AMME4710: COMPUTER VISION AND IMAGE PROCESSING WEEK 1

Dr. Mitch Bryson
School of Aerospace, Mechanical and Mechatronic Engineering, University of Sydney

Acknowledgement of Country

I would like to acknowledge the Traditional Owners of Australia and recognise their continuing connection to land, water and culture. I am currently on the land of the Cadigal people of the Eora Nation and pay my respects to their Elders, past, present and emerging.

I further acknowledge the Traditional Owners of the country on which you are on and pay respects to their Elders, past, present and future.

Do you have a disability that impacts on your studies?

You may not think of yourself as having a disability but the definition under the **Disability Discrimination Act (1992)** is broad and includes temporary or chronic medical conditions, physical or sensory disabilities, psychological conditions and learning disabilities.

Some types of disabilities we see include:
Anxiety // Arthritis // Asthma // Autism // ADHD
Bipolar disorder // Broken bones // Cancer
Cerebral palsy // Chronic fatigue syndrome
Crohn's disease // Cystic fibrosis // Depression
Diabetes // Dyslexia // Epilepsy // Hearing impairment // Learning disability // Mobility impairment // Multiple sclerosis // Post-traumatic stress // Schizophrenia // Vision impairment

Students needing assistance are advised to register with Disability Services as early as possible. Please contact us or review our website to find out more.

THE UNIVERSITY OF SYDNEY

Inclusion and Disability Services Office
sydney.edu.au/disability
02-8627-8422

About the lecturer: Dr. Mitch Bryson

- B.Eng (aerospace) (2000-2003), PhD in robotics (2004-2008)
- Research Fellow in Robotics (2008-2016)

- In 2022, I teach:
  - MTRX1702 Mechatronics 1 (programming)
  - AMME4710 Computer Vision and Image Processing

- Research Interests:
  - Aerial and marine robotic systems
  - Applications of computer vision in agriculture, forestry, marine science
- Teaching Interests:
  - Active teaching and learning
  - Problem-based Learning

Teaching Team

- Course Tutor:
  - Mr. Ryan Griffiths (Computational imaging and multi-sensor fusion)
  - Research into applying machine learning to automatically calibrate vision systems

- Additional/Guest Lecturers:
  - Dr. Stewart Worrell (Intelligent transportation systems)
  - Dr. Don Dansereau (Computational Imaging Systems)

Course Information

- Lectures:
  - Tuesdays 9-11am Mech Eng LT S213 (and online via Zoom)
  - Used to present new theory and applications, discuss techniques

- Tutorials/Labs:
  - Fridays 1-4pm:
    - Civil PC Lab 365 (weeks 1-7)
    - Mech PC Lab S345 (weeks 8-13)
    - Remote students: online via Zoom
  - Used to practice implementing and using algorithms, designing solutions to computer vision problems
  - Working on structured tutorial activities, assignments and group major design project

Course Website (Canvas)

- Go to:
- https://canvas.sydney.edu.au
  - Then login with your unikey, and click on "AMME4710"

- The website is used:
  - announcements
  - access to course material including lecture slides, videos, tutorial activity resources and assignments
  - Online discussion forum for asking questions outside of class

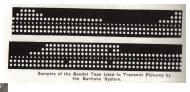
Introduction to Computer Vision and Image Processing

- ***Image Processing*:** The study of analysing and processing image data from a wide variety of sources, typically for improving quality or extracting information, and typically from a signal processing perspective:
  - Image restoration and enhancement, sharpening, filtering
  - Image segmentation, classification and object recognition

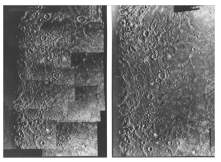
- ***Computer Vision*:** the study of automating (using computers) the process of gaining high-level understanding from images and video:
  - Automating tasks that the human visual system can perform
  - Detect, recognise and track objects in images
  - Estimating 3D structure from images, reconstructing 3D scenes from multiple images
  - Use vision for navigation, control and perception during interactive tasks (e.g. manipulation)

Brief History of Computer Vision and Image Processing

- **1920s**: Digital transmission of images for news via cable across the Atlantic (Bartlane System)
- **1940s, 1950s**: Basic use of digital techniques for interpretation of aerial photographs during war time
- **1960s**: increasing use in computers, DIP in space programs (planetary image reconstruction), remote sensing, beginnings of medical imaging



http://www.hffax.de/history/html/bartlane.html



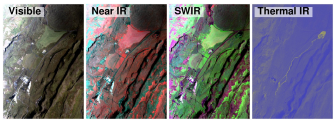
R. Greeley, "Planetary Landscapes", Springer.

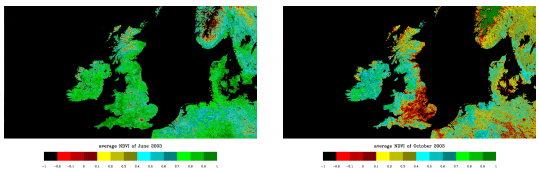
Brief History of Computer Vision and Image Processing

- **1960s**: Generally thought solving low-level vision tasks would be easy (higher-level logic thought to be harder)
  - Prof. Marvin Minsky asks his intern to solve the problem over one summer
- **1970s**: recognition that problem is hard, focus on synthetic images
- **1980s**: 3D from images, mathematical models of vision
- **1990s**: Object detection, facial recognition, large-scale use in industrial automation
- **2000s**: Machine learning, image category recognition, large-scale automated 3D reconstructions
- **2010s**: Dominance of machine learning, deep learning, learning from millions of online images

Remote Sensing

- Satellite multi-spectral and hyperspectral imagery





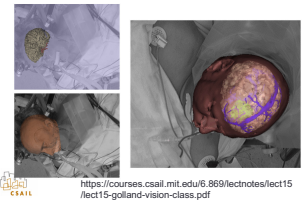
http://www.crazyverse.com/cappelluti/docs/anims/leicester/

