Learning-based Stereo Matching for 3D Reconstruction

by

A thesis submitted to the

School of Graduate Studies

in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

Memorial University of Newfoundland

September 2019

St. John's Newfoundland

Abstract

Stereo matching has been widely adopted for 3D reconstruction of real world scenes and has enormous applications in the fields of Computer Graphics, Vision, and Robotics. Being an ill-posed problem, estimating accurate disparity maps is a challenging task. However, humans rely on binocular vision to perceive 3D environments and can estimate 3D information more rapidly and robustly than many active and passive sensors that have been developed. One of the reasons is that human brains can utilize prior knowledge to understand the scene and to infer the most reasonable depth hypothesis even when the visual cues are lacking. Recent advances in machine learning have shown that the brain's discrimination power can be mimicked using deep convolutional neural networks. Hence, it is worth investigating how learning-based techniques can be used to enhance stereo matching for 3D reconstruction.

Toward this goal, a sequence of techniques were developed in this thesis: a novel disparity filtering approach that selects accurate disparity values through analyzing the corresponding cost volumes using 3D neural networks; a robust semi-dense stereo matching algorithm that utilizes two neural networks for computing matching cost and performing confidence-based filtering; a novel network structure that learns global smoothness constraints and directly performs multi-view stereo matching based on global information; and finally a point cloud consolidation method that uses a neural network to reproject noisy data generated by multi-view stereo matching under different viewpoints. Qualitative and quantitative comparisons with existing works demonstrate the respective merits of these presented techniques.

Acknowledgements

As this journey of Ph.D. study comes to an end, it falls to my lot to express special thanks and appreciation to all the people who supported me unconditionally. Without their help and accompany, it is impossible for me to continue chasing my goals.

First of all, I would like to dedicate this dissertation to my supervisor, Dr. Minglun Gong. Four years ago, you supported my application for Memorial University, which was one of the happiest moment in my life. All the time and efforts you have spent on helping me fulfill this research are beyond my expectation. I have learned from you not only academic skills, but also kindness and patience.

In addition, I would like to express many thanks to my fellow friends, Zili Yi, Songyuan Ji, Shiyao Wang, Cao Cai, Ali Farrokhtala, Xue Cui, Xin Huang, Mingjie Wang and Jun Zhou. I would also like to take this opportunity to acknowledge the advices from all the committee members of my comprehensive exam and the supervisory committee members: Drs. Todd Wareham, Miklos Bartha, Manrique Mata-Montero, Wlodek Zuberek, Ting Hu, David Churchill, Mohamed Shehata, Adrian Fiech, Yuanzhu Chen, and Oscar Meruvia-Pastor. My thankfulness is yet to all the examiners: Drs. Sharene Bungay, Yang Wang and Lourdes Pena-Castillo, and the administrative assistants: Andrew Kim, Erin Manning, Darlene Oliver, Jennifer Friesen and Sharene Bungay for their kindly assistance.

Finally, I would particularly like to thank my family, my Canadian brother Tyler Flynn and his family. Life is made of ups and downs, and you make me strong.

Thanks to everyone again!

Contents

List of Abbreviations

AUC [30]	Area Under the Curve
CNN	Convolutional Neural Network
DF	Disparity Filtering
DP	Dynamic Programing
DPCS	Dynamic Point Cloud Sequence
ELAS[21]	Efficient Large-Scale Stereo Matching
HMT [72]	Hidden Markov Trees
LOP	Locally Optimal Projector
LIDAR	Light Detection and Ranging
LRD [30]	Left Right Difference
MPE	Mean Percentage Error
MRF	Markov Random Field
MVS	Multi-View Stereo
NCC	Normalized Cross-Correlation
kNN	k-Nearest-Neighbors
RDP [23]	Reliability-based Dynamic Programing
RMS	Root-Mean-Square

ROC	Receiver Operating Characteristic
SAD	Sum of Absolute Differences
SGM[29]	Semi-Global Matching
SSD	Sum of Squared Differences
WTA	Winner-Take-All

Stereo Matching Imagery:

DTU [1], KITTI [61], Middlebury [77]

Binocular Stereo Matching Algorithms on Middlebury:

ICSG [82], IDR [47], INTS [34], MC-CNN [103], MPSV [5]

MotionStereo (anonymous), R-NCC (anonymous)

r200high [42], SED [66], SNCC [16], TMAP [72]

Confidence Measure Algorithms:

AML [62], APKR [45], CUR [15], CCNN [68], LRC [30]

MSM [15], O1 [67], PKRN [30], UCC [69], WMN [30]

Multi-View Stereo Matching and Consolidation Algorithms:

Camp [8], Furu [18], Gipuma [19], Ji [37], Tola [87]

EC-Net [99], PU-Net [100]

List of Tables

List of Figures

Chapter 1

Introduction

3D content is needed in a wide range of applications such as medical diagnosis [46], city planning [44] and archeology [105]. Recent developments in Augmented Reality (AR) and Virtual Reality (VR) techniques make 3D content even more accessible to average users. This cultivates a steadily growing market [85] and also increases the demand for 3D content. While active 3D sensing tools such as laser scanners [94], structured light cameras [1], and time-of-flight cameras [106] have been developed, reconstructing 3D models from images taken by regular cameras is often desirable due to its low-cost and passivity.

Being an ill-posed problem, reconstructing 3D models from 2D images is very challenging, especially under conditions such as lack of texture [29], presence of occlusion [95] and variations in lighting [78]. Conventionally, stereo matching algorithms rely on heuristically defined constraints to address the problem. Since humans can estimate 3D information from 2D observations robustly, and deep neural networks have demonstrated the capability of mimicking humans' object recognition capability [39],

one has to wonder how deep neural networks can be used to enhance existing stereo matching algorithms for 3D reconstruction.

The first attempt made in this thesis is to identify mismatches generated by traditional binocular matching algorithms using a deep neural network. Traditionally, this problem is studied under the topic of sparse stereo matching, which aims to output accurate disparity values for selected pixels only [78]. Instead of designing another disparity optimization method for sparse disparity matching, a novel disparity filtering step that detects and removes inaccurate matches is presented in Chapter 2. Based on 3D convolutional neutral networks, the proposed detector is trained directly on 3D matching cost volumes and hence can work with different matching cost generation approaches. The experimental results show that it can effectively filter out mismatches while preserving the correct ones. Evaluation shows that combining the proposed approach with even the simplest Winner-Take-All (WTA) optimization leads to better performance than most existing sparse stereo matching algorithms.

The above disparity filtering approach only removes mismatches and cannot correct them. While in many cases mismatches are isolated and hence the remaining accurate matches are sufficient for depth perception, large regions of mismatches often occur when lighting changes between the input stereo image pairs and/or object surfaces do not have sufficient textures. As a result, disparity filtering may cause some objects to not show up in the resulting disparity maps at all. To address this issue, a semi-dense stereo matching algorithm is presented in Chapter 3. It utilizes two Convolutional Neural Network (CNN) models for computing stereo matching cost and performing confidence-based filtering, respectively. Compared to existing CNNs-based matching cost generation approaches, the proposed method feeds ad-

ditional global information into the network so that the learned model can better handle challenging cases. By utilizing non-parametric transforms, the method is also more self-reliant than most existing semi-dense stereo approaches, which rely heavily on the adjustment of parameters. The experimental results based on the Middlebury Stereo dataset [77] demonstrate that the proposed approach is comparable to the state-of-the-art semi-dense stereo approaches.

Using the insights learned from designing the previous two approaches for binocular stereo matching, an application of neural networks to multi-view stereopsis, which refers to the perception of depth and 3D structure obtained on the basis of visual information [78], is considered next. Although a number of learning-based approaches have been proposed over the past few years, most of them train networks over small cropped image patches, so that the requirements on GPU processing power and memory space are manageable. The limitation of such approaches, however, is that the networks cannot effectively learn global information and hence have trouble handling large textureless regions. In addition, when testing on different datasets, these networks often need to be retrained to achieve optimal performances. To address this deficiency, a robust framework is presented in Chapter 4, which is trained on high-resolution (1280×1664) stereo images directly. It is therefore capable of learning global information and enforcing smoothness constraints across the whole image. To reduce the memory space requirement, the network is trained to output the matching scores of different pixels under each depth hypothesis at a time. A novel loss function is designed to properly handle the unbalanced distribution of matching scores. Finally, trained over binocular stereo datasets only, the network can directly handle the DTU [1] multi-view stereo dataset and generate results comparable to the

state-of-the-art approaches.

To generate a complete 3D model for an object or environment, it is necessary to convert 2D disparity maps generated under different viewpoints into 3D point clouds and then merge multiple point clouds together. An additional point consolidation procedure is often needed here for removing outliers and better aligning individual patches. Numerous approaches have been proposed for 3D point cloud consolidation, which include some that use neural networks for point-based surface smoothing [76], upsampling [100], and completion [27]. In Chapter 5, a novel network is presented, which consolidates 3D point clouds through directly projecting individual 3D points based on point distributions in their neighborhoods. Since only local information is used, the proposed network is scalable for handling point clouds of any sizes and is capable of processing selected areas of interest as well. Quantitative evaluation on the DTU [1] dataset, which is the largest multi-view stereo benchmark, demonstrates the proposed approach can effectively improve the accuracy of point clouds generated by existing multi-view stereo algorithms.

In summary, a number of learning-based algorithms are presented in this thesis for detecting mismatches in binocular stereo matching results, generating more accurate matches under challenging conditions, performing multi-view stereo matching with global smoothness constraint enforced, and consolidating point clouds obtained from different viewpoints. Some of the work has appeared in peer-reviewed conferences [53, 54, 57, 92], and some is under review for journal and conference publication [55, 56]. In the following chapters these algorithms, as well as the related works, are presented in details.

Chapter 2

Disparity Filtering with 3D Convolutional Neural Networks

A binocular stereo matching problem can be described as identifying matching pixels in two images captured at different horizontal positions [102]. Under the epipolar constraint, if the same 3D point p is projected to pixel (x_1, y_1) on the left image I_1 and pixel (x_2, y_2) on the right image I_2, then:

$$x_1 - d = x_2 \quad \text{and} \quad y_1 = y_2, \tag{2.1}$$

where d is the disparity between the two pixels, which reveals the depth of the 3D point p.

Due to its important applications, the stereo matching problem has been extensively studied over the past decades, with numerous algorithms proposed [78]. Nevertheless, even the state-of-the-art methods cannot guarantee the generation of accurate disparity maps under challenging situations, such as lighting changes, occlusions, low or no texture, non-Lambertian surfaces [86], reflections and translucency

of objects. Hence, sparse disparity matching algorithms were developed to output accurate disparity values for selected pixels only [49].

As reviewed by Scharstein and Szeliski [78], most stereo matching algorithms perform the following four steps: (1) matching cost computation, (2) cost aggregation, (3) disparity computation and optimization, and (4) refinement. Many existing sparse disparity matching algorithms [52, 91, 29] use customized disparity computation and optimization approaches to generate accurate matches only. Instead of designing yet another disparity optimization method for sparse disparity matching, the approach proposed herein focuses on the refinement step. In particular, a learning-based approach is presented to distinguish accurate disparity values from mismatches so that the latter group can be filtered out. The detector is trained directly based on the input 3D matching cost volumes and the accuracy of the output disparity maps. To work with 3D cost volumes, a 3D CNN architecture is designed for training.

In summary, this chapter demonstrates that it is possible to infer the accuracy of estimated disparity values based on input 3D cost volumes. The 3D CNNs designed accordingly can effectively filter out mismatches and produce sparse disparity maps.

2.1 Related work

2.1.1 Sparse disparity algorithms

Compared to dense stereo matching algorithms that assign disparities to all valid pixels [97], sparse (also referred to as semi-dense) stereo matching algorithms concentrate on outputting accurate disparity values for selected pixels.

An early work on sparse stereo matching was proposed by Manduchi and Tomasi [52], which applied matching algorithms on distinctive points first and further calculated disparities for the remaining pixels. Veksler [91] utilized graph cuts to detect textured areas as an alternative to unambiguous points and generated corresponding semi-dense results. By design, their approach can filter out mismatches caused by lack of textures, but not by occlusions. Semi-Global Matching (SGM) [29] utilized multiple 1D constraints to generate accurate semi-dense results based on peak removal and consistency checks. Gong and Yang [22] proposed a reliability measure to detect potential mismatches from disparity maps generated using Dynamic Programming (DP). This work was later extended and implemented on graphics hardware for real-time performance [24].

Inspiring algorithms were also proposed more recently to further improve accuracy of sparse stereo matching. The Efficient Large-scale Stereo Matching (ELAS) algorithm [21] creates a 2D mesh via a triangulation supported by a set of sparse matching points to reduce matching ambiguities of the remaining points and to compute disparities for high resolution images. Following the idea of triangulation, Jellal et al. [36] proposed a line segment extension of ELAS algorithm, referred to as LS-ELAS. A set of line segments and support points allow this algorithm to generate a more informative triangulation mesh which can better handle depth discontinuities. Assuming the association between color and disparity, the Hidden Markov Trees (HMT) method [72] creates minimum spanning trees for images, passes aggregated costs along the tree branches, and finally performs median filtering to remove isolated mismatches.

Generally, algorithms in this category utilize constraints and/or a customized disparity computation step for sparse stereo matching. It is therefore hard to apply

to different disparity optimization approaches.

2.1.2 Filtering through confidence measures

Approaches have also been proposed for computing confidence measures on disparity maps generated by dense stereo matching algorithms. These measures can then be used to filter out potential mismatches at the disparity refinement step. Quantitative evaluations on traditional confidence measures were presented in Hu et al. [30] and Poggi et al. [69]. How to improve error detection through combining multivariate confidence measures was introduced in Haeusler et al. [26]. Furthermore, Park and Yoon [65] proposed to apply a regression forest framework for selecting effective confidence measures. Relying on $O(1)$ features and machine learning, Poggi and Mattoccia [67] proposed an improved scanline aggregation strategy, which performs streaking detection on each path in the SGM algorithm to generate a confidence measure.

The approach proposed in this Chapter aims at filtering out mismatches at the disparity refinement step and hence belongs to this category. Nevertheless, instead of using customized confidence measures, a learning-based approach is employed to directly infer the confidence of disparity computation output. This makes the proposed algorithm similar to recent learning-based works [10, 80], which compute confidences through training 2D CNNs on 2D image or disparity patches. A key difference is, however, that a 3D CNN is employed in the presented approach. Utilizing a stereo dataset that has ground truth available, the 3D CNN is trained based on the accuracy of the disparity computation output and the respective 3D cost volume input. Experimental results show that this learning model can make accurate predictions

and therefore effectively filter out mismatches.

2.1.3 Convolutional neural networks

The pioneering work of CNNs was established by LeCun et al. [50] for recognizing 2D shapes, such as digital numbers and hand-written characters. Over the past few years, CNNs have shown their power in many computer vision problems, such as image classification [48], and point-wise localization and segmentation [14]. Extensions to 3D CNN were also proposed and applied to 3D object recognition or classification [59], spatiotemporal features extraction [90], scene flow estimation [60], human action recognition [38], and landing zone detection using light detection and ranging (LiDAR) sensors [58].

A few attempts were also made to apply CNNs to the stereo matching problem. Zbontar and LeCun [103] focused on the matching cost computation step and designed a CNN architecture that computes the matching cost for two 9×9 image patches. 2D CNNs were further applied to compute confidence measures in Park et al. [64] and Poggi et al. [68], where the training inputs are 2D patches from either the input images or the disparity maps. Luo et al. [51] treated the stereo matching problem as a multi-class classification problem, where the classes are all possible disparities. A matching network is proposed accordingly to efficiently produce accurate disparity maps. In comparison, Kendall et al. [41] proposed an end-to-end architecture to learn a stereo regression model. This architecture uses 2D convolutions to learn contextual information and further combines 3D convolutions and deconvolutions to regularize its disparity cost volume, from which disparities are regressed by a soft

argmin function. An end-to-end learning framework was also proposed for multi-view stereo (MVS) [37], which converts images to 3D voxel representations through projection and trains a 3D CNN model to predict the surface of the 3D object.

Instead of training a multi-class classifier [51] or a regression model [41] to output disparity values directly, the proposed approach trains a binary classifier to predict whether the disparity value generated at each pixel by a given algorithm is accurate. As a result, this approach can work with different dense stereo matching approaches and turn their noisy disparity map output into sparse but accurate matches.

2.2 Methodology

As mentioned above, the proposed approach focuses on the disparity refinement step and can work with different matching cost generation and disparity computation methods. Using the approach with the simple WTA optimization scheme will be illustrated here.

Figure 2.1 shows the pipeline of the whole sparse stereo matching process. First, the sum of squared differences (SSD) approach is used in the initial matching cost computation step. This is followed by cost aggregation through bilateral filtering [88]. Different window sizes were tested and in the end a 7×7 window size was used.

The above two steps generate a 3D cost volume, $C(x, y, d)$, where the value at location (x, y, d) stores the locally aggregated cost of matching pixel (x, y) in the left image with $(x - d, y)$ in the right image. A disparity map can then be generated by performing the following WTA optimization at each pixel location:

$$d(x, y) = \arg\min_{d} C(x, y, d). \tag{2.2}$$

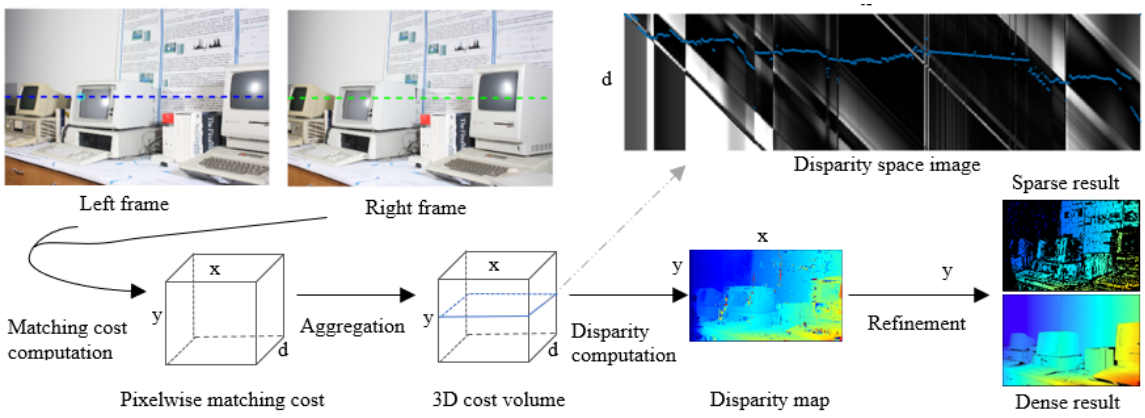


Figure 2.1: Sparse stereo matching pipeline. Given a pair of rectified images [77], a matching cost volume is first computed, followed by cost aggregation to suppress isolated noise. Each slice of the 3D cost volume corresponds to a disparity space image. A disparity map is computed though searching the minimum cost locations within the cost volume. The presented disparity filtering approach works at the final refinement stage to detect and remove mismatches.

The task now is to train a model that can predict whether the obtained $d(x, y)$ is an accurate disparity value based on the cost volume $C(x, y, d)$.

2.2.1 3D CNN architecture

Under an ideal situation, when a 3D point p is projected to pixel (x, y) in image I_1 and $(x - d, y)$ in I_2, $I_1(x, y)==I_2(x - d, y)$ and $I_1(x, y) \neq I_2(x - g, y)$ for all $g \neq d$. This implies that each column (x, y) of the matching cost volume $C(x, y, d)$, referred to as a matching cost curve, has a unique and clearly identifiable global minimum, which corresponds to the correct disparity value. However in practice, due to image capturing noise and matching ambiguities, it is often difficult to locate the minimum

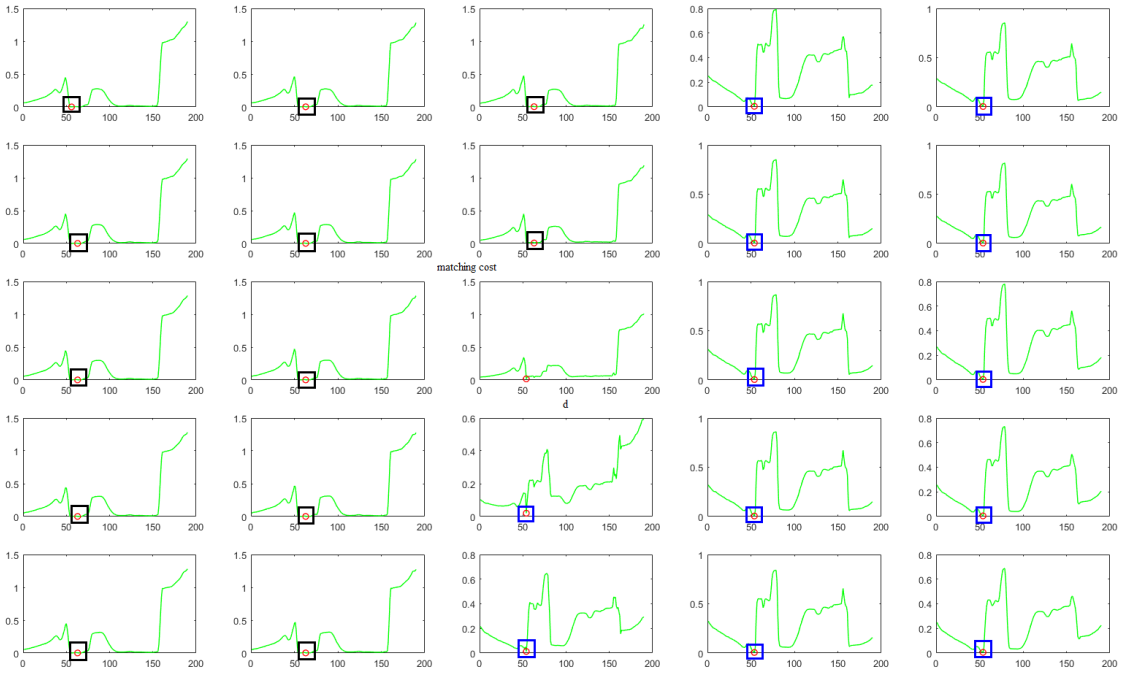


Figure 2.2: Matching cost curves of a given pixel and its 24 closest neighbors. Curves with unique minima are highlighted with blue rectangles, whereas those without clearly identifiable minima are shown in black.

as shown in Figure 2.2. To address this problem, additional constraints, such as local smoothness and left-right consistency [22], are often used. The problem can then be formulated as a global optimization problem, which is solved using various optimization techniques, such as dynamic programming, graph cuts, etc. Nevertheless, none of these algorithms can ensure the accuracy of the generated disparity maps.

