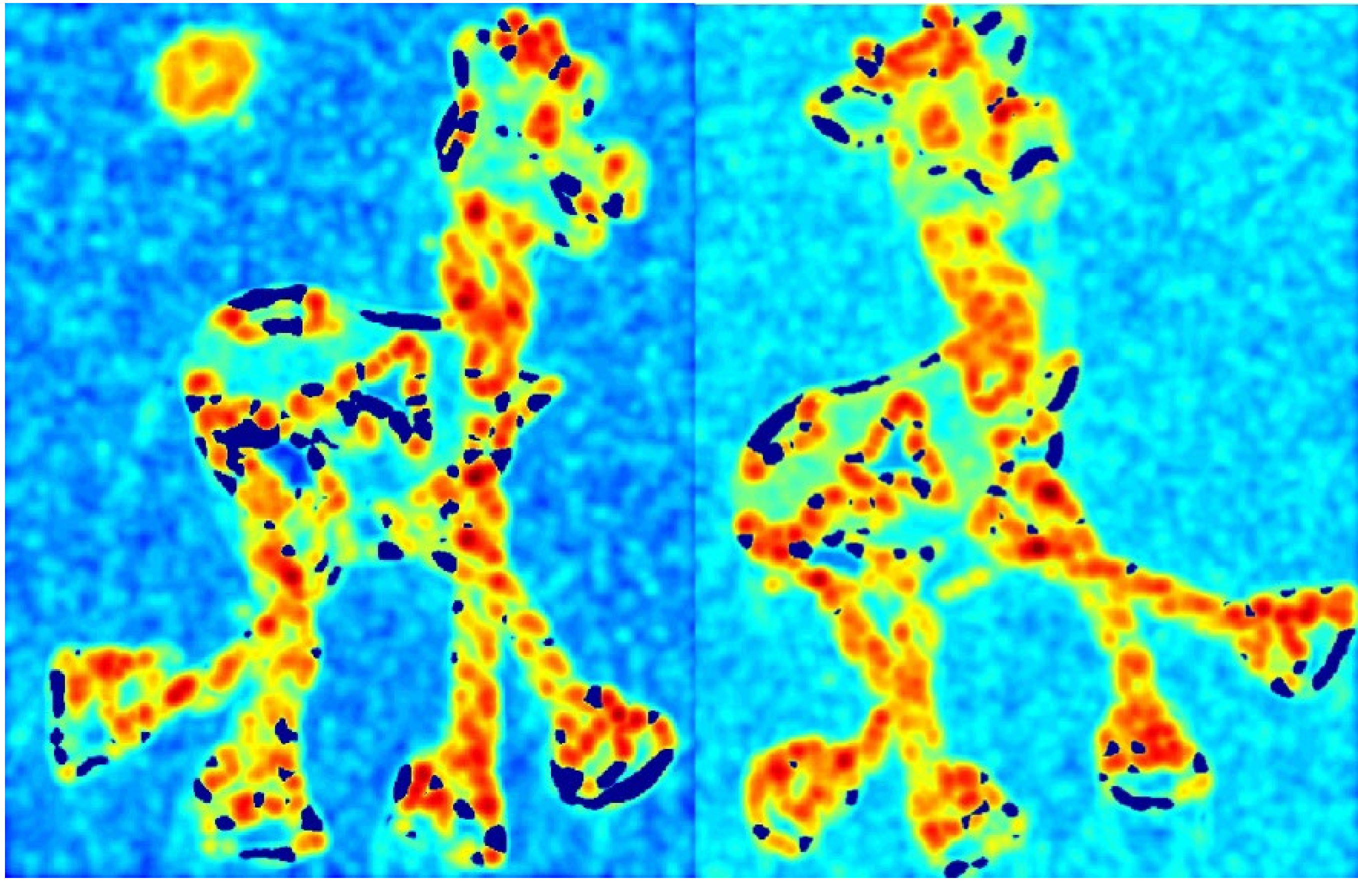
CMSC 491/691

Lecture 5

Image Features

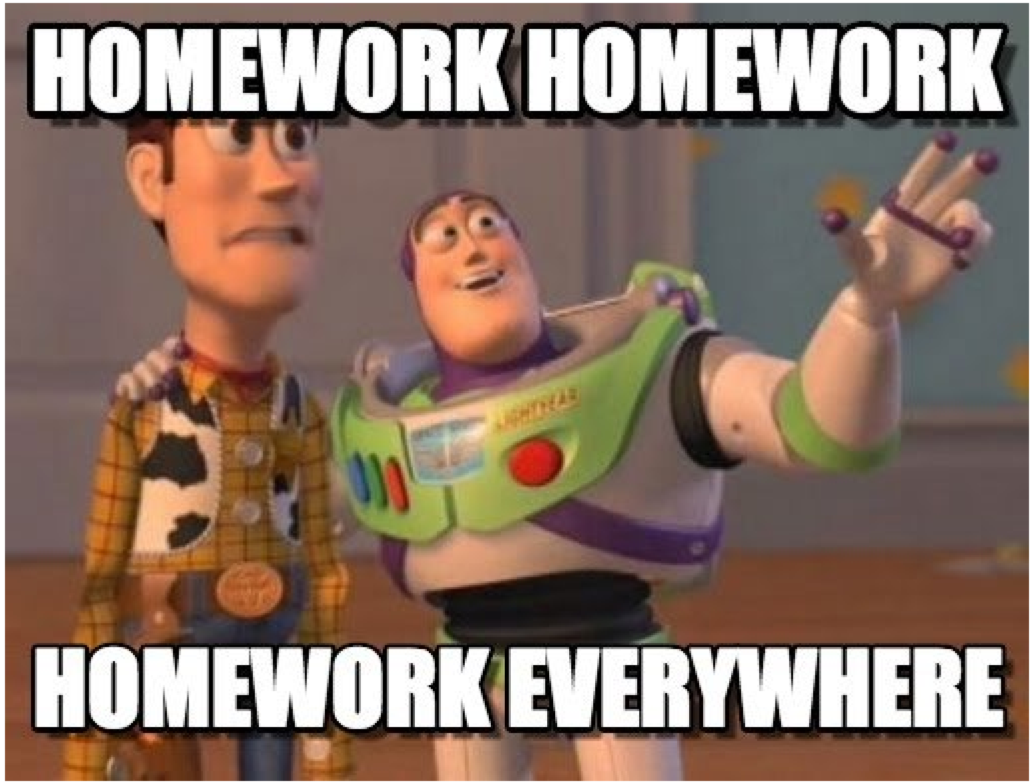


Some slides from Kitani, Jayasuriya, Snavely

Announcements

HW1 has been released

- Start early. Due on Feb 23.
- TA is an expert in Python and OpenCV
  - Seek help early!
- Submit on Blackboard
- What to submit?
  - See instructions in PDF
  - We want answers, code snippets, results, .. *in the PDF*



Announcements

Start looking for teammates for the group project:

- proposals will be due soon

https://redirect.cs.umbc.edu/courses/graduate/691cv/

Projects

The class has a mix of PhD, MS, and BS students. Projects will be judged on the basis of relative growth (from where you start to where you end).

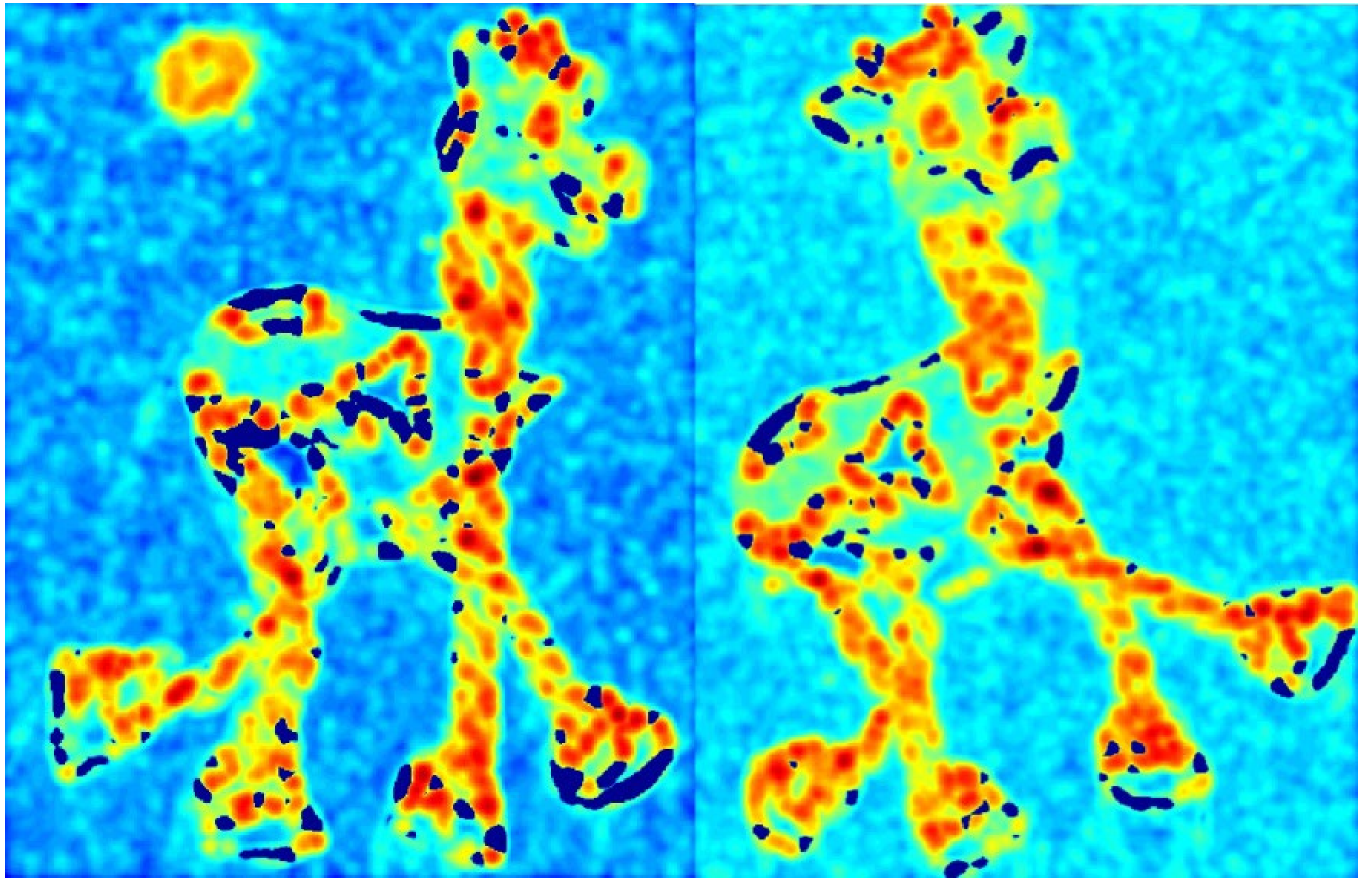
- *BS or MS (coursework) students:* Pick one of the suggested topics. If you want to work on a cool idea of your own, come see Tejas and we can create a concrete structure and gameplan. I recommend working in groups of 4 students.
- *PhD or MS (thesis) students:* Consult with Tejas during Office Hours and discuss your existing research agenda. We will integrate the course project into that agenda if possible. Group sizes (or individual projects) will be decided on a case-by-case basis.

- **Proposal:** Clearly state the following:
  - Problem you wish to tackle (and why)
  - Proposed approach and methods
  - Timeline
  - What each student in the group will do.
  - Expected Outcome and Worst-Case Outcome

CMSC 491/691

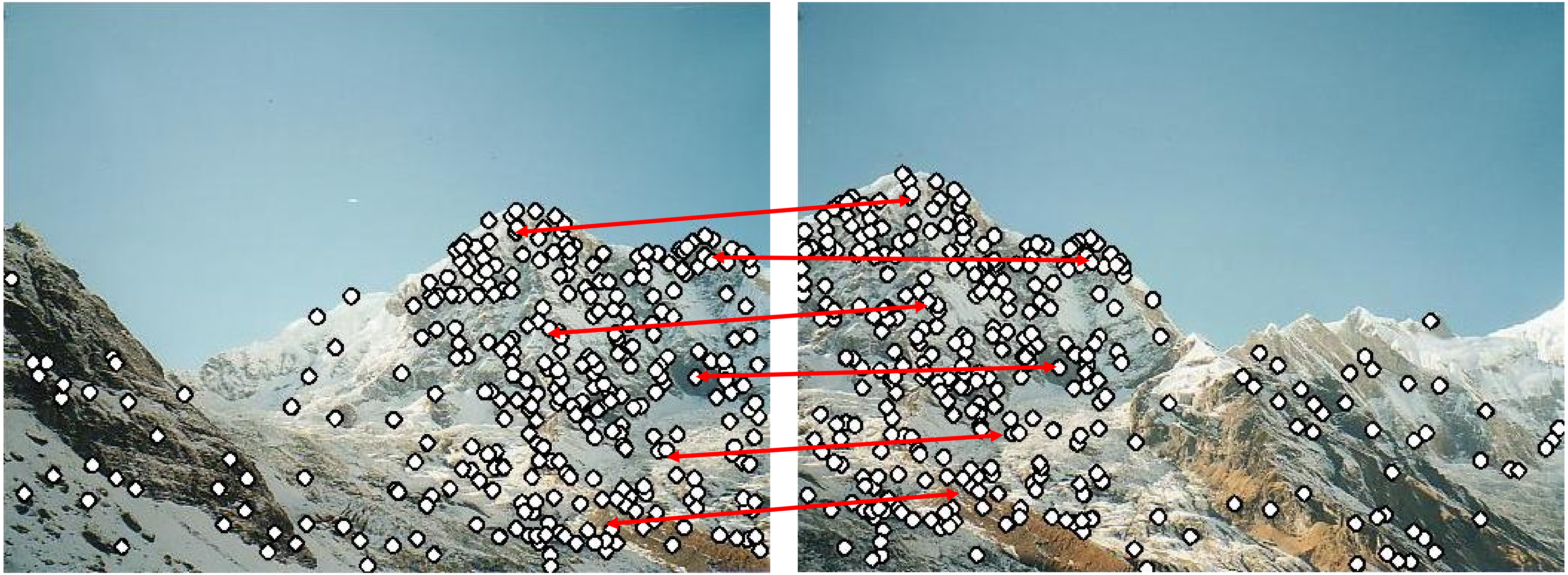
Lecture 5

Image Features



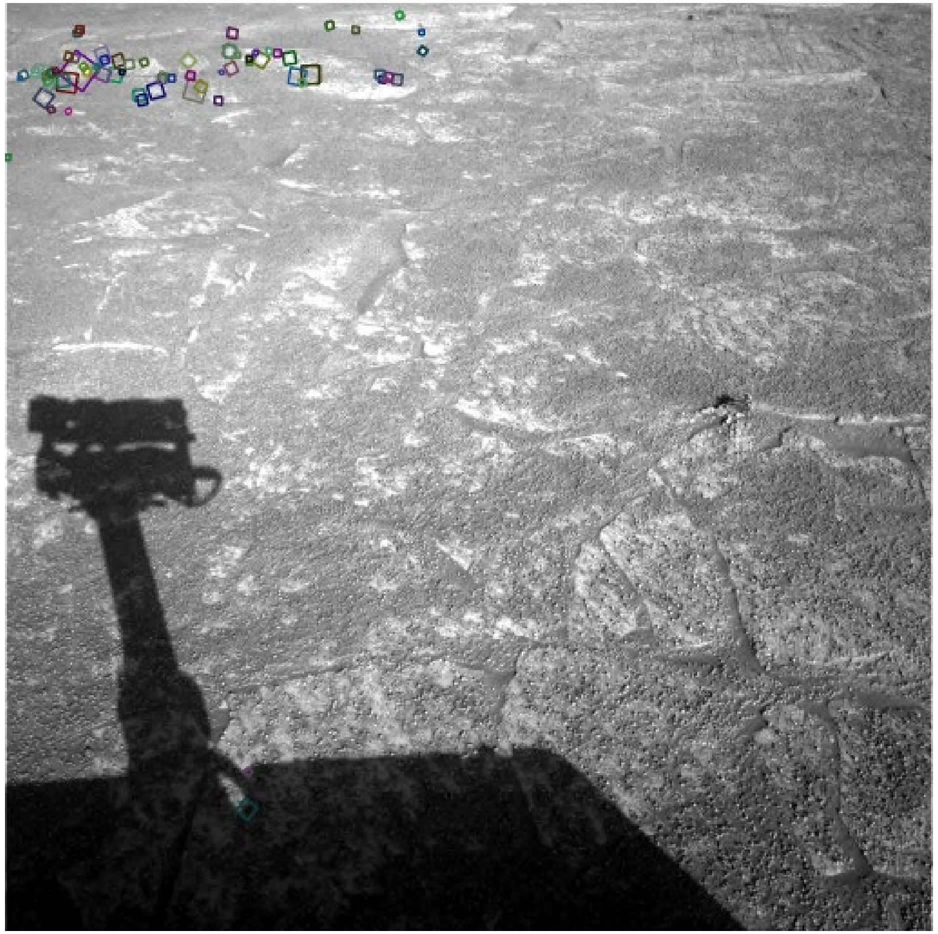
Are these images related?

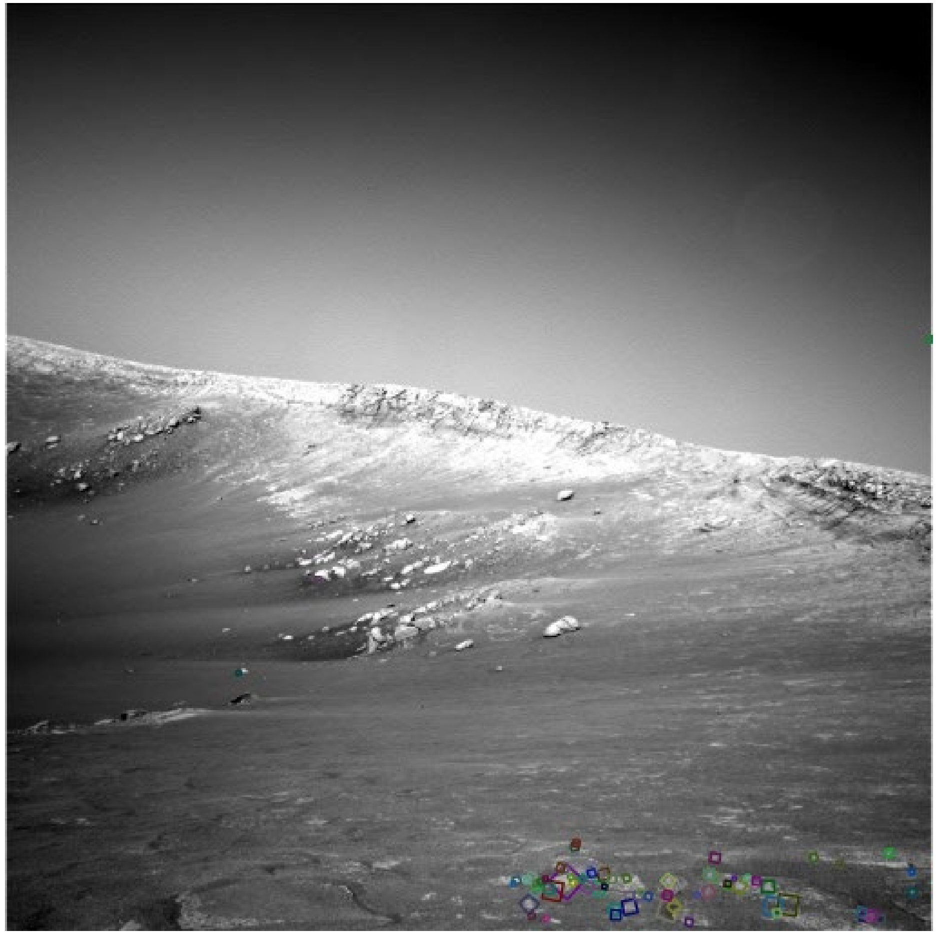
Are these images related?



Yes! They share common **features.**

Are these images related?





NASA Mars Rover images
with SIFT feature matches

What makes a good feature?

Properties of “Good Features”

- Image regions that are “important”
- Image regions that are “unusual”
- Uniqueness

How to define “unusual”, “important” ?

Why are we interested in features?

Motivation I:

Object Search

Why are we interested in features?

Motivation II:

Image Stitching

Step 1: extract features
Step 2: match features
Step 3: align images

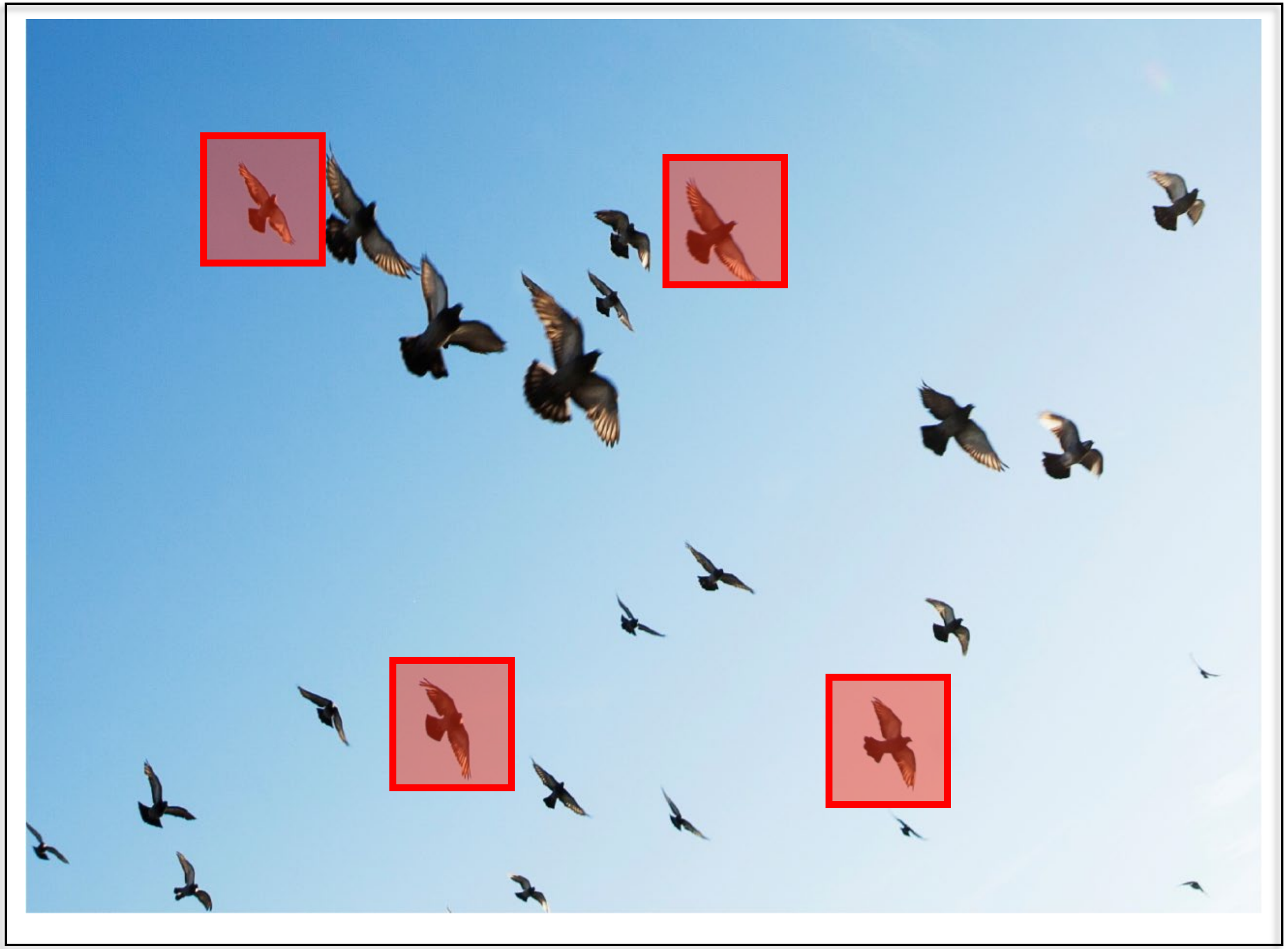
Why are we interested in features?