Medical Imaging

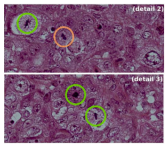
- Medical imaging: X-ray, MRI, CAT

https://commons.wikimedia.org/w/index.php?title=File%3AStructural_MRI_animation.ogv

- Image guided surgery and augmented reality



https://courses.csail.mit.edu/6.869/lectnotes/lect15/lect15-golland-vision-class.pdf



https://link.springer.com/content/pdf/10.1007%2F978-3-642-40763-5_51.pdf

Vision Based Navigation

P. Bristeau, F. Callou, D. Vissiere and N. Petit, "The Navigation and Control technology inside the AR.Drone micro UAV", 18th IFAC World Congress, 2011.

Jakob Engel and Thomas Schops and Daniel Cremers, "LSD-SLAM: Large-Scale Direct Monocular SLAM", ECCV 2014

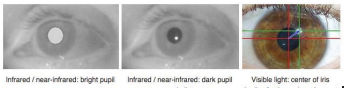
G. Conte and P. Doherty, "An Integrated UAV Navigation System Based on Aerial Image Matching", IEEE Aerospace Conference 2007.

3D Mapping and Structure From Motion

Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski, "Towards Internet-scale Multi-view Stereo", Computer Vision and Pattern Recognition, 2010

Face, Eye and Gaze Tracking

Massimo Gneo et. al., (http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3543256/figure/F1)

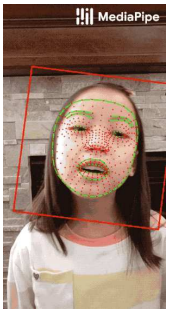


Infrared / near-infrared: bright pupil | Infrared / near-infrared: dark pupil and glint | Visible light: center of iris (red), glint (green), and output vector (blue)

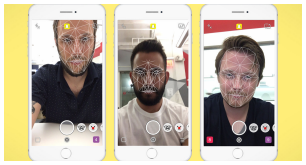
L. Fletcher and A. Zelinsky, "Driver Inattention Detection based on Eye Gaze–Road Event Correlation", International Journal of Robotics Research, Vol. 28, No. 6, June 2009.

Crabb D, Smith N, Rauscher F, Chisholm C, Barbur J, Edgar D, Garway-Heath D (2010). "Exploring Eye Movements in Patients with Glaucoma When Viewing a Driving Scene". PLOS ONE.
DOI:10.1371/journal.pone.0009710.

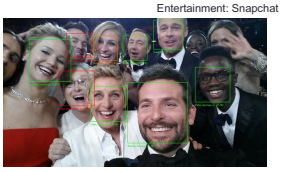
Face, eye and gaze tracking



Facial landmark tracking



Entertainment: Snapchat



Facial detection and recognition

Object detection and tracking

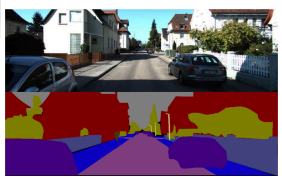
http://www.robots.ox.ac.uk/ActiveVision/Publications/benfold_reid_cvpr2011/benfold_reid_cvpr2011.html

Multi-view Self-supervised Deep Learning for 6D Pose Estimation in the Amazon Picking Challenge, ICRA, 2017

Segment2Regress: Monocular 3D Vehicle Localization in Two Stages, RSS 2019.

Image Classification, Scene Segmentation and Understanding

https://arxiv.org/abs/1411.4555



G. Ros et al. Vision-based Offline-Online Perception Paradigm for Autonomous Driving. WACV 2015.

Why computer vision is complex

- High-dimensional data:
  - Consider a single 1024x768 pixel colour image contains over *3 million* separate pieces of information
  - Computer vision algorithms are typically very computationally intensive
- We take for granted problems in computer vision because our own biological systems have already solved most of these problems:
  - But our understanding of the brain and visual cortex is limited, so it's difficult to replicate the way we do things

Optical Illusions

Optical Illusions

Optical Illusions

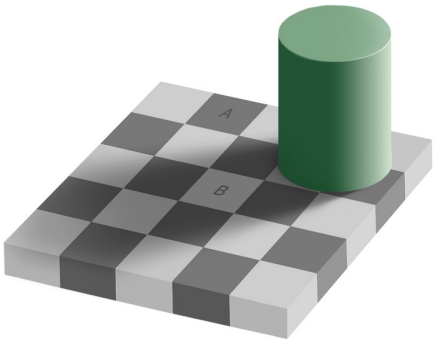
Optical Illusions

Optical Illusions

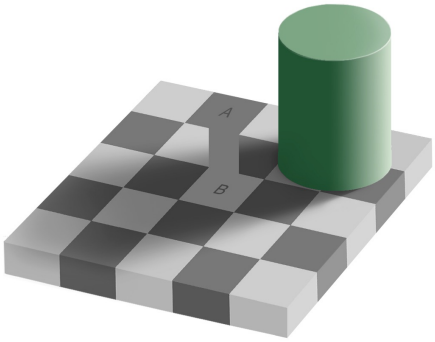
Old Man with a big nose?

Princess?

