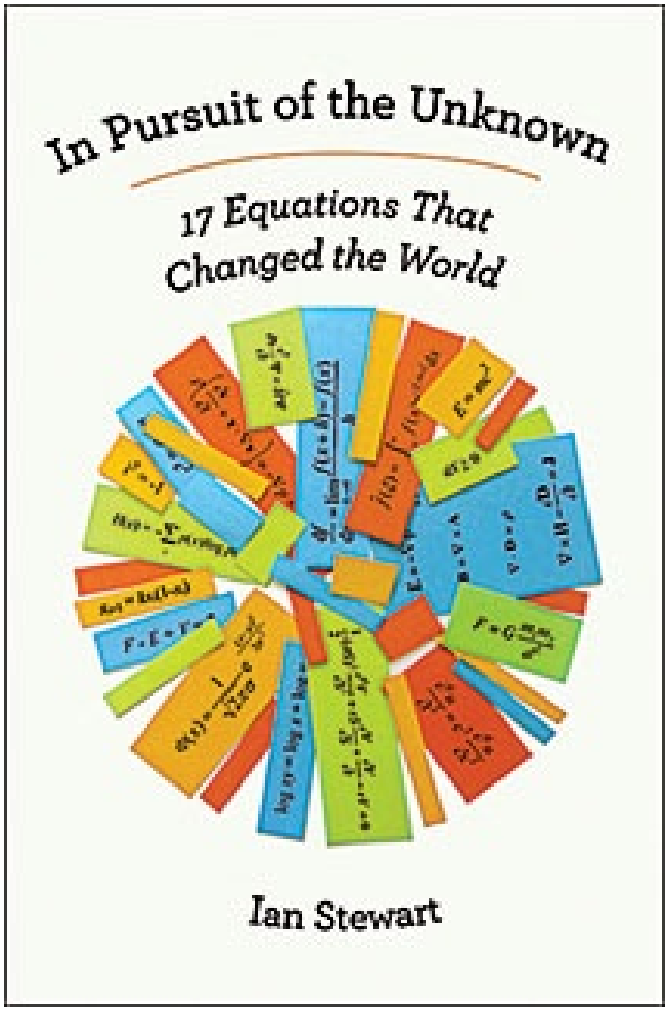
Image Features

So far

- In images (practically) the lowest level of organization (Pixel)
- Filtering (Extract structure from a collection of pixels)
- Convolution (a key operations) (Implemented in working systems through Fourier transform)

Seventeen Equations.....



Seventeen Equations.....

17 Equations That Changed the World
by Ian Stewart

1.	**Pythagoras's Theorem**	$a^2 + b^2 = c^2$	Pythagoras,530 BC
2.	**Logarithms**	$\log xy = \log x + \log y$	John Napier, 1610
3.	**Calculus**	$\frac{\mathrm{d}f}{\mathrm{d}t} = \lim_{h \to 0} \frac{f(t+h) - f(t)}{h}$	Newton, 1668
4.	**Law of Gravity**	$F = G\frac{m_1 m_2}{r^2}$	Newton, 1687
5.	**The Square Root of Minus One**	$i^2 = -1$	Euler, 1750
6.	**Euler's Formula for Polyhedra**	$V - E + F = 2$	Euler, 1751
7.	**Normal Distribution**	$\Phi(x) = \frac{1}{\sqrt{2\pi\rho}} e^{\frac{(x-\mu)^2}{2\rho^2}}$	C.F. Gauss, 1810
8.	**Wave Equation**	$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}$	J. d'Almbert, 1746
9.	**Fourier Transform**	$f(\omega) = \int_{\infty}^{\infty} f(x) e^{-2\pi i x \omega} \mathrm{d}\,x$	J. Fourier, 1822
10.	**Navier-Stokes Equation**	$\rho\left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v}\right) = -\nabla p + \nabla \cdot \mathbf{T} + \mathbf{f}$	C. Navier, G. Stokes, 1845
11.	**Maxwell's Equations**	$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0}$ $\quad \nabla \cdot \mathbf{H} = 0$ $\nabla \times \mathbf{E} = -\frac{1}{c}\frac{\partial \mathbf{H}}{\partial t}$ $\quad \nabla \times \mathbf{H} = \frac{1}{c}\frac{\partial E}{\partial t}$	J.C. Maxwell, 1865
12.	**Second Law of Thermodynamics**	$\mathrm{d}S \geq 0$	L. Boltzmann, 1874
13.	**Relativity**	$E = mc^2$	Einstein, 1905
14.	**Schrodinger's Equation**	$ih\frac{\partial}{\partial t}\Psi = H\Psi$	E. Schrodinger, 1927
15.	**Information Theory**	$H = -\sum p(x) \log p(x)$	C. Shannon, 1949
16.	**Chaos Theory**	$x_{t+1} = kx_t(1 - x_t)$	Robert May, 1975
17.	**Black-Scholes Equation**	$\frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} + \frac{\partial V}{\partial t} - rV = 0$	F. Black, M. Scholes, 1990

So far

- In images (practically) the lowest level of organization (Pixel)
- Filtering (Extract structure from a collection of pixels)
- Convolution (a key operations) (Implemented in working systems through Fourier transform)
- Smoothening (Gaussian)
- Edges (Good Abstraction)
- Role of Derivatives
- Scale of the images (Downsample Repeatedly Pyramid)
- Subsampling (Nyquist Rate)
- Upsampling (Interpolation)
- Interpolation also implemented through convolutions

Image Features

- What skyline is this?

Image Features

- What skyline is this?

Image Features

- What skyline is this?

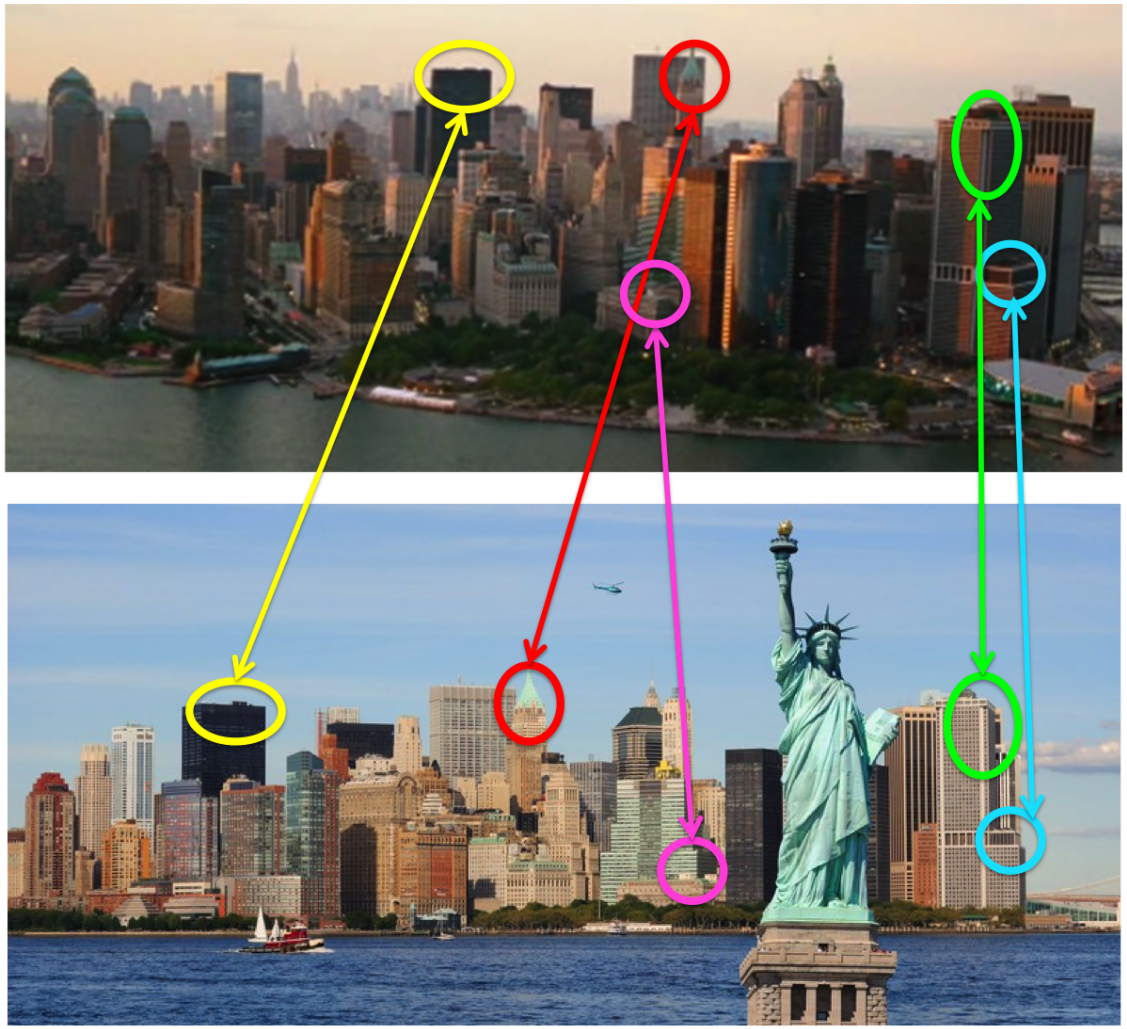


Image Features

- What skyline is this?

We matched in:

- Distinctive locations: **keypoints**
- Distinctive features: **descriptors**

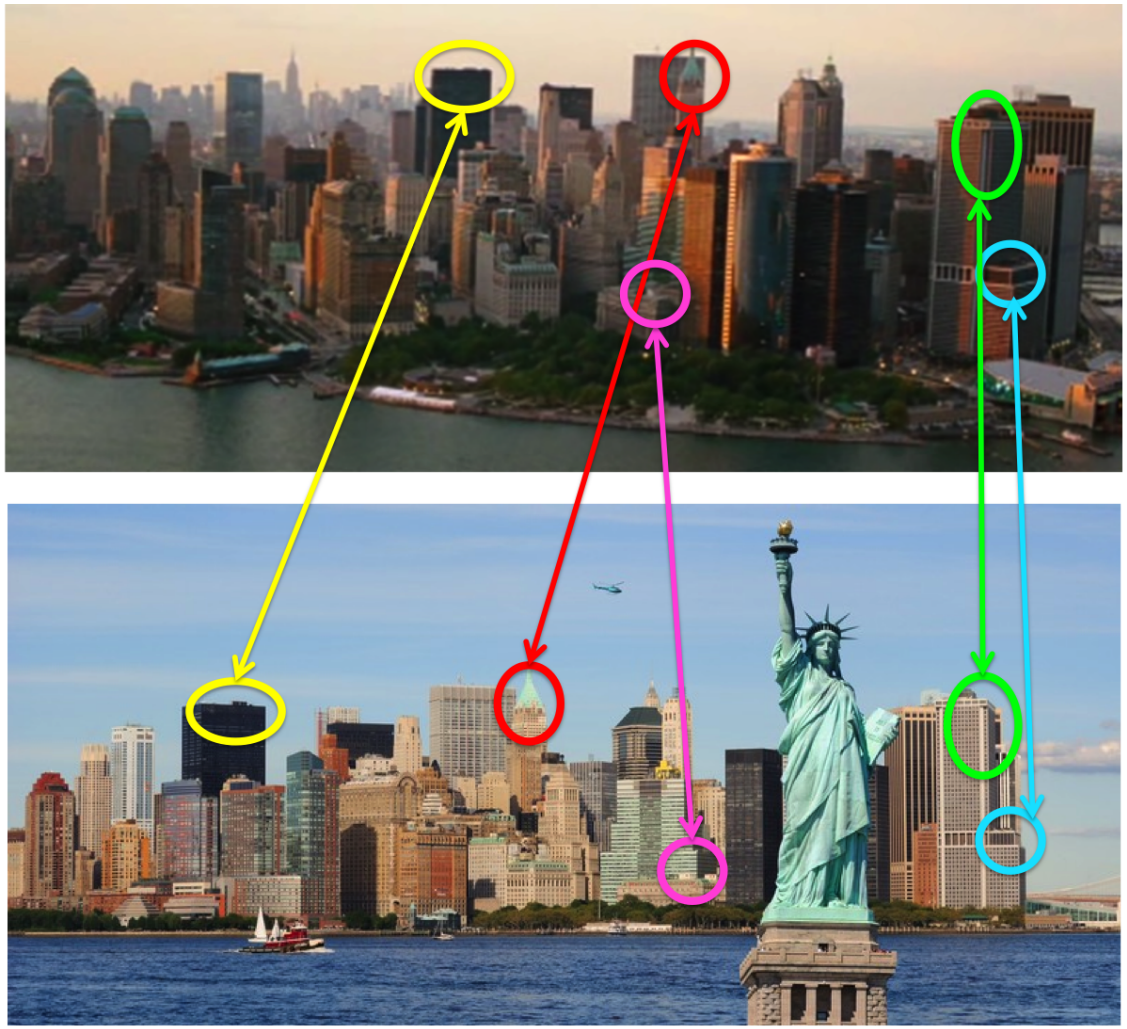
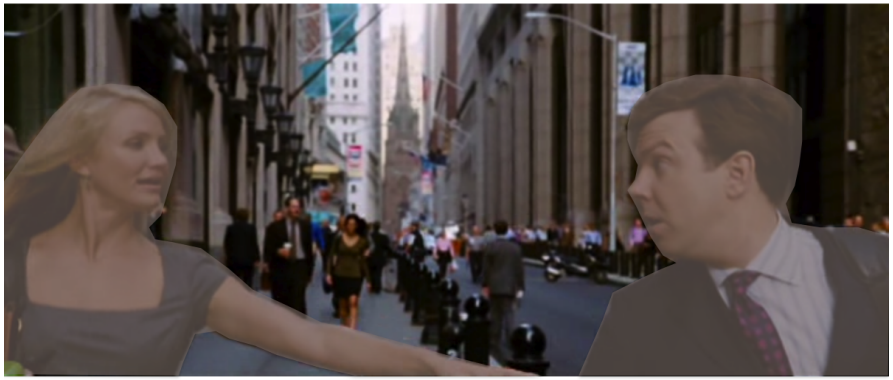


