Integrating Geometric Knowledge into Deep Learning for Dense Stereo Matching

In the course Geodesy and Geoinformatics

Masterthesis

of

Waseem Iqbal

Examiner: Prof. Dr.-Ing. Christian Heipke
Supervisor: M.Sc. Max Mehltretter

HANNOVER 2021

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremdem Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Hannover, 28. Februar 2021

Statement

I hereby declare that this dissertation presents my original work and to the best of my knowledge, works or ideas of other authors cited in this dissertation have been explicitly acknowledged under bibliography.

This work has not been submitted to any other examination authority in the same or a similar form and has not been published.

Hannover, 28. February 2021 Waseem Iqbal

Acknowledgement

I would like to express my sincere gratitude to Max Mehltretter for becoming the supervisor of my thesis. His professionalism, encouragement, support and guidance have been tremendous assets during my journey as a research student. The meetings with Max Mehltretter helped me to speed up the whole process towards completion of my thesis in a timely manner. I am thankful for his assistance and discussions, even learning the smallest details and ideas. He helped, encouraged and boosted my confidence level during my research. Alot of profound thanks go out to my parents Ghulam Rasool and Zubaida Nasreen and my family members especially my brother Naeem Iqbal who have always been a source of intuition for me and have sacrificed their time and money for my studies.

Abstract

Dense depth information can be reconstructed from stereo images using conventional and deep learning based methods. Such traditional methods extract more robust features for dense stereo matching but have limited performance in the areas with occlusions or in the untextured areas. On the other hand, deep learning methods appear to perform much better even in complicated areas but they learn geometrical principles from scratch, which are basically time-consuming and already well-known. Therefore, in this thesis, a convolutional neural network (CNN) is designed to better estimate dense depth map based on geometric knowledge. The key idea of designing the network is hybrid technique of learning the disparity maps. Thus, two different approaches using three different image transformations are formulated to estimate the depth maps in a more profound way. Firstly, GC-Net (Kendall et al., 2017) is trained jointly with a fourth channel apart from RGB channels. CNN is guided by the fourth channel with a collection of hand-crafted features. Additionally, a set of down-scaled features are fed in different resolutions into the decoder part of the network and the effects on disparity maps are investigated. For the evaluation of the network and different approaches, experiments are conducted on the FlyingThings3D dataset. The network is evaluated on the unseen Middlebury dataset and fine-tuned and tested on the KITTI datasets. In conclusion, the results prove that geometric information obtained through conventional methods can contribute to the enhancement of the original depth map without neither increasing the computational complexity nor the memory footprint.

Keywords Convententional Methods, Deep Learning, Convolutional Neural Network, Hybrid Approach, Depth Map Estimation

Contents

1 Introduction

Most of the popular digital cameras are capable of capturing a 2-dimensional (2D) projection of the scene while the components corresponding to the depth are lost (not recorded). Depth information is applied broadly in many applications, such as robotics (Häne et al., 2011), 3-dimensional (3D) model reconstruction (Krutikova et al., 2017), object classification (Badrinarayanan et al., 2017), medical imaging, and vehicle tracking (Ding et al., 2006).

Using such 2D digital images, 3D images may be constructed by extracting the depth information from 2D images using a variety of reconstruction techniques. The immediate way to get the distance information between two objects is to use a ruler to measure it, however, depth information for a 2D image may be acquired in several other methods namely active stereo vision such as structured light (Scharstein and Szeliski, 2003), time-of-flight (Zhu et al., 2011), and passive stereo vision including stereo matching (Ladicky et al., 2015). Passive stereo vision method, stereo matching is arguably the most widely applicable technique because it is more accurate and poses little assumption to the sensors and the imaging procedure.

The stereo vision system mainly consists of three components which are rectification, stereo matching, and triangulation. Owing to the epipolar constraint based image rectification, the search space for the matching can be limited to a 1-dimensional (1D) horizontal line, as compared to a 2D plane in optical flow (Lu et al., 2018). Namely, depth can be estimated by matching corresponding pixels on the two rectified images along the same scan line. We are specifically interested in computing the disparity of each pixel or at least a large majority of pixels between a rectified stereo pair of images, even in challenging regions of an image such as textureless areas. To achieve this, the core task of a stereo algorithm is to compute the correspondence of each pixel between two images, which is quite challenging to achieve robustly in the occluded or textureless areas. The last component includes the process of finding the position of a point in space, given its position in two rectified images involves the intersection of two known rays in space and is commonly known as triangulation.

In the early years, hand-crafted image features such as Haar-like features, Local Binary Patterns (LBP) (Ojala et al., 1994), Histogram of Oriented Gradients (HOG) (Dalal and Triggs, 2005), and Scale-Invariant Feature Transform (SIFT) (Lowe, 2004) were used in most image matching tasks. These features describe images from different aspects that are more robust but have limited description capabilities in challenging situations such as textureless regions. Some dense stereo matching techniques have also been used for this task such as Binary Census Transform (Hu et al.,

2016), Normalized Cross Correlation (Heo et al., 2011), and Absolute Intensity Difference (Hamzah et al., 2010). These techniques are well-considered using prevalent dense stereo matching pipeline but such techniques are prone to errors and lack an overall objective for optimization.

On the other hand, in recent years, especially deep learning based approaches have shown convincing results even in areas without texture. Yet, the problem is their low interpretability, it is difficult to describe what features are learned. In addition, these kinds of methods are mainly data-driven. As a result, these methods appear to generalize poorly to various data sets. Furthermore, they often learn geometrical principles from scratch, which are basically time-consuming and already well-known.

Our intention is not to naively construct a machine learning architecture as a black box to model stereo. Instead, we would like to develop a methodology that enables the incorporation of geometric information into a Convolutional Neural Network (CNN) trained end-to-end for the task of dense stereo matching. Therefore, it is investigated whether hand-crafted feature maps can provide complementary information for CNNs as constraints guiding the training process for the task of disparity estimation. In this research, we would like to investigate if the ideas of existing hand-crafted approaches namely LBP, HOG, and Sobel or Canny operator, etc. can be used to improve the performance of a CNN-based method and make the network more robust. For this purpose, a concept is developed to integrate these ideas on the architecture of an existing CNN specifically Geometry and Context Network (GC-Net) (Kendall et al., 2017).

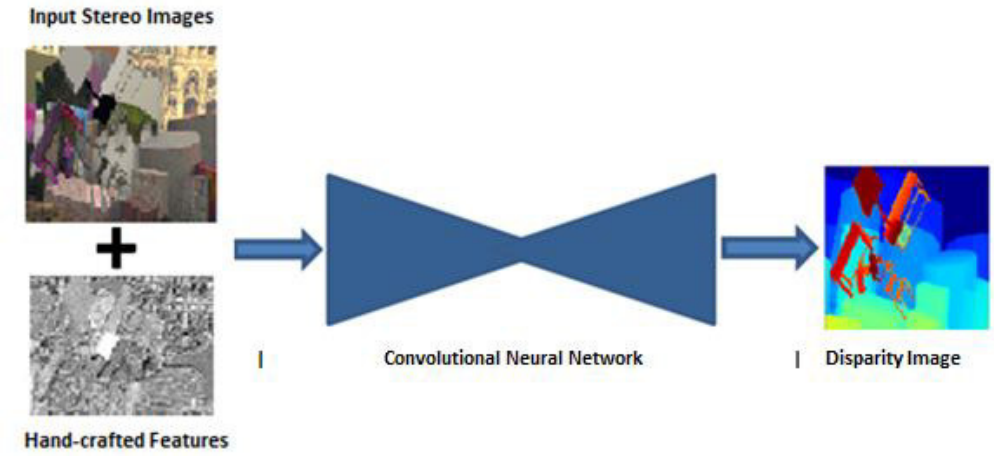


Figure 1.1: Basic Architecture of the designed network.

The key idea of designing the network architecture is to incorporate the geometric information into the learning strategy as shown in Figure 1.1. As the learning task of the network is to learn the essential features that help the network in better disparity estimation. So, this geometric information could be exceptionally valuable. Additionally, the approach is developed to examine its effects on the characteristics of the CNN, such as changes in the accuracy, or in the ability to

generalize unseen dataset during training. We expect this integration can make the network more robust, less data-driven, and might bring a finer final disparity map.

2 Fundamentals

The main purpose of this chapter is to give a basic introduction to general terms and different operations used in this thesis.

2.1 Stereo Image Reconstruction

Disparity (called parallax in photogrammetry) is defined as the displacement of a scene point shown on a stereoscopic image pair and is inversely proportional to the scene depth. A disparity map can be converted into a depth map and vice versa if the camera parameters (e.g., focal length, baseline) are known.

Stereo vision is a technique of inferring depth with two or more images. Stereo vision uses a pair of cameras to capture the images synchronously and then calculates the distance or depth of the object apart from the cameras through the differences between the two images shot by the two cameras separately. If the corresponding (homologous) points are found in two images, then depth can be inferred by utilizing the trigonometric relation.

Essentially in a disparity image, a grayscale heat map is produced whereby lighter shades of gray signifies objects close to the camera lens with progressively darker shades to distinguish objects further away, to proceed to pull depth information from stereo images using a naive approach. The depth of a point in real-world space can be calculated through the use of mathematical formulas and functions that incorporate the principles of epipolar geometry. The image below illustrates proof of these concepts in action.

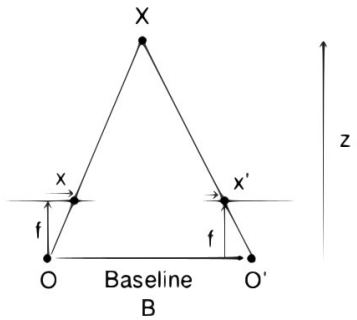


Figure 2.1: Calculation of binocular disparity. Here O and O' are optical centers of two cameras, f is the focal length of cameras, x and x' are the points on the 2D image plane, Z is the depth of point X, and X is 3D point in object space.

So let's break down the diagram above. The first thing to note, the optical centers of both cameras at O and O' in addition to the point in 3D space whose depth is being found; point X. Length f is the focal length of cameras which is assumed to be known, e.g. from previously performed camera calibration. Now x and x' are the points on the 2D image plane where point X has appeared in each image respectively. The idea here is that the equivalent triangles denoted as Ofx and O'fx' can be compared to each other in order to extract the depth Z of point X. The equation that allows accomplishing this is shown below.

$$Disparity = x - x' = \frac{Bf}{Z} \tag{2.1}$$

Here rectified images are used that means points only move in the x-direction and there is no movement in the y-direction. Therefore, the search space for the matching is restricted to a horizontal line, as compared to a 2D plane. To describe the above equation in as plain terms as possible, it tells us that the distance Z of point X away from cameras' optical centers is inversely proportional to the difference in distance of points x and x' (the image points of the object as they appear on the 2D plane).

- Small disparity - disparity value is small when a point is further away from the camera.
- Big disparity - disparity value is big when a point is close to the camera.
- Zero disparity - disparity value is zero when a point is at infinity. That is why it is cannot be negative by convention.

2.2 Convolutional Neural Networks

Convolutional neural networks (CNN) are one of the most popular types of neural network. These are widely used in the fields of image recognition, object detection, as well as in depth enhancement. The basic idea of CNNs is the application of convolutional operations so that various features (e.g., edges, shapes, objects) can be detected.

