Multiview Video View Synthesis and Quality Enhancement using Convolutional Neural Networks

A thesis submitted in accordance with the requirements of the
University of Liverpool
for the degree of Doctor in Philosophy
by

Samer JAMMAL

Department of Electrical Engineering and Electronics
School of Electrical Engineering and Electronics and
Computer Science
University of Liverpool

April, 2019

Abstract

Multiview videos, which are recorded from different viewpoints by multiple synchronized cameras, provide an immersive experience of 3D scene perception and more realistic 3D viewing experience. However, this imposes an enormous load on the acquisition, storage, compression, and transmission of multiview video data. Consequentially, new and advanced 3D video technologies for efficient representation and transmission of multiview data are important aspects for the success of multiview applications.

Various methods aiming at improving multiview video coding efficiency have been developed in this thesis, where convolutional neural networks are used as a core engine in these methods. The thesis includes two novel methods for accurate disparity estimation from stereo images. It proposes the use of convolutional neural networks with multi-scale correlation for disparity estimation. This method exploits the dependency between two feature maps by combining the benefits of using both a small correlating scale for fine details and a big scale for larger areas. Nevertheless, rendering accurate disparity maps for foreground and background objects with fine details in real scenarios is a challenging task. Thus, a framework with a three-stage strategy for the generation of high-quality disparity maps for both near and far objects is proposed.

Furthermore, the current techniques for multiview data representation, even if they exploit inter-view correlation, require large storage size or bandwidth for transmission. Such bandwidth is almost linear with the number of transmitted views. To address this problem, we proposed a novel view synthesis method for multiview video systems. In this approach the intermediate views are solely represented using their edges while dropping their texture content. These texture contents can get synthesized using a convolutional neural network, by matching and exploiting the edges and other information in the central view. Experimental results verify the effectiveness of the proposed framework.

Finally, highly compressed multiview videos produce severe quality degradation. Thus, it is necessary to enhance the visual quality of highly compressed views at the decoder side. Consequentially,a novel method for multiview quality enhancement that directly learns an end-to-end mapping between the low-quality and high-quality views and recovers the details of the low-quality view is proposed.

Key Words: Multiview video, view synthesis, quality enhancement, depth estimation, stereo vision, disparity estimation, convolutional neural network.

Acknowledgement

Four years of study, research, work have passed. It has been a fruitful journey: a journey across experiences will be my guide during my social life and academic career.

Foremost, I would like to express my sincere thanks to my supervisor Prof. Tammam TILLO, who gave me significant guidance and great support during my PhD study and research, and for offering the possibility of studying in his group as a Ph.D. candidate. His guidance helped me in all the period of my PhD research and writing this thesis. I could not have imagined having a better supervisor for my PhD research. Also, I would like to thank my supervisor Dr. Jimin XIAO for his support, his precious ideas, suggestions that in some cases really made the difference and his help for writing papers.

Besides my supervisors, I would like to thank my friends and colleagues in the MMT lab and in EEE department for their support, in particular, Dr. Mark LEACH, Prof. Kaizhu HUANG, Dr. Fei CHENG, Dr. Zhi JIN, Mr. Mingjie Sun and Mr. Yanchun Xie, who has always helped me solving technical problems. Special thanks to Dr. Haochuan JIANG, who has always offered his support and help during my Ph.D. life.

Especially, I would like to thank my parents and my wife Ruba for their encouragement, support, and love during the past few years. Without their support, I could not have such an achievement. Not everyone might be listed here, but all of you are in my memory.

Contents

Abstract **i**

Acknowledgement **iii**

Contents **v**

List of Figures **ix**

List of Tables **xiii**

List of Abbreviations **xiv**

1 Introduction **1**

1.1 Motivation 1

1.2 Overview of Multiview 3D Video 4

1.2.1 3D Dimensional Video Capturing and Display 4

1.2.2 3D Videos 6

1.3 Depth and Disparity Map Estimation 7

1.3.1 Passive methods 7

1.3.2 Active Methods 8

1.4 View Synthesis 10

1.4.1 View Synthesis using Stereo and Multiview Images 10

1.4.2 Depth Image-Based Rendering 11

1.5 Convolutional Neural Networks for Disparity Estimation 12

1.5.1 Architecture Overview 13

1.5.2 Convolutional Layer 15

1.5.3 Rectified Linear Unit 17

1.5.4 Pooling Layer 17

1.5.5 DispNetCorr1D Architecture 18

1.6 Image Fusion and Enhancement using Convolutional Neural Networks . . 22

1.6.1 Image and Video Enhancement 22

1.6.2 Multiview Depth-Based Quality Enhancement 23
1.7 Overview of This Thesis 24
1.7.1 Major Contributions 24
1.7.2 Brief Summary of the Remaining Chapters 25

2 Depth Map Estimation from Stereo Images using Convolutional Neural Networks 29
2.1 Motivation 29
2.2 The Proposed Depth Estimation Network Architecture 30
2.2.1 Training Method 32
2.3 Computer Graphic Dataset Generation 32
2.4 Real Stereo Dataset Generation 34
2.4.1 Calibration of SR-4000 TOF Camera 34
2.4.2 Disparity-Depth Mapping 36
2.5 Experimental Results 36
2.5.1 Computer Graphic Testing Set 37
2.5.2 Real Testing Set 38
2.6 Application and Limitation 38
2.7 Conclusion 39

3 Disparity Estimation from Stereo Images Using Convolutional Neural Networks with Multi-Scale Correlation 41
3.1 Introduction 41
3.2 Proposed Multi-Scale Correlation Approach 42
3.2.1 CNN Architecture 43
3.2.2 Multi-Scale Correlation 44
3.3 Experimental Results 45
3.3.1 Synthetic Training Set 46
3.3.2 Training the CNN with Multi-Scale Correlation 46
3.3.3 Results 47
3.4 Conclusion 50

4 Multi-Resolution Disparity Fusion Using Convolutional Neural Networks 51
4.1 Introduction 51
4.2 Motivation and Observation 53
4.2.1 Single-resolution Disparity Estimation 53
4.3 Multi-Resolution Disparity Fusion Approach 57
4.3.1 Multi-Resolution Disparity Estimation Framework 57

4.3.2 Convolution-based Multi-Resolution Disparity Fusion Network . 59
4.3.3 Convolution-Deconvolution-based Multi-Resolution Disparity Fusion Network 59
4.3.4 Training Procedure 62
4.4 Experimental Results 62
4.4.1 Single Resolution vs Multi-resolution Estimation 63
4.4.2 Benchmark Results 65
4.4.3 Re-sampling Horizontally and Vertically 66
4.4.4 Various Re-sampling Methods 67
4.5 Conclusion 67

5 Silhouette-Based Multiview Video Synthesis using Deep Convolutional Neural Networks 71
5.1 Introduction 71
5.2 Proposed Multiview View Synthesis Framework 73
5.2.1 Multiview Video Plus Silhouette 73
5.2.2 Network Architecture 74
5.3 Experimental Results 74
5.3.1 Computer Graphic Training Set 76
5.3.2 SVSNet Framework Evaluation 76
5.3.3 Various Edge Detector Thresholds 76
5.3.4 Visual Comparison with HEVC Baseline 79
5.3.5 Object Place Editing 80
5.4 Conclusion 81

6 Multiview Video Quality Enhancement without Depth Information 83
6.1 Introduction 83
6.2 Proposed Multiview Quality Enhancement Approach 84
6.2.1 Proposed Framework 85
6.2.2 Network Architecture 86
6.2.3 Training Procedure 86
6.3 Experimental Results 86
6.3.1 Implementation Details 86
6.3.2 Comparison with HEVC Baseline 89
6.3.3 Comparison with JPEG Approaches 91
6.4 Conclusion 94

7 Conclusions & Future Work 97

7.1 Summary 97

7.2 Perspectives for future work 98

Appendix: A list of Publications 101

Reference 103

List of Figures

1.1 Autostereoscopic technologies: parallax-barrier (upper image) and lenticular (lower image). 5
1.2 A conventional stereo video coding method with inter-frame and inter-view prediction. 6
1.3 An example coding structure in MVC for linear five camera setups and GOP size of eight pictures. 6
1.4 Depth map measurements 7
1.5 Horizontal displacement in a stereo vision system. 9
1.6 The monkey image pair. Two computer graphic images of a scene taken from different viewpoints. 10
1.7 The structure of view synthesis approach using a DIBR-based method. . . 12
1.8 A CNN represents the inputs/outputs of each layer in dimensions (width, height, depth). 14
1.9 Typical structure of Convolutional Neural Network. 14
1.10 This figure demonstrates a convolution within a CNN without padding. . . 16
1.11 This figure demonstrates a convolution within a CNN with padding. . . . 17
1.12 Principle of Max-pooling. 18
1.13 The architecture of DispNetC. 19
1.14 The single correlation unit. 21
1.15 The multiview depth based quality enhancement approach that have been employed. 24

2.1 The proposed scheme. 30
2.2 The DENet network architecture. 31
2.3 Examples of computer graphic generated images. 33
2.4 Experimental setup Swiss-ranger SR4000 and stereo cameras. 34
2.5 Texture and depth images captured by SR000 before and after correction. 35
2.6 Examples of calibrated stereo images 36
2.7 Disparity-depth mapping for both real and computer graphic images. . . . 37
2.8 Example of depth estimation for real images using the proposed DENet . 39

3.1 The CNN architecture with the proposed multi-scale correlation layer. . . 43
3.2 The matching process using three single-scale correlation kernels with different sizes. 43
3.3 The multi-scale correlation unit. 45
3.4 The matching process between the left and right features. 46
3.5 Training loss analysis of multi-scale network. 47
3.6 Performance comparison between multi-scale correlation and single scales. 49
3.7 Examples of disparity estimation using multi-scale correlation. 50

4.1 Examples where DispNetC fails to estimate large disparities. 52
4.2 Disparity estimation examples using different re-sampling factors in comparison with *DispNetC*. 54
4.3 The Multi-resolution disparity maps estimation and fusion framework. . . 56
4.4 The architecture of the merging MRDF-Conv network. 58
4.5 The architecture of the mergning MRDF-Conv-Deconv network. 61
4.6 The proposed multi-resolution merging networks MRDF-Conv and MRDF-Conv-Deconv in comparison with single resolutions 64
4.7 Shows some images for subject assessment of the proposed method. . . . 69

5.1 The silhouette-based view synthesis framework for multiview video systems. 73
5.2 The SVSNet framework. 75
5.3 Examples of view synthesis results from silhouette using the proposed framework on computer graphic datasets. 77
5.4 SVSNet evaluation on real outdoor scenes, KITTI 2015. 77
5.5 Examples of view synthesis from silhouette using different threshold values. 78
5.6 Examples of view synthesis from silhouette (proposed method) in comparison HEVC encoder. 79
5.7 Examples of view synthesis from silhouette before and after editing some objects place in the edges map. 80
5.8 Example of silhouette from texture image, disparity map and semantic segmentation map. 81

6.1 The proposed CNN-based framework for multiview video quality enhancement. 85
6.2 The architecture of MVENet. 87
6.3 Examples of HEVC multiview quality enhancement for various QP using the proposed framework on CityScapes Dataset. 88

6.4 Examples of HEVC multiview quality enhancement using the proposed MVENet framework on CityScapes Dataset. 90
6.5 Visual comparison results of the proposed CNN and other JPEG approaches. 93

List of Tables

2.1 Performance comparison (PSNR) of different computer graphic sets. . . . 38
2.2 Depth estimation (PSNR) using real images. 38

3.1 Disparity map errors (MAE). 48
3.2 Disparity errors comparison (MAE) for horizontally elongated patches and complete image. 49

4.1 The influence of using various re-sampling factors for disparity estimation. 55
4.2 Disparity estimation error for various re-sampling factors. 63
4.3 Performance comparison (MAE) of the two proposed approaches with other approaches. 65
4.4 Out-Noc error on KITTI 2015 testing dataset. 66
4.5 MAE comparison between horizontally elongated patches and re-sampling the patches on both horizontal and vertical directions. 67
4.6 Comparision between different re-sampling methods using MAE. 67

5.1 SVSNet evaluation (PSNR) using different edge detector thresholds. . . . 78

6.1 Quality enhancement comparison of the proposed MVENet with the HEVC standard. 89
6.2 Evaluation of the models on different QP factors on Cistscapes and FlythingThings3D datasets. 91
6.3 Quality enhancement comparison of MVENet on Cistyscapes dataset using encoded high-quality views with various QP_H factors. 91
6.4 Comparison with the state-of-the-art algorithms on Cistscapes dataset. . . 92
6.5 Comparison with the state-of-the-art algorithms on FlythingThings3D dataset. 92
6.6 Quality enhancement comparison of the proposed MVENet with the JPEG standard using various Q_H factors. 94
6.7 The the running time comparison of the MVENet with several state-of-the-art methods. 94

List of Abbreviations

AR Augmented Reality

AVC Advanced Video Coding

CG Computer Graphics

CGI Computer Graphics Images

CNN Convolutional Neural Network

Conv Convolutional

Deconv Deconvolution

DIBR Depth-Image-Based Rendering

DispNetC DispNetCorr1D

DENet Depth Estimation Network

FGO Foreground Object

FVT Free-Viewpoint Television

FVV Free-Viewpoint Video

HEVC High Efficiency Video Coding

IHF Ideal Hard Fusion

MAE Mean Absolute Error

MRDF-Conv Convolution-based Multi-Resolution Disparity Fusion network

MRDF-Conv-Deconv Convolution-Deconvolution based Multi-Resolution Disparity Fusion

MSE Mean Squared Error

MVC Multiview Video Coding

MVD Multiview Video plus Depth

MVENet MultiView quality Enhancement Network

MVS Multiview Video plus Silhouette

NN Neural Network

PSNR Peak Signal-to-Noise Ratio

QoE Quality of Experience

ReLU Rectified Linear Unit

SSIM Structural SIMilarity

SVSNet	Silhouette-Based View Synthesis Network
ToF	Time-of-Flight
VR	Virtual Reality
3DV	Three-Dimensional Video
3DTV	Three-Dimensional Television

Chapter 1

Introduction

1.1 Motivation

Multiview video is considered to be the next evolution of the classic stereoscopic video of motion picture formats towards a more natural and realistic 3D visual experience. While stereoscopic systems only require two views, the multiview displays require multiple videos. Multiview video delivery will be the basis to support various applications for scene communication, including Free-Viewpoint Video (FVV) [1], Free-Viewpoint Television (FVT) and Three-Dimensional Video (3DV), Three-Dimensional Television (3DTV) [2]. Unlike conventional 2D video systems, a multiview video system allows users to enjoy a 3D viewing experience and provides them with some special viewing experience such as view sweep and frozen moment. Besides, multiview technologies offer users the ability to freely navigate the scene from different viewpoints within a specific range. Multiview videos recorded by multiple cameras that are positioned and arranged to capture the same scene from different positions and angles, become feasible following the development of small-sized, cheaper texture and depth cameras and the more powerful processing units available in todays markets. An example is an autostereoscopic display which enables different viewers to perceive motion parallax and experience free viewpoint video. Although setting up more cameras leads to the capture of more video streams from different viewpoints, it can provide users with a more realistic 3D viewing experience. However, this imposes a huge load on the storage, compression, and transmission of multiview video data, in particular for wireless channels where the bandwidth is limited, making real-time multiview data delivery and interaction a challenging task.

Several multiview video data formats and associated coding technologies are emerging with the goal of achieving more flexible representation and more efficient multiview video compression methods. Over the years, Multiview Video Coding (MVC) was the process by which multiview video streams are efficiently encoded, exploiting not only the redundancies between temporal frames for a given view but also the similarities between frames of neighboring views. The topic of multiview video coding has been an active research area for more than three decades and several video formats have been developed to provide higher compression of multiview video content. In 1996 the international video coding standard H.262/MPEG-2 Video [3] was updated to MPEG-2 Multiview Profile [4] to support multiview video coding by using features designed for temporal scalability. However, this extension was never introduced to markets for several reasons including the lack of display technology and hardware processing capabilities, also the transition from analog standard to the high-definition digital video was a huge challenge in itself.

Later in 2008, the successful H.264/MPEG-4 Advanced Video Coding (AVC) [5] standard was extended to multiview video coding [6] featuring more efficient interview prediction schemes and higher compression gains. In the MVC standard, both interview correlation and temporal similarities are explored, images are not only predicted from temporally neighboring images but also from corresponding images in adjacent views. However, it was clear in MVC that the multiview data size is proportional to the number of views in the multiview video [7]. Consequently, in order to support a large number of views required by high-quality autostereoscopic multiview displays new solutions were needed. In addition, the displaying of realistic 3D scenes based on multiview videos that are captured from limited viewpoints at a high-quality was not supported by technology efficiently. Furthermore, conventional multiview camera systems need to be set accurately, which puts constraints on the post-processing stage and consequently limits potential applications.

Recently, a depth map, which represents the distance from objects in the scene to the capturing camera, together with aligned cameras, are exploited to describe 3D objects in the scene. Here is the Multiview Video plus Depth (MVD) format [7, 8, 9] is a promising method to represent 3D video content, and recent extensions supporting this format have been introduced [10, 11]. With the MVD format, only a small number of views associated with their depth data are required to represent the 3D multiview video, and the amount of bitrate allocated to texture and depth views can be properly tuned [12, 13]. At the decoder or display side, Depth-Image-Based Rendering (DIBR) algorithm [14, 15] is used to synthesize additional viewpoints (i.e, virtual views). DIBR is a technique for rendering virtual views for the MVD 3DTV format. This method is a key feature for the success of 3DTV delivery, solving the critical issue of MVC without introducing substantial bit rate increases. Both stereoscopic and multi-view displays can take advantage of this format: the former to adapt the content to the specific stereo baseline and allow users to adjust the depth perception, the latter to render a large number of output views and cover wide viewing angles. The multiview video plus depth representation used for 3DTV and FVV. For example, stereo image pairs can be generated by view interpolation from one video and depth data associated with each sample. From the coding efficiency point of view, the video plus depth format is very valuable, as the per sample depth data can be regarded as a monochromatic, luminance-only video signal. However, the main issue in this representation is that DIBR methods require high-quality depth maps. Over the past years, depth map estimation was an active research field due to the difficulties and challenges for estimating accurate, robust and high-resolution depth maps for different scenarios.

Moreover, in the recent decade, important breakthroughs have been achieved based on deep learning methods. In particular, Convolutional Neural Networks (CNN) have been successfully applied to many tasks in the area of computer vision, including object classification [16] and detection [17]. New models have been designed to provide per-pixel estimations like semantic segmentation [18], depth estimation from a single image [19] and disparity estimation from a pair of stereo images [20]. In [20], a deep CNN that performs the matching process between the left and right CNN feature maps is employed. However, in this network pooling is applied to input images which reduce the resolution of the features before the matching process. In addition, the maximum displacement in the correlation layer is static. Therefore, this network is unable to estimate reliable and accurate disparity maps for close objects with large displacements as well as far objects with small disparities. In this thesis, the feasibility of estimating an accurate depth map is investigated by proposing novel disparity estimation methods. In the proposed methods convolutional neural networks are used as a core for the depth estimation frameworks.

Additionally, in this research, a novel framework that addresses the main problem of multiple views representation is proposed. It is proposed to handle this problem by representing required views using only their edges while dropping their texture content. The texture content gets synthesized by a CNN from a high-quality reference view. The proposed concept of a multiview video system is similar to a multiview video plus depth system, but instead of using the disparity(depth) maps the edges of the view are used. Meanwhile, the edges help to generate accurate virtual views, without having shape deformation for synthesized objects.

Finally, another approach has been introduced aimed at the enhancement of compressed multiview videos. Large information redundancy and a vast amount of multiview video data can be reduced using asymmetric multiview video compression [21, 22, 23], where the views are encoded with different qualities. Only several viewpoints are kept with high-quality and other views are highly compressed to low-quality. However, highly compressed views may incur severe quality degradation. Thus, it is necessary to enhance the visual quality of highly compressed views on the decoder side. Exploiting similarities among the multiview images is the key to efficiently reconstructing the multiview compressed views. In this thesis, multiview quality enhancement is investigated. The main idea of this research work is to directly learn an end-to-end mapping between the low-quality and high-quality views and recover the details of the low-quality view. This network takes a low-quality image of one view and a high-quality image from a different viewpoint of the same scene as inputs and outputs an enhanced image for the low-quality view. To the best knowledge of the author, this is the first work for multiview video enhancement where neither a depth map nor a projected virtual view is required in the enhancement process.

The chapter is organized as follows. Background information on multiview 3D video capturing and coding is provided in Section 1.2. Section. 1.3 presents an overview of the most popular methods that have been used for depth and disparity estimation, which is a key principle for rendering virtual views for multiview video systems. The concept of view synthesis for multiview videos is provided in Section. 1.4. Since in this thesis the convolutional neural networks have been used as the main engine for introducing the proposed approaches, thus convolutional networks are explored in Section 1.5. After that, employing convolutional networks for multiview video enhancement is presented in Section 1.6. Finally, This chapter concludes with the contributions of the thesis research and a summary of the remaining chapters.

1.2 Overview of Multiview 3D Video

1.2.1 3D Dimensional Video Capturing and Display

Three-dimensional video is not a new concept at all. The first system that enables stereoscopic vision using a pair of stereo images was invented using a device with a mirror. This device was able to merge two perspective images into a single one and give the viewer the impression of depth. Then the first 3D motion pictures from stereo images, which uses the anaglyph technique, were invented in 1889 by William Friese-Green [24]. Until now, the anaglyph 3D technique has been used on traditional television.

Currently, there is an investment in digital media markets substantially in 3D displays and technologies, glasses-less 3D screen, virtual reality, and 3D cameras. Most of the 3D technologies still need glasses, passive glasses or active glasses. The passive glasses use a color filter or polarized glasses. While the active glasses use shutter like technology, which can be exploited on high frame rate televisions or monitors.

Viewing 3D content without glasses or goggles has proved to be one of the toughest things for interface designers to achieve, it is possible nowadays to provide it but the scene never really looks right when user observe it from different angles. The glasses-less 3D display technologies, called autostereoscopic enables viewing 3D content from different angles and different position (multiview video) without wearing special glasses. As shown in Fig. 1.1, there are two main principles used in autostereoscopic technologies: parallax-barrier and lenticular. Some 3D television has exploited autostereoscopic technology. However, the 3D depth perception can only be realized well at some fixed distance, which is a disadvantage of this technology.

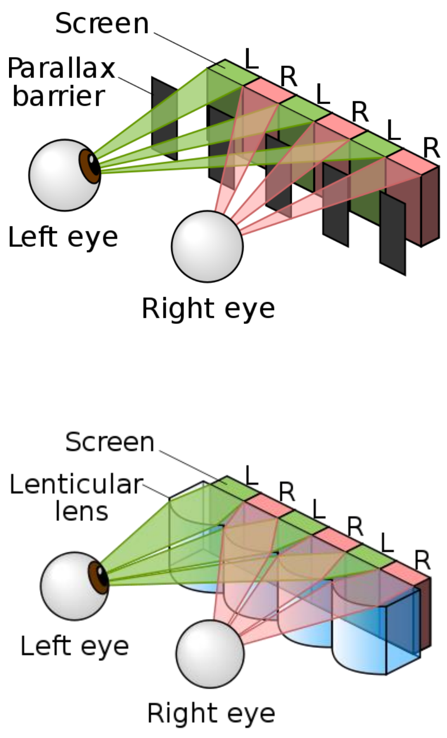

Fig. 1.1: Autostereoscopic technologies: parallax-barrier (upper image) and lenticular (lower image) [25].

Currently, the head mounted display for Virtual Reality (VR) and Augmented Reality (AR) are becoming more popular. A simple VR glasses consist of a display where both the left and right images are displayed on the left and right parts of the screen, and then the two eyes can see different views using special lenses.

In terms of capturing 3D video, the technologies can be divided into two types: multi-camera (normal texture camera) based capturing and range imaging (depth camera) based capturing. For multiview video capturing, an array of cameras is arranged to capture the scene from different view angles, where the camera settings, calibrations, and synchronization are very challenging tasks [26, 27]. Some manufacturers produced a series of integrated stereo or multiview cameras, but the limitations are image quality and flexibility. When an array of dense cameras is used, then almost every direction of light could be recorded, which is called light field camera [28].

Depth camera opens a new direction of capturing 3D image or video, which measures the distance between the scene objects and the camera (called depth map). A texture image with the corresponding depth map is able to build an incomplete 3D model, and then some nearby virtual views are generated from such a model. When there is an array of texture plus depth camera, the multiview plus depth capturing can provide free-view-point video.

1.2.2 3D Videos

For stereo video, where two cameras are used to record videos, it is not efficient to encode the two views independently because the two views represent the same scene from only slightly different view angles. Therefore, a conventional stereo video coding method uses inter-view prediction as shown in Fig. 1.2.

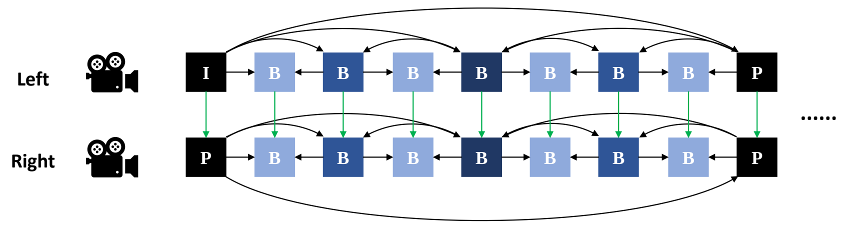

Fig. 1.2: A conventional stereo video coding method with inter-frame and interview prediction.

To reduce the number of views required to be transmitted and avoid redundancy among multiple viewpoints of a 3D video, the inter-view prediction [29] is also exploited in MVC [7] extension of the H.264/AVC and H.265/HEVC standard [30] as shown in Fig. 1.3.

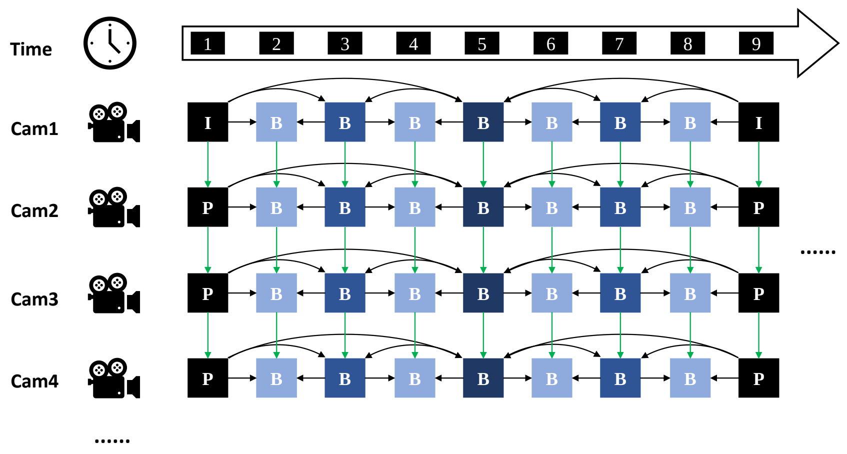

Fig. 1.3: An example coding structure in MVC for linear five camera setups and GOP size of eight pictures.

1.3 Depth and Disparity Map Estimation

Acquiring high-quality depth maps is a challenging task in the field of 3D computer vision and many 3D related applications such as image-based rendering, MVD, visual tracking, human-computer-interaction, robot vision, industrial inspection, virtual viewpoint synthesis, medical imaging, and 3D maps. These applications require high accuracy and resolution depth maps. Whilst acquiring textured images with high quality is relatively straightforward, acquiring real-time accurate depth maps is still difficult.

In this section, the main methods that solve the depth rendering task, shown in Fig 1.4 are briefly reviewed, among them, structured light, time-of-flight, and stereo vision based approaches. The conditions and limitations of these strategies are presented and discussed. Based on [31, 32, 33, 34] depth map measurements are shown in Fig. 1.4, which can broadly divided into two approaches, active and passive.

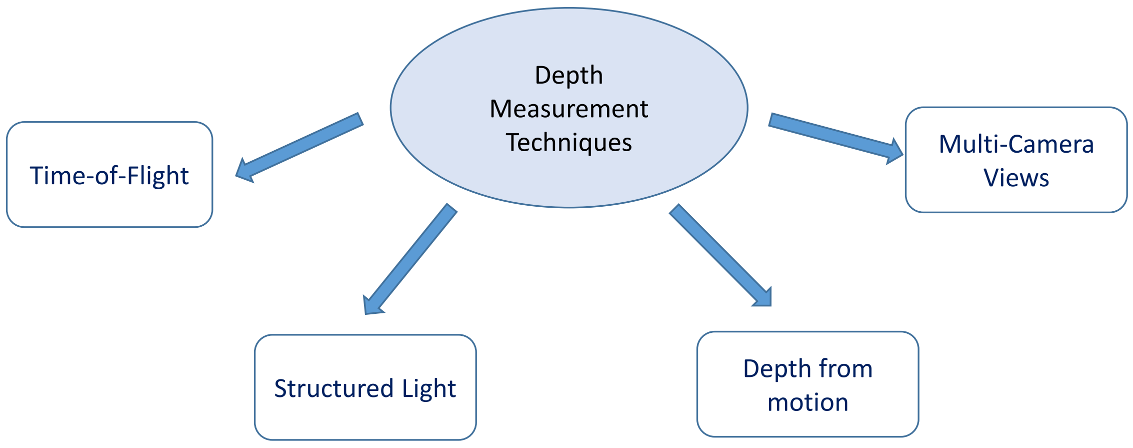

Fig. 1.4: Depth map measurements.

1.3.1 Passive methods

Passive methods capture the scene using one or more cameras without any additional illumination. The depth is then estimated based on the captured images. This approach includes several techniques which are explained in the following sections.

Depth from Motion Blur Blur is an artifact that occurs when a camera captures an image with irregular conditions such as camera shake, the failure of auto-focusing or object motion. In this motion blurred image the blur is greater for objects closer to the camera than for objects further away [35]. Deblurring and depth estimation are indeed highly related if the right setting is chosen; depth and blur are basically in one-to-one relation. If the depth is known then the blurriness for an object can be estimated. Conversely, if the blur for different objects can be measured in the blurred image then the depth map can be estimated [36]. Unfortunately, deblurring is a very difficult problem.

Depth from Focus Depth estimation from focused and defocused images is the problem of rendering the 3D information of a scene from a set of images. The images are captured with different camera settings, typically by changing the focal length settings or the sensor plane position [37]. The images are taken from the same point of view, thus the camera must be fixed and the scene static. Therefore, calculating the depth of the scene using the focus method is achieved by taking multiple images of the scene with several focus settings and applying a method to decide the best-focused image for each point or object in the scene. The best-focus for a point in the scene is related to the depth of that point. The main disadvantage of this technique is the trade-off between the accuracy and the spatial resolution and it has poor performance for regions with low texture content [38].

Stereovision A lot of research has been performed in the area of depth estimation based on stereo vision systems. Among the passive range sensing methods, stereovision is probably the most promising technique, least expensive and most widely applied. In this field, significant progress has been made in recent years [39]. A pair of stereo images consists of two rectified images of the same scene, taken from slightly horizontally separated points as shown in Fig. 1.5, before finding the correspondence between the two views by matching the two views, a calibration method is required. In many cases, finding the correspondence between images is a difficult task. This may be due to image sensor noise, repetitive structures and textures, texture-less regions, reflections, and occlusions. However, the correspondence problem can be solved by replacing one of the two cameras by a projector that emits a light pattern onto the measuring scene; this system is called structured light and is one of the active methods discussed in the next section.

1.3.2 Active Methods

Structured Light Illumination (Active Stereo) The correspondence problem in stereovision can be simplified by applying active methods based on the structured light concept instead of using passive methods. In this approach, one of the two stereo system cameras is replaced by a projector [31]. While the projector projects a pattern on the measuring surface the camera images the illuminated scene. Then the depth information can be extracted by analyzing the deformations of the captured pattern. Now the correspondence problem can be solved and the depth can be estimated after matching each point of the imaged pattern with the corresponding point of the projected pattern. Structured light systems are widely used for depth sensing and for the 3D reconstruction of static objects

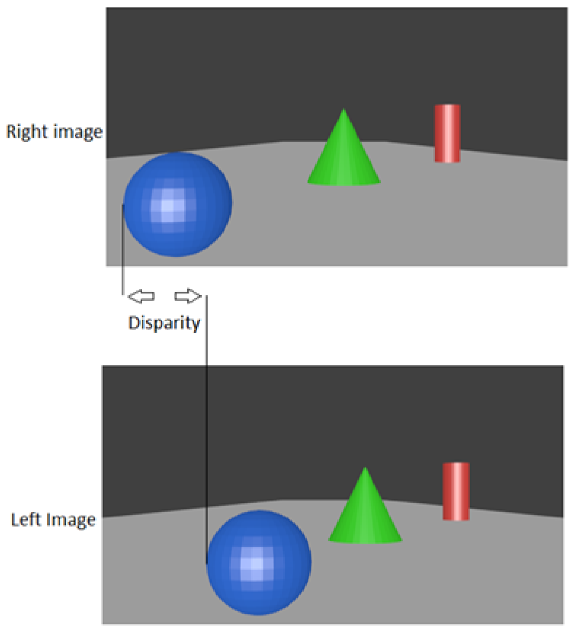

Fig. 1.5: Horizontal displacement in a stereo vision system.

for indoor scenarios. It is one of the most accurate methods for depth sensing and it is easy to implement.

Structured light illumination methods differ according to the way in which every point in the pattern is identified and with the number of patterns needed [33]. One of the successful structured systems is Kinect [40] which was released by Microsoft in 2010 and allowed great progress in Human-machine interaction. Without any sensors attached to the body, players can send commands to the Xbox 360 directly by simple hand or limb motion. The Kinect has a depth camera based on structured light for generating depth maps. The structured light system has the advantage of a large scanning area and low-cost hardware, however, the depth map acquired suffer from two main problems, severe occlusion problem, and low resolution. Many applications need accurate, high-quality and high-resolution depth maps, such as object recognition and DIBR.

Time-of-Flight Time-of-Flight (ToF) range cameras [41] are relatively modern devices that capture digital images together with distance information. In general, the ToF cameras provide more accurate depth maps than those of the structured light cameras and the ToF cameras almost has no occlusion problems unlike structured light approaches and the edges of the depth images are more accurate. However, cameras using this method currently have several limitations including low resolution, low frame rates and they are very expensive. Moreover, the depth map obtained by this technology can be very noisy and affected by the ambient light, limiting its use to indoor applications.

1.4 View Synthesis

In this section, the concept of view synthesis for multiview videos is presented. View synthesis is the task of generating new images of a scene from existing views as it would appear from certain viewpoints. View synthesis is a challenging problem, the fundamental limitation of rendering novel views is the degree of realism that can be achieved.

In the following, we will review two main approaches to synthesize virtual views.

1.4.1 View Synthesis using Stereo and Multiview Images

The basic problem considered is as follows: given a set of images of a scene from different viewpoints, can we predict an image of the scene from a new viewpoint? In recent years, image-based scene representation has received much attention. A possible solution provided by using stereo images, for example, consider the two images in Fig. 1.6 showing a monkey standing on a log in a forest. Given these two images, is it possible to predict a view from a new viewpoint?

