Image and video processing

Edge detection and segmentation

Dr. Yi-Zhe Song

Agenda

- Edge detection
  - Gradient
  - Laplacian filters
  - Unsharp masking
  - Canny edge detector
- Other image segmentation approaches
  - Thresholding
  - Region based segmentation

The Segmentation Problem

- Segmentation attempts to partition the pixels of an image into groups that strongly correlate with the objects in an image
- Typically the first step in any automated computer vision application

Segmentation Examples

- **Edge detection**
  - **Gradient**
  - Laplacian filters
  - Unsharp masking
  - Canny edge detector
- Other image segmentation approaches
  - Thresholding
  - Region based segmentation

Edges

Local

- Discontinuities: points, lines and edges

- Edges
  - define the shape of objects in a scene (example)
  - correspond to *large variations* in the values of the pixels (example)

Edges

- Edge detection
  - One of the most used operations in image processing
    - E.g., shape analysis and recognition
  - Local operation to determine intensity changes
  - Estimation of paths on the image dividing areas with different intensity values (→ segmentation)

Edges

What causes intensity changes?

- Various physical events cause intensity changes.
- Geometric events
  - object boundary (discontinuity in depth and/or surface color and texture)
  - surface boundary (discontinuity in surface orientation and/or surface color and texture)
- Non-geometric events
  - specularity (direct reflection of light, such as a mirror)
  - shadows (from other objects or from the same object)
  - inter-reflections

Examples of edges

Ideal edges: computer generated (synthetic images)

edges have directions

More realistic (but still idealised) – gradual transition

problem: where exactly should the edge be?

Edge detection

- Strategy
  - finding points of *large variation* in an image
  - Measure gray-level transitions in a meaningful way
- Requirement
  - a good edge detector must understand the difference between
    - Variations caused by image noise noise problem: local
    - Variations caused by textures in the objects a global pattern – unless you look at it from a global eye, you won't know it's the texture
    - Variations caused by true edges of objects

- Results of edge detection
  - often presented as a sub-set of image pixels representing its edges
  - shown
    - as a 2-level image of the edges only or binary case = segmentation map
    - by superimposing the edges with a different colour on the original image

Edge detection

- Differential methods
  - Gradient
  - Laplacian
- Template methods
  - Roberts
  - Prewit
  - Sobel
  - Kirsch
- Optimisation methods
  - based on modelisation of edges, noise, quality measure of edge detection
  - Marr
  - Canny

Edges and derivatives

- How derivatives are used to find discontinuities?
- 1st derivative tells us where an edge is
- 2nd derivative can be used to show edge direction

Computing the 1st derivative

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \approx f(x+1) - f(x) \quad (h=1)$$

convolutional mask

mask: $[-1 \quad 1]$

$$f'(x) = f(x) - f(x-1)$$ [–1 1] Backward difference

$$f'(x) = f(x+1) - f(x)$$ [–1 1] Forward difference

$$f'(x) = f(x+1) - f(x-1)$$ [–1 0 1] Central difference

Computing the 2nd derivative

$$f''(x) = \lim_{h \to 0} \frac{f'(x+h) - f'(x)}{h} \approx f'(x+1) - f'(x) =$$

$$f(x+2) - 2f(x+1) + f(x) \quad (h=1)$$

- This approximation is **centered** about $x+1$; by replacing $x+1$ by x we obtain:

$$f''(x) \approx f(x+1) - 2f(x) + f(x-1)$$

mask: $\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$

1st derivative - step edges mask [-1, 0, +1]

S_1			12	12	12	12	12	24	24	24	24	24
S_1	$\otimes$	M	0	0	0	0	12	12	0	0	0	0

(a) S_1 is an upward step edge

S_2			24	24	24	24	24	12	12	12	12	12
S_2	$\otimes$	M	0	0	0	0	-12	-12	0	0	0	0

(b) S_2 is a downward step edge

Step edge is detected well, but edge location imprecise.

1st derivative
- ramp and impulse mask [-1, 0, +1]

S_3			12	12	12	12	15	18	21	24	24	24
S_3	$\otimes$	M	0	0	0	3	6	6	6	3	0	0

(c) S_3 is an upward ramp

S_4			12	12	12	12	24	12	12	12	12	12
S_4	$\otimes$	M	0	0	0	12	0	-12	0	0	0	0

(d) S_4 is a bright impulse or "line"

Ramp edge now yields a broad weak response. Impulse response is a "whip", first up and then down.

边界情况：重复边界的值 – padding

12	12	14	23	15	14	8
-1	0	1				

2nd derivative using mask [-1, 2, -1]

S_1			12	12	12	12	12	24	24	24	24	24
S_1	$\otimes$	M	0	0	0	0	-12	12	0	0	0	0

(a) S_1 is an upward step edge

S_2			24	24	24	24	24	12	12	12	12	12
S_2	$\otimes$	M	0	0	0	0	12	-12	0	0	0	0

(b) S_2 is a downward step edge

Response is zero on constant region and a “double whip” amplifies and locates the step edge.

2nd derivative using mask [-1, 2, -1]

S_3			12	12	12	12	15	18	21	24	24	24
S_3	$\otimes$	M	0	0	0	-3	0	0	0	3	0	0

(c) S_3 is an upward ramp

S_4			12	12	12	12	24	12	12	12	12	12
S_4	$\otimes$	M	0	0	0	-12	24	-12	0	0	0	0

(d) S_4 is a bright impulse or "line"

the higher–order derivative you go to,
the more sensitive it becomes to noise

Weak response brackets the ramp edge. Bright impulse yields a double whip with gain of 3X original contrast.

Edges and derivatives

- Derivative based edge detectors are extremely sensitive to noise
- We need to keep this in mind

Gradient

- Gradient of an image $f(x, y)$ at location (x, y)

$$\nabla f(x,y) = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f(x,y)}{\partial x} \\ \dfrac{\partial f(x,y)}{\partial y} \end{bmatrix}$$

$\nabla f(x,y)$ Is a vector that points in the direction of the greatest rate of change of image f at coordinates (x, y).

Gradient

- The arbitrary angle derivative filter $\cos(\theta)G_x + \sin(\theta)G_y$

- Magnitude of the gradient vector

$$\nabla f(x,y) = mag(\nabla \mathbf{f}) = [G_x^2 + G_y^2]^{1/2}$$

- Direction of the gradient vector $\alpha(x,y) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$

  Where the angle is measured with respect to the *x*-axis

- The magnitude of gradient provides information about the <u>strength</u> of the edge.
- The direction of gradient is always <u>perpendicular</u> to the direction of the edge (the edge direction is rotated with respect to the gradient direction by -90 degrees).

Estimating the gradient with finite differences

$$\frac{\partial f}{\partial x} = \lim_{h \to 0} \frac{f(x+h, y) - f(x, y)}{h}$$

$$\frac{\partial f}{\partial y} = \lim_{h \to 0} \frac{f(x, y+h) - f(x, y)}{h}$$

- The gradient can be approximated by *finite differences*:

$$\frac{\partial f}{\partial x} = \frac{f(x+h_x, y) - f(x, y)}{h_y} = f(x+1, y) - f(x, y), \quad (h_x = 1)$$

$$\frac{\partial f}{\partial y} = \frac{f(x, y+h_y) - f(x, y)}{h_y} = f(x, y+1) - f(x, y), \quad (h_y = 1)$$

2–dimensional matrix mask

Roberts

diagonal changes

-1	0
0	1

0	-1
1	0

- Roberts cross-gradient operators

$$G_x=(z_9-z_5),\ G_y=(z_8-z_6)$$

- Gradient computed across diagonals
- Fast
- Do not have a clear centre

Prewitt

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

- Prewitt cross-gradient operators

$$G_x = (z_7+z_8+z_9)-(z_1+z_2+z_3),\ G_y = (z_3+z_6+z_9)-(z_1+z_4+z_7)$$

- Gradient of least-squares plane through the 9 pixels
- Simpler to implement

Sobel

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

- Sobel cross-gradient operators

$$G_x = (z_7+2z_8+z_9)-(z_1+2z_2+z_3),\ G_y = (z_3+2z_6+z_9)-(z_1+2z_4+z_7)$$

- A weight of 2 in the centre coefficient for smoothing
- Slightly superior noise-suppression - a comparison

Gradient method: flow diagram

- Solution A

Gradient method: flow diagram

- Solution B
  - Simpler → preferred

Examples

a b
c d

(a) Original image. (b) $|G_x|$, component of the gradient in the x-direction. (c) $|G_y|$, component in the y-direction. (d) Gradient image, $|G_x| + |G_y|$

Gradient ∇f

- Choice of gradient operator ∇f
  - This has an important effect on the edge detection output
    - **thresholding** to reduce the effect of noise
    - detection of **local maximum**
    - used as part of **post-processing** methods
      - to remove noise detected as a contour and
      - to correct thick edges

$$\nabla f \approx |G_x| + |G_y|$$

$$\theta = \arctan\left(\frac{G_y}{G_x}\right)$$

  - If $\nabla f > T$, then possible edge point

Gradient Operators

Prewitt masks for detecting diagonal edges

0	1	1
−1	0	1
−1	−1	0

−1	−1	0
−1	0	1
0	1	1

Prewitt

Sobel masks for detecting diagonal edges

0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

Sobel

Examples

0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

Next topics

- Edge detection
  - Gradient
  - **Laplacian filters**
  - Unsharp masking
  - Canny edge detector
- Other image segmentation approaches
  - Thresholding
  - Region based segmentation

Laplacian

- The Laplacian of a continuous 2D signal $f(x, y)$
  - Is a 2D isotropic measure of the 2nd spatial derivative of an image
  - It is very sensitive to noise → it should be applied to an image that has first been smoothed
  - The Laplacian is given by

$$\nabla^2 f(x, y) = \frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial y^2}$$

Laplacian

- The Laplacian of a discrete 2D signal
  - Can be approximation of its partial second-order derivative

$$\frac{\partial^2 f(x,y)}{\partial x^2} \cong f(x+1,y) + f(x-1,y) - 2f(x,y)$$

$$\frac{\partial^2 f(x,y)}{\partial y^2} \cong f(x,y+1) + f(x,y-1) - 2f(x,y)$$

Laplacian

- The Discrete two-dimensional Laplacian

$$\nabla^2 f = [f(x+1), y) + f(x-1, y) + f(x, y+1) + f(x, y-1)] - 4f(x, y)$$

- Can be implemented using

0	1	0
1	-4	1
0	1	0

$$\nabla^2 f = -4z_5 + (z_2 + z_4 + z_6 + z_8)$$

Laplacian method: flow diagram

- NB: Edge detection based on the Laplacian is very sensitive to noise

Laplacian filters

- Laplacian
  - can be calculated using a convolution filter
  - We need a discrete convolution mask that can approximate the second derivatives in the definition of the Laplacian
  - Two commonly used small masks

$$\frac{1}{1-a}\begin{bmatrix} a & 1-a & a \\ 1-a & -4 & 1-a \\ a & 1-a & a \end{bmatrix}$$

a=0

0	1	0
1	-4	1
0	1	0

a=0.5

1	1	1
1	-8	1
1	1	1

Properties of Laplacian

- It is an isotropic operator.
- It is cheaper to implement (one mask only).
- It does not provide information about edge direction.
- It is more sensitive to noise

Example: Laplacian filters

‘smaller window’

$$\begin{matrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{matrix}$$

$$\begin{matrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{matrix}$$

‘larger window’

Original image

Pixel differences

x direction

Pixel differences

y direction

Sobel: amplitude of the gradient

Laplacian filtering

Next topics

- Edge detection
  - Gradient
  - Laplacian filters
  - **Unsharp masking** to sharpen an image
  - Canny edge detector
- Other image segmentation approaches
  - Thresholding
  - Region based segmentation

Unsharp masking: rationale

- Mach banding
  - unsharp masking creates artificial Mach bands
  - Mach banding is a phenomenon in human vision
  - The eye and visual cortex perform pre-processing on the retinal image
  - The brain twists reality before our high centres of reasoning get the visual information
  - enhancing edges beyond the true natural appearance → is the same as Mach banding

Mach bands

How the eye works

brains are sensitive to contrast

An example of Mach bands

The true brightness matches the graph above on the left, but the perceived brightness matches the graph above on the right.