The values of each filter are manually specified for extracting those features in conventional image processing. In comparison, the filters in CNN can be optimized automatically during the network training process. By doing this, the network can autonomously pick the features that are more helpful for the prediction by using training data.

A typical CNN usually consist of three types of layers: convolutional layer, pooling layer, and fully connected layer. The convolutional layer is the core component of a CNN that does most of the computational work. Other than the convolutional layer, CNN often also uses the pooling layer to reduce the size of the representation and to speed the computation. Each neuron is linked with each previous layer neuron in the fully-connected layer, but each neuron in a convolutional layer receives input only from a local region of the previous layer. As shown in Figure 2.2, each

neuron in a later layer is the result of dot products between the filter and the subarea from the previous layer. In this case, the number of parameters to be learned depends only on the size and quantity of filters, which are drastically lower than the one in a fully connected layer. The same filter is applied through the whole input from top left to bottom right, and then a 2-dimensional (2D) activation map can be obtained, known as a feature map. Each feature map indicates the locations and strength of a specifically detected feature from the input.

One major advantage of CNN is that they reduce the parameters and reuse the learned weights. So, let's consider the filter (orange color) in the image below having 9 pixels and the input image (green color) having 64 pixels. In a fully connected layer, weights are calculated i.e. 9x64 = 576. While in a CNN this same filter keeps on moving (convolving) over the entire image. All pixel values in the image is multiplied with those same 9 values of filter (hence, weights are reused). Therefore, just 9 weights per filter are needed instead of 576 in the FC layer. The job of a filter is to identify features (such as texture, lines, edges, etc), which could be anywhere in an image. So, this same filter is reused over the entire image.

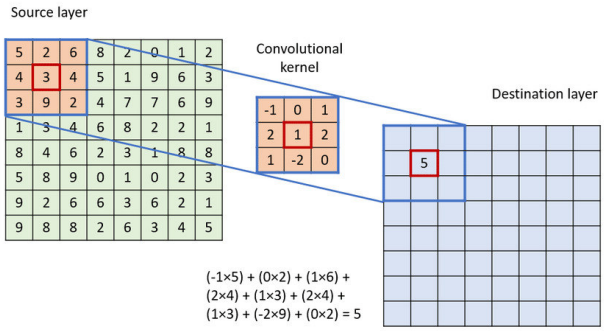


Figure 2.2: Convolutional operation: input image (green) is convolved with a 3x3 convolutional filter (orange) which results in an output image (blue) (Podareanu et al., 2019).

2.3 Image Transformations

This section describes the different image descriptors used to transform the original color images in this thesis and gives a basic understanding, how these operations work.

2.3.1 Local Binary Patterns (LBP)

Local Binary Patterns (LBP) is an effective texture pattern descriptor introduced by (Ojala et al., 1994) to describe the local texture patterns of an image. It is widely used in applications based on image processing (Tianyu et al., 2018).

Calculation procedure: The LBP works in a block size of 3×3, in which the center pixel is used as a threshold for the neighboring pixel. The mathematical expression of LBP is given as

(Hemanth et al., 2019):

$$LBP = \sum_{i=0}^{P-1} s(n_i - G_c)2^i \tag{2.2}$$

$$s(x) = \begin{cases} 1 & \text{, if x>0} \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

here P is the number of neighborhood pixels, n_i represents the ith neighboring pixel, and c represents the center pixel. Hence, for eight neighboring pixels, the histogram feature vector length of 256 is obtained. The LBP operation is shown in 2.3 with a G_c value of 55 and eight neighboring pixels.

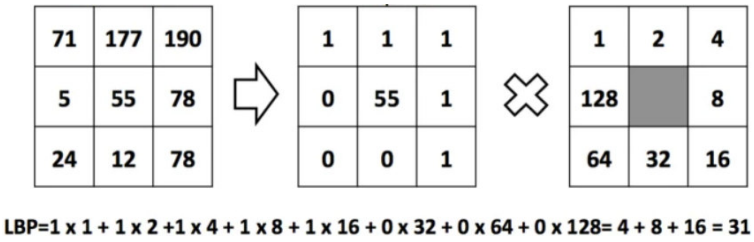


Figure 2.3: Example of Local Binary Pattern calculation for a 3 × 3 pixel neighbourhood. Grayscale pixel values (left), binary pattern (middle), weight matrix (right) (García-Olalla et al., 2013).

2.3.2 Histogram of Oriented Gradients (HOG)

Histogram of Oriented Gradients, is a feature descriptor that is often used to extract features from image data. The HOG descriptor focuses on the shape or the structure of an object.

Calculation procedure: Step-wise calculation procedure is as follows (Histogram of Oriented Gradients, 2018)

The first step is an optional global normalization of the image that is intended to minimize the effects of lighting or illumination. Gamma (power-law) compression is used in practice, either computing the square root or each color channel's log. The intensity of the image texture is generally proportional to the local surface illumination, so this compression helps to minimize the effects of differences in local shadowing and illumination.

The second step calculates image gradients of the first order. The contour, silhouette, and some texture details are captured while providing additional light variations resistance. The locally dominant color channel is used, which provides a large invariance of the color. Variant methods can also involve image derivatives of second-order, acting as rudimentary bar detectors, a useful feature for capturing, e.g. bar-like structures in bicycles and limbs in humans.

The third step is designed to create an encoding sensitive to local image content while resisting minor changes in pose or appearance. The adopted method pools gradient orientation information locally in the same way as the SIFT feature. The image window is divided into small spatial regions, called cells. For each cell, a local 1D histogram of gradient or edge orientations is accumulated over all the pixels in the cell. This combined cell-level 1D histogram forms the fundamental orientation histogram representation. Each orientation histogram divides the gradient angle range into a predetermined number of bins. The gradient magnitudes of the pixels in the cell are used to vote into the orientation histogram.

The fourth step computes normalization, which takes local groups of cells and contrast normalizes their overall responses before passing to the next step. Normalization leads to better invariance to illumination, shadowing, and edge contrast. It is performed by accumulating a measure of local histogram energy over local groups of cells that are called blocks. The result is used to normalize each cell in the block. Typically each individual cell is shared between several blocks, but its normalizations are block-dependent and thus different. The cell thus appears several times in the final output vector with various normalizations. This may seem redundant but it improves the performance. The normalized block descriptors are referred as Histogram of Oriented Gradient (HOG) descriptors.

The final step collects the HOG descriptors from all blocks of a dense overlapping grid of blocks covering the detection window into a combined feature vector for use in the window classifier.

2.3.3 Canny Operator

The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images.

Calculation procedure: The steps of the Canny edge detection algorithm are as follows (Sofiana Sahir, 2019)

Noise Reduction: Since the mathematics involved behind the scene are mainly based on derivatives (Step 2: Gradient calculation), edge detection results are highly sensitive to image noise. One way to get rid of the noise on the image is by applying Gaussian blur to smooth it. To do so, the image convolution technique is applied with a Gaussian Kernel (3x3, 5x5, 7x7 etc...). The kernel size depends on the expected blurring effect. Basically, the smaller the kernel, the less visible is the blur. The equation for a Gaussian filter kernel of size (2k+1)×(2k+1) is given as:

$$H_ij = \frac{1}{2\pi\sigma^2} \exp\left(\frac{-(i-(k+1))^2 + (j-(k+1))^2}{2\sigma^2}\right); 1 \leq i, j \leq (2k+1) \tag{2.4}$$

Gradient Calculation: The Gradient calculation step detects the edge intensity and direction by calculating the gradient of the image using edge detection operators. Edges correspond to a change of pixels' intensity. To detect it, the easiest way is to apply filters that highlight this intensity change in both directions: horizontal (x) and vertical (y) When the image is smoothed, the derivatives Ix

and Iy w.r.t. x and y are calculated. It can be implemented by convolving I with Sobel kernels Kx and Ky, respectively:

$$K_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} K_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \tag{2.5}$$

Then, the magnitude G and the slope θ of the gradient are calculated as follow:

$$|G| = \sqrt{I_x^2 + I_y^2} \tag{2.6}$$

$$\theta(x, y) = arctan\left(\frac{I_y}{I_x}\right) \tag{2.7}$$

Some of the edges are thick and others are thin. Non-Max Suppression step helps mitigate the thick ones. Moreover, the gradient intensity level is between 0 and 255 which is not uniform. The edges on the final result should have the same intensity (i-e. white pixel = 255).

Non-Maximum Suppression: Ideally, the final image should have thin edges. Thus, non-maximum suppression is performed to thin out the edges. The principle is simple: the algorithm goes through all the points on the gradient intensity matrix and finds the pixels with the maximum value in the edge directions. Create a matrix initialized to 0 of the same size of the original gradient intensity matrix; Identify the edge direction based on the angle value from the angle matrix; Check if the pixel in the same direction has a higher intensity than the pixel that is currently processed; Return the image processed with the non-max suppression algorithm. The result is the same image with thinner edges. However, it can still be noticed some variation regarding the edges' intensity: some pixels seem to be brighter than others, and it is tried to cover this shortcoming with the two final steps.

Double threshold: The double threshold step aims at identifying 3 kinds of pixels: strong, weak, and non-relevant: Strong pixels are pixels that have an intensity so high that they contribute to the final edge. Weak pixels are pixels that have an intensity value that is not enough to be considered as strong ones, but yet not small enough to be considered as non-relevant for edge detection. Other pixels are considered as non-relevant for the edge. Now what the double thresholds hold for: High threshold is used to identify the strong pixels (intensity higher than the high threshold) Low threshold is used to identify the non-relevant pixels (intensity lower than the low threshold) All pixels having intensity between both thresholds are flagged as weak and the Hysteresis mechanism (next step) will help to identify the ones that could be considered as strong and the ones that are considered as non-relevant.

Edge Tracking by Hysteresis: Based on the threshold results, the hysteresis consists of transforming weak pixels into strong ones, if and only if at least one of the pixels around the one being processed is a strong one. Finally, outputs the binary image.

3 Related Work

The problem of computing depth from stereo image pairs has been studied for quite some time (Barnard and Fischler, 1982). The requirement of dense output is motivated by modern applications of stereo such as 3D topographic mapping, the extraction of 3D point clouds for surface reconstructions, the generation of digital elevation models, robotics (Häne et al., 2011), autonomous driving, object detection (Ren et al., 2015), recognition, and orthophotographs (Aber et al., 2019), which require disparity estimates in all image regions, even those that are occluded or without texture. Stereo matching algorithms perform this task. Once the disparity is calculated by stereo matching algorithms, the depth of the point can be derived and 3D information is thus retrieved from stereo images. In order to support an informed comparison of stereo matching algorithms Scharstein and Szeliski (2002) develop a taxonomy and categorization scheme for such algorithms. Provided taxonomy performs some subset of:

Matching Cost Computation - the matching cost is a measure of pixel dissimilarity for potentially corresponding image locations (Hirschmuller and Scharstein, 2007). Matching cost computation is very often based on the absolute, squared or the sampling insensitive difference (Gurrieri and Dubois, 2014) of intensities or colors. Since these costs are sensitive to radiometric differences, costs based on image gradients are also used (Klaus et al., 2006). Different concepts have been developed in the past for matching costs computation, Žbontar and LeCun (2016) present a method for extracting depth information using a convolutional neural network. Luo et al. (2016) present a notably faster network for computing local matching costs as a multi-label classification of disparities using a Siamese network, which result in much better matching performance. They examine two network architectures (one tuned for speed, the other for accuracy) by learning a similarity measure on small image patches and Kendall et al. (2017) use a concatenation of deep unary features from which matching cost is computed by an encoder-decoder structure.