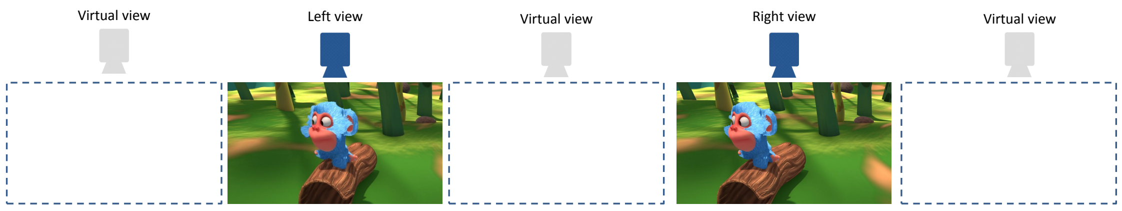

Fig. 1.6: The monkey image pair. Two computer graphic images of a scene taken from different viewpoints.

In multiview videos case, several works in this area focus on using multiple views to interpolate new views in-between [42, 43, 44, 45], however they have some limitations. For example the work [42] addresses content creation for multiview autostereoscopic displays using phase-based motion magnification. This method does not require disparity information but only supports small disparity ranges and may refine important details.

Recently, novel view synthesis methods have been addressed using neural networks, in [43] Ji et al. present a CNN based method. First, it rectifies the two input view images and uses the rectified images to estimate homography by deep networks, and then intermediate views using other deep networks are rendered. The recent work by Flynn et al. [46] uses a deep CNN to learn the geometry of a scene to synthesize an intermediate view of a scene from a set of input images. This method can produce good quality novel views for small baseline and has difficulties for large baseline and moving objects. Jaderberg et al. in [47] introduce a CNN model able to learn the relationship between the input image and the output image by performing spatial transformation on features maps. In [48] the work presents 2D to 3D synthesis framework by synthesizing stereo image pairs from a single input image. This network synthesizes a right view of the input left view, however, no parameters are used to control the viewpoint of the rendered image. In [44, 49] they use multiple images as input, which are captured from different four corners, to reconstruct a light field image. Srinivasan et al. [50] train a CNN model to render light field images from a single input image. However, using a single input view suffers from a critical problem that happens when regions covered by foreground objects in the original view are disoccluded in the synthesized views.

1.4.2 Depth Image-Based Rendering

Depth-image-based rendering is one of the most popular techniques that might be used at the receiver or display side by the decoder to synthesize additional realistic views at a slightly different viewpoint, using a textured image of a specific view with the associated disparity map, DIBR-based method is shown in Fig. 1.7. In general, DIBR techniques generate novel views in two steps: first, the points of the input texture image points are re-projected into the 3D world, using the corresponding depth data. After that, these 3D space points are projected into the image plane of a virtual camera, which is located at the required viewing position. The process of re-projection (2D-to-3D) and followed by projection (3D-to-2D) is usually called 3D image warping. The main drawback raised when occluded regions by foreground objects in the original views become visible in the rendered views. Disocclusions are typically less critical in view interpolation than single view-based approaches, as uncovered textures in background regions and objects' boundaries are usually visible by one of the neighbor views. Consequently, an inpainting method of disoccluded scene regions in the synthesized view is required.

More importantly, the view-quality obtained by DIBR is strongly affected by the accuracy of the estimated disparity. In an attempt to enhance rendered quality-views and more realistic views, accurate depth maps are required in the warping process. Consequentially, over the past years, depth estimation was an active research field. Eigen et al. [51] used a multi-scale CNN to estimate the depth from a single image, where a coarse scale network is used to perform depth prediction with low resolution, and a fine scale network is trained to perform local depth refinements with higher resolution. Laina et al. [19] introduced a deep fully convolutional neural network with residual learning to model the mapping problem between monocular images and depth maps.

CNNs have been also investigated as a possible solution to the stereo correspondence problem. Zagoruyko and Komokadis [52] exploited CNN to learn a general similarity function by using two channels siamese models. Zbontar et al. [53, 54] used a convolutional neural network to determine corresponding patches in left and right stereo images.

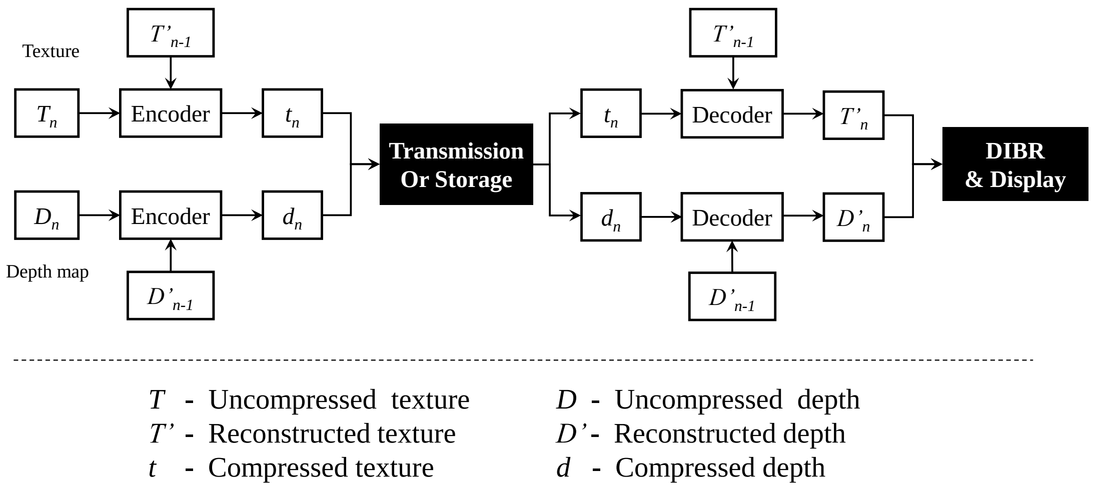

Fig. 1.7: The structure of view synthesis approach using a DIBR-based method.

These reconstruction methods include semi-global block matching for the depth of each pixel with a series of post-processing steps, which renders the model computationally intensive. Garg et al. [55] introduced an unsupervised approach for monocular depth estimation with CNNs. In their work, they propose a data augmentation technique to deal with the cost of acquiring real images with depth ground truth. However, the augmented dataset has to be generated from already acquired images, and thus this technique is unable to generate unseen environments. There has been steady progress in stereo depth reconstruction algorithms, but the quality is not yet good enough for 3DTV broadcast and 3D movies. It is especially difficult for the task of disparity estimation for scenes that contain textureless materials, defocus blur, motion blur, transparent objects, and specularity.

In this thesis, a novel and accurate method for depth estimations have been investigated.

1.5 Convolutional Neural Networks for Disparity Estimation

Traditionally, digital image processing is based on methods that relied on hand-crafted filters. These filters could capture specific, non-trivial features crucial to a problem in computer vision or multimedia. Examples of these features include edge detectors, object detectors, and other detectors. Researchers apply these hand-crafted filters to the input image in order to enhance predictions. However, such filters tend to produce very specialized representations for a problem. In other words, finding the most optimal filter for a specific task is both computationally and labor-intensive. In recent years, deep learning methods have been used to learn and extract optimal features. The majority of works are dominated by deep learning methods that leverage artificial Neural Networks (NNs) to make predictions and additionally learn representations. The ultimate goal of training deep networks is to produce abstract, useful representations that are generalizable and transferable to different inputs.

Using NNs have made a major breakthrough in image processing, including face and object recognition, person recertification, classification and scene understanding. The neural networks mainly have three architecture types:

- The multilayer perceptrons, that are simple feedforward networks that generate a set of outputs from a set of inputs.
- The Convolutional Neural Networks, particularly employed for image processing as they take images as inputs.
- The recurrent neural networks, used for consecutive data such as speech or text.

This thesis has introduced novel approaches using convolution neural networks for disparity estimation task, view synthesis and quality enhancement for multiview video. For this reason, this section provides the necessary technical background of CNNs. More comprehensive details and other architectures could be found in the deep learning book [56]. CNNs are a special type of neural networks that are originally used in image and video processing applications. The key advantage of using CNN compared to traditional methods in image and video processing is that CNNs are able to automatically detect the important features during the training process without the need for human selection or supervision. CNN is also computationally efficient, as it uses special convolution and pooling operations to reduce the forward runtime.

1.5.1 Architecture Overview

Convolutional neural networks consist of basic elements called neurons, these neurons have learnable weights and biases. Each neuron, which has one or a set of inputs, performs a dot product among the inputs and the weights after that a non-linear operation is followed. The convolutional neural network employs at least one loss function in the training process. All these elements including inputs, learnable kernels, biases are used in all neural network structures. What makes CNN architectures different is that they use images as inputs, which provides specific properties into the architecture. Consequentially, the architectures of CNN are designed in an efficient way by taking advantage of having images as inputs. In particular, the layers of CNNs have volumes of three dimensions:

width, height, depth. In other words, the output of each layer is a set of feature maps, each feature map has the $width \times height$ with a total number of $depth$. In the following sections, we will present a CNN in more details.

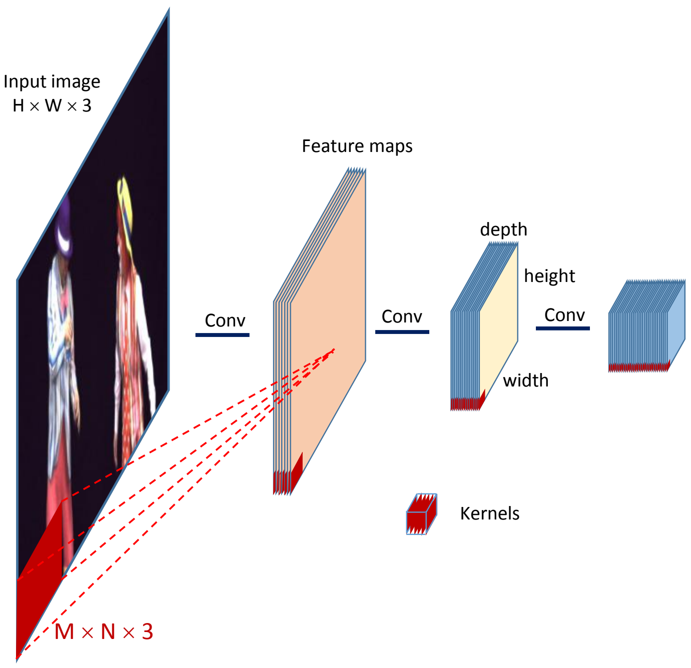

Fig. 1.8: A CNN represents the inputs/outputs of each layer in dimensions (width, height, depth). Each layer processes the 3D input volume into a different 3D output volume.

Usually, a CNN is composed of a sequence of layers and mainly three types of layers are used to design CNN architectures: convolutional or deconvolutional layer, pooling layer, and fully-connected layer. Each type of layer performers forward and backward propagation.

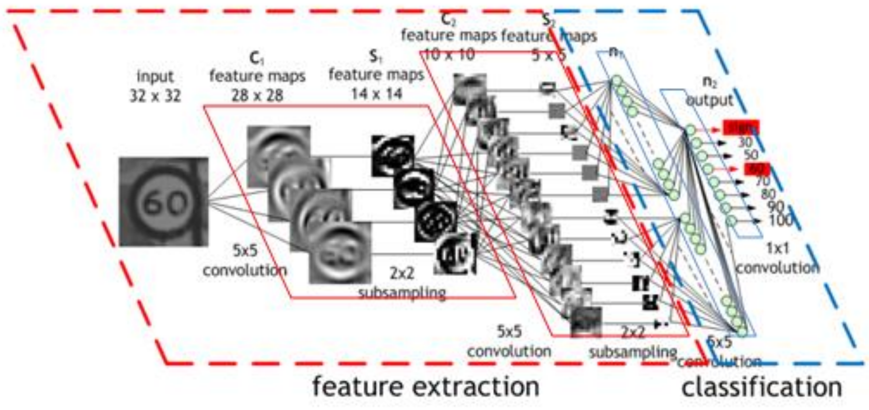

Fig. 1.9: Typical structure of Convolutional Neural Network [57].

Up to date, there is no specific way on how to stack the layers in order to build the network. However, most recent CNN structures process the inputs in two sections. In the first section, meaningful feature maps are extracted by using both convolutional and pooling layers. Whereas, the structure of the second section depends on the network task. Moreover, the depth of the network or the number of stacked layers also depends on the problem that the network is trying to solve. An example for CNN used for classification, where fully connected layers are used, is illustrated in Figure. 1.9.

1.5.2 Convolutional Layer

The Convolutional Layer (Conv) is the most important unit of a convolutional neural network. The input for the first convolutional layer is an image or a set of images, could be gray or colored images. Then the convolution is applied on the input images using a kernel to produce feature maps. Whereas, the input for the other Conv layers is feature maps extracted from previous layers. The Conv layer's parameters contain learnable kernels (filters). The input is convolved with a set of kernels to generate features, that will be the input for the next layer. In order to process the three dimensions inputs of a Conv layer kernels also have three dimensions: width, height, and its depth. For example, a kernel in the first layer might have the size of $7 \times 7 \times 3$ (i.e. 7 pixels width and height, and 3 for the case of using colored images with three 3 channels, i.e., the color channels). Also, the kernel in a first layer could have the size of $5 \times 5 \times 1$, in this case, the width and height of the kernel are 5 pixels and 1 for using a gray image with on channel. During the convolution process in the forward pass, the kernel is convolved over the width and height of the input volume and dot products between the elements of the kernel and the input at any position are computed. By sliding the kernel across the width and height of the input volume a two-dimensional feature map is generated, providing the responses of that kernel for each spatial position. The correlation between an input image I and a kernel K is defined by the following formula:

$$C(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n) \times K(i-m, j-n) \tag{1.1}$$

Which equivalently can be written:

$$C(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n) \times K(m,n) \tag{1.2}$$

Usually, the second formula is adopted in deep learning libraries, as it allows less range of valid values of m and n. This convolution can be viewed as sum-of-product, where the product is the dot product matrixes. The convolution of a single channel of an image with a filter is shown in Fig. 1.10. More precisely, the convolution operation is performed by sliding the kernel over the input. At each position, element-wise matrix multiplication is done and then the results are summed. The obtained values after the sum operation present the feature map. In this example, it's called a 3×3 convolution due to the size of the kernel.

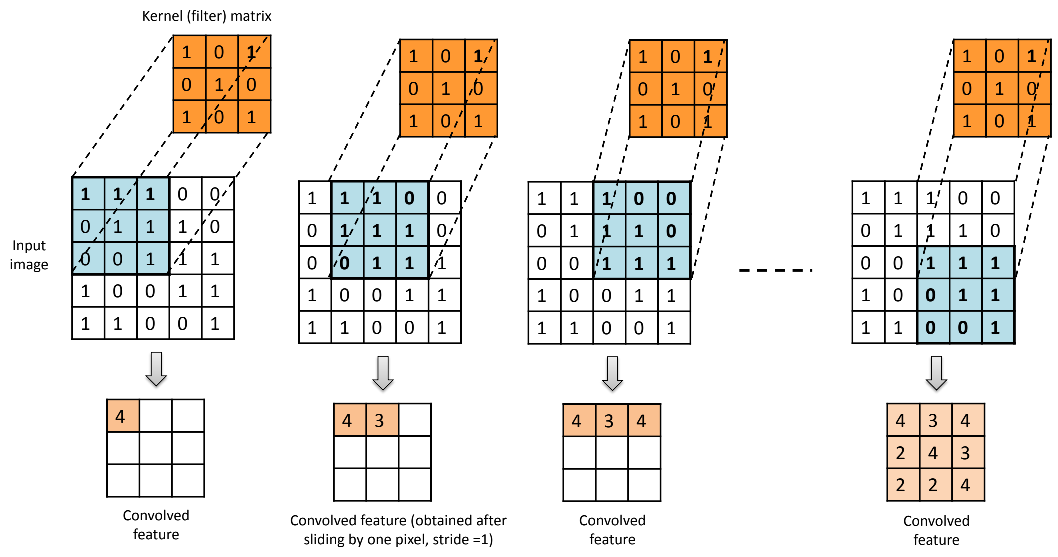

Fig. 1.10: This figure demonstrates a convolution within a CNN without padding. The matrix in the first row presents the kernel. Whereas, the big matrix in the second row defines the pixel values of the input image. Together these produce a convolved feature. The krnel slides over the image producing the final output on the right (down).

The step for shifting the kernel over the input during the convolution operation is decided by the stride, the default value is one. However, this default value can be modified to larger values. Using bigger strides reduces the overlap among receptive fields. Another advantage of using bigger strides is that it produces smaller feature maps than the input. Moreover, the padding parameter defines the kernels action toward the edges of the image, as the kernel should be contained in the input each time it convolves with the input. As a result, the size of the feature map is smaller than the input. For producing feature maps with the same size of the input, zero padding is used. In this case, the required amount of padding elements around the edges of input is inserted. This amount is determined by:

$$p = (M - 1) \div 2 \tag{1.3}$$

where M is the width of the used kernel. Otherwise, the feature map is reduced by the $2p$. Reduction of the size of the feature map can be in some cases desirable. Zero padding is illustrated on Fig. 1.11.

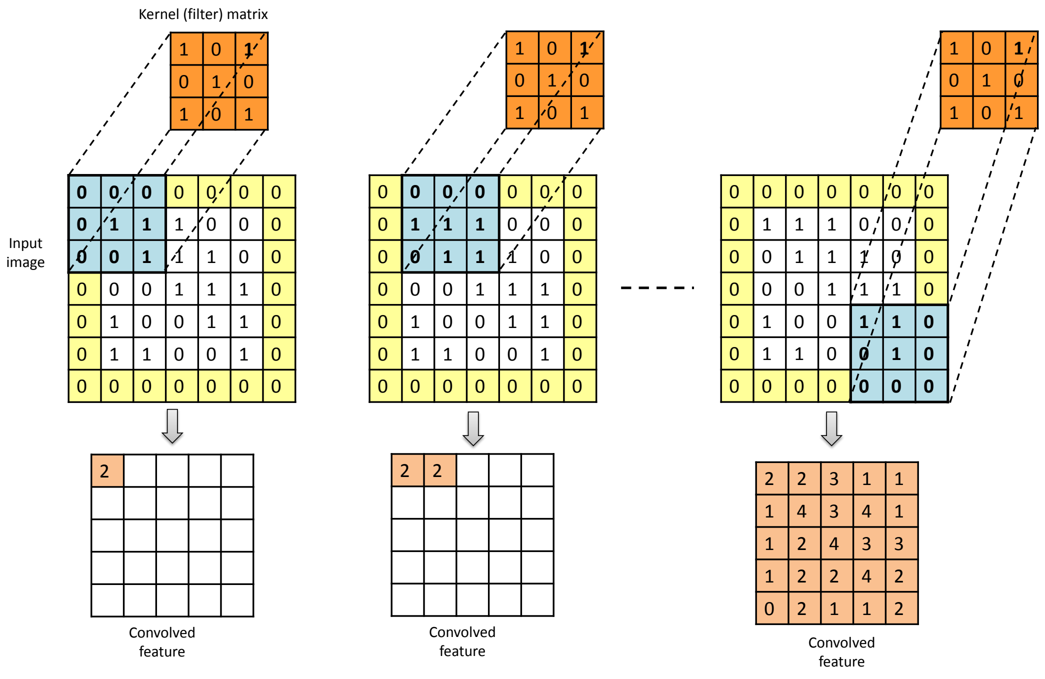

Fig. 1.11: This figure demonstrates a convolution within a CNN with padding (zero padding on the borders of the input image).

1.5.3 Rectified Linear Unit

In this step, an activation function is used to increase or introduce non-linearity in the CNN. The Rectified Linear Unit (ReLU) is the most commonly used activation function in convolutional neural networks architectures. The function returns zero if its input is a negative value, while for positive input values it returns the same value back. This function can be written as:

$$f(x) = max(x, 0) \tag{1.4}$$

where x is the input value. This function is computationally cheap, therefore the network model takes less time to train or run.

1.5.4 Pooling Layer

Spatial invariance is a concept where the position of a structure or an object in an image has no effect on the ability of the neural network to detect its specific features. Pooling enables CNNs to detect features in various images irrespective of the difference in lighting in the pictures and different angles of the images. In addition, pooling is used as an operation to reduce the number of parameters, which accelerates the training procedure and helps in avoiding overfitting.

Usually, pooling is performed after a convolution operation to reduce the size of the feature maps. This enables us to reduce the number of parameters, which both shortens the training time and combats overfitting. Pooling layers downsample each feature map independently, reducing the height and width, keeping the depth intact, and this layer typically requires no learning process. The main target is to divide the input into non overlapped output units. This helps to reduce the size of the feature maps while retaining the most important information contained in the input layer.

There are different types of pooling, for example, max pooling and min pooling. Max pooling, which is the most used type, works by placing a matrix of 2×2 on the feature map and picking the largest value in that box. The 2×2 matrix is moved from left to right through the entire feature map picking the largest value in each pass. The concept of max pooling is depicted in Fig. 1.12

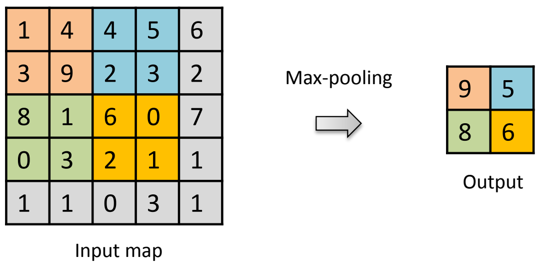

Fig. 1.12: Principle of Max-pooling.

These values then form a new matrix called a pooled feature map. Max pooling works to preserve the main features while also reducing the size of the image. This helps reduce overfilling, which would happen if the CNN is given too much information.

1.5.5 DispNetCorr1D Architecture

Since the convolutional neural network *DispNetCorr1D* (*DispNetC*) [20] is a core building block for our research work, we think it is useful to briefly introduce the *DispNetC* network, and show how our work integrates this network. It worth mention that *DispNetC* network has achieved state-of-the-art for the disparity estimation task from a pair of rectified stereo images. The architecture of the original *DispNetC* network is illustrated in Fig. 1.13.

Feature Extraction

In order to predict the disparity map, a matching process between input views is required. In this network two identical processing streams with convolutional layers of *DispNetC*, which separately process the two images, are designed to aid the network in

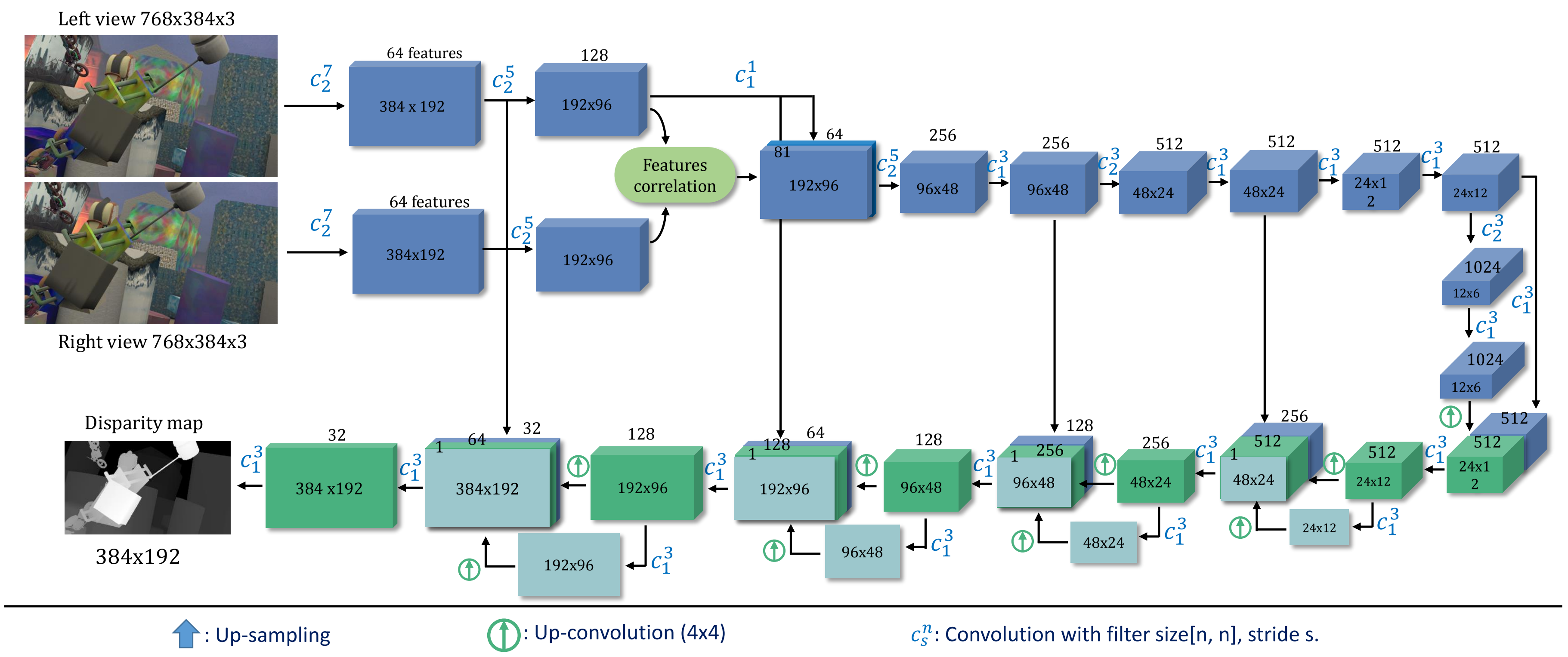

Fig. 1.13: The architecture of DispNetC.

the matching process. With this architecture, the network firstly extracts meaningful representations of the two images separately, and then the obtained features are used in the correlation process. The feature extraction module can be expressed as:

$$\mathbf{F}_{i,k} = max(0, \mathbf{W}^c_{i,k} * \mathbf{F}_{i,k-1} + \mathbf{b}^c_{i,k}), \tag{1.5}$$

where $\mathbf{W}^c_{i,k}$ and $\mathbf{b}^c_{i,k}$ (superscript c for convolution) denote the convolution filter and the bias for layer k and branch i, respectively; $\mathbf{F}_{i,k}$ indicates the output feature set of the layer k for branch i, with $\mathbf{F}_{i,k-1}$ denoting the input feature set. The operator $*$ represents convolution with a stride of 2. Each stream has two layers of feature extraction, and the result is a set of 128 feature maps. The two sets of feature maps will be passed to the correlation unit.

Correlation Unit The reconstruction process requires to find correspondence between the two views. The correlation unit (as depicted in Fig. 1.14) of *DispNetC* is the basic layer in the matching process. The following paragraphs explain how the correlation unit works. Let us suppose that $\mathbf{F}_{l,2}$ and $\mathbf{F}_{r,2}$ are the left and right set of feature maps for the second convolutional layer, respectively; $H \times W$ is the size of the feature and K is the total number of features for each branch. In addition, let $N \times M$ be the size of the correlation block.

For the parallel camera alignment setting, the most common setting for multiview cameras, the differences between the various views are mainly due to objects' horizontal translation. As a result, the 1D correlation which only covers the horizontal displacement is sufficient.

The correlation layer between the two features will produce feature maps with the same spatial size. The correlation map for s pixel displacement is:

$$F_s(x,y) = \sum_{n=1}^{K} \sum_{j=0}^{N} \sum_{i=0}^{M} F^n_{l,2}(x+i, y+j) \times F^n_{r,2}(x+i+s, y+j) \tag{1.6}$$

where $F^n_{l,2}$ and $F^n_{r,2}$ represent the left and right second layer feature map for channel n, respectively. In other words, $F^n_{l,2}$ and $F^n_{r,2}$ are n-th channel of feature map sets $\mathbf{F}_{l,2}$ and $\mathbf{F}_{r,2}$, respectively. The first block is copied from the left feature map for a given center position of correlation, while the second block is copied from the second feature (i.e., right feature map) with a displacement of s. An element-wise multiplication between the two blocks is conducted. The multiplication result, which is a vector, is summed up, and the obtained value is set to the same position in the new feature that represents the matching score for displacement s between the two views.

Contracting Part The feature maps obtained from the correlation unit are concatenated with the feature maps of the high-quality stream and passed to the contracting section.

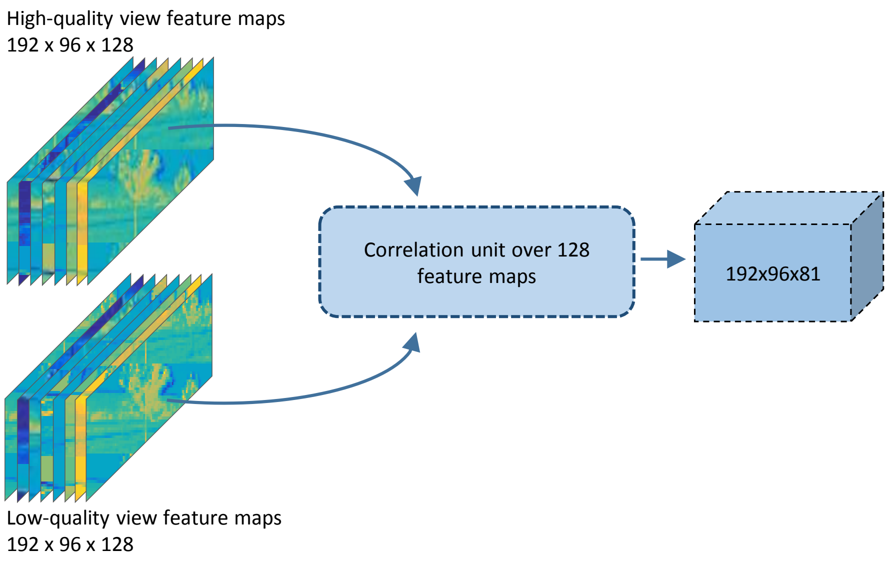
High-quality view feature maps
192 x 96 x 128
Correlation unit over 128 feature maps
192x96x81
Low-quality view feature maps
192 x 96 x 128

Fig. 1.14: The single correlation unit. The input images are separately processed, using two convolutional layers, to generate meaningful representations. Then the correlation layer performs multiplicative patch comparisons between these two set of feature maps. The obtained features are concatenated together then passed to the following network.

It is worth noticing that in this part the network involves the feature maps of the high-quality stream as they contain high-frequency information and details of the scene. In particular, the concatenated features are processed through a number of convolutional layers with a stride of 2, allowing to progressively decrease the spatial size of the extracted features, and providing large receptive fields for higher level convolutional layers, which in turn enables the network to capture more global information. The large receptive field is an important aspect of *DispNetC* architectural design, as it allows the network to deal with large object displacements between two views. The feature generation in the contracting part is given by the following equation:

$$\mathbf{F}_k = max(0, \mathbf{W}_k^c * \mathbf{F}_{k-1} + \mathbf{b}_k^c), \tag{1.7}$$

where we only have one main branch to process in the contracting part.

Expanding Part Given the low-resolution features of the contracting part, the up-sampling operation is performed to increase their spatial size to the desired resolution, the size of the input images. Instead of using standard interpolation methods, this network employs a learning-based up-sampling operation. This strategy is widely adopted in CNN for up-sampling, which is deconvolution [58]. Deconvolution uses backward convolution in which it diffuses a set of feature maps to another set of larger feature maps. Suppose that up-sampling by a factor of $s = 2^k$ is required, then k deconvolution layers (levels) are needed in the up-sampling process. The feature up-sampling can be expressed as follows:

$$\mathbf{F}_k = max(0, \mathbf{W}_k^d * \mathbf{F}_{k-1} + \mathbf{b}_k^d), \qquad (1.8)$$

where $\mathbf{W}_k^d$ *(superscript d for deconvolution)* and $\mathbf{b}_k^d$ denote the trainable kernels and bias of the layer k, respectively; $\mathbf{F}_k$ indicates the up-sampled features in the layer k (output), with $\mathbf{F}_{k-1}$ the low resolution feature maps in the layer $k-1$ (input).

In the expanding process, the feature maps obtained from the deconvolution layer are concatenated with their corresponding ones (which have the same spatial resolution) from the contracting part of the high-quality branch and input into a convolutional layer. The obtained feature maps are then fed to the following deconvolutional layer. Using all these features preserves both the local and high-level information. Each up-sampling level increases the resolution by a factor of 2.

1.6 Image Fusion and Enhancement using Convolutional Neural Networks

In the field of digital image processing, taking a photo or a video using a single shot from a camera where all the objects in a scene are in focus is a challenging task, as only the objects within the depth-of-field will have sharp edges while other objects will suffer from blurring effects. To solve the problem and generate an image where all the objects are in focus a multiple images of the same scene with various focal settings are captured and merged into a single image. In the past decades,a variety of image fusion using multi-focus images have been proposed [59, 60, 61, 62]. Furthermore, image fusion has been employed in medical imaging as a tool to provide more accurate information for clinical purposes [63, 64, 65]. In general, multi-focus image fusion can be mainly divided into two groups: fusion directly in the spatial domain and transform domain. These methods consist of three stages: decomposition, fusion, and reconstruction.

In recent years, convolutional neural networks have been used as a tool for performing image fusion targeting image enhancement [66, 67, 68, 69]. Moreover, CNNs have been investigated as a novel tool for image and video enhancement.

1.6.1 Image and Video Enhancement

Image and video enhancement is an important research topic, which has drawn a lot of attention from both academia and industry. Image enhancement methods attempt to improve the perceived quality of an image and provide better quality for other image processing applications. Image enhancement is applied in many fields where images are analyzed or displayed. Specifically, the method proposed by Liew et al. [70] reduces both block discontinuities and ringing artifacts of compressed images using an overcomplete wavelet representation. Foi et al. [71] used adaptive-shape image filtering approach to reduce blocking effects caused by JPEG compression. Later, Wang et al. [72] proposed to remove the blocking artifacts in compressed images by filtering the boundaries between neighboring blocks. Chang et al. [73] presented a JPEG image decompression approach to diminishing artifacts using dictionary learning.

Recently, CNN has also been successfully applied to improve the visual quality of decoded images. Dong et al. [74] introduced a 4-layer network (ARCNN) to reduce the compression caused artifacts by learning an end-to-end mapping between the low-quality and the original images, which achieves a significant quality improvement for the low-quality images. ARCNN is designed based on a three-layer CNN (SRCNN [75]). Extended from [74], Yu et al. [76] proposed a faster 5-layer CNN, called FastARCNN.

On the other hand, some works focus on quality enhancement for compressed video. Wang et al. [77] proposed a deep network to enhance the HEVC quality at the decoder side, which can be directly applied to the existing video streams. In [78], a decoder-side scalable convolutional neural network was presented for HEVC quality enhancement, which does not require any modification at the encoder side.

These methods mainly focused on enhancing the quality of a single image or video frame. However, in a multiview video, exploiting the inter-view similarity has an important role in the quality enhancement task, making these aforementioned works unsuitable for multiview enhancement.

1.6.2 Multiview Depth-Based Quality Enhancement

In asymmetric multiview video systems, only several viewpoints are kept with high-quality while other views are highly compressed to low-quality. At the decoder side, an enhancement technique is employed to improve the low-quality views. Diogo et al. [79] presented a super-resolution method, in which low-resolution views are enhanced with the aid of high-frequency content from neighboring full resolution views. More specifically, they utilized the DIBR technique to generate a virtual view from the full resolution viewpoint. The high-frequency components of the virtual view are extracted, and added to the interpolated low-resolution views, with a consistency check step. Motivated by [79], Xie et al. [67] introduced a similar method for up-sampling low-resolution viewpoints in a 3D video using a convolutional neural network. In this work, the neighboring high-resolution image was projected to the position of the low-resolution image using DIBR, and reconstruct the low-resolution images into high-resolution ones using the projected image information. Jin et al. [68] put forward a quality enhancement method for mixresolution multiview videos. The core idea of this method is to selectively pick pixels from either the interpolated image or the full resolution virtual view. Although the above approaches to improve the low-quality views, they require accurate and high-resolution depth maps, which is still a challenging task. In summary, these methods are computationally demanding as they need to use DIBR.

