IMPERIAL

Advanced Programming

Assessment 2: Group Project

Image Filters, Projections and Slices

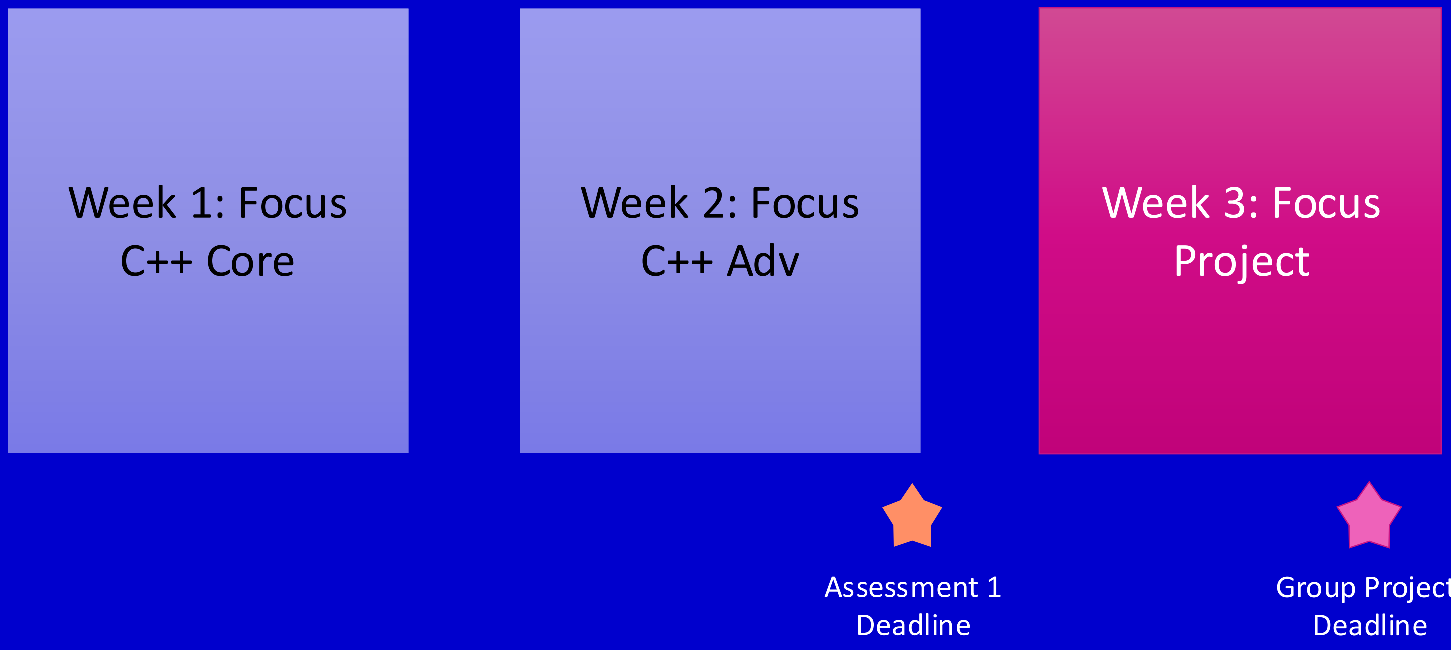
Tom Davison
thomas.davison@imperial.ac.uk
RSM 4.85

Core task: Image Processing with C++

- Build a C++ program using the programming techniques you have learned during the Advanced Programming course.
- Take inputs of 2D images or 3D data volumes (e.g. CT scans)
- Apply image filters, orthographic projections and slices
- Output result as an image

Advanced Programming Schedule

- Possible to complete project in **four days**.
- Apply knowledge learned over weeks 1 & 2 during the project in week 3



Week 1: Focus C++ Core

Week 2: Focus C++ Adv

Week 3: Focus Project

Assessment 1 Deadline

Group Project Deadline

Group working

GTA support Monday pm,
Tuesday, Wednesday, Thursday pm

Expectation is that you will work on campus with your group for most of the week

Any concerns about group work (e.g., a member is not in contact with the group) please let me know ASAP so we can fix the situation!

	Monday	Tuesday	Wednesday	Thursday
am	**Group work** 1.49/50, 1.51, 3.01D/E	**Group work** 1.49/50, 1.51, 3.01D/E GTA support	**Group work** 1.49/50, 1.51, 3.01D/E GTA support	**Group work** 1.49/50, 1.51, 3.01D/E GTA support
pm	**Group work** 1.49/50, 1.51, 3.01D/E GTA support	**Group work** 1.49/50, 1.51, 3.01D/E	**Group work** 1.49/50, 1.51, 3.01D/E	**Group work** 1.49/50, 1.51, 3.01D/E **Deadline 16:00**

Groups

- Groups have been automatically generated
- Aim to balance the number of ACSE/EDSML students in each group
- Also balance the average grade in other modules to date, to make it as fair as possible

- Repositories will be released at the end of this lecture
- We are **not** creating group channels in Teams, up to you to decide how best to communicate with your group

Group names: Algorithms

Barnes-Hut	Karger
Bellman-Ford	Kruskal
Binary-Search	Linear-search
Bisection	Merge-sort
Brent	Monte-Carlo
Bubble-sort	Newton
Crank-Nicholson	Prim
Dijkstra	Quicksort
Edmonds	Ridder
Euler	Runge-Kutta
Fibonacci	Secant
Floyd	Tarjan
Gosper	Ukkonen
Halley	Vincenty
Insertion-sort	Warnsdorff
Johnson	

Group working space

Should be a table for every group

For the rest of the week, please work in these rooms when in the department

GTAs will be available in all rooms

I will make sure to visit every room each day

3.01 D/E	1.51	1.49/1.50
Barnes-Hut	Dijkstra	Newton
Bellman-Ford	Edmonds	Prim
Binary-Search	Euler	Quicksort
Bisection	Fibonacci	Ridder
Brent	Floyd	Runge-Kutta
Bubble-sort	Gosper	Secant
Crank-Nicholson	Halley	Tarjan
	Insertion-sort	Ukkonen
	Johnson	Vincenty
	Karger	Warnsdorff
	Kruskal	
	Linear-search	
	Merge-sort	
	Monte-Carlo	

A welcoming environment

- We encourage and promote diversity in science
- Whoever you are, and whatever your background, we welcome you.
- We hope that everyone finds their experience welcoming, encouraging and rewarding.
- We want to foster a community based on mutual respect, tolerance, and encouragement and we kindly ask that you respect these principles.
- A part of that means being inclusive in your discussion and communicating in English as a group.
- If you experience or witness any unwelcome behaviour, we encourage you to challenge the behaviour or report it, in confidence, to the module coordinator or the course director.
- If you have questions about support or pastoral concerns, talk to PGT Senior Tutor: James Percival (j.percival@imperial.ac.uk)

Image processing

Image processing

Colour spaces

Most images we will work with will be in one of a few colour spaces:

- **Grayscale**: single channel, denotes brightness
  - Ranging from 0 (black) to 255 (white)

- **RGB**: three channels, brightness of red, green and blue component of each channel

- **RGBa**: four channels, RGB + alpha
  - Alpha is the transparency
  - Generally ignored for this project (i.e. can leave it as it is, or remove it and save as RGB)

Image Processing

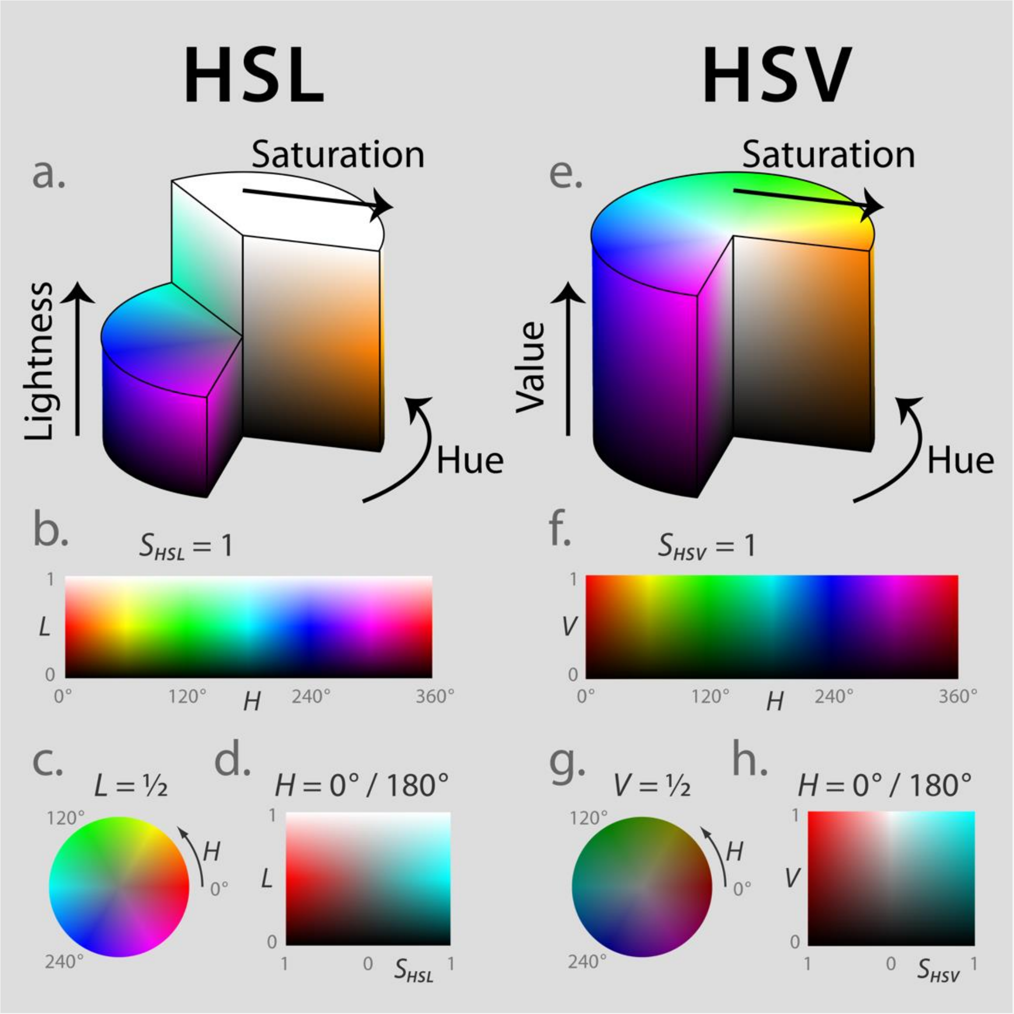
Colour space conversions

- Sometimes it is helpful to store the pixel information in a different colour space.

- Common examples include **HSV** and **HSL:**
  - **HSV**: Hue, Saturation, Value
  - **HSL**: Hue, Saturation, Luminance

- Note: Saturation is defined **differently** between HSL and HSV.