Cost Support Aggregation - cost aggregation connects the matching costs within a certain neighborhood. Often, costs are simply summed over a fixed-sized window at constant disparity (Mustaniemi et al., 2015) and refined by cross aggregation and semi-global matching (Zbontar and LeCun, 2015). Some methods additionally weigh each pixel within the window according to color similarity and proximity to the center pixel (Yoon and Kweon, 2006). Another possibility is to select the neighborhood according to segments of constant intensity or color.

Disparity Computation/Optimization - disparity computation is done for local algorithms by selecting the disparity with the lowest matching cost (Mustaniemi et al., 2015) i.e. winner takes all.

Global algorithms typically skip the cost aggregation step and define a global energy function that includes a data term and a smoothness term. The former sums pixel-wise matching costs, while the latter supports piecewise smooth disparity selection. Some methods use more terms for penalizing occlusions, alternatively treating visibility, enforcing a left/right or symmetric consistency between images, or weight the smoothness term according to segmentation information (Hirschmüller, 2006). The strategies for finding the minimum of the global energy function differ. Dynamic Programming (DP) approaches (Gurrieri and Dubois, 2014) perform the optimization in 1D for each scanline individually, which commonly leads to streaking effects. This is avoided by tree-based DP approaches (Abseher et al., 2017). A two-dimensional (2D) optimization is reached by Graph Cuts (Kolmogorov and Zabih, 2001) or Belief Propagation (Klaus et al., 2006). Layered approaches perform image segmentation and model planes in disparity space, which are iteratively optimized.

Disparity Refinement - disparity refinement is often done for removing outliers, checking the consistency, interpolating gaps, or increasing the accuracy by sub-pixel interpolation (Hirschmüller et al., 2002). The hybrid segmentation algorithm is proposed that is based on a combination of the Belief Propagation and Mean Shift algorithms with the aim to refine the disparity and depth map by using a stereo pair of images. This algorithm utilizes image filtering and modified Sum of Absolute Differences (SAD) stereo matching method (Kamencay et al., 2012).

Two main approaches will be discussed for estimating the disparity from stereo image pair:

Traditional Methods - traditional stereo matching methods are well studied following the above mentioned popular stereo pipeline. However, such a stage-wise pipeline is prone to errors in each step and lacks an overall objective for optimization. Traditional stereo methods find a dense set of correspondences that have high photoconsistency, while at the same time forming a (piece-wise) smooth surface. The key issue is to efficiently approximate the smoothness prior and most methods have limited performance in challenging situations, such as areas with poor texture, repetitive patterns, occlusions, and areas with large height discontinuities. In contrast, these hand-crafted approaches may provide additional feature maps supplementing learned ones. Stucker and Schindler (2020) put this idea in practice and propose an astonishingly simple scheme to use stereo networks for dense 3D reconstruction: instead of replacing existing (hand-crafted or learned) stereo matchers with a deep network, they complement them. The network receives as input both an initial digital elevation model (DEM) and the stereo images and estimates a depth correction that is added to the DEM to improve it. The main difficulty with this traditional approach is that it is necessary to choose which features (small interesting, descriptive, or informative patches) are important in each given image. As the number of objects increases, feature extraction becomes more and more cumbersome.

Deep Learning - In the last few years, the focus has been on stereo methods that harness the power of deep learning. Deep learning methods such as Convolutional Neural Networks (CNN) mostly improve prediction performance using big data and plentiful computing resources and have pushed the boundaries of what was possible. For stereo matching, convolutional neural networks

(CNN) have first been introduced to calculate matching costs (Zbontar and LeCun, 2015). Though CNN based methods generally out-perform the traditional ones, these algorithms suffer from a high computational burden because they calculate matching costs at all potential disparities. In addition, they require extra post-processing steps and hand-crafted regularization to produce complete disparity results, because matching cost computation is only one step of stereo matching. Therefore, it was suggested to integrate all steps into an end-to-end network to directly estimate the disparity from stereo images, disparity network (DispNet) is the first end-to-end learning framework (Mayer et al., 2015). Deep learning introduced the concept of this framework to estimate per-pixel disparity from a single rectified image pair. Additionally, the engineering design complexity is shifted. Hence, more recent methods use encoder-decoder architectures to directly output disparity maps.

DispNet utilizes an encoder-decoder architecture for disparity regression. Given a pair of rectified images, it explicitly extracts features in the encoder part and then directly estimates the disparity map in the decoder part by minimizing a regression training loss based on the absolute difference between prediction and ground truth disparity (Mayer et al., 2015). DispNet has achieved prominent performance and has become a baseline network in stereo matching. However, the loss function employed in DispNet results in over-smoothing output disparities, which leads to a loss of local details especially in large disparity discontinuity areas. In addition, it is found that DispNet has lower accuracy for large disparities (Kang et al., 2019). Therefore, several studies have improved the performance of DispNet, e.g. by stacking multiple networks. For instance, DispNet-css combines three separate networks (similar to DispNet) with residual connections to predict more accurate disparity (Ilg et al., 2018) and PSM-Net (Shaked and Wolf, 2016). Kang et al. (2019) use DispNet and add dilated convolution to exploit multi-scale context information, which is beneficial for disparity estimation in weakly texture areas. They construct the matching cost volume with patch-based correlation to handle larger disparities and incorporate disparity gradient information as a gradient regularizer into the loss function to preserve local structure details.

The network architecture of Geometry and Context Network (GC-Net) (Kendall et al., 2017) also follows a coarse-to-fine fashion called encoder-decoder structure. It also encodes information of a wider context at low resolution through successive convolutions and activations and then decodes the result back to the original resolution by successive deconvolutions. GC-Net model has the capacity to learn contextual information. While DispNet applies a 1-D correlation to mimic the cost volume, GC-Net regresses disparity by employing 3D convolutions over 4-D volume to obtain matching costs and finally applied a differentiable version of argmin to select the best disparity along this volume. Additionally, GC-Net uses a feature representation for computing the stereo matching cost rather than using raw pixel intensities. This is basically done to compare a descriptor that is more robust to the ambiguities in photometric appearance and can incorporate local context. It already gives us a good reason and accuracy to consider it as a baseline. Therefore, GC-Net is used as a base network for the stereo matching problem.

Mayer et al. (2015) extends further to the concept of optical flow estimation via convolutional

networks using large-scale datasets (35000 stereo image pairs with ground truth disparity, optical flow, and scene flow) for real-time disparity and scene flow estimation. Though the focus of this work is on the binocular stereo, it is worth noting that the representational power of deep convolutional networks also enables depth estimation from a single monocular image (Eigen et al., 2014). Finally, (Zhang and Seitz, 2007) present an approach for estimating the parameters for Markov Random Field (MRF) based stereo algorithms. This approach is based on a new formulation of stereo as a maximum a posterior (MAP) problem in which both a disparity map and MRF parameters are estimated from the stereo pair itself.

Hybrid Approach - There are clear trade-offs between traditional computer vision and deep learning-based approaches. Classic computer vision algorithms are well-established, transparent, robust, and optimized for performance and power efficiency, some areas where traditional computer vision techniques remain relevant such as being utilized in hybrid approaches to improve performance, panoramic vision, video stabilization (Zhang et al., 2017) and 3D vision (Alldieck et al., 2017) for which deep learning models have not yet been fully optimized (O'Mahony et al., 2020), while deep learning offers greater accuracy and versatility at the cost of large amounts of data and computing resources. Hybrid approaches combine traditional computer vision and deep learning and offer the advantages traits of both methodologies (Tianyu et al., 2018).

ResDepth (Stucker and Schindler, 2020) generates an approximate reconstruction with a stereo matcher, then rewrap the input image with that approximate model, with the initial reconstruction and the warped images as input, train a deep encoder-decoder network to upgrade that initial estimate by regressing an additive residual correction. They argue that such a purely learning based solution may be inefficient because it has to learn many things from data that are already well captured by existing stereo methods. Importantly, classical stereo matching algorithms are very robust in the sense that their outputs are usually correct as a coarse, global estimate of the scene surface but may suffer from local biases and errors. Our intention is quite similar to ResDepth, but, a traditional operator is used and fed this operated image to the convolutional neural networks as additional feature maps supplementing learned ones or to guide the training process and make the network more robust.

Therefore, rather than relying solely on learned features, a hybrid approach is followed to excel the advantages traits of both approaches for the computation of depth from stereo image pairs. This notion is utilized based on GC-Net.

4 Methodology

This chapter describes the various approaches applied in this thesis to improve the accuracy of the disparity map. The aim of this research is to improve the quality of stereo matching results based on Convolutional Neural Network (CNN) by guiding CNN with hand-crafted features. For this purpose influence of three different operators namely Local Binary Patterns (LBP), Histogram of Oriented Gradients (HOG), and Canny Operator is investigated.

A feature descriptor is a representation of an image or an image patch that simplifies the image by extracting useful information and throwing away extraneous information. These feature descriptors enable achieving a high recognition accuracy without requiring a costly model learning approach on massive data. They progressively extract features from input images that could also aid CNN in the estimation of disparity as it has been proven in the field of image classification (Tianyu et al., 2018).

This chapter is structured as follows: the first section explains the network architecture used in this thesis. The second section elaborates three different image transformations, which is followed by the introduction of transformed images as an additional channel. The last section describes the geometry guided decoder part.

4.1 Network Architecture

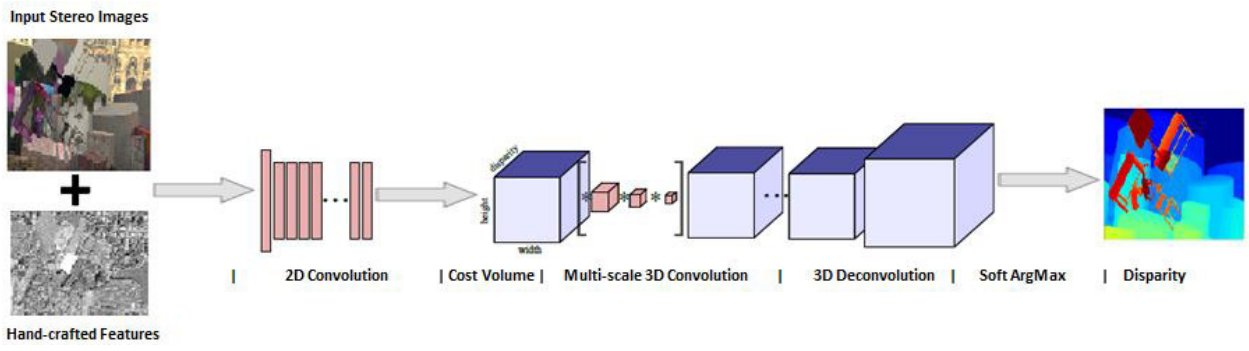


Figure 4.1: Adapted Network Architecture, 2D convolutions learn the features, which are used to form cost volume, 3D convolutions learn features from height, width and disparity dimensions, 3D deconvolutions upsample the volume, and Soft ArgMax is used to estimate disparity over the cost volumes disparity dimension.

