Lecture 5 Group Operators

COMP3204 Computer Vision

How do we combine points to make a new point in a new image?

Book
pp
84- 114

Department of
Electronics and
Computer Science

UNIVERSITY OF
Southampton
School of Electronics
and Computer Science

Content

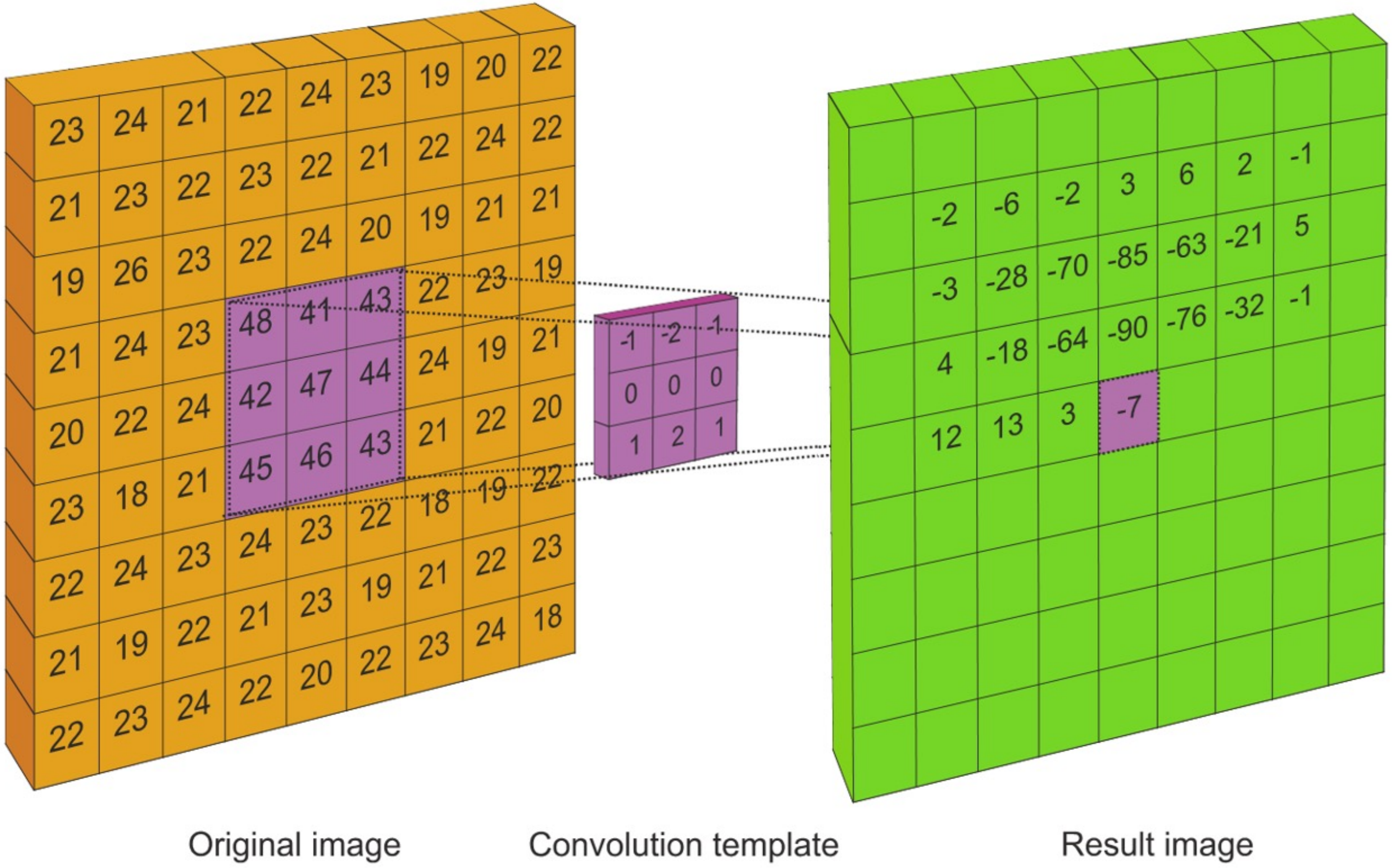
1. How can we collect points as a group?
2. How can we apply processes to that group?

Template convolution

Calculate a new image from the original

Template is convolved in a raster fashion

Template is inverted for convolution



Template convolution

Image

100	100	200	200	200
100	100	200	200	200
100	100	200	200	200
200	200	400	400	400
300	300	400	400	400

G_y

0	0	0	0	0
0	400	400	0	0
0	500	500	0	0
0	600	600	0	0
0	0	0	0	0

Result

0	0	0	0	0
0	707	400	0	0
0	640	860	800	0
0	1000	1000	800	0
0	0	0	0	0

G_x

0	0	0	0	0
0	0	0	0	0
0	-500	-700	-800	0
0	-800	-800	-800	0
0	0	0	0	0

3×3 template and weighting coefficients

w_0	w_1	w_2
w_3	w_4	w_5
w_6	w_7	w_8

$$\mathbf{N}_{x,y} = \sum_{i \in \text{template}} \sum_{j \in \text{template}} w_{i,j} \times \mathbf{O}_{x(i),y(j)}$$

where $w_{i,j}$ are the weights and $x(i), y(j)$ denote the position of the point that matches the weighting coefficient position

Result calculated for centre point

Border?