- May find it useful to write some **helper functions** to perform conversions to/from HSL and HSV



https://en.wikipedia.org/wiki/HSL_and_HSV

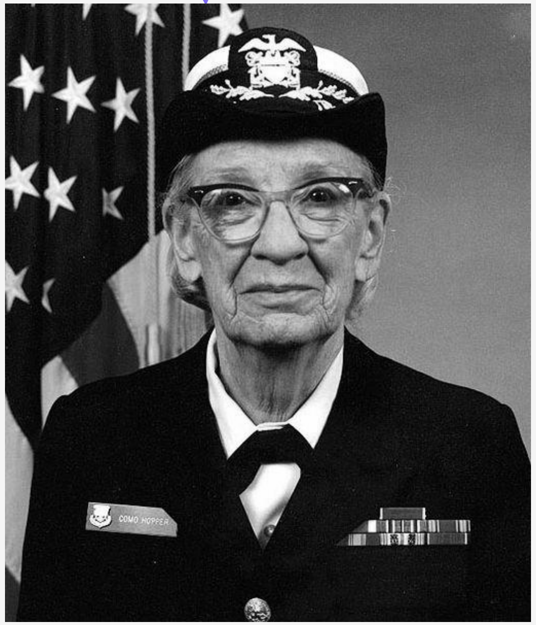
2D Image Filters

1. Colour correction / simple filters

- a) **Greyscale**
- b) Brightness
- c) Histogram equalisation
- d) Threshold
- e) Salt and Pepper Noise

