COMP3065
Computer Vision

Topic 2 – Describing Image Regions and Patches

Dr. Tianxiang Cui

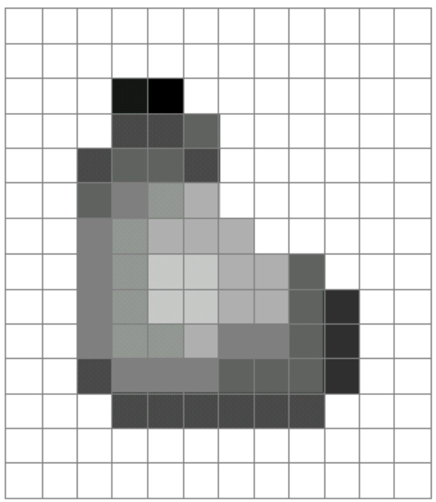
2025 Spring

Outline

- Image and its feature descriptor
- Common features used in CV:
  - Colour features
  - Texture features
  - Shape features
  - Edge features
- Some common feature vectors
  - Colour histograms
  - Local binary patterns
  - Histograms of Gradient Orientations (HoG)

What is an Image?

- A grid (matrix) of intensity values



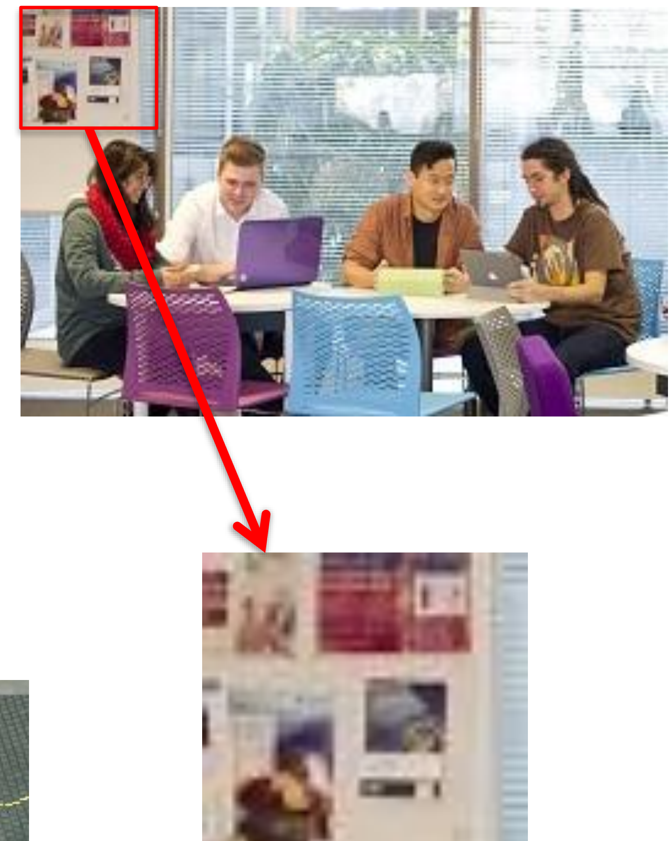
=

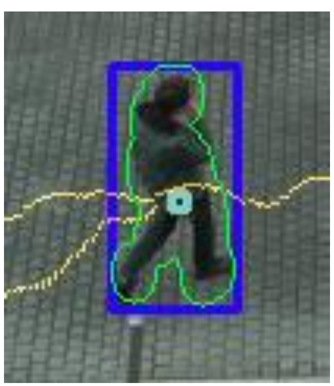
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	20	0	255	255	255	255	255	255	255
255	255	255	75	75	75	255	255	255	255	255	255
255	255	75	95	95	75	255	255	255	255	255	255
255	255	96	127	145	175	255	255	255	255	255	255
255	255	127	145	175	175	175	255	255	255	255	255
255	255	127	145	200	200	175	175	95	255	255	255
255	255	127	145	200	200	175	175	95	47	255	255
255	255	127	145	145	175	127	127	95	47	255	255
255	255	74	127	127	127	95	95	95	47	255	255
255	255	255	74	74	74	74	74	74	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255

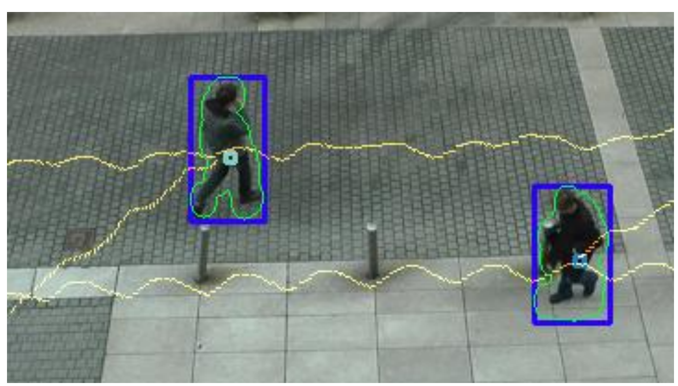
- Common to use one byte per value: 0 = black, 255 = white

Regions and Patches

- **Segments** (irregular) or rectangular image **patches** are widely used in computer vision tasks.
  - Matched between views to recover 3D.
  - Matched over time to track objects.
  - Compared to pre-stored models to detect object classes and recognize specific objects.
- To do this we need to describe them.

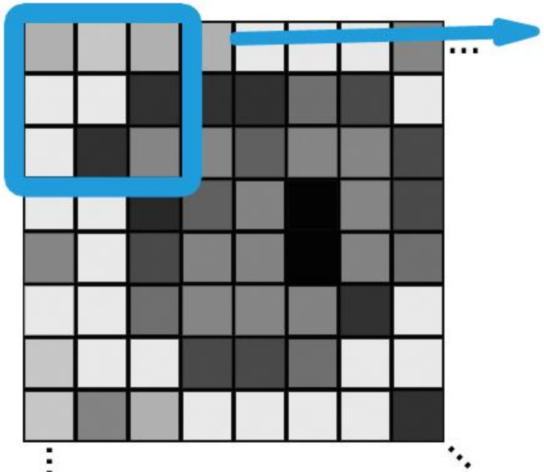






Features and Feature Vectors

- We need to define a set of descriptive ***features*** and concatenate them to produce a feature vector, e.g.



```
176, 199, 176
232, 232,  49
233,  49, 133

mean = 164
median = 176
max = 233
```

[176, 199, 176, 232, 232, 49, 233, 49, 133, 164, 176, 184]

- We should choose features that reflect the relevant properties of the viewed object, such as **color** features, **texture** features, **shape** features, etc.

Traditional CV vs. DL CV

- DL is sometimes overkill – traditional CV techniques can solve a problem much more efficiently (e.g., classify lemon and orange)
- Traditional CV techniques are not class-specific, they are very general and perform the same for any image – features learned from a deep neural net are specific to your training dataset
- Traditional CV techniques often used for the applications like image stitching/3D mesh reconstruction which don't require specific class knowledge – can also be achieved by training large datasets, require huge effort
- Traditional CV techniques can be deployed on low-cost microcontrollers

Colour Features

- Colour correlates well with class identity.

- Human vision works hard to preserve colour constancy: presumably because colour is useful.

- Histograms
  - Are invariant to translation and rotation.
  - Change slowly as viewing direction changes.
  - Change slowly with object size.
  - Change slowly with occlusion.

- Colour histograms summarise target objects quite well, and should match a good range of images.

Colour Histograms

- Histogram
  - X-axis: bins of intensity (colour) value intervals
  - Y-axis: number of pixels whose value falls into those bins.

- Which colour space? – depend on colour models
  - RGB: red, green, blue channels.
  - YUV: Y (luma), U (chrominance), V (chrominance) channels.
- How many bins? 256 (0 – 255) or 32 (0-7, 8-15, …)

Texture Features

- Colour is a property of a single pixel, texture features capture the frequency with which patterns of colour/grey level appear.
- E.g. Local Binary Patterns (LBP)
  - For each pixel p, create an 8-bit number $b_1\, b_2\, b_3\, b_4\, b_5\, b_6\, b_7\, b_8$, where $b_i=0$ if neighbor i has value less than or equal to p's value and 1 otherwise.

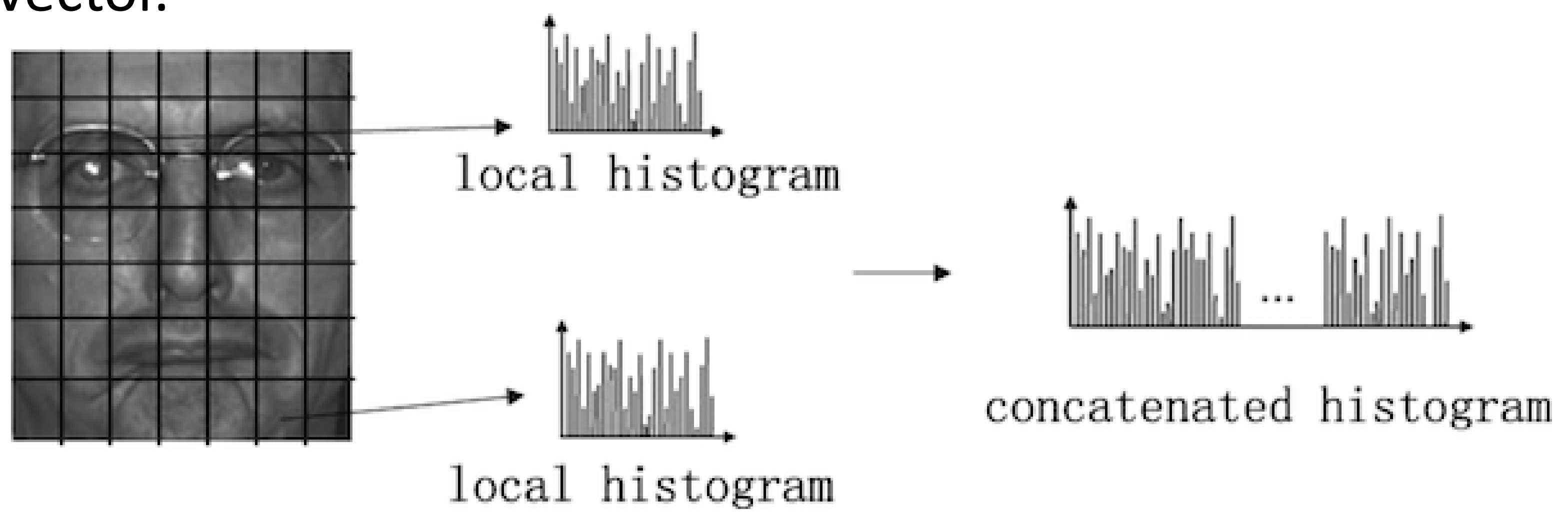
	1	2	3	
	100	101	103	
8	40	50	80	4
	50	60	90	5
	7	6		

$\longrightarrow$ $\boxed{1\ 1\ 1\ 1\ 1\ 1\ 0\ 0}$

Captures the pattern around one pixel

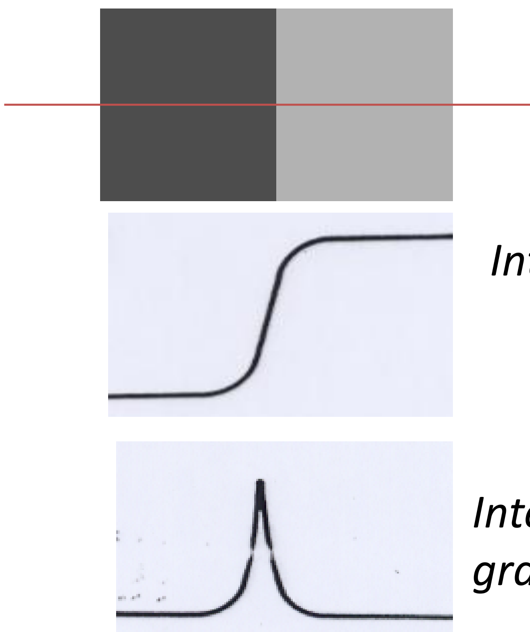
LBP Feature Vector

- Divide the patch into cells e.g. 16 x 16 pixels per cell.
- Compute the local patch description number of each pixel.
  - As described in previous slide.
- Histogram these numbers over each cell.
  - Usually a 256-d feature vector.
- Optionally normalize each histogram (so its bins sum to 1).
- Concatenate (normalized) histograms to make the feature vector.