How unsharp masking works

Unsharp masking

- How to make an image look sharper?

- Unsharp masking is performed by
  - blurring the original remove the high frequencies
  - subtracting the blurred image from the original → segmentation mask only the high frequencies remain
  - adding the mask to the original
    high frequencies double

Original image

Problem: Noises are doubled, too

Unsharp masking applied

Unsharp masking

- Edges can be enhanced with:

- … but the result is affected by noise

Convolution kernel for unsharp masking

- Can be implemented based on a convolution kernel
  - Laplacian kernels

0	-1	0
-1	4	-1
0	-1	0

-1	-1	-1
-1	8	-1
-1	-1	-1

1	-2	1
-2	4	-2
1	-2	1

- Scaling factor for masks:

Unsharp masking with Laplacian

Laplacian filters

- Laplacian filters are very sensitive to noise
  - due to the ill-posedness of the gradient estimations
- To deal with this problem
  - de-noising should be done first
  - usually Gaussian smoothing is used
  - since the convolution operation is associative, we can
    - first convolve the Gaussian filter with the Laplacian filter, and
    - then convolve this hybrid filter with the image
- This is called LoG operation (Laplacian of Gaussian)

Unsharp masking: Laplacian of Gaussian

less noise

The LoG operator

- To reduce the noise effect, the image is first smoothed with a low-pass filter.
- The LoG (`Laplacian of Gaussian') filter can be pre-calculated and stored in a filter bank
- In the case of the LoG, the low-pass filter is chosen to be a Gaussian.

- Consider the function

A Gaussian function

$$G(r) = e^{-\frac{r^2}{2s^2}} \quad \text{where} \quad r^2 = x^2 + y^2$$

and s: the standard deviation

The LoG operator

- The continuous 2D LoG function centered on zero is
- The Laplacian of *h* is

$$\nabla^2 G(r) = \left[\frac{r^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}}$$

The Laplacian of a Gaussian (LoG)

- The Laplacian of a Gaussian sometimes is called the Mexican hat function. It also can be computed by smoothing the image with the Gaussian smoothing mask, followed by application of the Laplacian mask.

The LoG operator

0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

Laplacian of a Gaussian (LoG). (a) 3-D plot. (b) Image (black is negative, gray is the zero plane, and white is positive). (c) Cross section showing zero crossings. (d) 5 × 5 mask approximation to the shape of (a).

Convolution filtering

- LOG Laplacian (2nd spatial derivative) of the Gaussian

- edge detection

Marr-Hildreth method: flow diagram

Marr-Hildreth method: flow diagram

The LoG operator

- At a reasonably sharp edge between two uniform regions with different intensities,
  the LoG response will be

0	at a long distance from the edge
>0	just to one side of the edge
<0	just to the other side of the edge
0	at some point in between, on the edge itself

The fact that the output of the filter passes through zero at edges can be used to detect those edges: Zero Crossings

Approximations of the LoG operator

- The LoG filter can be approximated by the difference of two differently sized Gaussians
  - this is called *DoG* filter → Difference of Gaussians

- An even cruder but faster approximation
  - DoB filter → Difference of Boxes → simply the difference between two mean filters of different sizes

超纲超纲

Zero Crossings

original

ZC at low scale

level of Gaussian blur

Zero Crossings

original

false positives

ZC at high scale

Zero Crossings

original

reconstructed ZCs

- Edge detection
  - Gradient
  - Laplacian filters
  - Unsharp masking
  - **Canny edge detector**
- Other image segmentation approaches
  - Thresholding
  - Region based segmentation

Criteria for Optimal Edge Detection

- Low error rate of detection
  - Minimize the probability of false positives (i.e., spurious edges).
  - Minimize the probability of false negatives (i.e., missing real edges).
  - Well match human perception results
- Good localization of edges
  - The distance between actual edges in an image and the edges found by a computational algorithm should be minimized
- Single response
  - Minimize the number of local maxima around the true edge.
  - The algorithm should not return multiple edges pixels when only a single one exists

Canny edge detector

- Detection/Localization trade-off
  - More smoothing improves detection
  - And hurts localization.

- Canny has shown that the **first derivative of the Gaussian** closely approximates the operator that optimizes the product of *signal-to-noise* ratio and localization.

J. Canny, ***A Computational Approach To Edge Detection***, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

Flow-chart of Canny Edge Detector

Original image

↓

Smoothing by Gaussian convolution

↓

Differential operators along x and y axis

↓

Non-maximum suppression
finds peaks in the image gradient

leaving the maxima
throwing the minima

↓

Hysteresis thresholding locates edge strings

↓

Edge map

Canny Edge Detector - Algorithm

1. Compute f_x and f_y

$$f_x = \frac{\partial}{\partial x}(f * G) = f * \frac{\partial}{\partial x} G = f * G_x$$

$$f_y = \frac{\partial}{\partial y}(f * G) = f * \frac{\partial}{\partial y} G = f * G_y$$

$G(x, y)$ is the Gaussian function

$G_x(x, y)$ is the derivate of $G(x, y)$ with respect to x: $G_x(x, y) = \frac{-x}{\sigma^2} G(x, y)$

$G_y(x, y)$ is the derivate of $G(x, y)$ with respect to y: $G_y(x, y) = \frac{-y}{\sigma^2} G(x, y)$

2. Compute the gradient magnitude

$$magn(i, j) = \sqrt{f_x^{\,2} + f_y^{\,2}}$$

3. Apply non-maxima suppression.

4. Apply hysteresis thresholding/edge linking.

Canny edge detector - example

original image

Canny edge detector - example

Gradient magnitude

Canny edge detector - example

Thresholded gradient magnitude

Canny edge detector - example

each edge is 1 pixel wide

Thinning (non-maxima suppression)

Non-maxima suppression

- Check if gradient magnitude at pixel location (*i,j*) is local maximum along gradient direction

Non-maxima suppression

<u>Warning:</u> requires checking interpolated pixels p and r

Algorithm

For each pixel (i, j) do:

if $magn(i, j) < magn(i_1, j_1)$ or $magn(i, j) < magn(i_2, j_2)$
  then $I_N(i, j) = 0$
else $I_N(i, j) = magn(i, j)$

Non-maxima suppression

- Predicting the next edge point

Hysteresis thresholding

- Standard thresholding:

$$E(x,y) = \begin{cases} 1 & \text{if } \|\nabla f(x,y)\| > T \text{ for some threshold } T \\ 0 & \text{otherwise} \end{cases}$$

- Can only select “strong” edges.
- Does not guarantee “continuity”.

gradient magnitude

low threshold

high threshold

Hysteresis thresholding

- Hysteresis thresholding uses two thresholds:

  - low threshold t_l
  - high threshold t_h (usually, $t_h = 2t_l$)

$$
\begin{array}{rrll}
 & \|\nabla f(x,y)\| & \geq & t_h \quad \text{definitely an edge} \\
t_l \leq & \|\nabla f(x,y)\| & < & t_h \quad \text{maybe an edge, depends on context} \\
 & \|\nabla f(x,y)\| & < & t_l \quad \text{definitely not an edge}
\end{array}
$$

- For "maybe" edges, decide on the edge if <u>neighbouring</u> pixel is a strong edge.

Hysteresis thresholding/Edge Linking

<u>Idea:</u> use a **high** threshold to start edge curves and a **low** threshold to continue them.

Use edge "direction" for linking edges

Hysteresis thresholding/Edge Linking

Algorithm

1. Produce two thresholded images $I_1(i, j)$ and $I_2(i, j)$.(using t_l and t_h)

(note: since $I_2(i, j)$ was formed with a high threshold, it will contain fewer false edges but there might be gaps in the contours)

2. Link the edges in $I_2(i, j)$ into contours

   2.1 Look in $I_1(i, j)$ when a gap is found.

   2.2 By examining the 8 neighbors in $I_1(i, j)$, gather edge points from $I_1(i, j)$ until the gap has been bridged to an edge in $I_2(i, j)$.

Hysteresis Thresholding

$$Threshold = 25$$

$$High = 35$$

$$Low = 15$$

Multi-scale Processing

- A formal theory for handling image structures at different scales.
- Process images **multiple scales**.
- Determine which structures (e.g., edges) are most significant by considering **the range of scales** over which they occur.

Noise is not applicable – not structured

Multi-scale Processing

σ=1

σ=2

σ=4

σ=8

σ=16

•**Interesting scales**: scales at which important structures are present.

e.g., in the image above, people can be detected at scales [1.0 - 4.0]

Effect of σ (Gaussian kernel size)

original

Canny with $\sigma = 1$

Canny with $\sigma = 2$

The choice of σ depends on desired behavior

- large σ detects large scale edges
- small σ detects fine features

Scale

- Smoothing
- Eliminates noise edges.
- Makes edges smoother.
- Removes fine detail.

(Forsyth & Ponce)

fine scale
high
threshold

coarse scale, high threshold

coarse
scale
low
threshold

Scale Space (Witkin 1983)

- Detect and plot the zero-crossing of a 1D function over a continuum of scales σ.

- Instead of treating zero-crossings at a single scale as a single point, we can now treat them at multiple scales as contours.

A. Witkin, "Scale-space filtering", 8th Int. Joint Conf. Art. Intell., Karlsruhe, Germany,1019–1022, 1983

Scale Space

local structures merge

never introducing details

- Properties of scale space (assuming Gaussian smoothing):
  - Zero-crossings may shift with increasing scale (σ).
  - Two zero-crossing may merge with increasing scale.
  - A contour may ***not*** split into two with increasing scale.