In contrast, this thesis proposes a novel approach for multiview video enhancement. Where a low-quality lateral view and a high-quality central view, of the same scene, are inputted to a CNN network to generate a high-quality lateral view. Previous works [79, 67, 68], as depicted in Fig. 1.15, require at least two stages to enhance the low-quality view, including disparity estimation (if the depth map is not provided), wrapping the high-quality view to the viewpoint of the low-quality one using DIBR method, and the final enhancement process. Meanwhile, in this thesis, a novel method with a single end-to-end process employed using the MVENet, where neither depth information nor post-processing stage is required, the details are presented in Chapter. 6.

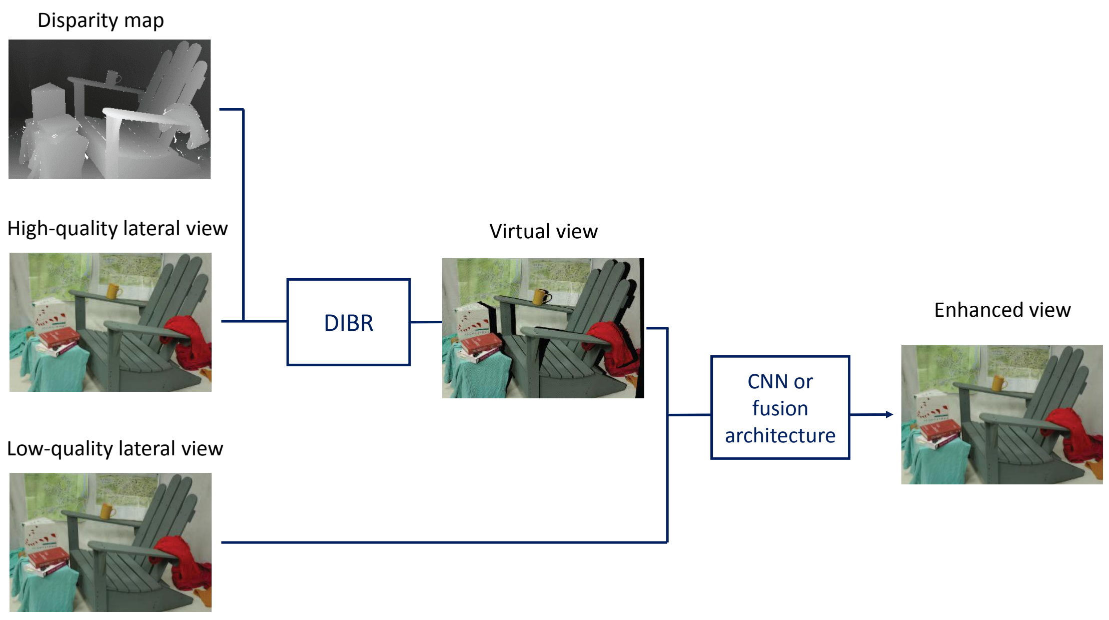

Fig. 1.15: The multiview depth based quality enhancement approach that have been employed.

1.7 Overview of This Thesis

1.7.1 Major Contributions

The major contributions of the research reported in this thesis are summarized as follows:

- In the field of disparity estimation, we proposed a novel multi-scale correlation layer integrated with a CNN using several correlation kernels and different scales to further improve the depth estimation accuracy for both fine and large objects in a real scene.

- A novel multi-resolution framework with a three-stage strategy for the generation of high-quality disparity maps, that handles both large and small disparities, is introduced. This method allow estimating accurate disparity maps for a wide range of disparities, as state-of-the-art methods have limitations in generating accurate disparities for large range of objects' displacements.

- A novel and interesting view synthesis approach for 3D videos is proposed in this thesis. This method represents lateral views solely using their edges, while dropping their texture content.

- In asymmetric multiview video systems, highly compressed views may incur in severe quality degradation. Thus, it is necessary to enhance the visual quality of highly compressed views on the decoder side. Therefore, a novel method for multiview quality enhancement is proposed in this. To the best of our knowledge, this is the first work for multiview video enhancement where neither a depth map nor a projected virtual view is required in the enhancement process.

1.7.2 Brief Summary of the Remaining Chapters

Chapter 2: Depth Estimation. In this chapter, a CNN model (DENet) that predicts a depth map directly without any post-processing is proposed. This network takes a pair of stereo images as input and outputs a single image representing the depth map for the middle view between the left and right views. For training the network, an end-to-end approach is employed. For the purpose of correct learning of depth estimation, computer graphics technology with accurate texture and depth sequences are used, which is suitable for generating training and testing sequences. Moreover, for the generation of real stereo images with a ground truth depth map, a platform was used that combined a ToF [41] (Swiss Ranger SR4000) depth camera with a pair of stereo cameras. In addition, a calibration method for the platform and a correction of the captured depth map is presented. Followed by an evaluation of the proposed approach on both computer graphics and real images. Finally, conclusions, limitations are introduced.

Chapter 3: Disparity Estimation Using Convolutional Neural Networks with Multi-Scale Correlation. In this chapter, a detailed literature overview of the relevant methods of disparity estimation from a pair of stereo images is first provided. Then a state-of-the-art work is discussed in details, advantages and limitations are studied as well. For further improving the performance of this network, a multi-scale correlation layer that integrates the disparity estimation network in [20] is proposed. The main idea of this work is to perform the matching process between the left and right convolutional features using multi-scale correlation unit that contains three kernels with different scales. Besides, several experiments were then conducted to evaluate the effectiveness of using multi-scale correlation unit. During the experiments, it was observed that the training loss exhibits a considerable amount of fluctuation as a result of the small batch size employed in the training process. This fluctuation is reduced by using a larger batch size. However, increasing the batch size increases the demand on memory and hence increasing the training time. To solve this problem, a new strategy for training the network using horizontally elongated images is implemented in this chapter.

Chapter 4: Multi-Resolution Disparity Fusion Using Convolutional Neural Networks. In this chapter, a new framework to mitigate limitations of current disparity estimation frameworks is investigated to further improve the accuracy. In particular existing methods fail to estimate reliable and accurate disparity map that covers a large range of disparities. The fundamental idea of this chapter is based on using multiple resolution versions of the initial stereo images, which provide scalable information to improve the feature map matching and expand the estimated disparity range. Hence, two end-to-end Multi-Resolution Disparity Fusion networks *MRDF-Conv* and *MRDF-Conv-Deconv* are proposed. *MRDF-Conv* is a light-weight and computationally efficient network with acceptable performance. *MRDF-Conv-Deconv* replaces all the standard interpolation units with learnable convolution and deconvolution layers and achieves better performance. To assess the proposed approach, extensive experiments in this chapter were conducted on KITTI-2015 and Sintel benchmarks. In the experimental part, the results demonstrate that there is significant performance gain by using a multi-scale scheme over one scale.

Chapter 5: Silhouette-Based View Synthesis for Multiview Video using Deep Convolutional Neural Networks. This chapter addresses the main problem in multiview videos applications, which is the vast amount of multiview video data required for representing various views. In a multiview video, a large number of different views, with a small baseline, are recorded by a set of cameras. First, an overview of relevant view synthesis methods is provided. Second, the novel Multiview Video plus Silhouette (MVS) approach is presented. In MVS, instead of using depth information, silhouette maps are used to generate accurate intermediate views from the existing views. At the encoder side, MVS extracts and sends only the edges of the lateral views instead of the depth map. While at the decoder side, the lateral views are rendered from the edge map of the original lateral view and the central view. Additionally, for the implementation of MVS approach, a deep CNN is used, which takes a high-quality central view and the edge map of a lateral view and outputs the synthesized view. The network in-paints and fills the edge map of the lateral view, taking advantage of the correlation between views. Finally, in this chapter experimental results demonstrate the feasibility of the proposed framework and show that the network is able to synthesize accurate and reliable lateral views starting from their edges. In the experimental part, several issues were investigated and discussed.

Chapter 6: Multiview Video Quality Enhancement without Depth Information. The key idea in this chapter is based on the fact that in asymmetric multiview video coding, the views are coded with different qualities to alleviate storage and transmission cost. On the other hand, noticeable visual artifacts are caused when a high compression ratio is adopted which limits multiview applications. To address this problem, a novel multiview video quality enhancement framework is proposed in this chapter. Specifically, the enhancement of low-quality views, using directly the high-quality views without any depth information, are investigated using deep CNN. The used network takes two inputs, low and high-quality views. The details in the low-quality view get synthesized directly from the high-quality view. The network takes advantage of the inter-view correlation and exploits the high-frequency information of the high-quality view to recover the details of the compressed view. In the proposed multiview 3D video coding system, the encoder requires no modification, the proposed network is only employed at the decoder side.

This chapter also provides experiments on both computer graphics and real-world data that demonstrate that the proposed approach can efficiently and accurately enhance the quality of the compressed views over other state-of-the-art approaches.

Chapter 7: Conclusion. This chapter, summarizes the main achievements and results obtained in this thesis and outlines future development directions in the field of multiview 3D video communication.

For each of these chapters mentioned above, we have tried to make them self-contained. Therefore, some of the crucial contents, demonstrations, model definitions, and depictive illustrations might be reiterated in the following chapters when necessary.

Chapter 2

Depth Map Estimation from Stereo Images using Convolutional Neural Networks

In the previous chapter an overview of conventional depth map estimation methods and techniques are summarized. In this chapter, a preliminary exploration of depth map estimation based on stereoscopies using deep convolutional neural networks is introduced.

2.1 Motivation

Human visual capability to perceive the world in three dimensions is based on binocular cues information. Due to the horizontal displacement of the two eyes, the images projected on the retina are slightly different. Closer objects show larger disparities than further objects. Conventional algorithms to estimate the depth map based on stereo images have been an active research topic in computer vision for decades due to its wide range of applications including: multiview video coding, view synthesis, autonomous driving, robotics, Free Viewpoint Television, 3D modeling and 3D Television, etc. These algorithms tend to produce sub-optimal results especially with less texture regions, specular highlights, or complex occlusion boundaries. Recently, convolutional neural networks have made significant progress and they have set new records in a wide range of computer vision tasks [80].

Convolutional neural networks have been applied to the task of depth map prediction from a single image. Eigen et al. [51] use a multi-scale CNN for estimating the depth from a single image, where a coarse scale network is used to perform depth prediction with low resolution, and a fine scale network is trained to perform local depth refinements with higher resolution. In [81] another framework with two levels, super-pixel level and pixel level, being applied to predict the depth of a scene from single monocular images. Liu et al. [82] used a CNN to learn the binary and unary potentials for their conditional random field framework for the task of estimating the depth from monocular images. Sun et al. [83] extract sparse feature vectors from monocular infrared images and train a fully-connected neural network with a single hidden layer to produce dense depth estimates. Laina et al. [19] introduced a deep fully convolutional neural network with residual learning to model the mapping between monocular images and depth maps. However, estimating depth from a single monocular image is still challenging due to the limited geometry information included within a single image.

Motivated by these works, in this chapter we investigated the feasibility of the depth estimation task from a pair of stereo images using convolutional neural networks where the depth map is located in the middle view between the other views.

2.2 The Proposed Depth Estimation Network Architecture

The main aim of the proposed scheme, as illustrated in Fig.2.1, is to exploit CNNs for rendering depth map directly without any post-processing. To achieve this target we designed the Depth Estimation Network (DENet). The proposed network takes a pair of stereo images as input, and outputs a single image representing the depth map for the middle view between the left and right views.

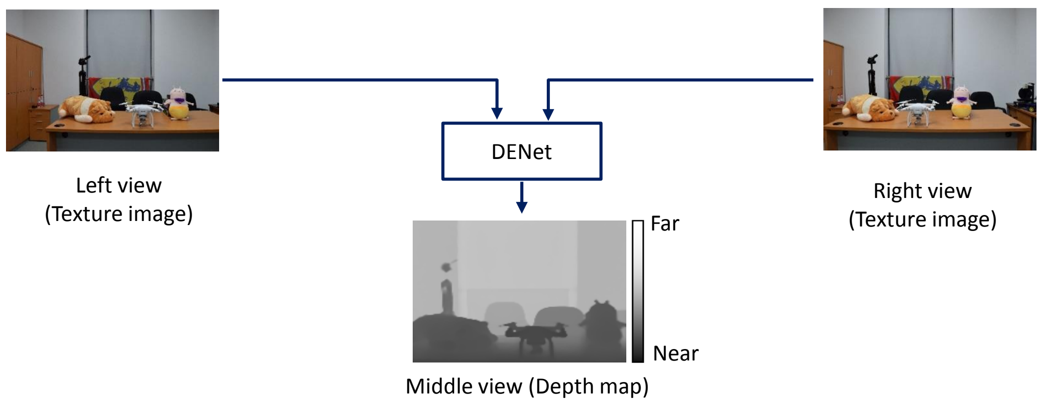

Fig. 2.1: The proposed scheme.

Learning to predict depth map directly from two stereo images is the main task for our network. And here comes an important question: what is the best architecture for this purpose? Convolutions account for short-range receptive field, limited by the size of kernels applied in each layer. Using pooling with small kernel's size would preserve long-range dependencies, however, this will lead to low resolution output. In order to avoid the loss of resolution brought by pooling we choose to have no pooling/subsampling in our network but convolutional layers with stride one [84]. Moreover, in this task CNN needs to exploit the displacement of objects between the left and the right views to render the depth. In this work the two cameras were aligned horizontally, thus the displacement between objects in the image plane is mainly horizontal. As a consequence, the kernel filters of the first layer were chosen to be elongated horizontally Fig. 2.2.

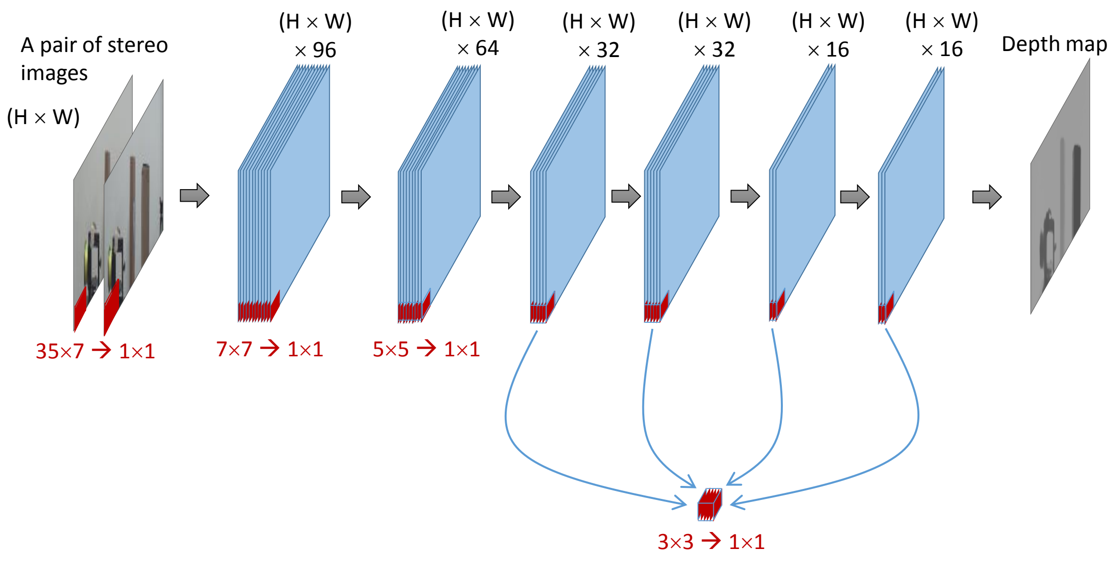

Fig. 2.2: The DENet network architecture.

More specially, Let $W_l \times H_l$ be the kernel size of layer L_l, and F_l is the number of features of layer L_l. We need to define the network parameters, that make the network able to infer the depth by exploiting the disparity between the two views. To detect the disparity we assume that we need to detect:

- Similar edges.
- Similar texture around the edges.

The first two layers have an important effect in addressing the previous two assumptions. To handle the displacement of D pixels in layer one, we need filters with width at least $D + (2 \times margin)$, where margin could be $\approx 3 \sim 5$. Moreover, we speculate that two filters are needed to detect disparity of D pixel, one is needed to detect edges and the second is needed to handle texture similarity.

Now we need to answer the following question: How many features are needed for layer one?

- Let suppose that the maximum displacement that needs to be handled in first layer is D_{max}. This means that we need to handle all the following displacements in first layer: $1, 2, 3, ..., D_{max}$.

- For each disparity we need $2 \sim 3$ features, this means that the total features we need in L_1 is $(2 \sim 3) \times D_{max}$ features. The other layers will complete the task of the first two layers to exploit the disparity of objects in the two views. In our experiments the maximum displacements was around 35 pixels, and thus based on the previous analysis the network configuration is shown in Fig. 2.2.

2.2.1 Training Method

Training the neural networks requires data with ground truth to optimize the filter parameters and to learn to perform the task. We therefore chose an end-to-end learning approach to learn the network to extract stereo features and predict the depth for the center view, between the left and right views. In this case the training data should be accurate to guide the network to learn the task in a proper way. based on this observation, the best option for training the network is to use Computer Graphics Images (CGI) for training the network, at the beginning we perform this training on several phases. The next phase will be to fine tune the network using real dataset. The generation of the computer graphic dataset is presented in detail in the following section.

2.3 Computer Graphic Dataset Generation

Computer Graphics (CG) technology produces accurate texture and depth sequences, which is suitable for generating training and testing sequences. Thus, Blender 2.76 [85] is used to produce the CG sequences including aligned texture and depth. The software is able to generate random virtual $3D$ views. The generated dataset includes $20K$ stereo pairs with different scenarios to have variety in the training data:

- CG-Set1: simple scenarios with simple background and basic foreground objects $\{$cube, cylinder, sphere, torus, etc.$\}$ with no texture.

- CG-Set2: additional training CG images with more complicated images: Set2 = $\{$ Set1 $+$ various kind of texture, as show in the second row in Fig. 2.3 $\}$.

- CG-Set3: include more objects in the scene: Set3 = $\{$ Set2 $+$ more objects located in each pair$\}$.

- CG-Set4: include many objects that overlapped and merged: Set4 = { Set3 + overlapped and merged objects}.

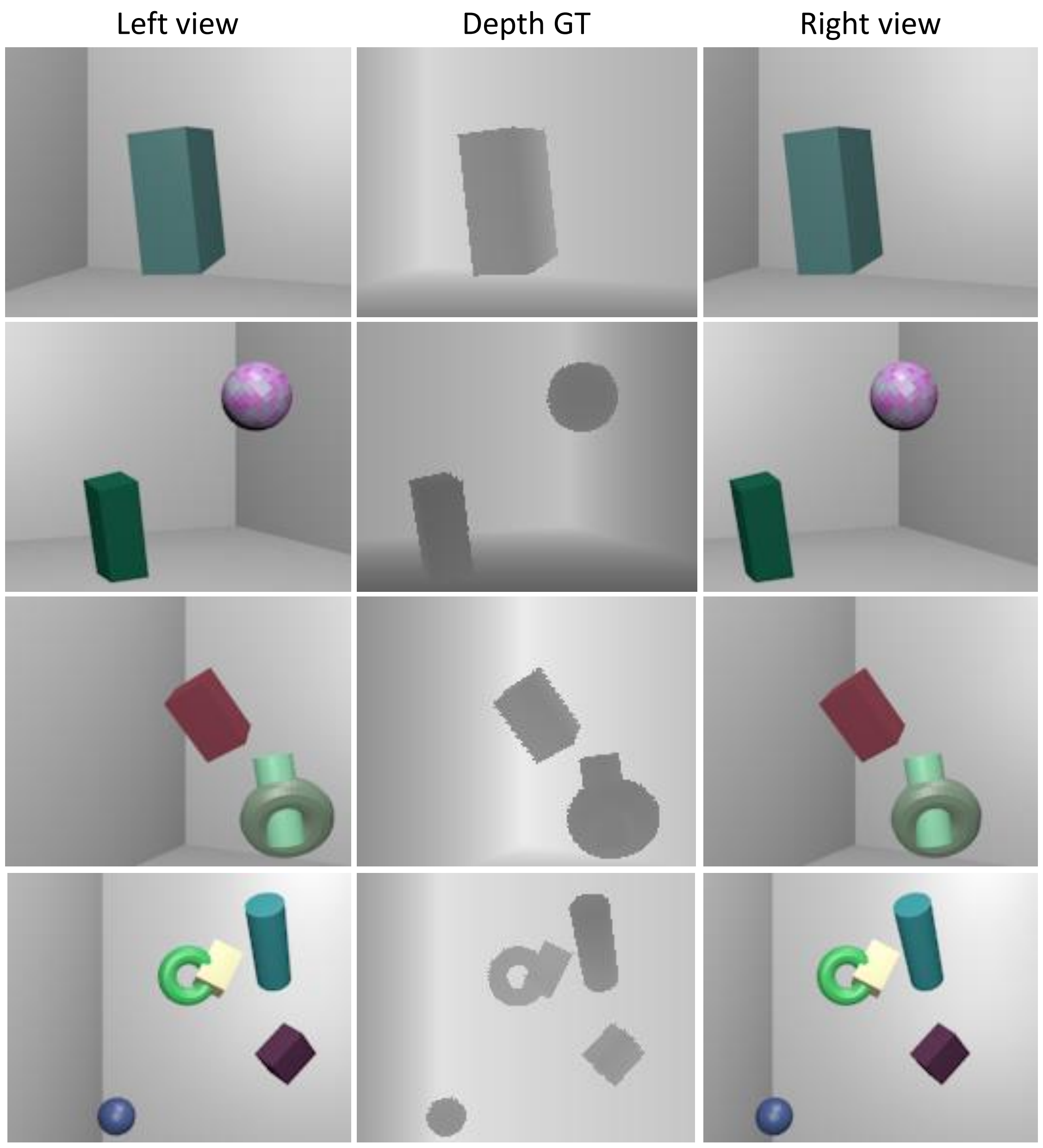

Fig. 2.3: Examples of computer graphic generated images with various foreground objects.

The objects are allowed to have arbitrary rotations. The objects' sizes and colors has no restrictions. In addition, the position of the objects is randomized as well. Finally, the viewpoint is randomized as well. Fig.2.3 shows various couples of sample scenes along with the corresponding depth maps for our generated datasets.

In the previous datasets the left and right images, I_L and I_R, are rendered as 8-bit 176×140 RGB Tiff images, while the depth map is rendered as a single channel 8-bit 176×140 Tiff image. The computer graphics generated datasets are used to train the network. Whereas the real dataset is used for evaluating the trained network. In the next section the real dataset generation is presented in detail.

2.4 Real Stereo Dataset Generation

To generate the real stereo dataset, which includes 175 stereo images, with a ground truth depth map in the middle view, a platform with cameras was constructed that combined a ToF [41] (Swiss Ranger SR4000) depth camera with a pair of stereo cameras as shown in Fig.2.4. The SR4000 provides depth maps of the scene with an operational range of up to five meters, with a resolution of 176×144 pixels. The ToF camera suffers from geometric distortion. In order to use the generated images, correction and enhancement of captured images using the Matlab camera calibration toolbox [86] is employed. The two stereo RGB cameras are Nikon D3300 with a maximum resolution of up to 6000×4000 pixels. In the tested setup, the two stereo cameras have a baseline separation of $22.6cm$ and the

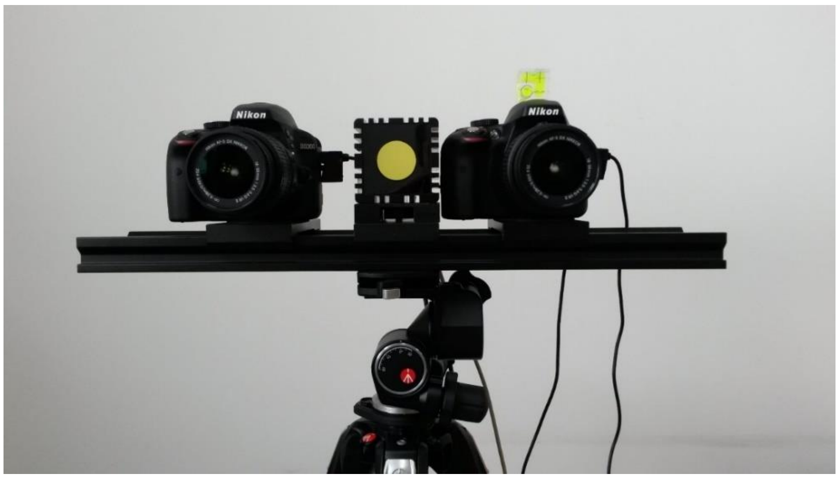
Fig. 2.4: Experimental setup Swiss-ranger SR4000 and stereo cameras.

depth camera placed in the center between the stereo cameras, which allows the texture cameras to include the view of the ToF camera.

2.4.1 Calibration of SR-4000 TOF Camera

In general the lens used in cameras have non-linear image distortion: radial distortion and tangential distortion. Radial distortion is caused by the shape of lens, if the light passes far from the lens center, the captured image will suffer from radial distortion. The raw depth and intensity images produced by the depth camera SR4000 require distortion correction. It is obvious from Fig. 2.5(a) that objects far from center of the image suffers from radial distortion, for example the edge of the wall on the left and clipboard of the intensity map appear bent. Moreover, these distortions appear in the depth map as well. Since

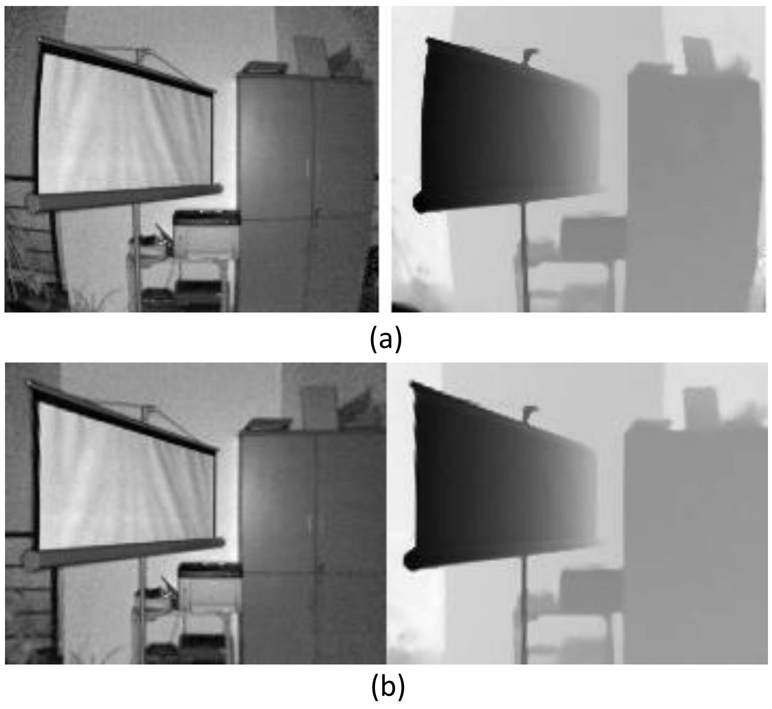

Fig. 2.5: Texture and depth images captured by SR000 (a) with distortion. (b) after correction.

the intensity map and the depth map in SR4000 have the same viewpoint, the calibration parameters computed for the intensity image can be used to remove the distortion of the depth image. In this work, the Matlab camera calibration toolbox [86] is used to measure the intrinsic parameters and correct the lens distortion. This method takes as input different checkerboard images and the checkerboard needs to be located in different positions.

Fig.2.5(b) illustrates the results from the depth camera after correction, and it is obvious that the wall edge, the clipboard and other objects in the scene have been adjusted to be aligned vertically.

After correcting the lens distortion of the depth images, a simple and accurate calibration method between the stereo cameras and the depth camera is applied. Fig. 2.6. shows a stereo pair images with depth map of the calibrated platform.

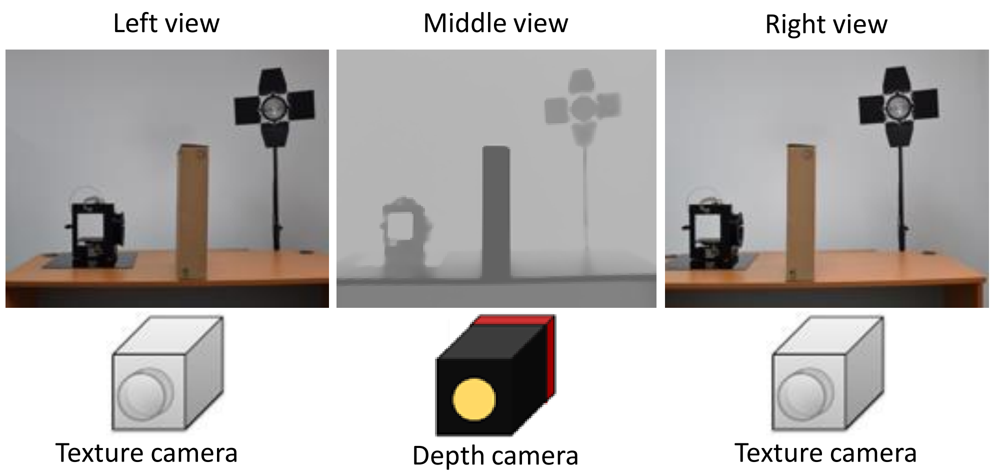

Fig. 2.6: Examples of calibrated stereo images (left and right) and depth (middle).

2.4.2 Disparity-Depth Mapping

Before the training stage, it is important to match the disparities and depth of generated images for the CG datset and real stereo images. In other words, the real stereo platform and the designed one in Blender should have the same parameters. In particular, the disparity obtained from the calibrated stereo rig within a range $1 \sim 5$ meters should match the one generated by Blender. To achieve this task, we perform several experiments to measure the disparity map of the stereo system. The data was collected by capturing images of a planer-pattern at several predefined positions within the range $1 \sim 5$ meters, located far from the camera. We set the cameras parameters in Blender for CG images to match the parameters of the platform.

Fig. 2.7 shows the results after setting the correct parameters in Blender. The x-axis indicates the distance from the camera position and the y-axis represents the disparity values in pixel. The disparity values are extracted from the left and right views. For example, this figure shows that if an object is located 2.6 meters far from the platform, then the displacement between the left and right images for this object is 20 pixels for both real and CG images. We can observe that the disparity map of CG images almost matches the results obtained from the real platform. These results allow us to train the network using CG images and test it on real images.

2.5 Experimental Results

The proposed seven layers fully convolutional neural network takes a pair of stereo images as input, and outputs a single image representing the depth map. The left and right views 176×140 images are concatenated to produce the $2 \times 176 \times 140$ input. The Caffe [87]

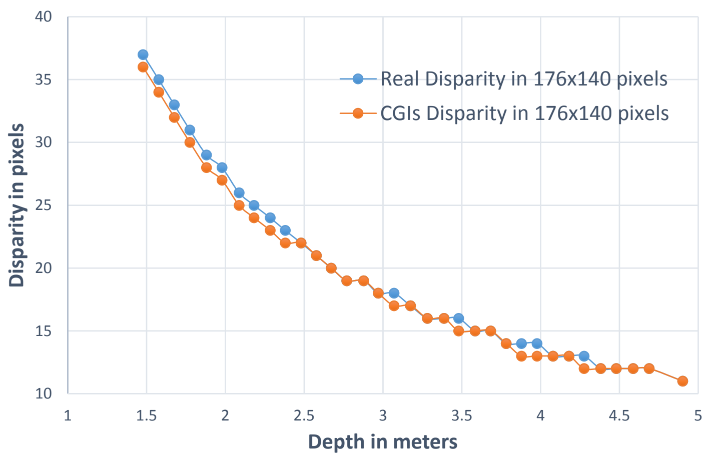

Fig. 2.7: Disparity-depth mapping for both real and computer graphic images.

framework is used for training CNNs. At training time, the input to the network was a batch of 128 pairs of images, and the learning rate is fixed to $\lambda = 1e^{-6}$.

2.5.1 Computer Graphic Testing Set

At this stage, the network is trained using only computer graphics images. Let $\mathbf{I}$ be the input left and right stereo images with size $H \times W$. We trained our network end-to-end by minimizing the norm $L2$ for the N training samples as follows:

$$L(\theta) = \frac{1}{N} \sum_{j=1}^{N} ||\hat{D}_j(\theta) - D_j^{gt}||^2, \tag{2.1}$$

where $\hat{D}_j(\theta)$ is the depth map obtained from the inputs $\mathbf{I}_j$ and the set of network parameters θ. The label D_j^{gt} is the ground truth depth map.

Initially, the network is trained using $CG - Set1$, $4K$ CGIs with a single foreground object *(1FGO)* in the scene. The kernels' values obtained from the first step are used as an initial values for the next training phase. In this step the network is trained using $CG - Set2$, $8K$ CGIs with two foreground objects *(2FGOs)*. Then we increased the training examples to $12K$ stereo images with three foreground objects *(3FGOs)*, $CG - Set3$ is used. Finally the network was trained using $CG - Set4$ dataset, which includes $16K$ stereo images with $4 - 5$ foreground objects *(5FGOs)*. The evaluation of the trained network was performed first on CG testing set, this testing set includes 160 images were randomly generated.

Table. 2.1 presents the results obtained after training the network with different training examples sizes. As expected, the results show that the PSNR increases when the training examples are increased and more objects are included.

Table 2.1: Performance comparison (PSNR) of different computer graphic sets.

Train \ Test	CGIs (1FGO)	CGIs (2FGOs)	CGIs (3FGOs)	CGIs (5FGOs)
CG-Set1	26.78	23.88	23.46	23.16
CG-Set2	27.47	26.80	26.21	25.25
CG-Set3	27.68	27.73	27.14	25.95
CG-Set4	**28.39**	**29.02**	**28.35**	**27.28**

2.5.2 Real Testing Set

To evaluate the performance of the proposed method and to check whether the network is truly learning stereo features for real stereo images, testing set is captured by the platform and used in the evaluation phase. Table.2.2 shows the results obtained after testing the network using real stereo images. In this table, using $CG - Set3$ provides the highest $PSNR$ among other scenarios, which is mainly a result of including $3 - 5$ foreground objects in the real images contains, and training the network using $CG - Set4$ leads to over fitting. In general, The depth estimation of the foreground objects is close to the ground truth. However, the predicted depth for the background and around the objects is still not accurate. This is due to the fact that the network was unable to estimate global depth information. Fig. 2.8 shows an example for the estimated depth map using our DENet.

Table 2.2: Depth estimation (PSNR) using real images.

Training set	Real testing set
CG-Set1	17.33
CG-Set2	19.19
CG-Set3	**20.38**
CG-Set4	19.38

2.6 Application and Limitation

This section discuss and analyzes the suitability and practicability of the proposed approach. If the application is interested in estimating the depth for foreground objects and not the background then this network could be used. In such systems a rig with two stereo camera is required to capture a pair of stereo images then feed them to the network.

