



CPSC 425: Computer Vision

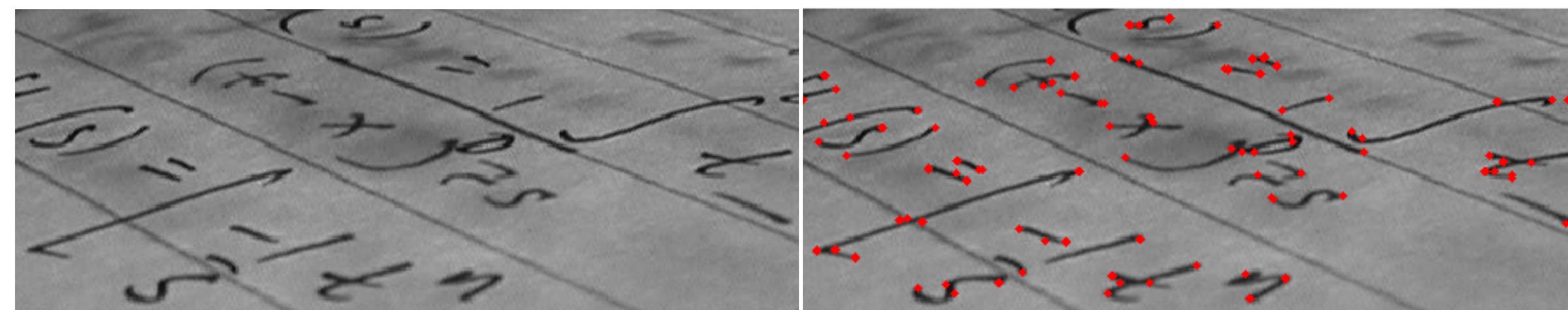


Image Credit: https://en.wikipedia.org/wiki/Corner_detection

Lecture 10: Corner Detection

(unless otherwise stated slides are taken or adopted from **Bob Woodham, Jim Little and Fred Tung**)

Menu for Today

Topics:

- Corner **Detection**
- Image **Structure**
- **Harris Corner** Detection

Readings:

- **Today's** Lecture: Szeliski 7.1-7.2, Forsyth & Ponce 5.3.0 - 5.3.1

Reminders:

- **Assignment 2:** Scaled Representations, Face Detection and Image Blending
(due **Oct 12** 23:59)
- **Midterm:** October 19th 5pm in class, 75 minutes, closed book

Midterm Example Questions + Solutions

The screenshot shows a web browser window with a dark blue header bar containing the text "CPSC 425 101 2023W1 Computer Vision". The main content area displays a course page for "2023W1". On the left, a sidebar lists various course navigation links: UBC logo, Account, Dashboard, Courses (with "2023W1" selected), Calendar, Inbox, History, Commons, Help, and several links under Library Online, Course Reserves, Chat, Item Banks, Course Evaluation, Evaluation Reports, Student Time Zones, Media Gallery, My Media, and Piazza.

The main content area starts with a note about the course webpage being the main source of information, mentioning lecture notes, slides, assignments, and discussions. It also mentions Piazza as the main forum for discussions and office hours starting week beginning September 11th. The "Office Hours" section lists days and times: Monday [redacted], Wednesday [redacted], and a large grayed-out block for the rest of the week.

The "Assignments" section lists assignment due dates:

- Assignment 1: [Image Filtering and Hybrid Images, Sep 28](#)
- Assignment 2: [Scaled Representations, Face Detection and Image Blending, Oct 12](#)
- Assignment 3: Texture Synthesis, Oct 26
- Assignment 4: RANSAC and Panorama Stitching, Nov 9
- Assignment 5: Scene Recognition with Bag of Words, Nov 23
- Assignment 6: Deep Learning, Dec 7

A red box highlights the "Midterm prep" section, which contains a link to "Midterm preparation problems and solutions (zip) ↓". A black arrow points from the text "Scroll down on homepage on Canvas" to this red box.

On the right side of the screen, there is a sidebar titled "Coming Up" with a "View Calendar" button. It shows two upcoming assignments:

- Grade Assignment 1: Image Filtering and Hybrid Images (60 points) due Sep 28 at 11:59p.m.
- Grade Assignment 2: Scaled Representations, Face Detection and Image Blending (45 points) due Oct 12 at 11:59p.m.

Below the calendar, it says "Nothing for the next week".

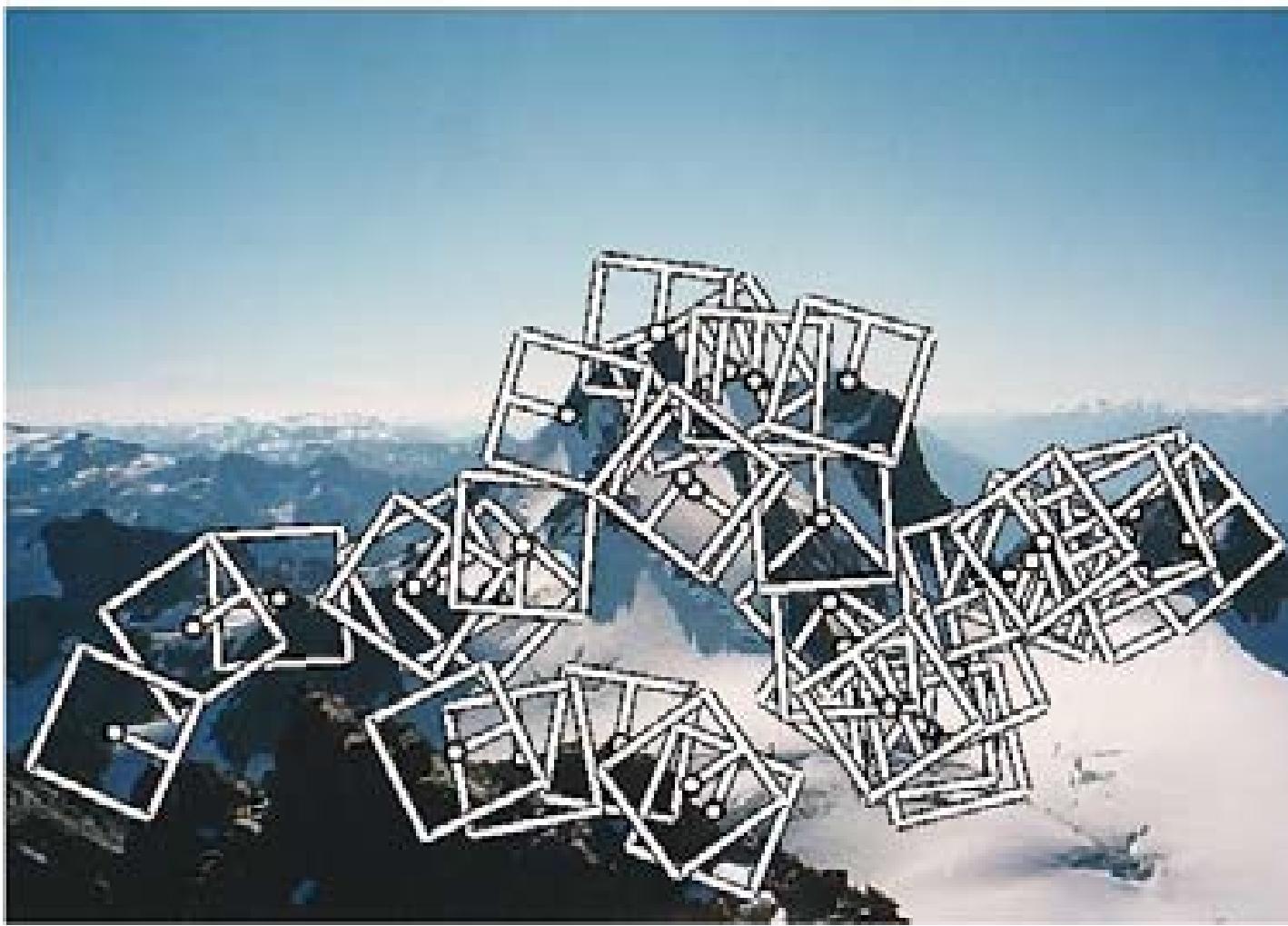
← Scroll down on homepage on Canvas

Correspondence Problem

- A basic problem in Computer Vision is to establish matches (correspondences) between images
- This has **many** applications: rigid/non-rigid tracking, object recognition, image registration, structure from motion, stereo...



Feature Detectors



Corners/Blobs



Regions



Edges



Straight Lines

Feature Descriptors

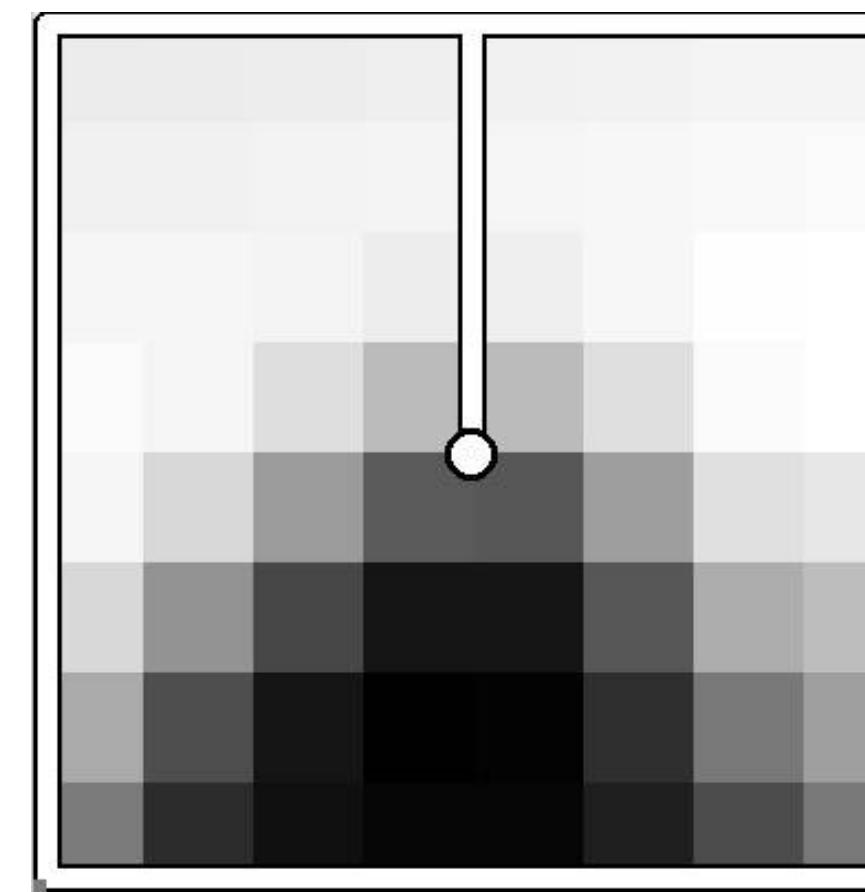
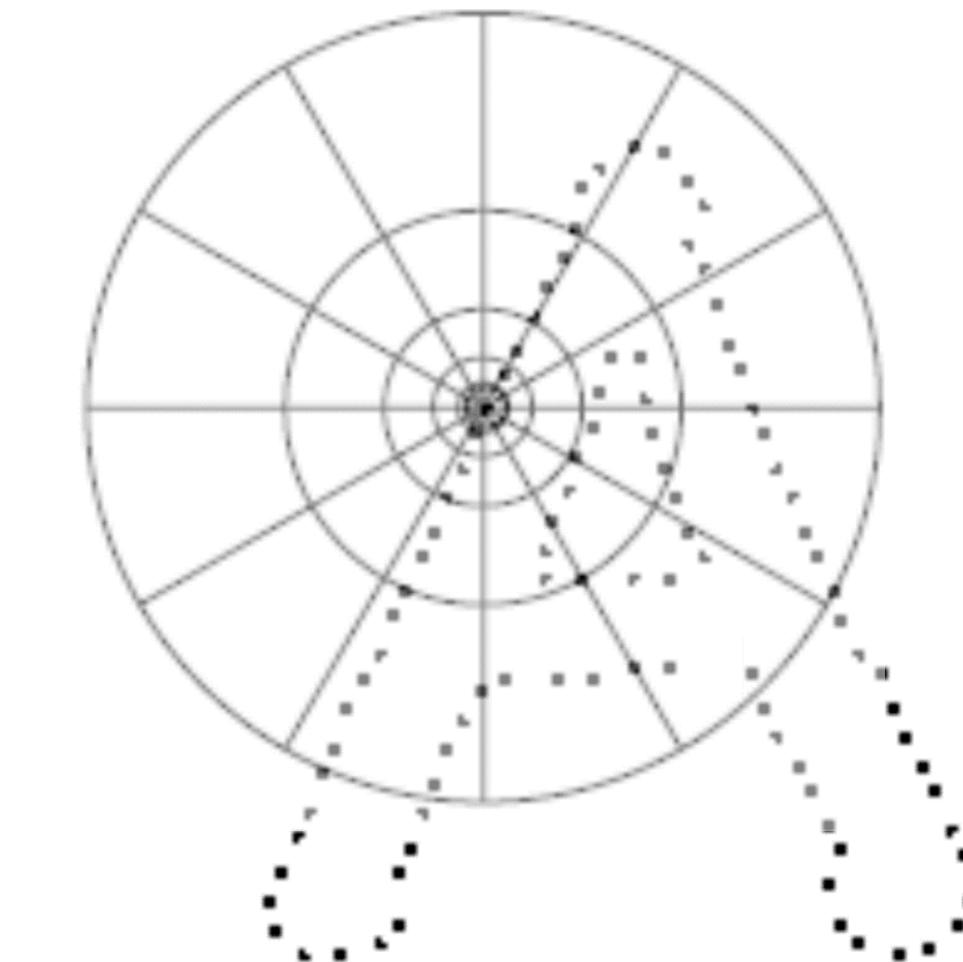
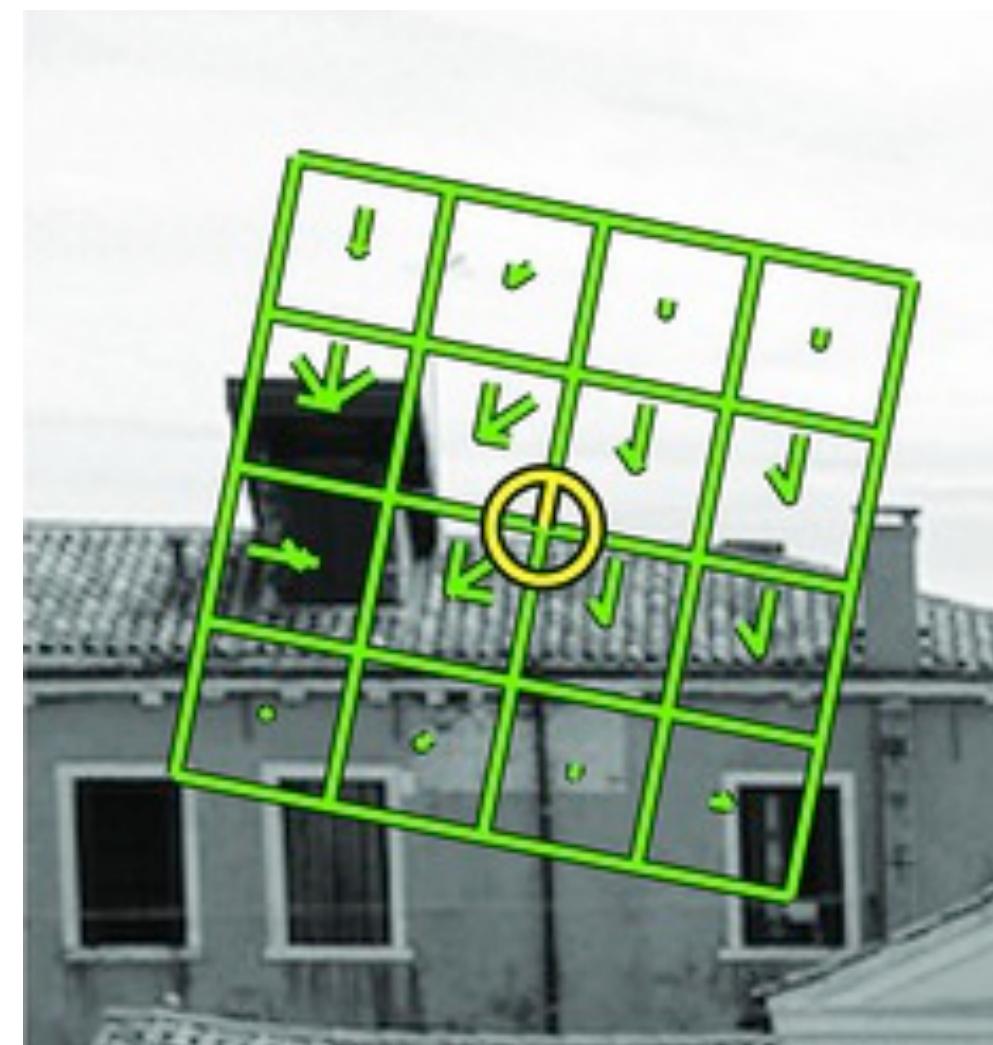


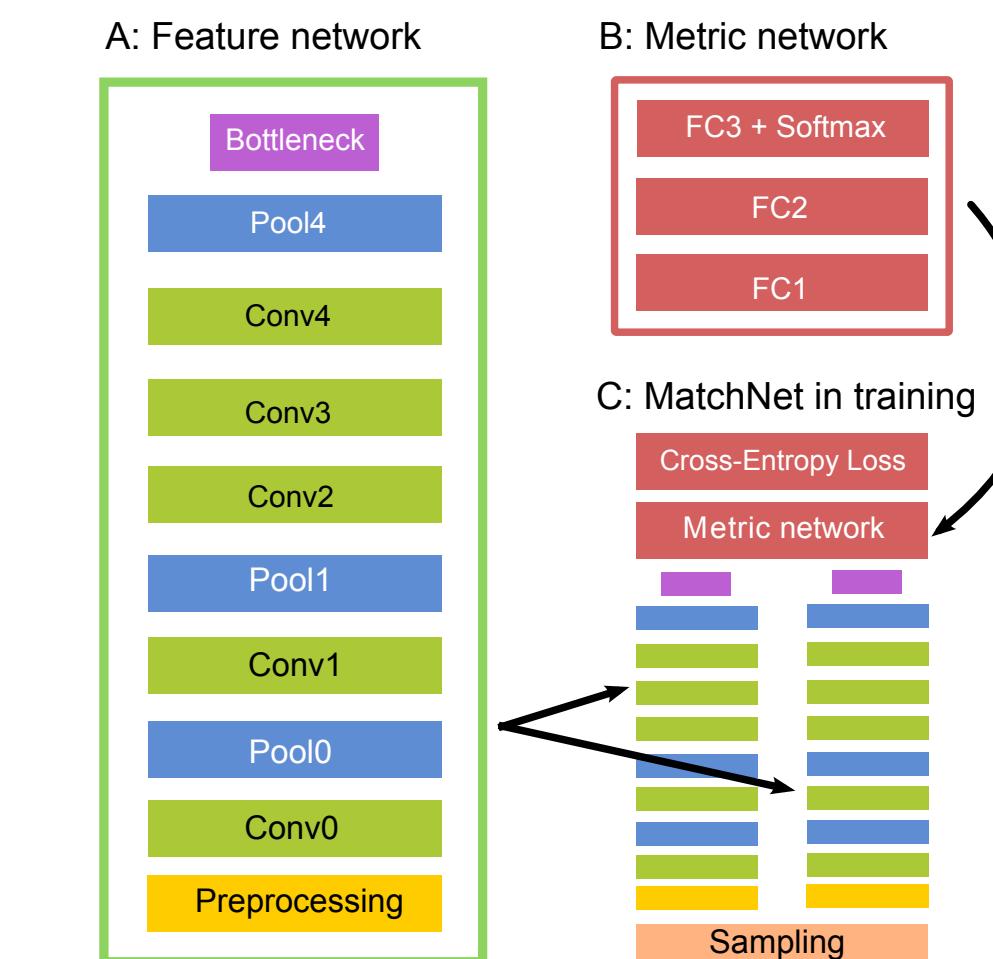
Image Patch



Shape Context



SIFT



Learned Descriptors

What is a Good Feature Detector?

Local: features are local, robust to occlusion and clutter

Accurate: precise localization

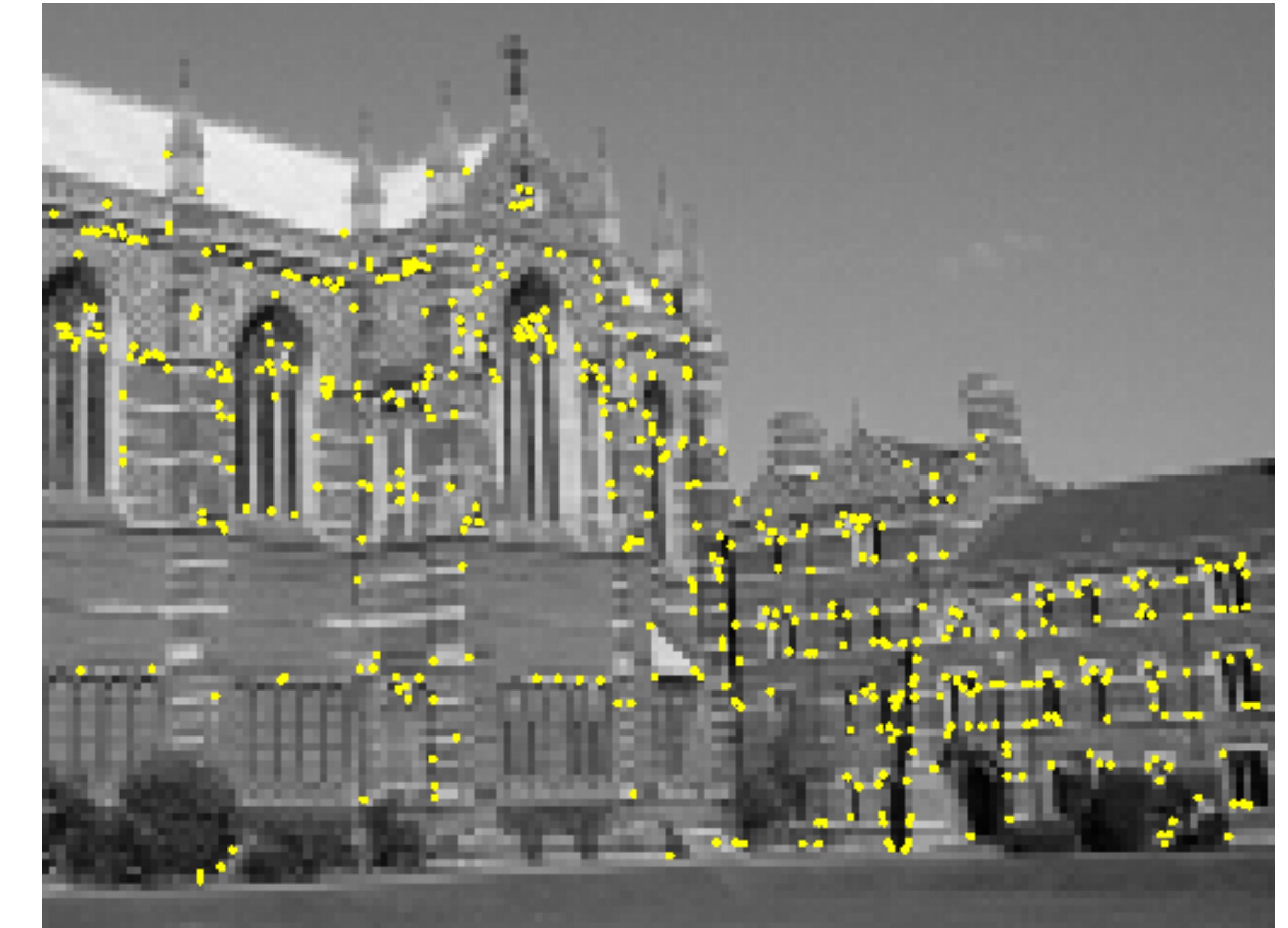
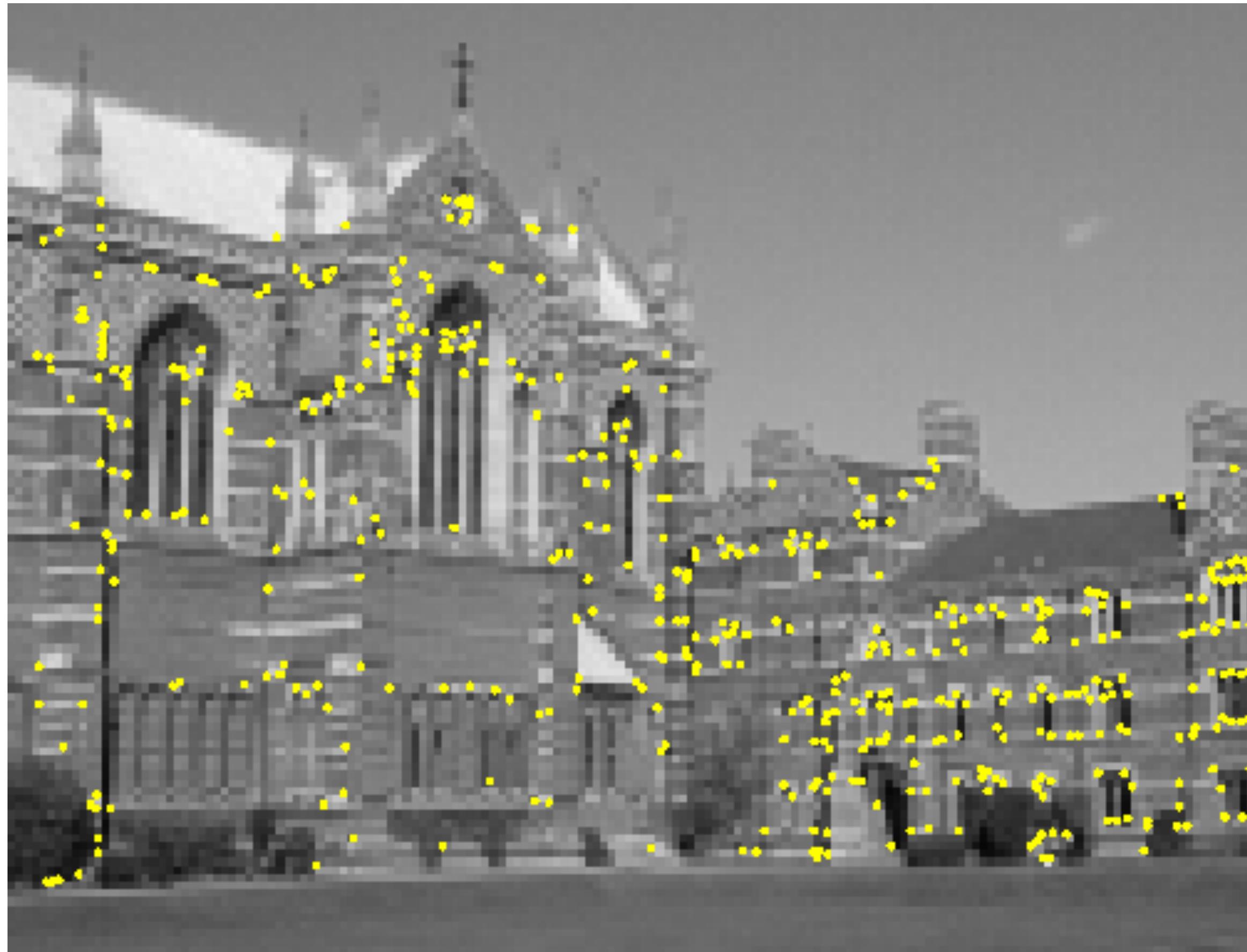
Robust: noise, blur, compression, etc. do not have a big impact on the feature.