The hypothesis is that it is possible to infer whether the disparity value computed by a given algorithm is accurate through the analysis of nearby matching cost curves. Following this idea, a 3D CNN model is constructed, which takes local matching costs as input and predicts whether the generated disparity value is accurate.

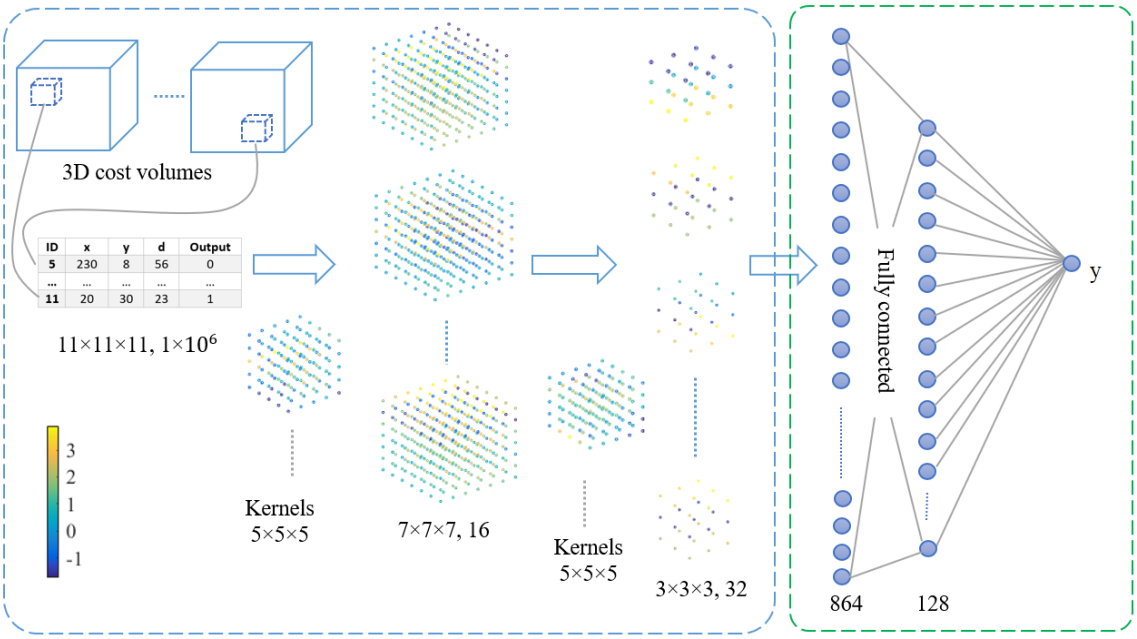


Figure 2.3: The proposed 3D CNN architecture. The input is represented as a set of 3D cost volumes (one for each stereo image pair) and a set of training samples. Each training sample is used to retrieve a $11 \times 11 \times 11$ local window from its associated cost volume, pixel coordinates, and estimated disparity. The samples go through two 3D convolutional layers to obtain 32 $3 \times 3 \times 3$ matrices. The 864 values in these matrices go through a fully connected layer before reaching the output node.

As shown in Figure 2.3, the experimental 3D CNN architecture has 5 layers, including input and output. Note that this architecture can be redesigned by including multiple convolutional layers and fully connected layers based on the training dataset. Cost values within $11 \times 11 \times 11$ local windows extracted from cost volumes are first processed using 16 $5 \times 5 \times 5$ convolutional kernels to extract features. The results are further processed by 512 $5 \times 5 \times 5$ convolutional kernels to generate 32 matrices.

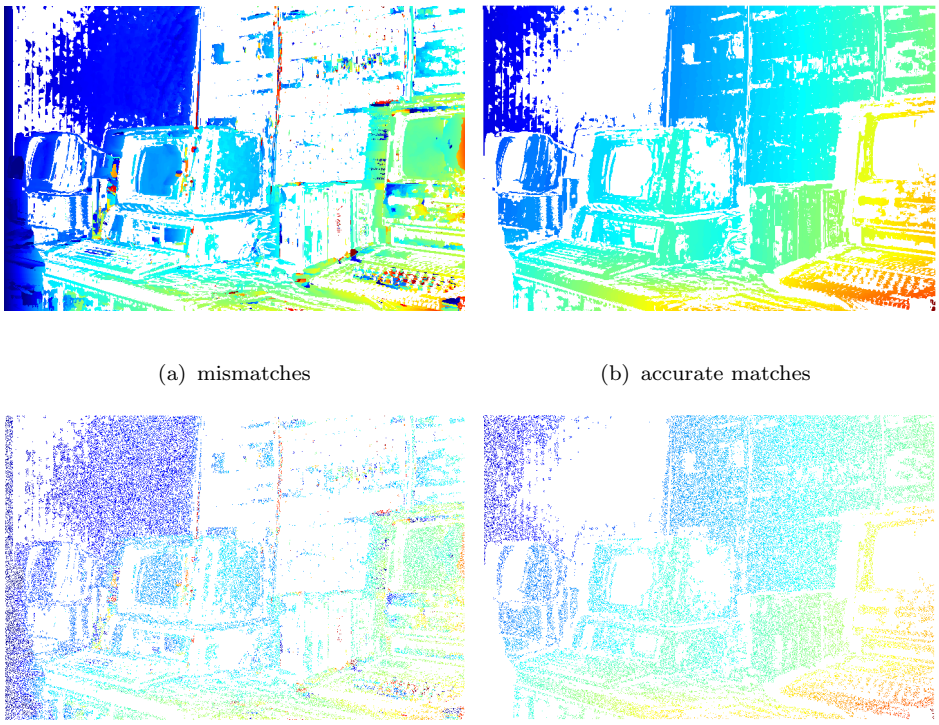
Values in these matrices are mapped to a total of 864 neurons, which go through a fully connected network to a hidden layer that has 128 neurons. Finally, these neurons are connected to the output neuron, which predicts the accuracy of the disparity value based on input local costs. The number of parameters between adjacent layers from left to right are 2016, 64032, 110720 and 128, respectively.

2.2.2 Training data

To obtain training data for the above CNN model, stereo image pairs with ground truth are needed. Here 11 image pairs were selected from the Middlebury 2014 stereo dataset [77] with consistent lighting condition between left and right images. The estimated disparity maps $D_e(x, y)$ are evaluated using the ground truth map $D_t(x, y)$ to identify mismatches. Here, a pixel (x, y) is considered as accurately matched if and only if $\|D_e(x, y) - D_t(x, y)\| < T$, where T is a threshold value. In many cases, the number of positive (accurately matched) and negative (mismatched) samples can be differ greatly as shown in Figure 2.4(a-b). To balance the two sides and to reduce the number of samples for training, the same number of samples from both sets are randomly chosen; see Figure 2.4(c-d). Note that pixels along the image boundary are excluded from sampling so that all selected samples have properly defined neighbors.

For each selected sample (x, y), cost values from an $11 \times 11 \times 11$ window centered at $(x, y, D_e(x, y))$ of the corresponding 3D cost volume are extracted to form a matrix M. For more effective training, costs in matrix M are normalized to zero-mean and one-variance:

$$M_{norm}(x, y, z) = \frac{M(x, y, z) - \text{mean}(M)}{\text{var}(M)}. \tag{2.3}$$



(a) mismatches (b) accurate matches

(c) negative samples (d) positive samples

Figure 2.4: Selection of training samples. Pixels in a given disparity map are first classified into mismatches (a) and accurate matches (b) using ground truth disparity. Negative (c) and positive samples (d) are randomly selected.

The CNN network is then trained to output value "1" for positive examples and "0" for negative examples.

2.2.3 Training method

In the described CNN architecture, two convolutional layers and two fully connected layers are used to implement a feedforward operation [50]. Gradient descent with momentum is used as the optimization algorithm. Each matrix $M_l^{n_l}$ in the convolutional layers (layer 2 and 3) is calculated by:

$$M_l^{n_l} = f\left(\sum_{i=1}^{N_{l-1}} \left(M_{l-1}^i * K_{(i,n_l)}\right) + b_l^{n_l}\right), \tag{2.4}$$

where l denotes the layer number, n_l refers toan individual matrix in layer l, and N_l is the total number of matrices in layer l. $K_{(i,j)}$ refers to the convolutional kernel, $*$ is 3D convolution operator, and $b_l^{n_l}$ is the bias.

Note that these two convolutional layers have different activation functions. $f(x) = \max(0, x)$ is applied to the first layer, whereas the sigmoid function $f(x) = (1+e^{-x})^{-1}$ is used for the second.

The two fully connected layers use the sigmoid functions for activation. Each neuron $a_l^{n_l}$ at location n_l of layer l is computed by:

$$a_l^{n_l} = f\left(\sum_{i=1}^{N_{l-1}} \left(a_{l-1}^i \cdot w_{(i,n_l)}\right) + b_l^{n_l}\right), \tag{2.5}$$

where $w_{(i,j)}$ is the weight between two neurons.

Once the final output a_t is computed for the last layer, its difference between the expected output y gives us the training error $e = (a_t - y)^2/2$. This training error is propagated backwards through 3D CNNs to update the weights and biases [50].

2.2.4 Postprocessing for subpixel accuracy

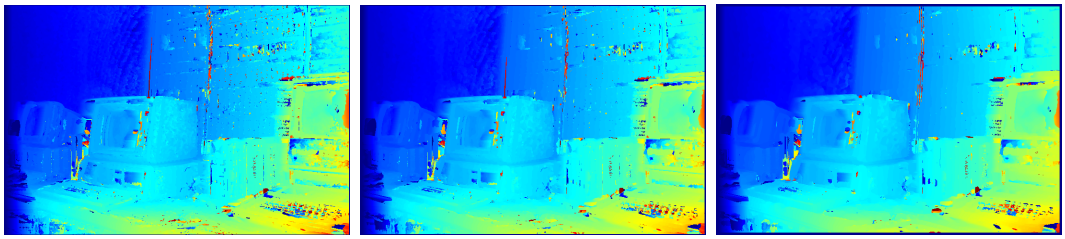
As shown in Figure 2.5, the sparse disparity maps generated through CNN-based filtering still contain isolated noises. As suggested by Taniai et al. [84], a 3×3 median filter are applied as postprocessing to remove the noise.

In addition, the disparity maps generated by WTA only have pixel level accuracy. To obtain disparity values at subpixel accuracy, an additional 3×3 bilateral filtering step can also be applied. During the bilateral filtering, if a pixel does not have disparity a value assigned, but 2/3 of its neighbors have disparity values, the result of the bilateral filtering will be assigned to this pixel. This approach helps to increase the density of output disparity maps.

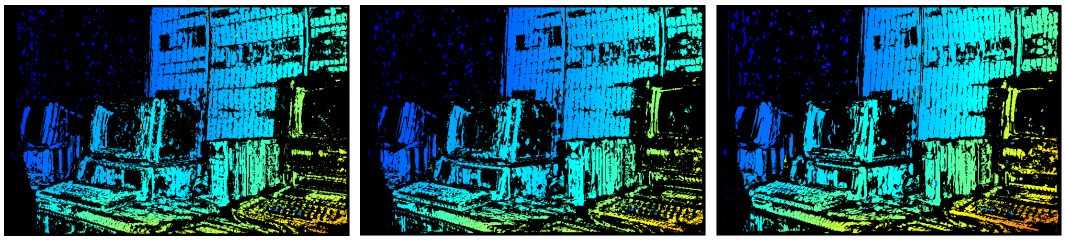
2.3 Experimental Results

To train a robust 3D CNN model, a large number of samples from different scenes are needed. Here, the Middlebury stereo 2014 datasets [77] are used, which have been commonly used to evaluate the state-of-art algorithms over the past few years. Since the proposed approach aim to detect mismatches with large disparity errors, quarter resolution images are used, which have resolution around 700×500 pixels. In addition, image pairs with dramatic lighting changes (i.e., ClassE, DjembL and PianoL) are not used here simply because the dense disparity maps generated using SSD cost aggregation and WTA disparity optimization contain too many mismatches to perform meaningful filtering. Hence, only the remaining 27 image pairs with similar lighting conditions are used for training and testing.

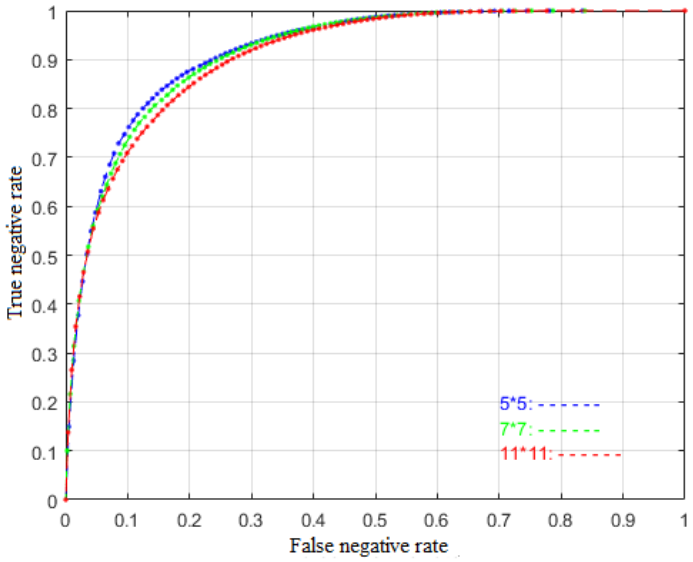
As mentioned in Section 2.2.2, when generating training samples, a pixel is selected



(a) 5×5 (b) 7×7 (c) 11×11



(d) error rate=2.45% (e) error rate=2.16% (f) error rate=2.54%



(g) ROC curve

Figure 2.5: Effectiveness of disparity filtering under different parameter settings: (a-c) disparity maps generated using WTA under different cost aggregation window sizes; (d-f) the corresponding CNN filtering results obtained under threshold $R = 0.5$; (g) the ROC curve obtained under different R values.

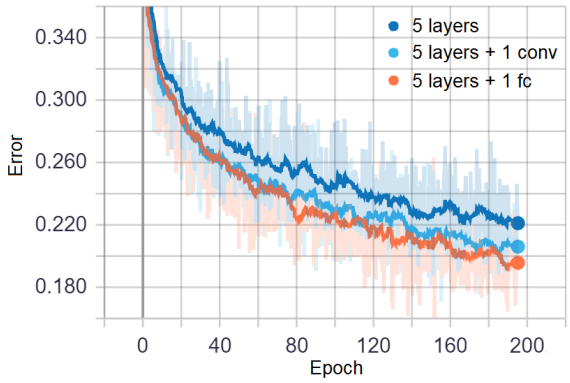


Figure 2.6: Change in training error as the number of iterations increases. Here, "conv" and "fc" denote the convolutional layer and the fully connected layer, respectively.

as a negative sample (mismatch) if the absolute difference between its disparity value and ground truth is greater than threshold T. To make sparse disparity matching results generated using quarter resolution images comparable with "bad 4.0" evaluation on the Middlebury Stereo Vision site, T was set to 1.

1 million samples are randomly ordered and organized as batches (64 samples in each batch) for unbiased training. The model was run for 200 iterations for all the samples. Figure 2.6 plots the training error that occurred during the training process. It shows that the classifier's performance gradually improves with more iterations and adding a convolutional layer or fully connected layer helps the model further reduce the error.

2.3.1 Impact of parameter settings

Threshold parameters Based on the input 3D cost matrix, the trained 3D CNN model outputs a single value a_t, which predicts the reliability of estimated disparity.

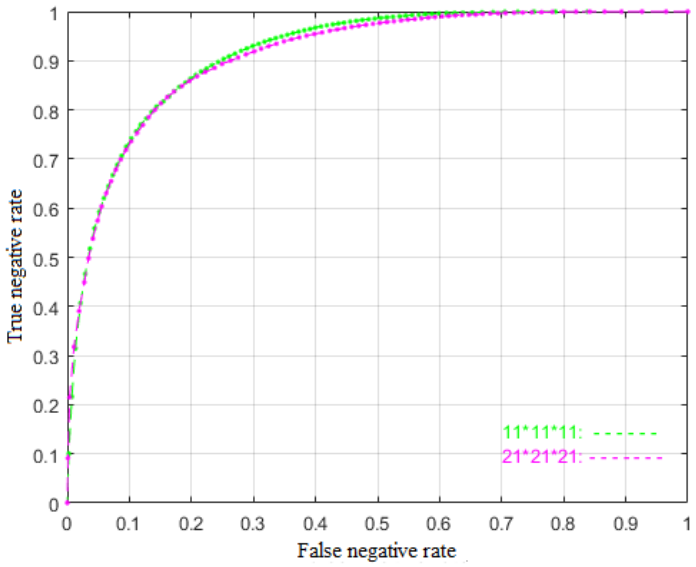
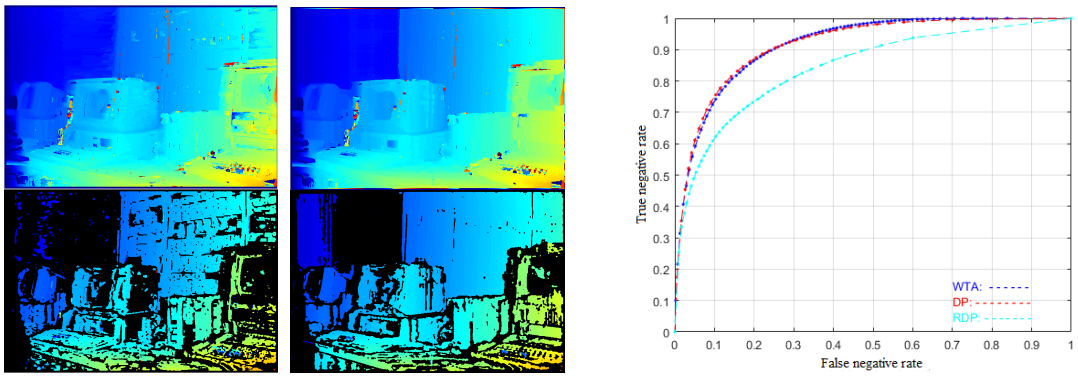


Figure 2.7: Results of 3D CNN models defined under different neighborhood sizes. Increasing the initial neighborhood size from $11 \times 11 \times 11$ to $21 \times 21 \times 21$ does not change the effectiveness of the filtering.

A second threshold value R is used here to filter out all disparity values with $a_t < R$. By definition, a higher R value leads to sparser but more accurate disparity matches, whereas a lower R yields denser but noisier matches. Changing the R value results in a ROC (Receiver Operating Characteristic) curve. The presented 3D CNN model can filter out more than 70% of mismatches with less than 10% false negatives, regardless of whether 5×5, 7×7 or 11×11 windows were used for cost aggregation.

Input neighborhood size Besides the network presented in Section 2.2.1, a different 3D CNN model that uses larger input neighborhood size is also tested. This second model takes a $21 \times 21 \times 21$ local cost matrix as input and uses an additional down-sampling layer after each convolutional layer to obtain the same number of neurons for feeding into the fully connected layer. Figure 2.7 compares the perfor-

mance between the two CNN models, which shows that larger input neighborhood size, which significantly increases computation time, does not necessarily improve the performance of disparity filtering. Hence, the model that uses a $11 \times 11 \times 11$ neighborhood size is chosen due to its lower computational cost.



(a) (b) (c)

Figure 2.8: Results from different matching cost generation and disparity optimization approaches: (a) result of DP optimization before and after CNN-based filtering; (b) result of RDP optimization; (c) the ROC curves obtained under different R values.

2.3.2 Performances on different stereo matching approaches

The 3D CNN model presented here is trained directly on the 3D cost volume and the corresponding disparity map. It does not make any assumption on how the disparity values are computed and hence can work with different stereo matching techniques. For test dataset, the 3D cost volumes are generated using local bilateral filtering cost aggregation, whereas the disparity maps are computed using simple WTA optimization. Here, the effectiveness of the disparity filtering on disparity maps generated

using Dynamic Programming (DP) and Reliability-based DP (RDP) [23] approaches is also tested. For the case of DP, the 3D CNN model is trained based on the same 3D cost volumes as the ones for WTA, but the corresponding disparity maps are generated using DP. The results demonstrate that the presented 3D CNN-based classifier can effectively label mismatches generated by a different disparity optimization technique; see Figure 2.8(a). The corresponding ROC curve (Figure 2.8(c)) is very close to the one obtained under WTA optimization.