GC-Net (Kendall et al., 2017) is used as a baseline network. It has the capacity to learn semantic reasoning and contextual information. The network is a systematic end-to-end mapping from a stereo image pair. GC-Net uses a feature representation for computing the stereo matching cost rather than using raw pixel intensities. This is primarily done to compare a descriptor that is more robust to the ambiguities in photometric appearance and can incorporate local context. Firstly, deep representation is learned through a number of 2D convolutional operations. Then deep unary features are used to compute the stereo matching cost by forming a cost volume of dimensionality height×width×(max disparity + 1)×feature size. Each encoder level consists of a convolutional layer. Each convolutional layer is followed by a batch normalization layer and a rectified linear unit (ReLU) non-linearity. Parameters between the left and right towers are shared to more effectively learn corresponding features (shown with square brackets 4.1). This approach guarantees to diminish much of the engineering design complexity. The decoder part is similar, except transposed convolutions for upsampling the result back to the original resolution. Finally, a cost volume is produced and the disparity is estimated by performing a soft argmin operation over the cost volumes disparity dimension. GC-Net yields good results but depth discontinuities e.g. object boundaries are identified as a weakness of the GC-Net.

Figure 4.1 shows the basic network architecture used in this thesis. GC-Net is not completely modified but adapted by adding an additional image channel. The main notion is to incorporate the results of existing hand-crafted approaches as additional feature maps supplementing learned ones or as constraints guiding the training process. Therefore, a supplement image is fed to initially guide the GC-Net to the actual problem. All inputs i.e. RGB channels + hand-crafted features' channel are simply stacked into a block and fed to the network. Dominant components of the network are described above. On the other hand, our naive approach is not just limited to GC-Net but can also be adapted to other convolutional neural networks e.g. DispNet.

In the second approach these additional hand-crafted features are down-scaled and inserted in the decoder part of the GC-Net in different resolutions until the original resolution is reached see Figure 4.8. The key idea is to concatenate down-scaled hand-crafted features in the decoder part by adding another dimension but spatial resolution remains unchanged.

4.2 Image Transformations

As stated in the previous section that depth discontinuities are identified as a downside of this network. Contrarily, LBP, HOG, and Canny descriptors describe images from a completely different perspective and output useful and robust information such as strong gradient map, oriented edge, and binary feature map. That is why it is a good idea to investigate whether transformed images can help boost the task of disparity estimation. This section explains three different transformations of the original RGB images. Images are transformed to extract more important information which can guide the network in addition to RGB images for better disparity estimates especially in challenging

areas e.g. thin objects, untextured areas, or incomplete object boundaries.

4.2.1 Local Binary Patterns (LBP)

Local Binary Patterns (LBP) is an effective texture pattern descriptor introduced by (Ojala et al., 1994) to describe the local texture patterns of an image, this is done by dividing an image into several small regions from which the robust features are extracted. LBP can be used for tasks such as classification, detection, and recognition. It has widely been used in applications based on image processing such as text identification (Jung and Oh, 2010), face recognition Alhindi et al. (2018), and image classification (Tianyu et al., 2018).

LBP is considered as the part of the network due to its discriminative power, computational simplicity, and probably the most important property in real-world applications i.e. robustness to monotonic gray-scale changes caused, for example, by illumination variations. Additionally, LBP has improved the performance of CNN in image classification (Tianyu et al., 2018). Therefore, it is also expected to guide CNN in the context of dense stereo matching i.e. disparity estimation rather than solely relying on the RGB images.

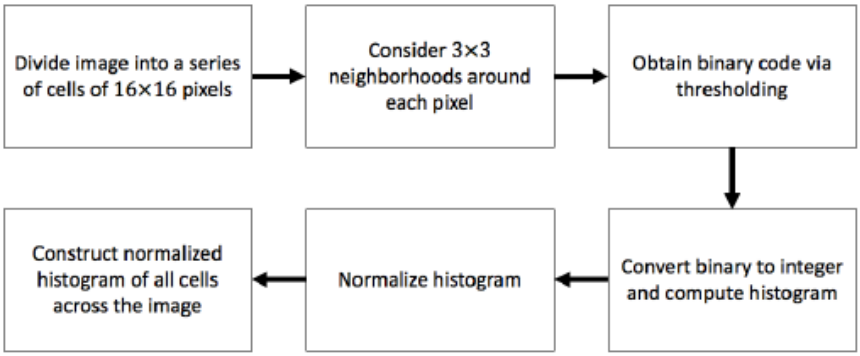
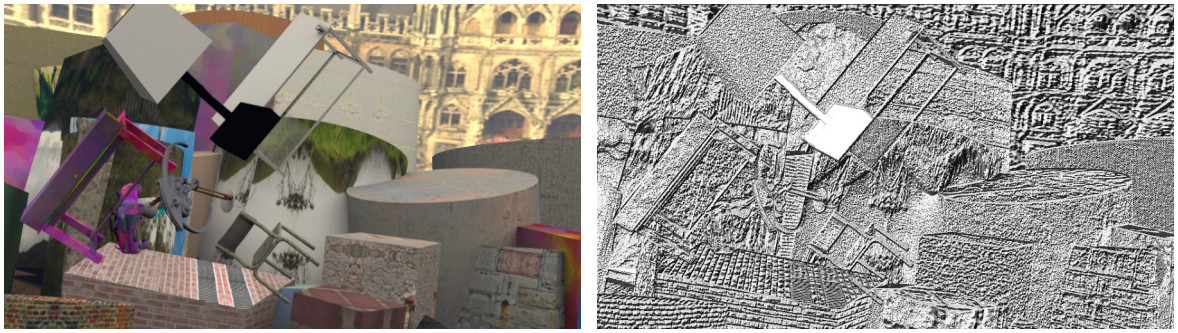


Figure 4.2: Flowchart summarizing the main steps of LBP feature extractor (Alhindi et al., 2018).



(a) Original RGB image (b) Transformed image

Figure 4.3: An example original image is transformed to LBP image

4.2.2 Histogram of Oriented Gradients (HOG)

HOG, or Histogram of Oriented Gradients, is a feature descriptor that is often used to extract features from image data. It is widely used in computer vision tasks for object detection. The HOG descriptor focuses on the shape or the structure of an object. This is different from the edge features we extract for images, in the case of edge features, it is only identified if the pixel is an edge or not. HOG provides the edge direction as well. It is done by extracting the gradient and orientation (magnitude and direction) of the edges.

Additionally, these orientations are calculated in localized portions. It means that the complete image is broken down into smaller regions and for each region, the gradients and orientation are calculated.

Finally, the HOG would create a histogram for each of these regions separately. The histograms are generated using the gradients and orientations of the pixel values. One of the main problems in the disparity estimation is the incomplete shape of the object boundaries which leads to wrong disparity estimation, as per the definition of HOG, it mainly focuses on the structure of the objects. This is one of the main reasons to consider HOG as a part of the network. Therefore, it is expected that it could help CNN to find the full shape or the structure of the objects at different disparity levels as it was useful in face recognition (Deniz et al., 2011), image classification (Tianyu et al., 2018), and in many other fields.

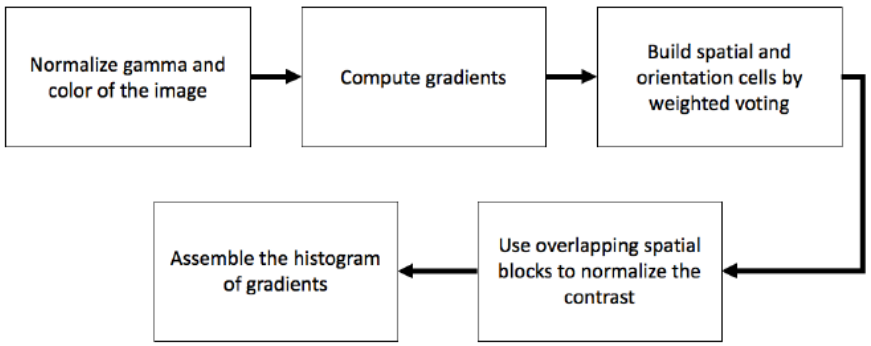
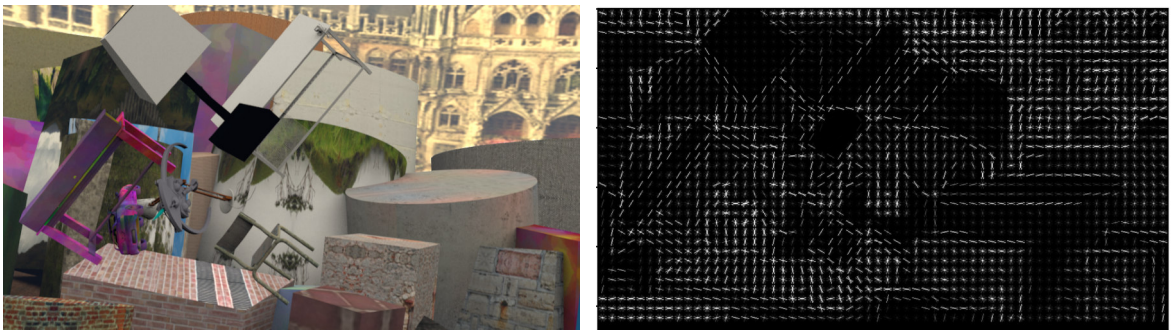


Figure 4.4: Flowchart summarizing the process of computing HOG(Alhindi et al., 2018).



(a) Original RGB image (b) Transformed image

Figure 4.5: An example original image is transformed to HOG image.

4.2.3 Canny Operator

The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. It was developed by John F. Canny in 1986 (Canny, 1986).

Canny edge detection is used to extract useful structural information from different vision objects and dramatically reduce the amount of data to be processed. Additionally, the requirements for the application of edge detection on diverse vision systems are relatively similar. The canny operator outcomes a binary complete edge map and one of the main issues in disparity estimation is very thin object structures such as human limbs, etc which disappear in the disparity images. The canny operator was basically designed to extract edges to one-pixel level, which makes it rather different from LBP and HOG. Edge contains elementary features of an image and also describe boundary of the image, which can already give the CNN essential features. Therefore, it is considered to be the part of the network.

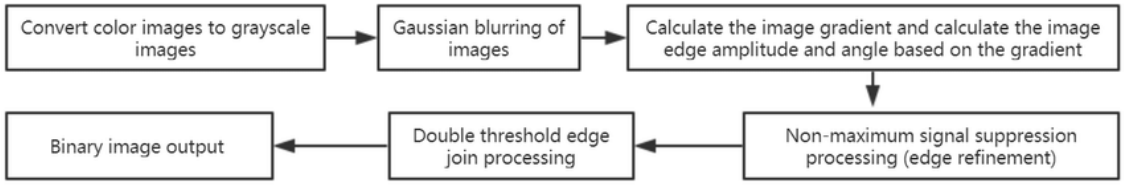
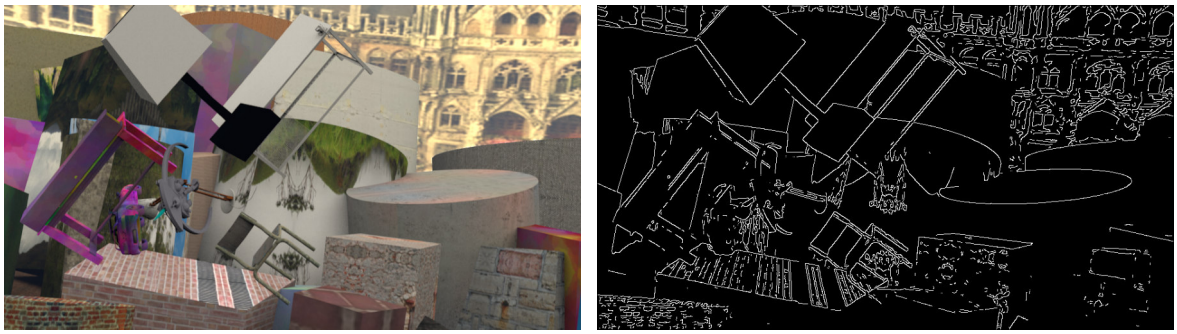


Figure 4.6: Flowchart summarizing the process of computing canny edge detection algorithm(Mao and Hu, 2019).