Image Features

- How could we tell which type of scene this is from a movie?



What kind of scene is behind the actors?
Kitchen? Bedroom? Street? Bar?

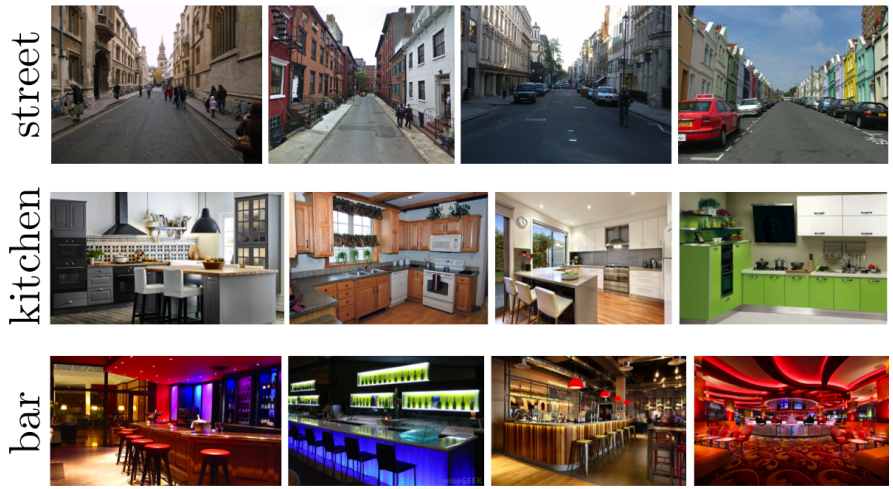
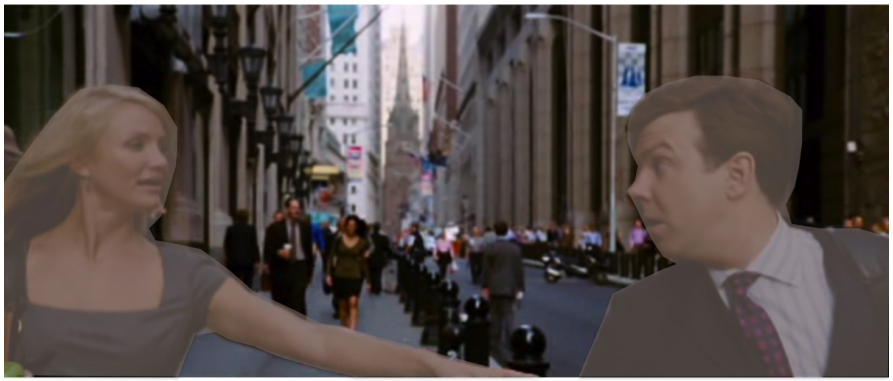


Image Features

- How could we tell which type of scene this is from a movie?

We matched:

- **Globally** – one descriptor for full image (?)
- More complex descriptor: color, gradients, "deep" features (learned), etc



What kind of scene is behind the actors?
Kitchen? Bedroom? Street? Bar?

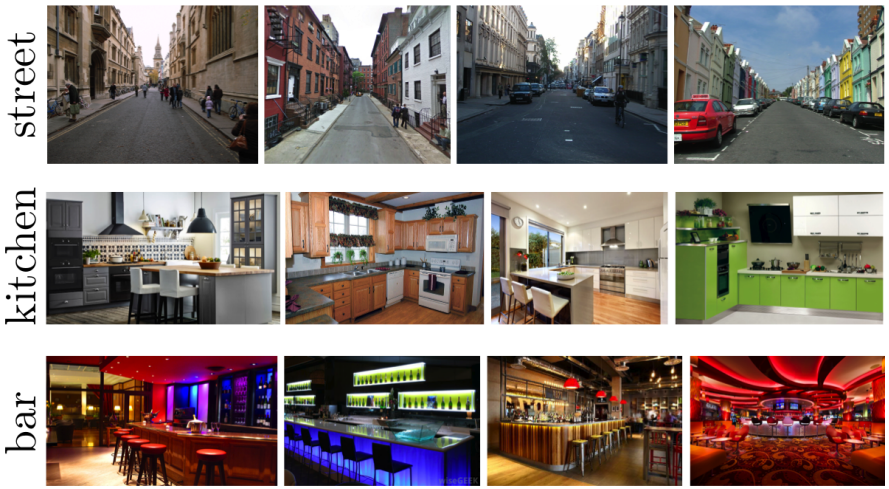


Image Features

- How would we solve this?

Are these two cups of the same type?

Image Features

- How would we solve this?

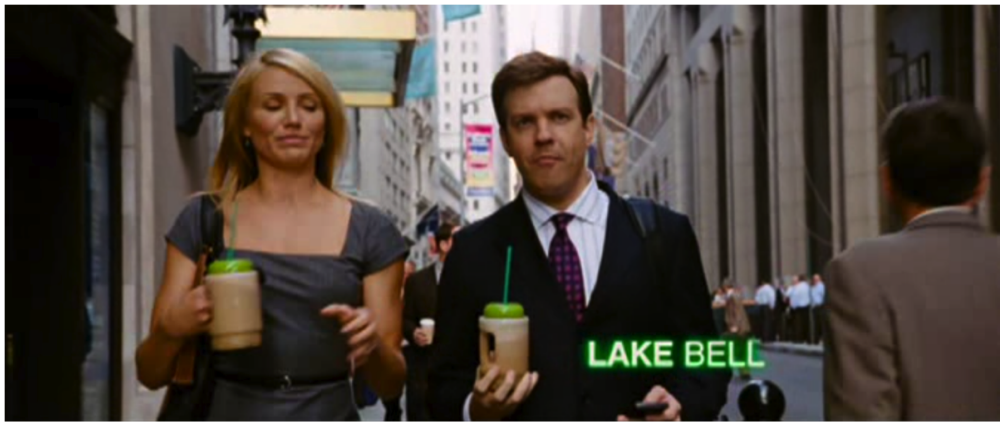
We matched:

- One descriptor for full **patch**
- Descriptor can be simple, e.g. **color**

Are these two cups of the same type?

Image Features

- How would we solve this?



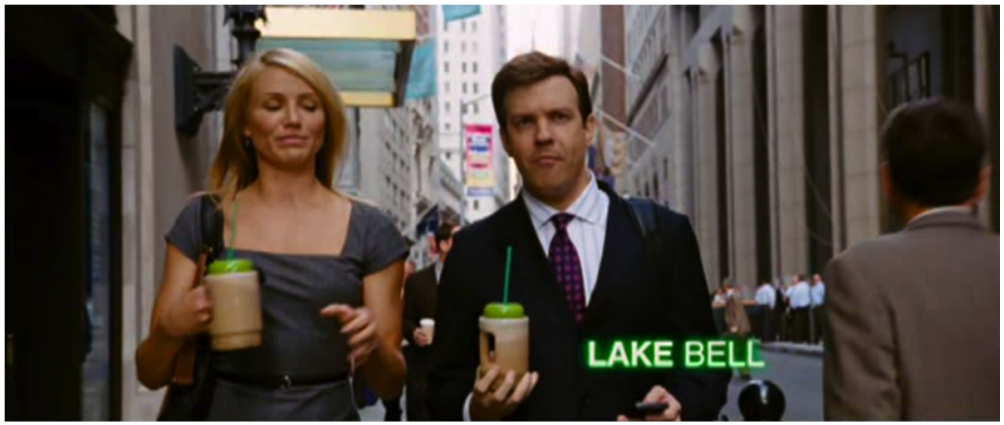
Where can I find this pattern? ·····→ 

Image Features

- How would we solve this?

We matched:

- At each location
- Compared pixel values



Where can I find this pattern? ⋯⋯→ 

Image Features

- How would we solve this?

Where can I find this pattern? ⋯⋯→ 

Image Features

- How would we solve this?

We matched:

- Distinctive locations
- Distinctive features
- Affine invariant

Where can I find this pattern? ⋯⋯→ 

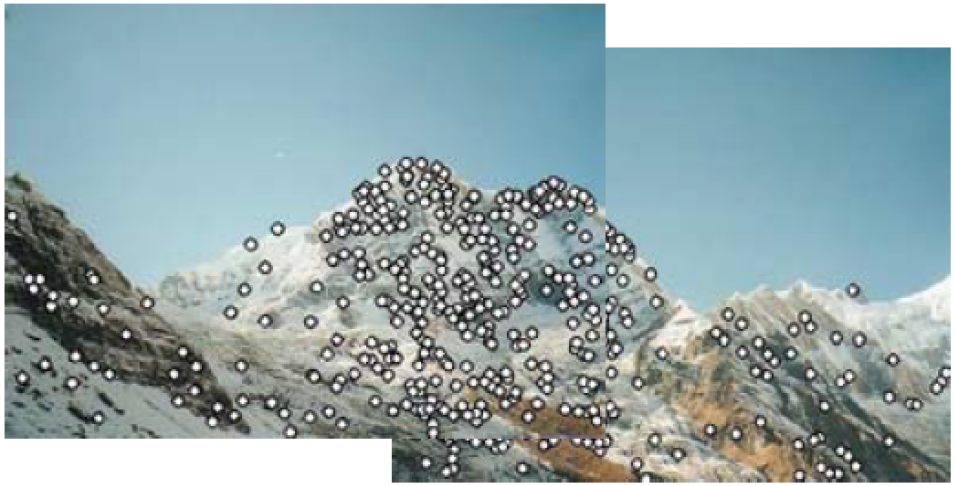
Image Features

- How would we solve this?

