Deep Learning for Computer Vision

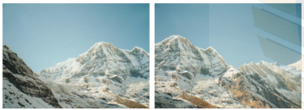
# Scale Space, Image Pyramids and Filter Banks


Vineeth N Balasubramanian

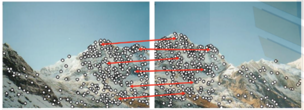
Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

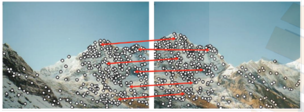
# Review

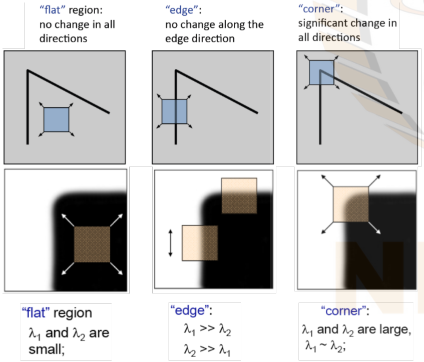


# Review

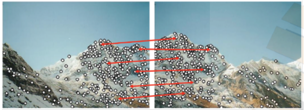


# Review




"flat" region:
no change in all
directions
"edge":
no change along the
edge direction
"corner":
significant change in
all directions
"flat" region
$\lambda_1$ and $\lambda_2$ are
small;
"edge":
$\lambda_1 >> \lambda_2$
$\lambda_2 >> \lambda_1$
"corner":
$\lambda_1$ and $\lambda_2$ are large,
$\lambda_1 \sim \lambda_2$;

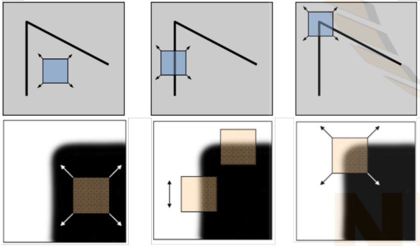
# Review



"flat" region:
no change in all
directions

"edge":
no change along the
edge direction

"corner":
significant change in
all directions

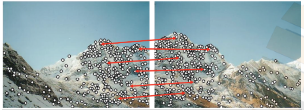


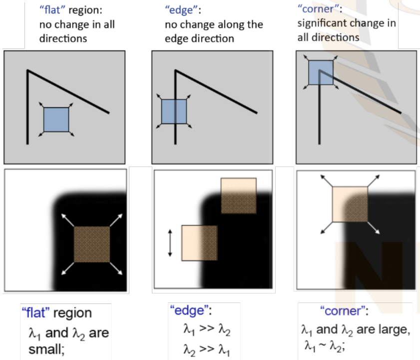
"flat" region
$\lambda_1$ and $\lambda_2$ are
small;

"edge":
$\lambda_1 >> \lambda_2$
$\lambda_2 >> \lambda_1$

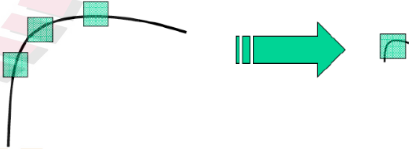
"corner":
$\lambda_1$ and $\lambda_2$ are large,
$\lambda_1 \sim \lambda_2$;

# Review

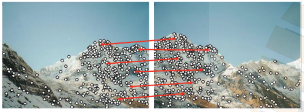




"flat" region:
no change in all directions
"edge":
no change along the edge direction
"corner":
significant change in all directions
"flat" region
$\lambda_1$ and $\lambda_2$ are small;
"edge":
$\lambda_1 >> \lambda_2$
$\lambda_2 >> \lambda_1$
"corner":
$\lambda_1$ and $\lambda_2$ are large,
$\lambda_1 \sim \lambda_2$;