Distinctive: individual features can be easily matched

Efficient: close to real-time performance

Corner Detection

- e.g., Harris corners are peaks of a local similarity function

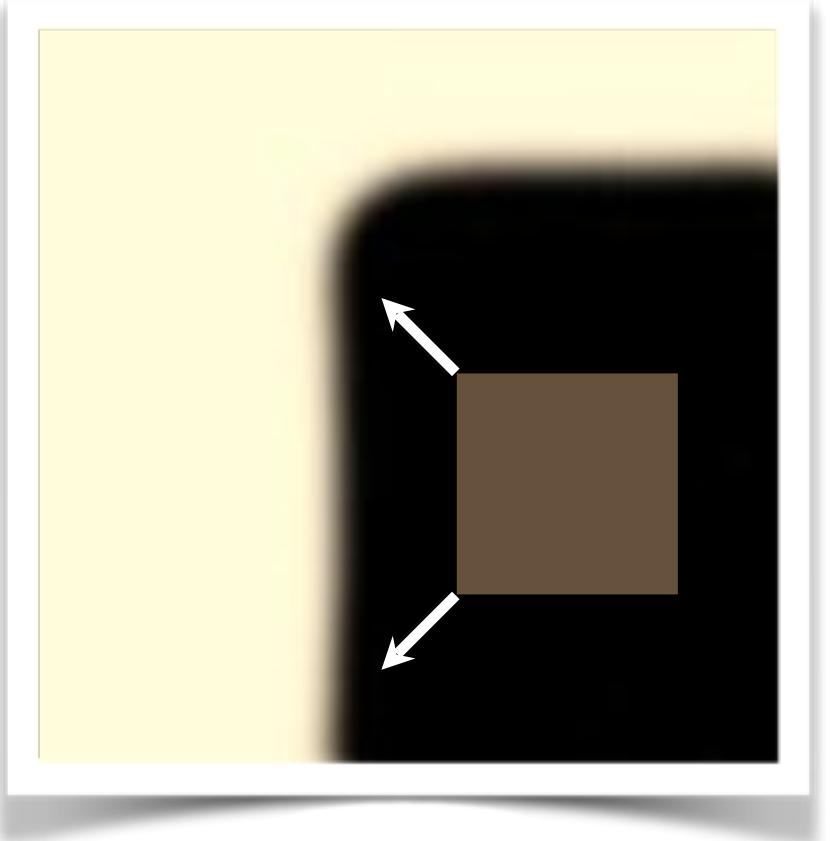


Why are corners **distinct**?

A corner can be **localized reliably**.

Thought experiment:

- Place a small window over a patch of constant image value.



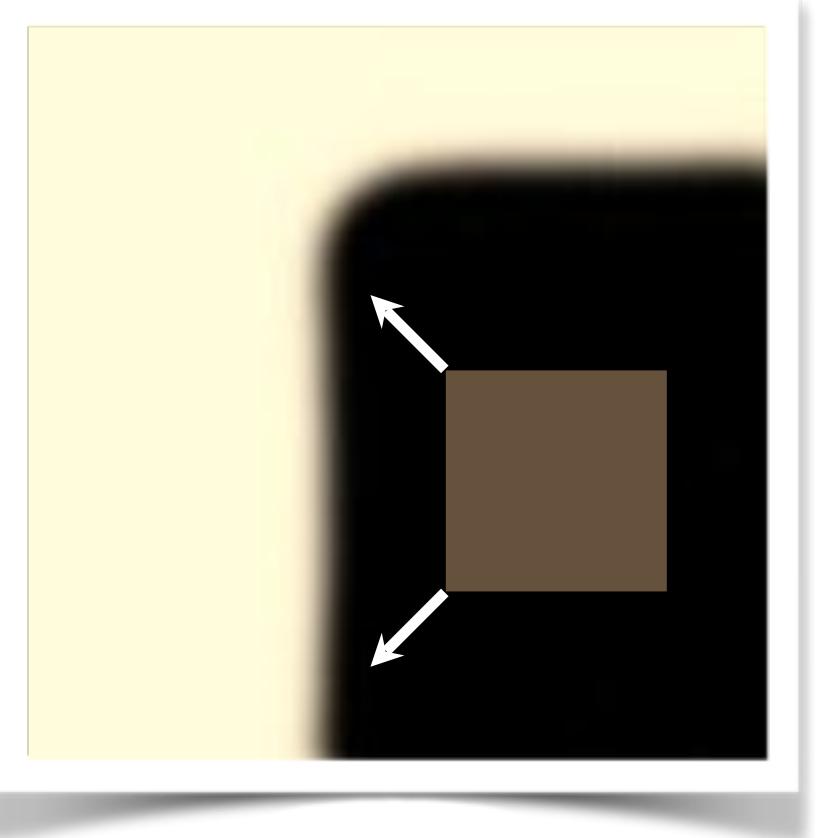
“flat” region:

Why are corners **distinct**?

A corner can be **localized reliably**.

Thought experiment:

- Place a small window over a patch of constant image value. If you slide the window in any direction, the image in the window will not change.



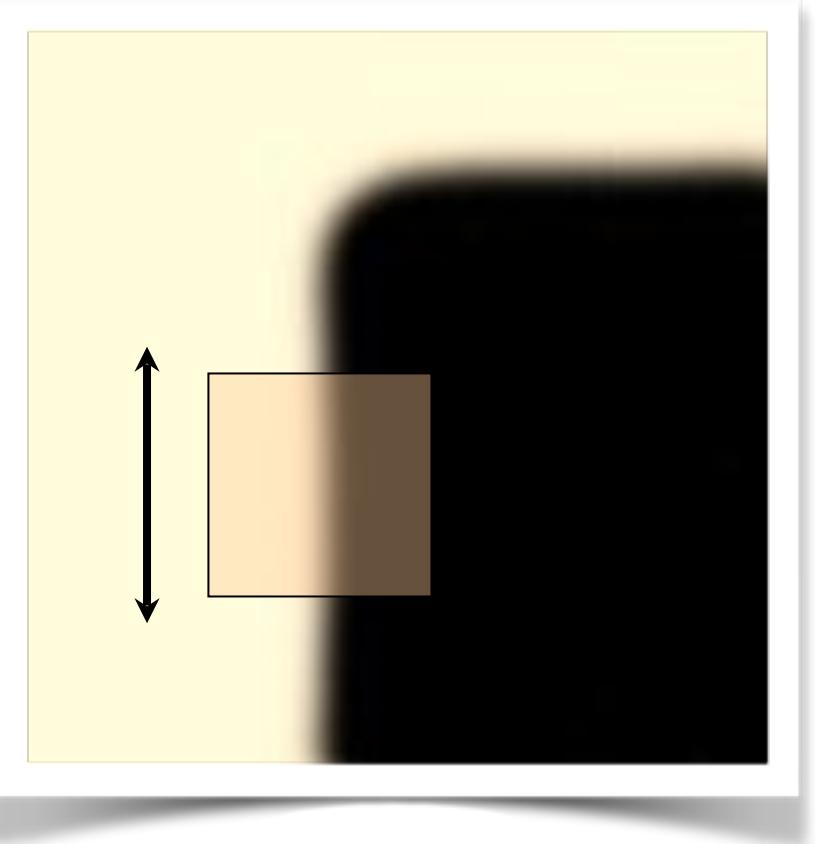
“flat” region:
no change in all
directions

Why are corners **distinct**?

A corner can be **localized reliably**.

Thought experiment:

- Place a small window over a patch of constant image value.
If you slide the window in any direction, the image in the window will not change.
- Place a small window over an edge.



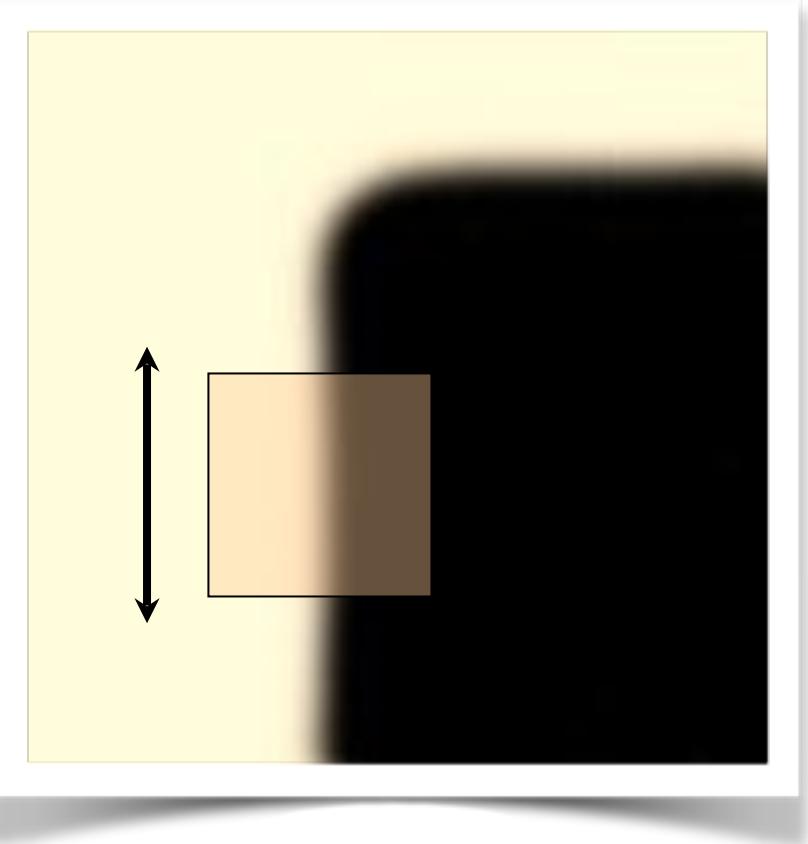
“edge”:

Why are corners **distinct**?

A corner can be **localized reliably**.

Thought experiment:

- Place a small window over a patch of constant image value.
If you slide the window in any direction, the image in the window will not change.
- Place a small window over an edge. If you slide the window in the direction of the edge, the image in the window will not change
 - Cannot estimate location along an edge (a.k.a., **aperture** problem)



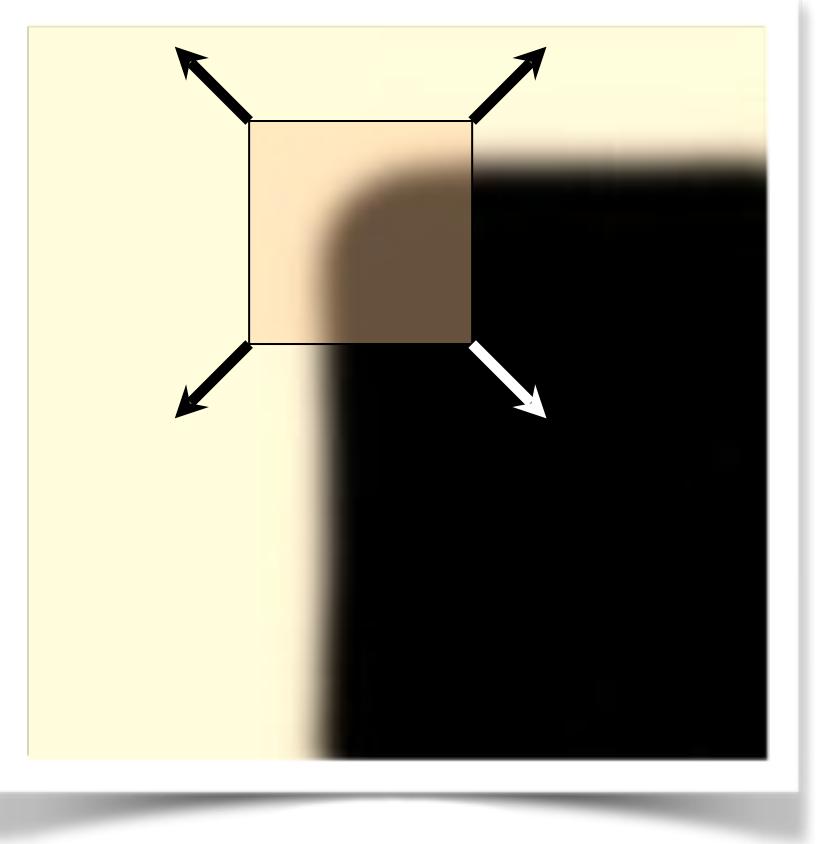
“**edge**”:
no change along
the edge direction

Why are corners **distinct**?

A corner can be **localized reliably**.

Thought experiment:

- Place a small window over a patch of constant image value.
If you slide the window in any direction, the image in the window will not change.
- Place a small window over an edge. If you slide the window in the direction of the edge, the image in the window will not change
→ Cannot estimate location along an edge (a.k.a., **aperture** problem)
- Place a small window over a corner.



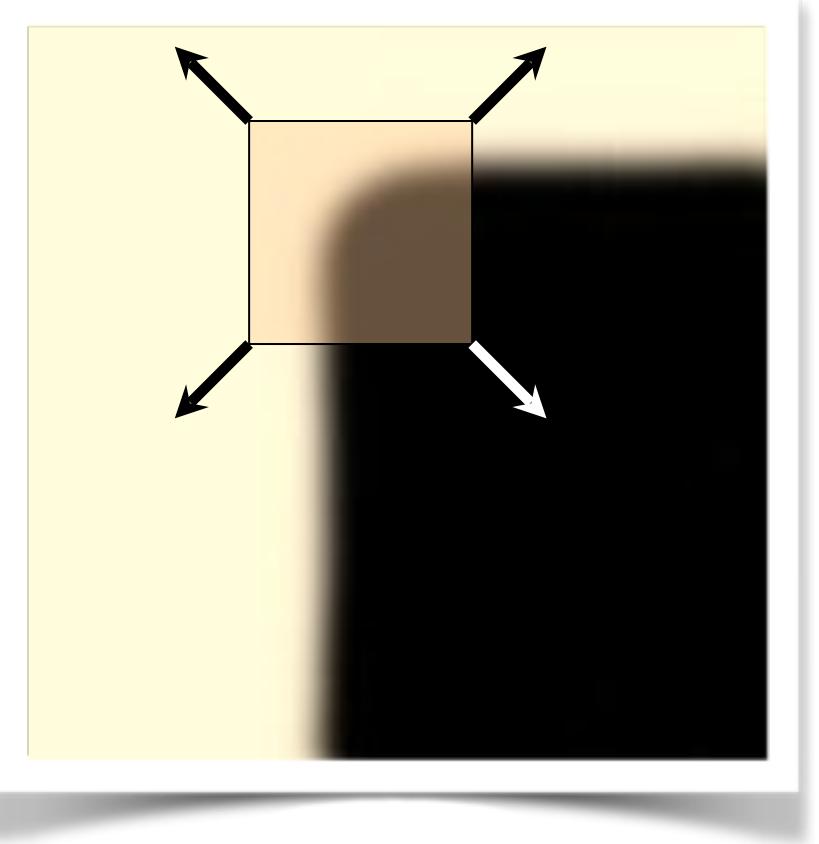
“corner”:

Why are corners **distinct**?

A corner can be **localized reliably**.

Thought experiment:

- Place a small window over a patch of constant image value.
If you slide the window in any direction, the image in the window will not change.
- Place a small window over an edge. If you slide the window in the direction of the edge, the image in the window will not change
→ Cannot estimate location along an edge (a.k.a., **aperture** problem)
- Place a small window over a corner. If you slide the window in any direction, the image in the window changes.



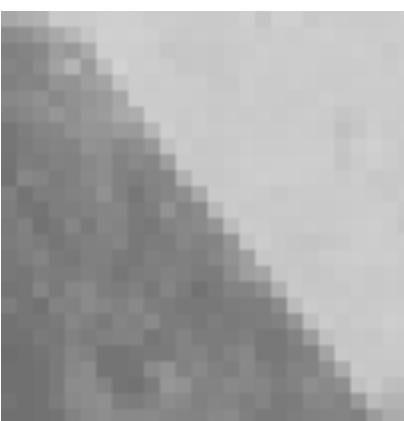
“corner”:
significant change
in all directions

Image Structure

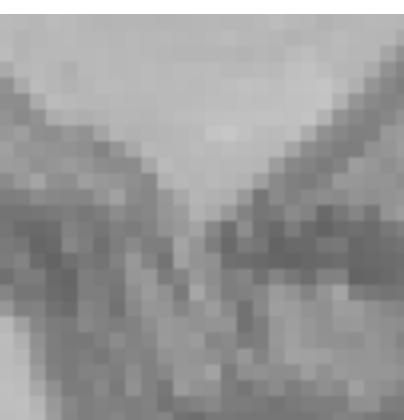
- What kind of structures are present in the image locally?



0D Structure: not useful for matching



1D Structure: edge, can be localised in one direction, subject to the “aperture problem”



2D Structure: corner, or interest point, can be localised in both directions, good for matching

Edge detectors find contours (1D structure), **Corner** or **Interest point** detectors find points with 2D structure.

Local SSD Function

- Consider the sum squared difference (SSD) of a patch with its local neighbourhood

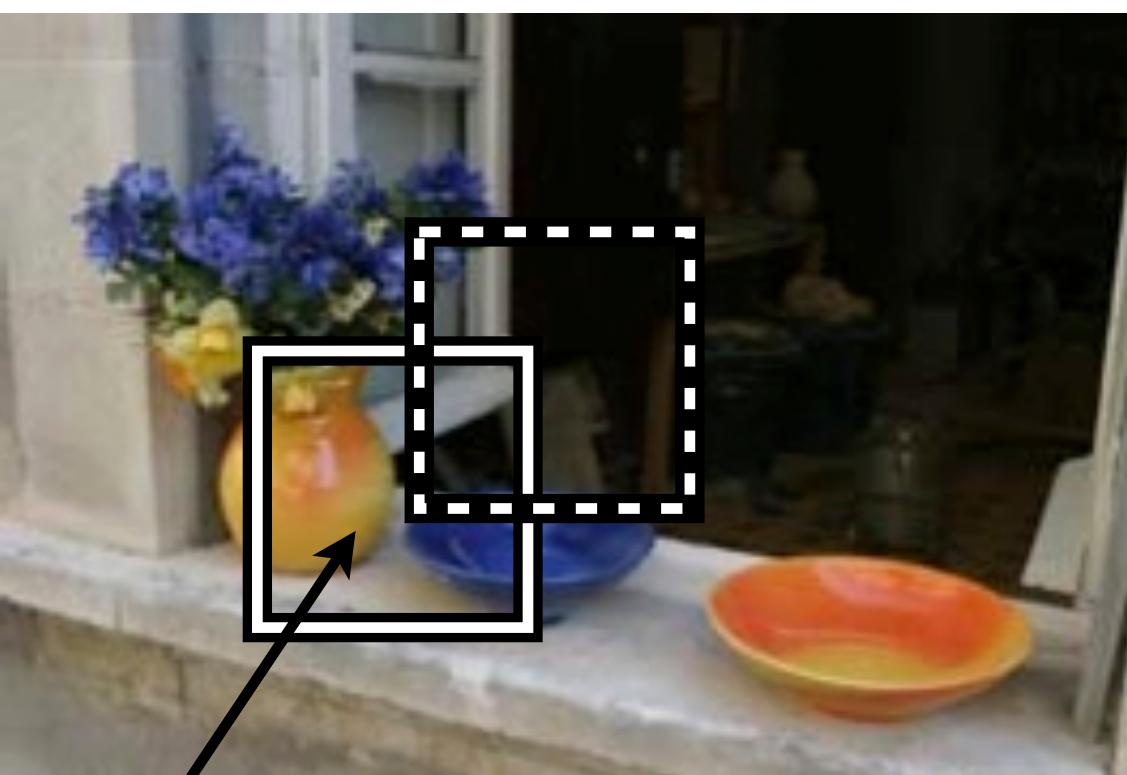
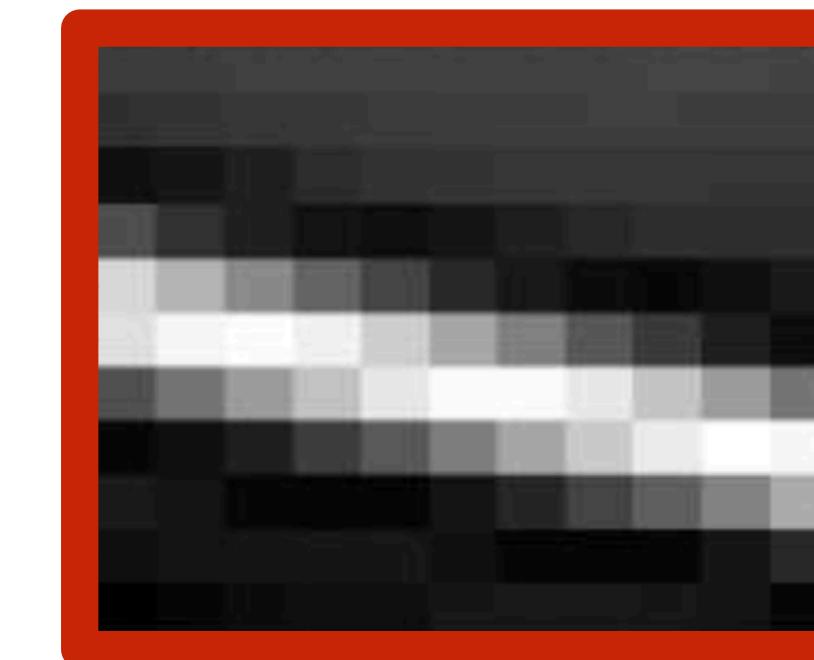
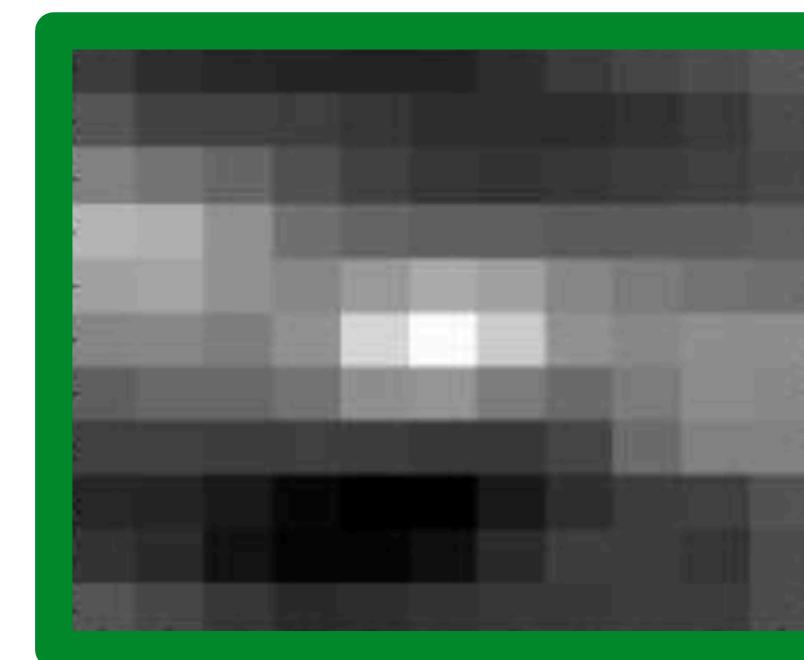