Image Features:

Interest Point (Keypoint) Detection

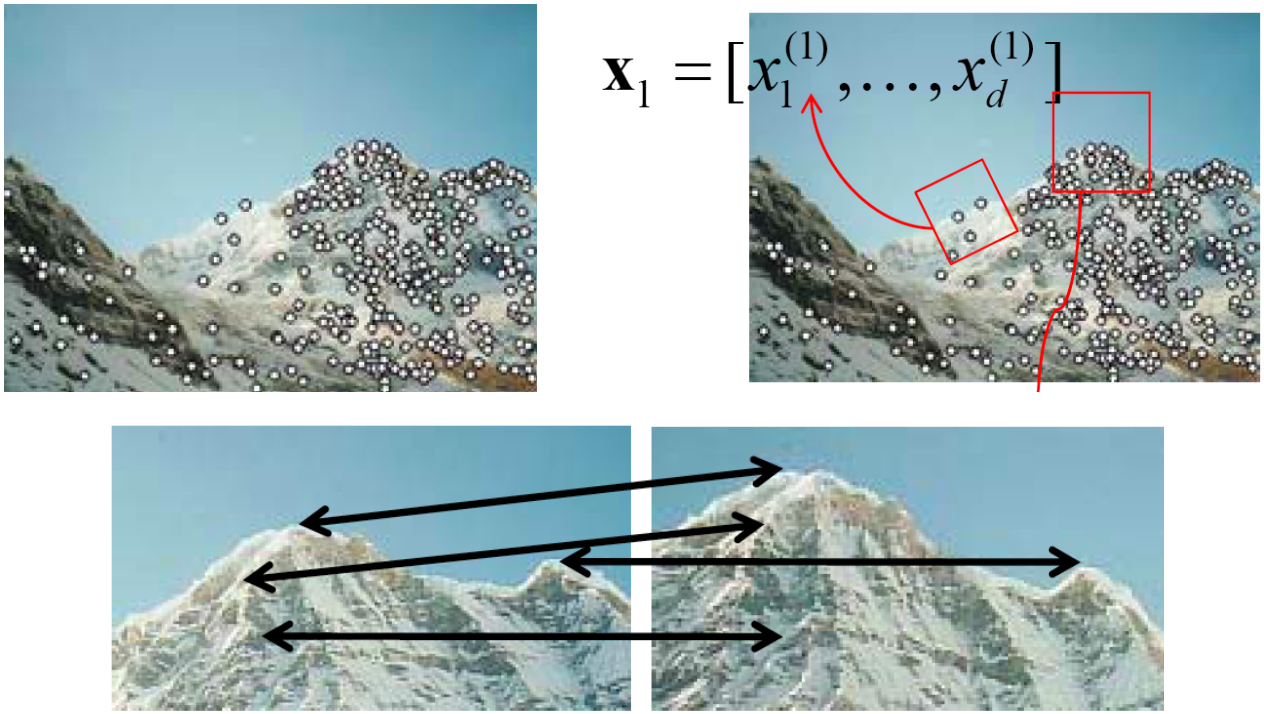
Application Example: Image Stitching



[Source: K. Grauman]

Local Features

- **Detection**: Identify the interest points.
- **Description**: Extract **feature vector** descriptor around each interest point.
- **Matching**: Determine correspondence between descriptors in two views.



[Source: K. Grauman]

Goal: Repeatability of the Interest Point Operator

- Our goal is to detect (at least some of) the same points in both images
- We have to be able to run the detection procedure independently per image
- We need to generate enough points to increase our chances of detecting matching points
- We shouldn't generate too many or our matching algorithm will be too slow

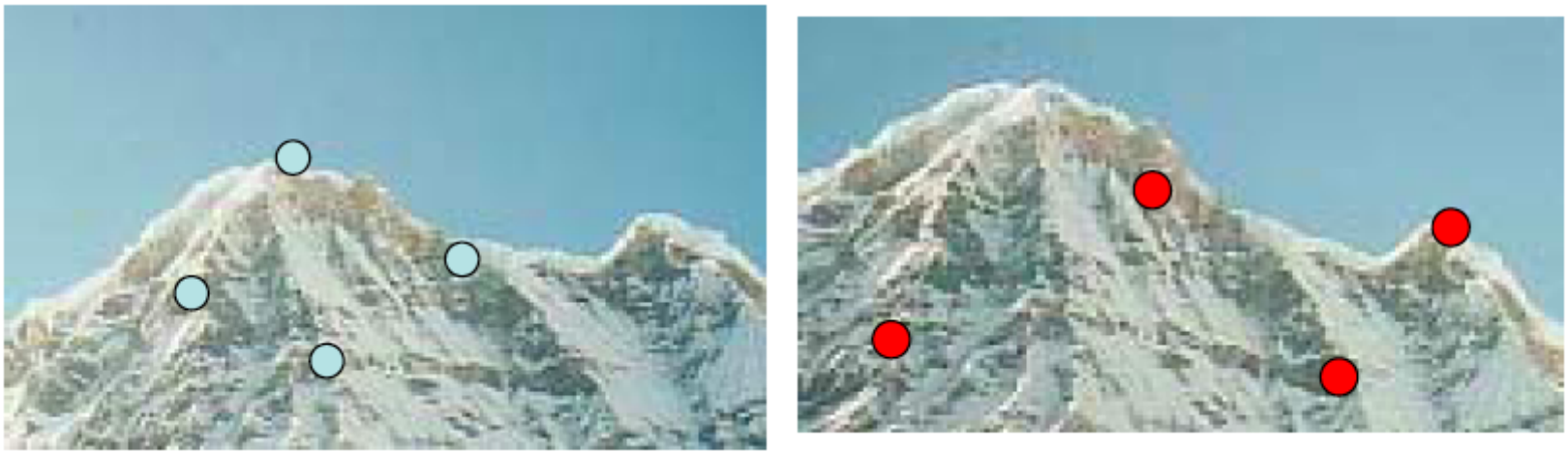


Figure: Too few keypoints $\rightarrow$ little chance to find the true matches

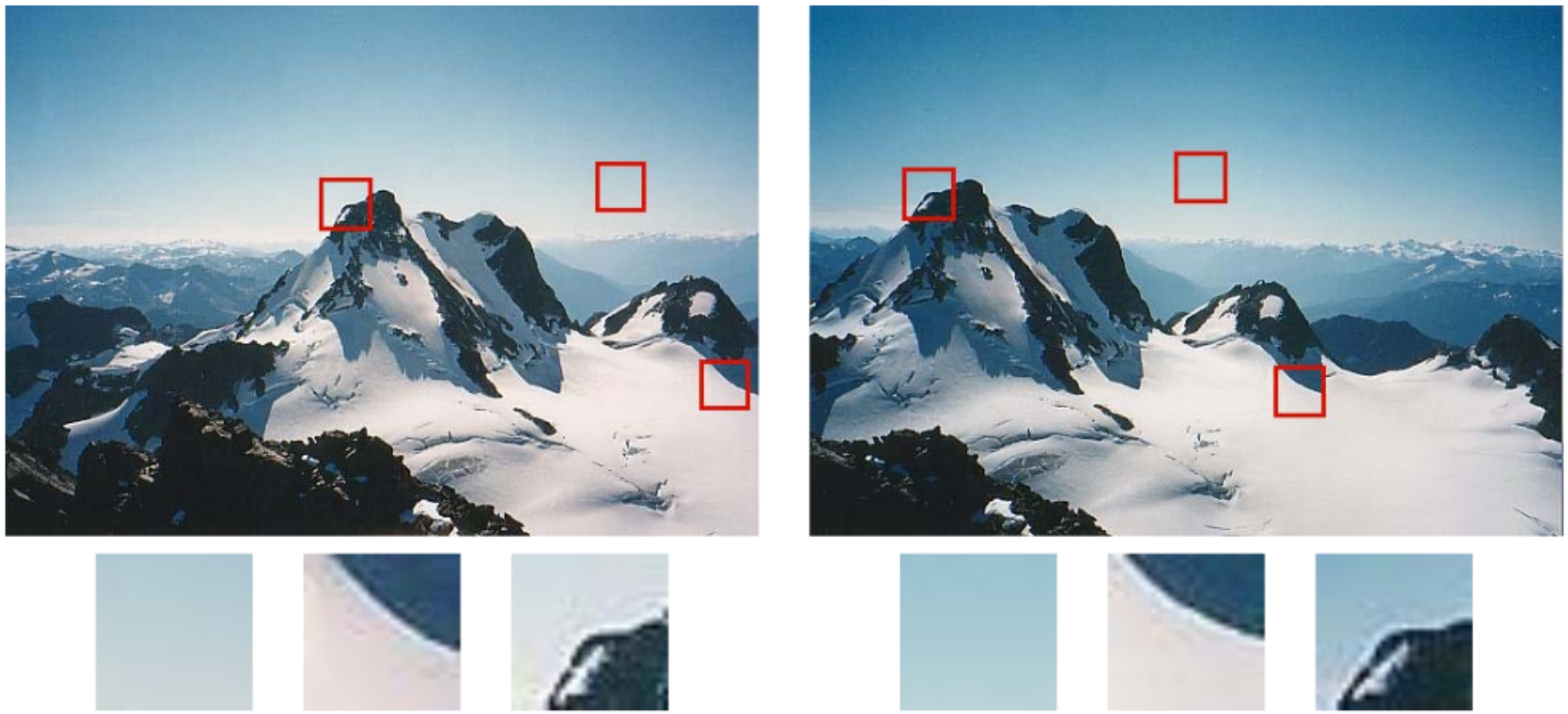
[Source: K. Grauman, slide credit: R. Urtasun]

What Points to Choose?

[Source: K. Grauman]

What Points to Choose for matching?

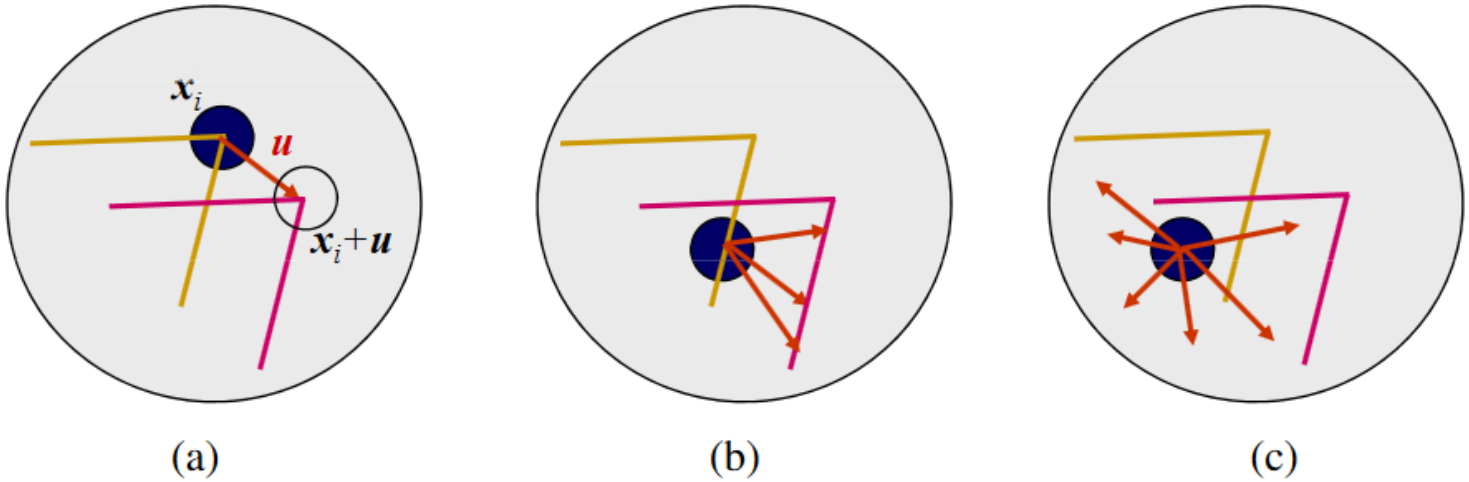
What Points to Choose for matching?



- Textureless patches are nearly impossible to localize.
- Patches with large contrast changes (gradients) are easier to localize.
- But straight line segments cannot be localized on lines segments with the same orientation (aperture problem)
- Gradients in at least two different orientations are easiest, e.g., **corners**!

[Adopted from: Szelski (Book)]

Aperture Problem



- "Corner-like" patch can be reliably matched
- A straight line patch can have multiple matches (Aperture Problem)
- Zero texture, useless, can have infinite matches

[Adopted from: Szelski (Book)]

Interest Points: Corners

- How can we find corners in an image?

Interest Points: Corners

- We should easily recognize the point by looking through a small window.
- Shifting a window in any direction should give a large change in intensity.