- Edge detection
  - Gradient
  - Laplacian filters
  - Unsharp masking
  - Canny edge detector
- **Other image segmentation approaches**
  - **Thresholding**
  - **Region based segmentation**

Thresholding

- Assumption: the range of intensity levels covered by objects of interest is different from the background.

$$g(x,y) = \begin{cases} 1 & \text{if} \quad f(x,y) > T \\ 0 & \text{if} \quad f(x,y) \le T \end{cases}$$

Single threshold

Multiple threshold

Thresholding

- The Role of Illumination

Thresholding

- The Role of Illumination

(a) Computer generated reflectance function.

(b) Histogram of reflectance function.

Thresholding

- Basic Global Thresholding

(a) Original image. (b) Image histogram. (c) Result of global thresholding with T midway between the maximum and minimum gray levels.

Thresholding

- Basic Global Thresholding

(a) Original image. (b) Image histogram. (c) Result of segmentation with the threshold estimated by iteration. (Original courtesy of the National Institute of Standards and Technology.)

Thresholding

- Basic Adaptive Thresholding

(a) Original image. (b) Result of global thresholding. (c) Image subdivided into individual subimages. (d) Result of adaptive thresholding.

Thresholding

- Basic Adaptive Thresholding

How to solve this problem?

Thresholding

- Basic Adaptive Thresholding

Answer: subdivision

- (a) histogram of small subimage at top left
- (b) further subdivided subimage
- (c) result of adaptively segmenting

Thresholding

- Thresholds Based on Several Variables

Colour image

(a) Original color image shown as a monochrome picture. (b) Segmentation of pixels with colors close to facial tones. (c) Segmentation of red components.

Region-Based Segmentation

- Edges and thresholds sometimes do not give good results for segmentation.
- Region-based segmentation is based on the connectivity of similar pixels in a region.
  - Each region must be uniform.
  - Connectivity of the pixels within the region is very important.
- There are two main approaches to region-based segmentation: region growing and region splitting.

Region-Based Segmentation

- Basic Formulation
  - Let R represent the entire image region.
  - Segmentation is a process that partitions R into subregions, $R_1, R_2, \ldots, R_n$, such that

$$\text{(a)} \ \bigcup_{i=1}^{n} R_i = R$$

$$\text{(b)} \ R_i \text{ is a connected region}, \ i = 1,2,\ldots,n$$

$$\text{(c)} \ R_i \cap R_j = \phi \ \text{ for all } i \text{ and } j, i \neq j$$

$$\text{(d)} \ P(R_i) = \text{TRUE for } i = 1,2,\ldots,n$$

$$\text{(e)} \ P(R_i \cup R_j) = \text{FALSE for any adjacent regions } R_i \text{ and } R_j$$

where $P(R_k)$: a logical predicate defined over the points in set R_k

For example: $P(R_k)$=TRUE if all pixels in R_k have the same gray level.

Region-Based Segmentation

Region Growing

bottom up

Seeds selection:

- A simple approach to image segmentation is to start from some pixels (seeds) representing distinct image regions and to grow them, until they cover the entire image
  - When a priori information is not available, the procedure is to compute at every pixel the same set of properties
  - If the result of these computation shows clusters of values, the pixels whose properties place them near the centroid of each cluster can be used as seeds

Region-Based Segmentation

Region Growing

Criteria for region growing

- For region growing we need a rule describing a growth mechanism and a rule checking the homogeneity of the regions after each growth step
- Descriptors: gray levels, spatial properties (moments or texture)
- Connectivity (or adjacency information)
- Stop rule: stop when no more pixels satisfy the criteria for inclusion in that region

Region-Based Segmentation

- Region Growing

(a) Image showing defective welds. (b) Seed points. (c) Result of region growing. (d) Boundaries of segmented defective welds (in black). (Original image courtesy of X-TEK Systems, Ltd.).

Region-Based Segmentation

- Region Growing

Region-Based Segmentation

- Region splitting is the opposite of region growing.
  - First there is a large region (possible the entire image).
  - Then a predicate (measurement) is used to determine if the region is uniform.
  - If not, then the method requires that the region be split into two regions.
  - Then each of these two regions is independently tested by the predicate (measurement).
  - This procedure continues until all resulting regions are uniform.

Region-Based Segmentation

Region Splitting

- The main problem with region splitting is determining where to split a region.
- One method to divide a region is to use a quadtree structure.
- Quadtree: a tree in which nodes have exactly four descendants.

(a) Partitioned image.
(b) Corresponding quadtree.

Region-Based Segmentation

Region Splitting and Merging

- The split and merge procedure:
  - Split into four disjoint quadrants any region R_i for which $P(R_i)$ = FALSE.
  - Merge any adjacent regions R_j and R_k for which $P(R_j \cup R_k)$ = TRUE. (the quadtree structure may not be preserved)
  - Stop when no further merging or splitting is possible.

(a) Original image. (b) Result of split and merge procedure. (c) Result of thresholding (a).

Applications

- 3D – Imaging : A basic task in 3-D image processing is the segmentation of an image which classifies voxels/pixels into objects or groups.
  - 3-D image segmentation makes it possible to create 3-D rendering for multiple objects and perform quantitative analysis for the size, density and other parameters of detected objects.

- Several applications in the field of Medicine like magnetic resonance imaging (MRI).

Results – Region grow

Results – Region Split

Results – Region Split and Merge

Example - point detection

top–down

X-ray image of a turbine blade

Result of point detection

Result of thresholding

Example – line detection

Binary image of a wire bond mask

After processing with -45° line detector

Result of thresholding filtering result

Example – Edge detection (Lena)

Example – Edge detection

The Lenna Story - www.lenna.org

- The Lenna (or Lena) picture is one of the most widely used standard test images used for compression algorithms.
- The original image is still available as part of the USC SIPI Image Database in their "miscellaneous" collection
- http://sipi.usc.edu/database/

- Lenna attended the 50th Anniversary IS&T conference in Boston held in May 1997!

Gradual transition

- An edge is a set of connected pixels that lie on the boundary between two regions

Model of an ideal digital edge

Gray-level profile
of a horizontal line
through the image

Model of a ramp digital edge

Gray-level profile
of a horizontal line
through the image

Comparison

sensitive to diagonal changes

Roberts

Sobel

a little more noise

Prewitt

Image and video processing

Interest points

Dr. Yi-Zhe Song

EBU723U

Today’s agenda

- Interest points
  - Definition
  - motivation
- Moravec operator
- Harris corner detector
- Evaluation of interest points

Last lecture: edges

$I(x,y)$

input image

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} / 16$$

noise smoothing

vertical
[-1 0 1]

edge enhancement

horizontal
$[-1\ 0\ 1]^T$

$$\frac{\partial I(x,y)}{\partial x}$$

$$\frac{\partial I(x,y)}{\partial y}$$

$$\partial I(x,y) = \left[\frac{\partial I(x,y)}{\partial x}^2 + \frac{\partial I(x,y)}{\partial y}^2 \right]^{\frac{1}{2}}$$

gradient image

threshold

edge image

Interest points

- it has a clear, preferably mathematically well-founded, definition,
- it has a well-defined *position* in image space,
- the local image structure around the interest point is rich in terms of local *information contents* (e.g.: significant 2D texture), such that the use of interest points simplify further processing in the vision system,
- it is *stable* under local and global perturbations in the image domain as illumination/brightness variations, such that the interest points can be reliably computed with high degree of *reproducibility*.

  regardless how things change, interesting points remain are still able to detect
- Optionally, the notion of interest point should include an attribute of *scale*, to make it possible to compute interest points from real-life images as well as under scale changes.

  less sensitive to scale

How to find corresponding points?

- Need to define local patches surrounding the interest points and extract feature **descriptors** from every patch.
- Match feature descriptors to find corresponding points.

Properties of good features

- **Local:** robust to occlusion and clutter.
- **Accurate:** precise localization.

- **Covariant**
- **Robust:** noise, blur, compression, etc.

Repeatable

- **Efficient:** close to real-time performance.

Interest point detectors should be covariant

- Features should be detected in corresponding locations despite <u>geometric</u> or <u>photometric</u> changes.

Interest point detectors should be invariant

Should be similar despite geometric
or photometric transformations

not all feature points are corners
corners are one of the most popular feature points

Corners

more discriminative: two edges meet

- where two edges come together
- corresponds to a point in both the world and image spaces
- point features / interest points / corners / feature points are all terms that are used
- A corner is where the image gradient has significant components in the x and y direction
- We can establish corners from the gradient rather than from the edge images

Motivation: Corners for recognition

Image search: find the book in an image.

Motivation: Corners for recognition

Motivation: Corners for augmented reality

Motivation: Corners for robotics

Motivation: Build a panorama

Input images

Output panorama 1

Motivation: Build a panorama

M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003

How do we build a panorama?

We need to match (align) images

How do we build a panorama?

- Detect corners (interest points) in both images
- Find corresponding corner pairs by comparing the corner descriptors

How do we build a panorama?

- Detect corners (interest points) in both images
- Find corresponding corner pairs by comparing the corner descriptors
- Use these pairs to align images

problems:
1) colour – lighting
2) following images have stronger rotation

More motivation

- Corner points are used also for:
  - Image alignment (homography, fundamental matrix)
  - 3D reconstruction, Object recognition, Indexing and database retrieval, Robot navigation, … other
- Corners define repeatable points for matching
- Not just intersection of two lines (pure corner) but pixels which have a “corner like” structure
- Corners sometimes called interest points because pixels that are “corner like” are interesting
- Observe that in the region around a corner the gradient has two or more distinct values

Corner Feature

- Corners are image locations that have large intensity changes in more than one direction
- For a pixel which is a corner shifting a window centered on that pixel in any direction should give a large change in the average intensity in that window.

在corner移动window窗里的变化很大

其他地方变化不大

Variance-based interest operators

- Variance-based interest operators
  - Method to find interest (corner) points
  - Measure the variance of the intensity values for pixels in a small window
  - the simplest and best known is Moravec

harris

Moravec operator

- The most popular interest point operator
- A simple difference-based operator
  - The sums of squared differences (SSD) of adjacent pixels in the four principal directions (vertical, horizontal and two diagonals) are measured over square overlapping windows of small size
  - For each sampling position the output of the interest operator is defined as the minimum of the sums of absolute differences of pixels adjacent in each of the four directions
  - Sampling positions whose filter response exceeds a certain threshold are declared interest points (corners)

Using the Moravec operator

$$V = \sum_{i=1}^{9} (A_i - B_i)^2 = 2 * 255^2$$

$$V = \sum_{i=1}^{9} (A_i - B_i)^2 = 3 * 255^2$$

Moravec operator

- Given a grayscale image *I(x,y)*
  - Centre a 3x3 pixel array on each pixel in turn
  - Define another array one pixel up to the right of the first array
  - Calculate the absolute differences between the values of the corresponding pairs of pixels in the two arrays
  - Sum all these absolute differences values (9 values)
  - Repeat for arrays moved one pixel relative to the initial array up, down, right, left, diagonally up left, diagonally down left and diagonally down right.
  - Take the minimum of these 8 values and this is the new value for the pixel in the Moravec image *M(i,j)*

8 shifts

$$M(i,j) = \cancel{\frac{1}{8}} \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} |I(k,l) - I(i,j)|$$

Not necessary

Moravec

- threshold the new image *M(i,j)* to isolate corners in the image

- Moravec operator
  - is maximal in pixels with high contrast
  - simple implementation
  - Diagonal edges will falsely be detected as corners
  - computationally efficient → works well for real-time applications
  - … there are better detectors in theory and in practice e.g. the Harris detector, but these take more computation

Example

problem: sensitive to horizontal and vertical edges

$I(x, y)$

$M(x, y)$

Example

$$M(x, y) > 50$$

$$M(x, y) > 65$$

Example

original

interest points
for a low threshold

Example

original

interest points
for a high threshold

Moravec

- Issue:
  - Only tries 4 shifts. We'd like to consider "all" shifts.
  - Uses a discrete rectangular window. We'd like to use a smooth circular (or later elliptical) window.
  - Uses a simple min function. We'd like to characterize variation with respect to direction.

