# CS 188-2
# Discussion-Week 5

Ali Hatamizadeh
10/25/2019

# Bilinear Interpolation

# Bilinear Interpolation

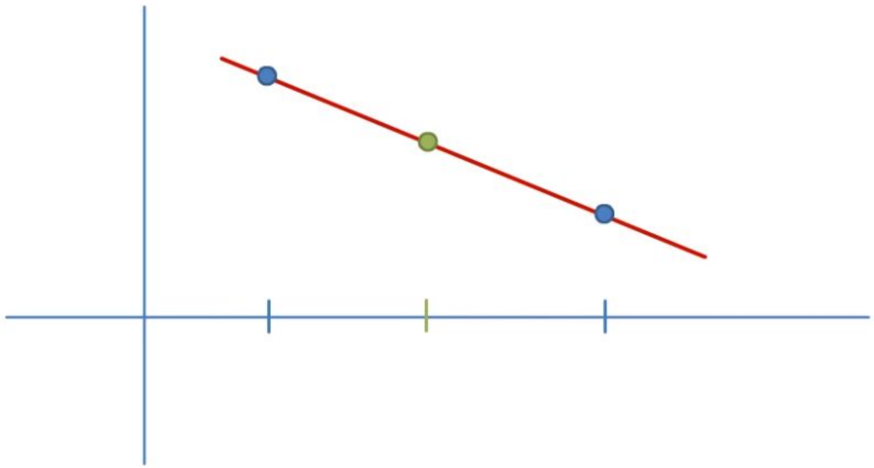
- From the first principle, what represents a line ?

$$m = \frac{(y - y_0)}{(x - x_0)}$$

$$(y - y_0) = m(x - x_0)$$

# Bilinear Interpolation

- Given the information of two points, we can use the equation of a line an solve for a point that is located anywhere on that line (for instance the middle point). This is called ***interpolation.***

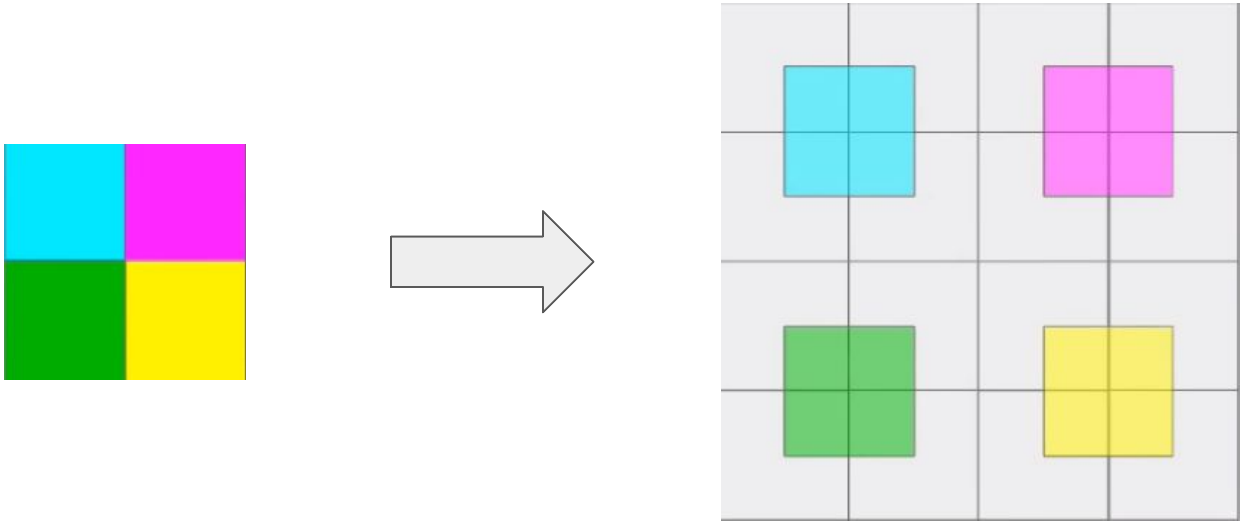


# Bilinear Interpolation

- Note that here we are dealing with two variables (e.g. x and y).
  - What is we had another variable added ?
- In the case of three variables, we need to use **bilinear interpolation.**
- The general idea is to hold two variable constant, perform a linear interpolation and repeat the process for the other set of two variables.
- Bilinear interpolation is a method commonly used for resizing an image.
- The 3D equivalent of bilinear interpolation is called **trilinear interpolation,** that follows the same strategy but deals with 4 variables ( one variable is a function of the other three).
- We focus on bilinear interpolation in this course.

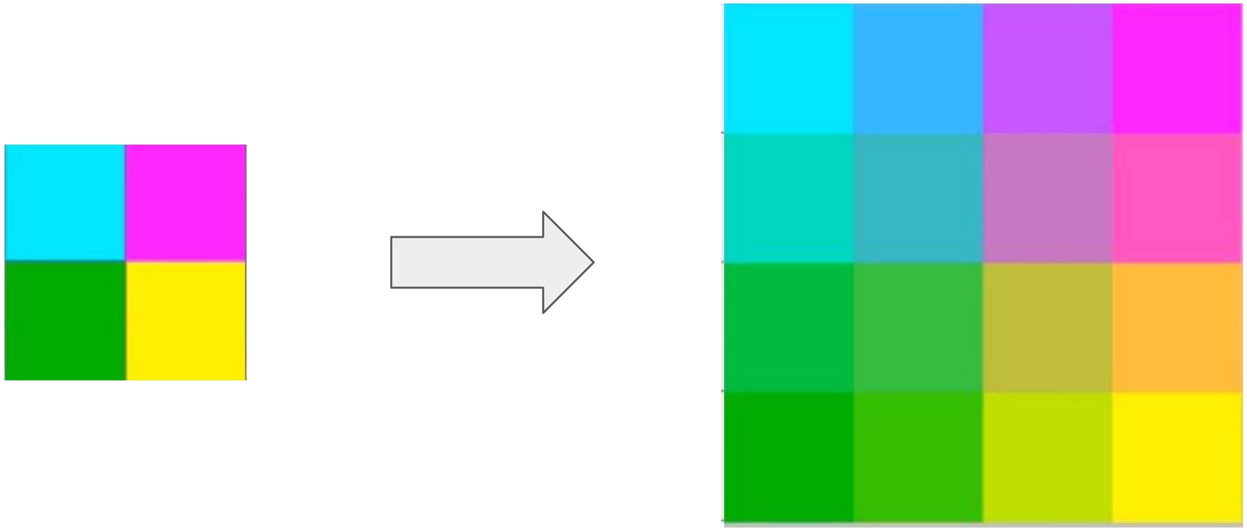
# Bilinear Interpolation

- Suppose, you want to upsample a given image as shown below. What's the intuition behind this process?