Grace Hopper:
pioneer of computer science

$0.2126R +$
$0.7152G +$
$0.0722B$



2D Image Filters

1. Colour correction / simple filters

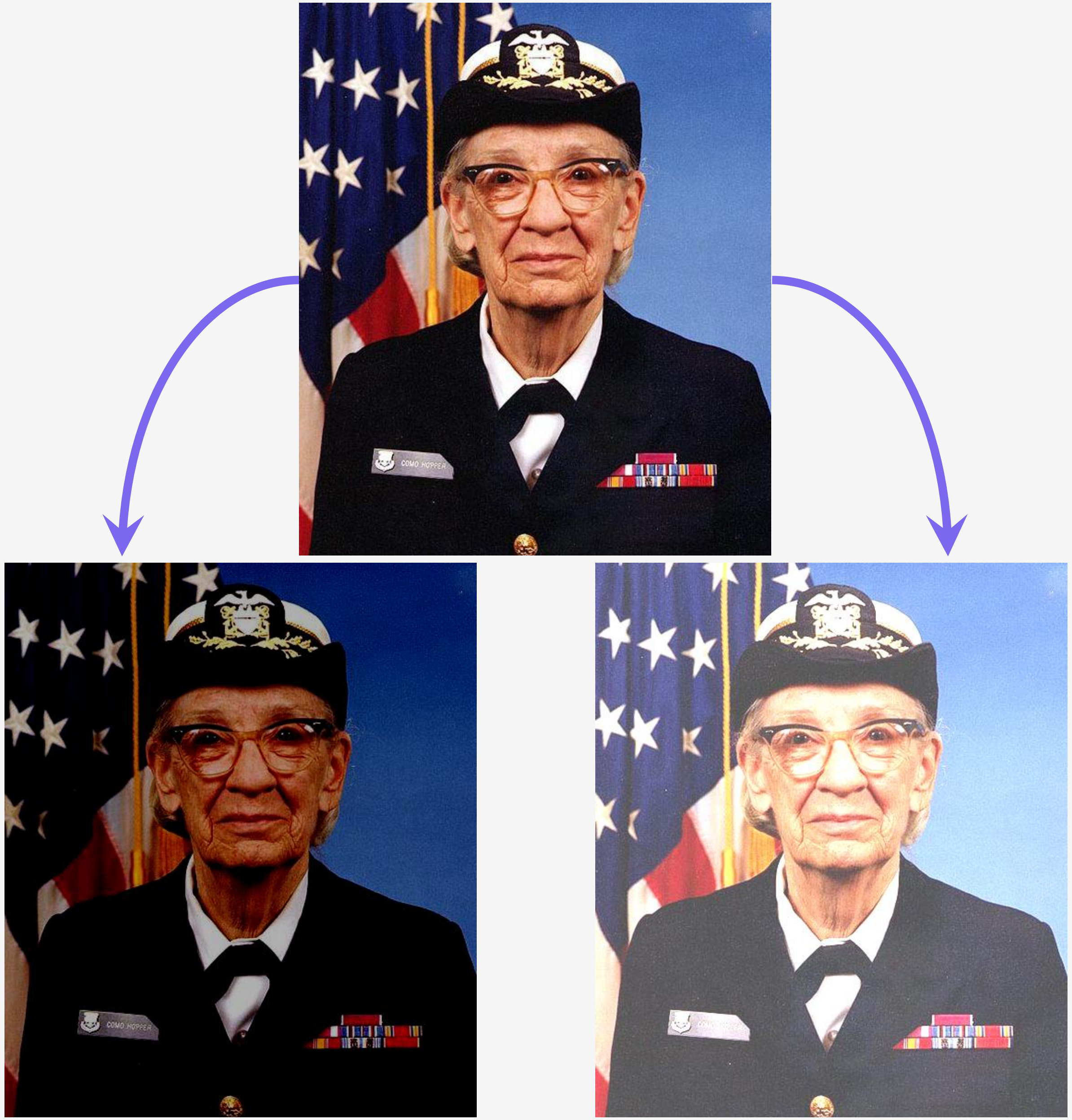
a) Greyscale

b) Brightness

c) Histogram equalisation

d) Threshold

e) Salt and Pepper Noise



2D Image Filters

1. Colour correction / simple filters

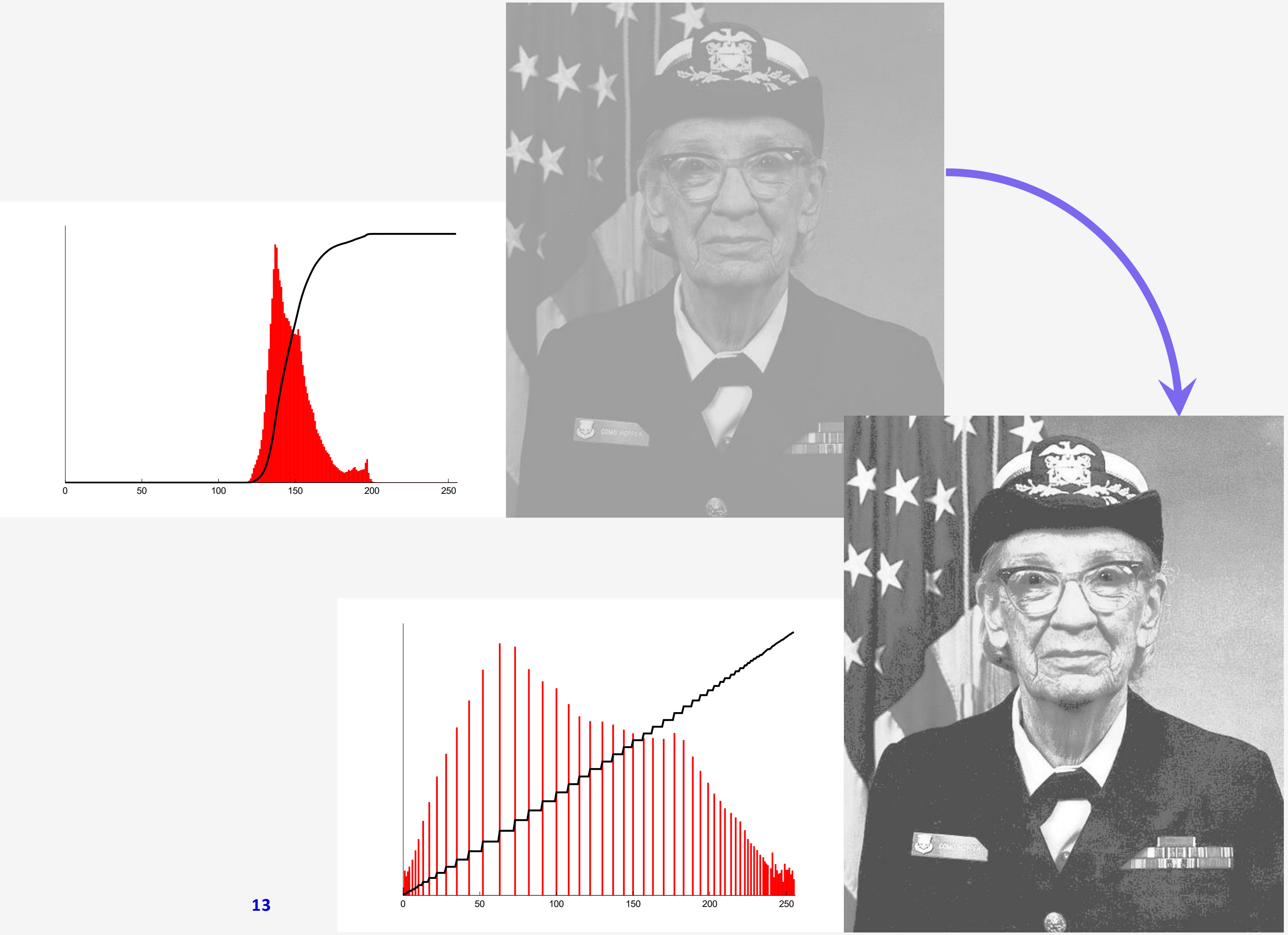
a) Greyscale

b) Brightness

c) Histogram equalisation

d) Threshold

e) Salt and Pepper Noise



2D Image Filters

1. Colour correction / simple filters

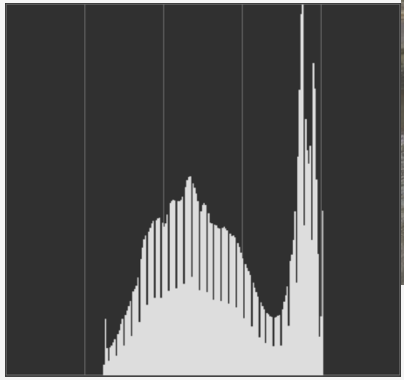
a) Greyscale

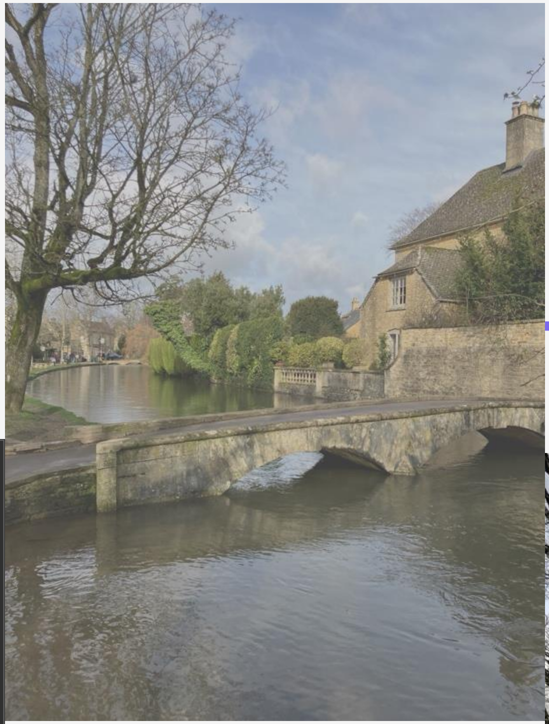
b) Brightness

c) **Histogram equalisation**

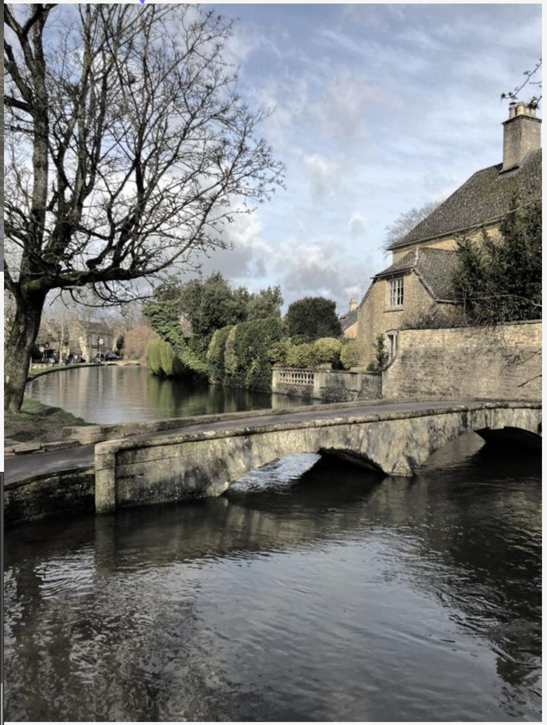
d) Threshold

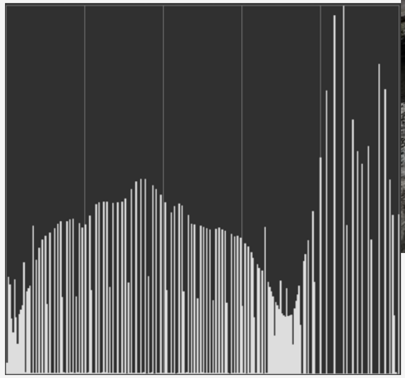
e) Salt and Pepper Noise





For RGB images, convert to HSL/HSV, and equalise the histogram of the L or V channel





