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

Assignment 1: Image formation and filtering

Due:	7pm, 19 Aug 2022
Submission:	Source code (in Jupyter Notebook) and written responses (as .pdf)
Marks:	The assignment will be marked out of 6 points, and will contribute 6% of your total mark.

1. Mapping between world and image coordinates [2 pt]

Figure 1: The full size version of this image (`Asst1_1_image.jpg`) has been provided with this assignment specification on Canvas.

What is the height of the sculpture in the photo `Asst1_1_image.jpg` (illustrated in Figure 1)? Use the pinhole projection model shown in lecture to work out the height. You can assume the ground plane is perfectly flat and horizontal. The camera is placed 56.35 m away from the statue at a height of 1.6 m, its optical axis is exactly parallel to the ground plane, and its vertical axis is exactly aligned

with the vertical axis in the world. The imaging sensor in this camera is 16.32 mm high by 9.77 mm wide. The image was taken with a focal length of 194 mm.

Show your work and include a diagram to illustrate how you set up the problem in your written report. Coding is optional for this question, but if you do not submit code, you must provide the formulas used to compute your answer in your written report.

2. Thinking with filters [4 pt]

Write an algorithm to count the number of intersections and dead ends in randomly-generated mazes. Your algorithm should take as input a maze like the one in Figure 2. Note that the annotations have been added for illustration purposes; the actual input images will not be annotated. Your algorithm should detect the intersections in the maze (points where the path branches into 2 or more paths) and return the total number of these. Your code should also detect the dead ends in the maze (points where the path ends, not counting the start/exit points) and return the total number of these.

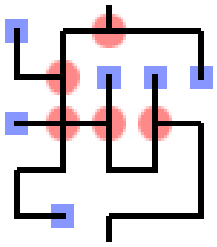

Figure 2: Example maze with five intersections (red circles) and six dead ends (blue squares). Maze image from www.mazegenerator.net.

Two sample mazes have been provided with this assignment spec. Each maze consists of black lines on a white background and has a start/exit point at the top/bottom edge of the image. You can assume the line width, line spacing, line colour, and background colour will be the same for all mazes.

Your algorithm should work correctly for any maze image from this generator. Importantly, your algorithm should also be efficient. You should only use spatial filtering and matrix operations to solve this problem. You can and should design your own filter kernels for this task. *Your code should not iterate over pixels.* Solutions that involve iterating over pixels will receive 0 marks for this problem.

Your written report should give the intersection and dead end counts for the two sample mazes, and briefly explain your method. Show the kernel values that you used in your filtering steps and include figures to illustrate the result of each filtering step on a sample maze.

Submission

You should make two submissions on the LMS: your code and a short written report explaining your method and results. Please note that although coding is optional for question 1, both questions require a written response. The response to each question should be no more than 500 words.

Submission will be made via the Canvas LMS. Please submit your code and written report separately under the **Assignment 1** link on Canvas.

- Your **code** submission should include the Jupyter Notebook (please use the provided template) with your code and any image files we will need to run your code. Please include the cell output in your notebook submission if possible.

- Your written **report** should be a .pdf with your answers to each of the questions. The report should address the questions posed in this assignment and include any images, diagrams, or tables required by the question.

Evaluation

Your submission will be marked on the correctness of your code/method, including the quality and efficiency of your code. You should use built-in Python functions where appropriate and use descriptive variable names. Your written report should clearly explain your approach and any experimentation used to produce your results, and include all of the specific outputs required by the question (e.g., images, diagrams, tables, or responses to sub-questions).

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 Misconduct policy (`http://academichonesty.unimelb.edu.au/policy.html`) where inappropriate levels of plagiarism or collusion are deemed to have taken place.