MLRF Lecture 02

J. Chazalon, LRDE/EPITA, 2019

Local feature detectors

Lecture 02 part 06

The need for local feature detectors

While **dense computation** of local feature descriptors is possible (grid of points), this is **rarely used in practice** (lots of computations, lots of useless features).

Will be **anchors** to describe a **feature of interest**.

- Edge / line
- Area around a corner / a stable point
- Blob (area of variable size)

A good feature of interest is **stable over the perturbations** our signal will face:

- Translation, rotation, zoom, perspective
- Illumination changes
- Noise, compression
- …

Some classical detectors

Edge (gradient detectors)

- Sobel
- Canny

Corner

- Harris & Stephens *and variants*
- FAST
- Laplacian of Gaussian, Difference of Gaussian, Determinant of Hessian

Blob

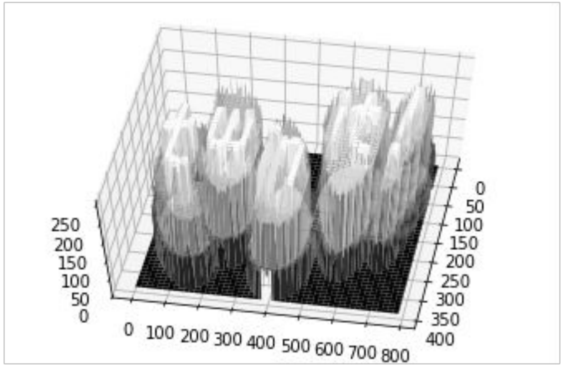
- MSER

Edge detectors

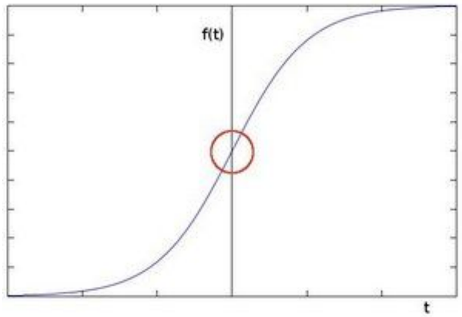
What’s an edge?

Image is a function

Edges are rapid changes in this function



The derivative of a function exhibits the edges



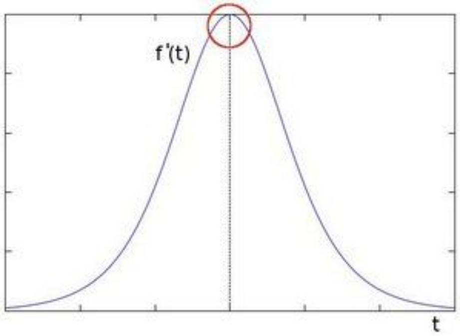


Image derivatives

Recall: $$f'(a) = \lim_{h \to 0} \frac{f(a+h) - f(a-h)}{2h}$$

We don't have an "actual" function, must estimate

Possibility: set h = 1

Apply filter

-1	0	+1

to the image (x gradient)

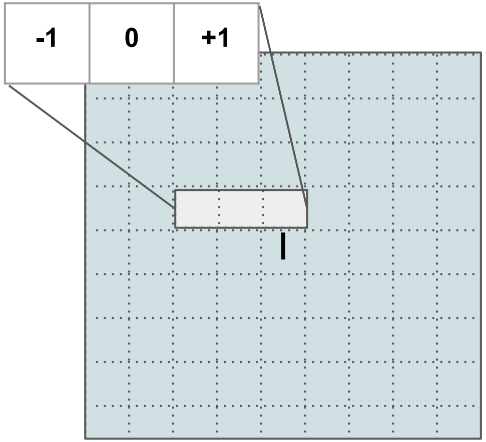
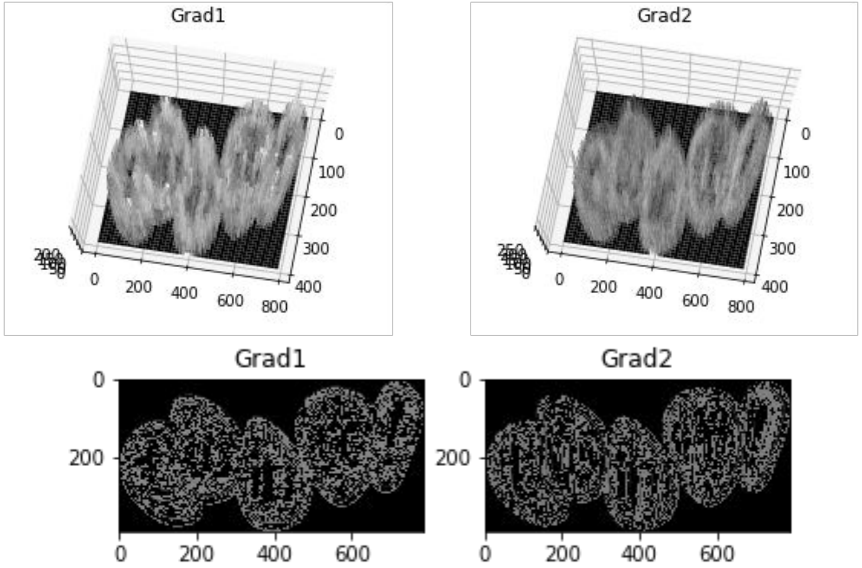


Image derivatives

We get terribly spiky results,
we need to interpolate / smooth.

⇒ Gaussian filter

We get a Sobel filter

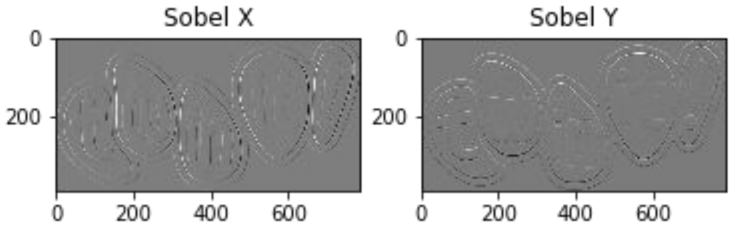


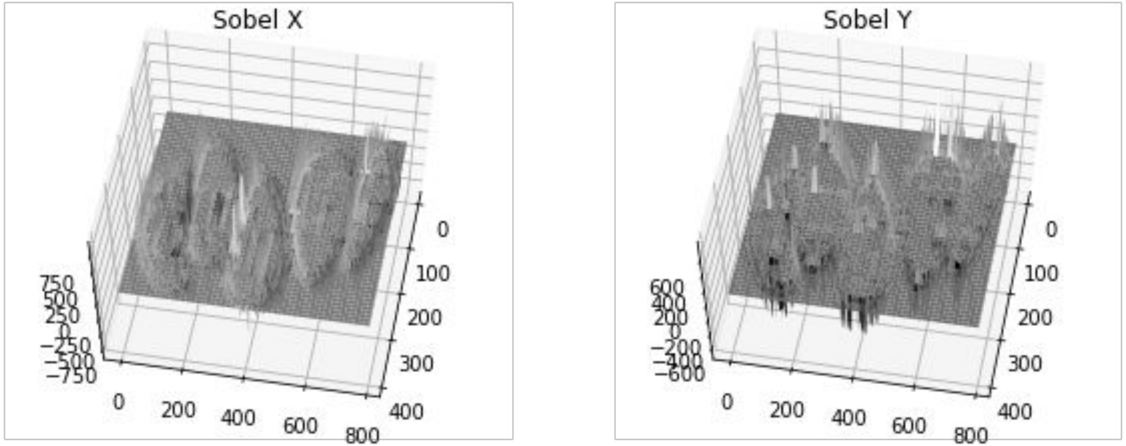
$$\frac{1}{2} \times \left(\begin{bmatrix} -1 & 0 & +1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \right) = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Horizontal Sobel

Vertical Sobel

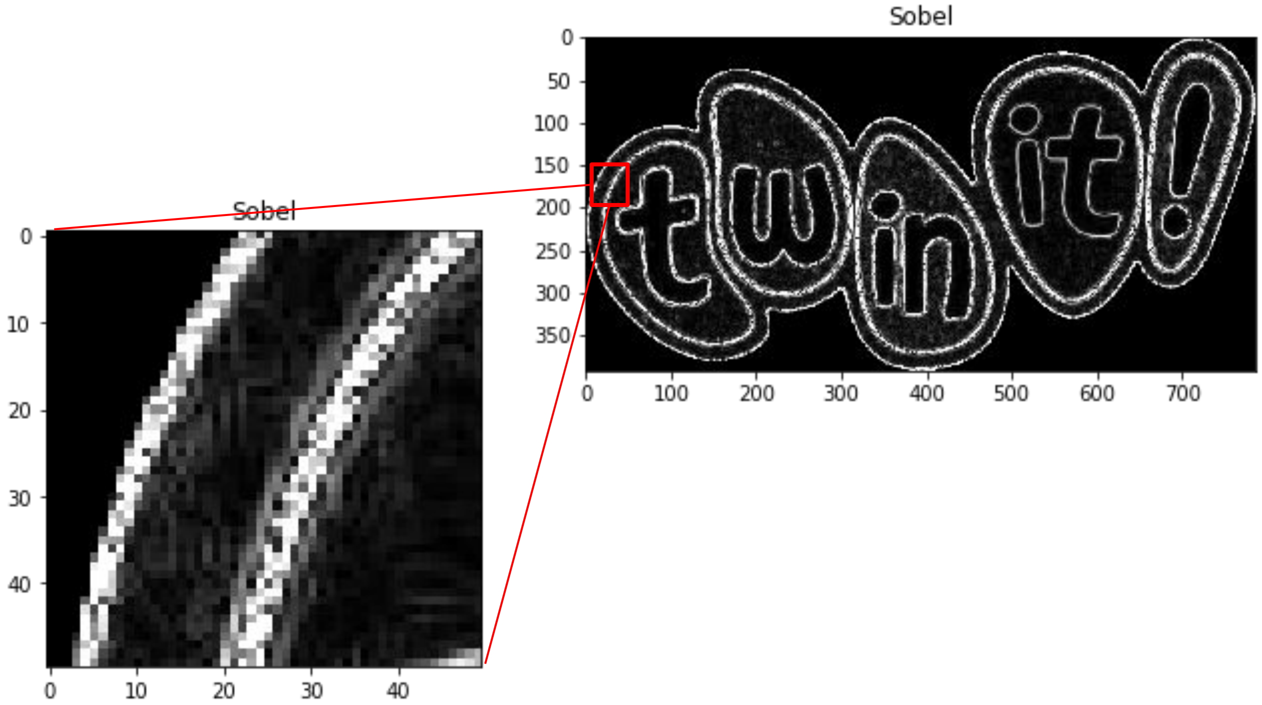
Sobel filter





Gradient magnitude with Sobel

sqrt(Sobel_x² + Sobel_y²)



Canny edge detection

Extract real lines!

