Image Structure 4

Feature Detection and Matching
4F12: Computer Vision

Combined slides from lecture and Q & A

Instructor: Samuel Albanie
Based on course material authored by Roberto Cipolla

VE Tag = Very Examinable
NE Tag = Non Examinable

Image Structure 4: Q&A

Feature Detection and Matching

4F12: Computer Vision

We will start at **10:05 am** (to allow everyone to join from previous in-person Q&A/jiu-jitsu/archaeological excavations/welding/yoga classes etc.)

If you are reading this before 10:05 am, you may wish to apply the computer vision researcher algorithm:

If 5 mins spare: profile your training code. Uh oh, it's I/O bound. Find out how to use LMDB. If this fails, try moving your files to a ramdisk. If this fails, hassle your supervisor to pay for faster servers.

If 3 mins spare: check out the latest NVIDIA Megatron model blog post (it has 530 billion parameters)!

If 1 min spare: Drink half a glass of water.

Recording notice

We will attempt to repeat the following protocol used in lecture 2:

1. We will record the "summary" part.
2. The rest will be non-recorded.

Examples paper

We will work through solutions during after slides summary

Instructor: Samuel Albanie

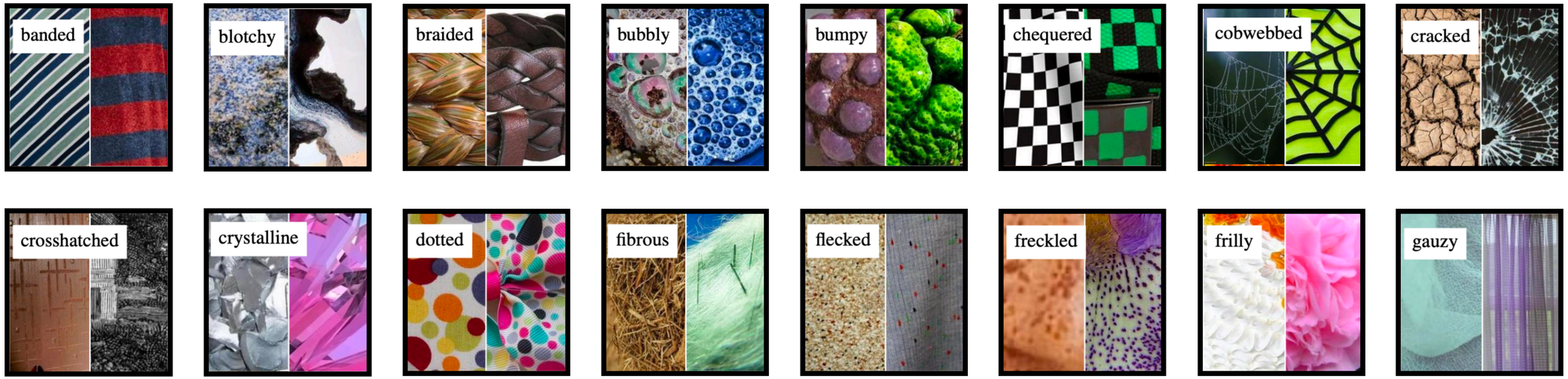
Based on course material authored by Roberto Cipolla

Image Textures

What is a texture?

Definition: a texture is a visual pattern on an infinite 2-D plane which, at some scale, has a stationary distribution[1] (**note**: there isn't a universally accepted definition of texture, but this one is useful).

Example textures



and many others....

Definition: [1]Efros and Leung. "Texture synthesis by non-parametric sampling." ICCV 1999
Images source: Cimpoi et al. Deep filter banks for texture recognition and segmentation, CVPR 2015

Historical context: preattentive vision NE

Texture and vision

Perception of **texture** has received a great deal of attention due to its potential to yield clues about how the visual system is able to group stimuli together in a way that supports their interpretation.

Gestalt

In the early 20th century, the *Gestalt school* of psychologists studied how humans perform perceptual grouping, and proposed a set of "laws" that govern how preattentive vision will group elements together.

Example: the law of symmetry

{ }[ ]{ }

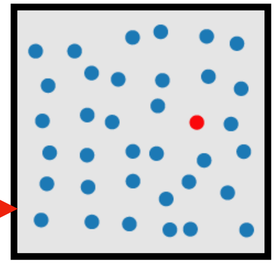
Six elements, but we see three groups.

Example: the law of proximity

Many dots, but we see four groups.

Preattentive processing is the kind of fast, parallel vision that happens (mostly) before attention is involved.

Question: Is there a red dot present?



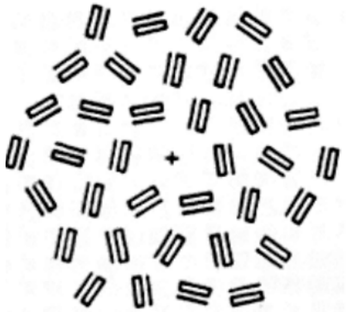
Julesz Textons

Between the 1960s and the 1980s, Bela Julesz studied the statistics of the kinds of textures humans could differentiate preattentively, and found the answers could not be explained by Gestalt rules.

Find  among 

Constant time

Find  among 



Time grows linearly with number of elements

Julesz defined textons to be the basic units recognised by preattentive vision that enable texture discrimination (these included oriented bars, crossings, terminators and other low-level structures).