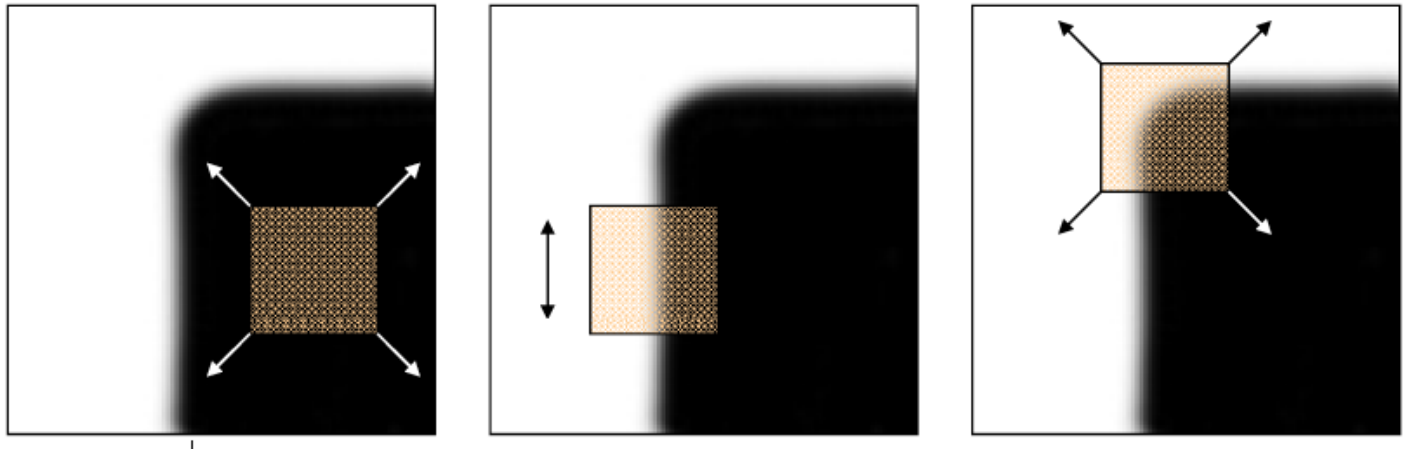
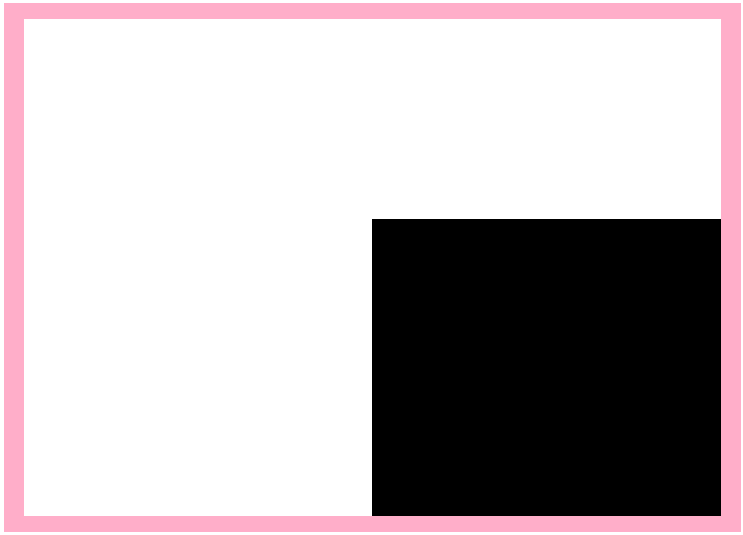


Figure: (left) flat region: no change in all directions, (center) edge: no change along the edge direction, (right) corner: significant change in all directions

[Source: Alyosha Efros, Darya Frolova, Denis Simakov]

Interest Points: Corners

- Harris Corner Detector: Idea



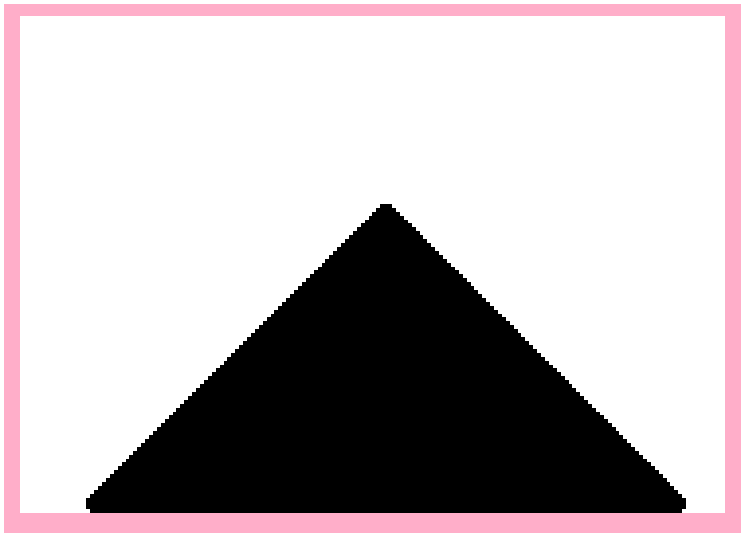
ΣI_x^2 is large

ΣI_y^2 is large

$\Longrightarrow$ Corner!

Interest Points: Corners

- Harris Corner Detector: Idea



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

$\Longrightarrow$ eigenvalues!

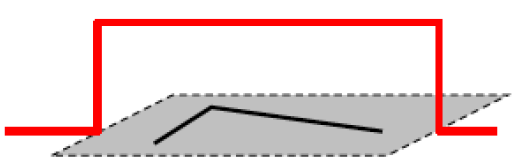
Interest Points: Corners

- Compare two image patches using (weighted) summed square difference
- Measures change in appearance of window $w(x, y)$ for the shift

$$E_{\mathrm{WSSD}}(u, v) = \sum_x \sum_y w(x, y)[I(x + u, y + v) - I(x, y)]^2$$

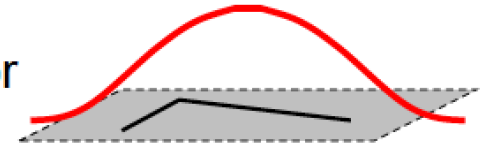
window function | shifted intensity | intensity

Window function $w(x,y)$ =



1 in window, 0 outside

or



Gaussian

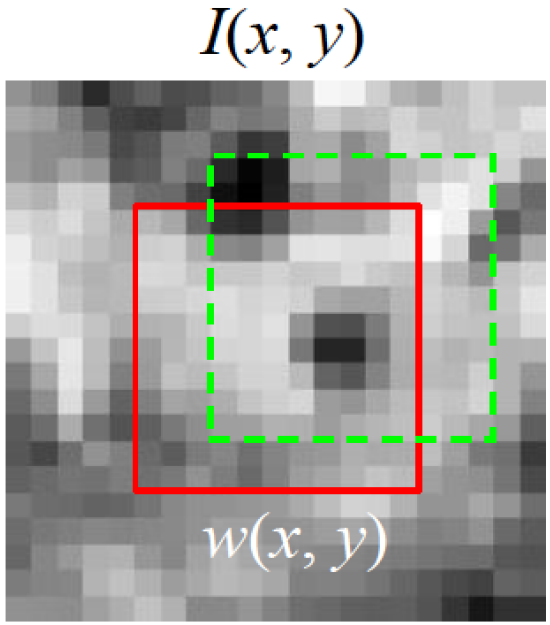
[Source: J. Hays]

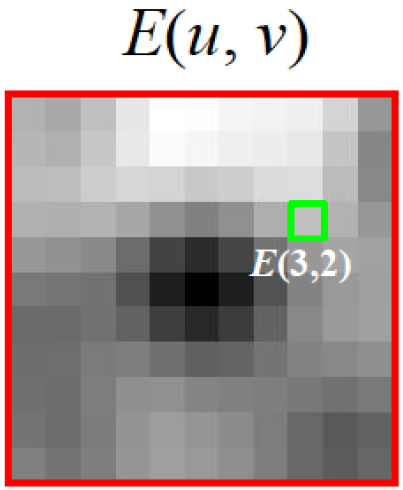
Interest Points: Corners

- Compare two image patches using (weighted) summed square difference
- Measures change in appearance of window $w(x, y)$ for the shift

$$E_{\text{WSSD}}(u, v) = \sum_x \sum_y w(x, y)[I(x + u, y + v) - I(x, y)]^2$$

window function | shifted intensity | intensity





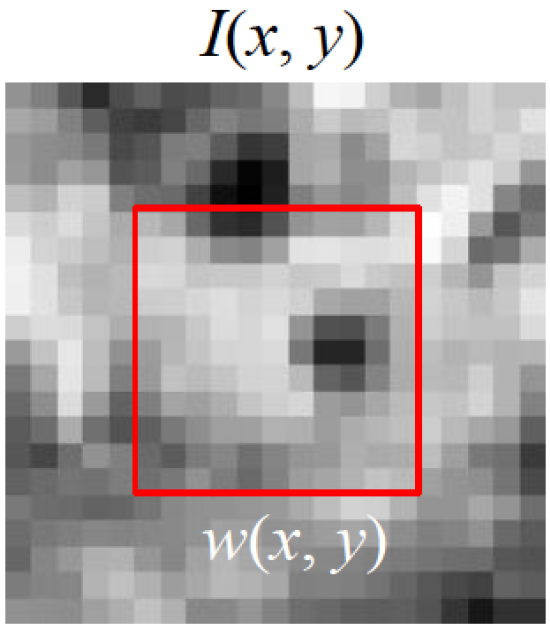
[Source: J. Hays]

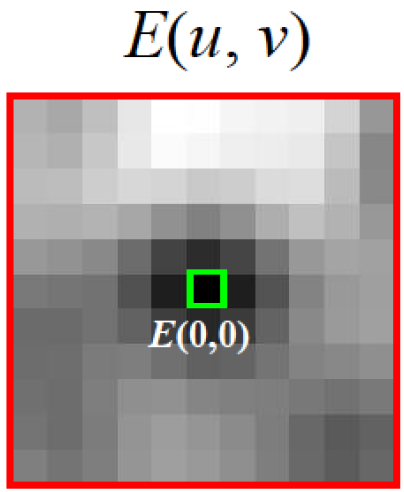
Interest Points: Corners

- Compare two image patches using (weighted) summed square difference
- Measures change in appearance of window $w(x, y)$ for the shift

$$E_{\text{WSSD}}(u, v) = \sum_x \sum_y w(x, y)[I(x + u, y + v) - I(x, y)]^2$$

window function | shifted intensity | intensity

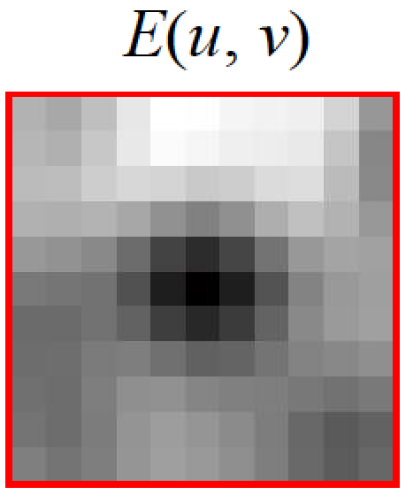




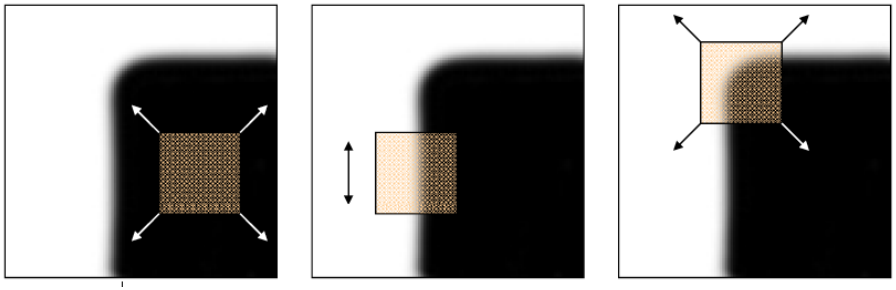
[Source: J. Hays]