Three options

1. Set border to black
2. Assume wrap-around
3. Make template smaller near edges

Normally we assume object of interest is near centre so set border to black

3×3 averaging operator

$$\mathbf{N}_{x,y} = \frac{1}{9} \sum_{i \in 3} \sum_{j \in 3} \mathbf{O}_{x(i),y(j)}$$

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

5×5, w = 1/25 and 7×7, w = 1/49 etc.

Illustrating the effect of window size

3×3

5×5

7×7

Larger operators remove more noise, but lose more detail

Nasty bit ....

Template is actually flipped around both axes

$$\mathbf{I} * \mathbf{T} = \sum_{(x,y)\in W} \mathbf{I}_{x,y} \mathbf{T}_{i-x,j-y}$$

This does not matter for symmetric templates

(i.e. the deep learning ones!)

Template convolution via the Fourier transform

Convolution theorem allows for fast computation via FFT for template size ≥ 7×7

$$\mathbf{P} * \mathbf{T} = \mathfrak{I}^{-1}\left(\mathfrak{I}(\mathbf{P}) .\times \mathfrak{I}(\mathbf{T})\right)$$

Template convolution $*$

Fourier transform of the picture, $\mathfrak{I}(\mathbf{P})$

Fourier transform of the template, $\mathfrak{I}(\mathbf{T})$

The inversion is implicit in Fourier

The theory is at end, for information only

Point by point multiplication $(\cdot\times)$ for sampled signals

This is fast!!