# Bilinear Interpolation

- In bilinear interpolation, each pixel looks at its 4 nearest neighboring pixels and takes into account the contribution by a weighted average.



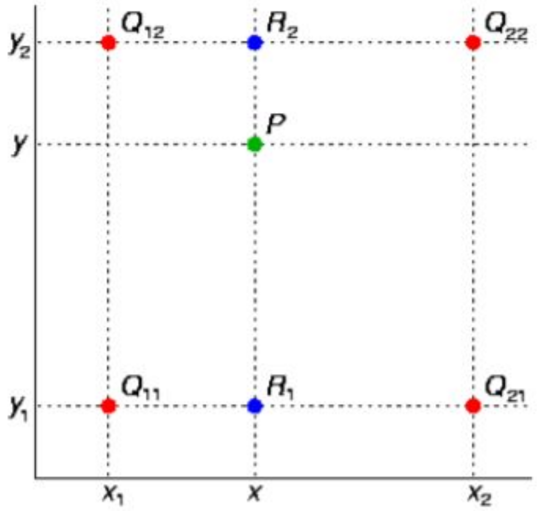
# Bilinear Interpolation

- Suppose we know the values at Q. If we were to interpolate for point P:
  - First interpolate horizontally to get $R_1$ and $R_2$ :

$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})$$
$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})$$

  - Then interpolate vertically to get P:

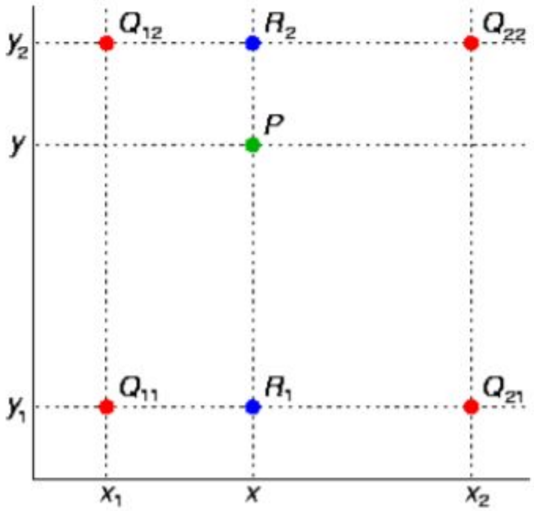
$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2).$$



# Bilinear Interpolation

- Assuming that $x_1 = 1, x_2 = 2, y_1 = 2, y_2 = 1, f(Q_{12}) = 100, f(Q_{22}) = 50, f(Q_{21}) = 70, f(Q_{11}) = 20$

  use bilinear interpolation to find the coordinates of point p(x=1.5,y=1.5).



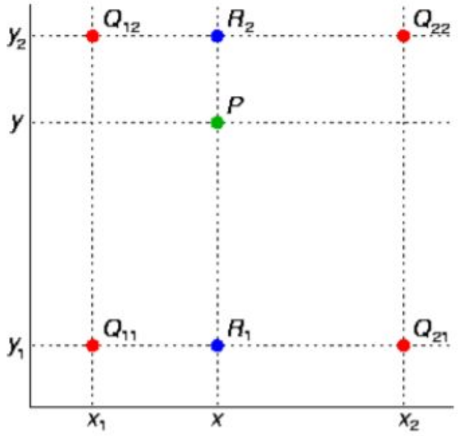
# Bilinear Interpolation

- Assuming that $x_1 = 1, x_2 = 2, y_1 = 2, y_2 = 1, f(Q_{12}) = 100, f(Q_{22}) = 50, f(Q_{21}) = 70, f(Q_{11}) = 20$

  use bilinear interpolation to find the coordinates of point p(x=1.5,y=1.5).

$$f(R_2) = (1.5 - 1)(100) + (2 - 1.5)(50) = 50 + 25 = 75$$

$$f(R_1) = (1.5 - 1)(20) + (2 - 1.5)(70) = 10 + 35 = 45$$



# Convolution

# What is a Convolution ?

- What does a convolution result in?
  - A convolution operation adds the contribution of each element in the image to its local neighbors and weights it by a kernel.
- How is it represented mathematically ?
  - For an original image of $x$ and a kernel filter of h:

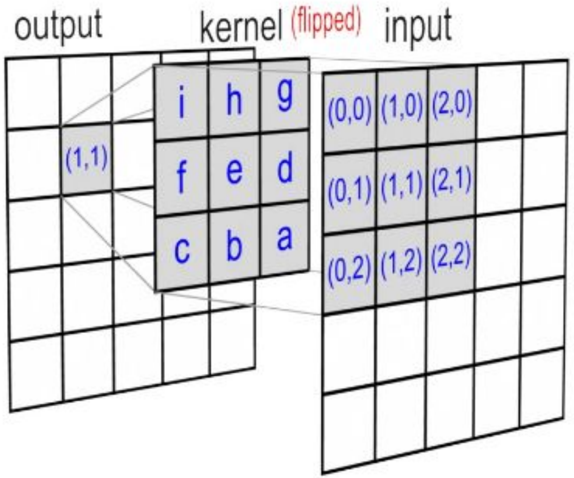
$$y[m,n] = x[m,n] * h[m,n] = \sum_{j=-a}^{a} \sum_{i=-b}^{b} x[i,j].\, h[m-i, n-j]$$

| n \ m | -1 | 0 | 1 |
|---|---|---|---|
| -1 | a | b | c |
| 0 | d | e | f |
| 1 | g | h | i |

$$\begin{aligned} y[1,1] &= \sum_{j=-1}^{1} \sum_{i=-1}^{1} x[i,j] \cdot h[1-i, 1-j] \\ &= x[0,0] \cdot h[1,1] + x[1,0] \cdot h[0,1] + x[2,0] \cdot h[-1,1] \\ &\quad + x[0,1] \cdot h[1,0] + x[1,1] \cdot h[0,0] + x[2,1] \cdot h[-1,0] \\ &\quad + x[0,2] \cdot h[1,-1] + x[1,2] \cdot h[0,-1] + x[2,2] \cdot h[-1,-1] \end{aligned}$$

# Convolution by hand

- You need to flip the filter horizontally and then vertically before multiplying it by the overlapped data.