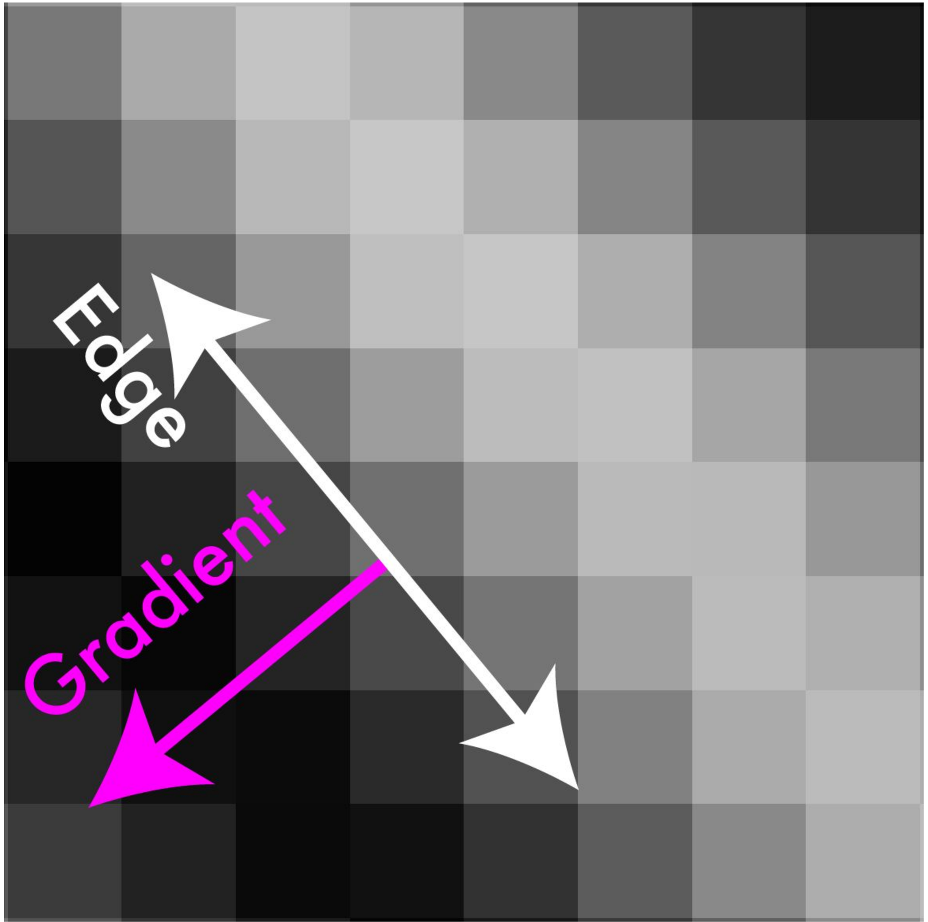
Algorithm:

Sobel operator

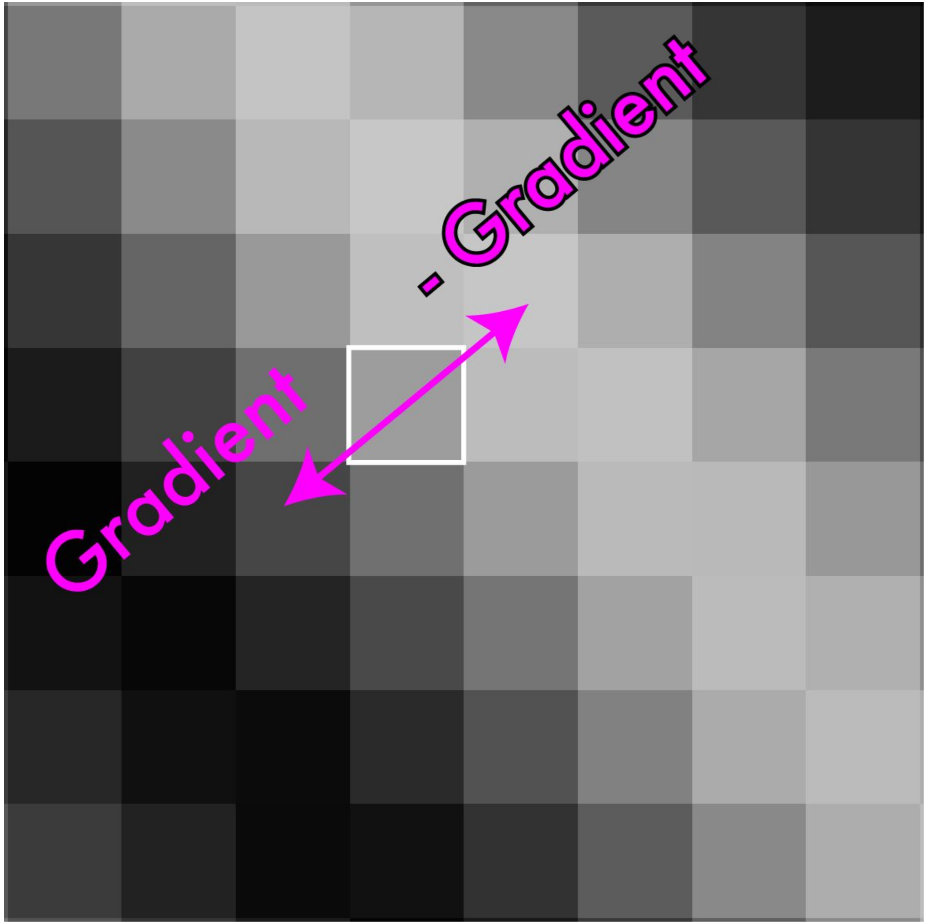
- Smooth image (only want “real” edges, not noise)
- Calculate gradient direction and magnitude
- Non-maximum suppression perpendicular to edge
- Threshold into strong, weak, no edge
- Connect together components

John Canny. A computational approach to edge detection. Pattern Analysis and Machine Intelligence, IEEE Transactions on, (6):679–698, 1986.