Optical Illusions



Optical Illusions



Journals and conferences

- Major international conferences on computer vision or image processing:
  - **International Conference on Computer Vision** (http://dblp.uni-trier.de/db/conf/iccv/index.html)
  - **European Conference on Computer Vision** (see 2016: http://www.eccv2016.org/proceedings/)
  - **Computer Vision and Pattern Recognition** (http://ieeexplore.ieee.org/xpl/conhome.jsp?punumber=1000147)
  - **International Conference of Computational Photography** (http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7486620)
- Major computer vision/image processing journals:
  - **International Journal of Computer Vision** (https://link.springer.com/journal/11263)
  - **IEEE Trans. on Image Processing** (http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=83)
  - **Computer Vision and Image Understanding** (https://www.journals.elsevier.com/computer-vision-and-image-understanding/)
  - **IEEE Trans. on Pattern Analysis and Machine Intelligence** (http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=34)

Software Packages

- MATLAB Image Processing Toolbox:
  - Provides implementations of a wide variety of processing methods including filtering, segmentation, object detection, 3D image processing etc.
  - We will use this predominantly throughout the course, however you should also be aware of the existence of other packages, i.e.:
- OpenCV:
  - Probably the most-widely used computer vision software toolkit
  - Library interface for C/C++, binding to functions in python, MATLAB and others
- Fiji/ImageJ:
  - Used extensively in scientific and medical image processing, GUI interface to customisable plug-ins in JAVA
- Many others: e.g. Octave/Scilab (alternatives to MATLAB) each have extensive image processing toolboxes

Course Outline

- **Week 1:** Introduction / Digital Image Fundamentals
- **Week 2:** Introduction to radiometry and colour
- **Week 3:** Image filtering and edge detection
- **Week 4:** Image features, matching, correspondence and detection
- **Week 5:** Stereo imaging, camera calibration, 2D/3D image projective relationships, image-based navigation
- **Week 6:** Image segmentation and clustering
- **Week 7:** Object recognition, image classification, introduction to machine learning
- **Week 8:** Image classification and deep learning in computer vision
- **Week 9:** Computer vision projects, software packages
- **Week 10:** Introduction to structure-from-motion, 3D image-based mapping
- **Week 11:** Advanced applications of computer vision: Face detection and recognition
- **Week 12:** Computer Vision Research Seminar
- **Week 13:** Project Presentations

Assessment

- Tutorials (10%):
  - There are six marked tutorial exercises (weeks 1,2,3,5,6,7)
  - These must be attended and completed during the tutorial
  - Remaining tutorials are used for working on assignments and the major project and attendance at these is also expected
- Assignments (40%):
  - There are two assignments in the course (due week 4 and 8)
  - Assignments will include a combination of written solutions to problems, MATLAB code for computer problems and written reports
  - Assignments are to be submitted using Turnitin and code submitted by email to the tutor
- Final Group Project (50%):
  - Major design task during weeks 9-13
  - Teams will be assessed on a presentation given to the group during week 13 and a final design report due week 13:
    - Group Presentation (20%)
    - Group Final Design Report (30%)

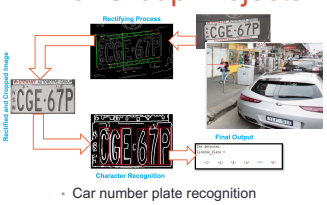
Assessment

- Tutorials (10%):
  - There are six marked tutorial exercises (weeks 1,2,3,5,6,7)
  - These must be attended and completed during the tutorial
  - Remaining tutorials are used for working on assignments and the major project and attendance at these is also expected
- Assignments (40%):
  - There are two assignments in the course (due week 4 and 8)
  - Assignments will include a combination of written solutions to problems, MATLAB code for computer problems and written reports
  - Assignments are to be submitted using Turnitin and code submitted by email to the tutor
- Final Group Project (50%):
  - Major design task during weeks 9-13
  - Teams will be assessed on a presentation given to the group during week 13 and a final design report due week 13:
    - Group Presentation (20%)
    - Group Final Design Report (30%)

Assessment

- Tutorials (10%):
  - There are six marked tutorial exercises (weeks 1,2,3,5,6,7)
  - These must be attended and completed during the tutorial
  - Remaining tutorials are used for working on assignments and the major project and attendance at these is also expected
- Assignments (40%):
  - There are two assignments in the course (due week 4 and 8)
  - Assignments will include a combination of written solutions to problems, MATLAB code for computer problems and written reports
  - Assignments are to be submitted using Turnitin and code submitted by email to the tutor
- Final Group Project (50%):
  - Major design task during weeks 9-13
  - Teams will be assessed on a presentation given to the group during week 13 and a final design report due week 13:
    - Group Presentation (20%)
    - Group Final Design Report (30%)

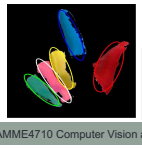
Final Group Projects

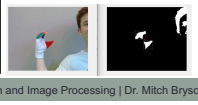


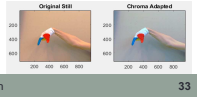
- Car number plate recognition

- Blackjack game tracker

- Sign language translation

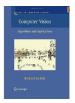


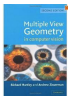




Textbooks and Resources

- There is no specific textbook for the course, however these books/notes are a great source of information:
  - R. Szeliski, “Computer Vision: Algorithms and Applications”, Springer, 2010.
  - D. A. Forsyth and J. Ponce, “Computer Vision - A Modern Approach”, Prentice Hall, 2002
  - R.C. Gonzalez and R.E. Woods, “Digital Image Processing”, Prentice Hall, 2008
  - R. I. Hartley and A. Zisserman, “Multiple View Geometry in Computer Vision”, Cambridge University Press, 2000



What is expected from you?

