Module	Introduction to Image Processing (IIP) / COMP 2032 / Semester 2
Module Convenor(s)	Tissa Chandesa

Assessment Name	Coursework	**Weight**	50%
Description and Deliverable(s)	The coursework (details below) requires you to **work in groups of 4** to develop a software solution, using Python, to a real image processing problem and collaboratively producing a conference paper describing and critically evaluating your solution. The deliverables required are: Per group submission: 1. Python codes: *.py* files (submitted as a ZIP folder) 2. A 6-page conference paper: 2000 words max, PDF format (**note:** *additional pages will result in a deduction of 5% for every additional page for the entire group*). 3. Result Images (submitted as a ZIP folder) – Output (*folder name*) – Image Processing Pipeline (*folder name*) Individual submission: 4. Each of you will need to make an additional submission, **separately** for your **peer assessment** form. Please click on the "**Peer Assessment**" link on Moodle to perform this submission. Aside from this **assessment sheet**, you will be provided with the following documentations: 1. **COMP2032-CW-GroupXXX.docx**: this is your conference paper template. Please change **XXX** to your group number. Also, please **do not** change the formatting and headers. Marks will be deducted if submitted document does not follow the original formatting style. The provided table as well as text coloured red and blue, respectively are to be replaced by your own written words. **Note:** *COMP2032-CW-GroupXXX.docx* **is to be converted to PDF prior to submission and all text should be coloured in black.** **Segmentation of Flowers** The need to separate one material from another, or identify a particular material, arises in many image processing applications. From industrial inspection, where you might be asked to identify pixels depicting correctly manufactured cloth, to human-computer interaction, where you might need to detect human skin, the core problem remains the same: develop a processing pipeline that converts a colour image into a binary image, labelling different materials. Biological science is a current growth area for image processing and computer vision. Biologists now have access to powerful tools that allow them to study the genetic structure of plants and animals, but lack the accurate, quantitative data on plant and animal structure as well as behaviour needed to understand the effect different genetic factors have. Plant science is an area of particular activity, as improved plant species are urgently needed to provide food to an expanding population. In this coursework, you will be asked to develop an image processing pipeline to separate flower material from background. This requires a processing pipeline that usually includes at least some of the following steps: **Colour space conversion**: choose a colour space. Any can be used, but most people choose to work in a lower-dimensional (<3) space whenever possible.		

Noise reduction: depending on image quality, some form of noise suppression may be required.

Thresholding/Segmentation: image regions corresponding to nuclei must be identified, and thresholding is a common approach. The method used varies and may be global or local. Methods that automatically determine the threshold value have obvious advantages over those that require user interaction.

Binary image processing: Identifying a perfect threshold value or segmentation method is almost impossible, and most methods will result in some mis-classified pixels. A further binary image processing stage is often needed to clean up the image, hopefully leaving it containing only regions that correspond to plants. Nothing, however, is perfect.

Dataset: Images of three species of flowers are collected in a folder available from Moodle. The images are selected based on the complexity of their foreground/background features and categorised into three sub-folders (e.g., easy, medium, and hard) within the *input-image* folder. To aid with analysing the effectiveness of your solution, **ground truth** for each image is provided within the *ground-truth* folder, again categorised within three sub-folders (e.g., easy, medium, and hard).

Design and implement a single **Python program** that inputs one of the images from the dataset at a time and ***outputs a binary image marking regions corresponding to flower material***. You may not need to employ all the steps listed above, but you will probably find it worthwhile to at least consider them all. You are also encouraged to use other strategies covered in this module to conduct the flower segmentation problem. ***Deep-learning-based methods and supervised data-driven learning approaches are not allowed in this coursework***. The focus is on **design and development of an image processing pipeline** to perform listed tasks. You must make sure to evaluate the final outcome. The evaluation can be both **qualitative and quantitative** – for example ***mIoU (mean Intersection over Union)*** can be used as the metrics to evaluate your image processing pipeline's performance.

You should seek a solution that is *as* ***automatic*** *as possible* (try to minimise the number of user-supplied parameters) and operates on all of the images without user intervention (one parameter set for all).

You are required to make your code easy to be executed by the marker. In other words, the marker can run your code and **reproduce** your given outputs. After the execution of your program, it is expected that each image in the ***input*-image folder** has a corresponding segmentation result represented by a .jpg file saved in the ***output folder***, categorised within three sub-folders (e.g., easy, medium, and hard) with the same name as the image. The segmentation result should be binary, i.e. 0 refers to the background and 1 represents the flower. Additionally, images generated as part of your image processing pipeline can be saved in the *image-processing-pipeline* folder whereby you can create sub-folders for each input image and label the sub-folders as the same name as the input image.

NOTE: A **readme file** is recommended to be produced alongside the source code.

When writing your conference paper, consider the following points:

1. *Detail* the chosen method(s) and specific image processing technique(s) employed.
2. *Explain* why you choose those technique(s) and method(s). You may include the **pseudocodes** of the chosen technique(s) and method(s) (if applicable).
3. *Present* the results obtained from your practical implementation.

	4. *Critically* evaluate your method on the basis of those results: what are its strength and weakness? This section should make explicit reference to features of the results you obtained. **Reference:** R.C. Gonzalez and R.E. Woods. (2018). *Digital Image Processing*. (Fourth Edition). Prentice Hall. Chris Solomon and Toby Breckon. (2010). Fundamentals of Digital Image Processing: A Practical Approach with Examples in Matlab. Wiley Prateek Joshi. (2015). OpenCV with Python by Example: Build Real-World Computer Vision Applications and Develop Cool Demos Using OpenCV for Python. Packt Publishing Sandipan Dey. (2020). Python Image Processing Cookbook: Over 60 Recipes to Help You Perform Complex Image Processing and Computer Vision Tasks with Ease. Packt Publishing Hypermedia Image Processing Reference (HIPR) at the University of Edinburgh https://homepages.inf.ed.ac.uk/rbf/HIPR2/ CVOnline, University of Edinburgh, https://homepages.inf.ed.ac.uk/rbf/CVonline/ 17 Categories of Flowers Dataset, Visual Geometry Group, University of Oxford https://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html
Release Date	Tuesday, 27 February 2024
Submission Date	Friday, 26 April 2024, by 11:59pm
Late Policy **(University of Nottingham default will apply, if blank)**	Work submitted after the deadline will be subject to a penalty of 5 marks (the standard 5% absolute) for each late **working day** out of the total 100 marks.
Feedback Mechanism and Date	Marks and written individual feedback will be returned via Moodle 3 June 2024
Assessment Criteria	Python code(s): **30%** [**unable to run codes will result in 0% being awarded**] Justification of chosen algorithms based on performed literature review: **15%** Critical analysis and discussion of your findings: **25%** Discussion of the strengths and weaknesses of the chosen algorithms (based on your findings), prior to concluding on the most optimum algorithm: **30%**

Assessment Name	Examination	**Weight**	50%
Description	In-person exam		
Release Date	TBA		
Submission Date	TBA		
Late Policy **(University of Nottingham default will apply, if blank)**			

Reassessment Method	**Weight**
Exam	100%