The RDP, on the other hand, can be considered a global cost-aggregation approach. It propagates local matching costs along scanlines, which produces a set of aggregated cost volumes (A^R, A^L, A^D, and A^U for results obtained from left-to-right, right-to-left, up-to-down, and down-to-up directions, respectively). A new 3D cost volume $C'(x, y, d)$ can be calculated by:

$$\begin{aligned} C'(x, y, d) = A^R(x, y, d) + A^L(x, y, d) + A^U(x, y, d) \\ + A^D(x, y, d) - 2 * C(x, y, d). \end{aligned} \tag{2.6}$$

The final disparity maps are generated by running WTA optimization on $C'(x, y, d)$. To evaluate the effectiveness of the presented disparity filtering approach, the 3D CNN model is trained on $C'(x, y, d)$ and the corresponding WTA disparity values. Figure 2.8(b) shows that the model can effectively filter out mismatches. However, since the disparity maps generated by RDP contain fewer mismatches than those of WTA and DP, the corresponding ROC curve is lower than those plotted for the latter two.

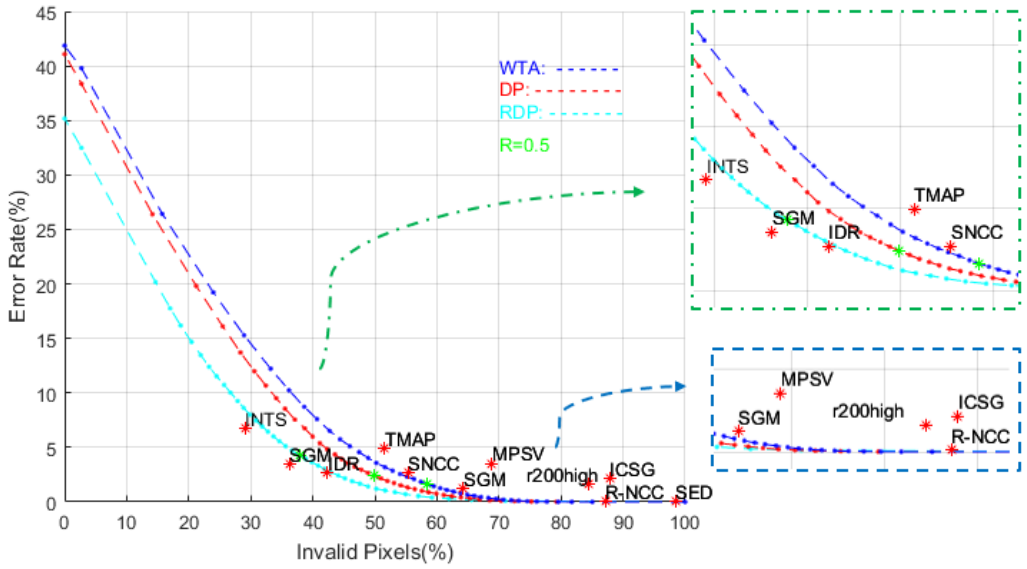
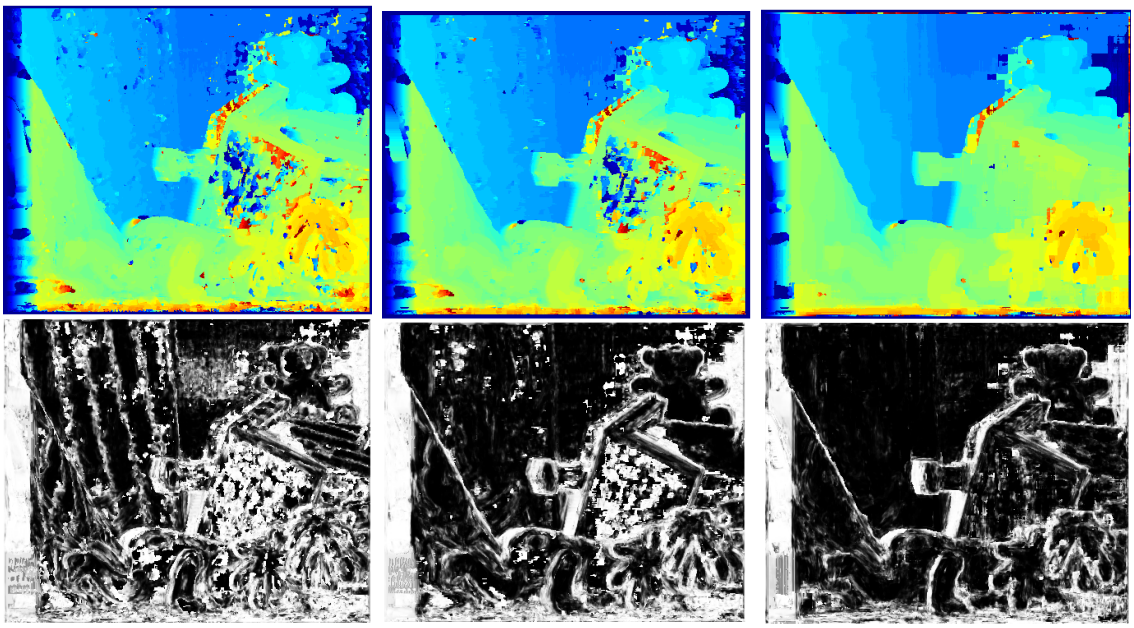


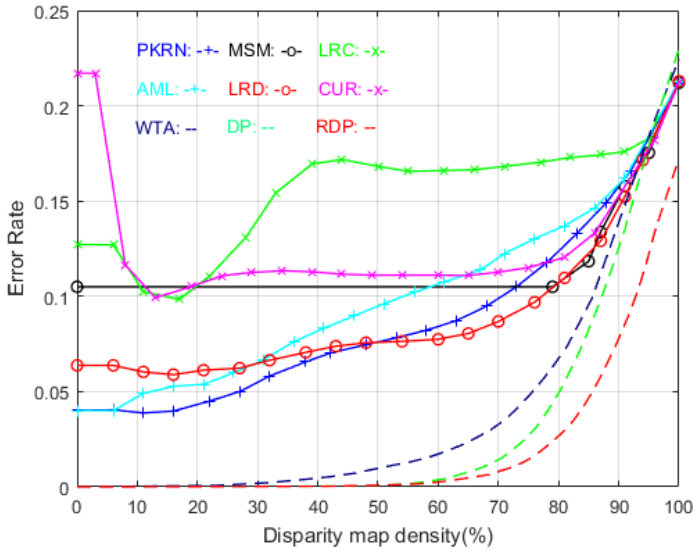
Figure 2.9: Comparison with existing approaches including SED [66], R-NCC (anonymous), r200high [43], MPSV [6], ICSG [82], SGM [29], IDR [47], TMAP [72] INTS [34] and SNCC [16] on the Middlebury Stereo Evaluation site.

2.3.3 Comparison with existing methods

The approach is performed on both "training sparse" and "test sparse" datasets provided by the Middlebury Stereo Evaluation site. The evaluation results on the "test sparse" dataset are accessible at `http://vision.middlebury.edu/stereo/eval3/`, which show that the approach (referred as "DF") ranks 3^{rd} under the "bad 4.0" category. The Middlebury evaluation ranks different algorithms based on the error rates of generated disparity maps only, and hence favors approaches that output fewer disparity matches (that is, more invalid pixels). To compare different algorithms based on both performance metrics, here the error rates vs. invalid pixels rate graph under different settings of the threshold parameter R is plotted in Figure 2.9. The evaluation clearly shows that the proposed approach outperforms 8 of the 11 existing



(a) (b) (c)



(d)

Figure 2.10: Confidence measures: (a-c) dense disparity generated by WTA, DP, and RDP (top), as well as the corresponding confidence maps for the "Teddy" dataset (bottom); (d) comparison with existing confidence measures on error rate over disparity map density [30]

approaches. The reason that the approach does not perform as well as INTS [34], SGM [29] and IDR [47], is mainly because the disparity maps generated by the WTA optimization contain a high percentage of mismatches.

Apart from filtering mismatches, the output of the 3D CNN model can also be used as confidence measures for estimated disparity values. Here, the approach is compared with existing confidence measures using the same "Teddy" pair [77]. The Area Under the Curve (AUC) metric introduced by Hu and Mordohai [30] is used as the metric and the curves for different approaches are plotted in Figure 2.10. It is worth noting that the dense disparity maps generated by RDP is more accurate than those by DP, which are also better than WTA. Hence, the AUC for RDP with the presented filtering is the smallest, whereas the AUC for WTA with the filtering is the largest. Nevertheless, in all 3 cases, the AUC values are smaller than existing approaches reported in Hu et al. [30]; see Table 2.1.

2.4 Summary

A novel disparity filtering approach is presented in this chapter, which is based on a binary classifier trained using a 3D CNN model. It is possible to infer whether the disparity values generated by a given disparity computation algorithm are accurate or not based on local 3D matching costs. Evaluations using the Middlebury Stereo Vision web page show that the proposed approach is comparable to most of the existing sparse stereo matching techniques. Additional comparisons also demonstrate that the approach is more effective than traditional confidence measures.

The matching costs involved in this chapter were computed using SSD, which

Table 2.1: AUC for different approaches.

Method	*AUC*
MSM [15]	0.162
CUR [15]	0.126
LRC [30]	0.115
AML [62]	0.096
LRD [30]	0.089
PKRN [30]	0.086
the proposed(WTA)	0.038
the proposed(DP)	0.030
the proposed(RDP)	0.020

lacks the capability to address image pairs with lighting changes and lack of texture. How to use a learning-based approach to enhance the cost computation process and address these challenges is investigated in Chapter 3.

Chapter 3

Semi-dense Stereo Matching using Dual CNNs

The disparity filtering approach in Chapter 2 is proposed to select accurate matches from disparity maps generated by a traditional stereo matching algorithm. When handling image pairs captured under varying lighting conditions or for textureless objects, the disparity maps typically suffer from a deficiency of valid matches, leading to extremely sparse results. In this chapter, a fully learning-based pipeline to tackle challenging situations is explored.

Approaches have been proposed for generating matching cost volumes (that is, disparity space images) using CNNs [64, 97, 103]. While inspiring results are generated, these existing approaches are not robust enough to handle ambiguous cases as referred above. Heuristically-defined post-processing steps are often applied to correct mismatches. The hypothesis here is that the performance of CNNs can be noticeably improved if more information is fed into the network. Hence, instead of

trying to correct mismatches as post-processing, a pre-processing step is introduced to perform image transforms that are robust against lighting changes and can add distinguishable patterns to textureless areas. The output of these transforms are used as additional information channels, together with grayscale images, for training a matching CNN model.

The experimental results show that the model learned can effectively separate correct stereo matches from mismatches so that accurate disparity maps can also be generated using the simplest WTA optimization as in Chapter 2.

Learning-based approaches have also been proposed to compute confidence measures for generated disparity values so that mismatches can be filtered out [10, 81, 97]. Following this idea, a second CNN model is designed to evaluate the disparity map generated through WTA. Trained with only one input image and the disparity map, this evaluation CNN model can effectively filter out mismatches and produce accurate semi-dense disparity maps.

Figure 3.1 shows the pipeline of the whole process. Since both matching cost generation and disparity confidence evaluation are performed using a learning-based approach, the algorithm contains very few handcrafted parameters. The experimental results on the Middlebury 2014 stereo dataset [77] demonstrate that the present dual-CNN algorithm is comparable to most existing sparse stereo techniques.

3.1 Related Work

In addition to the literature review in Chapter 2, Zhang et al. [104] used CNNs and SGM to generate initial disparity maps and further combine Left-Right Differ-

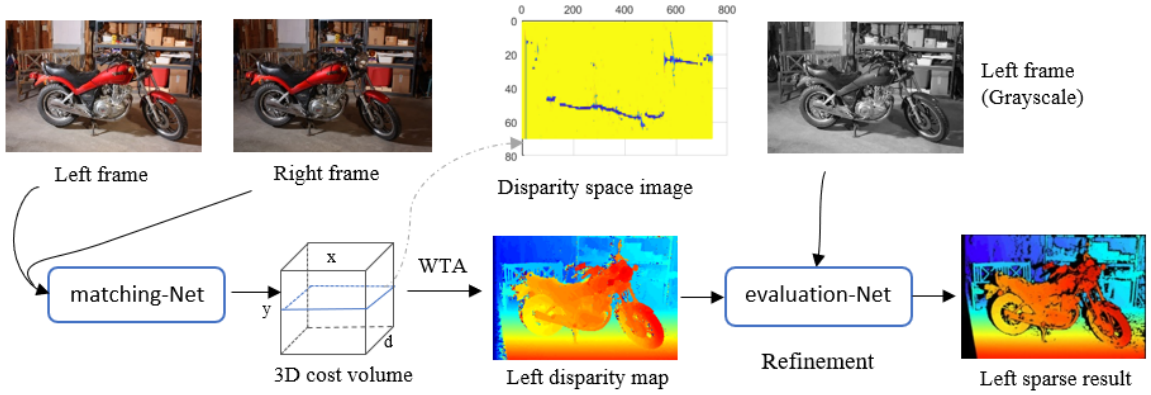


Figure 3.1: Semi-dense stereo matching pipeline. Given a pair of rectified images [77], how well a pair of image patches match is evaluated using *a matching-CNN model*. The results form a matching cost volume, from which a disparity map is generated using simple WTA optimization. Finally, an *evaluation-CNN model* is applied to filter out mismatches.

ence (LRD) [30] with disparity distance regarding local planes to perform confidence checks. In addition, they adopted segmentation and surface normal constraints within the post-processing to enhance the reliability of disparity estimation. To fully utilize the ability of CNNs in terms of feature extraction, Park and Lee [64] proposed a revised CNN model based on a large pooling window between convolutional layers for wider receptive fields to compute the matching cost, and they performed a similar post-processing pipeline as introduced in Zbontar et al. [103]. Another model revision, similar to Park and Lee's work [64], was introduced by Ye et al. [97], which used a multi-size and multi-layer pooling scheme to take wider neighboring information into consideration. Moreover, a disparity refinement CNN model was later demonstrated in their post-processing to blend the optimal and suboptimal disparity values. Both

the above revisions presented solid results in image areas with little or no texture, disparity discontinuities and occlusions.

Attempts were also made to train end-to-end deep learning architectures for predicting disparity maps from input images directly, without the needs of explicitly computing the matching cost volume [9, 41, 63]. As a result, these end-to-end models are efficient but require larger amount of GPU memory than the previous patch-based approaches. More importantly, these models were often trained on stereo datasets with specific image resolutions and disparity ranges and hence, cannot be applied to other input data. This restriction also limits the feasibility of training CNNs to concurrently preserve geometric and semantic similarity [12, 79, 93].

Once dense disparity results are generated, confidence measures can be applied to filter out inaccurate disparity values in the disparity refinement step. Quantitative evaluations on traditional confidence measures were presented by Hu and Mordohai [30], and the most recent review was given by Poggi et al. [69]. The proposed approach in this chapter is similar to these recent works [10, 81, 97], which compute confidences through training 2D CNNs on 2D image or/and disparity patches. A key difference is, however, that only the left image and its raw disparity map generated by WTA are used to train a confidence CNN model, whereas existing approaches require the generation of both left and right disparity maps.

3.2 Methodology

In this chapter, a robust and learning-based stereo matching approach is developed by assigning disparity values only for pixels with visual cues. As shown in Figure 3.1,

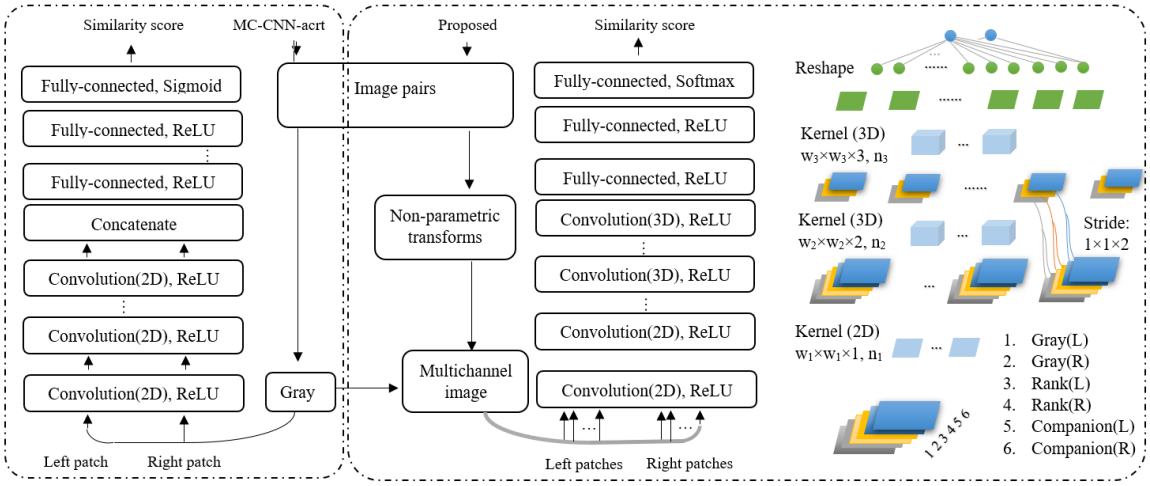


Figure 3.2: Comparison between the baseline architecture "MC-CNN-arct" in Zbontar et al. [103] and the proposed model matching-Net. The left and right image patches for the latter are selected from an image collection, which includes not only grayscale images, but also channels generated by non-parametric transforms. In addition, the concatenation operation is replaced by 3D convolution, which can separately group different transforms by adjusting stride size in the third dimension; see Section 3.3 for model configuration.

two CNN models, referred to as matching-Net and evaluation-Net, are utilized in the proposed pipeline: matching-Net is constructed as the substitution of matching cost computation and aggregation steps, and outputs a matching similarity measure for each pixel pair; evaluation-Net performs confidence measures on the raw disparity maps generated by WTA based on the similarity scores.

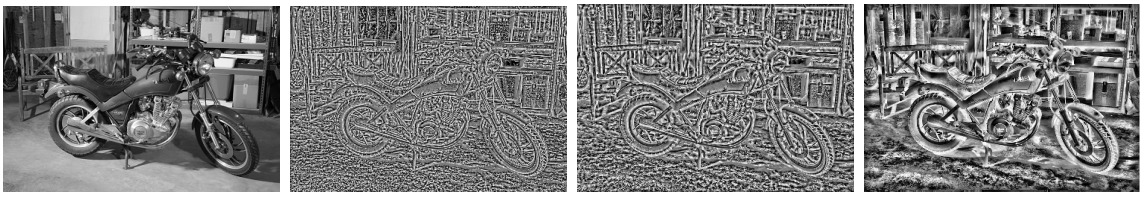
3.2.1 Matching-Net

The matching-Net model serves the same purpose as the "MC-CNN-arct" in Zhang et al. [103], but there are several key differences; see Figure 3.2. First of all, the neural network is fed with additional global information (namely, results of non-parametric transformations) that are difficult to generate through convolutions. Secondly, 3D convolution networks are employed, which was found to improve the performance. It is worth noting that the proposed approach is also different from other attempts to improve "MC-CNN-arct", which use very large image patches and multiple pooling sizes [64, 97]. These approaches require an extensive amount of GPU memory, which limits their usage. In order to feed global information into the network trained on small patches, the strategy adopted is to perform non-parametric transforms.

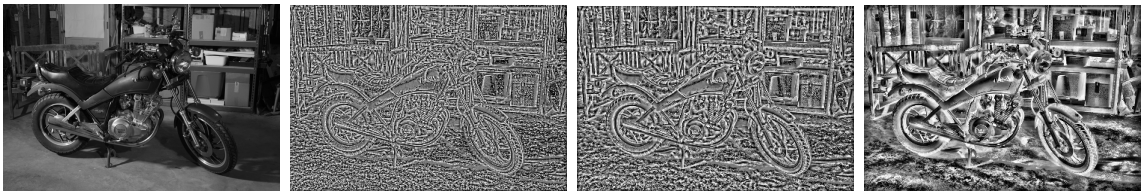
3.2.1.1 Lighting Difference

For robust stereo matching, lighting differences as an external factor cannot be neglected. To address this factor, "MC-CNN-arct" manually adjusts the brightness and contrast of image pairs to generate extra data for training. However, datasets with lighting differences may vary from one to another, making it hard to train a model that is robust to all cases.

Aiming for an approach with less human intervention, the used of rank transform is proposed to ameliorate lighting variations between image pairs. As a non-parametric local transform, rank transform was first introduced by Zabih [101] to achieve better visual correspondence near discontinuities in disparity. This endows stereo algorithms based on rank transform with the capability to perform similarity estimation for image



(a) Grayscale (b) 15×15 (c) 31×31 (d) 61×61



(e) Grayscale (f) 15×15 (g) 31×31 (h) 61×61

Figure 3.3: Results of "MotorE" dataset (left image on top row and right image on bottom) using rank transform under different neighborhood sizes. Larger windows generally lead to smoother results, but at the expense of losing fine information.

pairs with different lighting conditions.

