depth map. Eventually, the blur increases while the contrast (magnitude) reduces, and at large distances, features merge with the background. The algorithm, then, cannot reliably distinguish the depth. As most images have some blur due to distortions in the camera, the blur range improves when sharpening the images. Sharpening with Laplacian and Sobel methods also amplifies the image noise, which is a disadvantage. This calls for a learnable algorithm to detect local features and improve depth-range detection.

The calibration is an add-on application for depth algorithms. Other edge detection

methods, like Canny's method, can also improve the sparse depths. The method can be further improved with deep learning methods that can estimate depth and improve edge definitions without increasing noise.

Chapter 6

Loss Functions for Edge Enhancement

6.1 Introduction

Depth from monocular images uses CNN algorithms with crafted loss functions. The edges in an image capture the fine local features, which define the contours, structure, and regions of the image [14]. Strong and different intensities of neighboring pixels define sharp edges in an image. Blurred edges with reduced intensity dynamic range merge contours into the background, which leads to loss of feature definitions, poor image quality, and eventually loss of depth map details [71]. Hence, the accuracy of estimated depth-map resolution improves with edge enhancement [72] and provides robust feature structure [73]. The most effective edge detectors that are popular in computer vision applications are gradients. Additional smoothing and nonlinear filters also reduce noise and fragmented edges, thus improving structure detection.

6.2 AIM

There are some research reports on enhancing depth map edges that lead to new applications. Still, the available literature lacks analysis of known edge type loss functions w.r.t. depth map quality. This work studies and suggests a variety of edge loss functions for enhancing the details of feature in-depth maps and studying the performance and quality attainable. These loss functions are used to train a known model to analyze the areas for improvement. The work also studies the combinations of these loss functions to train the network together and analyze the performance results.

6.3 Network architecture

Deep learning networks process large amounts of structured data for classification. These can work on raw images without any preprocessing and are excellent at identifying underlying patterns. These consist of input layers, hidden layers, and output layers where the data propagates in a feed-forward manner. These layers contain convolutional functions that have trainable weights as filter kernels. The weights are updated through back-propagation based on a loss function. The hidden layer has weights identical to the input image size; the hidden layers have variable weight sizes depending on backpropagation and the output layers convert these weights to scores. Usually, a pooling operation and an activation function reduce the weight for faster convergence. The layers progressively identify more and more complex patterns through these trainable weights, while the loss function guides by attaching importance to the weights to learn useful features that resemble reference patterns. The reference pattern is given as the ground truth or label. The loss function is customized to keep the useful weights. Initially, these trainable weights are randomly valued and act as seeds for convergence through the minimization of the loss error from the loss function. This process takes several iterations or epochs, each time improving the weights. A Softmax function provides the probability distribution for decision thresholding. These networks are trained using a large number of images and reference data. This large dataset is fed to the network in batches. When the loss function is at its minimum, convergence is complete and the weights are final. These weights can now be applied to any image to get the desired inference. Overall, the process of learning mimics the human brain where the weights are eqivalent to neurons. Networks are computationally complex and intensive, and it isn't easy to understand the learned weights. These require large datasets to train, which mandates large memory. Small datasets lead to a loss of generalization and overfitting, which degrade accuracy. There are many networks developed, like LeNet-5, ResNet, VGGNet, DenseNet, AlexNet, GoogleNet, MobileNet, etc., each optimized for various intelligent functions. Deep learning elements were discussed in Section 2.4. Most networks for depth estimation have encoder-decoder architecture and usually resemble a UNet. The encoder downsamples to extract the feature details, while the decoder upsamples this information to provide the required output. These networks need training with enormous image data and corresponding labeled data to be reliable and robust. The training effort is therefore very intensive. A method of augmented training called transfer learning allows us to reuse an available trained model for a new task. One such SOTA method is [1], which saves time and computation effort. This network uses a pretrained DenseNet-169 as

the backbone encoder for its deep network architecture. The DenseNet encodes the RGB images into feature vectors. The decoder layers consist of serial bilinear upsamplers (2x), skip-connections, two (3x3) convolution layers, and leaky ReLU (except the last block). This network does not use the Batch Normalization layer. The estimated depth map resolution is half the input resolution (320×240). Fig. 6.1 shows the architecture.

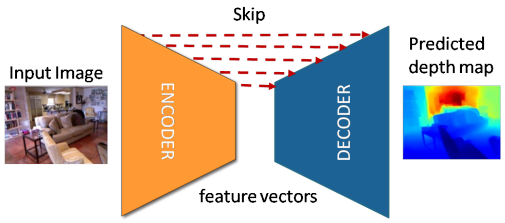


Figure 6.1: Network architecture [1].

6.3.1 Training a Network

Training a network can be either supervised or unsupervised, depending on the outcome and requirements. Classification, depth estimation, etc. need supervised training along with ground truth. Unsupervised training is possible for identifying a similar group of images using methods like k-means. The training starts with the forward propagation of inputs. In supervised learning, the hidden layers of the network learn the mapping between inputs and the corresponding ground truths. The learned model weights and biases predict the outcome. The error between the ground truth and the predicted output is computed using a defined loss function. The loss function gradients computed from the trainable parameters are back-propagated through the network, which evaluates and updates these trainable parameters through a gradient descent method. The gradient information is processed for the entire training set. All training samples are propagated forward and backward through the network to complete an epoch. The trainable parameters are updated after each epoch to reduce the loss function. The forward and backpropagation during the training of a network model is iterative and stops when the predicted error is at its minimum.

6.3.1.1 Transfer learning

In transfer learning, a trained model is re-trained for a new task. The method uses the information gained through earlier training to improve upon generalizations on a new problem. This saves computation and time.

6.4 Loss function

A deep-learning neural network learns to map an input set to a reference set. Since the weights to map are difficult to estimate due to many unknowns, these are solved as an optimization problem [74]. Training involves a stochastic gradient descent optimization algorithm with the weights updated through the backpropagation of the error algorithm. The function that has to be minimized or maximized is the loss function. The loss function compares the Predicted Output and the true output to provide a prediction error. If the compared values are far off, then the loss value is high, and the model is penalized during training by propagating through the network and updating the weights. Small losses do not affect the weights. The loss function reduces the various good and bad aspects of the complex system to a single scalar value, which allows the candidate solutions to be ranked. The loss function ensures that the model will work in an intended manner while providing practical flexibility in neural networks. The loss function is also critical for learning efficiency and prediction accuracy. Hence, the loss function is tuned to enhance the outcome. The various loss functions used are summarized below:

6.4.0.1 Mean Square Error Loss (MSE)

MSE, or L_2 loss is calculated between predicted (Y_{pred}) and actual values (Y). Large mistakes are penalized by squaring as:

$$MSE = \frac{1}{N}\sum_{i=0}^{n}(Y_i - Y_{pred_i})^2 \tag{6.1}$$

6.4.0.2 Mean Squared Logarithmic Error Loss (MLSE)

MSE also penalizes target values that are spread. This is avoided by taking the natural logarithm of each of the predicted values before MSE computation.

6.4.0.3 Mean Absolute Error Loss (MAE)

Gaussian data distribution may have outliers that are far from the mean value. The popular MAE loss or L_1 loss function has a large constant gradient and is more robust to outliers.

$$MAE = \frac{\sum_{i=1}^{n}|Y_i - Y_{pred_i}|}{N} \tag{6.2}$$

6.4.0.4 Huber Loss

Huber loss is less sensitive to outliers than MSE, differentiable at 0, and has a threshold δ. It approaches MSE when $\delta \sim 0$ and MAE when $\delta \sim \infty$.

$$L_\delta(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & for|y - f(x)| \leq \delta, \\ \delta|y - f(x)| - \frac{1}{2}\delta^2 & otherwise \end{cases} \tag{6.3}$$

The hyperparameter δ makes the loss flexible and can adapt to many distributions. A smooth approximation of Huber Loss is the Pseudo Huber loss.

6.4.0.5 BerHu Loss

This reversed Huber loss is appropriate for normally distributed errors with heavier tails, where many pixels have small depth values, leading to data imbalance problems. It switches to give higher weight to pixels with higher residuals as MSE loss and simultaneously allows smaller residuals to have a larger effect on the gradients as MAE loss. It is continuous and differentiable at the switch point c.

$$B(x) = \begin{cases} |Y_{pred_i} - Y_i| & |Y_{pred_i} - Y_i| \leq c, \\ ((Y_{pred_i} - Y_i)^2 + c^2)/2c & |Y_{pred_i} - Y_i| > c, \\ c = 0.2max_i(|Y_{pred_i} - Y_i|), & threshold \end{cases} \tag{6.4}$$

6.4.0.6 Log-Cosh Loss

It is used for tasks that are smoother than MSE. It works mostly like MSE without being strongly affected by occasional wild, incorrect predictions. It has all the advantages of Huber loss and is useful for hessian (second derivative) tasks. However, it works on a fixed scale.

$$L(Y, Y_{pred}) = \sum_{i=1} log(cosh(Y_{pred_i} - Y_i)) \tag{6.5}$$

6.4.0.7 Quantile Loss

Quantile loss predicts an interval instead of only a point for tree-based models. Quantile loss and quantile regression provide prediction even for residuals with non-constant variance or a non-normal distribution. It is an extension of MAE and gives different penalties

for overestimation and underestimation based on the chosen quantile value.

$$L_y(Y, Y_{pred}) = \sum_{i=Y_i<Y_{pred_i}} (\gamma - 1).|Y_i - Y_{pred_i}| + \sum_{i=Y_i<Y_{pred_i}} (\gamma).|Y_i - Y_{pred_i}| \tag{6.6}$$

The above loss functions are geometric loss types. These loss functions are summarized in Fig.6.2. Other loss functions are possible. Some are given below.

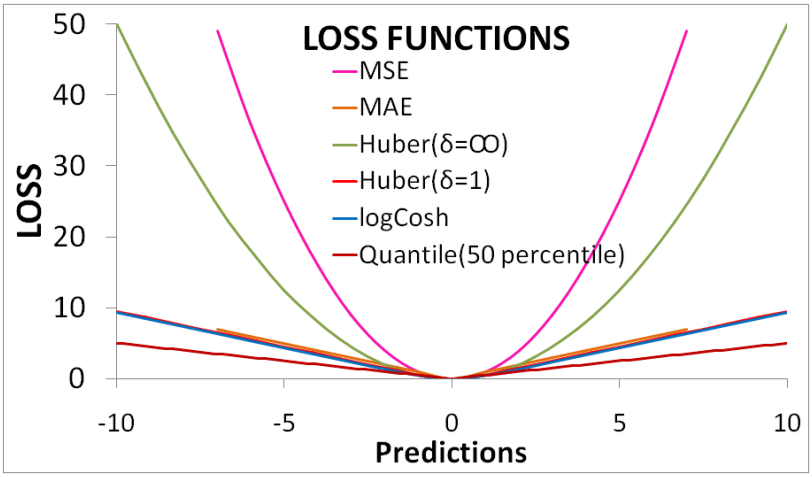


Figure 6.2: Plot of Loss Functions

6.4.0.8 LSSIM (Structural Similarity Index)

This loss computes the perceptual difference between two similar images. LSSIM is a non-geometric loss, and the structural loss information comes from spatially close pixels, which have strong neighboring interdependencies [41] and carry important information about the structure of the objects in the visual scene. Identical images score the lowest. The loss function is:

$$L_{SSIM} = 1 - SSIM(Y_i, Y_{pred_i}) \tag{6.7}$$

When two images are similar, the SSIM loss is 0. In most applications, the penalization is reduced by a standard weight of 0.5 on the loss.

6.4.0.9 Photometric loss

Provides the color differences computed on each pixel independently.

6.5 Formulation-Multi-Loss Function

The formulation of a loss function for training is an important activity for training a network. In this depth estimation task, the loss function should guide and penalize the model to converge, minimize loss, sharpen feature edges, and recover depth information similar to the labeled ground truth. Further, the loss function needs to be optimal and train the model within a short time by reducing iterations in the recursive methods. Researchers use multiple loss functions along with additional hyper-parameters (λ) to control the penalization [1, 41, 65]. The formulation may use multiple standard loss functions, as in the previous paragraph, or use custom loss functions or a hybrid combination. The loss functions used in the study are a hybrid combination of standard and custom loss functions. The choice of loss functions is discussed below. The Mean Absolute Error Loss (MAE) or L_1 loss function is a popular loss that is robust to outlier data. MAE computes the pixel-wise error as:

$$L_{pix} = \frac{\sum_{i=1}^{N} |Y_i - Y_{pred_i}|}{N} \tag{6.8}$$

where Y_i is a pixel in the ground truth depth map, Y_{pred_i} is a pixel in the estimated depth map, and N is the total number of pixels in the depth map. MAE is linear and provides equal weight to both lower and higher values, thus resulting in fewer predictions and poor training. The BerHu Loss [38, 37] overcomes this problem and applies mean square error (MSE) to ensure pixels with higher residuals get higher weight. Simultaneously, it applies MAE loss to smaller residuals for a larger effect on the gradients. BerHu uses a threshold c to provide this trade-off between L_1 loss and L_2 loss. A threshold of 20% of the maximum error in a batch is the standard level in most work. The BerHu Loss is:

$$L_{pix} = \begin{cases} |Y_{pred_i} - Y_i| & |Y_{pred_i} - Y_i| \leq c, \\ ((Y_{pred_i} - Y_i)^2 + c^2)/2c & |Y_{pred_i} - Y_i| > c \\ c = 0.2 max_i(|Y_{pred_i} - Y_i|)\} & \end{cases} \tag{6.9}$$

Fig.6.3 shows the simulated plots of MAE (L_1), MSE (L_2), and BerHu loss functions. The advantages are seen. Loss functions of a depth map are also required to define parameters for image structure and features. SSIM loss helps in constructing the image structure by assessing the perceptual difference between two similar images [41]. The structural loss information comes from close spatial pixels that have strong interdependencies. Constructing a depth map from a sharpened ground truth will lead to more details. Hence, this study

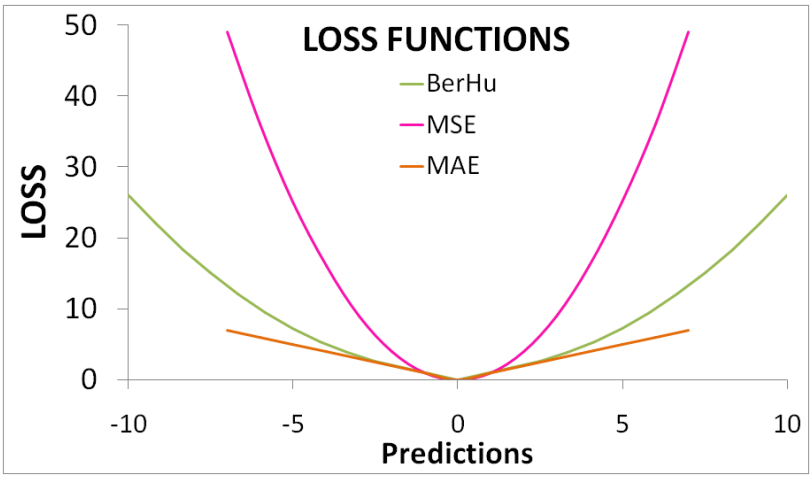


Figure 6.3: Plots of MAE, MSE, and BerHu functions and characteristics.

uses a new loss, which is an improved SSIM loss, where the ground truth image Y_i is replaced with an edge-enhanced, sharpened ground truth image Y_i' as:

$$L_{SSIM} = \frac{1 - SSIM(Y_i', Y_{pred_i})}{2} \tag{6.10}$$

A derivative Laplacian filter of second order sharpens the image Y_i by highlighting the inward and outward boundaries of feature edges present in the image. This filtered image is added to the original image. The Laplacian filter kernel is:

$$s = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

The features with high-frequency components of the structure need reconstruction through the edge loss function. The various edge loss functions studied are Sobel, Gradient, Difference of Gaussian, Laplacian, and Laplacian of Gaussian. The aim is to improve the details of the depth map by sharpening the features. In this study, the network model is trained one at a time with each type of edge loss, which totals a minimum of five training sessions along with several iterations. The Gradient operator is the simplest edge loss function and is the maxima of the derivative at the feature edges. The derivative produces a positive peak followed by a negative peak. Mathematically, the horizontal and vertical intensity gradient

of an image Y is:

$$\begin{aligned} \frac{\partial Y}{\partial x} &= Y(x+1,y) - Y(x-1,y), \\ \frac{\partial Y}{\partial y} &= Y(x,y+1) - Y(x,y-1), \\ L_{edges-1} &= mean\left(\left|\frac{\partial Y_{pred}}{\partial y} - \frac{\partial Y}{\partial y}\right| + \left|\frac{\partial Y_{pred}}{\partial x} - \frac{\partial Y}{\partial x}\right|\right) \end{aligned} \tag{6.11}$$

The gradient operation produces double edges for lines, which is unwanted. Further, image noise creates small peaks, which calls for removal by usual thresholding. The Sobel operator is a good edge detector and uses vertical and horizontal kernels (3x3 each) to provide the 2D gradient map. The advantage of this operator is its lower sensitivity to noise. The kernels are:

$$S_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \; S_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

The x and y kernels are applied separately to an image to get the gradient components in the required orientations. The absolute gradient magnitudes L_{mag}, computed from these components, are:

$$|L_{mag}| = |S_x| + |S_y| \tag{6.12}$$

The Sobel operator-based edge loss function L_{edges} is:

$$L_{edges-2} = mean(|Y_{pred} * S_x - Y * S_x| + |Y_{pred} * S_y - Y * S_y|) \tag{6.13}$$

The loss is the mean of the magnitudes obtained after taking the difference between the estimated depth and the ground truth depth gradient magnitudes. The algorithm iteratively penalizes large differences to obtain minimum errors for all depth values. Sobel operators have the limitation of providing first-order derivatives in only x and y directions. Further, a requirement for thresholding exists to reduce the noise. The Laplacian edge detector provides second-order spatial derivatives to detect rapid changes and zero crossings. This detector is insensitive to orientations. The Laplacian operates on image $Y(x,y)$ with a single kernel to provide the $L(x,y)$ coefficients as:

$$L(x,y) = \frac{\partial^2 Y(x,y)}{\partial x^2} + \frac{\partial^2 Y(x,y)}{\partial y^2} \tag{6.14}$$

The Laplacian kernel is:

$$L = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

The edge-loss function is then

$$L_{edges-3} = mean(|Y_{pred} * L - Y * L|) \tag{6.15}$$

Since the operator uses a single kernel, it is computationally faster. However, these second-order derivative operators are very sensitive to noise and need preprocessing with Gaussian filters to smooth the noise. The preprocessing stage reduces the noise from all the training RGB and depth images. The 2D representation of the Gaussian filter is:

$$G_\sigma(x, y) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{6.16}$$

The above preprocessing for the Laplacian edge detector leads to the 'Laplacian of Gaussian' (LOG) operator loss function. The LOG operator is hence useful for noisy images. The single equation for the LOG operator is:

$$LOG(x, y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2 + y^2}{2\sigma^2}\right] exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{6.17}$$

Here, zero crossing defines the edge position. Many filtering kernels are possible, and here a 3x3 Gaussian kernel is implemented as:

$$Gausssian = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

The edge loss function then becomes:

$$L_{edges-4} = mean(|Y_{pred} * (Gausssian * L) - Y * (Gausssian * L)|) \tag{6.18}$$

Most images inherently undergo noise filtering before further processing. If these filtered images are used, this loss function will have the advantage of computational speed, as with the Laplacian operation. However, noise filtering reduces high-frequency components or sharp edges in an image. This may lead to the loss of feature boundaries, thus increas-

ing errors. Two different Gaussian filtering kernels can act as an edge detector. Here, the difference of images operated upon by two different smoothing scales (σ) creates a 'Difference of Gaussian' (DOG) operation. The zero crossing gives the edge position. The DOG operator is:

$$DOG \triangleq \frac{1}{\sqrt{2\pi}} \left(\frac{1}{\sigma_1} e^{-(x^2+y^2)/2\sigma_1^2} - \frac{1}{\sigma_2} e^{-(x^2+y^2)/2\sigma_2^2} \right) \tag{6.19}$$

The DOG is useful for noisy images or those with rapidly varying contrast, like shadows. As with LOG operations, preprocessed images need only one filter operation. Further, since the DOG operation is derivative-independent, the computation is simple and fast. The first scale of the 3x3 Gaussian kernel is as given above. The second 5x5 Gaussian kernel, along with the edge loss function, are:

$$Gausssian2 = \frac{1}{256} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \end{bmatrix}$$

$$\begin{aligned} L_{edges\ 5} = mean(|(Y_{pred} * Gausssian - Y_{pred} * Gausssian2) \\ -(Y * Gausssian - Y * Gausssian2)|) \end{aligned} \tag{6.20}$$

The DOG loss function gives weak edges due to the loss of high-frequency components from double smoothing. This becomes prominent for already blurred background features in an image (Fig.2.2). The total loss function is the combination of the loss functions discussed above. In this work, this comprises pixel-wise loss (MAE or BerHu), edge loss (one of gradient, Laplacian, Sobel, LOG, DOG), and structural loss (SSIM or modified SSIM loss) as given below:

$$\begin{aligned} L_{total}(Y_i, Y_{pred_i}) = \lambda L_{pix}(Y_i, Y_{pred_i}) + L_{SSIM}(Y_i, Y_{pred_i}) \\ + L_{edges\ t}(Y_i, Y_{pred_i}) \end{aligned} \tag{6.21}$$

Where t is the suffix (1 to 5) for the type of edge loss function as above. Here, the L_{pix} loss (L_1 or BerHu loss) is penalized more with lower weights compared with structural and edge loss. The (λ) is taken as 0.1 based on reference [1]. Author [65] reported that the errors in predicted depth increase proportionally with object distances in the image. Usually, the compensation requires taking the reciprocal of the depth information in ground truth data for the predictions. The NYU dataset has a maximum depth of 10 m, so the reciprocal is $10/Y_i$.

6.5.1 Optimization

Optimization of the network involves minimization of the loss function during training. During training, large training sets consume time for gradient computations. Stochastic gradient descent here can speed up the process by computing gradients over a set of random samples or batches instead of the entire set. The learning rate hyperparameter also has an impact on training performance. An optimal training approach needs a large learning rate initially and, subsequently, a lower learning rate to avoid overshoot and local minimums. The Adaptive Moment Estimation (ADAM) optimizer adapts the learning rate individually for each parameter. ADAM is memory- and computationally efficient. This study uses a learning rate of 1e-4.