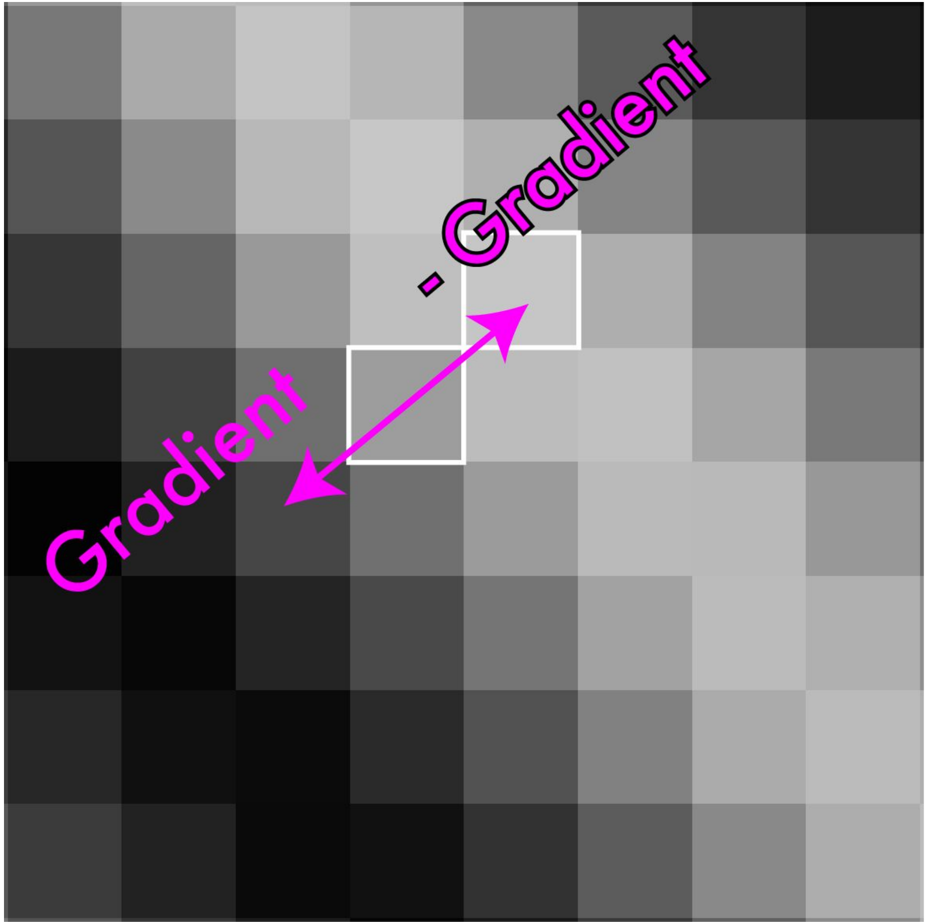
Canny: Non-maximum suppression



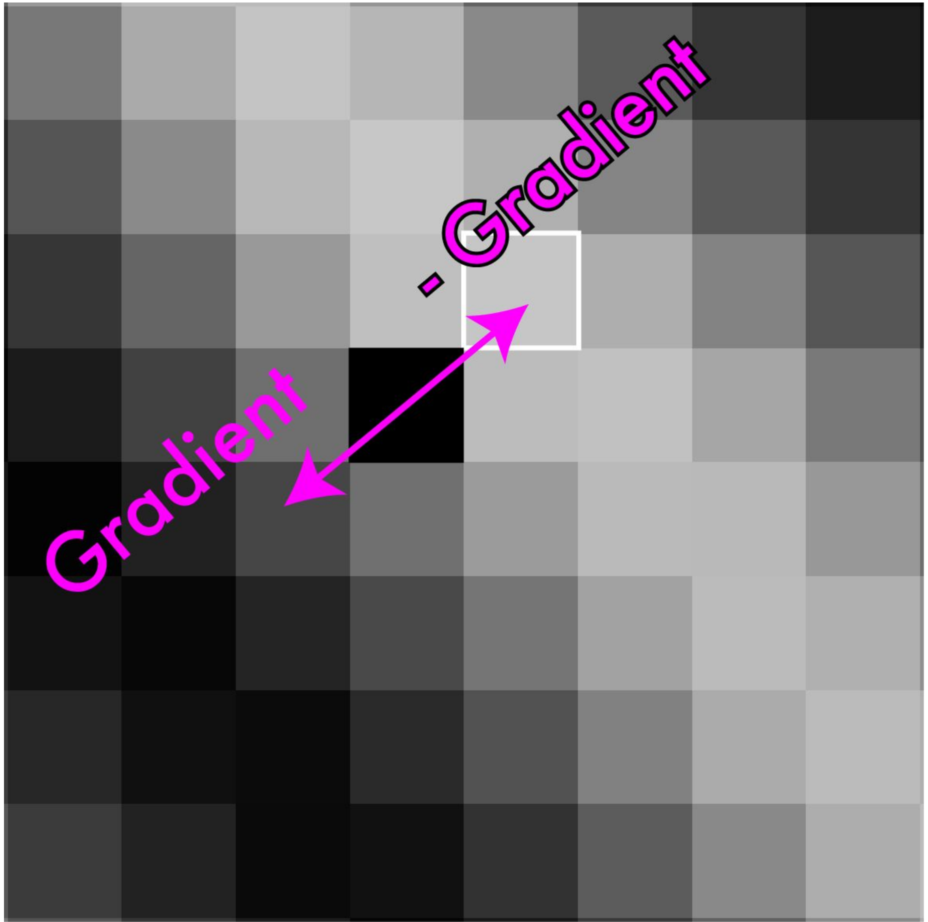
Canny: Non-maximum suppression



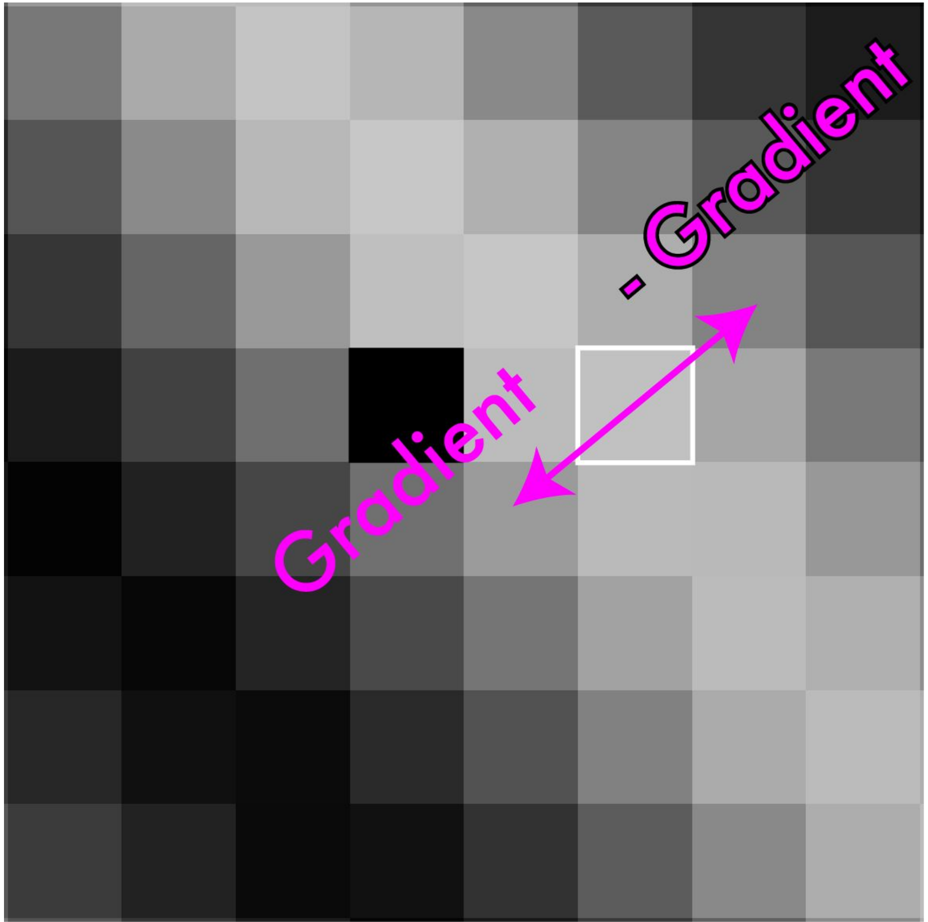
Canny: Non-maximum suppression



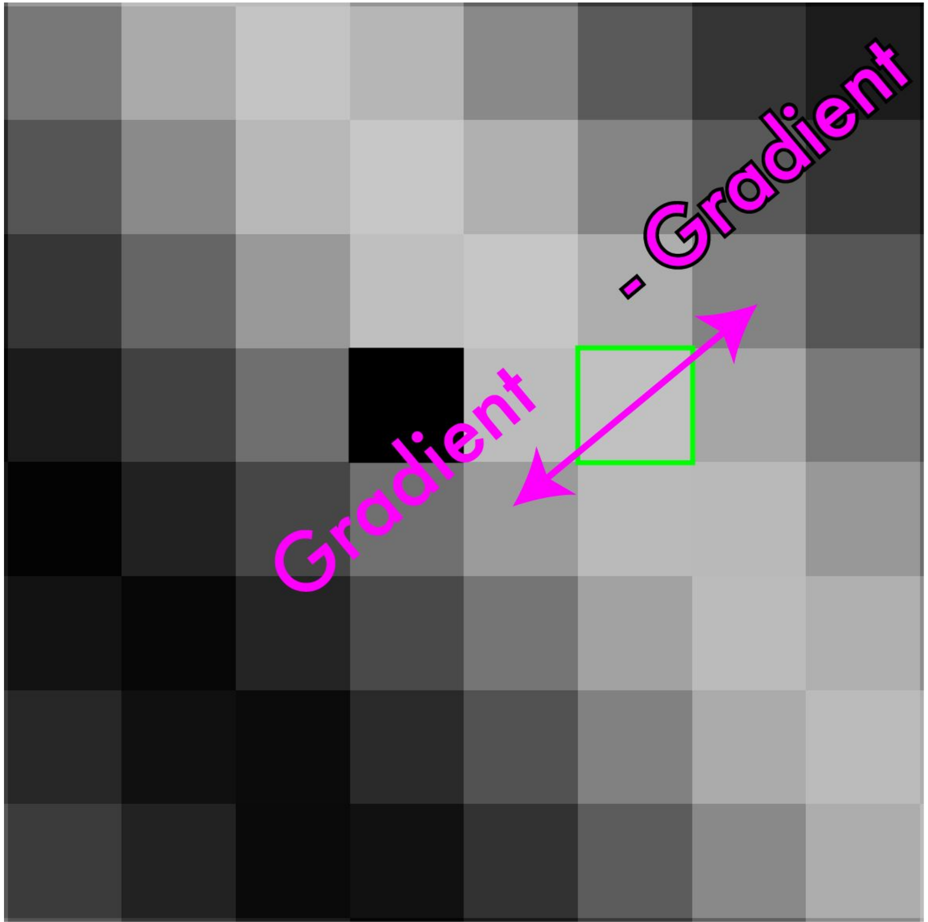
Canny: Non-maximum suppression



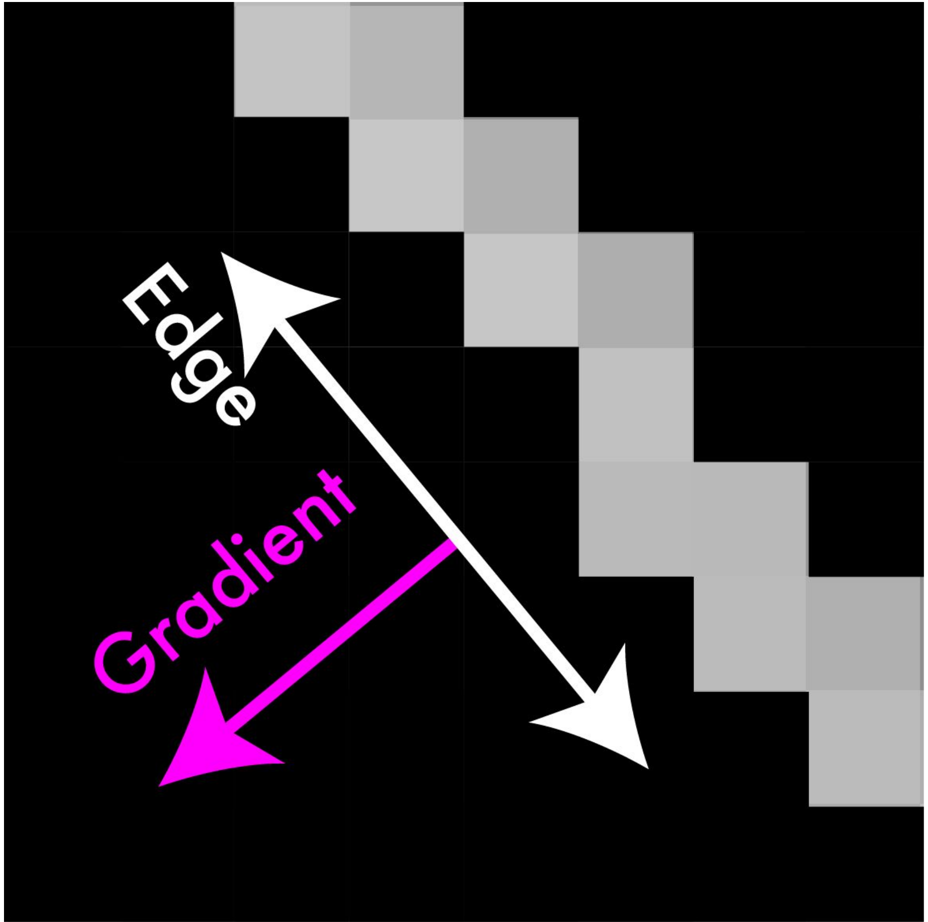
Canny: Non-maximum suppression



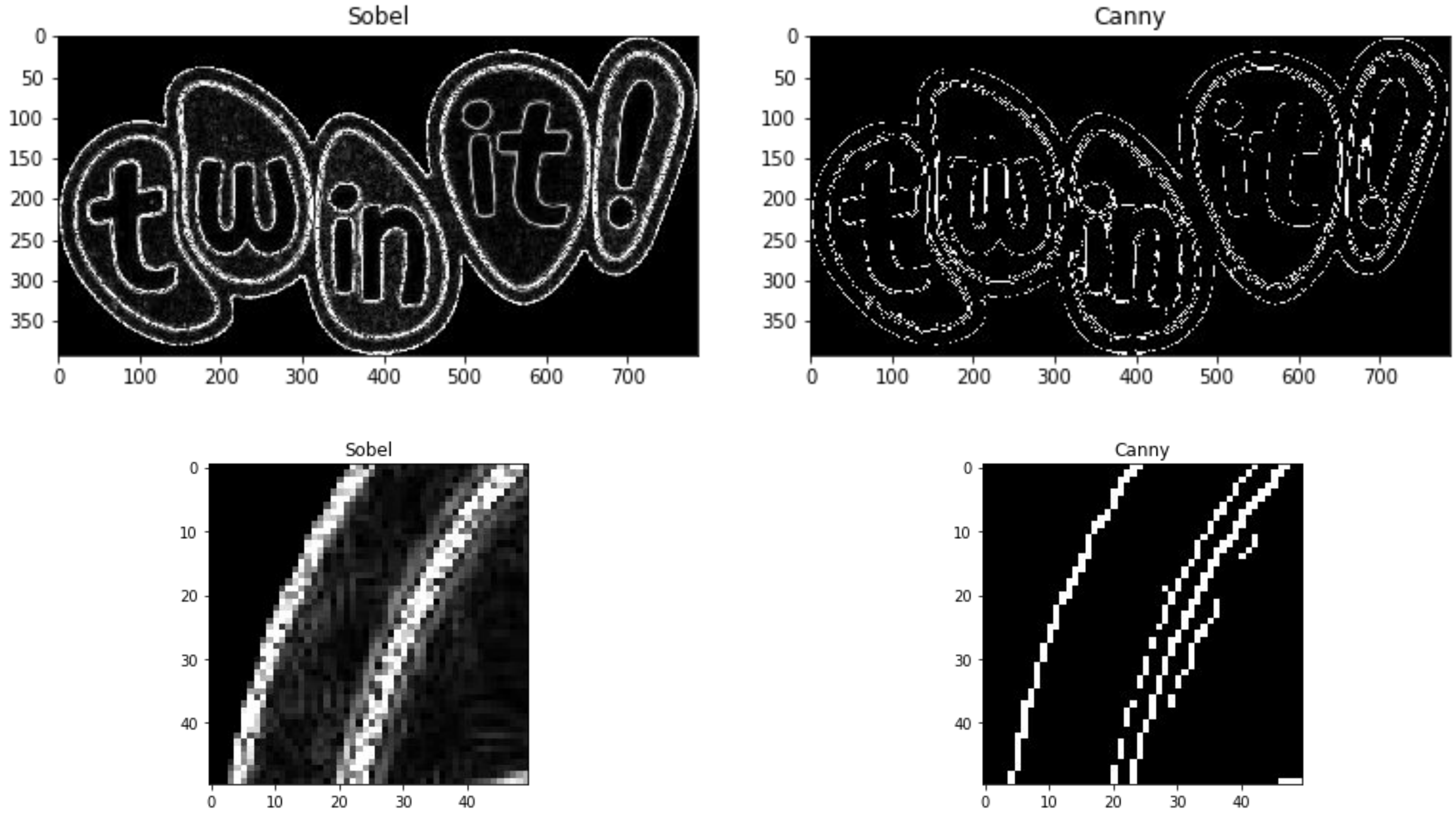
Canny: Non-maximum suppression



Canny: Non-maximum suppression



Canny: Non-maximum suppression



Canny: finalization

Threshold edges

- Still some noise
- Only want strong edges
- 2 thresholds, 3 cases
  - R > T: strong edge
  - R < T but R > t: weak edge
  - R < t: no edge
- Why two thresholds?

Connect weak edges to strong edges

- Strong edges are edges!
- Weak edges are edges
  iff they connect to strong
- Look in some neighborhood
  (usually 8 closest)

Corner detectors
Introduction & Harris detector

Good features

Reminder:

Good features are unique!

- Can find the “same” feature easily
- Not mistaken for “different” features

Good features are robust under perturbation

- Can detect them under translation, rotation…
- Intensity shift…
- Noise…

How close are two patches?