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \xrightarrow{\Delta x_1}$$
$$\Delta x_2 \uparrow$$
$$\text{SSD} = \sum_{\mathcal{R}} |I(\mathbf{x}) - I(\mathbf{x} + \Delta\mathbf{x})|^2$$

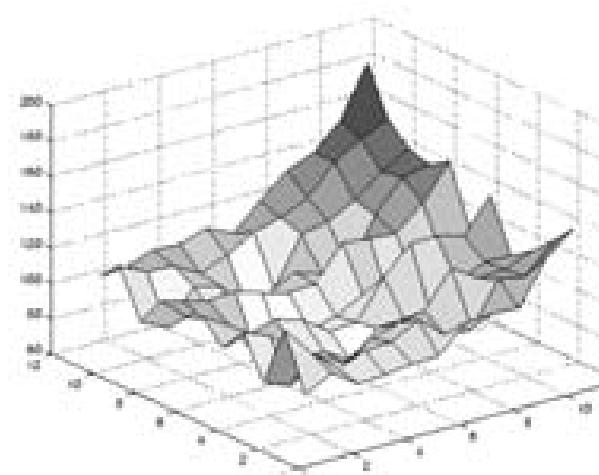
Image Structure



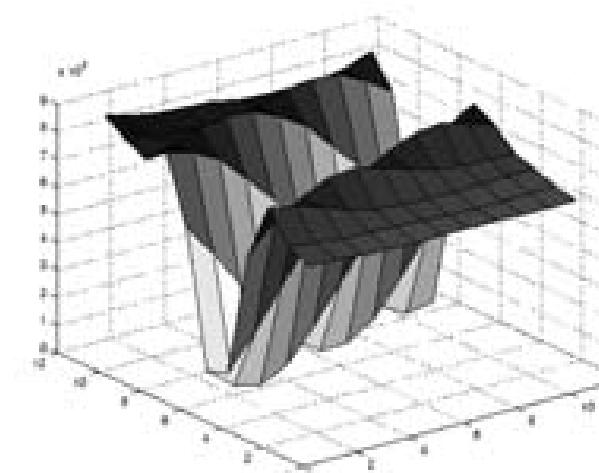
Szeliski, Figure 4.5

Local SSD Function

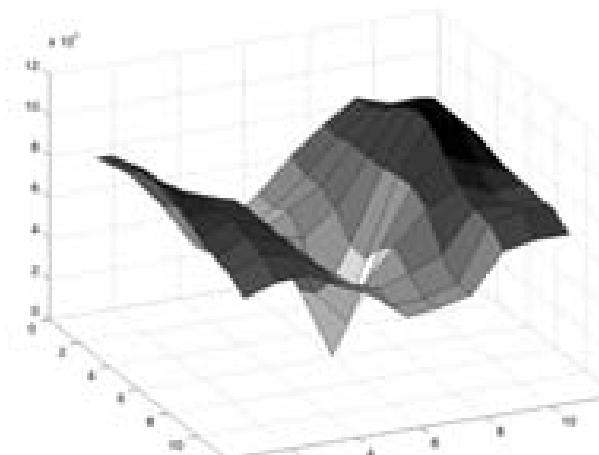
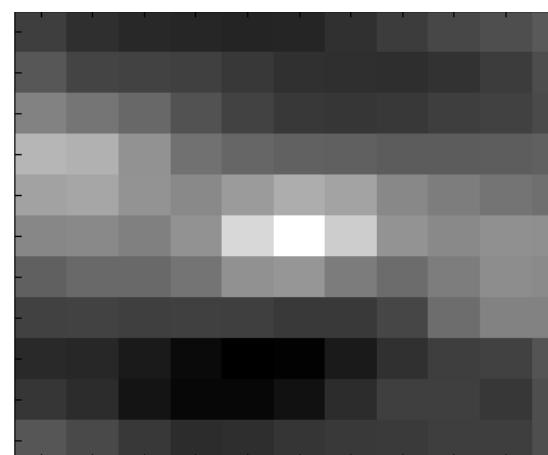
- Consider the local SSD function for different patches



High similarity locally



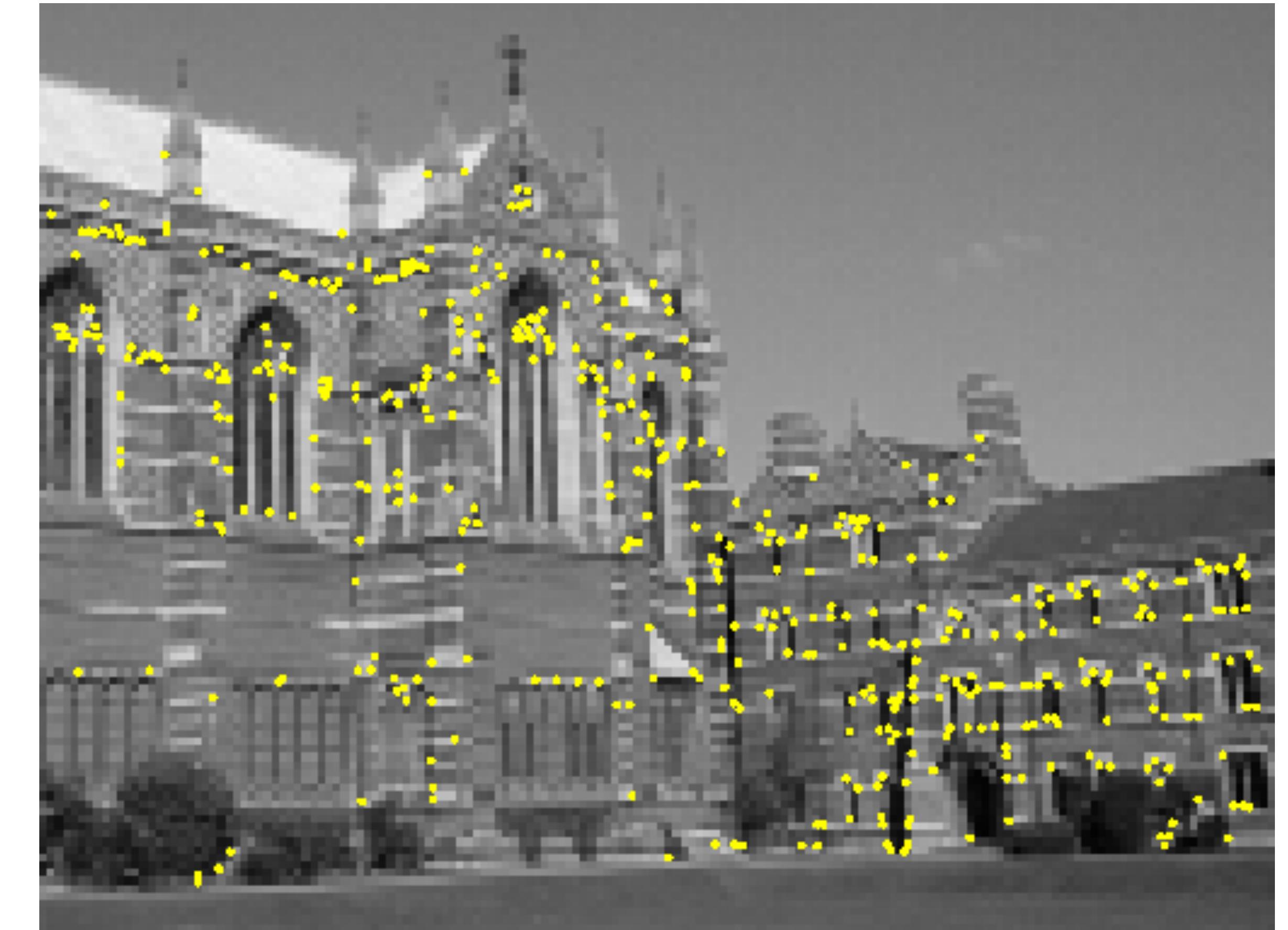
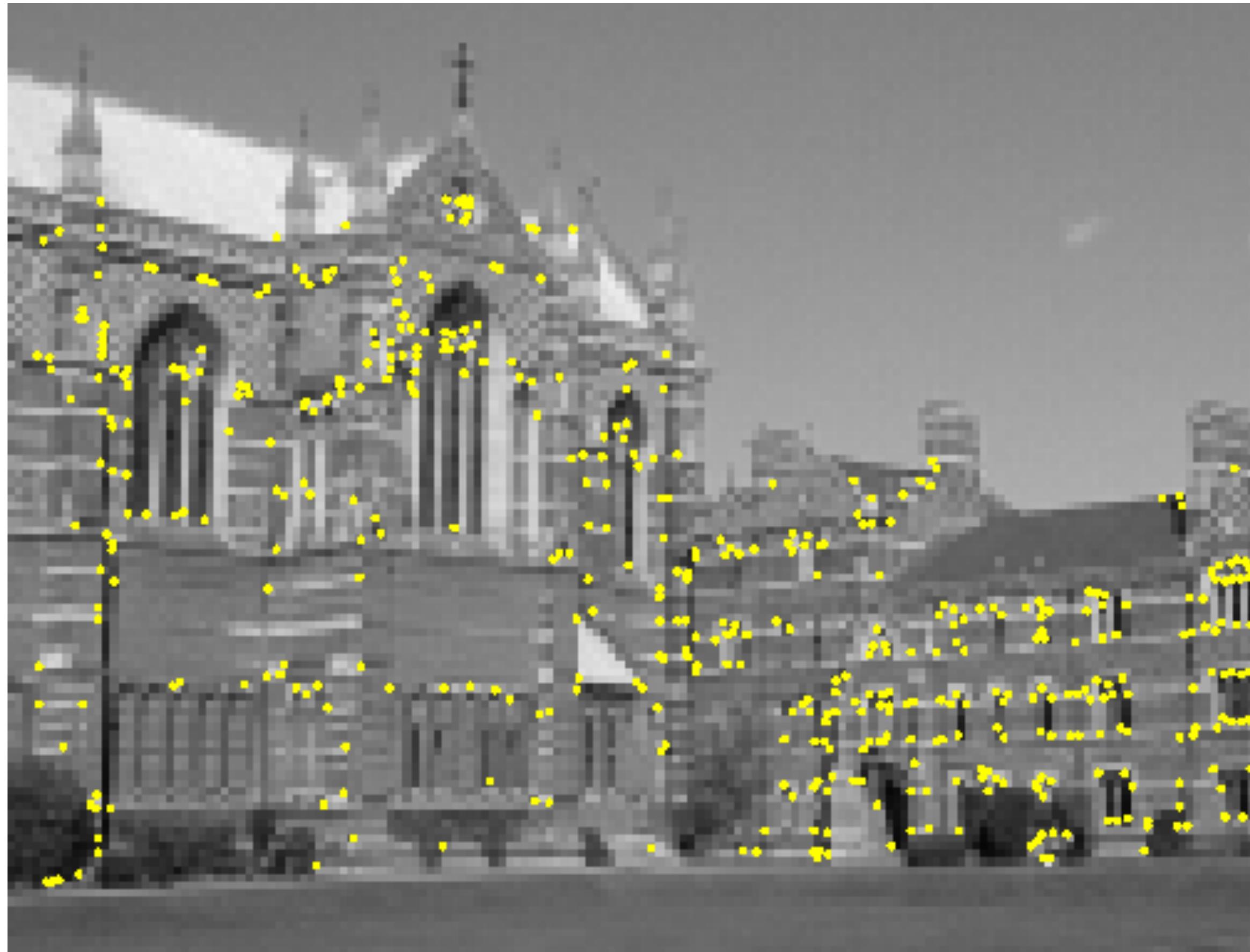
High similarity along the edge



Clear peak in similarity function

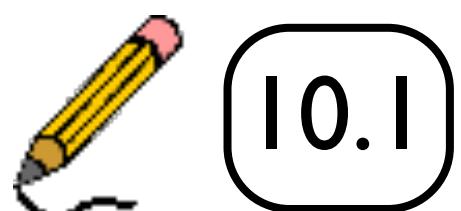
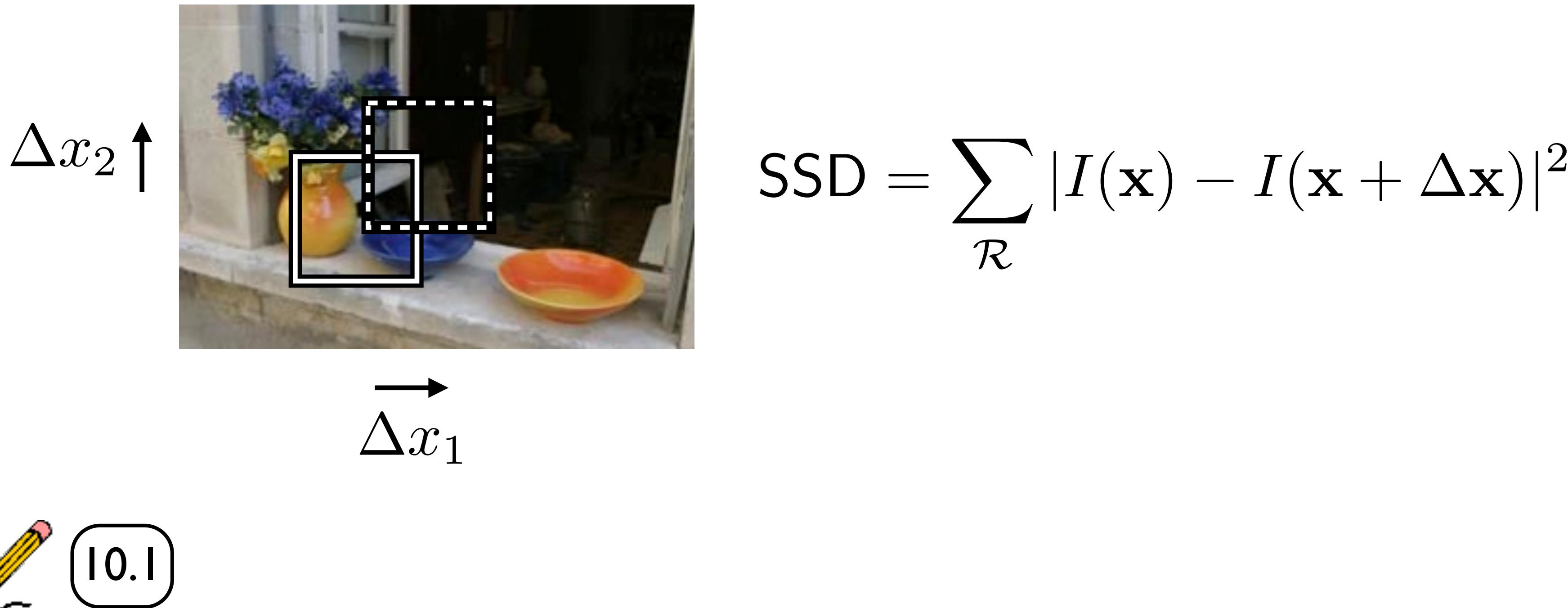
Harris Corners

- Harris corners are peaks of a local similarity function



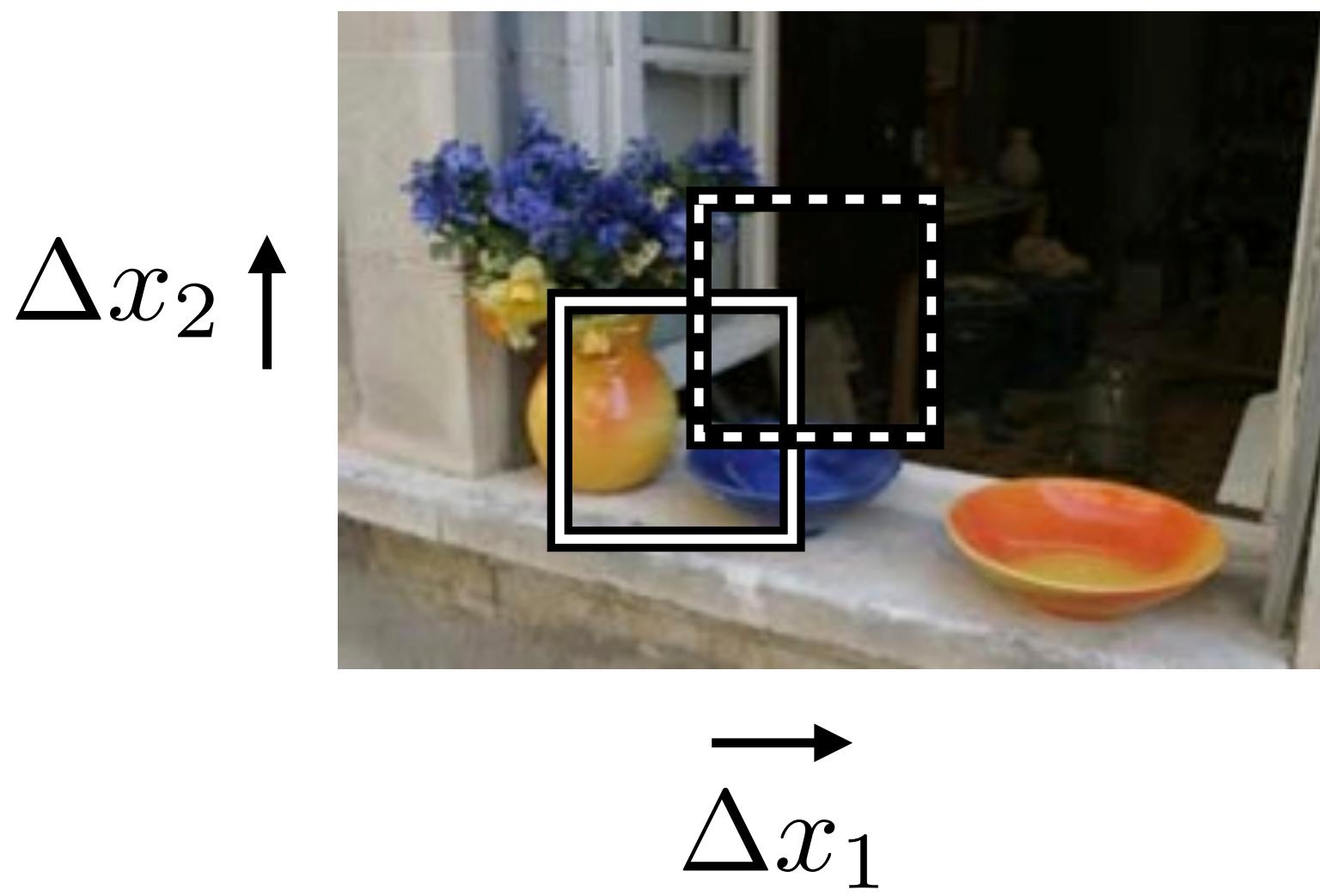
Harris Corners

- We will use a first order approximation to the local SSD function



Harris Corners

- We will use a first order approximation to the local SSD function



$$\text{SSD} = \sum_{\mathcal{R}} |I(\mathbf{x}) - I(\mathbf{x} + \Delta\mathbf{x})|^2$$

$$= \Delta\mathbf{x}^T \mathbf{H} \Delta\mathbf{x}$$

$$\mathbf{H} = \sum_{\mathcal{R}} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Compute the **covariance matrix** (a.k.a. 2nd moment matrix)

Sum over small region
around the corner

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

Compute the **covariance matrix** (a.k.a. 2nd moment matrix)

Sum over small region
around the corner

Gradient with respect to x, times
gradient with respect to y

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

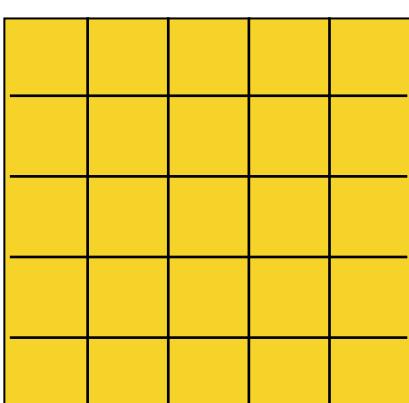
Compute the **covariance matrix** (a.k.a. 2nd moment matrix)

Sum over small region around the corner

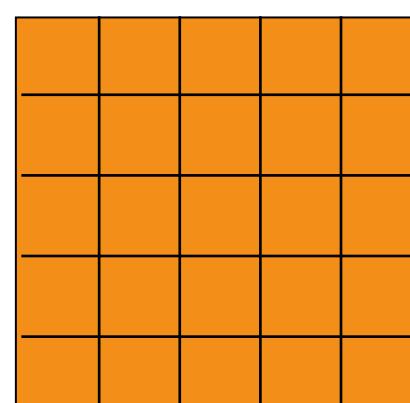
Gradient with respect to x, times gradient with respect to y

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

$$\sum_{p \in P} I_x I_y = \text{sum}($$

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


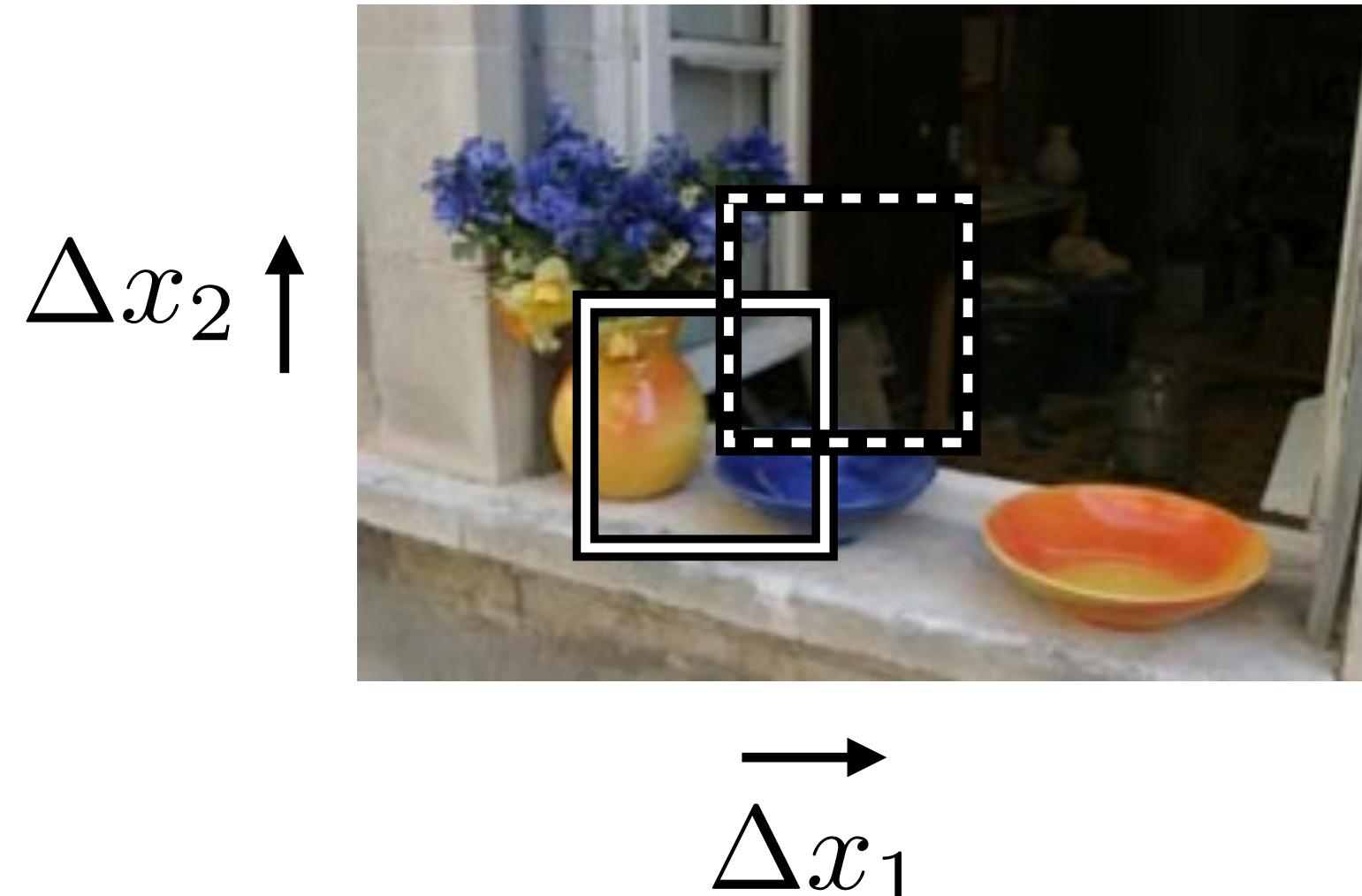
$\cdot \ast$

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


)

array of x gradients array of y gradients

Harris Corners



$$SSD = \sum_{\mathcal{R}} |I(\mathbf{x}) - I(\mathbf{x} + \Delta\mathbf{x})|^2$$

$$= \Delta\mathbf{x}^T \mathbf{H} \Delta\mathbf{x}$$

$$\mathbf{H} = \sum_{\mathcal{R}} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