6.5.2 Regularization

A network should perform well on training data and non-training data. This generalization is verified on a test dataset. A network with few parameters will perform poorly due to under-fitting. With very large parameters, the network may adapt to training data and perform poorly on test data due to overfitting. Regularization methods can improve generalization. In the drop-out regularization method, a random subset of hidden neurons is temporarily deleted during training. This random subset is defined by a hyperparameter. This parameter is typically set to 50% probability. This forces the neurons to avoid dependency on each other and makes the training less prone to overfitting. Under-fitting can be overcome with more data sets. However, for the limited available datasets, data augmentation is a way out. Here, the existing dataset is modified by random operations like rotation, flipping, cropping, translation, scaling, contrast changes, the addition of noise, mirroring, random color channel swapping, etc. These altered images, which are augmented with the current dataset, are treated as new inputs by the network. Data augmentation is adapted for the study. Information from homogeneous zones of the scene is difficult. Regularization, like the L_2 norm, is a simple method to overcome depth discontinuities. Other regularizers are based on edge-preserving methods that give sharper depth maps.

6.6 Ablation Studies

Various studies were conducted to tune the network. The weights for loss functions were tuned to get the optimal values. The results for model B+S+SSIM' are tabulated in Table

6.1. The overall best performance is achieved for λ_1, λ_2 and λ_3 = 1. These weights are used for our work.

Table 6.1: Performance of weights for B+S+SSIM' model

λ_1	λ_2	λ_3	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	**REL** $\downarrow$	**RMSE** $\downarrow$	**log10** $\downarrow$
0.1	1	1	0.811	0.971	0.994	0.133	0.606	0.06
1	0.1	1	0.842	0.972	0.9941	0.128	0.539	0.054
1	1	0.1	0.848	0.973	0.9939	0.126	0.534	0.054
1	1	1	0.845	0.973	0.9939	0.124	0.524	0.053

The work also studied the performance with two different encoders, viz., DenseNet169 and DenseNet201. Both models were trained for 10 epochs, and the performance is tabulated in Table 6.2. DenseNet201 has 201 layers with 20M parameters, thus being more accurate.

Table 6.2: Comparison of different encoders

Model	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	**REL** $\downarrow$	**RMSE** $\downarrow$	**log10** $\downarrow$
DenseNet169	0.8453	0.9732	0.9939	0.1238	0.5242	0.053
DenseNet201	0.8495	0.9736	0.9941	0.1233	0.5264	0.0528

Model 'B+S+SSIM' was trained with DenseNet 169 and DenseNet 201 encoders. DenseNet 201 performs better.

6.7 Experiments and Results

The work uses resources from Google CoLaboratory to train the model. A Tesla T4 with a single GPU is used on the cloud server with CUDA version 11.2. The training was for 20 epochs, as no significant improvements were observed. Each epoch runs for an hour, and to avoid disconnection, a batch of three epochs was run in one go. Further, the batch size was kept at 6 to meet the allocated memory limits. The paper [1] uses a batch size of 8 to obtain optimal results. However, the impact of batch sizes up to 16 was low, and hence performance errors due to batch sizes of 6 can be assumed to be negligible. Later, access to CoLab Pro was possible. This gave a faster Tesla P100 or V100 with a better GPU and a higher 100 MB of memory. This allowed a single run of batch sizes of 8 and higher. The algorithm used an ADAM optimizer for the training and randomly initialized the network decoder weights. All ground truth and predicted depth values are limited to 0.01 and 1 to avoid 'divide by zero' computing errors. The work implements a variety of combined loss functions for training to study performance. These are in the following format:

1. Sobel + Gradient + MAE **(Sobel)**

2. BerHu + SSIM + Gradient **(BerHu)**
3. SSIM-Sharpened + BerHu **(BSSIM')**
4. Laplacian + Gradient + MAE **(Laplacian)**
5. BerHu + Sobel + SSIM-Sharpened **(B+S+SSIM')**
6. BerHu + Sobel + SSIM **(B+Sobel)**
7. SSIM-Sharpened + Gradient + MAE **(SSIM')**
8. SSIM-Sharpened + BerHu + Gradient **(SSIM'+B)**
9. LOG + Gradient + MAE **(LOG)**
10. DOG + Gradient + MAE **(DOG)**

The initial training dataset was the NYU-V2 dataset (4.1 GB) hosted by the Silberman dataset site. The dataset was further divided into training, validation, and datasets. Since the prediction error is higher for higher ground-truth depth values, the inverse of the depth is always taken. To have the same level of ground for comparison, the model of reference [1] is also retrained. This is labeled as 'Trained' in Table-6.3. The required kernels are as given in the 'Loss function' sub-section. The standard performance metrics (Chapter 2) are used for evaluating the trained models along with the NYU-test sub-dataset. This allows reliable comparison of this work with other SOTA algorithms. Table-6.3 gives the results with the proposed loss functions and compares them with the original work. Fig. 6.4 compares the visual quality of estimated depth maps for different loss functions. The study also compares the visual improvements by taking the difference between actual and predicted values (Fig. 6.9). Further, the best loss function is chosen to train the model with a dataset provided by the author [1].

6.8 Results and Observation

The NYU dataset provides diverse, labeled complex indoor images covering big and small rooms, a variety of textures (furniture, wall corners, door, ladder, rack, roof, etc.), rooms with different illuminations, large and small features, etc., all at various distances. The proposed loss functions are verified with these images. The pixel-wise loss functions are compared, and it is found that BerHu gives better performance, as shown in Table-6.3 and

Table 6.3: Performance comparison of trained models

Model	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	**REL** $\downarrow$	**RMSE** $\downarrow$	**log10** $\downarrow$
Sobel	**0.8526**	0.9709	0.9919	0.1253	0.5585	0.0529
BerHu	0.851	0.972	0.993	0.126	0.552	0.053
BSSIM'	0.8505	0.9727	0.9932	0.1265	0.5367	0.0533
Laplacian	0.8504	0.9716	0.9926	0.1255	0.5441	0.0531
B+S+SSIM'	0.8494	0.9727	0.9940	0.1231	0.5234	**0.0528**
B+Sobel	0.8480	0.9725	0.9937	0.1259	0.5388	0.0534
SSIM'	0.847	0.9725	0.9929	0.1237	0.5429	0.054
SSIM'+B	0.8448	0.9706	0.9934	0.1275	0.5513	0.0539
LOG	0.8316	0.9669	0.9926	0.135	0.579	0.0565
DOG	0.8034	0.9588	0.9898	0.146	0.6032	0.0617
Baseline [1]	0.846	0.974	0.994	0.123	0.465	0.053

Note: The models are ranked as per δ_1, with the largest on top. The best results are in bold. Overall, the best performance is met by model B+S+SSIM'.

Fig. 6.6. Among all the loss functions studied, it was found that the model trained with the Sobel edge function provided better visible features with higher false colors (Fig. 6.4). Sobel has the best accuracy and depth range (distinct in images 3, 6, 7, and 8). Image 7 highlights the higher depth range for Sobel, as the range covers up to the corridor end. However, non-uniform lighting in the corridor causes depth errors here. The Sobel operator uses vertical and horizontal kernels to provide first-order derivatives in only x and y directions. It also has the advantage of lower sensitivity to noise. These properties resulted in better visible features with higher false colors. The performance is followed by BerHu, SSIM', SSIM'+ BerHu, and LOG, while the rest fare poorly. Sobel, however, performs poorly for near objects, as missing features are seen in Fig. 6.7. The near-depth predictions are better for BerHu, followed by Laplacian, Sobel, and DOG (image 4). BerHu and DOG provide good mid-range performance by detecting more details, as shown in image 6. Other loss-based training shows a wider range of colors here. SSIM' + BerHu brings more edge definitions and better dynamic range (images 2 and 4) compared to BerHu and SSIM'. However, the performance for near objects is relatively poor. LOG and DOG have poor depth range (image 9) and Fig. 6.8. Depth detection under poor scene illumination is poor for all loss functions, and Sobel gave a depth error for the notice board in image 7. Performance metrics show that $\delta 1$ values are higher for SSIM', SSIM'+BerHu, Sobel, and Laplacian; Sobel gives 85.26% compared to the original paper's 84.6%. This work also compares achieved performance with the performance metrics from the original paper [1]. This is also tabulated in Table-6.3 as the last row. Compared to these values, the thesis work performance is better in most columns. It may be noted that the authors used a modified dataset with inpainting. The study additionally uses this dataset to train the best loss

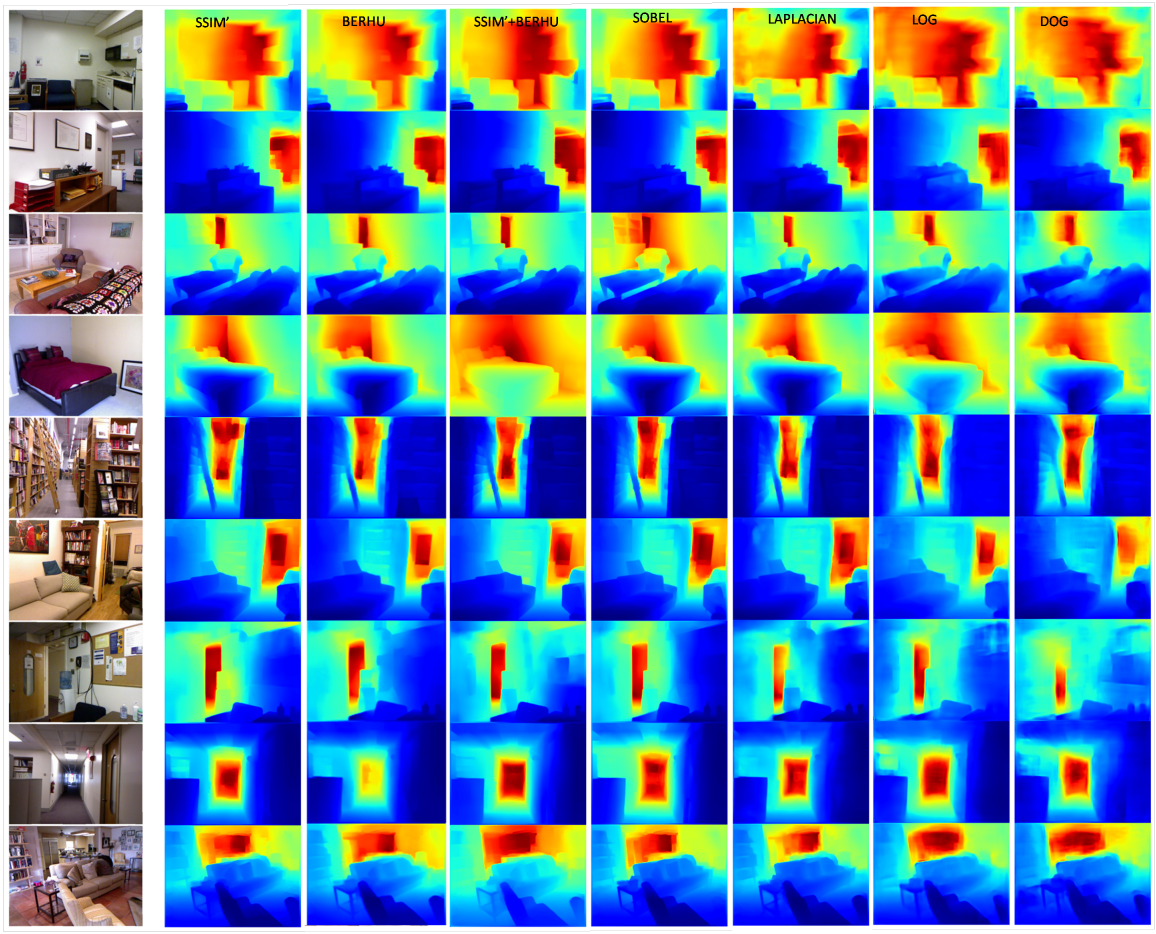


Figure 6.4: A visual comparison of predicted depths with proposed edge loss functions (a) SSIM', (b) BerHu, (c) SSIM' and BerHu, (d) Sobel, (e) Laplacian, (f) LOG, (g) DOG

functions, BerHu, Sobel, and SSIM'. The results showed good performance for the near range and poor performance for the mid and far ranges. Hence, the model is trained with BerHu, Sobel, and SSIM losses. Fig. 6.9 shows the results along with error maps.

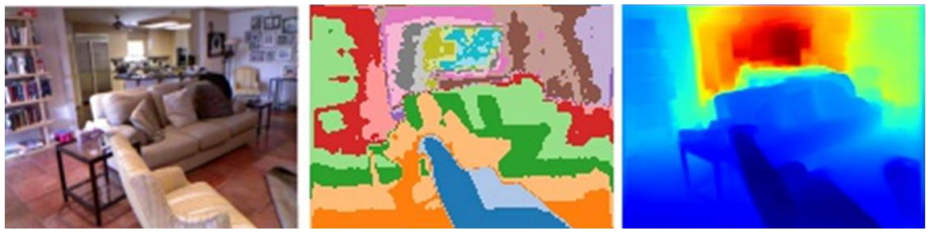


Figure 6.5: (a) NYU Dataset original image (b) Ground truth (c) Estimated depth map from [1]

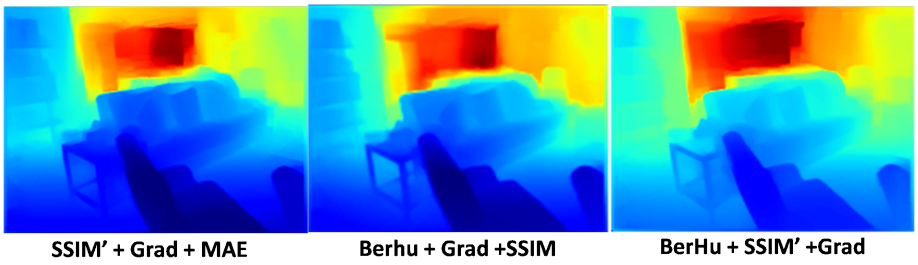


Figure 6.6: Predicted depth map with proposed loss functions (a) SSIM sharpened, (b) BerHu, (c) SSIM sharpened, and BerHu

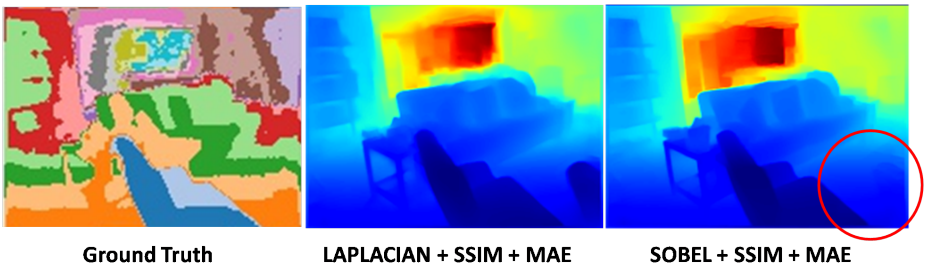


Figure 6.7: Performance of depth estimation (a) false color ground truth, (b) Laplacian loss function, (c) Sobel loss function

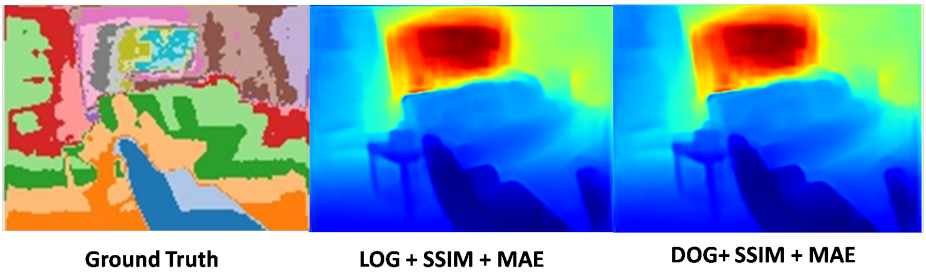


Figure 6.8: Edge loss functions and predicted depth-map (a) false color ground truth, (b) LOG, (c) DOG

6.9 Summary

The study proposed new loss functions to improve the feature details of estimated depth maps. To identify the improvements, the study uses a ready-made network and an established transfer learning method. Further, the network is trained with proposed edge loss functions using the standard NYU dataset and evaluated with popular evaluation metrics. The tailor-made edge loss functions are SSIM with sharpened image, Laplacian loss, Sobel loss, LOG loss, and DOG loss, and combinations of these loss functions with L_1 loss, reverse Huber loss and gradient loss. These loss functions are not reported in available

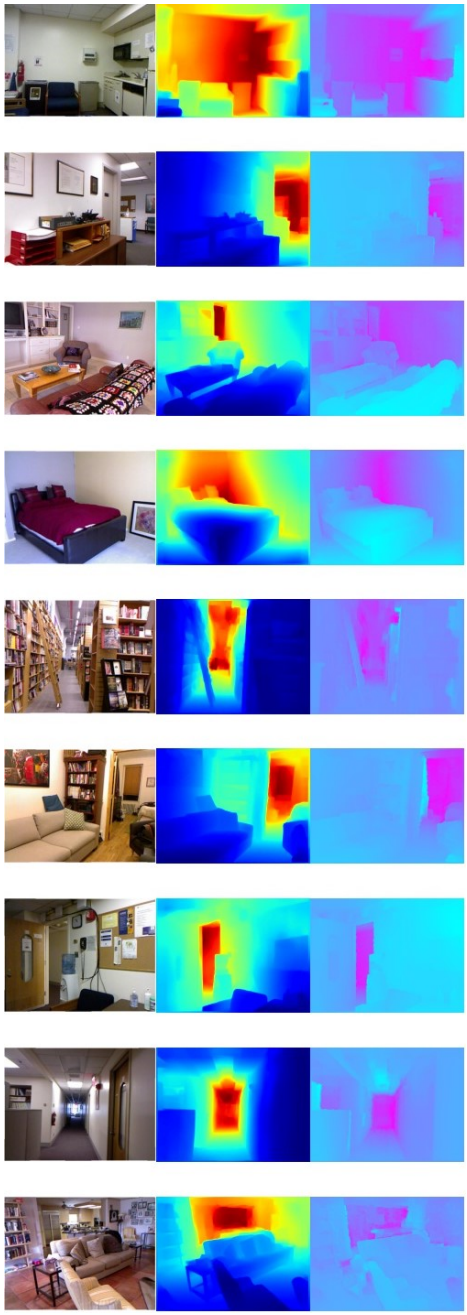


Figure 6.9: (a) NYU Dataset Original Image (b) Predicted depth map (c) Error map of ground truth and predicted

research works in this domain. The metric performance parameter $\delta 1$ was higher for loss functions with BerHu, Sobel, Laplacian, SSIM with sharpened depth map and a combination of SSIM with sharpened depth map plus BerHu. The Sobel edge function gave the best performance, while SSIM with a sharpened depth map was marginal. The performance of

LOG and DOG-based edge loss was poor. Apart from Sobel, Canny Edge detectors are also possible here. The important steps of Canny Edge Detectors are (a) smoothing filter to reduce noise, (b) deriving the intensity gradients, (c) Non-maximum suppression for single edges, (d) Double thresholding for strong and weak edges, and (e) Track edge by hysteresis and suppress non-connected edges. Here, in this study, steps (a) and (b) are used, that is Gaussian and Sobel filters with thresholding. The steps (c), (d), and (e) can lead to sharper depth images and may be tried in future studies. This may call for additional running time and feature weights. However, this edge detection will not perform on homogeneous surfaces like walls at an angle to the camera. The authors of the original paper used a custom-improved dataset from the NYU V2 dataset. The study used the original NYU V2 dataset as-is. The work also compares the achieved performances with their published report. They use the same dataset and model but use a different loss function. Here, the performance for Berhu, Sobel, Laplacian, and SSIM with sharpened depth maps is found to be superior. From this study, it is concluded that the choice of edge functions has a good impact on the estimation of depth maps. It is also observed that edge functions based on derivative and differentiation loss functions are better than smoothing-type loss functions. In this study, the following contributions are made to the research community:

1. Five new edge loss functions developed including SSIM loss function, which gives sharpened depth map features.

2. A multi-loss function with 4 different operators to improve the depth map

3. Standard performance analysis with a popular dataset and the visual quality of proposed edge-enhancing loss functions

The UNet-like network used for experimentation had a DenseNet backbone, which is complex and computationally intensive. Training needed about 10 epochs, and computation with the GPU took about 150 seconds per epoch. In most robotic applications, power and computing resources are constrained. Here, a simpler network will be an advantage, as mobile systems like smartphones do have a resource crunch. Further studies for better edge operators like Canny's can also be looked into.

Chapter 7

Nested Wavelet-Net for Depth Estimation

7.1 Introduction

Estimating depth from a single image is a well-known nonlinear problem with many solutions and therefore ill-poised. Researchers have successfully used convolutional neural networks (CNN) to solve these depth estimation problems for single images. Such networks need training with datasets having diverse images and labeled data (ground truth). After successful training, the network model can predict depth from live photos. The trainable network model can develop custom architecture with building blocks like convolution layers, pooling functions, activation layers, and expansion layers [1, 38, 39]. Further, the training is either supervised or unsupervised [37, 38, 42, 43, 44]. An image consists of global and local structures. Global structures provide an overview of the scene with semantic information, textures, sky, horizon, etc. The local structures enhance the details of these global structures with information like edges, local features, noise, contours, local regions, etc. [14]. In the frequency domain (Fourier, wavelet, etc.), the global features show up as low frequencies, while the local features that represent them show up as high-frequency components. Most local features have these high-frequency edges created by contrasting intensity gradients among structured neighboring pixels. As discussed earlier, smoothing functions like averaging degrade the edge gradients and merge the features with the background. Similarly, the averaging process of pooling or down-sampling layers in the network reduces these high-frequency components to introduce jagged edges and blur in images. The object definitions and details are significantly reduced, which causes errors in estimating depth [71]. Researchers have tried to address this degradation by implementing various methods for edge enhancements [72, 73, 75]. Recently, they have also used Discrete wavelet transform (DWT) as part of the network architecture. This transform converts the spatial images with contrast variations into frequency domain informa-

tion and has found wide acceptance in the down-sampling and up-sampling layers of the network. Since the high-frequency information is preserved throughout the process, artifacts are reduced [76, 77, 78, 79]. DWT breaks the image into one separate low-frequency and three high-frequency components. Learning all these coefficients during training improves the related boundary edges and reduces the noise. Up-sampling this information with Inverse DWT (IWT) will provide quality outputs without the loss of high frequencies. Networks for depth estimation use pretrained encoders like ResNet, DenseNet, VGG, etc., which are computationally and time-intensive. The simplest fully convolutional network is UNet [80] with encoder-decoder blocks. Each encoder block has convolution operators, a down-sampling operation to reduce the size by 2, and an activated output. The decoder block has an upsampling and convolution operation that expands the size of feature maps by 2. Skip connections, taken from the corresponding encoder block, are fed to the decoder block to enhance the output predictions. These connections preserve boundaries. Over the years, the research community has made several improvements to the UNet architecture. One significant improvement was the addition of soft attention [81], which allows the network to focus on relevant zones with low computational complexity. This feature reduces redundant image features and improves prediction accuracy. Another improvement was the development of Residual UNet [82], which overcame the problem of accuracy degradation while improving training ease. MobileNet [83, 84] is another light network that uses depth-wise separable convolutional layers to reduce computational effort. Wavelet transform has also been used in a variety of image processing applications, including depth map prediction [78, 85, 86, 87, 88, 89]. Some researchers have used multiple wavelets [77], while others have learned wavelet coefficients [79]. A nested UNET++ using dense skip paths and multiple decoder chains was an improvement from [68]. In the past, these lean networks catered to medical analysis and semantic applications with small databases, faster learning, good accuracy, and low computation resources [90]. Lately, the advantages of UNet-like architectures have been exploited for depth estimation [91, 92] for faster learning and implementation in less computationally intensive systems. The pooling and upsampling processes were subsequently replaced with DWT and IWT layers in a UNet variant by [93].