| n \ m | -1 | 0 | 1 |
|---|---|---|---|
| -1 | a | b | c |
| 0 | d | e | f |
| 1 | g | h | i |

$$
\begin{aligned}
y[1,1] &= \sum_{j=-1}^{1} \sum_{i=-1}^{1} x[i,j] \cdot h[1-i, 1-j] \\
&= \quad x[0,0] \cdot h[1,1] \quad + x[1,0] \cdot h[0,1] \quad + x[2,0] \cdot h[-1,1] \\
&\quad + x[0,1] \cdot h[1,0] \quad + x[1,1] \cdot h[0,0] \quad + x[2,1] \cdot h[-1,0] \\
&\quad + x[0,2] \cdot h[1,-1] + x[1,2] \cdot h[0,-1] + x[2,2] \cdot h[-1,-1]
\end{aligned}
$$

# Why flipping the filter?

- Why flipping the filter is needed ? take a closer look:

$$y(n_1, n_2) = \sum_k \sum_l x(k,l)h(n_1 - k, n_2 - l))$$

As a special case, just take $n_1 = 0, n_2 = 0$. Then essentially you are computing $h(-k, -l)$

# Convolution by hand

- Let's do a convolution operation by hand. Given the following kernel and input images, convolve the image. Apply appropriate padding to match the sizes.

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

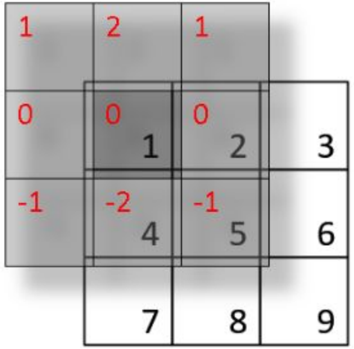
Input

| | | |
|---|---|---|
| -1 | -2 | -1 |
| 0 | 0 | 0 |
| 1 | 2 | 1 |

Kernel

# Convolution by hand-Solution

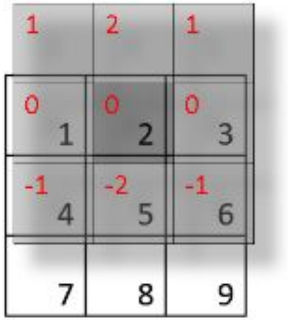
- First we **flip** the kernel. Then we pad the image where needed and overlay the kernel on top of image and start convolving:




$$
\begin{aligned}
y[0,0] &= \sum_j \sum_i x[i,j] \cdot h[0-i, 0-j] \\
&= \quad x[-1,-1] \cdot h[1,1] + x[0,-1] \cdot h[0,1] + x[1,-1] \cdot h[-1,1] \\
&\quad + x[-1,0] \cdot h[1,0] \quad + x[0,0] \cdot h[0,0] \quad + x[1,0] \cdot h[-1,0] \\
&\quad + x[-1,1] \cdot h[1,-1] + x[0,1] \cdot h[0,-1] + x[1,1] \cdot h[-1,-1] \\
&= \quad 0 \cdot 1 + 0 \cdot 2 + 0 \cdot 1 \\
&\quad + 0 \cdot 0 + 1 \cdot 0 + 2 \cdot 0 \\
&\quad + 0 \cdot (-1) + 4 \cdot (-2) + 5 \cdot (-1) \\
&= -13
\end{aligned}
$$

# Convolution by hand-Solution

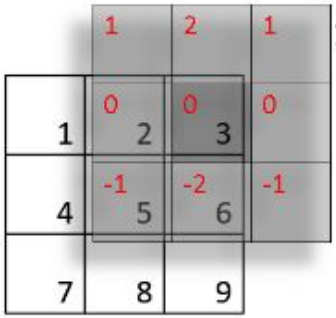
- 2nd element:




$$
\begin{aligned}
y[1,0] &= \sum_j \sum_i x[i,j] \cdot h[1-i, 0-j] \\
&= \quad x[0,-1] \cdot h[1,1] + x[1,-1] \cdot h[0,1] + x[2,-1] \cdot h[-1,1] \\
&\quad + x[0,0] \cdot h[1,0] \quad + x[1,0] \cdot h[0,0] \quad + x[2,0] \cdot h[-1,0] \\
&\quad + x[0,1] \cdot h[1,-1] + x[1,1] \cdot h[0,-1] + x[2,1] \cdot h[-1,-1] \\
&= \quad 0 \cdot 1 + 0 \cdot 2 + 0 \cdot 1 \\
&\quad + 1 \cdot 0 + 2 \cdot 0 + 3 \cdot 0 \\
&\quad + 4 \cdot (-1) + 5 \cdot (-2) + 6 \cdot (-1) \\
&= -20
\end{aligned}
$$

# Convolution by hand-Solution

- 3rd element:

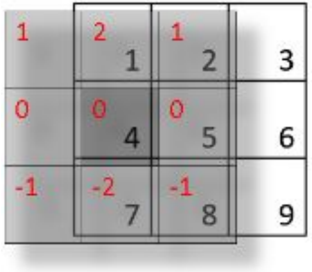



$$
\begin{aligned}
y[2,0] &= \sum_j \sum_i x[i,j] \cdot h[2-i, 0-j] \\
&= \quad x[1,-1] \cdot h[1,1] + x[2,-1] \cdot h[0,1] + x[3,-1] \cdot h[-1,1] \\
&\quad + x[1,0] \cdot h[1,0] \quad + x[2,0] \cdot h[0,0] \quad + x[3,0] \cdot h[-1,0] \\
&\quad + x[1,1] \cdot h[1,-1] + x[2,1] \cdot h[0,-1] + x[3,1] \cdot h[-1,-1] \\
&= \quad 0 \cdot 1 + 0 \cdot 2 + 0 \cdot 1 \\
&\quad + 2 \cdot 0 + 3 \cdot 0 + 0 \cdot 0 \\
&\quad + 5 \cdot (-1) + 6 \cdot (-2) + 0 \cdot (-1) \\
&= -17
\end{aligned}
$$

UCLA

# Convolution by hand-Solution

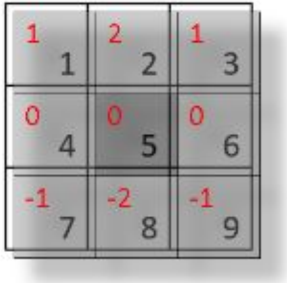
- 4th element:




$$
\begin{aligned}
y[0,1] &= \sum_j \sum_i x[i,j] \cdot h[0-i, 1-j] \\
&= \quad x[-1,0] \cdot h[1,1] \quad + x[0,0] \cdot h[0,1] \quad + x[1,0] \cdot h[-1,1] \\
&\quad + x[-1,1] \cdot h[1,0] \quad + x[0,1] \cdot h[0,0] \quad + x[1,1] \cdot h[-1,0] \\
&\quad + x[-1,2] \cdot h[1,-1] + x[0,2] \cdot h[0,-1] + x[1,2] \cdot h[-1,-1] \\
&= \quad 0 \cdot 1 + 1 \cdot 2 + 2 \cdot 1 \\
&\quad + 0 \cdot 0 + 4 \cdot 0 + 5 \cdot 0 \\
&\quad + 0 \cdot (-1) + 7 \cdot (-2) + 8 \cdot (-1) \\
&= -18
\end{aligned}
$$