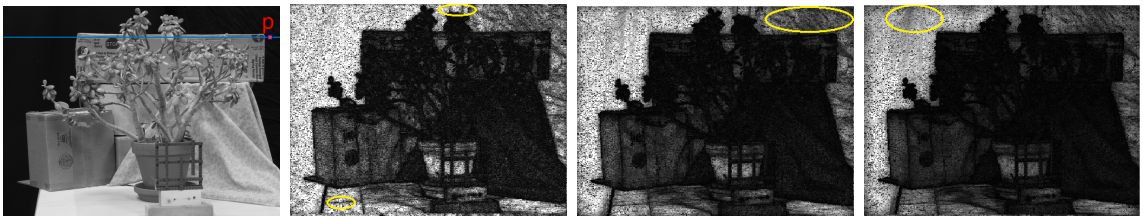
The rank transform $R(p)$ for pixel p in image I is computed as:

$$R(p) = \frac{1}{|N_p|} \sum_{q \in N_p} (I(q) > I(p)?1 : 0) \ , \tag{3.1}$$

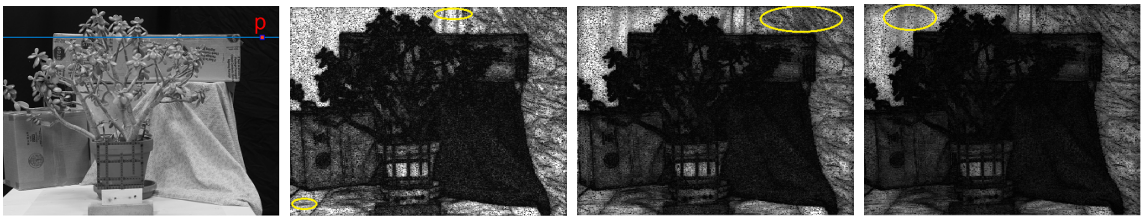
where N_p is a set containing pixels within a square window centered at p. $|S|$ is the size of set S. Figure 3.3 shows the results of rank transform under different window sizes, where the difference of lighting conditions is ameliorated effectively.

3.2.1.2 Low Texture

Besides lighting variations, low or no texture poses another challenge for stereo matching. For a given pixel p within a texture-less region, the best way to estimate its disparity is based on its neighbors who have similar depth but are in texture-rich



(a) Grayscale (b) 15×15 (c) 31×31 (d) 61×61



(e) Grayscale (f) 15×15 (g) 31×31 (h) 61×61

Figure 3.4: Results of companion transform under different neighborhood sizes for the "Jadepl" dataset (left image on top row and right image on bottom). The transformation results are brightened for better viewing. The results show that companion transform successfully adds distinguishable features to texture-less areas; see regions highlighted.

areas (that is, they have sufficient visual cues for accurate disparity estimation). Traditional stereo algorithms [78] mostly utilize cost aggregation, segmentation-based matching, or global optimization for disparity computation to handle ambiguous regions. As mentioned above, the intention here is to feed the neural networks with global information. Hence, a novel "companion transform" is designed and applied in the pre-processing step.

The idea of using a companion transform is inspired by SGM [29], which suggests performing smoothness control by minimizing an energy function on 16 directions. Here, the goal is to design a transformation that can add distinguishable features

to texture-less areas. Hence, for a given pixel p, the number of pixels that: 1) have the same intensity as p and 2) lie on one of the paths started from p as in Hirschmuller [29] is computed. These pixels are referred to as p's companions and the transform as companion transform. In practice, 8 ray directions (left, right, up, down, and 4 diagonal directions) work well, though other settings (4 or 16 directions) can also be used.

$$C(p) = \frac{1}{|N_p \cap \Omega_p|} \sum_{q \in N_p \cap \Omega_p} (I(q) == I(p)?1 : 0) \ , \tag{3.2}$$

where Ω_p is a set containing pixels on the paths starting from p.

Figure 3.4 shows the results of companion transform under different window sizes. Figure 3.5 further illustrates how the companion transform result adds distinguishable patterns to pixels in texture-less areas.

3.2.1.3 Training Data

To train the CNN model, the 15 image pairs from the Middlebury 2014 stereo training dataset [77], which contains examples of lighting variations and texture-less areas, are utilized. Each input image is first converted to grayscale before applying rank and companion transforms. The outputs of the two transforms, together with the grayscale images, form multi-channel images. Each training sample contains a pair of image patches centered at pixel (x, y) in the left image and $(x - d, y)$ in the right image, respectively. The input sample is assembled into a 3D matrix $M[w, w, 2 \times l]$, where w is the size of the patches and l is the number of channels in the multi-channel image. The ground-truth disparity values provided by the dataset are used to select matched samples and random disparity values other than the ground truth are used

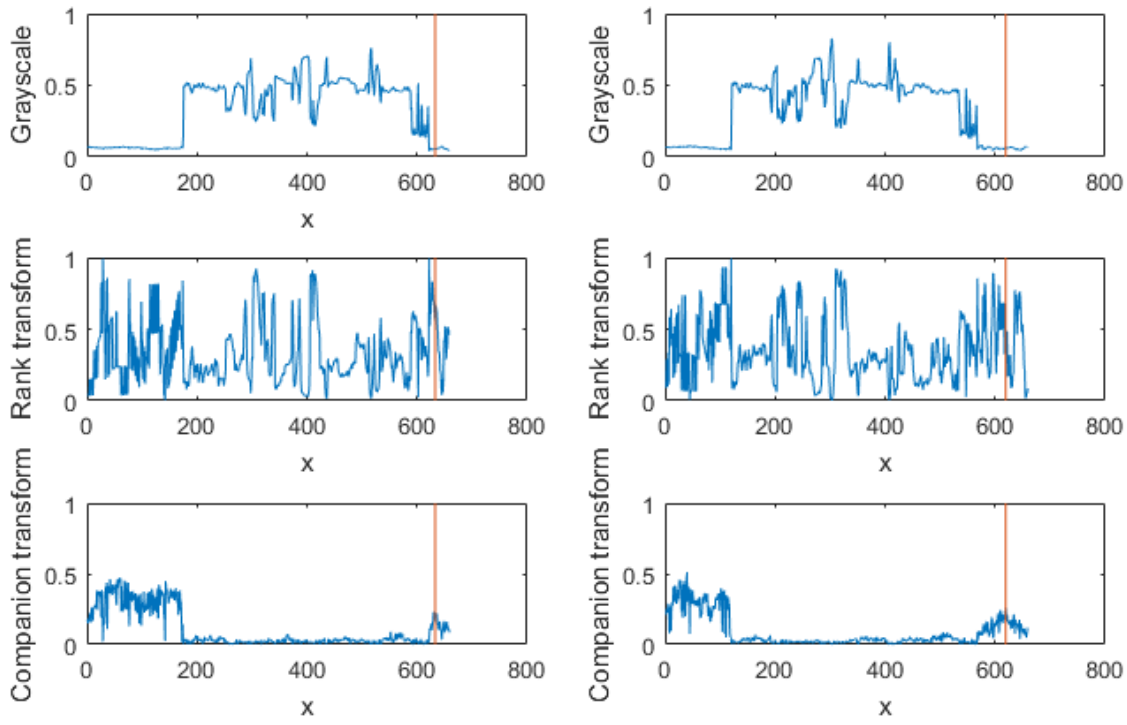


Figure 3.5: Comparison among information carried in different channels (grayscale, 31×31 rank transform, and 61×61 companion transform). The curves are plotted based on the values of different pixels on the same row marked in blue in Figure 3.4. The left side shows the left view, with the position of the target pixel p marked by red vertical lines. The right side shows the right view, where the red line shows the position of the correct corresponding pixel of p. Due to the lack of texture, neither the grayscale nor the rank transform channels provide distinguishable patterns for matching. The companion transform can amend information that is useful for the matching-CNN.

to generate mismatched samples. Similar to Zbontar et al. [103], the sample matching hypothesis is adopted so that the same number of matches and mismatches are used for training. The proposed matching-Net is then trained to output a value of "0" for correct matches and "1" for mismatches.

3.2.2 Disparity Computation

For each novel stereo image pair, the matching-Net trained above is used to generate a 3D cost volume $C_s(x, y, d)$, where the value at location (x, y, d) stores the cost of matching pixel (x, y) in the left image with $(x - d, y)$ in the right image. The higher the value, the more likely the corresponding pair of pixels are mismatches since the network is trained to output "1" for mismatches. Unlike many existing approaches that resort to complex and heuristically designed cost aggregation and disparity optimization approaches [78], this approach relies on the learning network to distinguish correct matches from mismatches. Expecting the correct matches to have the smallest values in the cost volume $C_s(x, y, d)$, the simplest WTA optimization is applied to compute the raw disparity map.

3.2.3 Evaluation-Net

The matching-Net is trained to measure how well two images patches match. It makes decisions locally and does not check the consistency among best matches found for neighboring pixels. When the raw disparity maps are computed by local WTA, they inevitably contain mismatches, especially in occluded and low-textured areas. To filter out these mismatches, another CNN model, evaluation-Net, is constructed to

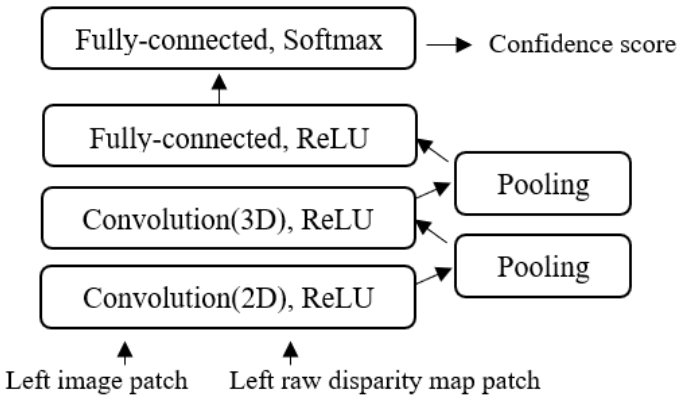
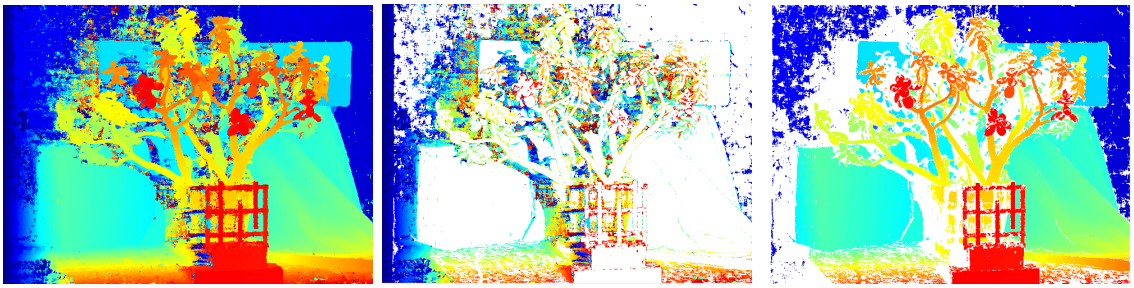


Figure 3.6: Architecture used for the evaluation-Net. The image patches here are generally bigger than those used in the matching-Net, therefore, multiple pooling layers are added for efficiency. Detailed model configuration can be found in Section 3.3.

implement consistency checks and compute confidence measures.

Learning-based confidence measures have been successfully applied to detecting mismatches and further improving the accuracy of stereo matching [69]. Similar to the 2D CNN model for error detection proposed in Ye et al. [97], only left images and their disparity maps are selected to train the model. A key difference, however, is that no handcrafted operation is involved in the proposed approach to fuse left and right disparity maps. In addition, the network contains both 2D and 3D convolutional layers to effectively identify mismatches from disparity maps; see Figure 3.6. 3D convolution is adopted here to allow the network to learn from the correlation between pixels' intensities and disparity values.

The evaluation-Net is trained using both matches and mismatches in the estimated disparity maps $D_e(x, y)$ for all training images. Mismatches are identified by comparing $D_e(x, y)$ with ground-truth disparity maps $D_t(x, y)$. Here, a pixel (x, y) is



(a) raw disparity (b) mismatches (c) matches

Figure 3.7: Training samples. Pixels in a given disparity map (a) are classified into mismatches (b) and accurate matches (c) using ground-truth disparity.

taken as mismatched if and only if

$$\|D_e(x,y) - D_t(x,y)\| > T_e \ , \tag{3.3}$$

where T_e is a threshold value commonly assigned with 1 pixel; see Figure 3.7(b-c).

In the estimated disparity map $D_e(x,y)$, the majority of pixels have correct disparity values, resulting in much more positive (accurately matched) samples than negative (mismatched) samples. Hence, all negative samples were collected and the same number of positive samples are randomly generated. For each selected sample (x,y), grayscale and estimated disparity values from patches centered at (x,y) are extracted to form a 3D matrix. The evaluation-Net is then trained to output value "0" for negative samples and "1" for positive samples. The output of the evaluation-Net can then be used to filter out potential mismatches that achieve scores lower than a confidence threshold R.

3.3 Experimental Results

In this section, the hyperparameters [103] for both of the proposed CNN models will be presented and followed by a set of performance evaluations. The goal of the evaluations is to determine: 1) whether the non-parametric transforms can help improve the disparity map accuracy generated using the matching-Net; and 2) how well the overall dual-CNN approach performs compared to the state-of-the-art sparse stereo matching techniques.

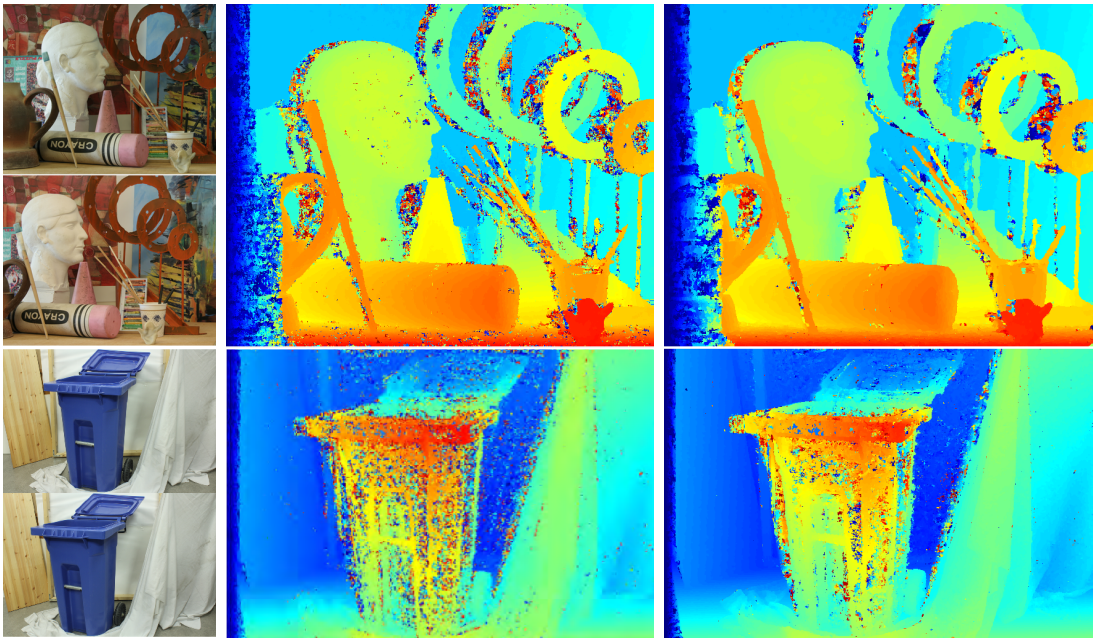
Hyperparameters and implementations: The input of the matching-Net is a 3D matrix that consists of $l = 6$ layers in the experiment. Both left and right images contains 3 layers, including the grayscale image, a rank transform, and a companion transform respectively. Here, the transform window sizes w_r and w_c are set to 31 and 61. Different layers from the left and right images are stored in the matrix in alternating order. For the evaluation-Net, the input contains only two layers of data: one is the grayscale image and the other the raw disparity map, both from the left image. Table 3.1 shows the hyperparameters of the experimental models.

The implementation of the CNN models are based on Tensorflow using classification cross-entropy loss, $-(t \log s + (1-t) \log(1-s))$, where s denotes the output value. Here, t was set to 1 for mismatches and to 0 for matches to train the matching-Net as in "MC-CNN-acrt", but $t = 1$ for positive samples and $t = 0$ for negative samples to perform confidence measure through the evaluation-Net. Both models utilize a gradually decreasing learning rate from 0.002 to 0.0001, and arrive a stable state after running 20 epochs on full training data.

Effectiveness of non-parametric transforms: The overall structure of "MC-

Table 3.1: Hyperparameters of the matching-Net and evaluation-Net. Here, "Conv", "Mp" and "Fc" denote the convolutional, the max pooling, and the fully connected layers respectively.

matching-Net		evaluation-Net	
Attributes	Kernel, quantity and stride	Attributes	Kernel, quantity and stride
Input	$11 \times 11 \times 6$, 1	Input	$101 \times 101 \times 2$, 1
Conv1(2D)	$3 \times 3 \times 1$, 32, $1 \times 1 \times 1$	Conv1(2D)	$3 \times 3 \times 1$, 16, $1 \times 1 \times 1$
Conv2(3D)	$3 \times 3 \times 2$, 128, $1 \times 1 \times 2$	Mp1	$2 \times 2 \times 1$, 16, $2 \times 2 \times 1$
Conv3(3D)	$3 \times 3 \times 3$, 64, $1 \times 1 \times 1$	Conv2(2D)	$3 \times 3 \times 1$, 32, $1 \times 1 \times 1$
FC1	1600	Mp2	$2 \times 2 \times 1$, 32, $2 \times 2 \times 1$
FC2	128	Conv3(2D)	$3 \times 3 \times 1$, 64, $1 \times 1 \times 1$
Output	2	Mp3	$2 \times 2 \times 1$, 64, $2 \times 2 \times 1$
		Conv4(3D)	$3 \times 3 \times 2$, 128, $1 \times 1 \times 1$
		Mp4	$2 \times 2 \times 1$, 128, $2 \times 2 \times 1$
		FC1	128
		Output	2



(a) (b) (c)

Figure 3.8: Comparison of dense disparity maps. The top stereo image pair ("ArtL") contains lighting condition changes, whereas the bottom pair ("Recyc") contains areas with low texture (a). Compared to the results generated by "MC-CNN-acrt" (b), the disparity maps generated by the proposed approach (c) are much smoother.

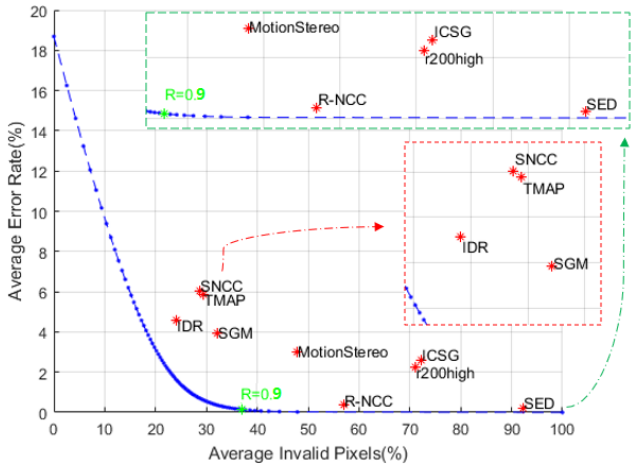
CNN-acrt" and matching-Net are quite similar. The key difference is that the input patches of "MC-CNN-acrt" are grayscale images only, whereas the matching-Net uses additional non-parametric transforms. Hence, to evaluate the effectiveness of non-parametric transforms, the raw disparity maps generated by the two approaches are compared. Based on the same training dataset from Middlebury [77], Figure 3.8 directly compares the raw disparity maps generated by "MC-CNN-acrt" and matching-Net. It suggests that the additional transforms allow the network to better handle challenging cases. The raw disparity maps of 15 training pairs generated by

the matching-Net achieves 18.69% regarding the mean percentage error (MPE) (over 1-pixel difference for half resolution) of non-occlusion areas compared to 22.91% by "MC-CNN-acrt".

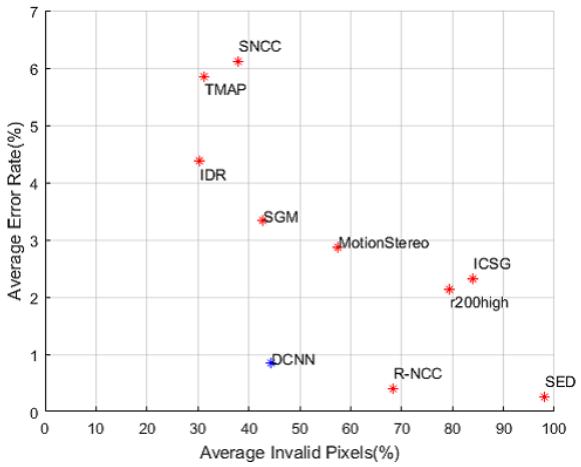
Comparison with sparse stereo matching approaches: Almost all state-of-the-art sparse stereo matching approaches have submitted their results to the Middlebury evaluation site [77]. The proposed approach (referred as "DCNN") on "test sparse" currently ranks 3^{rd} under the "bad 2.0" category. I would like to emphasize that simply comparing error rates of sparse disparities maps does not offer the whole picture on algorithm performance as it favors approaches that output fewer disparity values (that is, more invalid pixels). For a fair comparison, a non-occlusion error rates vs. invalid pixels rates plot is used to show the performance of different approaches on both the training and testing dataset; see Figure 3.9. The comparison suggests that the proposed approach under the $R = 0.9$ setting provides a very good balance between output disparity density and disparity accuracy. In addition, the plot on the training dataset also shows that, under the same output disparity density, The approach presented here provides lower non-occlusion error rates than existing approaches. Figure 3.10 further visually compares the disparity maps generated by different approaches.