7.2 AIM

A survey of research work on monocular depth estimation did not yield much work with UNet-like light networks, indicating these may be unexplored. This work discusses the

development of a moderately dense network for studying depth estimations. Looking at the advantages of DWT, the work looks at developing a custom network architecture using DWT and IWT. It is expected that this proposed nested DWT network (NDWTN), after training, will be better than UNet and UNet++. The work also creates variants of this network to understand the optimal and final network elements. The aims here are summarized as:

1. Explore light networks for depth estimates. Depth inference networks employ encoder and decoder architecture. The study also needs to identify the appropriate network elements.

2. Design of a multi-scale DWT network architecture for monocular depth estimation,

3. Investigate alternate lossless scaling methods like Wavelet,

4. Improve learning of local features with dense skip and attention functions,

5. Higher feature extraction and learning with dense convolution blocks.

The main novelty of our network is the use of multistage encoding and decoding; attention-based skip paths for information enhancements among the stages; DWT-based pooling instead of Max pooling to avoid information loss during scaling; UNet-like architecture to train faster; and Batch normalization and residual convolution layers. These features lead to higher training, validation, and evaluation scores. The main advantage comes from DWT, which preserves high and low frequencies (LH, HL, and HH) with localization and reduced coefficients. This reduces the size by two, similar to pooling. The DWT transformer is invertible and composes all coefficients back to the original image without any degradation, unlike pooling. All the proposed networks are trained with datasets. Ablation studies were also completed. Analysis was carried out and the results were compared with SOTA investigations in this domain.

7.3 Wavelets

Wavelets are various functions from the Wavelet Transform family that exhibit properties of rapid decay and orthogonality. These have advantages over Fourier functions, where the data contains discontinuities and sharp spikes, as these functions process data at various scales. These have large-scale-varying Basis Functions to provide frequency and temporal

information. A function in a discrete wavelet can be represented by a linear combination of two basis functions:

$$f(t) = \sum_{k} c_k \varphi(t-k) + \sum_{k} \sum_{j} d_{jk} \Phi(2^j t - k) \tag{7.1}$$

The above equation has the first part as a scaling function and the second as a wavelet function. The j represents scale or frequency, and the k is the position. The simplest wavelet is Haar, and this mother wavelet is represented in time t as:

$$\Phi(t) = \begin{cases} 1 & 0 \leq t < 0.5, \\ -1 & 0.5 \leq t < 1, \\ 0 & otherwise \end{cases} \tag{7.2}$$

The scaling function of Haar is:

$$\varphi(t) = \begin{cases} 1 & 0 \leq t < 1, \\ 0 & otherwise \end{cases} \tag{7.3}$$

Another popular wavelet is the Daubechies wavelet, commonly known as db1–db10. The index refers to the number of coefficients. This wavelet has the maximum vanishing moment, which is half the coefficients. The db1 is the same as the Haar wavelet. Wavelet transforms on an image provide spatial frequency information at lower scales and decompose an image into a low-frequency coefficient map (LL) and three high-frequency coefficient maps (LH, HL, and HH). Further decomposition is possible using the LL map only. Thus, many scales and multiple coefficients are arrived at, as shown in Fig. 7.1. The size of the derived coefficients is halved w.r.t. the input for each scale. The DWT transformer is invertible and, hence, can compose all four frequency coefficients back to the original image without any degradation. This needs an inverse DWT or IWT in the upsampler, which recovers the image to twice the coefficient resolution as the original. The DWT coefficients for image Y are:

$$LL, LH, HL, HH = DWT(Y) \tag{7.4}$$

Implemented wavelets have successfully improved accuracy in deep learning and are immune to noise [94]. In a network, the downsampling layer reduces the output size by 2 w.r.t. the input. This makes the DWT a good fit here and replaces the pooling layer. Pooling averages the features to reduce the high-frequency components and scale the input size

by 2. An additional advantage of the DWT layer is that the coefficients are learned and estimated with training, eventually ensuring that the high-frequency edges are preserved or improved. Among the wavelets, Haar and Daubechies are popular. The simplest basis is the Haar wavelet. The Daubechies wavelet has vanishing moments. This work implements a DWT scale of 1 to match the pooling size. The study uses both Daubechies (db4) with four vanishing moments and Haar wavelets as part of the experiments. The DWT implementation is similar to [95].

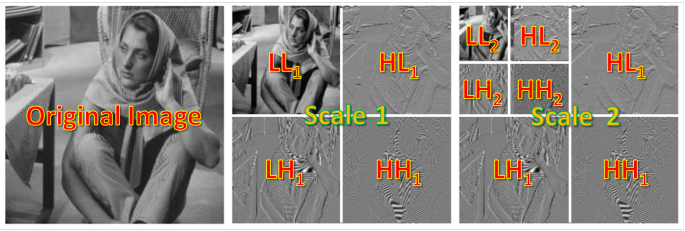


Figure 7.1: DWT decomposition and low-resolution coefficient maps. Here, the scale is 2, however, the downsampling operation requires a scale of 1.

7.4 Deep network architecture

A convolutional neural network (CNN) is a subset of artificial neural networks and comes from the family of machine learning. The network working principle and training were discussed in sections 2.4 and 6. CNN has wide range of image processing applications. These are Deep Learning methods that require less preprocessing compared to other classification techniques. These networks have neurons that self-optimize through learning and backpropagation by solving a non-convex optimization problem. An error or loss function computes the error between the input image and the labeled image (ground truth) for backpropagation. This error updates the learnable filter weights during back-propagation (see Fig. 7.2). This predicts statistical information through a series of hidden layers for multiple classifications. Deep neural networks (DNNs) based on VGG, ResNet, DenseNet, etc. are complex and computationally expensive. Therefore, this work investigates simpler networks with low complexity to estimate depth maps systematically with faster training.

The basic elements of a network are the convolutional layer, Pooling layer, and Activation layer.

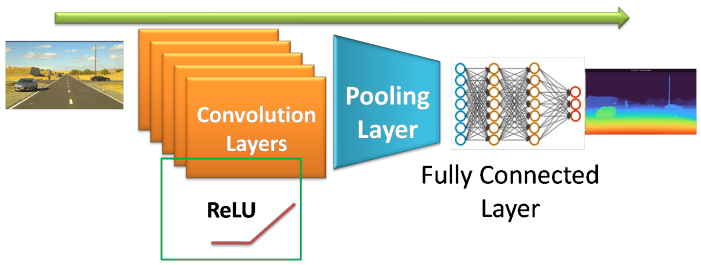


Figure 7.2: CNN- basic elements.

7.4.1 Convolution Layer

The convolutional layer convolved the input (usually an image) with a scanning trainable filter. The scanning step is defined by the stride. Here, neurons connect to different regions of the neurons of the previous layer. The region size depends on the filter size. Convolution results in a feature map that represents the underlying patterns or features. These can be edges, lines, and structures. The filter weights and biases are learned during training and define the types of features to detect. These parameters are shared across the image to detect similar features at various locations.

7.4.2 Pooling Layer

A pooling layer reduces the spatial size of feature maps. This reduces the number of learning parameters and improves the computational efficiency of the network. In this operation, A small, non-trainable filter operates over feature maps to provide a summary of neighboring values. Since only one value is returned for a region, the spatial size of the feature map is reduced or downsampled. The filter size and the stride control this size reduction. The max-pooling operation takes the maximum of the values within the filter region to indicate underlying features. Average pooling returns the average of all values within the filter region. Information on the exact spatial location of the feature is lost with these operations.

7.4.3 Activation functions

Activation functions are non-linear and make the model under training learn high-order polynomials between the input and ground truth. It determines the output of the neural network by mapping the input values. The most common and simple activation function is the rectified linear unit or ReLU. ReLU is nonlinear for negative values and linear for positive values. The function, therefore, provides only positive values and reduces the

vanishing gradient problems. For an input x, ReLU is given as

$$R(x) = max(0, x) = \begin{cases} \text{x, if } x \geqslant 0 \\ 0, \text{ if } x < 0 \end{cases} \tag{7.5}$$

Leaky Relu is a variant of ReLU and allows negative values with a small gain to solve the dying ReLU problem. The Sigmoid function, or Softmax function, has an S-shaped curve and is used to predict the output as a probability between 0 and 1. This function saturates for higher values and is suitable for multi-class classification. The function, for an input x, is

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{7.6}$$

7.4.4 Regularization

During the training, a covariance shift in layer outputs is expected after the update of trainable parameters. This leads to vanishing activations. Batch normalization [96] reduces this shift by normalizing the input to each layer per batch and prevents saturated activations.

7.4.5 Proposed Network Architecture

The proposed development is a moderately dense network architecture, the Nested Discrete Waveform Transform Net, or NDWTN. The proposed network comprises an encoder and multi-chain decoders. The encoder provides image compression to extract image features. This half of the network consists of a series of convolutional and downsampling layers that gradually reduce the spatial dimension and resolution of the input image. The second half of the network is a decoder that gradually expands the feature information through a series of convolutional and upsampling layers. This increases the spatial dimension to the original dimension while reducing the resolution of the feature dimension to 1. The decoder chains have four different scales. The encoders and decoders are connected at each level by nested dense skip paths. These help preserve fine-grained information from the input image and improve the accuracy of the segmentation results. Downsampling is usually a pooling operation. This is a convolution layer that reduces the size of the input image and the number of parameters, which improves training speed. The common pooling operations are Max-pool and Avg-pool. Max-pool layers select the maximum value within 2x2 pixel blocks, thus reducing the size by 2 in both dimensions of the image. This preserves the prominent features, and the reduced image is sharper. The AVG-pool layers select the average value

from 2x2 pixel blocks, again reducing the image size by 2. This smooths the image while retaining the essence of the features. Apart from these pooling layers, researchers have also used strided convolution layers to reduce image size and enhance feature information. The upsampling layers must expand the pooled output to the desired resolution by adding pixels. One such popular method is bi-linear upsampling. However, as the downsampling process is lossy, high-frequency information cannot be recovered during the upsampling process. Skip paths help here to some extent by concatenating information from encoders. To solve this issue, DWT for downsampling is implemented. This inherently preserves the high-frequency information after upsampling with IWT. Further, the wavelet coefficients (LL, LH, HL, and HH) are learned through training. The proposed network architecture is shown in Fig. 7.3. The encoder has five blocks, like UNet. Each of these downsamples the input by 2. Fig. 7.4 illustrates one encoder block, which consists of multiple convolution layers (green color) and a DWT layer for downsampling (red block). The convolution layers comprise filter kernels that have learnable weights to filter features from the input. Here, the information energy is compacted through the computation of cross-correlation, which creates a feature map representing the underlying patterns like edges, lines, structures, etc. Here, the filter weights and biases that are learned through training define the features to be classified across the image locations. Convolution layers increase the computation complexity and the feature extraction size. These encoder layers sandwich other layers like activation and batch normalization. The layer density is also customized for convolution and batch normalization. Fig. 7.5 shows one customized encoder block. This work also implements residual-type convolution by replacing the exiting convolution. This is indicated by the '+' sign in the blue block (Fig. 7.4). These have parallel identity skip connections from input that add to the convolution output. This provides easy training for deep networks while improving accuracy. Fig. 7.6 shows one customized encoder with residual convolution. Except for the first and last encoder, each encoder block feeds the next encoder, a parallel decoder, and a horizontal skip connector. A decoder block up-samples the input data and expands the information by 2. This block consists of an IWT layer for up-sampling (red block) and multiple convolution layers (green color), as shown in Fig. 7.7. An additional input 'a' is taken from the encoder through the skip connector and concatenated to the up-sampled outputs (blue block). The convolution layers are the same as those in the encoders and can be replaced with a residual convolution. The customization includes the density of convolution layers, batch norm layers, and activation layers (Fig. 7.8). Each decoder block feeds another decoder above it and totals ten in the four chains. Each decoder chain is on a different scale. The scales are visually rep-

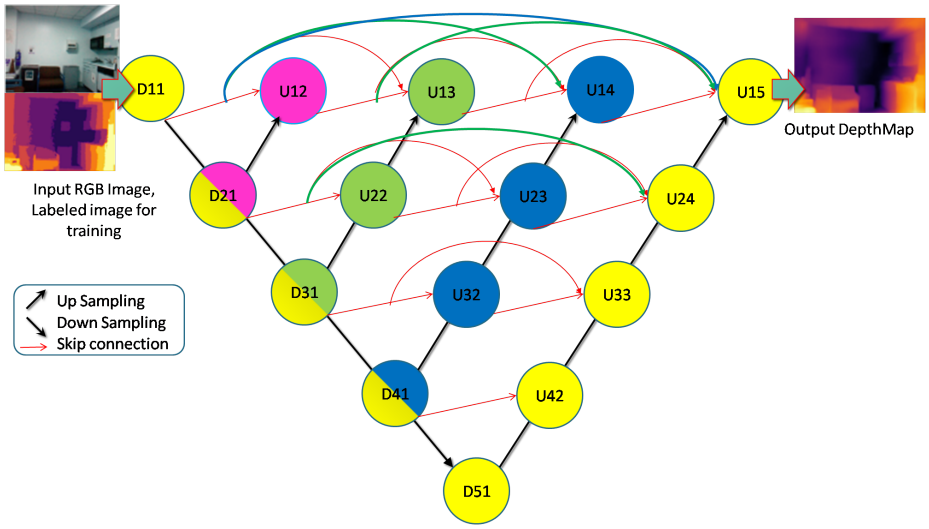


Figure 7.3: Proposed Network Architecture (NDWTN)

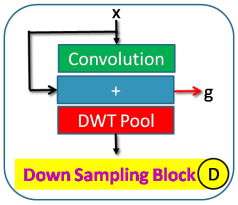


Figure 7.4: Structure of down-sampling block.

resented with pink, green, blue, and yellow colors. The output of each decoder chain is implemented with convolution layers, possibly a batch normalization layer, and a Sigmoid activation layer. These outputs independently predict the depth maps (Fig. 7.9). All three outputs are further added to the fourth chain through skip connections to provide a single inference and also improve the prediction accuracy. Activation function layers are used both in encoders and decoders. This layer decides whether a weight should be upgraded or dropped, thus activating a neuron. The decision uses a threshold along with a bias to regulate the output of neural networks across several domains, depending on the function it represents. These introduce non-linearity to make the model learn complex relationships between inputs and outputs.

$$f(a) = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b \tag{7.7}$$

Some widely used activation functions are:

- Linear: The activation function is directly proportional to the input and follows a

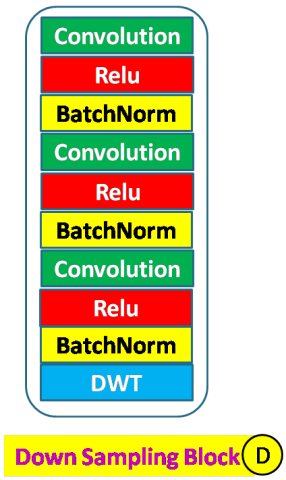


Figure 7.5: Down-sampling block details: The stack of convolution operators and the sequence of Batch-Norm and activation layers are customized.

straight line.

- ReLU: This function removes the negative values by rectification action. This avoids overfitting by not activating all neurons.
- Leaky Relu (LR): LR modifies ReLU and provides a small constant gain to negative weights instead of making the values zero. This avoids "dying ReLU" and improves accuracy.
- Parametric Relu: This function is an improvement over LR by giving the choice of best gain or gradient for the negative region.
- Exponential Linear Unit (ELU): This function is exponential and leads to faster convergence. with higher accuracy.
- Swish: This activation function is computationally more efficient than RelU.
- Sigmoid: This nonlinear S-shaped function maps real input values $[-\infty, -\infty]$ to within [0,1].
- Tanh: This non-linear activation function is similar to the Sigmoid function and maps the input to within [-1,1].
- Softmax: This function provides the probability of target classes and is used for multi-class classifications.

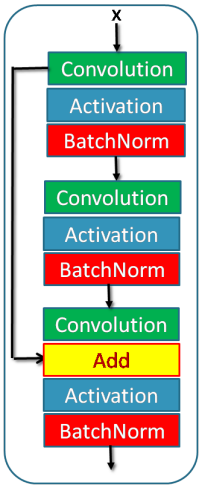


Figure 7.6: Residual convolution block. The stack of convolution operators and the sequence of Batch–Norm and activation layers are customized.

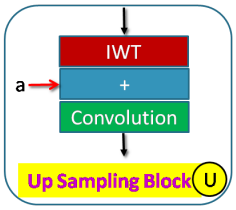


Figure 7.7: The upsampling block provides IWT and convolution operations. Information from the skip path 'a' is also concatenated.

The choice of activation function depends on the defined task and the input. This study uses ReLU, LR, and Sigmoid. Nesting of information in the proposed network is possible with skip connections. The encoder provides the convolution layer output to the skip layer, which concatenates with the decoder IWT output. Each level of encoder connects with the same level of decoder from each chain through multiple skip layers. This reduces semantic gaps between the encoder and decoder by providing high-level information and retaining structural information. This path can be an attention gate. The attention gate consists of convolution and activation layers, as shown in Fig. 7.10. It takes two inputs, 'x' and 'g', which relate to the encoder input and the convolution layer output, respectively. The 'x' is operated with a strided convolution to match the feature dimensions of 'g'. The outputs are added to induce higher aligned weights and reduce unaligned weights. A ReLU activation layer filters in the higher weights. A subsequent convolution layer reduces the feature dimensions to 1. A final Sigmoid activation layer scales the information to [0, 1]. These attention coefficients are upsampled to match the dimensions of the IWT output in the de-

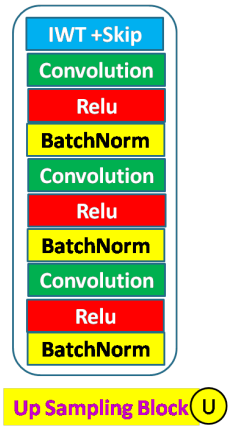


Figure 7.8: Up–sampling block details: The stack of convolution operators and the sequence of Batch–Norm and activation layers are customized. The convolution stack can be replaced with a residual block.

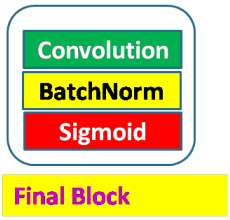


Figure 7.9: Output block with Sigmoid activation layer

coder. The attention gate preserves local feature details by adding the high-resolution local information from the encoder input to the global contextual information in the decoder. This gate concentrates the high-energy features by focusing on local contrasts and sharp edges to generate an attention grid and finally enhance the high frequency of local features in the estimates. Fig. 7.11 shows the proposed model with attention gates. The work uses

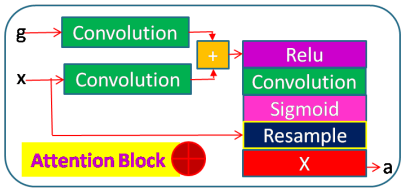


Figure 7.10: Skip layer with attention. This layer takes two inputs from encoder blocks of different scales, 'g' from the higher scale or input to the encoder and 'x' from the lower scale or output of the encoder, and feeds the decoder block with attention vectors 'a'.

two activation functions. The ReLU (Rectified Linear Unit) function is computationally efficient, allows backpropagation, and boosts all positive values. However, neglecting the negative values kills neurons with negative bias. The Leaky ReLU (LR) activation has a

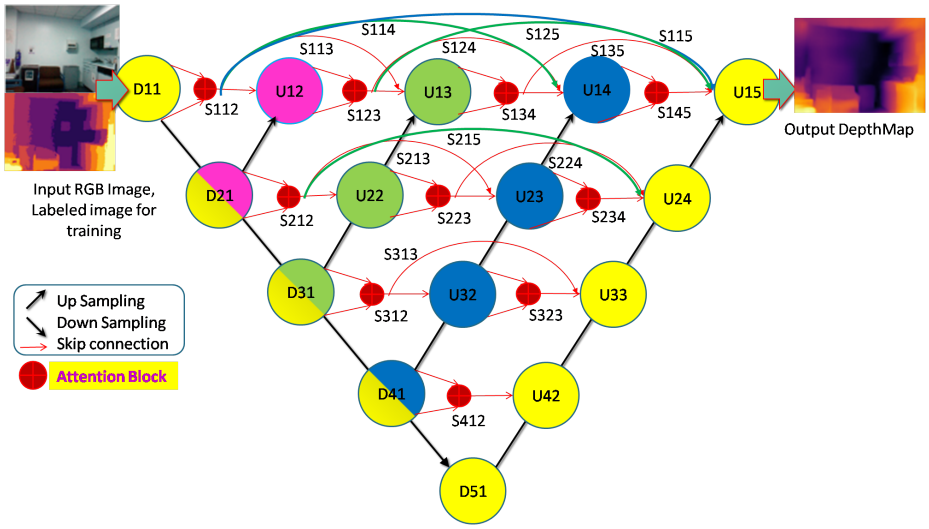


Figure 7.11: Proposed Network with Attention Gates (NADWT)

small positive gain for these negative values to overcome such problems. ELU (Exponential Linear Units) have advantages over these two functions but are computationally intensive and, hence, not considered. The work implements both activation functions in the network and studies the improvements in each case. The Batch Normalization layer is not implemented in the original UNet and UNET++ networks. This layer reduces Covariate shift to remove bias, improves training time, and reduces overfitting. This layer was proposed to be before the activation function [96]. However, many researchers reported improvements with this layer placed after the activation layer. This sequence of layers is also studied to finalize the model. The layer includes two learnable parameters and two non-learnable parameters (Mean and Variance Moving Averages). The NDWTN takes small input image shapes (240,240,3 or 240,320,3). This resizing is sufficient for most low-level applications and is popular among most researchers to save computation resources. Higher sizes are possible, with an impact on computation power and time. The NDWTN encoder reduces the size of the input image to (15, 15, 1024) after processing five blocks. The decoder blocks then restore the dimension to (240, 240, 1) for the depth map. The network is optimized through performance studies. The study includes the effects of layers, combinations, and sequences. The study works out 13 variants of NDWTN based on convolution layers, residual convolution layers, batch normalization layers, activation types, and attention features. These architecture variants are designated as:

- NDWT: Basic NDWTN;
- NADWT: NDWTN with attention on skip paths;

- NRDWT: NDWTN with residual blocks;
- NARDWT: NRDWT with attention on skip paths.