- Sum squared difference
- Images I, J
- $\Sigma_{x,y} (I(x,y) - J(x,y))^2$

How can we find unique patches?

Say we are stitching a panorama

Want patches in image to match to other image

Need to only match one spot

How can we find unique patches?

Sky? Bad!

- Very little variation
- Could match any other sky

How can we find unique patches?

Sky? Bad!

- Very little variation
- Could match any other sky

Edge? OK...

- Variation in one direction
- Could match other patches along same edge

How can we find unique patches?

Sky? Bad!

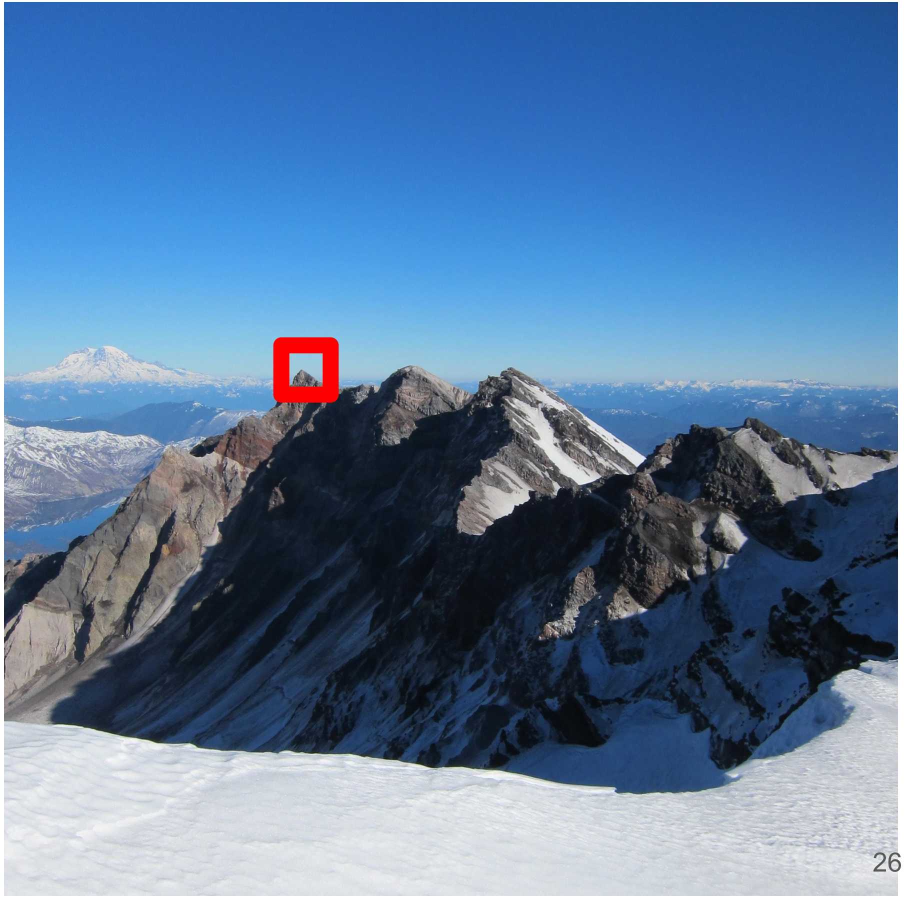
- Very little variation
- Could match any other sky

Edge? OK...

- Variation in one direction
- Could match other patches along same edge

Corners? good!

- Only one alignment matches



How can we find unique patches?

Want a patch that is unique in the image

Can calculate distance between patch and every other patch, lot of computation

How can we find unique patches?

Want a patch that is unique in the image

Can calculate distance between patch and every other patch, lot of computation

Instead, we could think about auto-correlation:

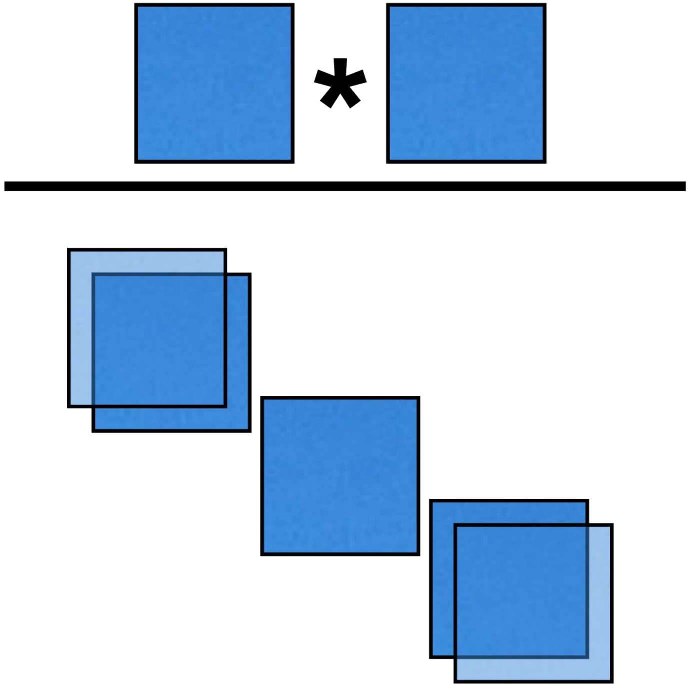
How well does image match shifted version of itself?

