Computer Vision:

General:

* Where deep learning has had the most success
* Deals with analysis of image instead of image processing [input🡪image, output🡪image interpretation]
* **Goals:** edge and feature detection, classification, pattern recognition and detect objects and establish depth in virtual world
* Issues with identification, object classification, verification and detection

Applications:

* Automated vehicles and drones + facial recognition [usually implemented by **law enforcing agencies**] and AR software [**smartphones and tablets nowadays**]
* **Inspection in manufacturing production lines** + organising information
* Driverless cars [CV & AI & ML]. **Waymo of Google** [optimise transportation]. With deep neural networks car could potentially navigate busy streets. Use ML to filter out noise for example in sensor data due to raindrops. Also, **Tesla vision**
* Financial services sector: **passports and ID and banks verify document authenticity using CV software**. Require robust software to work on poor quality images too [image fault resolution (distortion and poor lighting for example)]. Mobile apps **automating client identification eliminate human errors** [quick and secure execution of transactions]
* Healthcare [**diagnose disease**]. Used alongside sensors
* AWS DeepLens opensource system

Lecture material for CW:

* Computational models 🡪 relationships between physics and intrinsic characteristics. Establish relationship between pixel intensity and image properties [scene, illumination and sensor models]
* Photometric image formation: point and area light sources / BRDF [*f*(incident, reflected, wavelength, output/input power)]
* Diffuse vs specular [mirror like] reflection models
* CCD [high quality image data] and CMOS [smart phone cameras]
* From bayer to quad bayer colour filter array
* Colour models: additive and subtractive

Lecture 1:

* Filtering [finished with structure adaptive filtering]
  + Noise: make images clear. Might apply local averaging
  + A picture containing graphical user interface

    Description automatically generated
  + Text

    Description automatically generated
  + Timeline

    Description automatically generated
  + Might have issues with smoothing [not adequate]. Neighbourhood trade off (choice)
* Convolution:
  + Use convolution (flip mask and slide on signal on another and multiply beware if symmetric no need to flip)
  + Word

    Description automatically generated with low confidence
  + Text, whiteboard

    Description automatically generated
  + Text

    Description automatically generated
  + Can consider edges using padding
  + 1: odd numbers [if pixel is centred] [alignment, cannot with even number of rows and cols]. If image not to be shrunk, then apply padding
  + Gaussian kernel as a convolution mask [for line detection] 🡪 different weights [more weights for the nearest neighbours]
  + Finding differences between rows to highlight horizontal edges. Same idea for vertical
  + Convolution: commutativity, associativity, distributivity and differentiation
  + Computational complexity stuff (1 x k and k x 1 to reduce computational complexity). If kernel can be reduced to the convolution of 2 simpler ones

Lecture 2:

* Apply edge detection masks individually. Apply both masks and get magnitude and direction of gradient.
* Low probability for uniform areas
* A picture containing application

  Description automatically generated
* Canny algorithm [good detection (low probability of false positives and negatives), good localisation and single response]
* **Step 1**: gaussian filtering 🡪 suppress noise [derivative of gaussian along each direction and then convolve]
* **Step 2**: NMS 🡪 based on pixel gradient magnitude only at location of local maximum
* **Step 3**: Hysteresis thresholding
* A picture containing chart

  Description automatically generated
* Gaussian kernel [large sigma large scale edges, small sigma finer] need trade-off
* Laplace operator: 1 single mask and check for 0 crossings / Laplacian of gaussian
* Text

  Description automatically generated
* For joining 2 pixels apply threshold
* A picture containing graphical user interface

  Description automatically generated
* Hough transform: line parametrisation 🡪 from x y to m and c space [slope and intercept [characterised at intersection of lines in m and c space]
* Use r and theta representation (polar coordinates).
* Diagram

  Description automatically generated
* Can constrain values of theta unlike m [cartesian parametrisation] which can range from -inf to + inf
* Compute it:
* **Step 1**: compute accumulator array [histogram 🡪 2d matrix bins of histogram]
* **Step 2**: compute r and theta in range for every bin
* Find bins with sufficiently large number of votes
* Diagram, schematic

  Description automatically generated
* 3 parameters characterise a circle not theta
* For line only 1
* Displaced ellipse [5 unknowns includes theta for its rotation]
* Pairs of points with similar gradients [ellipse]

Lecture 3:

* Text

  Description automatically generated
* Text

  Description automatically generated
* **Spatial to frequency domain and then do calculations.** Then take inverse. For noise suppression. Ignore high frequency components in frequency domain
* Text

  Description automatically generated with medium confidence
* Chart

  Description automatically generated
* Above for image size MxN
* Use FFT to speed up the process [faster computation]
* **Phase [brings more info compared to amplitude] and amplitude are computed**
* Chart

  Description automatically generated
* FT for smoothing, edge detection [high pass filter]
* <https://akshaysin.github.io/fourier_transform.html#.YXKEJRrMJPY> [for edge detection]

Lecture 4:

* Shape representation and matching
* Text, whiteboard

  Description automatically generated**cross correlation for similarity**
* No flipping in cross correlation [**subtract the mean and divide by standard deviation**]
* If kernel symmetric then convolution and correlation yield same result
* Instead of template matching use **shape discriminants** [area, perimeter, elongatedness, moments, direction, energy, entropy and chord distribution] 🡪 mainly for **sorting and orienting**
* Use elongated Ness for very asymmetric objects
* Text

  Description automatically generated with medium confidenceMoments
* 0th order moment gives the area
* 1st order divided by 0th order gives centroid and 2nd order gives the spread of the object
* Diagram

  Description automatically generatedDirection
* Text

  Description automatically generatedOrientation can be estimated as function of 2nd order moments
* Signature
* Geometric discriminants [based on geometric features]
* Image segmentation: by use of histogram [quantisation of intensities dependent on bin size]🡪 histograms to assign threshold and labels
* For RGB systems the zero-intensity pixel is the black pixel
* Region merging divide into small patches and find uniform patch [this is the seed] 🡪 then expand area using uniformity criterion / opposite process for region splitting [based on uniformity criterion]. Both split and merging using quadtrees
* Uniformity criteria [thresholds for splitting and merging]🡪 use variance as shown below
* Text, letter

  Description automatically generated
* Variance assumes that intensities inside block [segmented] follow a normal distribution
* Diagram

  Description automatically generated with medium confidence
* Text

  Description automatically generated with medium confidenceconvert from RGB to HSI space [hue, saturation and intensity]
* Co-currence matrix 🡪 how many pairs of pixels have specific pixel intensities
* Text, letter

  Description automatically generated
* Text, letter

  Description automatically generated

Lecture 5 – Detecting corners:

* Text

  Description automatically generated
* A picture containing text, clock

  Description automatically generated
* Above high at corners [distance ε high at corner points, derivatives will be high due to significant changes in intensity]
* Diagram

  Description automatically generated with medium confidence
* Vertical edge means higher variation in Ix.
* Diagram

  Description automatically generated
* Uniform window [point], corner [circle], edge [ellipse]. Plot eigenvalues graph
* Diagram

  Description automatically generated
* Threshold for corners or consider local maxima [single response to corners]
* Image scale matters [corner vs edge example]. Also with rotation eigenvectors change
* A picture containing text

  Description automatically generated
* DoG vs LoG [faster]

Lecture 6:

* LoG second derivative along x and y [strong corners in scale space]
* SIFT descriptor: 4 by 4 and 8 orientations therefore 128-dimensional descriptor
* SIFT is illumination invariant [scale and orientation invariant] [difference between intensities remains the same [gradient and therefore orientation]]
* Diagram, schematic

  Description automatically generatedrelative to dominant orientation in neighbourhood
* Feature matching:
* Text

  Description automatically generated
* Brightness constancy, temporal persistence and spatial coherence for tracking in image sequences
* Text

  Description automatically generated
* A picture containing text, clock

  Description automatically generatedbrightness constancy
* Due to spatial coherence then:
* A picture containing diagram

  Description automatically generated
* Text

  Description automatically generated
* Text

  Description automatically generated
* Multivariate normal distribution:
* Text, whiteboard

  Description automatically generated
* Text

  Description automatically generated
* Kalman filter: Text, letter

  Description automatically generated
* Graphical user interface, text, application

  Description automatically generated with medium confidence
* A picture containing text

  Description automatically generated
* SVM: Chart, scatter chart

  Description automatically generated
* Graphical user interface, text, application

  Description automatically generatedMulti-class SVM / if class in voting use the model with least uncertainty and/or most accuracy
* Patching, extract salient features, descriptor, extract word, create vocab and describe scene and train, **classify in advance [highest value or voting scheme]**