References: M. Wertheimer, "Laws of organization in perceptual forms." (1938); Julesz, Bela. "Textons, the elements of texture perception, and their interactions." Nature 290.5802 (1981): 91-97.
Image sources: (red dot) Perception in Visualization, C.G. Healey; https://en.wikipedia.org/wiki/Gestalt_psychology#/media/File:Gestalt_proximity.svg; Zhu et al. "What are textons?." IJCV, 2005

Textons revisited

Describing textures

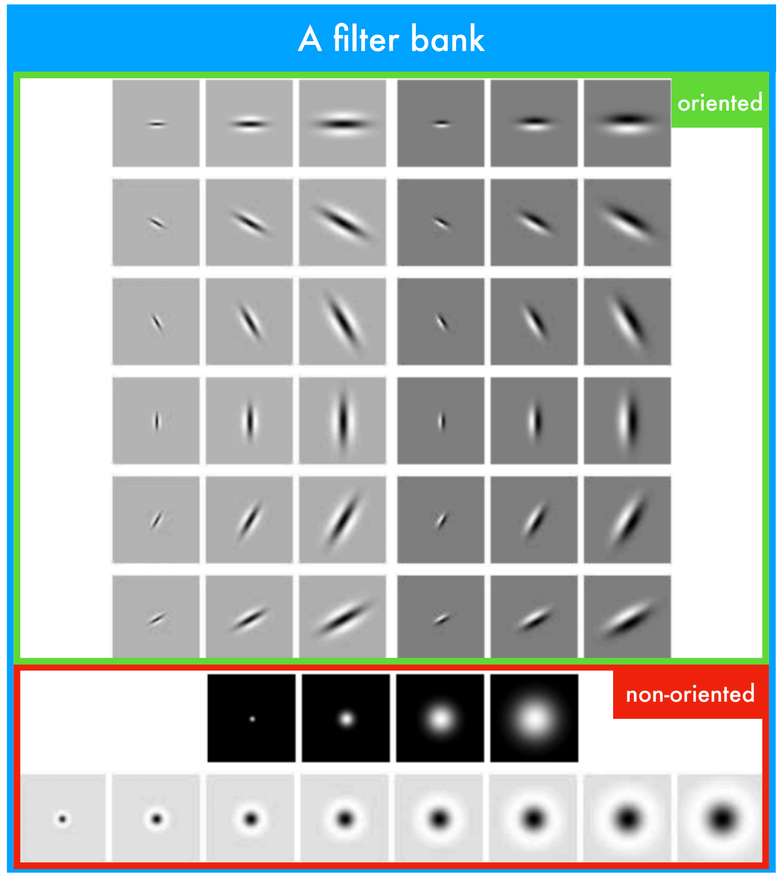
One way to characterise texture is through its response to a filter bank.

The example on the right consists of 48 filters:

- 8 LoG filters and 4 Gaussian filters at different scales to provide non-oriented responses
- 36 oriented filters at 6 angles, 3 scales, and 2 phases.

The two phases of oriented filters are first and second derivatives of Gaussians on the minor axis and elongated Gaussians on the major axis, and thus detect edges or bars respectively along their major axes.

The **descriptor** is simply the *concatenated responses* of all of the filters in the filter bank at a pixel.