- Improvements – Harris & Stephens corner detection
  - Instead of using patches circular, instead of squares
  - Considering the differential of the corner score with respect to direction

Harris corner detector

Based on the idea of auto-correlation

Important difference in all directions => interest point

- C.Harris, M.Stephens. “A Combined Corner and Edge Detector”. 1988

Harris corner detector

- The Basic Idea
  - We should easily recognize the point by looking through a small window
  - Shifting a window in *any direction* should give *a large change* in intensity

Harris Detector: Basic Idea

"flat" region:
no change in
all directions

"edge":
no change along
the edge direction

"corner":
significant change
in all directions

Harris Detector: Mathematics

Change of intensity for the shift [*u,v*]:

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

Window function

Shifted intensity

Intensity

Window function $w(x,y) =$

1 in window, 0 outside

or

Gaussian

Harris Detector: Mathematics

For small shifts [*u*, *v*] we have a *bilinear* approximation:

$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \; M \; \begin{bmatrix} u \\ v \end{bmatrix}$$

where *M* is a 2×2 matrix computed from image derivatives:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Ix Iy are first derivatives

Harris Detector: Mathematics

$$E(u,v) \cong [u,v] \quad M \begin{bmatrix} u \\ v \end{bmatrix}$$

λ_1, λ_2 – eigenvalues of M

Ellipse $E(u,v) = \text{const}$

Harris Detector: Mathematics

Measure of corner response:

$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

(k – empirical constant, k = 0.04-0.06)

Harris Detector: Mathematics

Classification of image points using eigenvalues of *M*:

Harris Detector: Mathematics

- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region

Harris Detector

- The Algorithm:
  - Find points with large corner response function R ($R >$ threshold)
  - Take the points of local maxima of R

Harris Detector: Workflow

Harris Detector: Workflow

Compute corner response *R*

Harris Detector: Workflow

Find points with large corner response: *R*>threshold

Harris Detector: Workflow

Take only the points of local maxima of *R*

Harris Detector: Workflow

Harris Detector: Summary

- Average intensity change in direction $[u, v]$ can be expressed as a bilinear form:

$$E(u,v) \cong [u,v] \; M \; \begin{bmatrix} u \\ v \end{bmatrix}$$

- Describe a point in terms of eigenvalues of M: *measure of corner response*

$$R = \lambda_1 \lambda_2 - k\left(\lambda_1 + \lambda_2\right)^2$$

- A good (corner) point should have a *large intensity change* in *all directions*, i.e. R should be large positive

Next agenda

- Interest points
  - Definition
  - motivation
- Moravec operator
- Harris corner detector
- **Evaluation of interest points**

Evaluation

- The two measures that are most useful for comparing techniques are Repeatability and Accuracy

- Repeatability
  - The Stability of interest points when
    - The view point changes and/or
    - geometric transformations are applied

- Accuracy
  - High accuracy in localisation (the position of the interest point)
  - A measure of the precision of the position of an estimated interest point compared to the ‘real’ interest point

Improvements on Moravec

- A better technique is the Scale-Space approach
- Versions of the image are smoothed with various values of σ^2 or various values of $t=\sigma^2$.
- As t increases this gives Gaussian filtering with a larger and larger filter i.e. increasing amounts of smoothing.

Scale-space approach

Scale-space approach

- Representation scale-space
  - Parametric set of images gradually smoothed
  - the most appropriate scale for extracting the interest points is not known

Original

Scale t = 5

Scale t = 10

Scale t = 30

Scale-space approach

- Selection of interest points
  - criterion: life-span of a track
  - the life-span
    - of points due to noise is short
    - of interest points is long

Interest points in scale space

Repeatability Evaluation

- A set of objects is rotated by a series of angles and interest points identified for each rotation
- If the method has high repeatability the same number of interest points (the total number of corners on the objects) should be identified for each rotation.

Evaluation: repeatability

Reference image (algo. Harris)

Rotated image (algo. Harris)

Reference image (algo. scale-space)

Rotated image (algo. scale-space)

Repeatability

- Repeatability = (# of interest points repeated) / (total # of interest points)

how many interest points remain

Scale space is
1. Larger
2. Smoother

Accuracy Evaluation

- The same method of rotating a set of objects by different degrees is used.
- The position of each interest point is measured.
- The difference between the position of the identified interest point and the true interest point is calculated
- An overall error for each position is calculated as the root mean square error,

Accuracy

- Accuracy
  - Root Mean Square Error $$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\left[(x_{n0}-x_n)^2+(y_{n0}-y_n)^2\right]}$$

distance of two points

smaller the better

What did we learn?

- Interest points
  - Definition
  - motivation
- Moravec operator
- Harris corner detector
- Evaluation of interest points

Image and video processing

Morphological Image Processing

Dr. Yi-Zhe Song

EBU723U

Today’s agenda

- What is morphology?
- Simple morphological operations
- Compound operations
- Morphological algorithms

What Is Morphology?

change of shapes

- Morphological image processing (or morphology) describes a range of image processing techniques that deal with the shape (or morphology) of features in an image
- Morphological operations are typically applied to
  - remove imperfections introduced during segmentation,
  - and so typically operate on bi-level images
  - These techniques can be extended to greyscale images.

Throughout all of the following slides whether 0 and 1 refer to white or black is a little interchangeable

What Is Morphology?

Quick example

Image after segmentation

Image after segmentation and morphological processing

Uses of Image Morphology

- image enhancement
- image segmentation
- image restoration
- edge detection
- texture analysis
- particle analysis
- feature generation
- skeletonization
- shape analysis
- image compression
- component analysis
- curve filling
- general thinning
- feature detection
- noise reduction
- space-time filtering

A Binary Image

This represents a digital image. Each square is one pixel.

not
very
useful
anyway
????

Support of an Image

- **The support** of a binary image is the **set of foreground pixel locations** within the image plane.
- **The complement of the support** is, therefore, the **set of background pixel locations** within the image plane.

Structuring Element (SE)

Structuring Element

A **structuring element** is a small image – used as a moving window – whose support delineates pixel neighborhoods in the image plane.

Structuring Element

- Structuring elements can be any size and make any shape
- It can be of any shape, size, or connectivity (more than 1 piece, have holes).
  - Zero-valued pixels of the structuring element are ignored, i.e. indicate points where the corresponding image value is irrelevant.
- In the figure the circles mark the location of the structuring element's origin which can be placed anywhere relative to its support.

Example SEs

FG is gray ;
BG is white

Reflected Structuring Elements

Let **s** be a SE and let $\mathfrak{S}$ be the square of pixel locations that contains the SE. Then

$$\tilde{s}(x, y) = s(-x, -y)$$

is the reflected structuring element.

$\tilde{s}$ is s rotated by 180º around its origin.

Fitting & Hitting

Structuring Element

- The structuring element is said to **fit** the image A
  - if, for each of its pixels set to 1, the corresponding image pixel is also 1.
- a structuring element is said to **hit**, or intersect, an image B
  - if, at least for one of its pixels set to 1 the corresponding image pixel is also 1.

Fundamental Operations

- Fundamentally morphological image processing is very like spatial filtering
- The structuring element is moved across every pixel in the original image to give a pixel in a new processed image
- The value of this new pixel depends on the operation performed
- There are two basic morphological operations:
  - **erosion**
  - **dilation**

Erosion

- Erosion of image f by structuring element s is given by $f \ominus s$
- The structuring element s is positioned with its origin at (x, y) and the new pixel value is determined using the rule:

$$g(x,y) = \begin{cases} 1 \text{ if } s \text{ fits } f \\ 0 \text{ otherwise} \end{cases}$$

Erosion Example

Erosion Example

Original Image

Processed Image

Structuring Element

edge detection for binary images

Erosion Example 1

Original image

Erosion by 3*3 square structuring element

Erosion by 5*5 square structuring element

Watch out: In these examples a 1 refers to a black pixel!

Erosion Example 2

Original image

After erosion with a disc of radius 10

After erosion with a disc of radius 5

After erosion with a disc of radius 20

What Is Erosion For?

Erosion can split apart joined objects

用Erosion分开重叠的图形

Erosion can strip away extrusions

Watch out: Erosion shrinks objects

What Is Erosion For?

- Separate touching objects

(a)

(b)

(c)

(d)

What Is Erosion For?

- Remove small spurious bright spots (salt noise)

(a)

(b)

Dilation

- Dilation of image f by structuring element s is given by $f \oplus s$
- The structuring element s is positioned with its origin at (x, y) and the new pixel value is determined using the rule:

$$g(x,y) = \begin{cases} 1 \text{ if } s \text{ hits } f \\ 0 \text{ otherwise} \end{cases}$$

Dilation Example

Dilation Example

Original Image

Processed Image With Dilated Pixels

Structuring Element

Dilation Example 1

Original image

Dilation by 3*3 square structuring element

Dilation by 5*5 square structuring element

Watch out: In these examples a 1 refers to a black pixel!

Dilation Example 2

Original image

After dilation

0	1	0
1	1	1
0	1	0

Structuring element

Erosion from Dilation / Dilation from Erosion

- Dilation and erosion are duals of each other with respect to complementation:

dilation of the FG is equivalent to the erosion of the BG

$$f \oplus s = f^c \ominus \tilde{s}$$

- That is, dilation with the reflected SE of the complement of a binary image is the complement of the erosion. Erosion with the reflected SE of the complement of the image is the complement of the dilation.

- erosion can be performed with dilation and vice versa. That implies that only one or the other must be implemented directly.

What Is Dilation For?

Dilation can repair breaks

Dilation can repair intrusions

Watch out: Dilation enlarges objects

What Is Dilation For?

- Fill in small spurious holes (`pepper noise') in images

What Is Dilation For?

- Edge detection by taking the dilation of an image and then subtracting away the original image

Compound Operations

More interesting morphological operations can be performed by performing combinations of erosions and dilations

Set operations on binary images: from left to right: a binary image f, a binary image g, the complement f^c of f, the intersection $f \cap g$, and the union $f \cup g$.

