Selected Topics in Image Super-Resolution: Guided Image and Learning Representation

Wentian Zhou

Dissertation submitted to the
Staler College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy
in
Electrical Engineering

Daryl Reynolds, Ph.D., Chair
Xin Li, Ph.D.
Matthew Valenti, Ph.D.
Natalia Schmid, Ph.D.
Hailin Li, Ph.D.

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2017

Keywords: Digital Imagery, Super-resolution, Interpolation, Depth map, Group Sparse Coding, Neural Network, Deep Learning

Copyright 2017 Wentian Zhou
Abstract

Selected Topics in Image Super-Resolution: Guided Image and Learning Representation

Wentian Zhou

Obtaining high-resolution images is a fundamental challenge for many vision related tasks. It is highly desirable to develop effective and efficient super-resolution algorithms to enhance the spatial resolution of acquired images. However, estimating high-resolution images is an ill-posed problem where a large set of possible solutions exist. To make this inverse problem more tractable, it is often necessary to constrain the targeted high-resolution image with a priori information (i.e., in model-based approaches) or to assume the availability of additional training data (i.e., in learning-based approaches). In this dissertation, we propose to (i) use simultaneous-sparse-coding and machine learning techniques to incorporate side information from a guided image and (ii) to study the role of data representations (e.g., spatial vs. frequency) in competing learning based approaches toward image super-resolution.

We first study the problem of depth map super-resolution, a special class of images captured by 3D depth sensing devices. The generated depth map is often in low resolution (LR) but paired with a high-resolution (HR) color image of the same scene. In one approach, simultaneous-sparse-coding (SSC) is employed to model the relationship between LR and HR depth maps. We extend the nonlocal similarity of the original SSC model from the guided color image to depth map in order to determine the unknown patch clusters. In another approach, we directly learn the relationship between LR depth map and HR color image using a recently developed deep neural network scheme.

The next problem in our study is the resolution-enhancement of natural images. Such a problem can be formulated in two different ways: interpolation and super-resolution (SR). We have developed a structural aware interpolation scheme capable of recognizing structural variations and inferring the unknown pixels based on the structural prior. Moreover, we propose a super-resolution scheme utilizing the local self-similarity property. Our approach overcomes the limitation of integer scaling factors and generates aesthetically pleasing SR results. Extensive evaluations have shown that the performance of the proposed methods achieve and exceed current state-of-the-art algorithms.

Overall, we propose to tackle three sub-problems of the image super-resolution problem from two different aspects: guided image and learning representation. To utilize the guided image, two competing approaches (model-based vs. learning-based) are presented to incorporate the side information. Additionally, spatial domain learning and progressively high-frequency learning frameworks are compared to tackle the image interpolation and image SR problems respectively.
To my parents and my wife
Acknowledgments

I would like to thank many people for contributing to and supporting my Ph.D. studies. It would not be possible to complete my Ph.D. study without their contributions.

First, I would like to express my deepest appreciation to my advisors Prof. Daryl Reynolds and Prof. Xin Li for their guidance and support throughout my Ph.D. study. It is their patience and expertise that have shaped me from an uncertain student into a professional researcher. I am very grateful for their teachings, not just about the aspects of signal processing, but for many valuable pieces of advice that will benefit me for life.

Secondly, I would like to thank all of the faculty and staff members in the Lane Department of Electrical Engineering and Computer Science. For over nine years of study at this department, including during my undergraduate studies, they have always been there to provide support. I would not reach this step without their dedication to teaching and their genuine helping hearts.

Thirdly, I am highly grateful to my colleagues Dr. Yu Zhu, Dr. Xiaoxia Sun, Yuan Li, Usha Alluri, Macro Piccirilli, Kenneth Hite, Yufeng Yu and Linhao Li for their kind help and assistance during my Ph.D. study.

Fourthly, I am thankful to my committee members, Prof. Natalia Schmid, Prof. Matthew Valenti, and Prof. Hailin Li. Thank you all for teaching me, encouraging me, and correcting me throughout my study. Special thanks to Prof. Yangqiu Song for guiding and directing me into the world of Machine Learning.

Lastly, I would like to thank my parents, for everything they have done to support me. I would also like to thank my aunt’s family for their encouragement and support. A big thank to my beloved wife, Shu Cao, for being always beside me. I could not complete this dissertation without their understanding, support, and love.
## Contents

Abstract ii

Acknowledgments iv

List of Figures vii

List of Tables ix

Nomenclature xi

1 Introduction 1
   1.1 Literature Review .................................................. 4
   1.2 Research Approach and Contributions .............................. 8
   1.3 Thesis Organization .................................................. 11

2 Background 13
   2.1 Introduction .......................................................... 13
   2.2 Sparse Representation ............................................... 16
      2.2.1 Sparse coding .................................................... 16
      2.2.2 Simultaneous Sparse Coding ................................... 19
   2.3 Deep Neural Networks ............................................... 20
      2.3.1 Supervised Learning .............................................. 20
      2.3.2 Neural Network .................................................. 22
      2.3.3 Optimization and Backpropagation ............................. 25

3 Image assisted upsampling of Depth Map via Non-local similarity 31
   3.1 Introduction .......................................................... 31
   3.2 Algorithm Development .............................................. 34
      3.2.1 Depth Map Registration ......................................... 34
      3.2.2 Dictionary Learning .............................................. 34
      3.2.3 Spatially Adaptive Iterative Singular-Value Thresholding ... 36
   3.3 Experiment Result .................................................... 39
      3.3.1 Evaluations using the Middlebury stereo dataset ............ 39
      3.3.2 Evaluation with synthetic noise ............................... 40
   3.4 Summary .............................................................. 42
# List of Figures

2.1 Low-resolution image acquisition model. ........................................ 14
2.2 Image sparse decomposition illustration. ....................................... 18
2.3 (a) sparsity VS (b) group sparsity .............................................. 20
2.4 Illustration for supervised learning. ............................................ 22
2.5 A diagram of biologically inspired model for a single neuron. Input $y_i$ are multiplied by weight $w_i$. results are accumulated in the cell then propagate to the next neurons if it surpass the activation threshold. .................. 23
2.6 An illustration for a three-layer full-connected network. .................. 27
2.7 An illustration of a convolutional layer. ...................................... 28
2.8 An illustration of a three layer convolutional neural network. .......... 29
2.9 An illustration of gradient back-propagation. ................................. 30
3.1 An example of upsampling on a low resolution depth map. ................. 33
3.2 Comparison of k-means clustering between depth map and color image of middlebury Books. ................................................................. 35
3.3 Sample process of finding a dictionary. ......................................... 36
3.4 The numerical (PSNR) results of iteratively upsampled depth map Art. Patch size and number of similar patches are varied to indicate the effectiveness of regular nonlocal parameters. ................................. 40
3.5 Subjective quality comparison of ×4 upsampling of middlebury Moebius dataset with added AWGN noise. ........................................... 43
3.6 Error map comparison of Books with upsampling factor of four between our algorithm and nonlocal means upsampling approach by Park et al. ...... 44
3.7 Enlarged comparison results of polygon inside Moebius from Fig. 3.5. .... 45
3.8 Subjective quality comparison of ×4 upsampling of Art with added AWGN noise. ................................................................. 46
4.1 Comparison of a typical depth edge with its counterparts. .................. 49
4.2 DEN+CBPN network illustrations. ................................................. 51
4.3 Examples of testing images, and their corrupted low-resolution counterparts. ................................................................. 57
4.4 Subjective quality comparison of ×8 upsampling of Art with added noise. 58
4.5 Subjective quality comparison of ×8 upsampling of Books with added noise. 59
4.6 Subjective quality comparison of ×8 upsampling of Moebius with added noise. 60
5.1 A formation of acquiring LR pixels from a HR image by direct downsampling. The solid dots are the missing HR pixels and the circles are the LR pixels. Blank dots and gray dots separate HR pixels geometrically of which 4 neighbors are associate with blank dots at diagonal direction and 2 neighbors are associate with gray dots at vertical or horizontal direction.

5.2 Illustration of interpolate HR pixels by (5.1).

5.3 Geometry interpretation of how LR pixels contribute to the estimation of HR pixels.

5.4 Architecture of our proposed Deep Interpolation Network. This network is trained with an end-to-end fashion. The output of this network is the three HR pixels.

5.5 (a) Run time comparison of state-of-the-art methods on Set5, (b) Run time comparison of state-of-the-art methods on Set14. PSNRs shown are averaged PSNR of each algorithm.

5.6 Interpolation results on test image baby.

5.7 Interpolation results on test image butterfly.

5.8 Interpolation results on test image flowers.

6.1 Illustration of model performance when the quality of the input LR image varies.

6.2 Illustration of Deep Progressive Zooming Network.

6.3 Illustration of Deep Progressive Zooming Network extended to larger scaling factors.

6.4 Convergence curve Comparison.

6.5 Digital zooming result of Baby with a scaling factor of two.

6.6 Digital zooming result of Lenna with a scaling factor of three.

6.7 Examples of learned kernels.

6.8 Enlarged examples of learned kernels.

6.9 Visual Comparison for ×2 SR on image baby of Set5.

6.10 Visual Comparison for ×2 SR on image barbara of Set14.

6.11 Visual Comparison for ×3 SR on image butterfly of Set14.

6.12 Visual Comparison for ×3 SR on image comic of Set14.
List of Tables

3.1 Objective quality comparison without noise on Middlebury dataset. The performance is measured as PSNR(dB) for two different factors ($\times 2, \times 4$). . . . . . . 41
3.2 Objective quality comparison results with noise. The performance is measured as PSNR(dB) for two different factors ($\times 2, \times 4$). . . . . . . . . . . . . . . . . 42
4.1 Quantitative evaluation when input low-resolution depth maps are clean. We compare DEN and DEN + CBPN with several state-of-the-art methods. The mean-square-error (MSE) results on Middlebury data sets are compared. The best result is bolded and the second best is underlined. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
4.2 Quantitative evaluation when input low-resolution depth maps are corrupted by noise. We compare DEN and DEN + CBPN with several state-of-the-art methods. The mean-square-error (MSE) results on Middlebury data sets are compared. The best result is bolded and the second best is underlined. . . . . . . . . . . . . 54
5.1 Quantitative evaluation on Set5 dataset. The mean-square-error (PSNR) results are presented. The best result is in bold and the second best is underlined. . . . . . . 72
5.2 Quantitative evaluation on Set14 dataset. The mean-square-error (PSNR) results are presented. The best result is in bold and the second best is underlined. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 72
6.1 Quantitative evaluation on Set5 dataset with a scaling factor of two. The mean-square-error (PSNR) results are presented. The best result is in bold and the second best is underlined. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 97
6.2 Quantitative evaluation on Set14 dataset with a scaling factor two. The mean-square-error (PSNR) results are presented. The best result is in bold and the second best is underlined. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 98
6.3 Quantitative evaluation on Set5 dataset with a scaling factor of two. The structure similarity index (SSIM) results are presented. The best result is in bold and the second best is underlined. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 99
6.4 Quantitative evaluation on Set14 dataset with a scaling factor of two. The structure similarity index (SSIM) results are presented. The best result is in bold and the second best is underlined. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 100
6.5 Quantitative evaluation on Set5 dataset with a scaling factor of three. The mean-square-error (PSNR) results are presented. The best result is in bold and the second best is underlined. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 100
6.6 Quantitative evaluation on \textit{Set14} dataset with a scaling factor three. The mean-square-error (\textbf{PSNR}) results are presented. The best result is in bold and the second best is underlined.  

6.7 Quantitative evaluation on \textit{Set5} dataset with a scaling factor of three. The structure similarity index (\textbf{SSIM}) results are presented. The best result is in bold and the second best is underlined.  

6.8 Quantitative evaluation on \textit{Set14} dataset with a scaling factor of three. The structure similarity index (\textbf{SSIM}) results are presented. The best result is in bold and the second best is underlined.
Nomenclature

A+ Adjusted Anchored Neighborhood Regression
ANR Anchored Neighborhood Regression
AWGN Additive White Gaussian Noise
BM3D Block-Matching and 3D Filtering
BSD Berkeley Segmentation Data-set
CBPN Color-Based Prediction Network
CNN Convolutional Neural Networks
DEN Depth Enhancement Network
DIN Deep Interpolation Network
DNC Deep Network Cascade
DPZN Deep Progressive Zooming Network
FCN Full-Connected Network
FCNN Fully Convolutional Neural Network
GTH Ground-Truth
HD High-Definition
HR High-Resolution
JBF Joint Bilateral Filtering
KNN  K-Nearest Neighbors
LapSRN  Deep Laplacian Pyramid Network
LMMSE  Linear Minimum Mean Square Error Estimator
LR  Low-Resolution
MISR  Multi-Image Super-Resolution
MRF  Markov Random Field
MRI  Magnetic Resonance Imaging
MSE  Mean-Square-Error
NARM  Non-local Auto-regressive Modeling
NE  Neighbor Embedding
NEDI  New Edge Directed Interpolation
NLSS  Non-Local Self-Similarity
NN  Neural Networks
PReLU  Parametric Rectified Linear Unit
PSF  Point Spread Function
PSNR  Peak Noise-to-Signal Ratio
ReLU  Rectified Linear Unit
RGB  Red-Green-Blue
SAI  Soft-decision Adaptive Interpolation
SAIST  Spatially Adaptive Iterative Singular-Value Thresholding
SCN  Sparse Coding Network
ScSR  Sparse Coding based Super-Resolution
SelfExSR  Self-Exemplar based Super-Resolution

SGD  Stochastic Gradient Descent

SISR  Single-Image Super-Resolution

SR  Super-Resolution

SRCNN  Super-Resolution via Convolutional Neural Network

SSC  Simultaneous Sparse Coding

SSIM  Structure Similarity Index

SVD  Singular Value Decomposition

TOF  Time-of-Flight

VDSR  Very Deep Super-Resolution
Chapter 1

Introduction

Digital images, an extension of human memory, is one of the most important information carriers for human activities. They play critical roles in many day-to-day applications, from online social networking to commercial advertising to medical imaging. Constrained by the physical characteristics of the digital sensor, e.g. size and density, the resolution of the captured image is limited. In some applications, e.g. Magnetic-Resonance-Imaging (MRI) that help doctors make correct diagnoses, the limited resolution has become a barrier for fast and accurate analysis. Therefore, it is highly desirable to breach the resolution limitation and acquire accurate high-resolution (HR) digital images.

One promising method is to use post-processing algorithms to reach the required resolution. The major advantage of choosing this software approach is that it costs less than upgrading hardware and the existing camera systems can be still utilized. Over the past four decades, many pioneers have developed various algorithms to improve the quality of reconstructed images. The common goal of these techniques is to provide finer details than the given low-resolution (LR) image by increasing the number of pixels per unit space. To achieve aesthetically pleasing images, preserving details (i.e. edges and textures) becomes critical in the super-resolution process.

In the early stage of imaging research, numerous methods including multiple-image fusion (MISR) were employed to enhance structural details. These algorithms fuse multiple LR images obtained from the same scene into one HR image. Aside from their performance, these methods often lead to additional challenges such as image registration and limited scaling
factors. The penalties paid for ignoring or assuming perfect alignment include aliasing of the reconstructed image and impracticality of deployment.

Recently, single image super-resolution (SISR) has gained more attention in the literature due to its versatility, and it reduces many practical constraints related to MISR. As indicated by its name, the essential difference between SISR and MISR is the number of inputs needed for the system. Since only one input LR image is required, the problem becomes under-determined/ill-posed. Although there is no unique solution to the ill-posed system, the reconstructed HR image can be regularized by a variety of priors. Meanwhile, various SISR models and algorithms (e.g. spatial domain methods, frequency domain methods, and sparse representations) have been proposed to incorporate and utilize these priors.

As more priors are proposed for the general SISR, specific super-resolution problems must be handled individually. In this dissertation, we focus on three sub-problems of super-resolution: depth map super-resolution, image interpolation, and natural image super-resolution. The depth map is a special class of images captured by dedicated distance measuring devices. In comparison with conventional natural images, the value of pixels stored in the depth map represent the physical distance between the camera and the scene; they are represented by a single-channel image. Because of the physical limitation of these dedicated devices, the resolution is poor, and the captured depth maps are often noisy. Additionally, image interpolation aims to increase image resolution as well. Different from conventional super-resolution, image interpolation only involves upsampling the LR image; it usually assumes that LR images are directly down-sampled from the HR image or that it is an incomplete version of the HR image. Therefore, image interpolation not only requires recovering the missing high-frequency components, but it also needs to remove the aliasing artifacts created by direct down-sampling. The third sub-problem that we study is the typical natural image super-resolution problem which aims to reconstruct HR images from their LR counterparts. The natural image capturing process is often modeled as sampling and quantizing processes of the real analog world. As a result, HR images obtained with a higher sampling rate preserves more details than LR images. Therefore, recovering HR details while increasing the number of pixels becomes the fundamental problem of natural image SR.
When countering these problems, there are a few questions that naturally arise. Ideally, as the knowledge about HR images accumulates, more accurate HR images can be estimated. Therefore, can we impose any additional information regarding HR images besides LR query images? For example, the output resolution of depth cameras is low, but modern stereo cameras produce higher resolution color images. With properly designed equipment and algorithms, the outputs should significantly advance the field of depth perception. Accordingly, models designed to characterize the relationship between LR and HR images play a significant role during super-resolution process. Thus, how can we find a suitable model for each application? Recent advances in Neural Networks (e.g., Alphago) have demonstrated the success of modeling certain processes by learning from themselves. It is intuitive to follow this philosophy and discover the natural models induced by pairs of LR and HR images.

This thesis tackles the aforementioned sub-problems of image super-resolution from guided image and learning representation points of view. Guided image refers to the set of image processing problems where additional guidance is available. Despite the common priors, such as piece-wise constant pixel values and sharp edges that represent distance discontinuity associated within the depth map, the corresponding HR image provides more information, e.g. structural details in higher resolution. In particular, we are interested in properly utilizing these priors and reconstructing the high-quality depth map under the guidance of an HR color image. Differing from image modeling, learning representation refers to the methods that exploit the rich inventory of existing data. In this dissertation, we are interested in utilizing machine learning techniques to describe the relationship between LR depth maps and HR depth maps under color guidance. Moreover, we aim to learn a structure-aware model that is capable of recognizing pixels variations and countering the aliasing artifacts for the problem of image interpolation. Also, we aim to build a learning framework for the natural image super-resolution problem that is not only capable of generating HR images with integer scaling factors, but also producing HR images with non-integer scaling factors.

In summary, we introduce new angles of handling depth map super-resolution and image interpolation problems efficiently and effectively. Our proposed algorithms have achieved comparable and often superior performance to current state-of-the-art algorithms. In the
next section, we identify the literature works associated with depth resolution, image interpolation, and natural image super-resolution, and outline our contributions.

1.1 Literature Review

Image super-resolution has been an active research topic over the past four decades. There are many sub-problems related to image super-resolution when increasing the spatial resolution of images, including Depth map super-resolution, image interpolation, and natural image super-resolution are typical examples. Numerous image models and algorithms related to these problems have been extensively studied. We shall start by overviewing some of these models and algorithms.

The depth map is a basic data carrier for various vision-related applications. The goal of super-resolution is to assist these applications by revealing the physical details at a higher resolution. Common super-resolution techniques can be roughly categorized into two different types: multiple depth maps fusion and image-assisted super-resolution. Similar to multiple images fusion, depth maps fusion merges multiple misaligned depth maps into a single high-resolution map, and therefore existing image fusion techniques can be utilized. An energy minimization framework [1] was developed to fuse multiple depth maps captured by the same sensor but with slightly different viewpoints. Based on this work, A probabilistic multi-scan alignment algorithm [2] was proposed to merge a larger set of depth maps with larger displacement. A tracking and mapping system was developed [3] to refine HR depth maps from coarse to fine detail using real-time depth data collected by a Microsoft Kinect [4].

With the success of SISR, more researchers started to focus on the single depth map super-resolution problem. Since the depth sensor often produces a considerably lower resolution map than a regular stereo camera, it is intuitive that high-resolution auxiliary stereo images of the same scene can provide additional cues to ease the super-resolution process. Homemade depth-stereo rigs [5, 6] was created to collect paired depth map and stereo images. An HR depth map was then generated with data fusion techniques. The basic assumption behind the fusion approach is that homogeneous texture regions of natural images tend to appear
on the same depth layer; the discontinuities of textures are more likely to be edges (depth variation) on the depth map [7, 8].