- Read the unit outline (in Canvas)
- Familiarise yourself with Academic Honesty responsibilities and policies:
  - https://www.sydney.edu.au/students/academic-integrity.html
- Attend both lectures and tutorials
  - Spend time in marked tutorials working on tutorial activities
  - Spend non-marked tutorial time working on assignments and major projects
- Spend time outside of contact hours:
  - The average student is expected to spend 2-3 hours per week working on this subject, outside of contact hours
  - Working on assignments, working on major design project and doing your own research into the latest research/developments to solving problems that spark your interest

Thursday’s Tutorial

- The tutorial is on this Thursday (4th August) and you will be marked on the activity during the tutorial
- The tutorial exercise will provide a chance to get introduced to the MATLAB Image Processing Toolbox, learn about basic operations with images, and solve some problems using image histograms and thresholding
- The tutorial sheet is available on the Canvas: you should have a quick look over the sheet before the tutorial

5 minute break

Introduction to Image Processing

- Image processing is applied in a wide variety of fields to perform tasks such as:
  - **Image Enhancement or Restoration:** improve the visual quality of an image by enhancing the contrast/visual distinctiveness for human interpretation, remove noise or degradation
  - **Image Segmentation:** break an image down into spatially-meaningful pieces, extract relevant information from background
  - **Image Classification and Recognition:** analyse the content of an image to label it or place it into the context of one of many classes, recognise and detect specific things/objects in images

- In this lecture and during the week 1 tutorial, we will:
  - Examine the basics of image representation and operations
  - Explore image histograms and contrast enhancement
  - Explore the basics of image thresholding
  - Use the MATLAB Image Processing Toolbox and explore basic commands

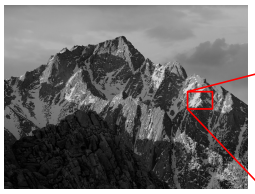
Introduction to Image Processing

- Image processing is applied in a wide variety of fields to perform tasks such as:
  - **Image Enhancement or Restoration:** improve the visual quality of an image by enhancing the contrast/visual distinctiveness for human interpretation, remove noise or degradation
  - **Image Segmentation:** break an image down into spatially-meaningful pieces, extract relevant information from background
  - **Image Classification and Recognition:** analyse the content of an image to label it or place it into the context of one of many classes, recognise and detect specific things/objects in images

- In this lecture and during the week 1 tutorial, we will:
  - Examine the basics of image representation and operations
  - Explore image histograms and contrast enhancement
  - Explore the basics of image thresholding
  - Use the MATLAB Image Processing Toolbox and explore basic commands

A Digital Image

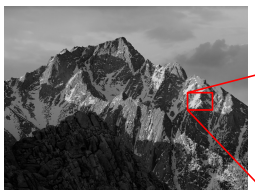
- A large array of discrete points (typically a rectangular grid), each point containing a discrete number representing (typically) the brightness of each point
  - Data organised in this way is referred to as a **raster**
  - Points referred to as **pixels**, and these usually take on integer values with finite range
  - Group of pixels surrounding a pixel is referred to it's **neighborhood**



218	192	168	154	138	119
189	171	157	136	115	104
172	152	136	122	109	102
157	137	121	108	98	88
130	119	108	97	87	74
111	106	98	88	81	69
108	98	90	81	75	67
94	82	74	71	71	72
79	68	68	70	70	73

A Digital Image

- A large array of discrete points (typically a rectangular grid), each point containing a discrete number representing (typically) the brightness of each point
  - Data organised in this way is referred to as a **raster**
  - Points referred to as **pixels**, and these usually take on integer values with finite range
  - Group of pixels surrounding a pixel is referred to it's **neighborhood**

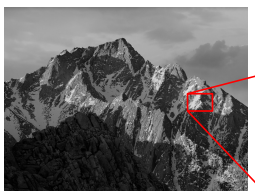


Pixel

218	192	168	154	138	119
189	171	157	136	115	104
172	152	136	122	109	102
157	137	121	108	98	88
130	119	108	97	87	74
111	106	98	88	81	69
108	98	90	81	75	67
94	82	74	71	71	72
79	68	68	70	70	73

A Digital Image

- A large array of discrete points (typically a rectangular grid), each point containing a discrete number representing (typically) the brightness of each point
  - Data organised in this way is referred to as a **raster**
  - Points referred to as **pixels**, and these usually take on integer values with finite range
  - Group of pixels surrounding a pixel is referred to it's **neighborhood**

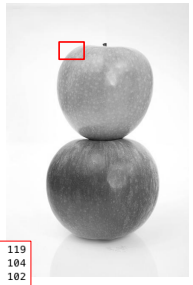


Neighborhood (5x5)

218	192	168	154	138	119
189	171	157	136	115	104
172	152	136	122	109	102
157	137	121	108	98	88
130	119	108	97	87	74
111	106	98	88	81	69
108	98	90	81	75	67
94	82	74	71	71	72
79	68	68	70	70	73

A Digital Image

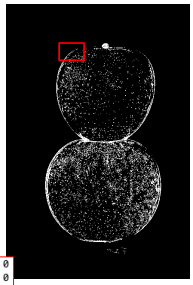
- Types of Images:
  - **Grayscale:** shades of gray from black (0) to white (255 for 8-bit images, 65535 for 16-bit images)
  - **Binary:** values can only be white or black, true or false, 1 or 0
  - **True Colour or RGB:** contains three separate channels: Red, Green and Blue. The balance of values in each channel create different colours



218	192	168	154	138	119
189	171	157	136	115	104
172	152	136	122	109	102
157	137	121	108	98	88
130	119	108	97	87	74
111	106	98	88	81	69
108	98	90	81	75	67
94	82	74	71	71	72
79	68	68	70	70	73

A Digital Image

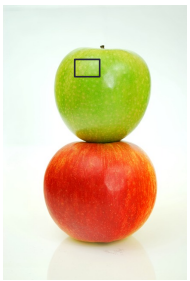
- Types of Images:
  - **Grayscale:** shades of gray from black (0) to white (255 for 8-bit images, 65535 for 16-bit images)
  - **Binary:** values can only be white or black, true or false, 1 or 0
  - **True Colour or RGB:** contains three separate channels: Red, Green and Blue. The balance of values in each channel create different colours