The main advantage of this network, that it gives the depth for the middle view between the left and right view. Although in this work CG sequences are used for training the network, finetuning the network using a real stereo dataset might lead to better performance for testing real images. The main limitations of this approach are the low resolution depth map obtained by the stereo platform, and failing in estimating the depth for

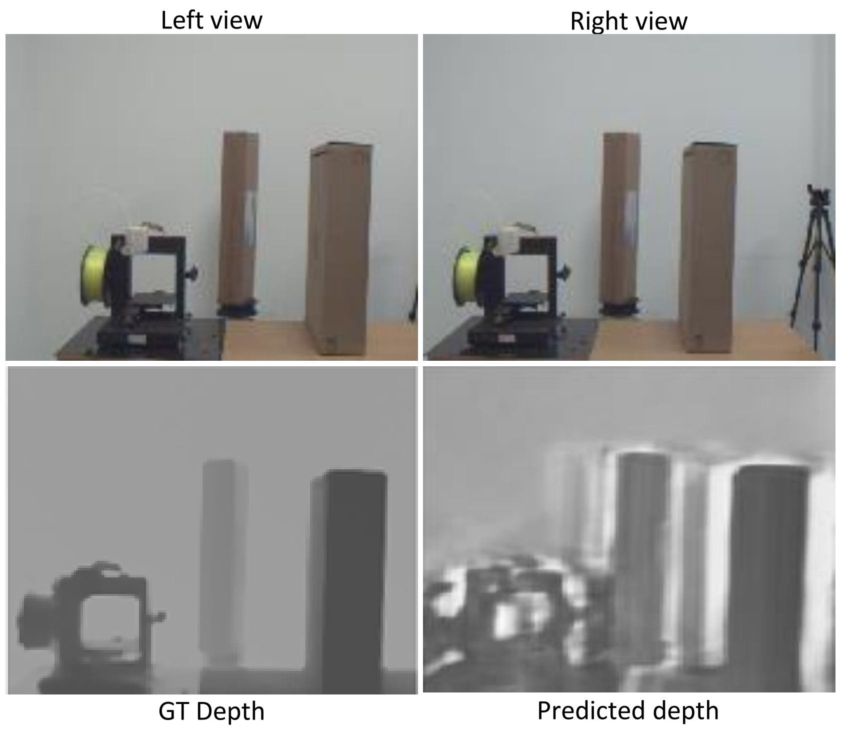

Fig. 2.8: Example of depth estimation for real images using the proposed DENet.

background objects. Nevertheless, the network is able to estimate depth for foreground objects.

2.7 Conclusion

In this part of the thesis a convolutional neural network was designed to render 3D information from a pair of stereo images and predict a depth map for the middle view. Results show the feasibility of depth information extraction using DENet without any post-processing. The proposed DENet was trained using only computer graphic images, to accurately learn to predict the depth from a pair of stereo images. Results prove the ability of predicting the depth with CNNs from stereo images.

The current predicted depth map method has two problems which need to be investigated and solved. First, the current network failed in measuring the correct depth for the areas around objects. Secondly, it seems the current network is not able to estimate the depth for the background with no texture information accurately, as the network was trained to estimate only local disparities for the objects in the scene but not for the background.

We believe that using high performance GPU, larger and complex real scenarios datasets, more accurate method for calibrating the platform, and different CNN configuration potentially will result in better performance.

Chapter 3

Disparity Estimation from Stereo Images Using Convolutional Neural Networks with Multi-Scale Correlation

This chapter starts by introducing background information and recent works in disparity estimation from stereo images, focusing on the main challenges and limitations of these works. Then a brief introduction to one of the early networks that have been used for disparity estimation is presented, which is the motivation behind this work. Although this network has achieved state-of-the-art performance, when it was released, in the disparity estimation task, it suffers from several limitations. Afterward, the proposed multi-scale correlation method is presented, specifying the adopted solution for further accuracy enhancement. Finally, experimental results and conclusion are provided.

3.1 Introduction

Disparity estimation and 3D reconstruction from a rectified stereo pair of images are fundamental tasks in both multimedia and computer vision communities. Several approaches have been introduced to improve the accuracy as well as reducing the computational cost for disparity estimation [88].

A typical disparity estimation algorithm [89] involves four stages: matching cost computation, cost aggregation, optimization, and disparity refinement [88]. The main task of a stereo algorithm, known as the correspondence problem, is to match the pixels in the first image to the pixels of the second one. The typical methods generally fall into two broad categories. On one hand, local methods [90, 91, 92, 93] tend to be applied in real-time applications where the computational efficiency is valued over accuracy. On the other hand, global approaches [94, 95, 96] provide accurate results but with longer run time. Between the two categories, some methods provide a trade-off between accuracy and speed such as the SGM algorithm [89]. In fact, for practical applications, both disparity estimation accuracy and computational efficiency are important factors.

Convolutional neural networks have been investigated as a possible solution to the stereo correspondence problem. The input for such networks is two rectified stereo images. Zagoruyko and Komokadis [52] exploited CNN to learn a general similarity function by using a two-channel siamese model. Zbontar et al. [53, 54] used a convolutional neural network to determine corresponding patches in left and right stereo images. These methods include semi-global block matching for each pixel with a series of post-processing steps, which renders the model computationally intensive. To reduce the computational complexity, [97] exploited a matching network with a product layer that computes the inner product between the two representations of a siamese architecture. However, the proposed stereo matching between patches also leads to incorrect correspondence due to various reasons such as occlusion, saturation, and pixel intensity noise. As a solution to this problem, Seki et al. [98] exploited a two-channel CNN to predict the correspondence confidence for disparity patches. After that, a fusion method that uses confidence is employed.

In contrast, Mayer et al. [20] designed an end-to-end fully convolutional neural network *DispNetC*, which inputs full-resolution left and right stereo images to predict a disparity map without pre-processing nor post-processing. *DispNetC* is fast with a runtime of 0.06 seconds, which is particularly important for real-time applications. *DispNetC* includes a contracting part that progressively decreases the spatial size of the convolutional features, providing large receptive fields for higher-level convolutional layers, which in turn enables the network to capture more global information.

In [20], a correlation layer with a single kernel that performs the matching process between the left and right CNN feature maps is employed. However, a single 1×1 kernel unit, as used in [20], is not suitable for large objects or repeated patterns, because a small number of pixels in the neighborhood involved is used in the matching process. In this chapter, we introduce a method that integrates the state-of-the-art disparity estimation network [20] with a multi-scale correlation layer to improve the accuracy and solve the matching problem for large regions with low texture. We also propose to train the network using horizontally elongated images instead of the whole image, which eventually reduces training time and increases prediction accuracy.

3.2 Proposed Multi-Scale Correlation Approach

In this section a CNN with Multi-scale correlation for disparity estimation without pre nor post-processing steps is proposed.

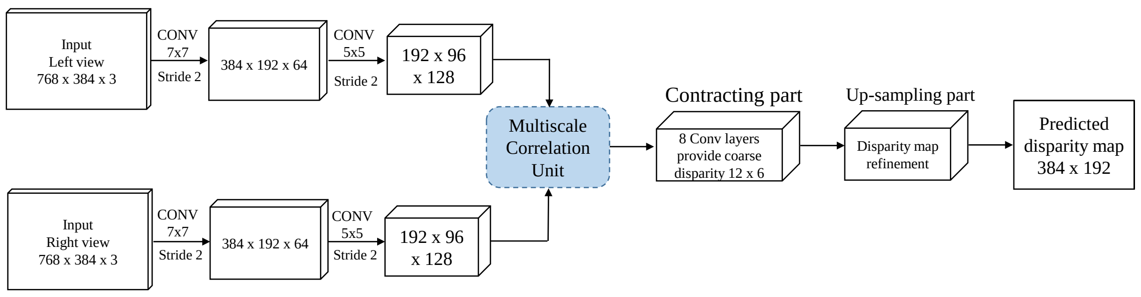

Fig. 3.1: The CNN architecture with the proposed multi-scale correlation layer. This layer perform the matching process between convolutional features using different correlation scales. The dashed block is presented in detail in Fig. 3.3.

3.2.1 CNN Architecture

In our work, we introduce several modifications to both the network architecture and the training process compared to [20]. The network architecture is shown in Fig. 3.1. First, we propose to perform the matching process between the left and right convolutional features using multi-scale correlation layer. Second, a new strategy for training the network using horizontally elongated images is implemented.

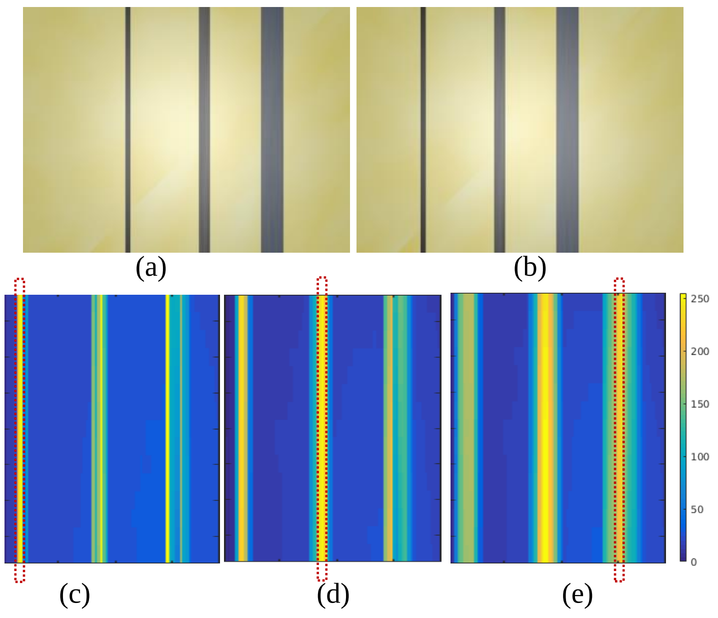

Fig. 3.2: The matching process using three single-scale correlation kernels with different sizes. In (a) and (b) the left and right views are illustrated. The figures (c), (d) and (e) show the matching results in the feature domain using the scales 1×1, 1×3 and 1×7.

3.2.2 Multi-Scale Correlation

In this section, the concept of the multi-scale correlation layer is introduced. We start with the case of the single-scale correlation unit for convolutional features. Although the single-scale correlation unit with a 1×1 kernel size is suitable for finding the correspondence between small objects and details, it has limitations for big objects with uniform areas, and objects with low texture and/or repeated patterns. Indeed this problem can be minimized by employing a correlation unit with multiple correlation kernels and different scales, which performs the matching process between a set of vectors in the neighborhood and not just a single vector in the case of the small kernel.

To illustrate this point, a pair of stereo synthetic images generated using the software package Blender are shown in Fig. 3.2 (a). The scene consists of three vertical dark objects with different sizes (small, medium and large). The two images are input into two identical and separate CNN streams to generate features. Then, three correlation units with different scales $1 \times 1, 1 \times 3$ and 1×7 are applied to the obtained features. By comparing the three figures Fig. 3.2 (c), (d) and (e) it is possible to observe that the maximum output for the small object is obtained in Fig. 3.2 (c) which correspond to the output of the smallest correlation unit. Whereas, for the middle-sized object, the largest correlation value is obtained using the scale of 1×3. Similarly, the large object is best matched using a bigger scale, i.e., 1×7, as shown in Fig. 3.2 (e). From this example, we find that using a multi-scale is suitable for objects and regions with various sizes, whereas a single-scale kernel will produce sub-optimal results especially for a complex scene with many objects of different sizes. In summary, small scales can aggregate the information within a set of small local regions, and bigger scales can match between large and global areas.

In light of our previous observations, a multi-scale correlation layer for convolutional features is proposed as depicted in Fig. 3.3, where the feature maps of the left and right streams are passed to three single-correlation units with the following scales $1 \times 1, 1 \times 3$ and 1×7. The output of these units are concatenated together and passed to the next convolutional layer.

Let us suppose that F_L and F_R are the left and right feature maps of the second convolutional layer, respectively, $H \times W$ is the size of the feature and K is the total number of features for each branch. In addition, let $N \times M$ be the size of the correlating block.

For parallel camera alignment, the most common setting for stereo cameras, the difference between the left and right views is mainly due to the objects' horizontal translation. As a result, the correlation process only covers the horizontal displacement. The correlation process between the left and right features will produce maps with the same size. The correlation map for d pixel displacement is:

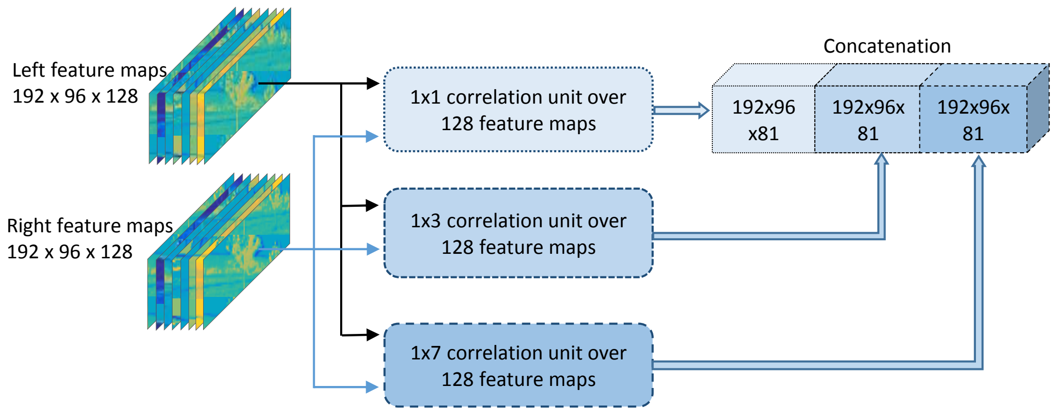
Left feature maps
192 x 96 x 128
Right feature maps
192 x 96 x 128
1x1 correlation unit over 128 feature maps
1x3 correlation unit over 128 feature maps
1x7 correlation unit over 128 feature maps
Concatenation
192x96 x81
192x96x 81
192x96x 81

Fig. 3.3: The multi-scale correlation layer contains three kernels with different scales $1 \times 1, 1 \times 3, 1 \times 7$. The obtained features are concatenated together then passed again to the CNN.

$$F_m(x,y) = \sum_{n=1}^{K} \sum_{j=0}^{N} \sum_{i=0}^{M} Fl_n(x+i, y+j) \times Fr_n(x+i+d, y+j) \tag{3.1}$$

where Fl_n and Fr_n represent channel n of the left and right feature, respectively, and $N \times M$ is the correlation kernel size. In our case, $N = 1$. The matching process in a single-scale correlation unit is performed between two feature blocks of size $1 \times M \times 128$. The first block copied from the left feature maps for a given center position of correlation, while the second block copied from the right features with a displacement d, and an element-wise multiplication between the two blocks is followed. The obtained vectors are then summed and set to the same position in the matched features that represent the matching for a displacement of d. In Fig. 3.4 the correlation process for both 1×1 and 1×3 is depicted.

3.3 Experimental Results

The model and architecture described in the previous section was trained on a large synthetic training set and evaluated on four data-sets in comparison to the state-of-the-art work [20].

The Mean Absolute Error (MAE) was used as an error measure to assess the quality of the estimated disparities:

$$MAE = \frac{1}{N} \sum_{p \in P} |\hat{d}(p) - d^{gt}(p)|, \tag{3.2}$$

where P is the pixel set, N is the size of the pixel set, $\hat{d}(p)$ and $d^{gt}(p)$ are estimated and ground-truth disparity values for pixel p, respectively.

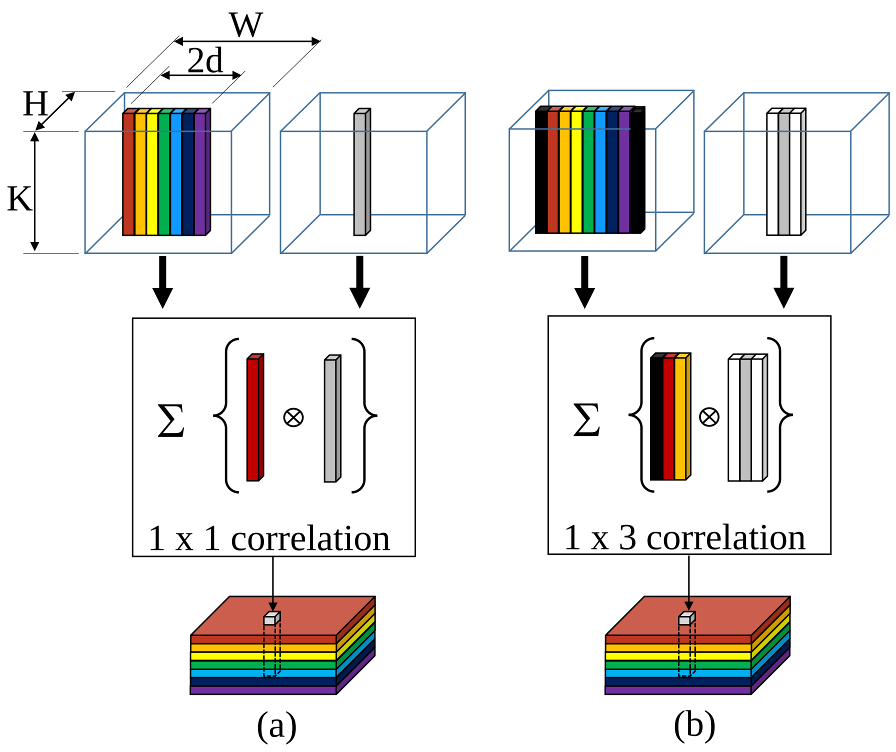

Fig. 3.4: The matching process between the left and right convolutional features. The correlation scale (a) 1×1 and (b) 1×3 are shown. Where black vectors indicate zero padding and $\otimes$ is the element-wise multiplication.

3.3.1 Synthetic Training Set

The network is trained using the FlyingThings3D synthetic stereo set. This dataset contains 22740 stereo frames for training and 4760 frames for testing. Each frame contains between $5-20$ everyday objects flying in the scene. The parameters of the objects (size, type, positions, texture, rotation) are randomly sampled, which makes this dataset suitable for training a deep CNN. An end-to-end learning approach is chosen to train the network to extract stereo features and predict the disparity map from the left and right views.

3.3.2 Training the CNN with Multi-Scale Correlation

The network is trained on the FlyingThings3D training dataset. Initially, the training was performed using patches of size $768 \times 384 \times 3$ which were cropped from the original image of size $960 \times 540 \times 3$ from the FlyingThings3D training set, and a batch size of 4 per iteration.

During the experiments, we found that the training loss exhibits a considerable amount of noise as a result of the small batch size employed in the training process. This noise can be reduced by using a larger batch size. Increasing the batch size increases the demand on memory and hence increasing the training time. Training the network with a batch size of 8 images takes 20 days to reach the $660k$ iterations using one Titan X GPU.

In contrast, we propose the slicing of each image into four horizontally elongated patches, each with a size of $768 \times 128 \times 3$ to speed up the training process, as the horizontal sliced patches have smaller size than the original images which results in reducing the processing time. The result is $91,040$ pairs of stereo images for training the network and leads to use of a batch size of 12 stereo images. The memory requirement for training the network with both bach size of 12 with sliced images is the same as that for a batch size of 4 for the complete image. On the other hand, even though the memory sizes used for training per iteration in both cases are the same, the use of sliced images provide more diversity and information than using one complete image, as the complete image may contain redundancy and repeated regions. Fig.3.5 shows the training loss using complete and sliced images, we observed that the training loss for sliced images is smoother and smaller.

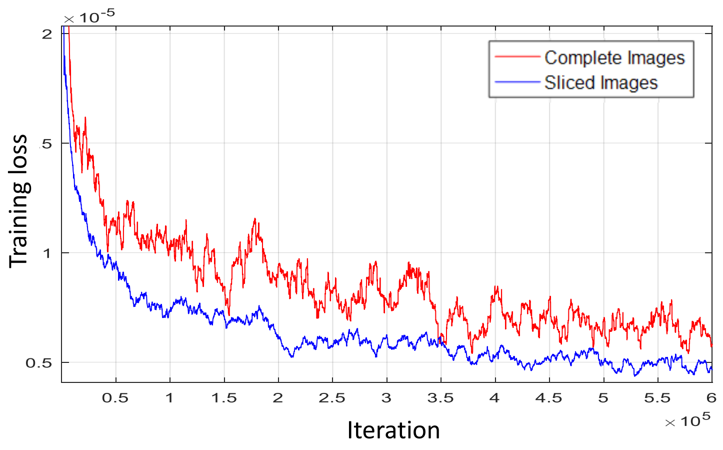

Fig. 3.5: Training loss analysis of multi-scale network with complete and sliced images on FlyingThings3D.

3.3.3 Results

The aim in this section is to test the performance of the disparity estimation using a single-scale correlation layer and the impact of using different scales (the different scales are 1×1, 1×3 and 1×7, where 1×1 is used in [20]) in comparison with the multi-scale correlation unit. For each model, one training experiment is conducted, with each taking 10 days to train using one Titan X GPU. All of the models are trained on the same FlyingThings3D training set in order to provide a fair comparison, and the evaluation is performed on different datasets including a FlyingThings3D clean test, Driving clean, Monkaa clean [20], Sintel clean train, KITTI train, and Middlebury. All the models are trained for 460k iterations. The feed forward run time in [20] is $0.06s$, whereas using the proposed model it is slightly increased and becomes $0.08s$, which a result of adding the multi-scale correlation unit into the correlation layer, while the other layers remain with no modifications.

Table 3.1: Disparity map errors (MAE). The disparity errors is measured by Mean Absolute Error on various disparity ranges

Method	KITTI 2012 Train	Driving clean	FlyingThings3D clean Test	Sintel clean	Middlebury 2014
Multi-scale (Proposed)	**36.06**	**14.73**	**4.54**	**8.70**	**41.29**
Single-scale 1 x 1 [6]	40.64	15.42	4.59	9.82	44.71
Single-scale 1 x 3	36.76	14.75	4.66	10.32	46.33
Single-scale 1 x 7	40.63	18.84	4.98	10.85	42.09

Table. 3.1 reports the MAE for each network on all testing sets, which is one of the s-tandard error measures used for Middlebury stereo evaluation, except for the KITTI where we follow the error measurement method provided by the KITTI. For the purpose of ensuring a fair comparison between the different models, the evaluation is performed on models of the last five iterations on each training set, then the average is calculated. From Table. 3.1 we noticed that the convolutional network with multi-scale correlation outperforms the other methods with single-scale kernels on KITTI 2012 Train, FlyingThings3D Clean, Monkaa Calean, Sintel Clean and Middlebury, which indicates the importance of using multi-scale correlation.

Meanwhile, we also notice that all the models have difficulties with large displacements as we can see in Table. 3.1 on Middlebury 2014, for which the disparity reaches 600 pixels. This is because the network is trained on the FlyingThings3D training set which contains small disparities (of less than 160 pixels). The network learns to estimate disparities of around 160 pixels or less, so it is unable to effectively estimate large disparities. Nevertheless, for Middlebury 2014, the gain of the proposed method is large in comparison to other methods. The proposed Multi-scale correlation CNN shows better performance for flat regions with uniform areas, as shown in Fig.3.6.

Fig. 3.7 illustrates that the proposed method learned to efficiently combine the information from different scales. In the first row (the image with chair) it could be observed that the proposed network uses the large scale to accurately estimate the disparity for the chair and from the small scales keep the edges. On the other hand, the network in [20] fails in estimating the disparity of the chairs back.

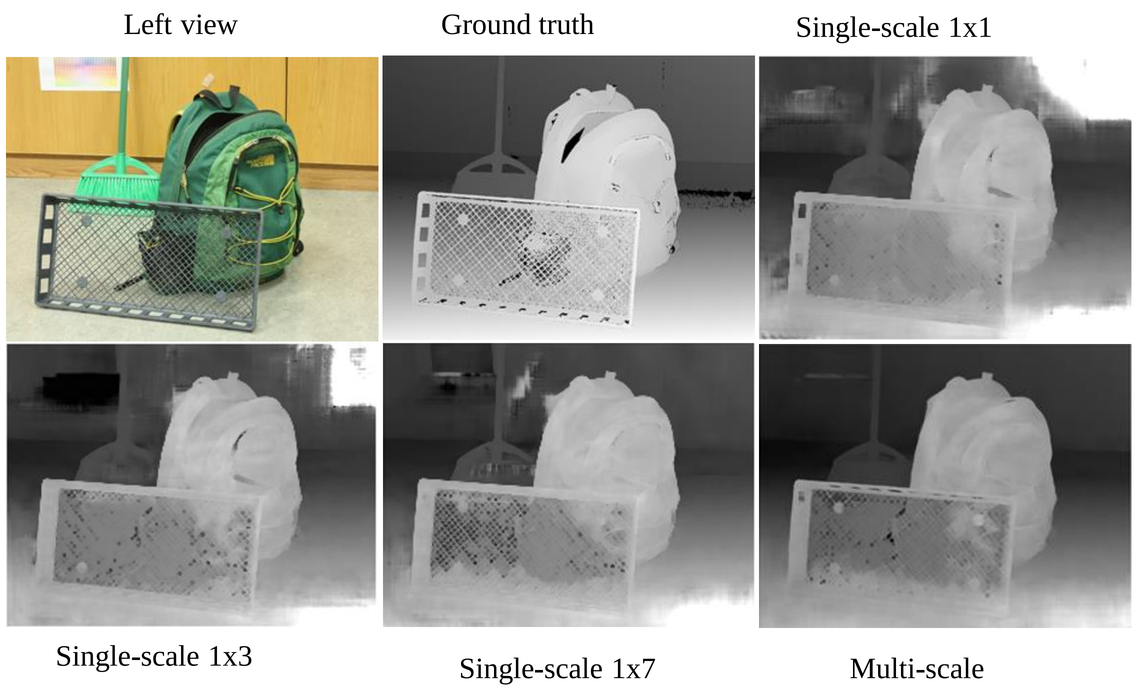

Fig. 3.6: Examples of disparity estimation for performance comparison between multi-scale correlation and single scales on Middlebury 2014 dataset.

In fact, each model with a different single correlation filter shows specific characteristics. The model using a 1×1 kernel size provides generally good performance with small objects and very rich texture while using the 1×7 scale achieves more accurate results for large objects and uniform areas. On the other hand, 1×3 shows good performance for objects with less complex texture and average size. The main advantage of this method is the ability to produce smooth predictions of the depth map as well as on texture-less regions.

Table 3.2: Disparity errors comparison (MAE)for two types of training; namely: horizontally elongated patches and complete image.

Training Multi-scale	KITTI 2012 Train	Driving clean	FlyingThings3D clean Test	Sintel clean	Middlebury 2014
Sliced images	**36.06**	**15.18**	4.54	8.70	**41.29**
Complete images	36.30	17.99	**4.43**	**7.01**	45.98

In Table. 3.2, we compare the performance of CNN with multi-scale correlation using the new training method with horizontally spliced images with that use the whole image for training as described in Section 3.3.

Finally it is worth reporting that a multi-scale correlation layer with kernels sizes: 2, 4,

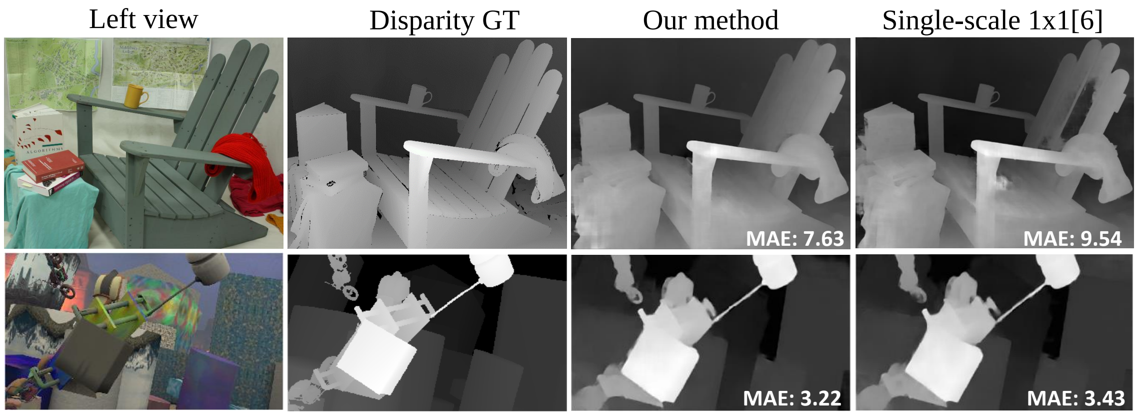

Fig. 3.7: Examples of disparity estimation using multi-scale correlation. The models trained on FlyingThings3D for 460k iterations. Rows from top to down: Middlebury 2014, FlyingThings3D (clean).

8, 16, 32 could have been chosen to form a base for different objects sizes, but practically odd-size kernels are more convenient, since destination feature maps are mapped directly onto the source neighborhood centers. Thus, two multi-scale networks were tested, namely: $\{1, 3, 7\}$ and $\{1, 3, 7, 15, 31\}$. It has been found that for the resolution we tested bigger, i.e. 15 and 31, kernels have a very limited contribution, nevertheless, we believe that for images with higher resolution large kernels will have some major contribution.

3.4 Conclusion

In this chapter, a novel method for disparity estimation from stereo images using CNN with a multi-scale correlation layer was proposed. It employs several correlation kernels and different scales. It was found that small kernels are suitable for the disparity estimation for small objects with fine details while larger scales are suitable for larger objects with uniform areas. The proposed model is able to capture the key stereo features of the two views and generate an accurate disparity map. Furthermore, we investigated the training approach using horizontally elongated patches to speed up the training process and improve accuracy. Experimental results showed the effectiveness of the proposed approach in comparison to the single kernel correlation, which achieves accurate results on both synthetic and real stereo images.

It is worth reporting that the proposed scheme in this section has led to the following publication:

1. Samer JAMMAL, Tammam TILLO, and Jimin XIAO, "Disparity Estimation Using Convolutional Neural Networks with Multi-scale Correlation," In International Conference on Neural Information Processing, October 2017.

Chapter 4

Multi-Resolution Disparity Fusion Using Convolutional Neural Networks

In this chapter, we first describe a major limitation in the state-of-the-art network *DispNetC* [20] that comes from the correlation unit, in particular, *DispNetC* fails to estimate accurate disparities for both foreground and background objects. Then, in Section 4.3 a framework with a three-stage strategy for the generation of high-quality disparity maps, that handles both large and small disparities, is introduced. The first stage down/upsamples images to different resolutions to improve the matching process between feature maps. The second stage uses a deep convolutional neural network to estimate the disparity maps using the re-sampled versions. Each estimated disparity maps fits a specific range of disparities, a small resolution is suitable for large disparities, whereas high resolutions are suitable for estimating small disparities for far regions and objects. In the third stage, to merge the disparity maps to a single representation, two end-to-end Multi-Resolution Disparity Fusion networks *MRDF-Conv* and *MRDF-Conv-Deconv* are proposed. In principle, the proposed techniques could be used for other networks no just for *DispNetC*. Finally, experimental results and conclusion will be presented in sections 4.4 and 4.5, respectively.

4.1 Introduction

In *DispNetC* [20] the maximum displacement between two features in the correlation layer makes the network unable to estimate large disparities of close objects. This renders the network unsuitable to estimate large disparities of foreground objects with large displacements between two features. First row of Fig.4.1 shows that the *DispNetC* network [20] fails to estimate large disparities for the foreground objects on the table. Meanwhile, the network also fails to preserve the details for far objects with small disparities in the second row.

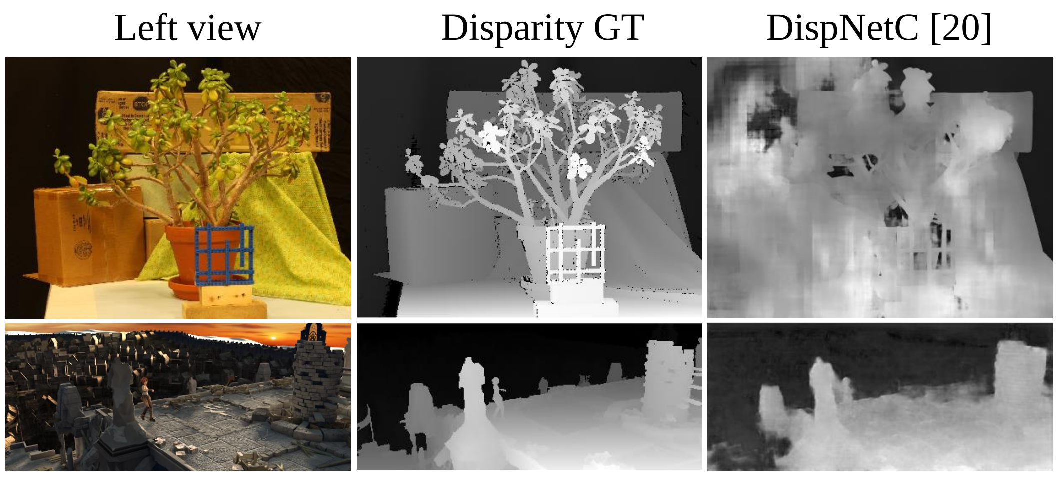

Fig. 4.1: Examples where DispNetC [20] fails, in the first row, to estimate the disparity map for close objects with large disparities (over 480 pixels), and for far objects with small disparities (less than 13 pixels), in the second row.

When the displacements between left and right views cover a large range from small to very large values, *DispNetC* fails to estimate reliable and accurate disparity map that covers the whole range of disparities. To address this problem, a multi-resolution based approach is proposed in this chapter. The proposed approach handles well both large and small disparities, and it includes three stages. In the first and second stage, several initial disparity maps are estimated from stereo images of the same scene with different spatial resolutions. These multiple resolution versions are obtained, at the first stage, by down-sampling and up-sampling the input stereo pair with different factors. Small resolution stereo pair can provide accurate disparity for close and foreground objects but lacks details for background objects. Large resolution one is able to estimate reliable disparity for background, preserving the far object details. For the third stage, two end-to-end Multi-Resolution Disparity Fusion networks *MRDF-Conv* and *MRDF-Conv-Deconv* are introduced. The task of the two networks is to combine the multi-resolution estimated disparities into a single accurate one. *MRDF-Conv* is a light-weight and computationally efficient network, where the multi-resolution disparity images are re-sampled directly using a standard interpolation method to the desired size, then the re-sampled versions are stacked together and processed through a number of convolutional layers for merging. This network shows competitive results compared with the state-of-the-art disparity estimation methods. In an attempt to further improve the overall performance and retain the sharpness of the disparity boundaries, we also design the *MRDF-Conv-Deconv* network. In this network, standard interpolation method is replaced with several convolution and deconvolution layers. This network learns rich hierarchical features at different levels to progressively resolve the ambiguity in the disparity map up-sampling process. Therefore, it leads to better performance than *MRDF-Conv*.

For texture images, a somehow similar paradigm fuses two or more partly focused images of the same scene to generate a “fully focused” image [99, 100, 101]. Unlike these multi-focus-images fusion methods, the proposed approach requires multi-resolution estimated disparities in order to generate a more accurate estimation of the disparity.

In summary, this chapter introduces:

1. A new framework to mitigate limitations of current disparity estimation frameworks. In particular existing methods fail to estimate reliable and accurate disparity map that covers a large range of disparities. We propose to use multiple resolution versions of the initial stereo images, which provide scalable information to improve the feature map matching, and expand the estimated disparity range.