2D Image Filters

1. Colour correction / simple filters

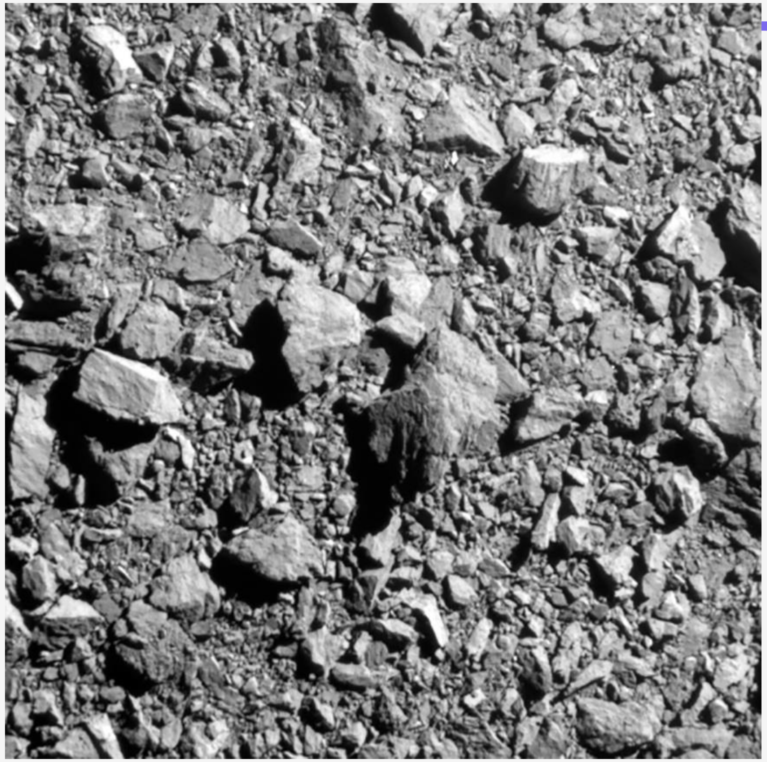
a) Greyscale

b) Brightness

c) Histogram equalisation

d) **Threshold**

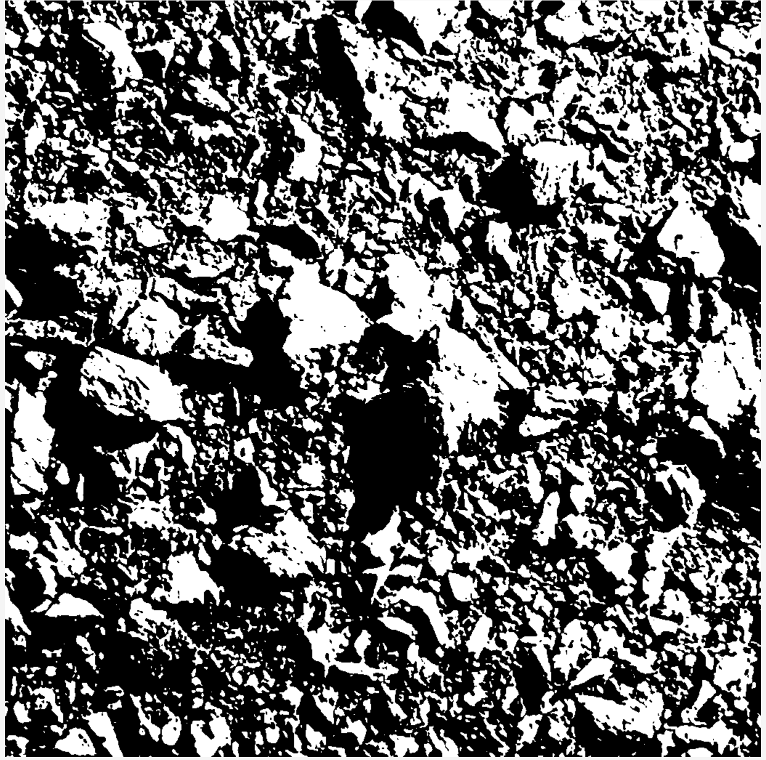
e) Salt and Pepper Noise



Asteroid Dimorphos surface
Image credit: NASA, APL

For RGB images, convert to HSL/HSV, and threshold the L or V channel

Threshold at a value of 127



2D Image Filters

1. Colour correction / simple filters

For RGB images, convert to HSL/HSV, and threshold the L or V channel

a) Greyscale

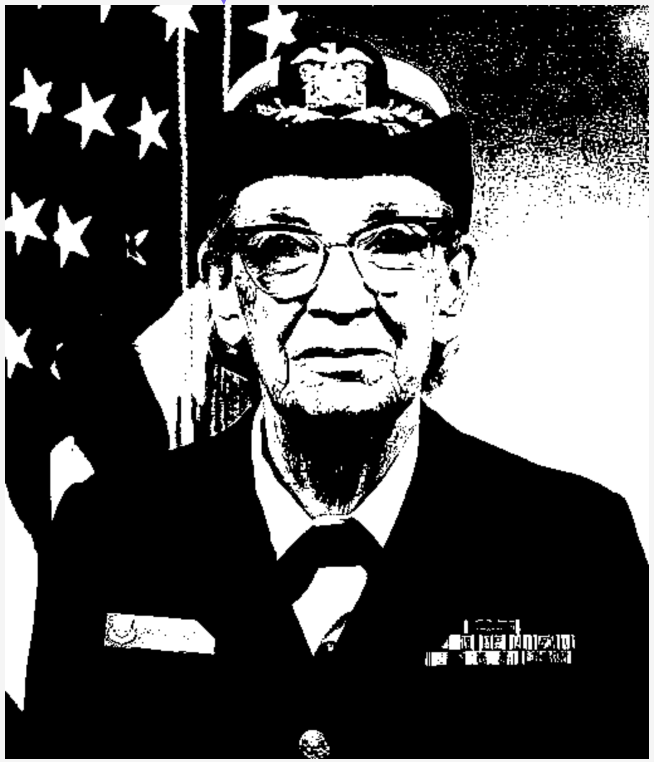
b) Brightness

c) Histogram equalisation

d) Threshold

e) Salt and Pepper Noise

Threshold luminance at a value of 127



2D Image Filters

1. Colour correction / simple filters

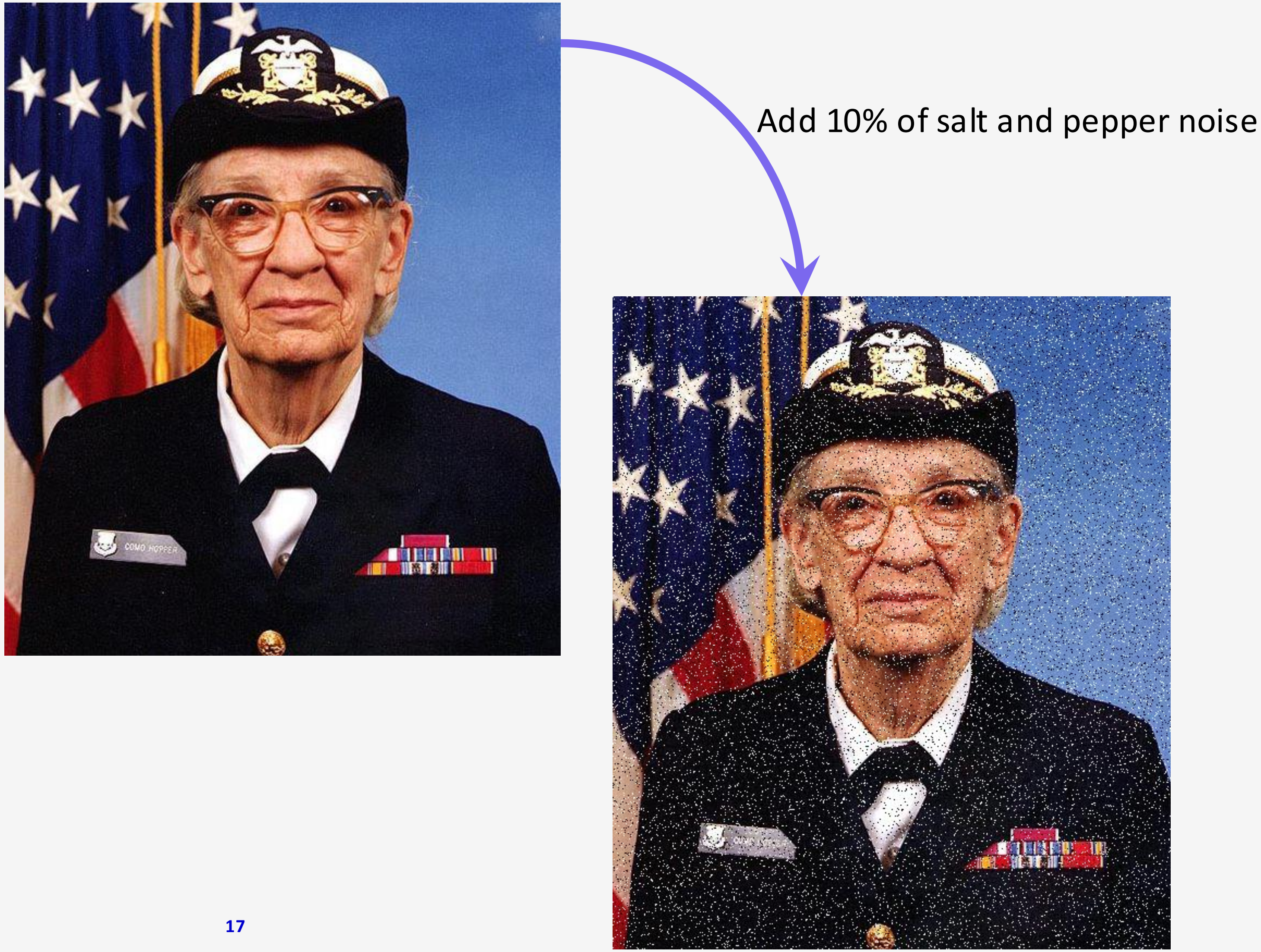
a) Greyscale

b) Brightness

c) Histogram equalisation

d) Threshold

e) Salt and Pepper Noise



Add 10% of salt and pepper noise

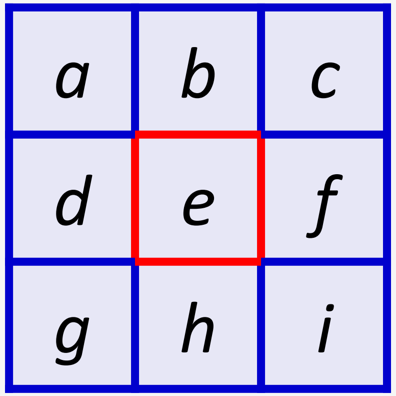
2D Image Filters

2. Convolutional filters for image blur (arbitrary kernel size)

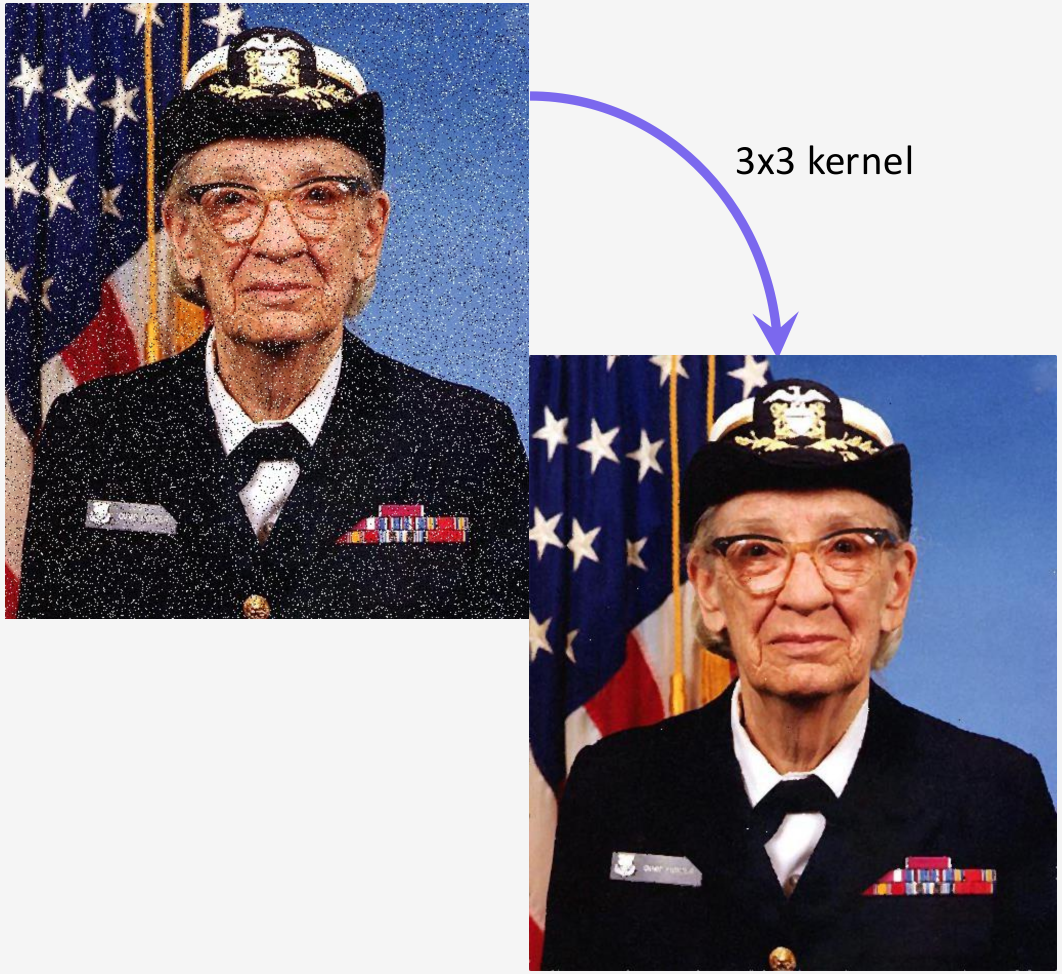
a) **Median Blur**

b) Box Blur

c) Gaussian Blur



a	*b*	*c*
d	*e*	*f*
g	*h*	*i*



(Original image)

2D Image Filters

2. Convolutional filters for image blur (arbitrary kernel size)

a) Median Blur

b) Box Blur

c) Gaussian Blur

Example 5x5 Box blur kernel

0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04

Example 5x5 Gaussian blur kernel
(st. dev. = 1)

0.003	0.013	0.022	0.013	0.003
0.013	0.060	0.098	0.060	0.013
0.022	0.098	0.162	0.098	0.022
0.013	0.060	0.098	0.060	0.013
0.003	0.013	0.022	0.013	0.003

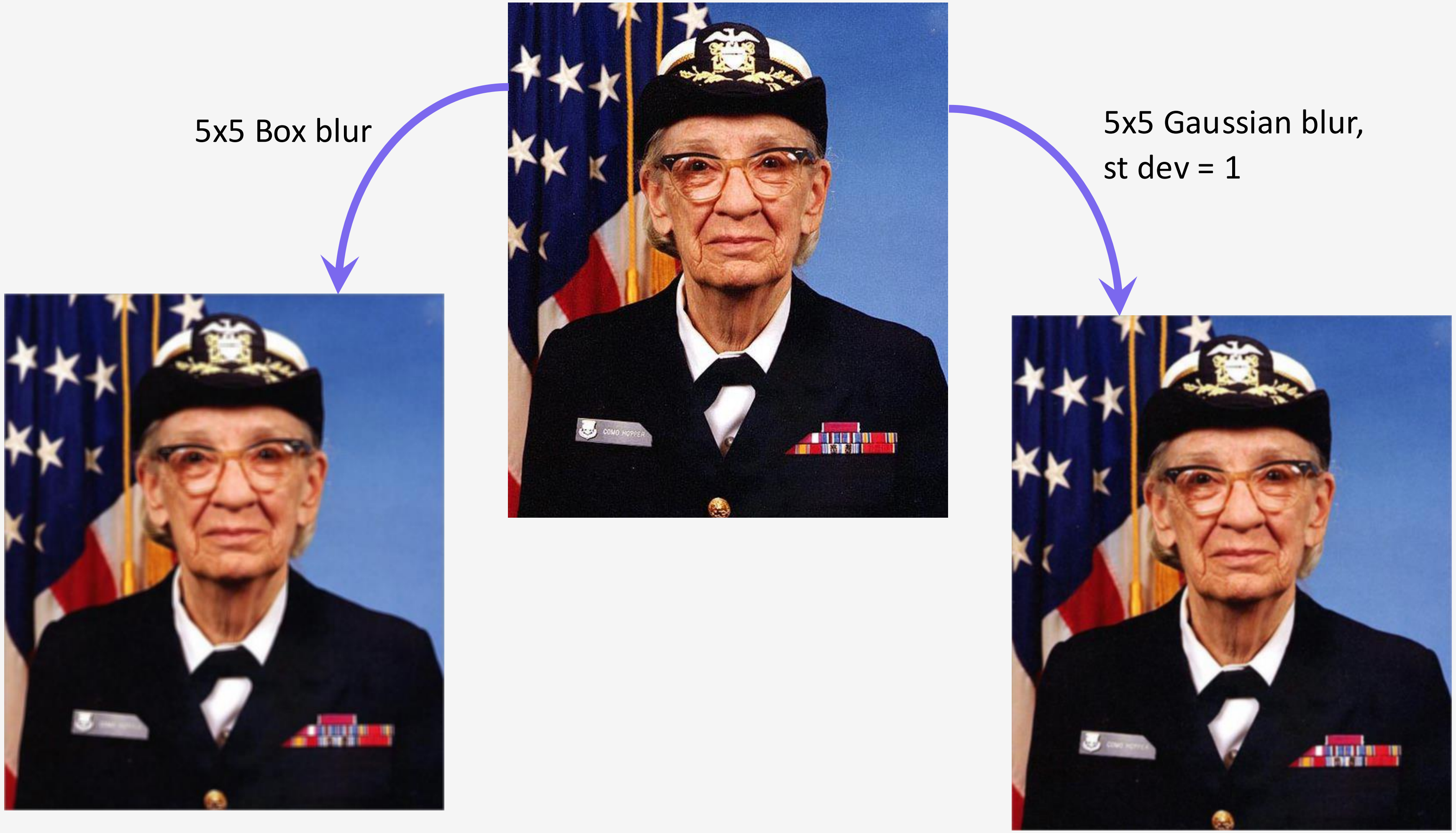
2D Image Filters

2. Convolutional filters for image blur (arbitrary kernel size)

a) Median Blur

b) Box Blur

c) Gaussian Blur



5x5 Box blur

5x5 Gaussian blur,
st dev = 1

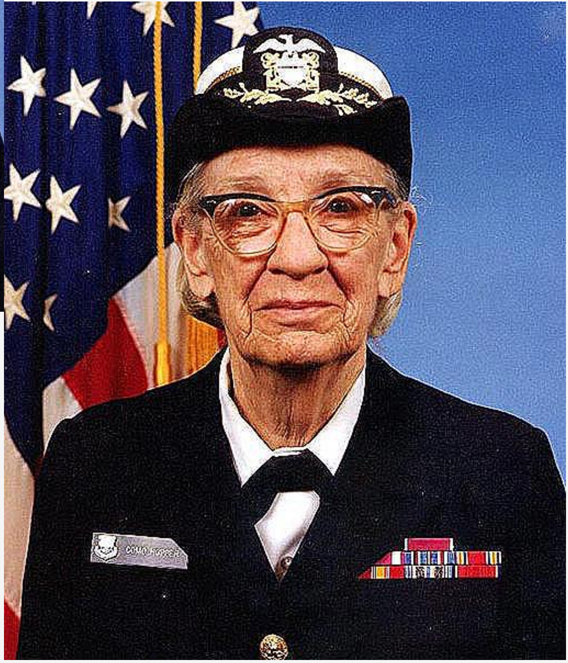
2D Image Filters

3. Convolutional filter for image sharpening

Apply Laplacian kernel to find edges, and add that result to original image.