The input for these networks is an RGB image with a resolution of 240 $\times$ 320 $\times$ 3 pixels and a corresponding labeled data resolution of 240 $\times$ 320 pixels. This ensures that the estimated depth map resolution is 240 $\times$ 320 pixels. The dimensions used are arbitrary. The trainable parameters of the proposed models are higher than basic UNet and UNET++. The non-trainable parameters are less than the DenseNet backbone UNet [1]. The computational costs (MACs, FLOPs) are computed after training the basic and DWT-based network models. These are compared with [1] who have provided a trained model. It can be seen that DWT-based models increase the number of filter weights. Attention methods further increase the parameters due to the number of nodes in the path. Higher numbers do improve the features of depth maps but may also include non-contributing weights. These can be pruned along with optimizations methods based on network node trimming, efficient convolutions, and weight trimming in future works. The non-trainable parameters also increase with DWT and attention methods. These are mostly due to batch normalization layers, which update mean and variance weights. Loss functions use specific filters that are also non-trainable and add to this parameter. The computation speed in floating operations per second is higher for NDWT and NADWT compared to DenseNet. This matches with the short time required for training and testing. Table.7.1 showcases proposed model benchmarks with other published models.

Table 7.1: Trained Models and parameters.

Models	Total Params (Millions)	Trainable Params (Millions)	Non-trainable Params (Thousand)	MACs (G)	FLOPs (G)
UNet	7.7	7.7	5.9	12.1	24.2
UNET++	13.2	13.2	10.9	38.9	77.8
MobileXNet [84]	24.9	–	–	4.9	9.8
NDWT	46.2	46.2	14.6	95.3	190.5
NADWT	53.9	53.9	21.3	**111.1**	**222.1**
DenseNet [1]	53.9	53.9	158.4	4.4	8.8
AdaBins [51]	78	–	–	–	–

DWT: UNet with DWT layers, ADWT: UNet with DWT + Attention layers.

7.5 Loss Function

The importance of the loss function was discussed in Section 6.5. This function is critical and tuned to get satisfactory results along with a faster training duration. It may consist of

many sub-loss functions. Hyper-parameters control the penalization impact of these sub-loss functions in the overall convergence algorithm. The model converges through gradient descent during the training steps. Here, this function helps to minimize the training loss by estimating depths that are comparable to the ground truth. The proposed comprehensive loss function contains pixel-wire loss (MAE or BerHu), edge loss (gradient), and structural loss (SSIM). This loss is:

$$\begin{aligned} L_{total}(Y_i, Y_{pred_i}) = \lambda_1 L_{pix}(Y_i, Y_{pred_i}) + \lambda_2 L_{SSIM}(Y_i, Y_{pred_i}) \\ + \lambda_3 L_{edges\ t}(Y_i, Y_{pred_i}) \end{aligned} \tag{7.8}$$

7.6 Experiments

The study uses T4, P100/V100, a single GPU, and 5GB of GPU memory from Google CoLaboratory to train the models. The core is a Tesla T4 with a single GPU from the cloud server with CUDA version 11.2. The training was limited to 10 epochs to compare model-to-model performances. Further, batch sizes of 4 and 8 for the models met the resource memory limits. The training hyperparameter for the learning rate was initially 0.0001 and decayed exponentially as the epochs increased. The study used an ADAM optimizer for the training and randomly initialized the network weights. All ground truth and predicted depth values are limited to 0.01 and 1 to avoid divide-by-zero computing errors. The batch normalization layer can be before or after the activation layer, and this study (ablation) analyzes the impact of adding this layer and the layer sequence position in the convolution block. The study also experiments with the number of convolution layers in the convolution blocks and varies the layers. More layers condense the image energy, increase learnable parameters, remove redundant or unwanted information (neurons), and hence lead to better decisions and estimations. However, these layers increase computation complexity, decrease image features, and consume training time. An activation layer further filters the information coming from the convolution layers. These have a non-linear function to remove neurons with low values and boost the selected neurons by increasing their weight. A computationally efficient ReLU (Rectified Linear Unit) is implemented, which removes the negative input values. This activation layer suffers from a 'dying ReLU' problem, and so many researchers have used the Leaky ReLU (LR). LR has a small positive slope to boost values in the negative zone. Since there is no evidence in the literature of the superiority of these activation functions, experiments to study their performances have been carried out. The dataset for training is important for benchmarking. Many usable datasets are available

(section 2.5). The study uses a common, popular database to enable easy and standard comparisons. Most researchers use the NYU dataset, and hence this is used. This dataset is segmented into three subsets for training, validation, and testing. Similarly, standard metrics (section 2.6) performed benchmarking. A multi-element-based loss function is used for training. Empirical experimentation with the hyper-parameters λ_1 to λ_3 provides optimum weights as $\lambda_1 = 0.5$, $\lambda_2 = 1$, and $\lambda_3 = 0.1$. The loss function is also modified, and MAE is replaced with Berhu. The impact of functional layers in the model is studied by developing various models, each with varying layers in the block structures. The analysis of the layers in the performance is also part of the study, as discussed below.

7.6.1 Network Models

The internal block architecture defines a network model. Since the work proposes a new network, a study and analysis of the contributions of various layers in a network block have been completed. These are (a) convolution layers and the density of these layers; (b) the existence of batch normalization layers; (c) the density of batch normalization layers in the network; (d) the sequence of this layer with the activation layer; and (e) types of activation layers (ReLU, LR). Further, the work analyzes the addition of attention and residual convolution in these variants. These studies called for the development of variants of the proposed model. The developed model variants have the following combinations:

1. NDWT (3C, 3R, 3Bs)+Bs
2. NADWT (3C, 3LR, 1Bs)
3. NADWT (3C,3LR,1Bs) +Bs
4. NADWT (3C, 3Bs, 3R) +Bs
5. NADWT (3C, 3R, 3Bs) +Bs
6. NRDWT (3C, 3R, 3Bs) +Bs
7. NRDWT (3C, 3Bs, 3R) +Bs
8. NARDWT(3C, 3LR, 3Bs)+Bs
9. NARDWT (3C, 3R, 3Bs) +Bs
10. NARDWT (3C, 3Bs, 3LR) +Bs

11. NARDWT (3C, 3Bs, 3LR)

12. NARDWT (3C,3LR)

13. NARDWT(4C,4Bs,4LR) +1Bs)

Where: C: Convolution Layer, R: ReLU, LR: Leaky ReLU, Bs: Batch Normalization, NUMBER: Number of Layers Implemented Among the various models, NDWT is the base model with dense skip layers and multistage decoders. This model has 5 encoder blocks (block D in Fig. 7.5) each having one input and two outputs. The outputs connect to the higher scale of the D block below and the skip layer (Fig. 6.1). DWT layers perform all the down-sampling operations. The decoder block has two inputs and one output. An IWT layer is the up sampler and takes input from the skip connection and the previous decoder (U, as in Fig. 7.8). Here, a concatenate layer adds the information for further expansion by IWT. The rest of the decoder block is similar to the encoder block. There are four separate decoder chains, each comprising 10 decoder blocks. The final output is obtained by combining the outputs of the four decoders through skip connections. The convolution layers in all encoder and decoder blocks of the residual NRDWT model have an additional skip path (Fig. 7.6). The NADWT model is a modified NDWT model that implements an attention layer. This layer takes one input and one output from each encoder and decoder block to feed attention gates in the skip connectors (Fig. 7.10). The output feeds the decoder on a higher scale. Adding attention layers to the NRDWT model gives the NARDWT model. Variations in structural layers in the encoder and decoder blocks of these models provide the other variants.

7.6.2 Ablation Studies and Results

This section provides detailed ablation studies for 13 different network variations based on (a) convolution layers and the density of these layers; (b) the existence of batch normalization layers; (c) the density of batch normalization layers in the network; (d) the sequence of this layer with the activation layer; and (e) types of activation layers. Further, the work analyzes the addition of attention and residual convolution in these variants. The work used NYU datasets from the Silberman dataset site for training the proposed models. The dataset consisted of training, validation, and test subsets. The NYU dataset has diverse, labeled, complex indoor images covering various rooms, textures, room elements at random distances, varying illuminations, etc. The loss functions are verified with these images. The performance of all 13 models is also compared with that of a UNET++

model trained with the same database. Fig. 7.12 summarizes the performance of all the models for a sample test image. The visual quality of estimated depth maps highlights the improvements of the proposed network w.r.t. the UNET++ model (Fig. 7.12C). The basic NDWT model (Fig. 7.12(1)) with 3 convolution layers, 3 ReLU layers, and post-batch normalization provides better feature detailing. The visibility of edges indicates that high-frequency components are well preserved and propagated by the skip connections. Here, the depth map shows more features. This result also highlights the performances with various variations in layers and sequence. Models with attention are NADWT(3C,3LR,1Bs), NADWT(3C,3LR,1Bs)+Bs, NADWT(3C,3Bs,3R)+Bs, and NADWT(3C,3R,3Bs)+Bs (see Fig. 7.12(2–5)). These models improved the object boundaries (Fig. 7.12(2)) through higher weight in these relevant areas. The NRDWT with residual convolution also improves the object details (Fig. 7.12(6,7) left corner objects) but blurs the edges lightly. Fig. 7.12(8–13) shows the proposed residual model with an attention gate. The combined LR and batch normalization only improved the depth dynamic range when implemented at the final stages (Fig. 7.12(3)). The model with this structure gave the best performance. In residual with attention models, R activation performed better, as in Fig. 7.12(8–13). Batch normalization layers before the activation layer also provided similar improvement (Fig. 7.12(4, 6, 7)). However, batch normalization after each activation layer degrades the dynamic range (Fig. 7.12(5)). This is mainly due to the pruning of the neurons with low weights by the activation layer. Reducing the batch norm layers at the final output stage leads to small losses, as seen in the sofa arm details (Fig. 7.12(11)). A model without any batch norm layer has reduced definitions of features in the estimated image (Fig. 7.12(12)). Models with higher convolution layers estimated stronger objects for near ranges (Fig. 7.12(13)). However, the definitions at the end of the room are poor. Reducing the convolution layer density to 2 layers from 3 layers degraded the performance. Increasing the layers from 3 to 4 in the NDWT model also degraded the performance. Hence, the optimal number is kept at 3. Figs. 7.13 and 7.14 compare the evaluation loss and accuracy performance of each model. The loss values should be low, and the accuracy should be high. Here, for comparison, the study again trains a UNet after replacing downsampling and upsampling with DWT. The performance of the proposed models is better than that of this UNet. Training with structural changes required many iterations; however, only the significant results were shortlisted. Figs. 7.15–7.18 show the loss and accuracy performances for training and validation datasets. The jagged lines indicate over-fitness or under-fitness during training. All the curves are very close, indicating small variations in performance. Since the accuracy achieved is greater than 0.9 in all cases, this shows the adequacy of

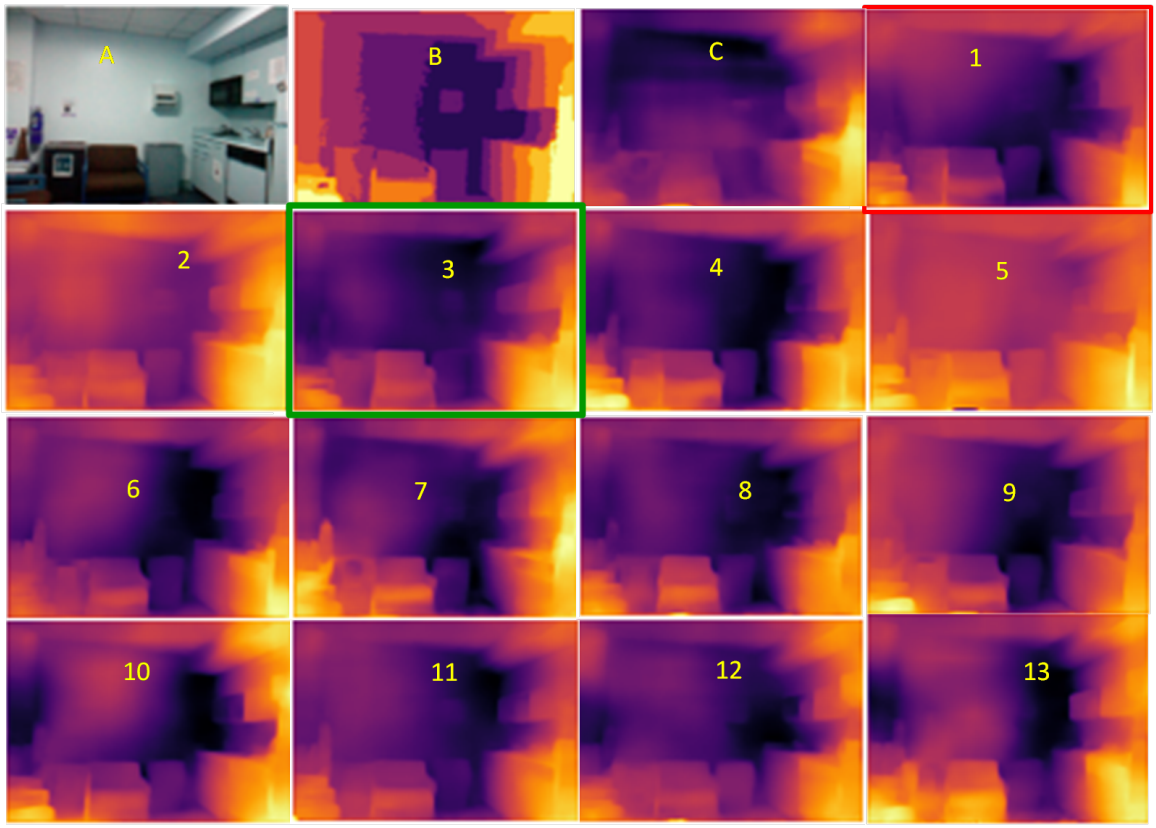


Figure 7.12: Depth map prediction after training, a visual comparison. **A**: input image, **B**: ground truth, **C**:UNET++, **1**: NDWT (3C, 3R, 3Bs) + Bs, **2**: NADWT (3C, 3LR, 1Bs), **3**: NADWT (3C, 3LR, 1Bs) + Bs, **4**: NADWT (3C, 3Bs, 3R) + Bs, **5**: NADWT (3C, 3R, 3Bs) + Bs, **6**: NRDWT (3C, 3R, 3Bs) + Bs, **7**: NRDWT (3C, 3Bs, 3R) + Bs, **8**: NARDWT (3C, 3LR, 3Bs) + Bs, **9**: NARDWT (3C, 3R, 3Bs) + Bs, **10**: NARDWT (3C, 3Bs, 3LR) + Bs, **11**: NARDWT (3C, 3Bs, 3LR), **12**: NARDWT (3C, 3LR), **13**: NARDWT (4C, 4Bs, 4LR) + 1Bs).

the training of the models. The training reaches above this value within 10 epochs, which is fast. The study also evaluates the NADWT (3C, 3LR, 1Bs) model with Haar wavelet for DWT and IWT. The performance improvement is not significant compared with db4 wavelets. Model NADWT (3C, 3LR, 1Bs)+Bs performed best with the lowest loss. The depth image in Fig. 7.12(3) and the evaluation accuracy (Fig. 7.14) support this observation.

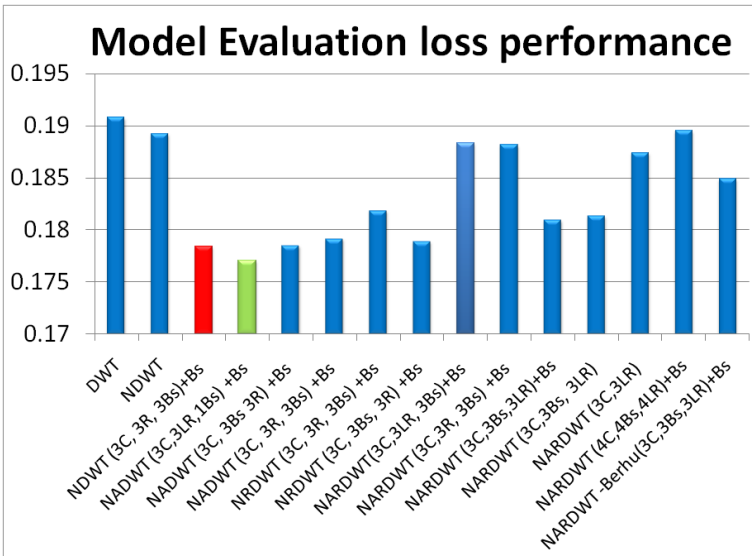


Figure 7.13: Model loss performance. The best is DWT + Attention, followed by Residual + Attention architecture.

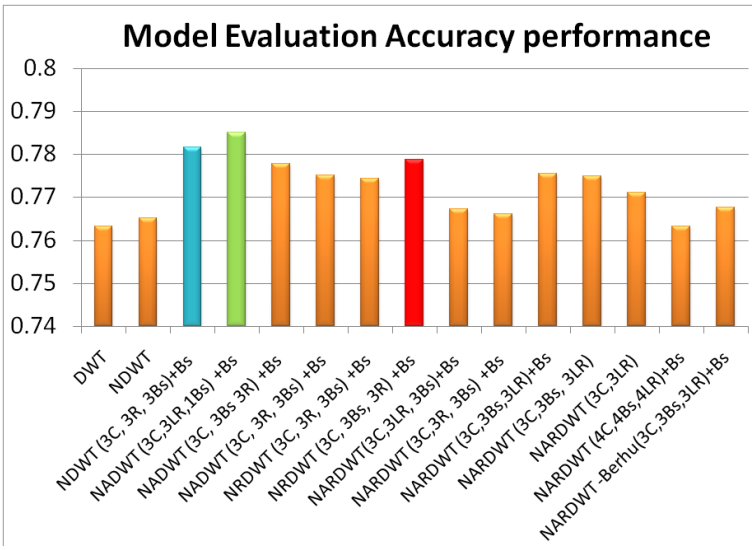


Figure 7.14: Model Evaluation Accuracy Performance

7.7 Observations

Attention model NADWT had the best overall loss performance. The base model NDWT is a close second, followed by the residual model NRDWT, as seen in Figs. 7.13 and 7.14. Among all the models, the NARDWT needs more training iterations. The NDWT model gave the best training accuracy and was followed by the NARDWT model. The NARDWT models tend to saturate faster, as seen from the validation accuracy plots (Fig. 7.18). The

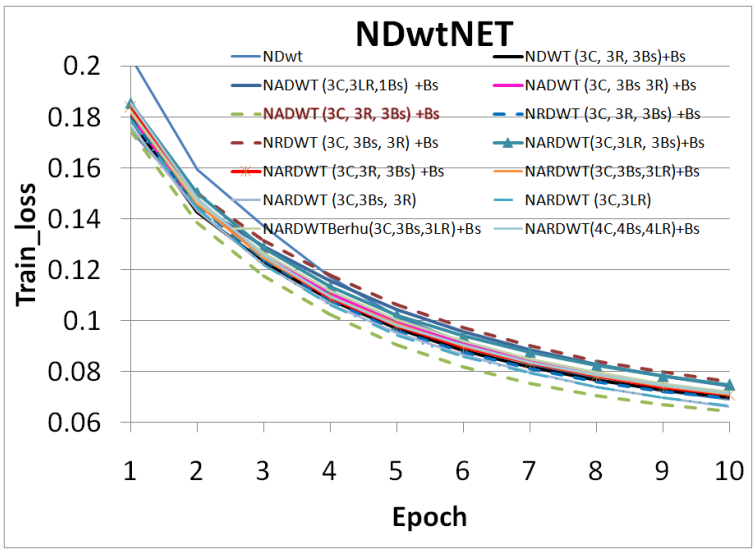


Figure 7.15: Model Training Loss Performance

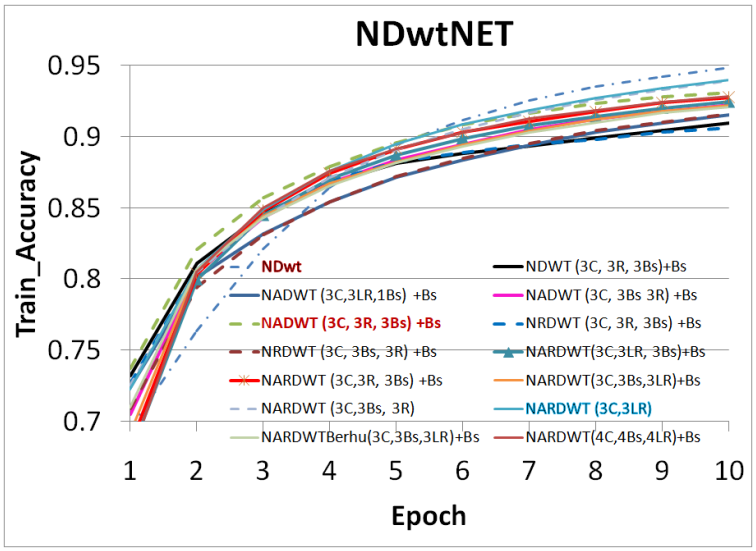


Figure 7.16: Model Training Accuracy Performance.