0	0	0	0	0	1	1	0
0	0	0	0	1	1	0	0
0	0	0	0	1	1	0	0
0	0	0	1	1	0	1	1
0	0	1	1	0	1	1	1
0	0	1	1	0	1	0	0
0	1	1	0	1	0	0	0

A Digital Image

- Types of Images:
  - **Grayscale:** shades of gray from black (0) to white (255 for 8-bit images, 65535 for 16-bit images)
  - **Binary:** values can only be white or black, true or false, 1 or 0
  - **True Colour or RGB:** contains three separate channels: Red, Green and Blue. The balance of values in each channel create different colours



191	194	202	198	193	195	189	178	174	
185	201	195	191	200	191	181	188	179	
193	194	184	196	189	173	176	189	173	
189	196	189	194	177	178	182	174	177	
190	194	187	175	175	184	192	187	187	
186	183	177	174	174	197	197	192	194	
192	181	177	181	180	201	209	204	198	

210	213	224	220	217	219	215	199	205	
204	220	217	214	225	217	211	219	213	
213	214	207	219	215	203	209	221	207	
210	220	214	220	206	200	214	205	210	
214	220	215	206	205	210	217	214	216	
212	211	208	207	205	222	220	218	221	
217	209	209	214	210	226	232	230	225	

69	71	81	75	71	73	66	47	50	
60	76	74	72	81	66	55	63	56	
62	63	57	76	66	43	44	58	48	
53	64	58	69	52	50	53	49	57	
54	61	53	42	45	61	74	73	74	
53	49	42	38	42	79	86	83	88	
63	46	40	47	52	83	98	97	92	

MATLAB Image Processing Toolbox

Image Formats

- Image data in memory is typically in an array or matrix data structure
- When stored on disk, various container formats are commonly used
- Most formats contain a header followed by raster data that may be:
  - **Uncompressed:** an ordered array of values representing the value at each pixel (examples: BMP, PPM, PGM)
  - **Lossless Compression:** compresses image data without losing information (original raster data can be recovered exactly) (Examples: PNG, TIFF, GIF)
  - **Lossy Compression:** image data is compressed in a way in which the original data cannot be recovered exactly. Typically smaller file size than lossless compression. (Examples: JPEG, TIFF, BPG)
- Different formats support differing bit depths (number of bits/precision used for each pixel) i.e. 1,2,4,8,16,32 bit integers and 16 or 32 bit floats
- Some image formats (i.e. GIF, TIFF, BMP) use a palette; an array of data immediately after the header which specifies a map from a low-depth index to a high-depth colour space

Image Formats

- Image data in memory is typically in an array or matrix data structure
- When stored on disk, various container formats are commonly used
- Most formats contain a header followed by raster data that may be:
  - **Uncompressed:** an ordered array of values representing the value at each pixel (examples: BMP, PPM, PGM)
  - **Lossless Compression:** compresses image data without losing information (original raster data can be recovered exactly) (Examples: PNG, TIFF, GIF)
  - **Lossy Compression:** image data is compressed in a way in which the original data cannot be recovered exactly. Typically smaller file size than lossless compression. (Examples: JPEG, TIFF, BPG)
- Different formats support differing bit depths (number of bits/precision used for each pixel) i.e. 1,2,4,8,16,32 bit integers and 16 or 32 bit floats
- Some image formats (i.e. GIF, TIFF, BMP) use a palette; an array of data immediately after the header which specifies a map from a low-depth index to a high-depth colour space

Arithmetic operations on images

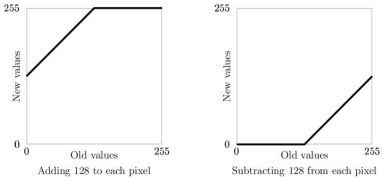
- Applying an arithmetic operation on an image constitutes applying a function y = f(x) to each pixel's intensity value x, arriving at a new image
- Simple operations are straight forward, but must observe that image values are discrete and have finite lower and upper bounds

Original Image | x + 50 | 2*x - 50

Arithmetic operations on images

- Applying an arithmetic operation on an image constitutes applying a function y = f(x) to each pixel's intensity value x, arriving at a new image
- Simple operations are straight forward, but must observe that image values are discrete and have finite lower and upper bounds

Addition and Subtraction

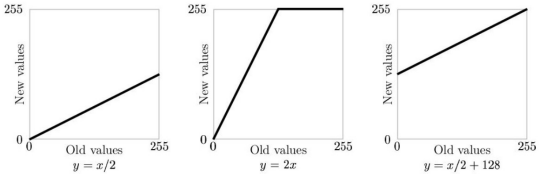


Adding 128 to each pixel | Subtracting 128 from each pixel

Arithmetic operations on images

- Applying an arithmetic operation on an image constitutes applying a function y = f(x) to each pixel's intensity value x, arriving at a new image
- Simple operations are straight forward, but must observe that image values are discrete and have finite lower and upper bounds

Multiplication and Division



Arithmetic operations on images

- Applying an arithmetic operation on an image constitutes applying a function y = f(x) to each pixel's intensity value x, arriving at a new image
- Simple operations are straight forward, but must observe that image values are discrete and have finite lower and upper bounds

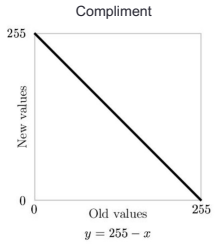


Image Histograms

- An image histogram is a graph of image value vs. frequency of occurrence for a specified image
- Histograms are a useful tool for visualising the dynamic range of an image, and can be used to assist thresholding and as feature for high-level vision algorithms