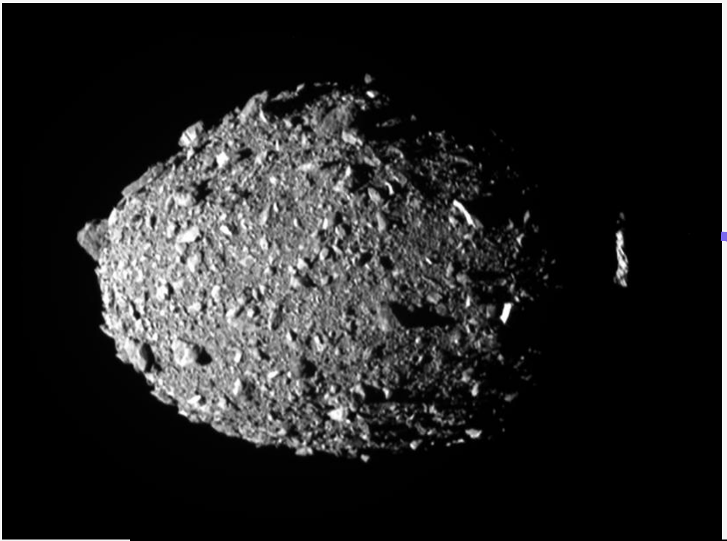
“Sharpens” images, making edges more well-defined.

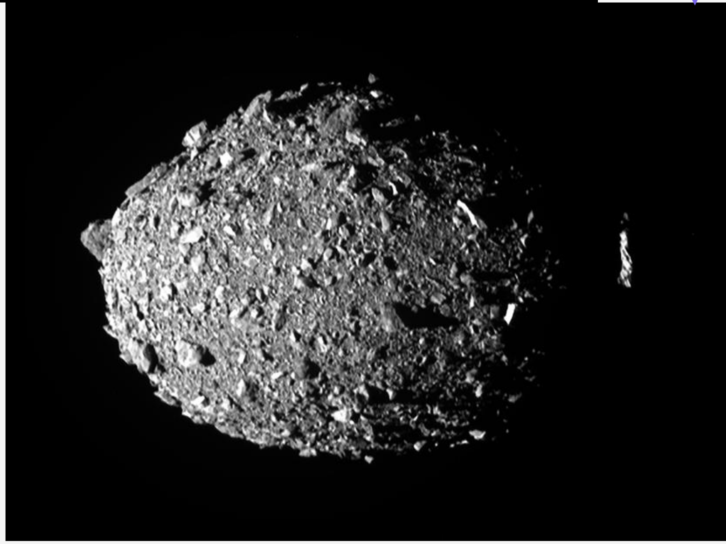
For RGB images, works on each channel separately



$$G = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$I_{\text{sharp}} = I_{\text{original}} + G$$





2D Image Filters

4. Convolution filters for edge detection (fixed kernel size)

a) Sobel

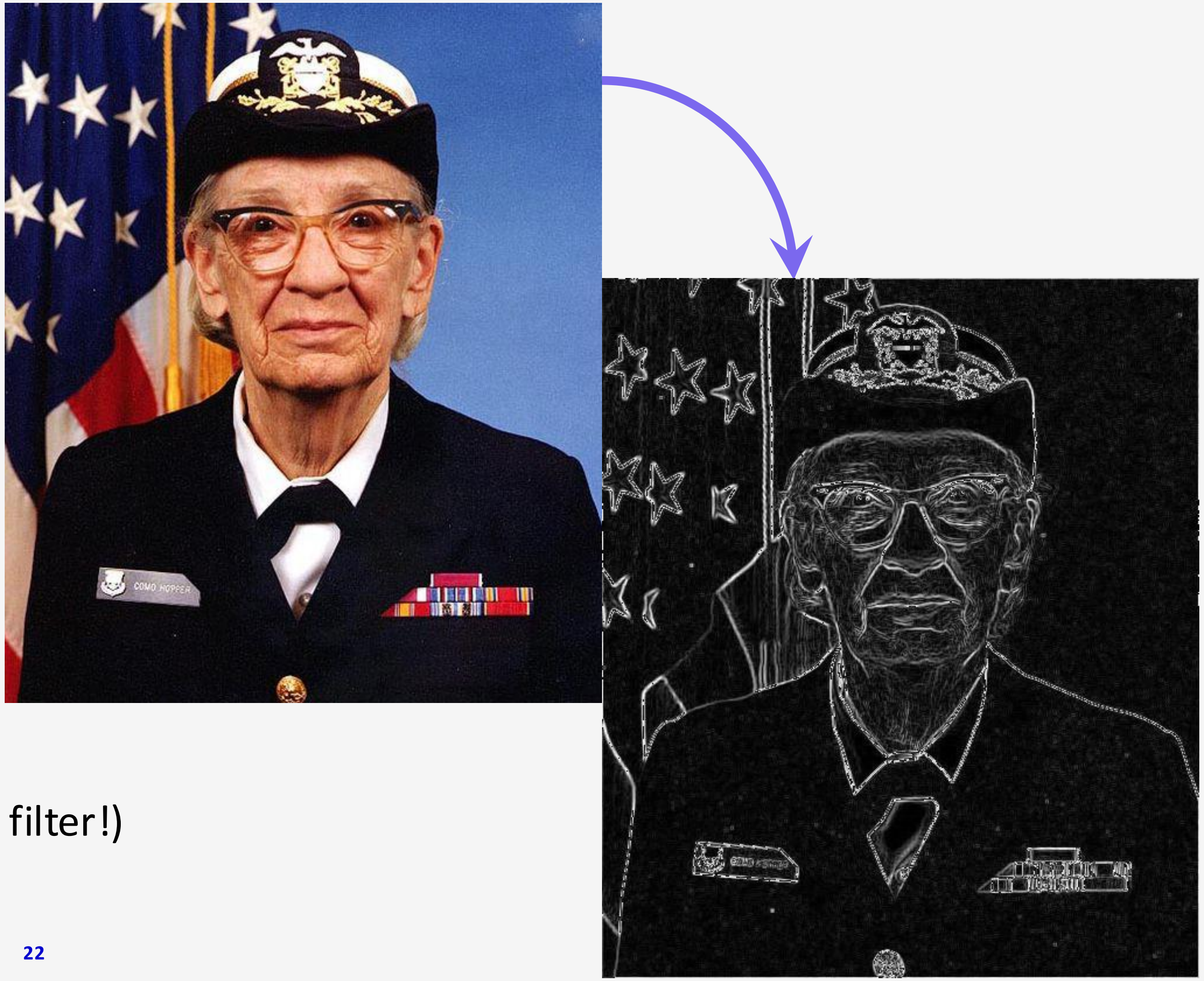
b) Prewitt

c) Scharr

d) Robert's Cross

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$



(convert to grayscale first – use your filter!)

2D image filters

Group working suggestions

- Each member of the group should write at least one 2D image filter
- Everyone gets a chance to write some code
- Paired programming is fine (write at least 2 filters as a pair)

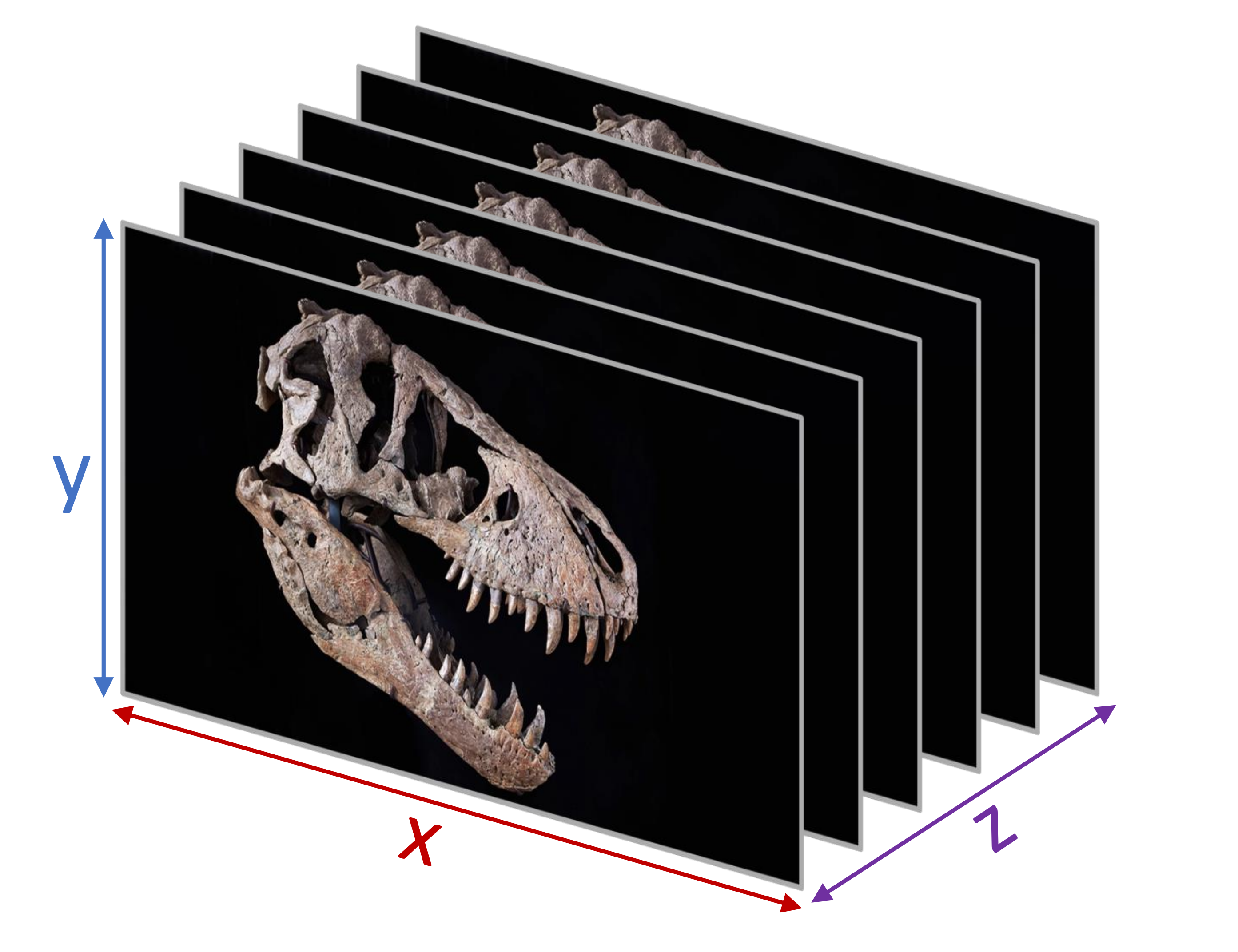
3D data volumes

3D Volumes

CT Scans

- Stack of images to give a data "volume"

- CT scans are good examples of such data, e.g.
  - Medical
  - Palaeontology
  - Porous media