The Joint Bilateral Filtering (JBF) approach [9] was developed to jointly filter a depth cost volume under the guidance of an HR color image; the algorithm acquires a better quality depth map by iteratively enhancing the depth map via the filtering process. An extension of this work [10] was proposed using a high-end stereo camera to detect discontinuity better and to avoid over-smoothing depth boundaries. Following the JBF formulation, a noise-aware bilateral filter [11] was developed that blended the results of the JBF and the standard bilateral upsampling processes to achieve depth maps with better quality. The JBF framework was also introduced into depth map super-resolution in video sequences [12] with extra time stamp information to maintain temporal coherence.

A Markov Random Fields (MRF) [13] formulation for depth map super-resolution was proposed; in the formulation, a data term was calculated from the depth map, and a smoothness term was weighted according to texture derivatives. Based on this formulation, a more complex approach was proposed [14]; in this work, a combination of different weighting terms of a least-squares optimization including non-local means, image gradient, edge salience map, and segmentation was used to regularize the outputs.

Besides JBF and MRF, other fusion-based frameworks were also introduced, including trilateral filtering [15], energy optimization by belief propagation [16], and the Bayesian framework [17]. The current state-of-the-art algorithm [18] was proposed to fuse an HR color image and LR depth map using the anisotropic total generalized variation formulation. A first order primal due algorithm was employed to optimize the HR outputs.

LR images in super-resolution algorithms usually modeled as the down-sampled version of its HR counterpart after blurring. Except for a special case when the blurring kernel is the Dirac Delta function, the image super-resolution problem becomes the image interpolation problem. As a result, conventional super-resolution algorithms become less effective as the high-frequency aliasing artifacts persist and restoring local structures becomes critical to obtain high fidelity HR images.

Edge-directed interpolation (NEDI) [19] was introduced to preserve the local structure details. The key contribution of this work was the ability to estimate HR patch covariances
A Soft-decision Adaptive Interpolation (SAI) [22] with the same philosophy of preserving spatial coherence was developed. In this work, block-based estimation instead of pixel estimation was proposed. In addition, SAI introduced additional constraints on the estimated unknown pixels. Such a formulation enforced the statistic coherence within a local region and therefore achieved better performance. Based on SAI, Robust Soft-decision Adaptive Interpolation (RSAI) [23,24] and Bilateral Soft-Decision Adaptive Interpolation (BSAI) [25] were proposed to replace the least-square estimator with a weighted least-square estimator to improved accuracy and robustness, and with bilateral filtering to reduce the computation complexity, respectively.

As the effectiveness of sparse representation was discovered in many other applications such as compressive sensing and image denoising, it is natural to incorporate such a powerful image modeling strategy for image interpolation problems. However, conventional Sparse Representation Modeling (SRM) methods become less effective because the data fidelity term would fail to estimate the missing pixels. To overcome this challenge, the state-of-the-art Non-local Autoregressive Modeling (NARM) based sparse representation method [26] was proposed. In this work, the concept of NARM was proposed to model a given pixel as a linear combination of its non-local neighboring pixels. By adopting and embedding NARM into SRM, the image interpolation problem was converted into a general SRM based image restoration problem which satisfied all the constraints related to SRM.

Natural image super-resolution is one of the fundamental challenges in image SR. It has very high theoretical and practical value in both academic and industrial environments. Due to the popularization of high-definition (HD) displays (e.g. 4K display) and because most digitally acquired media uses LR equipment, natural image super-resolution is in huge
demands nowadays to allow better utilization of these HD displays. Inherited from image SR, the natural image super-resolution problem is a highly ill-posed problem, and it has been an active research area over the past few decades. Recently, there have been a few breakthroughs which have greatly advanced the technologies in this area.

A sparse representation based super-resolution technique [27] was proposed; it is the first work that attempts to adopt a dictionary learning scheme in natural image super-resolution. This work utilizes the fact that image patches can be represented with a sparse linear combination of atoms from a well-defined over-complete dictionary. Under the context of natural image super-resolution, a pair of HR-LR dictionaries were jointly learned to enforce the similarity between resolutions. When increasing the resolution of an image, an LR sparse code was first learned by projecting the LR image onto the LR dictionary. The outputted HR image was then generated by applying the learned sparse code with the HR dictionary. Due to the intrinsic property of the local sparse modeling, this approach was naturally robust to noise.

A different approach to modeling natural images was proposed [28]; in this formulation, natural images were modeled as a composition of two separate signal bands: high-frequency and low-frequency. The base idea of this work was to utilize the local self-similarity approach toward upsampling for generating high-frequency details, and subsequently adding the details back to the upscaled low-frequency signal. The key observation of this approach was that the local self-similarity assumption holds better for small scaling factors in which large scaling factors (e.g. \( \times 2, \times 3 \)) were decomposed into non-integer small scaling factors (e.g. \( \times 1.25, \times 1.5 \)). Such a simple modeling strategy dramatically increased HR image quality and produced aesthetically pleasing results.

Following the scheme of the dictionary learning approach, an extended work [29] that adopted nearest neighbor embedding was developed. In this work, neighbor embedding (NE) approaches were also adopted and utilized. NE assumed that LR and HR image pairs lied on a low-dimensional nonlinear manifold with similar local geometries. Therefore, instead of using the global dictionaries learned from LR-HR pairs, it considered local neighborhoods of a given window. In such a case, dictionaries were re-calibrated to adapt local properties and produced better results. This work was extended in [30] with improved computation speed
and quality.

With the recent advances in neural networks, deep learning has become a popular tool to tackle many computer vision problems including natural image super-resolution. A convolutional neural network based approach was proposed in [31]. In this method, three convolutional layers were introduced to model the relationship between different resolutions. The network is pre-trained with synthetic LR-HR image pairs and applied to estimate HR images directly. This work has shown the possibilities and the capabilities of utilizing deep learning techniques in low-level computer vision fields and inspired numerous researchers later on.

In summary, Depth map super-resolution image interpolation and natural image super-resolution are both estimation/regression problems. Current state-of-the-art methods for depth map super-resolution fuses stereo and depth structures together to achieve accurate estimation. However, questions such as “What to fuse?” “When to fuse?”, moreover, “What are the other methodologies of imposing stereo image?”, are not very well addressed in the literature. By studying the literature for image interpolation problems, we can observe the trend of imposing additional knowledge to constrain the estimated unknown pixels, i.e. a single local neighboring constraint (NEDI) to a multiple constraints based autoregressive model (SAI) to a non-local constraint-based model (NARM). However, these methods only constrain the output by the query image alone when more knowledge could be learned from other query images. Similarly, to solve natural image super-resolution problem, more and more attentions have been shifted to learning-based approaches. Instead of manually deriving or composing a model to reveal the fundamental relationship between resolutions, the ability to learn a model from existing dataset has become valuable. Meanwhile, this approach also faces other challenges since the best learning framework/structure remains unknown.

1.2 Research Approach and Contributions

In this section, we briefly introduce our research approaches to solve the depth map super-resolution and image interpolation problems. We mainly attack these problems from guided image and learning representation points of view. Background knowledge related to
our approaches is discussed in Chapter 2.

In Chapter 3, we introduce our model-based approach to the depth map super-resolution problem. Unlike previously proposed methods, we do not fuse an HR stereo image with the depth map but search similar patches of the depth patch under the guidance of stereo structure. We first formulate depth maps using Simultaneous-Sparse-Coding modeling. Such a formulation allows us to assemble local and non-local coherences together; rather than searching non-local similar patches using the depth patch alone, we create a patch similarity dictionary from the stereo image that guides the searching process. Moreover, to avoid solving the sparse coefficient matrix based on some fixed/learned dictionary, we transform the SSC into a low-rank approximation problem where an iterative singular value thresholding strategy is employed to approximate the HR depth map.

Chapter 4 presents a deep neural network based approach to solve the depth map super-resolution problem. In this work, we attempt to fuse the LR depth map with its HR stereo counterpart. Using a deep convolutional neural network, we directly learn an end-to-end mapping function between LR depth maps and HR depth maps. Inspired by the recent success of direct depth map prediction from stereo images, we introduce a separate stereo prediction network that predicts relevant structure information to aid depth-map super-resolution. Moreover, this work attempts to partially shed some light onto the fundamental question in color-guided depth recovery: how much can color help in depth SR?

In Chapter 5, we introduce our deep neural network based approach for image interpolation problems. Current state-of-the-art super-resolution algorithms often estimate the missing high-frequency component based on the given LR image. Under the context of deep neural networks, a non-linear high-frequency estimator is learned by investigating the relationship between thousands of HR images and their directly down-sampled LR counterparts. However, such a patch-based estimation still lacks the ability to reveal local structure details. To overcome this difficulty, we proposed another network to estimate unknown pixels given their neighboring pixels; this process can be viewed as a mutant of the edge directed estimator. Experiment results have demonstrated that our method is more appreciate to handle image interpolation problem.

In Chapter 6, we discuss our approach to the natural image SR problem. In this work, we
attempt to utilize the deep convolutional neural network to handle natural image resolution changes. Unlike common deep learning approaches that learn a model between two different resolutions in a direct and end-to-end fashion, we propose to incorporate the local self-similarity properties in our deep neural network framework. Our approach is motivated from two perspectives: application and learning capabilities. In the real world, digital zooming is one of the practical applications of natural SR techniques. However, most of the learning-based techniques often only consider integer scaling factors (e.g. $\times 2, \times 3$) which ignore the scaling factors in between. We made a simple observation that the quality of the inputs matters significantly to the ability of the learned model. To overcome these challenges, we propose a network structure that not only produces intermediate resolution results but also gradually increases the quality of input LR images. Finally, in Chapter 7, we summarize and conclude our research approaches in this thesis.

Overall, we summarize our contributions in this dissertation as follows:

- **Guided Image:**
  
  - We develop two algorithms that utilize the additional information provided by the guided image. On the one hand, color images only serve as the guidance, which are not directly projected onto HR depth maps. On the other hand, color images are utilized to directly estimate the HR depth map and then merge with HR depth maps.
  
  - We observe that color images are relatively unreliable when the interference from noise is absent. Therefore, color guidance is more useful when the scaling factor is large and/or with the presence of noise. In another word, the contribution of guidance image increases when the reliability of the LR image decreases.

- **Learn Representation:**
  
  - The subproblems of the image SR problem are similar yet different tasks. To handle these problems, dedicated networks and learning procedures are needed. To handle image interpolation task, we propose to learn the model in the spatial
domain and directly estimating the missing pixels. On the contrary, estimating spectral information is promoted when tackling the natural image SR problem.

- We observe that the quality of input LR images impacts significantly on the learning outcomes. To ease the learning difficulties, we developed a progressive neural network that gradually increases the quality of the input LR image. As a result, better SR outputs can be achieved.

1.3 Thesis Organization

It is worth noting that we focus on the Guided image super-resolution problem in Chapter 3 and 4. Different approaches of how to utilize external guided imaging are demonstrated. In Chapter 3, the guiding color image is not directly fused with the LR image; it can be interpreted as guidance. This guidance rectifies the K-nearest-neighbor searching process under the assumption that similar textures often appear on the same depth layer. Therefore, the integrity of the image self-similarity property is ensured which leads to better performance even under heavy noise corruption. On the other hand, in Chapter 4, we fuse the guiding LR image onto the LR image. Unlike the conventional methods that merge both images together and then regularize the fused image, we only fuse their high-frequency components; the HR images are then created by adding high-frequency segment back to the initial interpolated LR images. Meanwhile, we also discussed the problem of how much color can help in the depth super-resolution process.

In Chapter 4, 5, and 6, learning representation of image super-resolution problem are studied and discussed. The essential innovation of these chapters is the deployment of the data-driven approach, i.e. Neural Network. In Chapter 4, two separate networks are trained jointly to recover the HR depth map. The first network is designed to learn the high-frequency structure information of the scene from the color guidance, and the second network is devoted to learning the high-frequency components from the LR depth map and to estimating noise when inputs are corrupted. We can summarize that the frequency domain representation of images is exploited by our proposed method to enhance structural details in the estimated HR images. Chapter 5 focuses on the image interpolation problem. We design
a network that estimates the missing HR pixels directly from the LR pixels. This approach, in fact, exploits the spatial domain representation of images which ensures geometric duality between lower resolution and higher resolution. Our experiment results have confirmed that learning spatial domain representation is more effective for the image interpolation problem. In Chapter 6, we propose a deep learning architecture for the natural image super-resolution problem. Our approach shares a similar insight with the local self-similarity property of natural images where intermediate layers are considered to better improve the trained model. Our experiments have confirmed the feasibility of our method and produced aesthetically pleasing results.
Chapter 2

Background

2.1 Introduction

Increasing image resolution is a fundamental problem in low-level vision research. The outcomes of image SR algorithms have proven to be useful in many areas, e.g. security, medical, and photography applications. Roughly speaking, the goal of this problem is to generate more image pixels based on a given low-resolution image. Although numerous algorithms have been proposed to increase image resolution, estimating accurate and reliable high-resolution images is still an open problem involving image acquisition, image modeling, image fitting.

To properly understand the single image super-resolution problem, a detailed illustration of how low-resolution images are acquired is necessary. In the literature, low-resolution images are often considered the degraded versions of high-resolution images and the acquisition process is modeled by the following operations: (i) PSF operator, (ii) down-sample operator, (iii)wrapping operator, and (iv) adding noise. The PSF operator is usually modeled by convolving the image with a low-pass filter and is followed by the down-sample operator that decimates pixels to reach the desired low resolution. Mathematically, we summarize the acquisition process as the following:

\[ y = DBWx + N. \] (2.1)

\( x \) is the high-resolution original image with a size of \( s_1 \times s_2 \). \( y \) is the low-resolution image
Figure 2.1: Low-resolution image acquisition model. Generally, the PSF determines the problem type. For instance, the PSF is considered as a Dirac Delta function for the image interpolation problem and treated as a Gaussian kernel when dealing with the natural image super-resolution problem.

observed with a size of $m_1 \times m_2$, where $m_1 = s_1/f_1$ and $m_2 = s_2/f_2$, where $D$, $B$, and $W$ are the decimation matrix, the Point Spread Function (PSF), and the wrapping matrix, respectively. Matrix $N$ represents the noise encountered during the acquisition process and is independent of $x$. See Fig. 2.1 for a graphic illustration of (2.1). Commonly, $f_1 = f_2 = f$ is a scale factor that is defined by the decimation matrix $D$. In practice, it is very common to consider the $D$, $B$, and $W$ matrices as a single transformation matrix $L$ and neglect the noise $N$. In doing so the formula now become

$$y = Lx. \quad (2.2)$$

Given (2.2), it is easy to see the ill-posed nature of the image super-resolution problem. Let us represent $y$ and $x$ in the lexicographic-ordered vector form (vertical concatenation of matrix columns), with a size of $m_1m_2 \times 1$ and $s_1s_2 \times 1$, respectively. Thus, the transformation matrix $L$ has a size of $m_1m_2 \times s_1s_2$, where $m_1m_2 < s_1s_2$. This is a under-determined system where infinitely many solutions exist.

It is tempting to calculate $x$ simply by the equation $\hat{x} = L^\dagger y$, where $L^\dagger$ denotes the pseudo-inverse of $L$ and $\hat{x}$ denotes the estimation of $x$. However, such a straightforward solution is not feasible. It is almost impossible to recover the exact $x$ from the observation $y$,
but many works and effects have been devoted to minimize the gap between the estimation \( \tilde{x} \) and \( x \). From now on, we shall name \( x \) as the *ground-truth*.

In order to have quantitative measurements and reflect the precision of \( \tilde{x} \), pixel-based similarity measures are often employed. Mean-Square-Error (MSE) is one of the more commonly used measurements. It is defined as

\[
MSE(\tilde{x}, x) = \frac{1}{s_1s_2} \sum_{i=1}^{s_1} \sum_{j=1}^{s_2} (\tilde{x}_{i,j} - x_{i,j})^2.
\] (2.3)

Peak Signal-to-Noise Ratio (PSNR) is another commonly used measurement for low-level vision task. Given an image \( x \) where each pixel is represented by 8-bit integers (uint8) in modern computers, the numeric value of representing one pixel is ranged from 0 to 255 and PSNR is defined as follows:

\[
PSNR(\tilde{x}, x) = 10 \log_{10} \left( \frac{(2^8 - 1)^2}{MSE(\tilde{x}, x)} \right).
\] (2.4)

Besides these pixel-based metrics, the Structural Similarity (SSIM) \[32\] index is also a well-known quality metric to measure the similarity between two images. The definition of SSIM is

\[
SSIM(\tilde{x}, x) = l(\tilde{x}, x)c(\tilde{x}, x)s(\tilde{x}, x),
\] (2.5)

\[
\begin{aligned}
l(\tilde{x}, x) &= \frac{2\mu_{\tilde{x}}\mu_x + C_1}{\mu_{\tilde{x}}^2 + \mu_x^2 + C_1} \\
s(\tilde{x}, x) &= \frac{2\sigma_{\tilde{x}}\sigma_x + C_2}{\sigma_{\tilde{x}}^2 + \sigma_x^2 + C_2} \\
c(\tilde{x}, x) &= \frac{\sigma_{\tilde{x}x} + C_3}{\sigma_{\tilde{x}}\sigma_x + C_3}
\end{aligned}
\]

where \( l, s \) and \( c \) denote three separate comparisons between \( \tilde{x} \) and \( x \): luminance, contrast and structure, respectively. SSIM is designed to be a perception-based model that considers change in structural information. However, it has been shown that an analytical link exists between PSNR and SSIM \[33\]. Therefore, there are no precise rules for selecting PSNR or SSIM as the evaluation metric to measure image quality. It is worth noting that the MSE, the PSNR and the SSIM are irrelevant to human perception and human judgment \[34\].
will elaborate this irrelevance property in detail in Chapter 6, but for now we accept them as our evaluation metrics.

Furthermore, we treat the image super-resolution process as an inverse process of image acquisition. To better understand our research, we prepare readers by introducing two modeling methods that are directly related to our research in Section 2.2 and Section 2.3.

## 2.2 Sparse Representation

In this section, we provide the necessary technical background and formulations on image sparse representation. Simultaneous sparse representation is then discussed to incorporate the non-local similarity property of image patches.

### 2.2.1 Sparse coding

Sparse signal representation, also called sparse coding, is a fundamental tool for the Compressive Sensing community and has been adopted by image processing community to model images. The fundamental philosophy of sparse representation is to decompose high-dimensional signals in the form of a linear combination of some basic elements. These elements are also known as atoms. Combining these atoms create a dictionary. Atoms in the dictionary are not require to be orthogonal, and they usually are over-complete column spanning sets. The advantage of using an over-complete dictionary over an under-complete dictionary is that it is able to capture more inherent structures and patterns.

Given an input signal \( x \in \mathbb{R}^m \), and an over-complete dictionary \( D = [d_1, d_2, \ldots, d_p] \), \( D \in \mathbb{R}^{m \times p} \), we can decompose the single as

\[
x = D\alpha, \tag{2.6}
\]

where \( \alpha = [\alpha_1, \alpha_2, \ldots, \alpha_p]^T \), and \( \alpha \in \mathbb{R}^p \), is the coefficient vector. In practice, the above strict equality is sometimes not feasible, and an approximation model is often employed,

\[
x \approx D\alpha \quad s.t. \|x - D\alpha\|_2^2 \leq \epsilon, \tag{2.7}
\]

where \( \epsilon \) is a predefined error parameter and \( \| \cdot \|_2 \) denotes the \( \ell_2 \)-norm. However, with an over-completed dictionary, the choice of \( \alpha \) is no longer uniquely determined by the signal \( x \).
Therefore, in sparse representation, an additional criterion of \textit{sparsity} is introduced to resolve the non-uniqueness. The sparsity implies that only a few entries in $\alpha$ are non-zero and the rest are zero, and the sparsity can be measured by $\|\alpha\|_0 = \#\{i : \alpha_i \neq 0, i = 1, \ldots, p\}$. In conjunction with the philosophy of finding a unique solution, it has been observed that most sensory data, such as high resolution images, can be recovered (almost) perfectly from its sufficiently sparse representation.

To find the sparsest representation, one has to solve

$$\hat{\alpha} = \arg \min_{\alpha} \|\alpha\|_0 \quad s.t. \|x - D\alpha\|_2^2 \leq \epsilon. \quad (2.8)$$

It is known to the literature that this problem is a \textbf{NP} hard problem and that the solution can be approximated by Matching Pursuit [35] [36] or Basis Pursuit [37] with a margin. Rather than directly solving it, research work [38] has shown that as long as the desired coefficients are sufficiently sparse, they can be discovered by instead minimizing $\ell_1$-norm,

$$\hat{\alpha} = \arg \min_{\alpha} \frac{1}{2}\|x - D\alpha\|_2^2 + \lambda\|\alpha\|_1, \quad (2.9)$$

where $\lambda$ is a regularization parameter that balances the sparsity of the solution and the fidelity of the single approximation. This is a convex problem, also known as \textit{Lasso} [39] in the statistic literature. In order to solve (2.9), the choice of the dictionary $D$ plays a significant role. An over-complete dictionary leading to a sparse solution can either be chosen as a pre-specified set of atoms or learned by adapting the content of the signal. In the literature, finding the proper dictionary for different applications is another challenge [40].