The root-mean-square (RMS) metric [77] is also used here for evaluation. Since square errors are used, the RMS metric provides a stronger penalty to large disparity errors than the average absolute error ("avgerr") metric. The proposed approach on the testing dataset currently ranks on the top under the "rms" category; see Table 3.2.

AUC evaluation: The AUC metric introduced by Hu and Mordohai [30] has



(a) training



(b) testing

Figure 3.9: Comparison with the top ten approaches on the Middlebury Stereo Evaluation site [77]: SED [66], R-NCC (anonymous), r200high [43], ICSG [82], SGM [29], MotionStereo (anonymous), IDR [47], TMAP [72] and SNCC [16]. Performances of different approaches on both training (a) and testing (b) datasets are plotted on non-occlusion error rates v.s. invalid pixels rates plot. The relative position of these approaches on the two datasets are similar. On training datasets, where the ground truth disparity maps are available, the performance of the proposed approach under different confidence threshold R settings is shown as a curve.

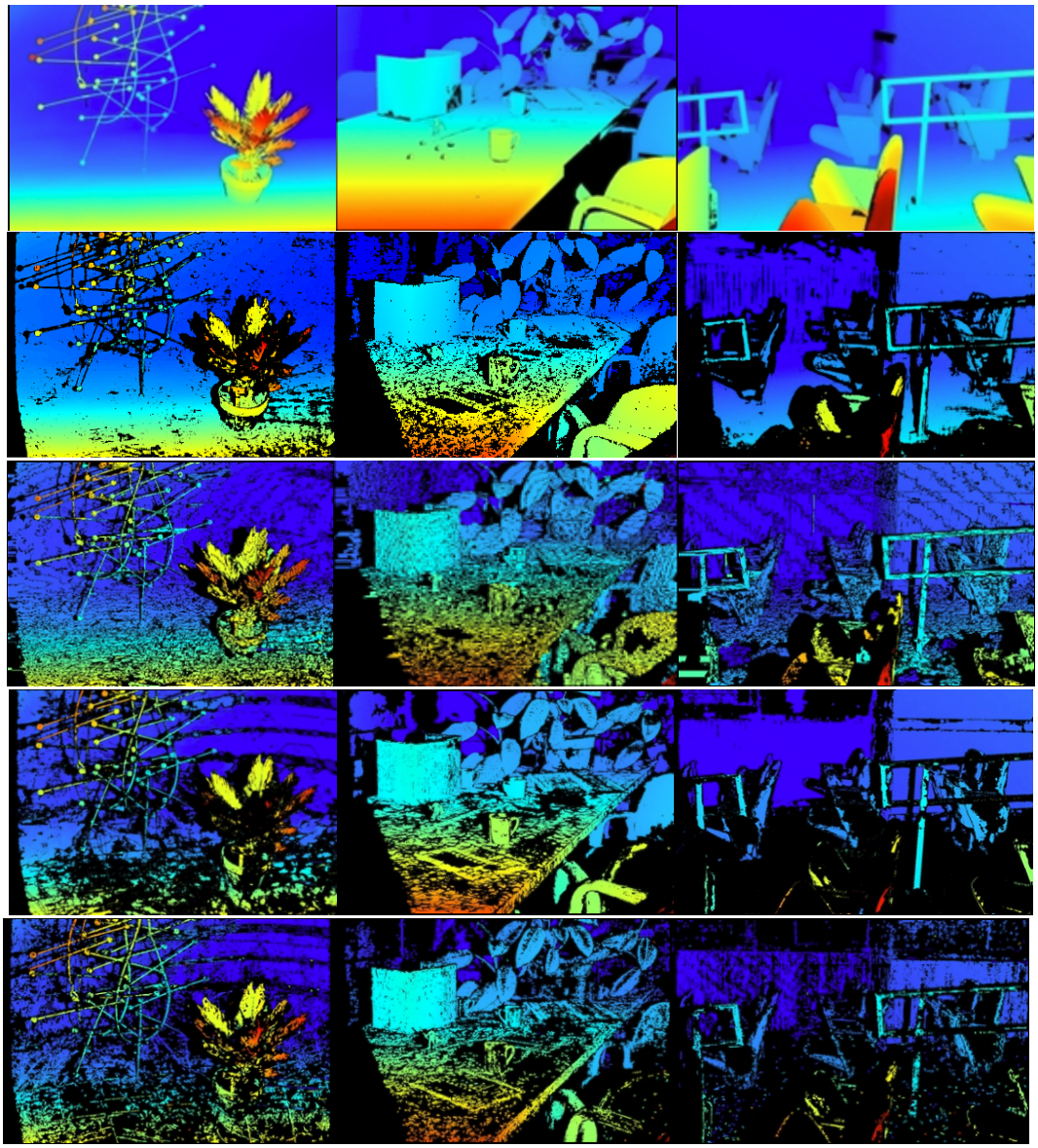


Figure 3.10: Comparison of sparse disparity maps for “Austr”, “Crusa” and “ClassE” (with lighting variation) of the testing dataset on the Middlebury Stereo Evaluation site [77]. The first row shows the ground truth, and rows 2 to 5 are the disparity maps generated by DCNN, TMAP [72], IDR [47] and SGM [29] respectively.

Table 3.2: Comparisons of the state-of-the-art approaches under the RMS metric.

Name	RMS
DCNN	3.86_1
R-NCC(anonymous)	4.61_2
IDR [47]	8.07_3
MPSV [5]	9.25_4
INTS [34]	10.6_5
SGM [29]	10.9_6

been used as a metric for evaluating various confidence measures over the past few years. It measures how effectively the confidence measures can filter out mismatches under different parameter settings, rather than only checking the performance under one set of parameters. Since a large set of sparse disparity maps need to be evaluated, this measure can only be computed on datasets with published ground truth. Following the practice in Tosi et al. [89], the dual-CNN approach is trained only on the 13 additional image pairs with ground truth from Middlebury [77] and then tested on the 15 training image pairs. The proposed approach achieves a competitive mean AUC value of 0.0522 compared to 0.0728, 0.0680 and 0.0637 attained respectively by the state-of-the-art approaches APKR [45], O1 [67] and CCNN [68] reported in Tosi et al. [89], which compares various confidence measures on the raw disparity maps from Zbontar et al. [103].

3.4 Summary

A novel learning-based semi-dense stereo matching algorithm is presented in this chapter. The algorithm employs two CNN models. The first model evaluates how well two image patches match. It serves the same purpose as "MC-CNN-acrt", but takes additional rank and companion transforms as input. These two transforms introduce global information and distinguishable patterns into the network; and hence areas with lighting changes and/or lack of texture can be more accurately matched. As a result, the optimal disparity values can be computed using the simplest WTA optimization. No complicated global disparity optimization algorithms or additional post-processing steps are required. The second CNN model is used for evaluating the disparity values generated and filtering out mismatches. Taking only one of the stereo images and the disparity map as input, the evaluation-Net can effectively label mismatches, without the needs for heuristically designed process such as left-right consistency check and median filtering.

The pipeline introduced in this chapter is limited to binocular stereo cases. How to handle multi-view stereo matching for 3D reconstruction applications is discussed next.

Chapter 4

A Global-Matching Framework for Multi-View Stereopsis

The previous two chapters focused on binocular stereo matching and proposed compelling solutions to address various challenges. This chapter explores how to reconstruct 3D models from images captured under different perspectives based on stereo correspondence, which is termed Multi-View Stereopsis (MVS). Substantial efforts have been made in this research field. A well-established pipeline starts from imagery collection to model refinement [17].

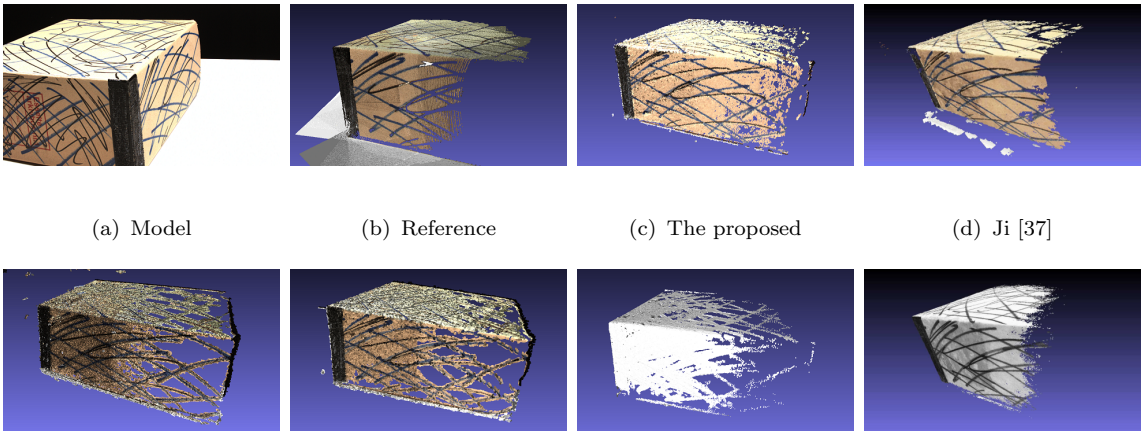
Given a 3D point p captured by a set of images, supporting domains from neighboring images are used to compute p's 3D location under the epipolar constraint. Although traditional matching algorithms generate promising results, many attempts on training neural networks to select potential matching pairs have been made over the past few years. Unlike many state-of-the-art methods, which use local cropped image patches as input for training and enforce smoothness of depth values in the

post-processing stage, the proposed framework aims at computing matching scores on entire images. This scheme allows pixels having the same depth value in the reference image to be computed at the same time, and hence the global smoothness of the depth map can be learned by the neural network. To reconstruct the scene, each depth map is further integrated into a point cloud using the camera transformation matrix.

The main contributions of this chapter are two-fold: 1) to present a novel network that can learn global smoothness constraint and directly perform MVS matching based on global information, and 2) to demonstrate that the method is highly robust and can be applied to different image datasets without the need for retraining, regardless of how the resolution and depth range change. Based on the evaluation on the DTU dataset [1], the proposed approach is comparable to existing algorithms in terms of completeness; see Figure 4.1.

4.1 Related work

Over the past decade, many practical approaches using traditional stereo algorithms for 3D modeling have been developed. Campbell et al. [8] utilized Normalized Cross-Correlation (NCC) to calculate patch-wise matching costs. To address false predictions in the depth maps caused by repeated texture, they enforced a spatial consistency constraint on neighboring pixels and demonstrated how to select accurate depth from multiple depth hypotheses using Markov Random Field (MRF) optimization. Furukawa and Ponce [18] proposed using photometric and visibility consistency to enhance the effectiveness of multi-view stereopsis based on epipolar geometry, and



(a) Model (b) Reference (c) The proposed (d) Ji [37]

(e) Camp [8] (f) Furu [18] (g) Tola [87] (h) Gipuma [19]

Figure 4.1: Comparison with the state of the arts on 3D reconstruction for model 10 from the DTU dataset [1]. The approach proposed here, (c), generates the most complete result.

introduced two filtering steps to remove patches lying outside and inside the surface separately. For modeling applications, they further merged the collected patches into meshes through smoothness control.

The above approaches tend to assume that pixels within a supporting patch have constant depth and therefore may miscalculate depth values for slanted surfaces. Targeting this challenge, Bleyer et al. [3] introduced an effective algorithm referred to as PatchMatch, which initializes a random 3D plane for each pixel and gradually discovers the optimal plane through iterations of spatial and view propagation. As advanced optical sensors were developed to catch images with higher resolution, attempts were also made to accelerate the reconstruction process for stereopsis. Following the idea of PatchMatch, Galliani et al. [19] presented a novel diffusion-like approach which categorizes pixels into different groups as a checkerboard pattern so that high-resolution

images can be addressed by an extensively parallel scheme implemented on a GPU. Tola et al. [87] proposed generating depth maps by using DAISY descriptors, which generate gradient histograms from different orientation layers to efficiently implement similarity score computation on whole images, and directly selected matching pairs with notably larger scores than other candidates. Moreover, depth prediction was performed on sparse areas first to restrict the disparity searching range on neighboring pixels for fast performance.

The past few years have witnessed a rapid expansion of learning-based approaches for 3D reconstruction. Here, these approaches are grouped into two categories: patch-wise and global matching.

4.1.1 Patch-wise Methods

Inspired by the successful practice of patch-wise stereo matching within traditional algorithms, early learning-based approaches are devoted to using neural networks to replace window-based matching cost computation. Works were first proposed to address binocular stereo matching and then extended to multi-view cases [35, 64, 97, 103].

Galliani and Schindler [20] opted for the matching algorithm proposed in their earlier work [19] to generate initial 3D points and vector fields, and trained a CNN model to perform normal prediction on raw image patches from multiple views. To obtain 3D models, depth and normal maps are merged together with Poisson reconstruction. The obtained surface normals are beneficial for reliable reconstruction of areas that have no valid MVS points. Huang et al. [33] presented a deep CNN model,

which exploits the structure of U-Net [75], to generate a bunch of plane-sweep volumes by performing stereo matching on 64×64 patches and thereafter compute depth maps for MVS. Hartmann et al. [28] proposed to directly learn multi-patch similarity using a N-way Siamese network architecture [7]. To implement this idea, a reference image patch, together with multiple matching patches from neighboring views, are assembled as an input sample for training. A similar work can be found in Yao et al. [96], where much wider patches and homography warping are harnessed to train an end-to-end deep learning framework. In addition, matching cost aggregation and depth map regression are integrated into the training pipeline. Ji et al. [37] proposed another end-to-end structure. They converted images to 3D voxel representations through projection and trained a CNN model, referred to as SurfaceNet, to predict each voxel's probability of lying on the surface of models. SurfaceNet takes cropped voxel cubes as input, thus the learning process is also based on local information.

Since patch-wise approaches generally perform prediction for each pixel individually, the computation cost tends to hinder their applicability to larger datasets with high-resolution images. A solution to accelerate patch-wise matching is to apply prediction only on image patches with enough vision cues, though with consequence being a loss in completeness [70] .

4.1.2 Global Methods

When it comes to objects lacking texture, patch-based approaches mostly require overall smoothness control during their post-processing steps. In contrast, this operation is automatically tackled by learning-based approaches using neural networks

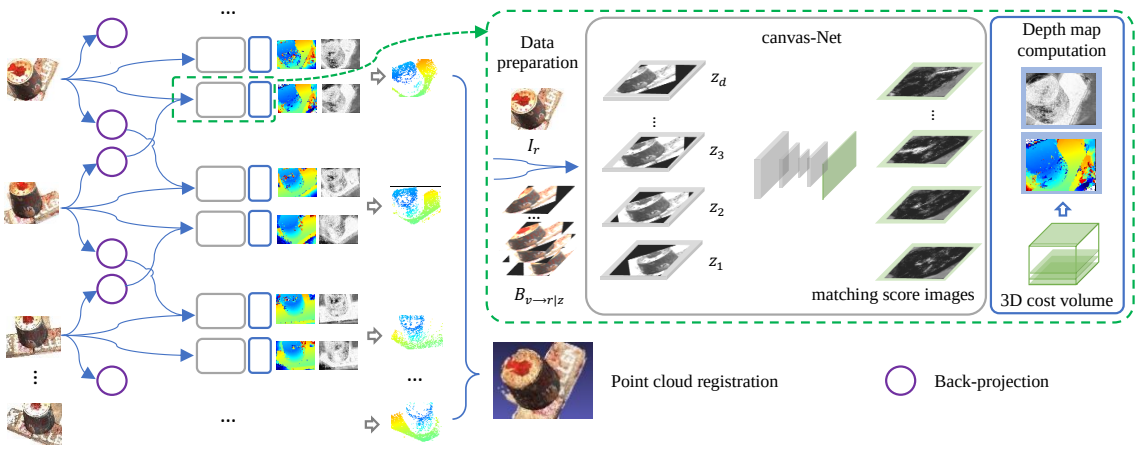


Figure 4.2: MVS framework. Given a set of images with geometry parameters [1], fronto-parallel back-projections are first obtained to generate matching score images stored in a matching cost volume, from which depth and confidence maps are then computed and filtered for point cloud registration. In the depth maps, the warmer the color, the higher the predicted depth values are.

trained on global information. Inspiring results have been achieved on binocular stereo cases [9, 41, 63], but few attempts have been made to apply global matching on MVS. Performing global matching on multiple high-resolution images directly demands tremendous parallel computational resources, but separately applying two-view global matching on MVS as presented here is likely to be manageable.

4.2 Methodology

As discussed above, the proposed approach in this chapter is robust enough to handle different datasets without the need of retraining. It only requires that the input images have known intrinsic/extrinsic camera parameters and are lens-distortion cor-

rected. All images from public stereo datasets, such as DTU [1], KITTI [61], and Middlebury [77], satisfy this requirement. A few notable differences among these datasets are image resolutions, experimental objects, and lighting conditions, which makes it challenging for a learning-based approach to process them without retraining.

The overall pipeline of the proposed MVS approach is shown in Figure 4.2. Given an MVS dataset, each image, I_r, is used as a reference image and its neighboring views are selected. I_r is then paired with each of its neighboring views, I_v, to compute a depth map and an associated confidence map (Section 4.2.1). The former map provides us the best depth hypothesis for each pixel in I_r, whereas the latter indicates how likely this depth hypothesis is correct. The depth map computed using an individual image pair can be noisy, and hence the depth/confidence maps computed using all of I_r's neighboring views need to be merged together to obtain a clean depth map under the image space of I_r. Finally the clean depth maps computed under the image spaces of different views are registered into a 3D point cloud (Section 4.2.3).

4.2.1 Pair-wise image matching

When matching points between a pair of images, the epipolar geometry defines that a 3D point seen in the reference image I_r only appears along the epipolar line in each neighboring view I_v. For a rectified binocular stereo image pair, the epipolar lines are parallel to scanlines. However, for general input images from multi-view stereo, the epipolar lines have arbitrary directions, making the search for matching points more difficult.

To simplify the problem, I_v will be back-projected to the fronto-parallel planes of I_r

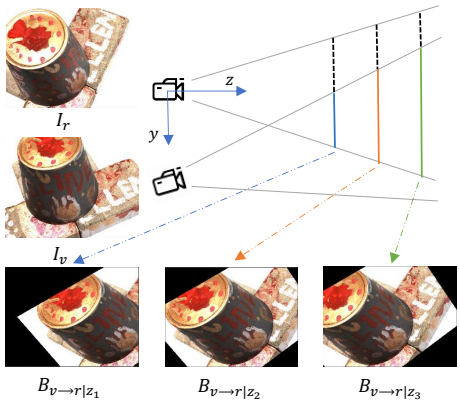


Figure 4.3: Fronto-parallel back-projection. Given a reference image I_r and one of its neighboring views I_v, fronto-parallel planes at different depth z are used to back-project I_v toward the image space of I_r. The resulting images, referred to as $B_{v\to r|z}$, are used to search matching pixels.

at the same resolution; see Figure 4.3. Here, the image obtained by back-projecting I_v to the fronto-parallel planes of I_r at depth z is denoted by $B_{v\to r|z}$. Under this strategy, to find the matching pixel for a given pixel p in I_r, one need only to search among pixels at the same coordinates as p in $B_{v\to r|z}$ under different z values. In addition, 3D points having the same depth value with respect to I_r will find matches in the same back-projected image [13] and hence matching smoothness can be effectively enforced. Note that the same scheme can be also applied to binocular stereo pairs, where the back-projection only shifts the image along the x axis, resulting matching pixels shown at the same location on the corresponding back-projection plane.

4.2.2 canvas-Net

As mentioned above, the goal here is to train a neural network that can directly process high resolution stereo images from a variety of datasets. Since images from these datasets have different resolutions, the network needs to accommodate the highest resolution images. When the reference image I_r and back-projected images $B_{v\to r|z}$ have lower resolutions, they are simply processed using the center portion of the network, without the need for scaling the images to match the network resolution. Here the network is referred to as canvas-Net and $H_C \times W_C$ denotes the resolution of the network (canvas).

However, when training the canvas-Net, always placing I_r and $B_{v\to r|z}$ at the center of the canvas will cause neurons in the area of missing data to be improperly trained. To address this problem, both I_r and $B_{v\to r|z}$ will be randomly shifted within the range of the canvas size. That is: r_h and r_w are random offsets used for shifting I_r and all back-projected images $B_{v\to r|z}$ in each training batch where $r_h = \text{rand}(e, H_C - h_I - e)$ and $r_w = \text{rand}(e, W_C - w_I - e)$. The padding size, e, is set to 5 to exclude invalid convolutional operation on the edges, and $h_I \times w_I$ is the resolution of I_r,. Owing to image regularization in the previous stage, two canvases carrying grayscale information from the reference image I_r and one of its neighboring back-projected images $B_{v\to r|z}$ can be stacked together as a 3D canvas to be fed into the canvas-Net. Note that the input can be extended to accommodate RGB channels of images or one reference image with multiple back-projected images. The main goal of the model is to select all accurate matches between corresponding pixels in I_r and $B_{v\to r|z}$.