Interest Points: Corners

- Let's look at E_{WSSD}
- We want to find out how this function behaves for small shifts



- Remember our goal to detect corners:



Interest Points: Corners

- Using a simple first-order Taylor Series expansion about x, y:

$$I(x+u, y+v) \approx I(x,y) + u \cdot \frac{\partial I}{\partial x}(x,y) + v \cdot \frac{\partial I}{\partial y}(x,y)$$

- Using a series of polynomials to approximate I, more info on Taylor Series here
- And plugging it in our expression for E_{WSSD}:

$$\begin{aligned}
E_{\text{WSSD}}(u,v) &= \sum_x \sum_y w(x,y)\Big(I(x+u, y+v) - I(x,y)\Big)^2 \\
&\approx \sum_x \sum_y w(x,y)\Big(I(x,y) + u \cdot I_x + v \cdot I_y - I(x,y)\Big)^2 \\
&= \sum_x \sum_y w(x,y)\Big(u^2 I_x^2 + 2u \cdot v \cdot I_x \cdot I_y + v^2\ I_y^2\Big) \\
&= \sum_x \sum_y w(x,y) \cdot \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}
\end{aligned}$$

Interest Points: Corners

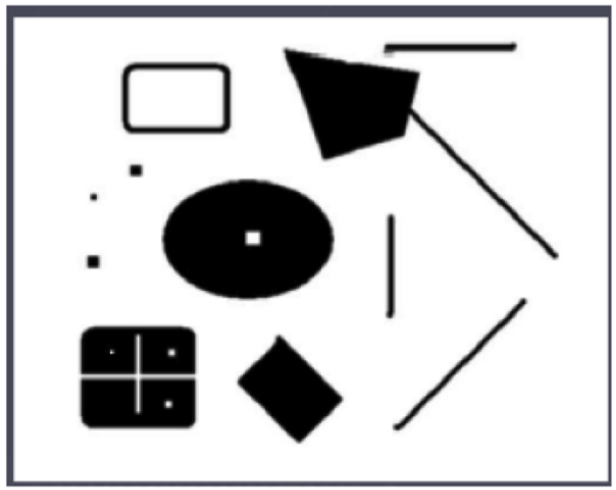
- Since (u, v) doesn't depend on (x, y) we can rewrite it slightly:

$$
\begin{aligned}
E_{\mathrm{WSSD}}(u, v) &= \sum_x \sum_y w(x, y) \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \\
&= \begin{bmatrix} u & v \end{bmatrix} \underbrace{\left(\sum_x \sum_y w(x, y) \begin{bmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{bmatrix} \right)}_{\text{Let's denotes this with } M} \begin{bmatrix} u \\ v \end{bmatrix} \\
&= \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}
\end{aligned}
$$

- M is a 2×2 *second moment matrix* computed from image gradients:

$$
M = \sum_x \sum_y w(x, y) \begin{bmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{bmatrix}
$$

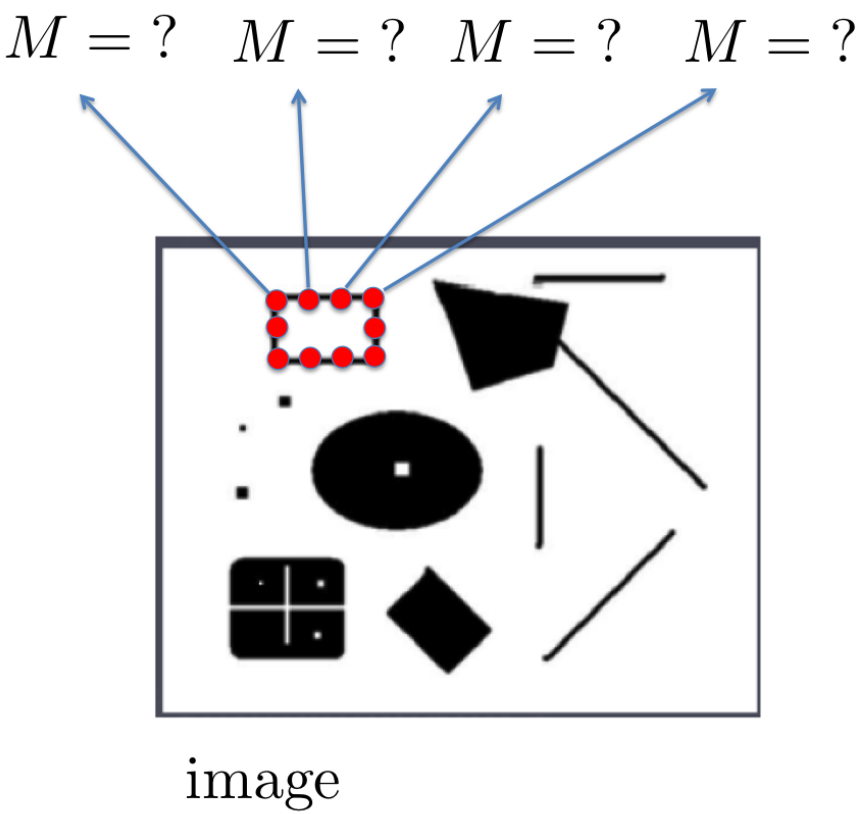
How Do I Compute M?



image

- Let's say I have this image

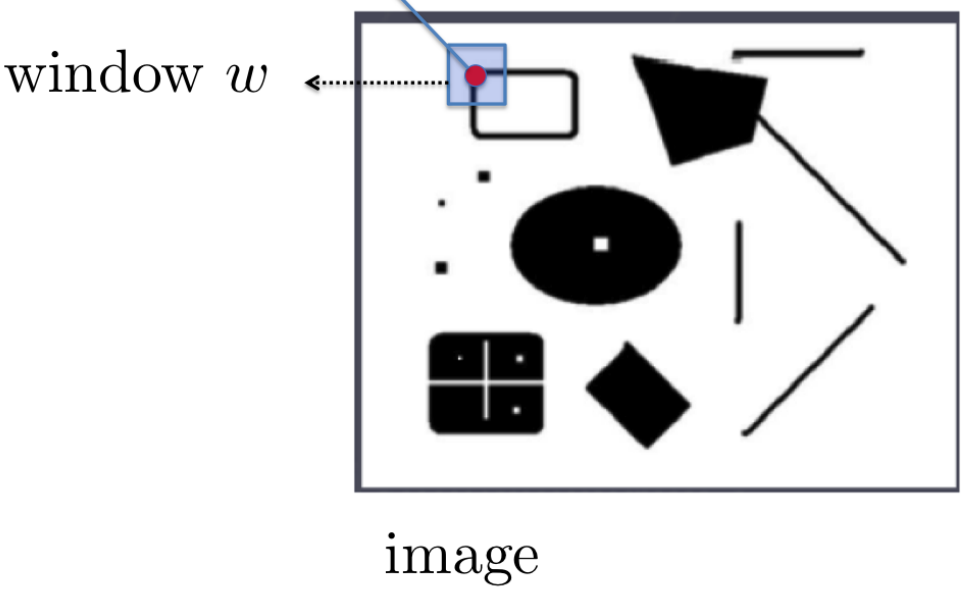
How Do I Compute M?



- Let's say I have this image
- I need to compute a 2×2 second moment matrix in each image location

How Do I Compute M?

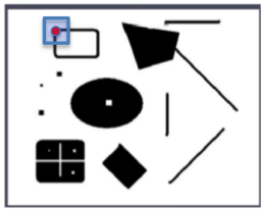
$$M = \sum_x \sum_y w(x,y) \begin{bmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{bmatrix}$$



- Let's say I have this image
- I need to compute a 2×2 second moment matrix in each image location
- In a particular location I need to compute M as a weighted average of gradients in a window

How Do I Compute M?

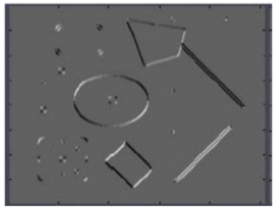
$$M = \sum_x \sum_y w(x,y) \begin{bmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{bmatrix}$$



image

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

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



$$I_x \cdot I_y$$

- Let's say I have this image
- I need to compute a 2×2 second moment matrix in each image location
- In a particular location I need to compute M as a weighted average of gradients in a window

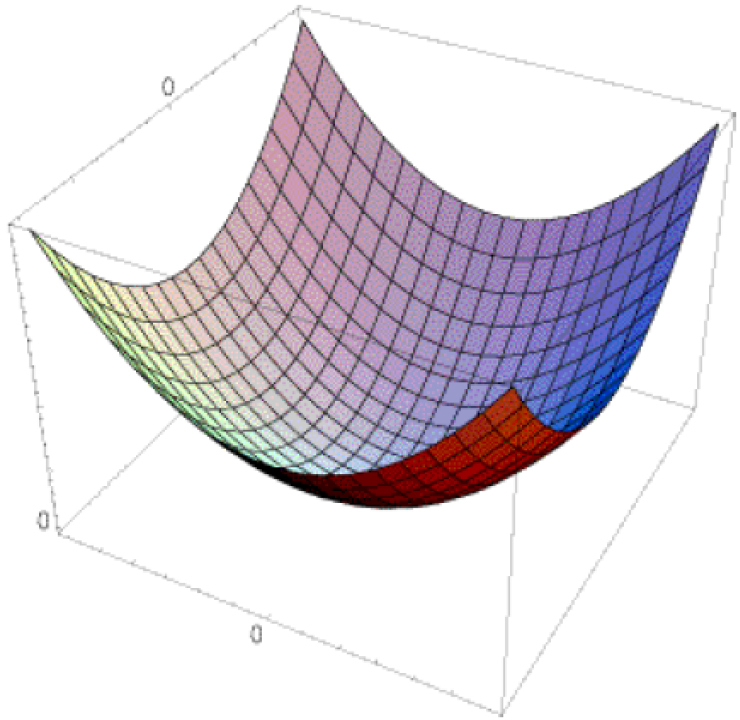
I can do this efficiently by computing three matrices, I_x^2, I_y^2 and $I_x \cdot I_y$, and convolving each one with a filter, e.g. a box or Gaussian filter

Interest Points: Corners

- We now have M computed in each image location
- Our E_{WSSD} is a **quadratic function** where M implies its shape

$$E_{\mathrm{WSSD}}(u, v) = \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$$

$$M = \sum_x \sum_y w(x, y) \begin{bmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{bmatrix}$$



[Source: J. Hays]

Interest Points: Corners

- Let's take a horizontal "slice" of $E_{\mathrm{WSSD}}(u, v)$:

$$\begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = \mathrm{const}$$

- This is the equation of an ellipse

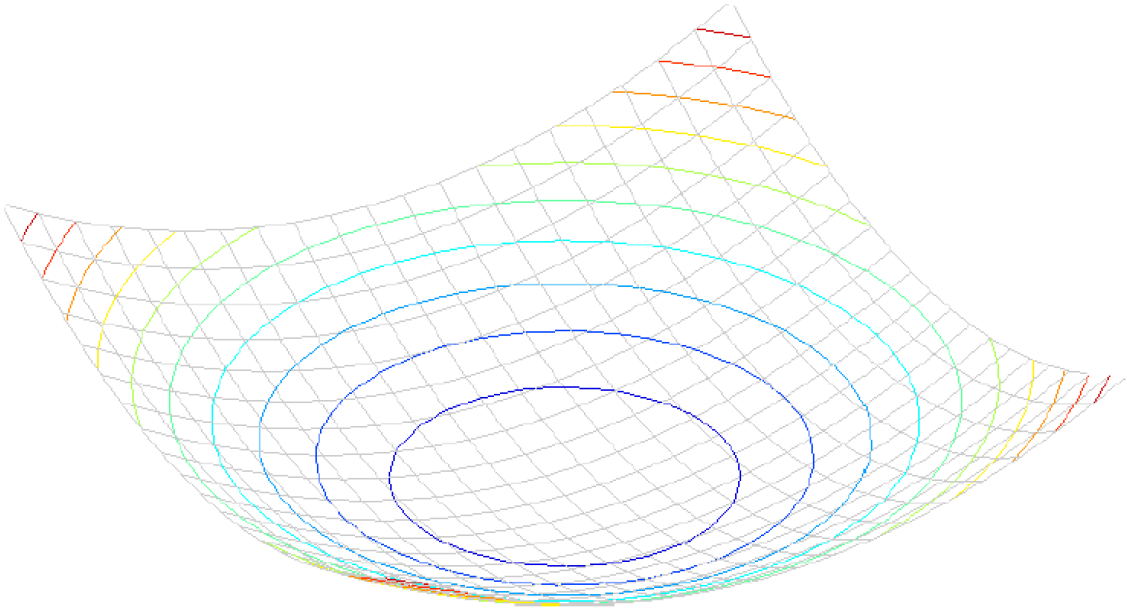


Figure: Different ellipses obtain by different horizontal "slices"