2. Two end-to-end Multi-Resolution Disparity Fusion networks *MRDF-Conv* and *MRDF-Conv-Deconv*. *MRDF-Conv* is a light-weight and computationally efficient network with acceptable performance. *MRDF-Conv-Deconv* replaces all the standard interpolation units with learnable convolution and deconvolution layers, and achieves better performance.

3. In principle, the proposed framework can be applied to the output of other stereo-matching methods. In general, it can be used to enhance the accuracy of other existing disparity estimation methods.

4. We extensively evaluate the proposed approach on KITTI-2015 and Sintel benchmarks. In particular, the mean absolute error is reduced to 9.89 from 16.12 for the *DispNetC* approach on the Driver dataset. The results demonstrate that there is significant performance gain by using multi-scale scheme over one single scale.

4.2 Motivation and Observation

In this section we present our observation and motivation behind the work.

4.2.1 Single-resolution Disparity Estimation

Stereo images, specifically in outdoor scenarios, contain structures for both close and far objects. Estimating accurate disparity for the whole scene, keeping the details for foreground and background structures, is a difficult task especially when the disparity range is large. This chapter introduces a solution to handle the disparities for close and far objects separately. Let $\mathbf{X}$ be the input left and right images with size $H \times W$.

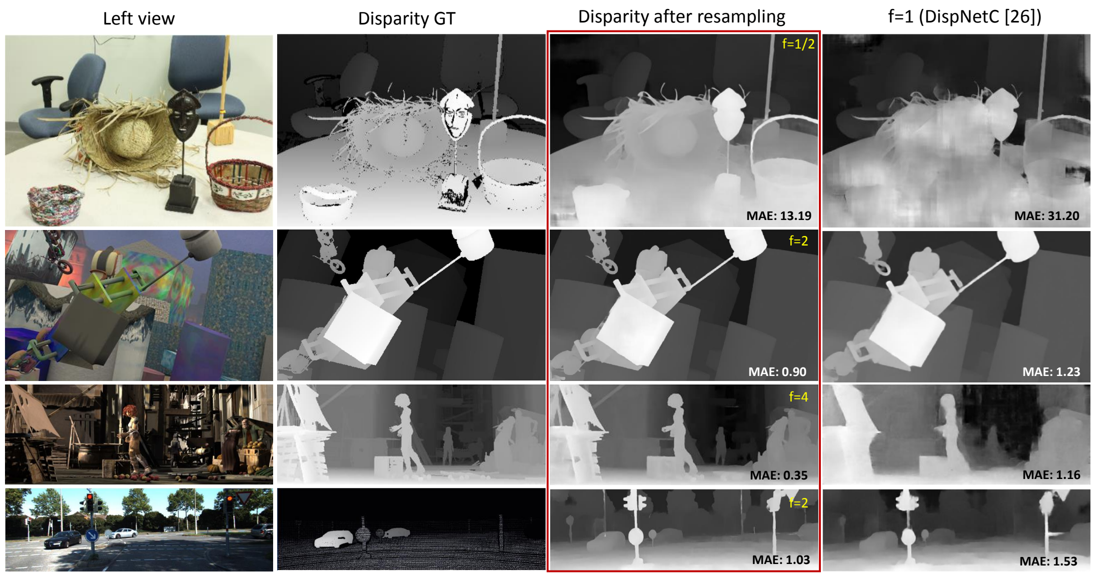

Fig. 4.2: Disparity estimation examples using different re-sampling factors in comparison with *DispNetC*. Each column from left to right : left view, disparity ground truth, disparity estimated using re-sampled stereo pairs (the re-sampling factor is indicated in the top right corner of each image), and disparity estimated using DispNetC. Rows from top to bottom: Middlebury 2014, FlyingThings3D (clean), Sintel (clean), KITTI 2015.

The input image is re-sampled to different sizes using several factors $f \in \{f_1, f_2, ..., f_n\}$. Let $s_f(.)$ be the re-sampling function that re-sample an $H \times W$ image X to the corresponding image X_f [1], in other words $X_f = s_f(X)$

The size of the re-sampled image X_f is $h_f \times w_f$, where $h_f = fH$ and $w_f = fW$. Each pair of re-sampled images then used to estimate a disparity map using, in this work, *DispNetC*. Given that *DispNetC* generates a disparity map with half the resolution of the input images, then we could write:

$$D_{f/2} = DispNetC(s_f(X)) \tag{4.1}$$

When $f > 1$ the upsampled image $s_f(X)$ aims to generate features with larger resolutions and keep the fine details for far objects.

In a similar way, the problem of estimating large disparities is solved by down-sampling the input images using factors $f < 1$.

Several experiments were conducted to analyze the mean absolute error versus disparities for different re-sampling factors. It was found that each re-sampling factor is suitable for a specific range of disparity, as shown in Table. 4.1. These experiments used the whole 1064 stereo images in Sintel dataset. In summary, the observations are as follows.

- Up-sampling the input stereo pair by a factor of 4 is suitable to estimate disparity for

[1]Here we used the superscript to indicate the scale factor

Table 4.1: The influence of using various re-sampling factors for disparity estimation on Sintel dataset. The disparity errors is measured by Mean Absolute Error (MAE) on various disparity ranges, D_m is 160 pixels.

Resampling factor	[0, $D_m/4$]	[$D_m/4 + 1$, $D_m/2$]	[$D_m/2 + 1$, D_m]	[$D_m + 1$, $2 \times D_m$]
f=1/2	3.30	3.90	11.50	**41.84**
f=1	1.40	2.44	**11.24**	91.52
f=2	0.74	**2.15**	15.79	197.13
f=4	**0.59**	9.32	21.75	312.42

a disparity range [0, 40], which is [0, $D_m/4$]; where D_m is the maximum disparity that the main estimating engine was trained to predict, in DispNetC this is 160.

- Up-sampling by factor 2 is suitable for [41, 80], which is [$D_m/4 + 1$, $D_m/2$].
- The original input stereo images are suitable if the disparity range is between [81, 160], which is equivalent to [$D_m/2 + 1$, D_m].
- Down-sampling by factor 2 is suitable for [$D_m + 1$, $2 \times D_m$].

Fig. 4.2 depicts some disparity estimation examples using different re-sampling factors in comparison with *DispNetC*. In the first row, and third column, the down-sampled version by a factor 2 (i.e., $f = 1/2$) is used, this result demonstrates the benefit of using down-sampling when the original stereo pair has disparities larger than 160 pixels. In the second and fourth rows where the disparities are ranged around 80 pixels, the up-sampled version by a factor 2 (i.e., $f = 2$) is used. In the third row, the disparities are less than 40; therefore up-sampling with a factor 4 (i.e., $f = 4$) is suitable since it allows the recognition of the disparities of far "objects". More specifically, each disparity map will generate accurate results within a specific range. Having a scene where all the structures have disparities within, for example, [$D_m/4 + 1$, $D_m/2$], then up-sampling by a factor 2 is suitable. However, typical outdoor stereo images contain both close and far objects, the disparities in a stereo pair might cover the range [0, $2D_m$], which poses a big challenge for stereo matching.

To estimate the disparity of an object with large disparity, we could increase the maximum displacement in the correlation layer in [20], or use a down-sampled version of the stereo pair. In our approach, we use a down-sampled version of the stereo pair. In fact, the receptive field of the CNN kernels on the down-sampled views is larger than that over the non-down-sampled view. In other words, the kernels of the CNN when applied to

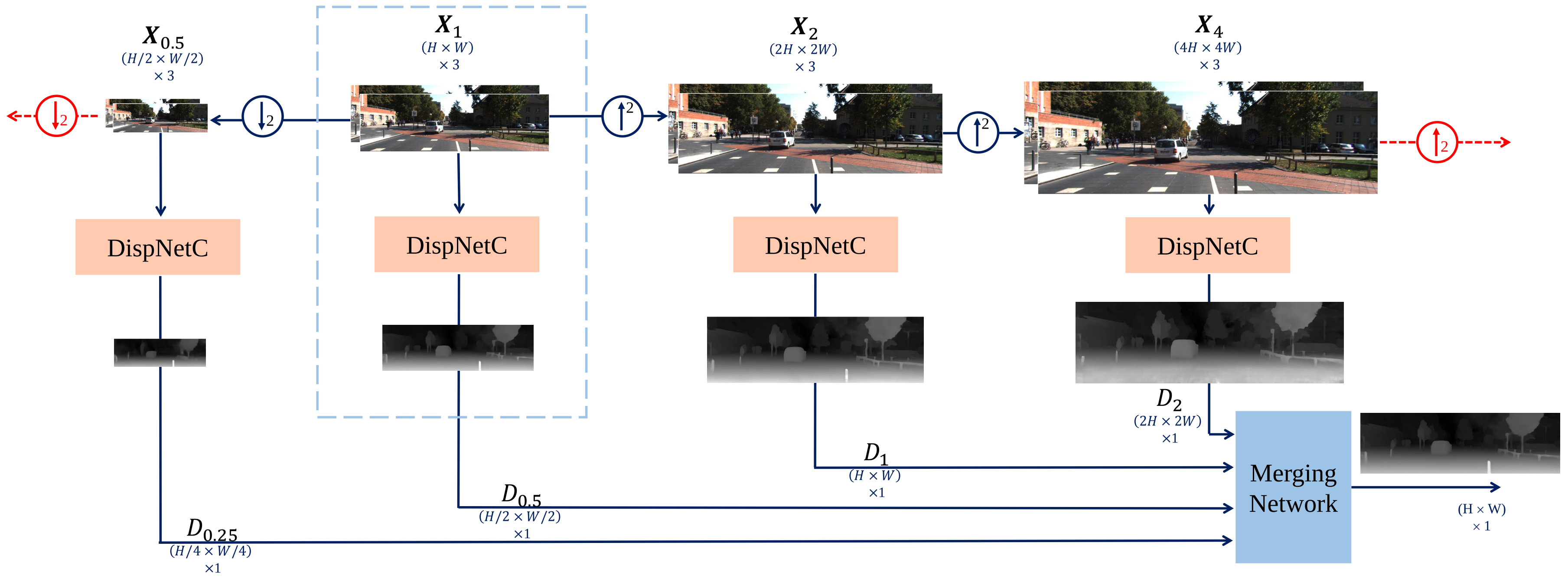

Fig. 4.3: The Multi-resolution disparity maps estimation and fusion framework.

the down-sampled views will effectively span larger area than that covered by the non-down-sampled. This is particularly important for near objects, which usually have large projected areas on the image plane due to their closeness to the camera. Similarly, using a down-sampled version results in better disparities for large texture-less structures. As for estimating small disparity, up-sampled versions are adopted. These observations are the key motivation to introduce the multi-resolution disparity estimation fusion using CNNs into the field of disparity estimation.

4.3 Multi-Resolution Disparity Fusion Approach

The basic idea of the proposed approach is that a number of disparity maps, estimated from a set of stereo images of the same scene with different spatial resolutions, can be merged into a single representation. In this chapter, two CNN based multi-resolution disparity fusion networks are proposed, namely *MRDF-Conv* and *MRDF-Conv-Deconv*. Both networks can handle complex scenarios where the stereo images contain both foreground and background objects. The *MRDF-Conv* is light-weight and uses a standard interpolation method to re-sample the disparity maps to the same resolution of the original size of the input stereo pair. Then a merging part with convolutional layers is employed. *MRDF-Conv-Deconv* generates features of the disparity maps and re-samples the obtained features using convolution and deconvolution layers. Using convolution and deconvolution layers for feature re-sampling enable end-to-end learning. The details and the performance of the proposed networks will be discussed in details in the following subsections.

4.3.1 Multi-Resolution Disparity Estimation Framework

The proposed framework is shown in Fig. 4.3. The upper part represents *stage I* where the input pairs are re-sampled to different resolutions. Then, in *stage II* each pair of re-sampled images will be employed to generate a disparity map. The *DispNetC* [20], which has foreword run time of 0.062 sec, was chosen as the main engine for disparity estimation, but other methods could be used as well.

Taking into consideration the size of images in the available stereo datasets, the disparity estimation range for *DispNetC* and the available dedicated graphics memory in the Titan X GPU, we use four different scales to generate the multi-resolution disparity maps. Moreover, using another engine for disparity estimation or more powerful GPUs will allow using more scales. Nevertheless, during our experiments, we found that using these scales generates accurate disparities that cover the disparity range of the available stereo datasets.

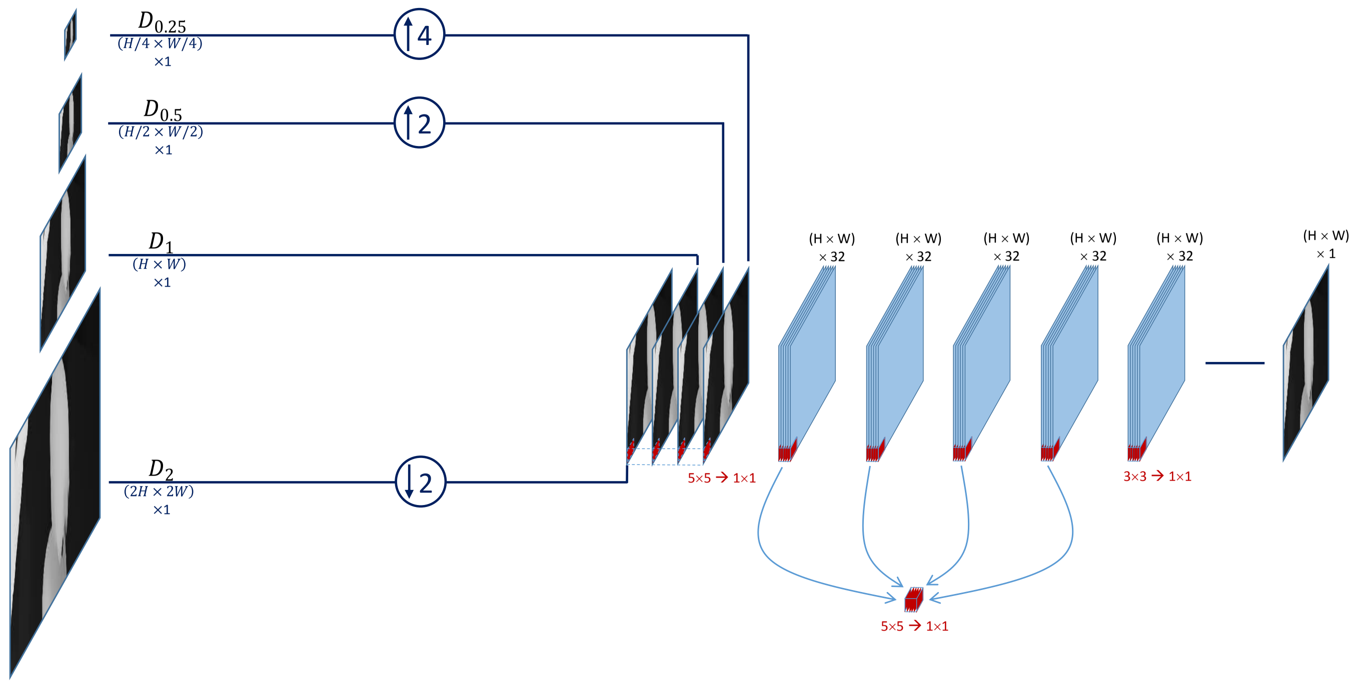

Fig. 4.4: The architecture of the merging MRDF-Conv network. The input is multiple disparity maps estimated for a particular scene using different resolutions. The proposed network, in this picture, uses two upsampled versions by factors 2, 4 and one downsampled version by 2. The output is a single disparity map that integrates all the input maps.

4.3.2 Convolution-based Multi-Resolution Disparity Fusion Network

We propose an end-to-end learning approach to merge a set of disparity maps into a single representation. We introduce the Convolution-based Multi-Resolution Disparity Fusion network (*MRDF-Conv*) as the first network for disparity map fusion. This multi-resolution scheme requires a traditional interpolation technique to re-sample the input images before the fusion stage.

The first choice is to stack the input images together and process them through a number of convolutional layers, allowing the network to learn how to combine the images. *MRDF-Conv* only consists of convolutional layers. In the *MRDF-Conv* network, neither pooling nor sub-sampling is applied, and the reason for this is to avoid reducing the resolution of the final merged map. Fig. 4.4 illustrates the architecture of the merging network. The red boxes in the figure present the kernel size for each layer.

The first step of the *MRDF-Conv* framework is to re-sample the input disparities $\{D_{f_1}, D_{f_2}, \ldots, D_{f_n}\}$ to the desired resolution, the same resolution of the original stereo pair ($R = H \times W$). A traditional interpolation method is applied directly on the disparity maps. After that, the re-sampled versions are concatenated to form a $H \times W \times 4$ feature map and provided as input for the merging network stage.

4.3.3 Convolution-Deconvolution-based Multi-Resolution Disparity Fusion Network

Disparity boundaries often lose sharpness when they are up-sampled to higher resolutions. This in return might affect the overall performance of the merging process of the MRDF-Conv approach. To address this problem we introduce the Convolution-Deconvolution-based Multi-Resolution Disparity Fusion (*MRDF-Conv-Deconv*) network. This framework adopts an alternative strategy and performs feature extraction directly on the original low-resolution disparities with convolution and deconvolution layers. These features are re-sampled progressively in multiple layers to the desired resolution. Replacing all the standard interpolation with learnable convolution and deconvolution layers enable to tackle the problem, of merging multi-resolution disparities, by an end-to-end learning approach. In other words, this network introduces multi-resolution disparity fusion into the field of disparity estimation, by jointly learning re-sampling and fusion of multi-resolution disparity maps using a CNN model.

MRDF-Conv-Deconv Architecture Fig. 4.5 shows an overview of the *MRDF-Conv-Deconv* architecture. It consists of various units including, feature extraction, down-sampling, up-sampling and merging. The details of each unit will be discussed in this section.

Feature Extraction The *MRDF-Conv-Deconv* trains a feature extractor for low resolution disparity maps, then use it to extract features before feeding them to the up-sampling network part. The feature extraction module can be expressed as:

$$\mathbf{F}_{i,1} = max(0, \mathbf{W}^c_{i,1} * D_i + \mathbf{b}^c_{i,1}), \tag{4.2}$$

where $\mathbf{W}^c_{i,1}$ *(subscript c for convolution)* and $\mathbf{b}^c_{i,1}$ denote the convolution filter and bias of the first layer for the low resolution branches i , respectively; $\mathbf{F}_{i,1}$ indicates the output feature maps of the first layer for the branch i, with D_i denoting the original low resolution input for the the branch i. The operator $*$ represents convolution. In a similar way, the feature extraction is applied to higher resolution disparity maps to generate features with the desired resolution using different convolution strides.

Multi-scale Feature Up-sampling Given the extracted features from the original low-resolution disparities, an upsampling operation is performed to increase their spatial size to the desired high resolution. Instead of using standard interpolation methods, we use a learning-based upsampling operation with an end-to-end training process. In this network, we consider a strategy widely adopted in CNN for up-sampling, which is deconvolution [58] (**deconv**). Deconvolution uses backward convolution in which it diffuses a set of feature maps to another set of larger feature maps. Suppose that up-sampling by a factor of $s = 2^k$ is required for the branch i, then k deconvolution layers (levels) are needed in the up-sampling process. The feature up-sampling can be expressed as follows:

$$\mathbf{F}_{i,k} = max(0, \mathbf{W}^c_{i,k} * \mathbf{FU}_{i,k-1} + \mathbf{b}^c_{i,k}), \tag{4.3}$$

where $\mathbf{W}^c_{i,k}$ and $\mathbf{b}^c_{i,k}$ denote the trainable kernels and bias of the layer k for the low resolution branches i , respectively; $\mathbf{F}_{i,k}$ indicates the upsampled features in the layer k (output) for the branch i, with $\mathbf{FU}_{i,k-1}$ denoting the features map obtained by upsampling the input feature maps $\mathbf{F}_{i,k-1}$, in the layer $k-1$ (input) for the branch i, by a factor of four by zero insertion.

In the proposed network, a different number of deconvolution layers is used to generate features with the desired resolution for each branch based on the input resolution. Overall, all the obtained features from each branch share the same resolution as the original stereo images $H \times W$.

Feature Fusion The extracted features from each branch, obtained from the CNN-base resampling phase, are concatenated together and fed through six convolution layers to learn the mapping process and predict the final disparity map. In this stage, convolution with stride one, and without pooling is applied in order to avoid reducing the resolution of the feature maps and generate a dense disparity map with the same size of the original stereo images.

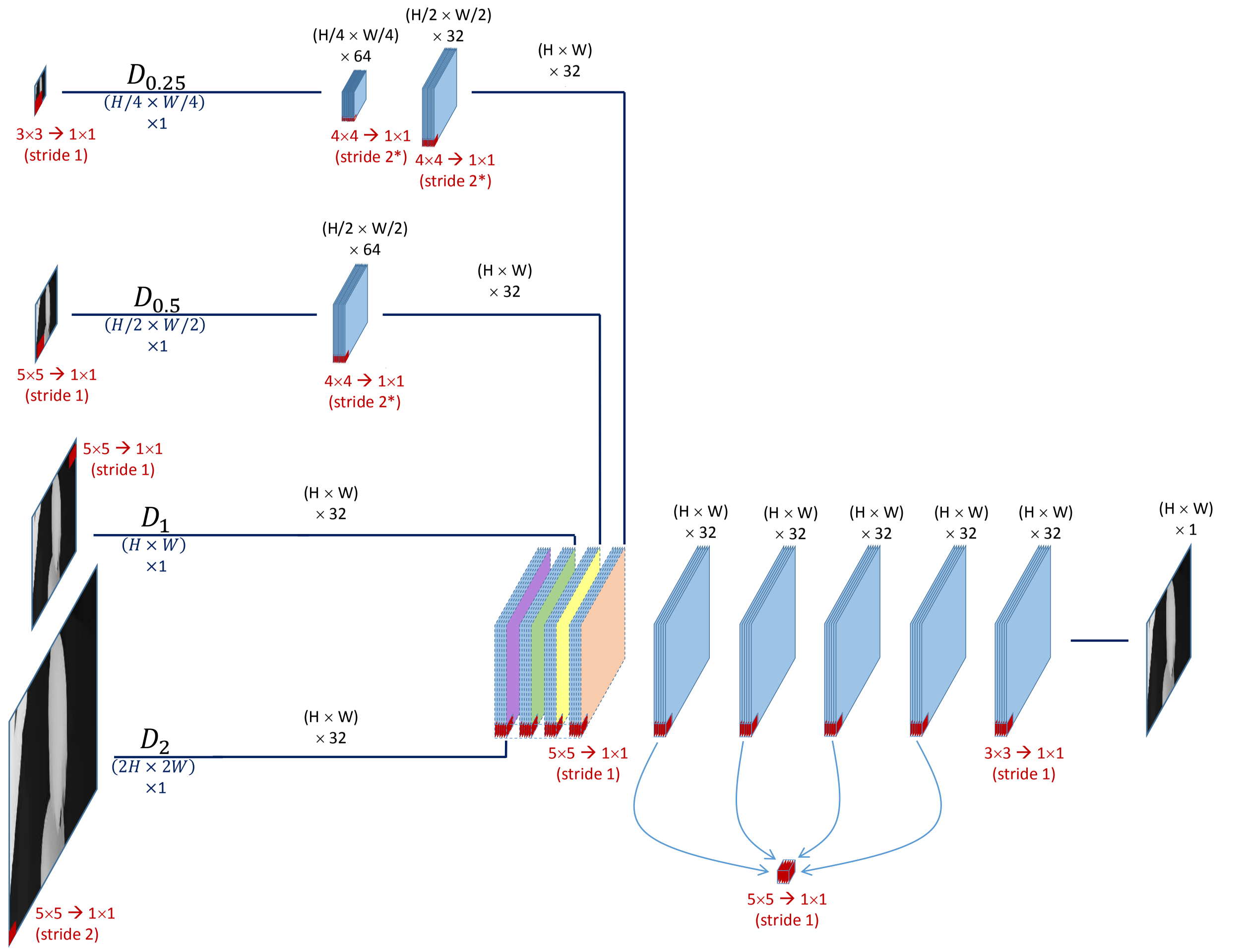

Fig. 4.5: The architecture of the merging MRDF-Conv-Deconv network. The network is fed with a set of multiple resolutions disparity maps. The output is a single disparity map that integrates all the input scales. (2^* indicates that the input feature map was upsampled by a factor of four by zero insertion, and the convolution is applied with stride 2 which results in upsampled feature map by factor two).

4.3.4 Training Procedure

Let us assume the N training inputs are $\{\mathbf{D}_j, D_j^{gt}\}_{j=1}^N$, where $\mathbf{D}_j = \{D_{f_1}, D_{f_2}, \ldots, D_{f_n}\}_j$ is the set of multi-resolution disparity maps generated from the stereo pair $\mathbf{X}_j$ using the first and second stage of the proposed framework, and the label D_j^{gt} is the ground truth disparity map of the pair $\mathbf{X}_j$. Denote $\theta = \{\mathbf{W}, \mathbf{b}, \mathbf{a}\}$ as the network parameters controlling the foreword process, our goal is to train the two models *MRDF-Conv* and *MRDF-Conv-Deconv* to combine the multi-resolution disparities into a single map $\hat{D}$. We trained our networks by minimizing the norm $L1$ for the N training samples as follows:

$$L(\theta) = \frac{1}{N} \sum_{j=1}^{N} ||\hat{D}_j(\theta) - D_j^{gt}||, \tag{4.4}$$

where $\hat{D}_j(\theta)$ is the merged disparity map obtained from the inputs $\mathbf{D}_j$ and the set of network parameters θ.

The optimization is conducted by the mini-batch stochastic gradient descent method, with momentum being 0.9 and weight decay being 0.005.

We used the Caffe framework [58] and the optimization is employed using Adam method [102]. We set $\beta_1 = 0.9$ and $\beta_2 = 0.999$ as in Kingma et al. [102]. All the filters in the convolution layers are randomly initialized using *MSRA* method [103], this method is proposed to solve the training of extremely deep rectified models directly from scratch. The filters in deconvolution layers are initialized from bilinear interpolation kernels. The learning rate is initially set to $1e^{-4}$ and then get divided by 2 every $20k$ iterations starting from iteration $40k$.

The proposed networks were trained using the MPI Sintel [104] dataset. Sintel is a synthetic dataset derived from a short open source animated 3D movie. It provides dense ground truth for the disparity. It contains sufficiently realistic scenes including natural image degradation effects such as fog and motion blur. This, in addition to the fact that the dataset contains 1,064 stereo frames for training, makes this dataset suitable for training the proposed models.

4.4 Experimental Results

We evaluated our approach on three real stereo datasets including the Middlebury dataset [105] which contains 81 image pairs with disparities larger than 600 pixels, the KITTI 2015 dataset [106, 107, 104] with 200 image pairs, and KITTI 2012 dataset with 194 image pairs. The evaluation was also performed on 3 synthetic datasets including FlyingThings3D [20] which contains 4760 frames for testing, Driving dataset [20] with 4392

frames and Monkaa dataset [20] with 8591 frames. The MAE was used as an error measure.

Table 4.2: Disparity estimation error for various re-sampling factors and several datasets within certain different disparity ranges. All measures are mean absolute errors (MAE).

Re-sampling factor	Sintel			FlyingThings3D	Middlebury2014
	disp≤40	disp≤80	disp>160	disp≤80	disp>160
f=1	1.26	1.99	14.29	0.88	18.82
f=$1/2$	2.70	4.35	**8.04**	2.48	**9.65**
f=2	0.63	**1.26**	30.65	**0.49**	30.36
f=4	**0.44**	1.49	69.82	1.25	85.12

We conducted several experiments to show the effectiveness of disparity estimation using the proposed approaches. In these experiments, in addition to the original images, the input stereo images were re-sampled by factors of $\{0.5,\ 1,\ 2,\ 4\}$. *Bicubic* is used as a method for down/up-sampling the stereo images. The *DispNetC* [20] was used to generate a disparity map for each pair of re-sampled images, thus obtaining a set $\mathbf{D} = \{D_{1/4}, D_{1/2}, D_1, D_2\}$. There are mainly two reasons for choosing these scales. First, these scales can cover a disparity range of [0, 320] pixels, which is suitable for most of the available datasets. Second, choosing a higher resolution, i.e., $f > 4$, will provide more details for far objects, however, it requires larger memory size and longer runtime. In summary, using these scales is suitable for available datasets and keeps the proposed approach computationally efficient.

The results reported in Table. 4.2 show how the accuracy of the estimated disparity, within certain disparity range, is influenced by the resolution of the input images. The results are reported in term of MAE for different re-sampling factors on Sintel, FlyingThings3D, and Middlebury 2014. All the obtained disparity maps are resized to the same size of the ground disparity map to evaluate the MAE. The results demonstrate that each resolution has superior performance in a certain disparity range. For example, in the Middlebury 2014 and Sintel (clean) datasets, it is possible to note that $f = 1/2$ leads to the lowest MAE for the pixels belonging to close objects (i.e., pixels with disparities larger than 160).

4.4.1 Single Resolution vs Multi-resolution Estimation

This subsection compares the performance of the two proposed merging networks with the disparities obtained from single resolution pairs of texture images; the results of these

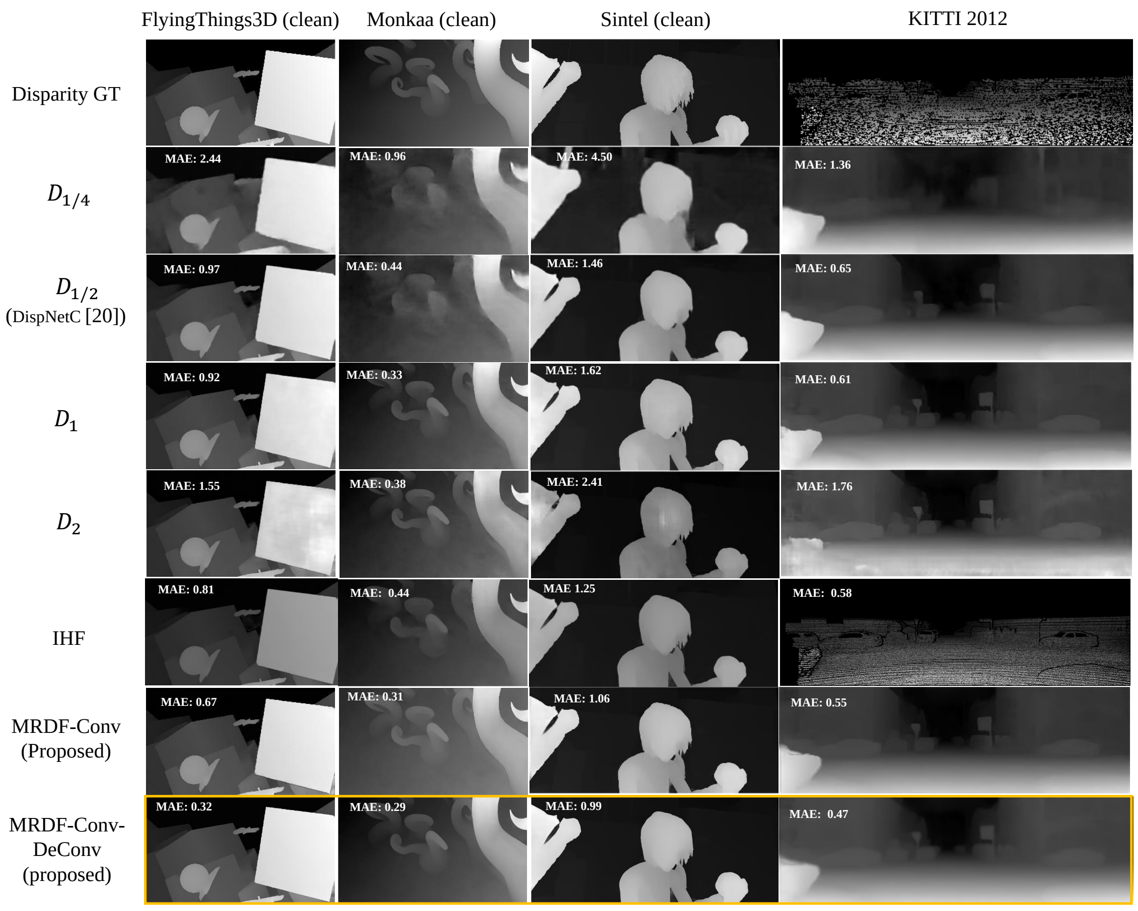

Fig. 4.6: Examples of disparity estimation using the proposed multi-resolution merging networks MRDF-Conv and MRDF-Conv-Deconv in comparison with single resolutions. MAE is reported in the top left corner. Rows from top to bottom: Disparity ground truth, $D_{1/4}$, $D_{1/2}$, D_1, D_2, IHF, MRDF-Conv, MRDF-Conv-Deconv. Each column from left to right: FlyingThings3D (clean), Monkaa (clean), Sintel (clean) and KITTI 2012.

experiments are reported in Fig. 4.6 From this figure, it is possible to note that the high-resolution versions show some errors for foreground objects but provide accurate estimations for the far objects. Whereas, the merged disparity renders accurate disparities for both close and far objects and it preserves the details. Moreover, in the second column, Monkaa dataset, where the disparity is relatively small, we could conjecture that the network relays more on the up-sampled versions. In other words, the network is able to successfully merge different ranges of disparity maps to obtain an accurate one for both close and far objects. In summary, *MRDF-Conv-Deconv* Network yields more robust results and sharper edges than *MRDF-Conv* network, as it employs learnable convolution and deconvolution layers for up-sampling and down-sampling disparity maps.

In order to show the importance of merging different scales, Table. 4.3 reports the obtained MAE with different scales and with the Ideal Hard Fusion (IHF) algorithm.

Table 4.3: Performance comparison (MAE) of the two proposed approaches with respect to other approaches.

Method	KITTI 2015	Driving	FlyingThings3D clean test	Monkaa clean	Time
IHF	1.10	**7.71**	1.19	**2.58**	0.04s
MRDF-Conv (proposed)	1.35	10.32	1.36	4.10	0.62s
MRDF-Conv-Deconv (proposed)	**0.77**	9.89	**1.11**	3.78	1.03s
f=1/2	3.02	12.32	3.38	5.70	0.04s
f=1 (DispNetC [20])	1.59	16.12	1.68	5.78	0.06s
f=2	1.54	20.24	1.74	10.40	0.17s
f=4	2.33	25.87	3.09	16.35	0.35s
DispNet [20]	2.19	15.62	2.02	5.99	0.06s
SGM [108]	7.21	40.19	8.70	20.16	1.1s
MC-CNN-fst [54]	-	19.58	4.09	6.71	0.8s

This one selects for each pixel the best-estimated disparity, among the various scales, this method is the best that could be done as a selection method. More specifically, for the estimations $\{D_{f_1}, D_{f_2}, \ldots, D_{f_n}\}$, for each pixel p in the *IHF* disparity map we choose the one with the minimum distance to the ground truth according to the following equation:

$$\overline{d(p)} = argmin_{\hat{d}(p) \in \{D_{f_i}(p), i \in \{1,\ldots,n\}\}} |\hat{d}(p) - d^{gt}(p)|, \tag{4.5}$$

where $\overline{d(p)}$ is the closest disparity value to $d^{gt}(p)$, the ground-truth disparity value for pixel p, among different estimates. It is interesting to see that the results, obtained by *MRDF-Conv-Deconv* network on FlyingThings3D and KITTI 2015 datasets, present a superior performance over the *IHF*, as shown in the third row of Table. 4.3. It is important to note that we fine-tune our network on both FlyingThings3D and KITTI 2015 datasets, while no fine-tuning was conducted for other datasets. It is probably the reason why only on these two datasets, *MRDF-Conv-Deconv* outperforms *IHF*. Nevertheless, *MRDF-Conv-Deconv* outperforms any other schemes, including SGM [108], MC-CNN-fst [54], *MRDF-Conv*, and DispNet [20] with various single scales.