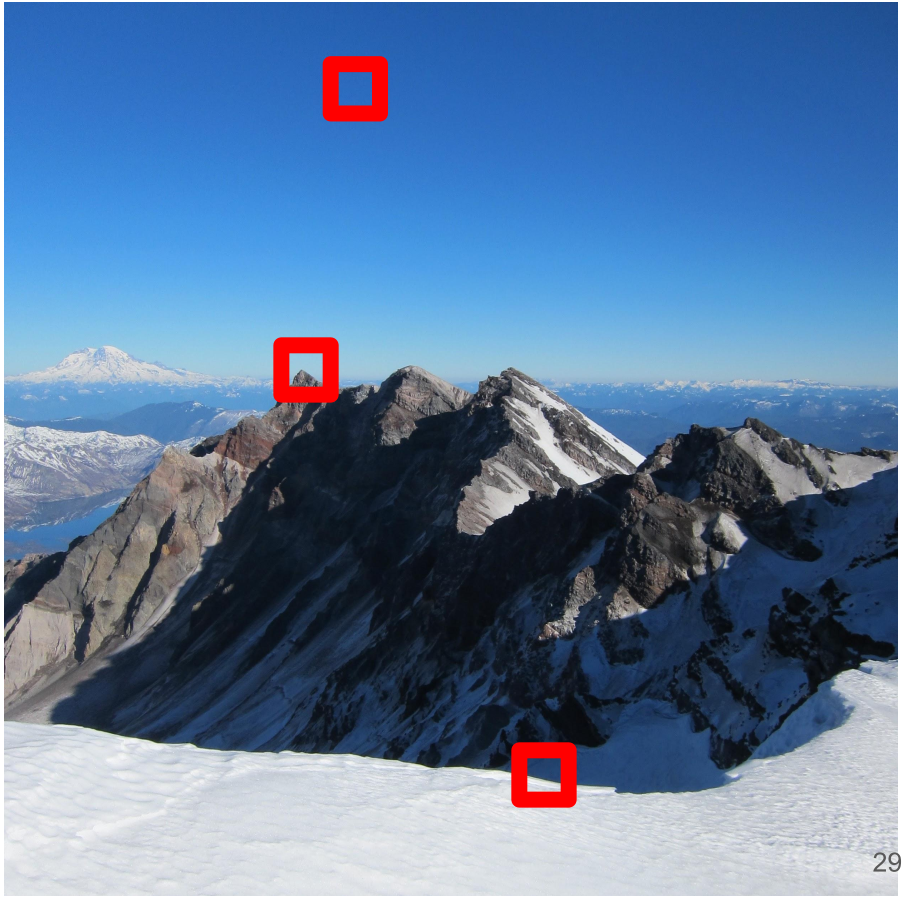
$$\Sigma_{\mathbf{d}}\Sigma_{x,y} (I(x,y) - I(x+\mathbf{d}_x,y+\mathbf{d}_y))^2$$

Measure of self-difference (how am I not myself?)