To summarize, we shall demonstrate the general concept of sparse decomposition by decomposing an image with a Discrete Cosine Transformation (DCT) dictionary and a learned dictionary in Fig. 2.2. Note that the calculated sparse coefficients are neglected due to space limitations.
Figure 2.2: Image sparse decomposition illustration. (a) original image, (b) DCT dictionary, (c) dictionary learned from the a by K-SVD [40], (d) reconstructed image using calculated sparse coefficients and b, (e) reconstructed image using calculated sparse coefficients and c. As we can observe from this figure that a high-resolution image can be recovered from its decomposition with negligible visual difference.
2.2.2 Simultaneous Sparse Coding

As we can observe from Fig. 2.2, sparse decomposition of an image is usually done at the patch level. That being said, images are usually decomposed into overlapping patches and these patches are assumed to be independent of each other. However, this assumption is questionable since they overlap. To better exploit the dependencies among patches, Simultaneous Sparse Coding [41] (SSC) has been proposed. The key idea of SSC, inspired by Non-Local Means filtering (NL-means) [42] and BM3D [43], is to group similar patches for a joint decomposition.

First, let us define a set of similar patches $Y_i = [y_1, y_2, \ldots, y_m] \in \mathbb{R}^{n \times m}$ for a random patch $y_i$ in an image, where $n$ is the size of the patch in lexicographic-order and $m$ denotes the number of similar patches including itself. Similar patches are usually defined by measuring Euclidean distances such that

$$\|y_i - y_j\|^2 < \xi, \quad j = [1, \ldots, m] \quad (2.10)$$

where $\xi$ is a predefined threshold. To utilize these similar-patch sets, a sparse coefficient matrix $A \in \mathbb{R}^{k \times m}$ is defined by stacking individual sparse coefficient vectors as

$$A = \begin{bmatrix} \alpha_{1,1}, & \alpha_{1,2}, \ldots, & \alpha_{1,m} \\ \alpha_{2,1}, & \alpha_{2,2}, \ldots, & \alpha_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{k,1}, & \alpha_{k,2}, \ldots, & \alpha_{k,m} \end{bmatrix}.$$  

A group sparsity regularizer is imposed to regularize the group sparsity of $A$ by $\ell_{p,q}$-norm,

$$\|A\|_{p,q} \triangleq \sum_{z=1}^{k} \|\alpha_z\|_q^p = \left[ \sum_{z=1}^{k} \left( \sum_{j=1}^{m} |\alpha_{z,j}|^q \right)^{q/p} \right]^{1/p}, \quad (2.11)$$

where $\alpha_z$ denotes the $z$-th row of $A$. In practice, the pair $(p, q)$ are usually chosen to be $(1, 2)$ which penalizes the number of non-zero rows in $A$ and enforces the alignment of sparse coefficients along the row direction. Fig. 2.3 demonstrates the difference between sparsity with group sparsity.

As the result, the similar patches $Y_i$ can be decomposed as
\[ Y_i \approx DA \quad s.t. \|Y_i - DA\|_2^2 < \epsilon. \]  

(2.12)

Combining with (2.12) the minimization problem (2.9) becomes

\[ A = \arg \min_A \frac{1}{2}\|Y_i - DA\|_2^2 + \lambda\|A\|_{1,2}. \]  

(2.13)

In this configuration, the independence assumption of individual patches is preserved while similar patches are forced to have similar sparse representation. Consequently, with the enforced non-local structural similarity, patches are more repellent to the small structural variations and therefore lead to better reconstruction results.

2.3 Deep Neural Networks

In this section, we provide necessary details on supervised machine learning and neural networks.

2.3.1 Supervised Learning

Many real world problems require computers to perform a “simple” paradigm to predict acceptable outputs with given inputs. Unfortunately, there is no simple solution to this
In the early stages of research, many algorithms have been custom-tailored for specific applications and often generated unsatisfying results. Supervised learning is an alternative approach that allows machines to learn desired mapping between inputs and outputs.

In order to teach and train machines to perform a specific task, related knowledge/examples are required. Fortunately, we can safely assume that the task of collecting training samples is relatively simple and straightforward, e.g. data collected and labeled by human beings. Given a training dataset of \( n \) samples \( \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \) that are independent and identically distributed (i.i.d) samples generated from a distribution \( \mathcal{D} \), supervised learning algorithms aim to find the mapping \( f : y \rightarrow x \) that best describes the relationship between them. More specifically, the mapping function \( f \) is often chosen from some particular class of functions \( \mathcal{F} \) and a loss function \( \mathcal{L}(x, y) \) is needed to measure the displacement between the predictions and ground-truths. Formally, the supervised learning algorithm is designed to solve

\[
f \approx \arg \min_{f \in \mathcal{F}} \mathcal{L}(f(y), x) \quad \text{s.t.} \quad (x, y) \in \mathcal{D}.
\]

Unfortunately, the above optimization problem is impossible to solve simply because we are not capable of including all samples generated by \( \mathcal{D} \) in our training dataset. Therefore, the evaluation of the supervised learning algorithm can only be done for the available training dataset at hand:

\[
f \approx \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(y_i), x_i)
\]

That being said, we find a mapping function \( f \) that optimizes the loss over the available training dataset and hope this function \( f \) can be generalize to other uncollected samples generated by \( \mathcal{D} \).

Similar to sparse representation, the choice of \( f \in \mathcal{F} \) is not unique. In addition, the uncertainty of whether \( f \) can be generalized outside of the training data rises quickly. Therefore, additional regularization is required for (2.15) to narrow down the selections

\[
f \approx \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(y_i), x_i) + \mathcal{R}(f),
\]
2.3.2 Neural Network

Neural Networks (NN) are designed to solve problems in the same way that a human would. Biologically inspired and modeled after the human brain, a neural network is consisted of multiple layers of "neurons". Each neuron is connected with many others in both the preceding layer and the succeeding layer. Within each neuron, a summation operator combines the value of all its inputs together and passes it to the activation operator. The total value must surpass a threshold before it can propagate to other neurons. See Fig. 2.5 for an illustration.

For read world applications, the simplest neural network, also called the vanilla neural network or full-connected network (FCN), only consists of matrix multiplication and element-wise non-linearity. Mathematically, a three-layer network, is modeled by

$$f(y) = W_3^T \sigma (W_2^T \sigma (W_1^T y + b_1) + b_2) + b_3,$$  \hspace{1cm} (2.17)
Figure 2.5: A diagram of biologically inspired model for a single neuron. Input $y_i$ are multiplied by weight $w_i$. Results are accumulated in the cell then propagate to the next neurons if it surpass the activation threshold.

Where $W_1, W_2, W_3$ are weight matrices and $b_1, b_2, b_3$ are biases, and all of them are hyper-parameters to be designed; $\sigma$ denotes an element-wise non-linearity. Commonly used non-linearities are the Sigmoid function, $1/(1+e^{-x})$, the Rectified Linear Unit (ReLU), $\max(0, x)$, and tanh, $2/(1+e^{-2x})−1$. Note that outputs from the last layer usually are not activated by any non-linearities. A graphical illustration of a three-layer vanilla neural network is shown in Fig. 2.6.

Convolutional Neural Networks (CNNs) [44] are neural network architectures designed to handle high-dimensional data, such as texts, sentences, images, and videos. The core block of a Convolutional Neural Network is the Convolutional Layer which takes an input tensor and produces an output tensor. Let us consider an image (a low-resolution image under the context of super-resolution) as the input to the network. Here, the multi-dimensional array
representing this image is a tensor. For example, a $256 \times 256$ grayscale image is a $256 \times 256 \times 1$ tensor, and a $256 \times 256$ color image is a $256 \times 256 \times 3$ tensor. Indicated by the name, the convolutional layer utilizes the convolution operator. Therefore, instead of connecting all neurons by a giant weight matrix $W$, we can define a filter $W_f$ with a much smaller size to connect neurons. Then, we can slide this filter across all spatial locations of the input tensor and convolve accordingly. The resulting tensor is often named the activation map or the feature map. For example, convolving the input tensor with $W_f$, a $5 \times 5 \times 3$ tensor, results in a $252 \times 252 \times 1$ tensor (size change due to convolution). We can also pad the input with zeros or reflect the borders to control the output dimension. It is easy to see that one set of filters and biases is linked with one feature map and shared by all spatial locations of the input tensor. In general, the Convolutional Layer possess more than one set of filters and biases, and convolving the input tensor with each of these filters generates a set of activation maps. See Fig. 2.7 for an illustration of a convolutional layer.

With the understanding of the basic building block, we can now construct a Fully Convolutional Neural Network (FCNN), e.g. a three-layer CNN. For example, the input is a degraded grayscale patch $y$ with size of $44 \times 44$, and the output is the non-degraded counterpart $x$ with the same size. We stack three layers of convolutional layers between them. The first layer possesses a set of 64 filters, each of them is a tensor with size $3 \times 3 \times 1$. We zero-pad the input tensor with a border of zeros ($45 \times 45$) to produce an output tensor with size $44 \times 44 \times 64$. After activation, the second layer takes it as its input. With zero-padding and convolving a set of 64 filters of size $3 \times 3 \times 64$, the activated output tensor is sent to the final layer. The final layer only possesses one set of filters with a size of $3 \times 3 \times 64$, and the output is an approximation of $x$. See Fig. 2.8 for an illustration of this example. Counting all the hyper-parameters in this neural network, we have $3 \cdot 3 \cdot 1 \cdot 64 + 64 + 3 \cdot 3 \cdot 64 \cdot 64 + 64 + 3 \cdot 3 \cdot 64 \cdot 1 + 1 = 38145$ total parameters that need to be determined. This is (almost) impossible to find manually. Therefore, a systemic optimization process is required; see Section 2.3.3 for details.

In general, the deeper the CNN gets, the more powerful it becomes; hence the name deep convolutional neural network. The only drawbacks are that the network becomes very complicated and very hard to train. There are many methods have been proposed to ease
the training process, such as *Batch-Normalization* [45], *drop-out* [46], *residual-learning* [47], *etc*. We do not elaborate them here because they are beyond the scope of this dissertation.

### 2.3.3 Optimization and Backpropagation

In this section, we introduce a derivative based optimization process and *backpropagation* to determine all the hyper-parameters needed in CNNs. Recall the optimization formula (2.16) we introduced in Section 2.3.1. We can rewrite it under the context of CNN as

\[
\Theta^\ast = \arg \min_{\Theta} g(\Theta, x, y) = \arg \min_{\Theta} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f_{\Theta}(y_i), x_i) + \mathcal{R}(f_{\Theta}),
\]

(2.18)

where \( \Theta \) represents all the hyperparameters and \( g(\cdot) \) denotes the total losses induced. The objective is to find the \( \Theta \) that minimizes \( g(\cdot) \).

One advantage of *neural networks* in general is that the total loss function is a differentiable function. We can compute the gradient \( \nabla_{\Theta} g \) with *backpropagation* (which will be discussed next). This gradient is a vector of partial derivatives where the geometric slope of \( g(\cdot) \) is calculated along every dimension of \( \Theta \). We then approximate \( g(\cdot) \) by the first order Taylor approximation as

\[
g(\Theta) \approx g(\Theta_1) + \nabla g(\Theta_1) \cdot (\Theta - \Theta_1),
\]

(2.20)

Where \( \Theta_1 \) is found by moving from \( \Theta \) a small amount in every dimension of \( \Theta \). Therefore, we can search for the optimal parameter \( \Theta^\ast \) by iteratively updating \( \Theta_i \) by adding to it a small amount of every negative gradient direction.

This observation inspired the famous *gradient descent* algorithm that iteratively updates the initial hyperparameters until the total loss reaches a minimum. We summarize the core of the *gradient descent* algorithm in Algorithm.1.
Algorithm 1 Gradient Descent Algorithm

1: **Input**: Training dataset \((x, y)\), a step size \(h\);

2: **Input Initialization**: Initialize the hyperparameters \(\Theta\) with a random guess;

3: while Not Converged do

4: Estimate the gradient \(\nabla_\Theta g(\Theta, x, y) \approx \nabla_\Theta \left[ \frac{1}{n} \sum_{i=1}^{n} L(f_\Theta(y_i), x_i) + R(f_\Theta) \right] ;

5: Compute with update gradient of each dimension with a small step size \(h\): \(\Delta \Theta \leftarrow -\nabla_\Theta g(\Theta, x, y) \cdot h\);

6: Apply the increments: \(\Theta = \Theta + \Delta \Theta\).

7: end while

Now, we know that the key element of acquiring \(\Theta\) is to evaluate the gradient of the loss function. Under the context of the neural network, the best way of evaluating the gradient at each iteration is to utilize backpropagation.

The **Backpropagation algorithm** is designed to compute gradients of scalar valued functions with respect to their inputs. The core of the backpropagation algorithm is the **Chain Rule**. Let us consider a simple example that \(g(\Theta) = [w_1(w_0x_0 + b_0)]^2\); hence the \(\nabla_\Theta g(\Theta) = \left[ \frac{\partial g}{\partial w_0}, \frac{\partial g}{\partial w_1}, \frac{\partial g}{\partial b_0} \right] \). Following the philosophy of the **Chain Rule**, we can introduce a few intermediate functions: \(a = w_0x_0; b = a + b_0; c = w_1b; g = c^2\), and the corresponding local gradients are: \(\frac{\partial a}{\partial w_0} = x_0; \frac{\partial b}{\partial a} = 1; \frac{\partial c}{\partial b} = 1; \frac{\partial c}{\partial w_1} = b; \frac{\partial c}{\partial b_0} = w_1; \frac{\partial g}{\partial c} = 2c\). Now we image each of these functions as a node in a graph, *i.e.* a directly acyclic graph (DAG), where vectors flow through them. There are two directions that vectors can flow: (i) forward pass that calculates \(g(\Theta)\), (ii) backward pass that passes gradients with respect to each node. It is easy to see that the gradient with respect to a vector is nothing but multiply the local gradient with the incoming gradient, *e.g.* \(\frac{\partial g}{\partial w_1} = \frac{\partial g}{\partial c} \frac{\partial c}{\partial w_1} = 2cb\), where \(\frac{\partial g}{\partial c}\) is the incoming gradient in the back pass and \(\frac{\partial c}{\partial w_1}\) is the local gradient with respect to the node. A graphical illustration for backpropagation of this example can be found in Fig. 2.9.
Figure 2.6: An illustration for a three-layer full-connected network. In this architecture, each neuron is connected with all neurons from both the preceding layer and the succeeding layer. Note that activation functions are omitted in this diagram and data propagate to the next layer only if that particular neuron is activated. Under the super-resolution framework, we can image \((y_1, y_2)\) as two pixels from a low-resolution image and \((x_1, \ldots, x_4)\) as four pixels from its corresponding high-resolution image. This neural network describes the relationship between them and we intend to find all the hyper-parameters that minimize the total loss.
Figure 2.7: An illustration of a *convolutional layer*. An input tensor, $256 \times 256 \times 3$, is convolved with a set of 64 filters where the size for each filter is $5 \times 5 \times 3$. Note that filters are independent to each other and always span the entire depth of the input tensor (3 in this case). The output tensor is generated by stacking the outputted *feature maps* together and has a size of $252 \times 252 \times 64$. The number of filters, 64, is not fixed; this number is determined by the user depending on the training resources. If this *convolutional layer* is followed by another *convolutional layer*, the activated output tensor becomes the input tensor, and the filter is a tensor with depth of 64.
Figure 2.8: An illustration of a three layer convolutional neural network. The input $y$ to this network is a $44 \times 44 \times 1$ tensor. We choose the filter with size of $3 \times 3$ and set the number of feature maps in layer 1 and 2 to be 64. After each convolution operation, an activation function is employed except in the output layer. The goal is to find these filter-and-bias sets that transform the input $y$ to the output $x$ with minimum loss. Note that padding and activation are omitted in this diagram for simplicity.
Figure 2.9: Figure (a): an example of backpropagation along a node of a graph. During forward pass, vectors $a$ and $b$ are combined by some local function $c$; the local gradients are $\frac{\partial c}{\partial a}$ and $\frac{\partial c}{\partial b}$. During backward pass, the incoming gradient $\frac{\partial g}{\partial c}$ flows through the node and propagates to the next set of nodes by multiplying the local gradient with respect to their inputs. Figure (b): a graphical illustration of example $g(\Theta) = [w_1(w_0x_0 + b_0)]^2$. Blue letters denote the vectors at the forward pass and red letters denote the incoming gradients at the backward pass. Local gradients are defined by the local operations: $\frac{\partial a}{\partial w_0} = x_0; \frac{\partial a}{\partial a} = 1; \frac{\partial b}{\partial b} = 1; \frac{\partial c}{\partial w_1} = b; \frac{\partial c}{\partial b} = w_1; \frac{\partial g}{\partial c} = 2c$. 
Chapter 3

Image assisted upsampling of Depth Map via Non-local similarity

3.1 Introduction

Capturing a high resolution depth map has been a challenging task. Before the introduction of time-of-flight (TOF) cameras, people used expensive laser scanners to obtain the depth information. This pixel-per-pixel scanning procedure often requires a large amount of time and computational resources which limits its use in practice. Recently, Microsoft announced their second-generation Kinect, which was an inexpensive and portable depth map capturing device. Unlike the previous generation, it was equipped with a TOF camera providing faster depth map acquisition. However, the depth resolution of even this new-generation Kinect is still low and the acquired depth map often suffers from the impact of acquisition noise. Consequently, developing powerful depth map processing tools such as upsampling and denoising has become desirable for the purpose of obtaining high quality depth maps.

Early upsampling methods include two major classes, one is filter-based [48], [11], and the other is Markov Random Field (MRF) based [13]. Joint Bilateral Upsampling (JBU) [48] and MRF [13] both promote the idea of extracting additional information from a high resolution image to assist the upsampling process and produce good results. Thus, various information can be obtained from a high resolution image and used for upsampling. More
recent works include guided image filtering [49] that can be used as an edge-preserving smoothing operator; Anisotropic Total Generalized variation approach [18] to formulate the upsampling as a convex optimization problem using primal-dual algorithm, and high resolution upsampling [14] to construct this problem using constrained optimization. Most recently, depth map denoising has also been studied in the literature (e.g., [50]).

In this Chapter, we view depth map denoising and upsampling as a joint problem and propose to exploit the patch dependency/similarity of color images to guide the upsampling process of low-resolution depth maps, as demonstrated in Fig. 3.1. Our method [51] relies on the fact that depth discontinuity information of a targeted scene is often embedded within the corresponding high resolution color image (with the exception of textured regions) [8]. In the meantime, even though homogeneous texture regions of natural images tend to appear on the same depth layer, such seemingly inconsistencies do not affect the patch similarity result. Therefore, it is plausible to leverage the patch clustering result of high resolution color images into the nonlocal similarity enforcement during the upsampling of low-resolution depth maps. Such new insights distinguish our approach from the previous ones.

More specifically, we propose to exploit the nonlocal similarity enforcement for depth map upsampling via low-rank methods. Low-rank methods have attracted increasingly more attention in recent years and been successfully applied to various applications including matrix completion [52], [53] and image restoration [54]. Recently, the connection between nonlocal sparse representations and low rank methods was explored for natural images in [54] and further extended by the tool of Graph-based Transform [50] for depth maps. By utilizing this connection, excellent image and depth map denoising results have been obtained. When compared with those works, this Chapter can be viewed as another extension of the proposed Spatially Adaptive Iterative Singular-Value Thresholding (SAIST) [54] algorithm into image-guided depth map upsampling where the patch dependency among high-resolution depth maps is “learned” from the accompanying color images. Experimental results with the widely used Middlebury database have shown that our method is comparable with previously developed image-guided depth map upsampling methods.

The outline of this chapter is as follows. First, we describe the proposed image-guided SAIST method in Section 3.2; in Section 3.3, we report our experimental results to demon-
strate the performance of our method; in the end, we discuss and summarize this chapter in Section 3.4.

Figure 3.1: Example of upsampling on a low resolution depth map: (a) Low resolution depth map with noise (b) Corresponding high resolution RGB image (c) Depth map after image-guided SAIST. The input depth map has a resolution of $273 \times 344$, and we increase its resolution with the factor $\times 4$. The high-resolution depth map is estimated by our proposed image-guided SAIST method.
3.2 Algorithm Development

In this section, we develop a depth map upsampling algorithm based on image-guided SAIST. The methodology of this approach can be divided into three steps: (1) Registering the low resolution depth map into the high resolution image coordinate (Section 3.2.1), (2) learning nonlocal dictionaries of high resolution images (Section 3.2.2), and (3) importing the dictionaries into SAIST (Section 3.2.3).