(a) Original RGB image (b) Transformed image

Figure 4.7: An example original image is transformed to canny image.

4.3 Introduction of Fourth Input Channel

One of the most important problems in the disparity estimation is the depth discontinuities e.g. contour of the objects which is also recognized as the weakness of the original GC-Net. The network commonly cannot distinguish between the foreground and the background objects, whether an object belongs to the foreground or background. Additionally, in the RGB images typically edges or object boundaries and disparity discontinuities coincide. In contrast, descriptors e.g. LBP, HOG, Canny, etc. extract useful information from the images such as binary edge maps, gradient information. Therefore, it is expected that these traditional descriptors might give abundant complementary knowledge to CNN.

Hand-crafted features are computed from the RGB images; it is then assumed as the fourth channel for both input images supplementing the RGB information. The fourth input channel is added to the network because it could aid the network a lot of information about the actual problem and it could also divert the network to the expected solution. In addition, it does not require notable additional parameters i.e. 0.02 % to be learned and also no increase of the memory footprint. Therefore, it is expected that the network might detect the edges in the early layers and maybe use these important features or information added as a fourth input channel. It might detect the part of the objects in the later layers and this information can also be used as a guide to the network.

Another notion of introducing the geometrical information as a fourth input channel is to speed up the learning process of the network by giving such useful information as input. So, it uses these essential features in the early layers and then mainly focuses on the more sophisticated features in the rest of the layers.

4.4 Geometry Guided Decoder

Hand-crafted information can also be used in different sections of the network such as decoder part, loss function, or as a prior. It is quite an interesting idea to introduce these hand-crafted features in the decoder part of the network in different resolutions. The main idea of using the hand-crafted features is to concatenate this information by setting an extra dimension i.e. number of features is increased in the decoder part that contains more crucial information. It is important to know that the spatial resolution of the images remains unchanged. Besides, it does not make major changes in the number of parameters i.e. 0.18 % increase. There is no enlargement in the size of the intermediate tensors of the model and no extra memory footprint is required. It is expected to get better accuracy without any significant changes in the size of the network, it can be seen in Table 4.1 below. Although, just LBP based images are used in the decoder part of the network due to time limitations, other descriptors' based images can also be used.

The intuition is that the network might have found these added image features unimportant as the images get smaller and smoothed in the encoder part of the network and sharp object boundaries do not look clear anymore. Hence, maybe all these features are not used in the end. On the other hand, as the images get smaller and blurry in the encoder part of the network. Thus, there is no way to get this information back to the network. Therefore, these additional hand-crafted features are added i.e. down-scaled LBP based image still contains all the important features so that the network could get this unavailable information or features and this knowledge might help the network to estimate the disparity map in a good resolution during upsampling.

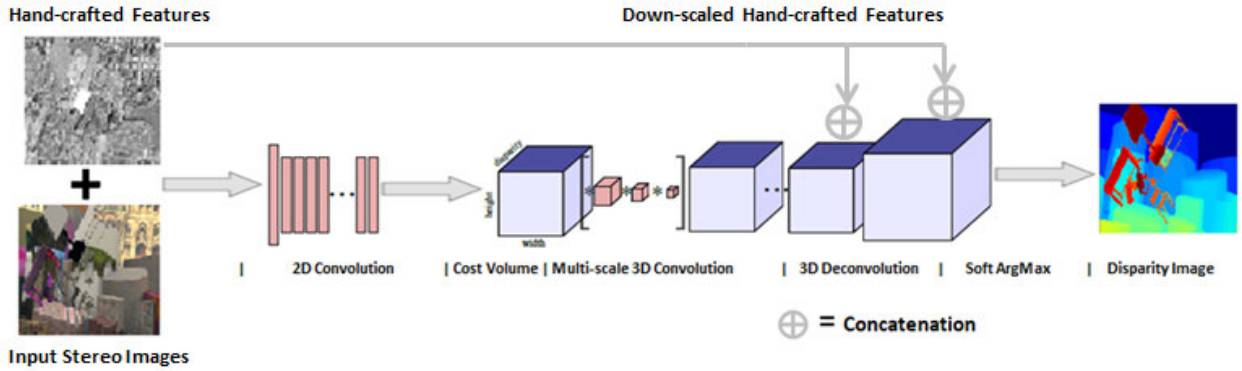


Figure 4.8: Adapted network architecture, down-scaled hand-crafted features are fed in the decoder part of the network in different resolutions. The network parts are described in Figure 4.1.

Table 4.1: Comparison of no. of parameters in the original network, by adding the fourth channel, and by adding no. of features in decoder part.

Layer (type)	No. of parameters		
	Original network	Fourth channel	Decoder
FeatureExtraction	164,256	165,056	165,056
CostVolume	0	0	0
EncoderDecoder3D	2,688,961	2,688,961	2,693,308
Soft Argmin	0	0	0
Total no. of parameters	2,853,217	2,854,017	2,858,364
Trainable parameters	2,848,417	2,849,217	2,853,564
Non-trainable parameters	4,800	4,800	4,800

5 Experiments

This chapter describes the datasets, experimental set-up, and evaluation applied in this thesis.

5.1 Datasets

In this thesis FlyingThings3D (Mayer et al., 2015) a publically available dataset is used. It is a synthetic dataset of flying objects with varied backgrounds. 12,000 stereo examples in 540x960 pixel resolution with ground truth from this dataset are used for training. The dataset also includes dense disparity maps which are again used for the validation of 500 images. Additionally, 30 random and unseen images are used for testing.

For experimentation on unseen datasets, the Middlebury (Scharstein et al., 2014) dataset is used. The Middlebury stereo dataset has been a de-facto standard in the field since 2007. Second, it provides real static indoor scenes with ground truth. Dataset is captured with structured light and has subpixel accuracy i.e. 0.2 pixels on most observed surfaces. In this thesis, 15 images are tested using the parameters of pre-trained network on the FlyingThings3D dataset.

The network is fine-tuned using the 350 training images from the KITTI Stereo (Menze and Geiger, 2015) and KITTI Stereo (Geiger et al., 2012) datasets. Both datasets were captured using vehicle mounted stereo camera set-ups and provide LIDAR based ground truth disparity maps with disparities for 30 % of the pixels. Containing various street scenes from urban as well as rural environments, these datasets still pose a challenge to dense stereo matching algorithms. Accuracy is lower than the Middlebury dataset i.e. pixel level. In our work, 10 training images are used for validation and 34 images are used for testing.

5.2 Training Settings

This research is carried out in TensorFlow 2.1.0 using Keras API. GC-Net is just trained on the training data set. A crop size of 128 x 256 is used due to hardware limitations.

- Image normalization - image normalization is done in the preprocessing step, which ensures that each input parameter (pixel, in this case) has a similar data distribution. This makes convergence faster while training the network. Data normalization is done by subtracting the mean from each pixel and then dividing the result by the standard deviation. The distribution of such data would resemble a Gaussian curve centered at zero.

- Batch size - stochastic mode is used, which means batch size is equal to one and the gradient and the neural network parameters are updated after each sample.

- Optimizer - RMSprop optimizer is used. The RMSprop optimizer is similar to the gradient descent algorithm with momentum. The RMSprop optimizer restricts the oscillations in the vertical direction. Therefore, the learning rate can be boosted and the algorithm could take larger steps in the horizontal direction converging faster.

- Learning rate - learning rate controls how quickly the model is adapted to the problem. Smaller learning rates require more training epochs given the smaller changes made to the weights each update, whereas larger learning rates result in rapid changes and require fewer training epochs. A learning rate that is too large can cause the model to converge too quickly to a suboptimal solution, whereas a learning rate that is too small can cause the process to get stuck. The challenge of training deep neural networks involves carefully selecting the learning rate. It may be the most important hyperparameter for the model. Therefore, a learning rate of 0.001 is used in our project.

LBP, HOG, and Canny are used as a fourth input channel and GC-Net is trained on the FlyingThings3D dataset using above mentioned parameters i.e. batch size = 1, RMSprop optimizer, and learning rate of 0.001. The network estimates the errors on the training data set and the validation data set. When minimum validation error is seen, training is halted. This means that the GC-Net can generalize to unseen data. If the training is stopped when the training error is minimum then the network has been over-fitted and the GC-Net cannot generalize to random and unseen data. The validation error is usually bigger than the training error in most cases. This process is called cross-validation. There are more complicated methods to do cross-validation. It is important to note that 30 epochs are used to train the network.

Secondly, LBP based images are used in different resolutions in the decoder part. As previously, same parameters are used to train GC-Net on the FlyingThings3D dataset for 30 epochs.

Finally, the network is fine-tuned using pre-trained weights of the FlyingThings3D dataset. Once again same parameters are used to train the GC-Net on KITTI datasets for 40 epochs. Training is stopped when minimum validation loss is noticed. It is important to note that all four variants are trained on KITTI datasets. The network is also tested on 34 images.

5.3 Evaluation Settings

Performance metrics (error measures) are vital components of the evaluation frameworks in the field of deep learning. In this work, Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and pixel error rates with different thresholds (see Table 6.1) are used to assess the performance of our designed network.

- Mean Absolute Error (MAE) - The MAE is the simplest regression error metric. It calculates the residual for every pixel and takes only the absolute value of each pixel so that negative and positive residuals do not cancel out. Then the average of all these residuals is calculated. Effectively, MAE describes the typical magnitude of the residuals. In our case, the pixels with valid ground truth disparity are considered to be evaluated. MAE is the absolute difference between the ground truth value and the value predicted by the network.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|d_i - \hat{d}_i| \tag{5.1}$$

  Here and in the equation below, N is the number of pixels with available ground truth disparity, d is the predicted disparity, and $\hat{d}$ is the ground truth disparity.

- Root Mean Square Error (RMSE) - RMSE is considered an excellent general-purpose error metric for numerical predictions. It is the square root of the mean of the square of all of the errors. In other words, it tells us how concentrated the data is around the line of best fit. In our case, it is calculated by the square root of the averaged squared difference between the ground truth value and the predicted value.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(d_i - \hat{d}_i)^2} \tag{5.2}$$

- Pixel error rate - 1, 3, and 5-pixel metrics are also used. If the absolute difference between the ground truth disparity and the predicted disparity is larger than a threshold of 1, 3 and 5 pixels, this pixel is considered an incorrect pixel.

Table 5.1: Summary of errors used.

Error	**Residual Operation**	**Robust To Outliers**
Mean Absolute Error	Absolute Value	Yes
Root Mean Square Error	Square	No

Evaluation of the network is done on the unseen Middlebury dataset (Scharstein et al., 2014). It is mainly carried out to see the domain gap i.e. network is not trained on the Middlebury dataset but pre-trained weights of FlyingThings3D are used. Additionally, the network is fine-tuned on the KITTI datasets i.e. trained on KITTI datasets using pre-trained weights of FlyingThings3D. It is worth mentioning that all four variants are used in this whole evaluation process.

