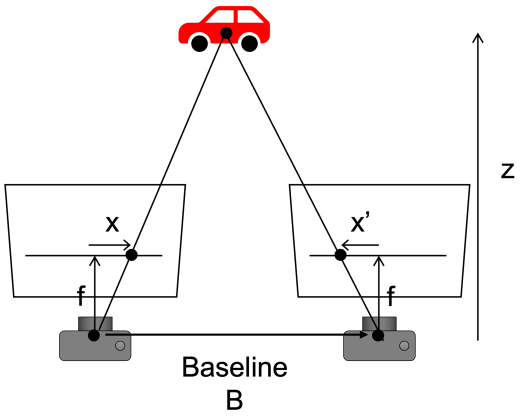
The University of Melbourne
School of Computing and Information Systems
COMP90086 Computer Vision, 2022 Semester 2

Final Project: Stereo Disparity

Project type: Group (teams of 2)
Due: 7pm, 22 Oct 2021
Submission: Source code and written report (as .pdf)
Marks: The assignment will be marked out of 30 points, and will contribute 30% of your total mark.

Introduction

Stereo matching is the problem of finding correspondences between two images that are taken simultaneously from two cameras that are mounted so that they are parallel and separated along their x-axis. The output of stereo matching is a disparity image that, for every pixel in the left image (x), indicates how many pixels to the left its correspondence (x') is in the right image, giving the disparity (x-x').



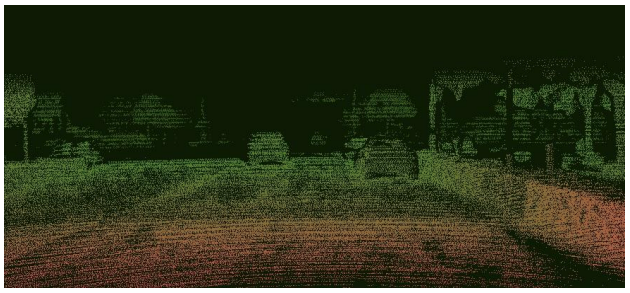
In this project, you will work with stereo image pairs taken from a moving vehicle, such as those shown below (with the right image on the left and vice versa so you can cross your eyes to see the stereo effect):

Tasks

Your tasks are:

1. To calculate a disparity map for the left image using classical (non deep learning) methods. You should create an image of floating point numbers that estimate the disparity (x-x') for every pixel in the left image.

2. To calculate some statistics using supplied ground truth depth images like the one below (crudely recoloured for this document to show the disparity as a heat map):



Specifically, you should compute:

a. The rms (root mean squared) error between the values in your disparity map and those in the ground truth
b. The fractions of pixels with errors less than 4, 2, 1, 0.5 and 0.25 pixels
c. The runtime of your algorithm in seconds per image

The ground truth file is a 16 bit png where the value it contains is 256 * disparity (so that it can represent sub-pixel precision). It does not have disparity values for every pixel, where the disparity is absent, the number zero is given. You should restrict your statistics calculations to pixels for which there are valid depths.

Approach and considerations

How you approach this problem is up to you - but you may wish to start with the methods discussed in lecture 13. You may also wish to consider:

- What matching function should you use? SSD and NCC were described in the lecture, can you develop others that will improve performance?
- How big should the window size be?
- Are all pixels in the matching window equally important?
- How are you going to get sub-pixel accuracy?
- Can you improve performance by encouraging a smooth output?
- How can you accelerate the process so that you don't have to loop over every pixel in python (perhaps by using tensor operations in numpy, pytorch or tensorflow)?

Note that these are only suggestions to help you get started; you are free to use your own ideas.

Whatever methods you choose, you are expected to evaluate these methods using the provided data, to critically analyse the results, and to justify your design choices in your final report. Your evaluation should include error analysis, where you attempt to understand where your method works well and where it fails.

You are encouraged to use existing computer vision libraries in your implementation. However, your method should be your own; you may not use library functions that solve the depth or disparity problem, even if you then build on top of these functions.