Malik Textons

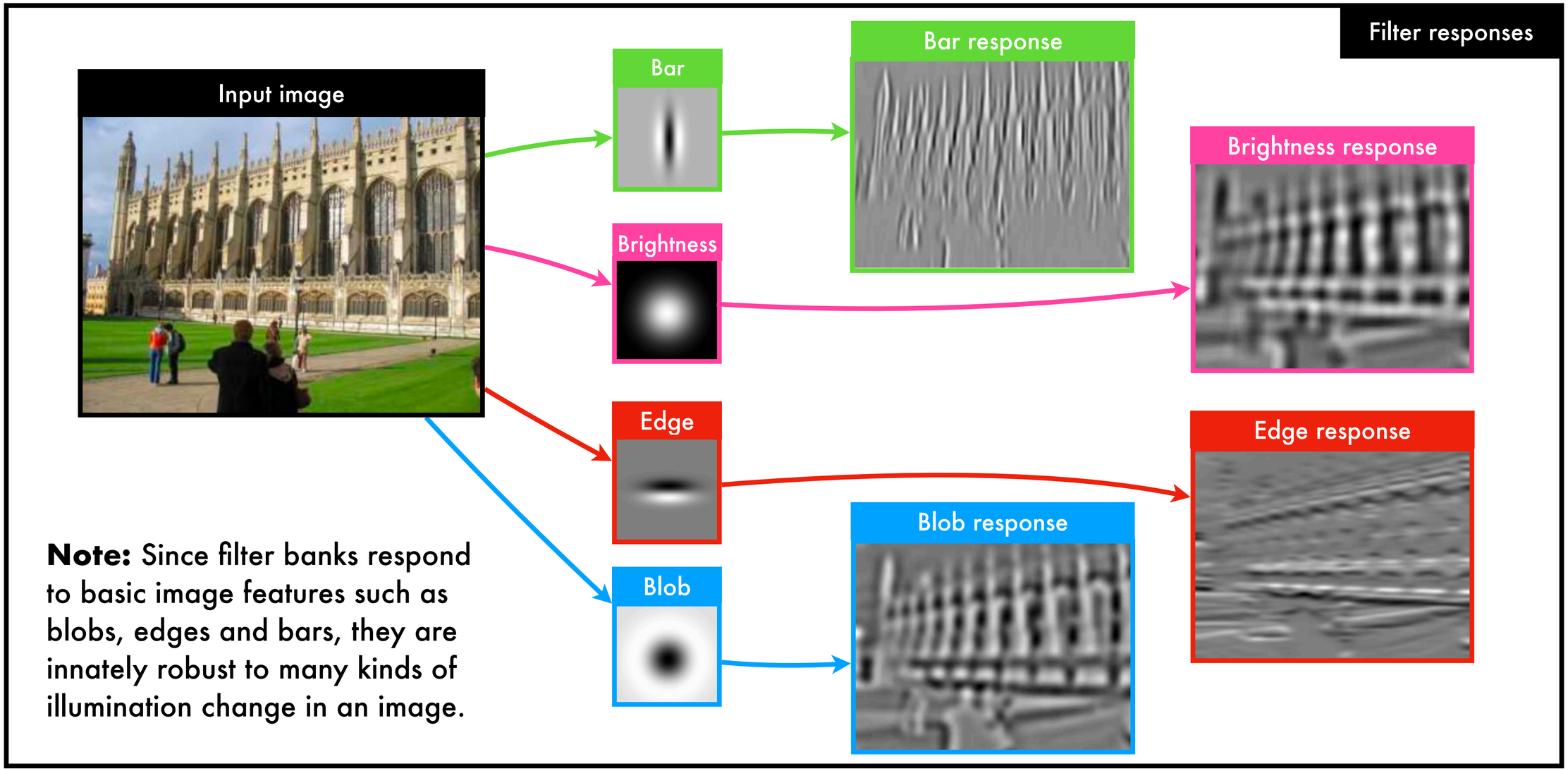
Julesz provided a qualitative definition of *textons*, rather than a mathematical one.

Malik et al. (2001) proposed to redefine textons as the prototypes that result from **clustering** the responses of a filter bank.

Empirically, they found that these tend to correspond to oriented bars and terminators (aligning with elements of the original definition of Julesz).

Reference: J. Malik et al. "Contour and texture analysis for image segmentation." IJCV (2001)

Filter banks visualised



Note: Since filter banks respond to basic image features such as blobs, edges and bars, they are innately robust to many kinds of illumination change in an image.

Link to Deep Learning

Deep Learning

In our study of image structure, the raw image has been pre-processed through "hand-crafted" feature extractors (for edges, corners, textons).

The feature extractors were not learned directly from data.

Later in the course, we will learn a hierarchy of feature extractors just by looking at examples - from low-level to mid-level invariant representations up to object identities. This is called Deep Learning.

End of Image Structures 4: questions?

Example paper

Storage requirements

Question 1

Images are stored as pixel arrays of quantised intensity values. Typically each pixel has a brightness value in the range 0 (black) to 255 (white), and is stored as a single byte (8 bits).

Compute storage requirements (in bytes per second) for a stereo pair of HD video cameras grabbing grey-level images of size 1920 × 1080 pixels at 25 frames per second. Approximately how many pages of text require the same amount of storage as one second of stereo video?

Solution

The storage requirements for one stream is: width (in pixels) * height (in pixels) * frame rate, which gives

1920 * 1080 * 25 = 51,840,000 bytes per second (52 Megabytes per second)

So the storage for the stereo pair is approximately 104 Megabytes per second.

To estimate pages of text, we need to make some assumptions.

A reasonable guess could be that a page of text contains perhaps 50 lines with 80 characters per line (each requiring a single byte as an ASCII code). So a page of text contains 4000 bytes.

One second of stereo camera streaming data corresponds to 104 MB / 4kB = 25,500 pages (more than 20 copies of War and Peace)!

The reproducing property of Gaussians

Question 2

A commonly used 1D smoothing filter is the Gaussian: $g_\sigma(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$ where σ determines the size of the filter. Show that repeated convolutions with a series of 1D Gaussians, each with standard deviation of σ_i, is equivalent to a single convolution with a Gaussian of variance $\sum_i \sigma_i^2$

Solution

<u>Simplest approach:</u> take Fourier transform of the Gaussian and multiply.

Note that: $\mathcal{F}(g_\sigma)(\omega) = \exp\left(-\frac{\omega^2\sigma^2}{2}\right)$

So, $g_{\sigma_2}(x) \circledast g_{\sigma_1}(x) \leftrightarrow \mathcal{F}(g_{\sigma_1})(\omega) \cdot \mathcal{F}(g_{\sigma_2})(\omega) = \exp\left(-\frac{\omega^2\sigma_1^2}{2}\right) \cdot \exp\left(-\frac{\omega^2\sigma_2^2}{2}\right)$

$= \exp\left(-\frac{\omega^2(\sigma_1^2+\sigma_2^2)}{2}\right) = \mathcal{F}(g_{\sigma_3})(\omega) \leftrightarrow g_{\sigma_3}(x)$ where $\sigma_3 = \sqrt{\sigma_1^2+\sigma_2^2}$

So convolving with consecutive 1D convolutions with standard deviations of σ_i is equivalent to convolving with a 1D Gaussian with a standard deviation of $\sqrt{\sum_i \sigma_i^2}$, and hence a variance of $\sum_i \sigma_i^2$

Constructing the discrete Gaussian kernel

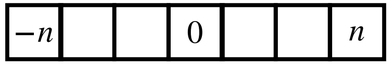
Question 3

A discrete approximation to a 1D Gaussian can be obtained by sampling the function $g_\sigma(x)$. In practice, samples are taken uniformly until the truncated values at the tails of the distribution are less than 1/1000 of the peak value.