3D Volumes

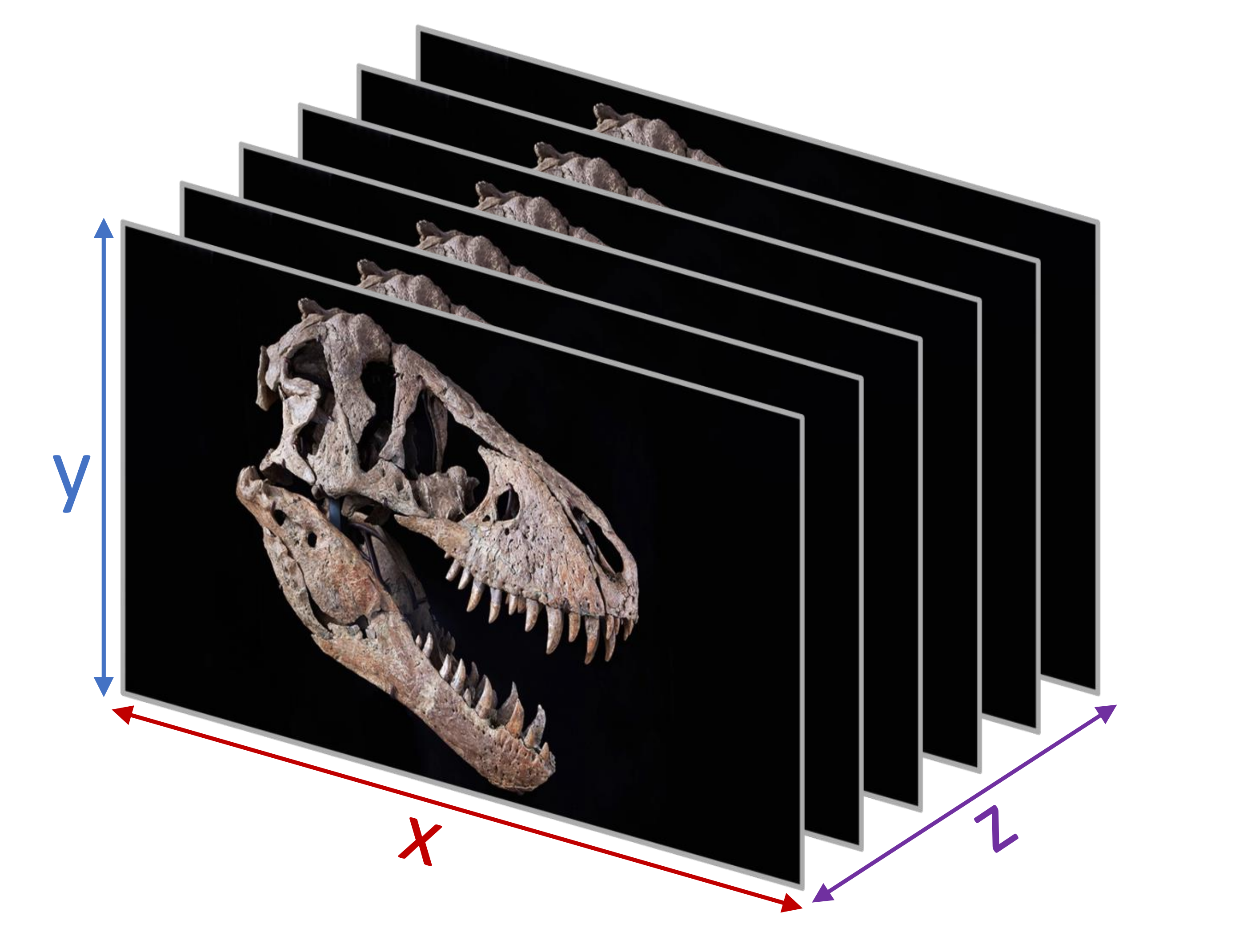
1. Filters

3D filters work in much the same way as 2D filters, but over the 3D volume

i.e., also need to consider *neighbouring images* in the stack as well as neighbouring pixels in the image

a) 3D Gaussian blur

b) 3D Median blur



3D Volumes

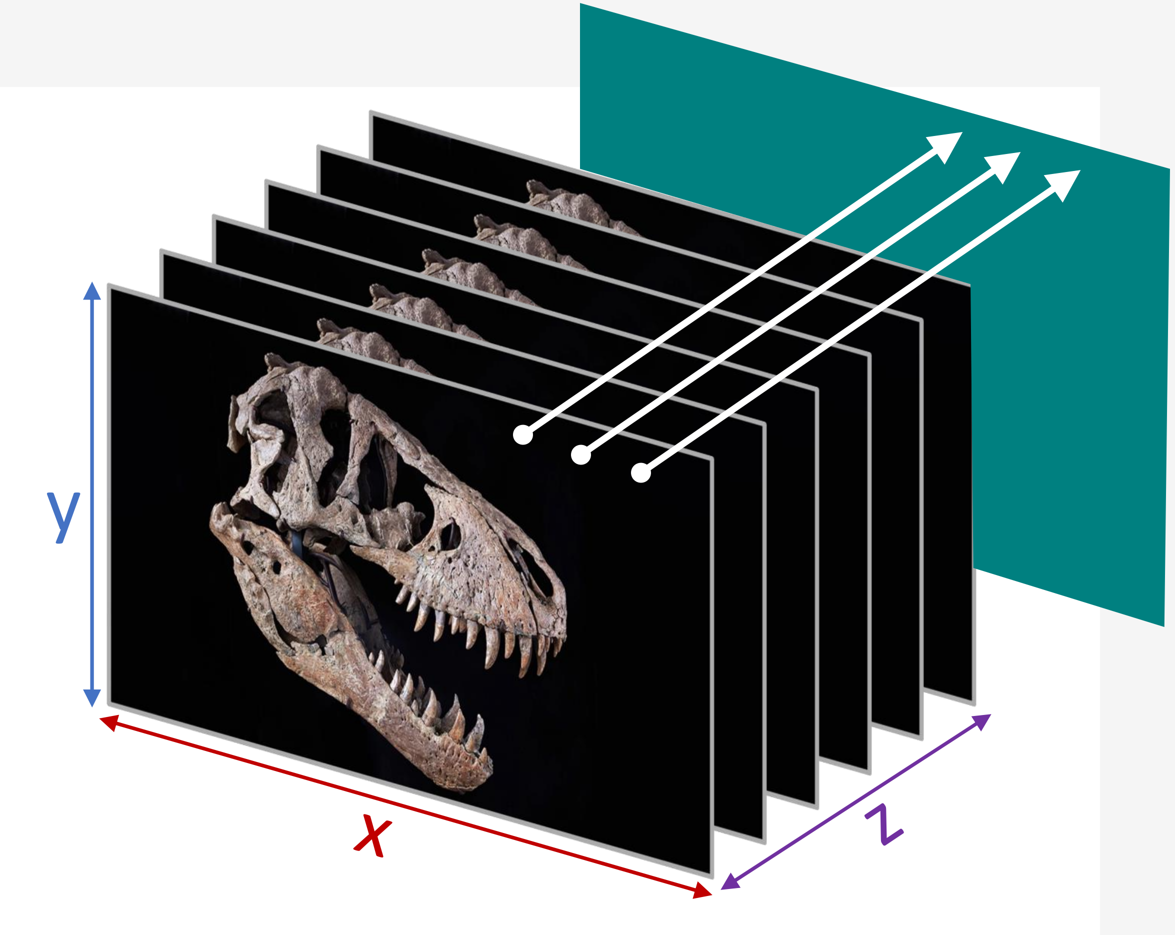
2. Orthographic projections

a) Maximum intensity projection (MIP)

b) Minimum intensity projection (MinIP)

c) Average intensity projection (AIP) using the mean
[**Additional challenge**: AIP using the median; computationally expensive]

Projections look through each image in the stack, and project a feature onto a single image of the same dimensions

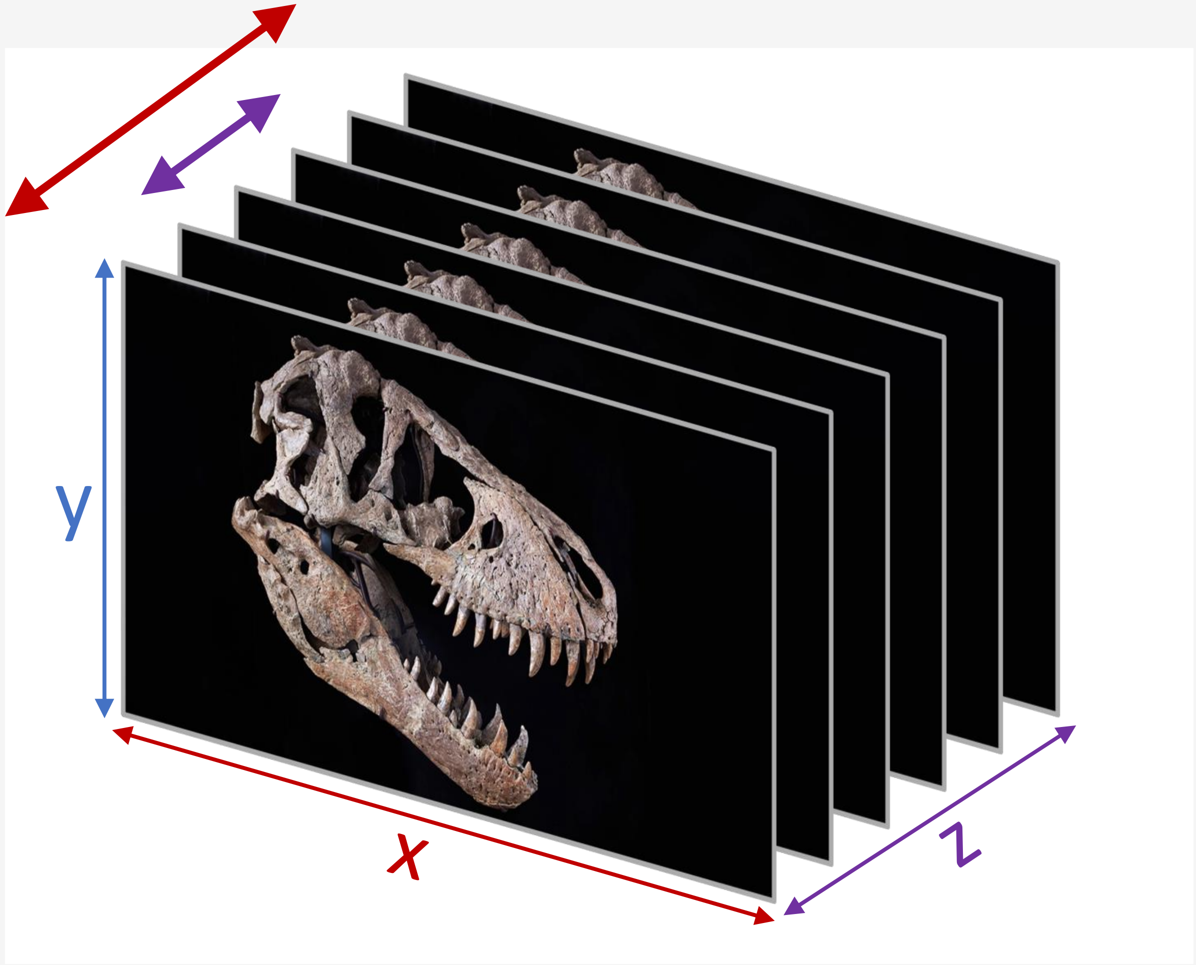


3D Volumes

2. Orthographic projections

Projections can operate on either:

- The whole volume available
- A thin slab (user defined region) in the z-direction