Self-difference

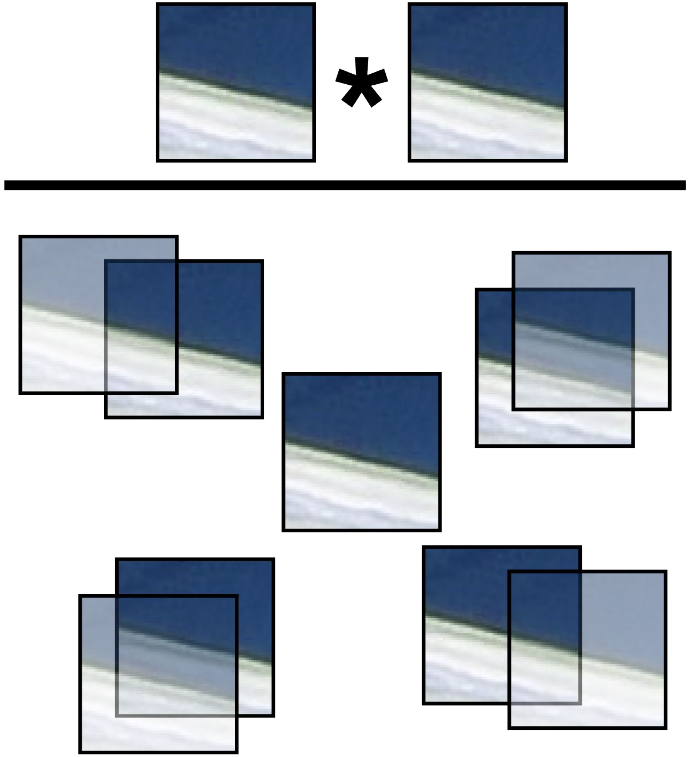
Sky: low everywhere

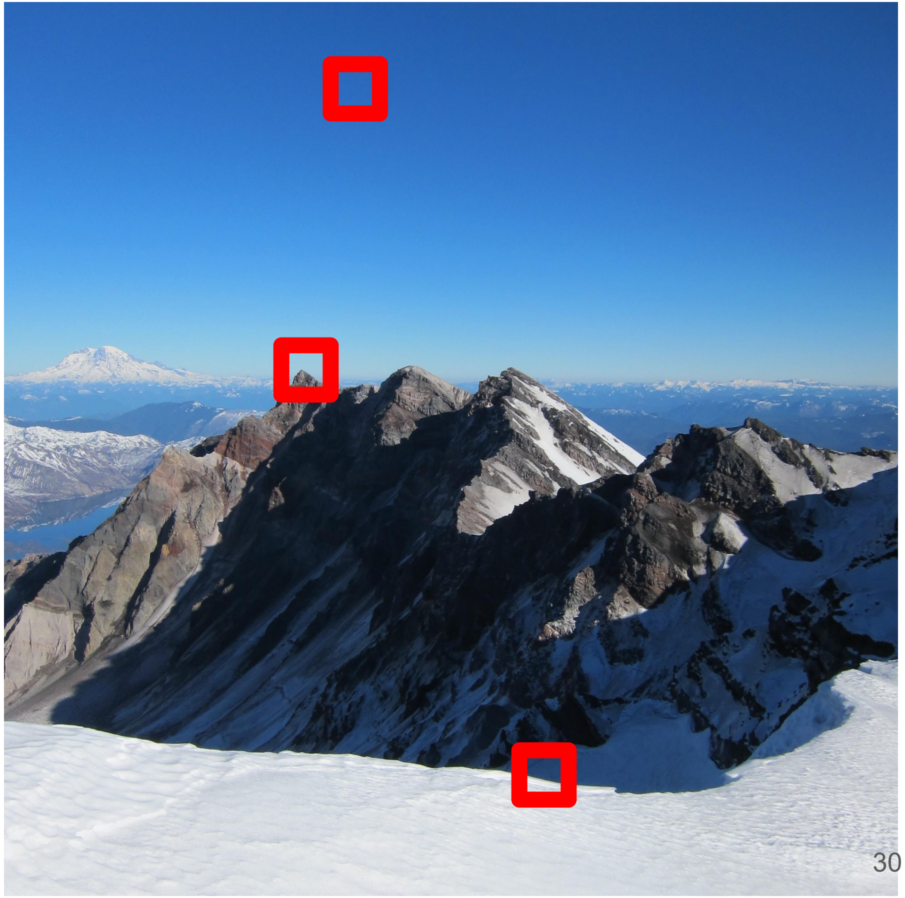




Self-difference

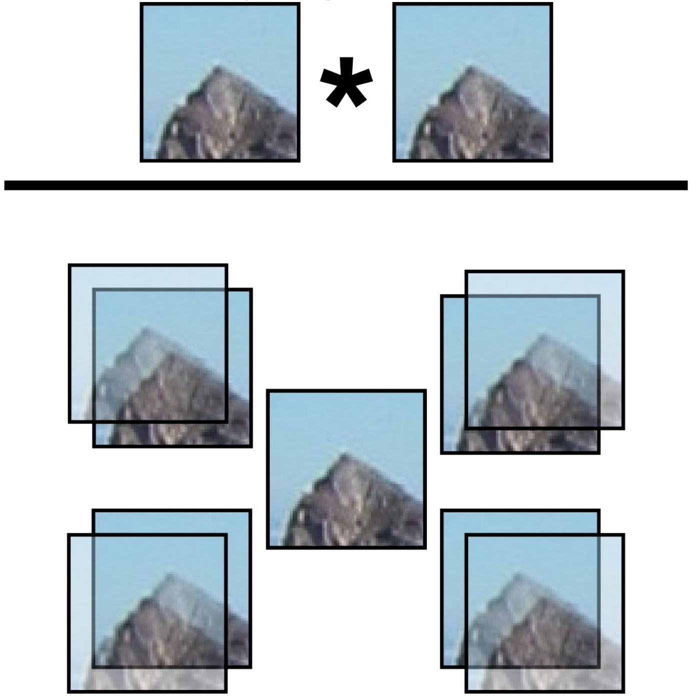
Edge: low along edge

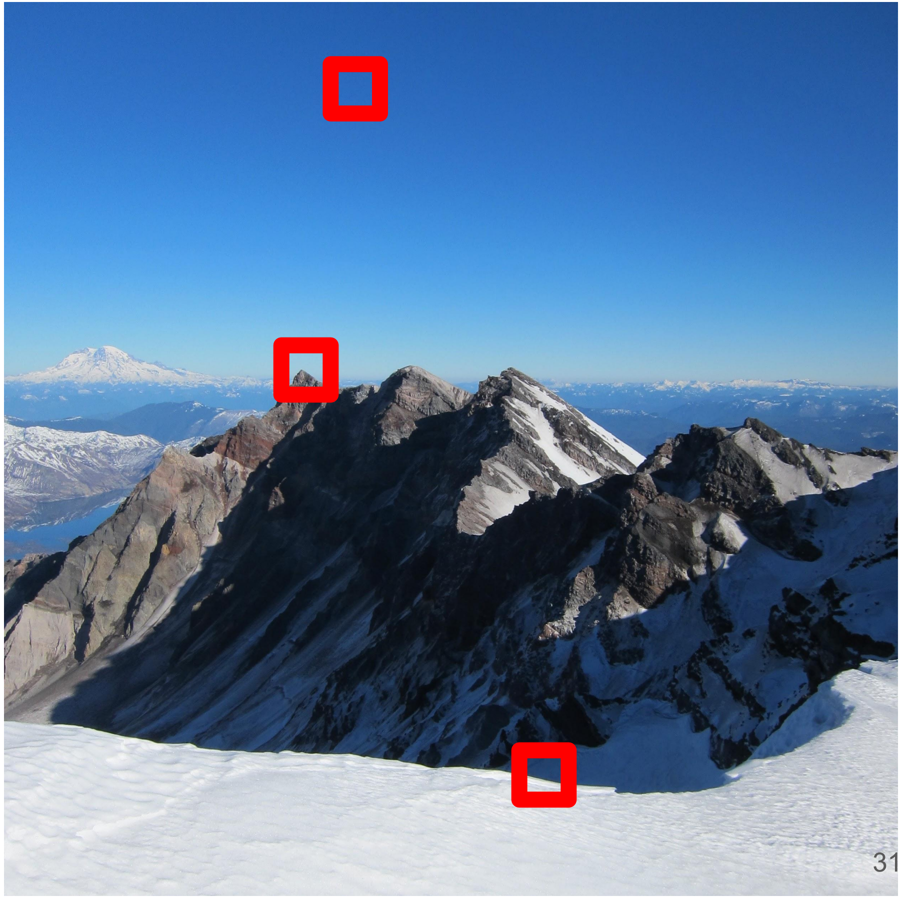




Self-difference

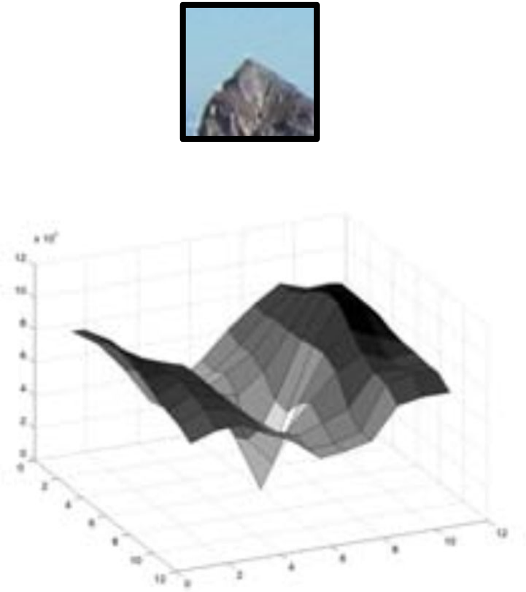
Corner: mostly high



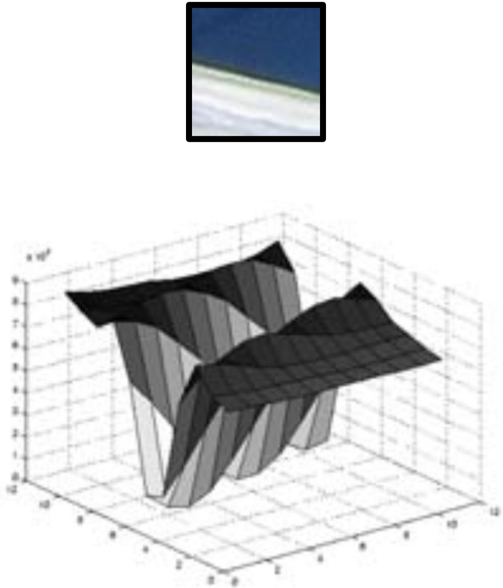


Self-difference

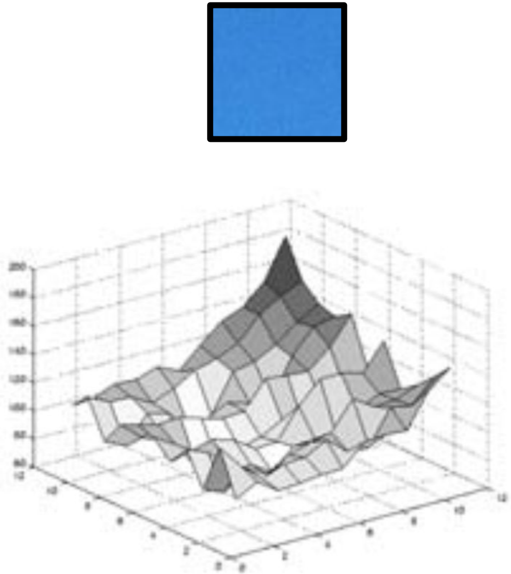
Corner: mostly high



Edge: low along edge



Sky: low everywhere



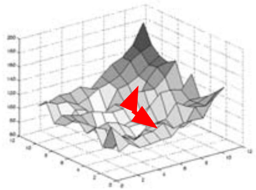
Self-difference is still expensive

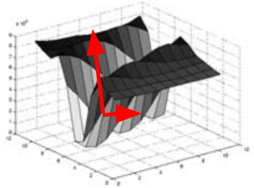
$$\Sigma_{\mathbf{d}}\Sigma_{x,y}\ (I(x,y) - I(x+\mathbf{d}_x, y+\mathbf{d}_y))^2$$

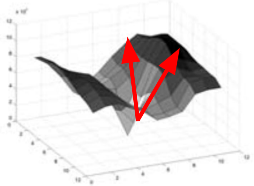
Lots of summing => Need an approximation

Look at nearby gradients Ix and Iy

- If gradients are **mostly zero**, not a lot going on
  ⇒ Low self-difference
- If gradients are **mostly in one direction**, edge
  ⇒ Still low self-difference
- If gradients are **in twoish directions**, corner!
  ⇒ High self-difference, good patch!



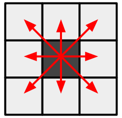




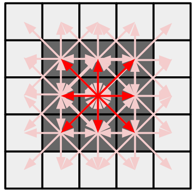
Harris corner detector

In practice we pool the previous indicator function over a small region (u,v) and we use a window $w(u,v)$ to weight the contribution of each displacement to the global sum.

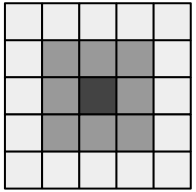
$$S(x,y) = \sum_u \sum_v w(u,v) \left(I(x+u+d_x, y+v+d_y) - I(x+u, y+v) \right)^2$$



(I(x,y) - I(x+$\mathbf{d}_x$,y+$\mathbf{d}_y$))^2



$$\sum_u \sum_v$$



$$w(u,v)$$

Harris corner detector

Trick to precompute the derivatives

$$I(x + d_x, y + d_y)$$

can be approximated by a Taylor expansion

$$I(x + d_x, y + d_y) \approx I(x, y) + d_x \frac{\partial I(x, y)}{\partial x} + d_y \frac{\partial I(x, y)}{\partial y} + \cdots$$

Harris corner detector

This allows us to "simplify" the original equation,

$$S(x, y) \approx \sum_u \sum_v w(u, v) \left(d_x \frac{\partial I(x+u, y+v)}{\partial x} + d_y \frac{\partial I(x+u, y+v)}{\partial y} \right)^2$$

and more important making it **faster to compute**,
thanks to simpler derivatives which can be **computed for the whole image**.

Harris corner detector

If we develop the equation and write is as usual matrix form, we get:

$$S(x,y) \approx \begin{pmatrix} d_x & d_y \end{pmatrix} A(x,y) \begin{pmatrix} d_x \\ d_y \end{pmatrix}$$

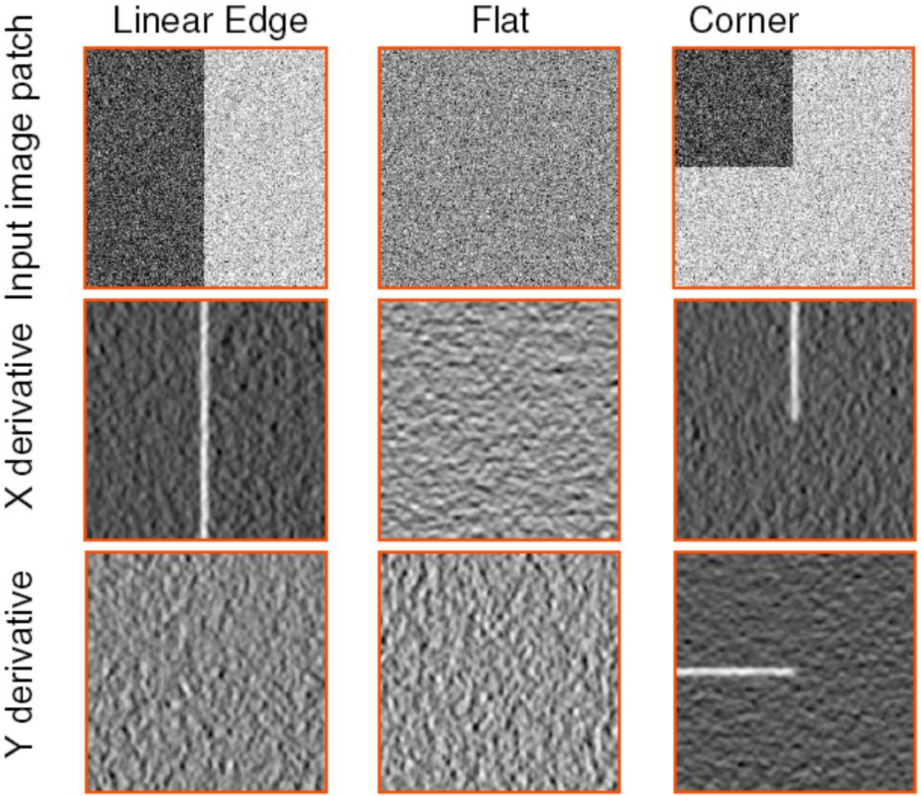
where $A(x,y)$ is the structure tensor:

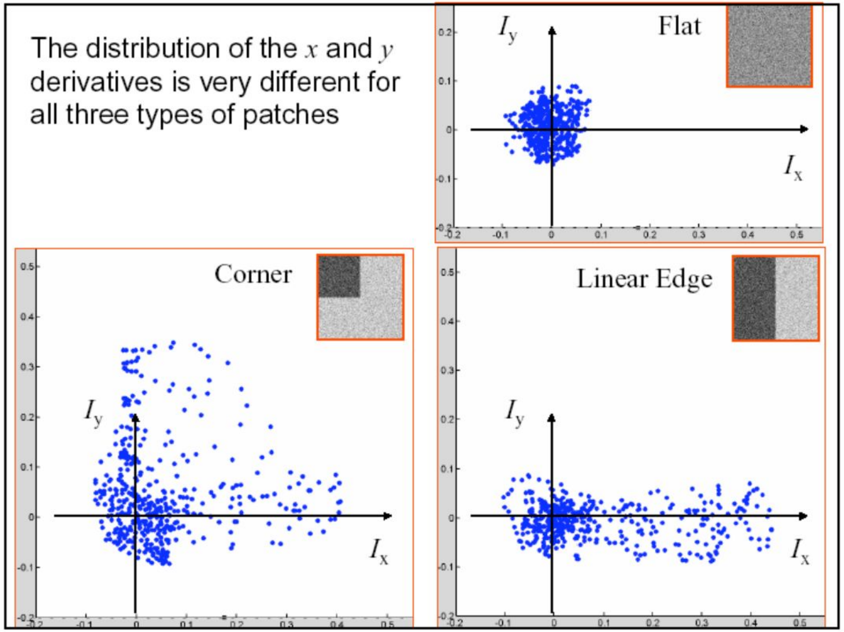
$$A = \sum_u \sum_v w(u,v) \begin{bmatrix} \frac{\partial I^2(x+u,y+v)}{\partial x} & \frac{\partial I(x+u,y+v)}{\partial x} \frac{\partial I(x+u,y+v)}{\partial y} \\ \frac{\partial I(x+u,y+v)}{\partial x} \frac{\partial I(x+u,y+v)}{\partial y} & \frac{\partial I^2(x+u,y+v)}{\partial y} \end{bmatrix}$$

$$= \begin{bmatrix} \langle I_x^2 \rangle & \langle I_x I_y \rangle \\ \langle I_x I_y \rangle & \langle I_y^2 \rangle \end{bmatrix}$$

This trick is useful because Ix and Iy can be precomputed very simply.

