Stereo Vision – Exercise

Dominik Schörkhuber dominik.schoerkhuber@tuwien.ac.at
Amelie Schäfer amelie.schaefer@tuwien.ac.at

Goal: The goal of the practical part of this course is the implementation of a local stereo matching algorithm based on the paper Hosni et al. "REAL-TIME LOCAL STEREO MATCHING USING GUIDED IMAGE FILTERING" [1]. The paper can be downloaded from the TUWEL course. The exercise is split into multiple parts; during the first task, you will implement a simple block-based stereo algorithm. In the second part, you will refine the previous results using guided filtering. After that, you will have to implement a post-processing procedure to enhance the quality of your final disparity map. For the final part of this exercise, we will calculate evaluation metrics to quantitatively assess the result.

Formalities: This exercise is part of the VU Stereo Vision (188.513). The LV consists of the lecture, and the corresponding written exam on the 26. June 2024 at 15:00, and this exercise which is due on the **12. June 2024** at **23:59**. Each of those is mandatory and needs to be passed, where the final grade is made up of 50% from the exam and 50% of this exercise.

Each of the four tasks within this exercise is mandatory and you must get at least 50% in each of them to get a passing grade. The exercises must be implemented in Python and Jupyter Notebooks. We recommend Python versions newer than Python 3.8. Your code should be well documented, readable and execute in a reasonable amount of time. We have prepared a Jupyter Notebook for all tasks in this exercise, with #TODO tags where code should be inserted. You may use any functions available in Numpy and OpenCV to solve the tasks.

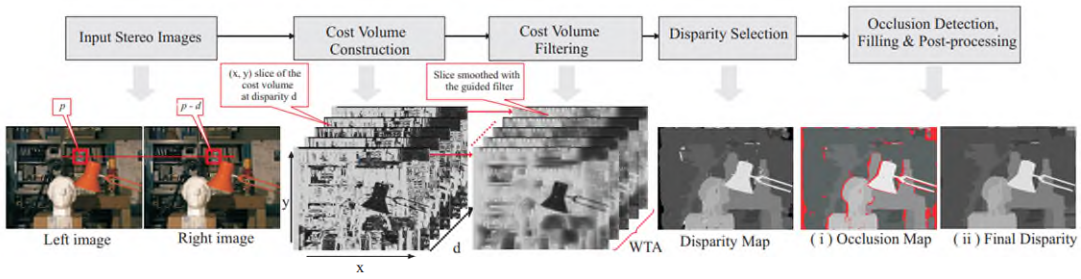
HINT: To separate installed Python packages between projects Conda[1] can be helpful. We suggest the use of Miniconda. You can find installation steps in the zip folder.

The exercise should be conducted in groups of two. Please register until the **18. April 2024** at **23:59** in TUWEL, you may find group members in the group finding forum. After the submission, there will be short submission interviews in the week of the 17. to 21. June 2024. The exact dates will be announced at the beginning of June.

Submission: The deadline for this exercise is the **12. June 2024** at **23:59** and the file must be uploaded via the TUWEL course. The submission must be a zip file containing your Jupyter Notebook and the notebook exported as an html file and the additional images from task 4.

Contact: If you have any questions about the exercise, please post them in the exercise forum in TUWEL and we will answer them in a timely manner.

[1] https://docs.anaconda.com/free/miniconda/index.html



Task 1 - 25 Points

Cost Volume Calculation

The goal of this task is to implement a function taking a pair of rectified stereo images L (left) and R (right) together with parameters for the maximum disparity, as well as color and gradient weighting, to calculate and visualize a disparity map.

The cost for a pixel p at a disparity d is calculated through the dissimilarity to its corresponding pixel in the second image. The following formula describes the cost calculation based on the absolute difference of colors.