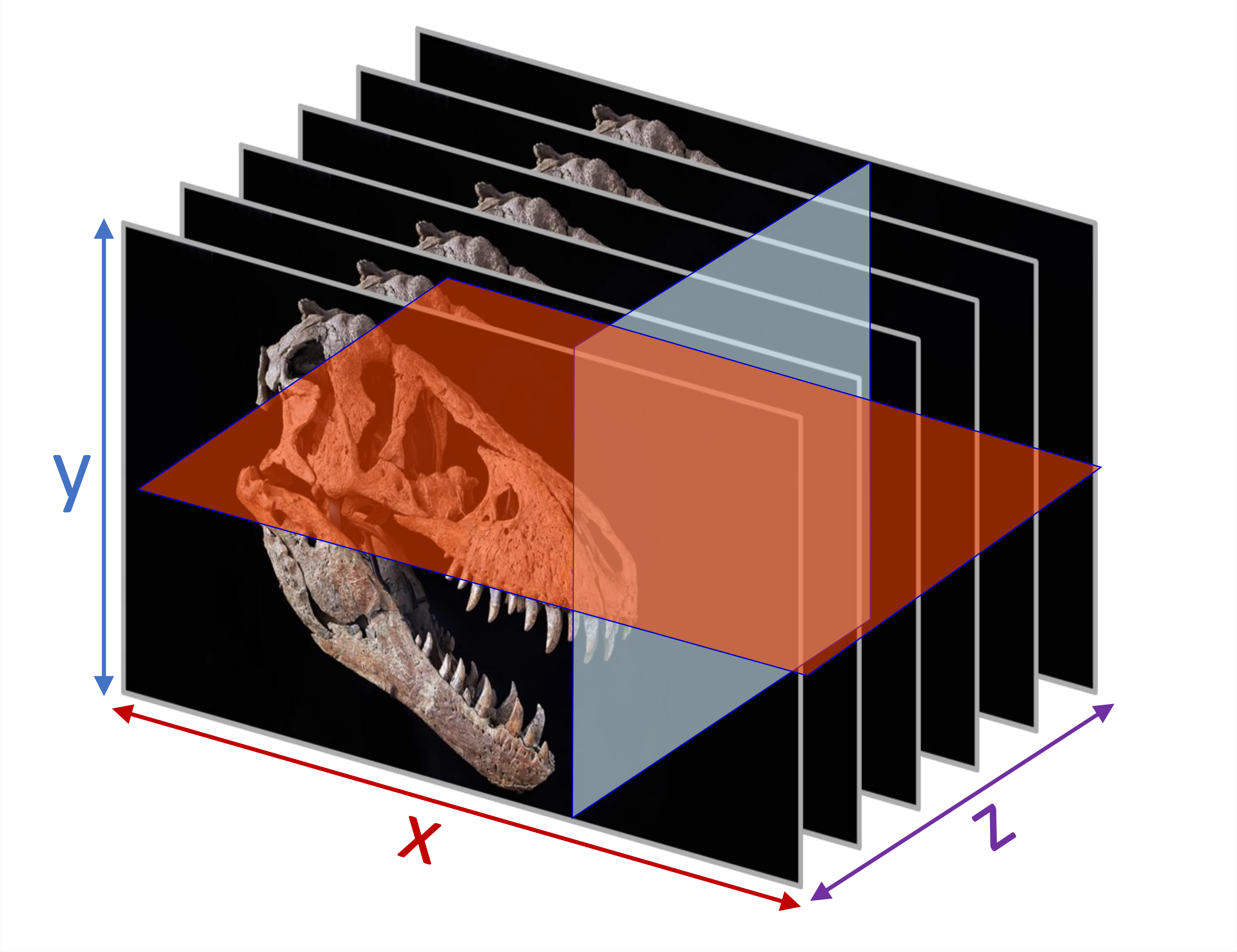
3D Volumes

3. Slicing

Images provided in the *x*-*y* orientation
Your code should be able to output an image in a different plane:

- *y*-*z* (user defined *x*)
- *x*-*z* (user defined *y*)

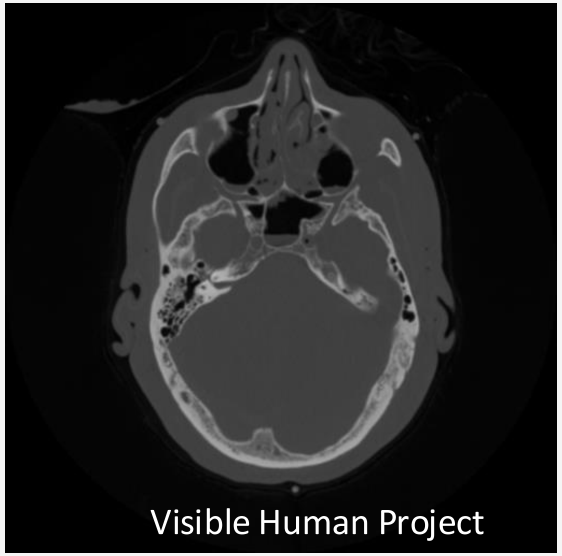
[**note**, index starting from 1 for each dimension; do not use individual image numbers]



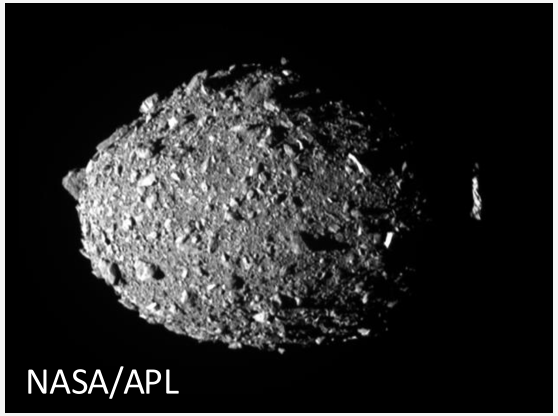
Example images

We have provided you with a set of example images

Please feel free to find other images which can demonstrate the capabilities of your code

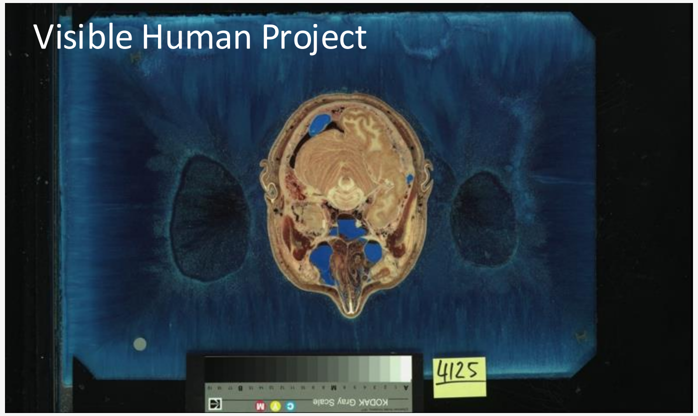


Visible Human Project



NASA/APL

NASA



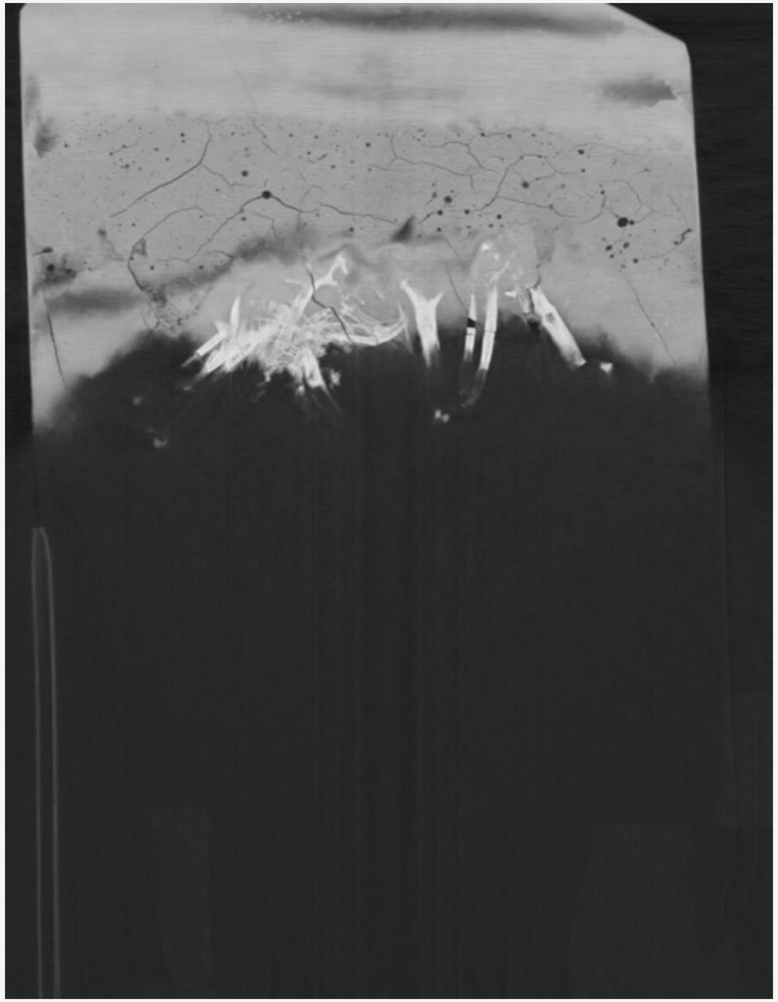
Visible Human Project

Provided 3D Volumes (CT Scans)

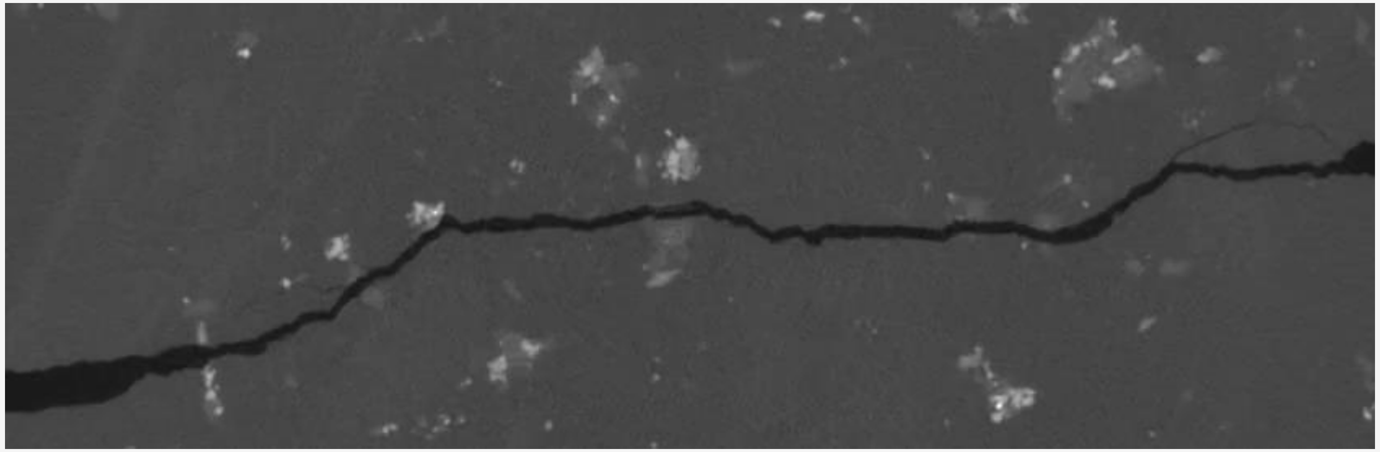
- Fossilized Confuciusornis (prehistoric bird) CT scan
- Fractured granite CT scan

See project description for a list of output images we would like you to provide from these scans

NOTE: DO NOT UPLOAD THESE SCANS TO YOUR GITHUB



https://doi.org/10.6084/m9.figshare.c.1612235_D59.v1



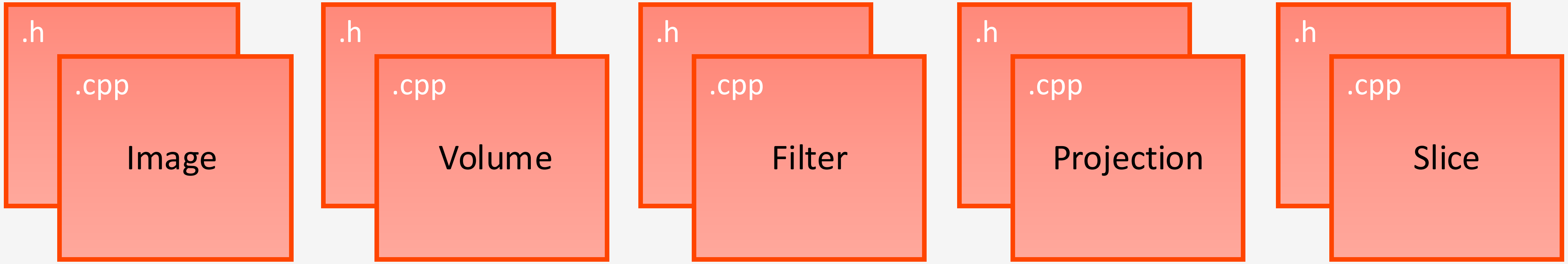
https://doi.org/10.17612/P7QX1X

Code structure

Code structure

A main program file is required with a `main()` function.
This should be run using command line options (see file in repo), no need to write a GUI!

