Computer Vision Challenge

Jens Ernstberger
Kevin Rexhepi
Achraf Friaa
Lida Kostis
Ghassen Laajailia

Research paper

Computer Vision Challenge

Jens Ernstberger
Kevin Rexhepi
Achraf Friaa
Lida Kostis
Ghassen Laajailia

July 19, 2019

Chair for Data Processing
Technische Universität München

Jens Ernstberger
Kevin Rexhepi
Achraf Friaa
Lida Kostis
Ghassen Laajailia . *Computer Vision Challenge.* Research paper, Technische Universität München, Munich, Germany, 2019.

Supervised by Prof. Dr.-Ing. Klaus Diepold ; submitted on July 19, 2019 to the Department of Electrical and Computer Engineering of the Technische Universität München.

Chair for Data Processing, Technische Universität München, 80290 München, Germany, `http://www.ldv.ei.tum.de/`.

Abstract

The Computer Vision Challenge 2019 is a group project of 3 to 5 students to the corresponding lecture 'Computer Vision'. The goal is to create a Disparity Map from a pair of stereo images. The dataset used was the Middlebury dataset [SHK+14]. It provides four pairs of stereo images with their ground truth and the corresponding calibration matrices. The images are rectified, for the sake of completeness the concept of rectification will be explained anyways. One of our goals is to have a fast execution time for the disparity algorithm. Evaluation is done by examining the Peak Signal To Noise Ratio (PSNR) and the execution time.

Contents

1 Fundamentals

1.1 Stereo Image Rectification

Stereo image rectification determines a transformation of a pair of images plane such that the conjugate epipolar lines become collinear and parallel to one of the image axes. The rectified images can be thought of as acquired by a new stereo rig, obtained by rotating the original cameras around the optical centre. The important advantage of rectification is that computing correspondences is reduced to a 1-D search problem, typically along the horizontal lines of the rectified images.

1.2 Block Matching

Block matching is the most common algorithm to find corresponding points in stereo images. This algorithm is used in the project to find corresponding points between the left and right images of the provided image pairs. The block matching algorithm is used to minimize the matching errors between the blocks at any position of the left image with regard to the right image. To find the most similar block we need to check all possible blocks in the right image which has the same row number but different columns. To measure similarity between reference block and the checked block, sum of squared differences (SSD) is used.

1.3 Dynamic Programming

The accuracy of the disparity map can be increased by taking into account the neighbor pixels disparity. This is implemented using dynamic programming. This algorithms increases the cost of the match whose matching block is far away from expected position, and therefore minimizes the problematic side of simple block matching, which is the lack of spacial consistency.

1.4 Image scaling

A critical point when looking at the performance of disparity map algorithms is the execution time. This project uses the 'imresize' function from MATLAB to resize the input images for faster computation. Figure 1.1 shows the principal functionality of image rescaling.

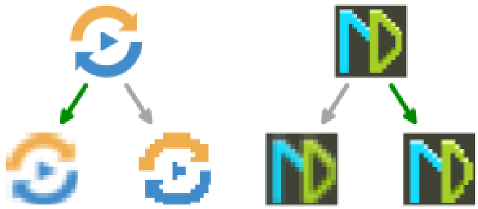


Figure 1.1: Image Scaling

In order to compare the resulting Disparity Map to the Ground Truth, a upscaling after the Disparity Map computation is necessary. The algorithm used to upsample in our Code is called Nearest Neighbour Interpolation [Mat14] and is a very simple approach towards interpolation. Rather than calculate an average value by some weighting criteria or generate an intermediate value based on complicated rules, this method simply determines the nearest neighbouring pixel, and assumes the intensity value of it. This might result in a bad Peak Signal To Noise Ratio, but will definitely decrease the computational complexity compared to other algorithms

2 Implementation

2.1 Overview

In the following the implementation we chose for our project will be explained in more detail. Chapter 2.2 explains how the rotation and translation is calculated from the stereo image pairs correspondences and the camera specific parameters. Chapter 2.3 is going to go into more detail with the implementation of our Block Matching algorithm. Chapter 2.4 will focus on the usability of our algorithm with the Graphical User Interface (GUI).