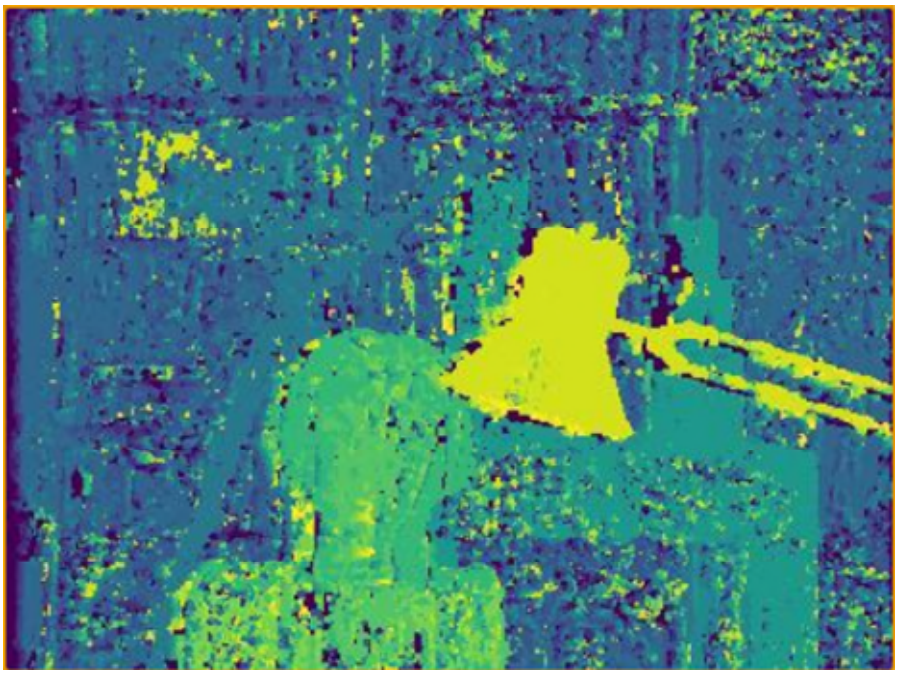
$$c(p,d) = \sum_{c\in\{r,g,b\}} |L_c(p) - R_c(p-d)|$$

The resulting cost volume should contain a cost matrix for each possible disparity, with the first element being the cost volume of disparity 0, the second of disparity 1 and so on. Implement the cost volume calculation according to Hosni et al. section 2.1. After constructing the cost volume, apply an average filter (which will be replaced in Task 2 by a guided filter) to each slice of the cost volume for better results. You can use suitable functions for this from Numpy or OpenCV. From the cost volume, we can select the disparity value d_p with the smallest cost for each pixel to create a disparity map.

$$d_p = \min_{\mathrm{d}\in\mathrm{D}} c(p,d)$$

- Read and display input images. (5 points)
- Compute cost volume with color cost and gradient cost terms. (10 points)
- Apply average filter and select best disparity values from cost volume. (5 points)
- Find suitable parameters and visualize disparity. (5 points)

Test your function with a maximum disparity of 15 and a window size of 5 for the average filter on the provided Tsukuba stereo image pair. Before you upload your exercise, try to find the best possible parameters. For visualization, you can use jupyter_compare_view.



Result for the Tsukuba example for Task1. Your results may vary depending on implementation details and selected parameters.

Task 2 – 25 Points

Guided Filtering

The goal of this task is to implement a function that filters the cost volumes (replacing the average filter from Task 1) before the disparity is calculated. To achieve this, we use a guided filter that modifies each value in the cost volume based on a weighted average of all pixels in the same slice. This new filter replaces any filters used in the previous task.

$$c'(p,d) = \sum_{q \in N_p} w_{p,q}(I) * c(p,d)$$

To simplify this task, we provide an implementation of the guided image filter in "guided_filter.py". The quality of your final disparity map heavily relies on the proper choice of parameters. By tweaking these parameters, you can influence the quality of your result, you should aim for achieving the best possible outcome. For your submission, make sure to have the best parameterization as the default setting.

- Implement and apply the guided image filter instead of the average filter of Task 1. (15 points)
- Select best parameters and visualize your results. (10 points)

Result for the Tsukuba example for Task2. Your results may vary depending on implementation details and selected parameters.