6 Results and Discussion

This chapter presents the evaluation results pertaining to the disparity estimation approaches described in Chapter 4. It is important to know that in this work original GC-Net (Kendall et al., 2017) is used as a baseline network. The aim of the evaluation is to investigate the effects of different methodologies on the quality of disparity estimates. A quality assessment of the results is made through a number of aspects. Firstly, a comparison of the performance is made between the original and the four variants using the FlyingThings3D dataset i.e. comparison of qualitative and quantitative results. Secondly, a comparison of the performance is made on unseen Middlebury dataset and fine-tuned KITTI datasets. These variants are:

Method 1: LBP - used as a fourth input channel

Method 2: HOG - used as a fourth input channel

Method 3: Canny - used as a fourth input channel

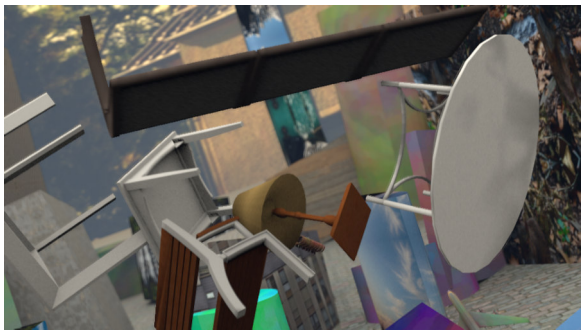
Method 4: LBP (decoder) - used in the decoder part

6.1 Qualitative and Quantitative Results based on the FlyingThings3D Dataset

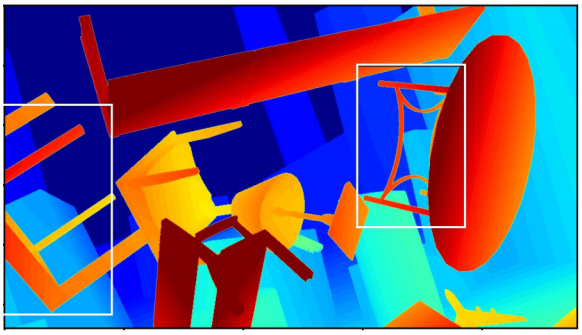
Table 6.1: Evaluation results of all variants on FlyingThings3D dataset.

	Original	**Canny**	**HOG**	**LBP**	**LBP (decoder)**
Pixel error 1	13.82 %	11.24 %	11.69 %	10.25 %	11.24 %
Pixel error 3	6.62 %	5.13 %	5.20 %	4.46 %	4.88 %
Pixel error 5	4.86 %	3.68 %	3.68 %	3.11 %	3.32 %
MAE [pixel]	1.432	1.088	1.095	0.927	0.991
RMSE [pixel]	6.131	4.975	5.079	4.446	4.639

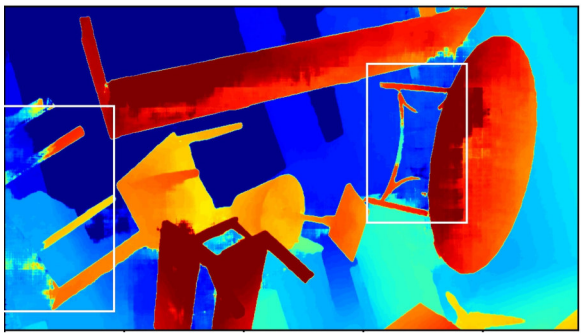
Table 6.1 summarizes the performance of each method based on five evaluation metrics. As expected all four variants performed better than the original variant. It can be seen that the values of MAE are already decreased by 24.02 %, 23.53 %, 35.26 %, and 30.80 % with the integration of Canny,



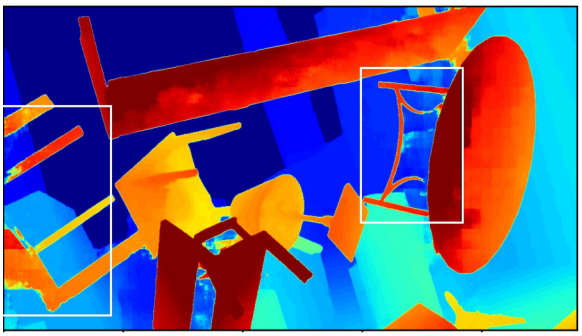
(a) Original RGB image



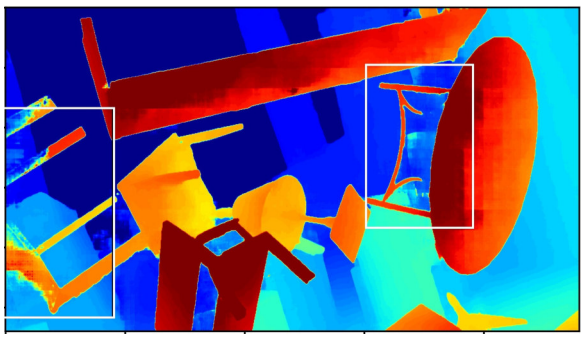
(b) Ground truth disparity image



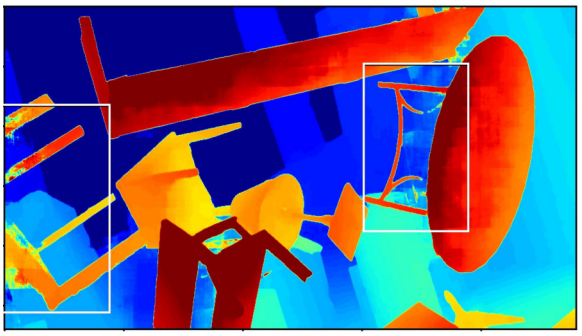
(c) Original disparity image



(d) LBP disparity image



(e) HOG disparity image



(f) Canny disparity image

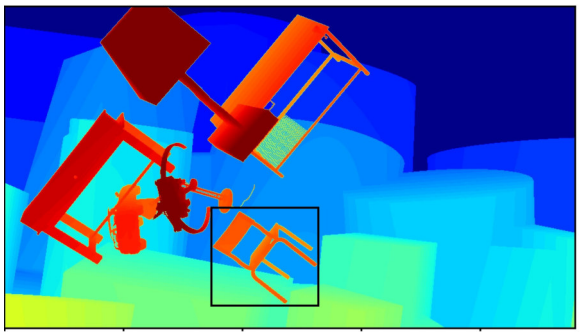
Figure 6.1: Qualitative result of all variants as well as ground truth and original disparity image on KITTI dataset. Problematic areas are shown in white rectangle comparing ground truth disparity image. Here red color indicates object is close to the camera (high disparity) and blue color indicates object is further away from the camera (low disparity). Colors range from dark blue (0 disparity) to red color ($\geq$ 100 disparity).

HOG, LBP, and LBP (decoder) respectively. Similar response is seen from the RMSE values i.e. dropped by 18.85 %, 17.16 %, 27.48 %, and 24.33 % respectively on the FlyingThings3D dataset. Overall LBP worked significantly better i.e. $\sim$10 % than HOG and Canny operators.

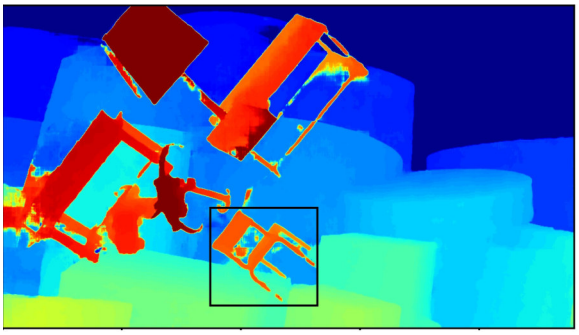
The table also shows the three-pixel-error metrics. LBP's performance is best among these methods as three-pixel-error and five-pixel-error are decreased by about 2.5 and 1.7 percentage points, while one-pixel-error is decreased by 3.5 percentage.

Additionally, Figure 6.1 shows that the thin limbs of the round-table on the right side of the original disparity image has a gap but it is better connected in the other disparity images. Furthermore, a small table like object (marked with the white rectangle) on the left side is almost completely not visible in the original disparity image but in all three variants is partially visible but the best appearance is seen in LBP. The reason for its better performance is perhaps its property of robustness to monotonic gray-scale changes such as illumination effects.

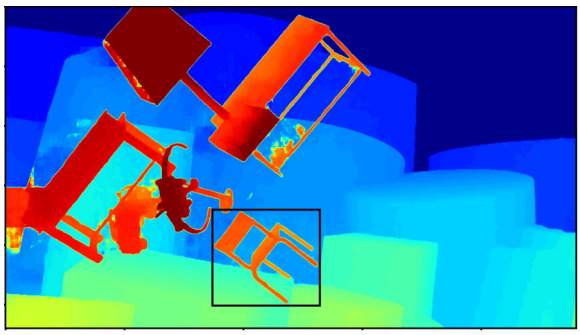
After seeing the performance of the LBP on qualitative and quantitative results as a fourth input channel, LBP based image was used in the decoder part of the network in different resolutions. Which also performed better than the original, HOG, and Canny variants but unfortunately not remarkably better than the LBP as a fourth input channel, Figure 6.2 shows that table lamp is almost completely visible in the disparity image of LBP as a fourth channel but it is not the case in LBP decoder variant. In addition, the contours of the objects are rather clear in the LBP than



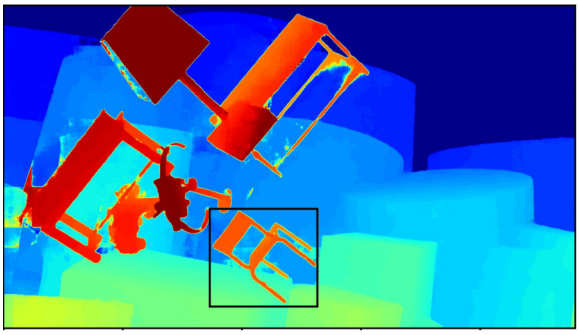
(a) Ground truth disparity image



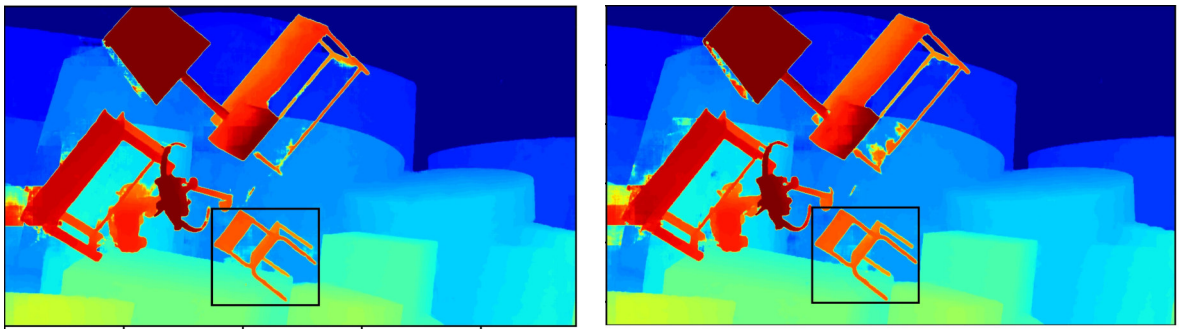
(b) Original disparity image



(c) LBP disparity image



(d) HOG disparity image



(e) Canny disparity image (f) LBP (decoder) disparity image

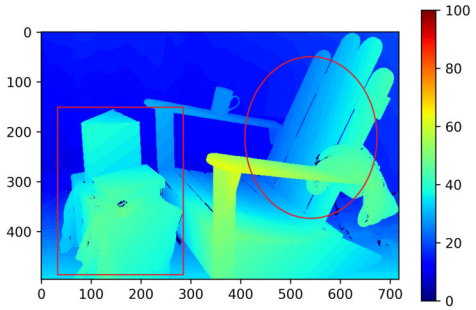
Figure 6.2: Qualitative result of all variants as well as ground truth and original disparity image on FlyingThings3D dataset. One of the major problematic areas is marked with black rectangle. Colorcoding is mentioned in Figure 6.1.

other variants see black rectangle marked in Figure 6.2. This can also be proven, see Table 6.1. The reason can be described as the network has already utilized the features when LBP based image is inserted into the encoder part of the network as an additional channel.