(a) For $\sigma = 1$, show that the filter obtained in this way has a size of 7 pixels and coefficients given by: [0.004, 0.054, 0.242, 0.399, 0.242, 0.054, 0.004]. What property of the coefficients ensures that regions of uniform intensity are unaffected by smoothing?

Solution

Plan: we will construct our kernel to contain $2n+1$ elements:



We can find the value of n required so that the $(n+1)^{th}$ element (i.e. the truncated one) achieves the desired accuracy by solving the following:

$$\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(n+1)^2}{2\sigma^2}\right) < \frac{1}{1000}\left(\frac{1}{\sqrt{2\pi\sigma^2}}\exp(0)\right)$$

Element to be truncated

Peak value of the Gaussian occurs at $x = 0$

$$\implies \exp\left(-\frac{(n+1)^2}{2\sigma^2}\right) < \frac{1}{1000} \implies -\frac{(n+1)^2}{2\sigma^2} < \log\left(\frac{1}{1000}\right) = -6.91$$

The coefficients of the filter sum to one, so smoothing uniform regions has no effect.

$(n+1)^2 >= -13.81\sigma^2 \implies n > 3.7\sigma - 1$ (where n must be an integer - equivalently, we can say that n is the closest integer to $3.7\sigma - 0.5$)

We can apply this formula for $\sigma = 1$ to find $n = 3$ (yielding a kernel size of $2n+1 = 7$ pixels).

We sample the kernel values from the 1D Gaussian $g_\sigma(x)$ at the points $x = \{-3, -2, -1, 0, 1, 2, 3\}$. $\quad g_1(3) = \frac{1}{\sqrt{2\pi}} \cdot \exp\left(-\frac{9}{2}\right) = 0.004$

Constructing the discrete Gaussian kernel

Question 3

(b) Using the same truncation criterion, what would be the size of the discrete filter kernel for $\sigma = 5$? Show that, in general, the size of the kernel can be approximated as $2n + 1$pixels, where n is the nearest integer to $3.7\sigma - 0.5$.

Solution

We already found the formula to solve (a). Applying it with $\sigma = 5$ gives $n = \text{round}(3.7 \cdot 5 - 0.5) = 18$, so the kernel size is $2n + 1 = 37$ pixels.

Bonus programming question

In the current version of Python, what is the value of round(2.5)? **NE** **2**

Bonus programming question 2

In Python2, what is the value of round(2.5)? **NE** **3**

Bonus programming question 3

What is the value of numpy.round(2.5)? **NE** **2**

For details, see IEEE *754*: "Round half to even" **NE**

Question 3

(c) The filter is used to smooth an image as part of edge detection. What factors affect the choice of an appropriate value for σ?

Solution

1. **The scale at which we wish to analyse the image:** a small σ will respond to edges at a fine scale, whereas a larger σ will respond to edges at a large scale (so the choice depends on the scale we are interested in).

2. **Noise**: smoothing suppresses noise, but also erodes fine-detail. Images with lots of noise will require more smoothing to be interpretable.

3. **Computational cost:** larger convolutional kernels are more expensive.

References: https://slate.com/technology/2019/10/round-floor-software-errors-stock-market-battlefield.html

Discrete Convolution

Question 4

A row of pixels is smoothed with the discrete 1D Gaussian kernel given in question 3(a) ($\sigma = 1$):

46	45	45	48	50	53	55	57	77	99	118	130	133	134	133	132	132	132	133

Calculate $S(10)$, the smoothed value of the pixel $I(10)$ with intensity $I(10) = 118$.

Solution

The kernel is

0.004	0.054	0.242	0.399	0.242	0.054	0.004

We can apply it via:

0.004	0.054	0.242	0.399	0.242	0.054	0.004

Kernel

46	45	45	48	50	53	55	57	77	99	118	130	133	134	133	132	132	132	133

Image

Element-wise product ↓

0.228	4.158	23.96	47.08	31.46	7.182	0.536

Summing the results gives $S(10) = 114.6$, i.e. 115 to the nearest integer

Derivative of convolution theorem

Question 5

(a) Show that smoothing an intensity signal with a Gaussian and then differentiating the smoothed signal is equivalent to convolution with the derivative of a Gaussian: $\frac{d}{dx}[g_\sigma(x) \circledast I(x)] = g'_\sigma(x) \circledast I(x)$ where $g'_\sigma(x)$ is the first derivative of the Gaussian function.

(b) Hence, or otherwise, show how "edges" in an intensity function $I(x)$ can be localised at the zero-crossings of $g''_\sigma(x) \circledast I(x)$, where $g''_\sigma(x)$ is the second derivative of the Gaussian function.