Haar wavelet is a 1st-order Daubechies wavelet (db1) and is discontinuous compared to db4 which is more localized and smooth. DWT is linear time complexity and is used here for dimensionality reduction similar to max pooling. Research papers have indicated that the results of mother wavelets vary depending on applications. Hence, we experimented with two types of wavelets for our database. In our applications, Haar wavelets performed slightly better with the third model. Table 7.2 tabulates the results. The performance of the best model NADWT (3C, 3LR, and 1Bs) is compared with the SOTA depth estimation methods from various researchers at the time of publication. The results are also compared

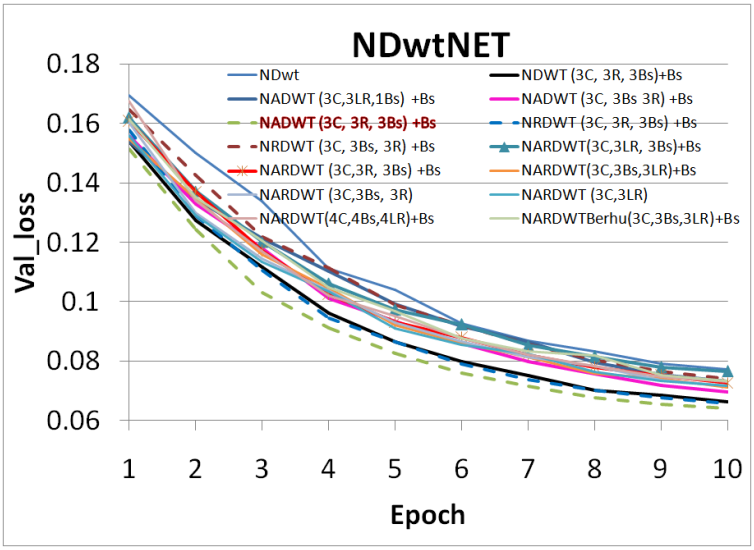


Figure 7.17: Model Validation Loss Performance

Table 7.2: Evaluation with different wavelets.

Models	$\delta_1 \uparrow$	$\delta 2 \uparrow$	$\delta 3 \uparrow$	**REL**$\downarrow$	**RMSE**$\downarrow$	**log10**$\downarrow$
Harr	0.34	0.62	0.82	0.39	0.15	0.17
db4	0.33	0.61	0.81	0.39	0.16	0.18

with published DWT-type and UNET++ models for reference. Table 7.3 compares the performance metrics of the models. Table 7.3, Figs. 7.13 and 7.14 show that the proposed models are superior w.r.t. the DWT and UNET++ models for depth prediction. Here, the proposed model outperforms these models in all six metric parameters. The reasons for such improvements are mainly dense convolution structures, the inclusion of the attention function, and regularization with batch normalization layers. Attention improved the learning of local features, thus providing more details. The batch normalization controls the covariate shift of the learned weights during training. Table 7.3 also tabulates the performance of a UNet with a DenseNet backbone encoder (DenseNet). This model has advantages like (1) encoder as a DenseNet, which is pre-trained with millions of images; (2) additional transfer training with an enhanced dataset of 50k image pairs; and (3) the model has large trainable parameters. In comparison, the study trains the proposed models from scratch with less than 400 image pairs. Still, the work achieves a lead in RMSE metrics for all tabulated SOTA model performances. The complexity of model structures like convolution density, residual functions, attention features, batch normalization density, and network blocks affects the training time. During model training, the NARDWT took approximately 17 minutes per epoch using the NYU dataset and an A100, 40 GB GPU

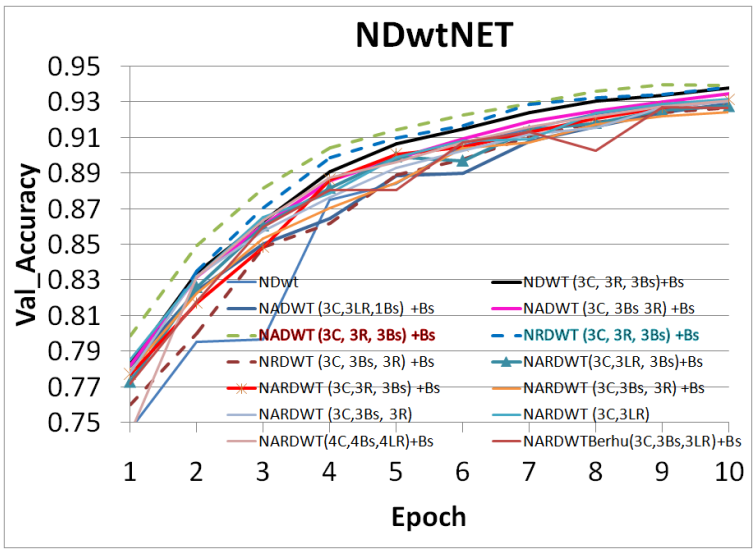


Figure 7.18: Model validation accuracy Performance

Table 7.3: Model performances.

Models	$\delta_1 \uparrow$	$\delta 2 \uparrow$	$\delta 3 \uparrow$	**REL**$\downarrow$	**RMSE**$\downarrow$	**log10**$\downarrow$	**Year**
DWT	0.27	0.52	0.73	0.54	1.76	0.21	2023*
ADWT	0.27	0.51	0.70	0.80	1.57	0.23	2023*
UNET++	0.29	0.55	0.75	0.66	1.69	0.21	2023*
DenseNet[1]	0.85	0.97	0.99	0.12	0.52	0.05	2018
DORN[97]	0.83	0.97	0.99	0.12	0.51	0.05	2018
MobileNet[84]	0.77	0.94	0.98	0.16	0.58	–	2019
MobileXNet[84]	0.79	0.95	0.988	0.15	0.54	–	2021
P3Depth[98]	0.898	0.98	0.996	0.1	0.36	0.04	2022
NewCRFs[99]	0.92	0.99	0.998	0.095	0.33	0.04	2022
ZoeD–M12– N[100]	**0.96**	**0.995**	**0.999**	**0.075**	0.27	**0.03**	2023
NADWT(3)	0.33	0.61	0.81	0.39	**0.16**	0.18	2023

*Trained with NYU dataset.

machine. For a standard T4 GPU, the training time consumed was about 97 minutes per epoch. Most of the training was completed within 10 epochs. This time was irrespective of Haar or db4 wavelet implementation. The UNet with DenseNet backbone [1], took about 150 min per epoch in the same machine, and researchers had trained for more than 20 epochs. This compares the training speed of the proposed models. In this work, UNet is modified with DWT and attention and trained along with NDWT models using the KITTI V2 dataset for 30 epochs. The dataset is used as-is without any preprocessing. The visual results are shown in Fig. 7.19. Here, UNet with DWT gives coarse depth estimates compared to NDWT. However, NDWT has errors, as seen by the discrete color patches in the image. The study observations are summarized below:

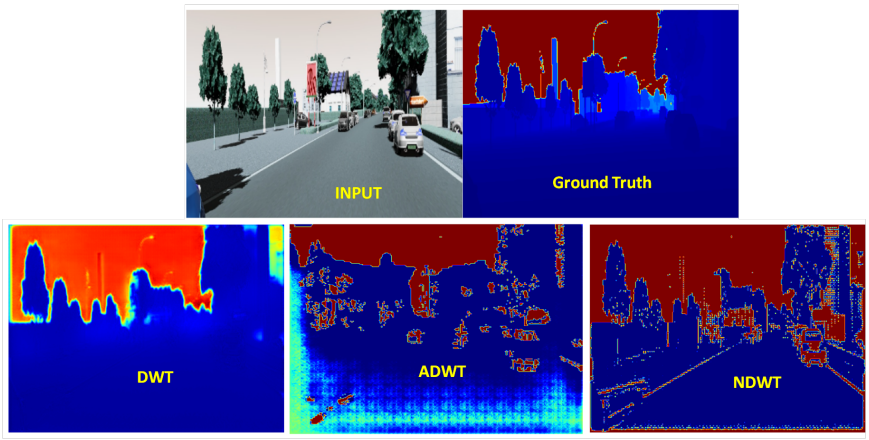


Figure 7.19: Training with the KITTI V2 dataset. Results show the visual performance of DWT (UNet with DWT), ADWT (UNet with DWT and attention), and NDWT.

- The proposed models are superior to published UNet, DWT-type UNet, and UNET++ models for all six types of performance metrics (Figs. 7.13 and 7.14). This improvement is due to dense convolution layers.
- NADWT (3C,3LR,1Bs)+Bs has the lowest loss and correlates with the depth image in Fig. 7.12(3). The model evaluation accuracy performance (Fig. 7.14) also supports this.
- The proposed light network trains faster (17 min) compared to UNet with a DenseNet backbone, which took about 150 min per epoch. The average accuracy reaches more than 92% within 10 epochs.
- The best performance is from the NDWT + Attention (NADWT) network, followed by the NDWT model, and third, the Residual (NRDWT) model (Figs. 7.13 and 7.14).
- The best training accuracy is from the NDWT model, followed by the NARDWT model (Figure 7.18). NARDWT models tend to saturate faster.
- Minor improvement is observed with Haar wavelets instead of db4 wavelets (Table 7.2).
- Batch normalization improves the depth range and loss. Batch normalization after the activation layer degrades loss This adds additional computations and trainable parameters.
- Activation: among activation layers, the LR activation function offered higher performance. Training and validation performance are better with ReLU.

- Attention: gives higher training, validation, and evaluation scores.
- Residual: gives lower training and validation accuracy, but the evaluation score is moderately better. Requires more training.
- Convolution: higher convolution layers do not improve performance but visually give better representation.
- Loss function: replacing MAE loss with BerHu loss did not show improvement.

7.8 Summary

This work shows the use of a simple network for a complex application like depth estimation from single images. The study develops a nested DWT network model to predict such depth maps. Initially, during training, the proposed network learns the DWT wavelet coefficients through a combined loss function. The components of this loss function are MAE, Berhu, SSIM, and gradient functions. The optimization-mandated study of 13 variants of networks and the experiments demonstrate that all proposed models are superior to UNet and UNet++. The proposed model has the best RMSE performance among the SOTA models. Further, the network trains within 17 minutes per epoch to achieve an average accuracy of >92% within 10 epochs. This training time is faster than other published models based on dense networks, which is an advantage for the models. This speed improvement is due to fewer trainable feature maps compared to other dense networks like DenseNet, RESNET, etc. The network structure with a Leaky ReLU followed by dense batch normalization showed improved performance. The optimum density of convolution layers in the models is three for best performance. This provides the optimal trainable feature map density for higher performance. The visual results with this optimum density are also better. As with most models, the model estimations suffer from smooth surfaces and poor illumination, as inferred from the far end of the scenes. These aspects require more analysis and study. Improving the accuracy further needs higher trainable parameters and pruning of non-performing weights in the model. This needs a trade-off study for speed and performance.

Chapter 8

Conclusions and Future work

Depth perception enhances the human-machine interface and productivity. Cameras and CV algorithms are largely used for such vision processing. Low-cost, single-camera-based systems are suited for mobile platforms due to resource constraints. Such systems eliminate elaborate setups and complex algorithms, as in stereo vision. The monocular depth estimation methods used with such systems leverage simple setups, lower costs, and lower power. This thesis surveys various literature and studies classical monocular depth estimation methods, as in Chapter 3) and in Chapter 4. Elaborate experiments were conducted with custom hardware and setups. The analysis of the results revealed that the methods are not suited for moving objects as multiple still images are required in sequence. It is also concluded that the depth estimations are ill-poised at weak patterns and that the sparse depth estimates are relative. These classical methods have good resolutions for near targets but degrade for distant targets. The focus of the work was therefore shifted to modern deep-learning methods.

In Chapter 5, a framework for a simple calibration technique to recover absolute depth from relative depth is developed using a new calibration target. The target addressed the ground truth data and the expected PSF for various depths. The two multi-scale blur targets developed related the estimated blur to ground truth through coefficients. The targets were implemented with various scales of known PSF. The coefficients of relation successfully recovered an absolute depth map from a relative depth map with good correlation. This method is immune to contrast variations within the image, magnification artifacts, and spectral sensitivity. The camera's optics and the relative depth map limit this depth range.

Researchers have successfully solved the ill-poised problems of monocular depth estimation with deep convolutional neural networks (CNN). This thesis explores various networks and concludes that training datasets, regression loss functions, and network architecture significantly impact the depth estimate quality and accuracy. The work identifies

standard performance metrics and datasets, which are popular among researchers. This enabled a comparison of performance results with those of current researchers. The selection aspects of datasets and the quality metrics were discussed in Chapter 2. Datasets need to be tuned to the application environment for best depth prediction; however, the present datasets were chosen for benchmarking.

A network minimizes the loss function during regression using backpropagation. The loss function is crucial for the training time as well as the estimated quality. Various loss functions were investigated for estimating depth. The survey inspired the development of new loss functions capable of enhancing the depth map details. An established deep-learning model was chosen for the study. This model was re-trained through transfer learning in this experiment with the new loss functions. The performances of the new loss functions are analyzed and reported in Chapter 6. The loss functions investigated were modified SSIM, reverse Huber, L1, gradient, Laplacian, LOG, and DOG-based loss functions and operators. The modified SSIM loss used a sharpened image in the SSIM loss function (SSIM'). State-of-the-art loss functions were constructed by combining multiple functions and operators such as SSIM, BerHu, Sobel, and Laplacian. These improved the feature definitions and accuracy of the depth maps. The algorithm scored higher – 85.26% for the $\delta 1$ performance parameter trained with NYU V2 benchmark datasets. The study deduces that the new edge loss functions improve depth map accuracy and range (Table-6.3, Fig. 6.4). Further, it is observed that loss functions based on derivatives and differentiation perform better than smoothing loss functions. Ablation studies here covered different network encoders and tuning of hyperparameters.

Deep learning networks for monocular depth estimations are based on complex large networks like DenseNet, LeNet, ResNet, etc. These are computationally intensive and need GPU-based hardware for long training sessions. Light networks were examined in Chapter 7. UNET-like networks are simple, robust, and traditionally used for medical analysis using small datasets. These networks have mostly been used for semantic segmentation, and only a few works exist for depth estimation. These networks influence this third thesis work. A new network, NDWTN, is conceived using a multi-scale encoder-decoder structure with skip functions. A lossless DWT-based pooling function is implemented to improve the efficiency of preserving the local features and their definitions. Other improvements are the incorporation of attention layers, skip paths, residual convolutions, and variants of activation functions, which resulted in 13 variants of the network. Experimental analysis, both qualitative and quantitative, demonstrated the effectiveness of these variant models for monocular depth estimation. These were trained with both NYU datasets and

KITTI datasets. Analysis revealed that better performance is achieved with the batch normalization regulator before the LR activation function and with the attention function in the skip path. These reduced co-variance shifts and vanishing neurons while improving training, validation, and evaluation scores. The quality metrics after training with the NYU dataset are tabulated in Table 7.3. These scores are also compared with current state-of-the-art algorithms. The designed NDWT has the best RMSE score and outperforms published DWT-type and UNET++ models for depth prediction in all six performance metrics. Further, all the models took approximately 17 minutes per epoch to train, compared to the earlier DenseNet backbone model, which took about 150 minutes per epoch with the same processing machine. The training accuracy reaches more than 92% within 10 epochs, which is fast and is due to lower trainable feature maps compared to dense networks like DenseNet, RESNET, etc. Ablation studies included wavelet variability, training parameters, and the density of convolution blocks.

8.1 Research Contributions

The thesis met the defined research goals and objectives. The outcomes of this research and experiments provide rich insight into new depth estimation techniques. These will be useful to the research community in further expanding the work done and may also open new avenues for lean networks. First, the thesis concludes that classical depth estimation methods are not suitable for mobile platforms. Most methods estimate relative depth, which inspired a new calibration method to recover absolute depth. The work further develops a framework for calibration targets and methods. These will be valuable for the scientific community. Secondly, five new efficient loss functions are conceptualized for training networks, which successfully improved the depth map estimation accuracy. These loss functions used a combination of multiple functions and operators and highlighted the need for tuning loss functions with applications. Third, a new network architecture is demonstrated that is less complex, provides faster training with a smaller database, and competes with state-of-the-art networks. The concept explores the encoder-decoder structure, dense layers, wavelets, attention, and regularization elements. Such studies are rare and open new avenues. The thesis further investigated various network and loss parameters and elements as ablation studies and experiments for optimizations.

8.2 Future Research Scopes

The thesis offers a fresh perspective on network designs and loss functions that will be beneficial for depth inference on mobile platforms. A novel calibration method to retrieve absolute depth is introduced, and additional research using calibrated settings can enhance the achieved depth range. Moreover, it encourages the investigation of new calibration goals and techniques.

There has been limited study on the loss functions required for depth inference. This study highlights the improvements when these functions are chosen or specially designed for the intended application and datasets. It is expected that the use of multiple loss functions may increase complexity, impacting training parameters and duration. There is room for further improvement in edge functions such as Canny's techniques, and several investigations are needed to optimize these.

The NDWTN network has numerous parameters that require significant time and computing resources. Some of these parameters may be unnecessary and can be removed. The work has shown that strided convolution can decrease parameter values at the cost of performance. It's common for deep learning networks to experience overfitting and vanishing neurons, which requires further investigation during pruning. Pruning involves retraining, fine-tuning filters, and quantization of weights. Additionally smaller network encoder can be studied as segmentation is not needed. Future research could focus on these areas. Additionally, there is a need to explore suitable loss functions for reducing filter weights and nodes. The majority of edge computing platforms rely on real-time inference, which mandates light networks. The challenges of deep learning networks on mobile platforms can be met by separating training and inference phases. Here, powerful remote computers can train the network while dedicated optimized hardware can make real-time inference. Specialized parallel calculations, made possible by hardware such as FPGAs and accelerators, help to achieve low-latency results. However, optimizing latency and embedding the trained model in hardware pose challenges. Model optimization, weight, and network node pruning for hardware implementation are key areas of future research.However, optimizing latency and embedding the trained model in hardware pose challenges. Model optimization, weight, and network node pruning for hardware implementation are key areas of future research. In addition, to deliver quick and accurate results, it's crucial to use suitable training datasets. Identifying and utilizing relevant datasets helps to expand the potential of the research.

Bibliography

[1] I. Alhashim and P. Wonka, "High quality monocular depth estimation via transfer learning," *arXiv e-prints*, vol. abs/1812.11941, 2018. [Online]. Available: https://arxiv.org/abs/1812.11941

[2] A. P. Pentland, "A new sense for depth of field," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 4, pp. 523–531, July 1987.

[3] G. Kreiman, *Biological and computer vision*. Cambridge University Press, 2021.

[4] U. Mudenagudi, A. Rajagopalan, and S. Chaudhuri, "Depth estimation and image restoration using defocused stereo pairs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 1521–1525, 11 2004. [Online]. Available: doi.ieeecomputersociety.org/10.1109/TPAMI.2004.102

[5] U. Shin, J. Park, and I. S. Kweon, "Deep depth estimation from thermal image," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 1043–1053.

[6] T. Laidlow, J. Czarnowski, and S. Leutenegger, "Deepfusion: Real-time dense 3d reconstruction for monocular slam using single-view depth and gradient predictions," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 4068–4074.

[7] X. Liu, A. Sinha, M. Ishii, G. D. Hager, A. Reiter, R. H. Taylor, and M. Unberath, "Dense depth estimation in monocular endoscopy with self-supervised learning methods," *IEEE Transactions on Medical Imaging*, vol. 39, no. 5, pp. 1438–1447, 2020.

[8] R. Hadsell, P. Sermanet, J. Ben, A. Erkan, M. Scoffier, K. Kavukcuoglu, U. Muller, and Y. LeCun, "Learning long-range vision for autonomous off-road driving,"

Journal of Field Robotics, vol. 26, no. 2, pp. 120–144, 2009. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20276

[9] P. R. Palafox, J. Betz, F. Nobis, K. Riedl, and M. Lienkamp, "Semanticdepth: Fusing semantic segmentation and monocular depth estimation for enabling autonomous driving in roads without lane lines," *Sensors*, vol. 19, no. 14, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/14/3224

[10] A. Saxena, S. H. Chung, and A. Ng, "3-d depth reconstruction from a single still image," *International Journal of Computer Vision*, vol. 76, pp. 53–69, 2007.

[11] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," 2014.

[12] L. Wang, Y. Wang, L. Wang, Y. Zhan, Y. Wang, and H. Lu, "Can scale-consistent monocular depth be learned in a self-supervised scale-invariant manner?" in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 12 707–12 716.

[13] V. Guizilini, I. Vasiljevic, D. Chen, R. Ambrus, and A. Gaidon, "Towards zero-shot scale-aware monocular depth estimation," 2023.

[14] T. van Dijk and G. C. Croon, "How do neural networks see depth in single images?" 2019.

[15] M. Watson and J. Enns, "Depth perception," in *Encyclopedia of Human Behavior (Second Edition)*, V. Ramachandran, Ed. San Diego: Academic Press, 2012, pp. 690–696. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780123750006001300

[16] Y. Y. Schechner and N. Kiryati, "Depth from defocus vs. stereo: how different really are they?" in *Proceedings. Fourteenth International Conference on Pattern Recognition (Cat. No.98EX170)*, vol. 2, Aug 1998, pp. 1784–1786 vol.2.

[17] S. Ullman, *The Interpretation of Visual Motion*. MIT Press, 01 1979. [Online]. Available: https://ieeexplore.ieee.org/servlet/opac?bknumber=6267349

[18] S. Suwajanakorn, C. Hernandez, and S. M. Seitz, "Depth from focus with your mobile phone," pp. 3497–3506, 2015.

[19] J. Ens and P. Lawrence, "An investigation of methods for determining depth from focus," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 2, pp. 97–108, 1993.

[20] A. Levin, D. Lischinski, and Y. Weiss, "A closed-form solution to natural image matting," pp. 228–242, 2008.

[21] M. Subbarao and G. Surya, "Depth from defocus: A spatial domain approach," *Int. J. Comput. Vision*, vol. 13, no. 3, p. 271–294, Dec. 1994. [Online]. Available: https://doi.org/10.1007/BF02028349

[22] V. Srikakulapu, H. Kumar, S. Gupta, and K. S. Venkatesh, "Depth estimation from single image using defocus and texture cues," in *2015 Fifth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG)*, 2015, pp. 1–4.

[23] T. Xian and M. Subbarao, "Performance evaluation of different depth from defocus (dfd) techniques," in *SPIE*, vol. 6000, 2005, pp. 87–99. [Online]. Available: https://doi.org/10.1117/12.629611

[24] Q. Dou and P. Favaro, "Off-axis aperture camera: 3d shape reconstruction and image restoration," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, June 2008, pp. 1–7.

[25] G. Surya and M. Subbarao, "Depth from defocus by changing camera aperture: A spatial domain approach," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 11 1999.

[26] Y. Amari and E. H. Adelson, "Single-eye range estimation by using displaced apertures with color filters," in *Proceedings of the 1992 International Conference on Industrial Electronics, Control, Instrumentation, and Automation*, Nov 1992, pp. 1588–1592 vol.3.

[27] V. Paramonov, I. Panchenko, V. Bucha, A. Drogolyub, and S. Zagoruyko, "Depth camera based on color-coded aperture," in *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016, pp. 910–918.