The canvas-Net has a similar structure as U-Net [75] but with a much larger

receptive field; see Table 4.1. To accommodate high-resolution image pairs from all selected datasets in the $H_c \times W_C \times 2$ canvas, H_C is set to 1280 and W_C is set to 1664. This allows for all high-resolution images to be handled without the need of down-sampling,, a process that would otherwise affect the accuracy of generated depth maps. For the reference image under each depth hypothesis, the canvas-Net outputs a matching score map to accentuate the locations of precisely matched pixels by marking them with higher confidence values.

To train the model for matching computation, stereo images with ground truth are desired. Since the input only consists of two layers, the training samples are extracted from multi-view and/or binocular imagery. Given I_r and $B_{v\to r|z=d}$, an ideal matching score image here should filter out all domains among them with the same coordinates offering invalid depth estimation. In practice, a pixel (x, y) in the expected depth map is considered as mismatched if and only if $\|D_t(x, y) - d\| > T$, where D_t denotes the ground truth and T is a threshold value.

Unlike other attempts to improve U-Net [75] by involving abundant layers and complicated substructures, the novelty of the canvas-Net lies in the well-designed scheme of loss calculation so that an effective learning process can be performed without greatly increasing the computational cost. The loss function in U-Net [75] cannot be directly adopted here since it equally addresses each pixel in the output, lacking a scheme to highlight those pixels that require more attention. To compute training loss between an estimated matching score image $S_{(e,d)}$ and its ground truth $S_{(t,d)}$, the peripheral regions generated by canvas fitting first need to removed. The masks of "0" M_0 and "1" M_1 divided by T within $S_{(t,d)}$ are highly unbalanced, i.e., the output at most pixel locations should be "0" whereas only a small number of

Table 4.1: Parameters of the canvas-Net. "Conv", "Dconv", "Mp" and "⌒" denote convolutional, deconvolutional, max pooling and concatenation operations, respectively.

Input	Operation	Kernel and channel	Stride	Output, size and channel
3D canvas	Conv	$1 \times 1 \times 2$, 16	$1 \times 1 \times 1$	O_1, $1280 \times 1664 \times 2, 16$
O_1	Conv	$5 \times 5 \times 1$, 16	$1 \times 1 \times 1$	O_2, $1280 \times 1664 \times 2, 16$
O_2	Conv	$5 \times 5 \times 2$, 16	$1 \times 1 \times 1$	O_3, $1280 \times 1664 \times 2, 16$
O_3	Conv	$5 \times 5 \times 2$, 16	$1 \times 1 \times 1$	O_4, $1280 \times 1664 \times 2, 16$
O_4	Mp	$2 \times 2 \times 2$, 16	$2 \times 2 \times 1$	O_5, $640 \times 832 \times 2, 16$
O_5	Conv	$5 \times 5 \times 2$, 32	$1 \times 1 \times 1$	O_6, $640 \times 832 \times 2, 32$
O_6	Mp	$2 \times 2 \times 2$, 32	$2 \times 2 \times 1$	O_7, $320 \times 416 \times 2, 32$
O_7	Conv	$5 \times 5 \times 2$, 64	$1 \times 1 \times 1$	O_8, $320 \times 416 \times 2, 64$
O_8	Mp	$2 \times 2 \times 2$, 64	$2 \times 2 \times 1$	O_9, $160 \times 208 \times 2, 64$
O_9	Conv	$5 \times 5 \times 2$, 64	$1 \times 1 \times 1$	O_{10}, $160 \times 208 \times 2, 64$
O_{10}	Mp	$2 \times 2 \times 2$, 64	$2 \times 2 \times 1$	O_{11}, $80 \times 104 \times 2, 64$
O_{11}	Conv	$5 \times 5 \times 2$, 64	$1 \times 1 \times 1$	O_{12}, $80 \times 104 \times 2, 64$
O_{12}	Mp	$2 \times 2 \times 2$, 64	$2 \times 2 \times 1$	O_{13}, $40 \times 52 \times 2, 64$
O_{13}	Dconv	$5 \times 5 \times 1$, 64	$2 \times 2 \times 1$	O_{14}, $80 \times 104 \times 2, 64$
$O_{14} \frown O_{12}$	Dconv	$5 \times 5 \times 1$, 64	$2 \times 2 \times 1$	O_{15}, $160 \times 208 \times 2, 64$
$O_{15} \frown O_{10}$	Dconv	$5 \times 5 \times 1$, 64	$2 \times 2 \times 1$	O_{16}, $320 \times 416 \times 2, 64$
$O_{16} \frown O_8$	Dconv	$5 \times 5 \times 1$, 32	$2 \times 2 \times 1$	O_{17}, $640 \times 832 \times 2, 32$
$O_{17} \frown O_6$	Dconv	$5 \times 5 \times 1$, 32	$2 \times 2 \times 1$	O_{18}, $1280 \times 1664 \times 2, 32$
$O_{18} \frown O_4$	Conv	$5 \times 5 \times 2$, 6	$1 \times 1 \times 1$	O_{19}, $1280 \times 1664 \times 2, 6$
O_{19}	Mp	$1 \times 1 \times 2$, 6	$1 \times 1 \times 2$	O_{20}, $1280 \times 1664 \times 1, 6$
O_{20}	Conv	$5 \times 5 \times 1$, 1	$1 \times 1 \times 1$	O_{21}, $1280 \times 1664 \times 1, 1$

pixels should output "1". Hence, a coefficient mask M is introduced to counteract the amount of variation. In addition, another threshold R in M_0 is used to isolate the minor mismatches forming mask M_a from the rest M_b by fulfilling $\|D_t(x,y)-d\| < R$, where R is slightly larger than T. M is then computed by:

$$M = (\frac{|M_0|M_1}{|M_1|} + w_1\frac{|M_1+M_b|M_a}{|M_a|} + w_2 M_b) \ , \tag{4.1}$$

where $|U|$ is the size of set U. The two weight parameters w_1 and w_2 can be set additionally to achieve an optimal trade-off among all regional influences. Moreover, the training loss is computed by:

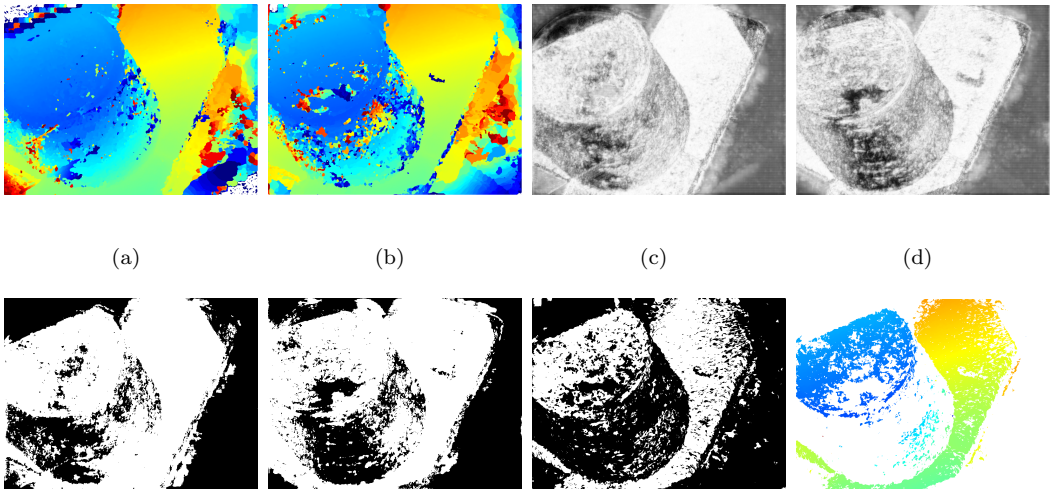
$$loss = \text{mean}(\|S_{(e,d)} - S_{(t,d)}\| \cdot M) \ . \tag{4.2}$$

The canvas-Net trained above computes a set of matching score images stacked as a 3D cost volume $C_s(x,y,d)$ for each reference image. To generate the depth map $D_e(x,y)$ and confidence map $F_e(x,y)$, the WTA algorithm is employed to select the best depth hypothesis for each pixel by $D_e(x,y) = \arg\max_d C_s(x,y,d)$ and $F_e(x,y) = \max_d C_s(x,y,d)$.

4.2.3 Point Cloud Registration

For a reliable point cloud registration, filtering is applied to both $F_e(x,y)$ and $D_e(x,y)$ mentioned above to exclude the depth outliers and isolated points in the reconstruction stage.

Dramatic errors are likely to be generated when 3D points projected in I_r do not occur in other views. An effective method to partially remove them is to demand $F_e(x,y) > G$, where G is the minimum matching score required for a valid depth



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 4.4: Depth filtering. Two separated depth maps (a,b) and their confidence maps (c,d) are generated when the same reference is paired with different neighboring views. Their confidence masks (e, f) and depth variation mask (g) are computed by setting $G = 0.7$ and $V = 1.0$, and merged into one mask to filter out invalid values for precise estimation (h).

value. By assembling the same reference image I_r with different neighboring views, multiple depth maps together as P_e can be generated and combined for an optimal one $D_o(x, y)$, where each valid depth shares no more than a variation threshold V with all its candidates from P_e. This integration scheme further filters out more outliers; see Figure 4.4. A 3D point cloud can then be reconstructed by merging all pruned depth maps together with the geometry parameters of camera views.

4.3 Experimental Results

As mentioned above, the goal in this chapter is to develop a robust approach that can handle different scenes without the need to retrain the network. To validate whether this goal is achieved, the DTU [1] dataset, which consists of a large variety of scenes compared to other accessible MVS datasets, is selected for testing. In addition, the binocular stereo datasets, KITTI [61] and Middlebury [77], were deliberately chosen to train the network. This is a more challenging experimental setup than existing approaches that retrain the network before testing it on a given dataset.

Unlike many existing approaches plainly practicing training and testing on the same dataset, Here, canvas-Net is forced to fulfill feature learning from the binocular stereo datasets KITTI [61] and Middlebury [77] and the attained filters are applied to other datasets. An added benefit of this setup is the avoidance of overfitting. The multi-view images from DTU are limited to objects in an experimental environment under stable lighting conditions. Combining KITTI and Middlebury, by contrast, populates the training dataset with indoor and outdoor scenes and facilitates feature extraction capability.

The algorithm is implemented with Tensorflow on a GTX 1080 Ti GPU. Around $4,000$ training samples with random offsets are generated for each training epoch, and a stable state can be achieved after 20 epochs by embedding an exponentially decreasing learning rate from 0.005 to 0.00001. Additionally, $T = 1.0$, $R = 3.0$, $w_1 = 0.5$ and $w_2 = 0.5$ are set to calculate the loss, and the entire process takes about 4 days to complete. When applying the canvas-Net on DTU, it takes around $0.3s$ to generate each matching score map.

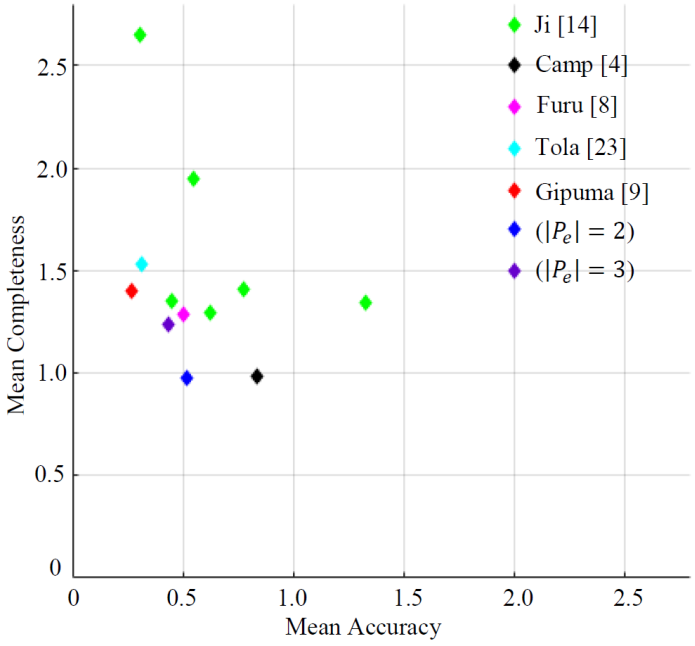
4.3.1 Testing on DTU

The DTU dataset [1] contains 124 experimental scenes in total, and 49 fixed positions are set up to capture views from different perspectives. For a fair and direct comparison with existing approaches, two metrics, *accuracy* and *completeness* in Aanaes et al. [1] are chosen to evaluate the proposed approach. The former is specified by measuring the Euclidean distance from a point cloud to its ground truth and vice versa for the latter. The better performance of the algorithm, the lower the values for both metrics.

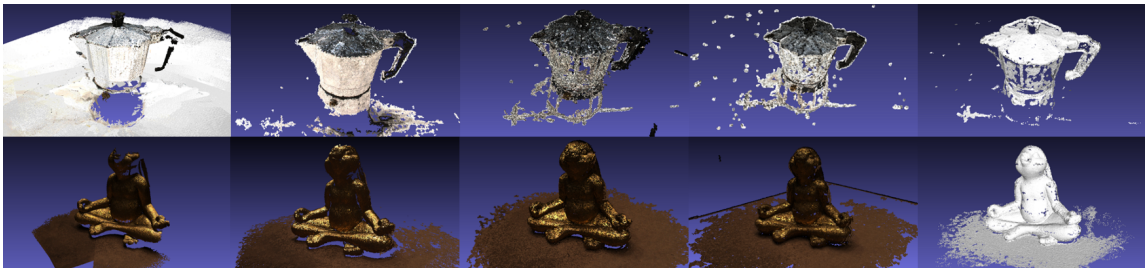
All results here are built on the calibrated 1200×1600 resolution images with both internal and external camera parameters. To comply with the trained model above, each reference image, along with one of its neighboring views, is assembled as an input pattern for testing. With regards to DTU, the sampling unit U_z was set to $0.5mm$ along the Z axis when generating the back-projected images. To remove invalid depth estimates precipitated by WTA, G was set to 0.7 and V was set to 1.0, and the initial point clouds were reconstructed; see Table 4.2 for more parameter settings.

4.3.2 Quantitative Comparison

When applying patch-wise CNN models as in other works [11, 33, 70, 96] on the DTU [1] dataset, the computational cost is barely manageable to generate quantitative results since a large number of high-resolution images are captured for each scene. In addition, existing binocular global-matching algorithms [9, 41, 63] lack the flexibility to address resolution and depth range variations. Therefore, the focus here



(a)



(b)

Figure 4.5: Qualitative comparison using 22 models [1] on completeness vs. accuracy (a) shows that the proposed framework is comparable to the state-of-the-art approaches on mean completeness. Visual comparisons (b) show the ground truth (1st column), the results from the proposed approach(2nd column), and the point clouds generated by Campbell et al. [8], Furukawa and Ponce[18], and Tola et al. [87] (colums 3, 4, and 5 respectively).

is on comparison with 3 traditional algorithms [8, 18, 87] and 2 learning-based methods [37, 19]. Note that the latter two approaches are directly trained on DTU, and therefore scenes selected for evaluation require isolation from their training data. For a fair comparison, the same 22 scenes suggested in Ji et al. [37] are used here.

Mean accuracy and completeness are calculated for all selected scenes. Although direct numerical comparison based on either metric can be made, an accuracy vs. completeness plot is used here to compare different algorithms on both aspects; see Figure 4.5 (a). The proposed framework makes full use of global feature correlation and therefore is more capable of performing stereo matching when lacking vision cues. Figure 4.5 (b) visually compares the point clouds produced by different approaches.

4.3.3 Real-world Application

Here, a cross-library framework is presented to eliminate the needs for retraining when handling different datasets. The canvas-Net trained above is directly applied to the task of reconstructing large scale outdoor scenes captured by a DJI drone. Even though the input images (750×1000 in resolution) barely resemble the training data, the proposed framework still can reconstruct dense 3D point clouds; see Figure 4.6.

4.4 Summary

A competent learning-based MVS approach is presented in this chapter. Unlike existing learning-based methods that work at patch level, the network in this chapter is trained over entire high-resolution images. As a result, the network can learn global features and implicitly enforce a global smoothness constraint. A novel data

preparation approach and loss function are proposed to reduce memory requirements and handle imbalanced classes. The experiments demonstrate the robustness of the proposed approach. When training on binocular datasets (KITTI and Middlebury) and tested on multi-view dataset (DTU), the proposed approach achieved overall best performance in terms of completeness vs. accuracy among the state-of-the-art approaches. While the point clouds generated using the proposed approach are promising, scattered outliers do exist, which need to be eliminated for clearer scene representations. In addition, when combining point clouds calculated under different views, a point set surface-thinning operation is needed. How to perform these tasks using a learning-based approach is addressed in the next chapter.

Table 4.2: Evaluation on model 13 in Aanaes et al. [1] under different settings.

Settings		Accuracy	Completeness
$U_z(mm)$	$\lvert P_e \rvert$		
1.0	2	0.526	**3.848**
0.5	2	0.460	3.879
0.5	3	0.369	4.359
Ji [37]		0.417	3.974
Camp [8]		0.477	4.517
Furu [18]		0.406	4.943
Tola [87]		**0.313**	5.041
Gipuma [19]		0.340	5.630

(a)

Figure 4.6: Reconstructing large scale outdoor scenes using the proposed MVS framework.

Chapter 5

Point Cloud Consolidation through Learning-based Projection

High-quality 3D representations for real objects are often needed in various Computer Vision and Graphics applications. In many of these cases, passive reconstruction through multi-view stereo matching is preferred, as introduced in the previous chapters. However, many factors, such as sensor noise, lack of texture, occlusion, and extreme lighting conditions, can contaminate the reconstructed depth information. To generate solid point clouds, consolidation is widely practiced as a subsequent process when primitive results are acquired. Various consolidation algorithms have been proposed to address the problem from the perspectives of denoising [76], outliers removal [2], upsampling [100], and completion [27]. Nevertheless, there is no perfect solution for this ill-posed problem and most existing approaches are designed for point clouds captured by 3D scanners, which are much cleaner than those generated by multi-view stereo matching.

In addition, many conventional consolidation approaches rely on smoothness assumptions and use tools such as the Poisson equation [40], smoothing filters [83], or normal propagation [31]. These tools can indeed suppress noise but also soften sharp edges, which occur ubiquitously in real-world scenes. Edge-preserving techniques [32] have been proposed to address this issue, but they rely on manually tuned parameters to distinguish fine geometry features from noise and the parameters likely need to be readjusted for different datasets.

Inspired by the success of deep neural networks, which automatically optimize the parameters when various cases are involved in training, a learning-based approach is proposed for point consolidation. The network, referred to as projection-Net, is trained to predict a 3D projection vector v for each point p in the point cloud based on p's neighboring points, so that $p+v$ is closer to the ground truth surface. Existing learning-based approaches, however, perform overall adjustment on full point clouds or local patches and thus lack the capability to selectively alter individual points.

5.1 Related Work

Consolidation of point clouds plays a vital role in 3D reconstruction. Early consolidating algorithms generally propose smoothness constraints to perform various adjustments, while the past few years have witnessed a number of inspiring works based on neural networks.

5.1.1 Smoothness Constraints

Noise is inevitably involved during scene reconstruction, and smoothness constraints have been principally applied to alleviate them with an underlying assumption that the target scenes have smooth surfaces. An early smoothing practice based on normal estimation was proposed by Huang et al. [31], where points are first thinned and equally scattered by a locally optimal projector (LOP) and then used to calculate normals via a predictor-corrector iteration. The consolidation of points is gradually achieved when the reliable normals are propagated. They further demonstrated [32] that iteratively applying bilateral smoothing on normals facilitates edge unveiling. Preiner et al. [71] proposed using a Gaussian mixture to describe the density of input points and applied a continuous LOP formulation to fast normal reconstruction. When it comes to recovering sharp features, Sun et al. [83] introduced a smoothing approach based on the L_0 norm. The implemented L_0-Minimization algorithm is capable of eliminating noise and maximizing smooth regions. To tackle moving objects represented by a dynamic point cloud sequence (DPCS), Arvanitis et al. [2] proposed enforcing spatial and temporal coherence between consecutive frames to exclude outliers and used a weighted Laplacian matrix for interpolation.

5.1.2 Neural Networks

Neural networks have been emerging as alternative solutions for consolidation as promising models [73, 74] are trained to resolve point set segmentation and classification. Roveri et al. [76] introduced using a generative neural network for consolidation. Trained on dense point clouds with ground truth, the proposed model is capable of

separately turning local noisy patches into organized ones, leading to favorable surface representations.