Solution

(a) To demonstrate the theorem, we can pass the differentiation through the integral:

$$s'(x) = \frac{d}{dx}[g_\sigma(x) \circledast I(x)] = \frac{d}{dx}\left[\int_{-\infty}^{\infty} g_\sigma(x-u)I(u)du\right]$$

$$= \int_{-\infty}^{\infty} \frac{d}{dx}[g_\sigma(x-u)]I(u)du$$

$$= \int_{-\infty}^{\infty} g'_\sigma(x-u)I(u)du = g'_\sigma(x) \circledast I(x)$$

(b) Edges occur at the extrema of $\frac{d}{dx}[g_\sigma(x) \circledast I(x)]$. The extrema can be found where the second derivative is zero, i.e. $\frac{d^2}{dx^2}[g_\sigma(x) \circledast I(x)] = 0$

We can apply the theorem twice to give $\frac{d^2}{dx^2}[g_\sigma(x) \circledast I(x)] = g''_\sigma(x) \circledast I(x)$, so edges can be found at the zero-crossings of $g''_\sigma(x) \circledast I(x)$

Differentiation and 1D edge detection

Question 6

(i) Show how an approximation to the first-order spatial derivative of $S(x)$ can be obtained by convolving samples of $S(x)$ with the kernel $[0.5, 0, -0.5]$.

Solution

Suppose we want to differentiate the signal $S(x)$. We can perform a Taylor expansion about x:

$$S(x+\delta x) = S(x) + \delta x \frac{\partial S}{\partial x} + \text{higher order terms}$$

Take $\delta x = 1$ and $\delta x = -1$ to give:

$$S(x+1) = S(x) + \frac{\partial S}{\partial x} + \text{higher order terms}, \text{ and } S(x-1) = S(x) - \frac{\partial S}{\partial x} + \text{higher order terms}$$

Subtract the latter term from the former term and divide by two to give $\frac{dS}{dx} \approx \frac{S(x+1) - S(x-1)}{2}$ which we can implement via convolution with $[0.5, 0, -0.5]$ (remember the flipping!)

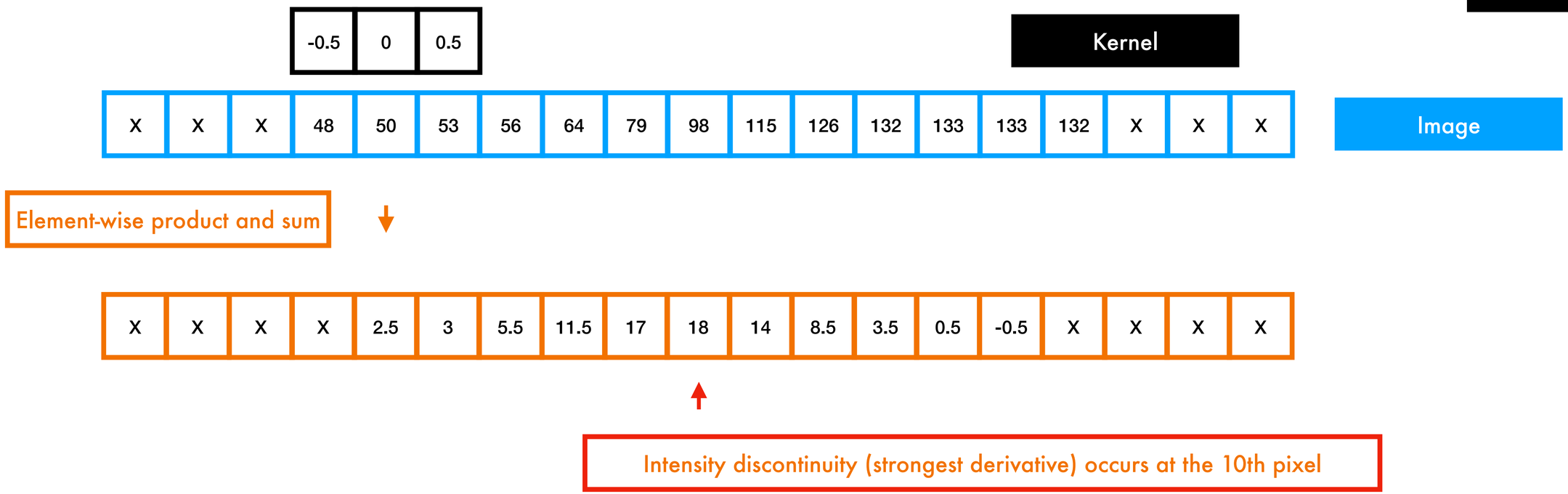
Differentiation and 1D edge detection

Question 6

The smoothed row of pixels in question 4 is shown below. Find the first order derivatives and localise the intensity discontinuity.

X	X	X	48	50	53	56	64	79	98	115	126	132	133	133	132	X	X	X

Solution



Decomposition of 2D convolution

Question 7

Smoothing a 2D image involves a 2D convolution with a 2D Gaussian: $G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$

Show that convolution with this 2D Gaussian can be performed via two 1D convolutions: $G_\sigma(x,y) \circledast I(x,y) = g_\sigma(x) \circledast [g_\sigma(y) \circledast I(x,y)]$. What is the advantage of performing two 1D convolutions instead of a 2D convolution?

