Computer Vision Challenge

Jens Ernstberger
Kevin Rexhepi

Research paper

Computer Vision Challenge

Jens Ernstberger
Kevin Rexhepi

July 18, 2019

Institute for Data Processing
Technische Universität München

Jens Ernstberger
Kevin Rexhepi. *Computer Vision Challenge.* Research paper, Technische Universität München, Munich, Germany, 2019.

Supervised by Prof. Dr.-Ing. K. Diepold ; submitted on July 18, 2019 to the Department of Electrical Engineering and Information Technology of the Technische Universität München.

Institute 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 [5]. 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

1.2 Block Matching

1.3 Dynamic Programming

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.

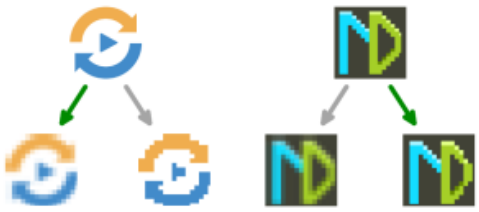

Figure 1.1: Image Scaling

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%

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 [3] 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 [2] 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 [**?**] 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

Sourcecode blockmatching: [4]
source ncc / ssd / sad [1]

2.4 Graphical User Interface

3 Results

Bild	Image size	Parameter	PSNR	Time
Motorcycle	0	0	0	0
Playground	0	0	0	0
Sword	0	0	0	0
Terrace	0	0	0	0

Table 3.1: Results

4 Conlusion

Bibliography

1. Y. Fouda and A.R. Khan. Normalize cross correlation algorithm in pattern matching based on 1-d information vector. In *Trends Appl. Sci. Res*, 10(4), pp. 195–206, 2015.

2. C.G. Harris, M. Stephens et al.. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pp. 10–5244. Citeseer, 1988.

3. MathWorks. Nearest neighbor, bilinear, and bicubic interpolation methods. 2014. URL `https://de.mathworks.com/help/vision/ug/interpolation-methods.html`.

4. C. McCormick. Stereo vison tutorial-part 1. 2014. URL `http://mccormickml.com/2014/01/10/stereo-vision-tutorial-part-i/`.

5. D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German conference on pattern recognition*, pp. 31–42. Springer, 2014.