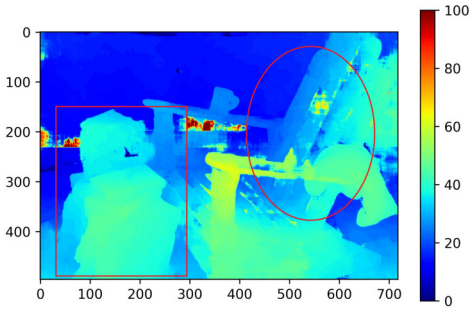
As stated in Section 4.1 i.e. depth discontinuities are identified as a weakness of the original GC-Net and qualitative and quantitative results prove that the adapted network architecture is appropriate and our approaches are capable of handling challenging scenarios, such as reflective, thin or texture-less surfaces, and occlusion. In addition, the evaluation results of strategies used indicate that by simply adding the fourth input channel can already provide sufficient information for detecting incorrect estimates and help a lot to improve the overall performance.

6.2 Qualitative and Quantitative Results based on the Middlebury Dataset

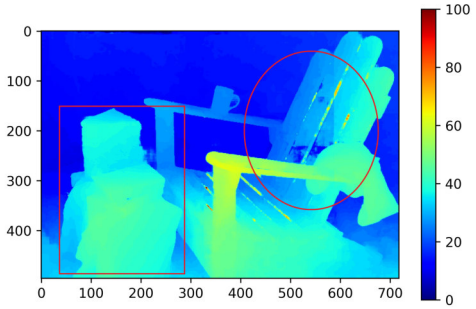
The generalization ability of the methods is verified on the Middlebury (Scharstein et al., 2014). As mentioned earlier 5.3, it is mainly carried out to see the domain difference. Most important thing to be noticed that all designed methods work noticeably better than the original variant. Figure 6.3 shows that small objects (shown in red rectangle) in front of the chair have rather clear boundaries in the disparity image of LBP as the fourth channel and LBP decoder. HOG and Canny's performance is better than original variant but worse than LBP as a fourth input channel and in the decoder part. In addition, chair splats (shown in red circle) are clear in disparity image of LBP and LBP decoder than other variants. On the contrary, it can be seen that the LBP decoder disparity image looks remarkably better than other variants. This also makes an important contribution to the second designed approach that hand-crafted features used in the decoder part still make difference. As previously the reason for LBP's best performance can be described as its robustness to illumination



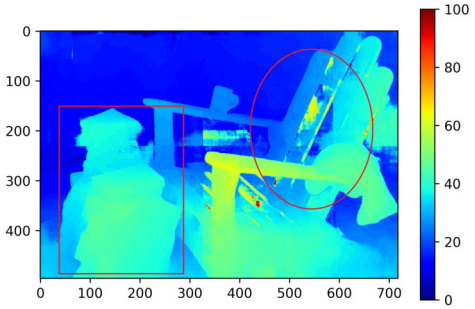
(a) Ground truth disparity image



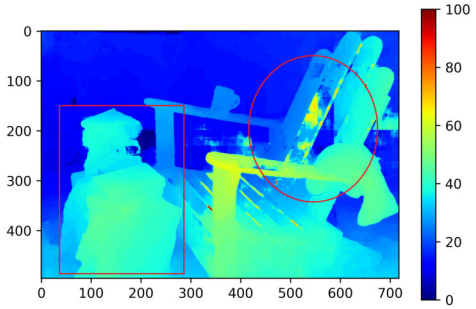
(b) Original disparity image



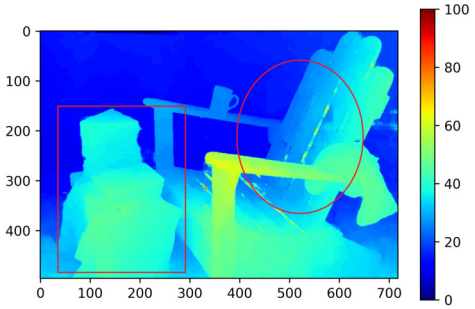
(c) LBP disparity image



(d) HOG disparity image



(e) Canny disparity image



(f) LBP (decoder) disparity image

Figure 6.3: Qualitative result of all variants as well as ground truth and original disparity image on Middlebury dataset. Major problematic areas are shown in red rectangle and circle comparing ground truth disparity image. Colorcoding is mentioned in 6.1.

Table 6.2: Evaluation results of all variants on Middlebury dataset.

	Original	**Canny**	**HOG**	**LBP**	**LBP (decoder)**
Pixel error 1	53.98 %	34.61 %	35.73 %	31.84 %	30.35 %
Pixel error 3	41.82 %	19.85 %	19.41 %	17.40 %	15.80 %
Pixel error 5	37.96 %	15.15 %	14.09 %	12.87 %	11.44 %
MAE [pixel]	13.577	4.046	3.747	4.418	3.461
RMSE [pixel]	18.040	9.280	8.596	10.369	8.535

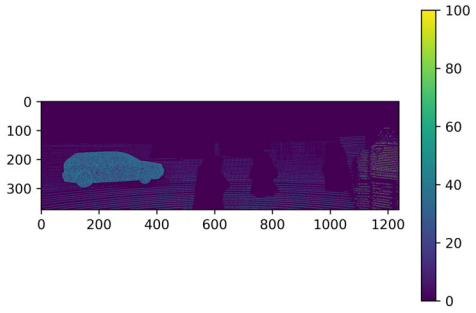
variations. Table 6.2 shows that the MAE is reduced by 70.20 %, 72.40 %, 67.45 %, and 74.50 % with the integration of Canny, HOG, LBP, and LBP (decoder) respectively relative to original variant on the Middlebury dataset. RMSE values showed a similar response i.e. dropped by 48.55 %, 52.35 %, 42.52 %, and 52.68 % respectively as compared to original variant. In conclusion, that proves that our designed methods work significantly better and handle small structures and object boundaries more profoundly.

6.3 Qualitative and Quantitative Results based on the KITTI Datasets

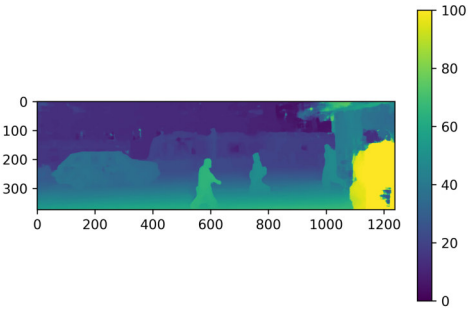
The network is fine-tuned on KITTI datasets. From the results big difference cannot be seen and it is rather different from the Middlebury dataset and FlyingThings3D dataset. It shows worse results, the reason could be twofold, firstly training is done on two separate datasets, i.e. FlyingThings3D and KITTI 2012 and 2015 so the results are worse. In addition, as described in Section 5.1 KITTI datasets have pixel-level accuracy. On the other hand, datasets contain outdoor environments, which is still challenging for dense stereo matching i.e. motion blur and no clear object boundaries. Therefore, it could not be noticeably improved but as expected LBP is the winner but HOG performed much worse, this is clearly seen in the Table 6.3.

Table 6.3: Evaluation results of all variants on KITTI datasets.

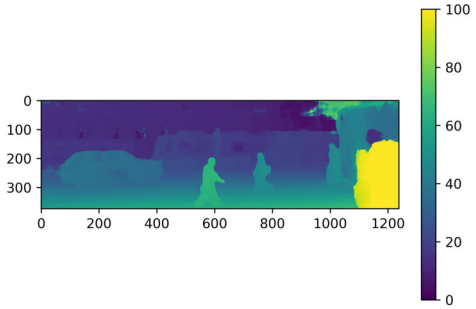
	Original	**Canny**	**HOG**	**LBP**	**LBP (decoder)**
Pixel error 1	19.33 %	20.07 %	21.42 %	19.91 %	19.37 %
Pixel error 3	4.65 %	4.50 %	5.00 %	4.37 %	4.52 %
Pixel error 5	2.61 %	2.47 %	2.68 %	2.34 %	2.44 %
MAE [pixel]	0.941	0.944	0.997	0.926	0.916
RMSE [pixel]	2.631	2.574	2.780	2.558	2.494



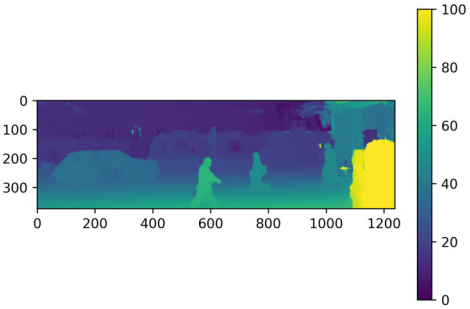
(a) Ground truth disparity image



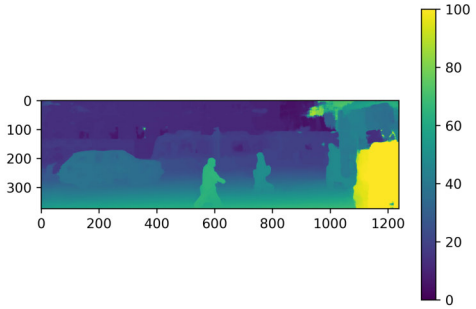
(b) Original disparity image



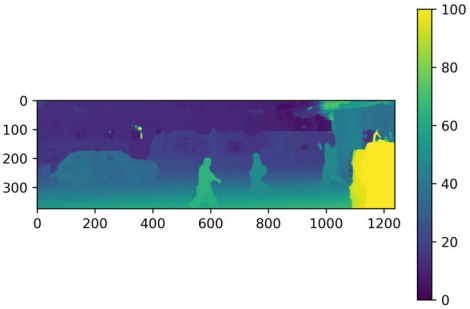
(c) LBP disparity image



(d) HOG disparity image



(e) Canny disparity image



(f) LBP (decoder) disparity image

Figure 6.4: Qualitative result of all variants as well as ground truth and original disparity image on KITTI dataset. colorcoding is mentioned in Figure 6.1.

7 Conclusions and Outlook

7.1 Conclusions

In this thesis, a Convolutional Neural Network (CNN) is designed and trained to incorporate geometric information to improve the performance of CNN for dense disparity estimates. GC-Net (Kendall et al., 2017) is used as the basic network architecture, which is an encoder-decoder architecture based on the concept of geometry and context learning. The key idea of this work is to put the focus of the network on learning disparity by leveraging the geometric knowledge. Primarily, two different approaches using three different image transformations as hand-crafted features are investigated, these hand-crafted features are used to guide the network for better disparity estimates. Three designed methods are used as a fourth input channel, while one is also used in the decoder part of the network.