Solution

Goal: show that convolution with this 2D Gaussian can be performed via two 1D convolutions: $G_\sigma(x,y) \circledast I(x,y) = g_\sigma(x) \circledast [g_\sigma(y) \circledast I(x,y)]$

Key idea: $G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{x^2}{2\sigma^2}\right)\cdot\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{y^2}{2\sigma^2}\right)$

Plug this in to the LHS of our **goal** equation: $G_\sigma(x,y) \circledast I(x,y) = \frac{1}{2\pi\sigma^2}\int_u\int_v I(x-u,y-v)\cdot\exp\left(-\frac{u^2+v^2}{2\sigma^2}\right)du\,dv$

$= \frac{1}{\sqrt{2\pi\sigma^2}}\int_u \exp\left(-\frac{u^2}{2\sigma^2}\right)\frac{1}{\sqrt{2\pi\sigma^2}}\int_v I(x-u,y-v)\cdot\exp\left(-\frac{v^2}{2\sigma^2}\right)dv\,du$ (using the key idea)

$= \frac{1}{\sqrt{2\pi\sigma^2}}\int_u \exp\left(-\frac{u^2}{2\sigma^2}\right)[g_\sigma(y) \circledast I(x-u,y)]du \qquad = g_\sigma(x) \circledast [g_\sigma(y) \circledast I(x,y)]$

This saves computation: a 1D convolution with kernel size N requires N multiplies and adds, whereas a 2D kernel of size N requires $N \times N = N^2$.

The saving is a factor of $\frac{N^2}{2N} = \frac{N}{2}$

Note: "separable" convolutions are very popular in deep learning architectures to save compute

Isotropic and directional edge finders

Question 8

(i) The Marr–Hildreth operator convolves the image with a discrete version of the Laplacian of a Gaussian and then localises edges at the resulting zero-crossings. Show that the Laplacian of a Gaussian is an isotropic (ie. rotationally symmetric) operator. Hence explain why the operator produces zero-crossings along an ideal step edge.

Solution

To show that the Marr-Hildreth edge detector is isotropic (uniform in all directions), we can use a change of coordinates.

The edge detector is the Laplacian of Gaussian: $\nabla^2 G_\sigma = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)\left[\frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)\right]$

First derivative

$$\frac{\partial}{\partial x}\left[\frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)\right] = \frac{1}{2\pi\sigma^2}\left(\frac{-x}{\sigma^2}\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

Second derivative

$$\frac{\partial^2}{\partial x^2}\left[\frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)\right] = \frac{1}{2\pi\sigma^2}\left(\frac{-1}{\sigma^2}\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) + \frac{1}{2\pi\sigma^2}\left(\frac{x^2}{\sigma^4}\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

$$\nabla^2 G_\sigma = \frac{1}{2\pi\sigma^2}\left(\frac{x^2}{\sigma^4} + \frac{y^2}{\sigma^4} - \frac{2}{\sigma^2}\right)\cdot\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$ (by plugging in the second derivatives for both x and y)

$$= \frac{1}{2\pi\sigma^2}\left(\frac{r^2}{\sigma^4} - \frac{2}{\sigma^2}\right)\cdot\exp\left(-\frac{r^2}{2\sigma^2}\right)$$ by converting to *polar coordinates* with $x = r\cos\theta, y = r\sin\theta$ (with $r^2 = x^2 + y^2$)

We can see that $\nabla^2 G_\sigma$ has no dependence on direction θ, so it must be isotropic!

Intuition for why Marr-Hildreth produces zero-crossings at an ideal step edge

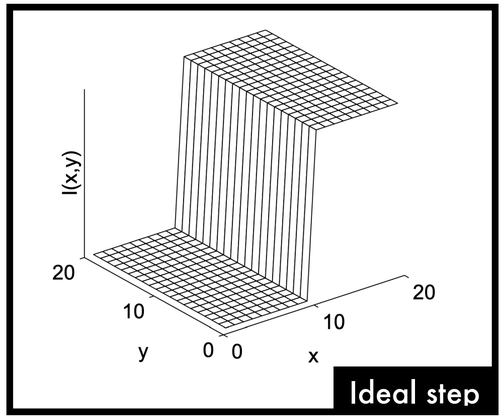
Question 8

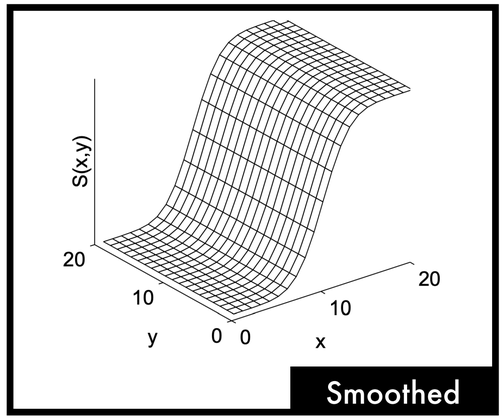
(ii) Hence explain why the operator produces zero-crossings along an ideal step edge

Solution

Suppose we have an ideal step edge that is axis-aligned in an image $I(x, y)$ (we know Marr-Hildreth is isotropic, so it doesn't matter which angle the edge has) as shown in the left figure.