# Convolution by hand-Solution

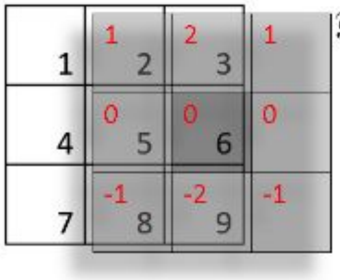
- 5th element:




$$
\begin{aligned}
y[1,1] &= \sum_j \sum_i x[i,j] \cdot h[1-i, 1-j] \\
&= \quad x[0,0] \cdot h[1,1] \quad + x[1,0] \cdot h[0,1] \quad + x[2,0] \cdot h[-1,1] \\
&\quad + x[0,1] \cdot h[1,0] \quad + x[1,1] \cdot h[0,0] \quad + x[2,1] \cdot h[-1,0] \\
&\quad + x[0,2] \cdot h[1,-1] + x[1,2] \cdot h[0,-1] + x[2,2] \cdot h[-1,-1] \\
&= \quad 1 \cdot 1 + 2 \cdot 2 + 3 \cdot 1 \\
&\quad + 4 \cdot 0 + 5 \cdot 0 + 6 \cdot 0 \\
&\quad + 7 \cdot (-1) + 8 \cdot (-2) + 9 \cdot (-1) \\
&= -24
\end{aligned}
$$

# Convolution by hand-Solution

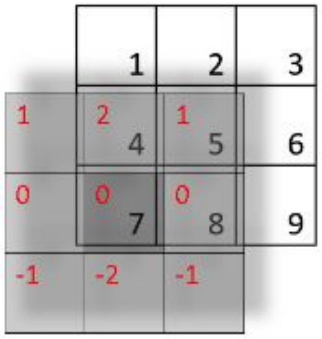
- 6th element:




$$
\begin{aligned}
y[2,1] &= \sum_j \sum_i x[i,j] \cdot h[2-i, 1-j] \\
&= \quad x[1,0]\cdot h[1,1] \quad + x[2,0]\cdot h[0,1] \quad + x[3,0]\cdot h[-1,1] \\
&\quad + x[1,1]\cdot h[1,0] \quad + x[2,1]\cdot h[0,0] \quad + x[3,1]\cdot h[-1,0] \\
&\quad + x[1,2]\cdot h[1,-1] + x[2,2]\cdot h[0,-1] + x[3,2]\cdot h[-1,-1] \\
&= \quad 2\cdot 1 + 3\cdot 2 + 0\cdot 1 \\
&\quad + 5\cdot 0 + 6\cdot 0 + 0\cdot 0 \\
&\quad + 8\cdot(-1) + 9\cdot(-2) + 0\cdot(-1) \\
&= -18
\end{aligned}
$$

# Convolution by hand-Solution

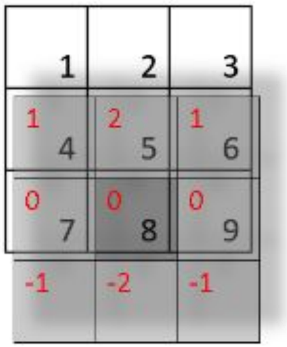
- 7th element:




$$
\begin{aligned}
y[0,2] &= \sum_j \sum_i x[i,j] \cdot h[0-i, 2-j] \\
&= \quad x[-1,1] \cdot h[1,1] \quad + x[0,1] \cdot h[0,1] \quad + x[1,1] \cdot h[-1,1] \\
&\quad + x[-1,2] \cdot h[1,0] \quad + x[0,2] \cdot h[0,0] \quad + x[1,2] \cdot h[-1,0] \\
&\quad + x[-1,3] \cdot h[1,-1] + x[0,3] \cdot h[0,-1] + x[1,3] \cdot h[-1,-1] \\
&= \quad 0 \cdot 1 + 4 \cdot 2 + 5 \cdot 1 \\
&\quad + 0 \cdot 0 + 7 \cdot 0 + 8 \cdot 0 \\
&\quad + 0 \cdot (-1) + 0 \cdot (-2) + 0 \cdot (-1) \\
&= 13
\end{aligned}
$$

# Convolution by hand-Solution

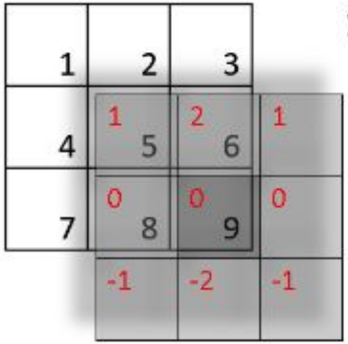
- 8th element:




$$
\begin{aligned}
y[1,2] &= \sum_j \sum_i x[i,j]\cdot h[1-i,2-j] \\
&= \quad x[0,1]\cdot h[1,1] \quad + x[1,1]\cdot h[0,1] \quad + x[2,1]\cdot h[-1,1] \\
&\quad + x[0,2]\cdot h[1,0] \quad + x[1,2]\cdot h[0,0] \quad + x[2,2]\cdot h[-1,0] \\
&\quad + x[0,3]\cdot h[1,-1] + x[1,3]\cdot h[0,-1] + x[2,3]\cdot h[-1,-1] \\
&= \quad 4\cdot 1 + 5\cdot 2 + 6\cdot 1 \\
&\quad + 7\cdot 0 + 8\cdot 0 + 9\cdot 0 \\
&\quad + 0\cdot(-1) + 0\cdot(-2) + 0\cdot(-1) \\
&= 20
\end{aligned}
$$

# Convolution by hand-Solution

- 9th element:




$$
\begin{aligned}
y[2,2] &= \sum_j \sum_i x[i,j] \cdot h[2-i, 2-j] \\
&= \quad x[1,1] \cdot h[1,1] \quad + x[2,1] \cdot h[0,1] \quad + x[3,1] \cdot h[-1,1] \\
&\quad + x[1,2] \cdot h[1,0] \quad + x[2,2] \cdot h[0,0] \quad + x[3,2] \cdot h[-1,0] \\
&\quad + x[1,3] \cdot h[1,-1] + x[2,3] \cdot h[0,-1] + x[3,3] \cdot h[-1,-1] \\
&= \quad 5 \cdot 1 + 6 \cdot 2 + 0 \cdot 1 \\
&\quad + 8 \cdot 0 + 9 \cdot 0 + 0 \cdot 0 \\
&\quad + 0 \cdot (-1) + 0 \cdot (-2) + 0 \cdot (-1) \\
&= 17
\end{aligned}
$$