Interest Points: Corners

- Let's take a horizontal "slice" of $E_{\mathrm{WSSD}}(u, v)$:

$$\begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$$

- This is the equation of an ellipse

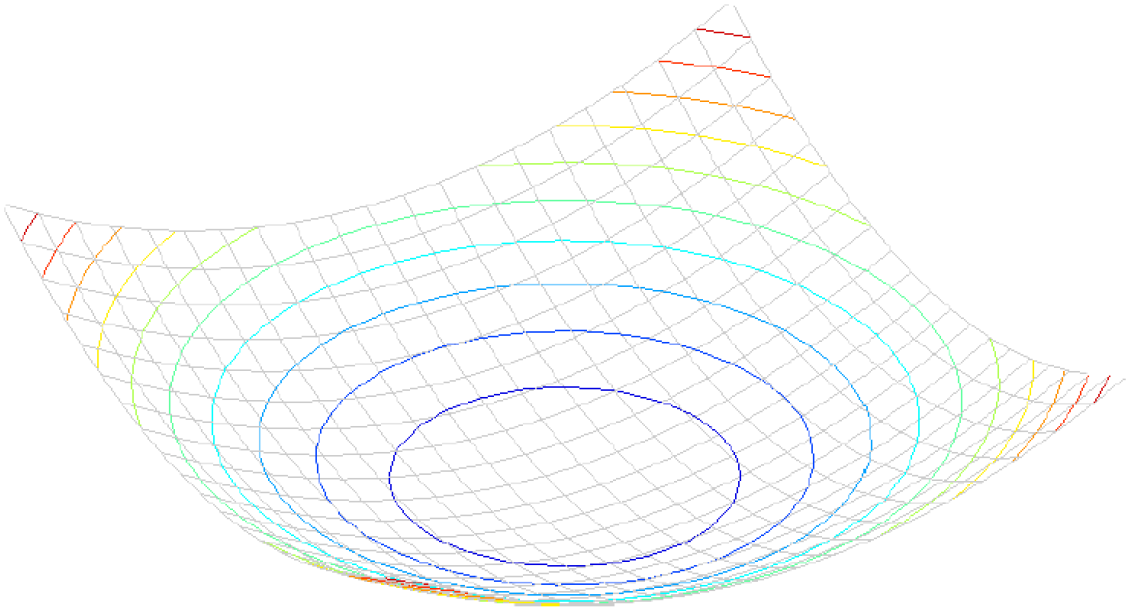


Figure: Different ellipses obtain by different horizontal "slices"

Interest Points: Corners

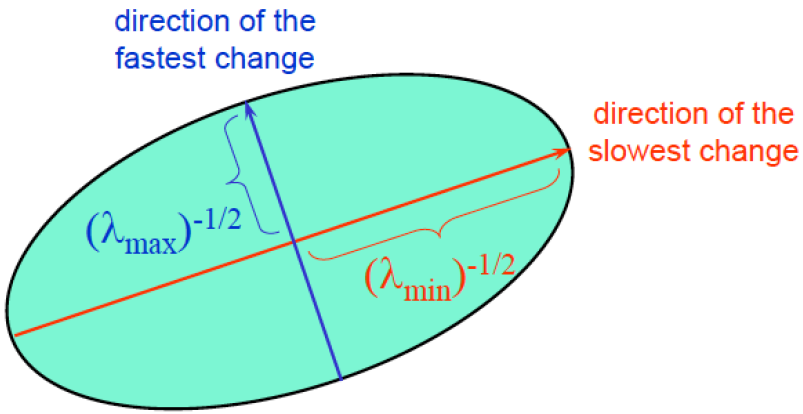
- Our matrix M is symmetric:

$$M = \sum_x \sum_y w(x,y) \begin{bmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{bmatrix}$$

- And thus we can diagonalize it (in Matlab: [V,D] = EIG(M)):

$$M = V \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} V^{-1}$$

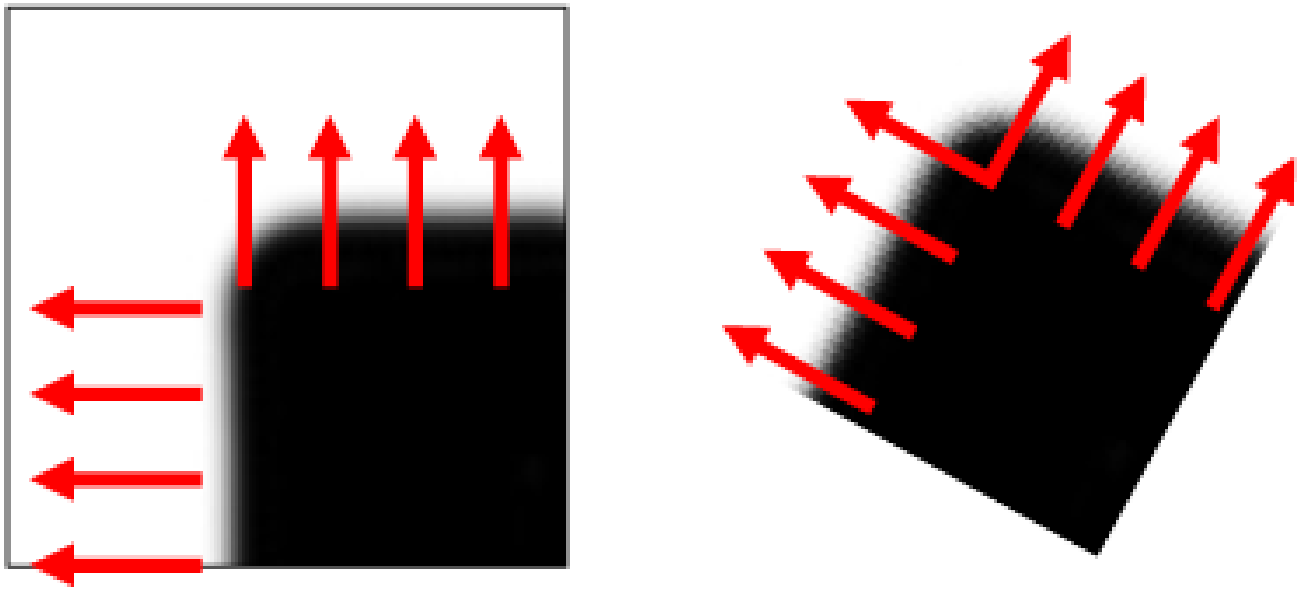
- Columns of V are major and minor axes of ellipse, the lengths of the radii proportional to $\lambda^{-1/2}$



[Source: J. Hays]

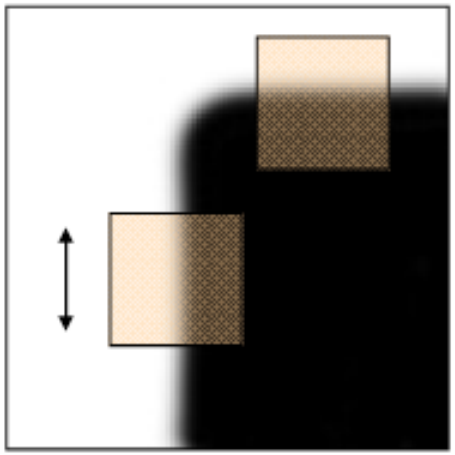
Interest Points: Corners

- The eigenvalues of M (λ_1, λ_2) reveal the amount of intensity change in the two principal orthogonal gradient directions in the window



[Source: R. Szeliski, slide credit: R. Urtasun]

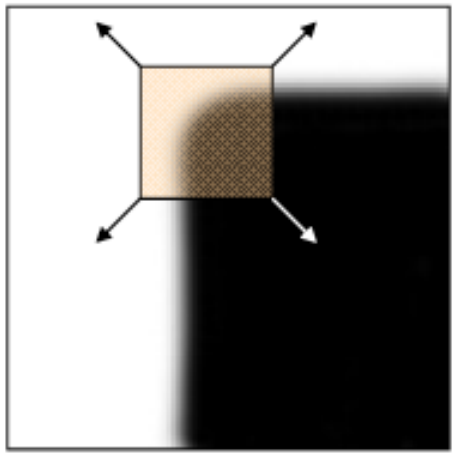
Interest Points: Corners



"edge":

$\lambda_1 >> \lambda_2$

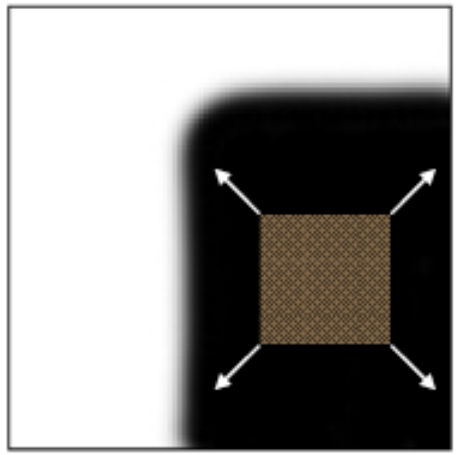
$\lambda_2 >> \lambda_1$



"corner":

λ_1 and λ_2 are large,

$\lambda_1 \sim \lambda_2$;



"flat" region

λ_1 and λ_2 are small;

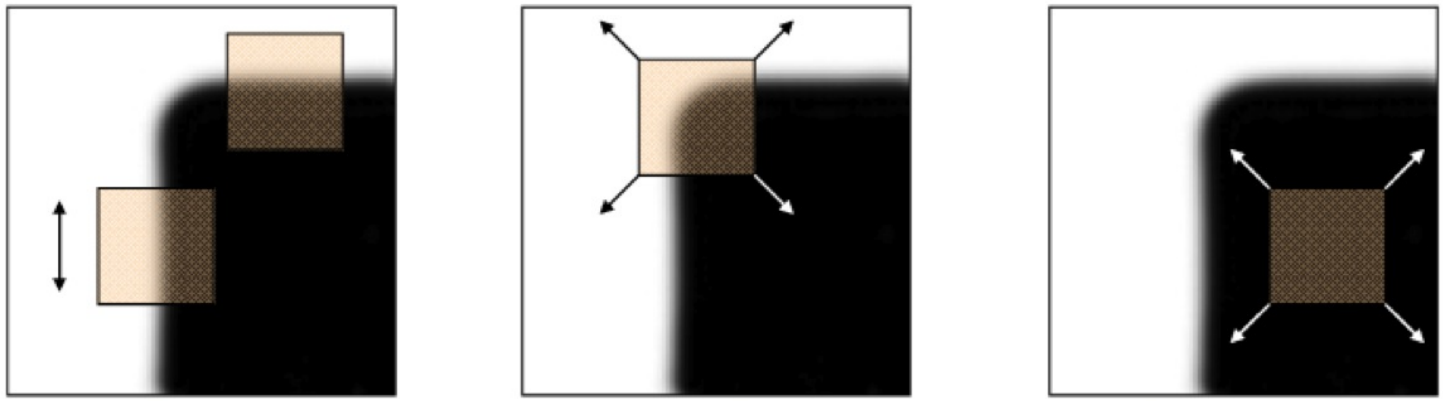
[Source: K. Grauman, slide credit: R. Urtasun]

Interest Points: Criteria to Find Corners

- Harris and Stephens, '88, is rotationally invariant and downweighs edge-like features where $\lambda_1 \gg \lambda_0$

$$R = \lambda_0\lambda_1 - \alpha(\lambda_0 + \lambda_1)^2 = \det(M) - \alpha \cdot \text{trace}(M)^2$$