Applying $\nabla^2 G_\sigma$ to the image is equivalent to first smoothing with a Gaussian with variance σ^2, to produce a smoother surface $S(x, y)$, as shown in the right figure.





$S(x, y)$ is constant in the y direction, $\frac{\partial^2 S}{\partial y^2}$ is zero everywhere (because we aligned the step to the y-axis).

Since $\frac{\partial S}{\partial x}$ reaches a maximum at $x = 10$, we also have that $\frac{\partial^2 S}{\partial x^2}$ is zero at the edge, so $\nabla^2 G_\sigma$ produces a zero at the edge.

Moving up/downhill flips the curvature $\frac{\partial^2 S}{\partial x^2}$ from to negative/positive, producing a *zero-crossing*.

Isotropic and directional edge finders

Question 8

(iii) The Canny operator is a directional edge finder. It first localises the orientation of the edge by computing $\mathbf{n} = \dfrac{\nabla(G_\sigma(x,y) \circledast I(x,y))}{|\nabla(G_\sigma(x,y) \circledast I(x,y)|}$ and then searches for a local maximum of $|\nabla(G_\sigma \circledast I)|$ in the direction of $\mathbf{n}$. Show that this is equivalent to finding zero-crossings in the directional second derivative of $(G_\sigma \circledast I)$ in the direction of $\mathbf{n}$, i.e. finding zero-crossings in $\dfrac{\partial^2(G_\sigma \circledast I)}{\partial s^2}$ where s is a length parameter in the direction $\mathbf{n}$.

Solution

Since the Canny edge detector searches for a maximum of $|\nabla(G_\sigma \circledast I)|$ in the direction $\mathbf{n}$, then if s is a length parameter in the direction $\mathbf{n}$, the detector seeks locations where:

$$\frac{\partial}{\partial s}|\nabla(G_\sigma \circledast I)| = 0$$

Remember that the maximum value of a directional derivative $\partial\phi/\partial s$ is given by $|\nabla\phi|$.

As a result, $\dfrac{\partial}{\partial s}(G_\sigma \circledast I) = |\nabla(G_\sigma \circledast I)|$ since s is in the direction of maximum intensity gradient.

We can substitute this into the earlier expression to give $\dfrac{\partial}{\partial s}\left(\dfrac{\partial}{\partial s}(G_\sigma \circledast I)\right) = 0$

Which shows that the Canny edge detector finds zero-crossings in the directional second derivative of $(G_\sigma \circledast I)$

Canny vs Marr-Hildreth (pros and cons)

Question 8

(iv) What are the advantages and disadvantages of isotropic and directional operators?

Marr-Hildreth

Pros

Computational efficiency: only requires a single convolution and detection of zero-crossings

Cons

Robustness: derivatives amplify noise (and Marr-Hildreth uses second derivatives)

Canny

Pros

Robustness: only uses first derivatives.

Useful extra output: directional edges

Cons

Computational efficiency: requires an extra, costly step to find local maxima along the gradient direction.

Auto-correlation and corner detection

Question 9

(a) Show that the weighted sum of squared differences (SSD) between a patch (window W) of pixels in image $S(x, y) = S(\mathbf{x})$ and another patch of pixels taken by shifting the window by a small amount in the direction $\mathbf{n}$ can be expressed approximately by:

$$C(\mathbf{n}) = \sum_{\mathbf{x} \in W} w(\mathbf{x})(S(\mathbf{x} + \mathbf{n}) - S(\mathbf{x}))^2 \approx \sum_{\mathbf{x} \in W} w(x) S_n^2 \quad \text{where } S_n = \nabla S(\mathbf{x}) \cdot \mathbf{n}$$

Solution

If in doubt, write down the Taylor Series expansion: $S(\mathbf{x} + \mathbf{n}) = S(\mathbf{x}) + \nabla S(\mathbf{x}) \cdot \mathbf{n} + \text{higher order terms}$

So, $S(\mathbf{x} + \mathbf{n}) - S(\mathbf{x}) \approx \nabla S(\mathbf{x}) \cdot \mathbf{n}$

$$\implies C(\mathbf{n}) = \sum_{\mathbf{x} \in W} w(\mathbf{x})(S(\mathbf{x} + \mathbf{n}) - S(\mathbf{x}))^2 \approx \sum_{\mathbf{x} \in W} w(x)(\nabla S(\mathbf{x}) \cdot \mathbf{n})^2$$

$$= \sum_{\mathbf{x} \in W} w(x) S_n^2 \text{ (using the definition of } S_n)$$

Auto-correlation and corner detection

Question 9

(b) Hence show that the weighted SSD can be represented by $C = \mathbf{n}^T A \mathbf{n}$ where A is a matrix of smoothed intensity gradients (sometimes called a second-moment or autocorrelation matrix) defined as follows:

$$A = \begin{bmatrix} \langle S_x^2 \rangle & \langle S_x S_y \rangle \\ \langle S_x S_y \rangle & \langle S_y^2 \rangle \end{bmatrix}$$

where $S_x \triangleq \partial S / \partial x$, $S_y \triangleq \partial S / \partial y$, and $\langle \rangle$ denotes a 2-dimensional weighting (smoothing) function.