Beware of clowns … Oxford

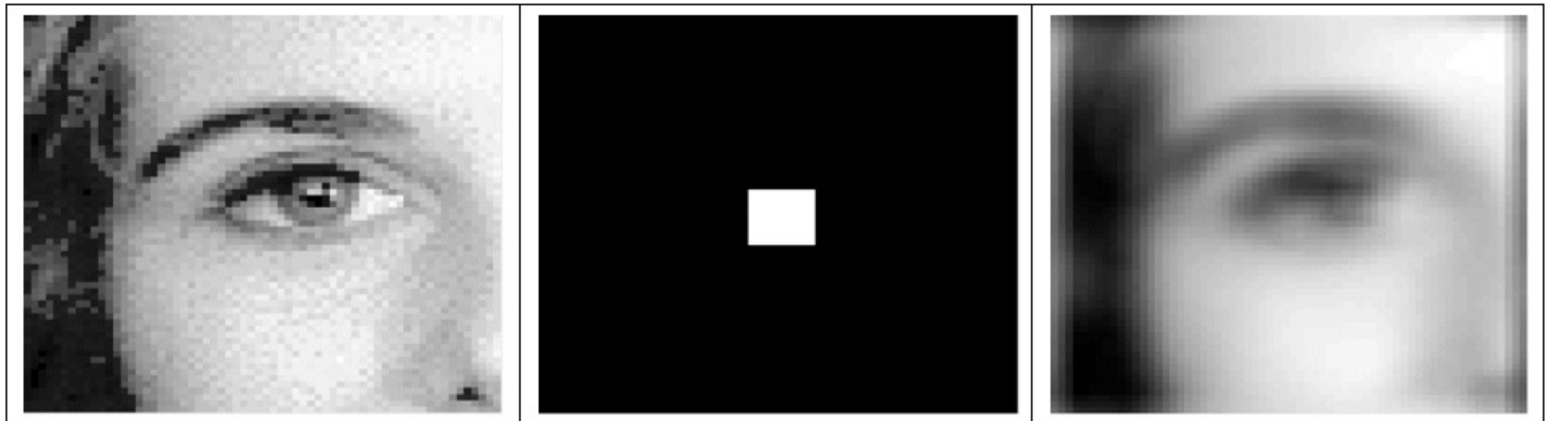
Imperial

$$f(x,y) * h(x,y) \Leftrightarrow F(u,v)H(u,v)$$

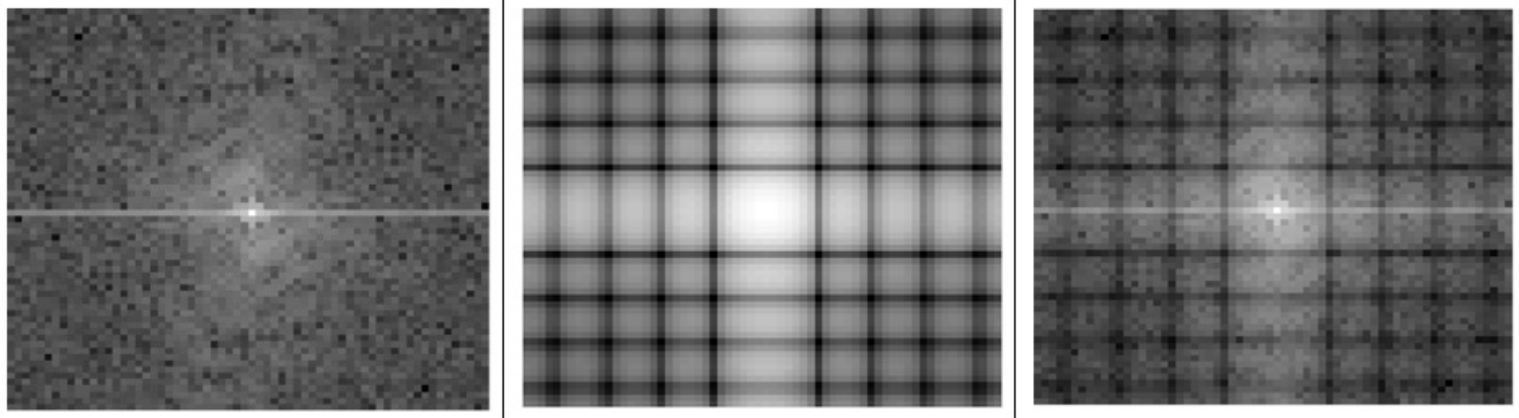
$$w(t) = u(t) * v(t) \Leftrightarrow W(f) = U(f)V(f)$$

it's point by point!!

Template Convolution via the Fourier Transform



(a) image of eye | (b) padded averaging template | (c) resulting averaged image



(d) image transform | (e) template transform | (f) multiplied transforms

2D Gaussian function

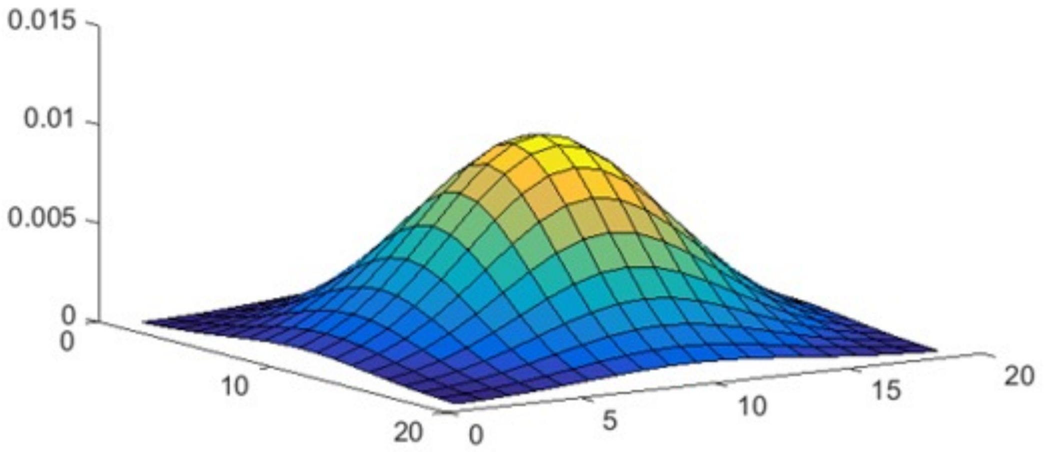
$$g(x,y,\sigma)=\frac{1}{2\pi\sigma^2}e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

Used to calculate template values

Note compromise between variance σ^2 and window size

Common choices 5×5, 1.0; 7×7, 1.2; 9×9, 1.4

2D Gaussian function and template



0.002	0.013	0.022	0.013	0. 002
0.013	0.060	0. 098	0. 060	0.013
0.022	0. 098	0.162	0. 098	0.022
0.013	0. 060	0.098	0. 060	0.013
0. 002	0.013	0.022	0.013	0. 002

Template for the 5×5 Gaussian Averaging Operator ($\sigma = 1.0$)