Data-driven attempts have also been made on surface completion. For low quality point clouds, Yu et al. [100] proposed an upsampling network, referred to as PU-Net, to output denser results. PU-Net extracts local patches from a point cloud and applies hierarchical feature learning [74] and multi-level aggregation to obtain local and global characteristics for feature expansion. They later extended this work for an edge-aware consolidation model referred to as EC-Net [99]. Another compelling work on upsampling was recently proposed by Wang et al. [98]. They presented a patch-based network, which consists of multiple sub-networks extracting details on different levels, to upsample the patch iteratively. Given an incomplete surface represented by 256^3 volumetric distance fields, Han et al. [27] proposed an end-to-end deep learning framework to recover missing portions. Two sub-networks are involved in this framework to infer overall structure and optimize local geometry, respectively. Attempts have also been made on training neural networks to study geometric properties such as surface normals and curvature from local point patches to facilitate consolidation [4, 25]. To date, little work has been done on relocating original points to improve accuracy.

5.2 Methodology

As mentioned above, various algorithms of consolidation can be developed and integrated to fine-tune point clouds. Outlier filtering (Section 5.2.1) is typically the initial process of consolidation. Since points alongside the surface have great potentials to be

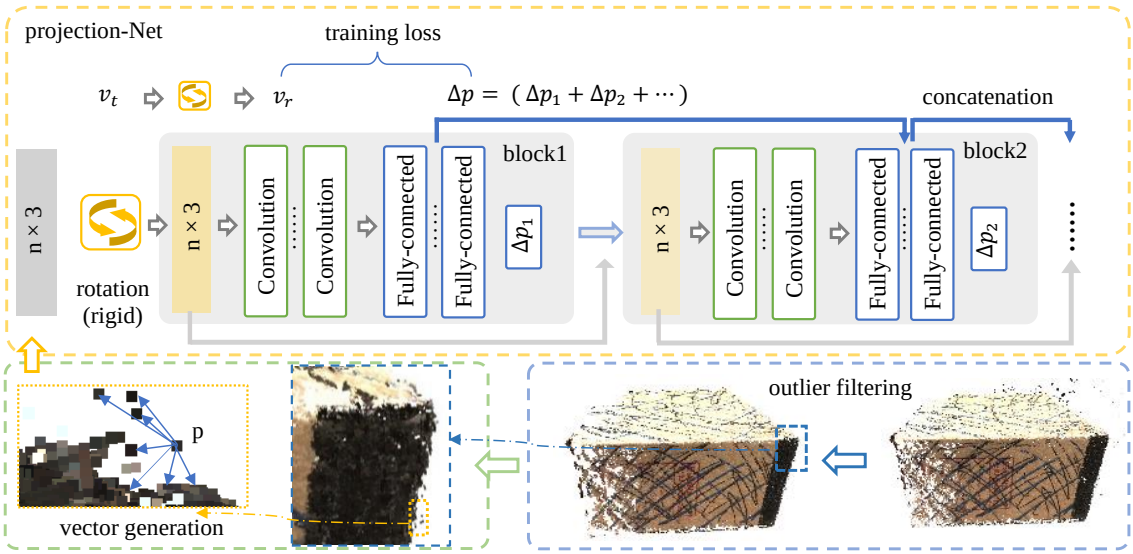


Figure 5.1: Algorithm pipeline and network architecture. Given a raw point cloud, scattered outliers are first identified and filtered. For each remaining point p, regardless of whether it is already on the surface or not, its neighboring points are located. These point locations, as well as the desired projection vector for p, are randomly rotated before being fed into the network for training. The network consists of multiple projection blocks that are chained together. Information exchange between blocks is implemented by building concatenation on fully-connected layers.

projected correctly by analyzing their correlation with those precisely placed points, local vectors (Section 5.2.2) containing spatial information between each point and its neighbors can be generated to train the network (5.2.3) to predict a projection vector; see Figure 5.1 for the consolidation pipeline.

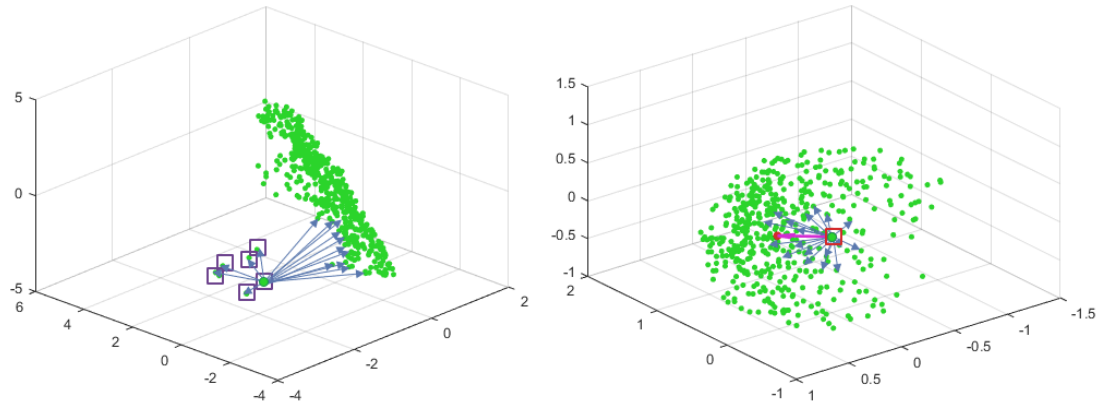
5.2.1 Outlier Filtering

Scattered outliers are normally involved in the generation of point clouds due to many factors such as lens contamination, sensor damage, and scene occlusion. These outliers typically appear at randomly distributed locations; see Figure 5.2 (a). In the proposed approach, the detection of outliers is integrated in the process of searching the supporting neighborhood for point projection. That is, for each point p, its neighborhood Ω_p can be found using the k-Nearest-Neighbors (kNN) algorithm. If p is an isolated outlier, its neighboring points generally spread over a large area, leading to a high mean distance $\frac{1}{\|\Omega_p\|}\sum_{q\in\Omega_p}\|p-q\|$. All points whose mean distance value is larger than a threshold D can be removed as a result.

5.2.2 Vector Generation

After outlier filtering, the remaining points all have sufficiently close neighbors and need to be projected onto the latent object surface to achieve the goal of point consolidation. This is done by computing a projection vector Δp for each point p so that $p+\Delta p$ is on or closer to the latent surface. As a learning-based approach, Δp is computed by training a network, instead of using a handcrafted algorithm based on a smoothness constraint. Hence at the training stage, Δp is computed to approximate the correct projection vector v_t, which gives the minimum distance between point p and the known ground truth surface.

A notable difference between the proposed approach and other learning-based consolidation algorithms [76, 98, 100, 99] is that the former is trained on local vectors instead of point patches. This scheme maintains the structure of local points



(a) (b)

Figure 5.2: kNN neighborhood search. A given point is labeled as an outlier and filtered (e.g., points highlighted using purple squares in (a)), if its kNN neighbors have high mean distance. Otherwise, vectors connecting the point and its neighbors are used to compute the desired projection vector (magenta vector in (b)).

but detaches their locations so that the training data is independent from various coordinate systems. In addition, the vectors are assembled here based length and normalized to $[-1.0, 1.0]$ to minimize the variation of length range and density when applied to different data.

It was observed that each vector starting from p to one of its closest neighbors plots a possible moving direction but the correct path tends to be determined by collective effects; see Figure 5.2 (b). Note that the points that have already been placed correctly may be trivially altered or remain in the same positions.

5.2.3 Projection-Net

The projection-Net is built to extract spatial information and predict a projection vector Δp for each 3D point p. As shown in Figure 5.1, the input of the network is a

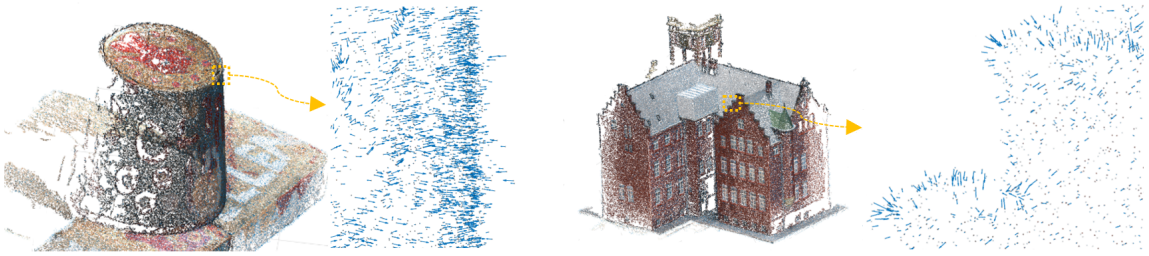
$n \times 3$ vector matrix, which stores the offset vectors $q - p$ for each point $q \in \Omega_p$, and the expected output, a 1×3 projection vector, gives the shortest path to project p onto the ground truth surface. By default, n is set to 500.

5.2.3.1 Rigid Rotation

Ideally, the ground truth projection vectors in the training data should uniformly sample all directions so that proper projection vectors can be inferred during testing, regardless of local path orientations. However, in practice, the ground truth projection vectors can be highly biased toward particular orientations. To reduce the bias, the input $n \times 3$ matrix and the ground truth v_t were deliberately and arbitrarily rotated to generate data with different orientations to train the network. A constraint for this process is that the distance between any two vectors need to remain constant as a rigid body, and the transformation using T-net introduced in Qi et al. [73] cannot be applied here. Hence, different 3×3 rigid rotation matrices are generated to randomly rotate vector samples within the same training batch.

5.2.3.2 Model Design

The projection-Net consists of multiple blocks, with each block containing mainly convolution and fully-connected layers. The use of multiple blocks allows the network to gradually project the input points onto the optimal positions. Within the i^{th} block, the input $n \times 3$ matrix first goes through convolution operations with 1×3 kernels to extract features along the x, y, and z axes, respectively. The feature maps computed are fed into fully-connected layers to compute an optimal projection vector Δp_i. The updated point location $p + \Delta p_i$ is then used to compute the new offset vectors to



(a) (b)

Figure 5.3: Point consolidation. Given the input raw point cloud generated by Furukawa and Ponce [18] for DTU model #1 (a) and #24 (b), the projection vectors (blue arrows) point toward the latent surface and hence help to clean up the data.

points in Ω_p. This forms a new $n \times 3$ matrix, which is used as input for the $(i+1)^{th}$ block. At the end of all blocks, the output projection vector is the sum of local projection vectors, i.e., $\Delta p = \sum_i \Delta p_i$.

Different blocks currently perform vector computation independently, but building information exchange across different blocks can provide successive projection stages with momentum effects. Concatenation is applied to fully-connected layers, since these layers conduct high-level features in each block. In terms of network optimization, the training loss is measured by $\|v_r - \Delta p\|^2$, where v_r is the rotated vector of v_t.

5.3 Experiments

Here, the above approach is applied to point clouds generated by existing MVS algorithms. Details of the implementation and results are presented in this section, and the validation is based on the DTU [1] dataset.

Table 5.1: Experimental settings for the Projection-Net. Here "Rt", "Conv", "Fc" and "$\frown$" denote rotation, convolutional, fully-connected, and concatenation operations respectively. The dropout rate of all fully-connected layers was set to 0.2.

Block	Input	Operation	Neurons	Output and Size
	I	Rt	3×3	I_1, 500×3
1:	I_1	Conv	1×3, 32	O_1, $500 \times 3 \times 32$
	O_1	Conv	1×3, 64	O_2, $96000(500 \times 3 \times 64)$
	O_2	Fc	256	O_3
	O_3	Fc	128	O_4
	O_4	Fc	3	Δp_1
2:	$I_1 - \Delta p_1$	Conv	1×3, 32	O_5, $500 \times 3 \times 64, 32$
	O_5	Conv	1×3, 64	O_6, $96000(500 \times 3 \times 64)$
	O_6	Fc	256	O_7
	$O_7 \frown O_3$	Fc	128	O_8
	O_8	Fc	3	Δp_2

5.3.1 Data and Parameters

To implement the network, point clouds with ground truth are needed for training. As a well-built 3D modeling dataset, DTU consists of 124 scenes covering various objects, of which the ground truth (reference data) was obtained using a structured light scanner and has been used widely as a MVS benchmark. The two metrics, *accuracy* and *completeness* [1] mentioned in Chapter 4, are used for evaluation. The former is evaluated based on a distance set $\Phi_{m \to t}$ that is computed from each point

in a MVS reconstruction S_m to its closest point in the ground truth S_t. That is:

$$\Phi_{m \to t} = \{\min_{y \in S_t}(\|x - y\|) \mid x \in S_m\}. \tag{5.1}$$

Similarly a second distance set, $\Phi_{t \to m}$, is computed from each point in the ground truth S_t to a reconstruction result S_m, which is used to evaluate the completeness.

Also, the DTU [1] dataset provides point clouds reconstructed by existing algorithms [8, 18, 87] for all the scenes. A value of $D = 1.5mm$ was chosen to apply outlier filtering to these results. Excluding the test group suggested in Ji et al. [37], 1.4 million points were randomly sampled from the remaining datasets for training. Each point is used to generate the corresponding 500×3 input matrix and the desired output projection vector using the ground truth.

An experimental structure (ProjNet) of the projection-Net is given in Table 5.1. The network is implemented using the open-source machine learning library Tensorflow. A stable state can be achieved after 200 training epochs when an exponentially decreasing learning rate from 0.005 to 0.00001 and a fixed batch size of 512 are employed. The entire process takes about 2 days on a Nvidia GTX 1080 Ti GPU. The projection vectors computed by the proposed network are visually presented in Figure 5.3. It shows that the proposed approach can effectively project points toward the latent surface for various scenes to obtain a clearer and thinner representation.

5.3.2 Ablation Study

To evaluate the importance of the proposed multi-block projection and concatenation strategy, an ablation study was conducted using two altered network structures. One removes the second projection block in the network structure presented by Table 5.1

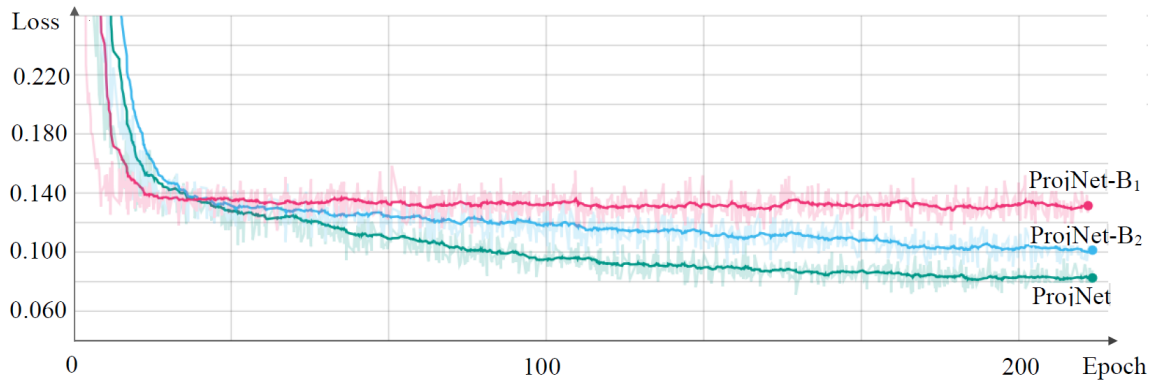


Figure 5.4: Loss comparison between the proposed network and two altered versions as a function of training epoch.

(referred to as ProjNet-B_1), whereas the other only breaks the concatenation shown in Figure 5.1 by replacing $O_7 \frown O_3$ with O_7 (referred to as ProjNet-B_2).

Figure 5.4 compares the training performances of the proposed architecture and the two altered ones. The proposed model that uses two blocks with concatenation achieves the best performance but takes more training time. On the other hand, using a single network block and repetitively processing the point clouds twice (referred to as ProjNet-$B_1 \times 2$) yields promising results, but is slower and still not as robust as ProjNet and ProjNet-B_2 based on all metrics; see Table 5.2. Even better results can be achieved when including more blocks, but at higher computational cost.

5.3.3 Validation on MVS

The proposed approach is developed as a general consolidation step and can be applied to point clouds generated by different MVS approaches. The validation here is performed on point clouds generated for 17 test scenes by 3 existing algorithms [8, 18, 87].

Table 5.2: Results of model 1 and 24 from Furukawa and Ponce [18]. Mean and median (Med) values of $\Phi_{m\to t}$ and $\Phi_{t\to m}$ are computed to compare ProjNet, ProjNet-B_2 and ProjNet-$B_1 \times 2$, and lower values here are better. ProjNet outperforms the others based on overall performance.

Model	Metric		Raw	ProjNet	ProjNet-B_2	ProjNet-$B_1 \times 2$
1:	$\Phi_{m\to t}$	Mean	0.255	0.210	0.205	**0.201**
		Med	0.145	0.129	**0.124**	0.125
	$\Phi_{t\to m}$	Mean	3.048	**3.006**	3.037	3.066
		Med	0.339	**0.282**	0.315	0.346
24:	$\Phi_{m\to t}$	Mean	0.318	0.281	0.279	**0.271**
		Med	0.221	0.203	**0.202**	0.203
	$\Phi_{t\to m}$	Mean	0.309	**0.288**	0.304	0.326
		Med	0.181	**0.171**	0.181	0.186

The mean and median values of both $\Phi_{m\to t}$ and $\Phi_{t\to m}$ are computable using the DTU evaluation code [1] for these point clouds before and after applying the consolidation approach.

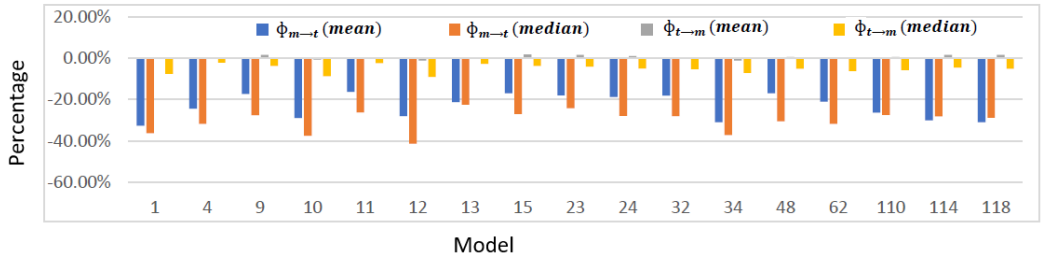
As Figure 5.5 illustrates, the proposed approach achieves considerable improvement on the results of all three algorithms with respect to $\Phi_{m\to t}$. The reasons why the proposed approach somewhat increases $\Phi_{t\to m}$ of a few scenes is because it tends to either filter out isolated points or project them to nearby point clusters. When these isolated points are close to latent surfaces, moving them reduces the level of completeness.

5.3.4 Consolidation Comparison

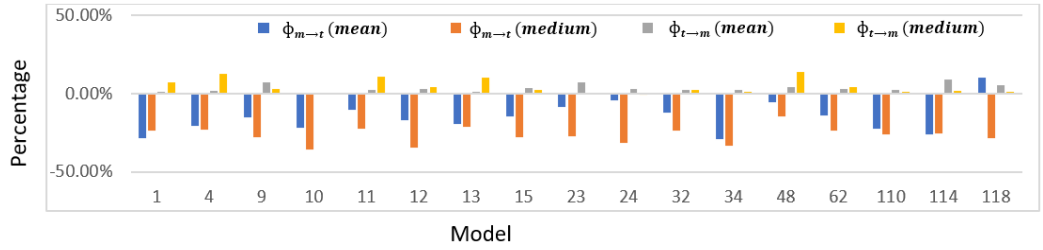
As of now, few attempts have been made on consolidation through projecting individual points. Compared to the state-of-the-art works [32, 76, 98, 99, 100], the approach proposed have no limitation on consolidating point clouds with various sizes. When testing on large-scale point clouds as in DTU, the patch-based networks proposed by Yu et al. [99] and Wang et al. [98] are the only existing approaches that can perform consolidation under reasonable memory cost (12 GB) and computation time. Here, comparison with these two works is presented; see Figure 5.6. The proposed consolidation method can generate convincing results.

5.4 Summary

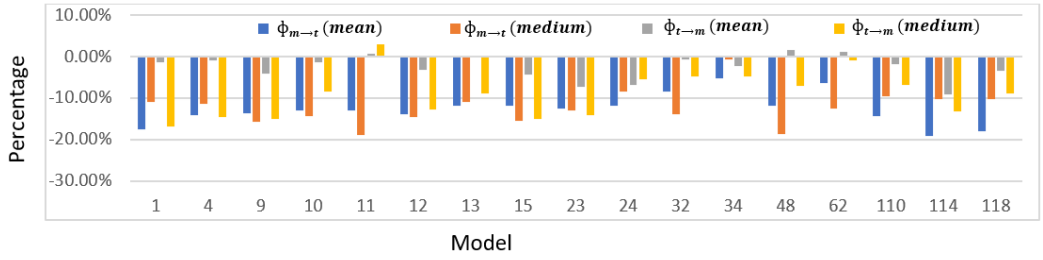
This chapter presented a learning-based approach for 3D point cloud consolidation. Trained on vectors extracted from a given point p and its neighboring points, the proposed network can effectively predict a projection vector Δp to move p closer to the latent surface. As a result, thinner and more accurate point clouds can be obtained without involving any heuristic algorithms. The experiments demonstrate that the proposed approach can effectively consolidate raw point clouds generated by the proposed MVS approach and 3 different MVS algorithms for DTU datasets.