Motivation III:

Object Detection

Object Counting

Pattern Recognition



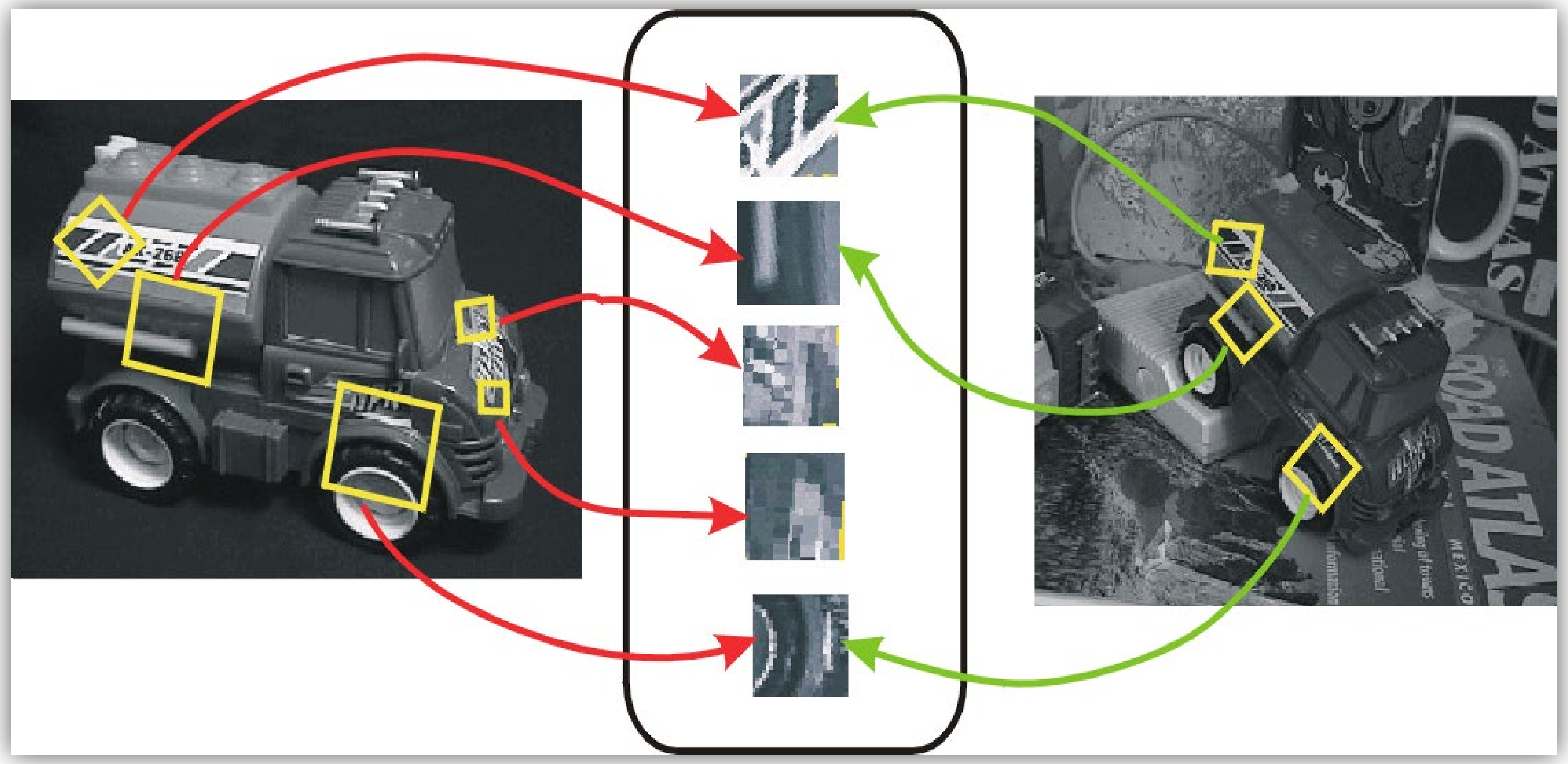
Features are used for …

- Image alignment, panoramas, mosaics …
- 3D reconstruction
- Motion tracking (e.g. for augmented reality)
- Object recognition
- Image retrieval
- Autonomous navigation
- …

Invariant Local Features

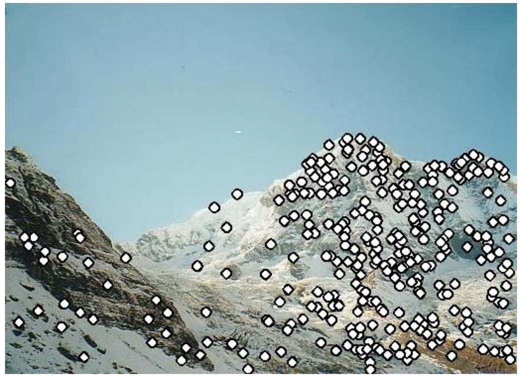
Main Idea: Find features that are invariant to transformations

- Geometric invariance (rotation, translation, scaling, ...)
- Photometric invariance (brightness, exposure, shadows, ...)

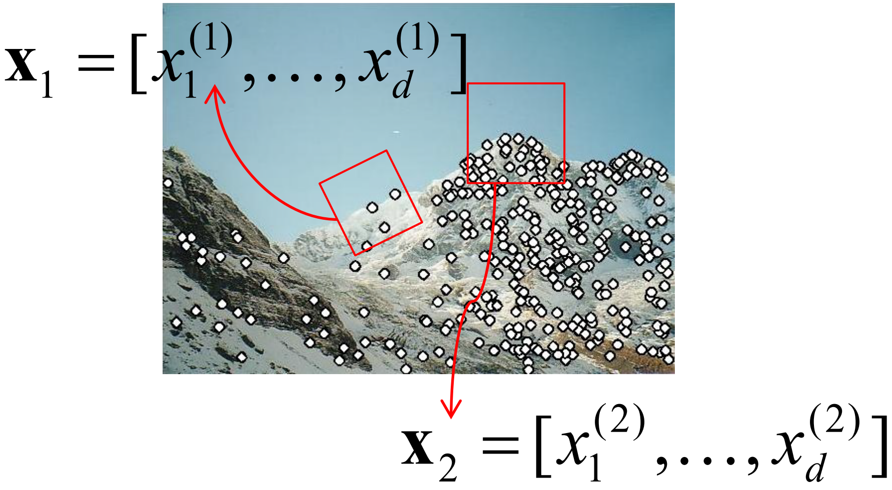


Local Features: Main Components

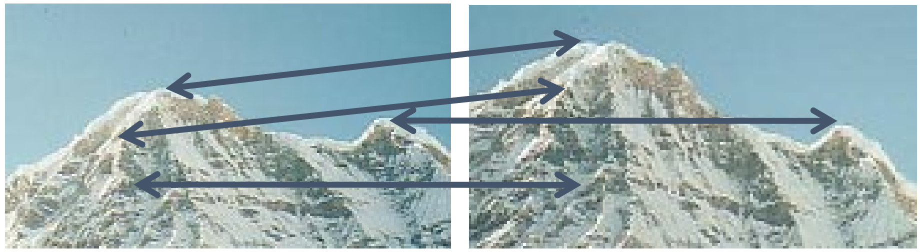
1. **DETECTION**
   Identify “interest points”



2. **DESCRIPTION**
   Extract “feature descriptor” vectors surrounding each interest point



3. **MATCHING**
   Determine correspondence between descriptors in two views



Slide Credit: Kristen Grauman

What makes a good feature?

Properties of "Good Features"

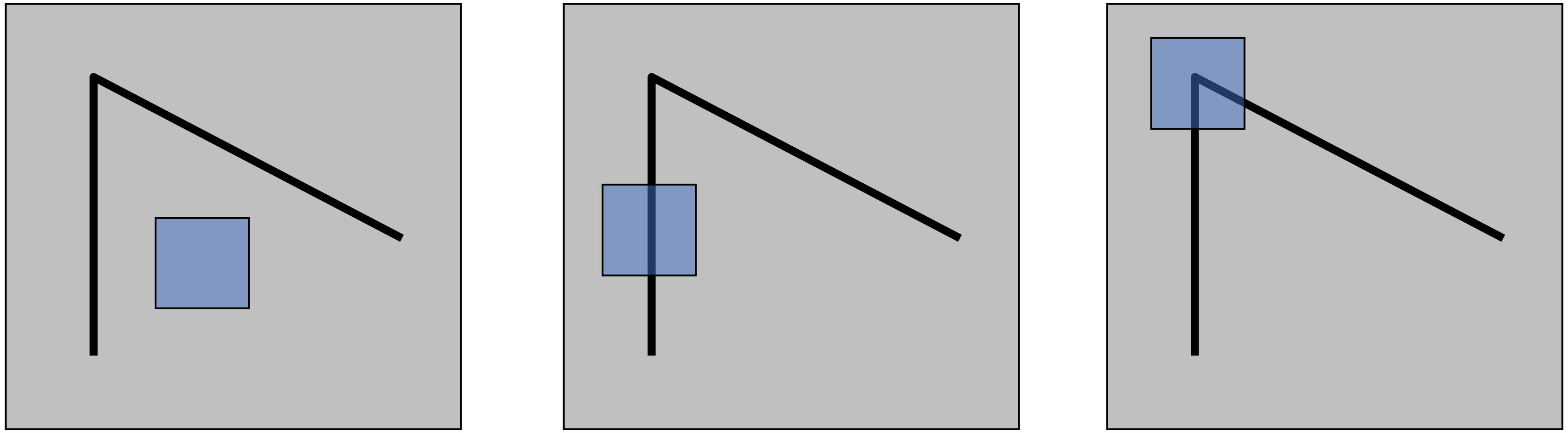
- Image regions that are **"important"**
- Image regions that are **"unusual"**
- Image regions that are **"unique"**

define "unusual", "important" ...

Harris Corner Detector [1988]

Suppose we only consider a small window of pixels

- What defines whether a feature is a good or bad candidate?

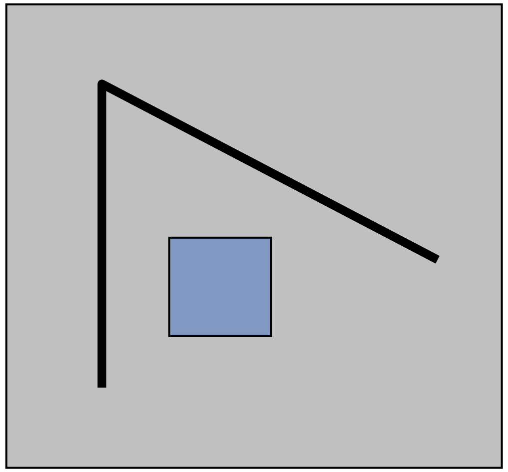


Credit: S. Seitz, D. Frolova, D. Simakov

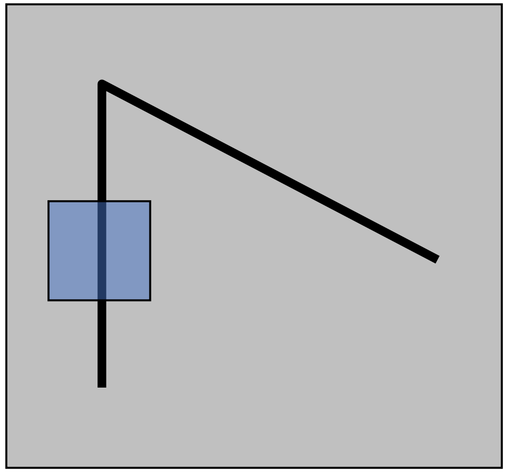
Harris Corner Detector: Intuition

Suppose we only consider a small window of pixels

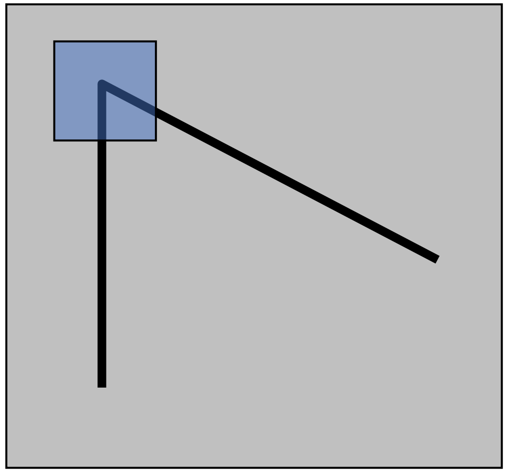
- What defines whether a feature is a good or bad candidate?



"flat" region:
no change in all directions



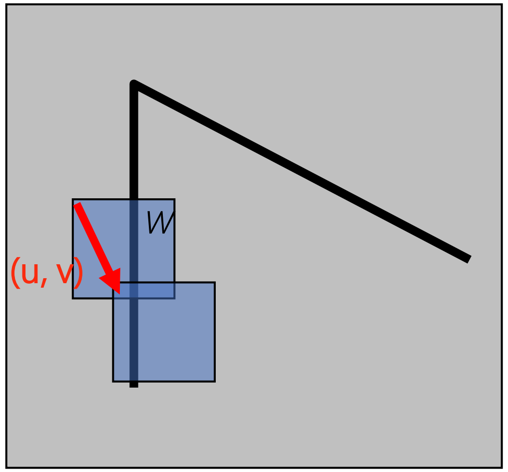
"edge":
no change along the edge direction



"corner":
significant change in all directions

Credit: S. Seitz, D. Frolova, D. Simakov

Harris Corner Detector: Intuition



- Consider a window operating over an image
- Shift the window by (u, v)
- How do pixels in W change?
  - Measure the change as the sum of squared differences (SSD)

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

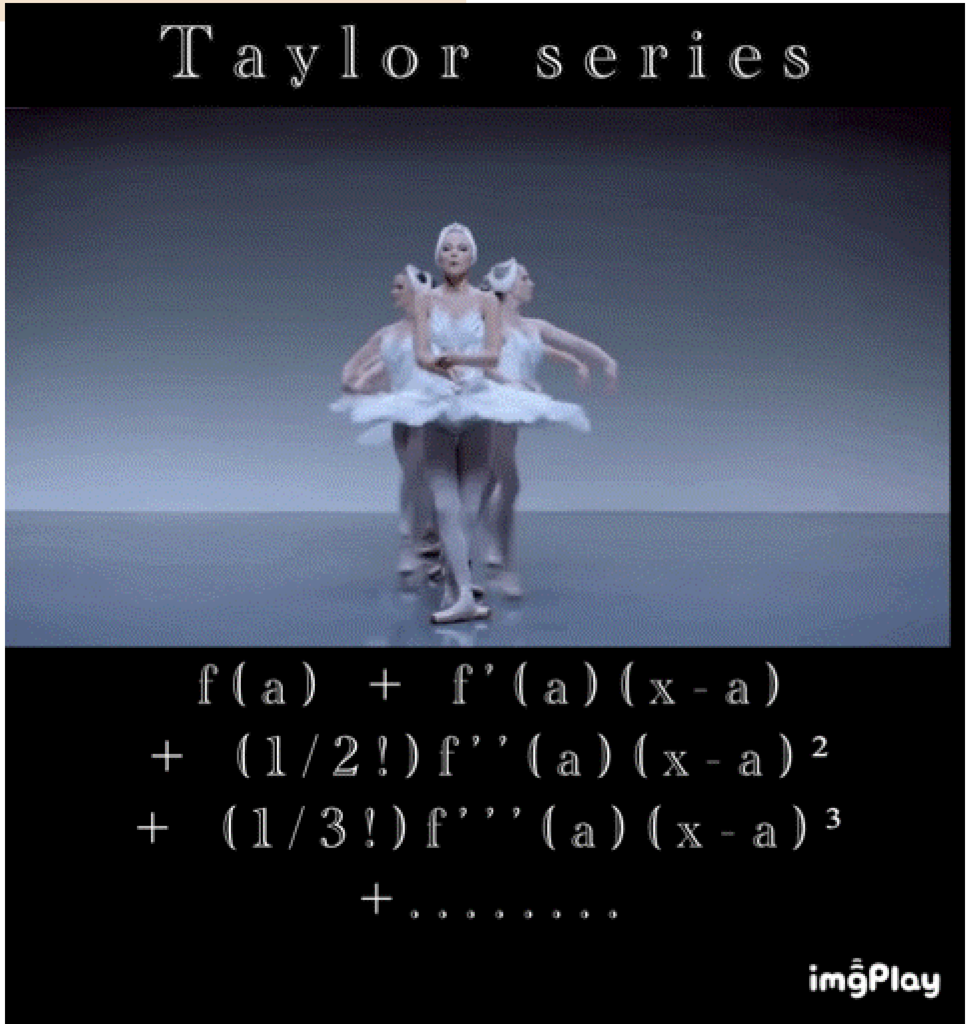
- Good feature $\Leftarrow$ High error !!!
  - We are happy if error is high
  - We are *very happy* if error is high for all shifts (u, v)
- Slow to compute error exactly for each pixel and each offset (u, v)

Small motion assumption

- We have: $$E(u,v) = \sum_{(x,y)\in W} \left[I(x+u,y+v) - I(x,y)\right]^2$$

- Taylor series expansion of I:

$$I(x+u,y+v) = I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$



Small motion assumption

- We have: $$E(u,v) = \sum_{(x,y)\in W} \left[I(x+u,y+v) - I(x,y)\right]^2$$
- Taylor series expansion of I: $I(x+u,y+v) = I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$
- If motion (u, v) is small … use first order approximation

$$I(x+u,y+v) \approx I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v \approx I(x,y) + [I_x \;\; I_y]\begin{bmatrix} u \\ v \end{bmatrix}$$

shorthand: $I_x = \frac{\partial I}{\partial x}$

- Plugging this in:

$$E(u,v) = \sum_{(x,y)\in W} \left[I(x+u,y+v) - I(x,y)\right]^2 \approx \sum_{(x,y)\in W} \left[I_x u + I_y v\right]^2$$

$$E(u,v) \approx \sum_{(x,y)\in W} [I_x u + I_y v]^2$$

$$E(u,v) \approx Au^2 + 2Buv + Cv^2$$

$$\approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y)\in W} I_x^2$$

$$B = \sum_{(x,y)\in W} I_x I_y$$

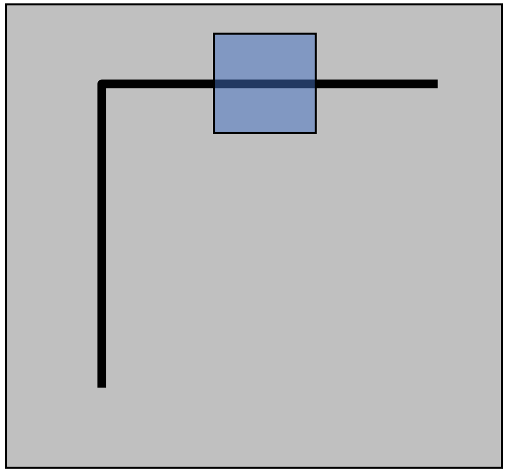
$$C = \sum_{(x,y)\in W} I_y^2$$

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y)\in W} I_x^2$$

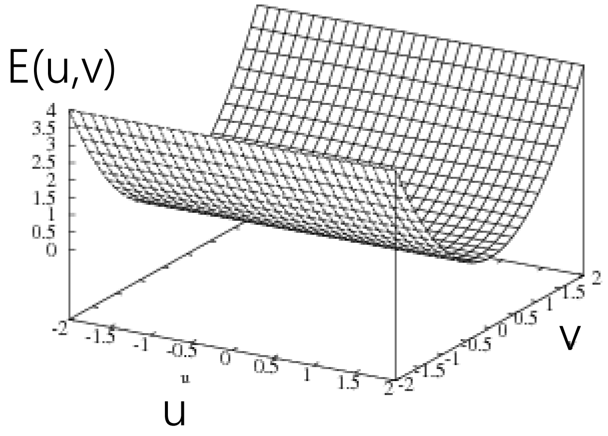
$$B = \sum_{(x,y)\in W} I_x I_y$$

$$C = \sum_{(x,y)\in W} I_y^2$$



Horizontal edge: $I_x = 0$

$$H = \begin{bmatrix} 0 & 0 \\ 0 & C \end{bmatrix}$$

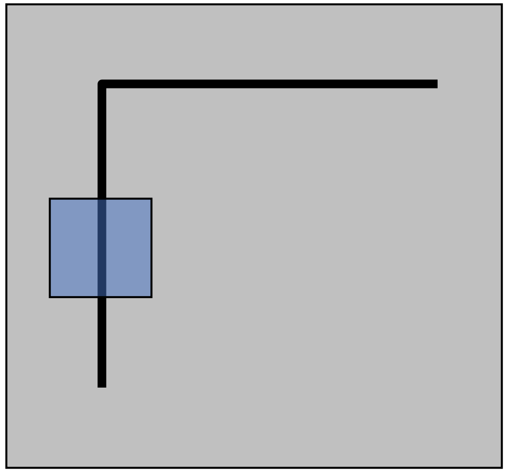


$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y)\in W} I_x^2$$

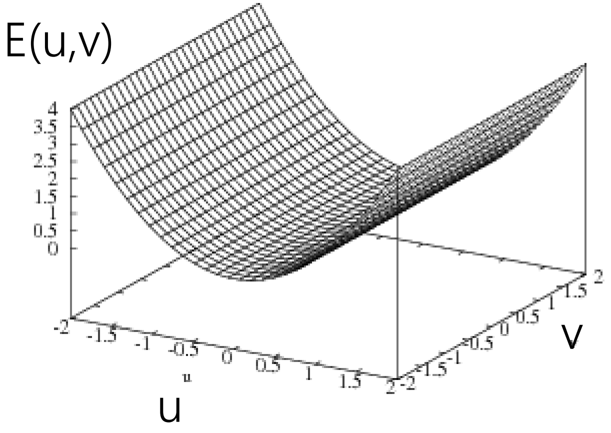
$$B = \sum_{(x,y)\in W} I_x I_y$$

$$C = \sum_{(x,y)\in W} I_y^2$$

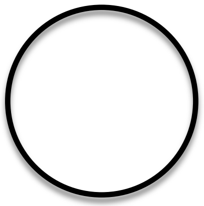


Vertical edge: $I_y = 0$

$$H = \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix}$$

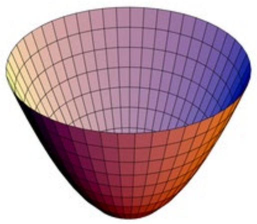


Quick Aside: Visualizing quadratics



Equation of a circle

$$1 = x^2 + y^2$$



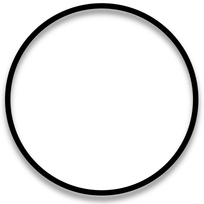
Equation of a 'bowl' (paraboloid)

$$f(x, y) = x^2 + y^2$$

If you slice the bowl at

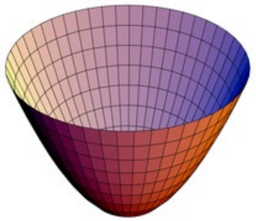
$$f(x, y) = 1$$

what do you get?