Applying Gaussian averaging

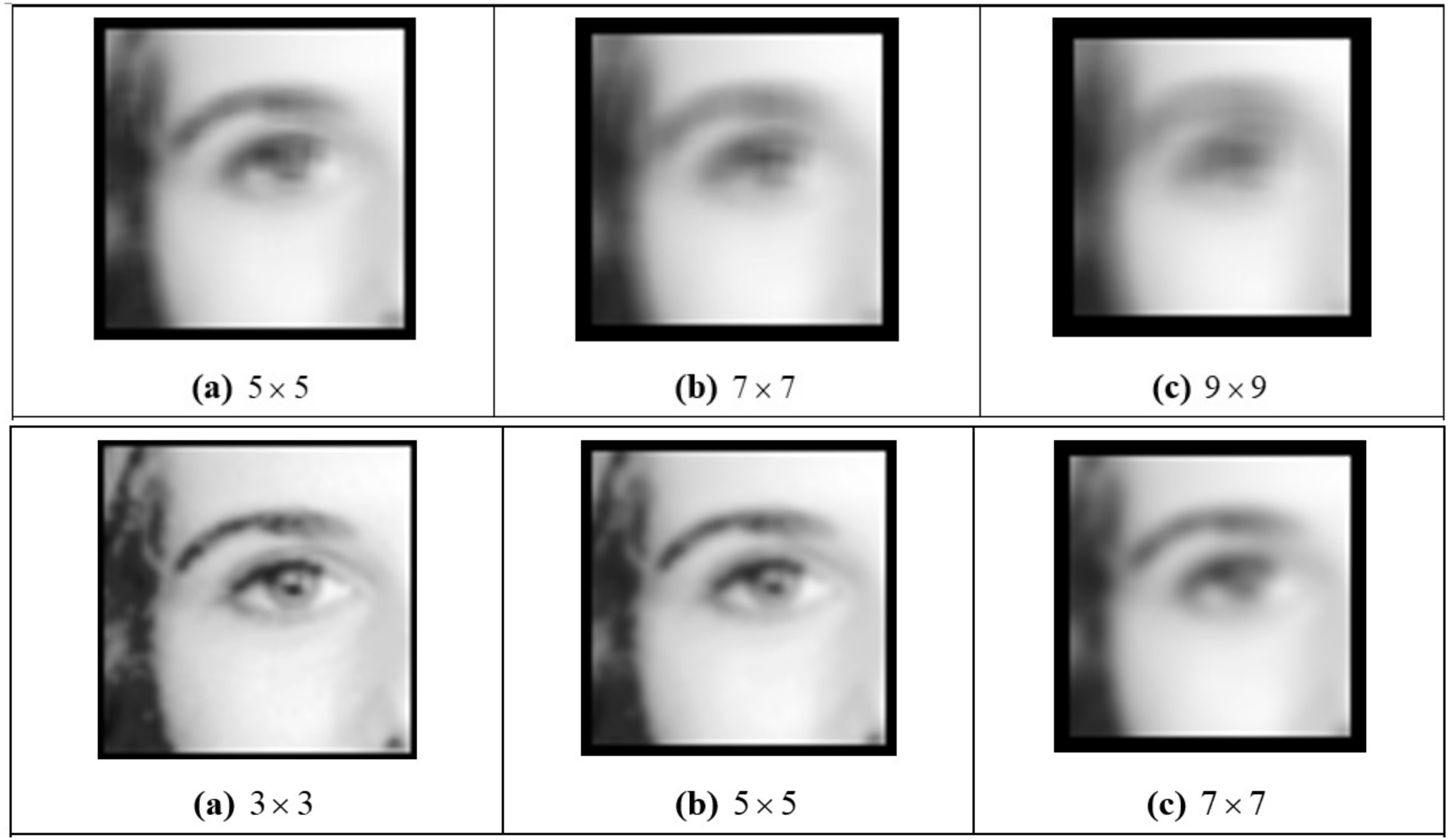
(a) 3×3 (b) 5×5 (c) 7×7

Comparison

Direct averaging

Which one is better?