3.2.1 Depth Map Registration

In our setup, we define $X_d = (X, Y, Z, 1)^T$ as the global coordinate of the captured 3D scene, and the TOF camera as the coordinate origin. Each measurement $d_{i,j}$ at low resolution pixel position $x_{ij} = (i, j, 1)^T$ represents the distance between the origin and a single point $X_{i,j,k}$ at the global coordinate $X_d$. To be noticed, this measurement is an averaged depth value of multiple points around $X_{i,j,k}$, and the entire depth map is treated as a projection of the 3D scene. We then re-project these measurements into its corresponding coordinate on the high resolution image space $\Omega_H$ by calculating

$$ \tilde{x}_{i,j} = P_H d_{i,j} P_L^\dagger, $$

where $P_L^\dagger$ is the pseudo-inverse of the low resolution projection matrix, and $P_H$ is the projection matrix for the high resolution depth map.

Therefore, a new depth map has been created with a sparse set of depth values at position $\tilde{x}_{i,j}$. Since the global minimum of upsampling result remains unknown in terms of MSE, we interpolate the missing pixels between known positions to initiate the algorithm to find a better local minimum and name the initialized depth map as $y$.

3.2.2 Dictionary Learning

We re-evaluate the strength of patch-based image processing by investigating nonlocal similarity properties under the context of upsampling. Previous studies on nonlocal upsampling for magnetic resonance images [55] and depth maps [14] confirm the advantages
of relying on nonlocal properties. Therefore, it is plausible that exploitation of nonlocal similarity by patch clustering could lead to fine details and structures.

Instead of collaborating patch clustering directly on the depth map, we apply it under the guidance of high resolution color images by leveraging the fact that the depth map and the corresponding color image are related. We can view this relationship subjectively by comparing a depth map and RGB color image pair to observe their structural similarity. Discarding the fact that depth map intensity values represent the physical distance while intensity values of color images stand for different textures, the remaining content is similar edge/structure information. In Fig. 3.2, we use k-means clustering with a patch size of five to show the structural similarity between the depth map and the color image. To put it another way, structure perimeter information "borrowed" from high resolution image will gradually increase the sharpness and accuracy of the upsampled result.

![Figure 3.2: Comparison of k-means clustering between depth map and color image of middlebury Books.](image)

Our upsampling method increases the low resolution of captured depth maps by making the best use of the strategies described above. For a given \( N \times M \) high resolution image \( I \) and initialized depth map \( y \), we extract all patches separately and formulate two matrices by converting every patch as a single column, named as \( M_I \) and \( M_y \). Both matrices are highly correlated where the corresponding columns are given the same labels by their position.
Then, for each column in $M_I$ (exemplar patch), we search for its KNNs by comparing its Euclidean distances with all other patches inside the searching window. Next, we construct our nonlocal dictionary $D$ by stacking the label of exemplar and the labels of its KNN as columns and we use a simplified example in Fig. 3.3 to demonstrate the process.

![Figure 3.3: Sample process of finding a dictionary.](image)

### 3.2.3 Spatially Adaptive Iterative Singular-Value Thresholding

We first overview simultaneous sparse coding as a standard low-rank approximation problem [54]. Sparse coding means that a patch can be represented by a learned dictionary $U_i$ and a collection of sparse weight vectors $\alpha_i$, as $y_i \approx U_i \alpha_i$. Therefore, a minimization problem for different image restoration tasks can be written as

$$\arg\min_{U_i, \alpha_i} \|y_i - U_i \alpha_i\|_2^2 - \tau \|\alpha_i\|_0,$$  

(3.2)

where $\tau$ is a Lagrange multiplier serving as a regularization parameter. However, regardless of the assumption of independence between patches, the idea of group sparse coding is to
exploit dependence among all patches by encoding the exemplar and its KNN jointly using the same learned dictionary $U$:

$$(U, A) = \arg\min_A \|Y - UA\|_F^2 - \tau \sum_{i=1}^{N} \|\alpha_i\|_1^2. \quad (3.3)$$

We use the predefined dictionary $D$ to find all KNNs and further write equation 3.3 using low-rank approximation as

$$(U, \Sigma, V) = \arg\min_{U, \Sigma, V} \|Y - U\Sigma V^T\|_F^2 - \tau \sum_{i=1}^{K} \lambda_i, \quad (3.4)$$

where,

$$(U, \Sigma, V) = \text{svd}(Y). \quad (3.5)$$

In addition, iterative regularization has been studied to update the result. The original idea of iterative regularization is to add filtered noise back to the image as

$$\hat{y}^{(k+1)} = \hat{y}^k + \delta(y - \hat{y}^k), \quad (3.6)$$

where $y$ is the input noisy low resolution depth map, $\hat{y}^k$ is the output of the $k$-th iteration and $\delta$ is a relaxation parameter. However, instead of using this technique on the upsampled depth map, signal variance and noise variance will be estimated and updated alternatively through

$$\hat{\sigma}_w^k = \gamma \sqrt{\sigma_w^2 - \|y - \hat{y}^{(k+1)}\|_2}, \quad (3.7)$$

$$\hat{\sigma}_i^{(k+1)} = \sqrt{\max((\hat{\lambda}_i^k)^2/m - (\hat{\sigma}_w^k)^2, 0)}. \quad (3.8)$$

As a result, the threshold/regularization parameter can be iteratively determined by $\tau = 2\sqrt{2}\sigma_w^2/\sigma_i$ [56] and applied to soft thresholding $\hat{\Sigma} = S_\tau(\Sigma)$. The new data matrix can be reconstructed by $\hat{Y} = U\hat{\Sigma}V^T$.

With the help of iterative regularization, we usually observe that the Mean-Square-Error (MSE) of upsampled depth map monotonically decreases and discloses more details. A step-by-step description of image-guided depth map upsampling algorithm is given in Algorithm 2.
Algorithm 2 Depth Map Upsampling via Image-Guided SAIST

1: **Input**: One noisy low resolution depth map and one high resolution image $I$;

2: **Input Initialization**: Register the low resolution depth map on the high resolution image space and create $y$;

3: **Dictionary Learning**: Find the KNN for each exemplar patch in $M_I$ and extract their labels to create the dictionary $D$;

4: **Initialization**: $\hat{y}^{(1)} = y$;

5: **for** $k = 1$ to iter **do**

6: Patch clustering: Search $D$ for the KNN labels of the $i$-th exemplar patch of $y$, and create data matrix $Y_i$ by stacking corresponding columns from $M_d$;

7: SVD for each data matrix $Y_i$: $(U_i, \Sigma_i, V_i) = \text{svd}(Y_i)$;

8: Signal variance and noise variance update: update both variance use Eqs. (3.7) and (3.8);

9: Thresholds update: $\tau = 2\sqrt{2}\sigma_w^2/\sigma_i$;

10: Singular value thresholding: $\hat{\Sigma} = S_r(\Sigma)$;

11: Depth map update: Obtain the improved depth map by combining all reconstructed data patches;

12: **end for**

13: **Output**: The upsampled depth map $\hat{y}^k$. 
3.3 Experiment Result

In this section, we present our experimental results with the image-guided SAIST upsampling algorithm developed in Section 3.2. We compare against several competing upsampling methods and further demonstrate its performance by showing its ability in depth map upsampling.

3.3.1 Evaluations using the Middlebury stereo dataset

We evaluate our image-guided SAIST algorithm with three Middlebury stereo depth maps \textit{Art}, \textit{Moebius}, and \textit{Books}. We downsample these depth maps by different upsampling factors (\textit{e.g.} $\times2, \times4$) to create low resolution depth maps. More specifically, we have a set of low resolution depth maps, three high resolution RGB color images and three high resolution depth maps as the ground-truth. We first demonstrate the strength of iterative regularization. In Fig. 3.4, we calculate the PSNR of the input $y$ (\textit{Art}) to indicate the starting point (red baseline). Two user adjustable parameters, patch size and number of similar patches, are varied to demonstrate the impact of nonlocal properties. For depth map \textit{Art}, the large patch size and increased number of similar patches will misleading in obtaining accurate structure information and patch locations, which leads to sub-optimal results. We can easily observe that the depth map is greatly intensified after the first iteration and continuously improved as iteration progresses. To be noticed, in Fig. 3.4 upsampled results tend to converge after several iterations.

Based on the iteratively updated results, we have compared our results with bilinear interpolation, MRF by Diebel and Thrun [13], guided image filtered approach by He \textit{et al.} [49], non-local means filtering by Park \textit{et al.} [14] and anisotropic total generalized variation approach by Ferstl \textit{et al.} [18]. The numerical results of these experiments in terms of Peak Signal-to-Noise Ratio (PSNR) are presented in the Table 3.1. These experiments provide an objective comparison on the effectiveness and accuracy of different algorithms. It is easy to observe that our algorithm produces the leading results, which achieved more than one dB gain for an upsampling ratio of two. For an upsampling ratio of four, our algorithm is still capable of producing better results on \textit{Art} and \textit{Moebius}. For \textit{Books}, we compare the error
Figure 3.4: The numerical (PSNR) results of iteratively upsampled depth map Art. Patch size and number of similar patches are varied to indicate the effectiveness of regular nonlocal parameters.

maps in Fig. 3.6. We observe an overall less error density, but there are two relatively large errors (highlighted in Fig. 3.6a). An possible explanation is that our algorithm misjudges the structural information at these particular locations and keeps reinforcing these nonexistent edges. Next, we compare these results with the outcomes of noisy depth map upsampling in Section 3.3.2 to demonstrate the consistency and robustness of our algorithm over noisy depth maps.

### 3.3.2 Evaluation with synthetic noise

To simulate the acquisition process, additive white Gaussian noise is added to all low resolution depth maps. Although the real noise distribution is more sophisticated than in
Table 3.1: Objective quality comparison without noise on Middlebury dataset. The performance is measured as PSNR(dB) for two different factors (×2, ×4).

<table>
<thead>
<tr>
<th></th>
<th>Art</th>
<th>No Noise</th>
<th>Art</th>
<th>No Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x2</td>
<td>x4</td>
<td>x2</td>
<td>x4</td>
</tr>
<tr>
<td>Bilinear</td>
<td>37.36</td>
<td>32.86</td>
<td>45.19</td>
<td>40.48</td>
</tr>
<tr>
<td>Diebel and Thrun [13]</td>
<td>38.25</td>
<td>36.55</td>
<td>46.51</td>
<td>44.34</td>
</tr>
<tr>
<td>He et al. [49]</td>
<td>38.78</td>
<td>36.56</td>
<td>46.82</td>
<td>44.20</td>
</tr>
<tr>
<td>Park et al. [14]</td>
<td>39.08</td>
<td>37.25</td>
<td>46.58</td>
<td>44.63</td>
</tr>
<tr>
<td>Ferstl et al. [18]</td>
<td>38.50</td>
<td>36.57</td>
<td>45.92</td>
<td>44.03</td>
</tr>
<tr>
<td><strong>OURS</strong></td>
<td>40.86</td>
<td>37.30</td>
<td>47.38</td>
<td>41.94</td>
</tr>
</tbody>
</table>

the Gaussian model, we implement it only for fair comparison with previously developed algorithms. In order to further demonstrate the superiority of our algorithm against noise, we compare all results to the noise-aware bilateral filter approach by Chan et al. [11]. The qualitative comparison results are summarized in the Table 3.2. We can clearly see that our results achieve more than two dB gain for a factor equal to two and more than one dB gain for a factor equal to four. Unlike other algorithms, the PSNR values of our algorithm remain at the same level while others dramatically decrease. These consistent results validate the robustness of our algorithm to depth noise. A subjective quality comparison of different methods is presented in Fig. 3.5 for noisy depth map *Moebius*, and we enlarge the polygon on the right side in Fig. 3.7 for further comparison. Similarly, a quality comparison for the noisy depth map *Art* is presented in Fig. 3.8. It can be observed that the subjective quality of our result exhibits clear structure, smooth surfaces and suppressed noise level, while others remain noisy and blurred along the edges. We confirm that similar textures in high resolution color image provide desirable references for our algorithm to locate similar patches in depth map. Hence, accurate patch clustering leads to exploiting the capacity of group sparsity, which is why our algorithm outperforms other compared depth map upsampling methods.
Table 3.2: Objective quality comparison results with noise. The performance is measured as PSNR(dB) for two different factors ($\times 2$, $\times 4$).

<table>
<thead>
<tr>
<th>Noise</th>
<th>Art $\times 2$</th>
<th>Art $\times 4$</th>
<th>Books $\times 2$</th>
<th>Books $\times 4$</th>
<th>Moebius $\times 2$</th>
<th>Moebius $\times 4$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>34.93 33.14</td>
<td>36.22 35.46</td>
<td>35.69 34.95</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>37.27 35.04</td>
<td>41.83 38.58</td>
<td>41.57 38.28</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>37.13 35.23</td>
<td>40.61 39.38</td>
<td>40.23 39.09</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>36.63 34.94</td>
<td>42.34 39.80</td>
<td>42.30 40.14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>38.06 35.95</td>
<td>44.48 41.23</td>
<td>44.75 41.98</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>37.40 35.13</td>
<td>41.72 39.27</td>
<td>41.78 39.31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>OURS</strong></td>
<td><strong>37.90 33.41</strong></td>
<td><strong>43.29 38.76</strong></td>
<td><strong>45.12 40.88</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.4 Summary

The depth resolution of TOF camera is poor and the resulting depth maps are noisy. Therefore, it is highly desirable to develop powerful image processing tools to enhance the resolution and suppress the noise of depth maps. In this chapter, we present a novel depth map upsampling algorithm using a low resolution depth map captured by TOF camera under the guidance of a high resolution color image. A spatially adaptive iterative singular-value thresholding (SAIST) with image-guided patch clustering strategy is developed and compared with previous image-guided depth map upsampling technique. Our algorithm exploited the similar structure relationship between color image and depth map. We formulate a dictionary using similar patch location in color image and pass it to form group sparse coding via low rank approximation. The upsampled depth map also can be iteratively updated with the help of iterative regularization. When testing our algorithm on the well-known dataset, we show our algorithm typically outperforms several competing approaches. We further demonstrate the robustness of our algorithm against noise. The numerical and visual results are consistent with the fact that nonlocal patch clustering is effective for reveling details in image domain.
Figure 3.5: Subjective quality comparison of ×4 upsampling of middlebury *Moeibus* dataset with added AWGN noise. (a) RGB high resolution color image. (b) Low resolution noisy input image. (c) Upsampling using MRF approach by Diebel and Thrun [13]. (d) Adaptive bilateral upsampling approach by Chan *et al.* [11]. (e) Nonlocal means upsampling approach by Park *et al.* [14]. (f) Anisotropic total generalized variation by Ferstl *et al.* [18]. (g) Our Upsampling result using image-guided SAIST. The results in (c),(d) and (e) are still suffer from noise. (f) removes majority of the noise and sharp edges around small structure. Our algorithm removes noise and retains sharp edges.
Figure 3.6: Error map comparison of *Books* with upsampling factor of four between our algorithm and nonlocal means upsampling approach by Park *et al.*
Figure 3.7: Enlarged comparison results of polygon inside Moebius from Fig. 3.5.
Figure 3.8: Subjective quality comparison of \( \times 4 \) upsampling of Art with added AWGN noise. (a) result produced by Diebel [13], (b) generated by Chan [11], (c) generated by Park [14], (d) generated by Ferstl [18], (e) Our result, (f) ground-truth. By comparing our proposed method with many state-of-the-art method, our result successfully suppresses the level of noise (negligible by human eyes) and preserved better structure details.
Chapter 4

Guided Deep Network for Depth Super-Resolution: How much can color help?

4.1 Introduction

Acquiring high-quality depth maps is a fundamental challenge for many vision related tasks, such as intelligent vehicles, gesture recognition, and 3D model rendering. In the past decade several model-based depth map super-resolution (SR) methods have been developed to improve the quality of depth maps. Diebel et al. [13] formulated depth map SR as an optimization problem, and integrated low-resolution depth maps with high-resolution color images using Markov Random Field (MRF). Park et al. [14] introduced non-local means into MRF to regularize depth maps. They also incorporated an edge-weighting scheme based on color image to preserve fine structural details. Ferstl et al. [18] proposed a total generalized variation model to regularize depth maps through an anisotropic diffusion tensor obtained from the color image. Zhou et al. [51] formed a dictionary by finding K-nearest neighbors (KNN) for each depth patch under the guidance of its corresponding color image, and iteratively solved a simultaneous sparse coding problem to refine depth details. Despite the success of these color guided approaches, the fundamental question - how much can color help? - largely remains open.
Recent breakthroughs in deep learning or deep neural networks have led to state-of-the-art performance in various computer vision applications including both low-level and high-level tasks. Under the context of single image super-resolution (SR) [27] [30] [57], Dong et al. [31] proposed a simple end-to-end deep convolutional neural network (CNN) to learn nonlinear mapping between low-resolution (LR) and high-resolution (HR) natural images. This work achieved excellent performance and inspired a deeper CNN architecture proposed by Kim et al. [58] as well as another deep recursive neural network [59] for image SR. These methods successfully demonstrate that end-to-end nonlinear mapping can be learned between low-resolution color images and their corresponding high-resolution images. However, these network models cannot be directly applied to depth map SR because depth maps acquired by TOF cameras have different intrinsic properties.

In this Chapter, we propose a novel CNN architecture to tackle learning-based color-guided depth map SR problem [60]. For depth maps distorted by noise, denoising and SR are treated jointly in our problem formulation. The first part of our network consists of a series of fully convolutional layers to estimate missing high-frequency and noise components simultaneously. We call this part of the network “Depth Enhancement Network” (DEN). The second part of our network is designed to explicitly exploit the structural correlation between color images and depth maps. Inspired by [61], which attempts to predict a depth map from a single color image, we propose to utilize the HR color image as a prediction network. This part of the network is called “Color-based Prediction Network” (CBPN). With two independent and competing networks, we also address the issue of auto-merging before the final reconstruction of depth maps. The proposed network architecture enables us to at least partially shed some light onto the aforementioned fundamental question.

Under the framework of deep CNN, the amount of reliable information (e.g., high-frequency components) in training images directly impacts the learning outcomes especially for low-level vision tasks such as SR. To see this, we have shown a visual comparison of depth-color pair along with the interpolated depth profiles with and without noise contamination in Fig. 4.1. It can be observed that when noise is absent, only small amount of high-frequency components are missing in the interpolated depth map (Fig. 4.1c); therefore the help from the corresponding color image (Fig. 4.1b) is limited. Moreover, depth values in
Figure 4.1: Comparison of a typical depth edge with its counterparts. High-frequency components in (c) is the most reliable; including (b) when SR (c) only confuses the network. On the contrary, including (b) when SR (d) provides relatively more reliable high-frequency components.

The noisy interpolated depth map (Fig. 4.1d) are only partially preserved - precise locations of depth discontinuities are severely distorted. In this situation, the high-frequency components contained in the supplementary color image could be highly useful. Apparently, how much can color help is question that is difficult to answer in traditional model-based color guided depth recovery framework; deep CNN at least suggests an alternative learning-based approach to joint depth SR and noise suppression.

Our contributions are summarized as follows:

1. We propose to directly learn end-to-end nonlinear mapping for the depth map super-

resolution problem under the framework of Convolutional-Neural-Network.

2. By analyzing the characteristics of low-resolution depth maps and color images, we observe that color images are more helpful when noise is present and/or the scale factor is large.

3. We propose a color-based prediction network for the learning processes that need color guidance. The combined network leverages the contribution from depth maps and color images automatically to finalize the nonlinear model.

4.2 Depth Super-Resolution Network

The goal of our proposed network is to learn a nonlinear mapping that describes the relationship between LR and HR depth maps. Our network consists of four components. The first component is a depth enhancement network (DEN), which estimates the missing high-frequency components from the LR depth map. The importance of estimating missing high-frequency components has been discussed in [58] and [62]. The second component is a color-based prediction network (CBPN), which predicts the high-frequency components for the HR depth map. The third component is an auto-merging part, in which feature maps produced by DEN and CBPN are automatically combined into a set of new feature maps. The last component of our network reconstructs the HR depth map from the merged feature maps. Note that CBPN and auto-merging become more important when the LR depth map becomes less reliable (e.g., due to presence of noise). A graphical illustration of the proposed network is shown in Fig. 4.2.