- **Why** go via det and trace and not use a criteria with λ?
- α a constant (0.04 to 0.06)



"edge": $R < 0$

"corner": $R > 0$

"flat" region $|R|$ small

- The corresponding detector is called **Harris corner detector**

Interest Points: Criteria to Find Corners

- Harris and Stephens, 88 is rotationally invariant and downweighs edge-like features where $\lambda_1 \gg \lambda_0$

$$R = \lambda_0\lambda_1 - \alpha(\lambda_0 + \lambda_1)^2 = \det(M) - \alpha \text{trace}(M)^2$$

- Shi and Tomasi, 94 proposed the smallest eigenvalue of $\mathbf{A}$, i.e., $\lambda_0^{-1/2}$.
- Triggs, 04 suggested

$$\lambda_0 - \alpha\lambda_1$$

  also reduces the response at 1D edges, where aliasing errors sometimes inflate the smaller eigenvalue

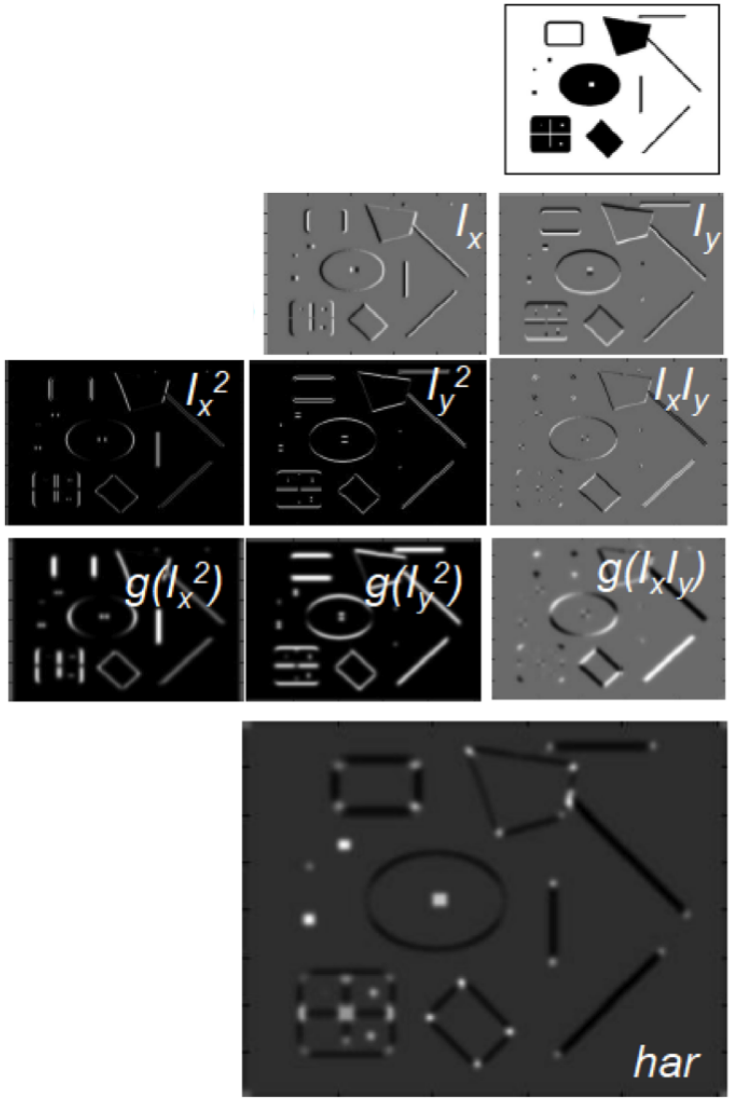
- Brown et al, 05 use the harmonic mean

$$\frac{\det(\mathbf{A})}{\text{trace}(\mathbf{A})} = \frac{\lambda_0\lambda_1}{\lambda_0 + \lambda_1}$$

[Source Mubarak Shah, Szelski]

Harris Corner detector

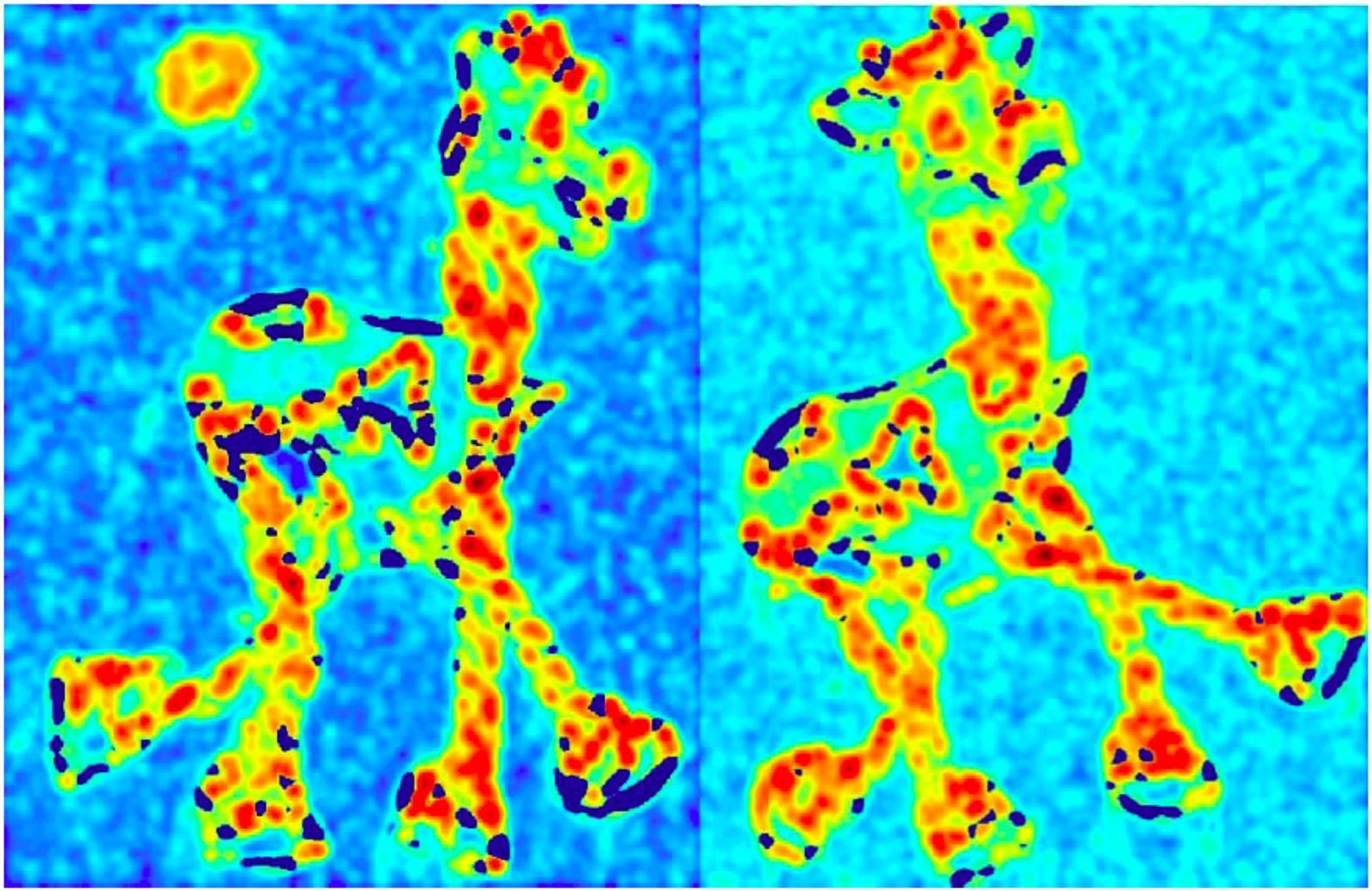
1. Compute gradients I_x and I_y
2. Compute I_x^2, I_y^2, $I_x \cdot I_y$
3. Average (Gaussian) $\rightarrow$ gives M per voxel
4. Compute $R = \det(M) - \alpha \,\mathrm{trace}(M)^2$ for each image window (*cornerness* score)
5. Find points with large R ($R >$ threshold).
6. Take only points of local maxima, i.e., perform non-maximum suppression



Example

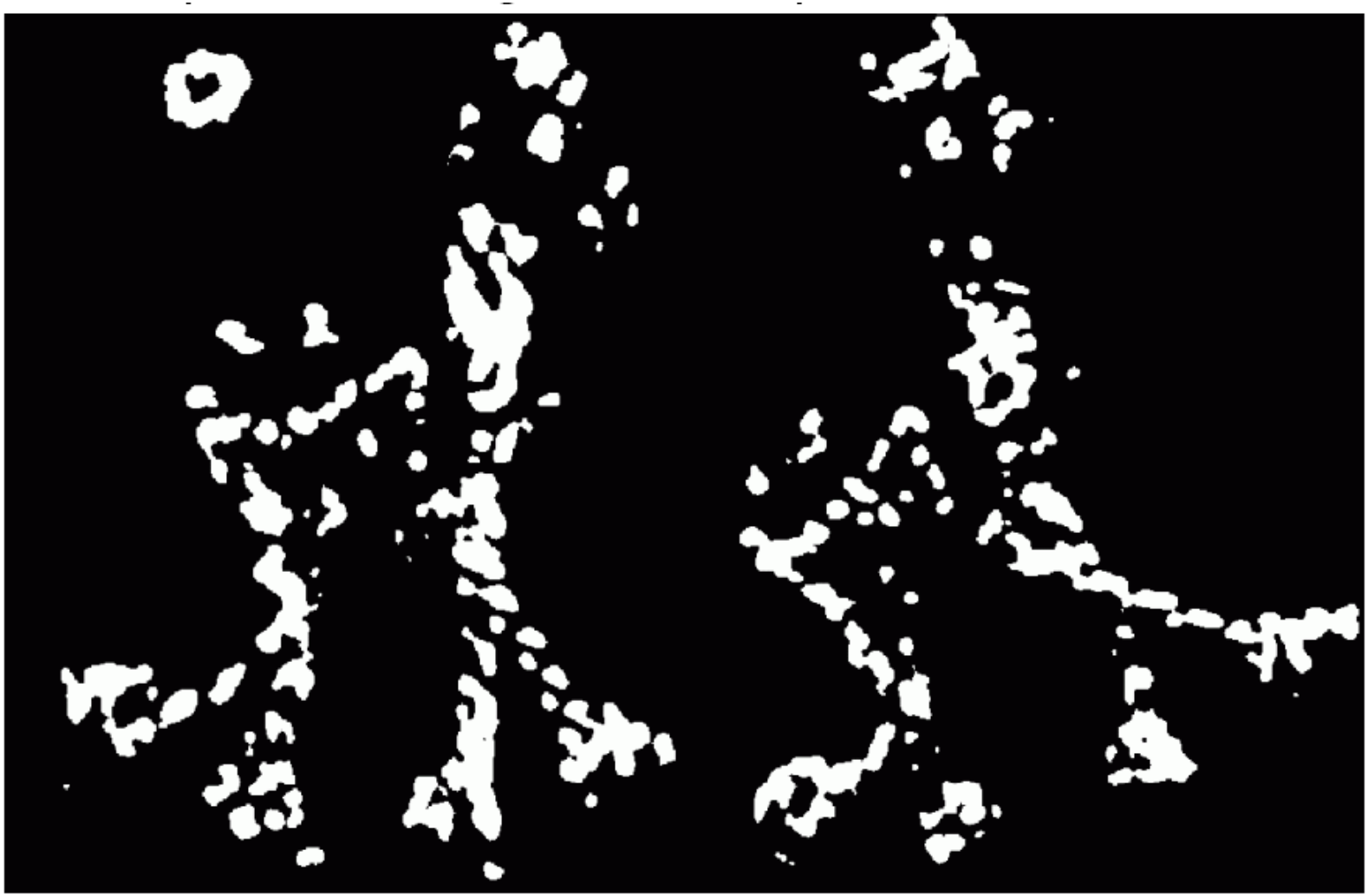
[Source: K. Grauman]