# Convolution by hand-Solution

- Final Solution

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

Input

| | | |
|---|---|---|
| -1 | -2 | -1 |
| 0 | 0 | 0 |
| 1 | 2 | 1 |

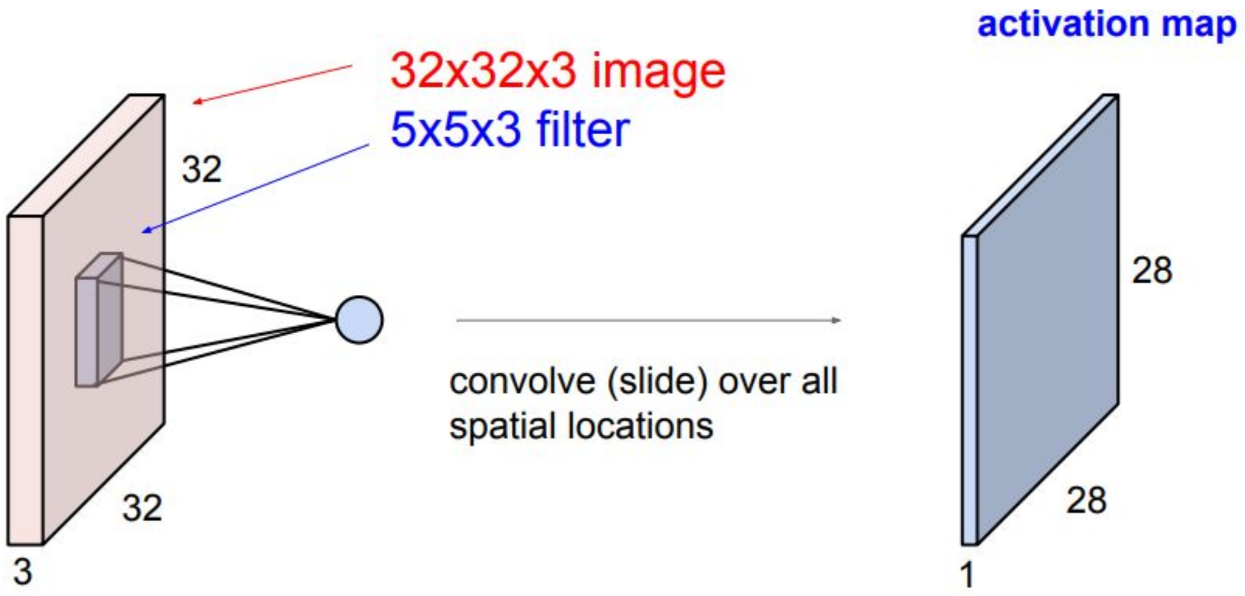
Kernel

| | | |
|---|---|---|
| -13 | -20 | -17 |
| -18 | -24 | -18 |
| 13 | 20 | 17 |

Output

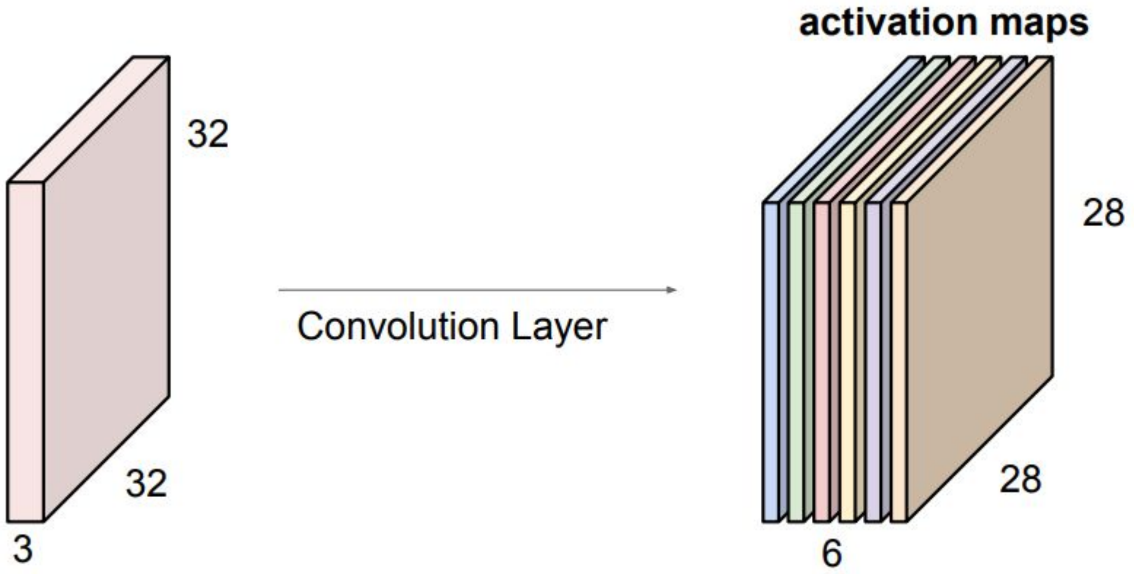
# Convolution of Colored Images

- Colored images have 3 channels. How's the convolution performed ?