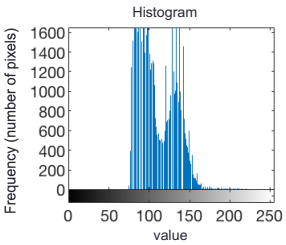
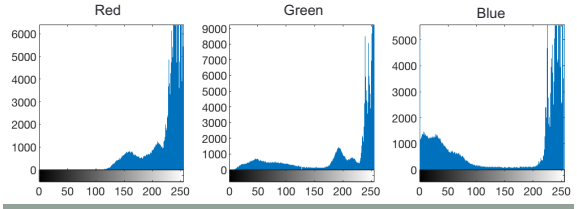


Image Histograms

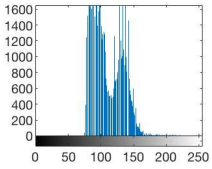
- Histograms for multi-channel images (i.e. colour) usually entail a separate histogram for each channel



Contrast Adjustment

- Contrast adjustment involves applying an intensity transformation y = f(x) (typically monotonic) to pixel values in order to improve the visual qualities of the image (for either human or machine interpretation)
- A simple scheme is to apply a linear function, with variable offset and scaling to improve appearance:

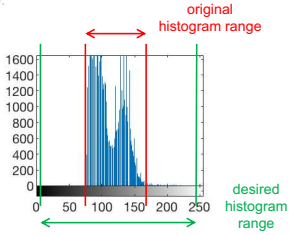
original image



Contrast Adjustment

- Contrast adjustment involves applying an intensity transformation y = f(x) (typically monotonic) to pixel values in order to improve the visual qualities of the image (for either human or machine interpretation)
- A simple scheme is to apply a linear function, with variable offset and scaling to improve appearance:

original image

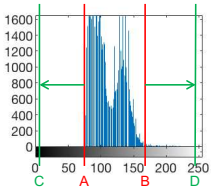


Contrast Adjustment

- Contrast adjustment involves applying an intensity transformation y = f(x) (typically monotonic) to pixel values in order to improve the visual qualities of the image (for either human or machine interpretation)
- A simple scheme is to apply a linear function, with variable offset and scaling to improve appearance:

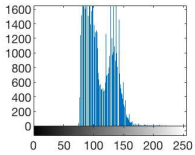
$$f(x) = \frac{d-c}{b-a}(x-a)+c$$

original image

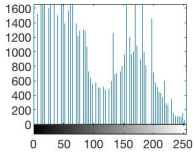


Contrast Adjustment

original image



contrast stretched image

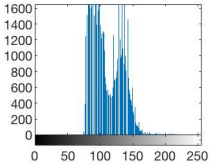


- When applying intensity transformations from one fixed precision to another (i.e. 8-bit image to 8-bit image) it important to keep in mind that these operations do not increase depth-resolution of the data itself:
  - in fact they often reduce the actual amount of information present

Histogram Equalisation

- Histogram Equalisation applies a contrast-enhancing, non-linear intensity transformation to an image with the aim of improving the value distance between image intensities present
- The aim of histogram equalisation is to create a non-linear transformation f(x) that "flattens" the resulting image histogram

original image

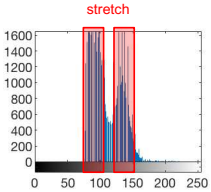


Histogram Equalisation

- Histogram Equalisation applies a contrast-enhancing, non-linear intensity transformation to an image with the aim of improving the value distance between image intensities present
- The aim of histogram equalisation is to create a non-linear transformation f(x) that "flattens" the resulting image histogram

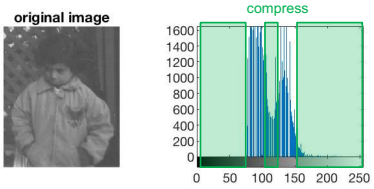
original image

stretch



Histogram Equalisation

- Histogram Equalisation applies a contrast-enhancing, non-linear intensity transformation to an image with the aim of improving the value distance between image intensities present
- The aim of histogram equalisation is to create a non-linear transformation f(x) that "flattens" the resulting image histogram

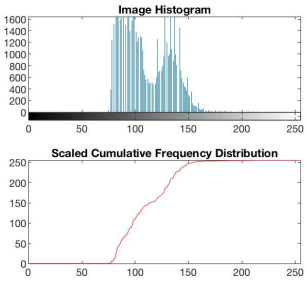


Histogram Equalisation

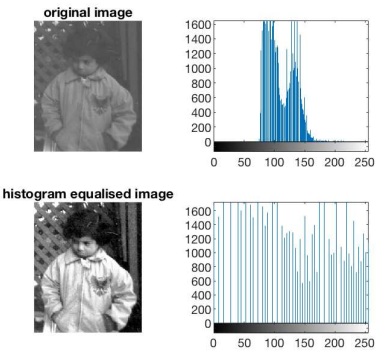
- This can be achieved by making f(x) equal to a scaled form of the cumulative frequency distribution of the original image histogram:

$$f(x) = \frac{L}{N}\sum_{i=0}^{x} h(x)$$

where L is the maximum image value (i.e. 255 for 8-bit), N is the total number of image pixels and h(x) is the frequency value of the histogram for intensity x

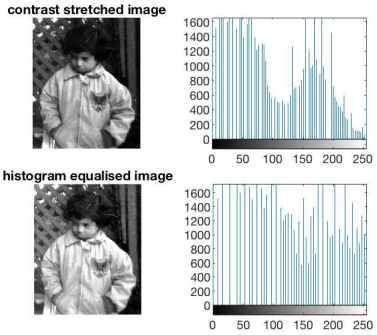


Histogram Equalisation



Histogram Equalisation

- Comparing to a contrast enhancement using a linear transformation, the histogram equalised image has almost equal contrast across all image values (bright and dark)