Formally, depth SR is formulated as a problem of estimating a HR depth map \( D_{HR} \in \mathbb{R}^{sM \times sN} \) from its LR counterpart \( D_{LR} \in \mathbb{R}^{M \times N} \). Scaling factors \( s \) of 2, 4, or 8 were commonly used in previous studies. Instead of supplying \( D_{LR} \) directly as the network input, we preprocess it with bicubic interpolation to reach the target resolution \( D_{bic} \in \mathbb{R}^{sM \times sN} \). Such a step of preprocessing dramatically reduces the computational burden for network training and helps relax the constraint on the input size. When \( D_{LR} \) is unreliable, we transform the RGB color image to a YCrCb image and only feed the Y channel \( Y_{HR} \) into the network. The
The final objective of network learning is to find a set of optimal parameters \( \Theta = \{W, B\} \) such that the following loss function

\[
L(\Theta) = \frac{1}{2n} \sum_{i=1}^{n} \| f(\Theta, D_{\text{bic}}^i, Y_{\text{HR}}^i) - D_{\text{HR}}^i \|_2^2, \tag{4.1}
\]

is minimized, where \( f(\cdot) \) estimate the HR depth map based on \( W, B \), i.e., weights and biases, and \( n \) is the total number of the training samples.

Figure 4.2: Network illustrations. LR depth map and HR color image are the inputs of DEN and CBPN. Auto-merging takes the outputs of these networks to estimate the missing high-frequency component and the noise. The final output of this network is the reconstructed HR depth map.

### 4.2.1 Depth Enhancement Network

Deep CNN is one of the most commonly used architectures in the literature of deep learning. It has shown state-of-the-art performance in various vision tasks including image SR. In Fig. 4.2 we have shown the construction of our own DEN using 10 convolutional layers along with a rectified linear unit (ReLU) [63] after each layer. We opt to keep our network as a 10-layer network for the purpose of balancing output accuracy and training resources. It is worth noting that more convolutional layers can be used to boost performance, at the price of increased computational complexity.

For each convolutional layer, we use an array of \( 3 \times 3 \) kernels to generate 64 feature maps. We then pass the feature maps through the ReLU activation function. In order to keep the...
feature maps the same size as the inputs, zero-padding is employed at each layer. Therefore, each pixel in the feature maps generated by the 10th layer has a receptive field of $21 \times 21$ pixels in the input depth map. After the network construction, bicubic-interpolated and ground-truth depth maps are used as inputs and outputs to train the nonlinear mapping. Note that in DEN, only high-frequency components are learned from the LR depth map because the bicubic-interpolated depth map will be fed forward to the final reconstruction step, as shown in Fig. 4.2.

4.2.2 Color-based Prediction Network

By comparing edges in Fig. 4.1b with Fig. 4.1d, we can use the differences between them complementarily. Depth values in Fig. 4.1d are much more reliable than blind guesses from a color image. On the other hand, edge locations in Fig. 4.1b are relatively more reliable than the corrupted depth edge. Such mutual complementary characteristics certainly can enhance accuracy and sharpness around depth edges.

Inspired by [61], we utilize color images to predict the locations of depth discontinuities. To fairly evaluate the contribution of color images, we employ a network structure that is identical to DEN. As shown in Fig. 4.2, the only difference between CBPN and DEN is the input. During the experiments in Sec.4.3, $Y_{HR}$ is chosen as the input to CBPN to eliminate the effect of color variation. A set of 64-feature maps is generated for $Y_{HR}$, which contains the predicted high-frequency content. Note that the feature maps have the same spatial size as $Y_{HR}$ and $D_{bic}$. This allows for an easy auto-merging process to fuse the feature maps.

4.2.3 Auto-merging and reconstruction

Instead of designing a switching network that selects from two sets of feature maps (DEN and CBPN), we argue that it is better to allow the network to learn an automatic merging scheme. As shown in Fig. 4.2, we concatenate the feature maps generated by DEN and CBPN into a new set of 128-feature maps. One convolutional layer is applied with filter size of $3 \times 3$ to merge the feature maps. The number of feature maps after this layer is reduced by half.

The last component in our network is to reconstruct the HR depth map from concatenated
feature maps. One convolutional layer is employed to project feature maps onto the depth domain and generate the final depth map. $D_{bic}$ is then added with this depth image to finalize the result. By adding $D_{bic}$ and minimizing the loss function (4.1), we explicitly force the network to learn the missing high-frequency components and suppress the unwanted noise.

<table>
<thead>
<tr>
<th></th>
<th>Art</th>
<th>Books</th>
<th>Moebius</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicubic</td>
<td>6.62</td>
<td>14.80</td>
<td>30.45</td>
</tr>
<tr>
<td>Park [14]</td>
<td>8.03</td>
<td>12.24</td>
<td>17.42</td>
</tr>
<tr>
<td>DEN</td>
<td>0.57</td>
<td>3.68</td>
<td>16.92</td>
</tr>
<tr>
<td>DEN + CBPN</td>
<td>0.58</td>
<td>4.15</td>
<td>12.94</td>
</tr>
</tbody>
</table>

Table 4.1: Quantitative evaluation when input low-resolution depth maps are clean. We compare DEN and DEN + CBPN with several state-of-the-art methods. The mean-square-error (MSE) results on Middlebury data sets are compared. The best result is bolded and the second best is underlined.

4.2.4 Training

During the training process, we learn nonlinear mapping using stochastic gradient decent [64] with momentum set to 0.9. We randomly initialize the weights and train the model from scratch. The learning rate is set to be 0.01 and reduces to $1e^{-3}$ after 20 epochs. Gradient clipping is especially required for CBPN to avoid gradient explosion during back-propagation. Finally, to ensure low-sparsity constraint on the filters, we penalize all weights with an $\ell_2$ penalty. Thus the total loss function becomes

$$L = \frac{1}{2n} \sum_{i=1}^{n} \| f(\Theta, D_{bic}^i, Y_{HR}^i) - D_{HR}^i \|_2^2 + \sum_{t=1}^{T} \lambda \| w^t \|_2,$$

(4.2)
where \( T \) is the total number of filters and the regularization parameter \( \lambda \) is \( 1e^{-4} \).

<table>
<thead>
<tr>
<th></th>
<th>Art ( \times 2 ) ( \times 4 ) ( \times 8 )</th>
<th>Books ( \times 2 ) ( \times 4 ) ( \times 8 )</th>
<th>Moebius ( \times 2 ) ( \times 4 ) ( \times 8 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicubic</td>
<td>30.18 38.52 54.19</td>
<td>24.69 26.43 28.79</td>
<td>24.38 25.71 27.80</td>
</tr>
<tr>
<td>Ferstl [18]</td>
<td>10.16 16.50 25.84</td>
<td>2.38 4.90 6.08</td>
<td>2.17 4.12 6.65</td>
</tr>
<tr>
<td>DEN</td>
<td>4.13 10.22 28.45</td>
<td>1.38 3.30 7.15</td>
<td>1.56 3.63 7.73</td>
</tr>
<tr>
<td>DEN + CBPN</td>
<td>4.13 9.44 23.31</td>
<td>1.42 3.26 7.05</td>
<td>1.43 3.25 6.71</td>
</tr>
</tbody>
</table>

Table 4.2: Quantitative evaluation when input low-resolution depth maps are corrupted by noise. We compare DEN and DEN + CBPN with several state-of-the-art methods. The mean-square-error (MSE) results on Middlebury data sets are compared. The best result is bolded and the second best is underlined.

4.3 Experiment

We evaluate the performance of the proposed network on widely-used Middlebury Stereo data sets [65] [66] [67]. Thirty-five subjects are obtained from the Middlebury 2001 - 2006 data sets and 32 of them are used as the training set. There are two depth maps provided for each subject along with their corresponding color images. We extract sub-images with size \( 44 \times 44 \) from these 64 depth maps. Data augmentation (e.g., flip and rotate) is used to expand our training set. To test the performance of our method, trained nonlinear mapping is applied to three test images: Art, Books, and Moebius.

4.3.1 Benchmark Comparison

First, we present a quantitative evaluation of the proposed method; Nonlinear mapping learned by DEN+CBPN are compared with several state-of-the-art methods [14] [13] [18] [9].
We set up two experiments, noise-free depth map SR and noisy depth map SR, to demonstrate the ability of our proposed method. SR factors of \( \times 2, \times 4, \) and \( \times 8 \) are considered, and mean-square-error (MSE) is employed as the performance metric. Noise-free LR depth maps are generated by down-sampling the HR depth maps with bicubic interpolation, and noisy LR depth maps are created by adding Gaussian noise after down-sampling. The quantitative comparison results of these experiments are shown in Table 4.1 and Table 4.2.

To demonstrate the strength of deep neural networks, we train the nonlinear mapping solely with DEN in comparison with DEN+CBPN. The SR depth maps generated by DEN are also shown in Table 4.1 and Table 4.2 to compare with aforementioned depth SR methods. Note that all compared methods are color-guided and that ours is the only learning-based approach. We can easily observe that DEN outperforms previous methods by a large margin when the LR depth maps are noise-free. This implies that the nonlinear mapping learned by DEN is capable of restoring most of the missing high-frequency components accurately. In the scenario that LR depth maps are corrupted by noise, DEN still outperforms [14] [13] [9] when SR factors are \( \times 2 \) and \( \times 4 \), and achieves the state-of-the-art performance when SR factor is \( \times 8 \).

In addition to quantitative comparisons, we demonstrate the ability of our proposed method qualitatively. Visual examples of Art, Books, and Moebius are presented. The input low-resolution depth maps are created by downsample the ground-truths with a factor of \( \times 8 \) and add noise; These inputs along with their ground-truths are shown in Fig. 4.3. The high-resolution depth maps estimated by our proposed method and our comparing methods are shown in Fig. 4.4, Fig. 4.5, and Fig. 4.6.

4.3.2 How much can color help?

Finally, we compare the performance of color-guided neural network with Depth Enhancement Network. We mainly compare the performance between DEN and DEN + CBPN to illustrate the benefits of including color images in depth map SR. We note that when training DEN + CBPN jointly, parameters learned in DEN are not completely equivalent to those learned in solo DEN. This is mainly due to the fact that two networks can cooperate with each other to achieve optimal accuracy.
For smaller SR factors, \( \times 2 \) and \( \times 4 \), including color image contributes effects benefit when LR depth maps are noise-free. This is consistent with our analysis on Fig. 4.1. When the interference from noise is absent, color images are relatively unreliable. As a result, including color images at these scenarios only “confuses” the network and reduces the learning accuracy of the network. For larger SR factors, \( \times 8 \), depth edges are significantly distorted. Thus, depth edges predicted by color images are more reliable, leading to a superior performance. Similar observations can be made when LR depth maps are corrupted with noise. When the depth edges become unreliable, our network tends to rely on CBPN for restoring more accurate depth edges. Therefore, contribution of color image increases when the reliability of the LR depth map decreases (e.g., as noise gets stronger).

4.4 Summary

Since the quality of depth maps produced by Time-of-Flight (TOF) cameras is low, color-guided recovery methods have been proposed to increase spatial resolution and suppress unwanted noise. Despite successful applications of deep neural networks in color image super-resolution (SR), their potential for depth map SR is largely unknown. In this Chapter, we have taken a learning-based approach for color-guided depth map super-resolution and present a deep neural network architecture to learn the end-to-end mapping between low-resolution and high-resolution depth maps [60]. Furthermore, we introduce a novel color-guided deep Fully Convolutional Network (FCN) and propose to jointly learn two nonlinear mapping functions (color-to-depth and LR-to-HR) in the presence of noise. Experimental results on several benchmark data sets show that our method outperforms several existing state-of-the-art depth SR algorithms. Moreover, this work attempts to partially shed some light onto the fundamental question in color-guided depth recovery — how much can color help in depth SR?
Figure 4.3: (a), (d), and (g) are the Y-luminance channels of *Art*, *Books*, and *Moebius*; (b), (e), and (h) are the high-resolution ground-truths; (c), (f), and (i) are the low-resolution input depth-maps that corrupted by noise.
Figure 4.4: Subjective quality comparison of ×8 upsampling of Art with added noise. (a) result produced by Yang [9], (b) generated by Park [14], (c) generated by Diebel [13], (d) generated by Ferstl [18], (e) Our result produced by the DEN network, (f) Our result Produced by the DEN+CBPN.
Figure 4.5: Subjective quality comparison of ×8 upsampling of *Books* with added noise. (a) result produced by Yang [9], (b) generated by Park [14], (c) generated by Diebel [13], (d) generated by Ferstl [18], (e) Our result produced by the DEN network, (f) Our result Produced by the DEN+CBPN.
Figure 4.6: Subjective quality comparison of ×8 upsampling of *Moebius* with added noise. (a) result produced by Yang [9], (b) generated by Park [14], (c) generated by Diebel [13], (d) generated by Ferstl [18], (e) Our result produced by the DEN network, (f) Our result Produced by the DEN+CBPN.
Chapter 5

Nonlinear Image Interpolation via Deep Neural Network

5.1 Introduction

Image interpolation is a problem of enhancing the spatial resolution of an image by guessing missing pixels. It is sometimes known as digital zooming which be viewed as a computational alternative to the more expensive optical zooming. Due to the ill-posed nature of image interpolation problem, the objective is to come up with the best possible approximations - i.e., the reconstructed high-resolution (HR) image should be as close to the ground-truth as possible or visually appear pleasant in the absence of ground-truth. Image interpolation algorithms have been adopted in various applications including digital zooming, image resizing, image compression etc.. The major difference between image interpolation and image SR is that image interpolation only reverses the downsample operation and regards the PSF kernel as a Dirac Delta function in the image acquiring model (please refer to Fig.2.1.) Thus, low-resolution images modeled in the image interpolation problem are often considered as the directly downsampled version of their high-resolution counterparts. If the downsample pixels of low-resolution image exceeds the Nyquist sampling limit, the aliasing artifacts are commonly presented in both low-resolution images and reconstructed high-resolution images. Accordingly, the de-aliasing ability of different algorithms play significant roles during the upsampling process.
Many image interpolation techniques have been developed over the last decades. Roughly speaking, we can classify these techniques into two categories: non-adaptive methods and adaptive methods. Conventional linear non-adaptive interpolation methods such as cubic interpolation [68] [69] and bilinear interpolation [70] still remain active in many modern applications. The major advantages are their simplicity to implement and speediness to compute. However, the output images often suffer from severe inherent artifacts including block effects and distorted details around edges. The main reason is the lack of the ability to adapt local structures (i.e. edges) during the upsampling process. Consequently, algorithms in conjunction with proper utilization of nearby latent structures can significantly improve the performance of image interpolation.

To adapt local properties, non-linear interpolation algorithms [19,20,22,71–75] have been proposed to retain sharp edges by adapting local structures. Li and Orchard [19] proposed to adapt edge orientations without specifying their directions by investigating the geometric duality between low-resolution covariance and high-resolution covariance, and proposed a winner-filtering like scheme to interpolate pixels along edge directions (NEDI). Zhang and Wu [20] proposed to interpolate a missing pixel in two directions, 45° and 135°, and then fuse the directional interpolation results together by a linear minimum mean square error estimator (LMMSE). Zhang and Wu [22] proposed the soft-decision adaptive interpolation (SAI) that extended the idea of [19]. They constrained the statistic consistency within the octagonal block region which comprised both the low-resolution pixels and the estimated high-resolution pixels; three autoregressive models were solved systematically to finalize the missing high-frequency pixels. Despite their success in preserving local structure, these methods share one common drawback: the lack of utilizing non-local similarity structure.

To overcome the limitation of local-structure based approaches, Dong et al. [26] introduced a sparse representation based interpolation scheme. They proposed a Nonlocal Autoregressive Model (NARM) which modeled a given pixel as the linear combination of its nonlocal neighboring pixels. Hence, the image nonlocal similarity propriety was seamlessly embedded into the data fidelity term. In addition, nonlocal similarity was also utilized to regularize the estimated sparse codes for each patch cluster. As a result, their method successfully reduced the coherence between the sampling matrix and the sparse representa-
tion dictionary, and generalized the image interpolation problem into a conventional sparse representation problem. In spite of their excellent interpolation performance, one critical drawback of the proposed method is the computation complexity. This bottleneck is mainly caused by the operation of searching nonlocal similarity patches for each query patch and by the iterative minimization process.

In this chapter, instead of manually deriving a local or nonlocal autoregressive model, we treat the image interpolation problem as a learning based discriminative regression problem, i.e. directly estimate missing pixels by neighboring pixels [76]. Specifically, we proposed to directly learn a discriminative model using a Convolutional Neural Network (CNN) [44]. The reasons for using CNN are as follows: first, CNN has shown outstanding performance in many vision-related applications due to its effectiveness and flexibility for exploiting natural image characteristics; a second, considerable amount of researches have been devoted to increase the performance of neural networks, including Rectifier Linear Unit (ReLU) [77], batch normalization [45] etc., which can be adopted to increase our performance; third, the interpolation process can be executed by a modern power Graphics Processing Unit (GPU) which Substantially reduces the computational time.

Our proposed Deep Interpolation Network (DIN) aims to estimate unknown high-resolution pixels from known low-resolution pixels; hence, DIN can be interpreted as a discriminative regressor. Specifically, pairs of low-resolution patches and their corresponding high-resolution pixels are fed into the network the during training phase. The goal is to find all the hyperparameters in the network that properly respond to numerous patch structures. Meanwhile, a nonlocal self-similarity property is planted; patches with similar edge orientation/structures are trained to activate identical sets of neurons. High-resolution pixels are then estimated based on the available low-resolution pixels (indicating local structure) and the pre-trained hyperparameters (based on nonlocal structure). Therefore, the conventional image interpolation problem is now converted into the challenges of designing proper network architecture that is capable of handling various structures and generating suitable training pairs that captures all desired structures.

Extensive Experiments show that our proposed DIN produces better interpolation results than local edge orientated methods (NEDI and SAI) and achieves state-of-the-art perfor-
mance with reduced computation time. Moreover, comparison with high-frequency learning based networks are reported to demonstrate the benefits of direct-estimation in the context of image interpolation. The rest of this chapter is organized as follows: Section 5.2 provides the related work. Section 5.3 presents our proposed Deep Interpolation Network. Experimental results and discussion are presented in Section 5.4. We summarize this chapter in Section 5.5.

5.2 Related Work

5.2.1 Neural Networks for Low-Level Vision Tasks

Recently, Deep Neural Networks have led to very successful results in the context of low-level vision. Dong et al. [31] introduced a CNN based super-resolution scheme (SRCNN); a three layer network architecture was utilized to directly estimate high-resolution images. This network architecture was re-designed in [78] to achieve real-time performance. Most importantly, their work inspired numerous deep learning based super-resolution approaches including [79], [80], [81], etc. In [82], deep neural networking was introduced to handle the depth map super-resolution problem, and in [83], a neural network based approach was proposed to handle the image denoising problem. One key observation made by some of these works is that better performance can be achieved by learning the residual noise in the case of image denoising and the high-frequency component in the case of image super-resolution, instead of directly estimating the ground-truth. However, such an observation does not hold in the context of image interpolation. In our observation, estimating the high-frequency component yields unsatisfied results especially around edges. We compare and discuss our proposed direct learning network with a deep residual network in Section 5.4.

5.2.2 Neural Networks for Image Interpolation

Some early work has been done to utilize neural networks for the image interpolation problem. Plaziac [84] introduced the one layer feedforward network to estimate missing high-resolution pixels. This work was later extended by Go et al. [85] with a three-layer
feedforward network to interpolate color pixels. Hu et al. [86] proposed a classification based neural network approach for image interpolation; they pre-classified low-resolution pixels into different classes and trained separate networks for each class. Recently, Chen et al. [87] proposed a saliency-directed interpolation scheme; they adopted a neural network to classify edges into eight orientations to guide the final interpolation process. In this work, we focus on the design and utilization of CNN. To the best of our knowledge, there is no work which investigates CNN for image interpolation at the time of writing this dissertation.

5.3 Deep Interpolation Network

In this section, we present the proposed Image interpolation CNN architecture, i.e., DIN. We first introduce some notations and the problem formulation that are necessary for the description of DIN. Then, a detailed network structure is presented and discussed. Finally, we discuss an interpretation of our trained model from an edge-directed point of view.

5.3.1 Problem Set-Up

For image interpolation, it is usually assumed that the low-resolution (LR) images are directly downsampled from high-resolution (HR) images. Let us denote \( y \in \mathbb{R}^{M_1 \times M_2} \) as the LR image and \( x \in \mathbb{R}^{S_1 \times S_2} \) as the HR image, where \( M_1 = S_1 / 2 \) and \( M_2 = S_2 / 2 \) if the scaling factor equals 2. Fig. 5.1 illustrates a formation of generating the LR image from the HR image. Let \( y_{i,j} \) and \( x_{i,j} \) be the LR and the HR pixels, respectively. The goal of interpolation is to estimate \( x_{i,j} \) from all available \( y_{i,j} \).