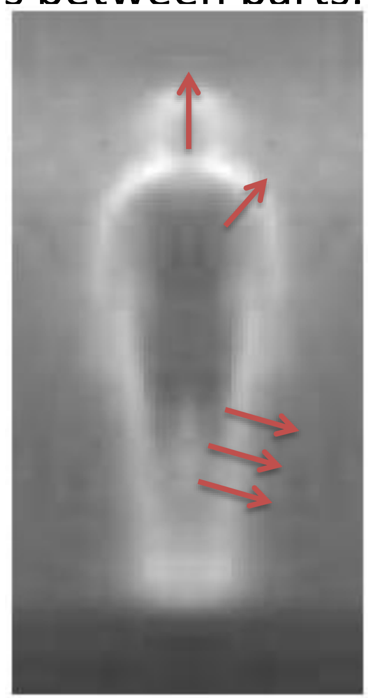
Shape Features

- Focus on **image gradient** measures:
  - The gradient of an image measures how it is changing.
  - The boundaries of objects are often associated with large gradients.
  - Distributions of gradients and gradient orientations reflect boundary shape (and internal boundaries between parts, surfaces, etc.).



Intensity

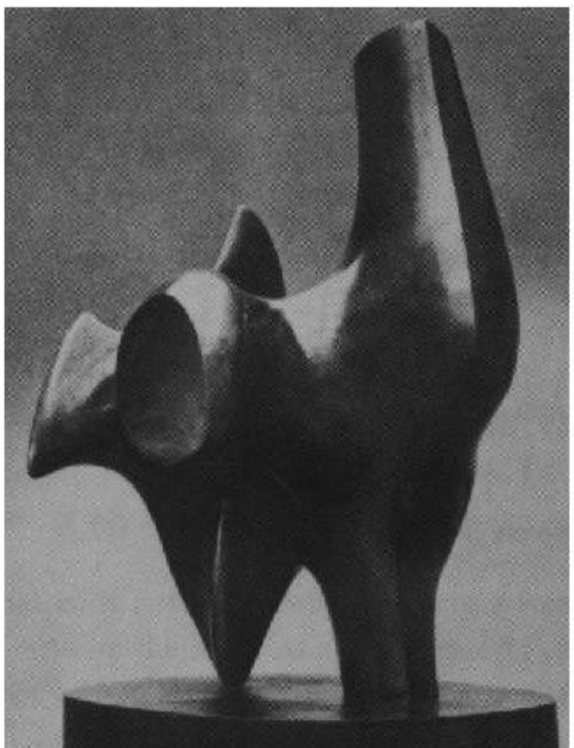
Intensity gradient

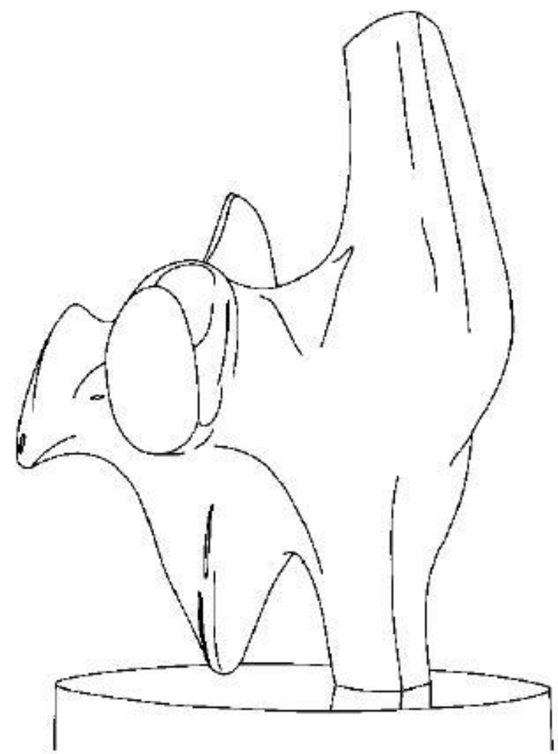


Mean gradient of a large set of person images

Edge Detection

- Convert a 2D image into a set of curves
  - Extracts salient features of the scene
  - More compact than pixels





Characterizing Edges

- An edge is a place of rapid change in the image intensity function

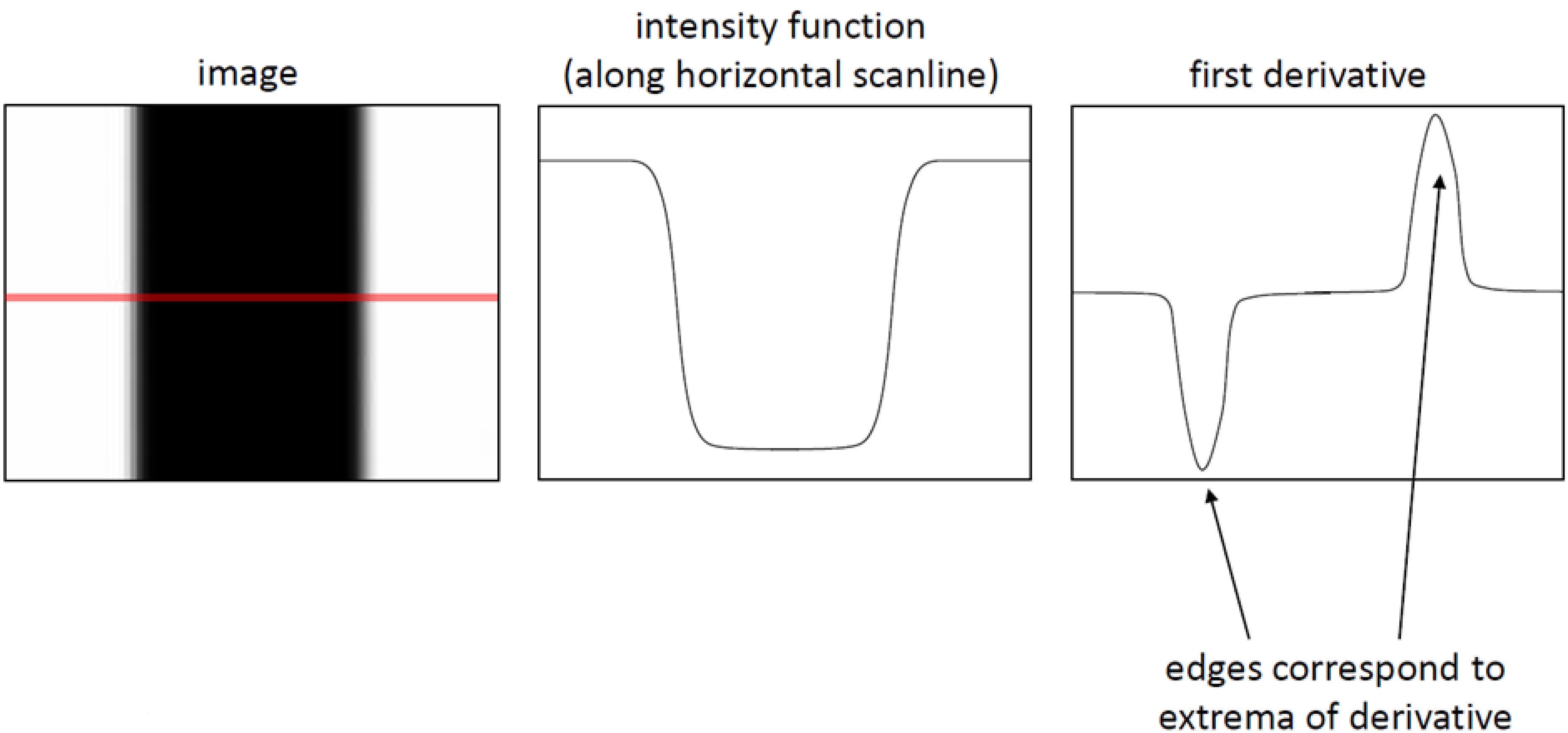


Image Derivatives

- In calculus, the derivative of a function represents its rate of change. For continuous valued functions:

$$\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x)$$

- With image data, the smallest possible delta *x* is 1

$$\frac{df}{dx} = \frac{f(x) - f(x-1)}{1} = f'(x)$$

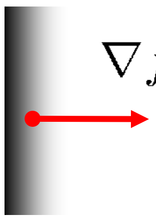
$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$

- Image gradient: a vector to measure the change in pixel values along the x-direction and the y-direction around each pixel

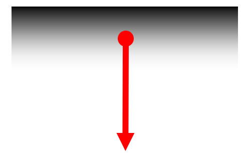
Image Gradient

- The gradient of an image:

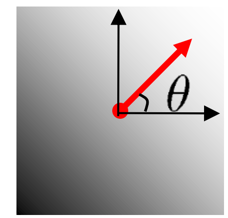
$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

- 

$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$



$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

The gradient points in the direction of most rapid increase in intensity

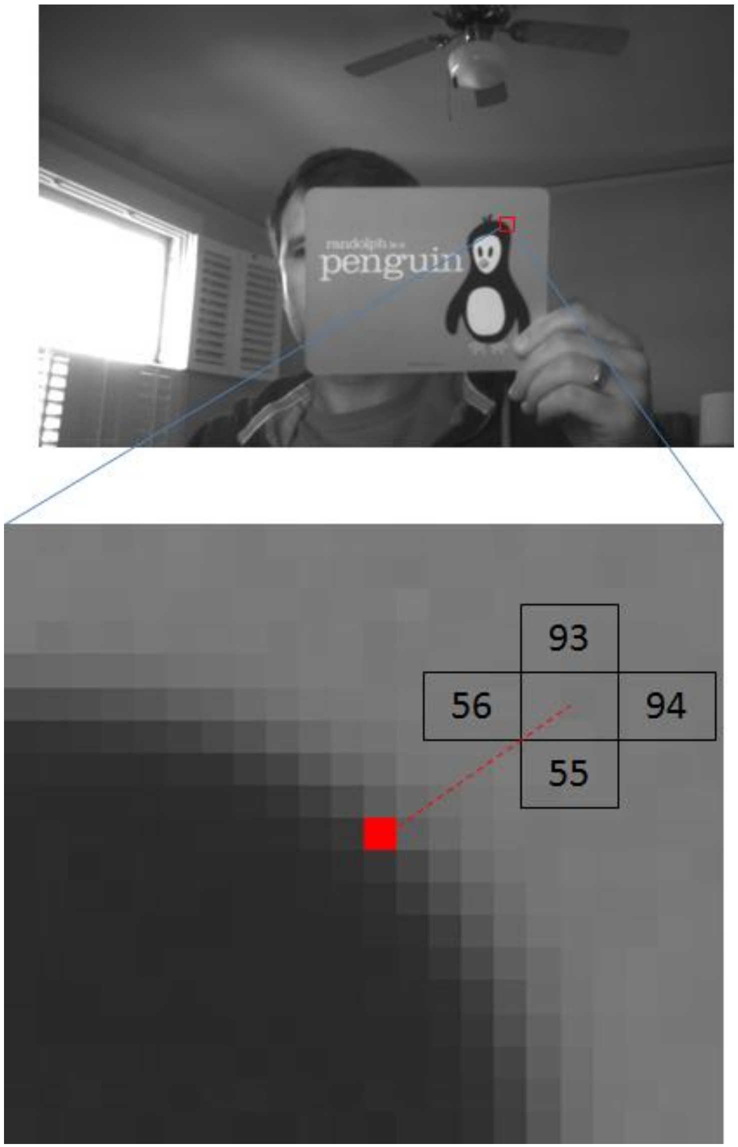
The gradient direction is given by:

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

The *edge strength* is given by the gradient magnitude

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Image Gradient: Example



Magnitude = $\sqrt{(38)^2+(38)^2} = 53.74$

Angle = $\arctan\left(\frac{38}{38}\right)$ = 0.785 rads

= 45 degrees

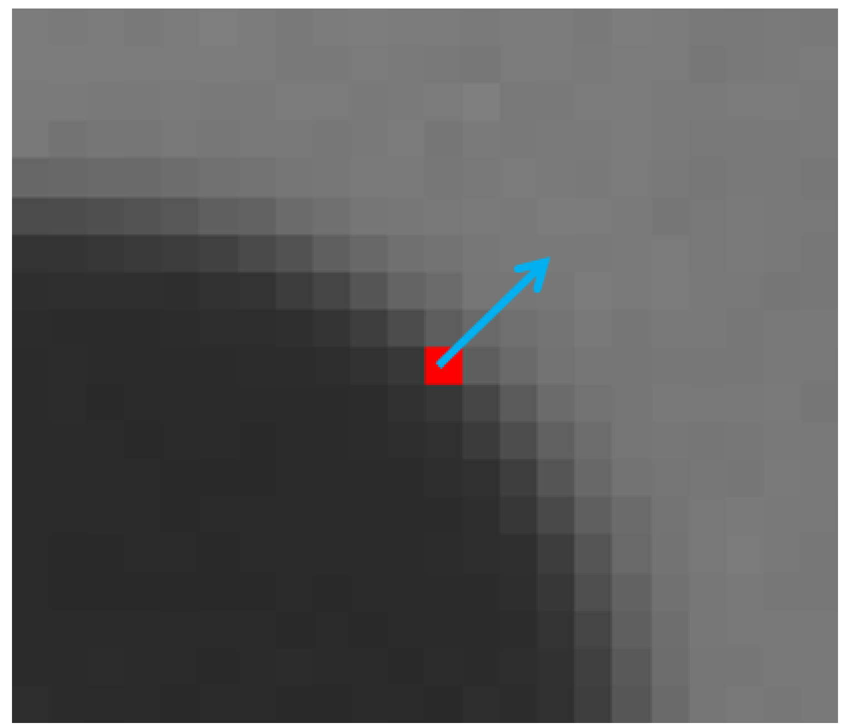
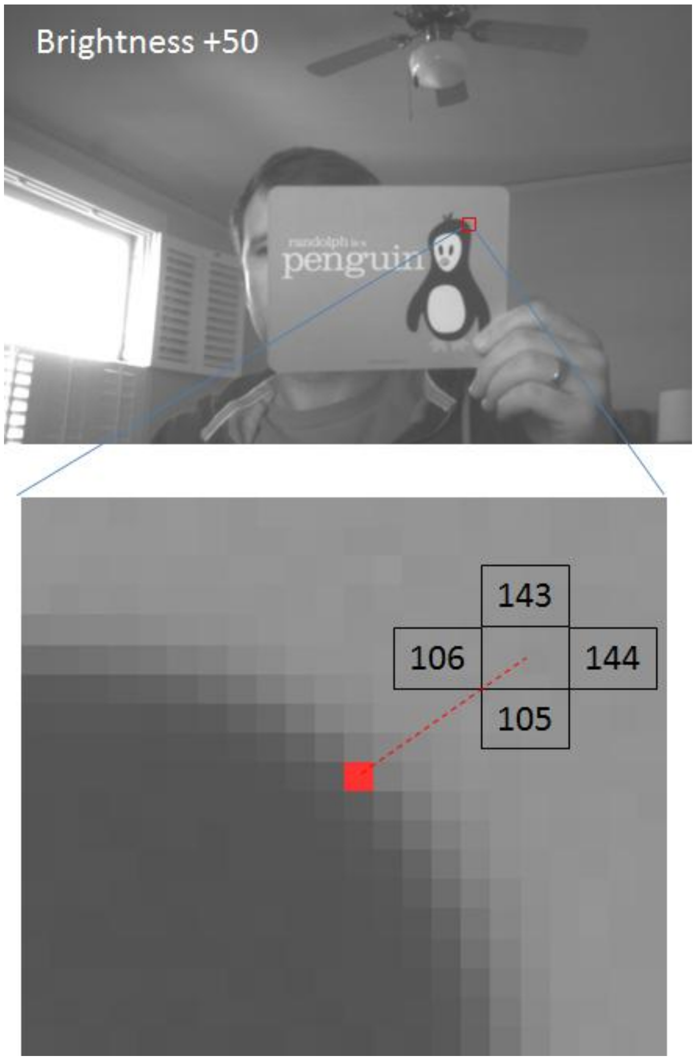


Image Gradient: Example

Change in x-direction

Change in y-direction

Image Gradient: Example



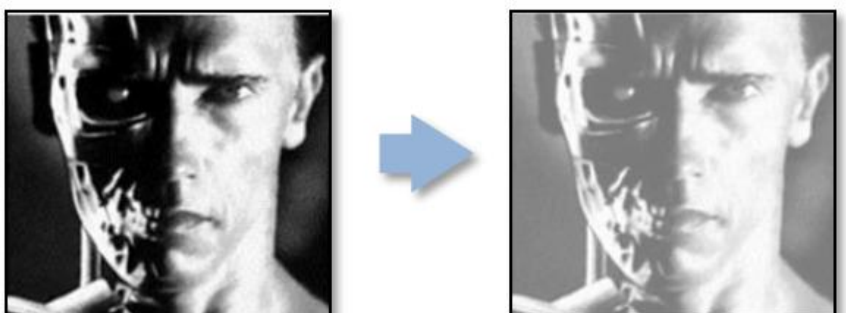
Magnitude = $\sqrt{(38)^2+(38)^2} = 53.74$

Angle = $\arctan\left(\frac{38}{38}\right)$ = 0.785 rads

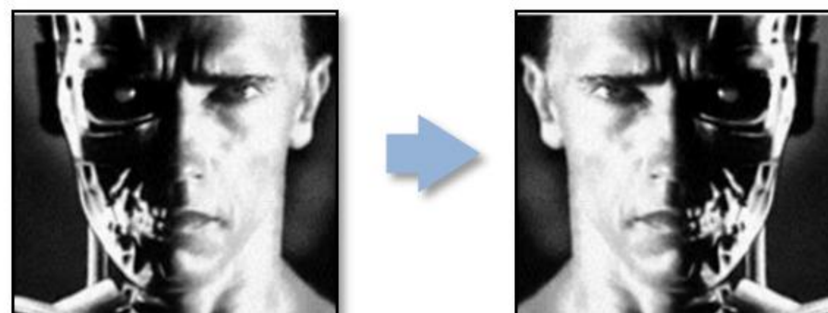
= 45 degrees

Image Transformations

- We can think of a (grayscale) image as a function, *f*, from R^2 to R
  - $f\,(x,y)$ gives the intensity at position (x,y)
- As with any function, we can apply operators to an image



$$g\,(x,y) = f\,(x,y) + 20$$



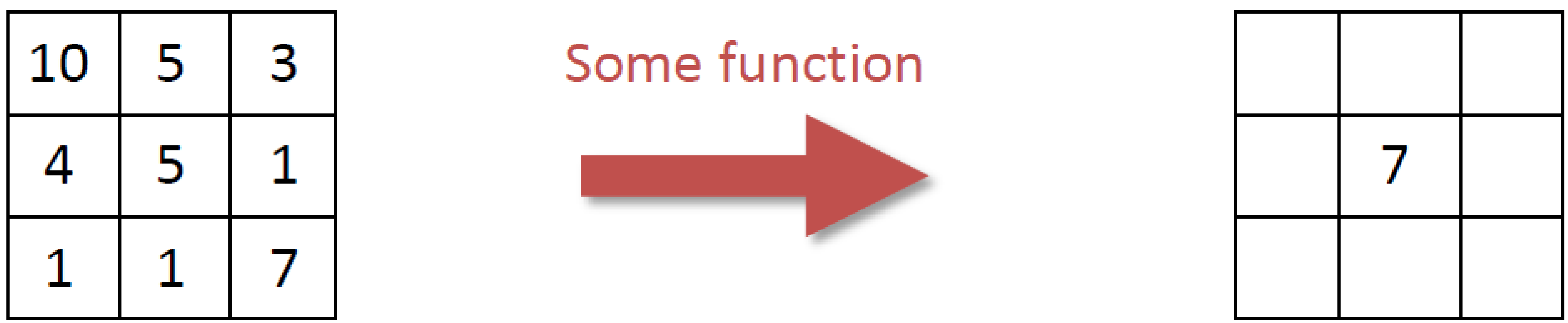
$$g\,(x,y) = f\,(-x,y)$$

Noise Reduction

- Given a camera and a still scene, how can you reduce noise?

Image Filtering

- Modify the pixels in an image based on some function of a local neighborhood of each pixel

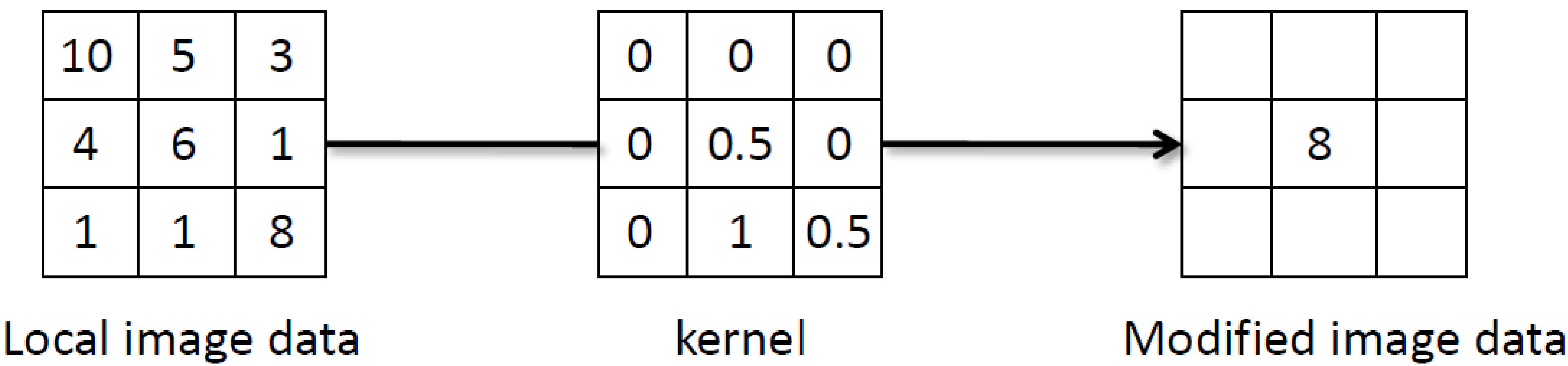


Local image data

Modified image data

Linear Filtering

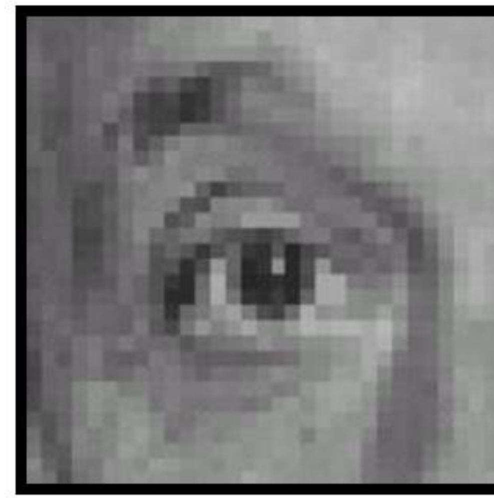
- Replace each pixel by a linear combination of its neighbors
- The prescription for the linear combination is called the "kernel" (or "mask", "filter")