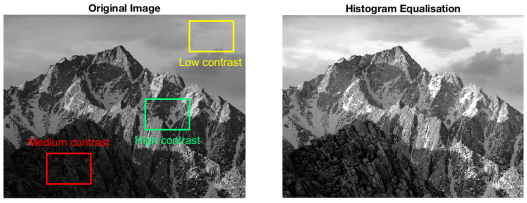
Spatially Adaptive Histogram Equalisation

- Histogram equalisation doesn't work particularly well when contrast in the image varies spatially

Spatially Adaptive Histogram Equalisation

- **Adaptive histogram equalisation** is a method that computes a local (per-pixel) intensity transformation that only considers the histogram in a local window around each pixel
  - In practice, a typical implementation will compute the intensity transformation at a number of discrete locations less than the number of pixels, and linearly interpolate the transformation parameters between these locations for each pixel

Spatially Adaptive Histogram Equalisation



- **Adaptive histogram equalisation** is a method that computes a local (per-pixel) intensity transformation that only considers the histogram in a local window around each pixel
  - In practice, a typical implementation will compute the intensity transformation at a number of discrete locations less than the number of pixels, and linearly interpolate the transformation parameters between these locations for each pixel

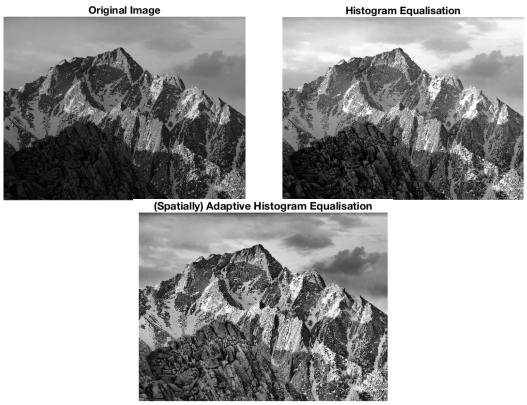


Image Thresholding

- Image thresholding is the process of assigning pixels into categories based on whether their value is above or below a threshold value
- Thresholding is a basic algorithm for segmenting an image (i.e. foreground/background extraction)

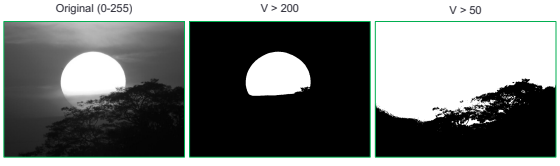
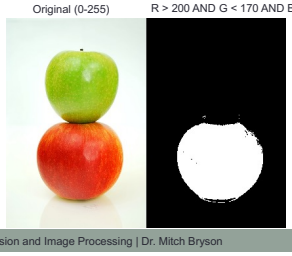


Image Thresholding

- Image thresholding is the process of assigning pixels into categories based on whether their value is above or below a threshold value
- Thresholding is a basic algorithm for segmenting an image (i.e. foreground/background extraction)
- Multiple thresholds can be applied by using per-pixel AND/OR/NOT operations

Original (0-255) R > 200 AND G < 170 AND B < 128



Automatic thresholding and Otsu's Method

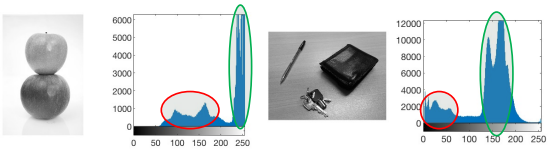
- Consider an image processing task for segmenting foreground/ background objects: threshold may vary depending on actual image conditions

Automatic thresholding and Otsu's Method

- Consider an image processing task for segmenting foreground/ background objects: threshold may vary depending on actual image conditions
- Assuming situations where foreground pixels have similar intensity and background pixels have similar intensity, the resulting image histogram should be roughly bi-modal
- In this case, the histograms indicate appropriate locations to threshold these images based on 'splitting' the two modes

Automatic thresholding and Otsu's Method

- Consider an image processing task for segmenting foreground/ background objects: threshold may vary depending on actual image conditions
- Assuming situations where foreground pixels have similar intensity and background pixels have similar intensity, the resulting image histogram should be roughly bi-modal
- In this case, the histograms indicate appropriate locations to threshold these images based on 'splitting' the two modes



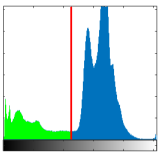
Automatic thresholding and Otsu's Method

- Otsu's method finds a threshold value that splits the histogram into two classes [0,1]
- The threshold (t) is selected that minimises the intra-class variance:

$$\sigma^2_{intra}(t) = \omega_0(t)\sigma^2_0(t) + \omega_1(t)\sigma^2_1(t)$$ where:

$\sigma^2_0(t)$ variance of intensity values < t $\omega_0(t) = \frac{1}{N}\sum_{i=0}^{t} h(x)$ proportion of intensity values < t

$\sigma^2_1(t)$ variance of intensity values > t $\omega_1(t) = \frac{1}{N}\sum_{i=t}^{L-1} h(x)$ proportion of intensity values > t



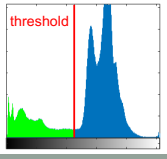
Automatic thresholding and Otsu's Method

- Otsu's method finds a threshold value that splits the histogram into two classes [0,1]
- The threshold (t) is selected that minimises the intra-class variance:

$$\sigma^2_{intra}(t) = \omega_0(t)\sigma^2_0(t) + \omega_1(t)\sigma^2_1(t)$$ where:

$\sigma^2_0(t)$ variance of intensity values < t $\omega_0(t) = \frac{1}{N}\sum_{i=0}^{t} h(x)$ proportion of intensity values < t

$\sigma^2_1(t)$ variance of intensity values > t $\omega_1(t) = \frac{1}{N}\sum_{i=t}^{L-1} h(x)$ proportion of intensity values > t