[28] S. Lee, M. H. Hayes, and J. Paik, "Distance estimation using a single computational camera with dual off-axis color filtered apertures," *Opt. Express*,

vol. 21, no. 20, pp. 23 116–23 129, Oct 2013. [Online]. Available: http://www.opticsexpress.org/abstract.cfm?URI=oe-21-20-23116

[29] S. Lee, N. Kim, K. Jung, M. H. Hayes, and J. Paik, "Single image-based depth estimation using dual off-axis color filtered aperture camera," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013, pp. 2247–2251.

[30] J. H. Elder and S. W. Zucker, "Local scale control for edge detection and blur estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 7, pp. 699–716, Jul 1998.

[31] S. Zhuo and T. Sim, "Defocus map estimation from a single image," pp. 1852–1858, 2011.

[32] A. Gershun, "The light field," *Journal of Mathematics and Physics*, vol. 18, no. 1-4, pp. 51–151, 1939. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/sapm193918151

[33] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '96. New York, NY, USA: Association for Computing Machinery, 1996, p. 31–42. [Online]. Available: https://doi.org/10.1145/237170.237199

[34] E. Adelson and J. Wang, "Single lens stereo with a plenoptic camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 99–106, 1992.

[35] R. Ng, M. Levoy, M. Br, G. Duval, M. Horowitz, P. Hanrahan, and D. Design, "Light field photography with a hand-held plenoptic camera," in *Stanford Computer Graphics Lab's archive of technical publications*, 2005.

[36] A. Veeraraghavan, R. Raskar, A. Agrawal, A. Mohan, and J. Tumblin, "Dappled photography: Mask enhanced cameras for heterodyned light fields and coded aperture refocusing," *ACM Trans. Graph.*, vol. 26, no. 3, p. 69–es, jul 2007. [Online]. Available: https://doi.org/10.1145/1276377.1276463

[37] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," 2016.

[38] K. Harsányi, A. Kiss, A. Majdik, and T. Sziranyi, "A hybrid cnn approach for single image depth estimation: A case study," in *Multimedia and Network Information Systems*, K. Choroś, M. Kopel, E. Kukla, and A. Siemiński, Eds. Cham: Springer International Publishing, 2019, pp. 372–381.

[39] S. S. Shivakumar, T. Nguyen, I. D. Miller, S. W. Chen, V. Kumar, and C. J. Taylor, "Dfusenet: Deep fusion of rgb and sparse depth information for image guided dense depth completion," 2019.

[40] S. Anwar, Z. Hayder, and F. Porikli, "Depth estimation and blur removal from a single out-of-focus image," *British Machine Vision Conference*, 2017.

[41] S. Gur and L. Wolf, "Single image depth estimation trained via depth from defocus cues," 2020.

[42] C. Zhao, Q. Sun, C. Zhang, Y. Tang, and F. Qian, "Monocular depth estimation based on deep learning: An overview," *Science China Technological Sciences*, vol. 63, no. 9, p. 1612–1627, Jun 2020. [Online]. Available: http://dx.doi.org/10.1007/s11431-020-1582-8

[43] L. He, G. Wang, and Z. Hu, "Learning depth from single images with deep neural network embedding focal length," *IEEE Transactions on Image Processing*, vol. 27, no. 9, p. 4676–4689, Sep 2018. [Online]. Available: http://dx.doi.org/10.1109/TIP.2018.2832296

[44] J. Chi, J. Gao, L. Qi, S. Zhang, J. Dong, and H. Yu, "Depth estimation of a single RGB image with semi-supervised two-stage regression," in *Proceedings of the 5th International Conference on Communication and Information Processing*. ACM, nov 2019. [Online]. Available: https://doi.org/10.11452F3369985.3370004

[45] N. Opatovski, D. Xiao, G. Harari, and Y. Shechtman, "Monocular kilometer-scale passive ranging by point-spread function engineering," *Opt. Express*, vol. 30, no. 21, pp. 37 925–37 937, Oct 2022. [Online]. Available: https://opg.optica.org/oe/abstract.cfm?URI=oe-30-21-37925

[46] D. Xu, E. Ricci, W. Ouyang, X. Wang, and N. Sebe, "Multi-scale continuous crfs as sequential deep networks for monocular depth estimation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 161–169.

[47] Z. Hao, Y. Li, S. You, and F. Lu, "Detail preserving depth estimation from a single image using attention guided networks," in *2018 International Conference on 3D Vision (3DV)*, 2018, pp. 304–313.

[48] A. Masoumian, H. A. Rashwan, J. Cristiano, M. S. Asif, and D. Puig, "Monocular depth estimation using deep learning: A review," *Sensors*, vol. 22, no. 14, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/14/5353

[49] J. Zhu, L. Liu, Y. Liu, W. Li, F. Wen, and H. Zhang, "Fg-depth: Flow-guided unsupervised monocular depth estimation," 2023. [Online]. Available: https://arxiv.org/abs/2301.08414

[50] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," 2016. [Online]. Available: https://arxiv.org/abs/1609.03677

[51] S. F. Bhat, I. Alhashim, and P. Wonka, "AdaBins: Depth estimation using adaptive bins," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2021. [Online]. Available: https://doi.org/10.1109$%$2Fcvpr46437.2021.00400

[52] B. Z. H. W. Z. L. C. Y. H. H. L. Li, "Unsupervised monocular depth estimation with aggregating image features and wavelet ssim (structural similarity) loss," pp. 84–98, 2021. [Online]. Available: http://dx.doi.org/10.20517/ir.2021.06

[53] S. Zhao, H. Fu, M. Gong, and D. Tao, "Geometry-aware symmetric domain adaptation for monocular depth estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[54] R. J. Woodham, "Photometric Method For Determining Surface Orientation From Multiple Images," *Optical Engineering*, vol. 19, no. 1, p. 191139, 1980. [Online]. Available: https://doi.org/10.1117/12.7972479

[55] G. Chen, K. Han, and K.-Y. K. Wong, "Ps-fcn: A flexible learning framework for photometric stereo," 2018. [Online]. Available: https://arxiv.org/abs/1807.08696

[56] G. Chen, K. Han, B. Shi, Y. Matsushita, and K.-Y. K. Wong, "Deep photometric stereo for non-lambertian surfaces," 2020. [Online]. Available: https://arxiv.org/abs/2007.13145

[57] Y. Ju, M. Jian, S. Guo, Y. Wang, H. Zhou, and J. Dong, "Incorporating lambertian priors into surface normals measurement," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2021.

[58] D. Antensteiner, S. Štolc, and R. Huber-Mörk, "Depth estimation with light field and photometric stereo data using energy minimization," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, C. Beltrán-Castañón, I. Nyström, and F. Famili, Eds. Cham: Springer International Publishing, 2017, pp. 175–183.

[59] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[60] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012.

[61] J. S. Lubor Ladicky and M. Pollefeys, "Pulling things out of perspective," in *CVPR '14: Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*. USA: IEEE Computer Society, 2014, p. 89–96.

[62] E. P. Örnek, S. Mudgal, J. Wald, Y. Wang, N. Navab, and F. Tombari, "From 2d to 3d: Re-thinking benchmarking of monocular depth prediction," 2022.

[63] M. Ramamonjisoa, Y. Du, and V. Lepetit, "Predicting sharp and accurate occlusion boundaries in monocular depth estimation using displacement fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[64] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1253–1260.

[65] Y. Wang, "Mobiledepth: Efficient monocular depth prediction on mobile devices," 2020.

[66] H. Si, B. Zhao, D. Wang, Y. Gao, M. Chen, Z. Wang, and X. Li, "Fully self-supervised depth estimation from defocus clue," 2023.

[67] A. Levin, "Analyzing depth from coded aperture sets," in *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part I 11*. Springer, 2010, pp. 214–227.

[68] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," 2018.

[69] R. Tsai, "A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses," *IEEE Journal on Robotics and Automation*, vol. 3, no. 4, pp. 323–344, 1987.

[70] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[71] H. Yue, J. Zhang, X. Wu, J. Wang, and W. Chen, "Edge enhancement in monocular depth prediction," in *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, 2020, pp. 1594–1599.

[72] J. Xie, R. S. Feris, and M.-T. Sun, "Edge-guided single depth image super resolution," *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 428–438, 2016.

[73] C. Zhang and Y. Tian, "Edge enhanced depth motion map for dynamic hand gesture recognition," in *2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2013, pp. 500–505.

[74] K. Janocha and W. M. Czarnecki, "On loss functions for deep neural networks in classification," 2017.

[75] S. Paul, B. Jhamb, D. Mishra, and M. S. Kumar, "Edge loss functions for deep-learning depth-map," *Machine Learning with Applications*, vol. 7, p. 100218, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666827021001092

[76] T. Williams and R. Li, "Wavelet pooling for convolutional neural networks," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rkhlb8lCZ

[77] A. Ferrà, E. Aguilar, and P. Radeva, "Multiple wavelet pooling for cnns," in *Computer Vision – ECCV 2018 Workshops*, L. Leal-Taixé and S. Roth, Eds. Cham: Springer International Publishing, 2019, pp. 671–675.

[78] H.-H. Yang, C.-H. H. Yang, and Y.-C. James Tsai, "Y-net: Multi-scale feature aggregation network with wavelet structure similarity loss function for single image dehazing," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 2628–2632.

[79] M. Ramamonjisoa, M. Firman, J. Watson, V. Lepetit, and D. Turmukhambetov, "Single image depth estimation using wavelet decomposition," *CoRR*, vol. abs/2106.02022, 2021. [Online]. Available: https://arxiv.org/abs/2106.02022

[80] P. F. Olaf Ronneberger and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention, MICCAI 2015*. Cham: Springer International Publishing, 2015, pp. 234–241.

[81] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz, B. Glocker, and D. Rueckert, "Attention u-net: Learning where to look for the pancreas," 2018.

[82] Z. Zhang, Q. Liu, and Y. Wang, "Road extraction by deep residual u-net," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 5, pp. 749–753, 2018.

[83] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017. [Online]. Available: http://arxiv.org/abs/1704.04861

[84] X. Dong, M. A. Garratt, S. G. Anavatti, and H. A. Abbass, "Mobilexnet: An efficient convolutional neural network for monocular depth estimation," *CoRR*, vol. abs/2111.12334, 2021. [Online]. Available: https://arxiv.org/abs/2111.12334

[85] Z. Yang, P. Wang, W. Xu, L. Zhao, and R. Nevatia, "Unsupervised learning of geometry with edge-aware depth-normal consistency," 2017.

[86] Y. Wang, X. Zhu, Y. Zhao, P. Wang, and J. Ma, "Enhancement of low-light image based on wavelet u-net," *Journal of Physics: Conference Series*, vol. 1345, p. 022030, nov 2019. [Online]. Available: https://doi.org/10.1088/1742-6596/1345/2/022030

[87] Y. Li, Y. Wang, T. Leng, and W. Zhijie, "Wavelet u-net for medical image segmentation," in *Artificial Neural Networks and Machine Learning – ICANN 2020:*

29th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 15–18, 2020, Proceedings, Part I. Berlin, Heidelberg: Springer-Verlag, 2020, p. 800–810. [Online]. Available: https://doi.org/10.1007/978-3-030-61609-0_63

[88] J. L. Chuter, G. B. Boullanger, and M. N. Saez, "U-n. o . 1 t: A u-net exploration, in depth," in *Computer Science*, 2018.

[89] M. Sharma, A. Sharma, K. R. Tushar, and A. Panneer, "A novel 3d-unet deep learning framework based on high-dimensional bilateral grid for edge consistent single image depth estimation," in *2020 International Conference on 3D Immersion (IC3D)*, 2020, pp. 01–08.

[90] D. Peng, Y. Zhang, and H. Guan, "End-to-end change detection for high resolution satellite images using improved unet++," *Remote. Sens.*, vol. 11, p. 1382, 2019.

[91] B. Yu, J. Wu, and M. J. Islam, "Udepth: Fast monocular depth estimation for visually-guided underwater robots," in *ccepted at the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023.

[92] N. Zioulis, G. Albanis, P. Drakoulis, F. Alvarez, D. Zarpalas, and P. Daras, "Hybrid skip: A biologically inspired skip connection for the UNet architecture," *IEEE Access*, vol. 10, pp. 53 928–53 939, 2022. [Online]. Available: https://doi.org/10.1109$%$2Faccess.2022.3175864

[93] C. Luo, Y. Li, K. Lin, G. Chen, S.-J. Lee, J. Choi, Y. F. Yoo, and M. O. Polley, "Wavelet synthesis net for disparity estimation to synthesize dslr calibre bokeh effect on smartphones," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2404–2412.

[94] Q. Li, L. Shen, S. Guo, and Z. Lai, "Wavelet integrated cnns for noise-robust image classification," 2020. [Online]. Available: https://arxiv.org/abs/2005.03337

[95] P. Liu, H. Zhang, W. Lian, and W. Zuo, "Multi-level wavelet convolutional neural networks," *CoRR*, vol. abs/1907.03128, 2019. [Online]. Available: http://arxiv.org/abs/1907.03128

[96] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *ArXiv*, vol. abs/1502.03167, 2015.

[97] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," 2018. [Online]. Available: https://arxiv.org/abs/1806.02446

[98] V. Patil, C. Sakaridis, A. Liniger, and L. Van Gool, "P3depth: Monocular depth estimation with a piecewise planarity prior," 2022. [Online]. Available: https://arxiv.org/abs/2204.02091

[99] W. Yuan, X. Gu, Z. Dai, S. Zhu, and P. Tan, "New crfs: Neural window fully-connected crfs for monocular depth estimation," 2022. [Online]. Available: https://arxiv.org/abs/2203.01502

[100] S. F. Bhat, R. Birkl, D. Wofk, P. Wonka, and M. Müller, "Zoedepth: Zero-shot transfer by combining relative and metric depth," 2023. [Online]. Available: https://arxiv.org/abs/2302.12288

[101] M. P. A., "The power of shadows: shadow stereopsis," *Journal of the Optical Society of America. A, Optics and image science, 6(2), 309–311*, vol. 6.2, p. 309–311, 1989. [Online]. Available: https://doi.org/10.1364/josaa.6.000309

[102] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," 2016. [Online]. Available: https://arxiv.org/abs/1604.01685

[103] I. Vasiljevic, N. Kolkin, S. Zhang, R. Luo, H. Wang, F. Z. Dai, A. F. Daniele, M. Mostajabi, S. Basart, M. R. Walter, and G. Shakhnarovich, "DIODE: A Dense Indoor and Outdoor DEpth Dataset," *CoRR*, vol. abs/1908.00463, 2019. [Online]. Available: http://arxiv.org/abs/1908.00463

[104] J. Spencer, C. S. Qian, C. Russell, S. Hadfield, E. Graf, W. Adams, A. J. Schofield, J. Elder, R. Bowden, H. Cong, S. Mattoccia, M. Poggi, Z. K. Suri, Y. Tang, F. Tosi, H. Wang, Y. Zhang, Y. Zhang, and C. Zhao, "The monocular depth estimation challenge," 2022.

[105] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *ECCV (1)*, 2008, pp. 44–57.

[106] X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, "The ApolloScape open dataset for autonomous driving and its application," *IEEE Transactions on*

Pattern Analysis and Machine Intelligence, vol. 42, no. 10, pp. 2702–2719, oct 2020. [Online]. Available: https://doi.org/10.1109$%$2Ftpami.2019.2926463

[107] S. Song, S. P. Lichtenberg, and J. Xiao, "Sun rgb-d: A rgb-d scene understanding benchmark suite," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 567–576.

[108] J. Xiao, A. Owens, and A. Torralba, "Sun3d: A database of big spaces reconstructed using sfm and object labels," in *2013 IEEE International Conference on Computer Vision*, 2013, pp. 1625–1632.

[109] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[110] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3d: Learning from rgb-d data in indoor environments," *International Conference on 3D Vision (3DV)*, 2017.

[111] D. Scharstein, R. Szeliski, and R. Zabih, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," in *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*, 2001, pp. 131–140.

[112] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 573–580.

[113] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 824–840, 2009.

[114] I. Armeni, S. Sax, A. R. Zamir, and S. Savarese, "Joint 2d-3d-semantic data for indoor scene understanding," *arXiv preprint arXiv:1702.01105*, 2017.

[115] Q.-H. Pham, M. A. Uy, B.-S. Hua, D. T. Nguyen, G. Roig, and S.-K. Yeung, "LCD: Learned cross-domain descriptors for 2D-3D matching," in *AAAI Conference on Artificial Intelligence*, 2020.

[116] A. R. Zamir, A. Sax, W. B. Shen, L. J. Guibas, J. Malik, and S. Savarese, "Taskonomy: Disentangling task transfer learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018.

[117] T. Schöps, J. L. Schönberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger, "A multi-view stereo benchmark with high-resolution images and multi-camera videos," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[118] G. Yang, X. Song, C. Huang, Z. Deng, J. Shi, and B. Zhou, "Drivingstereo: A large-scale dataset for stereo matching in autonomous driving scenarios," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[119] V. Guizilini, R. Ambrus, S. Pillai, A. Raventos, and A. Gaidon, "3d packing for self-supervised monocular depth estimation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[120] A. Wong, X. Fei, S. Tsuei, and S. Soatto, "Unsupervised depth completion from visual inertial odometry," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1899–1906, 2020.

[121] M. L. y and P. Hanrahan, "Light field rendering," 1996. [Online]. Available: https://graphics.stanford.edu/papers/light/

[122] A. Chatzitofis, L. Saroglou, P. Boutis, P. Drakoulis, N. Zioulis, S. Subramanyam, B. Kevelham, C. Charbonnier, P. Cesar, D. Zarpalas, S. Kollias, and P. Daras, "Human4d: A human-centric multimodal dataset for motions and immersive media," *IEEE Access*, vol. 8, pp. 176 241–176 262, 2020.

[123] Z. Bauer, F. Gomez-Donoso, E. Cruz, S. Orts-Escolano, and M. Cazorla, "Uasol, a large-scale high-resolution outdoor stereo dataset," *Scientific Data*, vol. 6, no. 1, pp. 1–14, 2019.

[124] L. Li, K. Ismail, H. Shum, and T. Breckon, "Durlar: A high-fidelity 128-channel lidar dataset with panoramic ambient and reflectivity imagery for multi-modal autonomous driving applications," in *Proc. Int. Conf. on 3D Vision*. IEEE, December 2021.

[125] A. Sharma and J. Ventura, "Unsupervised learning of depth and ego-motion from cylindrical panoramic video," in *2019 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*. Los Alamitos, CA, USA: IEEE Computer Society, dec 2019, pp. 58–587. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/AIVR46125.2019.00018

[126] G. Albanis, N. Zioulis, P. Drakoulis, V. Gkitsas, V. Sterzentsenko, F. Alvarez, D. Zarpalas, and P. Daras, "Pano3d: A holistic benchmark and a solid baseline for 360° depth estimation," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021, pp. 3722–3732.

[127] H. Hu, B. Yang, Z. Qiao, D. Zhao, and H. Wang, "Seasondepth: Cross-season monocular depth prediction dataset and benchmark under multiple environments," *arXiv preprint arXiv:2011.04408*, 2022.

[128] X. Wang, Z. Zhu, Y. Zhang, G. Huang, Y. Ye, W. Xu, Z. Chen, and X. Wang, "Are we ready for vision-centric driving streaming perception? the asap benchmark," *arXiv preprint arXiv:2212.08914*, 2022.

[129] M. Carvalho, B. Le Saux, P. Trouvé-Peloux, A. Almansa, and F. Champagnat, "Deep depth from defocus: how can defocus blur improve 3D estimation using dense neural networks?" *3DRW ECCV Workshop*, 2018.

[130] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," *Proceedings of 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[131] J. McCormac, A. Handa, S. Leutenegger, and A. J. Davison, "Scenenet rgb-d: 5m photorealistic images of synthetic indoor trajectories with ground truth," 2016. [Online]. Available: https://arxiv.org/abs/1612.05079

[132] L. Teixeira, M. R. Oswald, M. Pollefeys, and M. Chli, "Aerial single-view depth completion with image-guided uncertainty estimation," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1055–1062, 2020.

[133] M. Roberts, J. Ramapuram, A. Ranjan, A. Kumar, M. A. Bautista, N. Paczan, R. Webb, and J. M. Susskind, "Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding," in *International Conference on Computer Vision (ICCV) 2021*, 2021.

[134] D. G. Javier Hidalgo-Carrio and D. Scaramuzza, "Learning monocular dense depth from events," *IEEE International Conference on 3D Vision.(3DV)*, 2020. [Online]. Available: http://rpg.ifi.uzh.ch/docs/3DV20_Hidalgo.pdf

[135] N. Zioulis, A. Karakottas, D. Zarpalas, and P. Daras, "Omnidepth: Dense depth estimation for indoors spherical panoramas," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 448–465.

[136] A. Mahendran, H. Bilen, J. F. Henriques, and A. Vedaldi, "Researchdoom and cocodoom: Learning computer vision with games," 2016. [Online]. Available: https://arxiv.org/abs/1610.02431

[137] S. Niklaus, L. Mai, J. Yang, and F. Liu, "3d ken burns effect from a single image," *ACM Transactions on Graphics*, vol. 38, no. 6, pp. 184:1–184:15, 2019.

[138] K. Honauer, O. Johannsen, D. Kondermann, and B. Goldluecke, "A dataset and evaluation methodology for depth estimation on 4d light fields," in *Computer Vision – ACCV 2016*, S.-H. Lai, V. Lepetit, K. Nishino, and Y. Sato, Eds. Cham: Springer International Publishing, 2017, pp. 19–34.

[139] K. B. Ozyoruk, G. I. Gokceler, G. Coskun, K. Incetan, Y. Almalioglu, F. Mahmood, E. Curto, L. Perdigoto, M. Oliveira, H. Sahin, H. Araujo, H. Alexandrino, N. J. Durr, H. B. Gilbert, and M. Turan, "Endoslam dataset and an unsupervised monocular visual odometry and depth estimation approach for endoscopic videos: Endosfmlearner," 2020.

[140] V. Schmidt, A. Luccioni, M. Teng, T. Zhang, A. Reynaud, S. Raghupathi, G. Cosne, A. Juraver, V. Vardanyan, A. Hernández-García, and Y. Bengio, "ClimateGAN: Raising climate change awareness by generating images of floods," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=EZNOb_uNpJk

[141] H. Le, P. Das, T. Mensink, S. Karaoglu, and T. Gevers, "EDEN: Multimodal Synthetic Dataset of Enclosed garDEN Scenes," in *Proceedings of the IEEE/CVF Winter Conference of Applications on Computer Vision (WACV)*, 2021.