Linear Filter: Example

Original

$$* \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} =$$



Identical image

Linear Filter: Example

Original

$*$

0	0	0
1	0	0
0	0	0

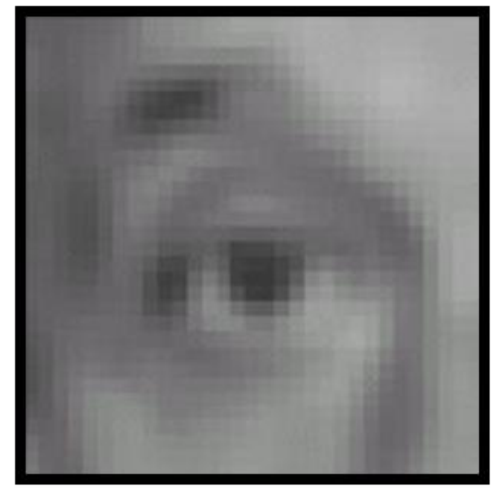
$=$

Shifted left
By 1 pixel

Linear Filter: Example

Original

$$* \ \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} =$$



Blur (with a mean filter)

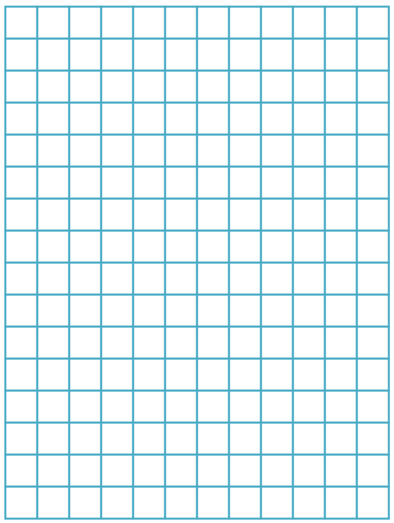
Linear Filter: Example

Original

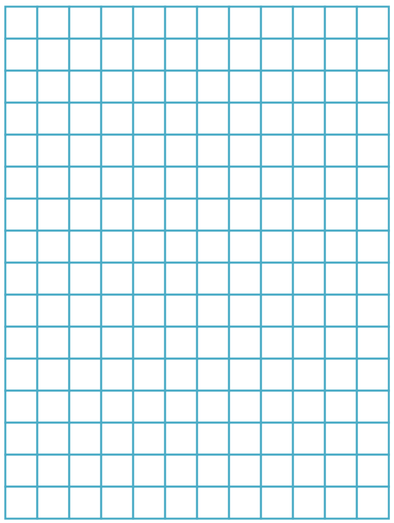
$$* \left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right) =$$

Sharpening filter
(accentuates edges)

