Module:	ICP3038 – Computer Vision
Department:	School of Computer Science & Electroni Engineering
Module credit:	20
Organiser:	Dr Franck P. Vidal
Assignment weight:	25%
Assignment deadline:	See on Blackboard (midnight)

PRIFYSGOL BANGOR UNIVERSITY

Assessment 1 – Image Processing Toolbox in C++

Implementating functionalities found in Photoshop, Gimp and ImageJ

Description

In this assignment, you will exploit the result of our initial development in the labs where we implemented a C++ class to handle a greyscale 2D image. You will add new methods to the Class Image to perform some image processing tasks, e.g. distance metrics between two images of the same size, spacial filters, and some additional point operators.

Contribution of this assessment

This assessment contributes to 25% of the overall module mark.

Mapping assessment to learning outcomes

- Effectively use image representations to solve computer vision problems
- Apply image processing filters and operators to achieve given goals of an imaging system.
- Programming skills for image processing

Tasks & Requirements

The assignment is based on the implementation of image processing algorithms. Your code should include:

1. The **Root Mean Squared Error (RMSE)** between two images
   (hint: if the images have different sizes, throw an error or return a large number, e.g. `FLT_MAX` that is declared in the `cfloat` header file). The RMSE can be used to measure how different two images are. It is:
   - 0 if they are the same; and
   - > 0 if they are different.
2. The **Zero mean Normalised-Cross Correlation (NCC)** between two images
   (hint: if the images have different sizes, throw an error). The ZNCC can be used to measure how similar two images are. It is:
   - 1 if they are fully correlated (i.e. they are extremely close to each other);
   - 0 if they are fully uncorrelated (i.e. they are extremely different); and
   - -1 if they are fully anticorrelated (i.e. one if the negative of the other one).

   (hint: if the images have different sizes, throw an error or return 0).

3. **Blending** of two images (hint: you may use the operators * and + that you implemented in the labs).
4. **Segmentation** using the **thresholding** technique.
5. **Spacial convolution** of an image by any kernel
   (hints: Remember to treat edge of the image as a special case, the kernel may be considered as an image).
6. Convolution using **preset kernels** (inc. Mean filter, Gaussian blur, Laplacian filter, Gradient magnitude using the Sobel Edge detector)
   (hint: Use the previous method).
7. **Sharpen filter** (hint: you may use the operators *, + and - that you implemented in the labs, as well as the Gaussian blur that you implemented for the assignment).

Note: The functionalities of the class have to be tested and validated. You may use unit testing, or just a visual examination of the output of your image processing, or even both.

Task 1: Implementation

You should use the the Image class from Assignment 1 as a starting point for your second assignment. We provide you with several files, but you should mainly change `Image.cxx` and `test_assignment.cxx`.

Your task is to complete the class in `Image.h` and to make sure every method is implemented in `Image.cxx`.

Task 2: Testing

The code has to be tested in `test_assignment.cxx`. Evidence of the results of the test will be assessed. For examples: You can compute the ZNCC and RMSE to measure how similar are two images. We provide test data to help you test the functionalities. `test_assignment.cxx` includes an example of tests for the segmentation using the thresholding technique. You should add similar tests for the other functionalities and filters.

Task 3: Write a short report

Your final submissions should include the code and a report that discusses the changes you made to the code and the experiments showing that the code works. It should describing what you have implemented, what works, what does not, and how you created the filters. Whenever possible, use pictures to illustrate your text. For example, you are expected to show the image before and after you apply a filter. You can use the test data provided to assess your code. Provide ZNCC and RMSE values to show that your filters provide the expected results.

At the end of your report, provide a critical analysis of your performance.
In appendix of your report, join your own copy of

- `Image.h`,
- `Image.cxx`, and
- `test_assignment.cxx`.

Submission procedure

Write your report in a Word document. You must submit a .doc or .pdf file of your short report in the following format: "username-ass1.pdf". Include your source code in the appendix.

The assignment is due on: See on Blackboard (midnight). Submissions must be made via Blackboard. Please take into account upload times and internet connections when considering how much time you have remaining.

Assessment method

Your report and your code will be marked.