Equation of a circle

$$1 = x^2 + y^2$$



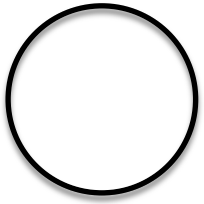
Equation of a 'bowl' (paraboloid)

$$f(x, y) = x^2 + y^2$$

If you slice the bowl at

$$f(x, y) = 1$$

what do you get?



$$f(x, y) = x^2 + y^2$$

can be written in matrix form like this…

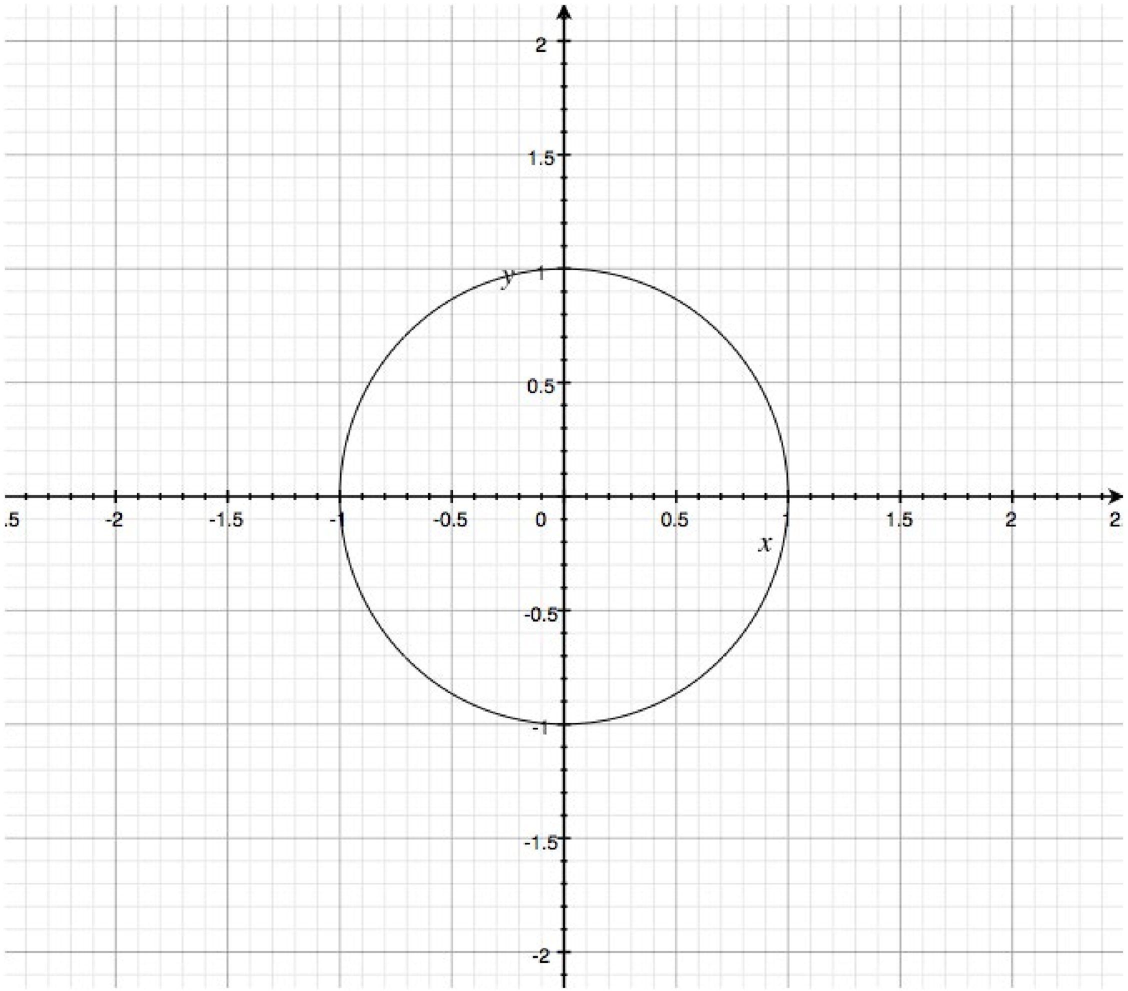
$$f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$f(x,y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

‘sliced at 1’

$$f(x,y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

'sliced at 1'



*What happens if you **increase** coefficient on **x**?*

$$f(x,y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

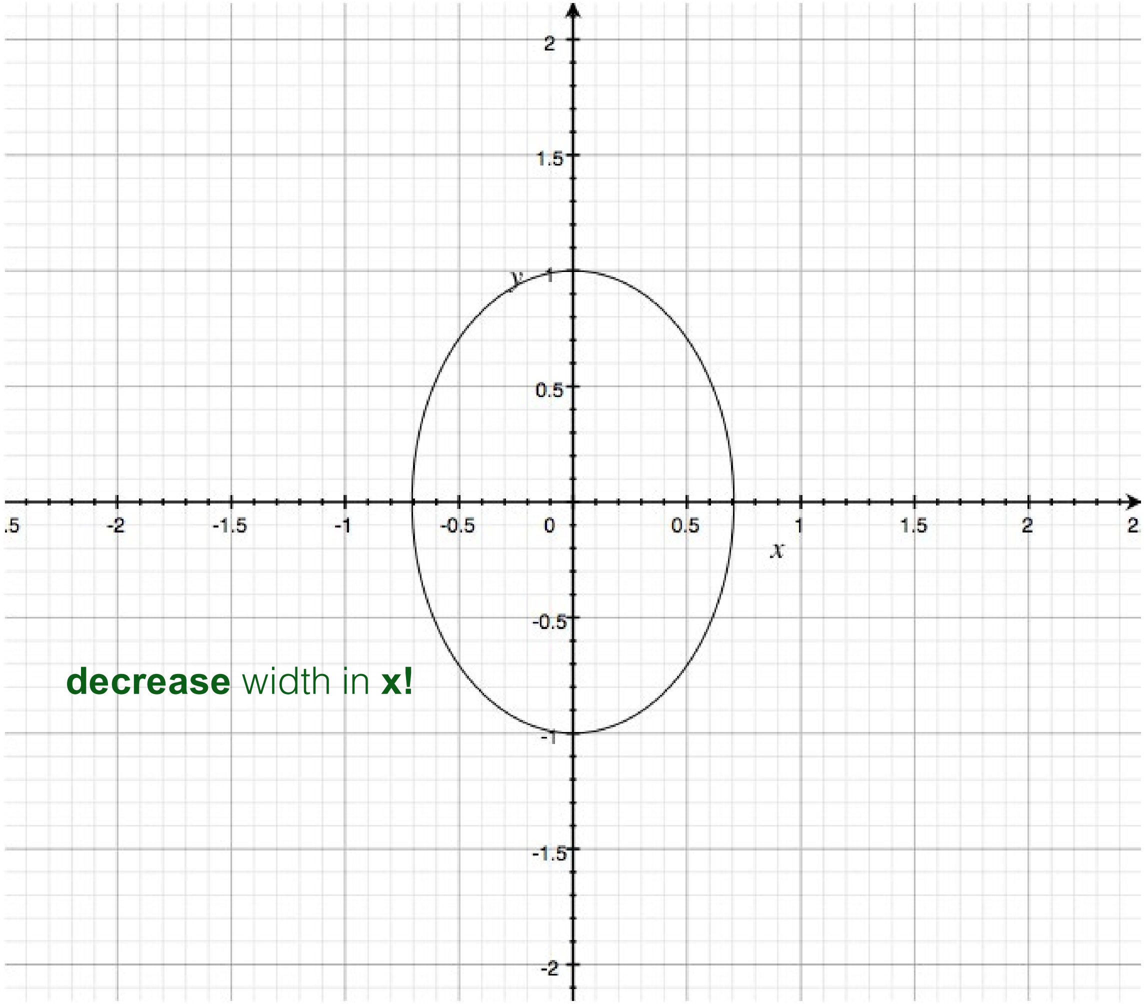
and slice at 1

decrease width in **x!**

What happens if you ***increase*** *coefficient on* ***x****?*

$$f(x,y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

and slice at 1



decrease width in **x!**

*What happens if you **increase** coefficient on **y**?*

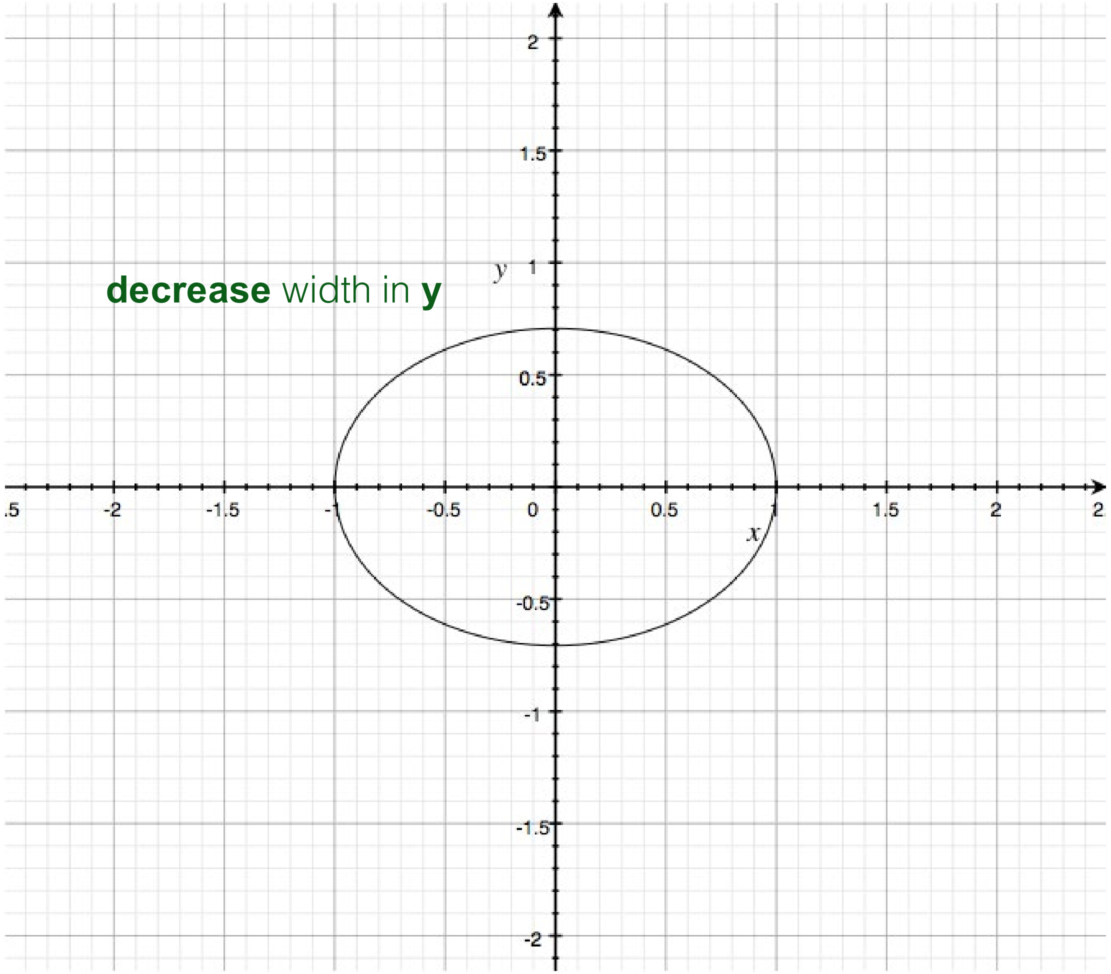
$$f(x,y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

and slice at 1

*What happens if you **increase** coefficient on **y**?*

$$f(x,y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

and slice at 1



$$f(x, y) = x^2 + y^2$$

can be written in matrix form like this…

$$f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

What's the shape?
What are the eigenvectors?
What are the eigenvalues?

$$f(x, y) = x^2 + y^2$$

can be written in matrix form like this…

$$f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Result of Singular Value Decomposition (SVD)

eigenvectors

eigenvalues along diagonal

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^\top$$

axis of the ‘ellipse slice’

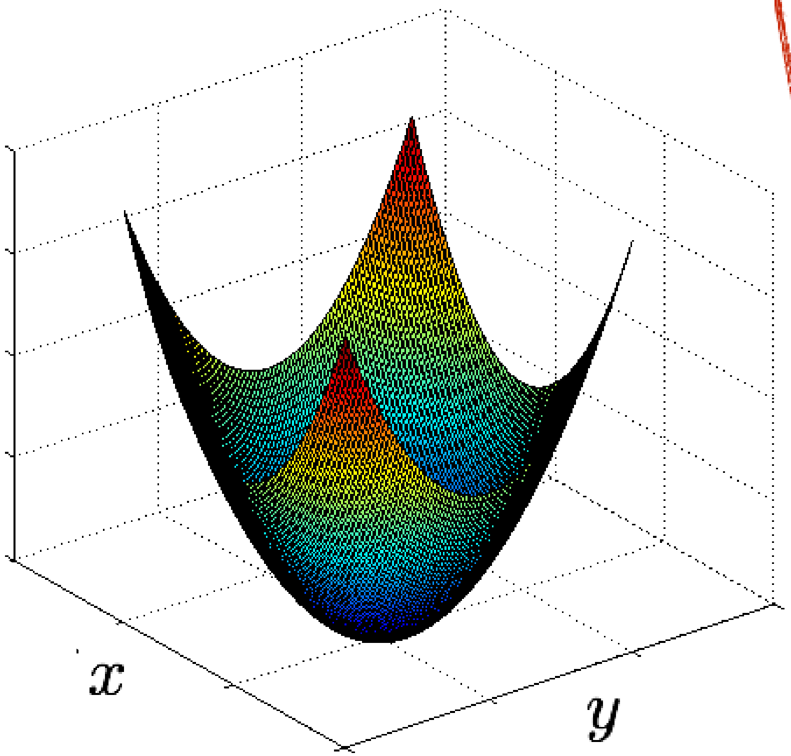
Inverse sqr of length of the quadratic along the axis

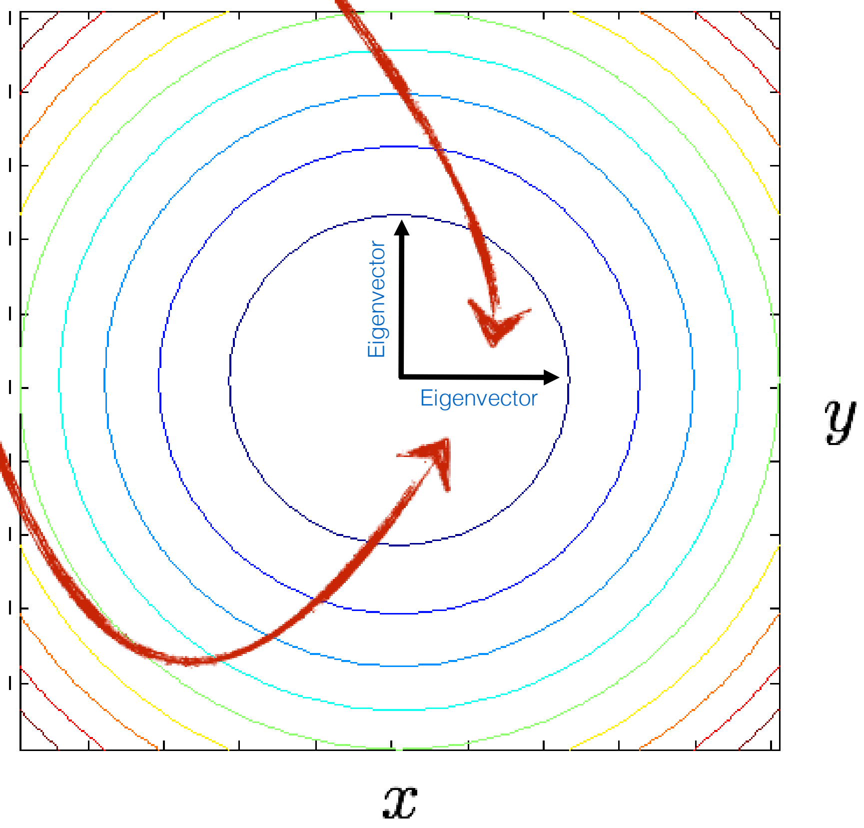
Eigenvectors Eigenvalues

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^T$$

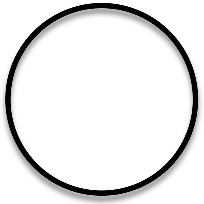
Eigenvectors

Inverse sqr of the size of the axis



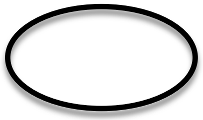


Recall:



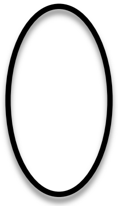
$$f(x,y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

you can smash this bowl in the **y** direction



$$f(x,y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

you can smash this bowl in the **x** direction



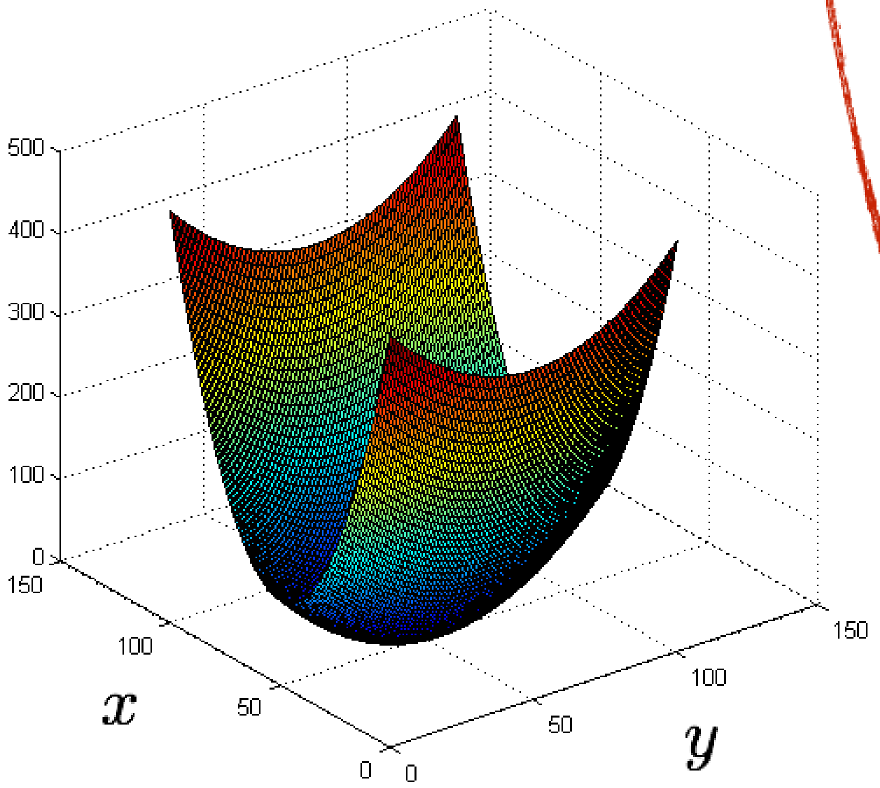
$$f(x,y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

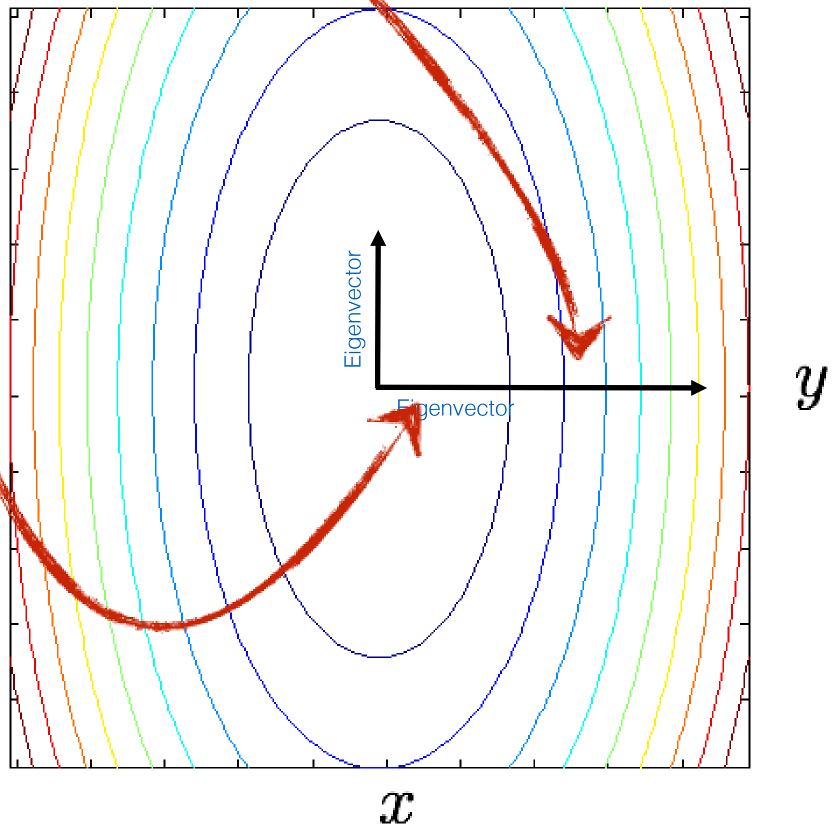
$$\mathbf{A} = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^T$$

Eigenvalues

Eigenvectors Eigenvectors

Inverse sqr of length of axis

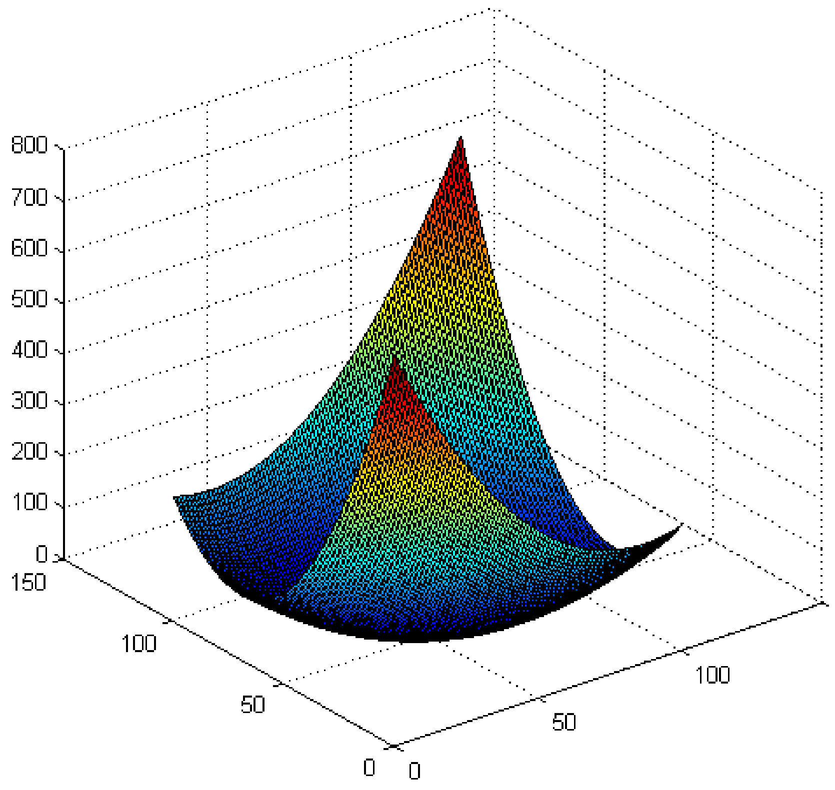


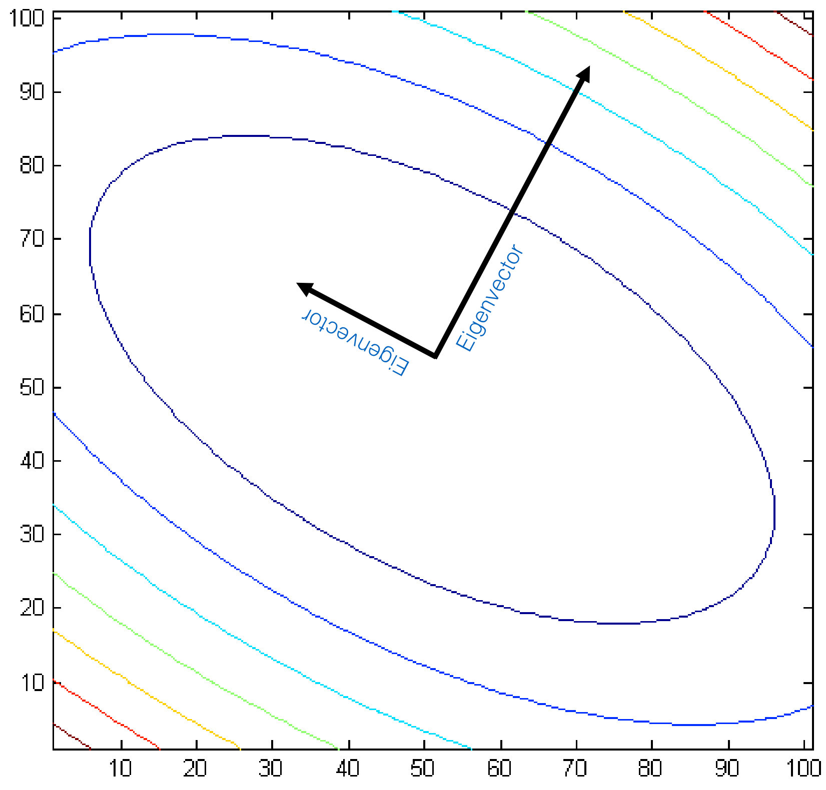


Eigenvalues

$$\mathbf{A} = \begin{bmatrix} 3.25 & 1.30 \\ 1.30 & 1.75 \end{bmatrix} = \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix}^T$$