Estimating Derivatives: Spatial Filtering

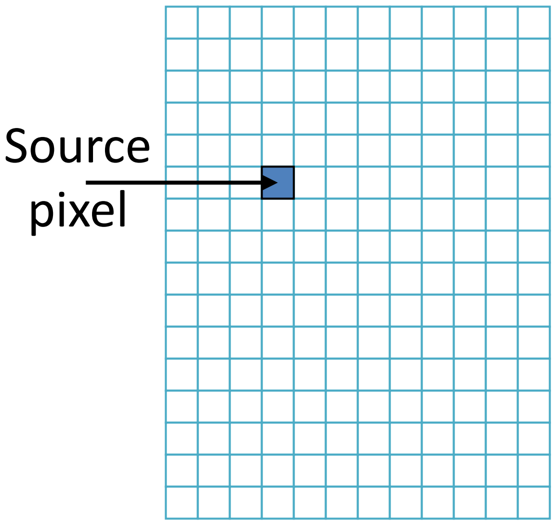


Source image

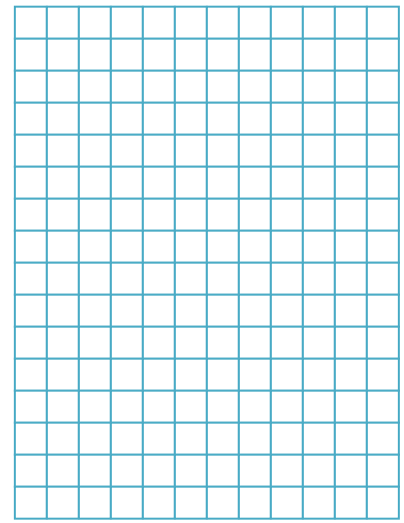


Target image

Estimating Derivatives: Spatial Filtering

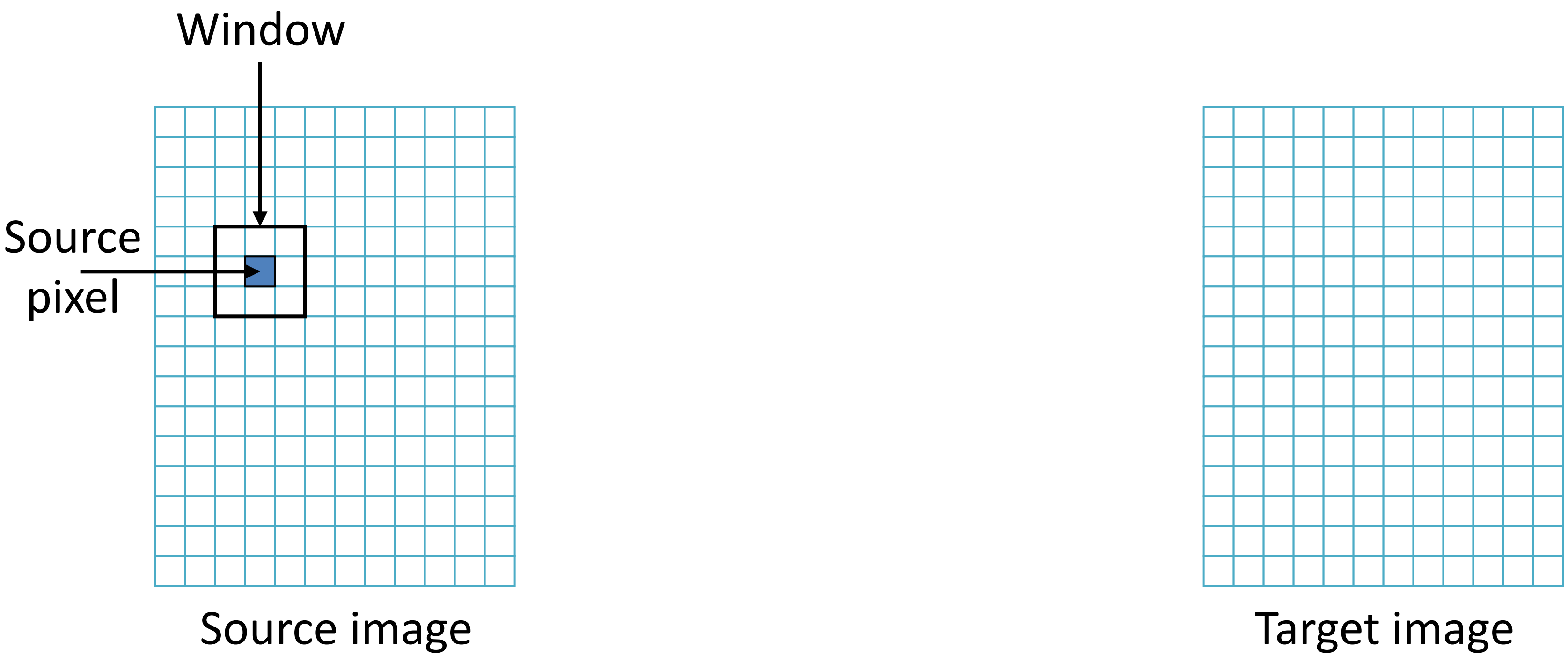


Source image

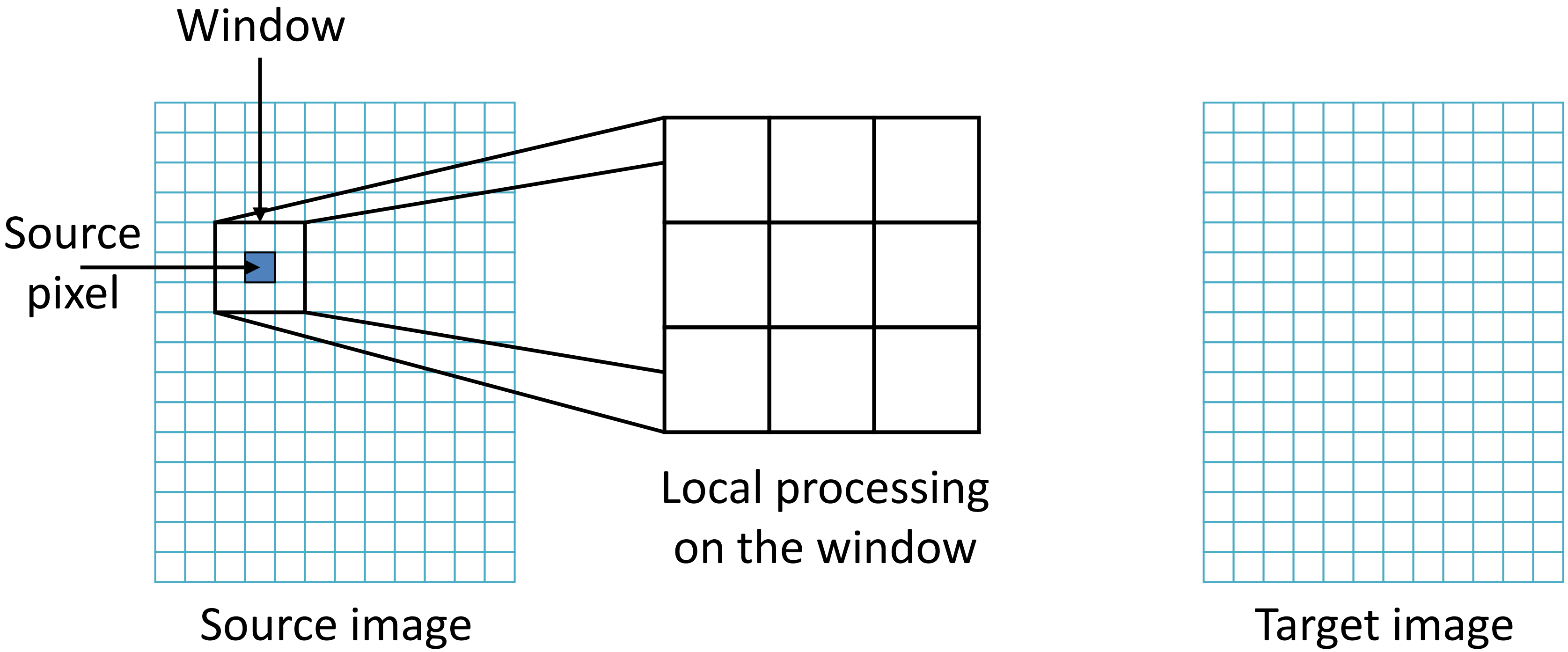


Target image

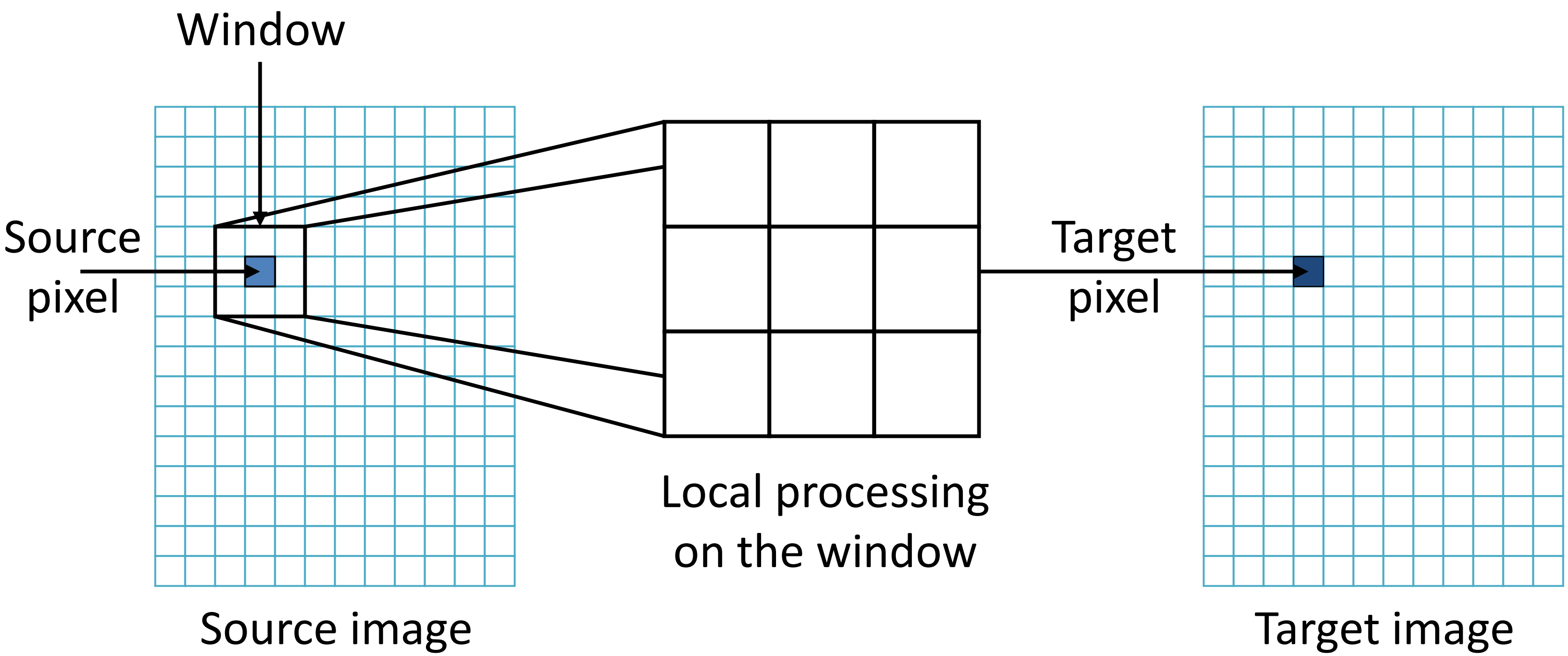
Estimating Derivatives: Spatial Filtering



Estimating Derivatives: Spatial Filtering



Estimating Derivatives: Spatial Filtering



Spatial Filtering: Convolution

- Many filters follow a similar pattern - multiplying each image value by a corresponding filter entry, and summing the results.

$F_{(-1,-1)}$	$F_{(0,-1)}$	$F_{(+1,-1)}$
$F_{(-1,0)}$	$F_{(0,0)}$	$F_{(+1,0)}$
$F_{(-1,+1)}$	$F_{(0,+1)}$	$F_{(+1,+1)}$

Filter Window

$P_{(x-1,y-1)}$	$P_{(x,y-1)}$	$P_{(x+1,y-1)}$
$P_{(x-1,y)}$	$P_{(x,y)}$	$P_{(x+1,y)}$
$P_{(x-1,y+1)}$	$P_{(x,y+1)}$	$P_{(x+1,y+1)}$

Picture Window

Result

Spatial Filtering: Convolution

- Many filters follow a similar pattern - multiplying each image value by a corresponding filter entry, and summing the results.

$F_{(-1,-1)}$	$F_{(0,-1)}$	$F_{(+1,-1)}$
$F_{(-1,0)}$	$F_{(0,0)}$	$F_{(+1,0)}$
$F_{(-1,+1)}$	$F_{(0,+1)}$	$F_{(+1,+1)}$

Filter Window

$P_{(x-1,y-1)}$	$P_{(x,y-1)}$	$P_{(x+1,y-1)}$
$P_{(x-1,y)}$	$P_{(x,y)}$	$P_{(x+1,y)}$
$P_{(x-1,y+1)}$	$P_{(x,y+1)}$	$P_{(x+1,y+1)}$

Picture Window

$$F_{(-1,-1)} \times P_{(x-1,y-1)}$$

Result

Spatial Filtering: Convolution

- Many filters follow a similar pattern - multiplying each image value by a corresponding filter entry, and summing the results

$F_{(-1,-1)}$	$F_{(0,-1)}$	$F_{(+1,-1)}$
$F_{(-1,0)}$	$F_{(0,0)}$	$F_{(+1,0)}$
$F_{(-1,+1)}$	$F_{(0,+1)}$	$F_{(+1,+1)}$

Filter Window

$P_{(x-1,y-1)}$	$P_{(x,y-1)}$	$P_{(x+1,y-1)}$
$P_{(x-1,y)}$	$P_{(x,y)}$	$P_{(x+1,y)}$
$P_{(x-1,y+1)}$	$P_{(x,y+1)}$	$P_{(x+1,y+1)}$

Picture Window

$F_{(-1,-1)} \times P_{(x-1,y-1)}$

$+ F_{(0,-1)} \times P_{(x,y-1)}$

Result

Spatial Filtering: Convolution

- Many filters follow a similar pattern - multiplying each image value by a corresponding filter entry, and summing the results

$F_{(-1,-1)}$	$F_{(0,-1)}$	$F_{(+1,-1)}$
$F_{(-1,0)}$	$F_{(0,0)}$	$F_{(+1,0)}$
$F_{(-1,+1)}$	$F_{(0,+1)}$	$F_{(+1,+1)}$

Filter Window

$P_{(x-1,y-1)}$	$P_{(x,y-1)}$	$P_{(x+1,y-1)}$
$P_{(x-1,y)}$	$P_{(x,y)}$	$P_{(x+1,y)}$
$P_{(x-1,y+1)}$	$P_{(x,y+1)}$	$P_{(x+1,y+1)}$

Picture Window

$$F_{(-1,-1)} \times P_{(x-1,y-1)}$$
$$+ F_{(0,-1)} \times P_{(x,y-1)}$$
$$+ F_{(+1,-1)} \times P_{(x+1,y-1)}$$

Result

Spatial Filtering: Convolution

- Many filters follow a similar pattern - multiplying each image value by a corresponding filter entry, and summing the results

$F_{(-1,-1)}$	$F_{(0,-1)}$	$F_{(+1,-1)}$
$F_{(-1,0)}$	$F_{(0,0)}$	$F_{(+1,0)}$
$F_{(-1,+1)}$	$F_{(0,+1)}$	$F_{(+1,+1)}$

Filter Window

$P_{(x-1,y-1)}$	$P_{(x,y-1)}$	$P_{(x+1,y-1)}$
$P_{(x-1,y)}$	$P_{(x,y)}$	$P_{(x+1,y)}$
$P_{(x-1,y+1)}$	$P_{(x,y+1)}$	$P_{(x+1,y+1)}$

Picture Window

$$F_{(-1,-1)}\times P_{(x-1,y-1)}$$
$$+ F_{(0,-1)}\times P_{(x,y-1)}$$
$$+ F_{(+1,-1)}\times P_{(x+1,y-1)}$$
$$+ F_{(-1,0)}\times P_{(x-1,y)}$$

Result

Spatial Filtering: Convolution

- Many filters follow a similar pattern - multiplying each image value by a corresponding filter entry, and summing the results

$F_{(-1,-1)}$	$F_{(0,-1)}$	$F_{(+1,-1)}$
$F_{(-1,0)}$	$F_{(0,0)}$	$F_{(+1,0)}$
$F_{(-1,+1)}$	$F_{(0,+1)}$	$F_{(+1,+1)}$

Filter Window

$P_{(x-1,y-1)}$	$P_{(x,y-1)}$	$P_{(x+1,y-1)}$
$P_{(x-1,y)}$	$P_{(x,y)}$	$P_{(x+1,y)}$
$P_{(x-1,y+1)}$	$P_{(x,y+1)}$	$P_{(x+1,y+1)}$

Picture Window

$$F_{(-1,-1)} \times P_{(x-1,y-1)}$$
$$+ F_{(0,-1)} \times P_{(x,y-1)}$$
$$+ F_{(+1,-1)} \times P_{(x+1,y-1)}$$
$$+ F_{(-1,0)} \times P_{(x-1,y)}$$
$$+ \ldots$$

Result

Spatial Filtering: Convolution

- Many filters follow a similar pattern - multiplying each image value by a corresponding filter entry, and summing the results

$F_{(-1,-1)}$	$F_{(0,-1)}$	$F_{(+1,-1)}$
$F_{(-1,0)}$	$F_{(0,0)}$	$F_{(+1,0)}$
$F_{(-1,+1)}$	$F_{(0,+1)}$	$F_{(+1,+1)}$

Filter Window

$P_{(x-1,y-1)}$	$P_{(x,y-1)}$	$P_{(x+1,y-1)}$
$P_{(x-1,y)}$	$P_{(x,y)}$	$P_{(x+1,y)}$
$P_{(x-1,y+1)}$	$P_{(x,y+1)}$	$P_{(x+1,y+1)}$

Picture Window

$$
\begin{aligned}
&F_{(-1,-1)} \times P_{(x-1,y-1)} \\
&+ F_{(0,-1)} \times P_{(x,y-1)} \\
&+ F_{(+1,-1)} \times P_{(x+1,y-1)} \\
&+ F_{(-1,0)} \times P_{(x-1,y)} \\
&+ \ldots \\
&+ F_{(+1,+1)} \times P_{(x+1,y+1)}
\end{aligned}
$$

Result

Derivative Filters

- Sobel Operators

G_x

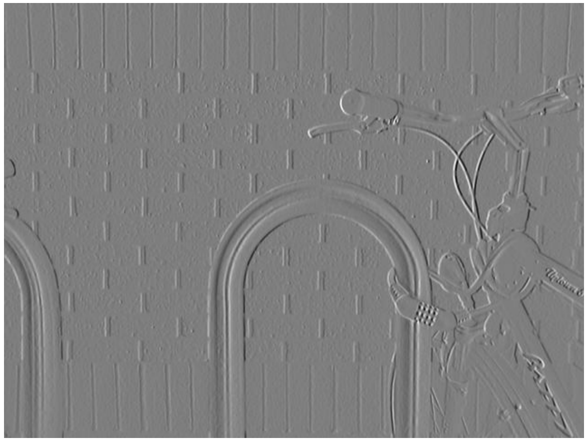
-1	0	1
-2	0	2
-1	0	1

G_y

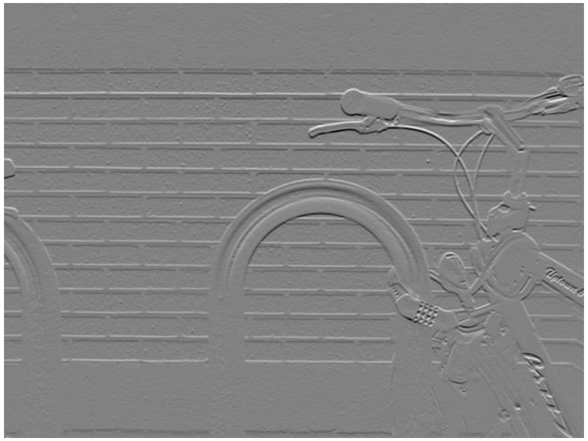
-1	-2	-1
0	0	0
1	2	1

- Applied separately and results combined to estimate overall gradient magnitude.

Derivative Filters



G_x



G_y

Oriented derivative filters only respond to edges in one direction.