[142] Z. Li, L. Wang, X. Huang, C. Pan, and J. Yang, "Phyir: Physics-based inverse rendering for panoramic indoor images," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[143] J. Shi, X. Jiang, and C. Guillemot, "A framework for learning depth from a flexible subset of dense and sparse light field views," *IEEE Transactions on Image Processing*, vol. 28, no. 12, pp. 5867–5880, 2019.

[144] L. T. Alessandro Bergamo, "Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach," in *Neural Information Processing Systems (NIPS)*, Dec. 2010. [Online]. Available: http://vlg.cs.dartmouth.edu/projects/domainadapt/

[145] Z. Li and N. Snavely, "Megadepth: Learning single-view depth prediction from internet photos," in *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[146] Z. Li, T. Dekel, F. Cole, R. Tucker, N. Snavely, C. Liu, and W. T. Freeman, "Learning the depths of moving people by watching frozen people," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[147] K. Xian, C. Shen, Z. Cao, H. Lu, Y. Xiao, R. Li, and Z. Luo, "Monocular relative depth perception with web stereo data supervision," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[148] W. Chen, Z. Fu, D. Yang, and J. Deng, "Single-image depth perception in the wild," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016. [Online]. Available: https://proceedings.neurips.cc/paper/2016/file/0deb1c54814305ca9ad266f53bc82511-Paper.pdf

[149] W. Chen, S. Qian, D. Fan, N. Kojima, M. Hamilton, and J. Deng, "Oasis: A large-scale dataset for single image 3d in the wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[150] Y. Hua, P. Kohli, P. Uplavikar, A. Ravi, S. Gunaseelan, J. Orozco, and E. Li, "Holopix50k: A large-scale in-the-wild stereo image dataset," 2020. [Online]. Available: https://arxiv.org/abs/2003.11172

[151] C. Wang, S. Lucey, F. Perazzi, and O. Wang, "Web stereo video supervision for depth prediction from dynamic scenes," in *2019 International Conference on 3D Vision (3DV)*, 2019, pp. 348–357.

[152] K. Xian, J. Zhang, O. Wang, L. Mai, Z. Lin, and Z. Cao, "Structure-guided ranking loss for single image depth prediction," in *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[153] C. Guérin, C. Rigaud, A. Mercier, F. Ammar-Boudjelal, K. Bertet, A. Bouju, J.-C. Burie, G. Louis, J.-M. Ogier, and A. Revel, "ebdtheque: A representative database of

comics," in *2013 12th International Conference on Document Analysis and Recognition*, 2013, pp. 1145–1149.

[154] N.-V. Nguyen, C. Rigaud, and J.-C. Burie, "Digital comics image indexing based on deep learning," *Journal of Imaging*, vol. 4, no. 7, 2018. [Online]. Available: http://www.mdpi.com/2313-433X/4/7/89

[155] H. Lee and J. Park, "Instance-wise Occlusion and Depth Orders in Natural Scenes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[156] Y. Jafarian and H. S. Park, "Learning high fidelity depths of dressed humans by watching social media dance videos," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 12 748–12 757.

[157] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

List of Publications

Journals

1. **Sandip Paul**, Bhuvan Jhamb, Deepak Mishra, M. Senthil Kumar, 2022. Edge loss functions for deep-learning depth map. *Machine Learning with Applications.* (Volume **7**, pp 100218, ISSN 2666-8270) https://doi.org/10.1016/j.mlwa.2021.100218. (https://www.sciencedirect.com/science/article/pii/S2666827021001092).

2. **Paul, Sandip**, Deepak Mishra, and Senthil Kumar Marimuthu. 2023. Nested DWT–Based CNN Architecture for Monocular Depth Estimation. *Sensors* (Volume **23**, No. 6). https://doi.org/10.3390/s23063066.

Conferences

1. **Paul, Sandip**, Mishra, Deepak, Senthil, M. (2020). Calibration of Depth Map Using a Novel Target. *Computer Vision and Image Processing 2019. Communications in Computer and Information Science* (Vol **1147**). Springer, Singapore. https://doi.org/10.1007/978-981-15-4015-8_22.

1. **Paul, Sandip**, Mishra, Deepak, Gadhia Jimit J., Paul Anirban (2024). AI/ML for onboard Remote Sensing. *9th National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIG-2024)*. IIST, Thiruvananthapuram.

Others

1. Summer School on Computer Vision, IIIT Hyderabad (3rd - 08th, July 2017). Participant

Chapter 9

Appendix A: Camera parameters

9.1 Computation methods

Diagonal

Diagonal is calculated by the use of the Pythagorean theorem:

$$Diagonal = \sqrt{w^2 + h^2} \tag{9.1}$$

where, $w = sensor\ width$ and $h = sensor\ height$ **Canon 400D diagonal:**
w = 22.20 mm
h = 14.80 mm

$$Diagonal = \sqrt{22.20^2 + 14.80^2} = 26.68\,mm \tag{9.2}$$

Surface area

Surface area is calculated by multiplying the width and height of a sensor.
Width = 22.20 mm
Height = 14.80 mm

$$Surface\ area = 22.20 * 14.80 = 328.56\,mm^2 \tag{9.3}$$

Pixel pitch

Pixel pitch is the distance from the center of one pixel to the center of the next measured in micrometers (μm). It can be calculated with the following formula:

$$Pixel\ pitch = \frac{sensor\ width\ in\ mm}{sensor\ resolution\ width\ in\ pixels} * 1000 \tag{9.4}$$

Canon 400D pixel pitch:

Sensor width = 22.20 mm

Sensor resolution width = 3893 pixels

$$Pixel\ pitch = \frac{22.20}{3893} * 1000 = 5.7\ \mu m \tag{9.5}$$

Pixel area

The area of one pixel can be calculated by simply squaring the pixel pitch:

$$Pixel\ area = pixel\ pitch^2 \tag{9.6}$$

You could also divide the sensor surface area with effective megapixels:

$$Pixel\ area = \frac{sensor\ surface\ area\ in\ mm^2}{effective\ megapixels} \tag{9.7}$$

Canon 400D pixel area:

Pixel pitch = 5.7 μm

$$Pixel\ area = 5.7^2 = 32.49\ \mu m^2 \tag{9.8}$$

Pixel density

Pixel density can be calculated with the following formula:

$$Pixel\ density = \frac{sensor\ resolution\ width\ in\ pixels}{(sensor\ width\ in\ cm)^2} * \frac{1}{1000000} \tag{9.9}$$

You could also use this formula:

$$Pixel\ density = \frac{effective\ megapixels * 1000000}{sensor\ surface\ area\ in\ mm^2} * \frac{1}{10000} \tag{9.10}$$

Canon 400D pixel density:

Sensor resolution width = 3893 pixels

Sensor width = 2.22 cm

$$Pixel\ density = \frac{(3893/2.22)^2}{1000000} = 3.08 MP/cm^2 \tag{9.11}$$

9.2 Canon EOS 400D Camera

Sensor size numbers are quite accurate so you can rely on them. Pixel–related numbers have more limitations, so take them as approximations (pixel pitch is usually within 2% accurate). The actual size is set to a 14.8" screen. This is the actual size of the 400D sensor: 22.2 x 14.8 mm. The sensor has a surface area of $328.6mm^2$. There are approx. 10,100,000 photosites (pixels) in this area. Pixel pitch, a measure of the distance between pixels, is 5.7 μm. Pixel pitch tells you the distance from the center of one pixel (photosite) to the center of the next. The pixel or photosite area is 32.49 μm^2. The larger the photosite, the more light it can capture and the more information can be recorded. Pixel density tells you how many million pixels fit or would fit in one square cm of the sensor. Canon 400D has a pixel density of 3.08 MP/cm^2. These numbers are important in terms of assessing the overall quality of a digital camera. Generally, the bigger (and newer) the sensor, pixel pitch, and photosite area; the smaller the pixel density, the better the camera.

9.2.1 Specifications of Camera

Brand: Canon
Model: EOS 400D
Also known as EOS Digital Rebel XTi (US), EOS Kiss Digital X (Japan)
Effective megapixels: 10.10
Total megapixels: 10.50
Sensor size: 22.2 x 14.8 mm
Sensor type: CMOS
Sensor resolution: 3893 x 2595
Max. image resolution: 3888 x 2592
Crop factor: 1.62
Optical zoom:
Digital zoom: No
ISO: Auto, 100, 200, 400, 800, 1600
RAW support:
Manual focus: y
Normal focus range:
Macro focus range:
Focal length (35mm equiv.):

Aperture priority: Yes
Max aperture:
Max. aperture (35mm equiv.): n/a
Depth of field: simulate
Metering: Center-weighted, Matrix, Spot
Exposure Compensation: ±2 EV (in 1/3 EV steps)
Shutter priority: Yes
Min. shutter speed: Bulb+30 sec
Max. shutter speed: 1/4000 sec
Built-in flash: y
External flash: y
Viewfinder: Optical (pentamirror)
White balance presets: 6
Screen size: 2.5"
Screen resolution: 230,000 dots
Video capture:
Storage types: CompactFlash type I, CompactFlash type II, Microdrive
USB: USB 2.0 (480 Mbit/sec)
HDMI:
Wireless:
GPS:
Battery: AA (2) batteries (NiMH recommended)
Weight: 556 g
Dimensions: 126 x 94 x 65 mm
Year: 2006

9.3 CM-030 GE / CB-030 GE

The CM-030GE and CB-030GE are designed following the GigE Vision standard. It transmits digital images over Cat5e or Cat6 Ethernet cables. All camera functions are also controlled via the GigE Vision interface. The camera can operate in continuous mode, providing an endless stream of images. For capturing individual images, related to a specific event, the camera can also be triggered. For precise triggering, it is recommended to use a hardware trigger applied to the Hirose 12-pin connector. It is also possible to initiate a software trigger through the GigE Vision interface. However, when using software triggers, certain latency inherent to the GigE interface must be anticipated. This latency, which manifests itself as jitter, greatly depends on the general conditions and traffic on the GigE connection. The frame rate described in this manual is for the ideal case and may deteriorate depending on conditions.

9.3.1 Main Features

Member of Compact series, covering VGA to UXGA resolution
656 (h) x 494 (v) 7.4 µm square pixels
1/3" progressive scan – Monochrome and Bayer mosaic color versions
High frame rate of 90.5 frames/second with full resolution in continuous operation
90 frames/second with external trigger and full resolution
Increased frame rate with vertical binning (CM-030GE only) and partial scan
Exposure time from 21.6μs to 2 sec. using Pulse Width Control trigger mode
Programmable exposure from 43.2μs to 11.037 ms in Full Frame scan
Sequencer trigger mode for on–the–fly change of gain, exposure, and ROI
Edge Pre-select and Pulse width trigger mode
LVAL-synchronous/-asynchronous operation (auto-detect)
Auto iris lens video output allows a wider range of light
GigE Vision Interface with 10 or 8-bit output
Programmable GPIO with optoisolated inputs and outputs
Comprehensive software tools and SDK for Windows XP/Vista

9.3.2 Specifications

11.2 Specification table

Specifications	CM-030GE	CB-030GE
Scanning system	Progressive scan	
Frame rate full frame	90.5 frames/sec. Progressive (511 lines/frame)	
Pixel clock	40 MHz	
Line frequency	46.296 kHz (1H = 21.6 µs), (864 pixel clock/line)	
CCD sensor	1/3". Monochrome ICX424AL	1/3" Bayer Color ICX424AQ
Sensing area	4.85 (h) x 3.66 (v) mm 1/3 inch diagonal	
Cell size	7.4 (h) x 7.4 (v) µm	
Active pixels	656 (h) x 494 (v)	
Pixels in video output. Full Scan 2/3 partial Scan 1/2 partial Scan 1/4 partial Scan 1/8 partial Scan Vertical Binning Region-of-interest (ROI)	656 (h) x 494 (v) 90.5 fps. H = 46.296 kHz 656(h) x 326 (v) 128 fps H= 46.296kHz 656 (h) x 246 (v) 159 fps. H = 46.296 kHz 656 (h) x 122 (v) 255 fps. H = 46.296kHz 656 (h) x 62 (v) 361 fps. H = 46.2965 kHz 656 (h) x 247 (v) 166.2 fps. H = 42.735 kHz (*Note) User Definable. Memory read-out *Note: Vertical binning is for CM-030GE only	
Sensitivity on sensor (minimum)	0.004 Lux (Max. gain, Shutter OFF, 50% video)	1.21 Lux (Max. gain, Shutter OFF,50% Green, w/IR cut filter)
S/N ratio	More than 50 dB (0dB gain)	
Digital Video output.	GigE Vision Compliant Mono8,Mono10,Mono10_Packed	GigE Vision Compliant BAYRG8,BAYGB8,BAYRG10,BAYGB10
Iris video output. Analogue	0.7 V p-p , enabled by internal switch	
Gain	Manual -3 to +12 dB	
Synchronization	Internal X-tal	
GPIO Module Input/output switch Clock Generator (One) Pulse Generators (Four)	Configurable 14-in / 9-out switch 12-bit counter based on 25MHz clock or Pixel clock 20-bit counter programmable for length, start point, stop point, repeat	
Hardware Trigger modes	Edge Pre-Select , Pulse Width Control, Frame Delay and Sequence	
OB area transfer mode	ON / OFF	
Event message	SYNC / ASYNC mode (Trigger mode status when exposure starts) Exposure start, Exposure end, Trigger IN, Video start, Video end	
Electronic Shutter Preset Shutter speed Programmable exposure Exposure Time (Abs) GPIO plus Pulse Width	OFF(1/90) and 1/100 to 1/10,000 in 9 steps 2L(43.2µs) to 511L (11.037ms) in 1L steps µsec – user definable. Same range as PE max. 2 sec (Can be set by 100µs unit or Pixel Clock unit)	
Control interface	Register based. GigE Vision / GenIcam compliant	
Functions controlled via GigE Vision Interface	Shutter, Gain, Black Level, Trigger mode, Read out mode, GPIO setup ,ROI (GenIcam mandatory functions)	
GigE Vision Streaming Control	Packet size, Delayed (Frame) read-out, inter-packet delay Jumbo frame can be set at max. 4K(4040) , Default packet size is 1428 Byte.	
Indicators on rear panel	Power, Hardware trigger, GigE Link, GigE activity	
Operating temperature	-5°C to +45°C	
Humidity	20 – 90% non-condensing	
Storage temp/humidity	-25°C to +60°C/20% to 90% non-condensing	
Vibration	10G (20Hz to 200Hz, XYZ)	
Shock	70G	
Regulatory	CE (EN61000-6-2 and EN61000-6-3), FCC part 15 class B, RoHS, WEEE	
Power	12V DC ± 10%. 3.6 w	
Lens mount	C-mount, Rear protrusion on C-mount lens must be less than 10.0mm	

IR-cut & Optical Low Pass Filter	Built in (only for CB-030GE)	
Dimensions	29 x 44 x 75 mm (HxWxD)	
Weight	125 g	125 g

For stable operation within the above specifications, allow approximately 30 minutes warm up.

Note: Above specifications are subject to change without notice

Chapter 10

Appendix B: Depth Perception by Humans

10.1 Human Eye

The human eye (Fig. 10.1) is a sensory organ that, by responding to visible light, provides humans with knowledge for various intelligent activities. The human eye is roughly 2.3 cm in diameter and consists of optical elements that consist of the cornea lens, which is the dome-shaped tissue covering the eye in front. This tissue lens focuses light from the external world, while the iris controls the pupil size (aperture) to regulate the amount of light entering the eye. A second lens is located just behind the iris and focuses the light on the light-sensitive retina at the back of the eye. The retina has photoreceptors like cones and rods. Cones are optic nerve cells that sense bright light and provide the perception of high-resolution, colored images. The rods are nerve cells that sense dim lights and provide low-resolution, black-and-white images, or peripheral vision. Ganglion cells are also present, which respond to the full light intensity range and help limit the amount of light reaching the retina. Light enters the eye through the cornea, pupil, and lens. The ciliary muscles adjust the lens shape for continuous focus (accommodation). The cones and rods convert light into electrical signals, which are transmitted to the brain through the optic nerve. No photoreceptors are present at the junction of the retina and optic nerve, which creates a blind spot.

The eye is sensitive to wavelengths within 380 nm to 800 nm. The non-linear response peaks at 555 nm (green light). The CIE standard defines the average spectral eye sensitivity during a daylight environment (photopic vision) as a spectral luminous efficiency function' $V(\lambda)$. Low light (scotopic vision) eye sensitivity is defined by the function $V'(\lambda)$. These eye spectral sensitivity functions are normalized as shown in Fig. 10.2. Humans cannot see infrared wavelengths (< 800 nm), which are exploited by many active depth cameras.

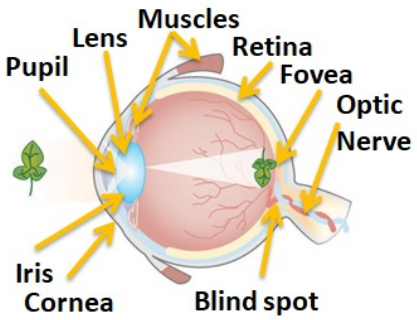


Figure 10.1: Basic components of an eye for human vision.

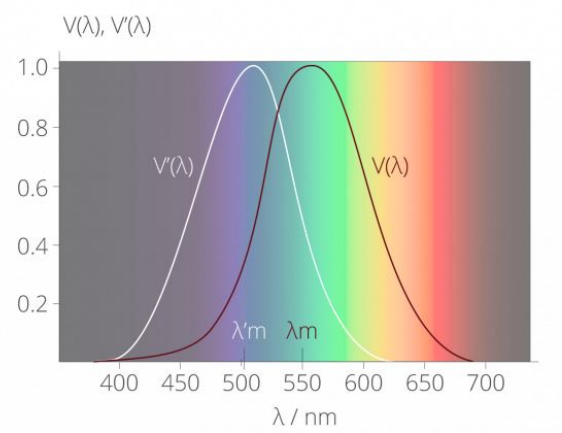


Figure 10.2: CIE spectral luminous efficiency function (a) photopic vision $V(\lambda)$ and (b) scotopic vision $V'(\lambda)$).

10.2 Depth Perception modes

Humans have two eyes and perceive depth primarily due to stereopsis and accommodation of the eye. Each eye sees a common view with small, different angles, which leads to stereopsis. The brain processes these two views into a single 3D image (Fig. 10.3). The subconscious processing includes prior knowledge to understand the surroundings around us. Humans can also judge depth with just one eye by using a variety of depth cues. Human depth perception can be summarized into three categories: binocular, monocular, and oculomotor.

10.2.1 Binocular cues

Depth perception through binocular cues uses both eyes. The cues can be categorized into $stereopsis$, $shadowstereopsis$, and $convergence$. Stereopsis, also known as retinal dis-

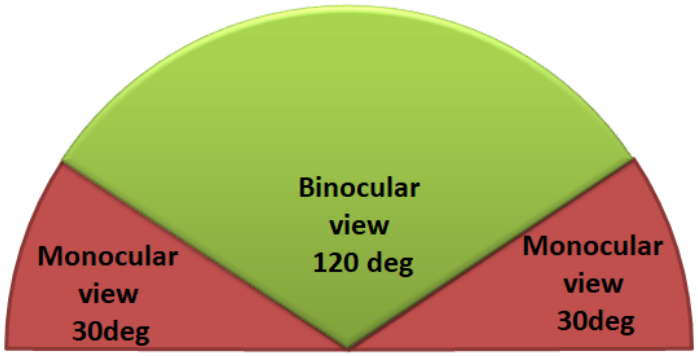


Figure 10.3: Binocular field of view limitation.

parity or binocular parallax, uses the frontal eyes to derive different projections of objects. The eyes are separated horizontally by about 6 cm and create parallax (Fig. 10.4). Each eye provides information about the scene with different view angles, leading to binocular disparity in the brain. Humans use this disparity to estimate the distance to an object with good accuracy through triangulation. The disparity is large for close objects and poor for distant objects. The disparity does improve with an increase in the distance between the eyes, which is possible in some animals. This property of human vision is exploited in 3-D movies and stereoscopic photos.

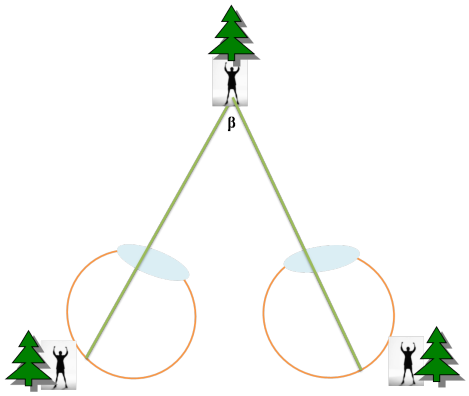


Figure 10.4: Stereopsis cue leads to disparity image in the brain by creating mirror image in L and R eye. Small distances give large disparities of β depth information.

Shadows are good cues for depth perception. Retinal images with different shadows are also processed by the brain as stereo images to get depths (Fig. 10.5). This is called shadow stereopsis [101].

For close objects, our eyes angle inward to focus on the same point, which leads to *convergence* (Fig. 10.6). The convergence stretches the extraocular muscles of the eye,

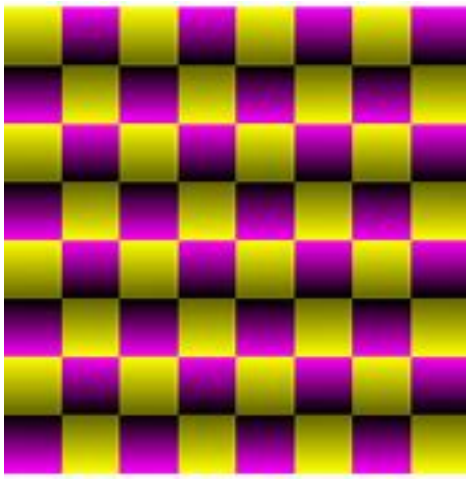


Figure 10.5: Shadow stereopsis cue provides depth perception.(Copyright Akiyoshi Kitaoka 2005)

which produces kinesthetic sensations that assist in depth perception. This is also called a binocular oculomotor cue. As the angle of convergence reduces for distant objects, the convergence cue is useful up to 10 meters in depth.