Eigenvectors   Eigenvectors

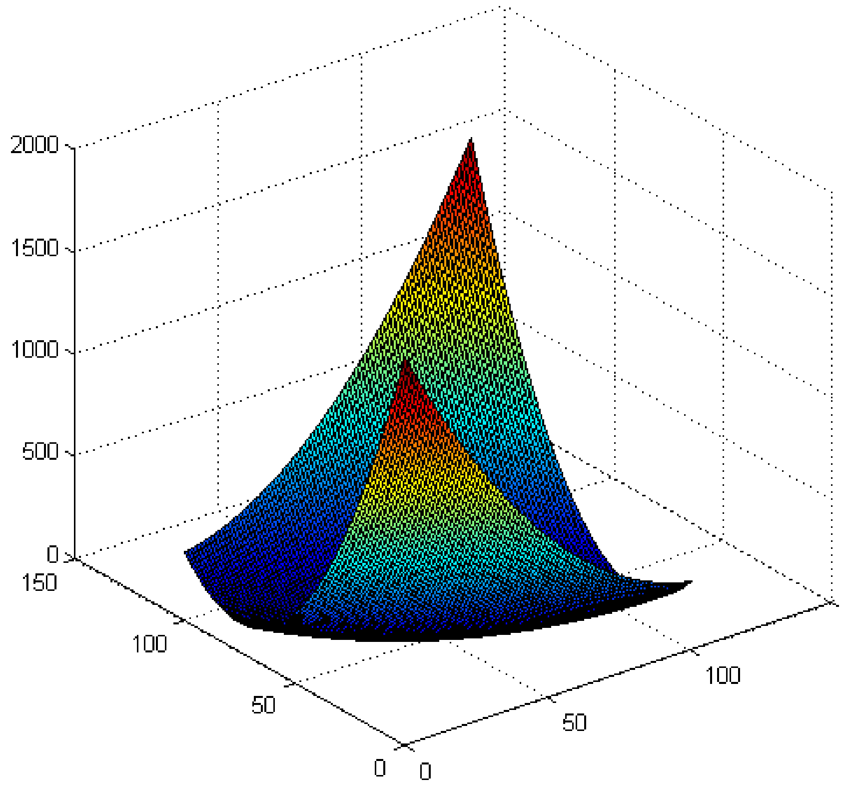


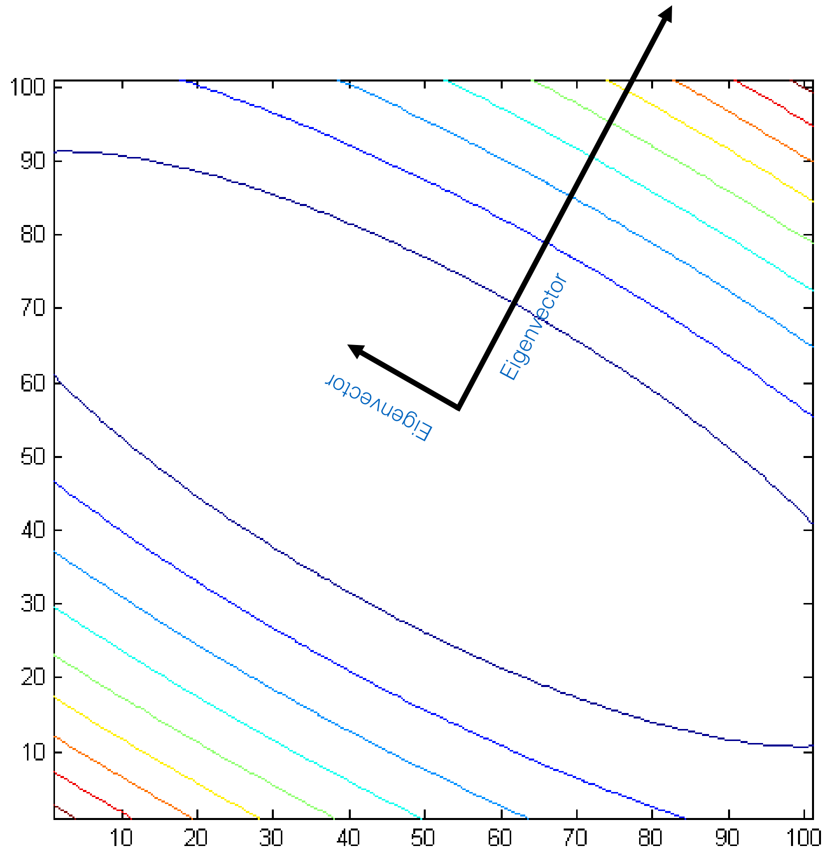


Eigenvalues

$$\mathbf{A} = \begin{bmatrix} 7.75 & 3.90 \\ 3.90 & 3.25 \end{bmatrix} = \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix} \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix}^T$$

Eigenvectors Eigenvectors



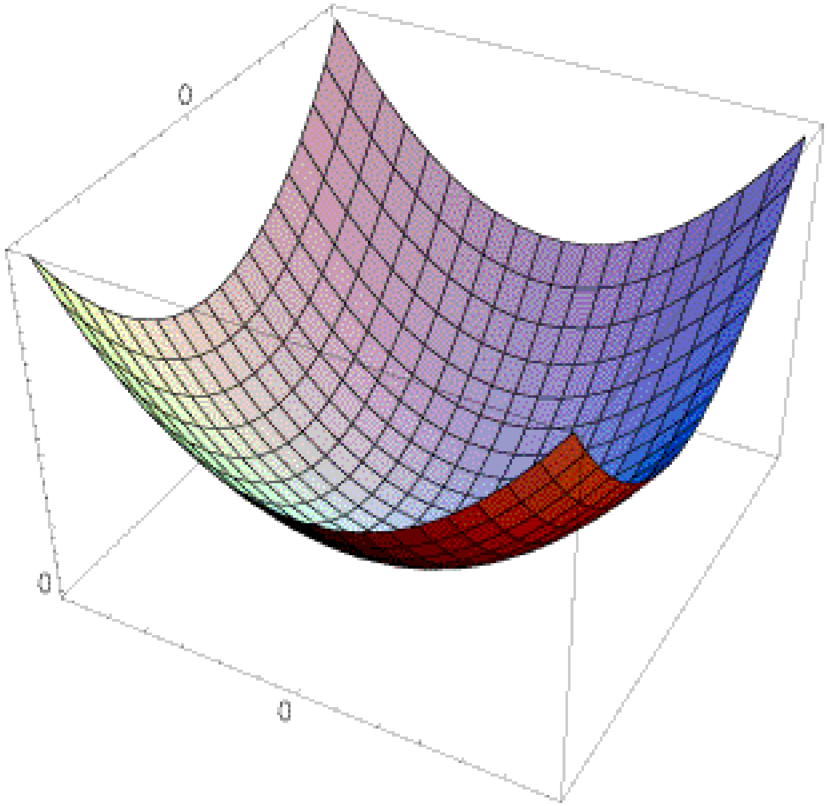


Error function for Harris Corners

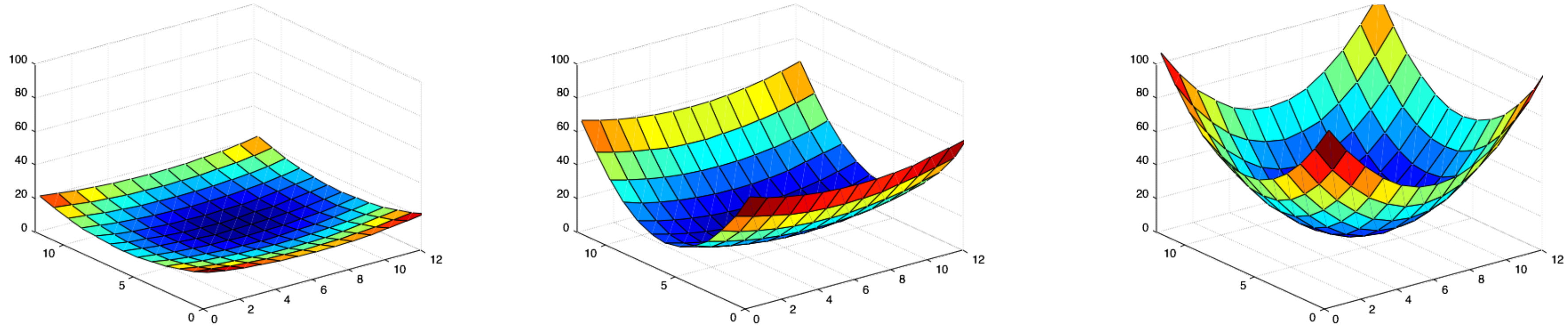
The surface *E*(*u*,*v*) is locally approximated by a quadratic form

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

$$M = \sum \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

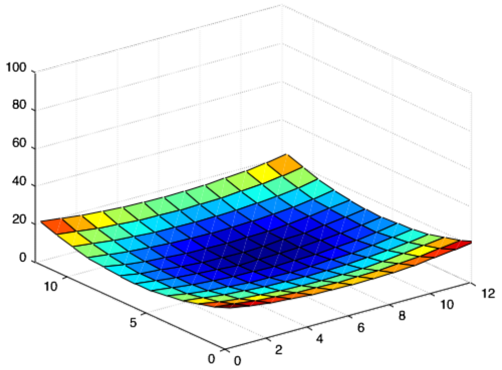


Which error surface indicates a good image feature?

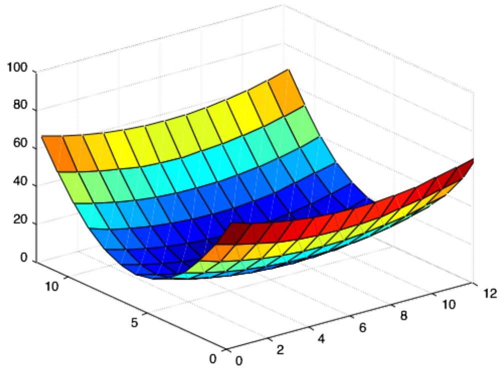


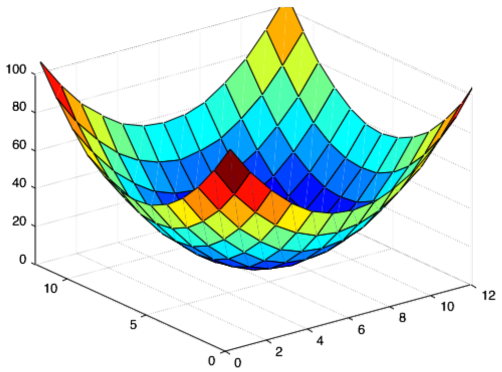
What kind of image patch do these surfaces represent?

Which error surface indicates a good image feature?

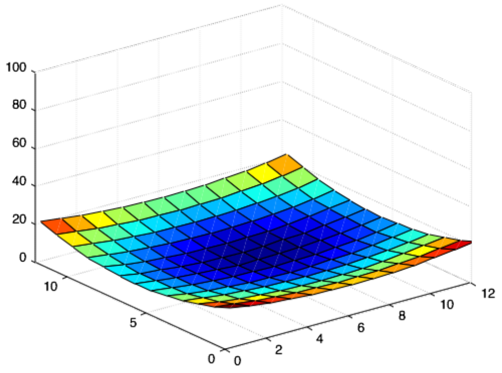


flat

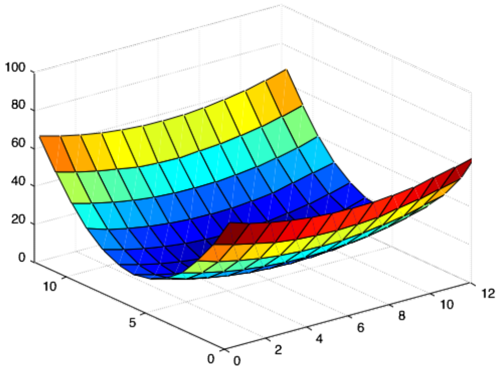




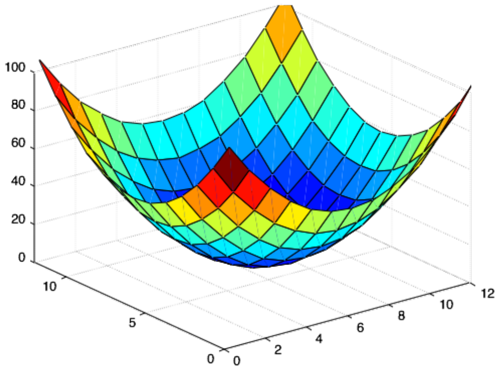
Which error surface indicates a good image feature?



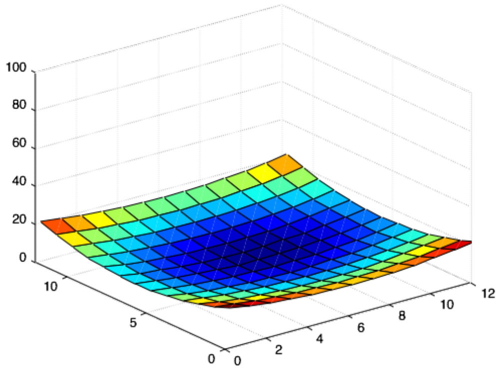
flat



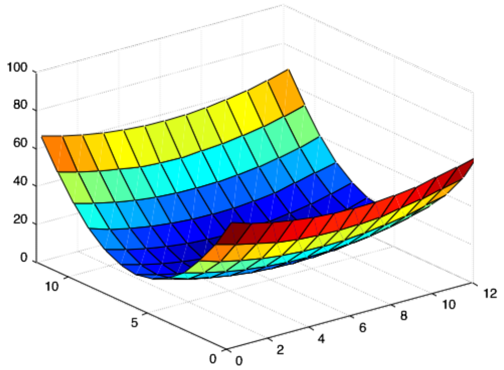
edge
'line'



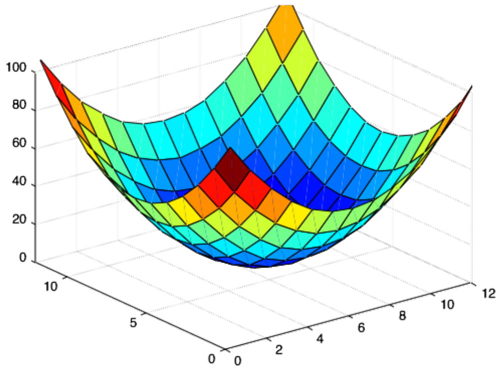
Which error surface indicates a good image feature?



flat



edge
'line'



corner
'dot'

Harris Corner Recipe

1. Compute image gradients over small region

$$I_x = \frac{\partial I}{\partial x} \qquad I_y = \frac{\partial I}{\partial y}$$

2. Subtract mean from each image gradient

3. Compute the covariance matrix

$$\begin{bmatrix} \sum\limits_{p \in P} I_x I_x & \sum\limits_{p \in P} I_x I_y \\ \sum\limits_{p \in P} I_y I_x & \sum\limits_{p \in P} I_y I_y \end{bmatrix}$$

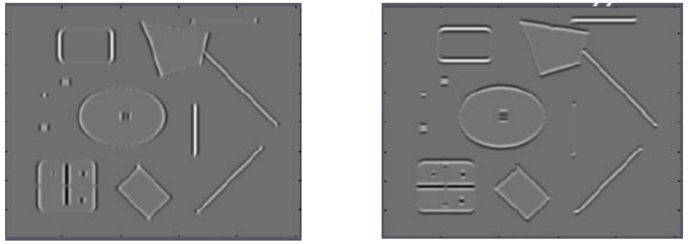
4. Compute eigenvectors and eigenvalues

5. Use threshold on eigenvalues to detect corners

Harris Corner Recipe

1. Compute image gradients over small region

$$I_x = \frac{\partial I}{\partial x} \qquad I_y = \frac{\partial I}{\partial y}$$



2. Subtract mean from each image gradient

3. Compute the covariance matrix

$$\begin{bmatrix} \sum\limits_{p \in P} I_x I_x & \sum\limits_{p \in P} I_x I_y \\ \sum\limits_{p \in P} I_y I_x & \sum\limits_{p \in P} I_y I_y \end{bmatrix}$$

4. Compute eigenvectors and eigenvalues

5. Use threshold on eigenvalues to detect corners

4. Compute eigenvalues and eigenvectors

4. Compute eigenvalues and eigenvectors

eigenvalue

$$M\boldsymbol{e} = \lambda \boldsymbol{e}$$

eigenvector

$$(M - \lambda I)\boldsymbol{e} = 0$$

4. Compute eigenvalues and eigenvectors

eigenvalue

$$M\mathbf{e} = \lambda\mathbf{e}$$

eigenvector

$$(M - \lambda I)\mathbf{e} = 0$$

1. Compute the determinant of $M - \lambda I$

(returns a polynomial)

4. Compute eigenvalues and eigenvectors

eigenvalue

$$Me = \lambda e$$

eigenvector

$$(M - \lambda I)e = 0$$

1. Compute the determinant of $M - \lambda I$
(returns a polynomial)

2. Find the roots of polynomial $\det(M - \lambda I) = 0$
(returns eigenvalues)

4. Compute eigenvalues and eigenvectors

eigenvalue

$$Me = \lambda e$$

eigenvector

$$(M - \lambda I)e = 0$$

1. Compute the determinant of $M - \lambda I$
(returns a polynomial)

2. Find the roots of polynomial $\det(M - \lambda I) = 0$
(returns eigenvalues)

3. For each eigenvalue, solve $(M - \lambda I)e = 0$
(returns eigenvectors)

Harris Corner Recipe

1. Compute image gradients over small region

2. Subtract mean from each image gradient

3. Compute the covariance matrix

4. Compute eigenvectors and eigenvalues

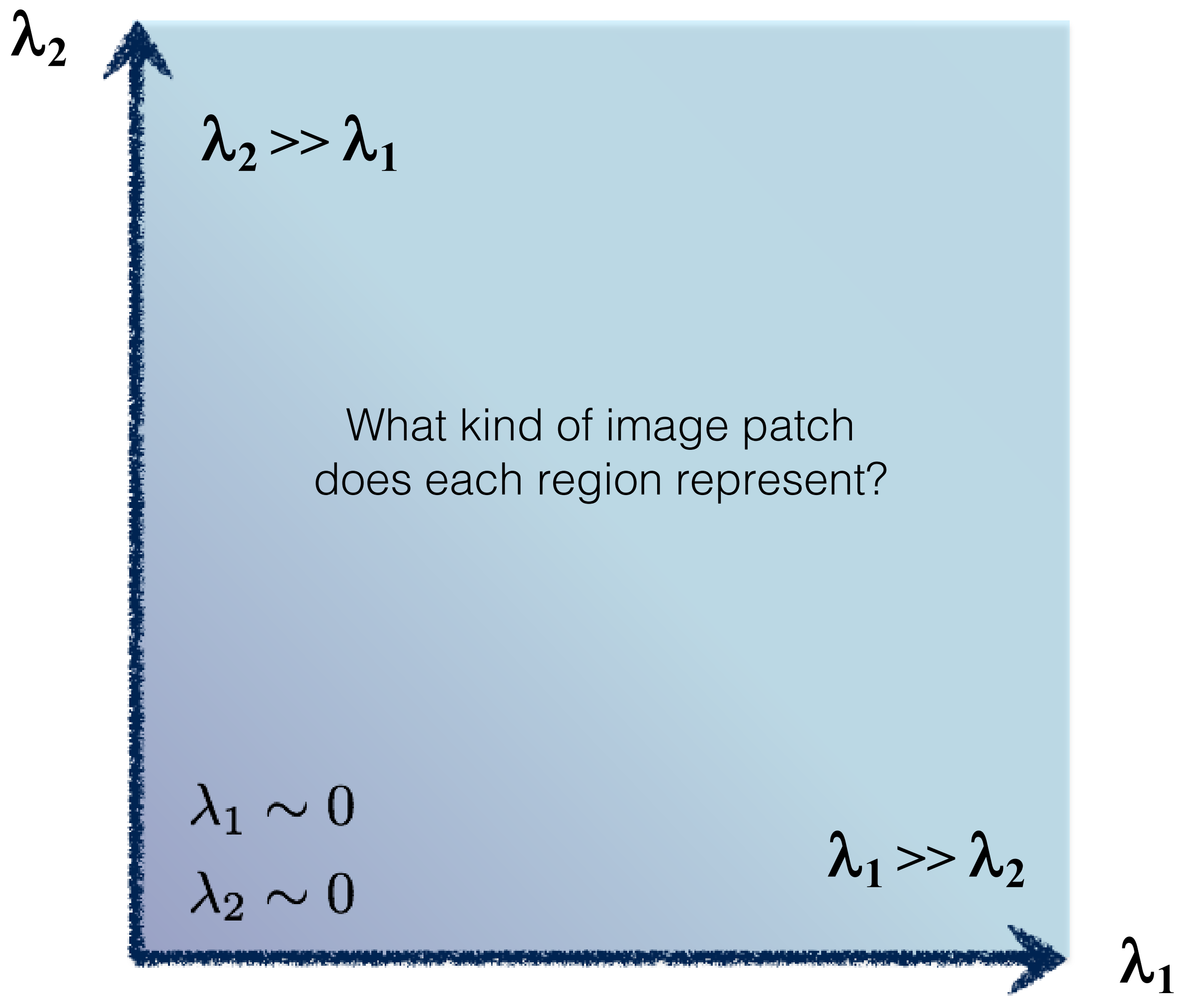
5. Use threshold on eigenvalues to detect corners

$I_x = \frac{\partial I}{\partial x}$

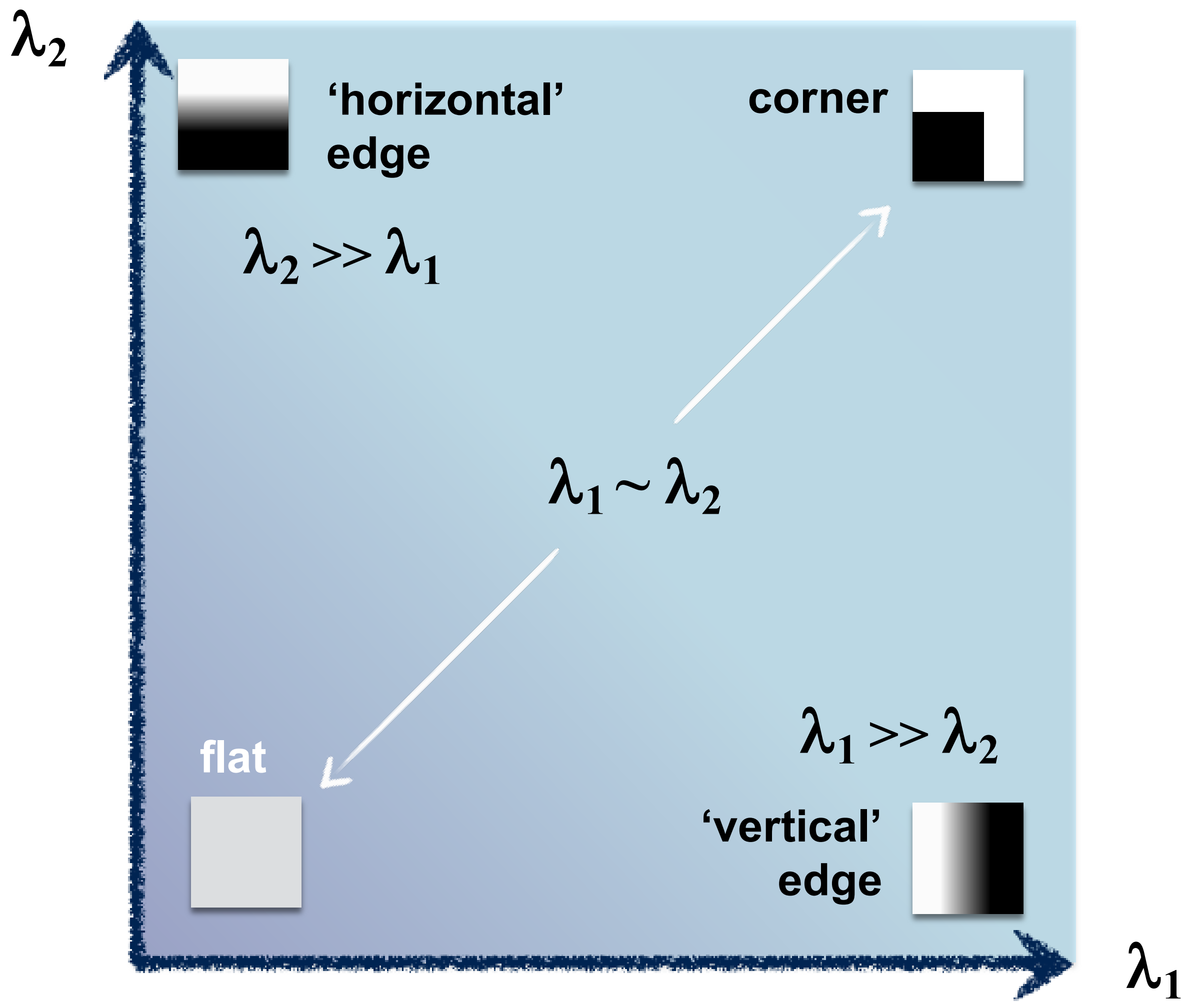
$I_y = \frac{\partial I}{\partial y}$

$$\begin{bmatrix} \sum_{p \in P} I_x I_x & \sum_{p \in P} I_x I_y \\ \sum_{p \in P} I_y I_x & \sum_{p \in P} I_y I_y \end{bmatrix}$$