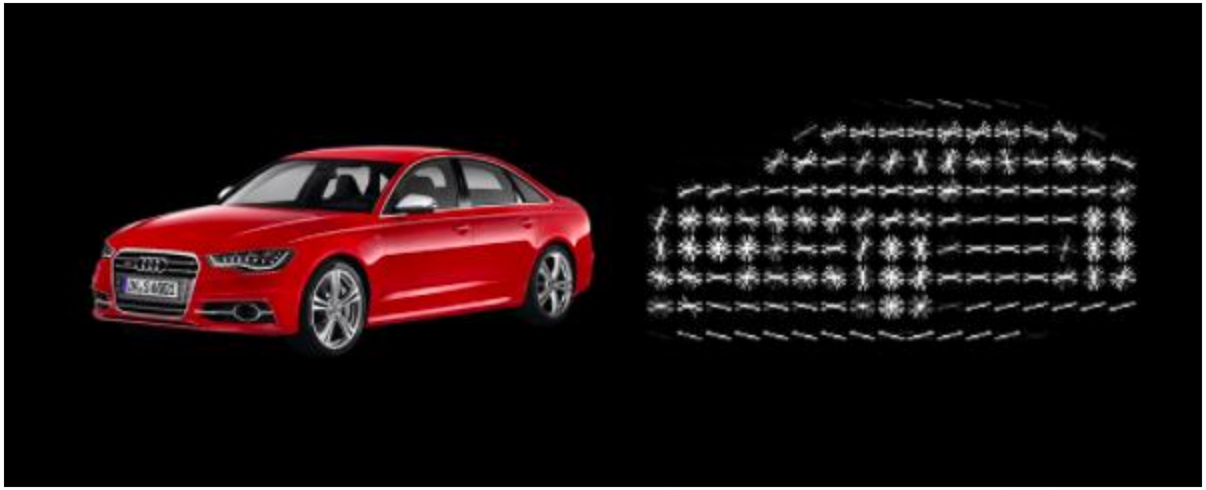
Gradient Magnitude

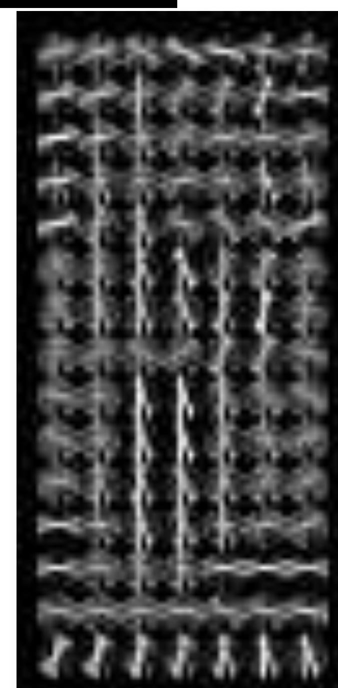

A few simple image processing operations provide image gradient and gradient direction at each pixel.

Histogram of Oriented Gradients (HoG)

- First used for person detection (Dalal and Triggs, CVPR 2005) cited in thousands (~47243) of computer vision papers
- Objective: human (object) recognition
- Basic idea:
  - Local shape information often well described by the distribution of intensity gradients or edge directions
  - Convert the image (width*height*channels) into a feature vector, then apply the classification algorithms
  - The intent is to generalize the object in such a way that the same object (e.g., person) produces as close as possible to the same feature descriptor when viewed under different conditions

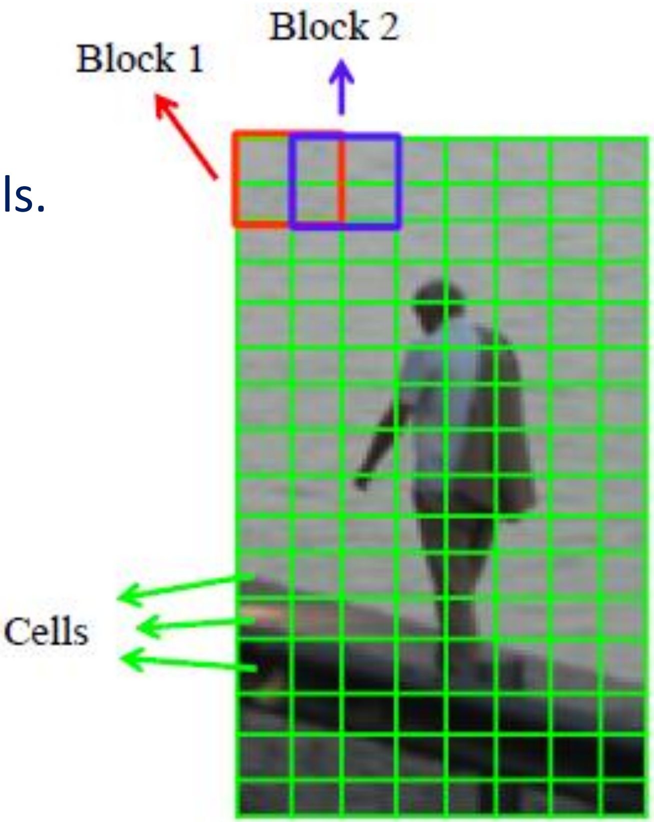
HOG Feature Descriptor





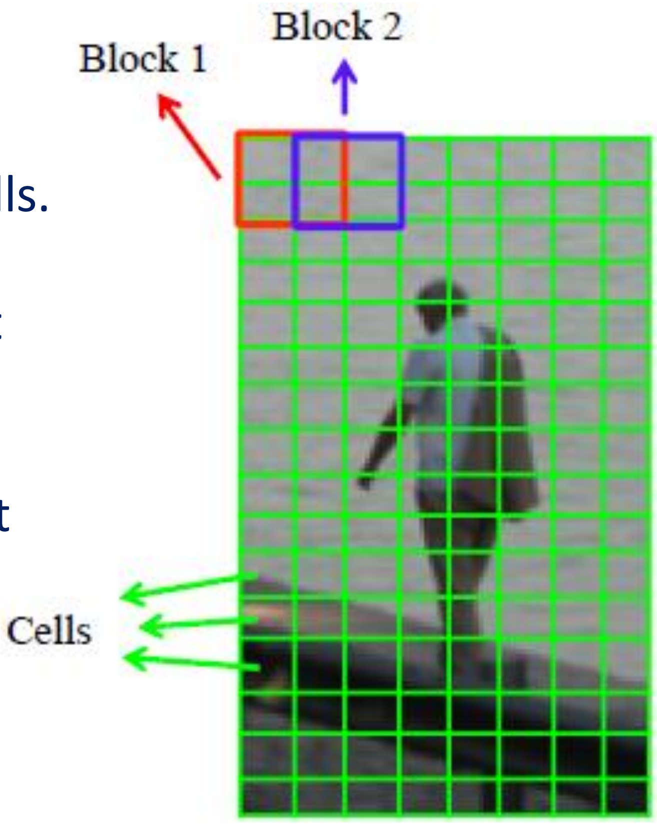
Histogram of Oriented Gradients (HoG)

- Divide the patch into small **cells**.
- Define slightly larger **blocks**, covering several cells.



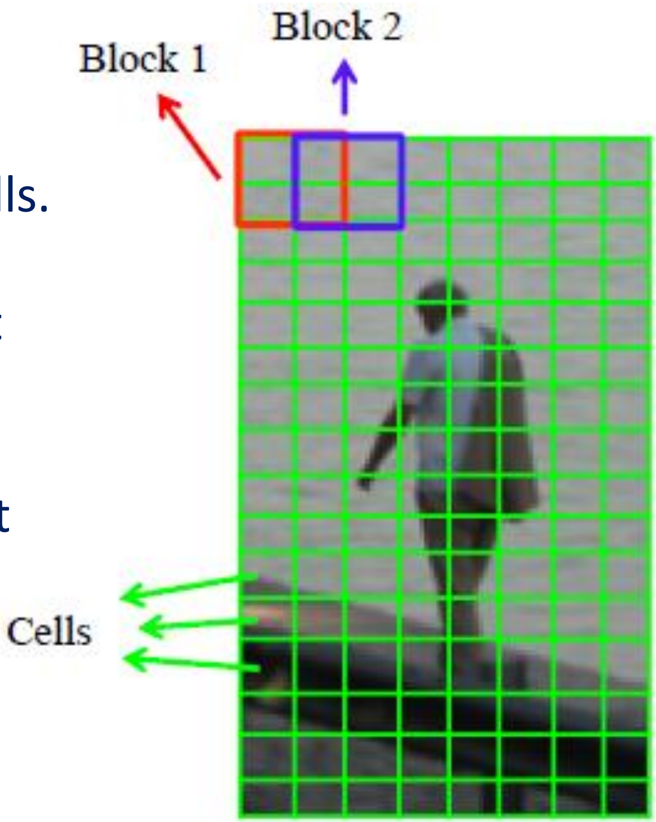
Histogram of Oriented Gradients (HoG)

- Divide the patch into small **cells**.
- Define slightly larger **blocks**, covering several cells.
- Compute gradient magnitude and orientation at each **pixel**.
- Compute a local **weighted histogram** of gradient orientations for each cell, weighting by some function of magnitude.



Histogram of Oriented Gradients (HoG)

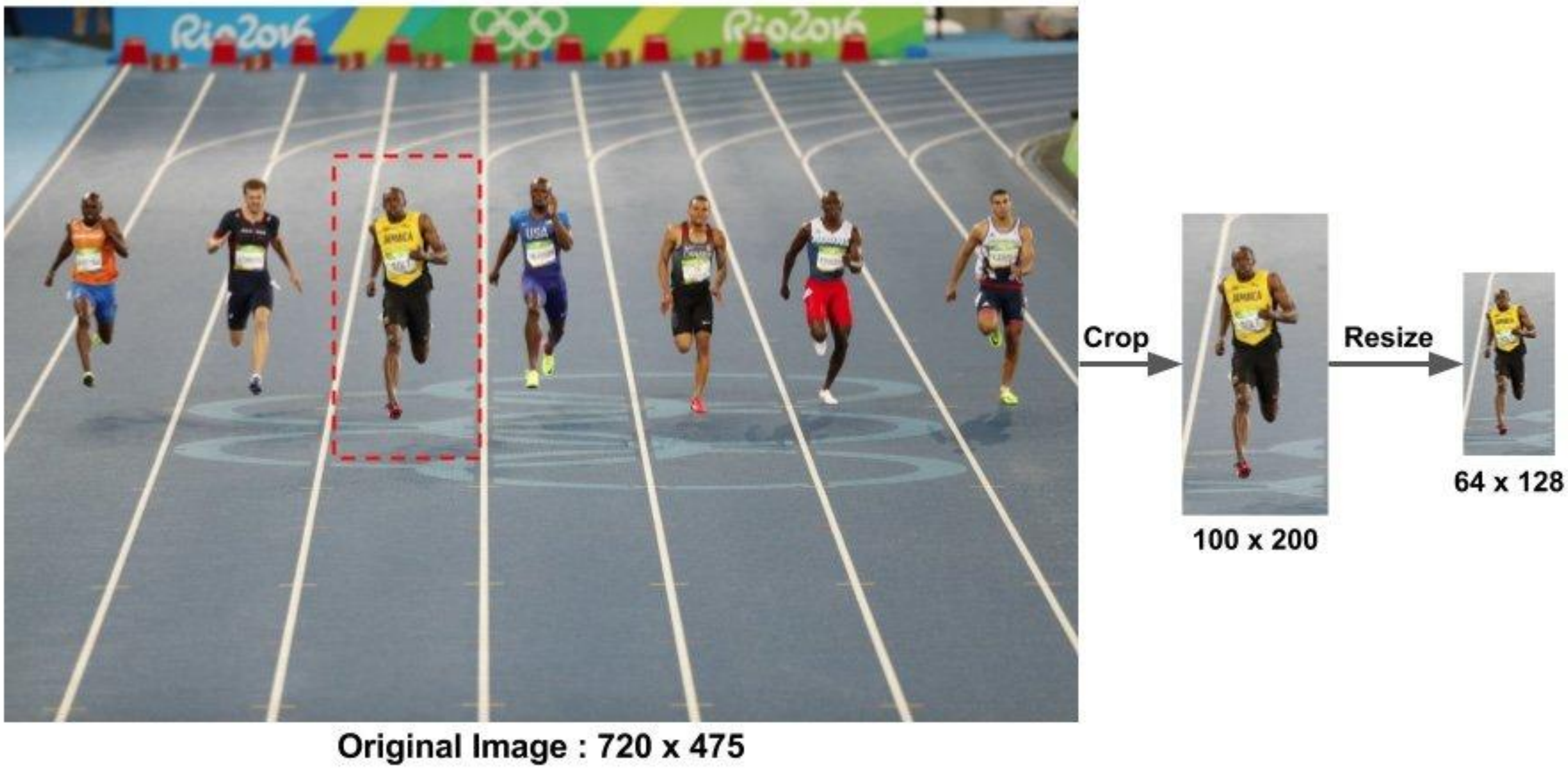
- Divide the patch into small **cells**.
- Define slightly larger **blocks**, covering several cells.
- Compute gradient magnitude and orientation at each **pixel**.
- Compute a local **weighted histogram** of gradient orientations for each cell, weighting by some function of magnitude.
- Concatenate histogram entries to form a HoG vector for each block.