SSD function must be large **for all shifts**

$\Delta\mathbf{x}$ for a corner / 2D structure

This implies that **both eigenvalues of \mathbf{H}** must be **large**

Note that \mathbf{H} is a **2x2 matrix**

Recap: Computing **Eigenvalues** and **Eigenvectors**

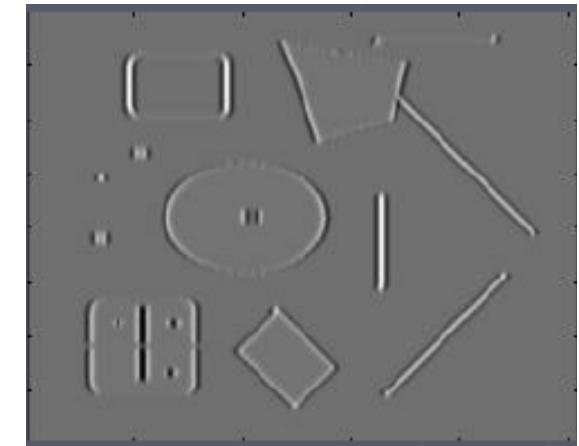


10.2

Harris Corner Detection

1. Compute image gradients over small region
2. Compute the covariance matrix
3. Compute eigenvectors and eigenvalues
4. Use threshold on eigenvalues to detect corners

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

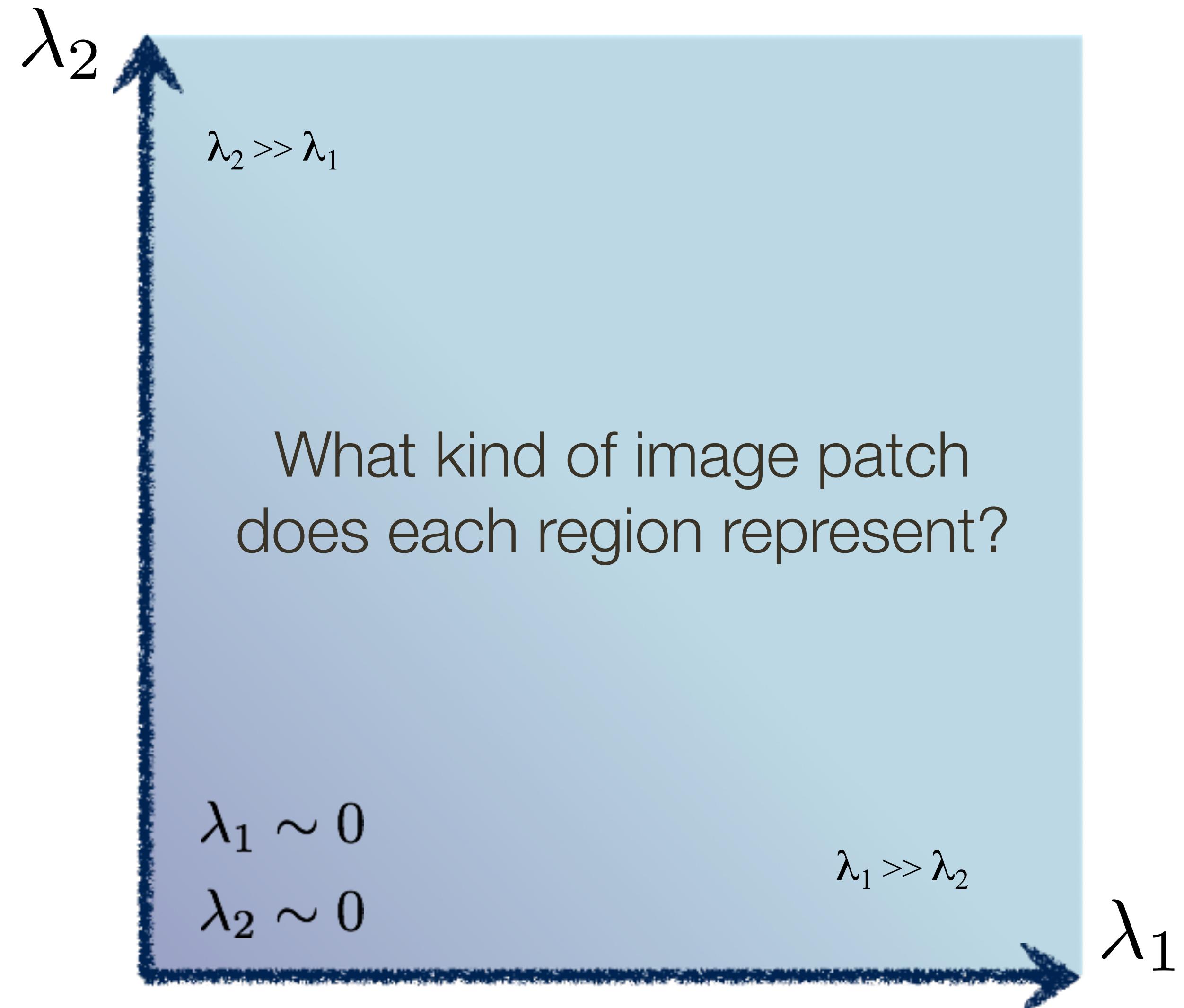


$$I_y = \frac{\partial I}{\partial y}$$

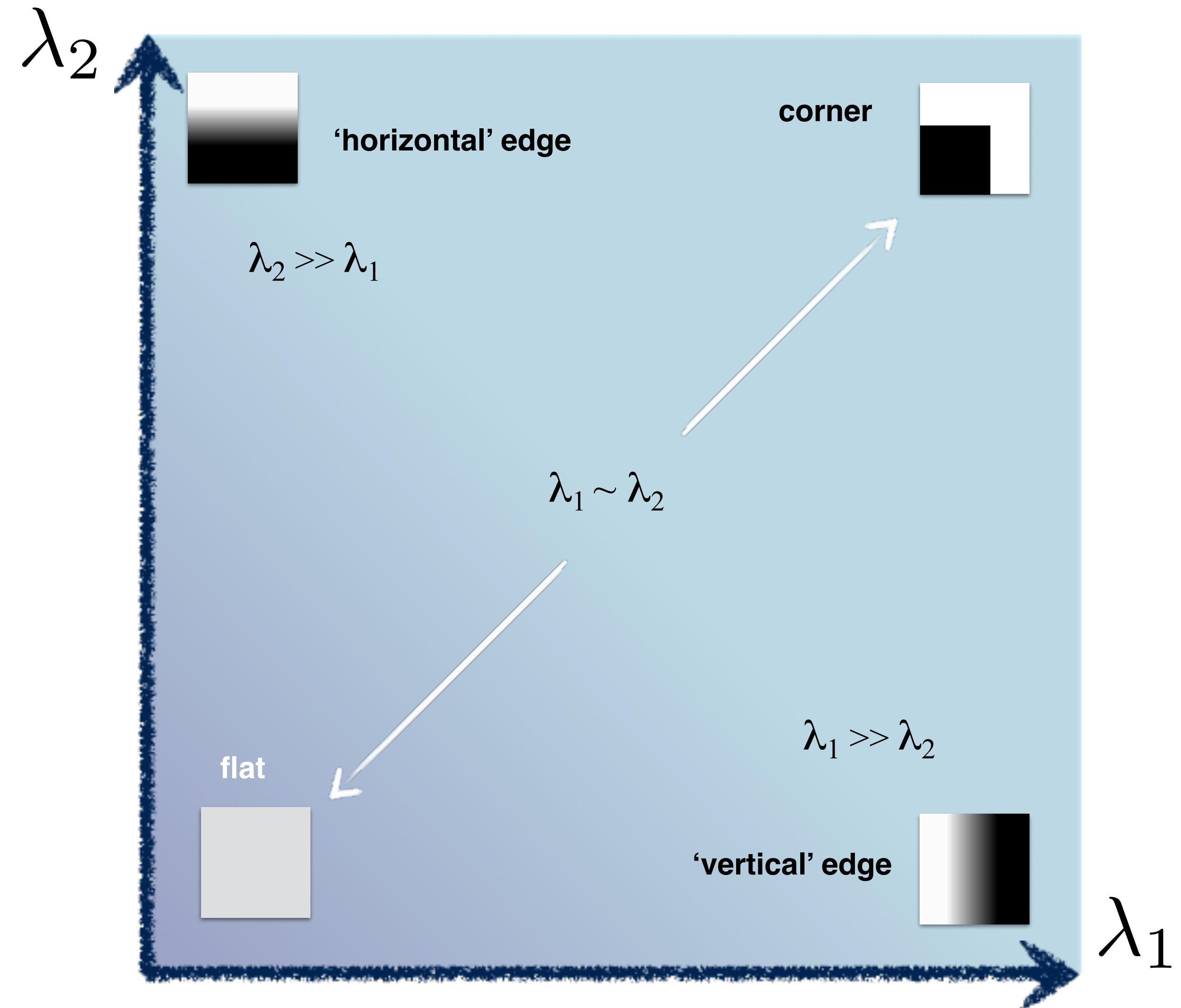


$$\begin{bmatrix} \sum_{p \in P} I_x I_x & \sum_{p \in P} I_x I_y \\ \sum_{p \in P} I_y I_x & \sum_{p \in P} I_y I_y \end{bmatrix}$$

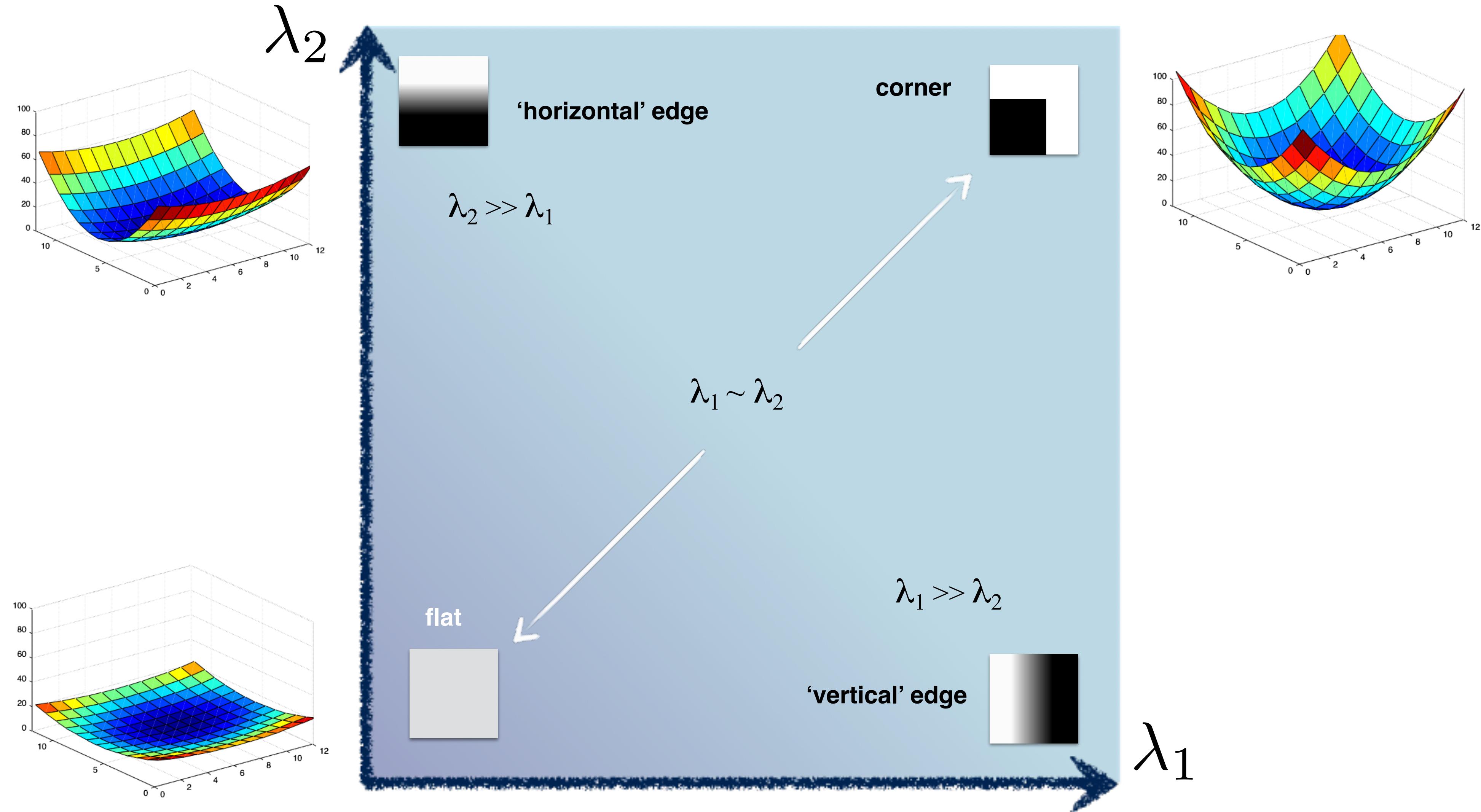
Interpreting Eigenvalues



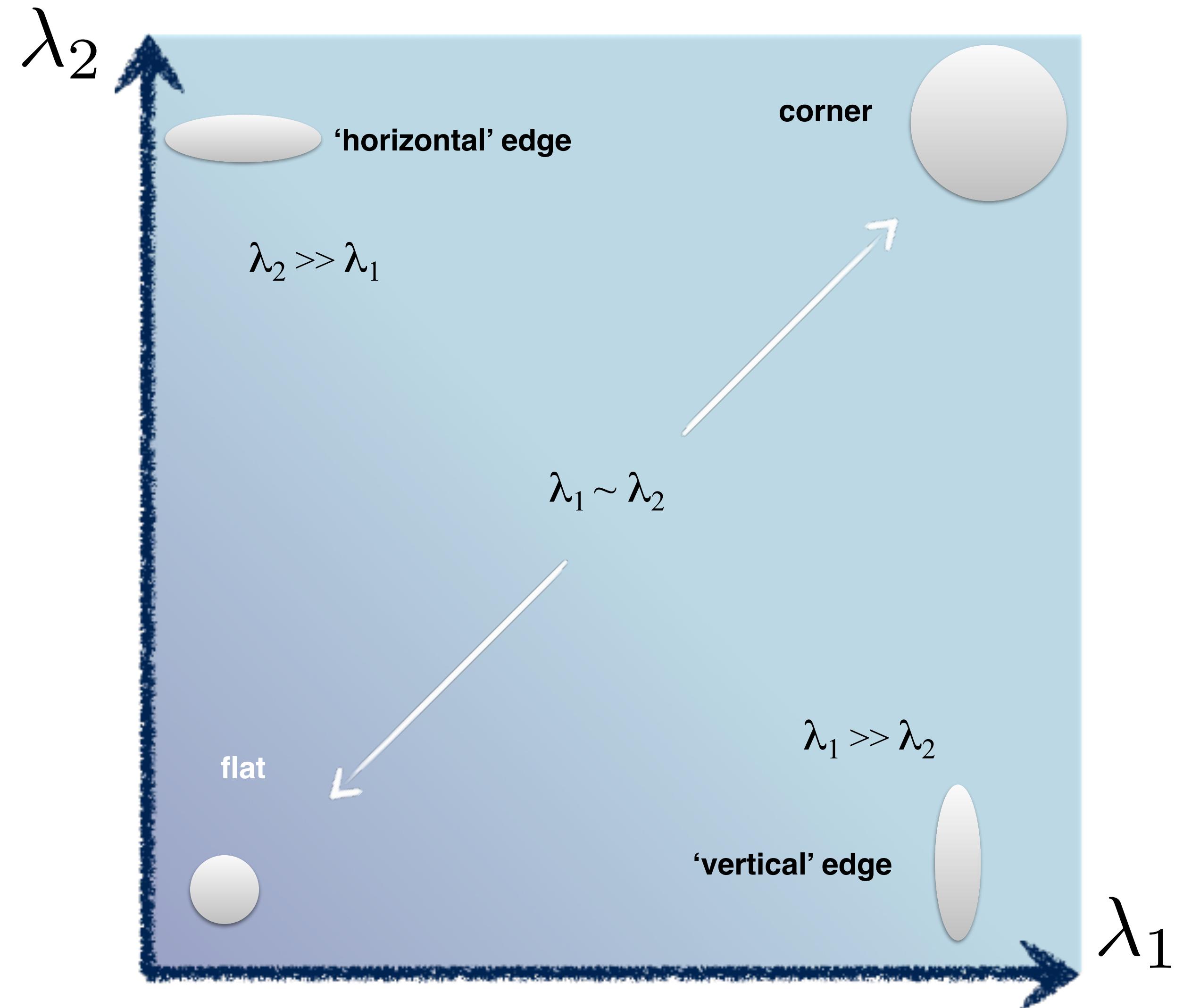
Interpreting Eigenvalues



Interpreting Eigenvalues



Interpreting Eigenvalues



Threshold on Eigenvalues to Detect Corners

(λ a function of)

Harris & Stephens (1988)

$$\det(C) - \kappa \text{trace}^2(C)$$

Kanade & Tomasi (1994)

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

Nobel (1998)

$$\frac{\det(C)}{\text{trace}(C) + \epsilon}$$



10.3

Example 1: Wagon Wheel (Harris Results)



$\sigma = 1$ (219 points)



$\sigma = 2$ (155 points)

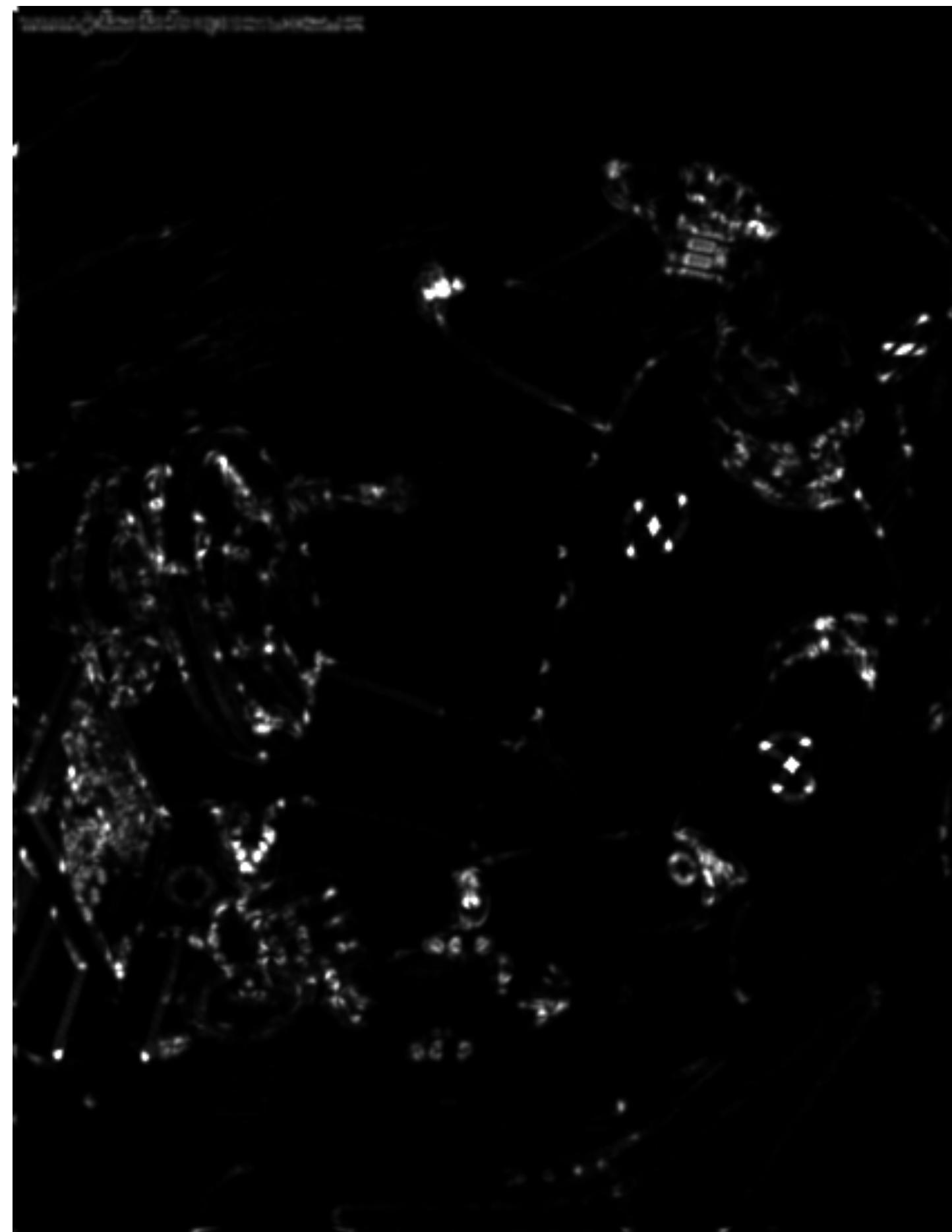


$\sigma = 3$ (110 points)

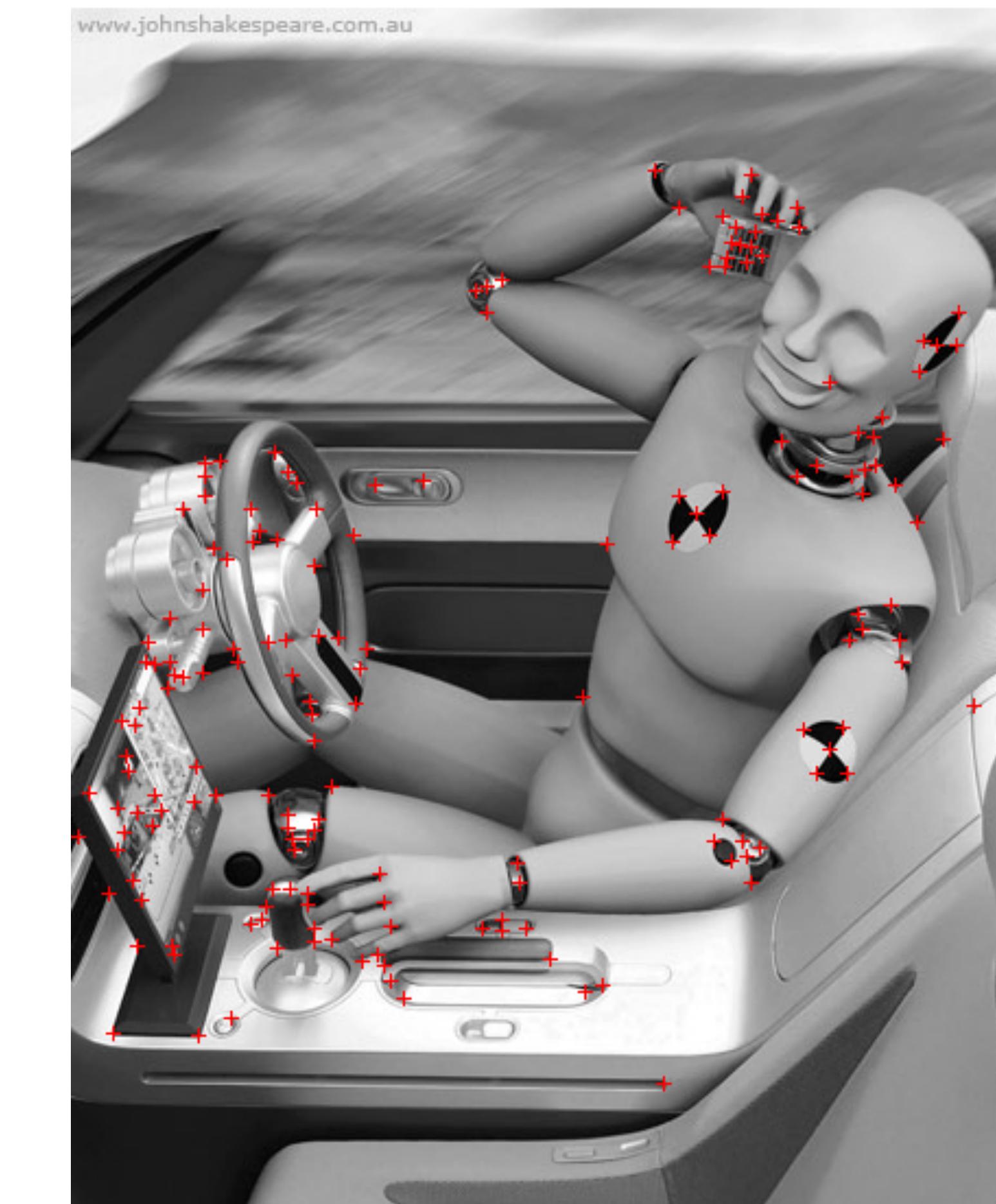


$\sigma = 4$ (87 points)

Example 2: Crash Test Dummy (Harris Result)



corner response image



$\sigma = 1$ (175 points)

Original Image Credit: John Shakespeare, Sydney Morning Herald

Harris Corner Detection Review

- Filter image with **Gaussian**
- Compute magnitude of the x and y **gradients** at each pixel
- Construct C in a window around each pixel
 - Harris uses a **Gaussian window**
- Compute Harris corner strength function $\det(C) - \kappa \text{trace}^2(C)$
- Threshold corner strength function, optionally apply non-maximal suppression

Example: Harris Corner Detection

0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	1	1	1	1	0	0
0	1	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0

Example: Harris Corner Detection

Lets compute a measure of “corner-ness” for the green pixel:

0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	1	1	1	1	0	0
0	1	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0

Example: Harris Corner Detection

Lets compute a measure of “corner-ness” for the green pixel:

0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	1	1	1	1	0	0
0	1	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0

Example: Harris Corner Detection

Lets compute a measure of “corner-ness” for the green pixel:

0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0
0	1	1	1	1	0	0	0
0	1	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0

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

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

Example: Harris Corner Detection

Lets compute a measure of “corner-ness” for the green pixel:

0	0	0	0	0	0	0	0
0	1	0	0	0	0	1	0
0	1	1	1	1	1	0	0
0	1	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0

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

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

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

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

Example: Harris Corner Detection

Lets compute a measure of “corner-ness” for the green pixel:

0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0
0	1	1	1	1	0	0	0
0	1	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0

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

$$\sum \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \odot \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 1 & 0 \end{bmatrix} = 3$$

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

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

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

Example: Harris Corner Detection

Lets compute a measure of “corner-ness” for the green pixel:

0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0
0	1	1	1	1	0	0	0
0	1	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0

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

$$\mathbf{C} = \begin{bmatrix} 3 & 2 \\ 2 & 4 \end{bmatrix}$$

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

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

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

Example: Harris Corner Detection

Lets compute a measure of “corner-ness” for the green pixel:

0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0
0	1	1	1	1	0	0	0
0	1	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0

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

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

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

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

$$\mathbf{C} = \begin{bmatrix} 3 & 2 \\ 2 & 4 \end{bmatrix} \Rightarrow \lambda_1 = 1.4384; \lambda_2 = 5.5616$$

Example: Harris Corner Detection

Lets compute a measure of “corner-ness” for the green pixel:

0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0
0	1	1	1	1	0	0	0
0	1	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0

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

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

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

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

$$\mathbf{C} = \begin{bmatrix} 3 & 2 \\ 2 & 4 \end{bmatrix} \Rightarrow \lambda_1 = 1.4384; \lambda_2 = 5.5616$$

$$\det(\mathbf{C}) - 0.04\text{trace}^2(\mathbf{C}) = 6.04$$

Example: Harris Corner Detection

Lets compute a measure of “corner-ness” for the green pixel:

0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	1	1	1	1	0	0
0	1	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0

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

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

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

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

$$\mathbf{C} = \begin{bmatrix} 3 & 0 \\ 0 & 0 \end{bmatrix} \Rightarrow \lambda_1 = 3; \lambda_2 = 0$$

$$\det(\mathbf{C}) - 0.04\text{trace}^2(\mathbf{C}) = -0.36$$

Example: Harris Corner Detection

Lets compute a measure of “corner-ness” for the green pixel:

0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	0	0
0	1	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0

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

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

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

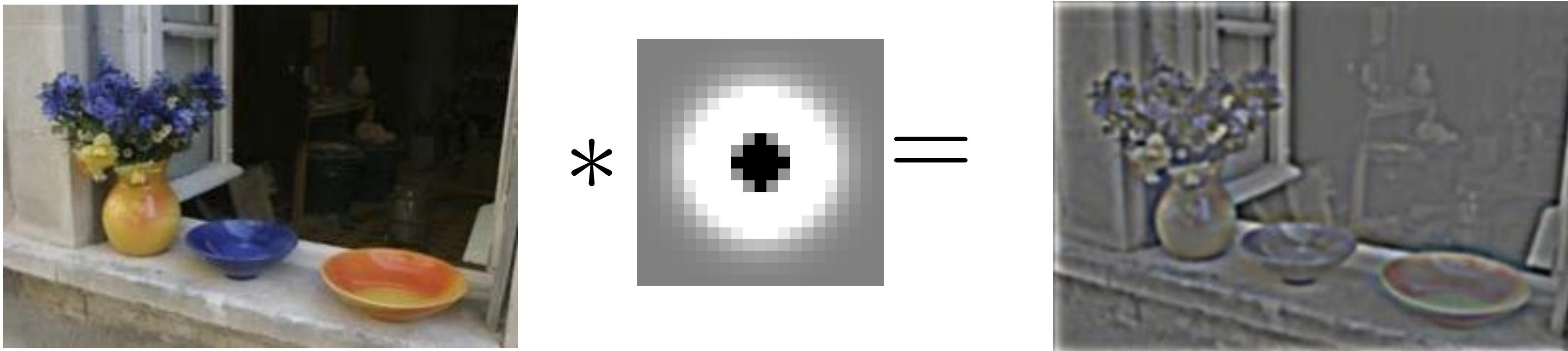
0	-1	0	0	0	-1	0
0	0	-1	-1	-1	1	0
0	0	0	0	0	0	0
0	1	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

$$\mathbf{C} = \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix} \Rightarrow \lambda_1 = 3; \lambda_2 = 2$$

$$\det(\mathbf{C}) - 0.04\text{trace}^2(\mathbf{C}) = 5$$

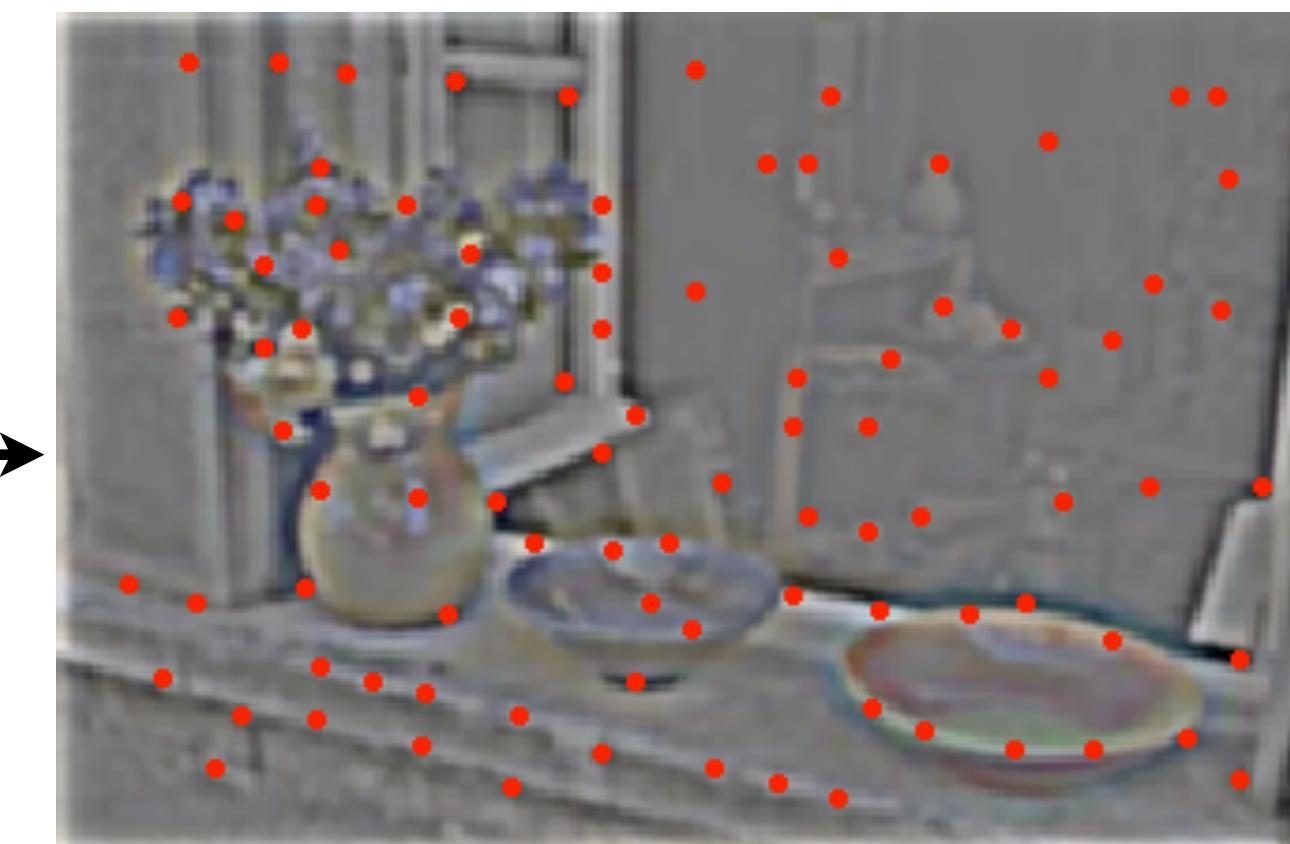
Difference of Gaussian

- DoG = centre-surround filter



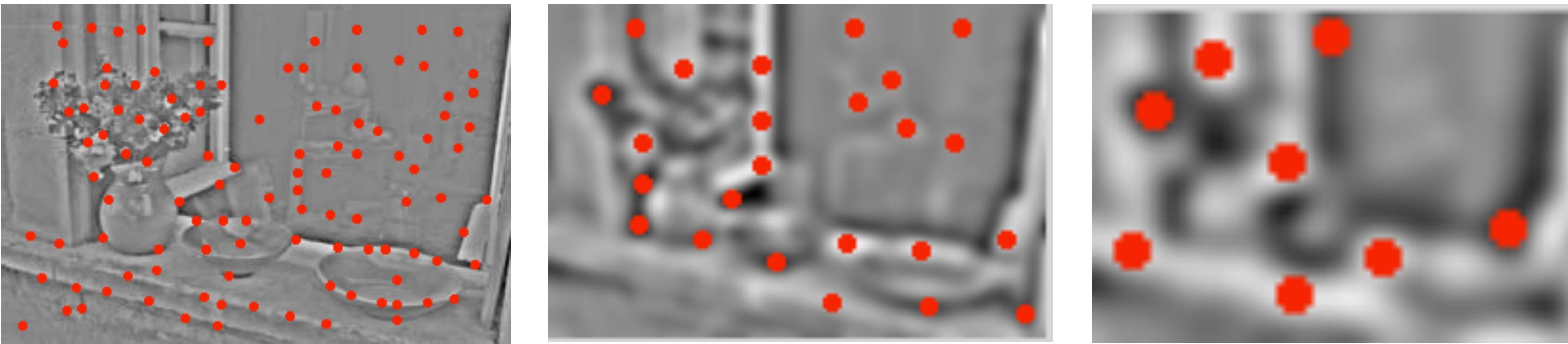
- Find local-maxima of the centre surround response

Non-maximal suppression:
These points are maxima in
a 10 pixel radius



Difference of Gaussian

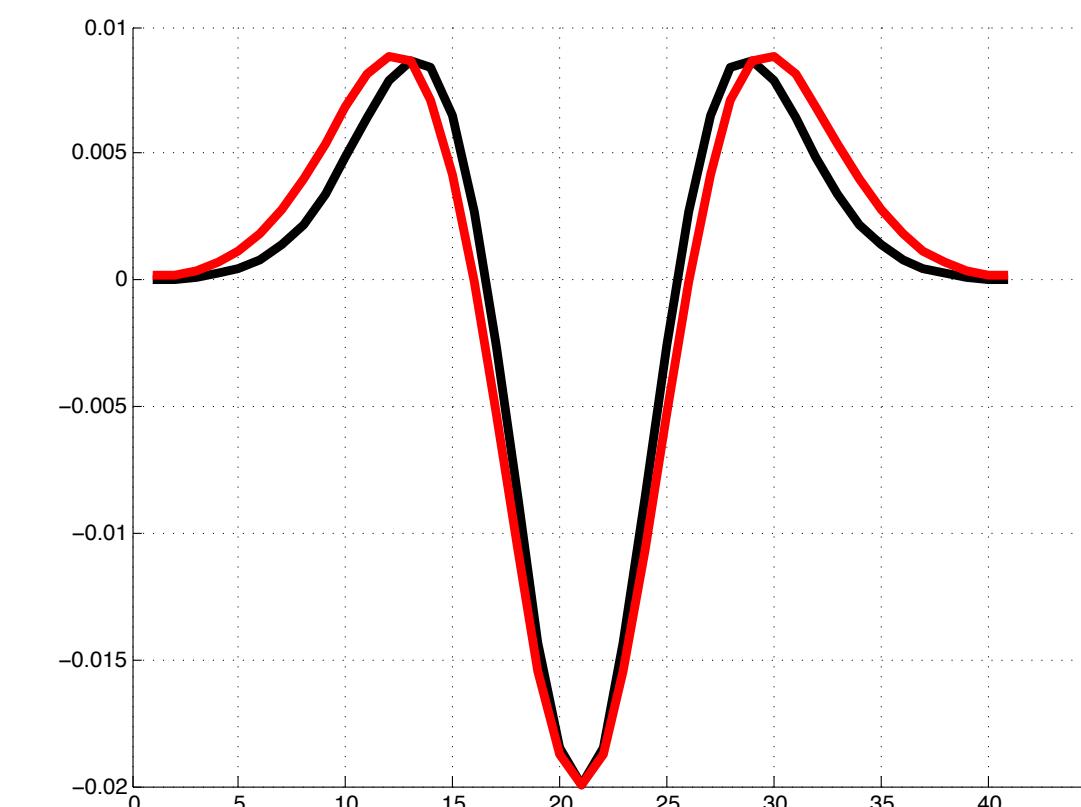
- DoG detects blobs at scale that depends on the Gaussian standard deviation(s)



Note: $\text{DOG} \approx \text{Laplacian of Gaussian}$

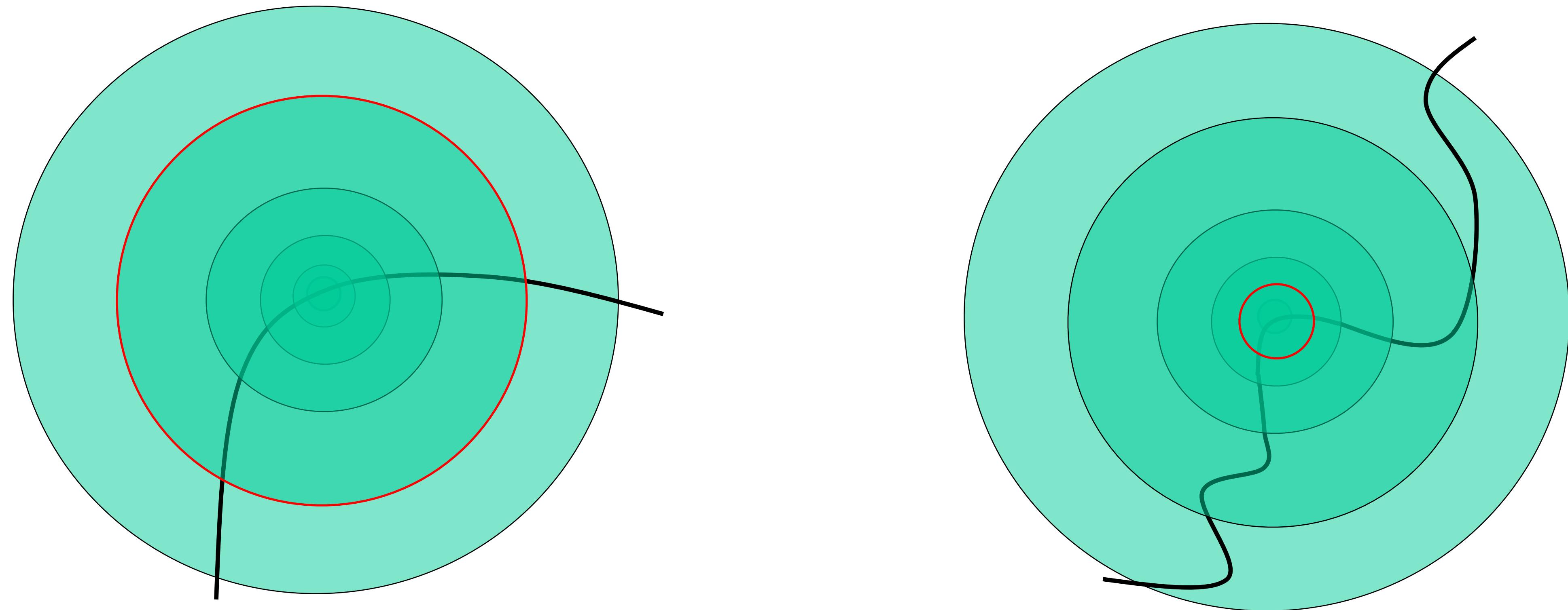
$$\text{red} = [1 \ -2 \ 1] * g(x; 5.0)$$

$$\text{black} = g(x; 5.0) - g(x; 4.0)$$



Scale Invariant Interest Point Detection

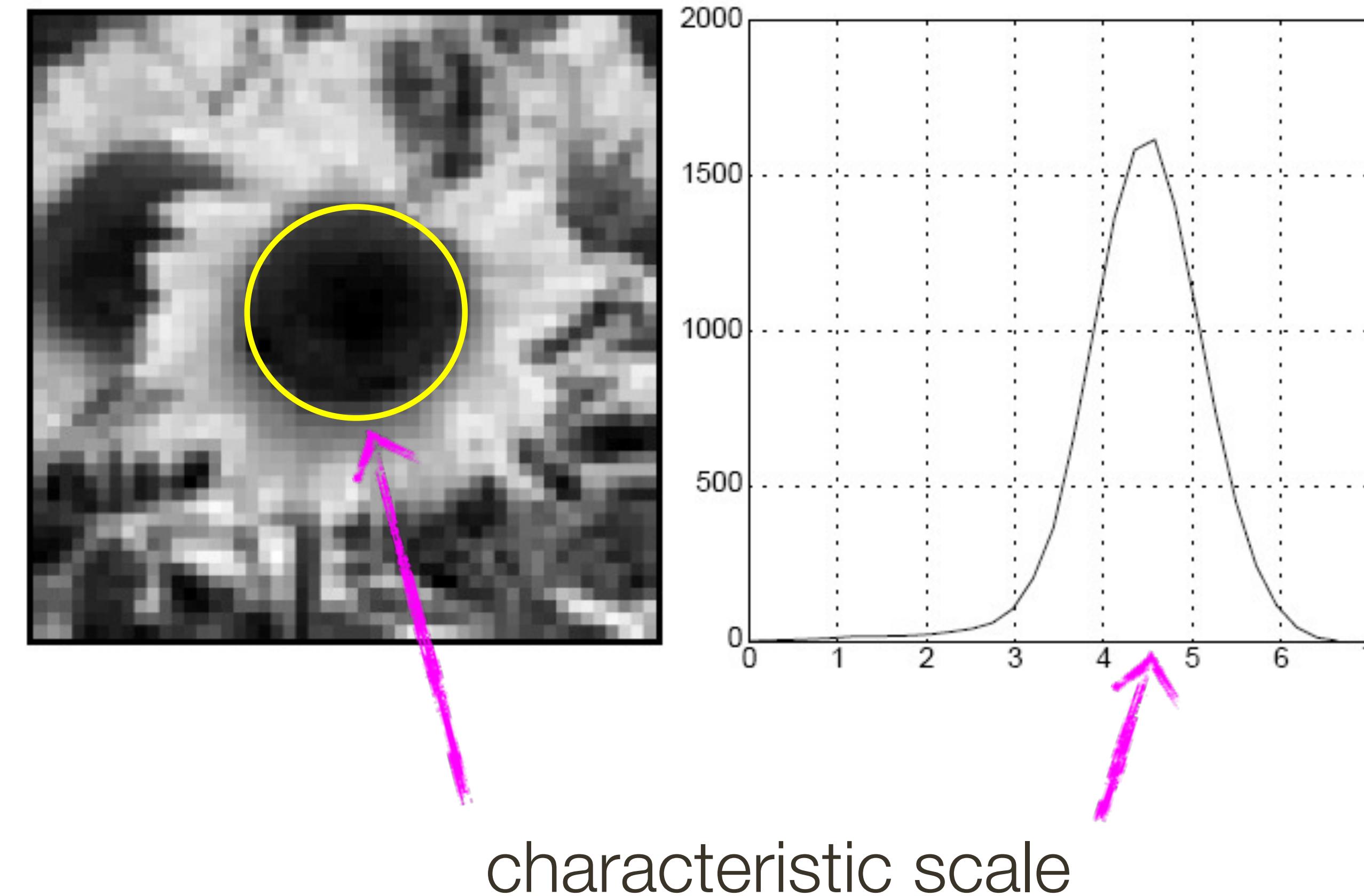
Find local maxima in both **position** and **scale**





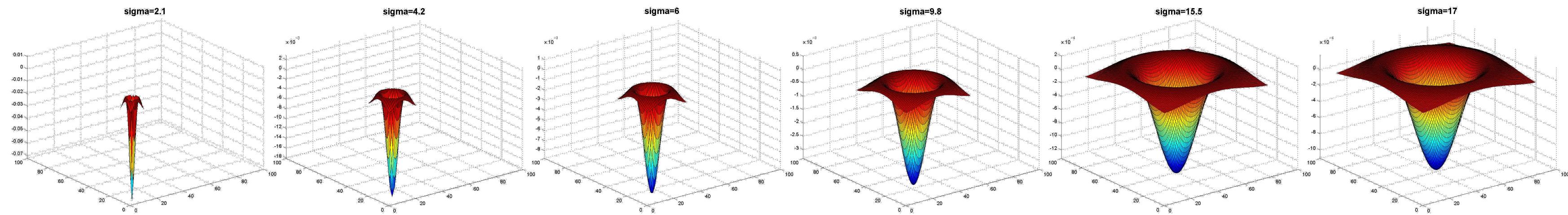
Characteristic Scale

characteristic scale - the scale that produces peak filter response



we need to search over characteristic scales

Applying Laplacian Filter at Different Scales



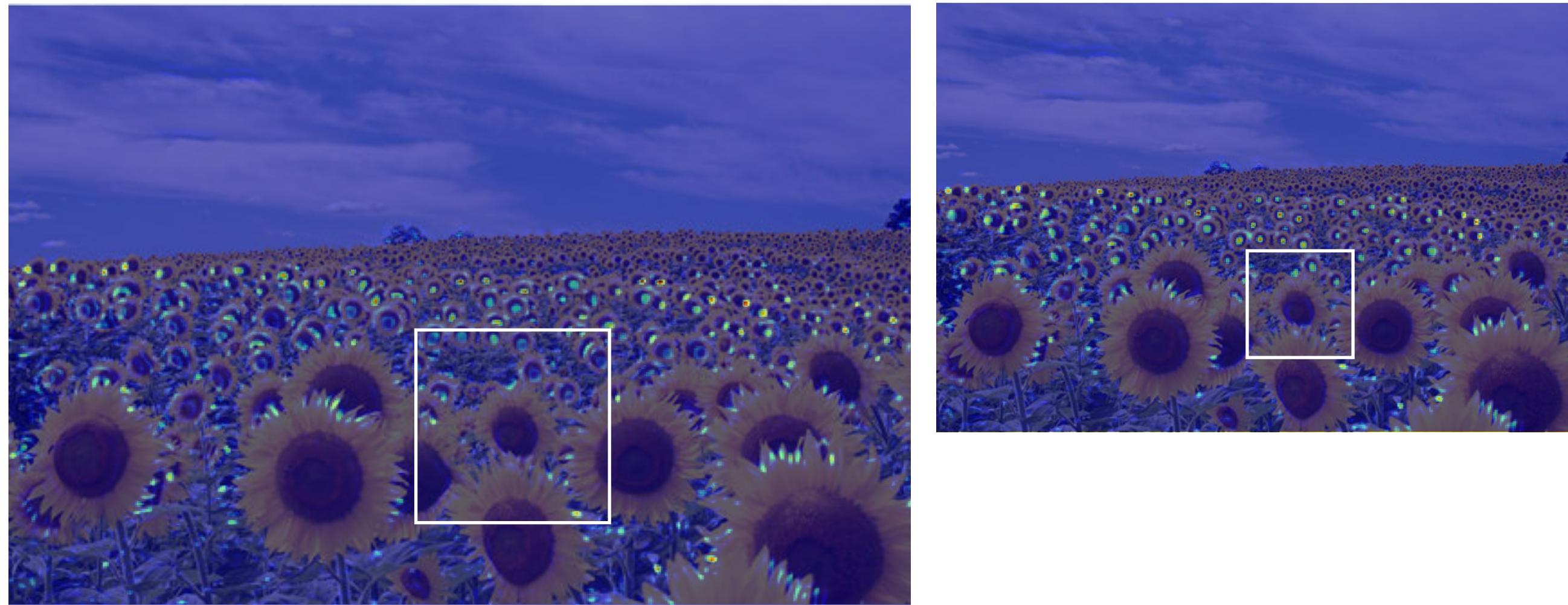
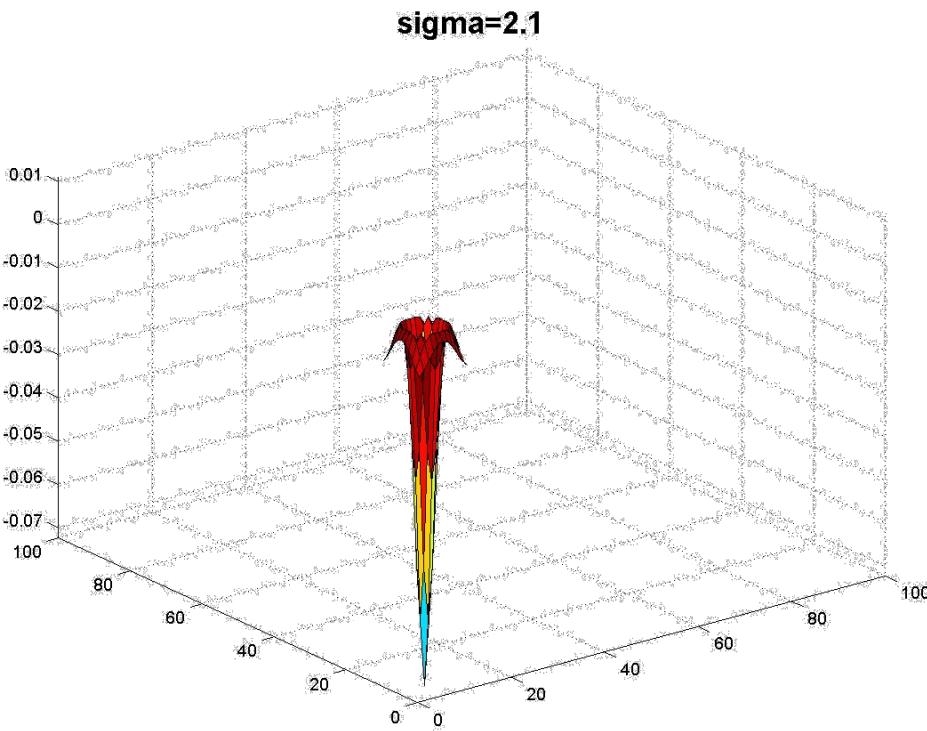
Full size