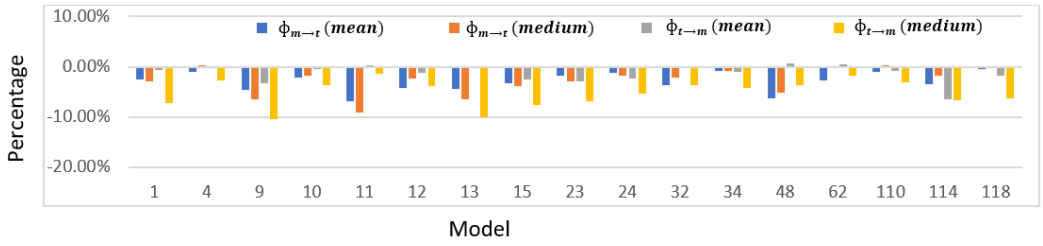
(a)



(b)



(c)



(d)

Figure 5.5: Consolidation performance. Results (a-d) are generated by consolidating point clouds generated in Chapter 4 and by Campbell et al. [8], Furukawa and Ponce [18] and Tola et al. [87], respectively. Percentage change of $\Phi_{m\to t}$ and $\Phi_{t\to m}$ are plotted here for better viewing. The proposed approach can improve various point clouds in regard to $\Phi_{m\to t}$.

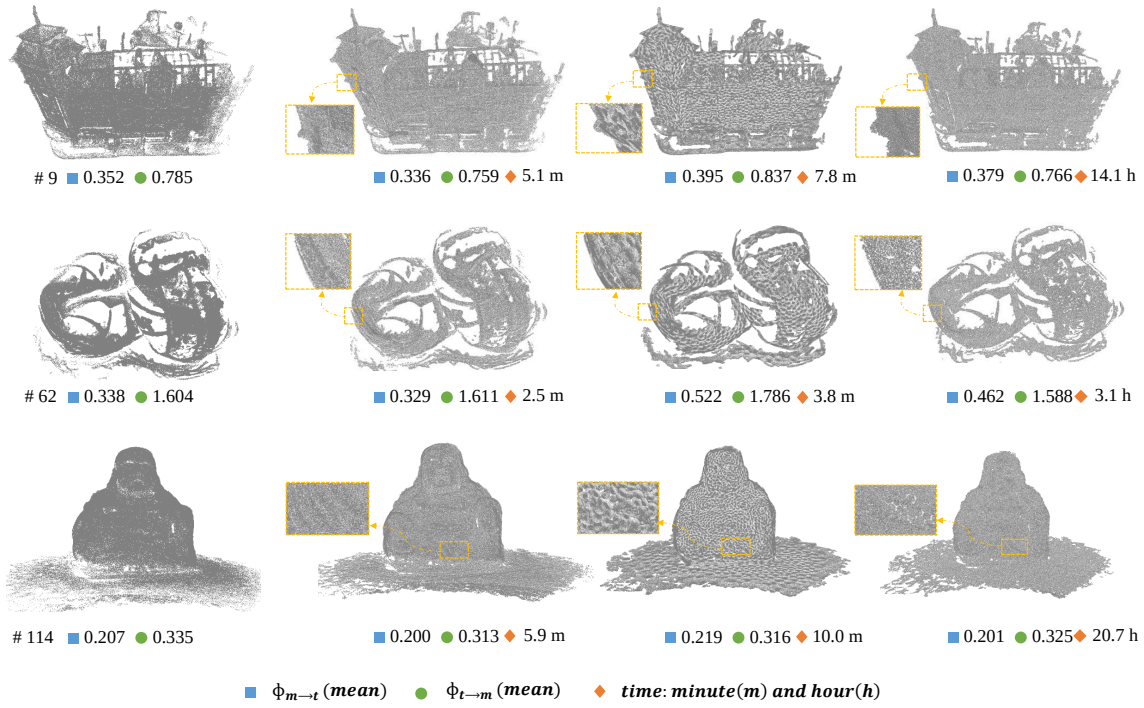


Figure 5.6: Consolidation on point clouds in Tola et al. [87]. Column 1 shows the raw results. Columns 2 to 4 are consolidated by the proposed approach, [99] and [98] respectively. The results generated by the proposed approach is promising based on all metrics.

Chapter 6

Conclusions and Future Work

In this thesis, a review of the existing works on binocular and multi-view stereo matching, matching-based 3D reconstruction, and point cloud consolidation was presented. Through the design of novel network architectures, four new deep learning-based algorithms were developed for detecting mismatches in binocular stereo matching results, generating more accurate semi-dense matches under challenging conditions, performing multi-view stereo matching using a network trained on binocular image pairs, and consolidating point clouds obtained from different viewpoints.

Currently, the proposed semi-dense stereo matching approach [57] still ranks at the top of the sparse results on the Middlebury site under the metric of "rms". This investigation suggests that, once sufficient information is fed to the network, CNN-based models can effectively predict the correct matches and detect mismatches. When it comes to stereo matching for 3D reconstruction, the presented MVS framework [56] trained on global features is capable of generating more complete DTU results compared to those of the state-of-the-art methods. Although the proposed consolidation

approach is still under review for publication, the experiments demonstrate that it can considerably enhance various MVS algorithms.

Many future directions are worth investigating to further improve the presented approaches. Firstly, the best depth hypothesis for each pixel in a given reference image is currently selected through WTA, and the depth maps in the MVS framework are merely validated among different image pairs. Replacing these heuristic operations with learning-based methods could further improve the robustness and performance of the overall matching algorithms. In addition, how to effectively exploit global information of the point clouds obtained from different reference images for overall consolidation is worth investigating. It would be interesting to validate the presented algorithms on large-scale real-world scenes captured by unmanned aerial vehicles. Finally, how to reduce the training and labeling costs so that the algorithms can be applied to real-time applications is another exciting direction.

Bibliography

[1] H. Aanæs, R. R. Jensen, G. Vogiatzis, E. Tola, and A. B. Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, pages 1–16, 2016.

[2] G. Arvanitis, A. Spathis-Papadiotis, A. S. Lalos, K. Moustakas, and N. Fakotakis. Outliers removal and consolidation of dynamic point cloud. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3888–3892. IEEE, 2018.

[3] M. Bleyer, C. Rhemann, and C. Rother. Patchmatch stereo-stereo matching with slanted support windows. In *Proceedings of the British Machine Vision Conference (BMVC)*, volume 11, pages 1–11, 2011.

[4] A. Boulch and R. Marlet. Deep learning for robust normal estimation in unstructured point clouds. In *Computer Graphics Forum*, volume 35, pages 281–290. Wiley Online Library, 2016.

[5] J.-C. Bricola, M. Bilodeau, and S. Beucher. Morphological processing of stereoscopic image superimpositions for disparity map estimation. working paper or preprint, Mar. 2016.

[6] J.-C. Bricola, M. Bilodeau, and S. Beucher. Morphological processing of stereoscopic image superimpositions for disparity map estimation. Mar. 2016. working paper or preprint.

[7] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a "siamese" time delay neural network. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*, NIPS'93, pages 737–744, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[8] N. Campbell, G. Vogiatzis, C. Hernandez, and R. Cipolla. Using multiple hypotheses to improve depth-maps for multi-view stereo. volume 5302, pages 766–779, 10 2008.

[9] J.-R. Chang and Y.-S. Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018.

[10] F. Cheng, X. He, and H. Zhang. Learning to refine depth for robust stereo estimation. *Pattern Recognition*, 74:122–133, 2018.

[11] S. Choi, S. Kim, K. Sohn, et al. Learning descriptor, confidence, and depth estimation in multi-view stereo. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 389–3896. IEEE, 2018.

[12] C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker. Universal correspondence network. In *Advances in Neural Information Processing Systems*, pages 2414–2422, 2016.

[13] R. T. Collins. A space-sweep approach to true multi-image matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 358–363. IEEE, 1996.

[14] T. Durand, T. Mordan, N. Thome, and M. Cord. Wildcat: Weakly supervised learning of deep convnets for image classification, pointwise localization and segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[15] G. Egnal, M. Mintz, and R. P. Wildes. A stereo confidence metric using single view imagery with comparison to five alternative approaches. *Image and vision computing*, 22(12):943–957, 2004.

[16] N. Einecke and J. Eggert. A two-stage correlation method for stereoscopic depth estimation. In *Digital Image Computing: Techniques and Applications (DICTA), 2010 International Conference on*, pages 227–234. IEEE, 2010.

[17] Y. Furukawa, C. Hernández, et al. Multi-view stereo: A tutorial. *Foundations and Trends in Computer Graphics and Vision*, 9(1-2):1–148, 2015.

[18] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2010.

[19] S. Galliani, K. Lasinger, and K. Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 873–881, 2015.

[20] S. Galliani and K. Schindler. Just look at the image: viewpoint-specific surface normal prediction for improved multi-view reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5479–5487, 2016.

[21] A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *Asian conference on computer vision*, pages 25–38. Springer, 2010.

[22] M. Gong and Y.-H. Yang. Fast stereo matching using reliability-based dynamic programming and consistency constraints. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 610–617, 2003.

[23] M. Gong and Y.-H. Yang. Fast unambiguous stereo matching using reliability-based dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):998–1003, 2005.

[24] M. Gong and Y.-H. Yang. Near real-time reliable stereo matching using programmable graphics hardware. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 924–931. IEEE, 2005.

[25] P. Guerrero, Y. Kleiman, M. Ovsjanikov, and N. J. Mitra. PCPNet: Learning local shape properties from raw point clouds. *Computer Graphics Forum*, 37(2):75–85, 2018.

[26] R. Haeusler, R. Nair, and D. Kondermann. Ensemble learning for confidence measures in stereo vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 305–312, 2013.

[27] X. Han, Z. Li, H. Huang, E. Kalogerakis, and Y. Yu. High-resolution shape completion using deep neural networks for global structure and local geometry inference. pages 85–93, 2017.

[28] W. Hartmann, S. Galliani, M. Havlena, L. Van Gool, and K. Schindler. Learned multi-patch similarity. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1595–1603. IEEE, 2017.

[29] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341, 2008.

[30] X. Hu and P. Mordohai. A quantitative evaluation of confidence measures for stereo vision. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2121–2133, 2012.

[31] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM transactions on graphics (TOG)*, 28(5):176, 2009.

[32] H. Huang, S. Wu, M. Gong, D. Cohen-Or, U. Ascher, and H. R. Zhang. Edge-aware point set resampling. *ACM transactions on graphics (TOG)*, 32(1):9, 2013.

[33] P.-H. Huang, K. Matzen, J. Kopf, N. Ahuja, and J.-B. Huang. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2821–2830, 2018.

[34] X. Huang, Y. Zhang, and Z. Yue. Image-guided non-local dense matching with three-steps optimization. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, 3(3), 2016.

[35] S. Im, H.-G. Jeon, S. Lin, and I. S. Kweon. DPSNet: End-to-end deep plane sweep stereo. In *International Conference on Learning Representations*, 2019.

[36] R. A. Jellal, M. Lange, B. Wassermann, A. Schilling, and A. Zell. Ls-elas: Line segment based efficient large scale stereo matching. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 146–152. IEEE, 2017.

[37] M. Ji, J. Gall, H. Zheng, Y. Liu, and L. Fang. SurfaceNet: An end-to-end 3D neural network for multiview stereopsis. *arXiv preprint arXiv:1708.01749*, 2017.

[38] S. Ji, W. Xu, M. Yang, and K. Yu. 3D convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.

[39] M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

[40] M. Kazhdan and H. Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):29, 2013.

[41] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry. End-to-end learning of geometry and context for deep stereo regression. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[42] L. Keselman, J. I. Woodfill, A. Grunnet-Jepsen, and A. Bhowmik. Intel realsense stereoscopic depth cameras. *arXiv preprint arXiv:1705.05548*, 2017.

[43] L. Keselman, J. I. Woodfill, A. Grunnet-Jepsen, and A. Bhowmik. Intel realsense stereoscopic depth cameras. *arXiv preprint arXiv:1705.05548*, 2017.

[44] P. Kim, J. Chen, and Y. K. Cho. Slam-driven robotic mapping and registration of 3D point clouds. *Automation in Construction*, 89:38–48, 2018.

[45] S. Kim, D.-g. Yoo, and Y. H. Kim. Stereo confidence metrics using the costs of surrounding pixels. In *Digital Signal Processing (DSP), 2014 19th International Conference on*, pages 98–103. IEEE, 2014.

[46] S.-H. Kim and K.-Y. Chung. Medical information service system based on human 3D anatomical model. *Multimedia Tools and Applications*, 74(20):8939–8950, 2015.

[47] J. Kowalczuk, E. T. Psota, and L. C. Perez. Real-time stereo matching on cuda using an iterative refinement method for adaptive support-weight correspondences. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(1):94–104, Jan 2013.

[48] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[49] N. Lazaros, G. C. Sirakoulis, and A. Gasteratos. Review of stereo vision algorithms: from software to hardware. *International Journal of Optomechatronics*, 2(4):435–462, 2008.

[50] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[51] W. Luo, A. G. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5695–5703, June 2016.

[52] R. Manduchi and C. Tomasi. Distinctiveness maps for image matching. In *Image Analysis and Processing*, pages 26–31. IEEE, 1999.

[53] W. Mao and M. Gong. Disparity filtering with 3D convolutional neural networks. In *15th Conference on Computer and Robot Vision (CRV)*, pages 246–253. IEEE, 2018.

[54] W. Mao, X. Huang, and M. Gong. A global-matching framework for multi-view stereopsis. In *18th international Conference Computer Analysis of Images and Patterns (CAIP)*, 2019.

[55] W. Mao, M. Wang, and M. Gong. Point cloud consolidation through learning-based projection. *Submitted to the 30th British Machine Vision Conference (BMVC)*, 2019.

[56] W. Mao, M. Wang, and M. Gong. A robust framework for multi-view stereopsis. *Submitted to International Journal of Computer Vision (IJCV)*, 2019.

[57] W. Mao, M. Wang, J. Zhou, and M. Gong. Semi-dense stereo matching using dual CNNs. In *IEEE Winter Conference on Applications of Computer Vision*, pages 1588–1597, 2019.

[58] D. Maturana and S. Scherer. 3D convolutional neural networks for landing zone detection from lidar. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 3471–3478. IEEE, 2015.

[59] D. Maturana and S. Scherer. VoxNet: A 3D convolutional neural network for real-time object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 922–928. IEEE, 2015.

[60] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4040–4048, 2016.

[61] M. Menze, C. Heipke, and A. Geiger. Joint 3D estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis (ISA)*, 2015.

[62] P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J.-M. Frahm, R. Yang, D. Nistér, and M. Pollefeys. Real-time visibility-based fusion of depth maps. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1–8. IEEE, 2007.

[63] J. Pang, W. Sun, J. S. Ren, C. Yang, and Q. Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *Proceedings of the International Conference on Computer Vision (ICCV)*, volume 7, 2017.

[64] H. Park and K. M. Lee. Look wider to match image patches with convolutional neural networks. *IEEE Signal Processing Letters*, 24(12):1788–1792, 2017.

[65] M.-G. Park and K.-J. Yoon. Leveraging stereo matching with learning-based confidence measures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 101–109, 2015.

[66] D. Peña and A. Sutherland. Disparity estimation by simultaneous edge drawing. In C.-S. Chen, J. Lu, and K.-K. Ma, editors, *Asian Conference on Computer Vision (ACCV) Workshops*, pages 124–135, Cham, 2017. Springer International Publishing.

[67] M. Poggi and S. Mattoccia. Learning a general-purpose confidence measure based on o (1) features and a smarter aggregation strategy for semi global matching. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 509–518. IEEE, 2016.

[68] M. Poggi and S. Mattoccia. Learning from scratch a confidence measure. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2016.

[69] M. Poggi, F. Tosi, and S. Mattoccia. Quantitative evaluation of confidence measures in a machine learning world. In *Proceedings of the International Conference on Computer Vision (ICCV)*, volume 206, page 17, 2017.

[70] A. Poms, C. Wu, S.-I. Yu, and Y. Sheikh. Learning patch reconstructability for accelerating multi-view stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3041–3050, 2018.

[71] R. Preiner, O. Mattausch, M. Arikan, R. Pajarola, and M. Wimmer. Continuous projection for fast l1 reconstruction. *ACM Trans. Graph.*, 33(4):47–1, 2014.

[72] E. T. Psota, J. Kowalczuk, M. Mittek, and L. C. Perez. Map disparity estimation using hidden markov trees. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2219–2227, 2015.

[73] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. *arXiv preprint arXiv:1612.00593*, 2016.

[74] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[75] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[76] R. Roveri, A. C. Öztireli, I. Pandele, and M. H. Gross. PointProNets: Consolidation of point clouds with convolutional neural networks. *Comput. Graph. Forum*, 37:87–99, 2018.

[77] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German Conference on Pattern Recognition*, pages 31–42. Springer, 2014.

[78] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3):7–42, 2002.

[79] T. Schmidt, R. Newcombe, and D. Fox. Self-supervised visual descriptor learning for dense correspondence. *IEEE Robotics and Automation Letters*, 2(2):420–427, 2017.

[80] A. Seki and M. Pollefeys. Patch based confidence prediction for dense disparity map. In E. R. H. Richard C. Wilson and W. A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 23.1–23.13. BMVA Press, September 2016.

[81] A. Seki and M. Pollefeys. Patch based confidence prediction for dense disparity map. In E. R. H. Richard C. Wilson and W. A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 23.1–23.13. BMVA Press, September 2016.

[82] M. Shahbazi, G. Sohn, J. Théau, and P. Ménard. Revisiting intrinsic curves for efficient dense stereo matching. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, 3(3), 2016.

[83] Y. Sun, S. Schaefer, and W. Wang. Denoising point sets via l0 minimization. *Computer Aided Geometric Design*, 35:2–15, 2015.

[84] T. Taniai, Y. Matsushita, and T. Naemura. Graph cut based continuous stereo matching using locally shared labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1613–1620, 2014.

[85] J. Tham, A. H. Duin, L. Gee, N. Ernst, B. Abdelqader, and M. McGrath. Understanding virtual reality: Presence, embodiment, and professional practice. *IEEE Transactions on Professional Communication*, 61(2):178–195, 2018.

[86] B. Tippetts, D. J. Lee, K. Lillywhite, and J. Archibald. Review of stereo vision algorithms and their suitability for resource-limited systems. *Journal of Real-Time Image Processing*, 11(1):5–25, 2016.

[87] E. Tola, C. Strecha, and P. Fua. Efficient large-scale multi-view stereo for ultra high-resolution image sets. *Machine Vision and Applications*, 23(5):903–920, 2012.

[88] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the International Conference on Computer Vision (ICCV)*, volume 98, page 2, 1998.

[89] F. Tosi, M. Poggi, A. Tonioni, L. Di Stefano, and S. Mattoccia. Learning confidence measures in the wild. In *Proceedings of the British Machine Vision Conference (BMVC)*, volume 2, 2017.

[90] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3D convolutional networks. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 4489–4497. IEEE, 2015.

[91] O. Veksler. Extracting dense features for visual correspondence with graph cuts. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2003.

[92] M. Wang, J. Zhou, W. Mao, and M. Gong. Multi-scale convolution aggregation and stochastic feature reuse for DenseNets. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 321–330. IEEE, 2019.

[93] C. S. Weerasekera, R. Garg, and I. Reid. Learning deeply supervised visual descriptors for dense monocular reconstruction. *arXiv preprint arXiv:1711.05919*, 2017.

[94] S. Wu, W. Sun, P. Long, H. Huang, D. Cohen-Or, M. Gong, O. Deussen, and B. Chen. Quality-driven poisson-guided autoscanning. *ACM Transactions on Graphics*, 33(6), 2014.

[95] T. Yan, Y. Gan, Z. Xia, and Q. Zhao. Segment-based disparity refinement with occlusion handling for stereo matching. *IEEE Transactions on Image Processing*, 2019.

[96] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan. MVSNet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783, 2018.

[97] X. Ye, J. Li, H. Wang, H. Huang, and X. Zhang. Efficient stereo matching leveraging deep local and context information. *IEEE Access*, 5:18745–18755, 2017.

[98] W. Yifan, S. Wu, H. Huang, D. Cohen-Or, and O. Sorkine-Hornung. Patch-base progressive 3D Point Set Upsampling. *ArXiv e-prints*, page arXiv:1811.11286, Nov. 2018.

[99] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng. EC-Net: an edge-aware point set consolidation network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[100] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng. PU-Net: Point cloud upsampling network. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[101] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. In *Proceedings of the Third European Conference-Volume II on Computer Vision - Volume II*, Proceedings of the European Conference on Computer Vision (ECCV), pages 151–158, London, UK. Springer-Verlag.

[102] J. Zbontar and Y. LeCun. Computing the stereo matching cost with a convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1592–1599, 2015.

[103] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17(1-32):2, 2016.

[104] S. Zhang, W. Xie, G. Zhang, H. Bao, and M. Kaess. Robust stereo matching with surface normal prediction. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 2540–2547. IEEE, 2017.

[105] M. Zollhöfer, C. Siegl, M. Vetter, B. Dreyer, M. Stamminger, S. Aybek, and F. Bauer. Low-cost real-time 3D reconstruction of large-scale excavation sites. *Journal on Computing and Cultural Heritage (JOCCH)*, 9(1):2, 2016.

[106] T. M. A. Zulcaffle, F. Kurugollu, D. Crookes, A. Bouridane, and M. Farid. Frontal view gait recognition with fusion of depth features from a time of flight camera. *IEEE Transactions on Information Forensics and Security*, 14(4):1067–1082, 2019.