- Normalize vector values by dividing by some function of vector length.
  - For improved accuracy, the local histograms can be contrast-normalized by calculating a measure of the intensity across a larger region of the image, called a block, and then using this value to normalize all cells within the block.

HoG: Example

- Preprocessing



HoG: Example

- Calculating the Gradients

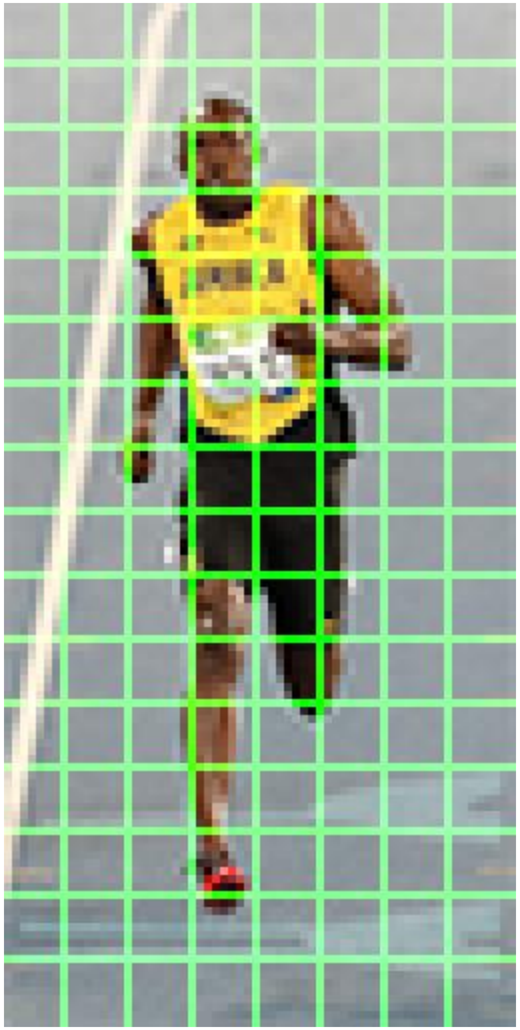
G_x

G_y

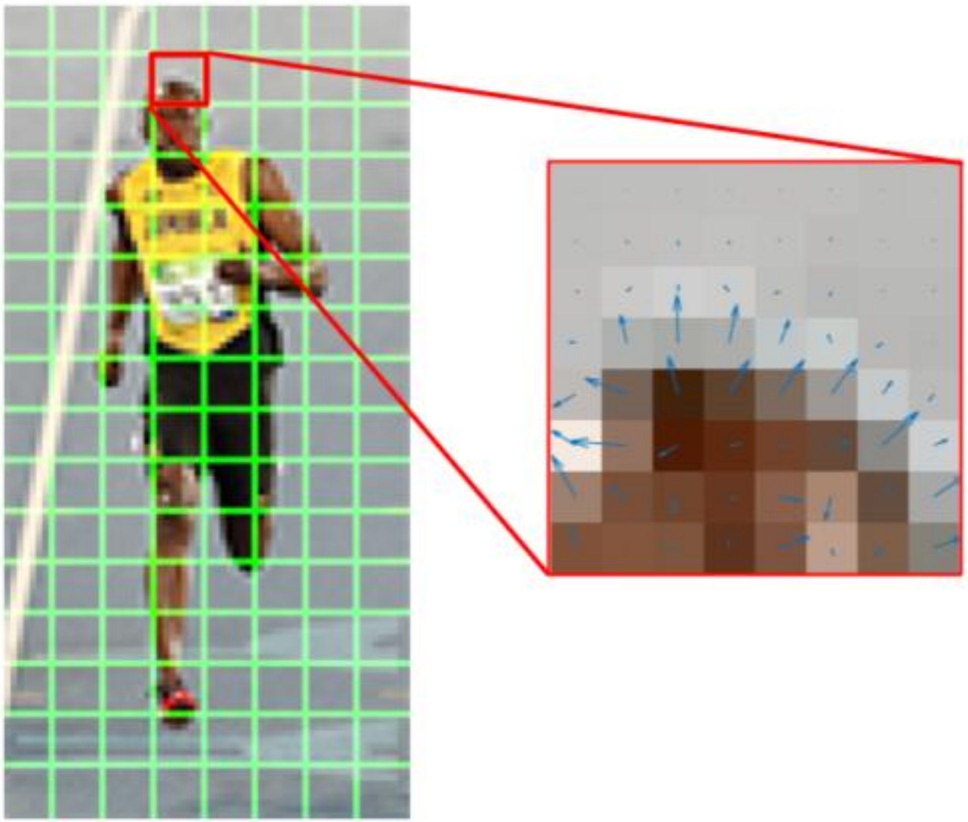
Magnitude

HoG: Example

- Calculate Histogram of Gradients in cells
- Human detector (8*8 to capture interesting features)
- The 8*8 cell can be represented by 128 numbers



HoG: Example



2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

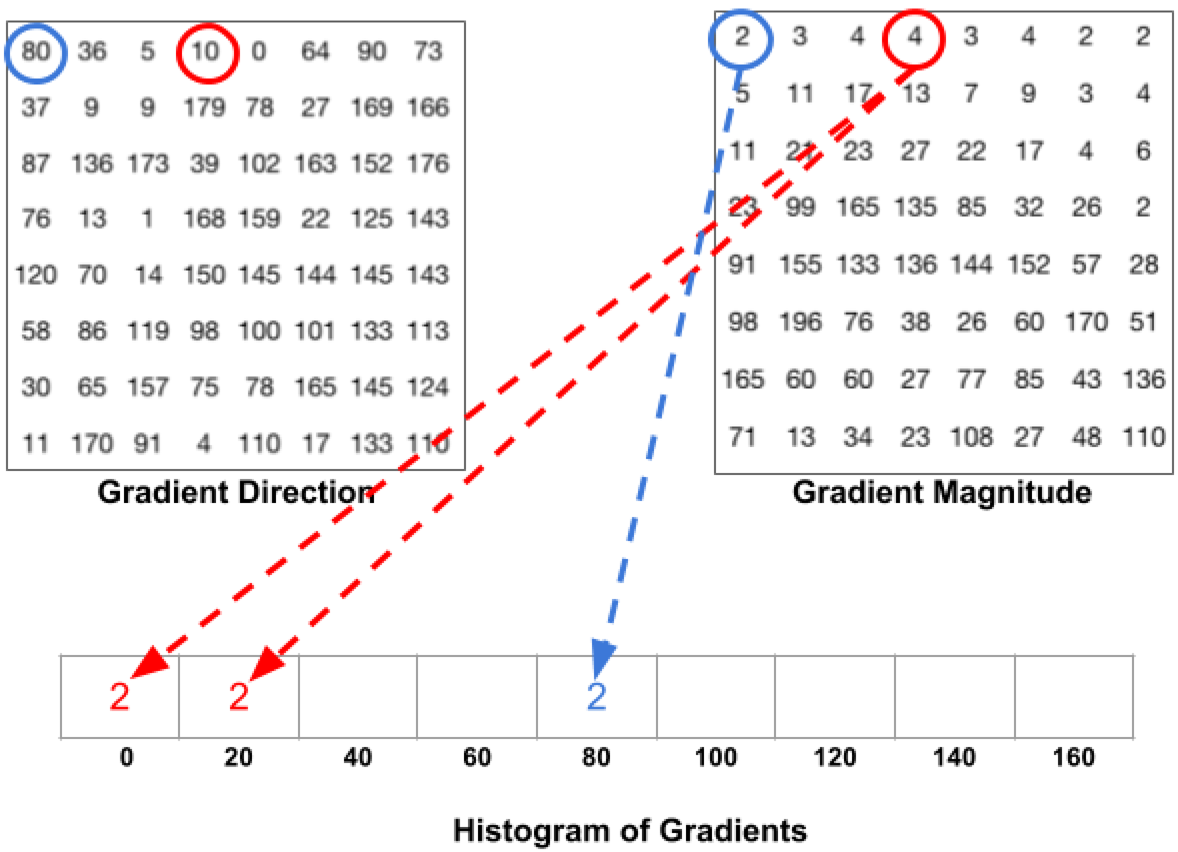
Gradient Magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

HoG: Example

- The histogram is a vector of 9 bins corresponding to angles 0, 20, 40, 60 … 160
- A bin is selected based on the direction, and the vote (the value that goes into the bin) is selected based on the magnitude



HoG: Example

- If the angle is greater than 160 degrees, split the contribution to 0 degree bin and 160 degree bin
- E.g, 165 = 160*0.75 + 180*0.25, apply the same for the magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

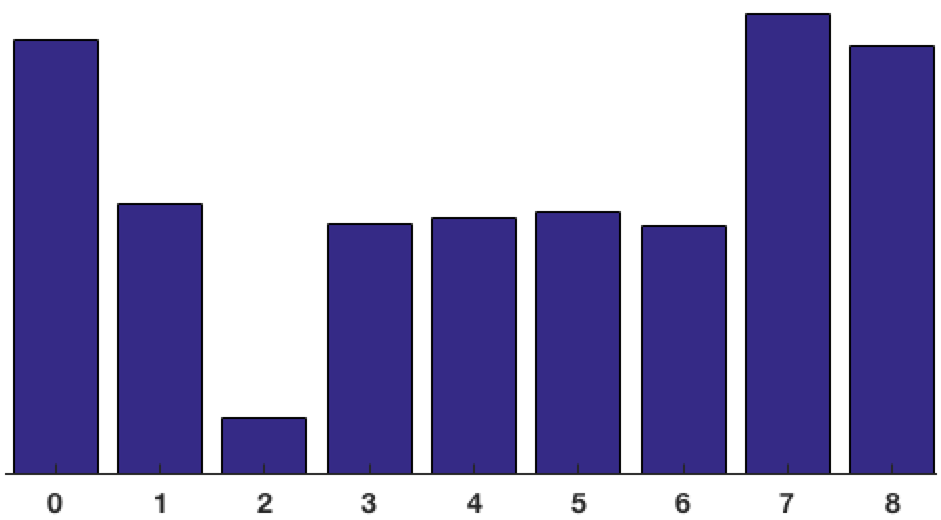
Gradient Magnitude

0	20	40	60	80	100	120	140	160
21.25								63.75

Histogram of Gradients

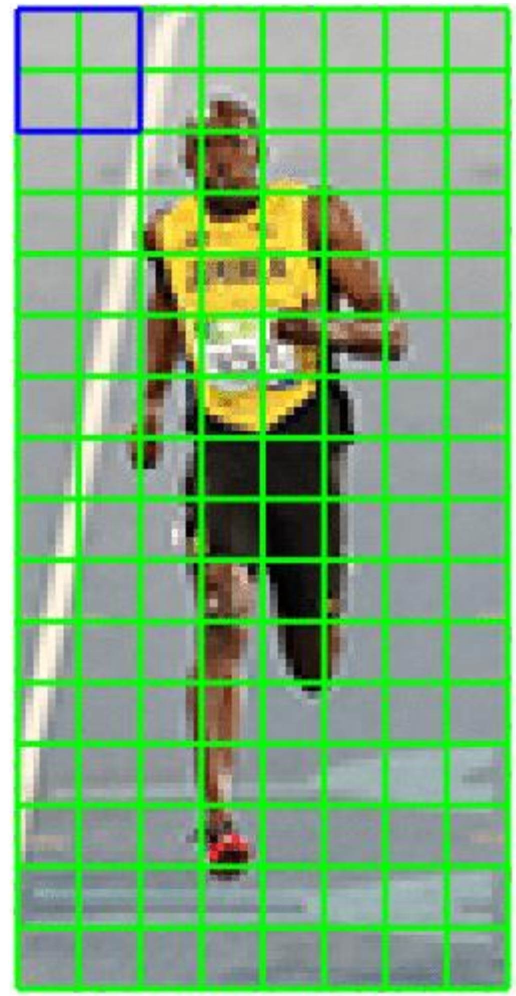
HoG: Example

- The contributions of all the pixels in the 8*8 cells are added up to create the 9-bin histogram