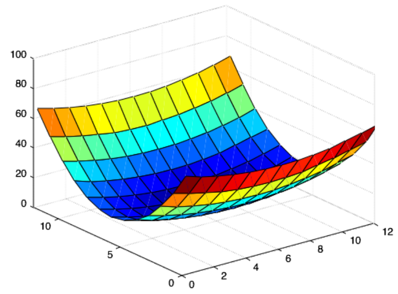
interpreting eigenvalues

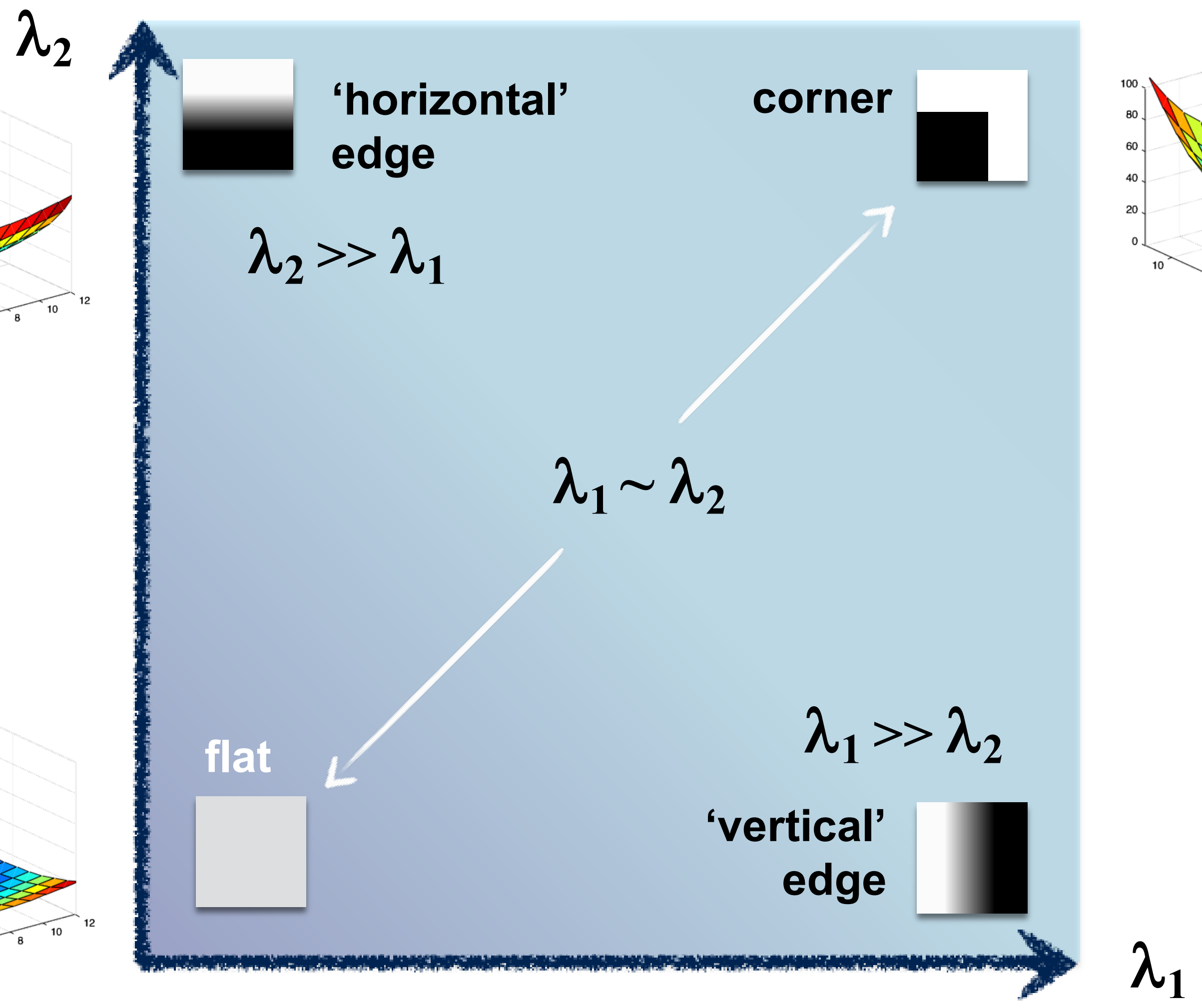


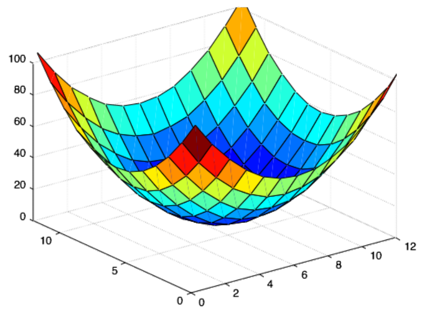
interpreting eigenvalues

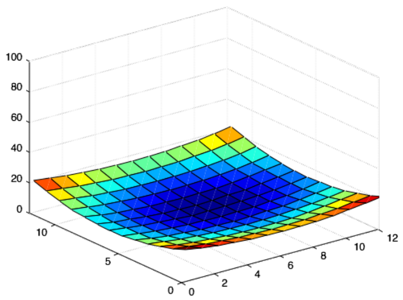


interpreting eigenvalues









Harris Corner Recipe

1. Compute image gradients over small region

$$I_x = \frac{\partial I}{\partial x} \qquad I_y = \frac{\partial I}{\partial y}$$

2. Subtract mean from each image gradient

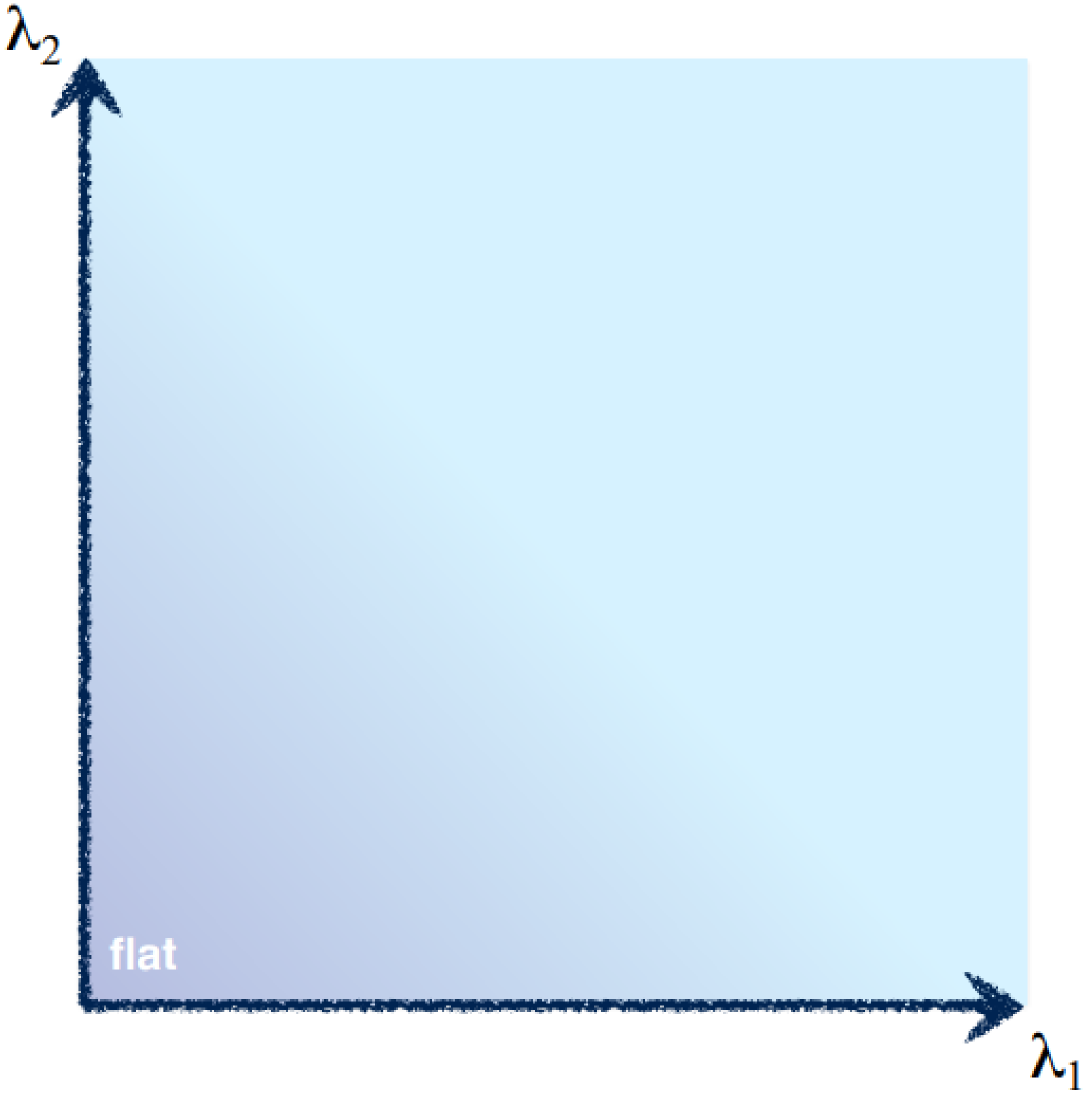
3. Compute the covariance matrix

$$\begin{bmatrix} \sum\limits_{p \in P} I_x I_x & \sum\limits_{p \in P} I_x I_y \\ \sum\limits_{p \in P} I_y I_x & \sum\limits_{p \in P} I_y I_y \end{bmatrix}$$

4. Compute eigenvectors and eigenvalues

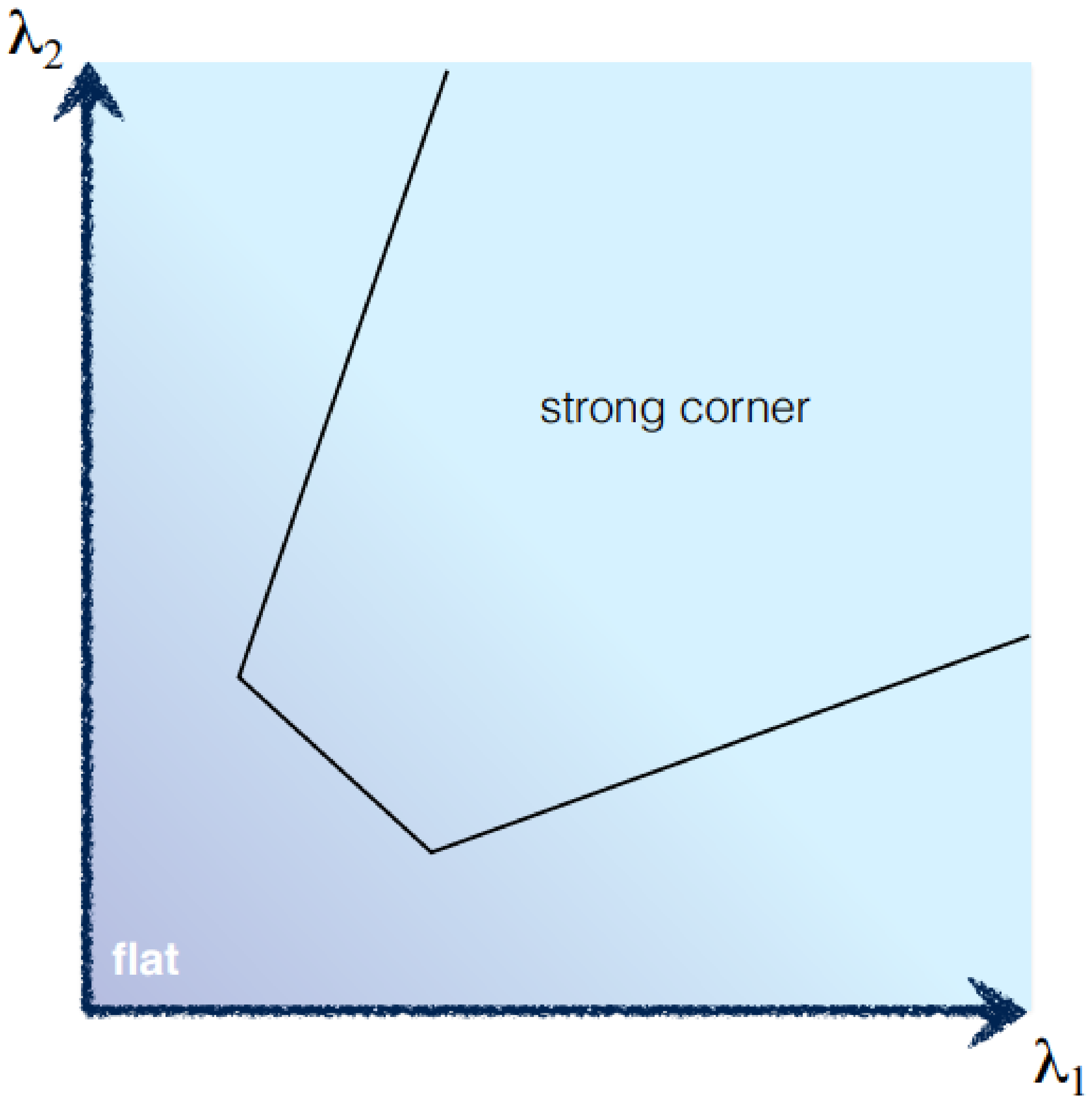
5. Use threshold on eigenvalues to detect corners

5. Use threshold on eigenvalues to detect corners



Think of a function to score 'cornerness'

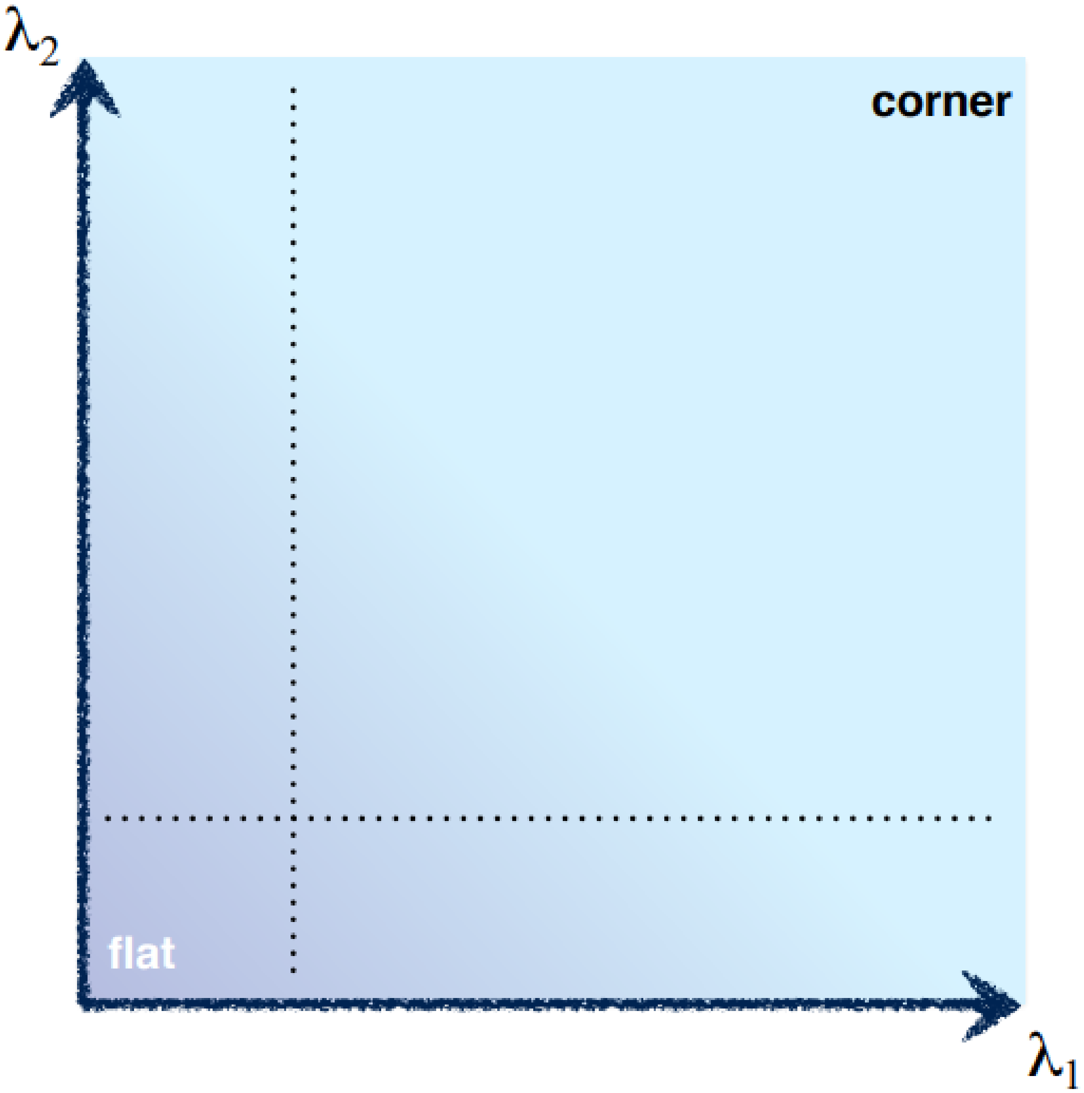
5. Use threshold on eigenvalues to detect corners



Think of a function to score ‘cornerness’

5. Use threshold on eigenvalues to detect corners

(a function of)

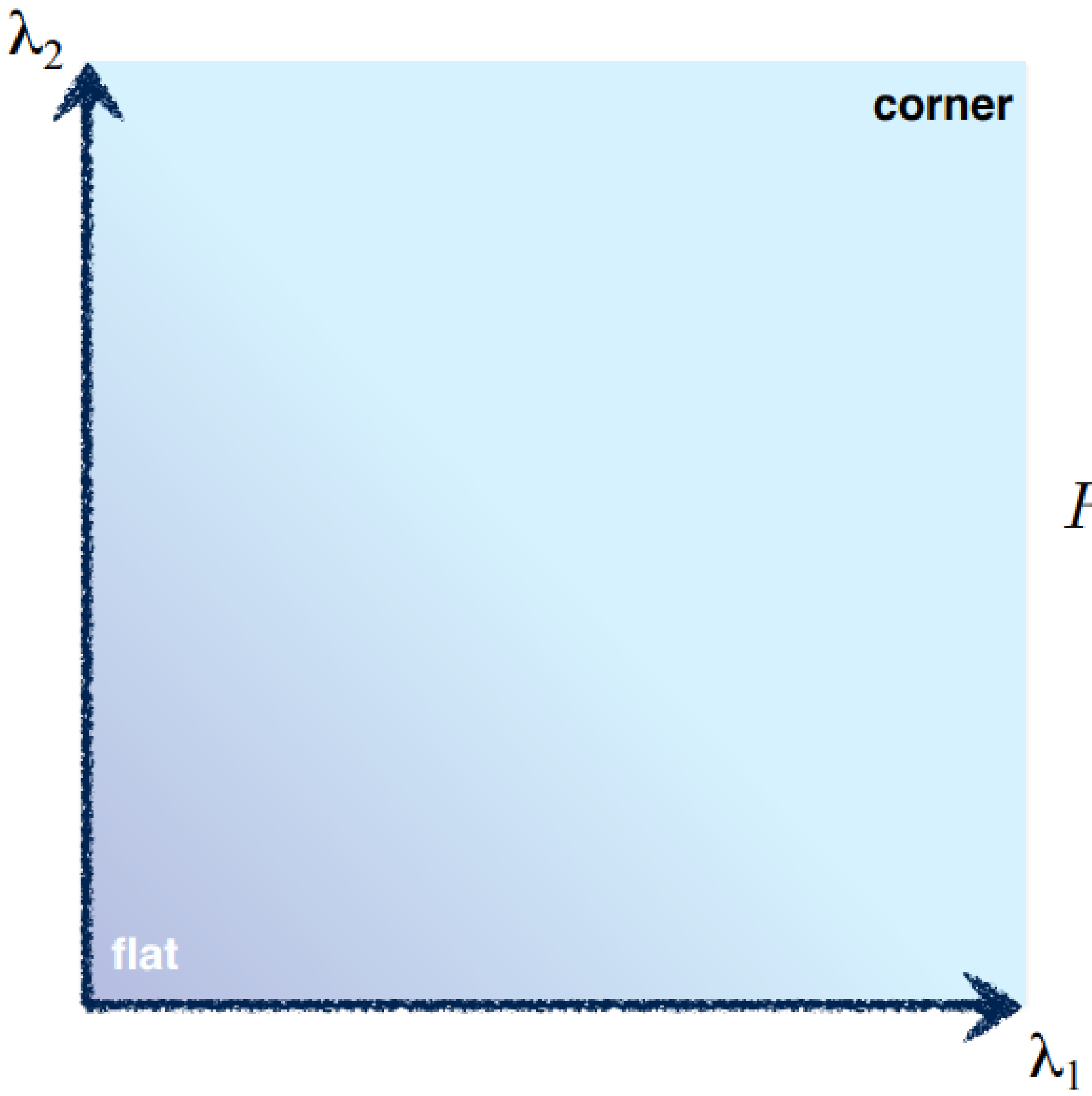


Use the smallest eigenvalue
as the response function

$$R = \min(\lambda_1, \lambda_2)$$

5. Use threshold on eigenvalues to detect corners

(a function of)



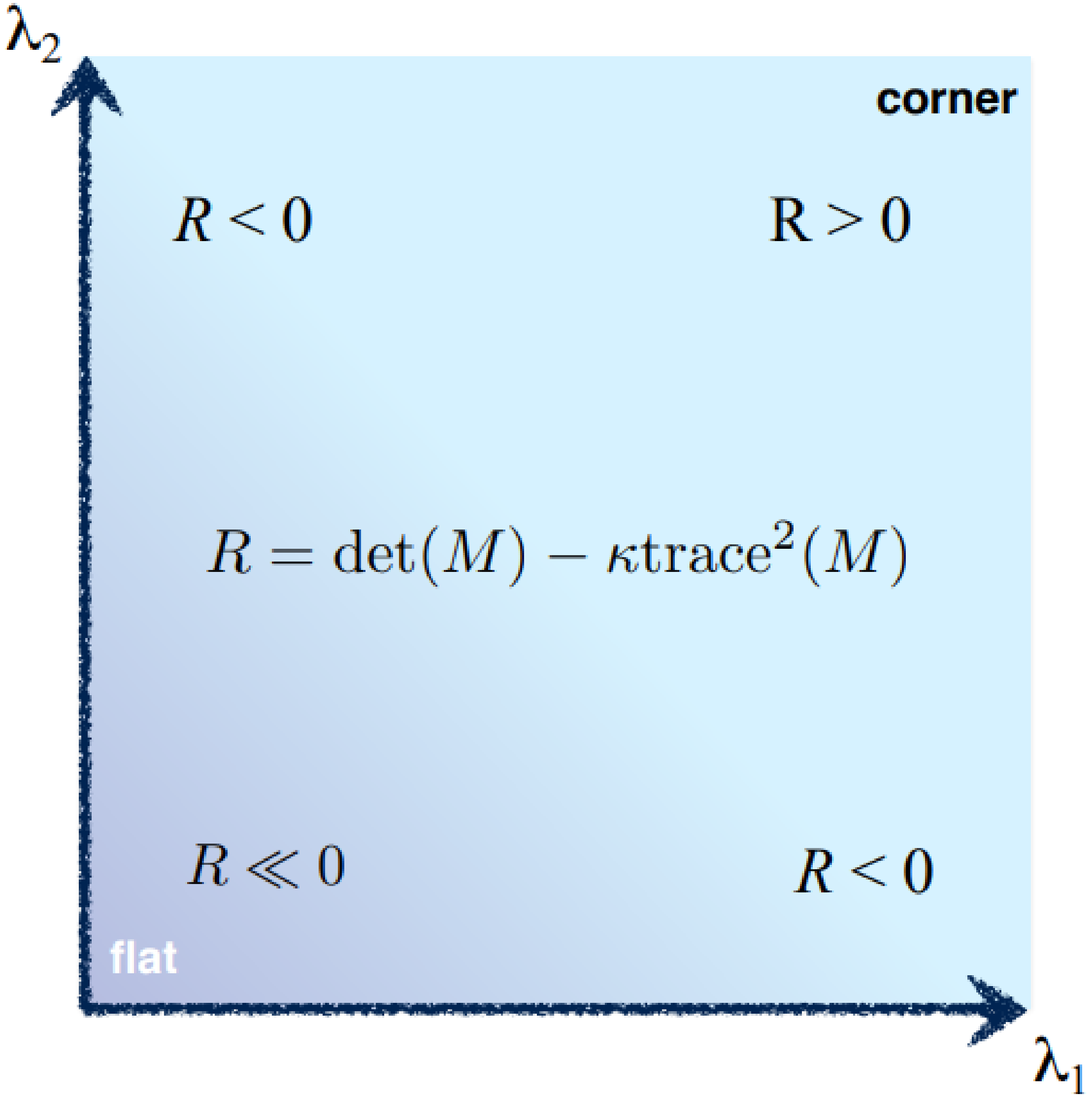
Eigenvalues need to be bigger than one.