Conventional methods often model \( x_{i,j} \) by its surrounding pixels, e.g. \( x_{i,j} = \sum_{(k,l) \in T} \alpha_{k,l} y_{k,l} \), where \( T \) is a spatial template (nearby LR pixels of \( x_{i,j} \)) for the interpolation operation. This is a linear autoregressive model where \( \alpha \) indicates the linear coefficients that controls the amount of contribution from the surrounding pixels. Mathematically, we can define the image interpolation problem as

\[
\hat{\alpha} = \arg\min_{\alpha} \| x_{i,j} - \sum_{(k,l) \in T} \alpha_{k,l} y_{k,l} \|_2^2 + \lambda R(\alpha),
\]

(5.1)
Figure 5.1: A formation of acquiring LR pixels from a HR image by direct downsampling. The solid dots are the missing HR pixels and the circles are the LR pixels. Blank dots and gray dots separate HR pixels geometrically of which 4 neighbors are associate with blank dots at diagonal direction and 2 neighbors are associate with gray dots at vertical or horizontal direction.

where \( \| \cdot \|_p \) denotes \( \ell_p \) norm (\( p \) is usually equals 2) and \( \mathcal{R} \) captures the constraints one wishes to enforce. Note that in Fig. 5.1, there are four diagonal neighbors associated with the black dots, and only two neighbors are available for the gray dots. It is very common to interpolate the black dots first and then interpolate the gray dots with respect to the estimated black dots. We illustrate this concept of exploiting the local information by limiting (5.1) to the fourth-order linear system in Fig. 5.2.

In order to obtain accurate HR pixels, \( \alpha \) has to capture the geometry structure insinuated by the available LR pixels, i.e. enforce the geometric duality. The aforementioned linear model is capable of achieving this requirement with some limits, but also has two major drawbacks. On one hand, in natural images, geometric structures are very complicated; therefore, they are insufficiently represented by linear models. On the other hand, different geometric structures should be treated separately and accordingly, e.g., edge models and texture models. To overcome these limitations, let’s relax the assumption of the linear model by an arbitrary function \( f(\cdot) \). (5.1) now becomes

\[
    f^*(\phi) = \arg \min_{f(\phi)} \| x_{i,j} - f(\phi, y_{k,l \in T}) \|_p^2 + \lambda \mathcal{R}(\phi),
\]  

(5.2)
Figure 5.2: Illustration of interpolate HR pixels by (5.1). (a), (b), and (c) indicate three geometry configurations of HR pixels and their linear coefficients associated with them. We name these three types of HR pixel as quincunx, vertical, and horizontal.

Where \( \phi \) is some parameters that are associated with function \( f(\cdot) \). Ideally, we would like this function to be flexible so that it adapts to the input LR structures, and it doesn’t necessarily have to be a linear function. Fortunately, we can approximate this function by neural networks. Thus, we decompose the image interpolation problem into two sub-problems: (i) design the neural network architecture and (ii) find the optimal parameters as

\[
\phi^* = \arg \min_{\phi} \|x_{i,j} - f(\phi, y_{k,l \in T})\|_p^2 + \lambda R(\phi).
\] (5.3)

### 5.3.2 Deep Interpolation Network

In this section, we introduce our CNN architecture, *i.e.*, Deep Interpolation Network. The input of our DIN is a LR patch \( y \). We set the size of \( y \) to be \( 4 \times 4 \) as demonstrated in Fig. 5.1 (impact of patch sizes will be discussed in Section 5.4); thus, 16 LR pixels are fed in to stimulate the network. Since the input volume is relatively small, we opt to keep our network as a 6-layer network (shallower deep neural network). As recommended by [88], we set the convolution filters to be \( 3 \times 3 \). Meanwhile, we pad the borders of the input volume spatially by one pixel with mirror-padding; the spatial size of the inputs is now \( 6 \times 6 \) which enforces the spatial size of the output of each layer to be \( 4 \times 4 \) (except the last layer). The rectified linear units (ReLU, \( \max(0, \cdot) \)) [77] are attached to each layer except the data layer
Figure 5.3: Geometry interpretation of how LR pixels contribute to the estimation of HR pixels. Red, blue and green dots represent the quincunx, vertical, and horizontal, respectively. Our network is capable of recognizing these subtle geometry variations and produce accurate HR pixels accordingly.

and the last layer. This simple operation is essential to the recognition of LR geometries. As the input data flow into the network, ReLU act as traffic lights which block the outputs of certain neuron from continuing; therefore, LR patches with similar geometries are forced to path through similar route. Mathematically, each convolutional layer can be expressed as follows:

\[ L_n = \max(0, L_{n-1} * W_n + b_n) \]  

(5.4)

The next component that needs to be defined is the output of our network. As shown in Fig. 5.2, three types of HR pixels, *i.e.*, quincunx, vertical, and horizontal, are associated to LR pixels with slightly varied geometry correlations. In contrast to the conventional methods that estimate quincunx pixels first, we argue that with carefully designed network and training data, they can be jointly estimated in a single network instead of three separate networks. As illustrated in Fig. 5.3, the trained network is expected to estimate them through separate nonlinear functions.

The last parameter we need to define is the number of feature maps produced by each layer. Although this number can vary for different layers, we fix it for reasons of simplicity. Theoretically, we would like 128-feature maps at each layer to be associated with one
5.3.3 Connection with State-of-the-Art Methods

The architecture shown in Fig. 5.3 can be used to directly estimate the HR pixels. It can be interpreted as an upgraded edge-directed interpolation when compared with NEDI [19] and SAI [22]. NEDI and SAI modeled the image interpolation problem as a linear system and aimed to estimate local statistics (covariance matrix) which implicitly evaluate the edge orientation. Similarly, DIN also considers the geometry orientation implicitly as the network responds differently to the input LR pixels. Furthermore, DIN is more flexible than NEDI and SAI. During the training phase, a significant amount of LR training patches containing numerous geometry structures are exposed to the network; DIN is trained to recognize these geometric variations and their corresponding local relationships to the HR pixels. Therefore, the specific estimator is tailored for each geometric structure, and accurate HR pixels are then generated accordingly.

As we mentioned before, our DIN also utilizes the image nonlocal self-similarity property as introduced in NARM [26]. Conventionally, the nonlocal property is often exploited by clustering similar patches; a patch can be represented as a combination of its similar-patch cluster. We can interpret DIN in a similar fashion. When preparing training data, we decom-
pose training images into patches of desired sizes. As similar patches flow into the network, they are trained to activate similar sets of neurons which leads to similar network responses. Furthermore, as more data are fed into the network, self-similarity are not necessarily constrained on one training image; similar patches from other images also contribute to help the network in recognizing the structure (similar idea of utilizing external image database for image denoising is proposed in [89]).

5.4 Experimental Results

5.4.1 Experiment details

Training/Testing Dataset

For the training dataset, we follow [31] to use 91 images of various sizes. When training our network, we extract image patches with sizes of 7 by 7 as the ground-truthes and generate the corresponding 4 by 4 input patches. Totally, we extract roughly 320,000 patches from these images as our training patches. Since the input volume of the network is relatively small, we can afford a larger training dataset by applying data augmentation (e.g. flip and rotate). We generally observe a increased performance by increasing the number of training patches.

To test the performance of our learned discriminative model and for the sake of comparison with existing algorithms, the Set5 and Set14 datasets are used for the evaluation process. All testing images are directly downsampled from their HR counterparts, and their evaluation results are shown in Section 5.4.3.

Network Training

During the training process, we randomly initialize the weights and train our model from scratch. We train our network using gradient descent (SGD) [44] with respect to Eqn. 5.3. The momentum of SGD is set to 0.9, and weight decay is set to 0.0001 for each convolutional layer. The initial learning rate is 0.01 in conjunction with gradient clipping to avoid gradient explosion during back-propagation, and it decays by a factor of 10 after every 20 epochs.
In our experiment, we do not define any finetuning stage, however, an additional finetuning stage could boost the performance slightly.

5.4.2 Comparing Methods

We compare the proposed DIN method with aforementioned state-of-the-art methods, including two edge-directed methods, NEDI [19] and SAI [22], and the nonlocal sparse-representing method NARM [26]. We also compare our DIN with a 15-layer deep CNN which is designed to estimate the missing high-frequency components [58]; we optimized this network by applying Batch Normalization (BN) [45] which increases the performance slightly; we name this network VDSR_BN.

5.4.3 Quantitative and Qualitative Evaluations

The proposed network is trained and tested, and the detailed PSNR results of different methods on Set5 and Set14 are shown in Table 5.1 and Table 5.2, respectively. The last column of both tables summarizes the average performance of each algorithm. As shown in these tables, DIN achieves and outperforms the state-of-the-art algorithms in both testing datasets. Specifically, the averaged results show that DIN outperforms NARM, the current state-of-the-art method, by 0.12dB on both datasets. Fig.5.6, Fig.5.7, and Fig.5.8 demonstrates the visual results of different methods. As aliasing is created due to directly downsampling, our network reacts upon the incoming LR structure and attempts to reconnect edges. Hence, DIN produces more visual pleasant results.

Our experimental results also raise attention to the fact that a better understanding of local structure leads to better interpolation performance. As researchers gain more knowledge about the intrinsic properties of natural images, more constrains are applied to advance image interpolation algorithms. In Table.5.1 and Table.5.2, we can clearly observe the boosted performance from NEDI to SAI by iteratively reinforcing the local constraints and the gain from SAI to NARM via adopting non-local similarity property. Following the trend, it is intuitive to understanding the performance gain of our DIN thanks to the memorization property of artificial neural networks. With more similar structures that stimulate the net-
work rather than a few non-local similar patches, our DIN is capable of differentiating subtle geometry variations in the input patch and estimates accurate pixel value correspondingly.

In addition to comparing with conventional model-based approaches, we compared DIN with VDSR_BN. The essential difference is that DIN learns a discriminative model which directly estimates missing pixels rather than refining an upsampled image with estimated high-frequency components. As shown in the experimental results, we have verified that DIN outperforms VDSR_BN with a margin of nearly 0.3dB on both datasets.

<table>
<thead>
<tr>
<th>Image</th>
<th>Baby</th>
<th>Butterfly</th>
<th>Head</th>
<th>Bird</th>
<th>Woman</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEDI [19]</td>
<td>34.79</td>
<td>26.93</td>
<td>32.69</td>
<td>35.55</td>
<td>31.53</td>
</tr>
<tr>
<td>SAI [22]</td>
<td>35.49</td>
<td>27.96</td>
<td>32.76</td>
<td>36.99</td>
<td>32.26</td>
</tr>
<tr>
<td>NARM [26]</td>
<td>35.32</td>
<td>29.20</td>
<td><strong>33.01</strong></td>
<td>37.29</td>
<td><strong>32.80</strong></td>
</tr>
<tr>
<td>VDSR [58]</td>
<td>35.21</td>
<td>28.67</td>
<td>32.94</td>
<td>37.38</td>
<td>32.38</td>
</tr>
<tr>
<td>DIN</td>
<td><strong>35.50</strong></td>
<td><strong>29.38</strong></td>
<td>32.91</td>
<td><strong>37.68</strong></td>
<td><strong>32.73</strong></td>
</tr>
</tbody>
</table>

Table 5.1: Quantitative evaluation on Set5 dataset. The mean-square-error (PSNR) results are presented. The best result is in bold and the second best is underlined.

<table>
<thead>
<tr>
<th>Image</th>
<th>Barbara</th>
<th>Flowers</th>
<th>Foreman</th>
<th>Lenna</th>
<th>Zebra</th>
<th>pepper</th>
<th>monarch</th>
<th>comic</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEDI [19]</td>
<td>23.56</td>
<td>28.52</td>
<td>35.39</td>
<td>32.95</td>
<td>28.73</td>
<td>33.34</td>
<td>31.68</td>
<td>24.39</td>
</tr>
<tr>
<td>SAI [22]</td>
<td>24.92</td>
<td>29.37</td>
<td>36.28</td>
<td>33.55</td>
<td>29.62</td>
<td>33.54</td>
<td>32.89</td>
<td><strong>25.16</strong></td>
</tr>
<tr>
<td>NARM [26]</td>
<td>25.05</td>
<td>29.56</td>
<td><strong>37.04</strong></td>
<td><strong>33.92</strong></td>
<td><strong>30.58</strong></td>
<td>34.00</td>
<td>33.93</td>
<td>25.05</td>
</tr>
<tr>
<td>VDSR [58]</td>
<td><strong>26.29</strong></td>
<td>29.36</td>
<td>36.65</td>
<td>33.45</td>
<td>29.61</td>
<td>33.90</td>
<td>33.50</td>
<td>25.06</td>
</tr>
<tr>
<td>DIN</td>
<td>25.30</td>
<td><strong>29.75</strong></td>
<td>36.84</td>
<td>33.72</td>
<td>30.30</td>
<td><strong>34.00</strong></td>
<td><strong>34.01</strong></td>
<td><strong>25.41</strong></td>
</tr>
</tbody>
</table>

Table 5.2: Quantitative evaluation on Set14 dataset. The mean-square-error (PSNR) results are presented. The best result is in bold and the second best is underlined.
Figure 5.5: (a) Run time comparison of state-of-the-art methods on Set5, (b) Run time comparison of state-of-the-art methods on Set14. PSNRs shown are averaged PSNR of each algorithm.

5.5 Summary

Machine learning methods for single-image super-resolution have been attracting more attention due to their outstanding performance. In this chapter, we extend learning based methods into image interpolation by investigating different architectures of Deep Neural Networks. In literature, low-resolution images in the image interpolation problem are usually modeled as a directly downsampled version (i.e., without blurring) of their high-resolution counterparts of which structure distortion and jaggy/ringing artifacts are often presented. Therefore, algorithms designed or learned to estimate missing high-frequency components may not be useful for image interpolation. To overcome this difficulty, we have proposed a neural network based estimator to directly estimate unknown pixels. Such a pixel-level estimator allows the network to better adapt for local structure details. Specifically, geometry duality are retained during interpolation which resolves ambiguity. Moreover, we compare our method with high-frequency domain learning networks to verify that the proposed deep neural network architectures can effectively restores edge structures and suppress artifacts.
Quantitative results also report that our method not only achieves state-of-the-art performance in terms of PSNR and SSIM, but also significantly reduces computation time by benefiting from GPU computing.
Figure 5.6: Interpolation results on test image baby.
Figure 5.7: Interpolation results on test image *butterfly*. 
Figure 5.8: Interpolation results on test image *flowers*.
Chapter 6

Deep Progressive Zooming Network

6.1 Introduction

Natural image super-resolution (SR) is the most researched sub-problem of image super-resolution. It is a popular yet very challenging topic with high industrial and academic potential. Inherited from image super-resolution, natural image SR is a technique that aims to increase the number of pixels that represent a natural image. Typically, when taking photos with everyday cameras, natural images are stored in three separate channels: red, green, and blue (hence the designation RGB images). In general, under the context of single image super-resolution (SISR), a single LR natural image is provided as the input to estimate an HR output where R, G, and B channels require simultaneous processing. Due to the ill-posed nature of image SR, there is a large set of image candidates available in the HR space. It is challenge to retrieve the best images that satisfies human perception and judgments.

Differing from the image interpolation problem we discussed in Chapter 5, natural image SR considers the PSF $B$ described in Fig.2.1. As a result, the pixels that represent LR natural images are no longer considered ground-truthes. Thus, every pixel in the HR space has to be coherently estimated to reach the best quality. Moreover, due to the $B$ kernel in natural image modeling, some high-frequency information is completely removed from LR natural images. This behavior makes natural image super-resolution a distinct yet more challenging problem than image interpolation.
6.1.1 Literature Review

There are many existing natural image super-resolution techniques in the literature. At an early stage, state-of-the-art methods such as reconstruction methods [90,91] and statistical methods [92] draw much attentions. Nowadays, we can generalize state-of-the-art natural image super-resolution techniques into four categories: sparse representation [27, 93–95], neighbor embedding [96,97], self-exemplar [28,98–100], and deep learning [31,58,59,101–103] -based methods. These four types of methods tackle the natural image super-resolution problem from different perspectives. As a result, different natural image priors have been utilized to estimate high quality HR natural images.

The sparse representation based natural image SR technique was first introduced in [27], from the realization that a natural image can be represented by only a few atoms from a well-learned dictionary. Hence, both LR and HR images could be represented by two sets of dictionaries. In this work, authors demonstrated that under mild conditions, the sparse representation/codes could be correctly recovered from the LR dictionary. By jointly training the two dictionaries together, LR and HR images were forced to be represented by a same set of codes. Therefore, when applying this technique to natural image super-resolution task, one could easily calculate the sparse codes of the LR images based on the learned LR dictionary and then apply these codes on HR dictionary to generate HR images. This idea was quickly adopted by the community and became state-of-the-art and inspired numerous research works afterward.

The nearest neighbor embedding based SR method [96] adopted the philosophy of locally linear embedding [104]. This work assumed that small image patches in the LR and HR images formed manifolds with local geometry in LR and HR spaces. Therefore, by learning the manifold, the local geometry of the low-resolution patches could be projected onto the high-resolution space as a linear combination of several nearing neighbors. Such information allows the representation of more image patch patterns with a smaller set of training pairs which will reduce the computation burden significantly.

The philosophy of local self-examples was proposed for natural image super-resolution in [28]. The key idea of this work was to explore internal patch-wise redundancy. In natural
images, many patches recur within and across scales of neighboring spaces. It was intu-
itive to use these redundancies, additional prior in a sense, to reconstruct a high-quality
HR image. Moreover, [28] modeled the image SR process as an additional result of missing
high-frequency details and low-frequency input content. The paired HF and LF information
was reconstructed gradually through smaller non-integer scaling factors. This simple for-
mulation progressively reconstructed HR images and produced excellent results. The local
self-similarity idea of this work was later extended in [100]. In this work, instead of searching
only for the translated similar patches of the query patch, they measured patches’ distances
after homography transformation and then wrapped the best matched patches onto the HR
space using the previously calculated homography matrix. This strategy of utilizing wrapped
similar patches breached the limitation of searching similar patches within a local window.
Therefore, similar patches could be found remotely (e.g. from an external database).

Generally, the aforementioned approaches model the natural image SR problem by utilizing
diverse image priors. In addition, the learning-based approaches have been introduced to
handle low-level vision problems. The deep learning based natural image super-resolution
method SRCNN [31] was developed and discussed. In this work, authors proposed to learn
end-to-end mapping across natural image resolutions directly, with the mapping represented
as a deep convolutional neural network (CNN). They proposed a deep three layer CNN that
mimicked a sparse code solver and successfully demonstrated the potential of deep CNN. Due
to recent advances of deep neural networks in the filed of artificial intelligence, many deep
learning platforms had become publicly available, and this work was studied and expanded.

Although these well-studied approaches utilized different priors to regulate this ill-posed
problem and to recover sharper and more visually pleasing results, there are some drawbacks
associated with these methods. Sparse representation based methods utilize the ”sparse”
prior of natural images, but the learning abilities of such a model are relatively low, and
the gap between the approximated HR image and the ground-truth is often large. Neighbor
Embedding based methods utilize the ”redundant” property of image patches. However, the
computational power required to find K-nearest-neighbors is a bottleneck, and it increases
proportionally to the number of K. Furthermore, the results of these methods often suf-
fer from blurring effects, due to over- or under-fitting. Self-exemplar based methods work
very well with repeated patterns, but their performance become unreliable when encountering non-structural images. Deep learning based method SRCNN failed to improve the performance significantly due to the difficulty of training at the time.

### 6.1.2 Proposed Architecture

As we can observe, the aforementioned SR methods explore different properties of natural images. It is intuitive to combine more properties to compensate for the drawbacks of each method. The anchored Neighborhood regression method [29] adopted both the concepts of sparse representation and neighboring embedding. Instead of using sparse representation as a global regression approach, Timofte et al. proposed localizing the LR and HR dictionaries within the K nearest neighbors. Therefore, more than one local projective matrix can be utilized, allowing more flexibility. Similarly, [58] adopted the idea of using CNN to represent the relationship between the resolutions and proposed a very deep (e.g. 20 layers) network. Furthermore, this work also embraced the idea in [28]. Instead of estimating the HR image directly, it was more robust to utilize low-frequency content as a base and to estimate high-frequency information accordingly. This work then became state-of-the-art, and the operation of estimating high-frequency information had become a standard process for deep learning based natural image super-resolution task.