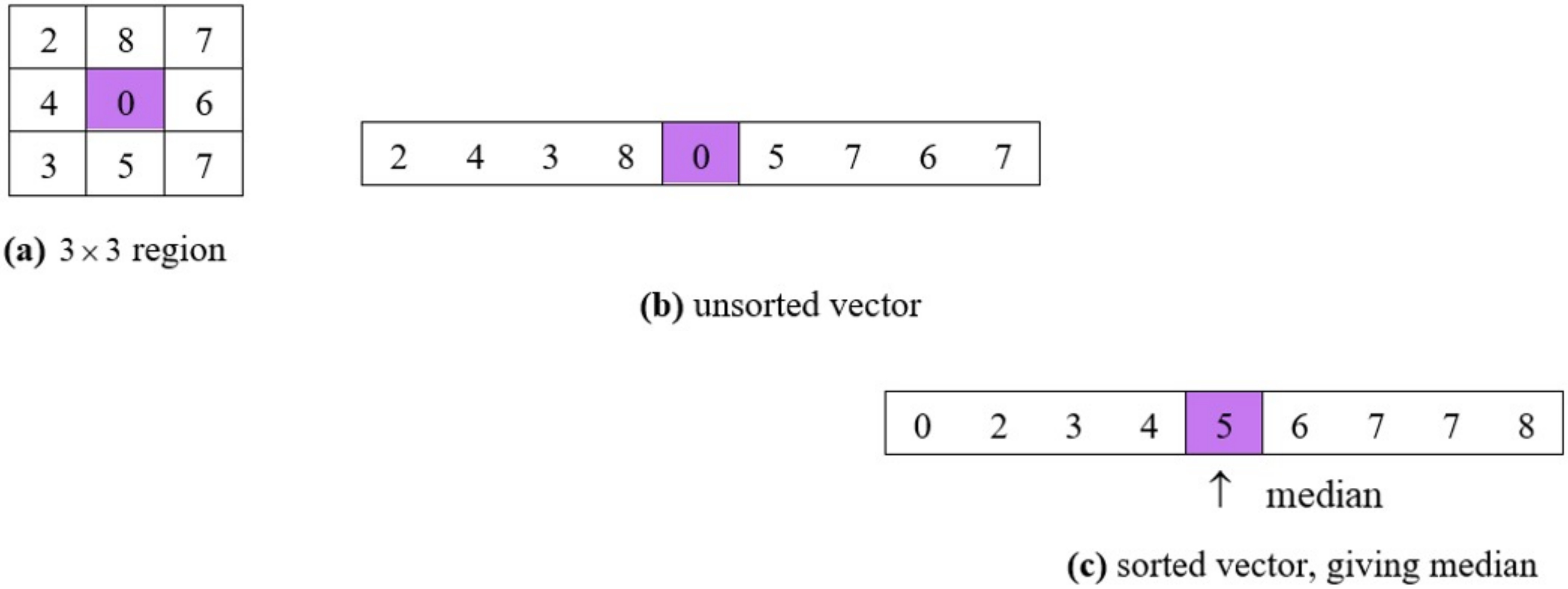
Gaussian averaging



(a) 5×5 (b) 7×7 (c) 9×9

(a) 3×3 (b) 5×5 (c) 7×7

Finding the median from a 3×3 template



(a) 3 × 3 region

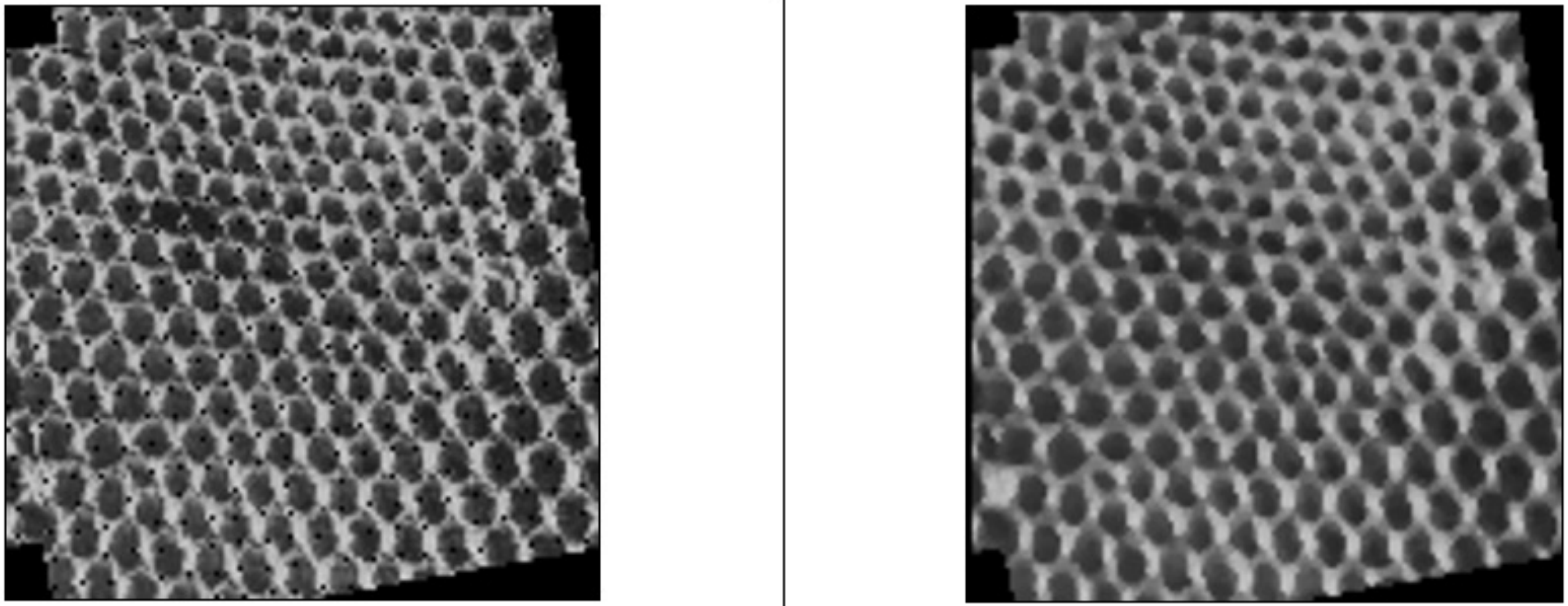
(b) unsorted vector

(c) sorted vector, giving median

The median is the centre element of a rank-ordered set of template points