$$R = \lambda_1 \lambda_2 - \kappa(\lambda_1 + \lambda_2)^2$$

Can compute this more efficiently…

5. Use threshold on eigenvalues to detect corners

(a function of)



$$\det M = \lambda_1 \lambda_2$$

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

$$det\left(\begin{bmatrix} a & b \\ c & d \end{bmatrix}\right) = ad - bc$$

$$trace\left(\begin{bmatrix} a & b \\ c & d \end{bmatrix}\right) = a + d$$

Harris & Stephens (1988)

$$R = \det(M) - \kappa \operatorname{trace}^2(M)$$

Kanade & Tomasi (1994)

$$R = \min(\lambda_1, \lambda_2)$$

Nobel (1998)

$$R = \frac{\det(M)}{\operatorname{trace}(M) + \epsilon}$$

Harris Corner Recipe

1. Compute image gradients over small region

$$I_x = \frac{\partial I}{\partial x}$$

$$I_y = \frac{\partial I}{\partial y}$$

2. Subtract mean from each image gradient

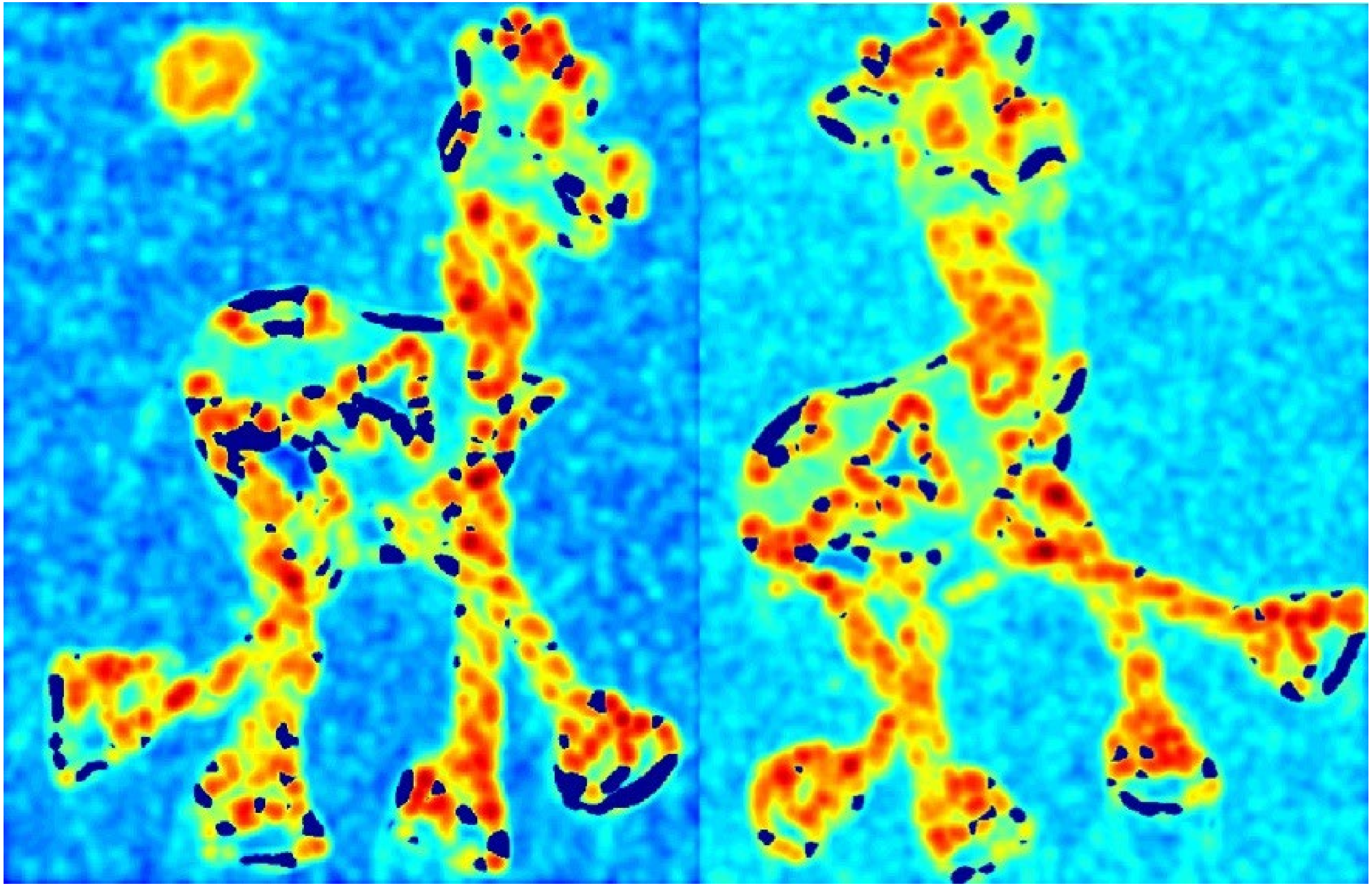
3. Compute the covariance matrix

$$\begin{bmatrix} \sum_{p \in P} I_x I_x & \sum_{p \in P} I_x I_y \\ \sum_{p \in P} I_y I_x & \sum_{p \in P} I_y I_y \end{bmatrix}$$

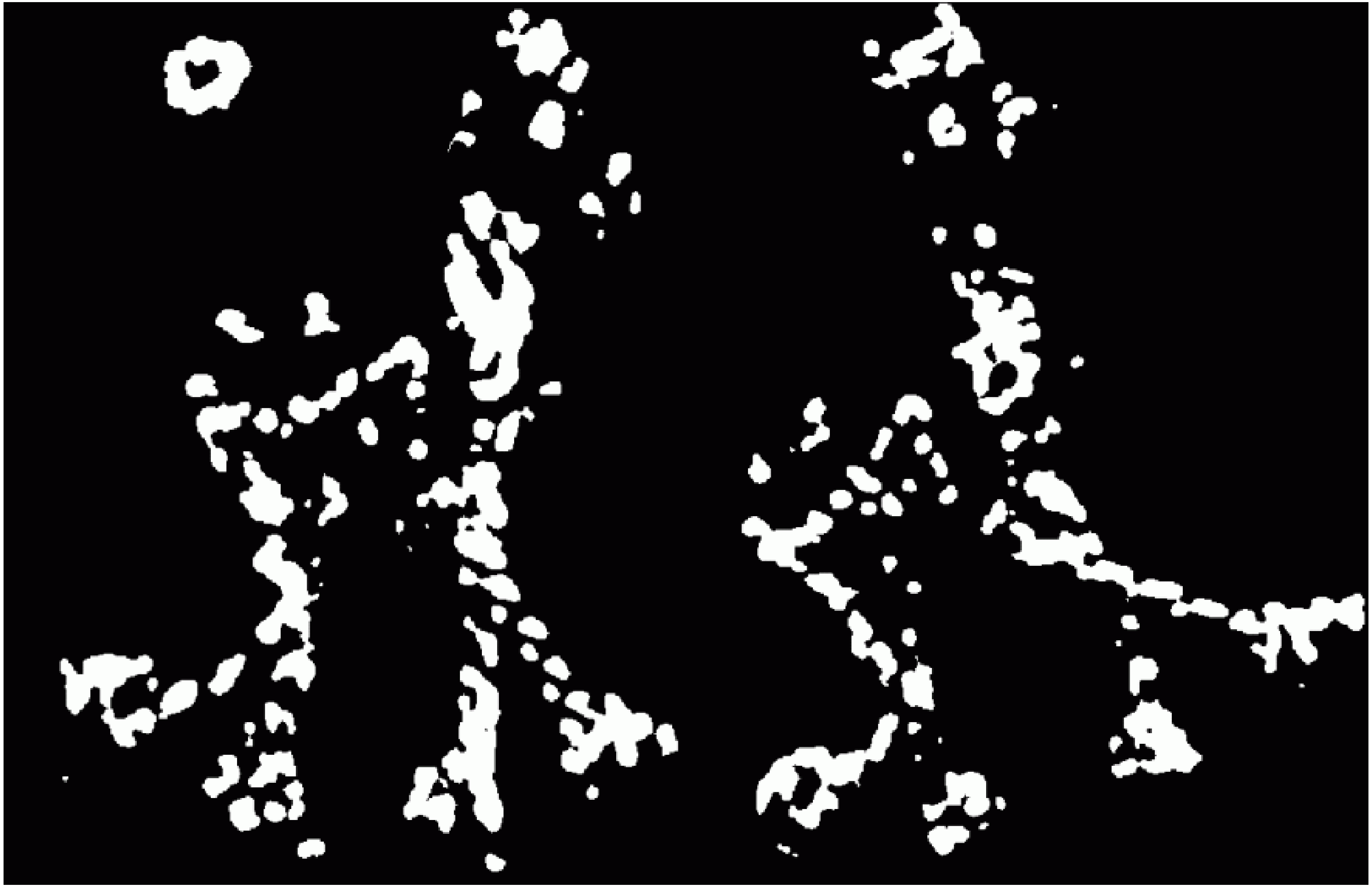
4. Compute eigenvectors and eigenvalues

5. Use threshold on eigenvalues to detect corners

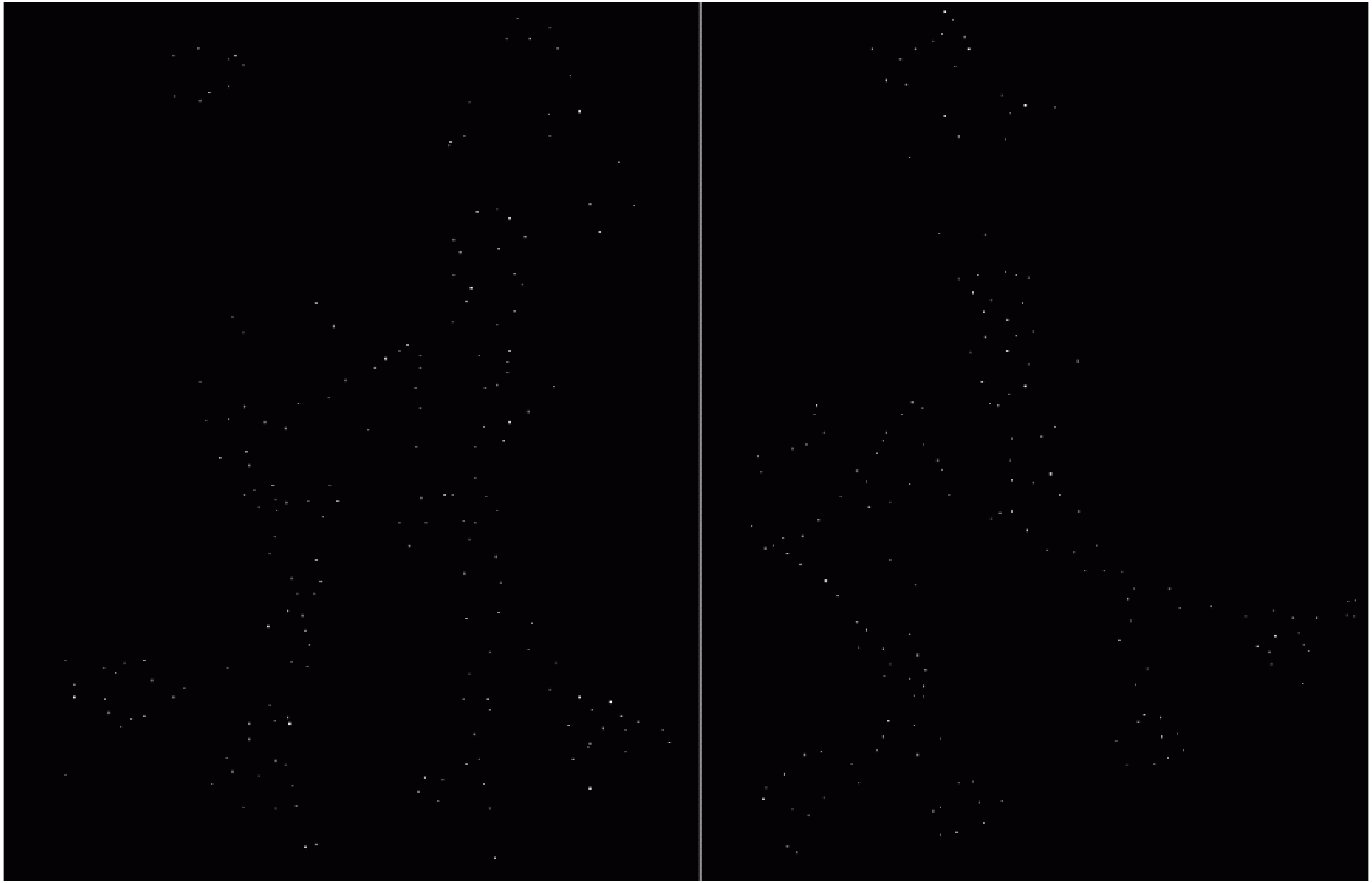
Corner response

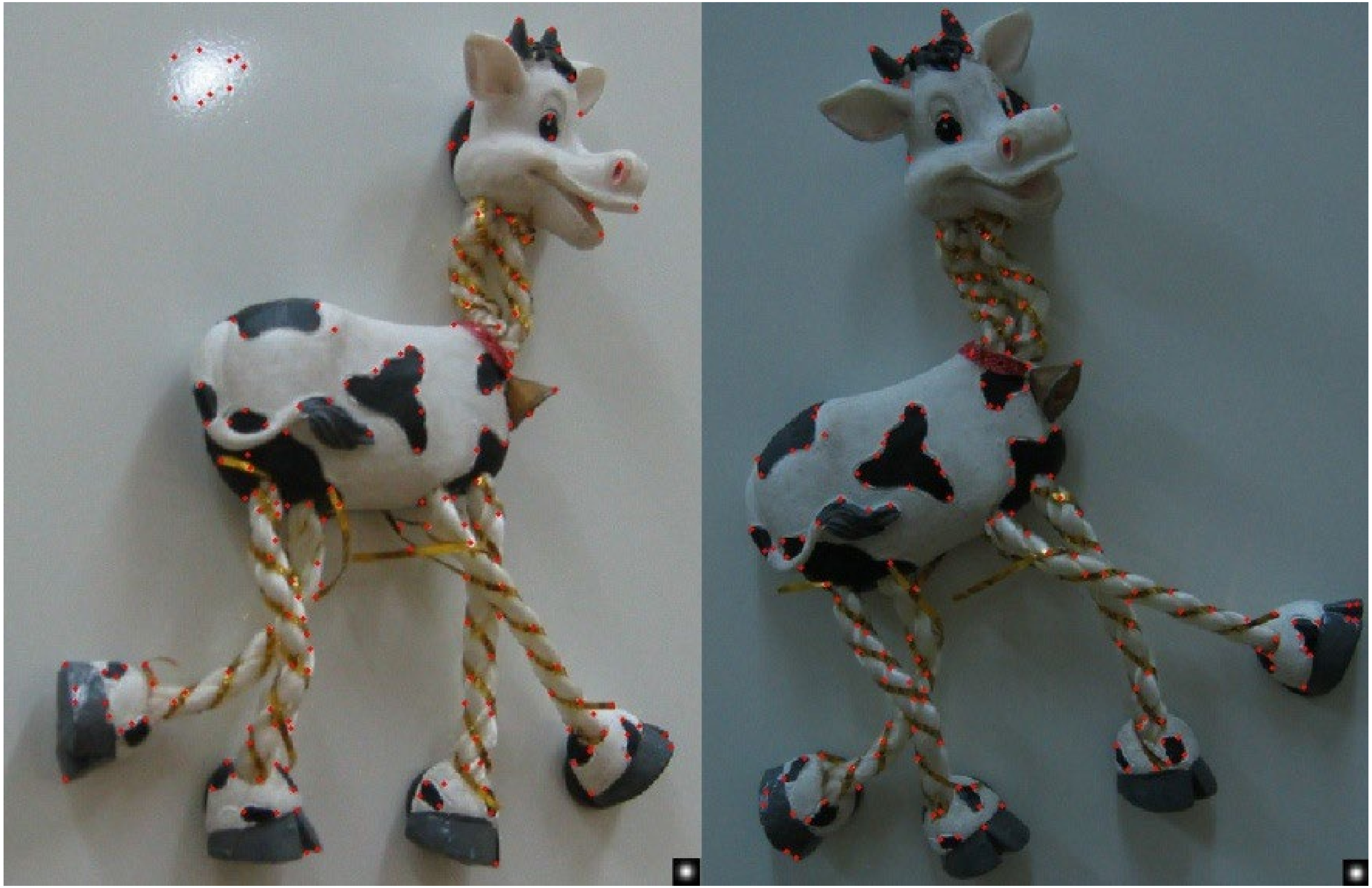


Thresholded corner response



Non-maximal suppression





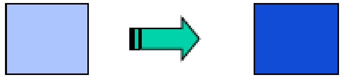
Properties of Harris Corners

Invariance and covariance

- We want corner locations to be *invariant* to photometric transformations and *covariant* to geometric transformations
  - **Invariance:** image is transformed and corner locations do not change
  - **Covariance:** if we have two transformed versions of the same image, features should be detected in corresponding locations

Intensity Invariance

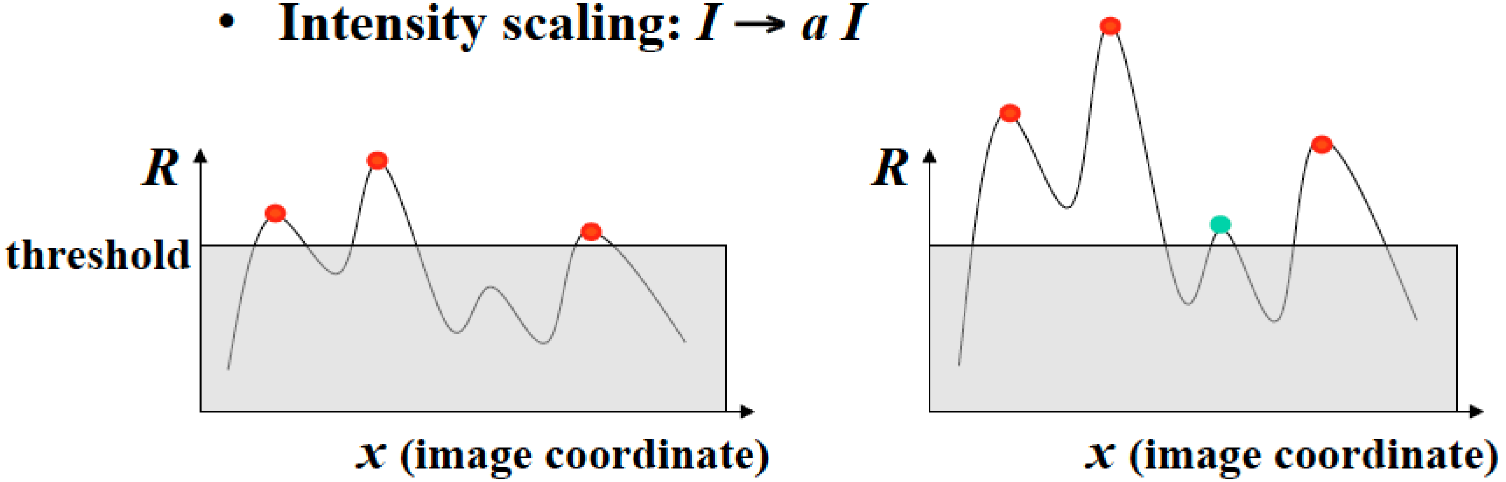
Affine intensity change



$I \rightarrow a\,I + b$

- **Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$**

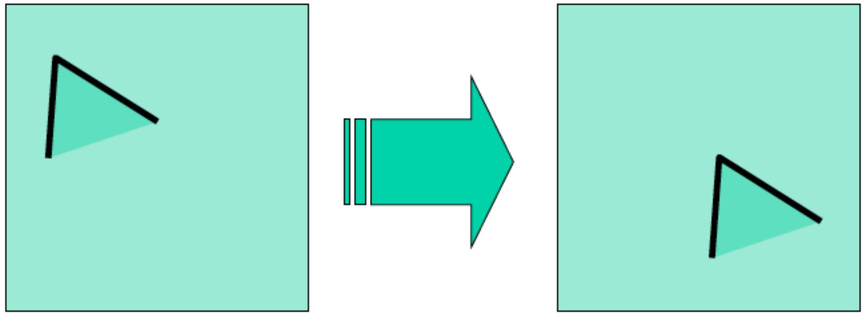
- **Intensity scaling: $I \rightarrow a\,I$**



***Partially invariant* to affine intensity change**

Translation/Rotation Covariance

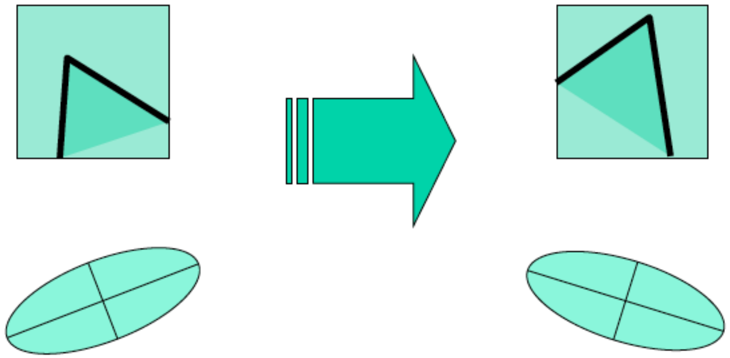
Image translation



- **Derivatives and window function are shift-invariant**

Corner location is covariant w.r.t. translation

Image rotation

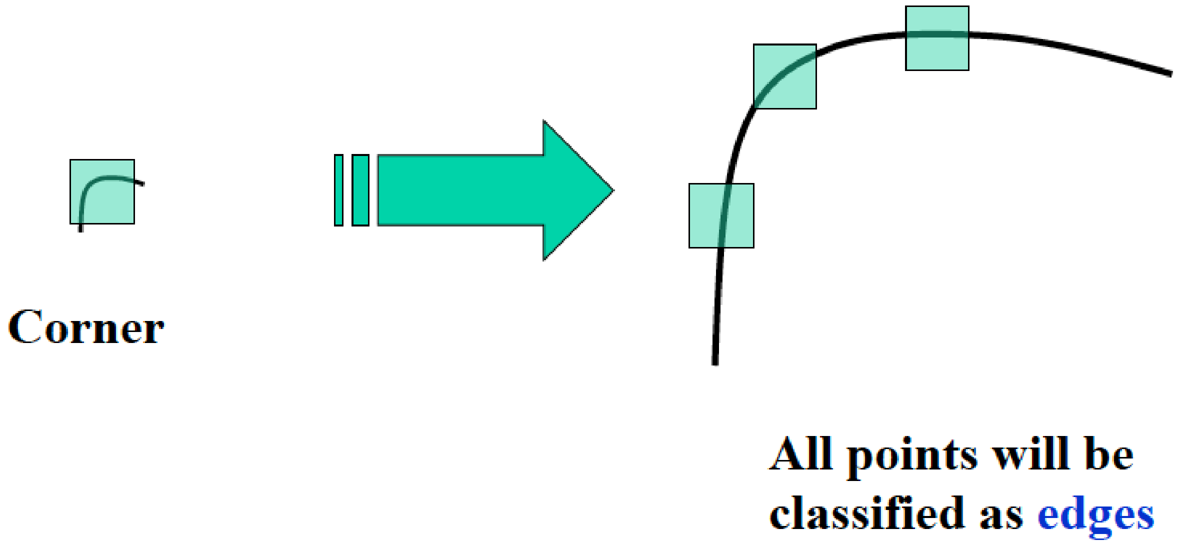


Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner location is covariant w.r.t. rotation

Scaling

Scaling



Corner location is not covariant to scaling!

How do we handle scale?

After feature detection, how do we match features in multiple images (feature description and matching)

How do we handle scale?