4.4.2 Benchmark Results

We further compared our method with several existing disparity estimation methods on the bench mark KITTI 2015, Driving, FlyingThings3D and Sintel datasets. We evaluate the performance of DispNet and DispNetC in [20], MC-CNN-fst [54] and the popular Semi-Global Matching approach [108] with a block matching implementation. The results in Table. 4.4 show that the proposed method has superior performance compared with *DispNetC* and *DispNet* [20], SGM [108], and MC-CNN-fst [54] on both synthetic and real stereo images for close and far objects. In Table. 4.4 the *D1-bg*, *D1-fg* and *D1-all* are error measure as reported by the KITTI evaluation server (percentage of pixels with estimation error $> 3pixels$ and $> 5\%$ of the true disparity), and *fg* and *bg* stands for foreground and background regions, respectively. From this table, it is possible to note that the proposed MRDF-Conv network provides accurate disparity not only on the background but also on foreground when compared with other methods.

Table 4.4: Out-Noc error on KITTI 2015 testing dataset. Rank is based on D1-all error.

Method	D1-bg	D1-fg	D1-all	Rune time
MRDF-Conv (proposed)	3.55 %	4.81 %	**3.76** %	0.62 s
MC-CNN-fst [54]	**2.89** %	8.88 %	3.89 %	67 s
CNN-SPS [109]	3.30 %	7.92 %	4.07 %	80 s
DispNetC [20]	4.32 %	**4.41** %	4.34 %	0.06 s
Content-CNN [110]	3.73 %	8.58 %	4.54 %	1 s
OSF+TC [111]	4.11 %	9.64 %	5.03 %	50 min
MDP [112]	4.19 %	11.25 %	5.36 %	11.4 s
FSF+MS [113]	5.72 %	11.84 %	6.74 %	2.7 s

Fig. 4.7 shows some disparity estimation examples using our approach compared with other methods. Our results are shown in the third and fourth rows. We can observe that the *MRDF-Conv* performs well and outperforms both SGM and MC-CNN on both synthetic and real datasets.

4.4.3 Re-sampling Horizontally and Vertically

In this subsection, we compare the performance of the proposed method when re-sampling on both vertical and horizontal directions versus re-sampling only horizontally. This latter case speeds up the estimation process and reduces the demand on memory. Table. 4.5, reports the results of this comparison using MRDF-Conv approach. From these results, we can see that both approaches yield to comparable results, nevertheless, the network trained with only horizontally up-sampled images will perform better in terms of run time, and memory demands.

Table 4.5: MAE comparison between horizontally elongated patches and re-sampling the patches on both horizontal and vertical directions. The measuring meathod MAE is used.

Re-sampling method	KITTI 2012		KITTI 2015	
	H & V	H	H & V	H
MRDF-Conv(Proposed)	1.39	1.22	1.35	1.44
f=1/2	3.00	2.23	3.02	2.24
f=1 (DispNetC [20])	1.75	1.75	1.59	1.59
f=2	1.82	1.83	1.54	1.65
f=4	3.78	3.20	2.33	2.3

4.4.4 Various Re-sampling Methods

In this experiment, the impact of different interpolation methods of the input stereo images on the disparity estimation was investigated including *Bicubic*, nearest neighbor, bilinear interpolation method.

The performance of these re-sampling methods was evaluated on two datasets KITTI 2012 and KITTI2015. Table. 4.6 reports the mean absolute error of these experiments; it is interesting to see that, although the single resolution performance get affected by the resampling method, the proposed merging approach provides almost the same performance, in other words, any of these methods can be used for the proposed approach, thus to enhance the runtime a fast method could be chosen.

Table 4.6: Comparision between different re-sampling methods using MAE.

Re-sampling method	KITTI 2012				KITTI 2015			
	f=1/2	f=2	f=3	MRDF-Conv	f=1/2	f=2	f=3	MRDF-Conv
Bicubic	3.00	1.82	3.78	**1.39**	3.02	1.54	2.33	**1.35**
Bilinear	3.01	1.87	3.77	**1.39**	3.02	1.54	2.23	**1.34**
Nearest	3.03	1.82	3.63	**1.39**	3.03	1.54	2.25	**1.35**
Average	3.02	1.84	3.78	**1.40**	3.03	1.55	2.31	**1.35**

4.5 Conclusion

To address the challenging task of wide range disparity estimation, we proposed a novel framework that can accurately estimate a disparity map over wide disparity range for both close and far objects with large and small disparities. Firstly, the input stereo images are re-sampled to different resolutions, then a deep CNN is used to predict several disparity maps from each pair of the re-sampled versions. High-resolution pairs allow the proper estimation of the disparities of far objects, whereas, low-resolution pairs are employed to estimate the disparities for close objects. Finally, two networks, *MRDF-Conv*, and *MRDF-Conv-Deconv*, are introduced to merge the multiple estimations of the disparity maps to generate more accurate disparity that handles a large range of displacements and keep the fine details. *MRDF-Conv* requires fewer parameters and costs less computational time than *MRDF-Conv-Deconv*. On the other hand, *MRDF-Conv-Deconv* retains the sharpness of the boundaries. Experiments performed on the Middlebury2014 and KITTI2015 benchmarks demonstrate the accuracy of the proposed methods.

It has been shown that the multi-resolution based hierarchical estimation of disparity maps yields accurate results while being computationally efficient as well. Future work could be devoted to integrating the whole framework into a single CNN that apply to resample on feature maps into various resolutions then concatenate and process them in a single channel and reconstruct a single disparity map.

It is worth reporting that the proposed scheme in this section has led to the following publications:

1. Samer JAMMAL, Tammam TILLO, and Jimin XIAO, "Multi-resolution for disparity estimation with convolutional neural networks," In Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Dec 2017, pp.17561761.

2.

3. Samer JAMMAL, Tammam TILLO, and Jimin XIAO, "Multi-Resolution Disparity Estimations Fusion Using Convolutional Neural Networks," IEEE transactions on multimedia, 2019 (submitted, under revision).

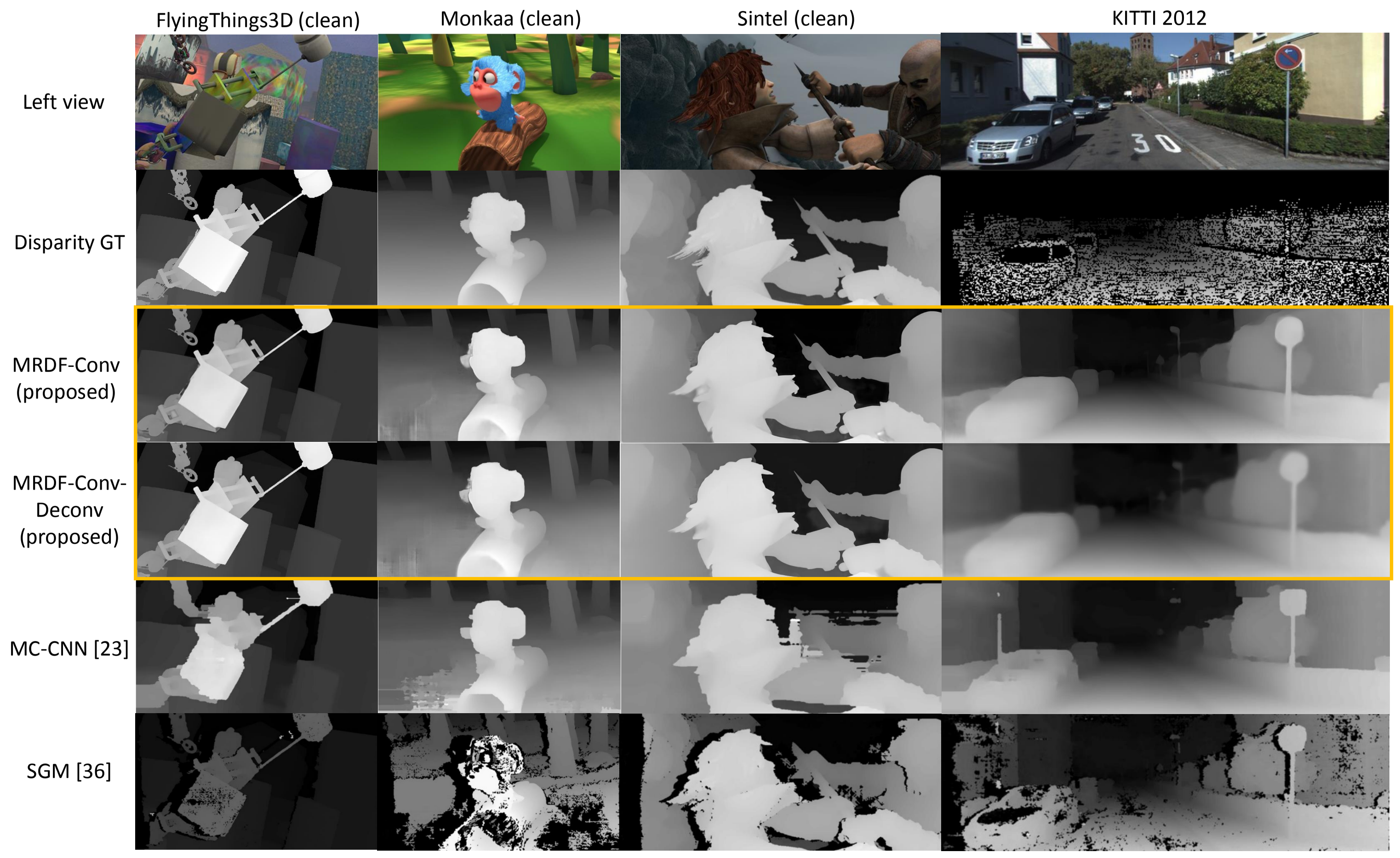

Fig. 4.7: Shows some images for subject assessment of the proposed method. Each column shows an example of one dataset, in particular from left to right these are : FlyingThings3D (clean), Monkaa (clean), Sintel (clean), KITTI 2012. Rows from top to down show: left view image, ground truth disparity, MRDF-Conv, MRDF-Conv-DeConv, MC-CNN and SGM.

Chapter 5

Silhouette-Based Multiview Video Synthesis using Deep Convolutional Neural Networks

In the previous chapters disparity estimation works have been presented. In this chapter, a novel view synthesis method for multiview video using deep convolutional neural networks are presented.

5.1 Introduction

Multiview video technologies have attracted considerable attention in recent years as it provides the depth perception of 3D scene. The main problem for multiview videos applications, as we have seen in Chapter 1, is that a large number of different views, with a small baseline, are recorded by multiple synchronized cameras. In fact, increasing the number of recording cameras can provide users with a more realistic 3D viewing experience. On the other hand, this imposes a huge load on the storage, compression, and transmission of multiview video data. A solution is to represent some views (let us call them *reference views*) and use view-interpolation methods to generate some additional views in between these reference views, this reduces the number of recorded videos and the transmission bitrate, systems using such approach would transmit a limited set of viewpoint videos. From now on we will denote these additional interpolated views by *intermediate views*.

View synthesis techniques are considered as a promising tool for rendering intermediate views from existing views. However, view synthesis is a challenging task, in particular for wide baseline views, where synthesized views might be significantly different from existing views. Existing approaches mainly focus on two key points, enhancing the quality of the synthesized view and avoiding the computational-intensive rendering process.

As we have seen in Chapter 1, depth image based rendering [14] is a popular technique for rendering virtual views for the 3DTV format [7, 8]. Instead of transmitting a large number of views to the receiver, only a small number of views are transmitted along with their corresponding depth maps. The missing views are synthesized from these transmitted views and their depth maps, as a result it helps to reduce storage and transmission bandwidth for multiview videos. Unfortunately, DIBR method suffers from the disocclusion problem, some regions in the virtual view are occluded by foreground objects. Generally, an inpainting method is used to fill in these areas [114]. Therefore, a hole filling post processing phase is required which makes such methods computationally expensive. Moreover, rendering high quality image-depth pairs is difficult, computationally expensive, and not always available.

Recently, few works [44, 46] employed convolutional neural networks to generate a new viewpoint from multiple posed images without depth maps. Kalantari et al. [44] used learning based method for synthesizing novel views from a set of input views captured with a consumer light field camera. Particularly, they were able to generate full 8×8 real scenes light field images from only four corner views. In [46] presented a novel deep architecture that takes two viewpoints as input to predict intermediate views. This technique is quite useful for many applications. However, these networks are unable to generate reliable results due to two major factors. First, some parts of the scene might be occluded in the input views but required for novel ones. Second, object displacement information, which is crucial to determine pixels correspondence between views, is not provided. Consequentially, the generated new viewpoints might suffer from blurred edges and more importantly shape deformation, in particular for moving objects.

Based on the above consideration, in this work we propose to address this problem by representing required views using only their edges, while dropping their texture content. The texture content get synthesized by a CNN from the central view. The proposed concept of a multiview video system is similar to a multiview video plus depth system, but instead of using the disparity(depth) maps we use the edges maps of the views. In the following, the view most likely to be watched is denoted *central view*,while the remaining views are called *lateral views*. It follows the rule that the central view is requested with high quality, while the lateral views have lower quality levels. Meanwhile, the edges help to generate accurate virtual views, without having shape deformation for synthesized objects.

To summarize, this chapter proposes a novel multiview video representation method. First, we propose to represent lateral views in a multiview viewo system simply using their edges, while dropping their texture content. These texture contents get synthesized at the receiver side using a convolutional neural network. The edges of the lateral views represent the location of the "objects" in their corresponding views. Moreover, an evaluation of the proposed method on different scenes along with a comparison to both the ground truth and the asymmetric multiview video approach is presented and discussed.

5.2 Proposed Multiview View Synthesis Framework

The multiview video plus silhouette representation is introduced in this section.

5.2.1 Multiview Video Plus Silhouette

In multiview video plus depth format, texture videos are often accompanied by depth information. The main benefit from using a texture-depth pairs is that intermediate views can be rendered. Therefore, MVD is an approach to increase the efficiency of multiview video data transmission by sending a limited number of texture videos along with additional depth data to generate additional videos at different viewpoints. However, as we reported in Section 5.1, synthesizing novel views requires accurate and high resolution disparity maps, which is not always available. In this work, instead of using depth information, we propose to use silhouette maps to generate accurate intermediate views from the existing views, as depicted in Fig. 5.1.

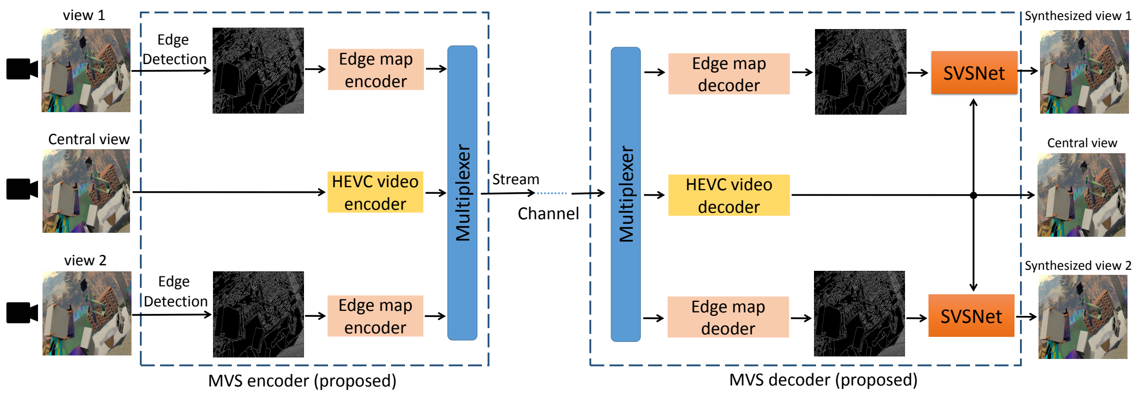

Fig. 5.1: The silhouette-based view synthesis framework for multiview video systems. In the proposed approach, silhouettes, which is the edges map of the lateral views, will sent with the central view. At the receiver side a deep CNN will synthesize the lateral views using their edges and the central view.

The proposed Multiview Video plus Silhouette (MVS) approach extracts and sends only the edges of the lateral view(s) instead of the depth map at the transmitter side. While at the receiver side, the lateral views are rendered from the edge map of the original lateral view and the central view. In MVS, the role of the edge map is of significant importance mainly for two reasons. First, it is very important, specially for large baseline case, to locate the objects in the synthesised view to their correct positions. The edge map solves this problem as it contains the exact location, boundaries, and some details for each object, and the texture of the central view will be used and copied to fill these object edges. Second, these edges help to avoid object blur or deformation when they are rendered in novel views. Therefore, with a deep network, the proposed method learns to fill not only small disocclusions for small baseline cases, but also large disocclusions for large baseline cases. The deep convolutional neural network learns the geometry structure between the central view and the edges of the lateral views, where the edge information is used to facilitate the matching between the central and lateral view. Finally, this method allows a consistent and realistic view rendering.

5.2.2 Network Architecture

In this section, the Silhouette-Based View Synthesis Network Framework (SVSNet), which is based on the DispNet network [20], is introduced as core engine for the proposed approach. The SVSNet is shown in Fig. 5.2. Given a high-quality central view and the edge map of a lateral view, the task of SVS is to synthesize the lateral view. The network inpaints and fills the edge map of the lateral view, taking advantage of the correlation between views. In other words, while view synthesis needs precise per-pixel localization, it also requires to find correspondences between the texture view and the edge map. This involves learning to match the features generated from the central view with the features generated from the edge map at different locations, for various objects and displacements in the two stream. The matching process is achieved using the correlation layer.

The SVSNet Framework is trained using a large dataset that includes the lateral edge map $e^{(i)}$, texture central view $c^{(i)}$, ground truth view (label) $l^{(i)}$. The goal is to train a SVS model that predicts $\hat{v} = SVS_\theta(e, c)$, where $\hat{v}$ is the synthesized lateral view and θ is the network parameters. In order to learn a mapping between $e^{(i)}$, $c^{(i)}$ and $l^{(i)}$, Mean Squared Error (MSE) is used as the objective function:

$$L(\theta) = \frac{1}{N} \sum_{i \in P} ||SVS_\theta(e^{(i)}, c^{(i)}) - l^{(i)}||^2 \qquad (5.1)$$

where P is the pixel set and N is the size of the pixel set. The SVSNet framework is end-to-end trained and no post-processing is required.

5.3 Experimental Results

The validity and effectiveness of the proposed approach were checked on several datasets, and several comparisons were conducted in this section.

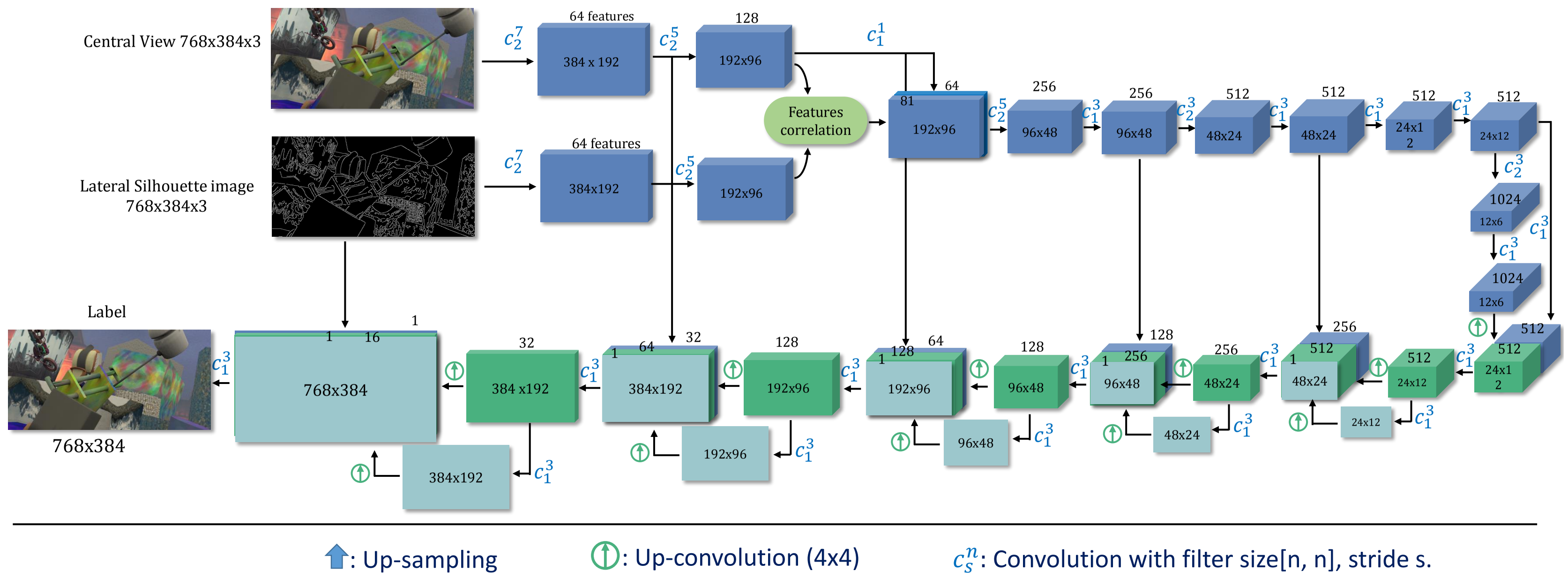

Fig. 5.2: The SVSNet framework which is based on DispNetC. An edge map obtained from the lateral view together with the central view pass through convolutional and upconvolutional layers. The two images are input into two identical and separate convolutional streams to generate features. Then, these features are combined and processed in the contracting part and then in the expanding part to synthesis.

5.3.1 Computer Graphic Training Set

The network is trained using the FlyingThings3D. During the learning stage, each training dataset was pre-processed to generate the silhouette images of the right views, thus, each training instance is composed of the left texture image, the silhouette image extracted from the right texture images (using Canny Edge detector [115]), and the ground truth is the right texture image. Canny Edge detector was used as method for silhouette generation. This method provides only one response per edge. However, other methods could also be chosen. Training SVSNet was performed using patches of size 768×384 which were cropped from the original images of size 960×540 from the FlyingThings3D training set, with a batch size of 4 per iteration. In the fine-tuning phase, the network is trained using the MPI Sintel dataset.

5.3.2 SVSNet Framework Evaluation

The proposed method was evaluated on both real-world datasets KITTI 2015 and synthetic datasets including FlyingThings3D, Sintel, Driving, and Monkaa datasets.

Fig. 5.3 shows that, used network learned to efficiently exploit the central view texture information to generate the texture of the lateral view based on its edges. In general, it could be observed that the network was able to synthesis small details and texture content for both small and large displacement and the occluded areas. For visual results, Fig. 5.4 depicts some examples using the used network for outdoor scenes KITTI 2015 where illumination and edges are more complex. The results show that the synthesized view is robust and accurate for both close and far objects with large and small displacements.

5.3.3 Various Edge Detector Thresholds

Our approach tackle the view synthesis problem by representing lateral views using their edges, while dropping their texture content. Thus, edge information are crucial during the rendering process. Therefore,it is speculated that the more accurate the edge information, the better the obtained results, however, the richer in details the silhouette map the more bitrate is required to transmit this map. Consequently, there is a tradeoff between compression efficiency and edge details. Thus, in this section the impact of different amount of extracted edge was investigated, by using various edge detector thresholds on the quality of view synthesis from silhouette.

The first experiment was conducted using Canny method in Matlab with the default threshold ratio. This automatic(default) threshold provides rich edge map with details about texture objects. The second set of experiments, uses various thresholds related to the automatically generated threshold. This approach allows to analyze the effect of the

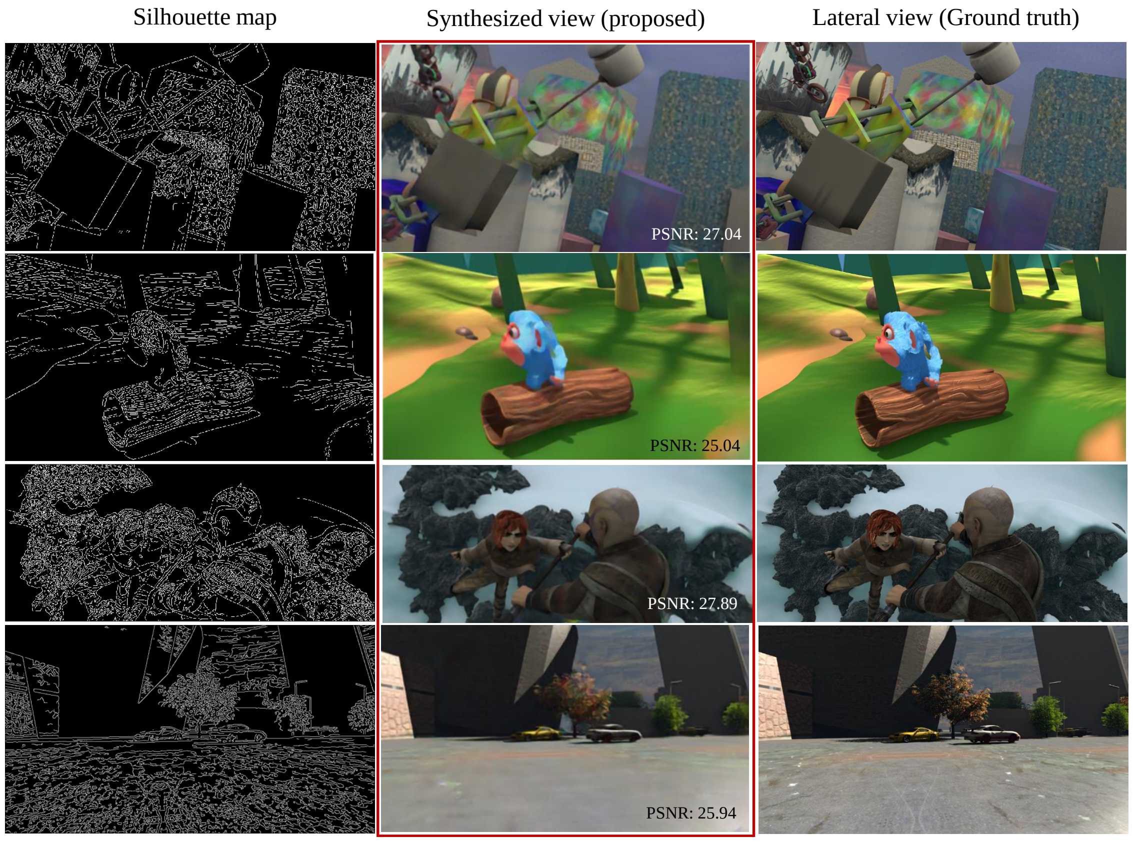

Fig. 5.3: Examples of view synthesis results from silhouette using the proposed framework on computer graphic datasets. Each column from left to right: lateral edge map, synthesized view and the ground truth label. Rows from top to down: FlyingThings3D (clean), Monkaa, Sintel(clean) and Driving.

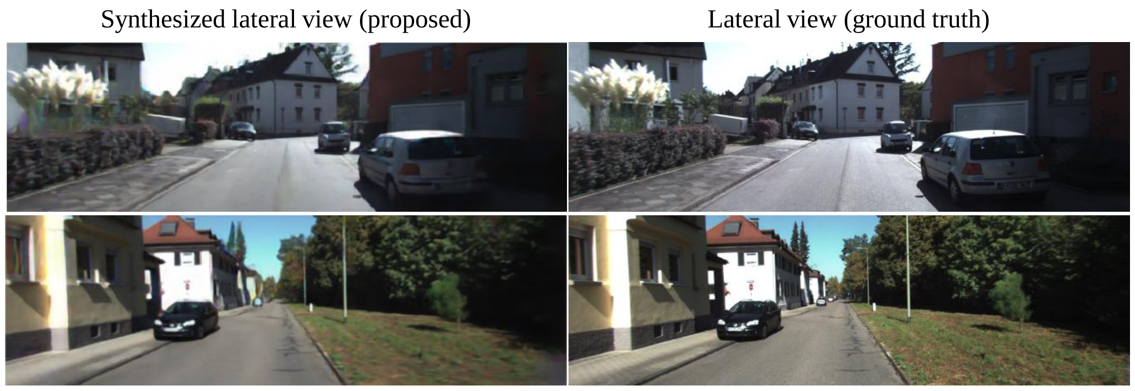
Synthesized lateral view (proposed)
Lateral view (ground truth)

Fig. 5.4: SVSNet evaluation on real outdoor scenes, KITTI 2015. Columns from left to right: synthesized lateral view and the ground truth.

threshold value while using the same mechanism to generate the threshold. The experimental results using these thresholds values ThresholdIn = Auto-threshold $\times\{1, 2, 3, 4\}$ are reported in Table. 5.1. This table demonstrates that, in general, using rich edge map helps to synthesize more accurate views, where the objects are clearly distinguished, whereas, using less complex edge maps reduces the quality of the rendered view, for example, for Sintel dataset the gain with rich edge map could be up to $2.73dB$. Whereas, for Driving dataset the performance is almost similar for the used thresholds. It is worth reporting that for the Driving dataset, which has naturalistic scenery with streets from driving car viewpoint (this dataset is made to resemble the KITTI datasets), the tested thresholds were not suitable due to the nature of this dataset.

Table 5.1: SVSNet evaluation (PSNR) using different edge detector thresholds on Driving, FlyingThings3D test, Monkaa and Sintel datasets

Threshold	Driving	Flying3D	Monkaa	Sintel
Auto-threshold	19.61	**25.13**	**23.11**	**21.89**
Auto-threshold$\times$2	19.72	24.50	22.67	19.16
Auto-threshold$\times$3	19.68	24.35	22.38	20.60
Auto-threshold$\times$4	**19.76**	24.03	21.38	20.26
Average	19.69	24.50	22.39	20.48

Fig. 5.5 illustrates some examples, showing the role of using various edge detector thresholds on the quality of view synthesis from silhouette. It can be observed from the second column, that using a rich edge map leads to synthesis all the objects in the scene with their texture, e.g., for the objects and texture inside the yellow ellipses. In contrast, using different threshold (Auto-threshold$\times$4), where the boundaries of some objects are removed, leads to objects distortion and texture deformation, e.g., red ellipses areas in the fourth column.

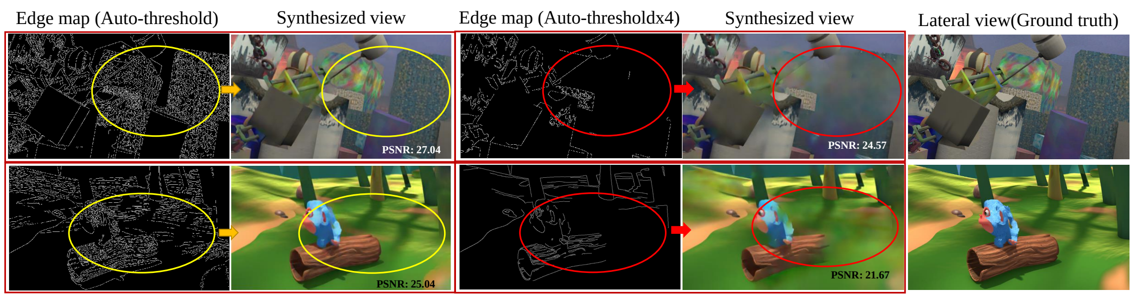
Edge map (Auto-threshold)
Synthesized view
Edge map (Auto-thresholdx4)
Synthesized view
Lateral view(Ground truth)
PSNR: 27.04
PSNR: 24.57
PSNR: 25.04
PSNR: 21.67

Fig. 5.5: Examples of view synthesis from silhouette using different threshold values. Each column from left to right: Edge map, Synthesized view using the left edge map, Edge map, Synthesized view using the left edge map and ground truth label view. Rows from top to bottom: FlyingThings3D (test) and Monkaa.

5.3.4 Visual Comparison with HEVC Baseline

In this section, we further investigate the effectiveness of our method in comparison with the High Efficiency Video Coding (HEVC) standard for representing multiview video views. The main idea in asymmetric multiview video coding is that lateral views are encoded to low qualities as a solution to reduce the size of the multiview data, encoding these views affects their qualities. In this section, we compare the visual quality of synthesised views by our method and views encoded using HEVC.

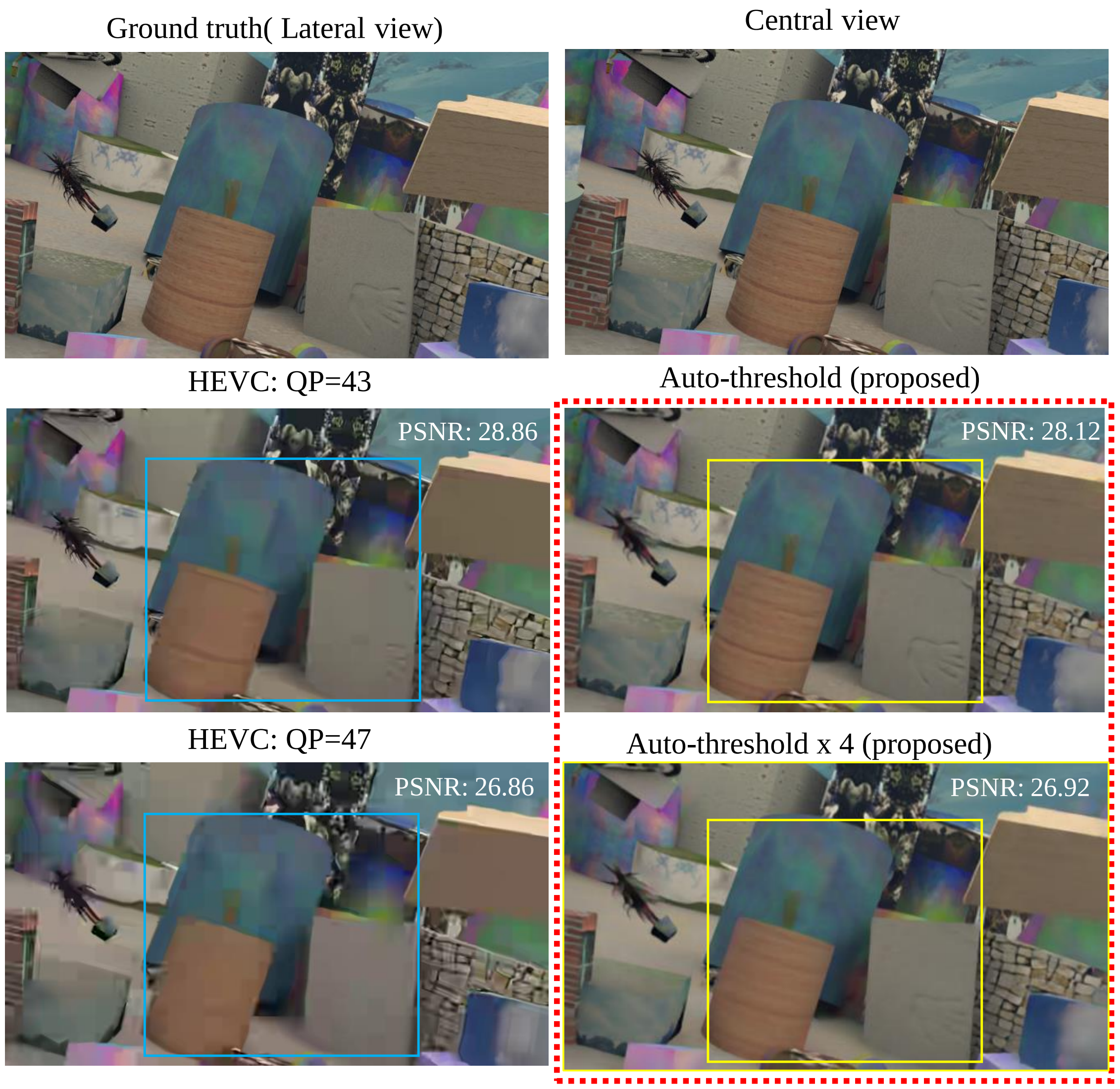
Ground truth(Lateral view)
Central view
HEVC: QP=43
PSNR: 28.86
Auto-threshold (proposed)
PSNR: 28.12
HEVC: QP=47
PSNR: 26.86
Auto-threshold x 4 (proposed)
PSNR: 26.92

Fig. 5.6: Examples of view synthesis from silhouette (proposed method) in the second row using different edge detector method's threshold values in comparison with HEVC encoder (in the first row) where the images were compressed using different QP values.