The most widely used of these *compound operations* are:

- **Opening**
- **Closing**

Opening – to separate touched objects

The opening of image f by structuring element s, denoted $f \circ s$ is simply an erosion followed by a dilation

$$f \circ s = (f \ominus s) \oplus s$$

Original shape | After erosion | After dilation (opening)

Note a disc shaped structuring element is used

Opening Example

Original Image

Image After Opening

Opening Example

Opening Example

Original Image

Processed Image

Structuring Element

Closing

The closing of image f by structuring element s, denoted $f \bullet s$ is simply a dilation followed by an erosion

$$f \bullet s = (f \oplus s) \ominus s$$

Original shape

After dilation

After erosion (closing)

Note a disc shaped structuring element is used

Closing Example

Original Image

Image After Closing

Closing Example

Morphological Processing Example

Morphological Algorithms

Using the simple technique we have looked at so far we can begin to consider some more interesting morphological algorithms

We will look at:

- Boundary extraction
- Region filling
- Extraction of connected components
- Thinning/thickening
- Skeletonisation compression

Boundary Extraction

Extracting the boundary (or outline) of an object is often extremely useful

The boundary can be given simply as

$$\beta(A) = A - (A \ominus B)$$

Boundary Extraction Example

- A simple image and the result of performing boundary extraction using a square 3*3 structuring element

Original Image Extracted Boundary

Boundary Extraction

binary image

original

8-connected SE

erosion by square

4-conn inside bdry

difference

This is a piece of a larger image. Boundary effects are not apparent

Boundary Extraction

binary image

original

4-connected SE

erosion by plus

8-conn inside bdry

difference

This is a piece of a larger image. Boundary effects are not apparent

Boundary Extraction

binary image

original

erosion by square is

in erosion by plus

8-conn inside bdry is

in 4-conn inside bdry

This is a piece of a larger image. Boundary effects are not apparent

Boundary Extraction

binary image

original

4-connected SE

dilation by plus

8-conn outside bdry

difference

This is a piece of a larger image. Boundary effects are not apparent

Boundary Extraction

binary image

original

8-connected SE

dilation by square

4-conn outside bdry

difference

This is a piece of a larger image. Boundary effects are not apparent

Boundary Extraction

binary image

original

dilation by plus is

in dilation by square

8-conn outside bdry is

in 4-conn outside bdry

This is a piece of a larger image. Boundary effects are not apparent

Region Filling

- Given a pixel inside a boundary, *region filling* attempts to fill that boundary with object pixels (1s)

Given a point inside here, can we fill the whole circle?

Region Filling

The key equation for region filling is

A: original image

$$X_k = (X_{k-1} \oplus B) \cap A^c \qquad k = 1,2,3.....$$

keep dilating the hole until cannot dilate anymore

Where X_0 is simply the starting point inside the boundary, B is a simple structuring element and A^c is the complement of A

This equation is applied repeatedly until X_k is equal to X_{k-1}

Finally the result is unioned with the original boundary

Region Filling Step By Step

til now, X7 U Ac doesn't change anymore

Region Filling Example

Original Image | One Region Filled | All Regions Filled

Conditional Dilation

original image

mask over original

dilated original

dilation inside a mask

Conditional Dilation

mask over dilated

masked dilated

masked dilated union with original

conditionally dilated with respect to mask

Connected Component Extraction

Connected Component Extraction

tags

masked tags

masked dilated tags

dilated tags

dilated tags

connected component

Binary Reconstruction

Used after opening to *grow back* pieces of the original image that are connected to the opening.

original

opened

reconstructed

Removes of small regions that are disjoint from larger objects without distorting the small features of the large objects.

Summary

- The purpose of morphological processing is primarily to remove imperfections added during segmentation
- The basic operations are *erosion* and *dilation*
- Using the basic operations we can perform *opening* and *closing*
- More advanced morphological operation can then be implemented using combinations of all of these

Image and video processing

Processing Colour Images

Dr. Yi-Zhe Song

The agenda

- Introduction to colour image processing
- Pseudo colour image processing
- Full-colour image processing basics
- Transforming colours
- Smoothing and sharpening

Preview

- Motivation
  - Colour is a powerful descriptor that often simplifies object identification and extraction from a scene.
  - Human can discern thousands of color shades and intensities, compared to about only two dozen shades of gray.

Preview

Preview

Preview

- Color image processing is divide into two major area:
  - *Full-Color* Processing
  - *Pseudo-Color* Processing

Introduction

- Different systems of representing colours were introduced in an earlier lecture
  - RGB and HSV are the most common

- The number of distinct colours that can be reproduced on a computer display or be recorded by a camera will depend on the number of bits allocated to the colour components and the display capabilities

Introduction

- For 24-bit each colour is represented by three two hex digit values e.g. purest red is FF0000, 000000 is black and FFFFFF is white.
  - A 24-bit RGB image can render 16,777,216 colours. Computer systems and other devices now normally have 24-bit capability.
  - Limitations on displays and operating systems limited the colours that could be relied on to reproduce on all systems to 216 safe colours – often called safe web colours or safe browser colours, but this limitation is no longer necessary.

- Top left of colour squares is FFFFFF, next right FFFFCC, next right FFFF99 etc.

Safe RGB Colours

Safe RGB Colors (Safe Web colors)

Number System	Color Equivalents					
Hex	00	33	66	99	CC	FF
Decimal	0	51	102	153	204	255

Valid values of each RGB component in a safe color.

(a) The 216 safe RGB colors.
(b) All the grays in the 256-color RGB system (grays that are part of the safe color group are shown underlined).

Safe RGB Colours

The RGB safe-color cube.

RGB 24-bit color cube.

Representing 24-bit colours

- Some systems use 30-bit colour, known as deep colour, or even 48-bit.
  - The human eye can only distinguish 24-bit colours,
  - but more bits are useful for post-processing.
- Displays will be driven by RGB values, but human perception of colours is better matched by HSV descriptions.
- Converting between the two systems is fairly simple

Pseudocolour image processing

- Pseudocolour or false colour image processing is assigning colours to grey values
- This is done to improve the understanding of images such as medical images, rainfall or temperature maps
  - Humans can distinguish thousands of colour shades but only about 25 shades of grey
- A method of allocating colours to grey levels is intensity slicing
  - Each grey level is considered to be a different slice
  - Any pixel value higher than the slice will given one colour
  - Pixel values lower than the slice will be given a different value
  - Values on the slice will be arbitrarily assigned one of the two colours

Intensity slicing

- The grey levels are from 0 (black) to L-1 (white)
- *P* planes for different intensities ($l_1, l_2 \ldots l_p$) that partition the grey scale into *P+1* intervals
- Each interval is assigned a different colour

Intensity slicing example

- Picker Thyroid Phantom
- Monochrome image of a radiation pattern coloured with eight colour regions
- The coloured version makes it much easier to distinguish the different grey levels

Weld X-ray

- Here problems in the weld are revealed by the X-ray saturating in the problem region
- The best conversion to colour is one for white (saturation) and a contrasting one for all other grey levels
- Here yellow for white and blue for everything else.

Rainfall pattern

- Rain sensors produces gray-scale images in which intensity corresponds to average monthly rainfall
- Colours are assigned to intensity values producing colour coded images
- Zoom of the South America region

Intensity to colour

- More range of pseudocolour enhancement can be achieved by three separate transformations of the pixel value to give red, green and blue values for the coloured pixel
- Changing the transformations can then produce very different colourings

Intensity to colour

- Functional block diagram for pseudocolour image processing
- f_R, f_G, and f_B are fed into the corresponding red, green and blue inputs of an RGB colour monitor

The effect of the transformations

Monochrome images with and without the explosives cylinder
Colour images clearly show the cylinder

The effect of the transformations

Explosives and garment bag received essentially the same colours

Intensity to colour

- A pseudocolour coding approach used when several monochrome images are available

Colour coding of multispectral images

4 spectral satellite images of Washington D. C.

Thematic bands in NASA's LANDSAT satellite

Band No.	Name	Wavelength (μm)	Characteristics and Uses
1	Visible blue	0.45–0.52	Maximum water penetration
2	Visible green	0.52–0.60	Good for measuring plant vigor
3	Visible red	0.63–0.69	Vegetation discrimination
4	Near infrared	0.76–0.90	Biomass and shoreline mapping
5	Middle infrared	1.55–1.75	Moisture content of soil and vegetation
6	Thermal infrared	10.4–12.5	Soil moisture; thermal mapping
7	Middle infrared	2.08–2.35	Mineral mapping

Combining several monochrome images

Jupiter Moon Io (NASA)

Bottom picture is a close-up of part of the top picture

Pseudocolour by combining images from sensors that respond to different spectral regions – some beyond the capability of the human eye

Full-colour image processing

- Two approaches
  - Process each colour component individually
  - Work with colour pixels directly
- As full-colour images typically have three components, the pixels are vectors
  - This can be represented in RGB colour space as

$$c(x,y) = \begin{bmatrix} c_R(x,y) \\ c_G(x,y) \\ c_B(x,y) \end{bmatrix} = \begin{bmatrix} R(x,y) \\ G(x,y) \\ B(x,y) \end{bmatrix}$$

Spatial mask in RGB colour space

- Major categories of full-colour Image processing:
  - Per-colour-component processing
  - Vector-based processing

Colour Transformation

Processing the components of a colour image within the context of a single colour model.

$$g(x,y) = T\left[f(x,y)\right]$$

$$s_i = T_i\left(r_1, r_2, \ldots, r_n\right), \qquad i = 1,2,\ldots,n$$

Colour components of ***g***

Colour components of ***f***

Colour mapping functions

Colour transformations

- The transformations will be operations such as
  - Changing image intensity
  - Replacing colours with their complements
  - Highlighting a specific range of colours in an image
  - Correcting image tone
  - Correcting image colours
  - Image smoothing or sharpening
  - Colour image edge detection
  - Segmentation based on colour
- The transformations can be carried out in any colour space but some are easier to achieve in particular colour spaces

Colour transformations

Full color

Cyan Magenta Yellow Black

CMYK

Red Green Blue

RGB

Hue Saturation Intensity

HSI

- Some difficulty in interpreting the HUE:
  - Discontinuity where 0 and 360° meet.
  - Hue is undefined for a saturation 0

Changing image intensity

Below the intensity is reduced by 30%, so values are multiplied by 0.7.

$$g(x,y) = kf(x,y)$$

$$s_i = kr_i \qquad i = 1,2,3$$

$$s_i = kr_i + (1-k) \qquad i = 1,2,3$$

$$s_1 = r_1$$
$$s_2 = r_2$$
$$s_3 = kr_3$$

Colour complements

- These are like grey-scale negatives and are the opposite colours in the colour circle

Colour complement values

- Here the new value for each component is calculated for 8-bit colour as 255 - current value.
- If R=120, G= 200, B= 50 for the current pixel the colour complement pixel values will be R=135, G=55, B= 205

Colour slicing

Motivation: Highlighting a specific range of colors in an image

Basic Idea:

- Display the color of interest so that they stand out from background
- Use the region defined by the colors as a mask for further processing

- The simplest technique is to replace all colours outside the selected range with a neutral colour (e.g. mid-grey; R=G=B= 127 for 8-bit)

Colour slicing

1. Colors of interest are enclosed by ***cube*** (or ***hypercube*** for n>3)

$$s_i = \begin{cases} 0.5 & if \left[\left| r_j - a_j \right| > \frac{W}{2} \right]_{any 1 \le j \le n} \\ r_i & otherwise \end{cases} , \qquad i = 1,2,\ldots,n$$

2. Colors of interest are enclosed by ***Sphere***

$$s_i = \begin{cases} 0.5 & if \sum_{j=1}^{n} (r_j - a_j)^2 > R_0^2 \\ r_i & otherwise \end{cases} , \qquad i = 1,2,\ldots,n$$

Colour slicing results

Cube

Sphere

Tone and Colour Correction

In conjunction with digital cameras, flatbed scanners, and inkjet printers, they turn a personal computer into a digital *Darkroom*

The Colors of monitor should represent accurately any digitally scanned source images, as well as the final printed out

Device Independent Color Model

The Model of choice for many color management system (CMS) is

CIE L*a*b* **(also called CIELAB) model**

Tone and Colour Correction

*L*a*b Color Components:*

$$L^* = 116 \cdot h\left(\frac{Y}{Y_w}\right) - 16$$

$$a^* = 500\left[h\left(\frac{X}{X_W}\right) - h\left(\frac{Y}{Y_W}\right)\right]$$

$$b^* = 200\left[h\left(\frac{Y}{Y_W}\right) - h\left(\frac{Z}{Z_W}\right)\right]$$

$$where:\ h(q) = \begin{cases} \sqrt[3]{q} & q > 0.008856 \\ 7.7.87q + 16/116 & q \le 0.008856 \end{cases}$$

X_W , Y_W and Z_W are reference white tristimulus values (defined by x=0.3127 and y=0.3290 in the *chromaticity diagram*

Tone and Colour Correction

The tonal range of an image, also called its key-type, refers to its general distribution of color intensities.

- *High-key images: Most of the information is concentrated at high intensities.*
- *Low-key images: Most of the information is concentrated at low intensities.*
- *Middle-key images: lie in between*

Tone transformation

Middle-key Image

Flat

Corrected

Tone correction 2

High-key Image

Light

Corrected

Tone correction 3

Low-key Image

Dark

Corrected

Colour balancing (correction)

The proportion of any color can be increased by decreasing the amount of the opposite (or complementary) color in the image or by raising the proportion of the two immediately adjacent colors or decreasing the percentage of the two colors adjacent to the complement.

Magenta ↓

Removing Red and Blue

Adding Green

Colour balancing (correction)

Original/Corrected

Correction in black

Correction in cyan

Colour balancing (correction)

Original/Corrected

Correction in magenta

Correction in yellow

Correction using histograms

- Histogram equalization automatically seeks to give the image a uniform histogram of intensity values
- If each of the colour components was equalized independently this would change the colours
- The colour intensities should be distributed uniformly but the colours left unaltered, so the HSV colour space is required

Histogram equalization on intensity

Histogram Equalizing the Intensity

Saturation correction

Histogram Equalizing the Intensity

Saturation Adjustment

Smoothing

- The techniques used for smoothing grey scale images can be used for colour images

$$\bar{\mathbf{c}}(x,y) = \frac{1}{K} \sum_{(x,y)\in S_{xy}} \mathbf{c}(x,y) \qquad \bar{\mathbf{c}}(x,y) = \begin{bmatrix} \frac{1}{K} \sum_{(x,y)\in S_{xy}} R(x,y) \\ \frac{1}{K} \sum_{(x,y)\in S_{xy}} G(x,y) \\ \frac{1}{K} \sum_{(x,y)\in S_{xy}} B(x,y) \end{bmatrix}$$

  - Each colour plane (R G and B) is smoothed independently using a spatial filter identical to the one we used for grey scale images
  - a mask that gives each pixel a new value that is the average of the surrounding pixels

Smoothing

- The techniques used for smoothing grey scale images can be used for colour images
  - Alternatively we can work in the HSV colour space. In this case we would only apply the smoothing filter to the intensity component.
  - As smoothing in RGB colour space averages colour pixel values the smoothing will result in the colour of each pixel being the average of the colour of the neighbours but using HSV the colour and saturation of each pixel remains unchanged

Smoothing

Red

Green

Blue

Smoothing

Hue

Saturation

Intensity

Smoothing

Averaging R,G and B

Averaging Intensity

Difference

Sharpening

- We can use the Laplacian to add a smoothed image to the original image as with grey scale images and again we can either apply the Laplacian to each colour plane in RGB colour space

The Laplacian of Vector **c** :

$$\nabla^2[\mathbf{c}(x,y)] = \begin{bmatrix} \nabla^2 R(x,y) \\ \nabla^2 G(x,y) \\ \nabla^2 B(x,y) \end{bmatrix}$$

Sharpening

- Sharpening by applying the Laplacian to the intensity component in HSV colour space

Sharpening R,G and B

Sharpening Intensity

Difference

Segmentation

- Segmentation is normally performed in HSV (HSI) colour space as this separates colour as a separate component
- Colour is often a very useful way of identifying objects in an image
  - Segmentation in HIS Colour Space
  - Segmentation in RGB Vector Space
  - Colour Edge Detection

What we have learnt

- Use of different colour spaces
- Pseudocolouring of images to improve understanding
  - Image slicing
  - Using transformation functions
- Intensity transformation
- Colour complement transformation
- Correcting image tone
- Correcting image colours
- Smoothing and sharpening colour images

Image and video processing

Image coding

Dr. Yi-Zhe Song

EBU723U

The agenda

- JPEG
- JPEG vs. JPEG2000
- JPEG2000 details
  - 2D wavelet decomposition
  - Tier
  - Layer
  - Region of interest (ROI)

Need for Image Compression

- For storage
  - A single high-quality image - 25-50 MB file
  - Typical uncompressed colour image requires about 1MB storage
- For transmission over the Internet, local networks and other networks with restricted bandwidth

Image Storage System

Lossless Compression

Lossless Compression

reorganising pixels – to be more compact

Lossless Compression

- A very basic method of lossless compression is Run Length Encoding
  - 11111111111111111110000011111111
    - 4 Bytes long
  - 18,5,9
    - 3 Bytes long

Lossless Compression

- Compresses data by storing it in a more appropriate way
- Rarely gives compression better than 2:1
- Guaranteed not to lose important data

Lossy Compression

- Provides compression by removing parts of the data
- Takes advantage of the properties of the human sensory system – vision and hearing
- Tries to remove data without it being noticed e.g.
  - The eye is relatively poor in distinguishing differences in chrominance (changes in colour)

JPEG Introduction - The background

- A standard image compression method is needed to enable interoperability of equipment from different manufacturer
- It is the first international digital image compression standard for continuous-tone images (greyscale or colour)
- The history of JPEG – the selection process

JPEG Introduction - The background

- JPEG standard is a collaboration among :
  - International Telecommunication Union (ITU)
  - International Organization for Standardization (ISO)
  - International Electrotechnical Commission (IEC)

- The official names of JPEG :
  - Joint Photographic Experts Group
  - ISO/IEC 10918-1 Digital compression and coding of continuous-tone still image
  - ITU-T Recommendation T.81

JPEG Introduction - The background

- JPEG have the following modes :
  - Lossless mode, predictive coding
  - Sequential mode, DCT-based coding
  - Progressive mode, DCT-based coding
  - Hierarchical mode

- Baseline system
  - Sequential mode, DCT-based coding, Huffman coding for entropy encoding
  - The most widely used mode in practice

JPEG overview

Baseline JPEG encoder

JPEG image compression

- The compression ratio of lossless methods (e.g., Huffman, Arithmetic) is not high enough for image and video compression

- JPEG uses lossy compression methods, by reducing
  - chrominance information
  - high-frequency information

Colour space conversion

(R,G,B)

(Luminance,
Chrominance)

Colour space conversion

- Transform function:

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.299000 & 0.587000 & 0.114000 \\ -0.168736 & -0.331264 & 0.500002 \\ 0.500000 & -0.418688 & -0.081312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}$$

(a) translate from RGB to YC_bC_r

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.0 & 0.0 & 1.40210 \\ 1.0 & -0.34414 & -0.71414 \\ 1.0 & 1.77180 & 0.0 \end{bmatrix} \begin{bmatrix} Y \\ C_b - 128 \\ C_r - 128 \end{bmatrix}$$

(b) translate from YC_bC_r to RGB

- *Y*: the luminance represents the brightness
- *Cb*: the difference between the gray and blue
- *Cr*: the difference between the gray and red

Downsampling formats of YCbCr

JPEG overview

- DCT of each 8x8 pixel array $f(x,y) \rightarrow_T F(u,v)$
- Quantization using a table or using a constant
- Zig-zag scan to exploit redundancy
- Differential Pulse Code Modulation (DPCM) on the *DC component* and
- Run Length Coding (RLC) of the *AC components*
- Entropy coding of the final output

Recap: Fourier theory

- Any complex waveform can be equated to an infinite sum of simple sinusoidal waves of varying frequencies and amplitudes.

Ref. - JPEG baseline structure

Why does it work?