After feature detection, how do we match features in multiple images (feature description and matching)

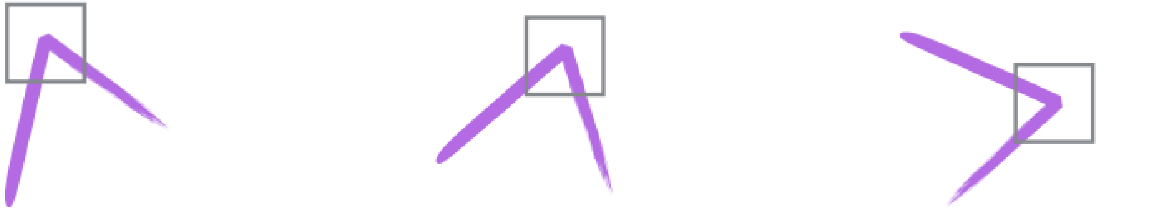
Harris Corner Detector

Rotation invariant?

Scale invariant?

Harris Corner Detector

Rotation invariant?



Scale invariant?

Two Questions

1. How can we make a feature detector ***scale invariant ?***

2. How can we ***automatically select the scale ?***

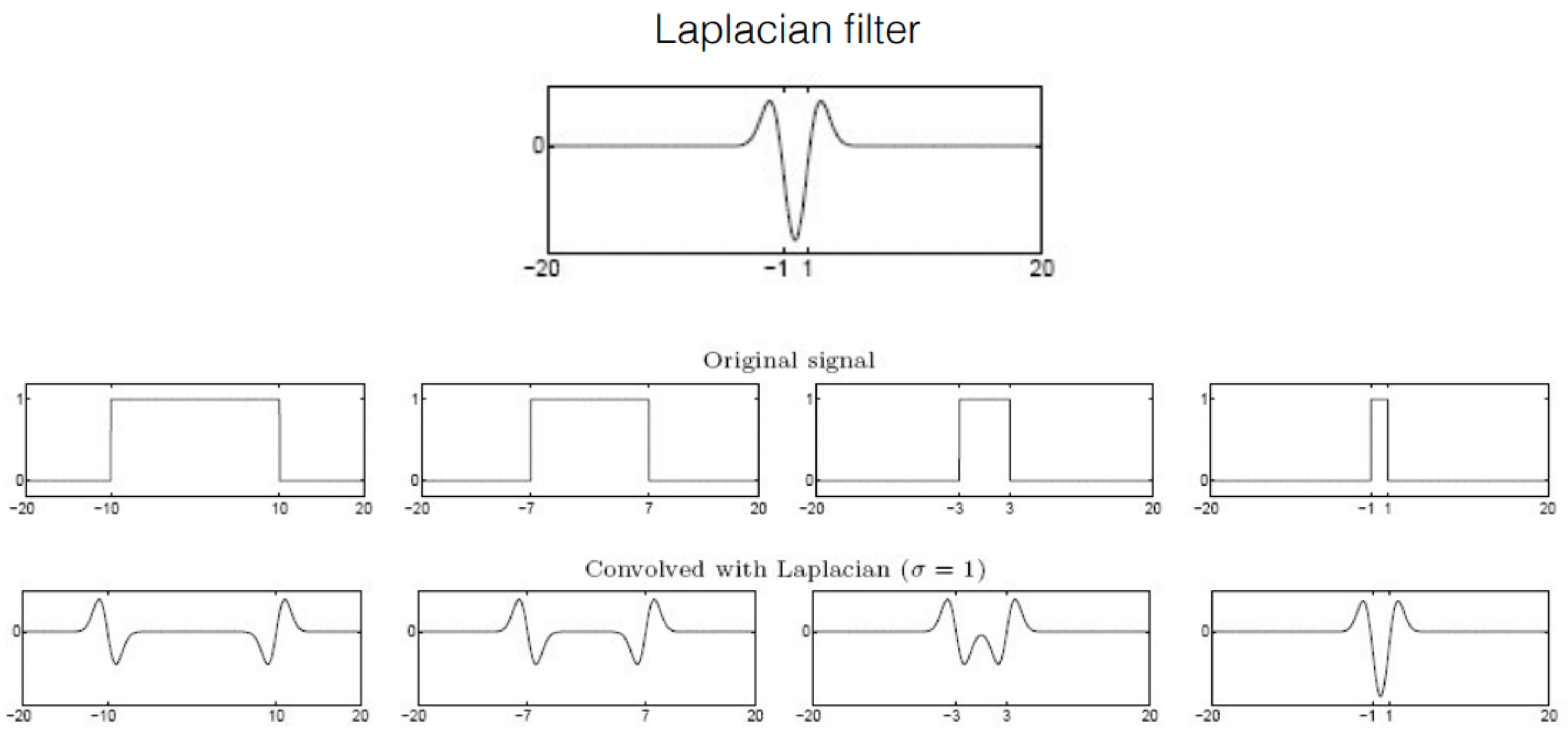
Multi-Scale Methods

1. Multi-Scale Detection

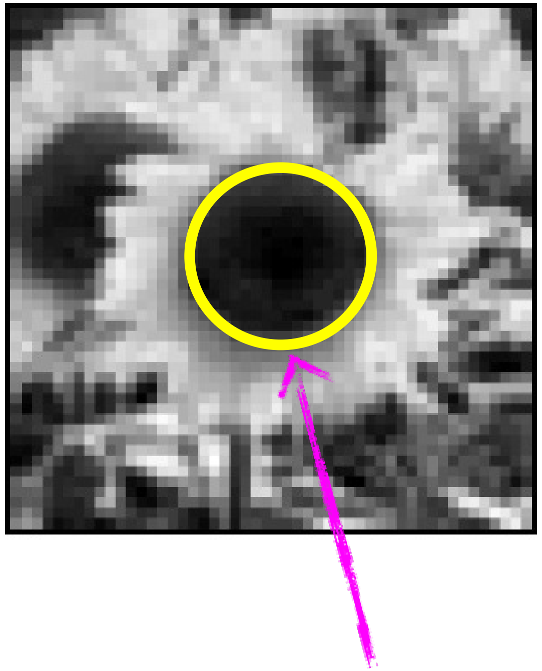
2. Scale-Space Normalization

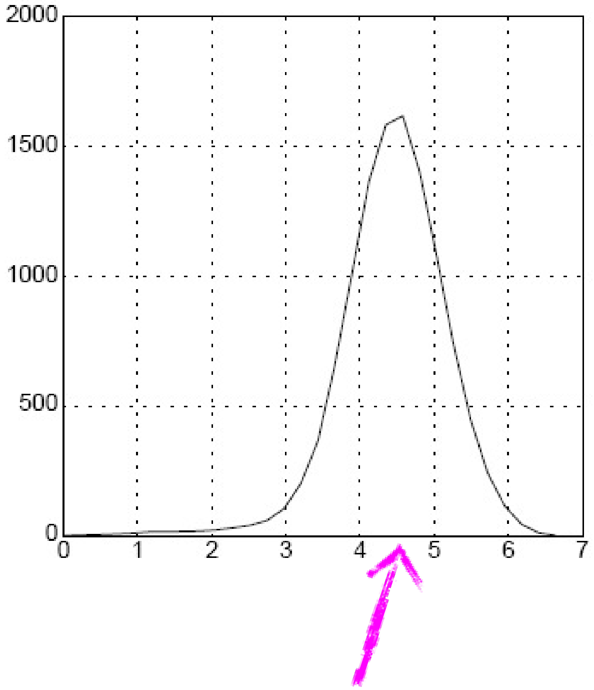
Multi-Scale 2D Blob Detector

Laplacian Filter !!!



Highest response when the signal has the same **characteristic scale** as the filter characteristic scale - the scale that produces peak filter response

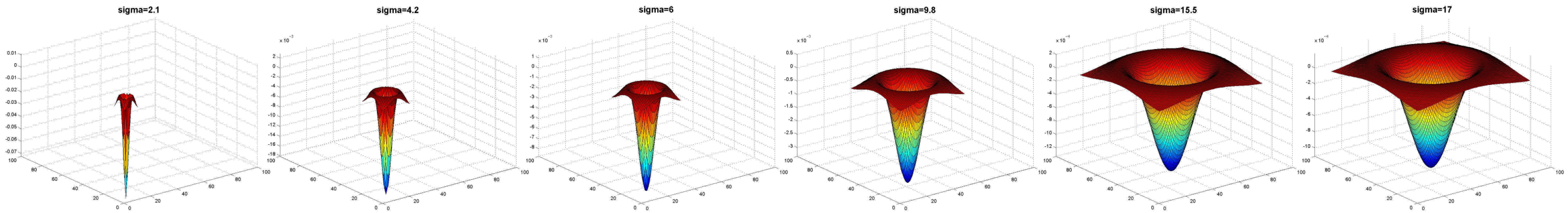




characteristic scale

we need to search over characteristic scales

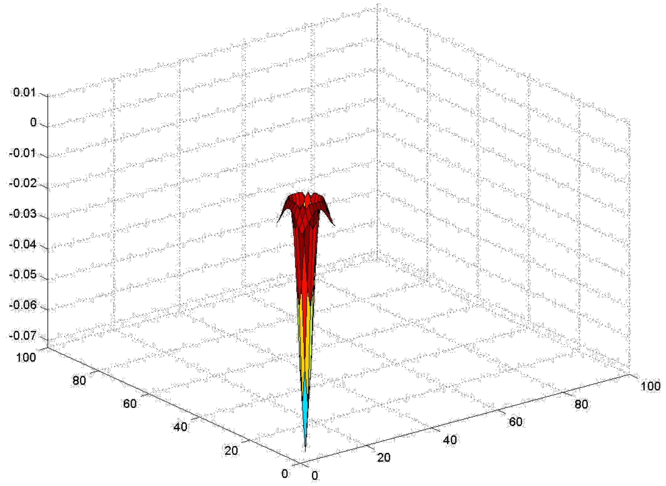
What happens if you apply different Laplacian filters?

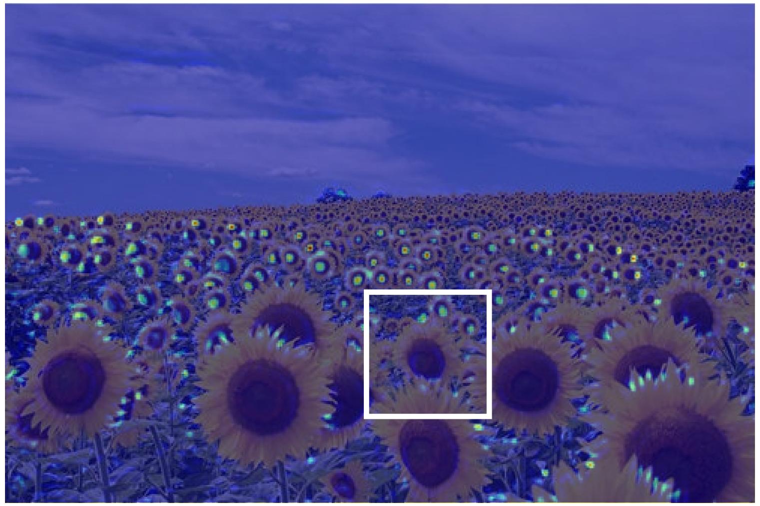


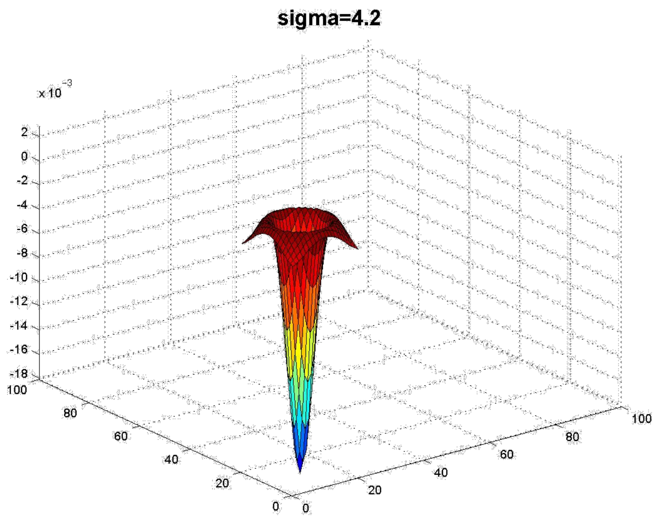
Full size

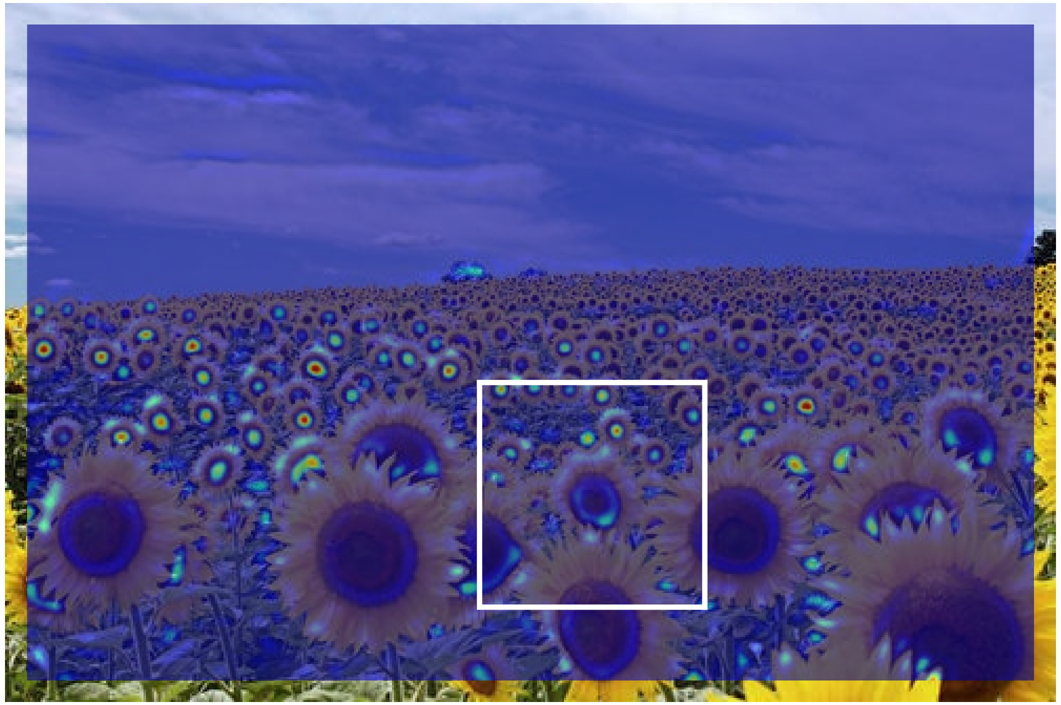
3/4 size

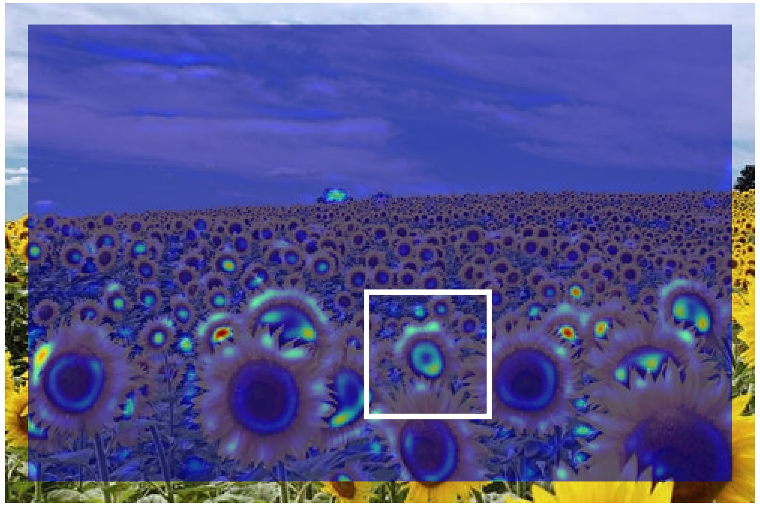
sigma=2.1

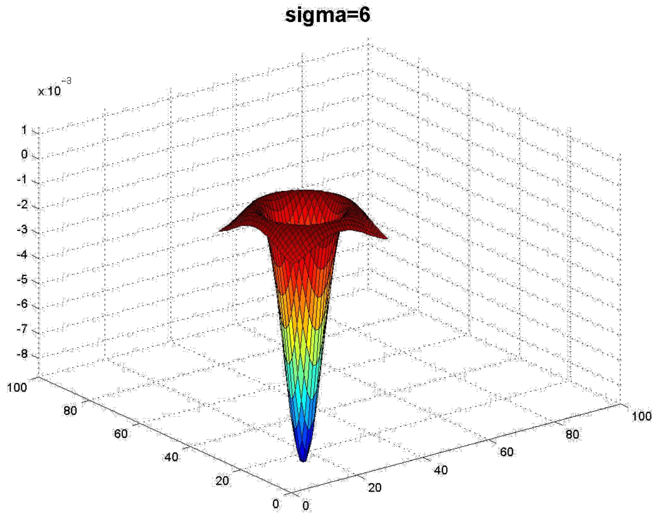


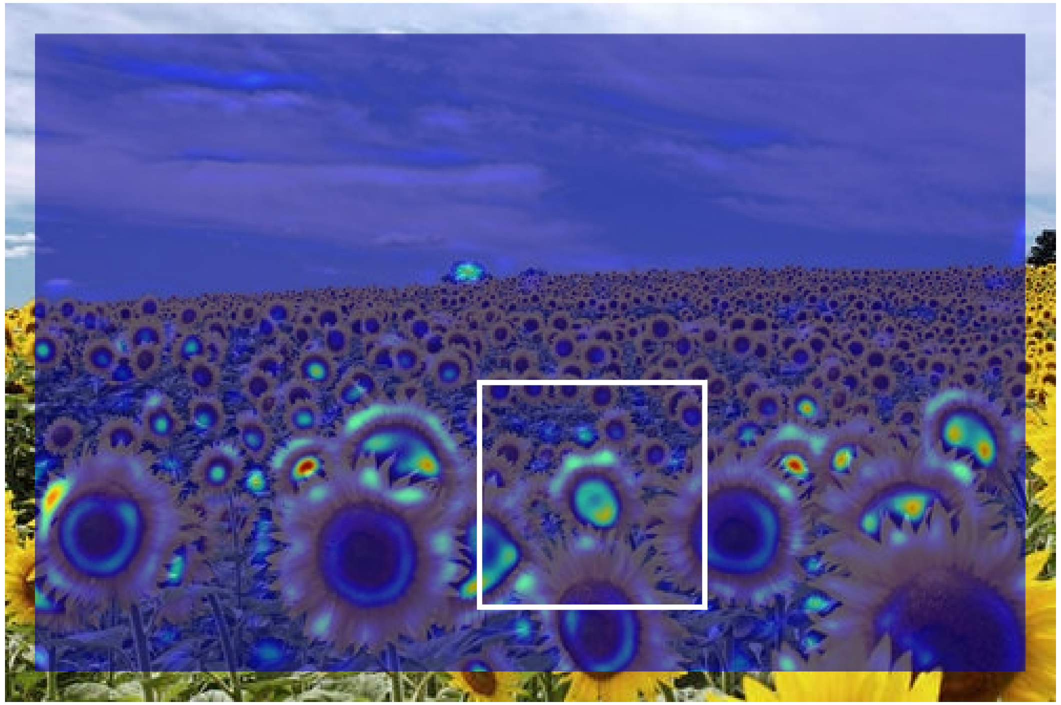



sigma=4.2
x 10^-3
2
0
-2
-4
-6
-8
-10
-12
-14
-16
-18
100
80
60
40
20
0
0
20
40
60
80
100

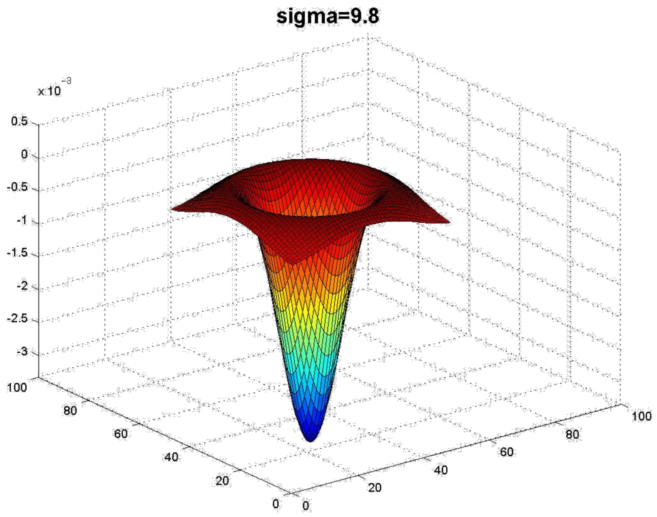




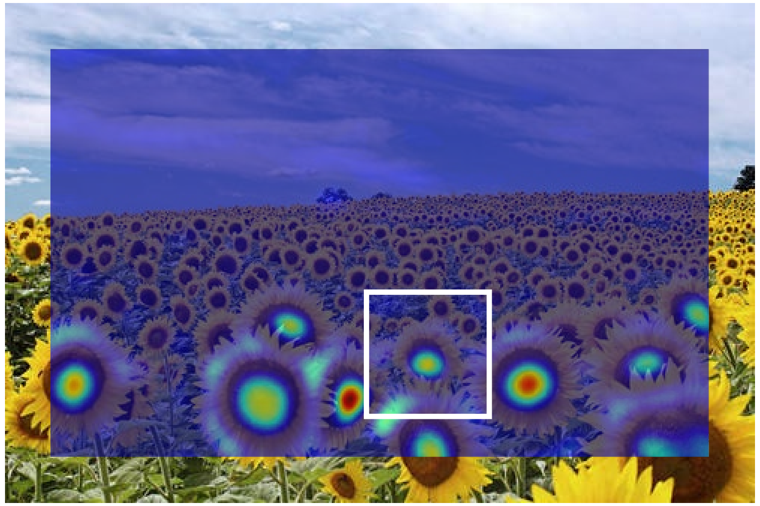

sigma=6
x 10^-3
1
0
-1
-2
-3
-4
-5
-6
-7
-8
100
80
60
40
20
0
0
20
40
60
80
100



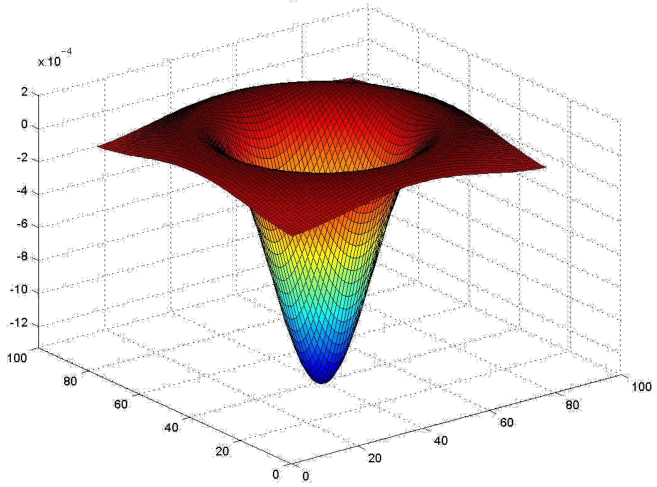

peak!