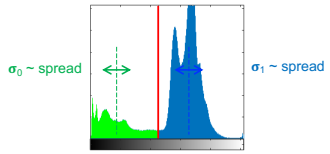
Automatic thresholding and Otsu's Method

- Otsu's method finds a threshold value that splits the histogram into two classes [0,1]
- The threshold (t) is selected that minimises the intra-class variance:

$$\sigma^2_{intra}(t) = \omega_0(t)\sigma^2_0(t) + \omega_1(t)\sigma^2_1(t)$$ where:

$\sigma^2_0(t)$ variance of intensity values < t $\omega_0(t) = \frac{1}{N}\sum_{i=0}^{t} h(x)$ proportion of intensity values < t

$\sigma^2_1(t)$ variance of intensity values > t $\omega_1(t) = \frac{1}{N}\sum_{i=t}^{L-1} h(x)$ proportion of intensity values > t



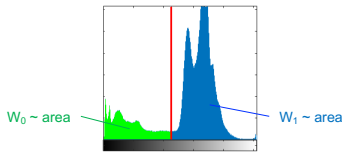
Automatic thresholding and Otsu's Method

- Otsu's method finds a threshold value that splits the histogram into two classes [0,1]
- The threshold (t) is selected that minimises the intra-class variance:

$$\sigma^2_{intra}(t) = \omega_0(t)\sigma^2_0(t) + \omega_1(t)\sigma^2_1(t)$$ where:

$\sigma^2_0(t)$ variance of intensity values < t $\omega_0(t) = \frac{1}{N}\sum_{i=0}^{t} h(x)$ proportion of intensity values < t

$\sigma^2_1(t)$ variance of intensity values > t $\omega_1(t) = \frac{1}{N}\sum_{i=t}^{L-1} h(x)$ proportion of intensity values > t



Automatic thresholding and Otsu's Method

- Otsu's method finds a threshold value that splits the histogram into two classes [0,1]
- The threshold (t) is selected that minimises the intra-class variance:

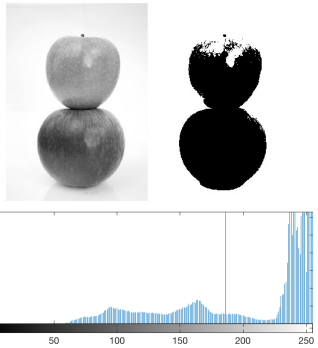
$$\sigma^2_{intra}(t) = \omega_0(t)\sigma^2_0(t) + \omega_1(t)\sigma^2_1(t)$$ where:

$\sigma^2_0(t)$ variance of intensity values < t $\omega_0(t) = \frac{1}{N}\sum_{i=0}^{t} h(x)$ proportion of intensity values < t

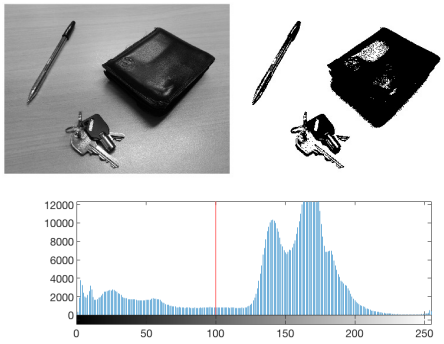
$\sigma^2_1(t)$ variance of intensity values > t $\omega_1(t) = \frac{1}{N}\sum_{i=t}^{L-1} h(x)$ proportion of intensity values > t

- Minimisation of the intra-class variance is equivalent to maximising the inter-class variance $\sigma^2_{inter}(t) = \sigma^2 - \sigma^2_{intra}(t)$ where σ^2 is the variance of all image intensities, which is easier to compute in practice
- The algorithm is implemented by computing the inter-class variance for all possible thresholds, and finding the threshold with maximum inter-class variance

Automatic thresholding and Otsu's Method



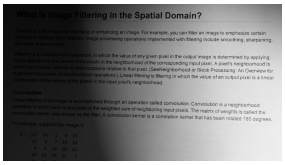
Automatic thresholding and Otsu's Method



Adaptive thresholding

- In the same way as with histogram equalisation, Otsu's method for thresholding can be made spatially adaptive by making a separate threshold for each pixel based on the histogram of it's local neighborhood:

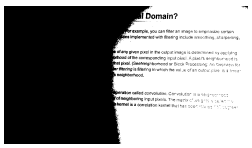
Original image



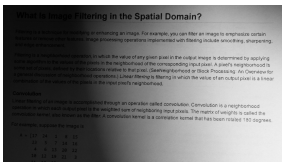
Adaptive thresholding

- In the same way as with histogram equalisation, Otsu's method for thresholding can be made spatially adaptive by making a separate threshold for each pixel based on the histogram of it's local neighborhood:

Single threshold value



Original image

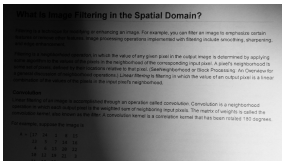


Adaptive thresholding

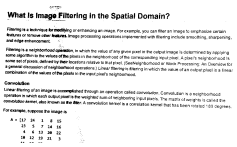
- In the same way as with histogram equalisation, Otsu's method for thresholding can be made spatially adaptive by making a separate threshold for each pixel based on the histogram of it's local neighborhood:

Single threshold value

Original image



Adaptive threshold



Further Reading and Next Week

- References:
  - R.C. Gonzalez and R.E. Woods, "Digital Image Processing", Prentice Hall, 2008
  - N. Otsu (1979). "A threshold selection method from gray-level histograms", IEEE Trans. Sys., Man., Cyber. 9 (1): 62–66
- Next Week: Introduction to radiometry, colour and colour image processing