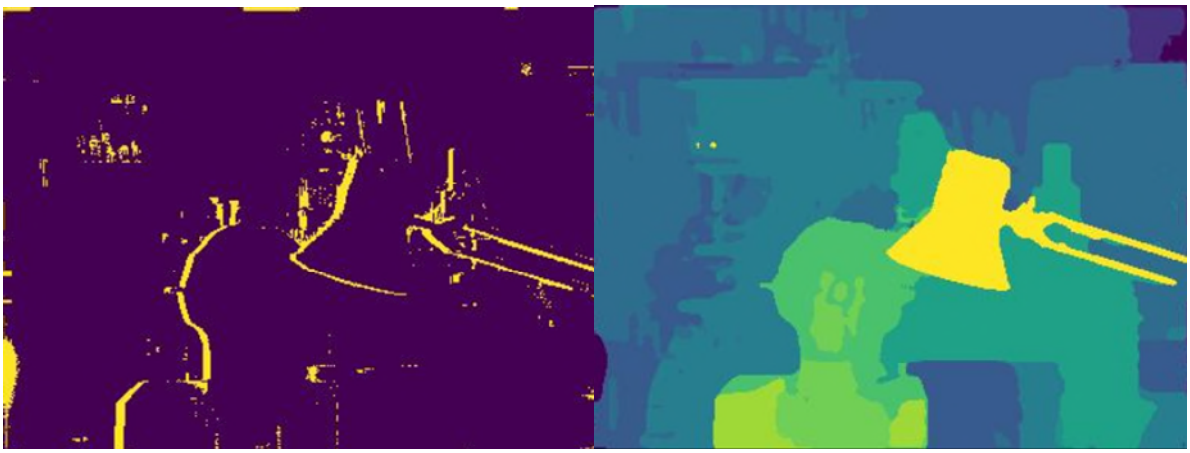
Task 3 - 25 Points

Post-Processing

The paper of Hosni et al. explains a procedure to increase the quality of the disparity map by applying an additional post-processing step after its calculation in section “2.4. Occlusion Detection, Filling and Post-processing”.

First, apply left-right consistency checking and mark each pixel in the left disparity map as inconsistent, if the disparity value of its matching pixel in the right disparity map differs by a value larger than one pixel. This consistency check typically fails for occluded and mismatched pixels. Next, fill all inconsistent pixels by the disparity of the closest consistent pixel. To this end, find the disparity of the closest consistent pixel to the left and right. The inconsistent pixel is then assigned the smaller disparity value of those two. For the median filter you can use a suitable function from Numpy or OpenCV.

- Compute disparity maps for the left and right reference image. (5 points)
- Mark inconsistent pixels. (5 points)
- Infill inconsistent pixels with nearby disparity values. (10 points)
- Apply median filter and visualize the result. (5 points)



Left: Inconsistent Pixels, Right: Disparity map with infilled inconsistent pixels.

Task 4 - 25 Points

Evaluation

There are different metrics to compare the correctness of an algorithm to the ground-truth information. Implement the mean difference and the accuracy of your created result with the ground-truth data. To see the progress and compare the results between the tasks, show the result from tasks 1 – 3 alongside the ground truth. Calculate the two metrics for each of the image stages. Repeat these steps with two other images from the Middlebury dataset (https://vision.middlebury.edu/stereo/data/). You might need to adjust some parameters like the maximum disparity, provide the used parameters for each image as well. Discuss the results including possible improvements and differences between the results.

- Compute the mean difference and the accuracy of the results. (5 Points)
- Compare the results from the different tasks, by showing them side-by-side additionally to the ground truth and by comparing the metrics applied to the results. (5 Points)
- Repeat those steps with two other images from the Middlebury dataset. (10 Points)
- Discuss the results. (5 Points)

[1] A. Hosni, M. Blezer, C. Rhemann, M. Gelautz und C. Rother, „REAL-TIME LOCAL STEREO MATCHING USING GUIDED IMAGE FILTERING,“ in *IEEE International Conference on Multimedia and Expo*, Barcelona, 2011, pp.1-6.