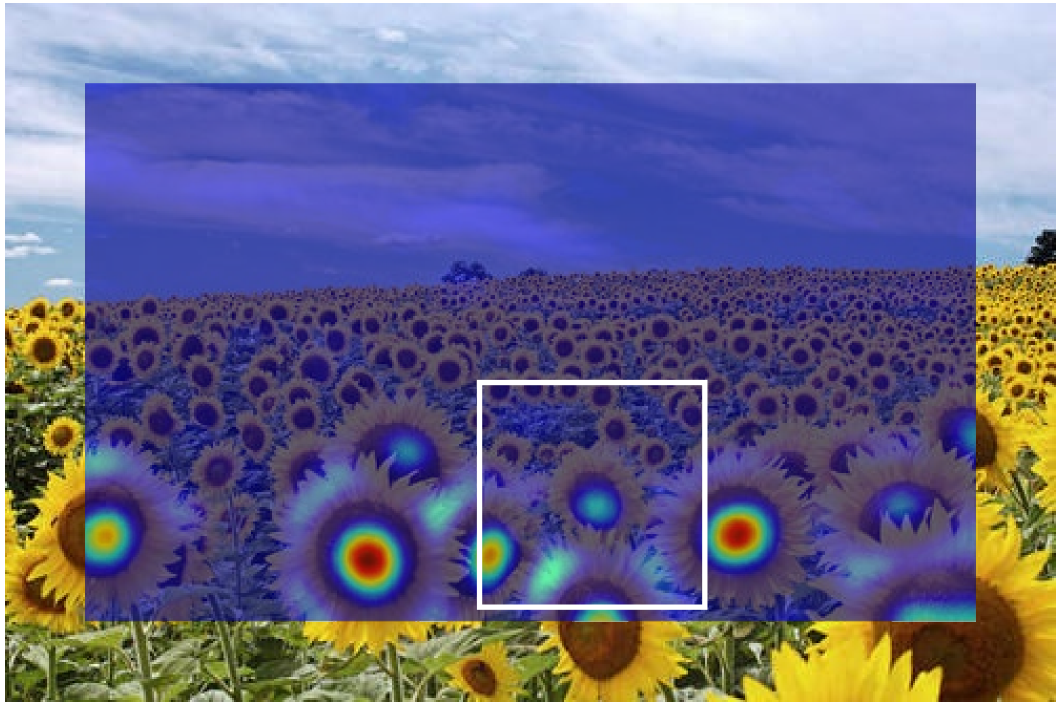

sigma=9.8
x 10^-3
0.5
0
-0.5
-1
-1.5
-2
-2.5
-3
100
80
60
40
20
0
0
20
40
60
80
100

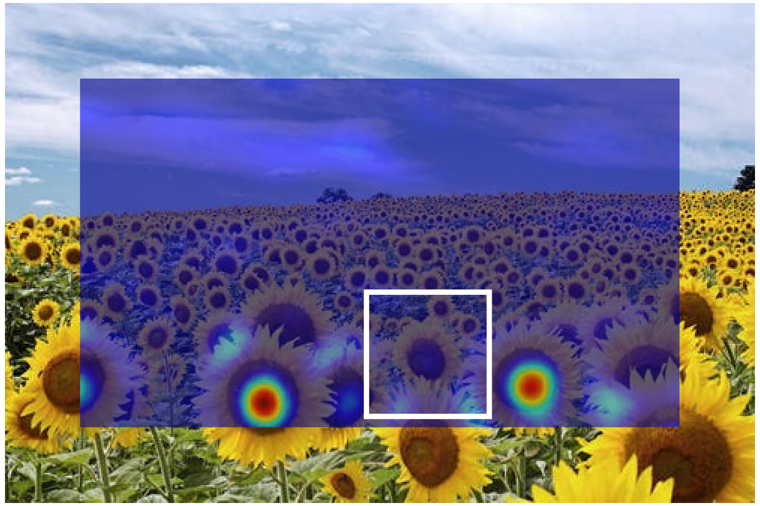

peak!

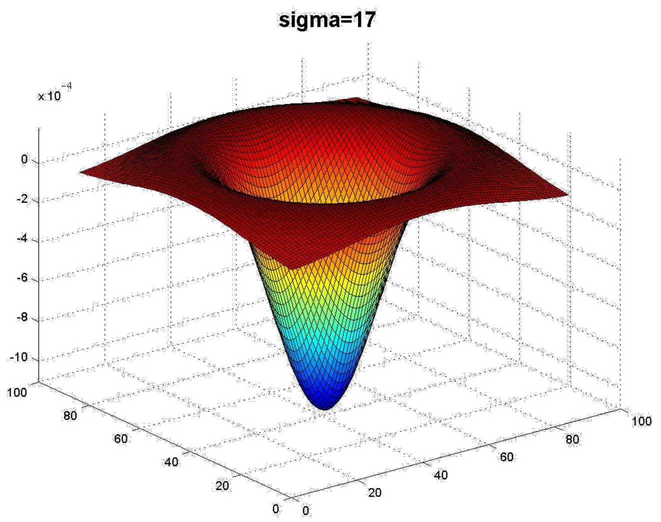


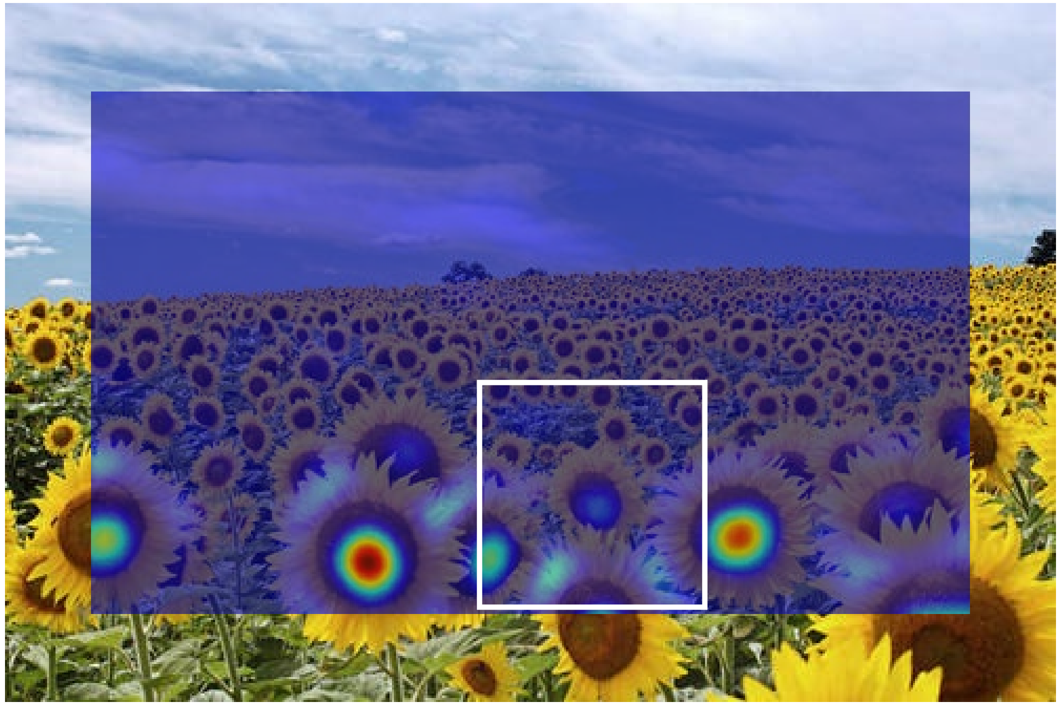
sigma=15.5

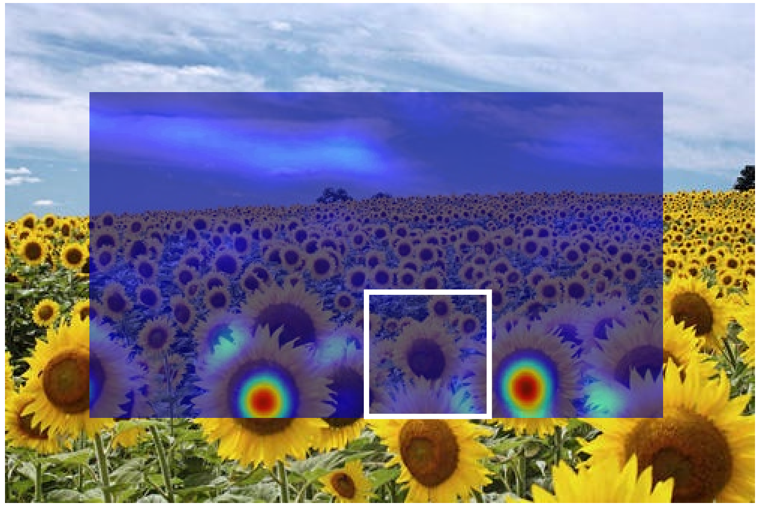


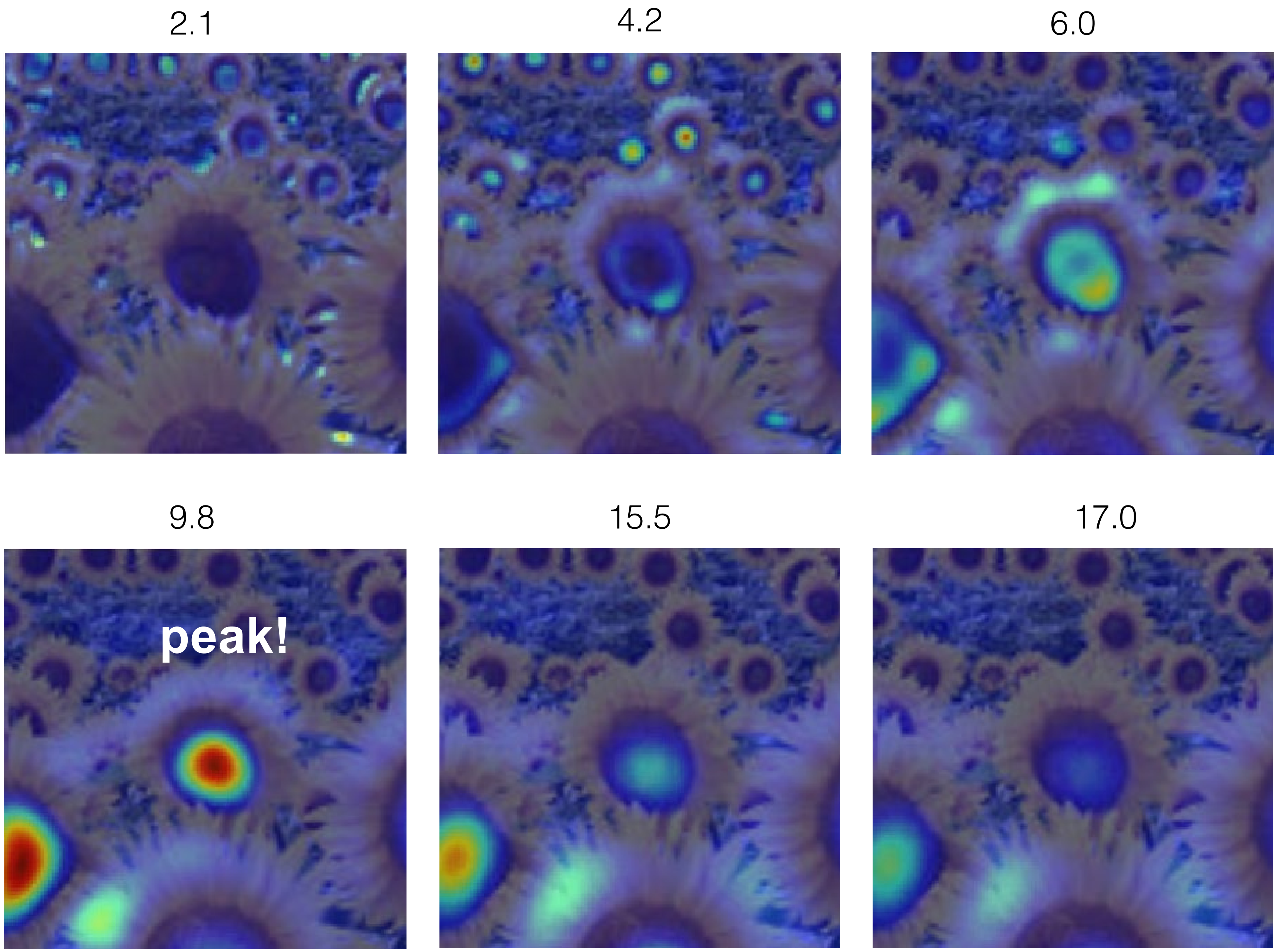


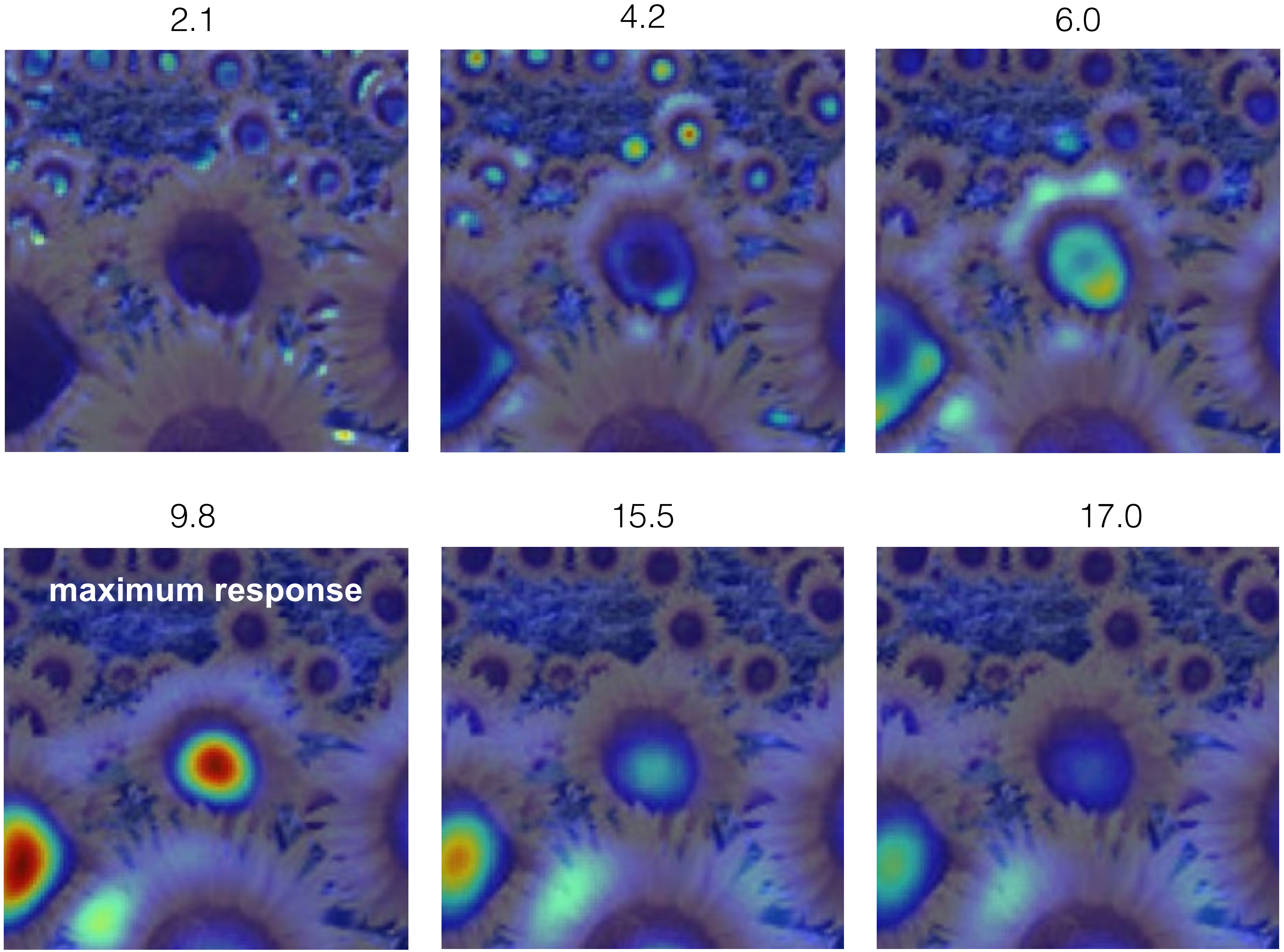



sigma=17
x 10^-4
0
-2
-4
-6
-8
-10
100
80
60
40
20
0
0
20
40
60
80
100



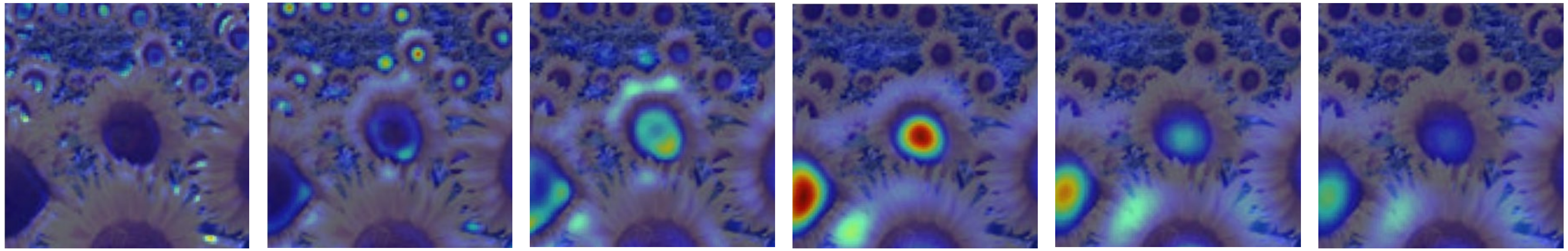



2.1
4.2
6.0
9.8
peak!
15.5
17.0


2.1
4.2
6.0
9.8
maximum response
15.5
17.0

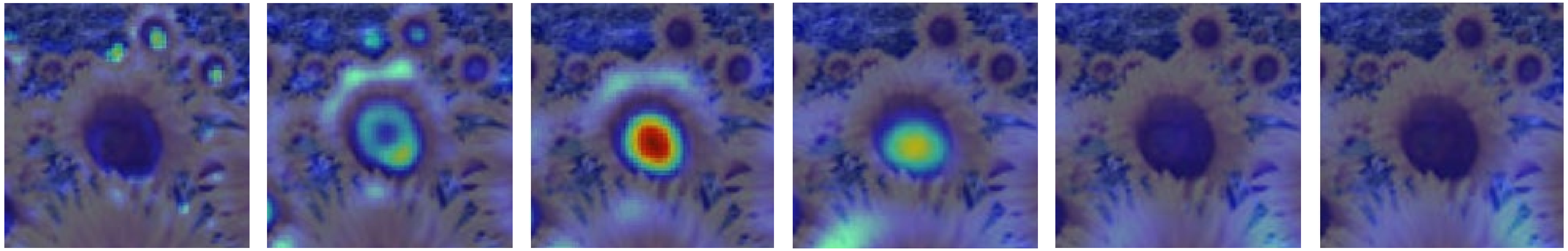
optimal scale

2.1 4.2 6.0 9.8 15.5 17.0



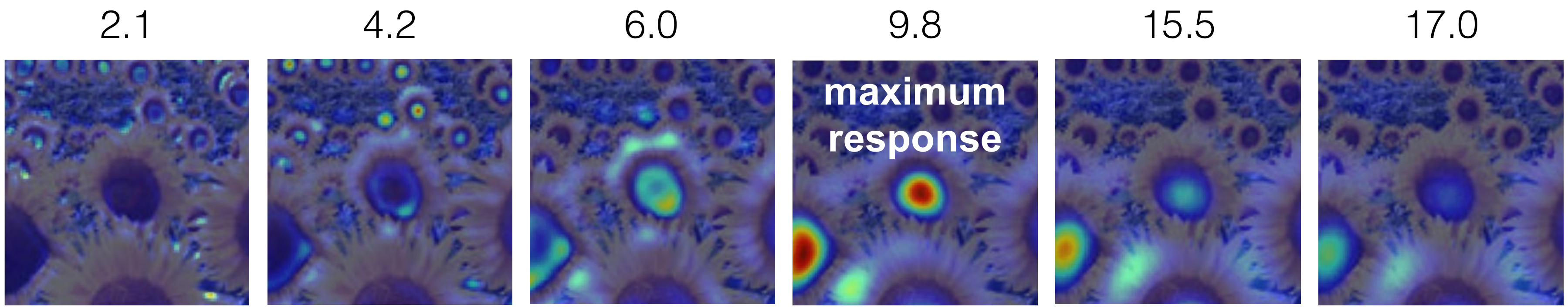
Full size image

2.1 4.2 6.0 9.8 15.5 17.0

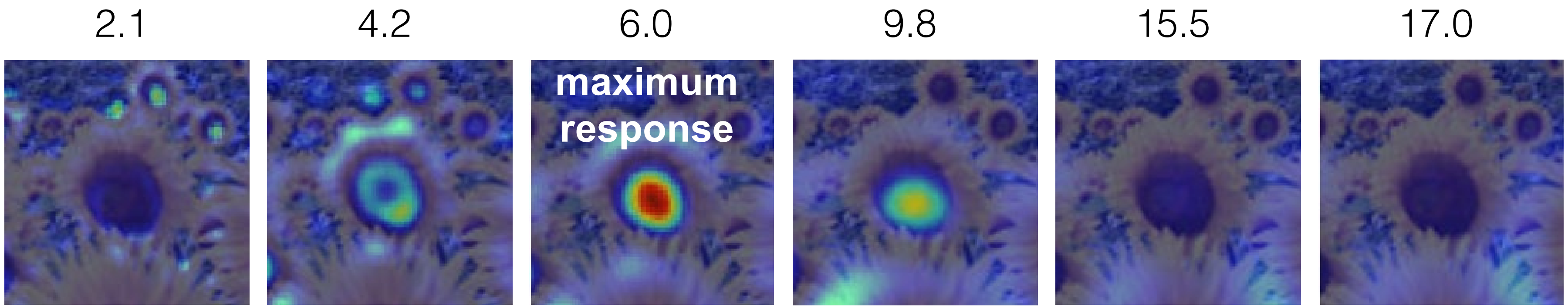


3/4 size image

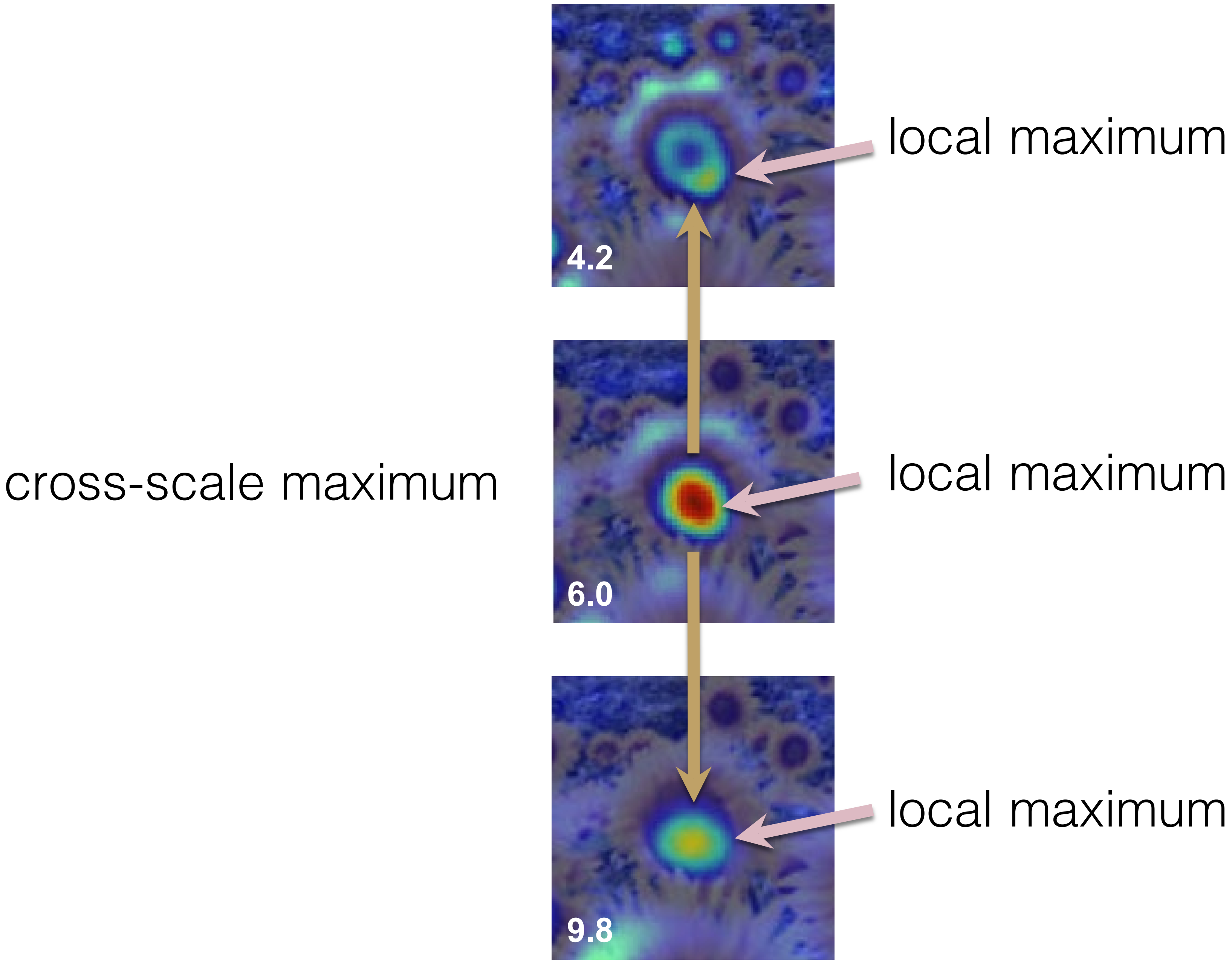
optimal scale



Full size image



3/4 size image


local maximum
4.2
cross-scale maximum
local maximum
6.0
local maximum
9.8

Multi-Scale 2D Blob Detector

Implementation

`For each level of the Gaussian Pyramid:`

- `Compute feature response`
- `If` *`local maximum`* `AND` *`cross-scale`*
    - `Save location and scale of feature` (x, y, s)