2.2 Rotation & Translation with Correspondences

In order to extract matching points between stereo images, one needs first to determine the Harris Corners. In this context, Harris Corner Detection is an algorithm introduced by Chris Harris and Mike Stephens [HS$^+$88] in 1988 which is used to extract corners and edges from images.

The corners detected in the images has to be compared using Normalized Cross Correlation which compares two points and determines if they match or not. To improve and select the robust points one need to use RANSAC Algorithm [Can81] which selects the matching points that have high rates. The description below resumes the stated process.

1. Convert image to Grayscale (if it is colored)
2. Calculate Harris features
3. Find correspondences & filter for stable ones with RANSAC
4. Calculate essential matrix by using the camera parameters & robust correspondences
5. Calculate Rotation & Translation by using the essential matrix

The resulting rotation & translation show how the positon of the two cameras differs in 3D space and is one of the outputs of the *challenge.m* function.

2.3 Disparity Algorithm

2.3.1 Preprocessing

Before the block matching algorithm can be started, the images are pre-processed. The first step is to check whether the images to be checked are color images. If this is the case with the stereo image pair, these must first be converted into gray images using the following formula:

$$Gray = 0.299 \cdot Red + 0.587 \cdot Green + 0.114 \cdot Blue$$

The last preprocessing step is to resize the images. The calculation effort of the algorithm is proportional to the size of the image. For this reason, the images are reduced by a certain factor before the main part of the algorithm begins.

The factor itself again depends on the image size. The basic principle of how the image is resized is described in section 1.4. We chose to implement the image downsizing in the following substeps:

1. # Rows in image < 500: Size down to 50%
2. 500 < # Rows in image < 500: Size down to 33%
3. 1000 < # Rows in image: Size down to 25%

The values for the factor for the different image sizes were determined empirically. This method considerably reduces the calculation effort, but also reduces the accuracy of the calculation, which leads to worse results for the disparity map.

Rectification Another necessary pre-processing step for the creation of a disparity map is the rectification of the stereo image pair. Rectification is a transformation in which the images are projected onto a common image plane. It is used to simplify the problem of searching for corresponding points between two images. In rectified images, the coordinates of any two point correspondences differ only in the x-coordinate, i.e. pixel column. This step was omitted because all stereo image pairs are already rectified.

2.3.2 Blockmatching

By first calculating the distance in pixels between the position of one feature in one image and its position in the other image, depth information can be calculated from a pair of stereo images. Objects closer to the camera have a larger pixel pitch than those further away from the camera. Based on the assumption that all images used are already rectified, the correspondence point search can be restricted to the respective pixel line. Block matching was used for the correspondence search. A small area of pixels in the left image is viewed. The goal is to find the same area in the right image. Fig. 2.1 shows how the algorithm works. How good the algorithm works, depends on the two parameters “block size" and "disparity range". "block size" is the size of the area of pixels, which is viewed. The "disparity range" decides the greatest possible value of the disparity. When looking for the corresponding box in the other image, the start is at the same coordinates as the template (white box) and search to the left and right up to a maximum, which is the "disparity range" [McC14]. The search is reduced to just one direction, to the left, to save computational costs. This is only possible, because it is already clear which image is taken by which camera.

Figure 2.1: Extract of the code: Image Scaling

The similarity metric for finding the corresponding matching block is “sum of squared differences” or “SSD” [FK15]. The formula is given below:

$$SSD = \sum_{i=1}^{N}\sum_{j=1}^{N}(I_l(i,j) - I_r(i,j))^2$$

To find the corresponding block, the SSD values between the template and each block in the search range are calculated. The block with the lowest SSD value is chosen as the corresponding block. Different values for the parameters were tested in the experiment. More details can be found in the chapter “Results”.

2.3.3 Interpolation/Subpixelaccuracy

The disparity values calculated with block matching are all integers, since they correspond to the pixel shift. To obtain more accurate results, the disparity value is estimated by including the neighboring blocks. Figure 2.5 shows an illustration of interpolation:

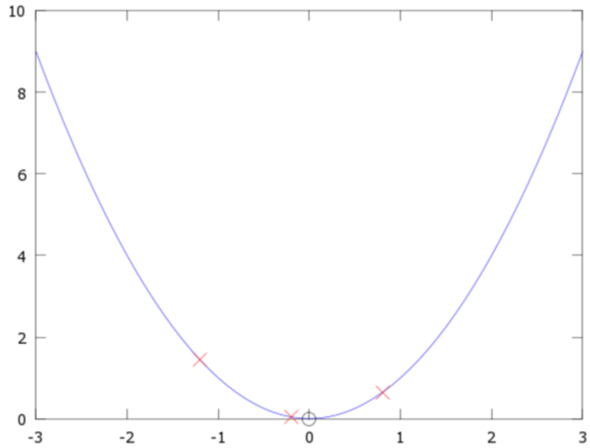


Figure 2.2: Overview of Interpolation

The illustration shows three points marked by red crosses. The points are evenly distributed in the x-direction and all lie on the same parabola. The middle cross represents the best block with the lowest SSD value. The other crosses represent the neighbors. It can be seen that the SSD value is slightly above zero and therefore not in the minimum of the parabola. Using the following equation the location of the parabola minimum can be determined, with $SSD_{1/3}$ as SSD value of the left/right neighbor and SSD_2 as SSD value of the corresponding block. d_{old} corresponds to the integer value of the pixel distance, while d_{int} represents the interpolated disparity value:

$$d_{int} = d_{old} - 0.5 \cdot \frac{SSD_3 - SSD_1}{SSD_1 - 2SSD_2 + SSD_3}$$

2.4 Graphical User Interface

The Graphical User Interface displays three figures, a table, and has two buttons. The first button opens a path selector where you can choose the folder where the stereo pairs are saved. The images pairs are then displayed in the GUI (Left and right image). The second button displays the disparity map and a table with the values of rotation (matrix) and translation (vector).

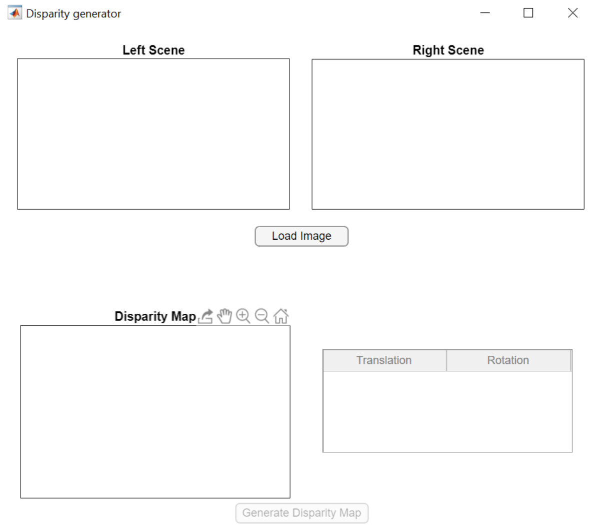


Figure 2.3: Graphical User interface

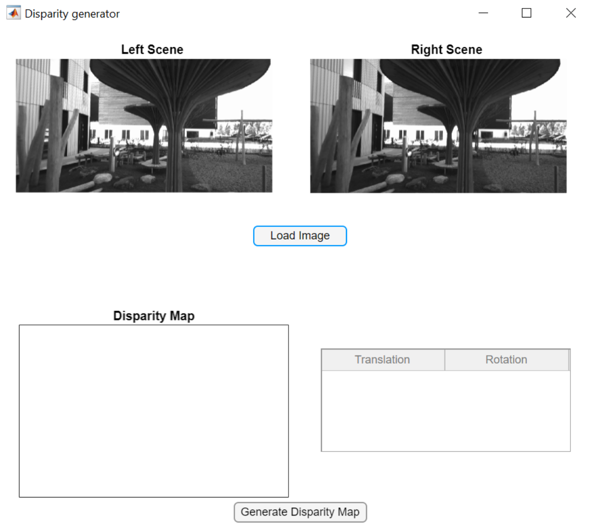


Figure 2.4: Displaying the image pairs after pushing the first button

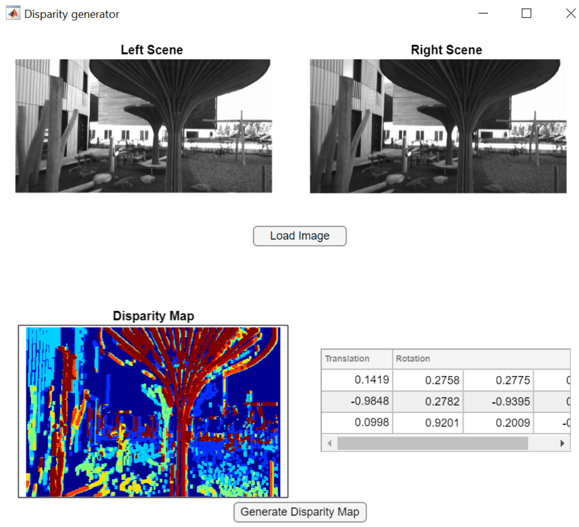


Figure 2.5: Displaying the disparity map and table data after pushing the second button