- Lossy encoding
- HVS is generally more sensitive to low frequencies
- Natural images

Frequency sensitivity of Human Visual System

Discrete Cosine Transform

DCT

- changes spatial intensity values to spatial frequency values.
- roughly arranges values from lowest frequency to highest frequency:
  - lowest frequencies → represent coarse details
  - highest frequencies → represent fine details
    → some high frequency parts can be dropped
- exploits features of the human eye
  - the eye is unable to perceive brightness levels above or below certain thresholds
  - gentle gradations of brightness of colour are more important to the eye than abrupt changes

Discrete Cosine Transform

Forward DCT:

$$F(u,v) = \frac{1}{4} C(u)C(v) \sum_{x=0}^{7} \sum_{y=0}^{7} f(x,y) \cos\left[\frac{\pi(2x+1)u}{16}\right] \cos\left[\frac{\pi(2y+1)v}{16}\right]$$

for $u = 0,\ldots,7$ and $v = 0,\ldots,7$

where $C(k) = \begin{cases} 1/\sqrt{2} & \text{for } k = 0 \\ 1 & \text{otherwise} \end{cases}$

Inverse DCT:

$$f(x,y) = \frac{1}{4} \sum_{u=0}^{7} \sum_{v=0}^{7} C(u)C(v)F(u,v) \cos\left[\frac{\pi(2x+1)u}{16}\right] \cos\left[\frac{\pi(2y+1)v}{16}\right]$$

for $x = 0,\ldots,7$ and $y = 0,\ldots,7$

2D DCT basis

- The DCT coefficients as an array. Top left (DC coefficient) represents least detail and bottom right the finest detail

- In frequency terms bottom right is the highest frequency component

The 8x8 DCT basis

Discrete Cosine Transform

Example :

the luminance of an image

48	39	40	68	60	38	50	121
149	82	79	101	113	106	27	62
58	63	77	69	124	107	74	125
80	97	74	54	59	71	91	66
18	34	33	46	64	61	32	37
149	108	80	106	116	61	73	92
211	233	159	88	107	158	161	109
212	104	40	44	71	136	113	66

8x8 values of luminance

DCT

699.25	43.18	55.25	72.11	24.00	-25.51	11.21	-4.14
-129.78	-71.50	-70.26	-73.35	59.43	-24.02	22.61	-2.05
85.71	30.32	61.78	44.87	14.84	17.35	15.51	-13.19
-40.81	10.17	-17.53	-55.81	30.50	-2.28	-21.00	-1.26
-157.50	-49.39	13.27	-1.78	-8.75	22.47	-8.47	-9.23
92.49	-9.03	45.72	-48.13	-58.51	-9.01	-28.54	10.38
-53.09	-62.97	-3.49	-19.62	56.09	-2.25	-3.28	11.91
-20.54	-55.90	-20.59	-18.19	-26.58	-27.07	8.47	0.31

8x8 DCT coefficiences

Ref. - JPEG baseline structure

Quantization

- Why quantization?
  - to achieve further compression by representing DCT coefficients with no greater precision than is necessary to achieve the desired image quality
  - Generally, the “high frequency coefficients” has larger quantization values
  - Quantization makes most coefficients to be zero, it makes the compression system efficient, but it’s the main source that make the system “lossy”

$$F_q(u,v) = Round\left(\frac{F(u,v)}{Q(u,v)}\right)$$

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

Luminance and Chrominance quantization matrix

Quantization

Example :

699.25	43.18	55.25	72.11	24.00	-25.51	11.21	-4.14
-129.78	-71.50	-70.26	-73.35	59.43	-24.02	22.61	-2.05
85.71	30.32	61.78	44.87	14.84	17.35	15.51	-13.19
-40.81	10.17	-17.53	-55.81	30.50	-2.28	-21.00	-1.26
-157.50	-49.39	13.27	-1.78	-8.75	22.47	-8.47	-9.23
92.49	-9.03	45.72	-48.13	-58.51	-9.01	-28.54	10.38
-53.09	-62.97	-3.49	-19.62	56.09	-2.25	-3.28	11.91
-20.54	-55.90	-20.59	-18.19	-26.58	-27.07	8.47	0.31

$F(u,v)$

8x8 DCT coefficiences

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

$Q(u,v)$

Quantization matrix

Quantization

Example :

43.70	3.93	5.52	4.51	1.00	-0.64	0.22	-0.07
-10.82	-5.96	-5.02	-3.86	2.29	-0.41	0.38	-0.04
6.12	2.33	3.86	1.87	0.37	0.30	0.22	-0.24
-2.91	0.60	-0.80	-1.92	0.60	-0.03	-0.26	-0.02
-8.75	-2.25	0.36	-0.03	-0.13	0.21	-0.08	-0.12
3.85	-0.26	0.83	-0.75	-0.72	-0.09	-0.25	0.11
-1.08	-0.98	-0.04	-0.23	0.54	-0.02	-0.03	0.12
-0.29	-0.61	-0.22	-0.19	-0.24	-0.27	0.08	0.00

$$\frac{F(u,v)}{Q(u,v)}$$

44	4	6	5	1	-1	0	0
-11	-6	-5	-4	2	0	0	0
6	2	4	2	0	0	0	0
-3	1	-1	-2	1	0	0	0
-9	-2	0	0	0	0	0	0
4	0	1	-1	-1	0	0	0
-1	-1	0	0	1	0	0	0
0	-1	0	0	0	0	0	0

$$F_q(u,v) = Round\left(\frac{F(u,v)}{Q(u,v)}\right)$$

Ref. - JPEG baseline structure

Zig-zag scan

- Zig-zag scan
  - To group low frequency coefficients in top of vector
  - Maps 8 x 8 to a 1 x 64 vector

Zig-zag scan

Example :

44	4	6	5	1	-1	0	0
-11	-6	-5	-4	2	0	0	0
6	2	4	2	0	0	0	0
-3	1	-1	-2	1	0	0	0
-9	-2	0	0	0	0	0	0
4	0	1	-1	-1	0	0	0
-1	-1	0	0	1	0	0	0
0	-1	0	0	0	0	0	0

$$F_q(u,v)$$

Zig-Zag Reordering :

44,
4,-11,
6,-6,6,
5,-5,2,-3,
-9,1,4,-4,1,
-1,2,2,-1,-2,4,
-1,0,0,-2,0,0,0,
0,0,0,1,0,1,-1,0,
-1,0,-1,0,0,0,0,
0,0,0,-1,0,0,
0,1,0,0,0,
0,0,0,0,
0,0,0,
0,0,
0

Ref. - JPEG baseline structure

DPCM on DC Components

- Differential Pulse Code Modulation (DPCM)
  - Encode the difference between the current and previous 8x8 block
  - The DC component value in each 8x8 block is large and varies across blocks, but is often close to that in the previous block.

Encode :

$Diff_i = DC_i - DC_{i-1}$

Decode :

$DC_i = DC_{i-1} + Diff_i$

Ref. - JPEG baseline structure

RLE on AC Components

- Run Length Coding (RLC)
  - The 1x64 vectors have a lot of zeros in them, more so towards the end of the vector.
    - Higher up entries in the vector capture higher frequency (DCT) components which tend to capture less of the relevant content
    - Could have been as a result of using a quantization table
  - Encode a series of 0s as a (*skip,value*) pair, where *skip* is the number of zeros and *value* is the next non-zero component

RLE on AC Components

- Example 1

63 AC coefficient:

57, 45, 0, 0, 0, 0, 23, 0, -30, -8, 0, 0, 1, 0, $\underbrace{0, 0, 0, 0, 0, 0, 0,..., 0}_{\text{50 zeros}}$

Run Length Coding :

(0,57) ; (0,45) ; (4,23) ; (1,-30) ; (0,-8) ; (2,1) ; EOB → (0,0)

end of bitstream

- Example 2

63 AC coefficient:

57, $\underbrace{0, 0, ... , 0}_{\text{18 zeros}}$, 3, 0, 0, 0, 0, 2, $\underbrace{0, 0, ... , 0}_{\text{33 zeros}}$, 895, EOB

Run Length Coding :

(0,57) ; (15,0) ; (2,3) ; (4,2) ; (15,0) ; (15,0) ; (1,895) ; (0,0)

Ref. - JPEG baseline structure

Entropy (Huffman) coding

- DC components are coded as `(SIZE,Value)`
- AC components are coded as `(RUNLENGTH, SIZE,Value)`

size	Bits for the value	value
1	-1,1	0,1
2	-3,-2,2,3	00,01,10,11
3	-7,-6,-5,-4,4,5,6,7	000,001,010,011,100,101,110,111
4	-15,...,-8,8,...,15	0000,...,0111,1000,...,1111
5	-31,...,-16,16,...31	00000,...,01111,10000,...,11111
6	-63,...,-32,32,...63	000000,...,011111,100000,...,111111
7	-127,...,-64,64,...,127	0000000,...,0111111,1000000,...,1111111
8	-255,..,-128,128,..,255	...
9	-511,..,-256,256,..,511	...
10	-1023,..,-512,512,..,1023	...
11	-2047,..,-1024,1024,..,2047	...

The codeword table for VALUE of the coefficients

Entropy (Huffman) coding

run/size	code length	code word
0/0	4	1010
...		
0/6	7	1111000
...		
0/10	16	1111111110000011
1/1	4	1100
...		
1/5	11	11111110110
...		
4/5	16	1111111110011000
...		
15/10	16	1111111111111110

The Run/Size Huffman table for the luminance AC coefficients

Entropy (Huffman) coding

Example

Run length coding of 63 AC coefficients :

(0,57) ; (0,45) ; (4,23) ; (1,-30) ; (0,-8) ; (2,1) ; EOB

Encode the right value of these pair as category and bits for the value, except the special markers like (0,0) or (15,0) :

(0,6,111001) ; (0,6,101101) ; (4,5,10111);

(1,5,00001) ; (0,4,0111) ; (2,1,1) ; EOB

Entropy (Huffman) coding

SIZE	Code Length	Code
0	2	00
1	3	010
2	3	011
3	3	100
4	3	101
5	3	110
6	4	1110
7	5	11110
8	6	111110
9	7	1111110
10	8	11111110
11	9	111111110

The Huffman table for SIZE of the luminance DC coefficients

- DC components are differentially coded as (SIZE,Value). The code for a SIZE is derived from the table on the left

Example: If a DC component is 40 and the previous DC component is 48 → the difference is -8 → it is coded as:

1010111

0111: The value for representing –8

Ref.: The codeword table for VALUE of the coefficients

101: The size from the same table reads 4. The corresponding code from the table on the left is 101.

Entropy (Huffman) coding

The DC difference

- 511

Ref.: The codeword table for VALUE of the coefficients

The AC coefficients :

(0,57) ; (0,45) ; (4,23) ; (1,-30) ; (0,-8) ; (2,1) ; EOB

Encode the left two value in () using Huffman encoding :

- 1111110 000000000 , 1111000 111001 , 1111000 101101 , 1111111110011000 10111 , 11111110110 00001 , 1011 0111 , 11100 1 , 1010

Ref. - JPEG baseline structure

Next agenda

- JPEG
- JPEG vs. JPEG2000
- JPEG2000 details
  - 2D wavelet decomposition
  - Tier
  - Layer
  - Region of interest (ROI)

JPEG2000

- JPEG2000
  - Created in 2000 by JPEG committee
  - File extension:
    - jp2 for ISO/IEC 15444-1 conforming files
    - image/jp2 for MIME type
  - new image coding system that uses state-of-the-art compression techniques based on wavelet technology.
  - Its architecture lends itself to a wide range of uses from portable digital cameras through to advanced pre-press, medical imaging, …

Why JPEG2000?