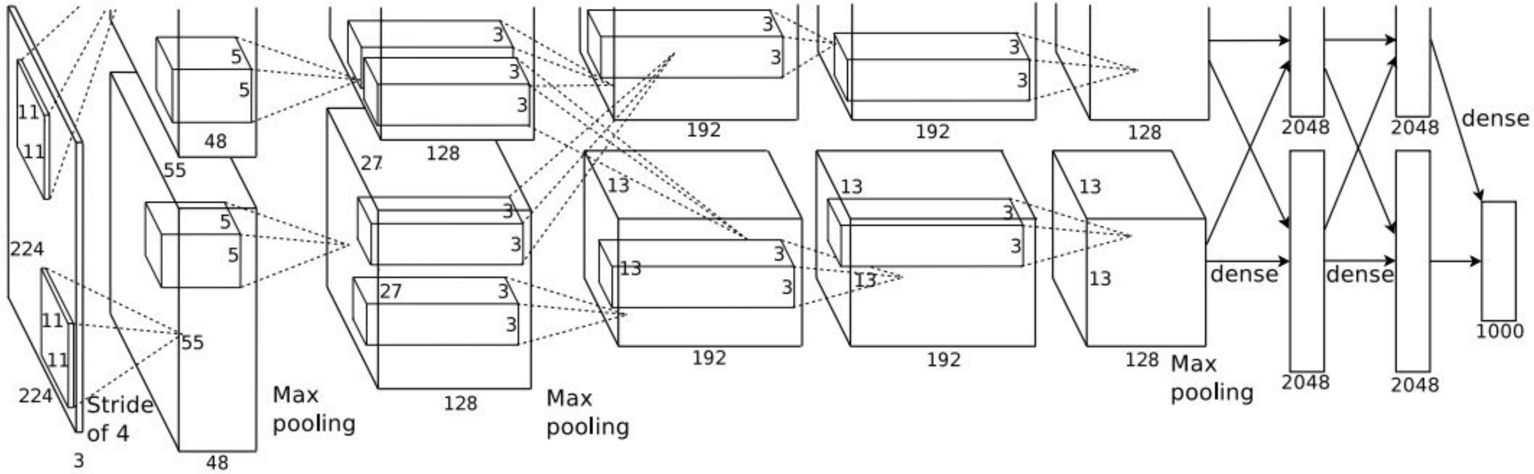
# Convolution of Colored Images

- Can we have more than 1 filter ?



# Convolutions in Deep Learning

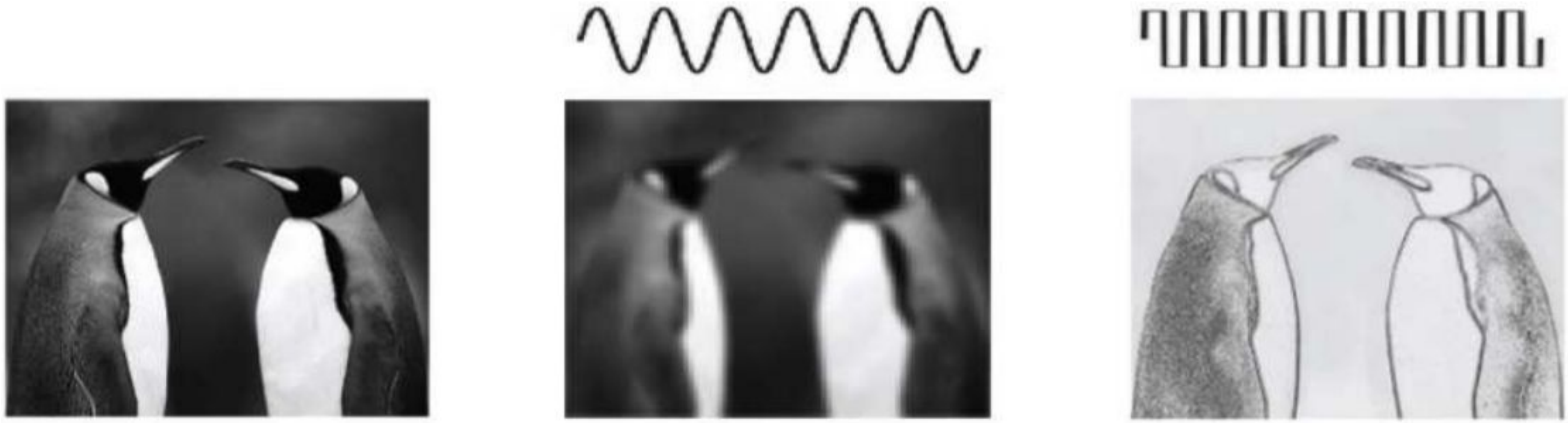
- This is the AlexNet architecture,winner of ImageNet 2012 Challenge.
- Can you identify the **number** of **convolutions** in the **first layer** ?



# Edge & Corner Detection

# Some Intuition

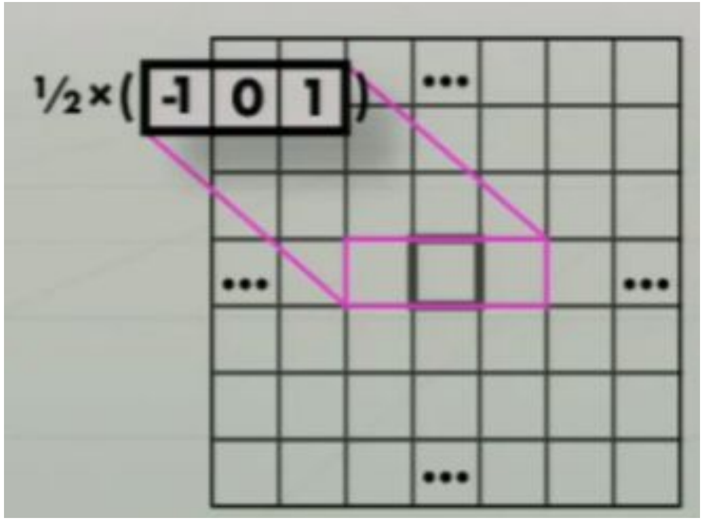
- Let's separate the image into low and high frequency spectrums:

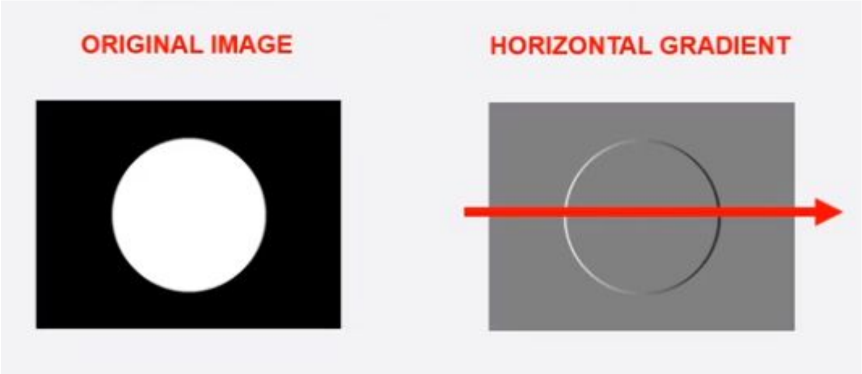


- Low frequencies encode the global structures while high frequencies encode the details and sharp edges !