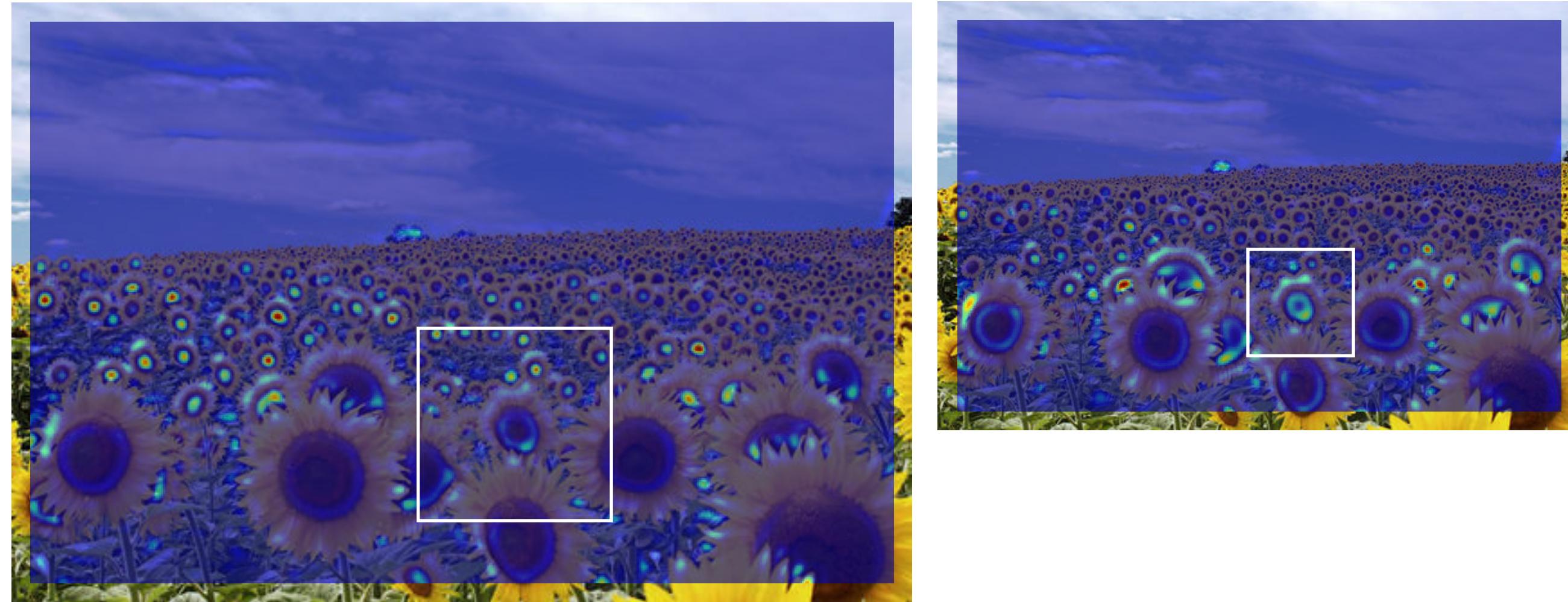
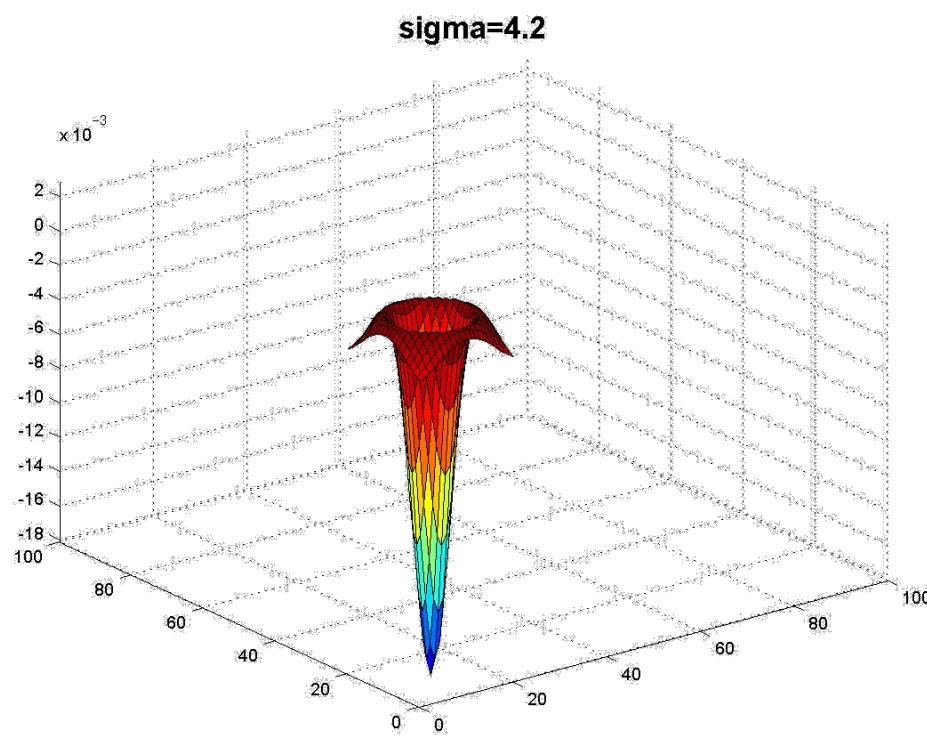
3/4 size



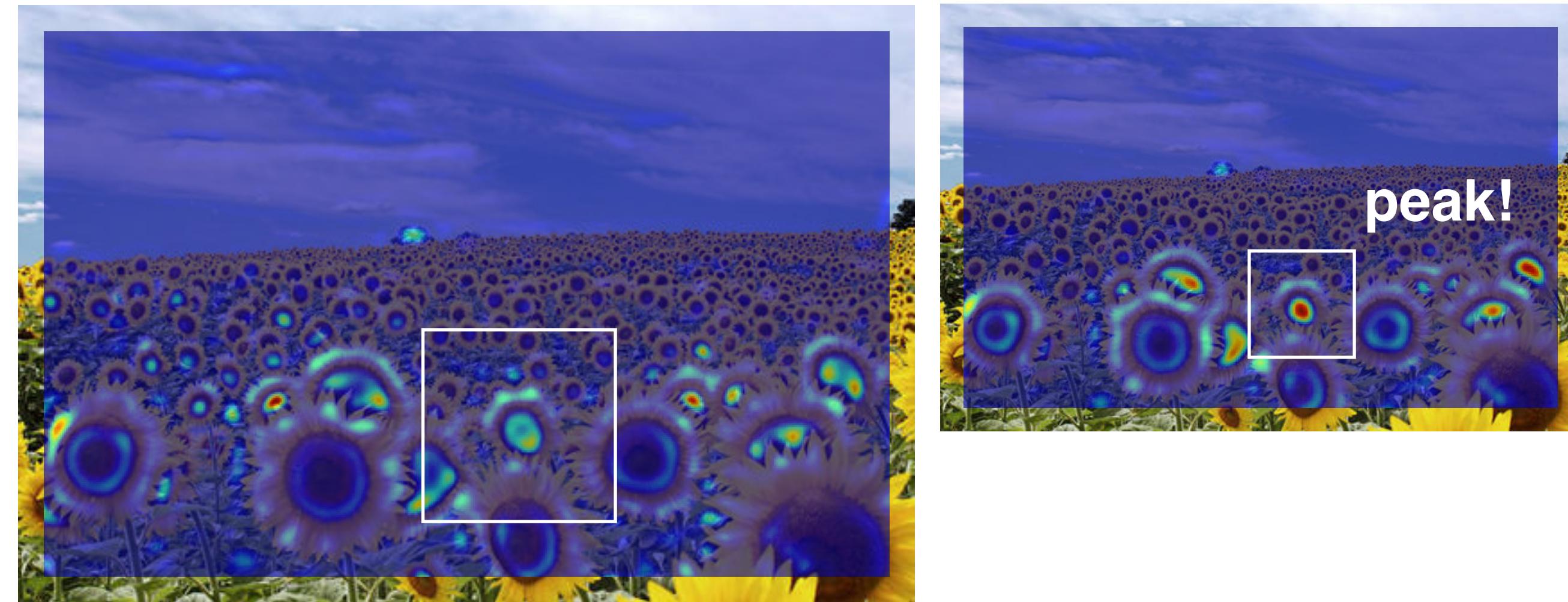
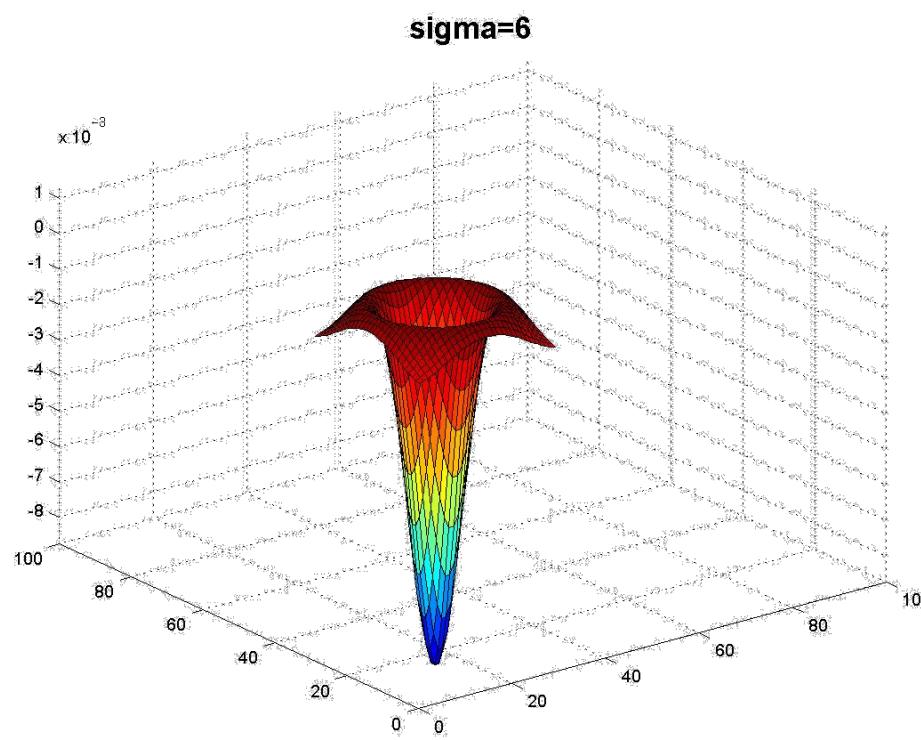
Applying Laplacian Filter at Different Scales



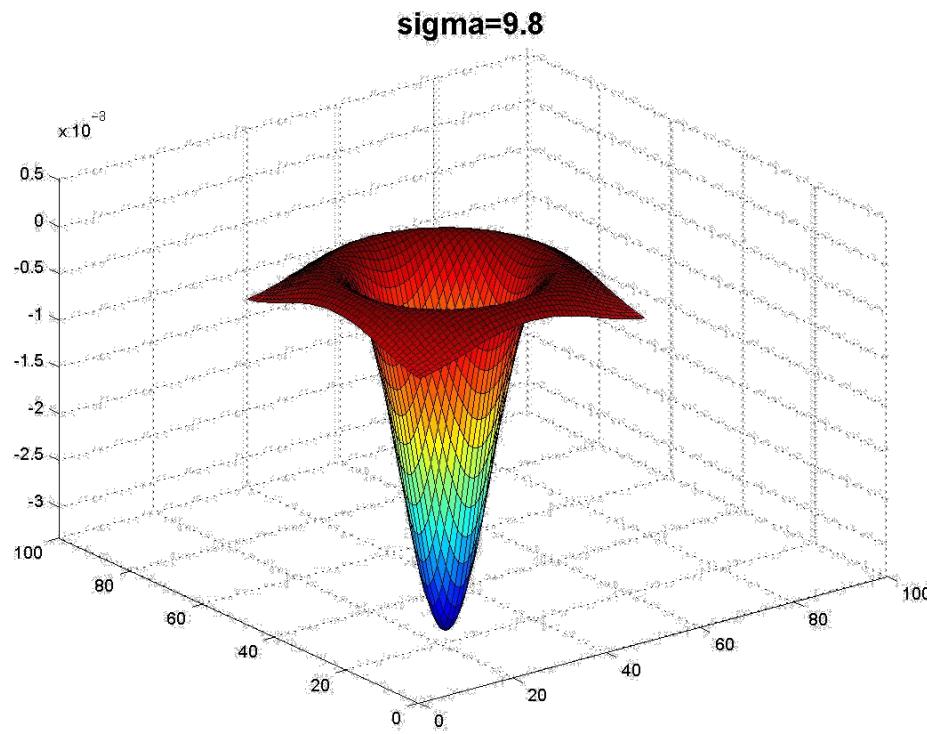
Applying Laplacian Filter at Different Scales



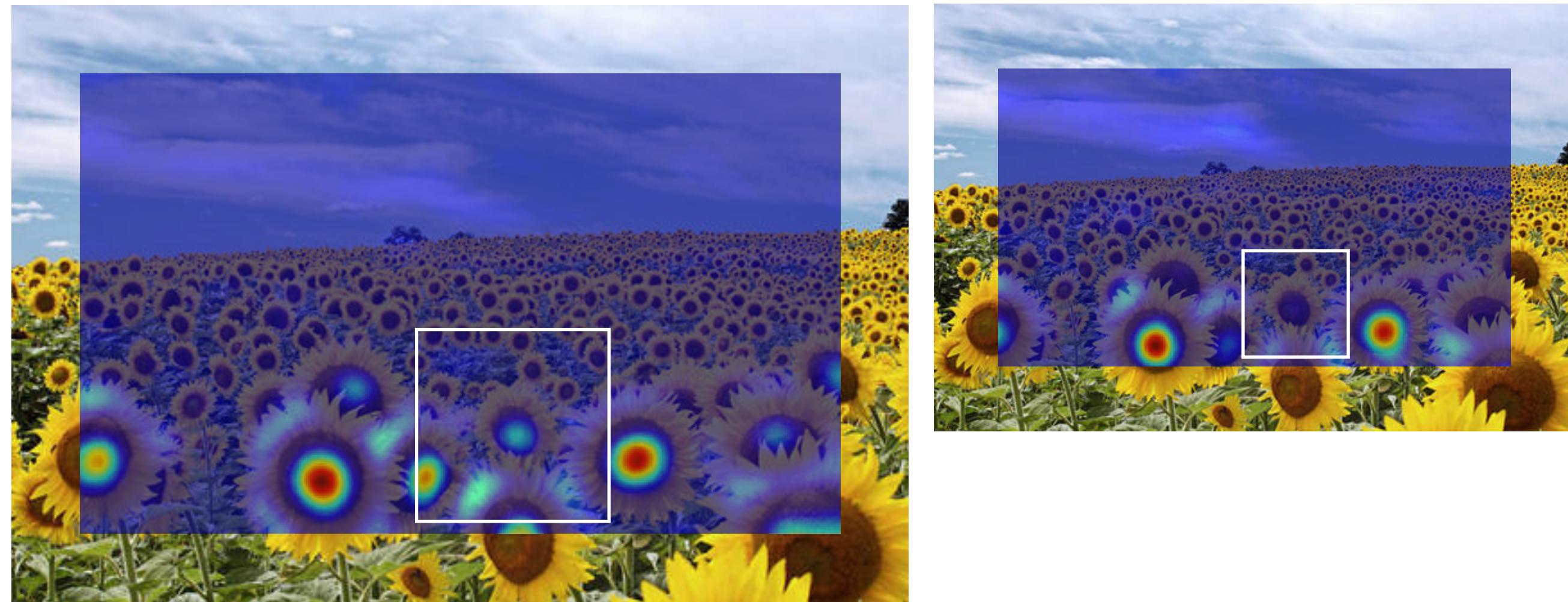
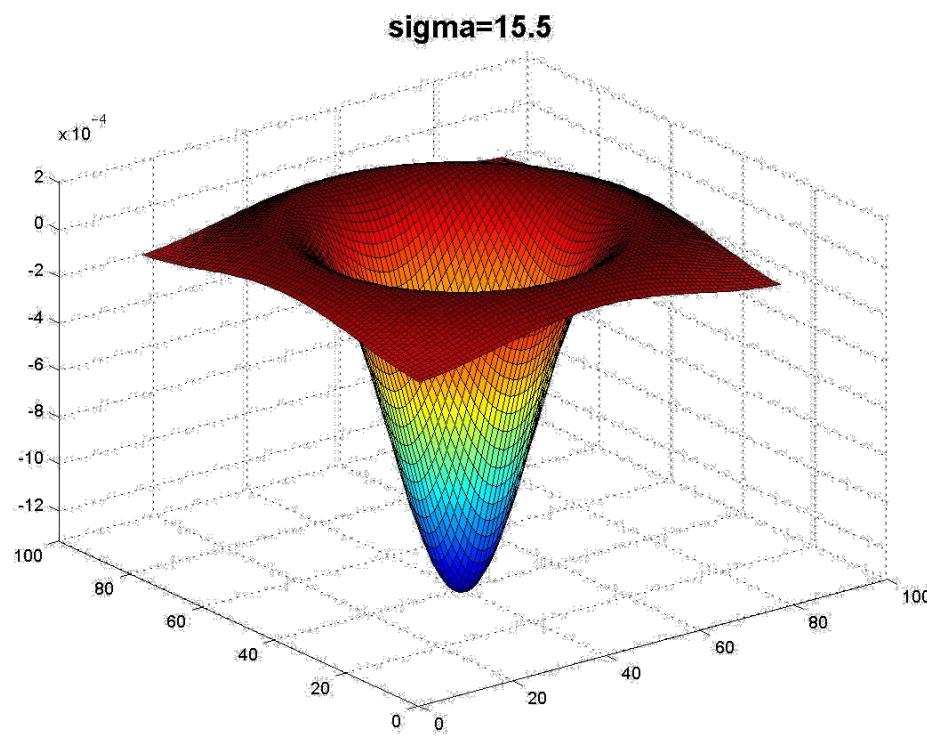
Applying Laplacian Filter at Different Scales



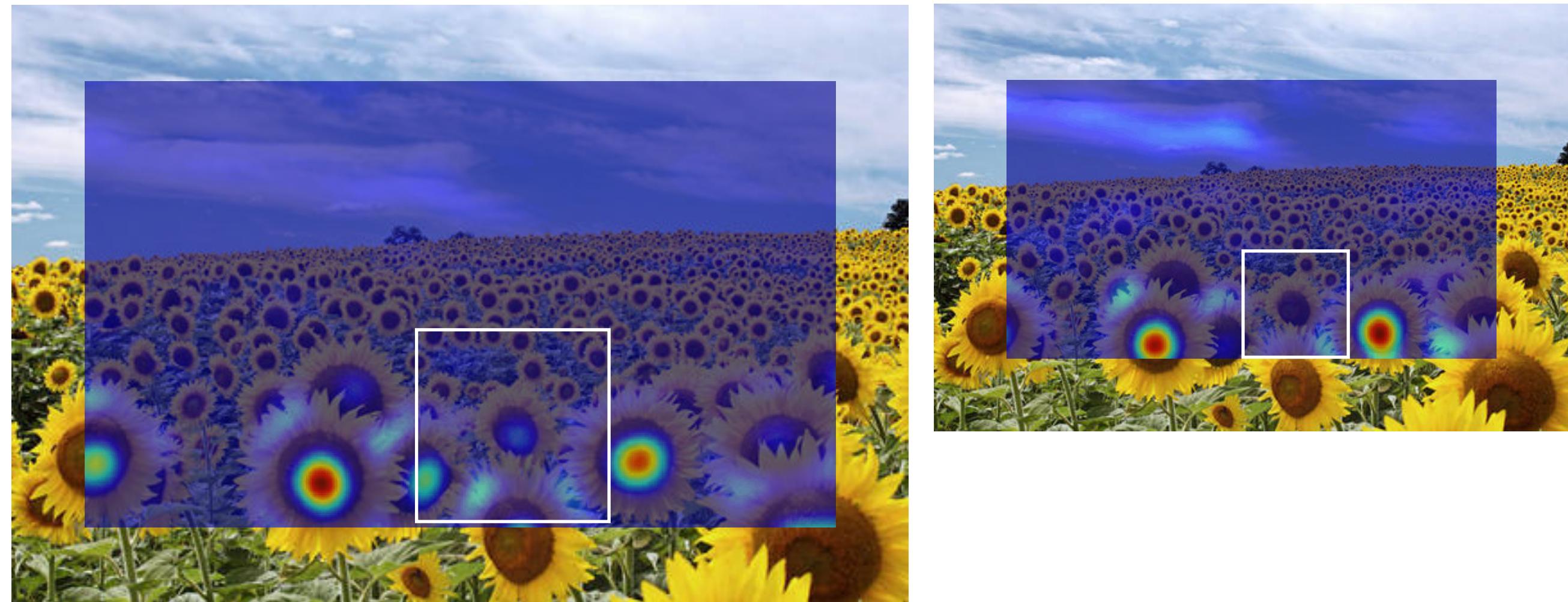
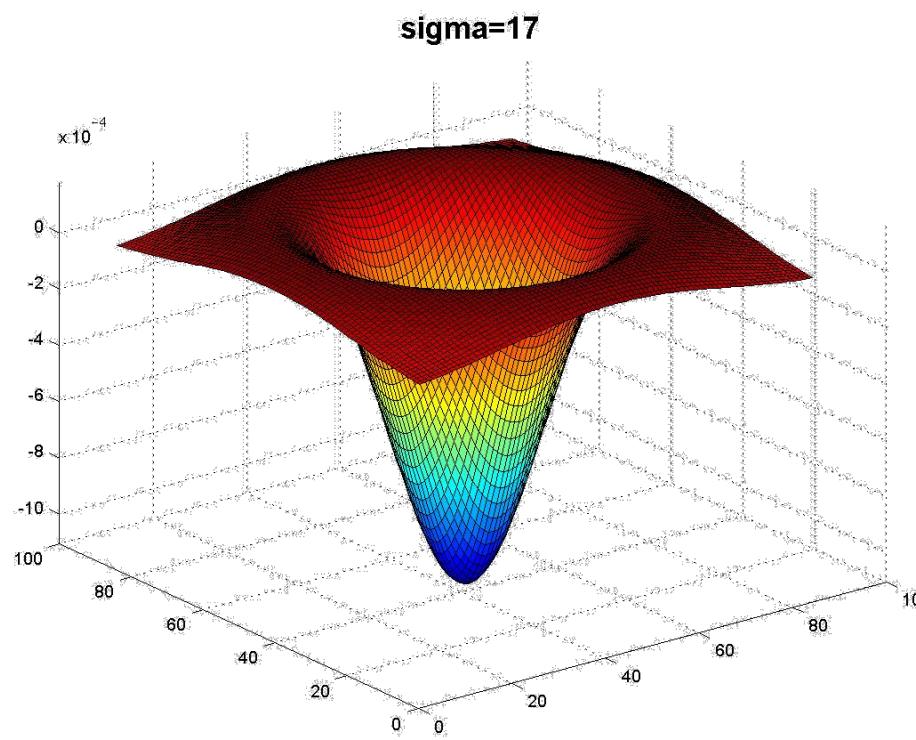
Applying Laplacian Filter at Different Scales