3 Results

In Order to compute how well our algorithm performs, we compared the disparity map with the ground truth disparity map provided by the "Middlebury Stereo Datasets". The ground truth of the disparity matrix is given in the format "pfm". To open this type of file a script was used, which is part of the software development kit for the evaluation v.3 [Sch14]. As comparison measure size, the Peak signal-to-noise ratio (dB) is used.

In the following we will display and discuss the results we found by completing test runs using different parameters for the Disparity range and Block size for the block matching algorithm. In Figure 3.1 we plot the disparity map for different block sizes for the block matching algorithm and constant Disparity map range.

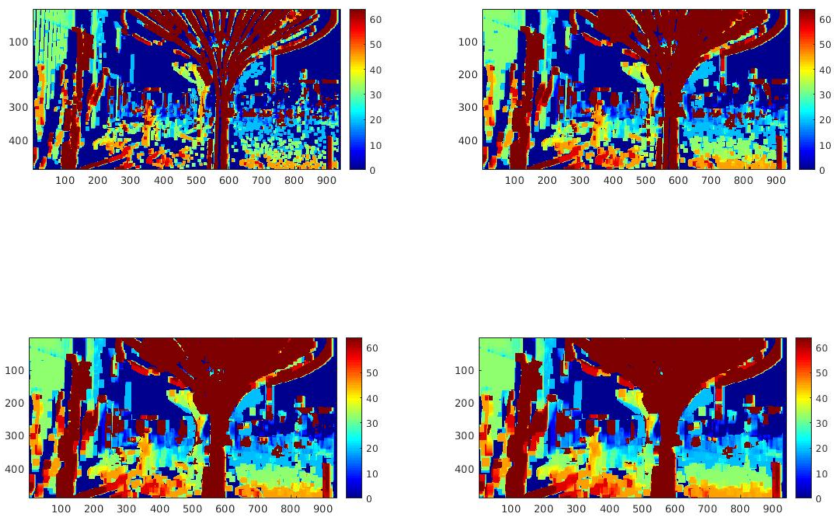


Figure 3.1: Disparity map for image playground. The chosen range for the disparity map is 20 and the block size is varying from 2 to 5 respectively.

In Figure 3.1 we notice that increasing the block size decreases the granularity of the color of the image.

After trying out different values for the parameters we the values that give the best Peak signal-to-noise ratio. Figures 3.2 to 3.5 show the disparity matrices that we computed and the correspond Ground Truth disparity matrix and table 3.1 shows the an overview of the parameters chosen for each image and the corresponding PSNR and excursion time.

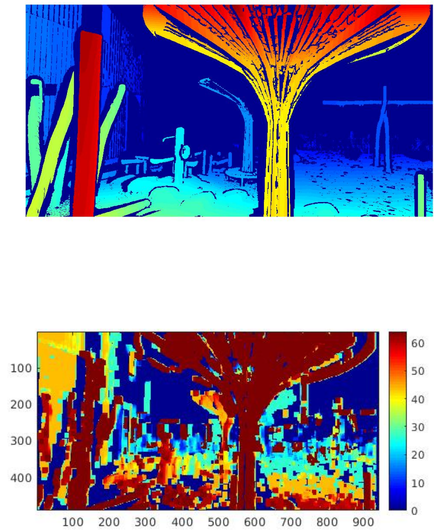


Figure 3.2: Playground: Disparity range: 15, Block size: 3

Bild	**Image size**	**Disp. range/Block size**	**PSNR**	**Time**
Motorcycle	2964x1988	250/5	5.9924	5.9924
Playground	941x490	15/3	27.5185	5.9924
Sword	2884x1956	308/7	24.5183	275.5578
Terrace	713x434	50/3	27.2458	11.5018