# Edge Detection

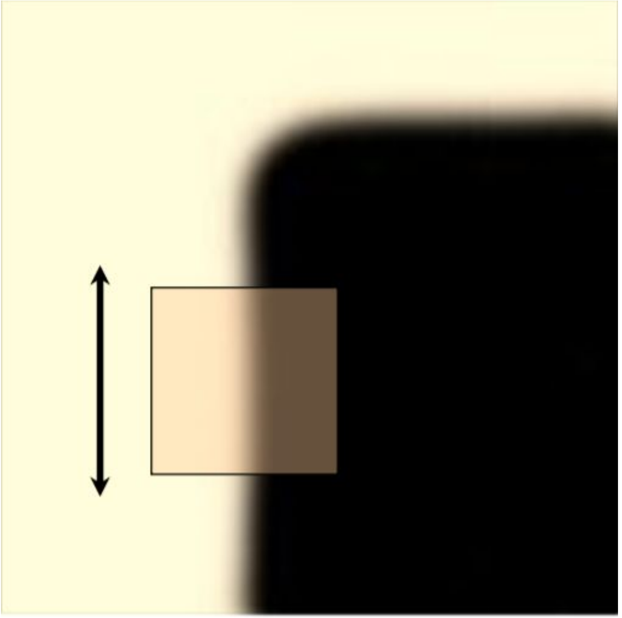
- One idea is to take the first derivative of the image and find the areas in which we have the highest response ( why ? )
- For example in the x-direction:



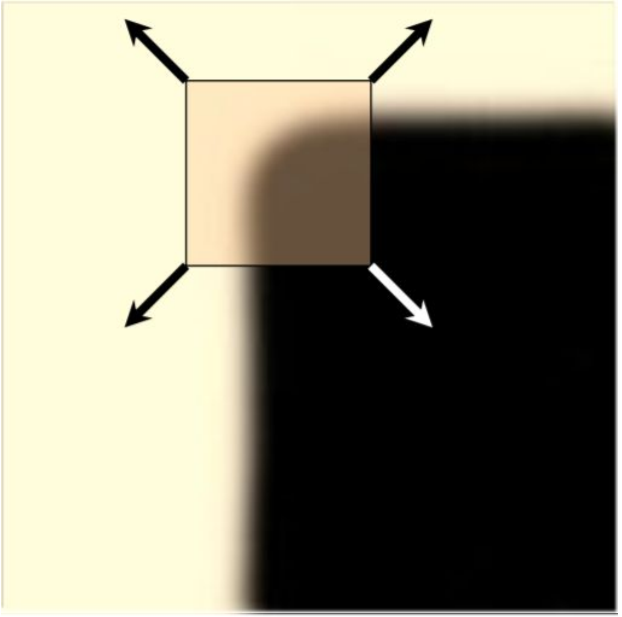
# Edge Detection

- In particular, no changes occur along the edge direction, while the maximum change occurs in the direction perpendicular to the edge.



# Corner Detection

- How about the corners ? in this case we have the highest changes in all directions.



UCLA

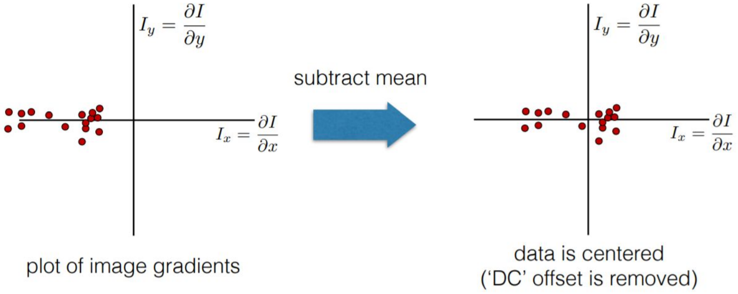
# Harris Corner Detection

- We attempt to check if the condition of **significant changes in all directions** holds
- The loss function for Harris corner detection, for change of intensity in [u,v] can be summarized as:

$$E(u,v) = \sum_{x,y} \underbrace{w(x,y)}_{\text{Window function}} \left[ \underbrace{I(x+u, y+v)}_{\text{Shifted intensity}} - \underbrace{I(x,y)}_{\text{Intensity}} \right]^2$$

# Steps in Harris Corner Detection

- **Step 1**: Compute image gradients over small region ( Error can be approximated by taking a Taylor Series Expansion for only small shifts)
- **Step 2**: Subtract mean from each image gradient (what does dc mean here?)




- **Step 3**: Compute the covariance matrix (how did we get such a matrix intuitively ?)

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

# Steps in Harris Corner Detection

- **Step 4**: Compute eigenvectors and eigenvalues (the following is the quadratic form that approximates E(u,v) )

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

eigenvalue

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

eigenvector

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

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

# Steps in Harris Corner Detection

- **Step 4**: Compute eigenvectors and eigenvalues boils down to:

eigenvalue

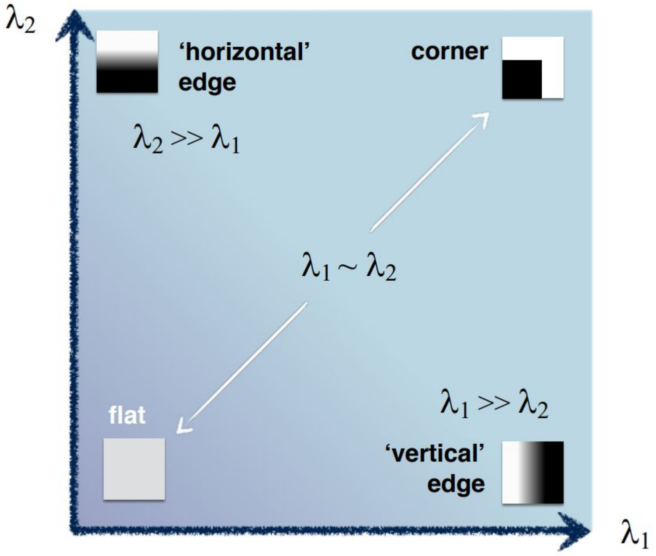
$$Me = \lambda e \qquad\qquad (M - \lambda I)e = 0$$

eigenvector

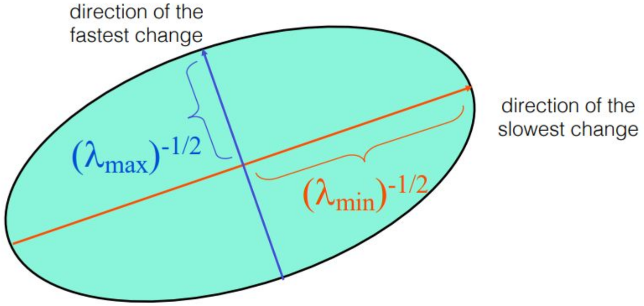
1. Compute the determinant of $M - \lambda I$
   (returns a polynomial)
2. Find the roots of polynomial $\det(M - \lambda I) = 0$
   (returns eigenvalues)
3. For each eigenvalue, solve $(M - \lambda I)e = 0$
   (returns eigenvectors)