Applying Laplacian Filter at Different Scales

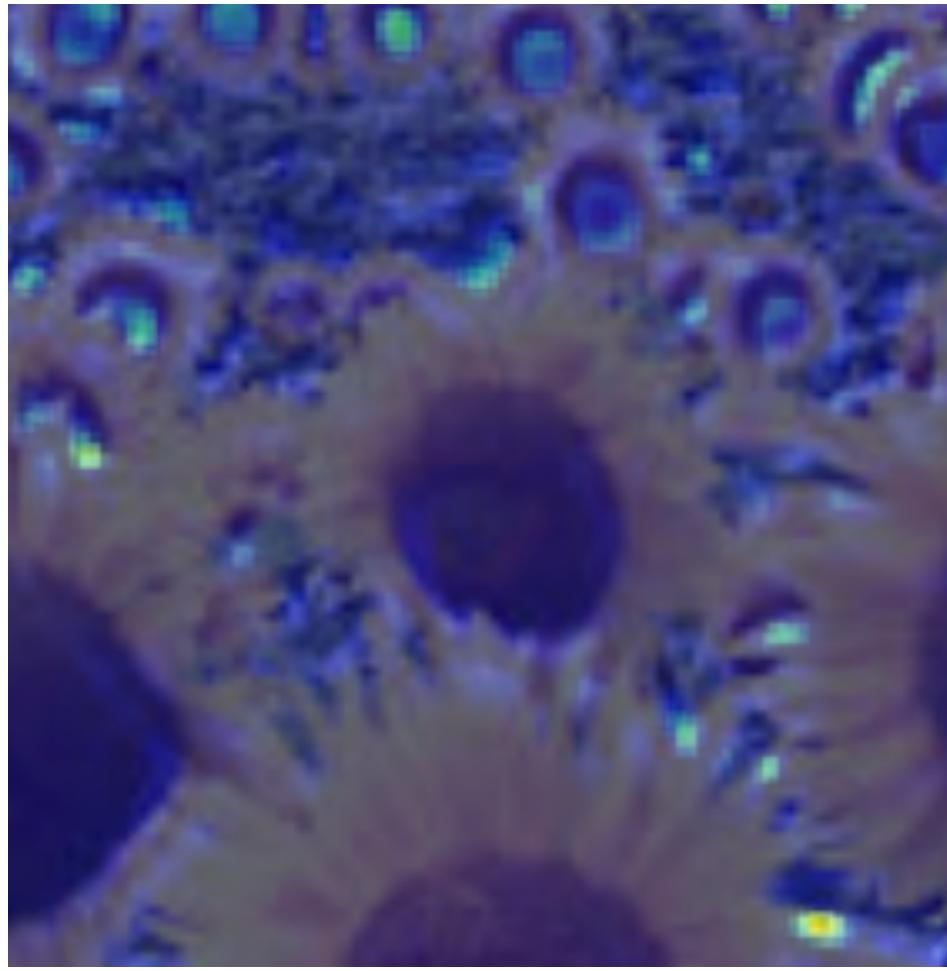


Applying Laplacian Filter at Different Scales

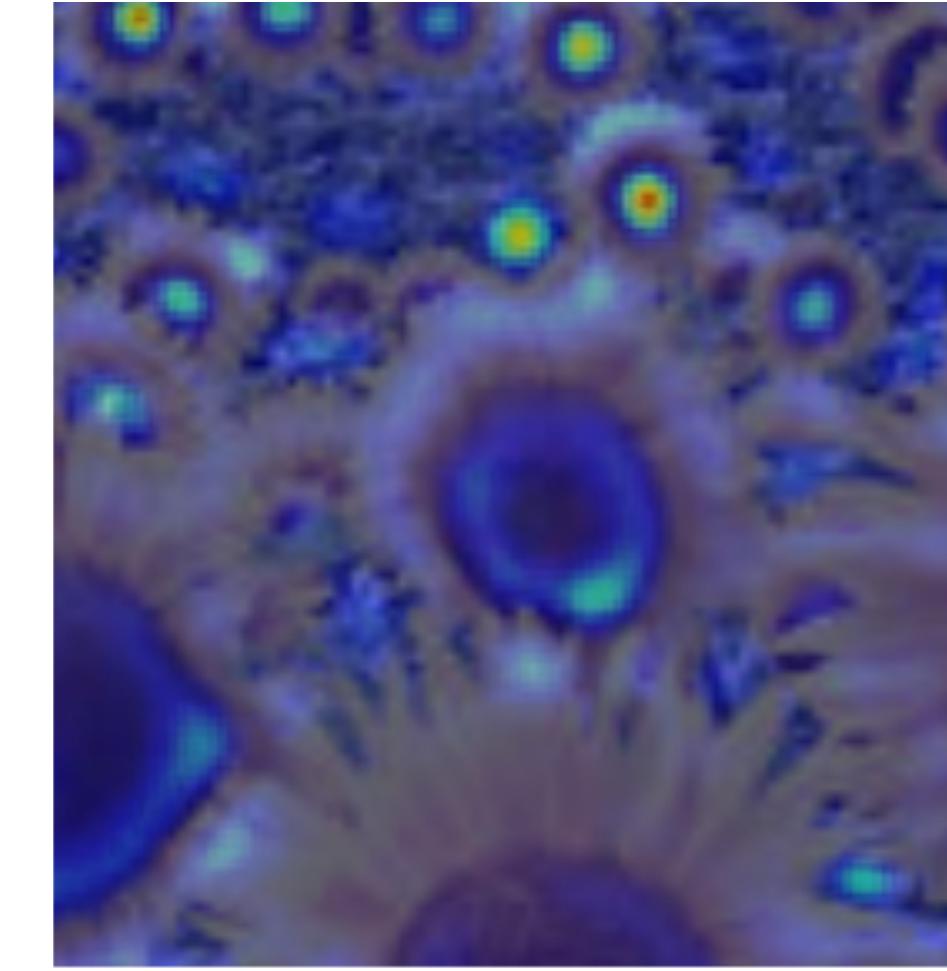


Applying Laplacian Filter at Different Scales

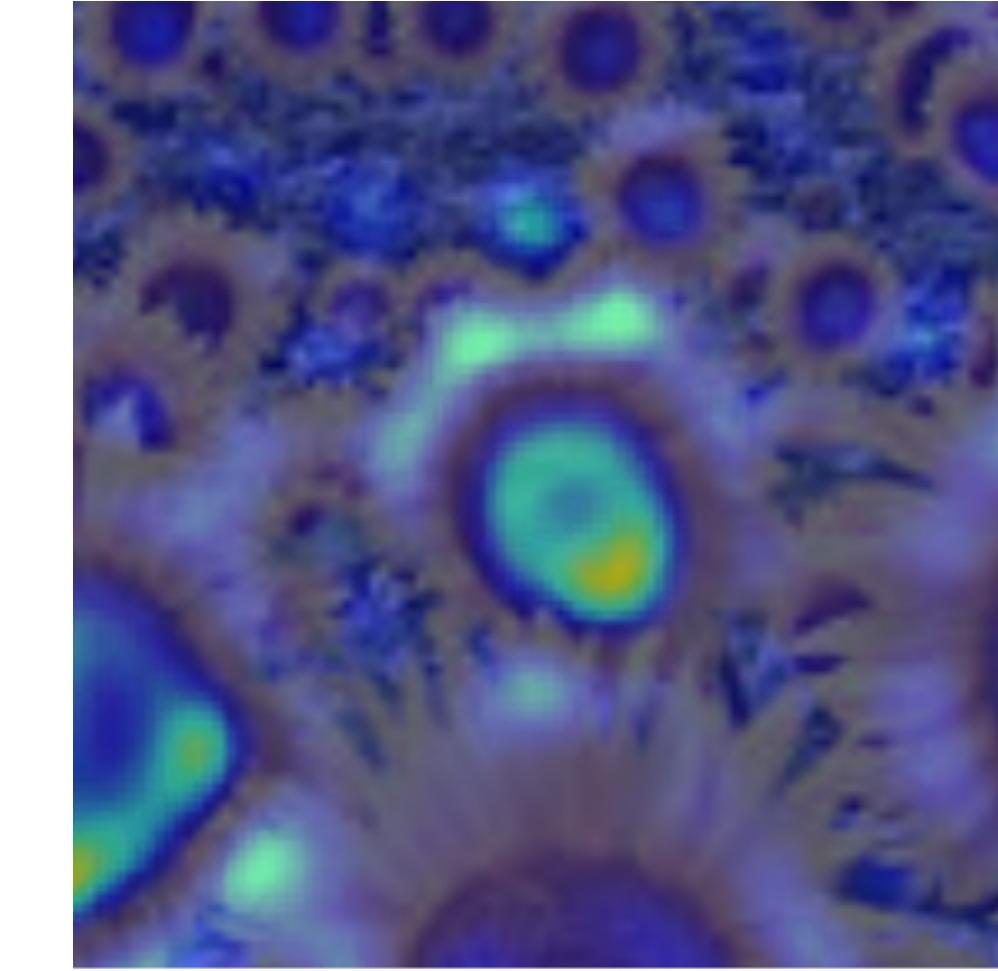
2.1



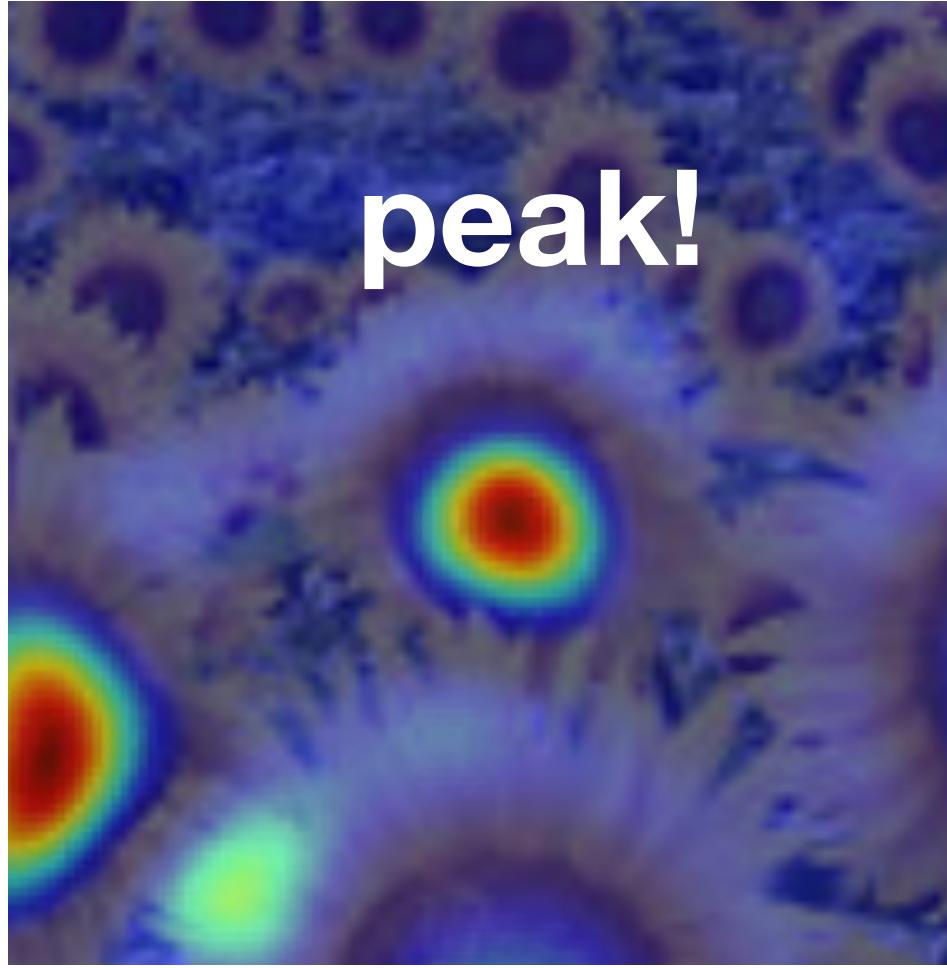
4.2



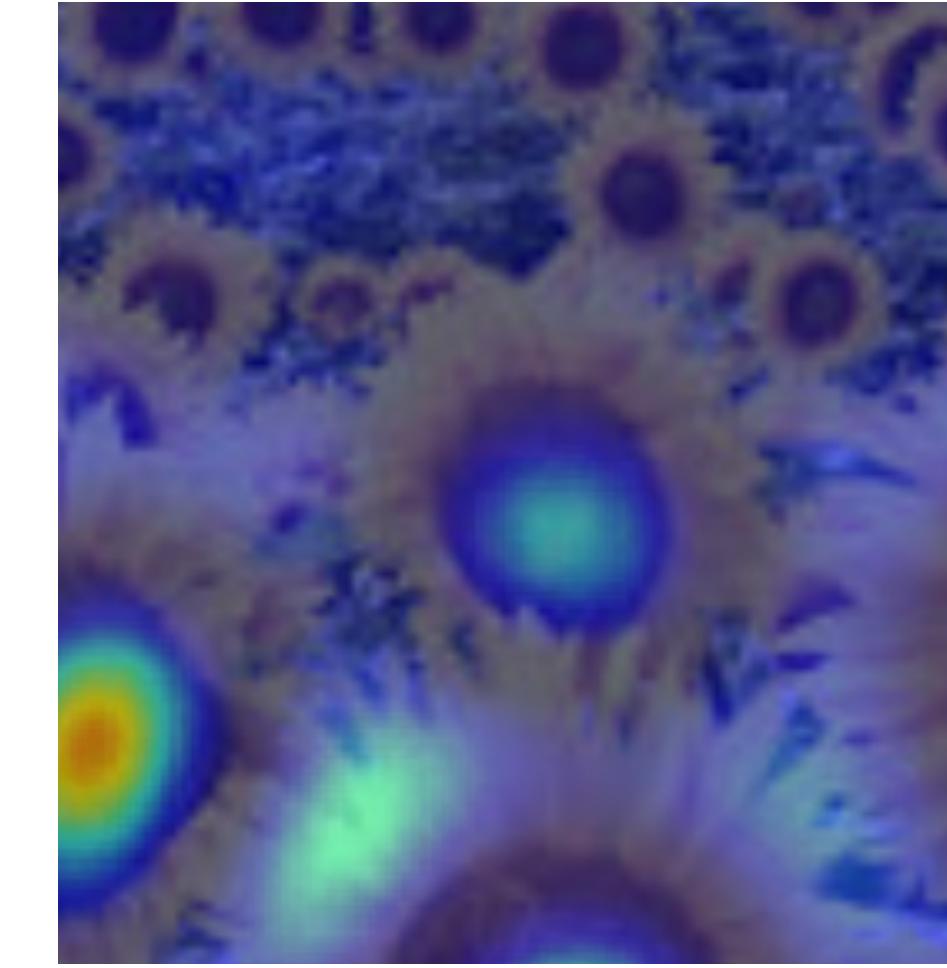
6.0



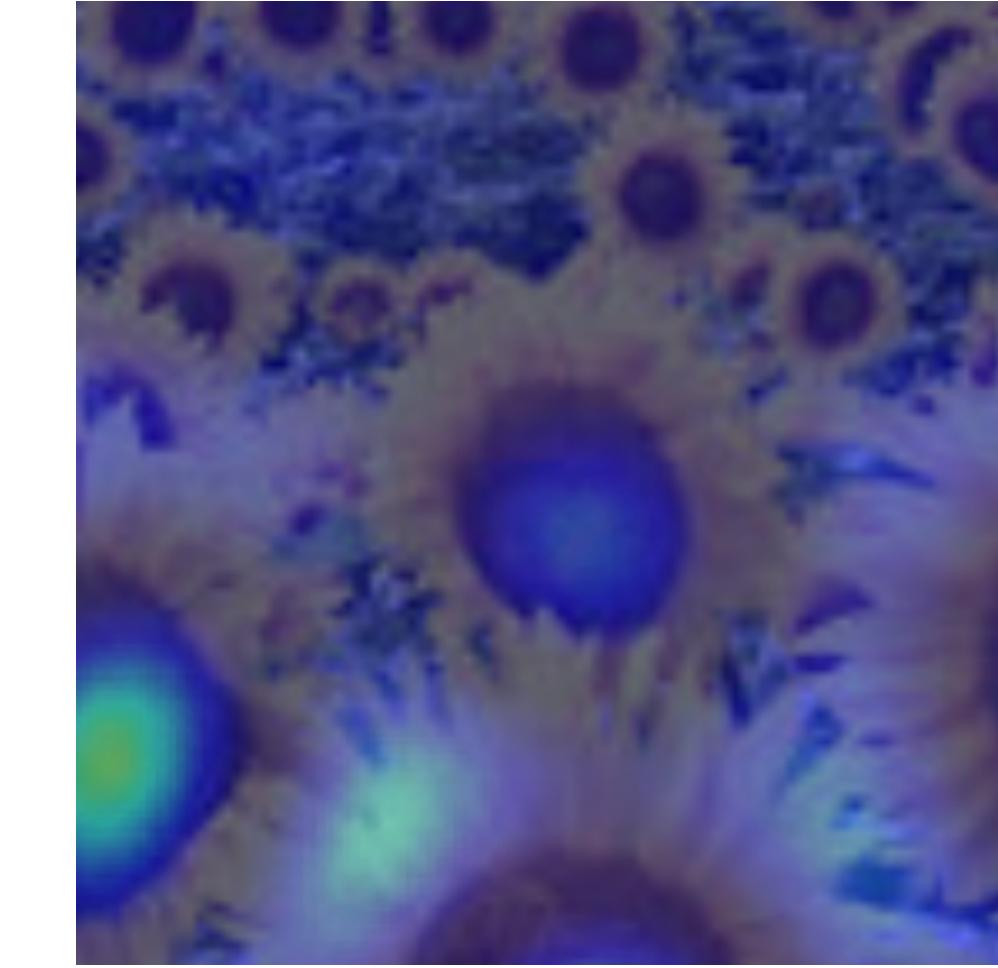
9.8



15.5

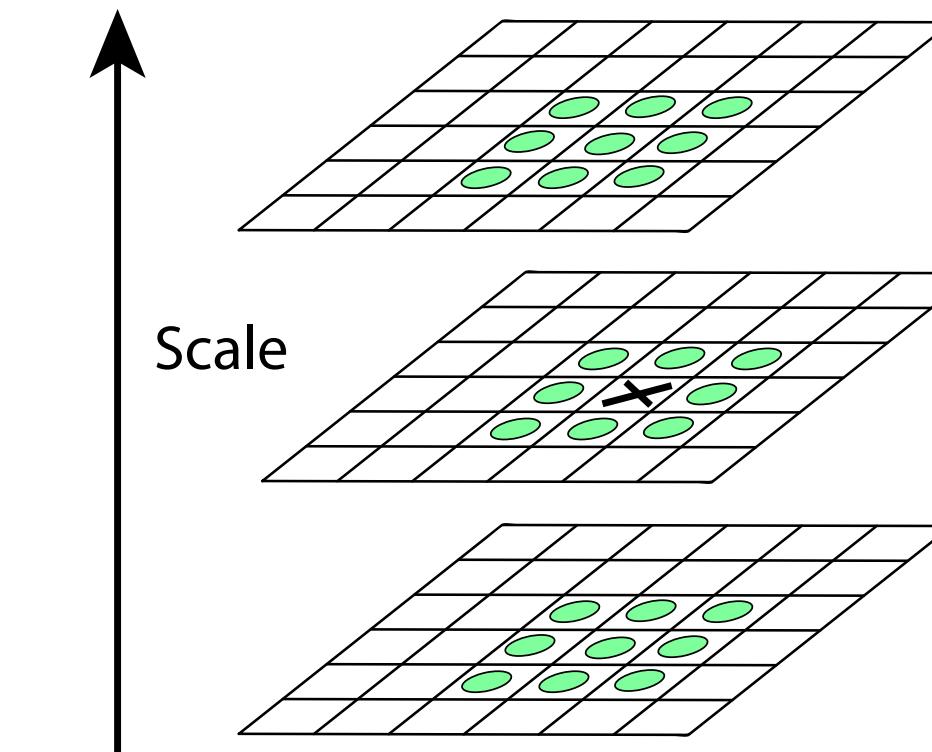
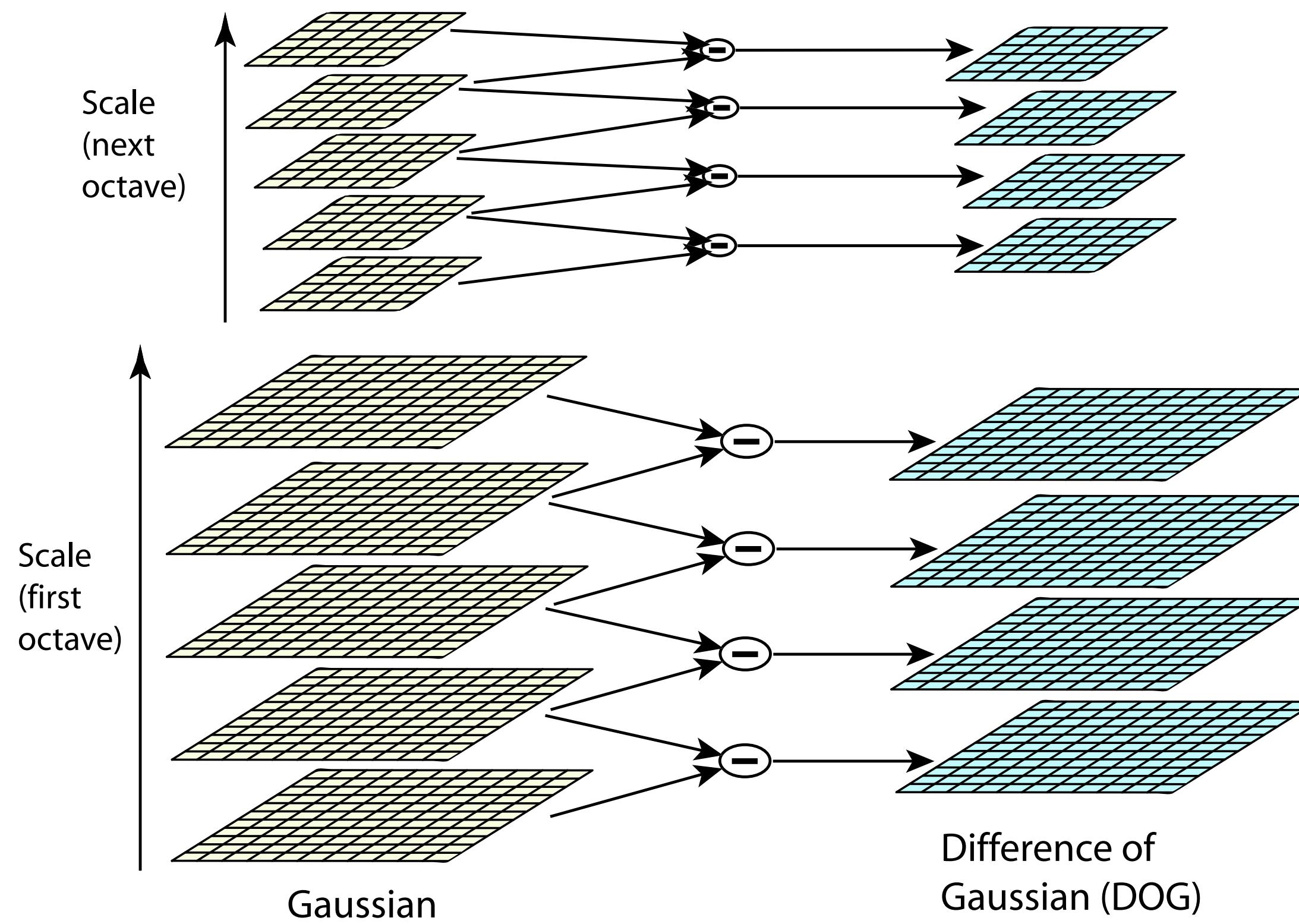