1) Compute Cornerness



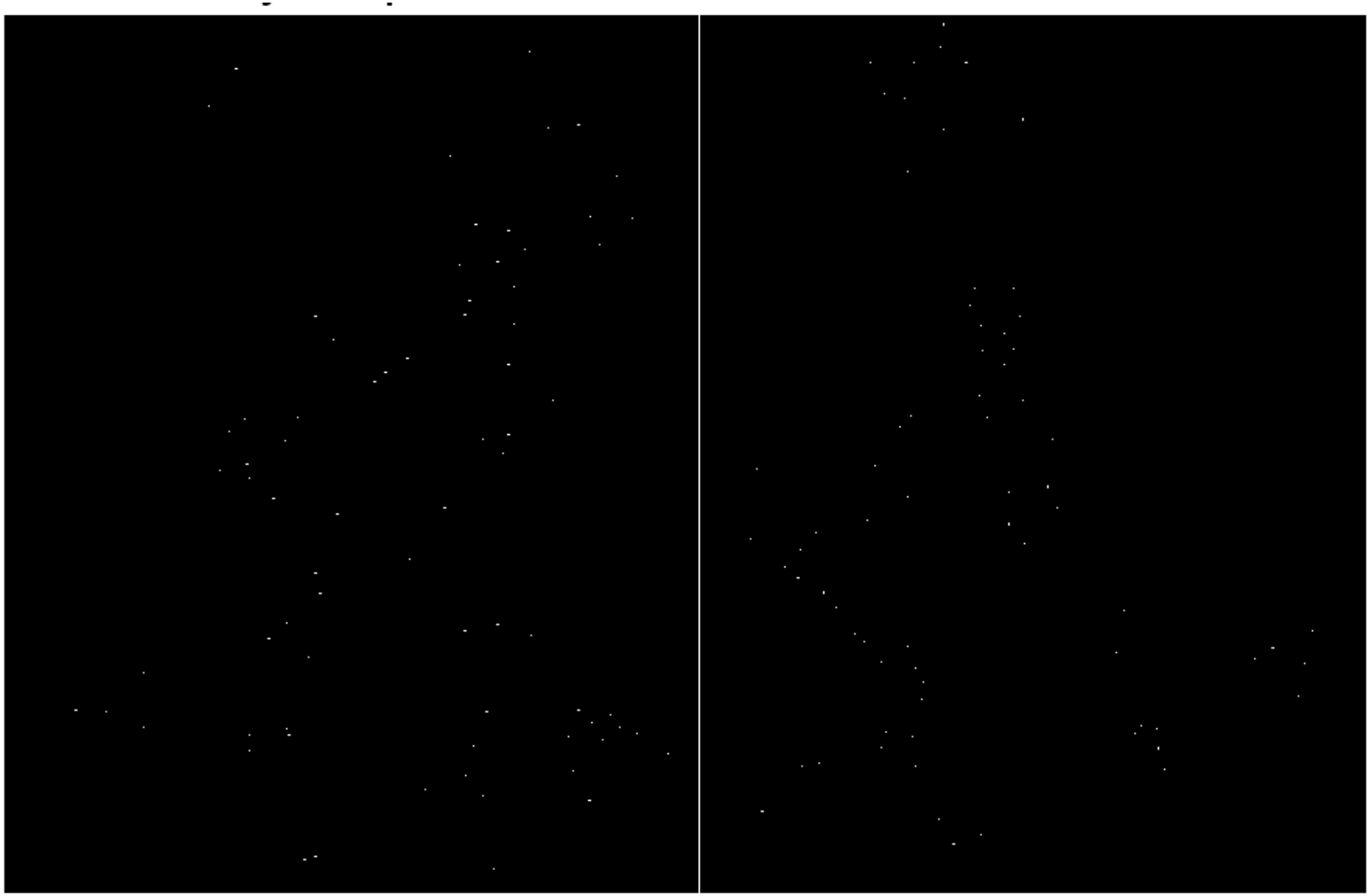
[Source: K. Grauman]

2) Find High Response



[Source: K. Grauman]

3) Non-maxima Suppresion



[Source: K. Grauman]

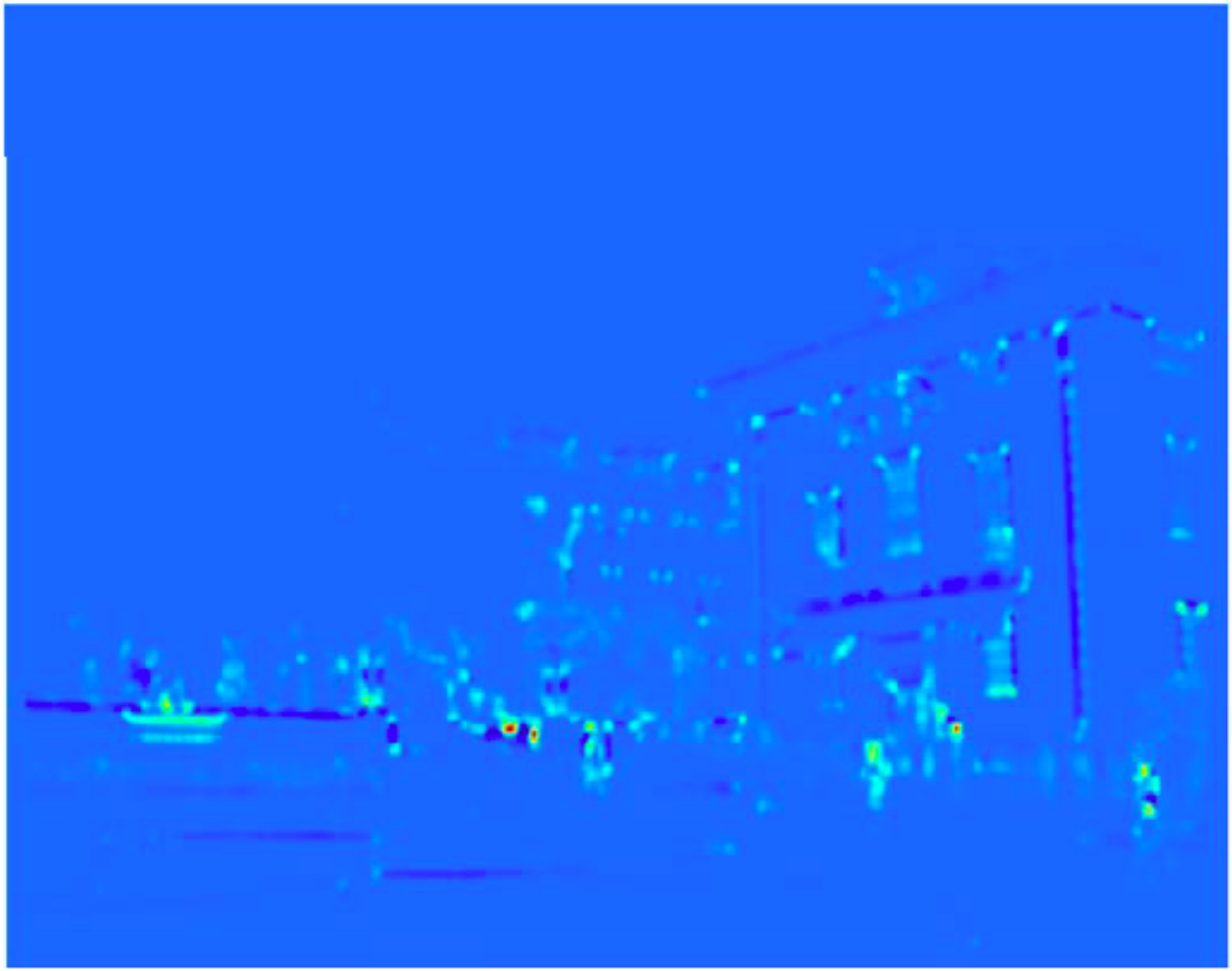
Results

[Source: K. Grauman]

Another Example

[Source: K. Grauman]

Cornerness



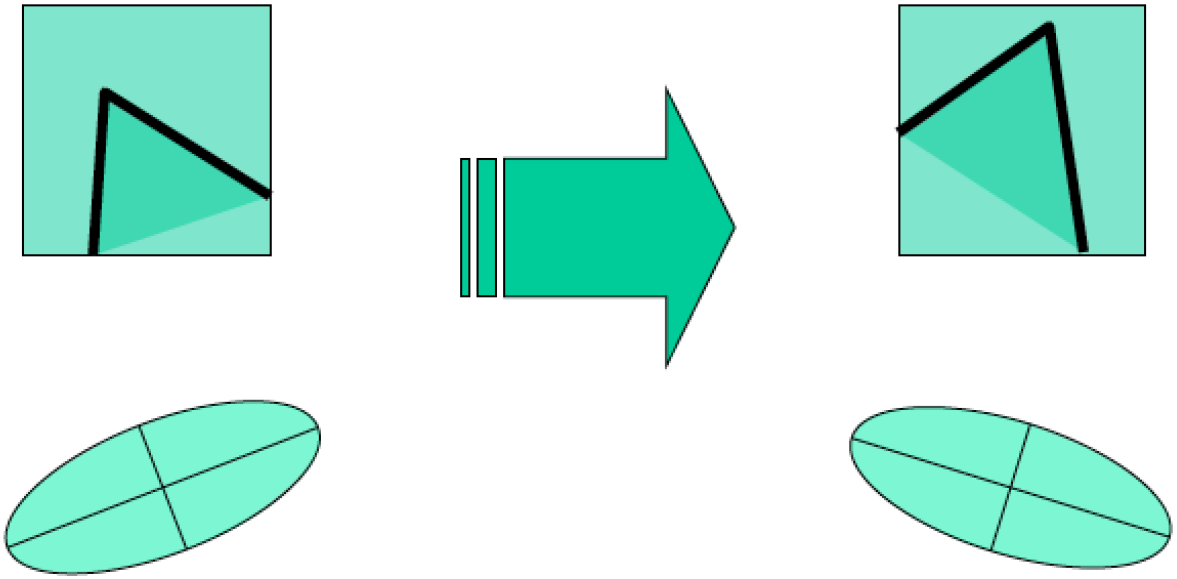
[Source: K. Grauman]

Interest Points

[Source: K. Grauman]

Properties of Harris Corner Detector

- Rotation and Shift Invariance of Corners

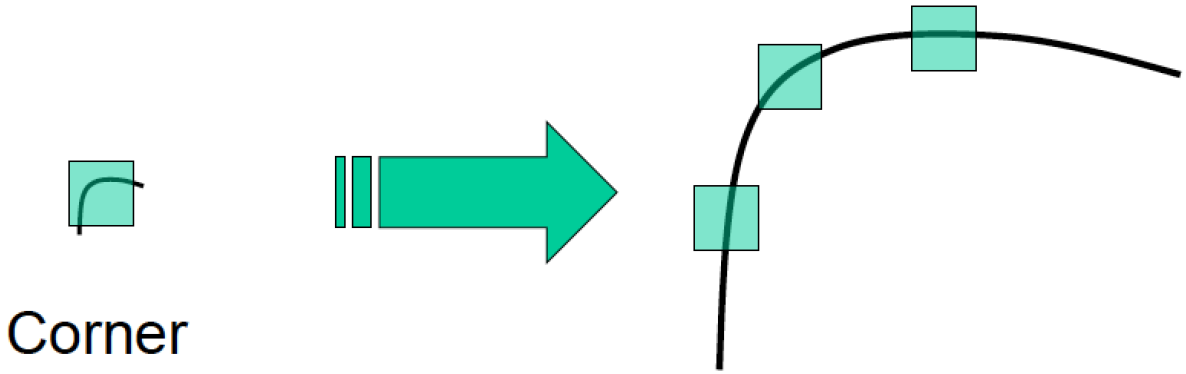


- Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same
- Harris corner detector is rotation-covariant

[Source: J. Hays]

Properties of Harris Corner Detector

- Scale?



- Corner location is **not scale invariant/covariant**!

[Source: J. Hays]

Next Time

- Can we also define keypoints that are shift, rotation and scale invariant/covariant?
- What should be our **description** around keypoint?