The network and the designed methods are evaluated and tested on the FlyingThings3D dataset. As expected some methods performed better than others but all of them performed remarkably better than the original RGB images i.e. MAE is reduced by 24.02 %, 23.53 %, 35.26 %, and 30.80 % with the integration of Canny, HOG, LBP, and LBP (decoder) respectively. LBP is found best among them. From the results of the experiments, it can be concluded that our network succeeds in reconstructing an improved dense depth map with the usage of geometric information. Especially in the case of LBP as a fourth input channel and in the decoder part, both MAE and RMSE are decreased and all three error-pixels are also dropped. Depth discontinuities, especially thin object structures, object boundaries, and untextured areas are improved. Besides, our network does not increase the computational complexity and the memory footprint.

The Middlebury dataset is also used to evaluate our approaches on an unseen dataset. The same response is seen from all four methods and once again LBP is the winner. KITTI 2012 and KITTI 2015 datasets are used to fine-tune and test the network. They work very well but due to the use of two separate training datasets, and lower accuracy of KITTI datasets i.e. pixel-level the result is not as good as the result of FlyingThings3D alone. LBP is the best performer among other descriptors.

7.2 Outlook

For future work, we are interested in exploring a more explicit representation of semantics to improve the disparity estimation. Additionally, one possible idea is to improve the disparity estimates with

Bayesian convolutional neural networks.

Furthermore, the errors contained in the disparity map are significantly reduced. That means the performance of the designed methods works reasonably well. Therefore, for future work, we could focus on feature fusion of two methods e.g. HOG, Canny, or LBP, and joint train all features in a convolutional neural network. HOG and Canny give us better contour information than LBP but LBP performs overall well but sometimes its performance is slightly lower than HOG see 6.2. Another possible direction could be planning to find which part of features is redundant, and how to reduce the influence of it. Moreover, as it has been proven that gradient information is quite helpful and we can also consider adding this information in the loss function as it is done in Kang et al. (2019).

In addition, currently, our approach is limited to terrestrial images and it would also be interesting to know the performance of our designed network on airborne images.

Bibliography

Aber, J., Marzolff, I., Ries, J. and Aber, S., 2019. Principles of photogrammetry. pp. 19–38.

Abseher, M., Musliu, N. and Woltran, S., 2017. Improving the efficiency of dynamic programming on tree decompositions via machine learning. *Journal of Artificial Intelligence Research* 58, pp. 829–858.

Alhindi, T., Kalra, S., Ng, K., Afrin, A. and Tizhoosh, H., 2018. Comparing lbp, hog and deep features for classification of histopathology images.

Alldieck, T., Kassubeck, M., Wandt, B., Rosenhahn, B. and Magnor, M., 2017. Optical flow-based 3d human motion estimation from monocular video. In: V. Roth and T. Vetter (eds), *Proc. German Conference on Pattern Recognition (GCPR)*, pp. 347–360.

Badrinarayanan, V., Kendall, A. and Cipolla, R., 2017. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. Vol. 39Number 12, pp. 2481–2495.

Barnard, S. T. and Fischler, M. A., 1982. Computational stereo. In: *ACM Computing Survey*, pp. 553–572.

Žbontar, J. and LeCun, Y., 2016. Stereo matching by training a convolutional neural network to compare image patches. In: *The Journal of Machine Learning Research*, pp. 1–32.

Canny, J., 1986. A computational approach to edge detection. *IEEE transactions on pattern analysis and machine intelligence* PAMI-8(6), pp. 679–698.

Dalal, N. and Triggs, B., 2005. Histograms of oriented gradients for human detection. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 1, pp. 886–893 vol. 1.

Deniz, O., Bueno, G., Salido, J. and De la Torre, F., 2011. Face recognition using histograms of oriented gradients. Vol. 32, pp. 1598–1603.

Ding, W., Zou, H., Xie, S. and Gong, Z., 2006. A pan-tilt camera control system of uav visual tracking based on biomimetic eye. In: *IEEE International Conference on Robotics and Biomimetics - ROBIO2006*, IEEE Computer Society, Los Alamitos, CA, USA, pp. 1477–1482.

Eigen, D., Puhrsch, C. and Fergus, R., 2014. Depth map prediction from a single image using a multi-scale deep network. In: *Advances in Neural Information Processing Systems 27*, Curran Associates, Inc., pp. 2366–2374.

García-Olalla, O., Alegre, E., Fernández-Robles, L., García-Ordás, M. T. and García-Ordás, D., 2013. Adaptive local binary pattern with oriented standard deviation (albps) for texture classification. Vol. 2013Number 1.

Geiger, A., Lenz, P. and Urtasun, R., 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, pp. 3354–3361.

Gurrieri, L. E. and Dubois, E., 2014. Depth consistency and vertical disparities in stereoscopic panoramas. Vol. 23Number 1, SPIE, pp. 1 – 15.

Hamzah, R., Abd Rahim, R. and Noh, Z. M., 2010. Sum of absolute differences algorithm in stereo correspondence problem for stereo matching in computer vision application.

Hemanth, D. J., Gupta, D. and Balas, V. E., 2019. Intelligent data analysis for biomedical applications: Challenges and solutions / edited by jude hemanth, deepak gupta, valentina emilia balas. Intelligent data centric systems, Academic Press, London.

Heo, Y. S., Lee, K. M. and Lee, S. U., 2011. Robust stereo matching using adaptive normalized cross-correlation. *IEEE transactions on pattern analysis and machine intelligence* 33(4), pp. 807–822.

Hirschmüller, H., 2006. Stereo vision in structured environments by consistent semi-global matching. In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, IEEE Computer Society, Los Alamitos, CA, USA, pp. 2386–2393.

Hirschmüller, H., Innocent, P. R. and Garibaldi, J., 2002. Real-time correlation-based stereo vision with reduced border errors. Vol. 47, Kluwer Academic Publishers, pp. 229–246.

Hirschmuller, H. and Scharstein, D., 2007. Evaluation of cost functions for stereo matching. In: *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8.

Histogram of Oriented Gradients, 2018.

Häne, C., Zach, C., Lim, J., Ranganathan, A. and Pollefeys, M., 2011. Stereo depth map fusion for robot navigation. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1618–1625.

Hu, H., Chen, C., Wu, B., Yang, X., Zhu, Q. and Ding, Y., 2016. Texture-aware dense image matching using ternary census transform. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* III-3, pp. 59–66.

Ilg, E., Saikia, T., Keuper, M. and Brox, T., 2018. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In: V. Ferrari, M. Hebert, C. Sminchisescu and Y. Weiss (eds), *Computer Vision - ECCV 2018*, Springer International Publishing, Cham, pp. 626–643.

Jung, I. and Oh, I., 2010. Local binary pattern-based features for text identification of web images. pp. 4320–4323.

Kamencay, P., Breznan, M., Jarina, R., Lukac, P. and Radilova, M., 2012. Improved depth map estimation from stereo images based on hybrid method. *Radioengineering*.

Kang, J., Chen, L., Deng, F. and Heipke, C., 2019. Context pyramidal network for stereo matching regularized by disparity gradients. In: *ISPRS Journal of Photogrammetry and Remote Sensing*, pp. 201–215.

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

Klaus, A., Sormann, M. and Karner, K., 2006. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In: *18th International Conference on Pattern Recognition (ICPR'06)*, Vol. 3, pp. 15–18.

Kolmogorov, V. and Zabih, R., 2001. Computing visual correspondence with occlusions using graph cuts. In: *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, pp. 508–515 vol.2.

Krutikova, O., Sisojevs, A. and Kovalovs, M., 2017. Creation of a depth map from stereo images of faces for 3d model reconstruction. Vol. 104, pp. 452–459.

Ladicky, L., Häne, C. and Pollefeys, M., 2015. Learning the matching function. Vol. abs/1502.00652.

Lowe, D. G., 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60(2), pp. 91–110.

Lu, C., Uchiyama, H., Thomas, D., Shimada, A. and Taniguchi, R. I., 2018. Sparse cost volume for efficient stereo matching. Vol. 10Number 11, Multidisciplinary Digital Publishing Institute (MDPI).

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

Mao, J. and Hu, Y., 2019. Obstacle contour extraction method based on improved grabcut algorithm. *Journal of Physics: Conference Series* 1303, pp. 012051.

Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A. and Brox, T., 2015. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:1512.02134.

Menze, M. and Geiger, A., 2015. Object scene flow for autonomous vehicles. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3061–3070.

Mustaniemi, J., Kannala, J. and Heikkilä, J., 2015. Disparity estimation for image fusion in a multi-aperture camera. In: G. Azzopardi and N. Petkov (eds), *Computer Analysis of Images and Patterns*, Springer International Publishing, Cham, pp. 158–170.

Ojala, T., Pietikainen, M. and Harwood, D., 1994. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. Vol. 1, pp. 582–585 vol.1.

O'Mahony, N., Campbell, S., Carvalho, A., Harapanahalli, S., Hernandez, G., Krpalkova, L., Riordan, D. and Walsh, J., 2020. Deep Learning vs. Traditional Computer Vision. Vol. 943, pp. 128–144.

Podareanu, D., Codreanu, V., Aigner, S., Leeuwen, C. and Weinberg, V., 2019. Best practice guide - deep learning.

Ren, S., He, K., B. Girshick, R. and Sun, J., 2015. Faster R-CNN: towards real-time object detection with region proposal networks. Vol. abs/1506.01497.

Scharstein, D. and Szeliski, R., 2002. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. In: *International Journal of Computer Vision*, pp. 7–42.

Scharstein, D. and Szeliski, R., 2003. High-accuracy stereo depth maps using structured light. In: *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, Vol. 1, pp. I–I.

Scharstein, D., Hirschmüller, H., Kitajima, Y., Krathwohl, G., Nešić, N., Wang, X. and Westling, P., 2014. High-resolution stereo datasets with subpixel-accurate ground truth. Vol. 8753, pp. 31–42.

Shaked, A. and Wolf, L., 2016. Improved stereo matching with constant highway networks and reflective confidence learning.

Sofiana Sahir, 2019. Canny edge detection step by step in python — computer vision.

Stucker, C. and Schindler, K., 2020. Resdepth: Learned residual stereo reconstruction. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*.

Tianyu, Z., Zhenjiang, M. and Jianhu, Z., 2018. Combining cnn with hand-crafted features for image classification. In: *2018 14th IEEE International Conference on Signal Processing (ICSP)*, pp. 554–557.

Yoon, K.-J. and Kweon, I. S., 2006. Adaptive support-weight approach for correspondence search. Vol. 28Number 4, pp. 650–656.

Zbontar, J. and LeCun, Y., 2015. Computing the stereo matching cost with a convolutional neural network. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Zhang, L. and Seitz, S., 2007. stimating optimal parameters for MRF stereo from a single image pair. In: *IEEE Trans Pattern Anal Mach Intell*, pp. 331–342.

Zhang, X. C., Lee, J.-Y., Sunkavalli, K. and Wang, Z., 2017. Photometric stabilization for fast-forward videos.

Zhu, J., Wang, L., Yang, R., Davis, J. and Pan, Z., 2011. Reliability fusion of time-of-flight depth and stereo geometry for high quality depth maps. In: *IEEE Trans Pattern Anal Mach Intell. 2011*, Vol. 1, pp. 1400–14.