Finding the median from a 3×3 template

Preserves edges; Removes salt and pepper noise

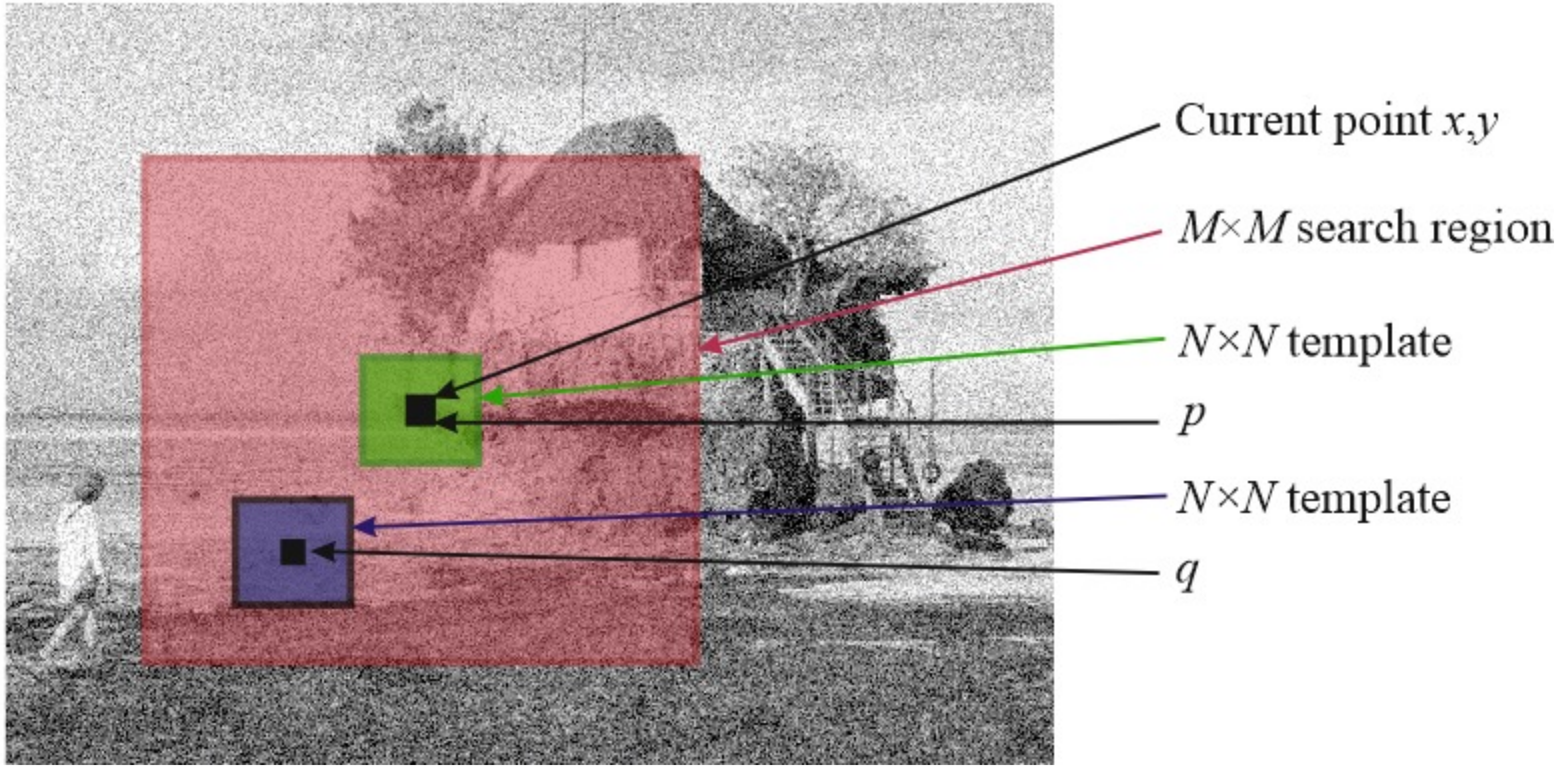


(a) rotated fence

(b) median filtered

Non-local means

Averaging which preserves regions



Applying non local means

(a) original image

(b) Gaussian averaging

(c) nonlocal means

(a) Original

(b) (a) with added Gaussian noise

(c) Averaged