In this chapter, we are interested in utilizing the similar intuition to further develop deep learning based approaches. Similar to Chapter 5, we treat natural image super-resolution problem as a discriminative regression problem, and we are estimating all pixels in HR space based on the LR inputs. Moreover, we proposed to follow and extend the idea explored in [31, 58] that is to directly learn a discriminative model based on a Convolutional Neural Network [44]. We call our network the Deep Progressive Zooming Network (DPZN). There are three key building blocks in our proposed architecture including convolution layer, convolution-transpose layer, and activation layer. Convolution layers are implemented to extract image features, learning the mapping, and reconstruct HR images. Convolution-transpose layers allow the network to modify the spatial dimensions of the input LR images. Therefore, the proposed DPZN is capable of projecting LR images onto HR space by the learned kernels
instead of completely relying on conventional Bicubic [69] interpolation algorithm. Unlike other deep learning based SR networks, DPZN utilizes *Parametric Rectified Linear Unit* (PReLU) [105] as the activation function since it helps alleviated the ”dead feature” problem caused by the ”vanishing” gradients of ReLU function.

Our proposed DPZN not only explores powerful learning representation abilities of deep CNN but also leverages domain expertise to ease the learning burden. Specifically, our network is separated into two sub-structures including low-frequency part and high-frequency part. Our DPZN employs Bicubic interpolation to handle low-frequency component estimation and allocates its learning power to estimate high-frequency component. Meanwhile, both components are estimated in a progressive fashion where large scaling factors are split into smaller non-integer factors. Therefore, the network gradually increases the quality of input LR images to partially alleviates the learning difficulties.

Additionally, we consider our DPZN as an exemplar-based learning network. Specific patch size is designed to fully utilizing convolution-transpose operation when estimating high-frequency component, and the final HR image is combined using the weighted averaging strategy. One advantage of such an architecture is that the progressively learned kernels for different patch sizes can be reused or re-trained within the network. We also analyze and discuss the contribution of various loss penalties to further improve the performance of our DPZN. Furthermore, we consider the natural image super-resolution problem from a practical application perspective. The formulation of our architecture performs digital zooming where more intermediate SR results are revealed.

This rest of this chapter is organized as follows: we formulation the natural image super-resolution problem and discuss our proposed architecture in section 6.2. The quantitative and qualitative evaluation results and discussion of our proposed architecture are presented in 6.3. We summarize this chapter in 6.4.

### 6.2 Deep Progressive Zooming Network

In this section, we first formulate the natural image SR problem and then discuss the motivation of our proposed deep architecture. Then, we will present the Deep Progressive
Zooming Network. Finally, we discuss the details of the progressive learning and the weight sharing strategy and conclude this section by comparing DPZN with existing progressive learning methods.

6.2.1 Problem Formulation

To help understand the proposed architecture, we recap the formulation of the image SR problem introduced in 2.1 and present our notations. Let us denote \( y \in \mathbb{R}^{M_{yr} \times M_{yc} \times 3} \) as the LR natural images with three separate channels (RGB image). The goal of natural image SR is to reach a higher resolution image \( x \in \mathbb{R}^{M_{xr} \times M_{xc} \times 3} \), where \( M_{yr} = M_{xr}/s \) and \( M_{yc} = M_{xc}/s \). Conventionally, scaling factor \( s \) equals to 2, 3, or powers of 2 are largely investigated. Mathematically, we can summarize this problem as follows:

\[
\hat{F}_s = \arg \min_{F_s} \| F_s(y) - x \|_p^2, \tag{6.1}
\]

\[
= \arg \min_{F_s} \left\| \sum_{ch=1}^{3} \left( F_{ch}^s(y) - x_{ch}^s \right) \right\|_p^2, \tag{6.2}
\]

where \( F_s(\cdot) \) denote the mapping function with respect to the scaling factor \( s \), and \( \| \cdot \|_p \) denotes \( l_p \) norm, where \( p \) generally equals 2 and reflects the Euclidean distance. Please note that in this formulation, the HR image of the R, G, and B channels are not estimated separately, but are approximated jointly through a single mapping operation.

To adapt (6.2) into our exemplar-based SR strategy, we focus on smaller image patches. Let us assume there are a total of \( K \) overlapping LR patches \( y_k \in \mathbb{R}^{m_{yr} \times m_{yc} \times 3} \) that can be extracted from a LR natural image. Correspondingly, there are a total of \( K \) overlapping HR patches \( x_k \in \mathbb{R}^{m_{xr} \times m_{xc} \times 3} \) that we aim to estimate. Meanwhile, \((m_{yr}, m_{yc}), (m_{xr}, m_{xc})\) still obeys the same spatial ratio \( s \). Thus, we have

\[
\hat{F}_s = \arg \min_{F_s} \left\| \sum_{ch=1}^{3} \sum_{k=1}^{K} \left( F_{ch}^s(y_k) - x_{ch}^s_k \right) \right\|_p^2, \tag{6.3}
\]

where now \( F_s \) only operates on patches instead of images.

Based on (6.3), the problem of natural image SR reduces to the problem of finding one or more appropriate mapping functions that are not only able to adopt different local structural variations but also approximate the ground-truthes with minimum error. Fortunately,
It has been proven that this mapping function can be learned using deep neural networks (e.g., CNN). Therefore, this problem is further reduced into two sub-problems: (i) designing the network architecture and (ii) learning the optimal parameters that represent the mapping function $F_s$. Combined with separate frequency contents learning, we show the formal mathematical formulation as follows:

$$\hat{F}_s(\Theta) = \arg \min_{F_s(\Theta)} \left\| \sum_{c_h=1}^{3} \sum_{k=1}^{K} [F_{c_h}^s(\Theta, y_k) - (x_k^{c_h} - G_{lf}(y_k^{c_h}))] \right\|^2_p + \lambda \mathcal{R}(\Theta), \tag{6.4}$$

Where $\Theta$ captures all the learnable parameters associated with the designed network architecture. Function $G_{lf}$ indicates the resolution projection method that estimates the low-frequency component. We choose Bicubic interpolation in our DPZN, but other basic interpolation mechanisms can also be utilized. Please note that the function $G_{lf}$ is applied to individual R, G, and B channels, which does not consider the correlations between channels. The function $\mathcal{R}$ applies the constraints one wishes to enforce. Generally, under the context deep neural network, learned parameters $\Theta$ are constrained to avoid energy concentration in kernels and to alleviate the over-fitting problem.

To tackle the two sub-problems above induced from natural image super-resolution, we first discuss the proposed deep architecture in the rest of this chapter. We then present our detailed learning procedure in section 6.3 for finding the optimal parameters.

### 6.2.2 Motivation Behind the Proposed Architecture

With the recent development of deep learning platforms, learning barriers, such as computational and memory cost, have been greatly reduced. Meanwhile, the flexibility of deep neural networks to approximate any nonlinear mapping function has been proven to achieve superior reconstruction results compared with many previously designed model-based methods. To fully utilize the learning power of deep CNN, we propose a simple and intuitive architecture to handle natural image resolution changes. The motivations of our proposed architecture are threefold: (i) the efficiency of learning upscaling filters within the network is confirmed in [103]; (ii) digital zooming in the real world requires more resolution results than a few fixed scaling factors; (iii) we made an observation that the quality of the input LR images impacts the learning abilities significantly.
Figure 6.1: Illustration of model performance when the quality of the input LR image varies. Figure (a), (b), (c), and (d) are samples of four different quality inputs, where (a) represents the lowest quality and (d) demonstrates the highest quality. Figure (e) is the targeted HR groundtruthes. The scaling factor is $\frac{8}{7}$. Figure (f) shows the training losses associated with these four sets of LR input images.

In recent deep learning based natural image SR algorithms (e.g. [58, 106]), the resolution of the input LR images is commonly increased before the first layer of the network. However, there are some drawbacks associated with this approach. Firstly, the computation complexity increases. Secondly, simple interpolation methods do not introduce any additional information to ease learning difficulties. To handle these drawbacks, the convolution-
transpose operation is introduced in [107] and the importance and efficiency of such an in-network upscaling operation is discussed in [103]. Thus, we are motivated to learn upscaling filters within our architecture. Nevertheless, we cannot directly apply a full image convolution-transpose operation due to the progressive nature of our architecture. Therefore, a patch-based convolution-transpose operation is introduced to learn the image upscaling.

Meanwhile, we are inspired by real-world applications such as digital zooming. Often, when performing digital zooming, higher resolution images with small or non-integer scaling factors are in demand. However, models that are designed and learned for integer scaling factors, such as \( \times 2, \times 3, \) and \( \times 4 \), fail to produce any intermediate resolution results. Therefore, we are motivated to design a deep architecture that not only generates integer SR results but also performs digital zooming.

Furthermore, we have made an observation that the quality of the input LR images impacts the performance of learned models significantly. To illustrate this observation, we designed a five-layers deep network to learn the mapping between resolutions. As shown in Fig.6.1, four sets of LR images with the same spatial dimensions are presented, and their qualities vary from low to high. In this network, the spatial dimension changes from Fig.6.1(a, b, c, d) to Fig.6.1(e) are handled by a convolution-transpose layer. Fig.6.1(f) records the training losses associated with them. As we can easily observe, the training losses are reduced in proportion to the quality of LR images. Therefore, it is intuitive to improve the quality of the input LR images before feeding them into networks.

### 6.2.3 Deep Progressive Zooming Network

In this section, we present the proposed neutral image super-resolution architecture (i.e. DPZN). Combined with our motivations and intuitions, we aim to design a network that (i) handles the spatial dimensions changes within the network, (ii) generates intermediate resolution results, and (iii) improves the quality of the input LR images to alleviate learning difficulties. To better describe our proposed DPZN, let us only focus on the scaling factor \( \times 2 \) for the moment.

As shown in Fig.6.2, we increase the resolution of the LR images with a deep network
Figure 6.2: Illustration of Deep Progressive Zooming Network. The scaling factor for this illustration is 2.

of four stages. Within each stage, a convolution-transpose layer is employed to learn the resolution projection from a lower resolution space to a higher resolution space. Then, ten convolutional layers are utilized to estimate the high-frequency components. Meanwhile, a mere 2-D interpolation algorithm is applied to generate low-frequency content. The combined higher resolution images are treated as inputs to the next stage. Such a formulation is also called *Residual Learning* [28, 58].
To be more specific, a scaling factor of two is decomposed into four sub-scaling factors: $\times \frac{5}{4}, \times \frac{6}{5}, \times \frac{7}{6}$, and $\times \frac{8}{7}$. To increase the spatial resolution within each stage, commonly deployed convolution-transpose operations with filters of the size $4 \times 4$ and the stride of $2$ [107, 108] are no longer feasible in our architecture. The main reason is that such a convolution-transpose set-up only generate results for integer scaling factors because fractional pixels do not physically exist in digital imagery. To address this problem, we decomposed LR images into smaller patches (i.e., $16 \times 16 \times 3$). By setting the convolution-transpose layer with filters of the size $5 \times 5$ and a stride of 1, we are able to increase the resolution of each patch by 4 pixels at every stage. Doing so, the spatial dimensions of an image patch is increased from 16 to 20, to 24, to 28, and to 32, which is the result of the scaling factor of two.

To fully utilize this deep architecture and perform digital zooming, additional constraints are required. Intermediate loss layers have to be attached to the higher resolution patches during the training stage. Without these intermediate loss layers, DPZN is free to learn any representations that we are yet to comprehend. Another benefit of attaching these intermediate layers is that we can guarantee the quality increments of the LR patches in the last stage. Compared to the extracted input feature maps in VDSR [58] (i.e., first layer outputs), the convolution-transpose feature maps in our last stage are extracted from higher quality LR patches due to the high-frequency insertion from previous stages. Therefore, the learning difficulties of generating an HR natural image with a scaling factor of two are greatly reduced, and better quality natural images can be estimated.

Our DPZN can also be efficiently extended to larger integer scaling factors (e.g., $\times 3, \times 4$). As shown in Fig.6.3, the estimated HR natural patches for a scaling factor of two can be decomposed into four sub-patches with the same spatial dimension as the input of the network. Therefore, it is intuitive to re-utilize the same network architecture to increase the resolution of these sub-patches. To further illustrate our proposed DPZN, we rewrite (6.4) to adapt our progressive architecture as:

$$[\Theta_1 \ldots \Theta_S] = \arg \min_{\Theta_s} \left\| \sum_{s=1}^{S} \sum_{ch=1}^{3} \sum_{k=1}^{K} \left[ F_{s,ch}^{ch} (\Theta_s, y^s_{ch}) - \left( x_{k,ch}^{s,ch} - G_{lf} (y_{k,lf}^{s,ch}) \right) \right] \right\|_p^2 + \lambda \sum_{s=1}^{S} \mathcal{R} (\Theta_s),$$

(6.5)

where $s$ denotes different stages, and $S$ represents the total number of stages needed to reach
Figure 6.3: Illustration of Deep Progressive Zooming Network extended to larger scaling factors. (a) is the image resolution flow diagram for scaling factor of two. (b) extends the result of (a) to reach larger scaling factors with the same network architecture. (c) illustrates SR result for scaling factor of three. (d) represents SR result for scaling factor of four.

certain resolution (e.g., $S = 4, 6, 8$ for $\times 2, \times 3, \times 4$, respectively). Our objective now is to optimize all the parameters $\Theta = [\Theta_1, \ldots, \Theta_S]$ associated with DPZN with respected to the distances between estimated patches and ground-truthes.

When increasing the spatial resolution of a natural image with a scaling factor of three, the stage five and the stage six can be considered as paralleled stage one and two. Due to the repetitive nature of our architecture, we can re-use the same parameters for higher stages. Thus, we re-distribute these parameters so that they can be retrained within the network. For instance, parameter $\Theta_1$ for SR with scaling factor of three or four is trained five times, making $\Theta_1 = \Theta_5$. This strategy efficiently limits the total number of parameters required for higher scaling factors without losing the progressive property.

To summarize, we propose a deep architecture that effectively increases the spatial resolution of the the input LR images. At each stage, the input patches are three channel
RGB color patches, and the corresponding feature maps are extracted by 64 filters with size $3 \times 3$. The spatial dimensions of these feature maps are increased by four pixels using a convolution-transpose layer with 64 filters of the size $5 \times 5$. Moreover, we learn the discriminative model using ten convolutional layers. There are 64 filters with size $3 \times 3$ within each convolutional layer except the high-frequency estimating layer. Additionally, each of these hidden layers is followed by a PReLU layer to introduce nonlinearities, with the exception of the high-frequency estimating layer.

### 6.2.4 Connection to Existing Progressive Learning Methods

There are a few progressive learning based SR architectures that exist in the literature. Each of them have some advantages and disadvantages. We compare each of them with our DPZN and summarize them as follows.

Deep network cascade (DNC) [109] proposes to learn the discriminative model using a deep cascaded auto-encoder network. This is the first work that proposed a deep progressive learning scheme. Similarly, this work realizes the degraded image input problem and tries to bypass this assumption. There are two major differences between DNC and our DPZN. Firstly, DNC depends on the non-local self-similarity (NLSS) search on the input image to compensate for high-frequency losses while DPZN gradually estimates the missing information. Secondly, DNC embeds auto-encoder networks in each layer while DPZN utilizes deep CNN. In brief, DNC was not designed to be an end-to-end solution, and bypassing image degradation using NLSS is not reliable.

Deep network with sparse prior (SCN) [101] proposes to join the deep network and the sparse prior of natural images together. This work is also a cascade network where two convolutional layers and three custom layers are employed in each stage. The main difference between the architectures is that SCN directly estimates the HR patches while DPZN falls into the residual learning category. Moreover, DPZN employs ten convolutional layers which have the ability to represent more complex models than SCN. Therefore, our DPZN learns a better representation of the mapping between resolutions and produces better results.

Deep Laplacian pyramid network (LapSRN) [108] is the most recently developed deep
progressive architecture. Our architectures share many similar insights: we both utilize the convolution-transpose operation to learn the initial upscaling function, and we both estimate missing high-frequency components. However, there are two major differences between our architectures. Firstly, LapSRN is a full-image SR network where the learned upscale kernels can only generate results for integer scaling factors (e.g., 2 or 3). Secondly, the feature maps used to generate high-frequency components are not extracted from intermediate high-resolution images, making this architecture sub-optimal when compared with DPZN.

6.3 Experiments

In this section, we train and evaluate our proposed deep natural image SR architecture. We first provide our detailed experiment setting. Then, the numerical and visual results are presented and compared with several state-of-the-art methods. Finally, we discuss the limitation and potential improvements of our proposed DPZN.

6.3.1 Experiment Setting

Training and Testing Datasets

During the training phase, we use publicly available datasets including the Berkeley segmentations dataset [110] BSD200 which provides 200 training images and the 91 images set [27] as our training dataset. For the experiments associated with the scaling factor two, we extract image patches with the size of $32 \times 32$ as our ground-truthes. Similarly, patches with the size of $48 \times 48$ are extracted for experiments with scaling factor three. To generate our input patches, all training images are downsampled through the Bicubic interpolation algorithm provided by MATLAB. Input patches with size of $16 \times 16$ are then cropped according to their HR patch locations. We extract roughly 480,000 and 200,000 training pairs for experiments with scaling factors two and three, respectively. Additionally, due to the progressive nature of our proposed architecture, intermediate ground-truthes are also produced according to non-integer scaling factors (e.g., $\times 1.25$, $\times 1.5$, and $\times 1.75$). To evaluate our trained models, popular comparison datasets
emph5 and Set14 are employed. The testing patches are extracted from the LR inputs with a slight overlap. After projecting these LR patches, the HR images are created by combining HR patches with averaged overlap regions.

**Training Setup**

To demonstrate the basic ability of our proposed architecture, we train our network with a simple training strategy. Thus, no data augmentation or finetuning is employed. We train our network on the open-sourced deep learning platform Tensorflow [111]. The Adam optimization scheme [112] is utilized to calculate the gradients and to minimize the losses. Meanwhile, all training parameters are initialized following the Xavier [113] process. To properly train our network, the learning rate is initially set to $1e^{-4}$ and then decreases by a factor of 10 every 20 epochs. The gradient clipping is also applied to avoid the initial gradient explosion problem.

**The Loss Penalty**

The loss penalty $\ell_p$ is one of the most critical components in our DPZN. Most deep learning related methods adapt to $\ell_2$ loss in their loss layer [31,58] because they are directly linked to the evaluation criterion (e.g., PSNR). However, models trained with $\ell_2$ penalty are known to generate blurring artifacts around sharp edges. This drawback is caused by the compromising nature of the $\ell_2$ penalty when countering outliers. Recently, [108] investigated and confirmed the superior performance of the $\ell_1$ loss penalty due to its robustness in handling outliers. Therefore, in our experiments, we adopt $\ell_1$ as our loss penalty. The Charbonnier penalty function, a differentiable variant of $\ell_1$, is shown as follows:

$$\| \cdot \|_p = \sqrt{x^2 + \epsilon^2},$$

(6.6)

where we set $\epsilon$ to $1e^{-4}$.

**Comparing Methods**

We compare the proposed DPZN method with aforementioned state-of-the-art methods, including adjusted ANR (A+) [30], Sparse representation SR (SCSR) [27], self-exemplar
based (selfExSR) [100], deep learning based (SRCNN) [31], and (VDSR) [58]. Bicubic interpolation is also included in the comparison as the baseline. All of the compared SR images are either acquired from the publicly released results or generated via available codecs. We evaluate the SR images with two commonly used image quality metrics: PSNR and SSIM. The numerical results are presented in section 6.3.2.

![Convergence curves](image)

Figure 6.4: Convergence curve analysis on the progressive architecture. (a) Convergence curves associated with the scaling factor of two. (b) Convergence curves associated with the scaling factor of three.

### 6.3.2 Experiment Results

**Deep Progressive Zooming Results**

We first present the digital zooming ability of our proposed DPZN. Fig.6.4 shows the convergence curves in terms of PSNR on the baby image in Set5. As we can observe from Fig.6.4, all four stages that are associated with the scaling factor of two and all six stages that are associated with the scaling factor of three are successfully converged after 80 epochs. It is interesting to see that the performance of stage one and two of Fig.6.4(b) decreases in the beginning, but quickly adjust themselves after a few epochs. One possible explanation for this abnormal phenomenon is that the weight sharing strategy we enforced are biased to stage five and six at the beginning since the loss ratio is 4 to 1. Additionally, these
convergence curves also demonstrate the capability of sharing kernels across the network for same scaling factors with different input qualities. Nevertheless, the total number of parameters required to represent the mapping function between resolutions is still unknown. We will investigate the impact on the number of parameters in future works.

In Figs.6.5 and 6.6, we present the visual results of our DPZN and the recovered high-frequency components. Please note that some high-frequency components are subtle and difficult to visualize. Thus, when plotting these high-frequency images, we alter the color maps for better visualization purposes. It can be seen that our DPZN progressively recovers the missing high-frequency components. Therefore, the qualities of the input images at each stage are enhanced, leading to better estimation at higher resolutions. Another phenomenon we observe is that the content of high-frequency components needed increases as the resolution increases. This is counter-intuitive since the scaling factor between two adjacent resolutions decreases (e.g., from $\frac{5}{4}$ to $\frac{6}{5}$ to $\frac{7}{6}$ to $\frac{8}{7}$). The main reason is that LR components are directly upsampled input image. Therefore, more high-frequency components are required to fill the gap at higher stages.