You should have a class for (at least) each of the following:



They should all have a `.cpp` and `.h` file

Add more classes/files if you think they are appropriate; you might consider adding derived classes from some of the classes listed above or deriving some of the above classes from each other.

Add a comment header to each source code file with the names and GitHub usernames of each group member

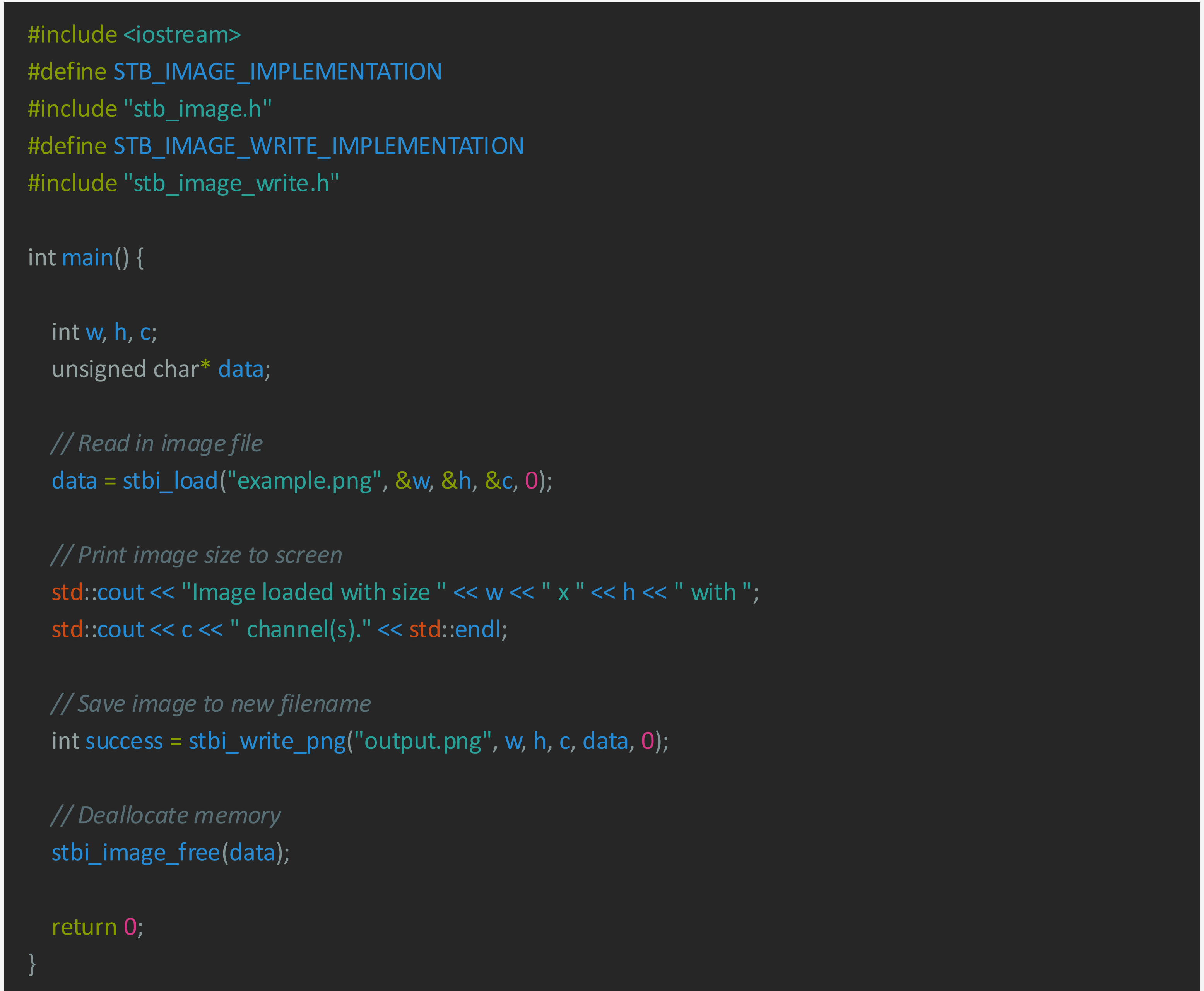
Image reading and writing

stbi image headers

- `stb_image.h` and `stb_image_write.h`
- Provided in your group repositories
- Open-source image I/O header-only library
- Works with a range of image formats, e.g., png, jpeg, gif

- Minimal example in repository showing how to import the headers and how to read and write an image

Image reading and writing with the stbi image library



```cpp
#include <iostream>
#define STB_IMAGE_IMPLEMENTATION
#include "stb_image.h"
#define STB_IMAGE_WRITE_IMPLEMENTATION
#include "stb_image_write.h"

int main() {

    int w, h, c;
    unsigned char* data;

    // Read in image file
    data = stbi_load("example.png", &w, &h, &c, 0);

    // Print image size to screen
    std::cout << "Image loaded with size " << w << " x " << h << " with ";
    std::cout << c << " channel(s)." << std::endl;

    // Save image to new filename
    int success = stbi_write_png("output.png", w, h, c, data, 0);

    // Deallocate memory
    stbi_image_free(data);

    return 0;
}
```

Most useful functions for this project:

- stbi_load
- stbi_write_png
- stbi_image_free

External libraries

- Other than the `stb_image.h` and `stb_image_write.h` headers, **no other external libraries are allowed**
- Standard libraries are fine
  e.g., iostream, string, vector, cmath
- No need to use openMP/threading—I will be testing performance with `OMP_NUM_THREADS=1`
- If you are using a sorting/searching algorithm for one of your filters, **must write it yourselves**!
- Best way to learn programming is to build something from scratch!
- **Do not copy code from the internet or other sources (that includes sharing code between groups)**

Cmake

I have included a CMakeLists.txt file, which should aid compilation for your group.

Instructions are in the README on how to use this.

You will need to add new .cpp files as you create them.

Make sure you code compiles using this file! Check the GitHub actions set up to build and run some small tests.

This will be used to evaluate your project, so if it doesn’t compile, try to fix it.

Project working

Sustainability

- Follow all best-practices you have learned in other courses up to now
- Make sure all code is commented well
- Provide documentation for users to know how to compile and run your code (ok to use documentation packages such as Doxygen)
- Include a readme and license file in your repository
- Use pull requests, code review, issues, etc. on GitHub
- Add a testing framework (unit tests, etc.)

Teamwork

Remember – the main aim of this module and project are for you to learn how to write C++ code collaboratively

- Some of you have more experience than others
- Doesn’t mean we want one or two people writing all the code!
- Everyone should write at least one image filter, but we would prefer you all to write more than that!
- Team effort, work together and support each other

Paired programming can help. Ask each other questions if you are unsure about anything in the code

Use of generative AI tools in this assessment

- As usual, generative AI tools (such as ChatGPT, Copilot, etc.) are permitted during this assessment, **if used responsibly**

- Any code that is generated by AI should be thoroughly **tested and verified**!

- Remember, any use of AI should be **acknowledged/referenced** in your submitted work

- **If in doubt about what is allowed, ask me**

Recommendations

- Start simple
- Build some of the easy filters and parts of the interface first
- Add comments and unit tests as you go
- Build complexity later (e.g. convolution filters, 3D projections, etc.)

Evaluation

Your code will be compiled and executed as part of your evaluation. Code that does not compile or does not output an image with the appropriate filter/projection applied correctly will not be able to score highly. **Write tests to check!**

We will mark the projects based on:

- Functionality and performance (40%)
- Implementation (20%)
- Sustainability (code documentation, commenting, and testing) (15%)
- Short 4-page report (15%)
- Peer evaluation (completed after the project) (10%)

Report

- Maximum 4 pages (single space, 11pt, pdf)
- Code design – discuss how you adopted the key principles learned in this module, e.g. classes, inheritance, polymorphism, etc.
- Performance – how well does your library scale with image/volume size and kernel size
- Breakdown of who worked on which aspects of the project
- Other interesting features or novel implementations

Peer evaluation

Contribution Score	Meaning
35-40*	I believe that this person made an outstanding contribution to the success of this project and worked in an inclusive and supportive manner throughout. Substantial extra credit should be given to this student for their role in the project.
26-34	I believe this person made a major contribution to the success of this project and worked in an inclusive and supportive manner throughout. This person should be given extra credit for their role in this project.
21-25	I believe this person made an above average contribution to the success of this project and worked in an inclusive and supportive manner throughout. This person should be given some additional credit for their contribution.
20	I believe this person made an average contribution to the work on this project. This person worked in an inclusive and supportive manner during the project.
19-15	I believe this person made a reasonable contribution to the success of this project and worked in an inclusive and supportive manner. This person's contribution, however, was not as substantial as other team members, and they should receive less credit for the project.
6-14	I believe this person engaged with the project at all times and worked in an inclusive and supportive way. However, this person made a reduced contribution to the success of this project compared to other team members and should receive substantially less credit for the project than some other team members.
1-5*	I believe this person made very little contribution to the project or their behaviour was detrimental to the project. This person did not work in an inclusive or supportive manner during the project. This person was absent or uncontactable without reason for a large portion of the week. I understand that this rating may result in this student's mark being capped for a lack of contribution and engagement.
0*	I believe this person did not engage with the project in any meaningful way and made no contribution to the project. This person was absent or uncontactable without reason for most of the week. This person's lack of engagement/contribution was reported to the teaching staff during the week. This student should get no credit for the project.

Submission

Upload your code by committing to the main branch on GitHub by:

4pm on Thursday 20th March 2025

Anything included in commits after this time/date will not be assessed.

Your repository should include:

1) Your C++ source code.
2) Documentation (and readme) of how to install and use your program.
3) Your code testing framework.
4) An appropriate licence.
5) Your 4-page PDF report.

IMPERIAL

Any questions?

thomas.davison@imperial.ac.uk