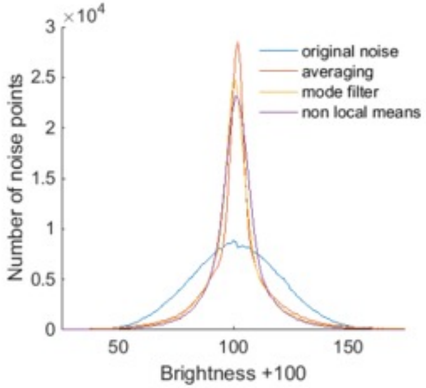
(d) Gaussian smoothed

(e) Median

(f) Truncated Median

(g) Anisotropic diffusion

(h) Non-local-means



(i) Effect of filtering on noise

Comparison of Filtering Operators

Bilateral filtering

Averaging which preserves regions

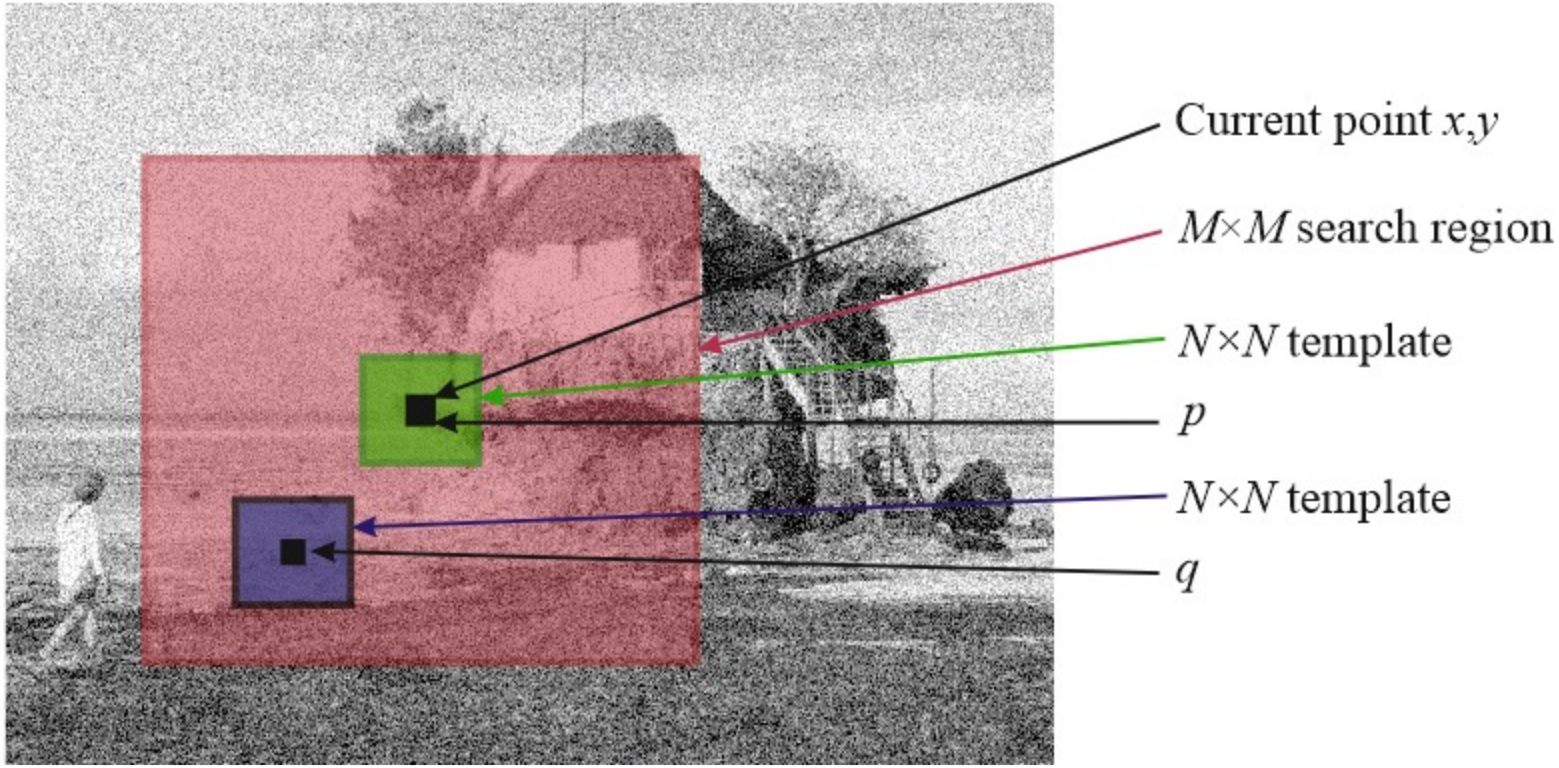
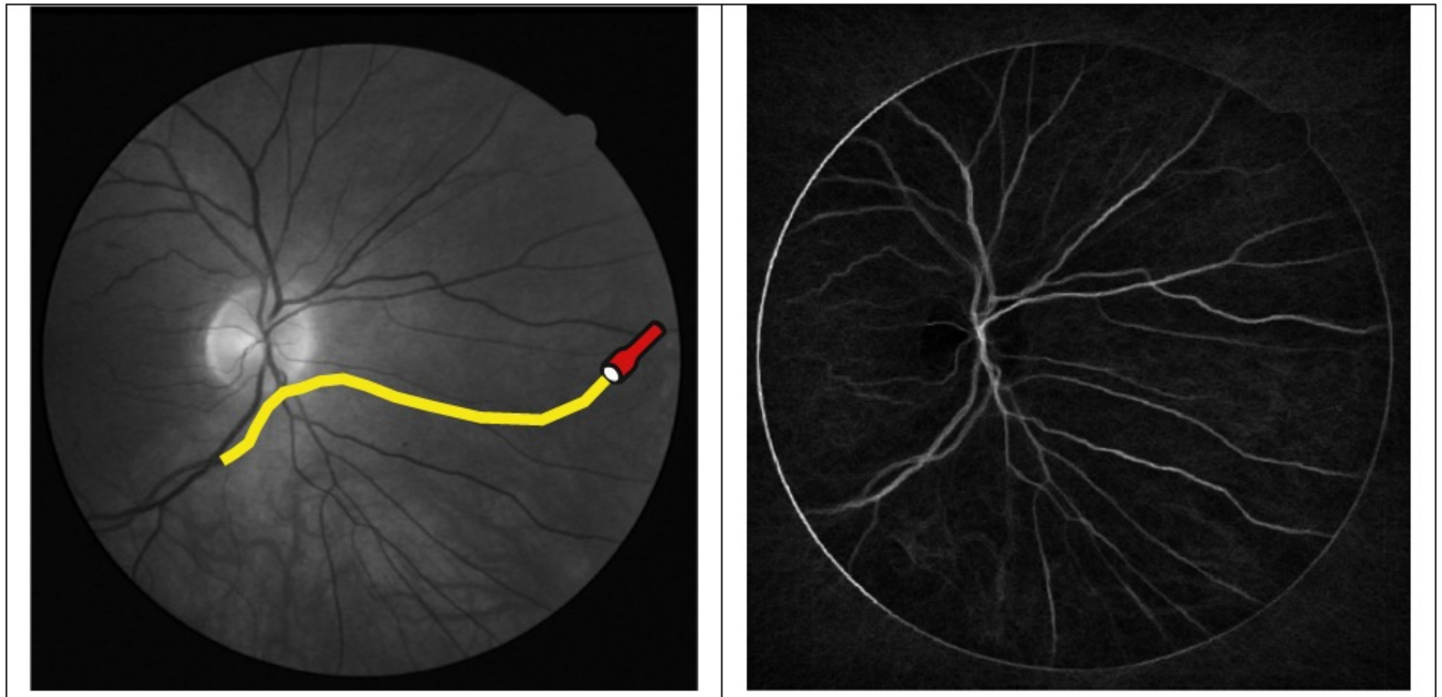