Solution

$$C(\mathbf{n}) = \sum_{\mathbf{x} \in W} w(\mathbf{x}) (\nabla S \cdot \mathbf{n})^2 = \sum_{\mathbf{x} \in W} w(\mathbf{x}) (\mathbf{n}^T \nabla S^T)(\nabla S \cdot \mathbf{n}) = \sum_{\mathbf{x} \in W} w(\mathbf{x}) \left(\mathbf{n}^T \begin{bmatrix} S_x^2 & S_x S_y \\ S_x S_y & S_y^2 \end{bmatrix} \mathbf{n} \right)$$

which we can write as $C(\mathbf{n}) = \mathbf{n}^T A \mathbf{n}$, where $A = w(\mathbf{x}) \circledast \begin{bmatrix} S_x^2 & S_x S_y \\ S_x S_y & S_y^2 \end{bmatrix} = \begin{bmatrix} \langle S_x^2 \rangle & \langle S_x S_y \rangle \\ \langle S_x S_y \rangle & \langle S_y^2 \rangle \end{bmatrix}$

Auto-correlation and corner detection

Question 9

(c) How are the directional derivatives computed from the raw intensities, $I(x, y)$? How are the 2D weighted (smoothed) values obtained? Comment on the different smoothing/weighting parameters for derivatives and the autocorrelation matrix.

Solution

We obtain directional derivatives after smoothing $I(x, y)$ by a Gaussian to obtain $S(x, y)$, by using the 1D kernels first order derivative kernel (from Q6) in the x and y direction to produce S_x and S_y.

For efficiency, we will ideally use separated Gaussians along each axis. We use a smaller kernel for the first smoothing (prior to differentiation) to suppress high frequency noise, use a larger kernel for smoothing the neighbourhood derivatives to avoid sharp discontinuities stemming from using a square region.

Question 9

(d) Show how A can be analysed to detect corner features and give details of the Harris-Stephens corner detection algorithm.

Solution

We can examine $\det(A)$ (i.e. $\lambda_{min}\lambda_{max}$) and $\text{trace}(A)$ (i.e. $\lambda_{min} + \lambda_{max}$) to avoid computing eigenvalues explicitly. We expect corners to occur when both eigenvalues are large, which corresponds to matrices for which $\frac{\det(A)}{1 + \kappa \text{trace}(A)}$ exceeds some threshold.

Feature detection and scale space

Question 10

Consider an algorithm to detect interest points (features of interest) in a 2D image for use in matching.

(a) Show how different resolutions of the image can be represented efficiently in an *image pyramid*. Your answer should include details of the implementation of smoothing within an octave and subsampling of the image between octaves.

Solution

We can build an image pyramid by *convolving* the image with Gaussians of varying sizes:

$L(x, y, t) = G(x, y, t) \circledast I(x, y)$ where $G(x, y, t) = \frac{1}{2\pi t} \exp\left(-\frac{x^2 + y^2}{2t} \right)$ and $t = \sigma^2$

For efficiency, we sample a discrete set of scales, rather than densely constructing all possible values of $L(x, y, t)$ according to $\sigma_{i+1} = 2^{\frac{i}{s}} \sigma_0$

This set of sampled scales can be divided into *octaves* (each containing s blurred images) which correspond to scale doublings. The last blurred image is subsampled by a factor of two in each dimension to construct the start of the next octave.

Within an octave, we use incremental blurring, using the reproducing property of Gaussians to note that we can write $G(\sigma_{i+1}) = G(\sigma_i) \circledast G(\sigma_{k_i})$, and solve explicitly for σ_{k_i} to keep the cost low by only computing small Gaussian blurs, and re-using the kernels:

$\sigma_{k_i} = \sqrt{\sigma_{i+1}^2 - \sigma_i^2}$ (reproducing property) + $\sigma_{i+1} = 2^{\frac{1}{s}} \sigma_i$ (by definition) $\Longrightarrow$ $\sigma_{k_i} = \sqrt{2^{\frac{2}{s}} \sigma_i^2 - \sigma_i^2} = \sigma_i \sqrt{2^{\frac{2}{s}} - 1}$

Feature detection and scale space

Question 10

(b) How can band-pass filtering at different scales be implemented efficiently using the image pyramid? Show how image features such as blob-like shapes can be localized in both position and scale using band-pass filtering.

Solution

The Laplacian of Gaussian is a band-pass filter, which we approximate with the Difference of Gaussians using the approximatIon: $G(x, y, k\sigma) - G(x, y, \sigma) \approx (k-1)\sigma^2 \nabla^2 G(x, y, \sigma)$

We can then simply subtract layers of the pyramid to get DoG responses.

We can localise blob-like shapes by searching local neighbourhoods for maxima/minima in the DoG responses, selecting the characteristic scale and location.

Question 10

(c) Explain how interest points in different images can be matched. Give details of 3 suitable descriptors.

Solution

The matching process is: (i) find the position and scale of the interest points. (ii) Compute gradient histograms and find dominant orientations, (iii) Then compute appropriate descriptors for each interest point, (iv) find matches via a suitable metric (e.g. Euclidean distance between descriptors). Suitable descriptors could include zero-normalised patches, HoG, SIFT descriptors.