Table 3.1: Results

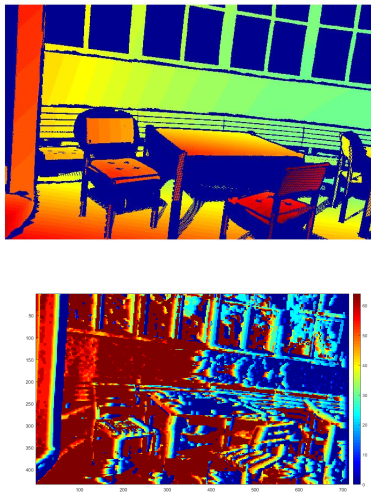


Figure 3.3: Terrace: Disparity rangee: 50, Block size: 3

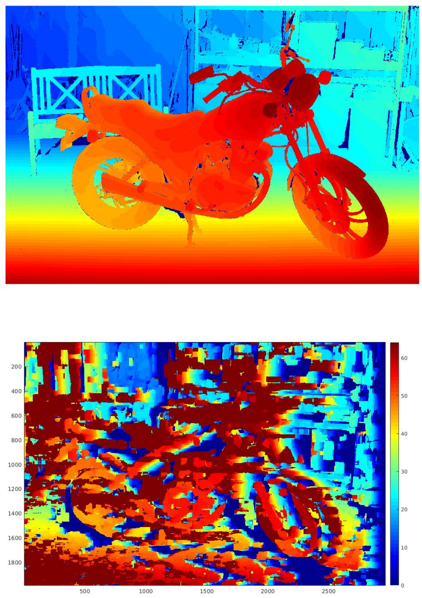
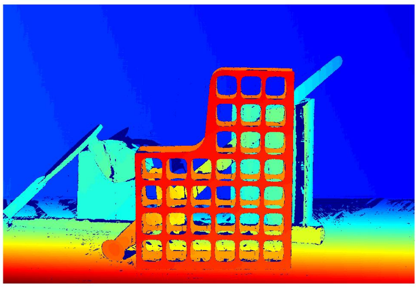


Figure 3.4: Terrace: Disparity range: 50, Block size: 3



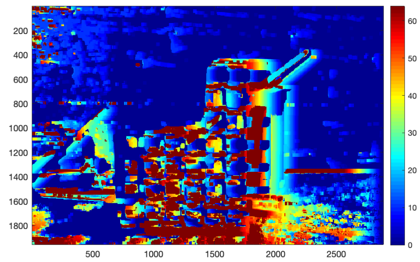


Figure 3.5: Sword: Disparity range: 308, Block size: 7

4 Conclusion

This work describes the algorithmic approach followed to obtain a Disparity Map from a pair of two image views. The implementation was based on computer vision techniques. The goal was to implement an algorithm with an acceptable computation time while achieving a satisfiable disparity estimation. Block matching did not deliver the anticipated results on the stereo pair image dataset chosen. An improvement in accuracy might be achieved by choosing another similarity metric, for example Normalized Cross Correlation.
The difference in PSNR between the images is highly differing due to image size and image specific characteristics. A PSNR of 27,2 was achieved for one of the images, which can be pointed out as satisfiable for a simple block matching algorithm. Some improvements are still necessary in order to optimize the processing speed and increase the quality of estimation. Nevertheless, a satisfiable computation time was achieved by choosing a simple algorithm and downsampling of the images.

Bibliography

Can81. H Cantzler. Random sample consensus (ransac). *Institute for Perception, Action and Behaviour, Division of Informatics, University of Edinburgh*, 1981.

FK15. YM Fouda and Abdul Raouf Khan. Normalize cross correlation algorithm in pattern matching based on 1-d information vector. *Trends Appl. Sci. Res*, 10(4):195–206, 2015.

HS+88. Christopher G Harris, Mike Stephens, et al. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer, 1988.

Mat14. MathWorks. Nearest neighbor, bilinear, and bicubic interpolation methods, 2014.

McC14. Chris McCormick. Stereo vison tutorial-part 1, 2014.

Sch14. Daniel Scharstein. pfmread, 2014.

SHK+14. Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German conference on pattern recognition*, pages 31–42. Springer, 2014.