# Steps in Harris Corner Detection

- **Step 4**: Interpret eigenvalues:
  - If $\lambda_2 >> \lambda_1$ then: horizontal edge( $I_y$ should have higher variance)
  - If $\lambda_1 >> \lambda_2$ then: vertical edge( $I_x$ should have higher variance)




$$[u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const} \qquad M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$



# Steps in Harris Corner Detection

- **Step 4**: Use a threshold on a response function of eignevalues to detect the corners :
  - Just a simple min function $R = \min(\lambda_1, \lambda_2)$
  - Eigenvalues greater than one $R = \lambda_1\lambda_2 - \kappa(\lambda_1 + \lambda_2)^2$
  - Or other choices in the literature

Harris & Stephens (1988)

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

Kanade & Tomasi (1994)

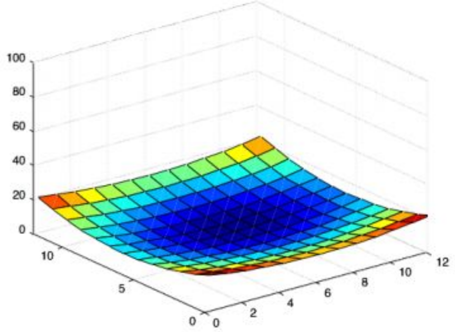
$$R = \min(\lambda_1, \lambda_2)$$

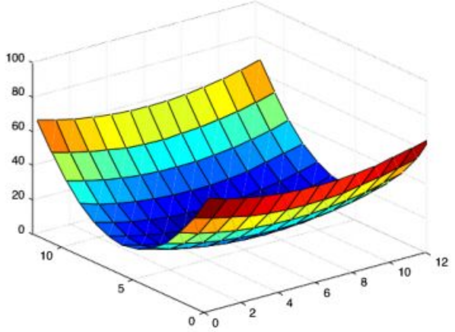
Nobel (1998)

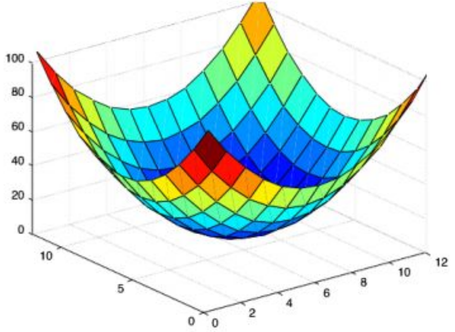
$$R = \frac{\det(M)}{\operatorname{trace}(M) + \epsilon}$$

# Corner Detection

- Which error function corresponds to only an **edge line** ?

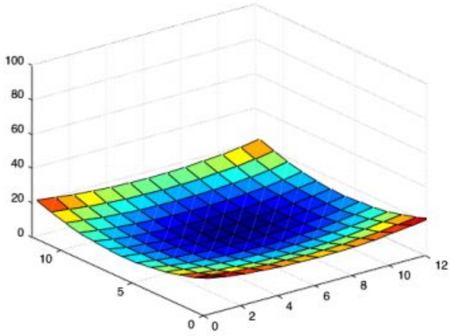


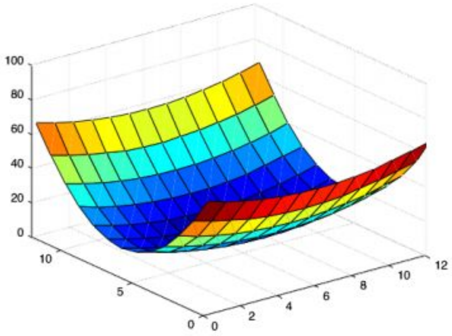



# Corner Detection

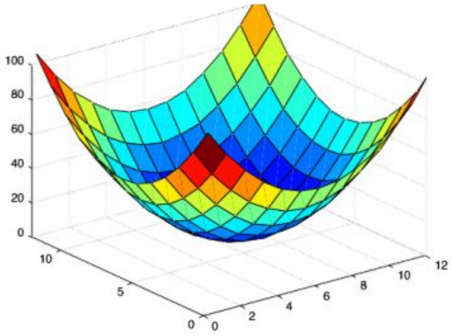
- Which error function corresponds to only an **edge line** ?




flat




edge
‘line’




corner
‘dot’

# Corner Detection

- Is Harris corner detection invariant to rotation ?

# Corner Detection

- Is Harris corner detection invariant to rotation ?
    - Yes ! eigenvalues remain the same!

# Corner Detection

- Is Harris corner detection invariant to scale ?

# Corner Detection

- Is Harris corner detection invariant to scale ?
  - No! curvature can vary across scales and one corner might not be considered as a corner in different scales

# Corner Detection

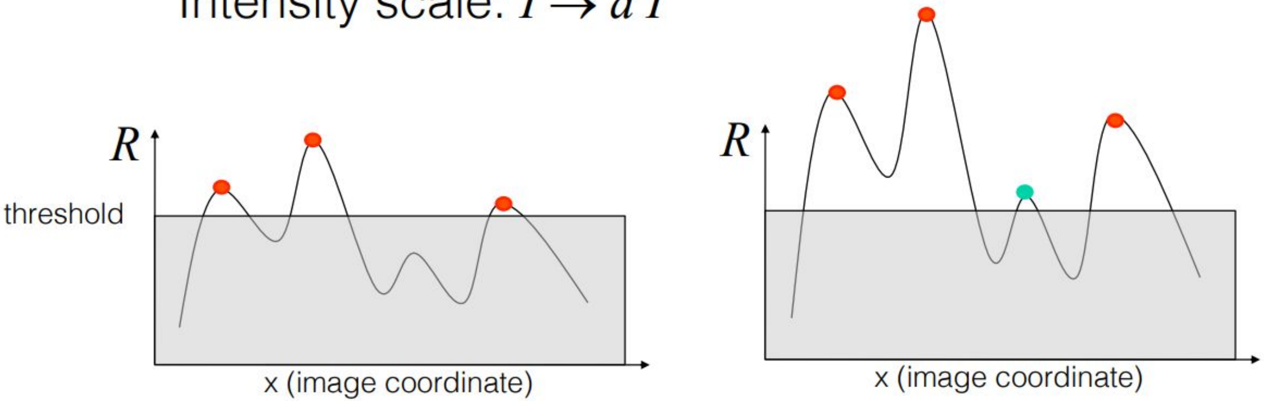
- Is Harris corner detection invariant to intensity changes?

# Corner Detection

- Is Harris corner detection invariant to intensity changes?
  - It’s partially invariant to affine intensity changes.

intensity shift $I \rightarrow I + b$

Intensity scale: $I \rightarrow a\,I$



# Thank you!