17.0



Scale Selection

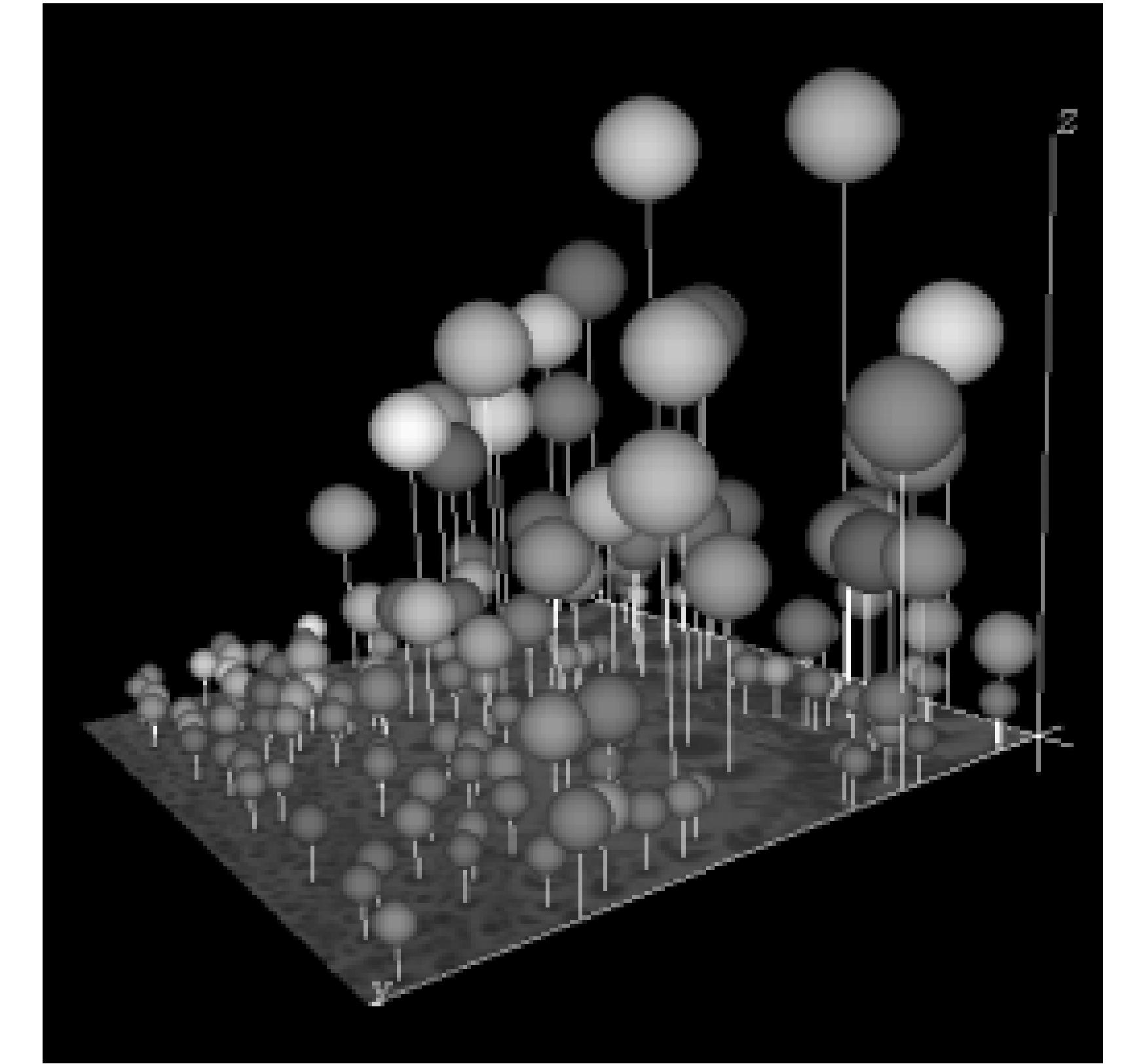
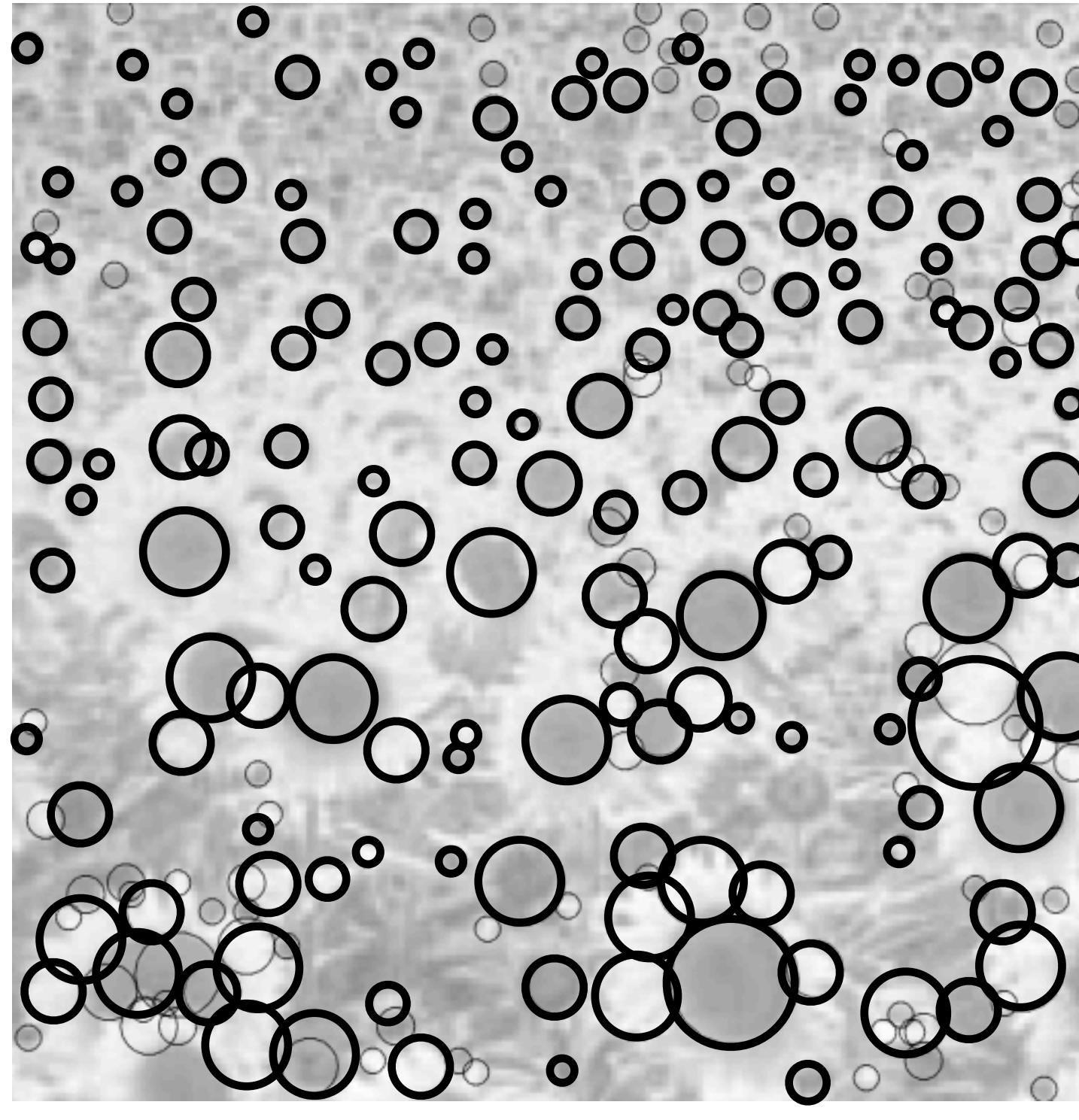
- A DOG (Laplacian) Pyramid is formed with multiple scales per octave



Detections are local
maxima in a $3 \times 3 \times 3$
scale-space window

Scale Selection

- Maximising the DOG function in scale as well as space performs scale selection



Multi-Scale Harris Corners

For each level of the Gaussian pyramid

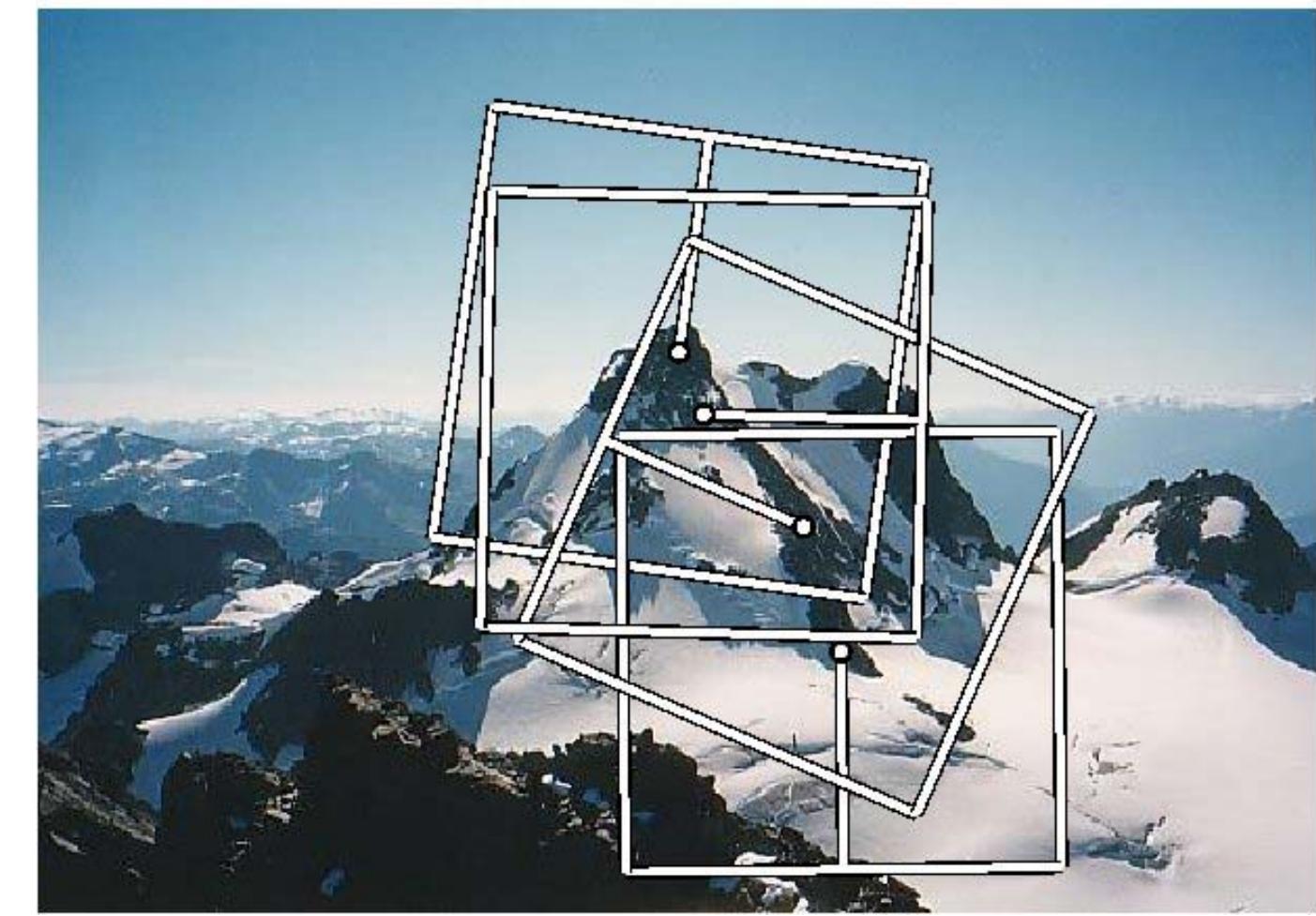
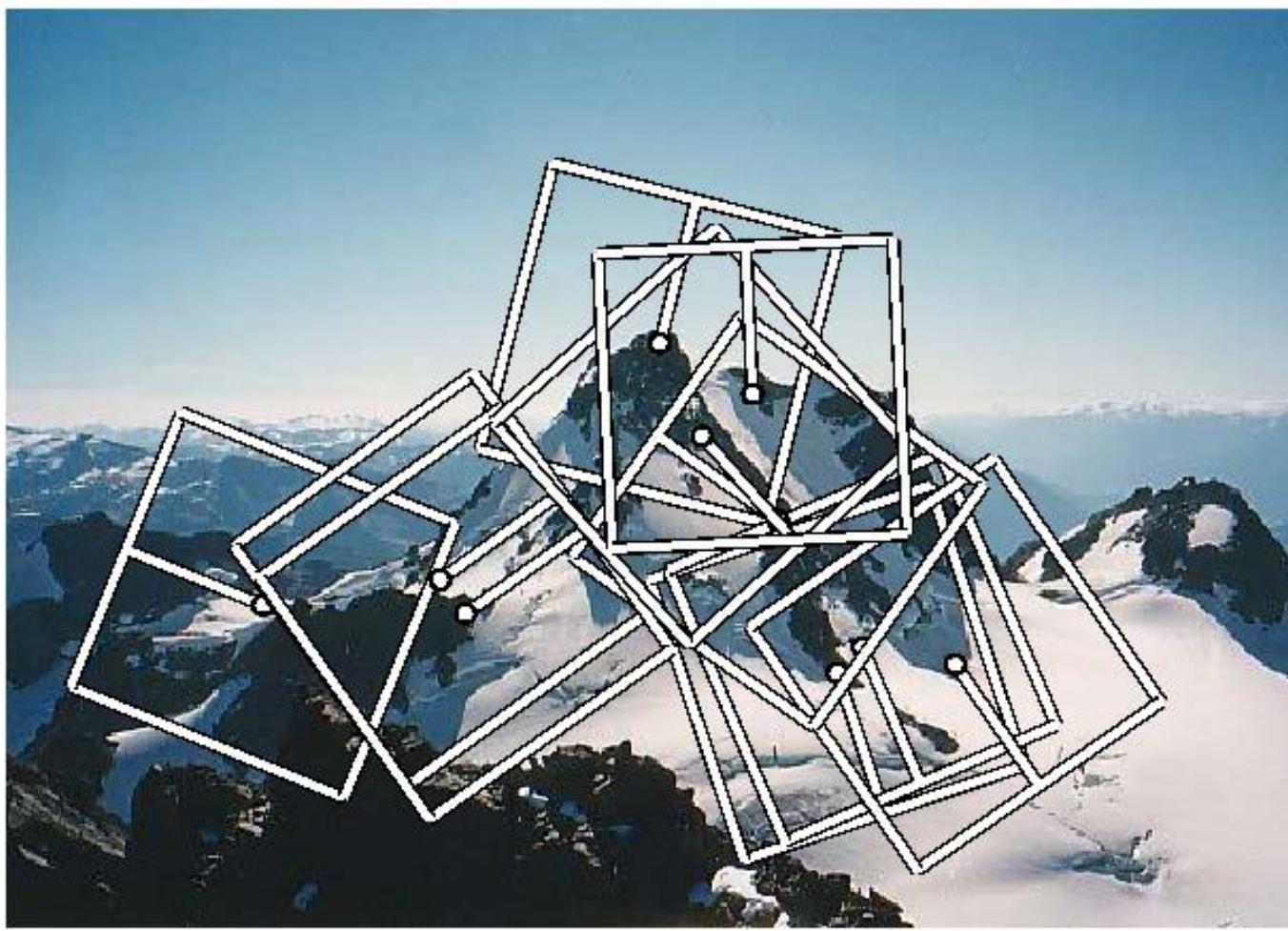
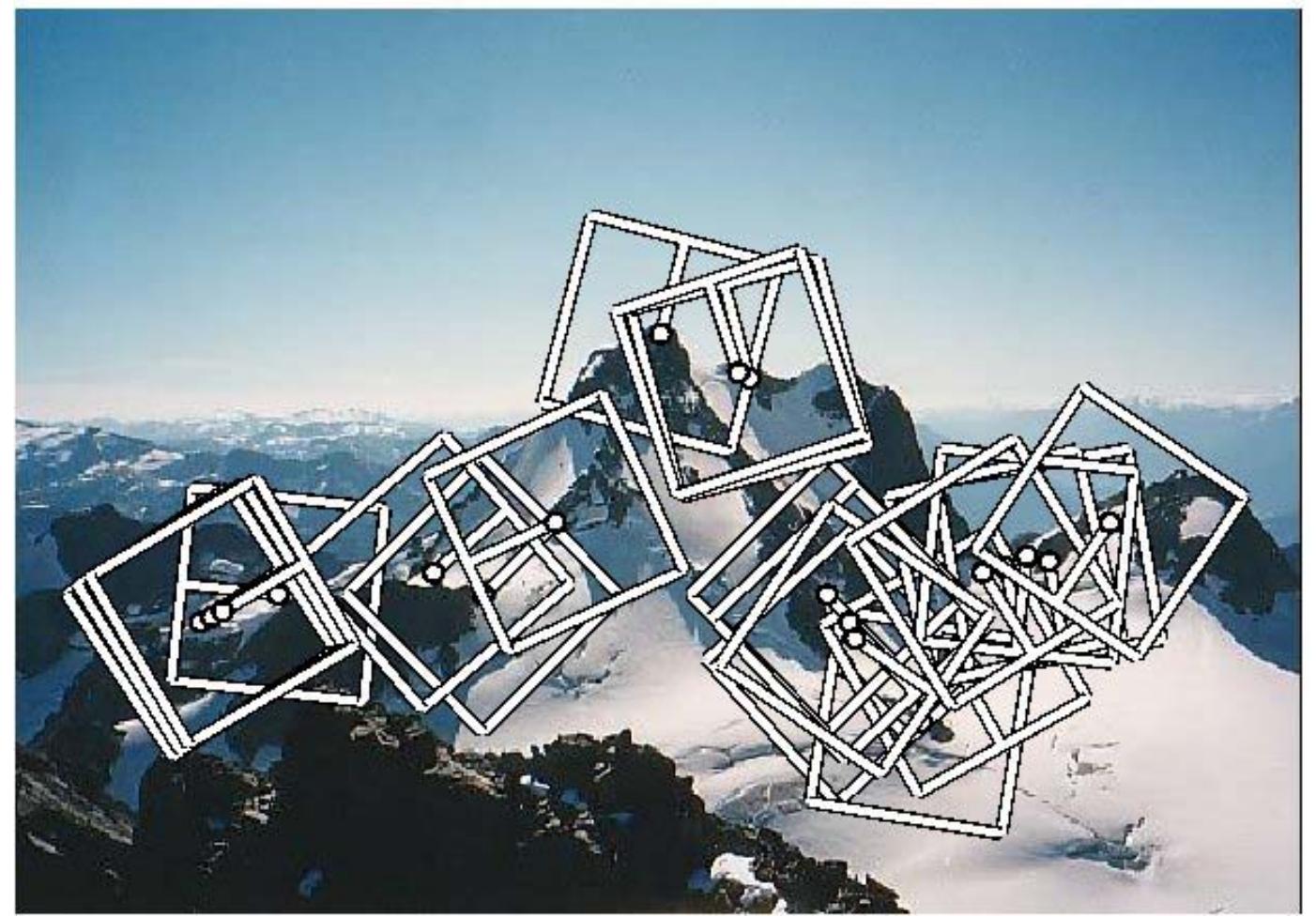
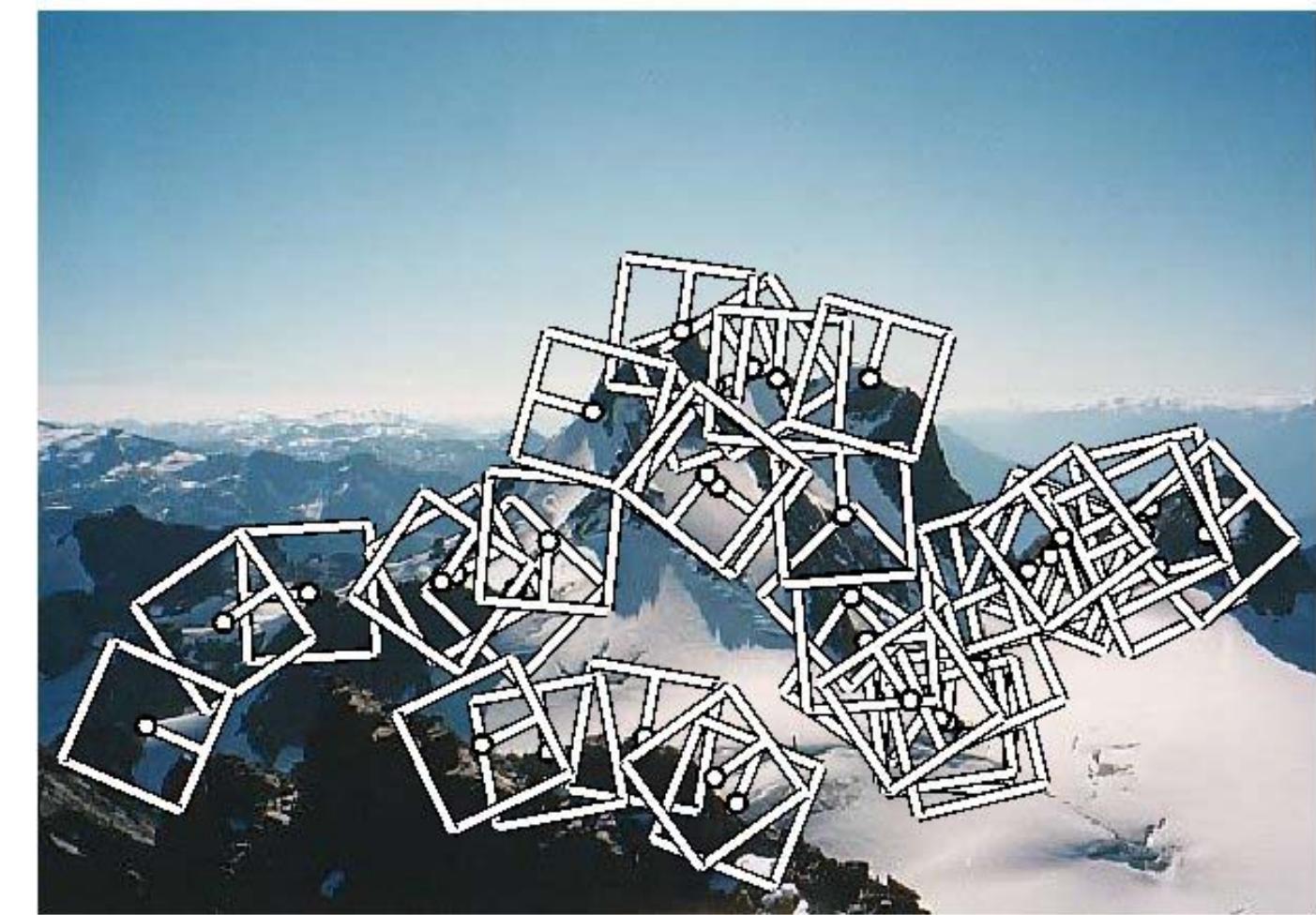
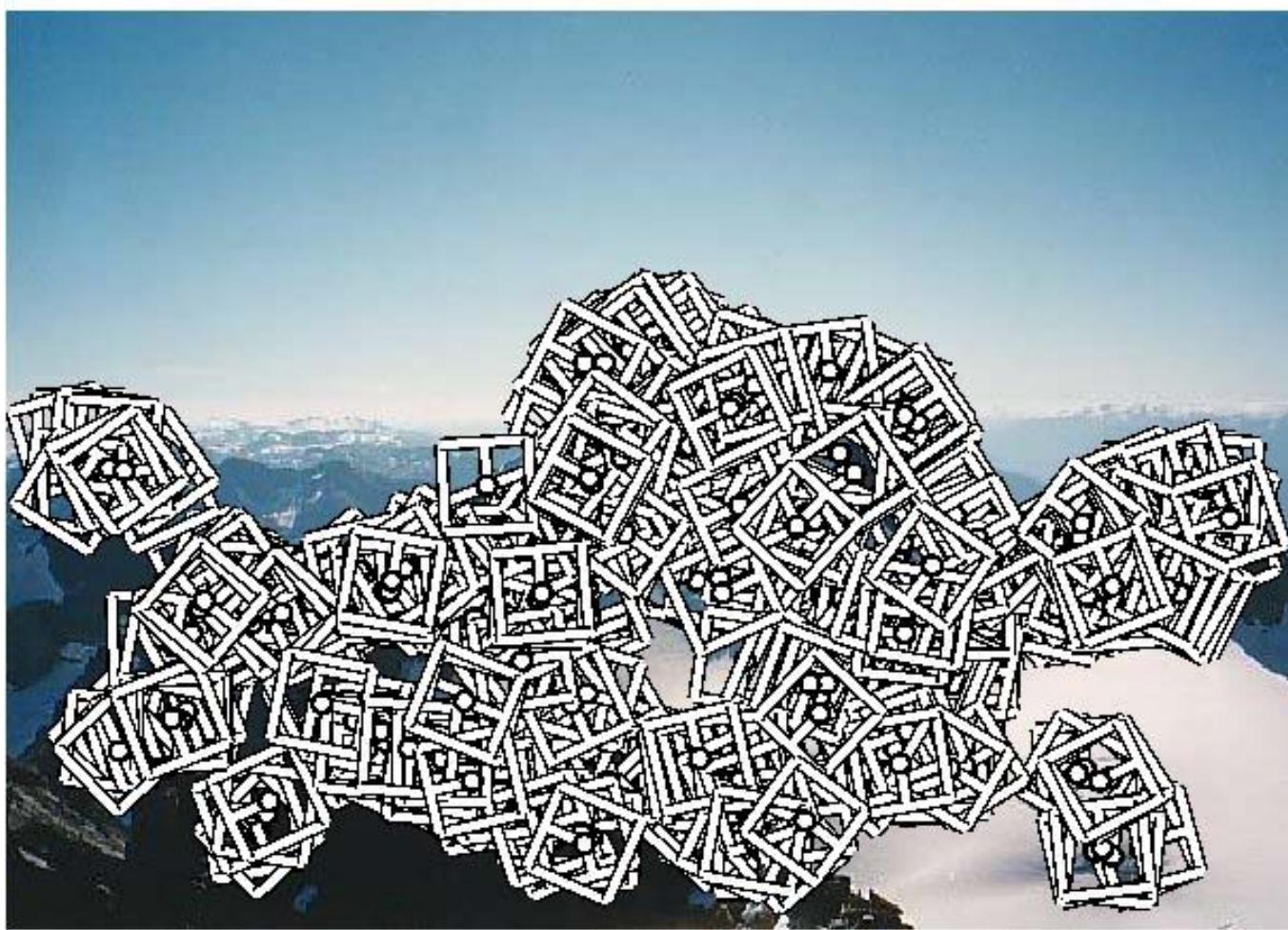
compute Harris feature response

For each level of the Gaussian pyramid

if local maximum and cross-scale

save scale and location of feature (x, y, s)

Multi-Scale Harris Corners



Summary

Edges are useful image features for many applications, but suffer from the aperture problem

Canny Edge detector combines edge filtering with linking and hysteresis steps

Corners / Interest Points have 2D structure and are useful for correspondence

Harris corners are minima of a local SSD function

DoG maxima can be reliably located in scale-space and are useful as interest points

Menu for Today

Topics:

- Corner **Detection**
- Image **Structure**
- **Harris Corner** Detection

Readings:

- **Today's** Lecture: Szeliski 7.1-7.2, Forsyth & Ponce 5.3.0 - 5.3.1

Reminders:

- **Assignment 2:** Scaled Representations, Face Detection and Image Blending
(due **Oct 12** 23:59)
- **Midterm:** October 19th 5pm in class, 75 minutes, closed book