- Very high compression ratios
  - e.g., < 0.25 bpp
- Lossy and lossless compression
  - There was no standard enabling lossy and lossless compression efficiently with the same approach
- Computer-generated images
  - JPEG was developed for natural images and is not appropriate for synthetic images
- Composite images
  - JPEG has limited performance when compressing composite images
- Error resilience
- Scalable and flexible encoding/decoding

The JPEG2000 standard

- Part 1 → Core coding system
  - intended as royalty and license-fee free - NOT patent-free
- Part 2 → Extensions
  - adds more features and sophistication to the core
- Part 3 → Motion JPEG 2000
- Part 4 → Conformance
- Part 5 → Reference software
  - Java and C implementations are available
- Part 6 → Compound image file format
  - document imaging, for pre-press and fax-like applications, etc.
- Part 7 → has been abandoned
- Part 8 → JPSEC (security aspects)
- Part 9 → JPIP (interactive protocols and API)
- Part 10 → JP3D (volumetric imaging)
- Part 11 → JPWL (wireless applications)
- Part 12 → ISO Base Media File Format (common with MPEG-4)

JPEG2000: applications

- Internet
- Mobile
- Printing
- Scanning
- Digital Photos
- Aerial imaging
- Fax
- Medical imaging
- E-commerce

JPEG2000: compression paradigm

A single compressed file can be decompressed to different sizes or qualities using different parts of the compressed file

Next agenda

- JPEG
- JPEG vs. JPEG2000
- JPEG2000 details
  - 2D wavelet decomposition
  - Tier
  - Layer
  - Region of interest (ROI)

JPEG - 0.125 bpp (192:1)

JPEG 2000 - 0.125 bpp (192:1)

JPEG - 0.25 bpp (96:1)

JPEG 2000 - 0.25 bpp (96:1)

JPEG - 0.5 bpp (48:1)

JPEG 2000 - 0.5 bpp (48:1)

JPEG - 1 bpp (8:1)

Dear Pam,
I was delighted to hear from you last week. Patti and I had a wonderful time during our week-long summer vacation. The weather was excellent, and the food was absolutely exquisite. I hope that we can repeat this next year and that you will join us too.
We came back with a lot of fantastic memories, which we would like to share with you through some snapshots that we took.
Summer of 1994.
I Love New York.
Our favorite is this picture of us aboard the "Top Hat", which I have pasted into this letter using some really neat advanced digital imaging technology on my home computer. We will ship the rest to you on a CD-ROM soon. Wishing you the best.
Love,
Susan
We came back with a lot of
like to share with you thro
Summer of 1994.

JPEG 2000 - 1 bpp (8:1)

Next agenda

- JPEG
- JPEG vs. JPEG2000
- JPEG2000 details
  - 2D wavelet decomposition
  - Tier
  - Layer
  - Region of interest (ROI)

JPEG2000 overview

DWT betters DCT in that it contains local information

Pre-processing

- In tiling, the input image is partitioned into rectangular and non-overlapping tiles of equal size
  - Except possibly for those tiles at the image borders
  - Each tile is compressed independently using its own set of specified compression parameters.
- Level offset pre-processing

Colour Component Transformation

- JPEG-2000: transformation from RGB to YC_bC_r or YUV
  - Irreversible Colour Transform (ICT):
    - uses the well known YC_bC_r color space. It is called "irreversible" because it has to be implemented in floating or fix-point and causes round-off errors.
  - Reversible Colour Transform (RCT):
    - uses a modified YUV color space that does not introduce quantization errors, so it is fully reversible.

Two methods of compression

- DCT-based coder
  - New baseline JPEG algorithm required for backward compatibility with existing JPEG
- Wavelet-based coder
  - This method permits coding of still images with high coding efficiency as well as spatial and SNR (signal-to-noise ratio) scalability at fine granularity

Discrete Wavelet Transform

- Wavelets (portions of waves) are discretely sampled
- Wavelet transforms capture both spatial (location in time) and frequency information
- There is a reversible version of the transform that can give lossless compression and an irreversible one that gives higher but lossy compression

Transformation in JPEG2000 Part 1

- Advantages of DWT
  - Multi-resolution image representation is inherent to DWT
  - The full-frame nature of the transform de-correlates the image across a larger scale and eliminates blocking artifacts at high compression
  - Use of integer DWT filters allows for both lossless and lossy compression within a single compressed bit-stream
  - DWT provides a frequency band decomposition of the image where each sub-band can be quantized according to its visual importance

L: lowpass
H: highpass

Wavelet Decomposition

Resolution 0: LL3

Res 1: Res 0 + LH3 + HL3 + HH3

Res 2: Res 1 + LH2 + HL2 + HH2

Res 3: Res 2 + LH1 + HL1 + HH1

Wavelet decomposition

- Several passes of the transform give successively higher quality results.
- LL3 is the lowest quality (or size)
- Adding the coefficient blocks HL3, HH3 and LH3 gives higher quality or a larger image
- Adding HL2, HH2 and LH2 increases quality again
- Full quality is produced when HL1, HH1 and LH1 are added

Decomposition: multiresolution

Decomposition: multiresolution

Decomposition: multiresolution

Decomposition: multiresolution

Another example of the discrete wavelet transform

- The original image is high-pass filtered, yielding the three large images, each describing local changes in brightness (details) in the original image.
- It is then low-pass filtered and downscaled, yielding an approximation image;
- This image is high-pass filtered to produce the three smaller detail images, and low-pass filtered to produce the final approximation image in the upper-left.

Ref. - JPEG 2000 structure

Quantization in JPEG2000 Part 1

- Quantization of all the wavelet coefficients
  - uniform quantization with dead-zone
  - for each sub-band *b*, a basic quantizer step size is selected *by the user* and is used to quantize all the coefficients in that sub-band
  - choice of the quantizer step size for each sub-band can be based on *visual models*.
  - This gives higher compression ratio for same visual quality.

Uniform scalar quantizer with dead-zone

- Quantization rule

$$q_j(m,n) = \mathrm{sign}(y_j(m,n)) \left\lfloor \frac{|W_j(m,n)|}{\Delta_j} \right\rfloor$$

y: input to the quantizer
Δ_j: scalar quantizer step size
q: resulting quantizer index
sign(y): sign of y,
$W_j(m,n)$: a DWT coefficient in subband j

Next agenda

- JPEG
- JPEG vs. JPEG2000
- JPEG2000 details
  - 2D wavelet decomposition
  - Tier
  - Layer
  - Region of interest (ROI)

Ref. - JPEG 2000 structure

Entropy Coding in JPEG2000 Part 1

- EBCOT: Embedded Block Coding with Optimized Truncation - Tier 1 and Tier 2

Entropy Coding in JPEG2000 Part 1

- Code block: each sub-band from each tile component is partitioned into code blocks
- Bitplane: each code block will be independently entropy-encoded bitplane by bitplane

Bit-plane Conversion

Embedded quantization by bit-plane coding - EBCOT

Fractional bitplane coding

To compress each individual bit plane

- each bit-plane of each block of a sub-band is encoded in three sub bit plane passes instead of a single pass.
  - Significance propagation pass
  - Magnitude refinement pass
  - Clean up pass
- the bit-stream can be truncated at the end of each pass.

Arithmetic coding

- Then, the binary value of a sample in a block of a bitplane of a subband is coded as a binary symbol with the JBIG2 MQCoder.

Performance measures

$$RMSE = \sqrt{\frac{\sum_{x=0}^{W-1}\sum_{y=0}^{H-1}\left[f(x,y) - f'(x,y)\right]^2}{WH}}$$

W : the width
H : the height of the original image

$$PSNR = 20\log_{10}\frac{255}{MSE}$$

Sig. Prop.	=	0
Refine	=	0
Cleanup	=	21
Total Bytes	=	21

Bit plane **1**

Compression ratio = 12483 : 1

RMSE = 39.69 PSNR = 16.16 db

Specific bitplanes of all DWT coefficients

Sig. Prop.	=	38
Refine	=	13
Cleanup	=	57
Total Bytes	=	108

Bit plane **3**

Compression ratio = 1533 : 1

RMSE = 21.59 PSNR = 21.45 db

Specific bitplanes of all DWT coefficients

Sig. Prop.	=	224
Refine	=	73
Cleanup	=	383
Total Bytes	=	680

Bit plane **5**

Compression ratio = 233 : 1

RMSE = 12.11 PSNR = 26.47 db

Specific bitplanes of all DWT coefficients

Sig. Prop.	=	2315
Refine	=	932
Cleanup	=	2570
Total Bytes	=	5817

Bit plane **8**

Compression ratio = 23 : 1

RMSE = 4.18 PSNR = 35.70 db

Specific bitplanes of all DWT coefficients

Sig. Prop.	=	4593
Refine	=	1925
Cleanup	=	5465
Total Bytes	=	11983

Bit plane **9**

Compression ratio = 11.2 : 1

RMSE = 2.90 PSNR = 38.87 db

Specific bitplanes of all DWT coefficients

Tier 2 role

- Tier 1
  - generates a collection of bit-streams
    - One independent bit-stream for each code-block
    - Each bit-stream is embedded
- Tier 2
  - multiplexes the bit-streams for inclusion in the code-stream and
  - signals the ordering of the resulting coded bit-plane passes in an efficient manner
  - coded data can be rather easily parsed
  - enables SNR, resolution, spatial, ROI and arbitrary progression scalability

Ref. - JPEG 2000 structure

Example of bit-plane pass coded data

256x256 image

Code-blocks (64x64)

Example of bit-plane pass coded data

JPEG 2000 Progressive Coding

Progressive by resolution

Lowest resolution, highest quality

Medium resolution, highest quality

Highest resolution, highest quality

Next agenda

- JPEG
- JPEG vs. JPEG2000
- JPEG2000 details
  - 2D wavelet decomposition
  - Tier – resolution
  - Layer – qualilty
  - Region of interest (ROI)

Layers

- Layer
  - a collection of some consecutive bit-plane coding passes from all code-blocks in all sub-bands and components
  - Each code-block can contribute an arbitrary number of bit-plane coding passes to a layer
  - Each layer successively increases the image quality. Most often associated with SNR or visual quality levels
  - Layers are explicitly signalled and can be arbitrarily determined by the encoder
  - The number of layers can range from 1 to 65535. Typically around 20. Larger numbers are intended for interactive sessions were each layer is generated depending on user feedback

Example of layer organisation

Layer (SNR) progressive example

Layer (SNR) progressive example

Layer (SNR) progressive example

Resolution progressive example

Resolution progressive example

Resolution progressive example

Resolution progressive example

Final agenda

- JPEG
- JPEG vs. JPEG2000
- JPEG2000 details
  - 2D wavelet decomposition
  - Tier
  - Layer
  - Region of interest (ROI)

Region of Interest coding principle

- Region of Interest (ROI)
  - allows a non-uniform distribution of quality. The ROI is coded with a higher quality than the background (BG)
  - A higher compression ratio can be achieved with same or higher quality inside ROIs.
  - Static ROIs
    - defined at encoding time
    - commonly referred to as ROI coding
    - suitable for storage, fixed transmission, remote sensing
  - Dynamic ROIs
    - defined interactively by a user in a client/server situation during a progressive transmission
    - dynamic generation of layers matching the user's request
    - suitable for telemedicine, PDAs, mobile communications

ROI mask generation

ROI example

256:1

ROI

45:1 (almost all ROI decoded)

No ROI

ROI covers 5% of image, 2 lowest resolution levels in ROI mask.
Magnified portion shown

ROI example

16:1

ROI

4:1 (complete decode)

No ROI

ROI covers 5% of image, 2 lowest resolution levels in ROI mask.
Magnified portion shown

What did we learn?

- JPEG
- JPEG vs. JPEG2000
- JPEG2000 details
  - 2D wavelet decomposition
  - Tier
  - Layer
  - Region of interest (ROI)