HoG: Example

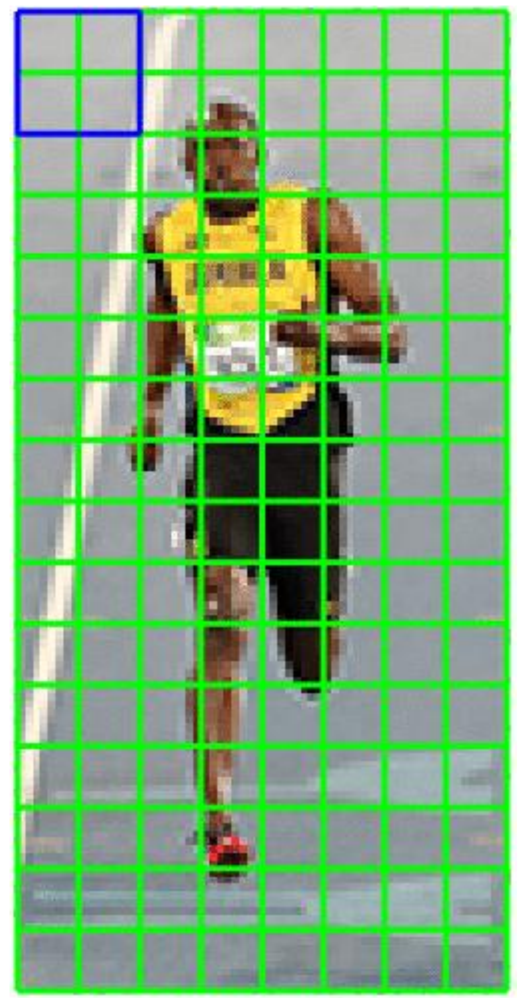
- Gradients of an image are sensitive to overall lighting
- We would like to normalize the histogram so they are not affected by lighting variations
- A 16*16 block has 4 histograms which can be concatenated to form a 36*1 element vector
- Divide each element of a vector by its vector length (l2-norm)



HoG: Example

- How many positions of the 16*16 block do we have for this 64*128 image?

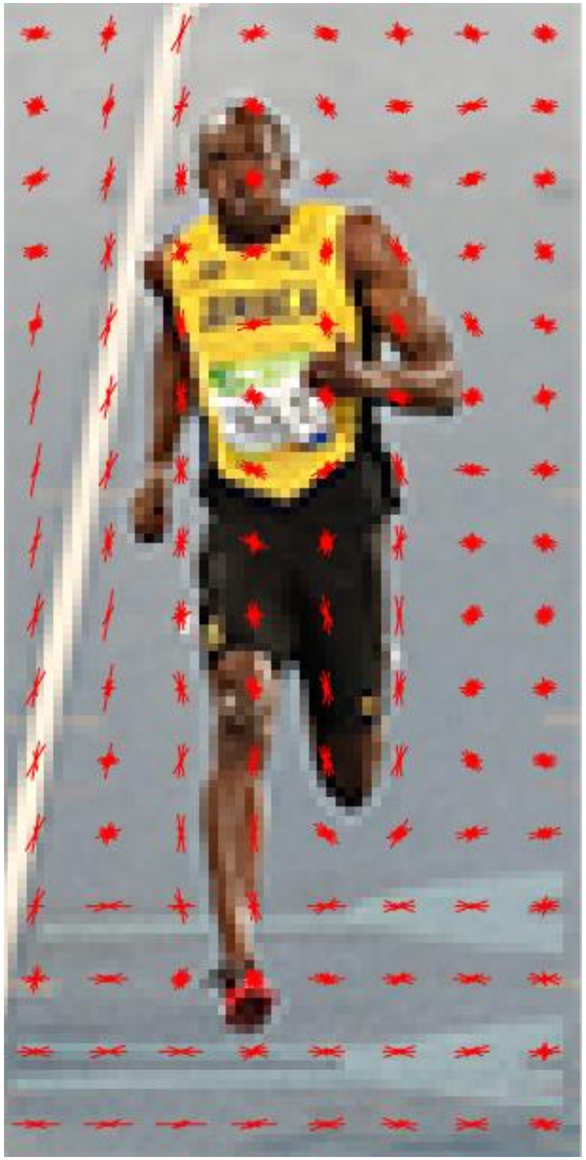
- What is the dimension of the HoG vector for this 64*128 image?



HoG: Example

- The HOG descriptor of an image patch is usually visualized by plotting the 9*1 normalized histograms in the 8*8 cells

- Figure right: dominant direction of the histogram captures the shape of the person



Variations on a Theme

Alternative derivative filters are available.

-1	0	1

centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected

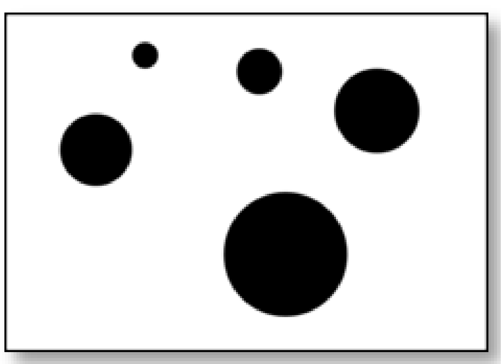
0	1
-1	0

diagonal

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

Sobel

Different cell and block sizes



Variations on a Theme

Alternative derivative filters are available.

-1	0	1

centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected

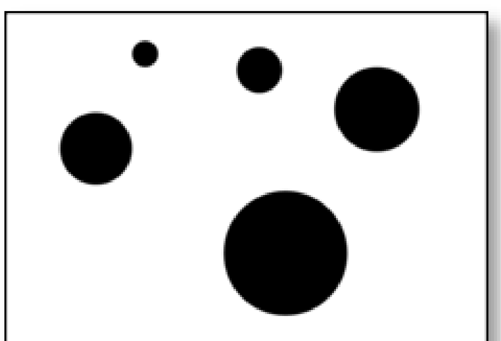
0	1
-1	0

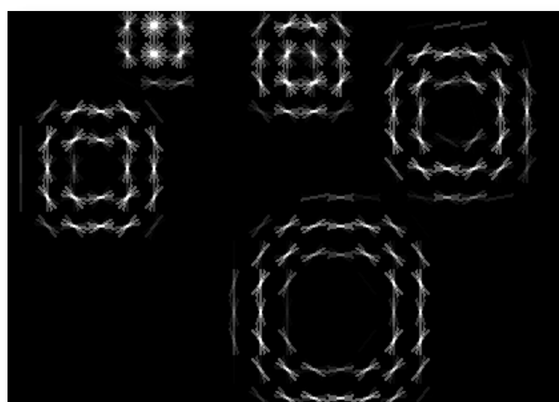
diagonal

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

Sobel

Different cell and block sizes





10x10 cells

Variations on a Theme

Alternative derivative filters are available.

-1	0	1

centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected

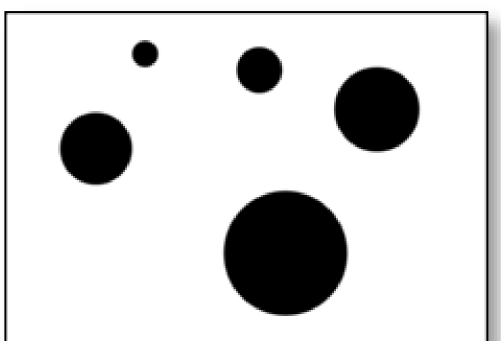
0	1
-1	0

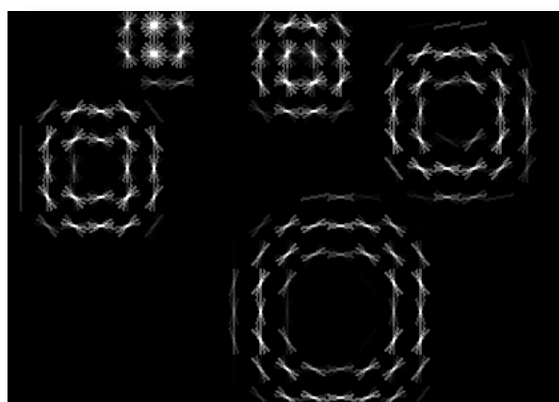
diagonal

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

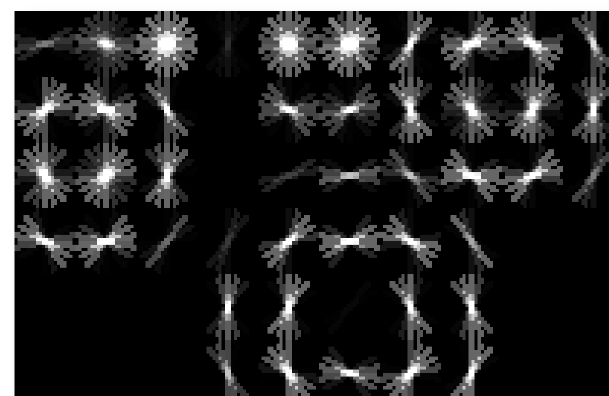
Sobel

Different cell and block sizes





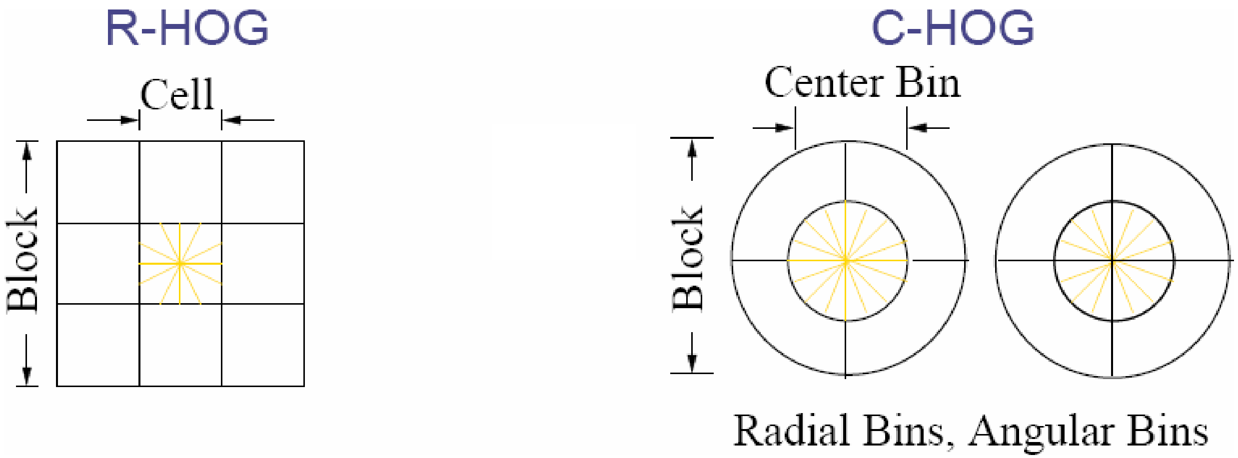
10x10 cells



20x20 cells

Variations on a Theme

Different block geometries (Rectangular or Circular)



Different weighting functions e.g. magnitude^2.

Different normalization functions.

Person Identification

- Dalal and Triggs used HoGs very successfully to detect pedestrians in natural images: More later.

Conclusion

- Rectangular image patches are often used.
- To examine and compare them we need to produce descriptions of them: feature vectors.
- Some common feature vectors.
  - Colour histograms
  - Local binary patterns
  - Histograms of Gradient Orientations (HoG)