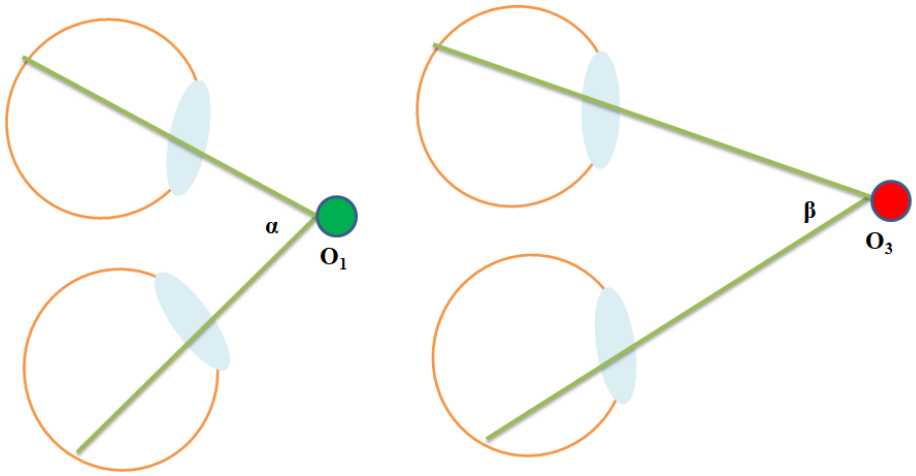


Figure 10.6: Convergence requires the eye to angle inward. Small distances give large angles.

10.2.2 Monocular cues

Monocular depth cues come from one eye. One cue type is $Motion\ parallax$ and comes from the differential movement or speed of static objects relative to the background. The motion parallax can provide absolute depth information along with additional information

like direction and velocity. Examples include driving in a car where near objects move faster than distant objects.

In $Depth\ from\ motion$, the changing object size with time provides a cue. Examples include a ball coming towards a person. This dynamic change in object size leads to a deeper perception of the moving object. Rotating objects also create depth cues due to the kinetic effect.

Near objects provide finer granular details compared to distant objects. This leads to cues based on $Texture\ gradient$. The fine details like shape, size, color, etc. on nearby objects are clearer than those at a distance where the features merge. Examples are long gravel roads, tiled footpaths, etc., where the texture merges uniformly at a distance (Fig. 10.7).

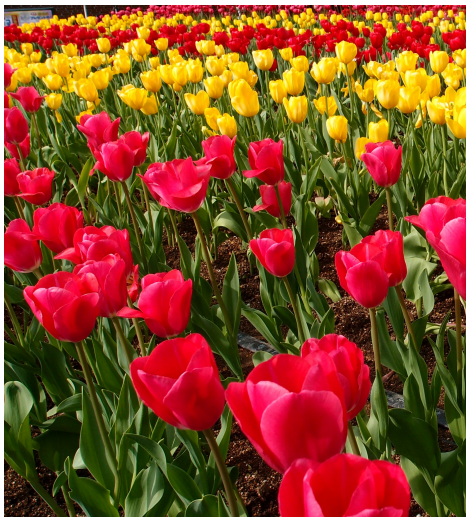


Figure 10.7: Texture gradient as a depth cue. The flowers in front have more details than those in the background.(https://pxhere.com/id/photo/912200)

The limited depth of focus of the human eye creates a selective blurring of objects as seen in top left corner of (Fig. 10.7). This $defocus\ blur$ provides the perception of depth.

$Relative\ sizes$ between similar known objects also provide depth cues. One can also estimate depth from the perceived size of a known object. Thus, a small object is perceived to be far away compared to a bigger object (flowers in Fig. 10.7) when we know that they are the same. Prior information on the object's size also provides a depth cue. This is called the $Familiar\ size$ cue. Similarly, a small object is perceived as far away when we are aware of the absolute size or $Absolute\ cue$.

$Occlusion$ or the interposition of objects, provides information about depth as the object in front partially blocks the background (Fig. 10.8). The object with a discontinuous

border is perceived as being behind the foreground.

Figure 10.8: The objects are identical; the occluded object is behind.

In $Perspective$, parallel lines converge in the distance, which provides the relative distance of objects (Fig. 10.9). An object in the larger area is treated as near, while those at converging ends are far.

Figure 10.9: Convergence of parallel lines creates a depth cue.

Similarly, $elevation$ provides information on the depth of objects relative to the hori-

zon. An object position that is higher than the horizon will be perceived as distant compared to one below the horizon. The buildings in Fig. 10.9 are of the same size.

Lighting and shadows provide the perception of position with knowledge of the light source (Fig. 10.10). The atmosphere scatters light, and hence distant objects have lower luminance contrast, poor color saturation, and appear hazy compared to the foreground. The distant objects shift towards blue (Fig. 10.11).

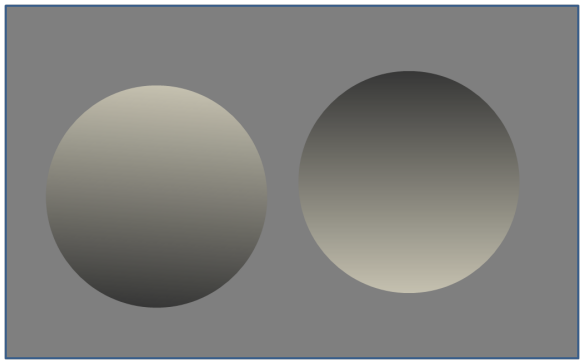


Figure 10.10: Light and shade create depth perspective.

Figure 10.11: The mountains at a distance have a bluish tinge due to atmospheric light scattering, giving rise to the color cue.(https://pixy.org/4801691/)

10.2.3 Oculomotor cues

The human eye has oculomotor and ciliary muscles. The oculomotor muscles control the movement of the eye, constriction of the pupil, focusing of the eyes, and the position of

the upper eyelid. The ciliary muscles change the focal length by compressing the eye lens. Oculomotor depth cues are proprioceptive information from oculomotor muscles and ciliary muscles.

The ciliary muscles stretch or contract the lens to focus on the object. This information about stretch or contract is the cue to interpret depth and is called $Accommodation$.

$Vergence$ makes both eyes move in appropriate directions to focus on an object. Near objects require both eyes oriented inwards ($convergence$) to have the object foveated, whereas far objects need the eyes to be aligned along parallel lines of sight.

Convergence, accommodation, and familiar size among human vision depth cues provide absolute distance information. Other cues provide relative information.

10.2.4 Camera

In machine vision, the equivalent of the human eye is the camera (Fig.10.12). The camera is an optical instrument that captures mostly 2D images. Some advanced cameras can also capture 3D images. The camera has evolved from a pinhole camera to an optics-based lens camera. The image recorder or photosensitive zone has also evolved from photosensitive chemical plates, films, analog vacuum tubes, and solid-state CCDs to modern active-pixel digital sensors. The basic camera consists of a light-sealed box. The light enters the box through a lens system, a small controllable hole (aperture) that regulates light input, and a mechanical or electronic shutter to integrate photons. The lens focuses the light onto the digital sensor to capture the image. The camera focuses on objects of interest by moving the lens along the optical axis to sharpen the object's features. The aperture consists of a ring of overlapping plates (the aperture ring), which adjusts the light intensity by changing the opening. The digital sensor has a cluster of photosensitive pixels. The dimension of the pixel defines the spatial resolving power, contrast resolution, and dynamic range characteristics. Present sensors have more than 64 megapixels. Color digital sensors use red, green, and blue filters to capture 2D RGB images. Modern sensors use pixel-size lattice filters arranged in a Bayes pattern mosaic to mimic human spectral sensitivity. This mosaic is a repeated 2 x 2-pixel set that has a pair of diagonal green pixels, a red pixel, and a blue pixel. The electronics or image processor uses a decoding algorithm to interpolate the RGB information. Table.10.1 compares the properties of the eye and camera.

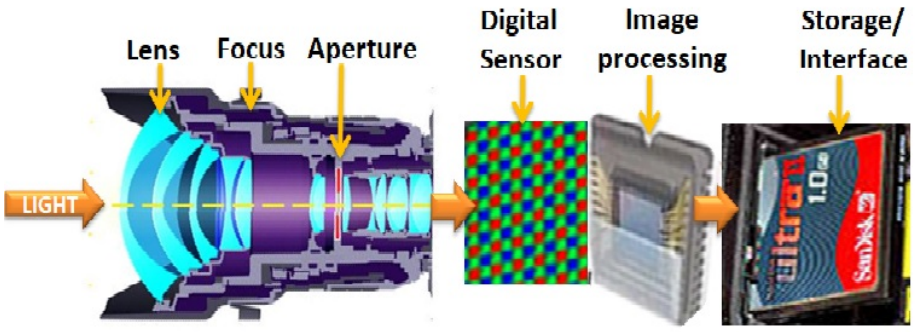


Figure 10.12: Basic digital camera schematic. Most are available as systems-on-chip.

Table 10.1: Properties of eye and camera: similarity and difference.

Sl.#	Eye	Camera
1	Living organ	Technology/ instrument
2	Consists of lenses and light-sensitive surfaces.	Consists of lenses and light-sensitive surfaces.
3	Need an external light source to image	Most need an external light source to image, active cameras have a built–in light source.
4	The iris controls the light entering the eye.	The aperture controls the light entering the camera.
5	Fixed 2.3 cm diameter spherical-shaped ball. Eyes are flexible and are filled with fluid.	Cameras are rigid boxes with a rigid lens.
6	An inverted image is formed on the retina.	May or may not record an inverted version of the image.
7	The lenses change shape continuously to stay focused on objects of interest.	Rigid lens cannot change focal length, so objects within a narrow range of distances are in focus. The camera focus needs adjustment beyond this range. Alternatively, replacing the lens with a different focal length will focus on objects at a new range of distances.
8	Retinas contain rods and cones. Rods cater to low-light monochrome images, while red, green, and blue cones respond to different spectral wavelengths.	Cameras have the same photosensitive pixels, which respond to monochrome light passed through spectral filters (color).
9	The retina center has no photoreceptors. This creates a blind spot.	All pixels on the image plane are evenly distributed, and no blind spot exists.
10	Recording of images is not possible,	Cameras can record images and videos
11	Can adjust to the dark in a matter of seconds	Cannot adapt to the dark.
13	Only responds to 390nm to 780 nm	Can respond to beyond 390nm to 780 nm

Chapter 11

Appendix C: Datasets

11.1 Datasets using cameras

Standard datasets available from researchers, organizations, and universities that are acquired with cameras are summarized in Table 11.1.

Table 11.1: Various Datasets for Training

Sl. No.	Database	Resources	Benchmark, papers
1	Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) [59]	>93k RGB traffic scene images with aligned sparse depth maps	120, 2459
2	Cityscapes, Cityscapes3D [102]	Outdoor dataset with 5k high-quality pixel-level annotations with 30 classes and 8 categories.	37, 2564
3	NYU Depth V1/V2 [60]	Indoor dataset of 464 scenes, 407,024 unlabeled frames and semi-synthetic depth data. The depth range is 10m.	18, 612
4	Dense Indoor and Outdoor DEpth (DIODE) [103]	Diverse set of RGBD images of 8574 indoor, 16884 outdoor with dense wide-range depth data. Provides 50m and 300m depth ranges for indoor and outdoor data respectively.	2, 32

Continued on next page

Table 11.1 – continued from previous page

Sl. No.	Database	Resources	Benchmark, papers
5	Southampton-York Natural Scenes (SYNS), https://syns.soton.ac.uk.	Has 1175 urban and natural images, panorama of stereo image pairs, high-dynamic range (HDR) spherical images, spherical LiDAR data with 120m range	
6	SYNS-Patches [104]	Has 1656 images and LiDAR views of 92 scenes belonging to a wide variety of environments like residential, industrial, agriculture, natural forests and fields, indoor, etc.	
7	Cambridge-driving labeled Video Database (CamVid) [105]	Manually annotated outdoor image sets with 32 classes of annotation. The images are useful for testing but not training.	
8	Baidu ApolloScape [106]	A large-scale open dataset for self-driving technologies width high-resolution RGB videos (>100k frames).	
9	Sun RGB-D [107]	Indoor dataset of 10,335 images, corresponding reference depth maps and dense annotations with 2D and 3D bounding boxes.	
10	SUN3D [108]	Has 8 annotated sequences of large-scale RGBD videos of buildings, with each frame in the scene providing information about the camera pose and semantic segmentation of the objects.	

Continued on next page

Table 11.1 – continued from previous page

Sl. No.	Database	Resources	Benchmark, papers
11	ScanNet [109]	An 2D and 3D instance-level indoor RGBD dataset with labeled voxels and 1513 annotated scans, surface reconstructions, instance-level semantic segmentation and 3D camera poses.	15, 701
12	Matterport3D [110]	Has 194,400 RGBD indoor images with 10,800 panoramic inside views of 90 real building-scale scenes, annotations like semantic segmentation, surface construction and camera poses.	4,272
13	Middlebury (Middlebury Stereo) [111]	Has high-resolution stereo sequences and accurate, pixel level disparity ground truth using structured light.	5,186
14	TUM RGB-D [112]	Contains RGBD images from Microsoft Kinect camera	
15	Make3D [113]	Has 400 single RGB image and depth map pairs for training monocular depth estimation algorithms and 134 test samples.	1, 117
16	2D-3D-S (2D-3D-Semantic) [114]	Has >70k RGB images from 6 large indoor areas with depths, surface normals, camera information, multiple annotations, 3D meshes, point clouds, and global XYZ images to provide co-registered modalities from 2D, 2.5D, and 3D domains.	5,103
17	2D-3D Match Dataset [115]	Has 110 RGBD scans from 3DMatch and SceneNN 3D data. The dataset provides about 1.4 million 2D-3D correspondences.	

Continued on next page

Table 11.1 – continued from previous page

Sl. No.	Database	Resources	Benchmark, papers
18	Taskonomy [116]	This indoor scene dataset distils knowledge from ImageNet, MS COCO, and MIT with Semantic information, pixel-level geometric information, camera poses and camera intrinsics.	2, 97
19	ETH3D [117]	Has indoor and outdoor multi-view stereo and 3D reconstruction benchmark.	1,53
20	DrivingStereo [118]	Provides a diverse set of driving scenarios of >180k images with high-quality disparity labels	
21	Dense Depth for Autonomous Driving (DDAD) [119]	Have monocular videos from self-driving cars of a variety of urban environments with a long range of 250m.	1
22	Visual Odometry with Inertial and Depth (VOID)} [120]	Outdoor and indoor dataset having 56 sequences of >47K motion frames and time stamped sparse depth maps at 1500, 500, and 150 density levels.	1
23	Stanford Light Field} [121]	Light field data from microscope, gantry, Stanford Multi-Camera Array and a simple Lego Mindstorms gantry.	
24	HUMAN4D [122]	4D dataset of multi—modal, diverse social and physical human activities along with audio by 2 male and 2 female actors for a wide range of motions and poses.	

Continued on next page

Table 11.1 – continued from previous page

Sl. No.	Database	Resources	Benchmark, papers
25	A large-scale high-resolution outdoor stereo dataset (UA-SOL) [123]	Has 2280x1282 pixel RGBD frames from 33 different scenes, totaling to 160902 frames. Each scene of 2k to 10k frames gives a pedestrian's perspective along with a GPS geo-localization tag.	1
26	DurLAR [124]	Multi-modal autonomous driving dataset with diverse 2048×128 panoramic ambient (near infrared), high-fidelity 3D images and high-fidelity GNSS/INS data.	
27	Headcam [125]	A helmet-mounted camera collected this panoramic video dataset	
28	Pano3D [126]	Provides spherical panoramas consistently and holistically for evaluating the transfer performance of zero-shot cross-datasets and provides generalization by decomposing data into three different splits.	
29	SeasonDepth [127]	Derived from the CMU Visual Localization dataset, this provides cross-season scaleless data using SfM.	
30	Autonomous-driving Streaming Perception Benchmark (ASAP) [128]	Aims to evaluate online performance of vision–centric perception in autonomous driving.	
31	Indoor and outdoor DFD dataset [129]	Has 110 indoor images with related depth images for training and 29 images for testing.	

Continued on next page

Table 11.1 – continued from previous page

Sl. No.	Database	Resources	Benchmark, papers
32	Mars DTM Estimation	Provides Mars surface 250k stereo image patches of a 3–channel 512 x 512 raster and digital terrain map (DTM).	
33	VIS-TIR	Provides dual–spectrum of visible light and thermal infrared images.	

11.2 Synthetic and simulated datasets

Many synthetic and simulated datasets are available for training and benchmarking. These are listed in the table. 11.2.

Table 11.2: Various Datasets for Training

Sl. No.	Database		Resources	Benchmark, papers
1	SUNCG [130]		A large-scale dataset for indoor scenes with over 45K manually created realistic diverse scenes of furniture layouts and room, which have semantic annotations at object level.	
2	SceneNet [131]	RGB-D	A large scale photo realistic data contains 5m indoor images with corresponding pixel wise depth maps with randomly generated room conditions, lighting, and textures.	

Continued on next page

Table 11.2 – continued from previous page

Sl. No.	Database	Resources	Benchmark, papers
3	SYNTHIA [131]	Virtual urban driving scenes comprising street blocks, suburban areas, highways, etc. YNTHIA–Seqs and SYNTHIA–Rand are the two complementary datasets. A panoramic version of this dataset is the SYNTHIA–PANO.	
	Aerial depth dataset [132]	Artificial outdoor provides large viewpoint and depth variations from 18 virtual 3D models and contains 83797 single-view RGBD images.	
4	Hypersim [133]	Contains photo-realistic 461 indoor scenes totaling to 77,400 images along with ground truth geometry. The data provides a per–pixel, detailed ground truth label .	1, 25
5	Depth Estimation oN Synthetic Events (DENSE) [134]	Contains synthetic events with perfect ground truth from the CARLA simulator and caters to low contrast.	1,24
6	3D60 [135]	A synthetic spherical depth estimation dataset with ray and annotations.	
7	CocoDoom [136]	Generated from Doom 3D gaming has annotations in the MS Coco format useful at the level of instances, training and evaluation.	
8	3D Ken Burns [137]	Large-scale photo-realistic scenes along with accurate ground truth depth.	

Continued on next page

Table 11.2 – continued from previous page

Sl. No.	Database	Resources	Benchmark, papers
9	4D Light Field [138]	Ddensely sampled 4D light fields with highly accurate disparity ground truth providing camera settings, disparity ranges, 5120x5120 disparity maps, 9x9x512x512x3 light fields, 512x512 depth maps, and evaluation masks of resolutions 512x512 and 5120x5120.	
10	Endoscopic SLAM (EndoSLAM) [139]	Endoscopic videos for depth estimation with 35 sub-datasets and ex-vivo data.	
11	Mila Simulated Floods [140]	A 1.5 square km virtual world of before and after flood pairs of rural suburban and urban areas.	1
12	Enclosed garDEN (EDEN) [141]	Has > 100 garden models and >300k images along with annotations for depth, semantic segmentation, optical flow, intrinsic colors and surface normal.	
13	FutureHouse [142]	A virtual, photo–realistic, large-scale and panoramic dataset with >70k models, high-resolution meshes , 1,752 house-scale scenes with 28,579 panoramic views.	

Continued on next page

Table 11.2 – continued from previous page

Sl. No.	Database	Resources	Benchmark, papers
14	INRIA Dense Light Field (INRIADLFD) [143]	A light field dataset using Blender, provides 512x512 pixels with 9x9 angular resolution with two subsets. The Dense Light Field Dataset (DLFD), with 39 scenes, provides a disparity range of [-4, 4] pixels. The Sparse Light Field Dataset (SLFD) provides 53 scenes with a disparity range of [-20, 20] pixels.	
15	Virtual KITTI	Synthetic videos from the KITTI with changing weather, modified camera parameters, poses and vehicle locations	

11.3 Database from the Web

A good number of databases are created from information available on the Web. These are listed in the table. 11.3.

Table 11.3: Various Datasets for Training

Sl. No.	Database	Resources	Benchmark, papers
1	Office-31 [144]	Has 795 low-resolution indoor office images of online merchants (Amazon, DSLR) against a clean background within a normalized scale with noise, color, and white balance artifacts.	**8, 466**

Continued on next page

Table 11.3 – continued from previous page

Sl. No.	Database	Resources	Benchmark, papers
2	MegaDepth [145]	A multi-view photo collection of 196 different locations caters to depth prediction. Models trained on MegaDepth exhibit strong generalization.	0,87
3	MannequinChallenge [146]	In-the-wild dataset contains video clips (>170k frames) of people in static pose (imitating mannequins) leading to natural diverse poses for training, validation, and testing.	
4	Relative Depth from Web (ReDWeb) [147]	A wide variety of RGB images and related dense relative depth with 3600 image pairs and includes various non-rigid objects.	
5	Depth in the Wild [148]	A variety of 495K single images from the wild with corresponding relative depths annotated between a pair of randomly sampled points.	
6	Open Annotations of Single Image Surfaces (OASIS) [149]	Crowdsourced 140k single images with rich annotations like relative depth, surface normal, fold boundary, occlusion boundary, relatively normal, and planarity. OASIS V2 has annotations for an additional 102k images.	
7	Holopix50k [150]	A diverse, in the wild, crowd sourced mobile stereo images (49,368).	
8	Web Stereo Video Dataset (WSVD) [151]	A in the Wild 553 stereo video for reconstruction of non-rigid scenes. Pairs of frames within the videos provide the depth information.	

Continued on next page

Table 11.3 – continued from previous page

Sl. No.	Database	Resources	Benchmark, papers
9	High-Resolution Web Stereo Image (HRWSI) [152]	A wide variety of high-resolution RGB images and related relative depth obtained from stereo images with invalid pixel masks, sky segmentation masks and instance segmentation masks.	
10	eBDtheque [153]	Contains selected and manually annotated 100 comic pages with information of semantic annotation and visual segmentation.	1
11	DCM [154]	The data Contains 772 images from 27 comic books (Digital Comics Museum) with annotation like human-like, animal-like, ground truth bounding, etc.	
12	InstaOrder [155]	Provides 101K natural scenes and information to understand the geometrical relationship instances in an image with relative depth from the camera and occlusion.	
13	TikTok [156]	A collection of >300 human dance videos and >100k RGB images taken from the TikTok social media for training high–fidelity human depths.	

11.4 Tools for Database

Tools are also available for custom synthetic database development 9Table.11.4).

Table 11.4: Tools for database development

Sl.No.	Tool	Capability
1	CALRA simulator [157]	An open-source simulator for autonomous driving research with a good collection of depth maps and corresponding RGB images. CARLA provides open digital assets (urban layouts, buildings, vehicles) and a sensor suite including LIDARs, multiple cameras, depth sensors, and GPS, among others.
2	MineNav	Minecraft (sandbox game) generates these synthetic data w.r.t. scenes from a large game community. Several plug-in programs allow rendered image sequences with time-aligned depth maps, camera poses and surface normal.
3	SuperCaustics	This is a tool made with Unreal Engine to simulate massive computer vision datasets. The simulated data can include transparent objects.