Image Ray Transform

Use analogy to light to find shapes, removing remainder



(a) method of operation (b) result of transform

Cummings, Nixon and Prugel-Bennett, *PRL* 2012

Applying Image Ray Transform

(a) tubular structure

(b) circular structure

(c) car

Good results

Poor result

Main points so far

1 – collection of points is called a template

2 – application to an image is called template convolution

3 – can use Fourier to improve speed

4 – averaging reduces noise

How do we find features? That’s edge detection, coming next

Convolution theorem, for completeness only!

1-D convolution is defined as $\mathbf{p} * \mathbf{q} = \sum_{i=0}^{N-1} p_i \, q_{m-i}$

by the DFT, for component *u* $\quad \mathcal{F}(\mathbf{p} * \mathbf{q})_u = \frac{1}{N}\sum_{m=0}^{N-1}\left(\sum_{i=0}^{N-1} p_i \, q_{m-i}\right) e^{-j\frac{2\pi}{N}mu}$

by re-ordering $\quad = \frac{1}{N}\sum_{i=0}^{N-1} p_i \sum_{m=0}^{N-1} q_{m-i} \, e^{-j\frac{2\pi}{N}mu}$

by shift th^m $\mathcal{F}(\mathbf{q}[i-\Delta]) = \mathbf{Fq}[i] \times e^{-j\omega\Delta}$ $\quad = \frac{1}{N}\sum_{i=0}^{N-1} p_i \sum_{m=0}^{N-1} q_m \, e^{-j\frac{2\pi}{N}mu} e^{-j\frac{2\pi}{N}iu}$

by grouping like terms $\quad = \frac{1}{N}\sum_{i=0}^{N-1} p_i e^{-j\frac{2\pi}{N}iu} \sum_{m=0}^{N-1} q_i \, e^{-j\frac{2\pi}{N}mu}$

and (by serendipity?) $\quad = \left(\mathcal{F}(\mathbf{p}) \times \mathcal{F}(\mathbf{q})\right)_u$

By this, the implementation of discrete convolution using the DFT is achieved by multiplication. For two sampled signals each with N points we have

$$\mathcal{F}(\mathbf{p} * \mathbf{q}) = \mathcal{F}(\mathbf{p}) .\times \mathcal{F}(\mathbf{q})$$

So convolution is the point-wise multiplication of the two transforms, and the template does not need to be inverted. The inversion is implicit in the use of the Fourier transform.

$$F_{f*g}(w) = F_f(w) \times F_g(w)$$

F : Fourier transform

Proof:

$$F_{f*g}(w) = \int_{-\infty}^{+\infty} f*g(t)\, e^{-j2\pi w t}\, dt$$

$$= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(\tau)\, g(t-\tau)\, d\tau\, e^{-j2\pi w t}\, dt$$

$$= \int_{-\infty}^{+\infty} \left(\int_{-\infty}^{+\infty} g(t-\tau)\, e^{-j2\pi w (t-\tau)}\, dt \right) f(\tau) \cdot e^{-j2\pi w \tau}\, d\tau$$

$$= \int_{-\infty}^{+\infty} F_g(w) \cdot f(\tau) \cdot e^{-j2\pi w \tau}\, d\tau$$

$$= F_f(w) \times F_g(w) \quad \square$$