![Input](image1)

![High Frequency Components](image2)

Figure 6.5: Digital zooming result of Baby with a scaling factor of two.
Figure 6.6: Digital zooming result of Lenna with a scaling factor of three.

Learned Filters

Fig.6.7 shows examples of learned extraction layer kernels. It is challenging to fully comprehend the learned kernels especially with the PReLU activation function, which leaks negative information to subsequent layers. However, we can still make a few interesting observations. Firstly, 6912 weights are in Fig.6.7, and 97% of these weights are between range $-0.1$ to $0.1$. Thus, starting from the first extraction layer, our DPZN suppresses the pixel values in the feature maps only to estimate the high-frequency components at the later layers. Secondly, kernels learned for R, G, and B channels vary significantly. This behavior indicates that the information extracted from these channels are different and have to be considered jointly. Thirdly, out of all these filters, there are a few we can comprehend with a physical meaning. We enlarged a few of these kernels in Fig.6.8 for better visualization. For instance, kernels in Fig.6.8(a) are like edge detectors at different directions, kernels in Fig.6.8(b) are like gradient operators, and (c) is like a Laplacian filter.

Quantitative and Qualitative Evaluation

In this section, we present the quantitative and qualitative evaluations of PSNR and SSIM as well as the qualitative results. To compare with aforementioned state-of-the-art
methods, we only present evaluation results with integer scaling factors. Tables 6.1, 6.2 present PSNR results and tables 6.3, 6.4 show the SSIM results of images in Set5 and Set14 with a scaling factor of two. The best result for each image is highlighted in bold, and the second best result is underlined. Similarly, we present the PSNR and SSIM results of images in Set5 and Set14 with a scaling factor of three in the tables 6.5, 6.6, and 6.8. As one can see, our DPZN achieves comparable performance with these start-of-the-art methods. Please note that we only train our DPZN with the simplest training strategy to demonstrate the feasibility of our proposed framework, and that its performance is not fully optimized. We shall discuss the limitations and potential improvements in section 6.3.3.

In Fig.6.9 and Fig.6.10, we show visual comparison results on Set5 and Set14 with a scaling factor of two. When compared to with state-of-the-art methods, our DPZN is visually indistinguishable with the VDSR and superior to all other methods. Therefore, our DPZN
can accurately estimate the missing details due to its progressive architecture. Additionally, DPZN tends to generate sharper edges due to the $\ell_1$ loss function we enforce. For SR with a scaling factor of three, we present the visual comparison results in Fig. 6.11 and Fig. 6.12. As one can see, even for a larger scaling factor, DPZN is capable of generating aesthetically please results.

![Figure 6.8: Enlarged examples of learned kernels.](image)

<table>
<thead>
<tr>
<th>Image</th>
<th>Baby</th>
<th>Butterfly</th>
<th>Head</th>
<th>Bird</th>
<th>Woman</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicubic</td>
<td>36.69</td>
<td>31.78</td>
<td>32.90</td>
<td>37.05</td>
<td>34.58</td>
</tr>
<tr>
<td>ScSR [27]</td>
<td>37.27</td>
<td>33.39</td>
<td>33.08</td>
<td>38.30</td>
<td>35.70</td>
</tr>
<tr>
<td>SelfEx [28]</td>
<td>37.70</td>
<td>33.96</td>
<td>33.21</td>
<td><strong>39.82</strong></td>
<td>36.26</td>
</tr>
<tr>
<td>SRCNN [31]</td>
<td>37.20</td>
<td>33.67</td>
<td>33.06</td>
<td>38.31</td>
<td>35.77</td>
</tr>
<tr>
<td>VDSR [58]</td>
<td><strong>37.73</strong></td>
<td><strong>35.21</strong></td>
<td><strong>33.25</strong></td>
<td>39.53</td>
<td><strong>36.47</strong></td>
</tr>
<tr>
<td>DPZN</td>
<td>37.60</td>
<td>34.65</td>
<td>33.20</td>
<td>38.84</td>
<td>36.26</td>
</tr>
</tbody>
</table>

Table 6.1: Quantitative evaluation on Set5 dataset with a scaling factor of two. The mean-square-error (PSNR) results are presented. The best result is in bold and the second best is underlined.
Table 6.2: Quantitative evaluation on Set14 dataset with a scaling factor two. The mean-square-error (PSNR) results are presented. The best result is in bold and the second best is underlined.

### 6.3.3 Current Limitations

Although DPZN achieves the state-of-the-art performances, there are still some limitations with our current set-ups. We briefly summarize them as follows. Firstly, training DPZN with a one-time training strategy is difficult. Arguably, our DPZN for a scaling factor $\times 2$ can be considered as four ten-layers sub-networks. When jointly training all sub-networks, gradients can back-propagate as far as forty layers, increasing the possibility of generating vanished gradients. Therefore, a more elegant training procedure is needed to fully utilize the potential of DPZN. Secondly, increasing input channels from a single channel (e.g., Y-luminance channel) to R, G, and B channels introduces more challenges. The main reason is that the number of patterns available in input RGB images increases exponentially. Thus, we may need to increase the number of kernels or adjust the feature extracting layer to accommodate different feature combinations. We will further investigate DPZN to overcome these limitations and to maximize its performance.

### 6.4 Summary

The Natural image SR task is a challenging problem due to its ill-posed nature. With the recent success of utilizing machine learning algorithms on low-level vision tasks, directly learning the mapping function between resolutions attracts more attention. In this chapter,
we propose a deep progressive zooming network that aims to handle the natural image SR resolution. To increase the capability of a network, we limit our architecture to only process image patches. Therefore, we can rely on the network to learn the initial upsampling phase and estimate results with non-integer scaling factors. Unlike many traditional SR methods, our DPZN not only produces the desired HR image with an integer scaling factor but also generates intermediate results. Because of these intermediate results, the quality of feature maps at a higher resolution stage is increased, leading to better results. Our Experiments on testing datasets have confirmed that DPZN is capable of reaching state-of-the-art performance and generating aesthetically pleasing results.
Table 6.4: Quantitative evaluation on *Set14* dataset with a scaling factor of two. The structure similarity index (SSIM) results are presented. The best result is in bold and the second best is underlined.

<table>
<thead>
<tr>
<th>Image</th>
<th>Baby</th>
<th>Butterfly</th>
<th>Head</th>
<th>Bird</th>
<th>Woman</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicubic</td>
<td>34.44</td>
<td>30.71</td>
<td>32.16</td>
<td>34.42</td>
<td>33.11</td>
</tr>
<tr>
<td>ScSR [27]</td>
<td>34.72</td>
<td>31.08</td>
<td>32.26</td>
<td>34.75</td>
<td>33.44</td>
</tr>
<tr>
<td>SelfEx [28]</td>
<td>35.14</td>
<td>32.09</td>
<td>32.42</td>
<td><strong>36.01</strong></td>
<td>34.38</td>
</tr>
<tr>
<td>SRCNN [31]</td>
<td>34.79</td>
<td>31.80</td>
<td>32.27</td>
<td>34.93</td>
<td>33.92</td>
</tr>
<tr>
<td>A+ [30]</td>
<td>35.03</td>
<td>32.02</td>
<td>32.40</td>
<td>35.35</td>
<td>34.26</td>
</tr>
<tr>
<td>VDSR [58]</td>
<td>35.14</td>
<td><strong>33.24</strong></td>
<td><strong>32.44</strong></td>
<td>35.77</td>
<td><strong>34.66</strong></td>
</tr>
<tr>
<td>DPZN</td>
<td>35.03</td>
<td>32.66</td>
<td>32.34</td>
<td>35.22</td>
<td>34.43</td>
</tr>
</tbody>
</table>

Table 6.5: Quantitative evaluation on *Set5* dataset with a scaling factor of three. The mean-square-error (PSNR) results are presented. The best result is in bold and the second best is underlined.

<table>
<thead>
<tr>
<th>Image</th>
<th>Baby</th>
<th>Butterfly</th>
<th>Head</th>
<th>Bird</th>
<th>Woman</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicubic</td>
<td>0.686</td>
<td>0.826</td>
<td>0.864</td>
<td>0.827</td>
<td>0.901</td>
</tr>
<tr>
<td>ScSR [27]</td>
<td>0.714</td>
<td>0.845</td>
<td>0.898</td>
<td>0.834</td>
<td>0.912</td>
</tr>
<tr>
<td>SelfEx [28]</td>
<td><strong>0.741</strong></td>
<td>0.858</td>
<td><strong>0.918</strong></td>
<td><strong>0.850</strong></td>
<td><strong>0.934</strong></td>
</tr>
<tr>
<td>SRCNN [31]</td>
<td>0.729</td>
<td>0.855</td>
<td>0.908</td>
<td>0.842</td>
<td>0.910</td>
</tr>
<tr>
<td>A+ [30]</td>
<td>0.730</td>
<td><strong>0.859</strong></td>
<td>0.912</td>
<td>0.846</td>
<td>0.916</td>
</tr>
<tr>
<td>VDSR [58]</td>
<td>0.739</td>
<td>0.858</td>
<td><strong>0.927</strong></td>
<td><strong>0.850</strong></td>
<td><strong>0.920</strong></td>
</tr>
<tr>
<td>DPZN</td>
<td>0.733</td>
<td>0.856</td>
<td><strong>0.921</strong></td>
<td>0.847</td>
<td><strong>0.920</strong></td>
</tr>
</tbody>
</table>

Table 6.6: Quantitative evaluation on *Set14* dataset with a scaling factor of three. The structure similarity index (SSIM) results are presented. The best result is in bold and the second best is underlined.

<table>
<thead>
<tr>
<th>Image</th>
<th>Baby</th>
<th>Butterfly</th>
<th>Head</th>
<th>Bird</th>
<th>Woman</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicubic</td>
<td>0.714</td>
<td>0.845</td>
<td>0.898</td>
<td>0.834</td>
<td>0.912</td>
</tr>
<tr>
<td>ScSR [27]</td>
<td>0.741</td>
<td>0.858</td>
<td>0.918</td>
<td><strong>0.850</strong></td>
<td><strong>0.934</strong></td>
</tr>
<tr>
<td>SelfEx [28]</td>
<td>0.729</td>
<td>0.855</td>
<td>0.908</td>
<td>0.842</td>
<td>0.910</td>
</tr>
<tr>
<td>SRCNN [31]</td>
<td>0.730</td>
<td><strong>0.859</strong></td>
<td>0.912</td>
<td>0.846</td>
<td>0.916</td>
</tr>
<tr>
<td>A+ [30]</td>
<td>0.739</td>
<td>0.858</td>
<td><strong>0.927</strong></td>
<td><strong>0.850</strong></td>
<td><strong>0.920</strong></td>
</tr>
<tr>
<td>VDSR [58]</td>
<td>0.739</td>
<td>0.858</td>
<td><strong>0.927</strong></td>
<td><strong>0.850</strong></td>
<td><strong>0.920</strong></td>
</tr>
<tr>
<td>DPZN</td>
<td>0.733</td>
<td>0.856</td>
<td><strong>0.921</strong></td>
<td>0.847</td>
<td><strong>0.920</strong></td>
</tr>
</tbody>
</table>
Table 6.6: Quantitative evaluation on *Set14* dataset with a scaling factor three. The mean-square-error (PSNR) results are presented. The best result is in bold and the second best is underlined.

<table>
<thead>
<tr>
<th>Image</th>
<th>Babn.</th>
<th>Bar.</th>
<th>Comic</th>
<th>Face</th>
<th>Flow.</th>
<th>Fman</th>
<th>Lenna</th>
<th>Mrch</th>
<th>Pepper</th>
<th>ppt3</th>
<th>Zebra</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicubic</td>
<td>29.22</td>
<td>31.26</td>
<td>29.75</td>
<td>32.11</td>
<td>31.41</td>
<td>34.78</td>
<td>32.90</td>
<td>34.59</td>
<td>33.15</td>
<td>33.43</td>
<td>30.78</td>
</tr>
<tr>
<td>ScSR [27]</td>
<td>29.25</td>
<td>31.42</td>
<td>29.91</td>
<td>32.20</td>
<td>31.53</td>
<td>35.26</td>
<td>33.04</td>
<td>34.87</td>
<td>33.23</td>
<td>33.74</td>
<td>31.11</td>
</tr>
<tr>
<td>SelfEx [28]</td>
<td><strong>30.30</strong></td>
<td><strong>31.69</strong></td>
<td>30.18</td>
<td><strong>32.38</strong></td>
<td><strong>32.04</strong></td>
<td>36.20</td>
<td><strong>33.40</strong></td>
<td>35.70</td>
<td>33.62</td>
<td>35.04</td>
<td>31.56</td>
</tr>
<tr>
<td>SRCNN [31]</td>
<td>29.25</td>
<td>31.46</td>
<td>30.04</td>
<td>32.23</td>
<td>31.57</td>
<td>35.59</td>
<td>33.13</td>
<td>35.26</td>
<td>33.18</td>
<td>33.86</td>
<td>31.33</td>
</tr>
<tr>
<td>A+ [30]</td>
<td>29.26</td>
<td>31.51</td>
<td>30.11</td>
<td>32.34</td>
<td>31.73</td>
<td>36.22</td>
<td>32.22</td>
<td>35.47</td>
<td>33.37</td>
<td>34.69</td>
<td>31.58</td>
</tr>
<tr>
<td>VDSR [58]</td>
<td><strong>29.30</strong></td>
<td>31.60</td>
<td><strong>30.37</strong></td>
<td><strong>32.39</strong></td>
<td><strong>31.97</strong></td>
<td><strong>36.63</strong></td>
<td>33.35</td>
<td><strong>36.46</strong></td>
<td>33.44</td>
<td><strong>35.06</strong></td>
<td><strong>31.70</strong></td>
</tr>
<tr>
<td>DPZN</td>
<td>29.27</td>
<td>31.53</td>
<td>30.25</td>
<td>32.32</td>
<td>31.75</td>
<td>36.29</td>
<td>33.28</td>
<td>35.88</td>
<td>33.26</td>
<td>34.66</td>
<td>31.55</td>
</tr>
</tbody>
</table>

Table 6.7: Quantitative evaluation on *Set5* dataset with a scaling factor of three. The structure similarity index (SSIM) results are presented. The best result is in bold and the second best is underlined.

<table>
<thead>
<tr>
<th>Image</th>
<th>Baby</th>
<th>Butterfly</th>
<th>Head</th>
<th>Bird</th>
<th>Woman</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicubic</td>
<td>0.900</td>
<td>0.807</td>
<td>0.765</td>
<td>0.922</td>
<td>0.896</td>
</tr>
<tr>
<td>ScSR [27]</td>
<td>0.907</td>
<td>0.843</td>
<td>0.777</td>
<td>0.930</td>
<td>0.908</td>
</tr>
<tr>
<td>SelfEx [28]</td>
<td><strong>0.919</strong></td>
<td>0.896</td>
<td>0.794</td>
<td><strong>0.954</strong></td>
<td>0.936</td>
</tr>
<tr>
<td>SRCNN [31]</td>
<td>0.911</td>
<td>0.889</td>
<td>0.782</td>
<td>0.931</td>
<td>0.924</td>
</tr>
<tr>
<td>A+ [30]</td>
<td>0.916</td>
<td>0.900</td>
<td>0.790</td>
<td>0.940</td>
<td>0.931</td>
</tr>
<tr>
<td>VDSR [58]</td>
<td><strong>0.918</strong></td>
<td><strong>0.930</strong></td>
<td><strong>0.796</strong></td>
<td>0.946</td>
<td><strong>0.942</strong></td>
</tr>
<tr>
<td>DPZN</td>
<td>0.917</td>
<td><strong>0.913</strong></td>
<td>0.788</td>
<td>0.939</td>
<td>0.936</td>
</tr>
</tbody>
</table>
Table 6.8: Quantitative evaluation on Set14 dataset with a scaling factor of three. The structure similarity index (SSIM) results are presented. The best result is in bold and the second best is underlined.
Figure 6.9: Visual Comparison for \( \times 2 \) SR on image `baby` of `Set5`. 
Figure 6.10: Visual Comparison for ×2 SR on image barbara of Set14.
Figure 6.11: Visual Comparison for $\times3$ SR on image butterfly of Set14.
Figure 6.12: Visual Comparison for $\times 3$ SR on image comic of Set14.
Chapter 7

Summary

Digital imagery has become an important information carrier in our daily life. It conveys our emotions, wishes, knowledge, and other useful information. The demand for high quality and aesthetically pleasing images never decreases. There are many challenges related to the modern digital image acquiring and transmission tasks, in which image super-resolution is one of the most common obstacles. Image super-resolution aims to increase image resolutions from a lower resolution image. From the digital zooming function on any digital camera to MRI images used in hospitals, there are many beneficiaries of image super-resolution. It is known to the community that such an inverse process is highly ill-posed; recovering the exact ground-truth is almost impossible. Over the past decades, numerous techniques have been adopted to ease this problem. In this thesis, we devote to studying and attacking this problem from a guided image and a learning representation point of view.

Guided imaging is a special class of the image super-resolution problem. The motivation behind guided imaging is that for some domain specific super-resolution tasks extra information may provide more useful information in helping the estimation process (i.e. constrain the estimation towards the ground-truth). The depth map super-resolution problem is a perfect example of where additional information might help. Typically, devices that measure physical distances between a scene and the camera produce low-quality, low-resolution maps. However, high-resolution RGB images are easier to acquire. It is intuitive to combine them to achieve high-resolution depth maps.

In this dissertation. Chapter 3 and 4 study the guided image super-resolution problem.
More specifically, the guided depth map super-solution problem is tackled. In Chapter 3, we model high resolution depth maps through simultaneous-sparse-representation. High-resolution RGB images are employed as guidance in finding similar patches. Since patch clustering is a crucial step to the SSC modeling, such guidance is very effective in finding accurate clusters, and therefore leads to better results. In chapter 4, we intend to solve this problem by directly fusing both images together. Inspired by many depth prediction works, we learn the nonlinear mapping between color and depth. Furthermore, only the missing high-frequency components are recovered to enhance the visual appearance of the high-resolution estimations.

Learning representation is another issue to be addressed to tackle the image super-resolution problem. The technique behind learning representation is to utilize the rich repository of existing high-resolution images and to discover the natural representation of high-resolution images. Conventionally, to increase image resolution, carefully engineered models are often deployed. These models (including many state-of-the-art methods) indeed reveal many image properties related to high-resolution images and enhance our understanding of digital imagery. However, with powerful machine learning techniques, we can model the image super-resolution problem implicitly. Therefore, image properties that are hard to model or employ can now be incorporated.

Chapter 4, 5, and 6 study the machine learning approaches to the image super-resolution problem. In chapter 4, two separate neural networks are employed to learn the relationship between high-resolution RGB images and high-resolution depth maps and the relationship between low-resolution depth maps and high-resolution depth maps. When estimating high-frequency components, these images are projected onto some high-dimensional feature spaces. Their high-level representations are then merged and projected back to the high-resolution space. In chapter 5, we utilize deep learning techniques for the image interpolation problem. Instead of estimating high-frequency components as we exploited in chapter 4, we directly estimate high-resolution pixels by their local neighbors. We have verified that such a process benefits the image interpolation process by retaining the geometric duality property and achieving state-of-the-art performance with less run time. In chapter 6, we continue to exploit the residual learning technique discussed in chapter 4 for the natural image SR...
task. We proposed a deep progressive network that gradually increases the quality of the input feature maps. Therefore, we can learn better representations for the natural image SR problem. Moreover, we extend the idea of learning upscaling filtering into our architecture for non-integer scaling factors and estimate intermediate results, which performs digital zooming.

The image interpolation task and the natural image SR problem are similar yet different tasks. In this dissertation, the differences between these two problems are modeled after the appearances of the blurring kernel. In chapter 5, we ignore the blurring kernel when modeling LR images. Thus, we keep our network in the spatial domain and directly estimate the missing pixels. The learned model, therefore, recognizes local geometries and acts accordingly. In chapter 6, high-frequency information is removed when modeling LR images. Therefore, the priority of the reconstruction process is to recover the lost high-frequency content. With the proposed architecture, intermediate high-frequency components are estimated and utilized prior to the feature map extraction phase of the final resolution, leading to better high-frequency estimation. Overall, both of our strategies demonstrate state-of-the-art performance and yield aesthetically pleasing results.
References


111


[38] David L. Donoho, “For most large underdetermined systems of linear equations the minimal 1-norm solution is also the sparsest solution,” Communications on Pure and Applied Mathematics, vol. 59, no. 6, pp. 797–829, 2006.