Dataset

The dataset is curated from https://drivingstereo-dataset.github.io

The images have been renamed so that they have the form:

<something>-left.jpg
<something>-right.jpg
<something>-disparity.png

The left and right images are 8 bit rgb images. The disparity images are 16 bit monochrome images.

Group Formation

You should complete this project in a group of 2. You are required to register your group membership on Canvas by completing the “Project Group Registration” survey under “Quizzes.” You may modify your group membership at any time up until the survey due date, but after the survey closes we will consider the group membership final.

Submission

Submission will be made via the Canvas LMS. Please submit your code and written report separately under the Final Project: Code and the Final Project: Report links on Canvas. Your code submission should include your model code, a readme file that explains how to run your code, and any additional files we would need to recreate your results. You should not include the provided images in your code submission, but your readme file should explain where your code expects to find these images. Your written report should be a .pdf that includes the description, analysis, and comparative assessment of the method(s) you developed to solve this problem. The report should follow the style of a short conference paper with no more than four A4 pages of content (excluding references, which can extend to a 5th page). The report should follow the style and format of an IEEE conference short paper. The IEEE Conference Template for Word, LaTeX and Overleaf is available here:

https://www.ieee.org/conferences/publishing/templates.html

Your report should explain the design choices in your method and justify these based on your understanding of computer vision theory. You should explain the experimentation steps you followed to develop and improve on your basic method, and report your final evaluation

result. Your method, experiments, and evaluation results should be explained in sufficient detail for readers to understand them without having to look at your code. You should include an error analysis which assesses where your method performs well and where it fails, provide an explanation of the errors based on your understanding of the method, and give suggestions for future improvements. Your report should include tables, graphs, figures, and/or images as appropriate to explain and illustrate your results.

Evaluation

Your submission will be marked on the following grounds:

Component	Marks	Criteria
Report writing	5	Clarity of writing and report organisation; use of tables, figures and/or images to illustrate and support results
Report method and justification	10	Correctness of method; motivation and justification of design choices based on computer vision theory
Report experimentation and evaluation	10	Quality of experimentation, evaluation and error analysis; interpretation of results and experimental conclusions
Statistical performance figures	3	How your method performs according to the computed statistics
Team contribution	2	Group self-assessment

The report is marked out of 25 marks, distributed between the writing, method and justification, and experimentation and evaluation as shown above.

In addition to the report marks, up to 3 marks will be given for methods that perform reasonably above a simple baseline. 1-2 marks will be given for methods that perform at or only marginally better than the baseline. 0 marks will be given for methods that perform below the baseline.

Up to 2 marks will be given for team contribution. Each group member will be asked to provide a self-assessment of their own and their teammate's contribution to the group project, and to mark themselves and their teammate out of 2 (2 = contributed strongly to the project, 1 = made a small contribution to the project, 0 = minimal or no contribution to the project). Your final team contribution mark will be based on the mark assigned to you by your teammate (and their team contribution mark will be based on the mark you assign to them).

Late submission

The submission mechanism will stay open for one week after the submission deadline. Late submissions will be penalised at 10% of the total possible mark per 24-hour period after the original deadline. Submissions will be closed 7 days (168 hours) after the published assignment deadline, and no further submissions will be accepted after this point.

Updates to the assignment specifications

If any changes or clarifications are made to the project specification, these will be posted on the LMS.

Academic misconduct

You are welcome — indeed encouraged — to collaborate with your peers in terms of the conceptualisation and framing of the problem. For example, we encourage you to discuss what the assignment specification is asking you to do, or what you would need to implement to be able to respond to a question. However, sharing materials — for example, showing other students your code or colluding in writing responses to questions — or plagiarising existing code or material will be considered cheating. Your submission must be your own original, individual work. We will invoke University's Academic policy (http://academichonesty.unimelb.edu.au/policy.html) where inappropriate levels of plagiarism or collusion are deemed to have taken place.