In the conducted experiments, HEVC encoder with $QP = \{50, 47, 43\}$ [116] are used to generate the compressed (low quality) views whose PSNR is in a range of $25 - 29$ dB for FlyingThings3D clean test set. In SVS, thresholds values = Auto-threshold $\times\{1, 2, 4\}$ are used to render synthesized views with equivalent PSNR. The visual results reported in Fig. 5.6 demonstrate the effectiveness of the proposed approach over HEVC. In fact it is possible to observe that for similar PSNR the proposed approach preserve the details and has clearer edges for the objects, e.g., texture inside yellow squares, while the encoded views using HEVC have lost details and texture for some objects, e.g., blue squares.

5.3.5 Object Place Editing

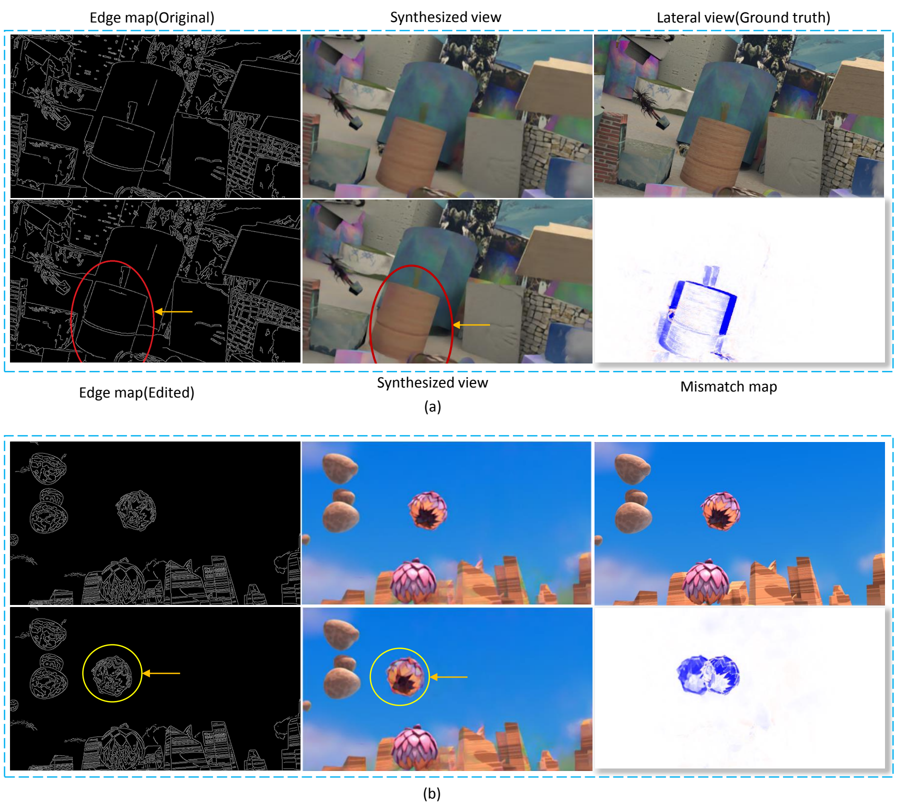

Fig. 5.7: Examples of view synthesis from silhouette before and after editing some objects place in the edges map.

To show that the edge map has a key component in the synthesizing process. We conducted an experiment by editing the generated edge map manually to change the place of the objects only in the edges map. The "manually edited object" result shows how the algorithm could fill the texture of the objects starting from their edges, even when it get placed in a different place. The displacement of the object is equivalent to having the same object at a different distances from the image plane with respect to the original setting of the testing data. In other words, the edge map provides the motion or displacement information of the objects in the scene between the central view and the lateral view as we can see in Fig. 5.7. In the first row we shifted the cylindrical object in the edge map to a new location to the left of the original position. The SVS uses the new edited edge map and generates the new view where the place of the cylindrical objected is located according to the edited edge map.

The mismatch map [1], with its dominant white color, shows that the proposed algorithm does not affect the filling of most of non-displaced objects. Similarly, the bottom row shows the synthesis quality when moving the silhouette of the bud to the left with respect to it original place. Also here the algorithm managed to fill the texture effectively.

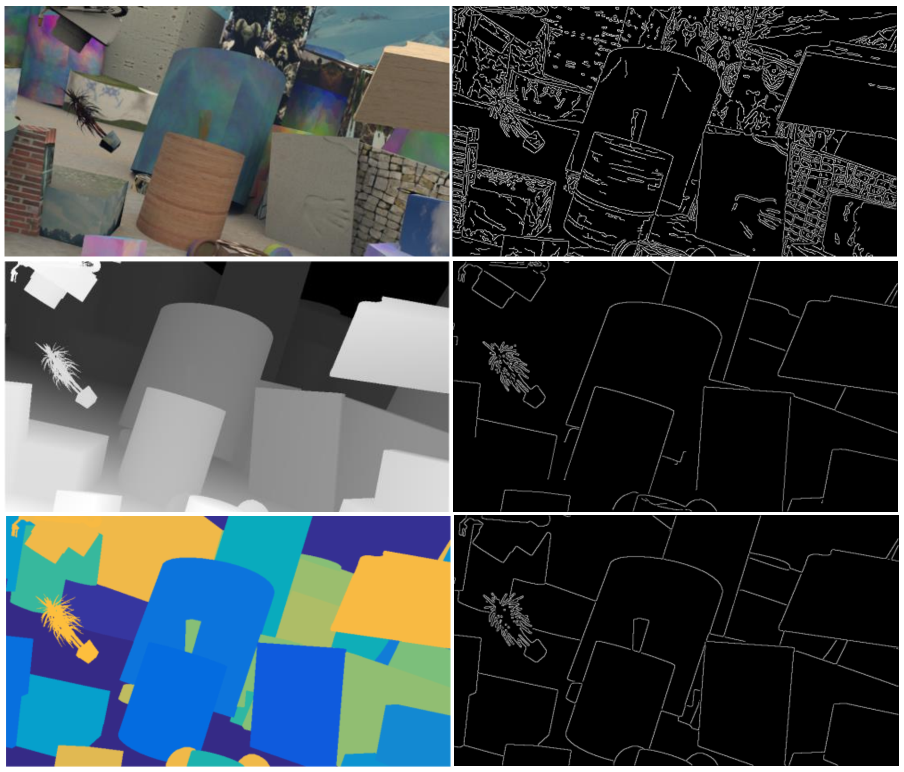

Fig. 5.8: Example of silhouette from texture image, disparity map and semantic segmentation map.

5.4 Conclusion

In this work we proposed silhouette-based view synthesis framework by representing lateral views using their edges, while dropping their texture content. These texture contents

[1]The mismatch map compares two error images. The first error image is the absolute error between the synthesis view from the non-edited silhouette and the ground-truth. Whereas, the second is the absolute error between the synthesis view from the edited silhouette and the ground-truth. Blue color indicate that the first error image has smaller values in comparison with the second error image, whereas, red areas indicate the opposite, and white indicate that both error images have almost the same error.

get synthesized by a CNN exploiting the edges and the information in the central view. The edges of the lateral views represent the location of the objects in their corresponding views. Moreover, in this work only information from neighbours views is exploited, however for multiview videos applications further performance improvement can be expected by exploiting temporal information from other frames. We believe that this is an interesting direction for future research. Furthermore, another direction for this research might be investigating the generation of edge map using semantic segmentation images or disparity maps at the decoder side, and example for generating the edge map using different representation for the same scene is depicted in Fig.5.8.

It is worth reporting that the proposed scheme in this section has led to the following publication:

1. Samer JAMMAL, Tammam TILLO, and Jimin XIAO, "View Synthesis from Silhouette Using Deep Convolutional Neural Network," In International Conference on Future Computer and Communication, 2019.

Chapter 6

Multiview Video Quality Enhancement without Depth Information

In the previous chapter silhouette-based view synthesis for multiview video using deep convolutional neural networks was presented. In this chapter, a new framework for multiview quality enhancement is introduced. The details of the proposed method are introduced in Section 6.2. In Section 6.3 subjective and objective evaluations of synthetic and real stereo datasets highlight the efficacy of the proposed merging scheme. Finally, we present our conclusions in Section 6.4.

6.1 Introduction

As it was presented in previous chapters, recording multiview video using multiple cameras produces vast amount of multiview video data. Asymmetric multiview video provides a solution for this problem. In asymmetric multiview video coding, the views are coded with different qualities. Based on the theory of binocular suppression [117], the perceived quality is close to that of the high-quality view if the lowest quality is above a certain level. However, the compressed view may incur severe quality degradation, particularly at very low bit-rate. Thus, it is necessary to enhance the visual quality of compressed views at the decoder side.

Over the past decade, there have been increasing interests in enhancing the visual quality of decoded images [118]. Deep learning approaches have been successfully applied in enhancing the visual quality of decoded images. Dong et al. [74] designed a four-layer CNN to learn an end-to-end mapping between low-resolution and high-resolution images, named ARCNN, for improving the quality of JPEG images. Afterwards, other deep networks [119, 120], were proposed to reduce the artifacts. However, for enhancing low-quality views in multiview systems, the similarities between low-quality and high-quality views should be exploited. These similarities can be classified into two types, inter-view similarity between adjacent camera views and temporal redundancy between successive frames of each video.

Recently, some works [67, 68, 69] exploited the inter-view correlation by projecting the high-quality view to the position of the low-quality one to render a high-quality virtual view using the DIBR method, that requires the depth map of the high-quality view. Then a network exploits (or in [68] a handcrafted method) the virtual view information to improve the low-quality view. These networks are able to learn the mapping process between the two views. However, using DIBR makes these methods computationally expensive. More importantly, these methods require high-quality and high-resolution image-depth pairs, which is not always available.

Based on the above consideration, we propose a new framework for MultiView quality Enhancement using a deep neural Networks approach (MVENet). The details in the low-quality view get synthesized using a convolutional neural network directly from the high-quality view. The used network takes advantage of the inter-view correlation and exploits the information of the high-quality view to recover the fine details of the compressed view. In the proposed multiview 3D video coding system, the encoder requires no modification, the used network is only employed at the decoder side. The MVENet is trained end-to-end and no post-processing phases are required. Experimental results on both computer graphic and real datasets demonstrate the effectiveness of the proposed approach with a PSNR gain of up to 2 dB over the low-quality compressed views using HEVC on the benchmark Cityscapes. More importantly, unlike previous works on multiview video enhancement, our method does not require any depth information during the enhancement process. Therefore, a stereo video could be a potential application for our approach, where normally no depth information is provided.

To summarize, this chapter MVENet framework. First, we introduce a quality enhancement framework for multiview 3D video, where a convolutional neural network is employed at the decoder side. The MVENet directly takes two inputs, low and high-quality views, and renders the enhanced view. Second, an end-to-end training method for the network is introduced using a large computer graphic dataset. Finally, extensive experiments on both computer graphic and real-world data demonstrate that the proposed approach can efficiently enhance the quality of the compressed views as compared with other state-of-the-art approaches.

6.2 Proposed Multiview Quality Enhancement Approach

The key principle of the presented framework is based on the fact that multiview data is intrinsically redundant. In fact, the semantic contents of different views are almost similar.

By taking this advantage, the proposed approach exploits the inter-view similarities to remove the artifacts and enhances the low-quality view using information of the high-quality view. To address this task, we introduce several modifications to the *DispNetC* network architecture compared to [20]. First, to meet the new task L_2 was used as loss function instead of $L1$. Second, since the original network outputs an image with quarter resolution of the input images, a $deconv$ layer was introduced to perform the upsampling to the same size of the original inputs. One key novelty of the proposed framework is that no depth information is required during the whole enhancement process, which was a major drawback in previous works where the depth map is needed. The details and the performance of the employed network will be discussed in the following subsections.

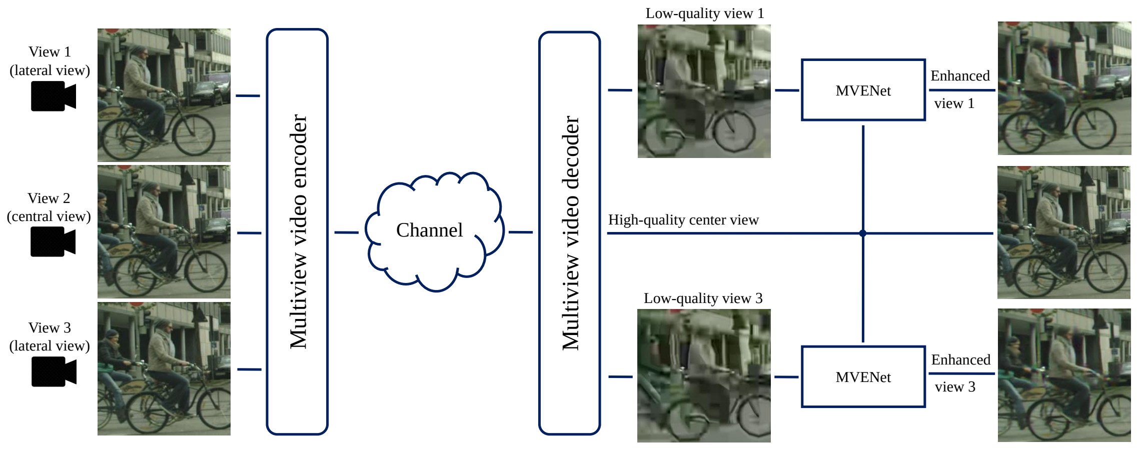

Fig. 6.1: The CNN-based framework for multiview video quality enhancement. In the proposed approach, the multiview encoder requires no modification. At the decoder side, the low-quality views will be enhanced by exploiting information from the high-quality ones.

6.2.1 Proposed Framework

Fig. 6.1 shows the framework of the proposed multiview video quality enhancement approach, along with the used network. In this framework, the center view is kept with high-quality while other views (lateral views) are compressed to low-quality. The multiview data are stored or transmitted through a communication channel to the receiver then to the decoder. After the decoding stage, the lateral views are recovered with the help of the high-quality center view using a deep network. The used network was trained end-to-end to learn the geometry mapping between the central view and lateral views, which in turn allows a consistent and realistic rendering for the lateral views.

6.2.2 Network Architecture

Convolutional neural networks, in general, are able to learn input-output relations with enough training data. We, therefore, propose to address the asymmetric multiview quality enhancement task using an end-to-end learning approach.

Fig.6.2 depicts the architecture of the proposed multiview video enhancement framework, with *DispNetC* [20] network as the main processing engine. Given a high-quality central view and a low-quality lateral view, the goal of the MVENet is to enhance the low-quality view by exploiting the high-frequency information and texture details from the high-quality view.

6.2.3 Training Procedure

Given N training inputs $\{HQ_j, LQ_j, GT_j\}_{j=1}^{N}$, where HQ_j is the high-quality central view, LQ_j is the low-quality lateral view, and the label GT_j is the ground truth lateral view. $\theta = \{\mathbf{W}, \mathbf{b}, \mathbf{a}\}$ denotes the network parameters controlling the forward process, the goal is to train the used network to recover and enhance the low-quality view. To achieve this task the network was trained by minimizing the $L2$ norm for the N training samples as follows:

$$L(\theta) = \frac{1}{N}\sum_{j=1}^{N} ||\hat{LQ}_j(\theta) - GT_j||^2, \tag{6.1}$$

where $\hat{LQ}_j(\theta)$ is the recovered view using network parameter θ and the inputs $\{HQ_j, LQ_j\}$.

6.3 Experimental Results

In this section, experimental results are presented to validate the effectiveness of the proposed MVENet framework. The compared benchmarks include the High Efficiency Video Coding (HEVC) standard [121], the JPEG image compression standard, and the latest quality enhancement methods, including SR-CNN [75], ARCNN [74], FastARCNN [76]. The proposed approach was evaluated on the benchmark dataset CityScapes [122] and FlyingThings3D synthetic dataset [20].

In the following part, we discuss the implementation details in section 6.3.1. We compare the performance of quality enhancement with HEVC in Section 6.3.2. Finally, we compare the proposed framework with state-of-the-art approaches in Section 6.3.3.

6.3.1 Implementation Details

The network is trained end-to-end using the FlyingThings3D (clean train set). The training set was generated as follows, one view (the right view) of each pair, j, in this dataset

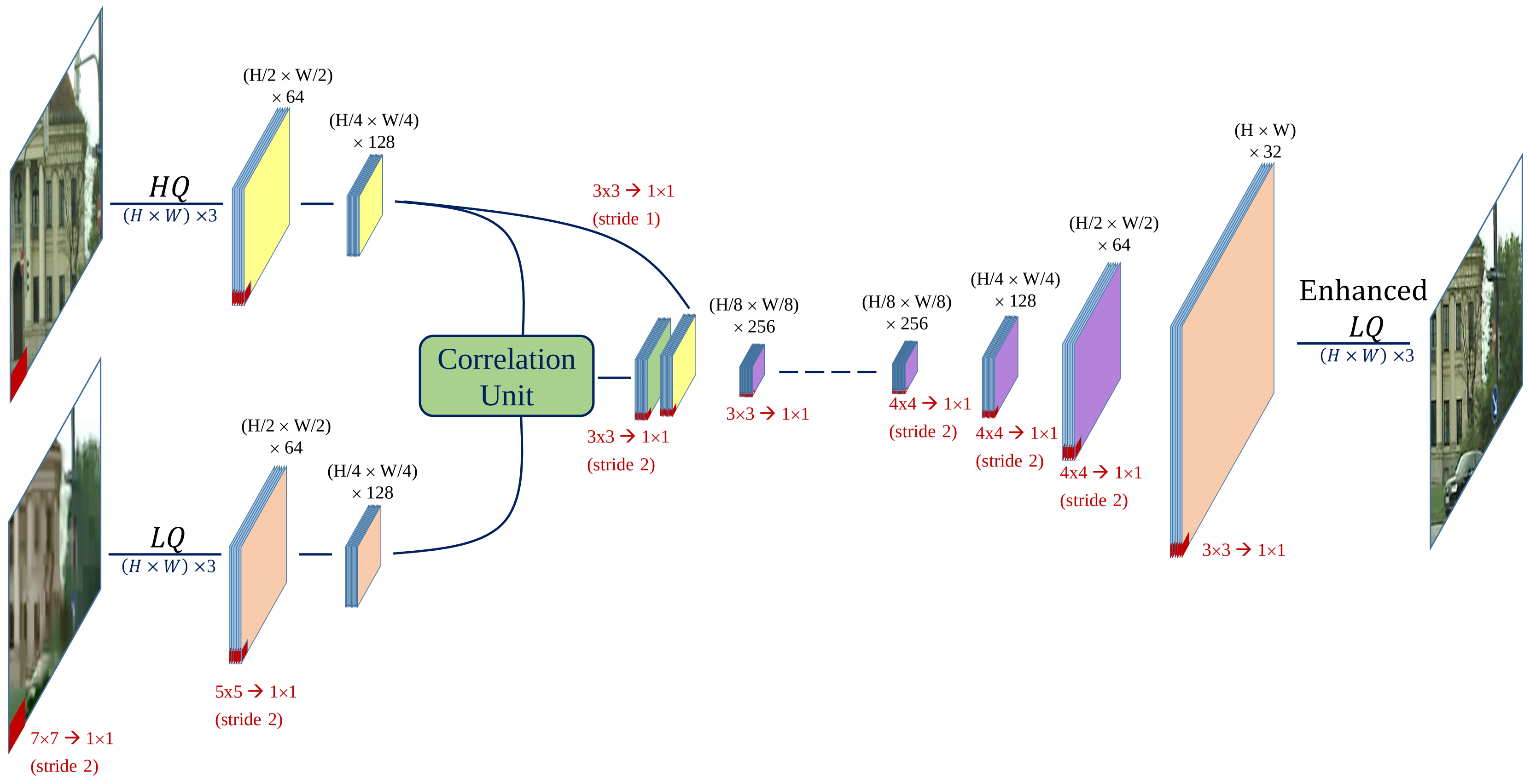

Fig. 6.2: The architecture of MVENet, *DispNetC* after modification.

was compressed to generate a low-quality lateral view, LQ_j, whereas, the other view (the left view) was kept uncompressed to provide the high-quality view, this view will be denoted by HQ_j. The target image, GT_j, which is used to evaluate the loss function and to train the network, for the pair j, is the original uncompressed right view. Initially, the training was performed using patches of size $768 \times 384 \times 3$ which were cropped from the original image of size $960 \times 540 \times 3$ from the FlyingThings3D training set. A batch size of 8 is used. Training the network with a batch size of 8 takes less than one day to run 80k iterations using one Titan X GPU.

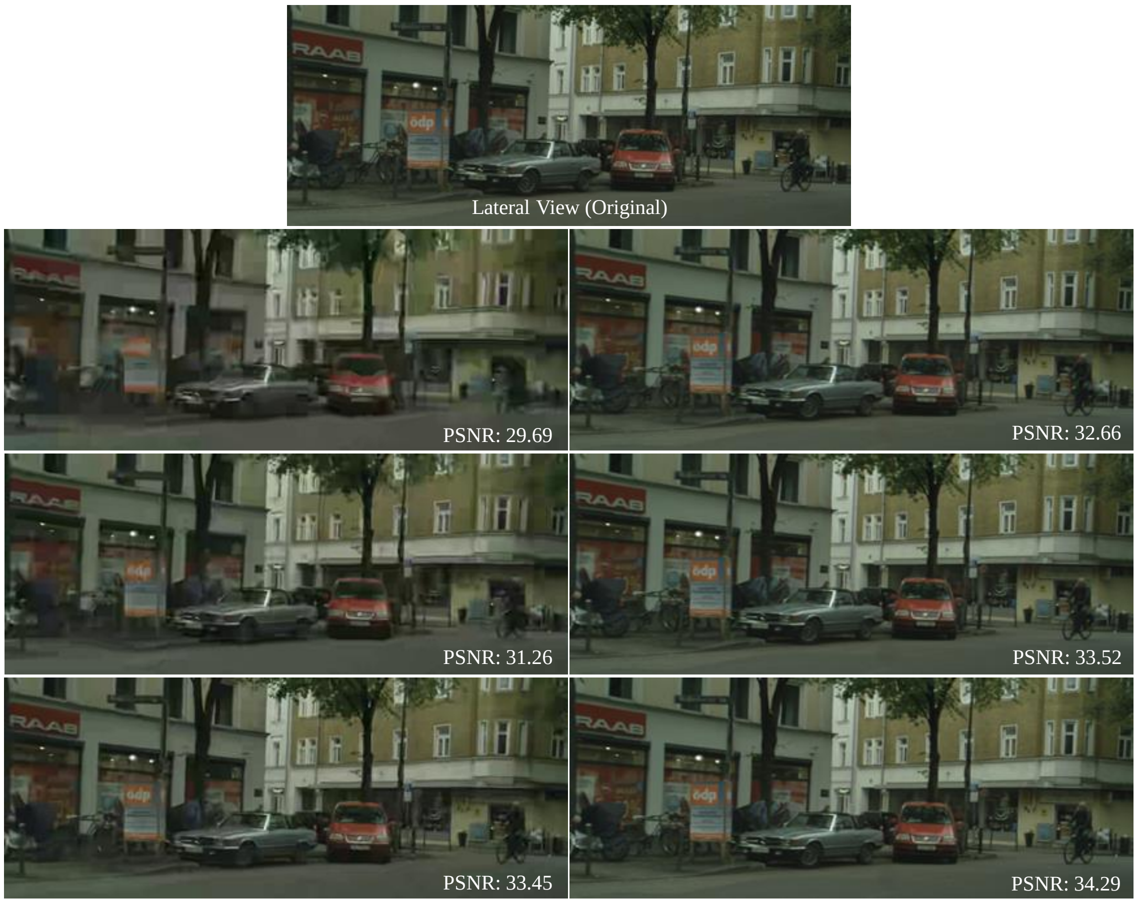

Fig. 6.3: Examples of HEVC multiview quality enhancement for various QP using the proposed framework on CityScapes Dataset. The image in the center of the first row: the original lateral view (i.e., the ground truth). The left column shows the compressed lateral image using HEVC; the right column shows the corresponding recovered image using MVENet. The second row to the fourth show the images for $QP_L = 50, 47$ and 43, respectively.

6.3.2 Comparison with HEVC Baseline

The HEVC standard has significantly improved video coding efficiency. HEVC is able to achieve a bit rate saving of 50% with similar subjective quality compared with the previous H.264/AVC standard. However, under very low bit rates, HEVC videos also incur artifacts, such as blocking artifacts, ringing effects, and blurring. Such artifacts may cause severe degradation of Quality of Experience (QoE) at the decoder side. In this subsection, we investigate the feasibility of highly compressed HEVC videos enhancement using MVENet.

In the conducted experiments, QP_L = $\{50, 47, 43\}$ are used to generate the highly compressed (low-quality) views whose PSNR is in a range of $29 - 34$ dB for CityScapes dataset and a range of $24 - 28$ dB for FlyingThings3D clean test set. In these experiments, the high-quality views are the original non-compressed views. The corresponding experimental results on both Cityscapes and FlyingThings3D datasets are reported in Table. 6.1. In this table, inaddtion to PSNR, the Structural SIMilarity (SSIM) metric, which is a method for measuring the similarity between two images, has been used. The results demonstrate the effectiveness of the proposed approach. A PSNR gain of 2.13 dB is obtained on the benchmark Cityscapes for $QP_L = 50$. We observed that the proposed approach achieves a larger gain at lower bit rates (larger QP). This is mainly because highly compressed views lose more details and the blocking artifacts are more severe. Consequently, the margin for quality enhancement is larger.

Dataset	**Method**	QP_L=50		QP_L=47		QP_L=43	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
CityScapes (test)	HEVC	29.56	0.81	31.05	0.84	33.11	0.88
	MVENet	**31.69**	**0.89**	**32.49**	**0.90**	**33.17**	**0.91**
	Gain	*2.13*	*0.08*	*1.44*	*0.06*	*0.06*	*0.03*
FlyingThings3D (test)	HEVC	24.63	0.65	25.94	0.70	27.81	0.76
	MVENet	**26.43**	**0.77**	**27.16**	**0.79**	**28.17**	**0.81**
	Gain	*1.80*	*0.12*	*1.18*	*0.9*	*0.36*	*0.50*

Table 6.1: Quality enhancement comparison of the proposed MVENet with the HEVC standard on both Cistscapes and FlythingThings3D datasets.

Besides the numeric values, some subjective results are reported in Fig.6.3. From this figure, it is possible to observe that the proposed approach guarantees much clearer edges and details than HEVC. More experiments were conducted using the stereoscopic 3D video sequences Balloons, Cafe and Kendo, and the results are reported in Fig. 6.4. These sequences differ greatly in content, motion, and displacement between views. The HEVC encoder with $QP_L = 50$ was used to encode the low-quality views of the three sequences. The MVENet model trained on Cityscapes dataset was used without any fine-tuning. From Fig. 6.4, it is possible to note that the encoded view of the Ballons video sequence, first row, has lost many details, e.g., the musical notes on the wall cannot be recognized after compression. However, the proposed MVENet successfully enhances the visual quality and recovers the musical notes on the wall, a zoom in images are shown in Fig. 6.4.

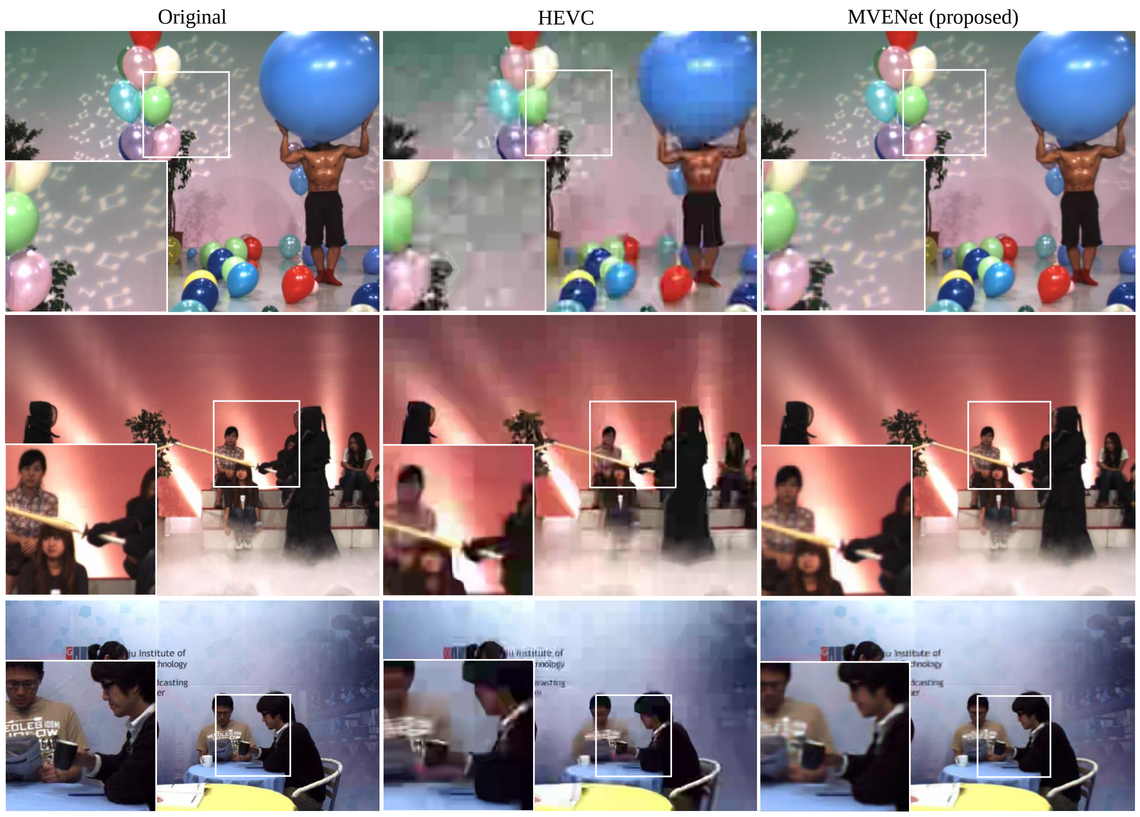

Fig. 6.4: Examples of HEVC multiview quality enhancement using the proposed MVENet framework on CityScapes Dataset. From left to right: original view frame, the encoded view using HEVC with QP=50, and the recovered view. From top to down the multiview sequences are Balloons, Kendo and Cafe.

In Table. 6.2, the robustness of the network against QP mismatch between the training QP, QP_L, and the testing QP, QP_{Test}, is assessed. The results are for both Cityscapes and FlyingThings datasets. As expected, the results show that each model is more suitable to enhance a low quality video sequence compressed with the same QP as the one used to train the network. Nevertheless, if there is a small mismatch between the training QP and testing QP, for example, if the gap is 3, the model can still work well without obvious performance degradation (less than 0.35dB).

In previous experiments, a high-quality view without any HEVC compression was used to demonstrate the effectiveness of the proposed framework. However, in practice, the high-quality views in a multiview video system are also compressed (normally with

Dataset		QP_L=50		QP_L=47		QP_L=43	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
CityScapes (test)	QP_{Test}=50	**31.69**	**0.89**	31.64	0.89	30.73	0.87
	QP_{Test}=47	32.14	0.90	**32.49**	**0.90**	32.17	0.89
	QP_{Test}=43	32.47	0.90	33.00	0.91	**33.17**	**0.91**
FlyingThings3D (test)	QP_{Test}=50	**26.43**	**0.77**	26.38	0.76	25.92	0.74
	QP_{Test}=47	27.08	0.79	**27.16**	**0.79**	27.13	0.78
	QP_{Test}=43	27.63	0.80	27.88	0.81	**28.17**	**0.81**

Table 6.2: Evaluation of the models on different QP factors on Cistscapes and FlythingThings3D datasets. In this table, the obtained model for each QP_L is tested with sequences coded with different QP values, QP_{Test}.

small QP factors, QP_H). Thus, in this part, we investigate the effect of using encoded views on the robustness of the reconstruction process. Hence, QP_H = $\{22, 27, 32, 37\}$ are used to generate the high-quality views, while QP_L = $\{50, 47, 43\}$ are used to generate low-quality images (high compression ratio). The experimental results on the two datasets are shown in Table 6.3. From these results, it is clear that using non-compressed images provides the best performance. However, using high-quality views shows that comparable results could be obtained. For example, using QP_L = 50 with QP_H = 22 results in a 0.07dB PSNR drop in comparison with the case when the high-quality view is not compressed at all. Whereas, when QP_L = 50 is used with $QP_H = 27$ the PSNR drop becomes 0.1dB. In summary, the proposed framework is able to enhance the low-quality views using the information from the encoded high-quality views. We also tested the proposed framework when JPEG standard is used as compression engine.

HEVC Compression		low-quality view					
		$QP_L = 50$		$QP_L = 47$		$QP_L = 43$	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
High-quality view	No compression	**31.69**	**0.89**	**32.49**	**0.90**	**33.17**	**0.91**
	QP_H=22	31.62	**0.89**	32.42	**0.90**	33.10	**0.91**
	QP_H=27	31.59	**0.89**	32.38	**0.90**	33.05	**0.91**
	QP_H=32	31.50	0.88	32.28	0.89	32.93	0.90
	QP_H=37	31.29	0.87	32.02	0.88	32.65	0.89

Table 6.3: Quality enhancement comparison of MVENet on Cistyscapes dataset using encoded high-quality views with various QP_H factors.

6.3.3 Comparison with JPEG Approaches

In this section, we compare the proposed method with ARCNN [74] and FastARCNN [76] networks on the benchmark dataset CityScapes [122] and FlyingThings3D synthetic dataset [20]. In these experiments, the JPEG standard is used for the generations of the low-quality images. In particular, JPEG quality $Q_L = \{1, 3, 5, 7\}$ are used to generate the highly compressed images whose PSNR quality is in the range of $27 - 32$dB for CityScapes dataset and $25 - 30$dB for FlyhtingThings3D dataset. The comparison on the two datasets are reported in Table 6.4 and 6.5.