Note that care will be given to details. You are expected to use initialisation lists in constructors. No memory leak is allowed. Variable and function names should be meaningful. An appropriate coding standard should be used consistently. The code should compile without error. The code should compile without warning if possible. You should complete the header information at the top of each file.

Plagiarism and Unfair Practice

Plagiarised work will be given a mark of zero. Remember when you submit you agree to the standard agreement:

> This piece of work is a result of my own work except where it is a group assignment for which approved collaboration has been granted. Material from the work of others (from a book, a journal or the Web) used in this assignment has been acknowledged and quotations and paraphrasing suitably indicated. I appreciate that to imply that such work is mine, could lead to a nil mark, failing the module or being excluded from the University. I also testify that no substantial part of this work has been previously submitted for assessment.

Late Submission

Work submitted within one week of the stated deadline will be marked but the mark will be capped at 40%. A mark of 0% will be awarded for any work submitted 1 week after the deadline.

Acceptable reasons for submitting work late include: Serious personal illness with a doctors certificate (a self-certified medical note should not be accepted). The death of a relative or close friend. Serious family problems such as divorce, separation and eviction. Examples of unacceptable reasons for failing to submit work on time include: Having exams; Having other work to do; Not having access to a computer; Having computer related problems; Being on holiday; Not being able to find information about a subject.

Marking Scheme

Please remember that marks are provisional until they are confirmed by a board of examiners.

The marking will take into account:

- What is implemented, what is not implemented;
- The amount and usefulness of comments;

- The naming of functions, variables, etc. (including the consistency in the coding standard);
- Testing;
- Verbal explanation in the lab after the submission if needed.

The marks will be split as follows:

1.	RMSE:	10%
2.	ZNCC:	15%
3.	Thresholding:	10%
4.	Blending:	15%
5.	Spacial convolution:	15%
6.	Kernel preset:	25%
7.	Sharpen:	10%

To understand the final grade, refer to the table below:

> 80, exceptional image processing skills in C++. Clear demonstrable understanding use of image processing techniques in C++. Outstanding development of all the methods. Superb use of coding standards and comments. Overall an exceptional and well-designed test program that demonstrates the validity of every functionality. The report is exceptional and provides a comprehensive and clear critical analysis of the work performed. An exemplar solution that could be used to demonstrate good practice of parallel programming to colleagues.
> 70, good implementation that demonstrates a good understanding of image processing using C++ programming. The code works effectively and the test program is effective to check the validity of every functionality. The code is written using coding standards. The code is well commented. The report is comprehensive and makes a good critical analysis of the work. Overall a very good solution to this assignment. A well-structured report is provided (including a good critical analysis of the work provided).
> 60, a good implementation that demonstrates a suitable understanding of both image processing and C++ programming. The code works effectively and the test checks the validity of most functionalities. Coding standards are relatively well applied and there are some comments. Some limitations may exist in the work, however a good attempt made, and although there may be some limitations with the program a comprehensive report (with an excellent and well critiqued section) is included.
> 50, appropriate demonstration of the challenge. Maybe full functionality is not provided, however the student demonstrates that they understand the processes required to achieve the assignment and understand some of the challenges of image processing in C++.Comments are sparse and the use of coding standards is approximate. Even with these limitations the report is well presented, and their achievements are well criticised and discussed. Limitations to the work are clearly presented in the report and the student clearly understands what they have achieved and the limitations thereof. Overall, maybe some flaws, but a reasonable submission.
> 40, threshold performance. Demonstrates some understanding of image processing techniques and some idea of testing the functionalities. Some attempt has been made over creating the metrics and filters. The code is not well commented and the coding standards are not used appropriately. The report discusses some of the issues, and provides a basic critique of the work submitted.
< 40, below threshold performance, with little demonstration of knowledge of image processing skills. Little thought has been made over this assessment, and understanding is confused.

Feedback details

	Description	Timeframe
Formative (On-going)	Verbal Feedback – Verbal feedback will be available by request. It is suggested that you keep a written note of this feedback to aid in your personal development.	Instant
Summative (Post Assessment)	Written Feedback – Written feedback will be made available through blackboard after an assignment is submitted. To access your written feedback see the comments section of your assignment submission.	1-2 weeks