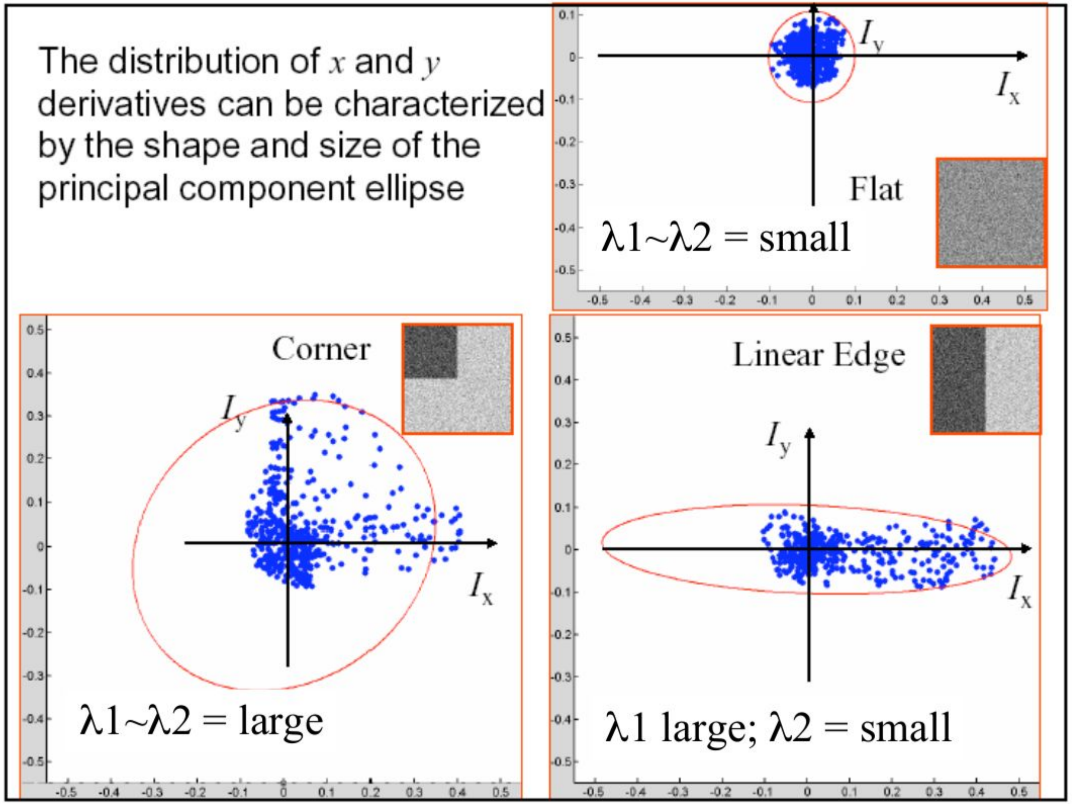
Harris corner detector





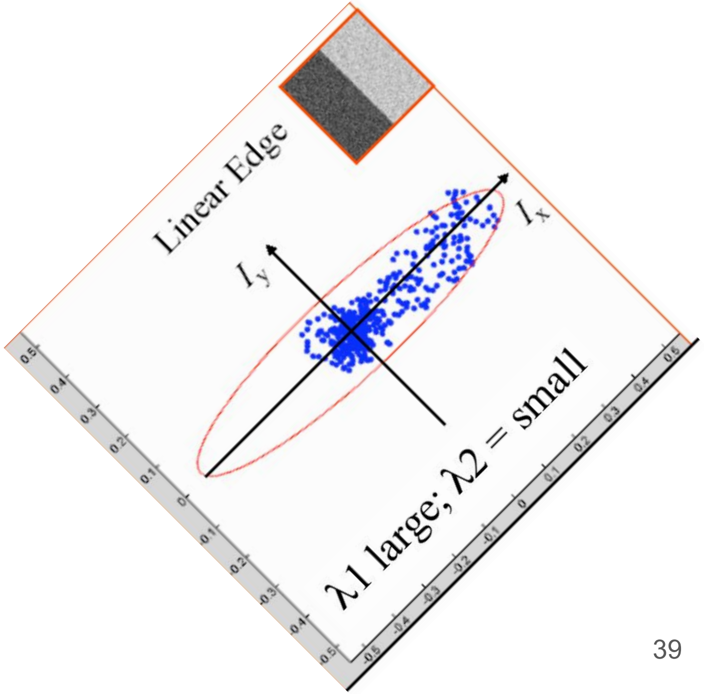
Illustrations: Robert Collins

Harris corner detector



The need for eigenvalues:
If the edge is rotated,
so are the values of I_x and I_y.

Eigenvalues give us the ellipsis axis len.



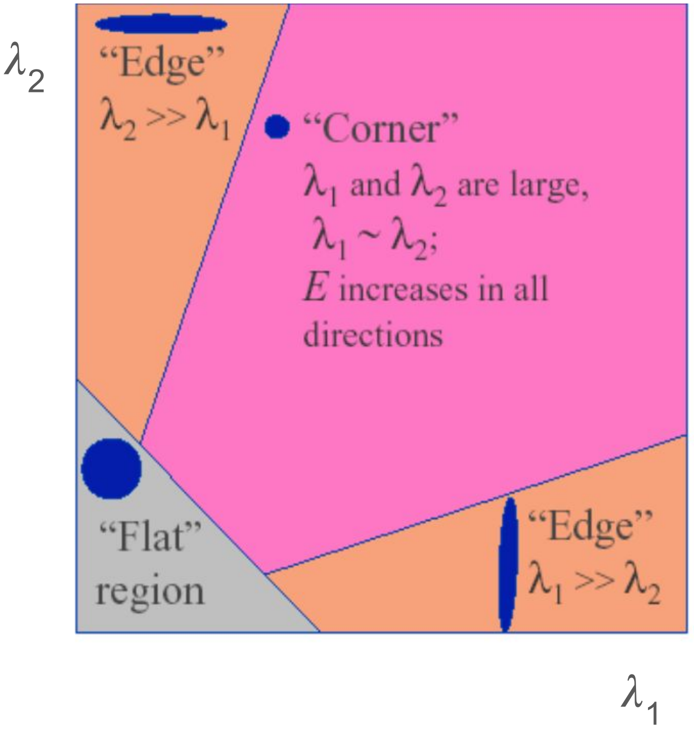
Illustrations: Robert Collins

Harris corner detector

A corner is characterized by a large variation of S in all directions of the vector $(x\ y)$.

Analyse the eigenvalues of A to check whether we have two large variations.

- If $\lambda_1 \approx 0$ and $\lambda_2 \approx 0$ then this pixel (x,y) has no features of interest.
- If $\lambda_1 \approx 0$ and λ_2 has some large positive value, then an edge is found.
- If λ_1 and λ_2 have large positive values, then a corner is found.



Illustrations: Robert Collins

Harris corner detector

To avoid the computation of the eigenvalues, which used to be expensive, Harris and Stephens instead suggest the following function Mc , where κ is a tunable sensitivity parameter:

$$M_c = \lambda_1 \lambda_2 - \kappa(\lambda_1 + \lambda_2)^2 = \underbrace{\det(A) - \kappa \,\mathrm{trace}^2(A)}_{\text{approximation}}$$

We will use Noble's trick to remove κ:

$$M_c' = 2\frac{\det(A)}{\mathrm{trace}(A) + \epsilon}$$

ϵ being a small positive constant.

Harris corner detector

A being a 2x2 matrix, we have the following relations:

- $\det(A) = A_{1,1}A_{2,2} - A_{2,1}A_{1,2}$
- $\mathrm{trace}(A) = A_{1,1} + A_{2,2}$

Using previous definitions, we obtain:

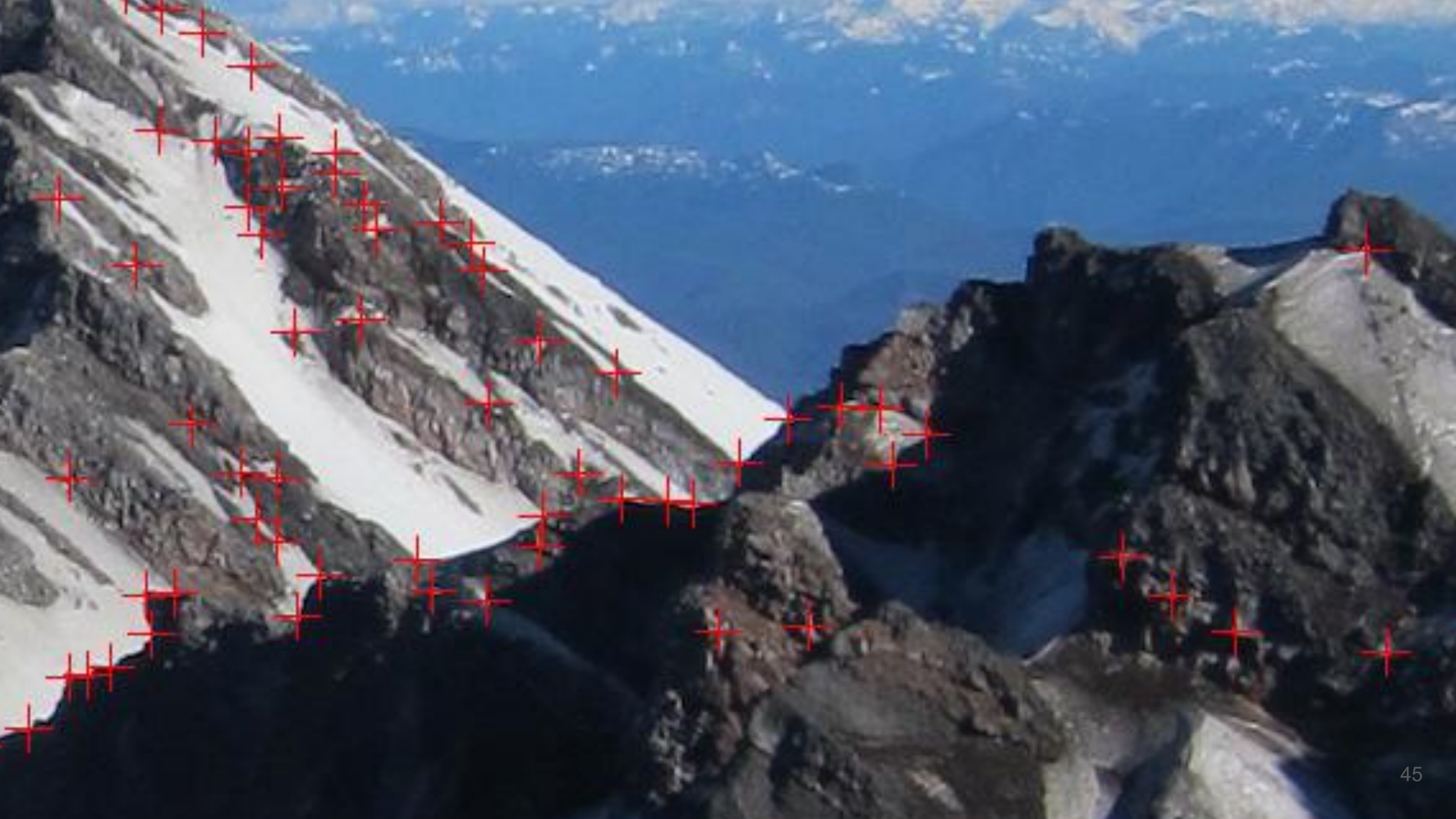
- $\det(A) = \langle I^2x \rangle \langle I^2y \rangle - \langle IxIy \rangle^2$
- $\mathrm{trace}(A) = \langle I^2x \rangle + \langle I^2y \rangle$

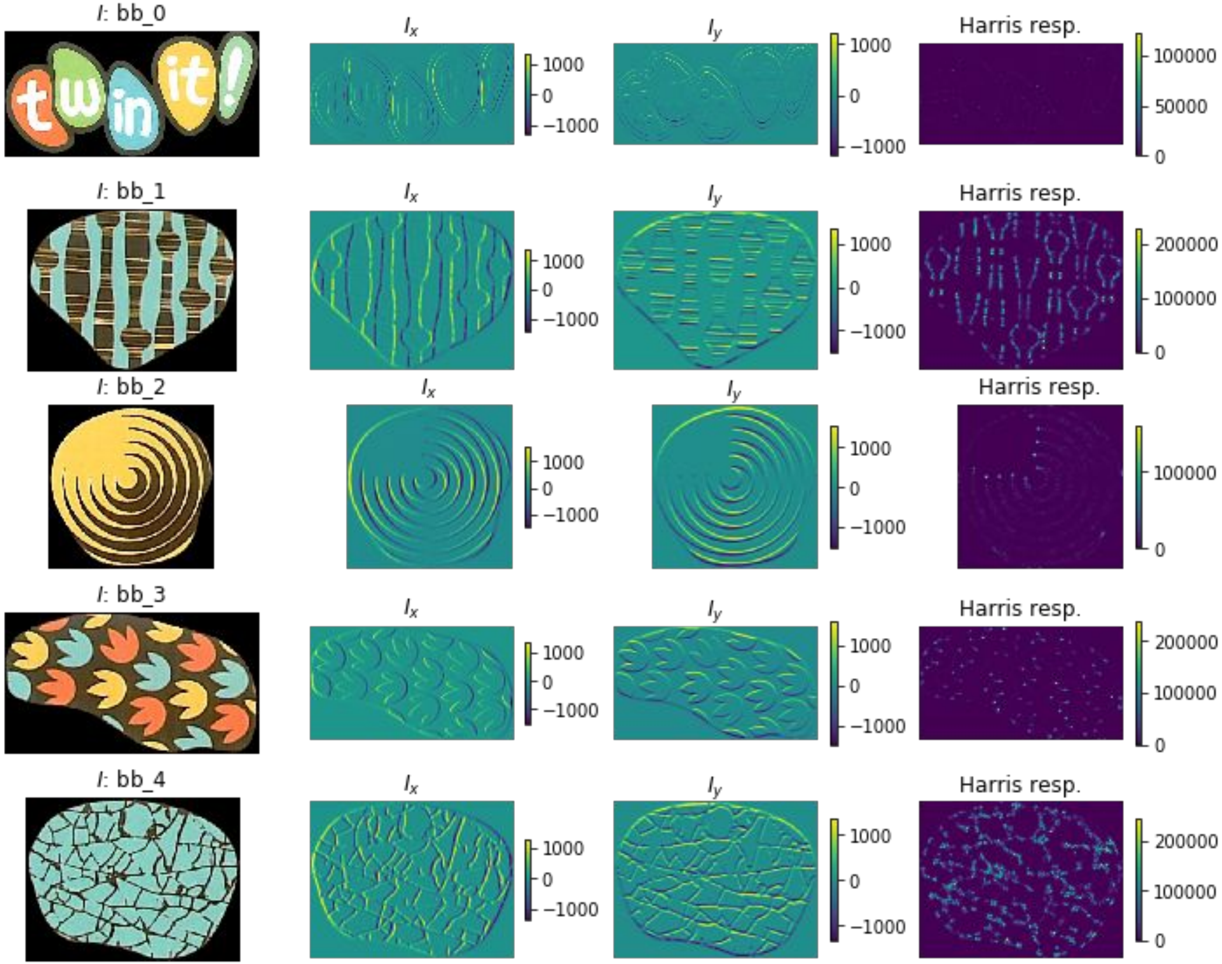
Harris corner detector

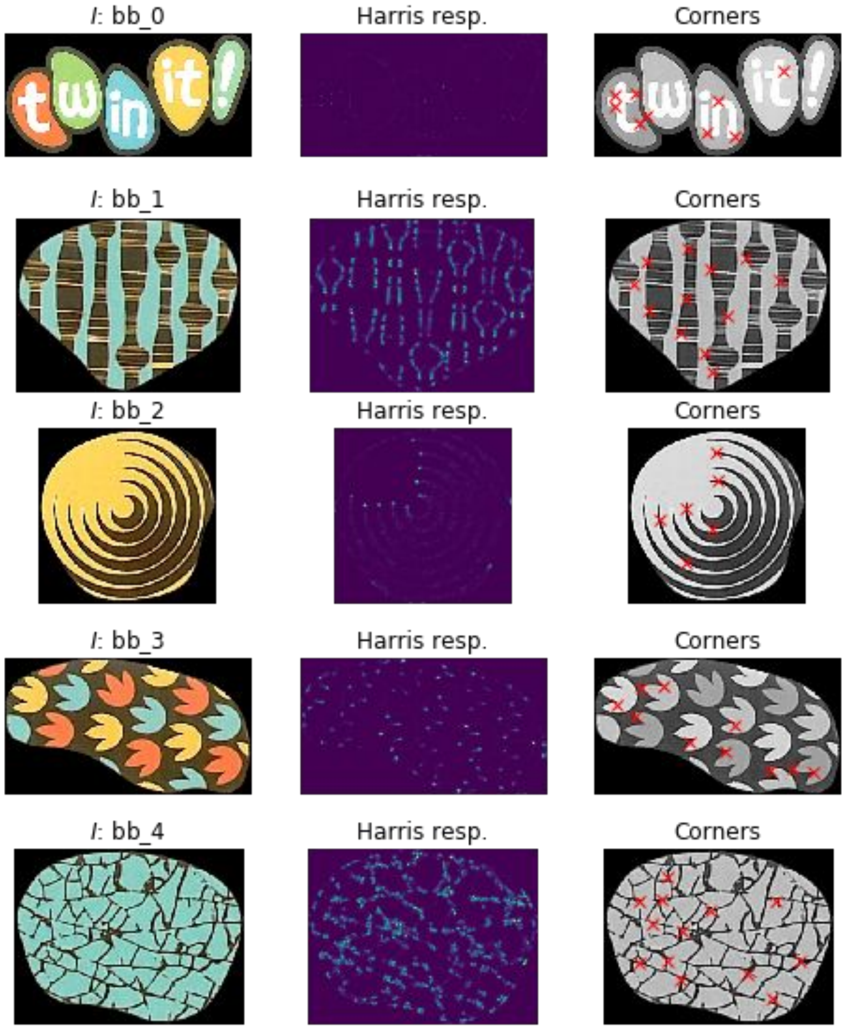
In summary, given an image, we can compute the Harris corner response image by simply computing:

- Ix : I 's smoothed (interpolated) partial derivative with respect to x ;
- Iy : I 's smoothed (interpolated) partial derivative with respect to y ;
- $\langle I^2x \rangle$: the windowed sum of I^2x ;
- $\langle I^2y \rangle$: the windowed sum of I^2y ;
- $\langle IxIy \rangle$: the windowed sum of $IxIy$;
- det(A) ;
- trace(A) ;
- M''_c=det(A) / (trace(A)+ϵ).

Then, we just perform **non-maximal suppression** to keep local maximas.




I: bb_0
twin it!
I_x
I_y
Harris resp.
1000
0
−1000
100000
50000
0
I: bb_1
200000
100000
I: bb_2
I: bb_3
I: bb_4


I: bb_0
Harris resp.
Corners
twin it!
I: bb_1
Harris resp.
Corners
I: bb_2
Harris resp.
Corners
I: bb_3
Harris resp.
Corners
I: bb_4
Harris resp.
Corners