We can perceive that MVENet outperforms the other two methods on both datasets by providing the highest PSNR and SSIM results. The PSNR gains are up to 3.74dB and 2.40dB over FastARCNN on CityScapes dataset for $Q_L = 3$ and 5, respectively. The PSNR gains are up to 2.42dB and 0.80dB over FastARCNN on FlyThings3D dataset for $Q_L = 3$ and 5, respectively. It's worth report that the input for ARCNN and FastARCNN networks is only the low-quality view and the output presents the enhanced view.

Method	Q_L=1		Q_L=3		Q_L=5		Q_L=7	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
JPEG	26.92	0.78	27.13	0.79	30.37	0.83	32.44	0.87
ARCNN [74]	27.60	0.80	27.63	0.81	31.46	0.87	33.97	0.91
FastARCNN [76]	27.62	0.80	27.90	0.82	31.70	0.88	34.20	0.91
MVENet (Proposed)	**31.29**	**0.89**	**31.64**	**0.90**	**34.10**	**0.92**	**34.40**	**0.93**

Table 6.4: Comparison with the state-of-the-art algorithms on Cistscapes dataset.

Method	Q_L=1		Q_L=3		Q_L=5		Q_L=7	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
JPEG	25.40	0.68	25.74	0.70	28.29	0.76	29.91	0.80
ARCNN [74]	26.08	0.71	26.50	0.73	29.61	0.81	31.50	0.85
FastARCNN [76]	26.23	0.71	26.67	0.73	29.75	0.81	31.67	0.86
MVENet (Proposed)	**28.68**	**0.79**	**29.09**	**0.81**	**30.55**	**0.84**	**31.72**	**0.86**

Table 6.5: Comparison with the state-of-the-art algorithms on FlythingThings3D dataset.

We also compare the subjective visual quality of the reconstructed images, as shown in Fig.6.5 on both datasets. It can be observed that the compressed image contains obvious blocking artifacts and many texture details of the objects have been blurred. The processed image by ARCNN reduces blocking artifacts, but most details and texture could not be recovered. While the images generated by MVENet suppresses most artifacts, and leads to better subjective visual quality.

Here, a set of experiments were conducted using $Q_H = \{80, 60, 40, 20\}$ to generate a high-quality compressed views, while $Q_L = \{1, 3, 5, 7\}$ are used to generate the low-quality images (i.e., highly compression ratio). The corresponding experimental results on the two datasets are reported in Table. 6.6. From these results, it can be observed that using non-compressed images provides best performance over the compressed ones. However,

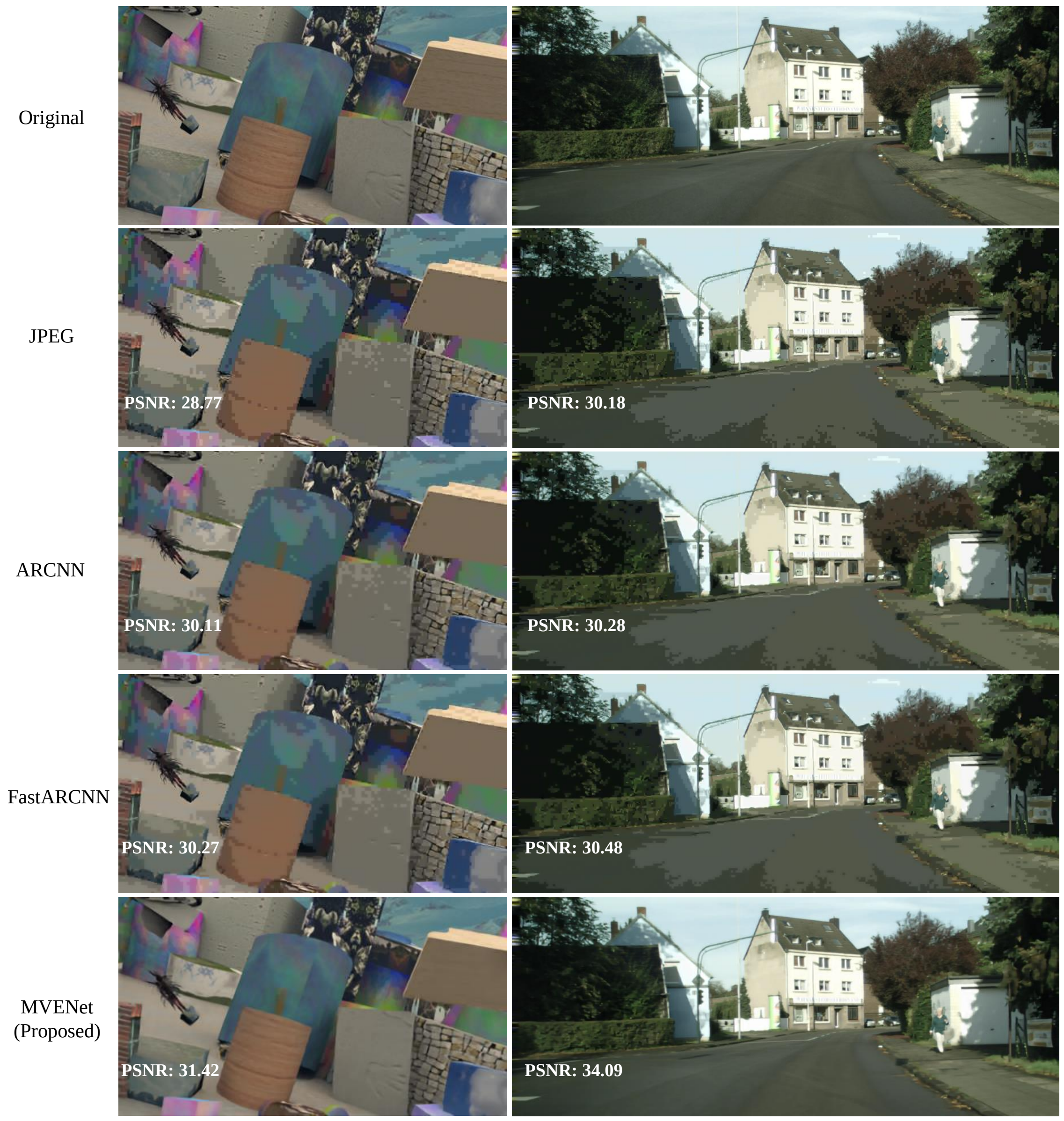

Fig. 6.5: Visual comparison results of the proposed CNN and other JPEG approaches on both FlyingThings3D and CityScapes datasets using JPEG as a compression engine.

using high-quality views compressed with high Q_H provides comparable results as well. For example, using high-quality views of $Q_H = 80$ and low-quality views of $Q_L = 5$ results in 0.01dB PSNR drop. For the case of using high-quality views of $Q_H = 40$ and low-quality views of $Q_L = 5$ the PSNR drop is 0.06dB. Consequently, the proposed framework is suitable for real scenarios where high-quality views are also compressed using a high Q_H factor.

JPEG Compression		Low-quality view					
		Q_L=3		Q_L=5		Q_L=7	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
High-quality view	No Compression	**31.64**	**0.90**	**34.10**	**0.92**	**34.40**	**0.93**
	Q_H=80	31.63	**0.90**	34.09	**0.92**	34.39	**0.93**
	Q_H=60	31.62	**0.90**	34.07	**0.92**	34.37	**0.93**
	Q_H=40	31.58	0.89	34.04	**0.92**	34.33	**0.93**
	Q_H=20	31.44	0.89	33.86	**0.92**	34.18	0.92

Table 6.6: Quality enhancement comparison of the proposed MVENet with the JPEG standard on CistyScapes datset after compressing the hight-quality view with various Q_H factors.

Finally, Table. 6.7 shows the running time comparison of ARCNN [74], FastARCNN [76] methods, along with MVENet. All baseline methods are obtained from the corresponding authors' implementation. It is observed that the overall running time of the three methods are comparable though MVENet leads to the best performance.

Method	**Runtime (second)**
ARCNN [74]	0.075
FastARCNN [76]	0.033
MVENet	0.060

Table 6.7: The the running time comparison of the MVENet with several state-of-the-art methods.

6.4 Conclusion

To address the challenging task of multiview video quality enhancement without depth information approach, we proposed a novel framework that can directly enhance low-quality views. In the proposed work, neither a depth map nor a projected virtual view is required in the enhancement process. The MVENet network improves the low-quality views by exploiting the information in the high-quality view to recover the details of the low-quality view. The experimental results have shown that the proposed MVENet outperforms the state-of-the-art methods on both HEVC and JPEG compressed images.

In this work, only information from neighboring views is exploited. We believe further performance improvement can be obtained by exploiting temporal information from other frames for multiview videos applications, which could be an interesting direction for future research.

It is worth reporting that the proposed scheme in this section has led to the following publication:

1. Samer JAMMAL, Tammam TILLO, and Jimin XIAO, "Multiview Video Quality Enhancement without Depth Information," In Signal Processing: Image Communication 75 (2019) 2231.

Chapter 7

Conclusions & Future Work

In this chapter, the final summary of this thesis will be presented, followed by some proposed work in relevant domains.

7.1 Summary

The video plus depth representation for multi-view video sequences enables functionalities like 3D television and free viewpoint video. The main benefit of using a multi-view video plus depth representation is that intermediate views can be easily rendered. However, the image quality of rendered arbitrary intermediate views depends on the accuracy of depth maps. Thus, the first purpose of this thesis was to develop original methods for estimating reliable and accurate disparity maps from stereo images using convolutional neural networks.

To take this task, we proposed a disparity estimation method based on a convolutional neural network with a multi-scale correlation layer employing several correlation kernels and different scales was proposed. We found that small kernels are suitable for the disparity estimation of small objects with fine details while larger scales are suitable for larger objects with uniform areas. The proposed model is able to capture the key stereo features of the two views and generate an accurate disparity map. However, when analyzing the network for wide disparity range, we find that the integrated network fails in estimating accurate disparities for both close foreground objects with large disparities and far background objects with small disparities; in the latter case, the details are lost. Consequently, to address the challenging task of wide range disparity estimation, we proposed a novel CNN-based framework that can accurately estimate a disparity map over wide disparity range for both close and far objects with large and small disparities. Firstly, the input stereo images are re-sampled to different resolutions, and then a deep CNN is used to predict several disparity maps from each pair of the re-sampled versions. High-resolution pairs allow the proper estimation of the disparities of far objects, whereas, low-resolution pairs are employed to estimate the disparities for close objects. Finally, two networks, MRDF-Conv and MRDF-Conv-Deconv, are introduced to merge the multiple estimations of the disparity maps to generate more accurate disparity that handles a large range of displacements while trying to keep the fine details. MRDF-Conv requires fewer parameters and costs less computational time than MRDF-Conv-Deconv. On the other hand, MRDF-Conv-Deconv retains the sharpness of the boundaries. Experiments performed on the Middlebury2014 and KITTI2015 benchmarks demonstrate the accuracy of the proposed methods.

In this thesis, we also proposed a silhouette-based view synthesis framework, which could be used as a method for multiview compression, by representing intermediate views using their edges, while dropping their texture content. These texture contents get synthesized by a CNN exploiting the edges and the information in the central view. The edges of the intermediate views represent the location of the objects in their corresponding views. The experimental results have shown that the proposed MVENet outperforms the state-of-the-art methods on both HEVC and JPEG compressed images.

Finally, to address the challenging task of multiview video quality enhancement, we proposed a novel framework that can directly enhance low-quality views. In the proposed work, neither a depth map nor a projected virtual view is required in the enhancement process. The MVENet network improves the low-quality views by searching similarities between the high-quality and low-quality views and exploiting high-frequency related information in the high-quality view to recover the details of the low-quality view. The experimental results have shown that the proposed MVENet outperforms the state-of-the-art methods on both HEVC and JPEG compressed images.

7.2 Perspectives for future work

At the time of concluding this manuscript, several interesting perspectives can be proposed to further continue the work done in this thesis. The primary points concern the multi-resolution based hierarchical estimation of disparity maps framework. It was observed that using multiple resolution stereo images to generate multiple disparities maps then merging these maps into a single representation disparity map improves the accuracy and provide more details for far objects. Furthermore, interesting future research for this work might be to use a single end-to-end convolutional neural network, where the concept of using multiple resolutions could be implemented in features domain. Then an end-to-end CNN that takes a pair of stereo images and multi-scales of features during the rendering process could be devised. Moreover, in silhouette-based view synthesis framework, only information from neighbors views is exploited, however for multiview videos applications further performance improvement could be expected by exploiting temporal information from other frames. We believe this is an interesting direction for future research. Finally, in the multiview video quality enhancement without depth information approach, the proposed framework improves the low-quality views by exploiting only information from neighboring views. However, using temporal information from other frames could result in better accuracy and more accurate details.

Appendix: A list of Publications

Here is a brief list of my research publications during my Ph.D. studies:

- **Journal Papers**:

    1. Samer JAMMAL, Tammam TILLO, and Jimin XIAO, "Multiview Video Quality Enhancement without Depth Information," In Signal Processing: Image Communication 75 (2019) 2231.
    2. Samer JAMMAL, Tammam TILLO, and Jimin XIAO, "Multi-Resolution Disparity Estimations Fusion Using Convolutional Neural Networks," IEEE transactions on multimedia, 2019 (submitted, under revision).

- **Conference Papers**:

    1. Samer JAMMAL, Tammam TILLO, and Jimin XIAO, "View Synthesis from Silhouette Using Deep Convolutional Neural Network," In International Conference on Future Computer and Communication, 2019.
    2. Samer JAMMAL, Tammam TILLO, and Jimin XIAO, "Multi-resolution for disparity estimation with convolutional neural networks," In Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Dec 2017, pp.17561761.
    3. Samer JAMMAL, Tammam TILLO, and Jimin XIAO, "Disparity Estimation Using Convolutional Neural Networks with Multi-scale Correlation," In International Conference on Neural Information Processing (ICONIP), October 2017.

Reference

[1] Y. Mori, N. Fukushima, T. Yendo, T. Fujii, and M. Tanimoto, “View generation with 3D warping using depth information for FTV,” *Image Communication*, vol. 24, no. 1-2, pp. 65–72, Jan. 2009.

[2] L. Zhang and W. J. Tam, “Stereoscopic image generation based on depth images for 3D TV,” *IEEE Transactions on Broadcasting*, vol. 51, no. 2, pp. 191–199, Jun. 2005.

[3] ITU-T and ISO/IEC JTC 1, Generic coding of moving pictures and associated audio information - Part 2: Video, ITU-T recommendation H.262 and ISO/IEC 13818-2 (MPEG-2 Video), 1994.

[4] ITU-T and ISO/IEC JTC1, Final draft amendment 3, Amendment 3 to ITU-T Recommendation H.262 and ISO/IEC 13818-2 (MPEG-2 Video), ISO/IEC JTC 1/SC 29/WG 11 (MPEG) Doc. N1366, Sep. 1996.

[5] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, “Overview of the H.264/AVC video coding standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[6] A. Vetro, P. Pandit, H. Kimata, A. Smolicand, and Y.-K. Wang, “Joint draft 8 of multiview video coding, Hannover, Germany, joint video team (JVT) Doc,” *IEEE Transactions on Circuits and Systems for Video Technology*, no. JVT-AB204, Jul. 2008.

[7] A. Vetro, T. Wiegand, and G. J. Sullivan, “Overview of the stereo and multiview video coding extensions of the H.264/MPEG-4 AVC standard,” *Proceedings of the IEEE*, vol. 99, no. 4, pp. 626–642, Apr. 2011.

[8] K. Muller, H. Schwarz, D. Marpe, C. Bartnik, S. Bosse, H. Brust, T. Hinz, H. Lakshman, P. Merkle, F. H. Rhee, G. Tech, M. Winken, and T. Wiegand, “3D High-efficiency video coding for multi-view video and depth data,” *Transactions Image Processing*, vol. 22, no. 9, pp. 3366–3378, Sep. 2013.

[9] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, “Multi-view video plus depth representation and coding,” in *IEEE International Conference on Image Processing*, vol. 1, Sep. 2007, pp. I – 201.

[10] Y. Chen, M. M. Hannuksela, T. Suzuki, and S. Hattori, “Overview of the MVC+D 3D video coding standard,” *J. Vis. Comun. Image Represent.*, vol. 25, no. 4, pp. 679–688, May 2014.

[11] Y. Chen, X. Zhao, L. Zhang, and J. Kang, “Multiview and 3D video compression using neighboring block based disparity vectors,” *IEEE Transactions on Multimedia*, vol. 18, no. 4, pp. 576–589, Apr. 2016.

[12] F. Shao, G. Jiang, W. Lin, M. Yu, and Q. Dai, “Joint bit allocation and rate control for coding multi-view video plus depth based 3D video,” *IEEE Transactions on Multimedia*, vol. 15, no. 8, pp. 1843–1854, Dec. 2013.

[13] H. Yuan, S. Kwong, X. Wang, W. Gao, and Y. Zhang, “Rate distortion optimized inter-view frame level bit allocation method for mv-hevc,” *IEEE Transactions on Multimedia*, vol. 17, no. 12, pp. 2134–2146, Dec. 2015.

[14] C. Fehn, “Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV,” in *Proceedings of SPIE*, vol. 5291, 2004.

[15] C. Fehn, “A 3D-TV system based on video plus depth information,” in *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers*, vol. 2, Nov. 2003, pp. 1529–1533.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, 2012.

[17] C. Szegedy, A. Toshev, and D. Erhan, “Deep neural networks for object detection,” in *Advances in Neural Information Processing Systems*, 2013, pp. 2553–2561.

[18] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 3431–3440.

[19] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, “Deeper depth prediction with fully convolutional residual networks,” *Computing Research Repository*, 2016.

[20] N. Mayer, E. Ilg, P. Husser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, “A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 4040–4048.

[21] Y. Lin and J. Wu, “Quality assessment of stereoscopic 3D image compression by binocular integration behaviors,” *IEEE Transactions on Image Processing*, vol. 23, no. 4, pp. 1527–1542, Apr. 2014.

[22] F. Shao, G. Jiang, M. Yu, K. Chen, and Y. Ho, “Asymmetric coding of multi-view video plus depth based 3-D video for view rendering,” *IEEE Transactions on Multimedia*, vol. 14, no. 1, pp. 157–167, Feb. 2012.

[23] G. Saygili, C. G. Gurler, and A. M. Tekalp, “Evaluation of asymmetric stereo video coding and rate scaling for adaptive 3d video streaming,” *IEEE Transactions on Broadcasting*, vol. 57, no. 2, pp. 593–601, 2011.

[24] R. Zone, *Stereoscopic Cinema and the Origins of 3-D Film*, ser. JSTOR EBA. University Press of Kentucky, 2014.

[25] N. A. Dodgson, “Autostereoscopic 3D displays,” *Computer*, vol. 38, no. 8, pp. 31–36, Aug 2005.

[26] C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen, “On benchmarking camera calibration and multi-view stereo for high resolution imagery,” in *IEEE Conference on Computer Vision and Pattern Recognition*, Jun 2008, pp. 1–8.

[27] H. Aghajan and A. Cavallaro, *Multi-Camera Networks: Principles and Applications*. Orlando, FL, USA: Academic Press, Inc., 2009.

[28] J. C. Yang, M. Everett, C. Buehler, and L. McMillan, “A real-time distributed light field camera,” in *Proceedings of the 13th Eurographics Workshop on Rendering*, ser. EGRW ’02. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2002, pp. 77–86.

[29] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, “Efficient prediction structures for multiview video coding,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 11, pp. 1461–1473, Nov. 2007.

[30] G. Tech, Y. Chen, K. Mller, J. Ohm, A. Vetro, and Y. Wang, “Overview of the multiview and 3D extensions of high efficiency video coding,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 1, pp. 35–49, Jan. 2016.

[31] J. Batlle, E. Mouaddib, and J. Salvi, "Recent progress in coded structured light as a technique to solve the correspondence problem: a survey," 1998.

[32] J. Salvi, S. Fernandez, T. Pribanic, and X. Llado, "A state of the art in structured light patterns for surface profilometry," *Pattern Recognition*, vol. 43, no. 8, pp. 2666–2680, Aug. 2010.

[33] J. Salvi, J. Pags, and J. Batlle, "Pattern codification strategies in structured light systems," *Pattern Recognition*, vol. 37, pp. 827–849, 2004.

[34] J. Salvi, J. Batlle, and E. Mouaddib, "A robust-coded pattern projection for dynamic 3D scene measurement," *Pattern Recognition*, vol. 19, no. 11, pp. 1055–1065, Sep. 1998.

[35] F. Shuai, Q. Tong, C. Yang, and C. Fengyun, "Depth recovery from a single defocused image based on depth locally consistency," in *Proceedings of the Fifth International Conference on Internet Multimedia Computing and Service*, ser. ICIMCS '13. ACM, 2013, pp. 56–61.

[36] Y. Y. Schechner and N. Kiryati, "Depth from defocus vs. stereo: How different really are they?" *Int. J. Comput. Vision*, vol. 39, no. 2, pp. 141–162, Sep. 2000.

[37] M. Subbarao and J.-K. Tyan, "The optimal focus measure for passive autofocusing and depth-from-focus," in *Proceedings of SPIE conference on Videometrics IV*, 1995, pp. 89–99.

[38] B.-Z. Jing and D. S. Yeung, "Recovering depth from images using adaptive depth from focus." in *ICMLC*. IEEE, 2012, pp. 1205–1211.

[39] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, vol. 47, no. 1-3, pp. 7–42, Apr. 2002.

[40] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*. IEEE, 2011, pp. 127–136.

[41] S. Foix, G. Alenya, and C. Torras, "Lock-in Time-of-Flight (ToF) cameras: A survey," *IEEE Sensors Journal*, vol. 11, no. 9, pp. 1917–1926, Sep 2011.

[42] P. Didyk, P. Sitthi-Amorn, W. Freeman, F. Durand, and W. Matusik, “Joint view expansion and filtering for automultiscopic 3D displays,” *ACM Transactions on Graphics*, vol. 32, no. 6, pp. 221:1–221:8, Nov. 2013.

[43] D. Ji, J. Kwon, M. McFarland, and S. Savarese, “Deep view morphing,” *Computing Research Repository*, 2017.

[44] N. K. Kalantari, T.-C. Wang, and R. Ramamoorthi, “Learning-based view synthesis for light field cameras,” *ACM Transactions on Graphics*, vol. 35, no. 6, 2016.

[45] P. Kellnhofer, P. Didyk, S.-P. Wang, P. Sitthi-Amorn, W. Freeman, F. Durand, and W. Matusik, “3DTV at home: Eulerian-lagrangian stereo-to-multiview conversion,” *ACM Transactions on Graphics*, vol. 36, no. 4, 2017.

[46] J. Flynn, I. Neulander, J. Philbin, and N. Snavely, “Deepstereo: Learning to predict new views from the world’s imagery,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[47] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, “Spatial transformer networks,” in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS’15. Cambridge, MA, USA: MIT Press, 2015, pp. 2017–2025.

[48] J. Xie, R. B. Girshick, and A. Farhadi, “Deep3d: Fully automatic 2D-to-3D video conversion with deep convolutional neural networks,” *Computing Research Repository*, vol. abs/1604.03650, 2016.

[49] G. Wu, M. Zhao, L. Wang, Q. Dai, T. Chai, and Y. Liu, “Light field reconstruction using deep convolutional network on EPI,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[50] P. P. Srinivasan, T. Wang, A. Sreelal, R. Ramamoorthi, and R. Ng, “Learning to synthesize a 4D RGBD light field from a single image,” *Computing Research Repository*, vol. abs/1708.03292, 2017.

[51] D. Eigen, C. Puhrsch, and R. Fergus, “Depth map prediction from a single image using a multi-scale deep network,” pp. 2366–2374, 2014.

[52] S. Zagoruyko and N. Komodakis, “Learning to compare image patches via convolutional neural networks,” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4353–4361, 2015.

[53] J. Zbontar and Y. LeCun, “Computing the stereo matching cost with a convolutional neural network,” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[54] J. bontar and Y. Lecun, “Stereo matching by training a convolutional neural network to compare image patches,” *Journal of Machine Learning Research*, vol. 17, 4 2016.

[55] R. Garg, V. K. B. G, and I. D. Reid, “Unsupervised CNN for single view depth estimation: Geometry to the rescue,” *Computing Research Repository*, 2016.

[56] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[57] An image of a traffic sign is filtered by 4 55 convolutional kernels. nvidia developer.

[58] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” in *Proceedings of the 22Nd ACM International Conference on Multimedia*, ser. MM ’14, 2014, pp. 675–678.

[59] Y. Xia and M. S. Kamel, “Novel cooperative neural fusion algorithms for image restoration and image fusion,” *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 367–381, Feb. 2007.

[60] H. Hariharan, A. Koschan, B. Abidi, A. Gribok, and M. Abidi, “Fusion of visible and infrared images using empirical mode decomposition to improve face recognition,” in *International Conference on Image Processing*, Oct. 2006, pp. 2049–2052.

[61] W. Yang and J. Liu, “Research and development of medical image fusion,” in *2013 IEEE International Conference on Medical Imaging Physics and Engineering*, Oct 2013, pp. 307–309.

[62] H. Hariharan, A. Koschan, and M. Abidi, “Multifocus image fusion by establishing focal connectivity,” in *IEEE International Conference on Image Processing*, vol. 3, 2007, pp. III – 321–III – 324.

[63] A. P. James and B. V. Dasarathy, “Medical image fusion: A survey of the state of the art,” *Information Fusion*, vol. 19, pp. 4 – 19, 2014, special Issue on Information Fusion in Medical Image Computing and Systems.

[64] L. Tao and Z. Qian, “An improved medical image fusion algorithm based on wavelet transform,” in *Seventh International Conference on Natural Computation*, vol. 1, Jul. 2011, pp. 76–78.

[65] Q. P. Zhang, M. Liang, and W. C. Sun, “Medical diagnostic image fusion based on feature mapping wavelet neural networks,” in *Third International Conference on Image and Graphics (ICIG’04)*, Dec 2004, pp. 51–54.

[66] Y. Liu, X. Chen, H. Peng, and Z. Wang, “Multi-focus image fusion with a deep convolutional neural network,” *Information Fusion*, vol. 36, pp. 191 – 207, 2017.

[67] Y. Xie, J. Xiao, T. Tillo, Y. Wei, and Y. Zhao, “3D video super-resolution using fully convolutional neural networks,” in *IEEE International Conference on Multimedia and Expo (ICME)*, Jul 2016, pp. 1–6.

[68] Z. Jin, T. Tillo, C. Yao, J. Xiao, and Y. Zhao, “Virtual-view-assisted video super-resolution and enhancement,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 3, pp. 467–478, Mar 2016.

[69] L. Yu, T. Tillo, J. Xiao, and M. Grangetto, “Convolutional neural network for intermediate view enhancement in multiview streaming,” *IEEE Transactions on Multimedia*, vol. 20, no. 1, pp. 15–28, Jan 2018.

[70] A. W.-C. Liew and H. Yan, “Blocking artifacts suppression in block-coded images using overcomplete wavelet representation,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 4, pp. 450–461, April 2004.

[71] A. Foi, V. Katkovnik, and K. Egiazarian, “Pointwise shape-adaptive DCT for high-quality denoising and deblocking of grayscale and color images,” *IEEE Transactions on Image Processing*, vol. 16, no. 5, pp. 1395–1411, May 2007.

[72] S. Shen, X. Fang, and C. Wang, “Adaptive non-local means filtering for image deblocking,” in *International Congress on Image and Signal Processing*, vol. 2, Oct 2011, pp. 656–659.

[73] H. Chang, M. K. Ng, and T. Zeng, “Reducing artifacts in JPEG decompression via a learned dictionary,” *IEEE Transactions on Signal Processing*, vol. 62, no. 3, pp. 718–728, Feb 2014.

[74] C. Dong, Y. Deng, C. Change Loy, and X. Tang, “Compression artifacts reduction by a deep convolutional network,” *In Proceedings of International Conference on Computer Vision (ICCV)*, pp. 145–167, 2015.

[75] C. Dong, C. C. Loy, K. He, and X. Tang, “Image super-resolution using deep convolutional networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, Feb 2016.

[76] K. Yu, C. Dong, C. C. Loy, and X. Tang, "Deep convolution networks for compression artifacts reduction," *Computing Research Repository*, 2016.

[77] T. Wang, M. Chen, and H. Chao, "A novel deep learning-based method of improving coding efficiency from the decoder-end for HEVC," in *Data Compression Conference (DCC)*, Apr 2017, pp. 410–419.

[78] R. Yang, M. Xu, and Z. Wang, "Decoder-side hevc quality enhancement with scalable convolutional neural network," in *IEEE International Conference on Multimedia and Expo (ICME)*, Jul 2017, pp. 817–822.

[79] E. M. Hung, C. Dorea, D. C. Garcia, and R. L. de Queiroz, "Transform-domain super-resolution for multiview images using depth information," in *European Signal Processing Conference*, Aug 2011, pp. 398–401.

[80] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," *Computing Research Repository*, 2014.

[81] B. Li, C. Shen, Y. Dai, A. van den Hengel, and M. He, "Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015.

[82] F. Liu, C. Shen, and G. Lin, "Deep convolutional neural fields for depth estimation from a single image," *Computing Research Repository*, 2014.

[83] "Svm model monocular depth estimation method based on infrared images," Oct. 2015, cN Patent 102,708,569.

[84] V. Jain, J. F. Murray, F. Roth, S. C. Turaga, V. P. Zhigulin, K. L. Briggman, M. Helmstaedter, W. Denk, and H. S. Seung, "Supervised learning of image restoration with convolutional networks," in *IEEE 11th International Conference on Computer Vision, ICCV 2007, Rio de Janeiro, Brazil, October 14-20, 2007*, 2007, pp. 1–8.

[85] "Blender." [Online]. Available: http://download.blender.org/release/

[86] Camera calibration toolbox for matlab,.

[87] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *Computing Research Repository*, 2014.

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

[89] H. Hirschmuller, “Stereo processing by semiglobal matching and mutual information,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, Feb. 2008.

[90] L. De-Maeztu, S. Mattoccia, A. Villanueva, and R. Cabeza, “Linear stereo matching,” in *IEEE International Conference on Computer Vision (ICCV)*, Nov 2011.

[91] Y. Geng, “Local stereo matching based on information entropy of image,” *3D Research*, vol. 7, no. 3, pp. 103:1–103:10, Sep. 2016.

[92] N. Y. C. Chang, T. H. Tsai, B. H. Hsu, Y. C. Chen, and T. S. Chang, “Algorithm and architecture of disparity estimation with mini-census adaptive support weight,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 6, pp. 792–805, Jun 2010.

[93] Y. C. Tseng and T. S. Chang, “Architecture design of belief propagation for real-time disparity estimation,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 11, pp. 1555–1564, Nov 2010.

[94] J. Sun, N.-N. Zheng, and H.-Y. Shum, “Stereo matching using belief propagation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 787–800, Jul 2003.

[95] L. Wang and R. Yang, “Global stereo matching leveraged by sparse ground control points,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2011, pp. 3033–3040.

[96] Y. Zhan, Y. Gu, K. Huang, C. Zhang, and K. Hu, “Accurate image-guided stereo matching with efficient matching cost and disparity refinement,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1632–1645, Sep 2016.

[97] W. Luo, A. Schwing, and R. Urtasun, “Efficient deep learning for stereo matching,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016.

[98] A. Seki and M. Pollefeys, “Patch based confidence prediction for dense disparity map,” in *Proceedings of the British Machine Vision Conference*, S.l., 2016, pp. 23–.

[99] S. Pertuz, D. Puig, M. A. Garcia, and A. Fusiello, "Generation of all-in-focus images by noise-robust selective fusion of limited depth-of-field images," *IEEE Transactions on Image Processing*, vol. 22, no. 3, pp. 1242–1251, March 2013.

[100] R. Shen, I. Cheng, J. Shi, and A. Basu, "Generalized random walks for fusion of multi-exposure images," *IEEE Transactions on Image Processing*, vol. 20, no. 12, pp. 3634–3646, Dec 2011.

[101] Y. Zeng, "Generation of all-focus images and depth-adjustable images based on pixel blurriness," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, Oct 2013, pp. 1–9.

[102] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization." *Computing Research Repository*, vol. abs/1412.6980, 2014.

[103] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *Computing Research Repository*, 2015.

[104] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conference on Computer Vision (ECCV)*, ser. Part IV, LNCS 7577. Springer-Verlag, Oct. 2012, pp. 611–625.

[105] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling, *High-Resolution Stereo Datasets with Subpixel-Accurate Ground Truth*. Cham: Springer International Publishing, 2014, pp. 31–42.

[106] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[107] D. Murray and J. J. Little, "Using real-time stereo vision for mobile robot navigation," *Autonomous Robots*, vol. 8, no. 2, pp. 161–171, Apr. 2000.

[108] H. Hirschm, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, pp. 328–341, 2007.

[109] L. Chen, J. Chen, and L. Fan, "A convolutional neural networks based full density stereo matching framework." Computer Vision and Image Understanding, 2015.

[110] W. Luo, A. G. Schwing, and R. Urtasun, "Efficient deep learning for stereo matching," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016, pp. 5695–5703.

[111] M. Neoral and J. ochman, “Object scene flow with temporal consistency.” Computer Vision Winter Workshop (CVWW), 2017.

[112] Y. L. A. Li, D. Chen and Z. Yuan, “Coordinating multiple disparity proposals for stereo computation,” in *IEEE Computer Vision and Pattern Recognition (CVPR)*, Mar 2016.

[113] T. Taniai, S. Sinha, and Y. Sato, “Fast multi-frame stereo scene flow with motion segmentation,” in *IEEE Computer Vision and Pattern Recognition (CVPR)*, Mar 2017.

[114] K.-J. Oh, S. Yea, and Y.-S. Ho, “Hole filling method using depth based in-painting for view synthesis in free viewpoint television and 3-D video,” in *Proceedings of the 27th Conference on Picture Coding Symposium*, ser. PCS’09, 2009, pp. 233–236.

[115] J. Canny, “A computational approach to edge detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, Nov 1986.

[116] F. Battisti, M. Carli, P. Le Callet, and P. Paudyal, “Toward the assessment of quality of experience for asymmetric encoding in immersive media,” *IEEE Transactions on Broadcasting*, vol. 64, no. 2, pp. 392–406, June 2018.

[117] R. Blake, “A neural theory of binocular rivalry,” *Psychological Review*, pp. 145–167, 1989.

[118] H. Chang, M. K. Ng, and T. Zeng, “Reducing artifacts in JPEG decompression via a learned dictionary,” vol. 62, no. 3, pp. 718–728, Feb. 2014.

[119] C. Dong, C. C. Loy, K. He, and X. Tang, “Image super-resolution using deep convolutional networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, Feb. 2016.

[120] D. Chao, L. C. Change, H. Kaiming, and T. Xiaoou, “Learning a deep convolutional network for image super-resolution,” in *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2014, pp. 184–199.

[121] G. J. Sullivan, J. Ohm, W. Han, and T. Wiegand, “Overview of the high efficiency video coding (hevc) standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[122] 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,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.