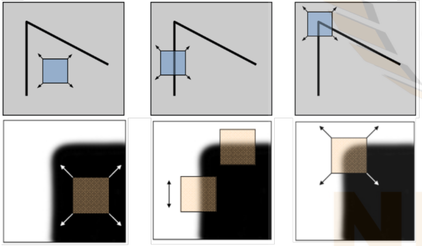
# Review



"flat" region:
no change in all
directions

"edge":
no change along the
edge direction

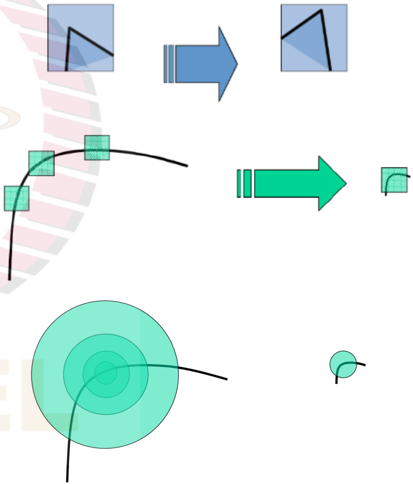
"corner":
significant change in
all directions



"flat" region
$\lambda_1$ and $\lambda_2$ are small;

"edge":
$\lambda_1 >> \lambda_2$
$\lambda_2 >> \lambda_1$

"corner":
$\lambda_1$ and $\lambda_2$ are large,
$\lambda_1 \sim \lambda_2$;



# Scale-Invariant Interest Point Detection

- How can we independently select interest points in each image, such that detections are repeatable across different scales?

# Scale-Invariant Interest Point Detection

- How can we independently select interest points in each image, such that detections are repeatable across different scales?
  - Extract features at a variety of scales, e.g., by using multiple resolutions in a pyramid, and then matching features at the same level.

# Scale-Invariant Interest Point Detection

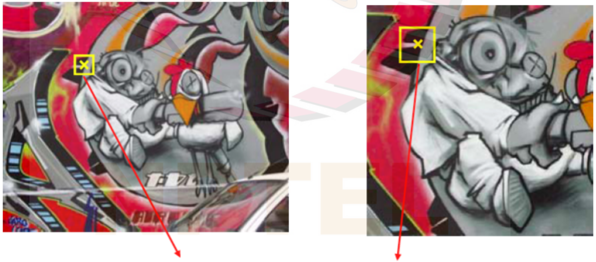
- How can we independently select interest points in each image, such that detections are repeatable across different scales?
  - Extract features at a variety of scales, e.g., by using multiple resolutions in a pyramid, and then matching features at the same level.
  - When does this work?

# Scale-Invariant Interest Point Detection

- How can we independently select interest points in each image, such that detections are repeatable across different scales?
  - Extract features at a variety of scales, e.g., by using multiple resolutions in a pyramid, and then matching features at the same level.
  - When does this work?
  - More efficient to extract features stable in both location and scale.

# Scale-Invariant Interest Point Detection

- How can we independently select interest points in each image, such that detections are repeatable across different scales?
  - Extract features at a variety of scales, e.g., by using multiple resolutions in a pyramid, and then matching features at the same level.
  - When does this work?
  - More efficient to extract features stable in both location and scale.
  - Find scale that gives local maxima of a function $f$ in both position and scale.



$$f(I_{i_1 \dots i_m}(x,\sigma)) = f(I_{i_1 \dots i_m}(x',\sigma'))$$

# Automatic Scale Selection

Function responses for increasing scale (scale signature).


*Credit: R Urtasun*

# Automatic Scale Selection

Function responses for increasing scale (scale signature).


*Credit: R Urtasun*

# Automatic Scale Selection

Function responses for increasing scale (scale signature).


*Credit: R Urtasun*

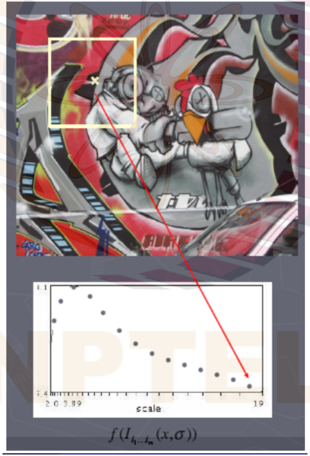
# Automatic Scale Selection

Function responses for increasing scale (scale signature).


*Credit: R Urtasun*

# Automatic Scale Selection

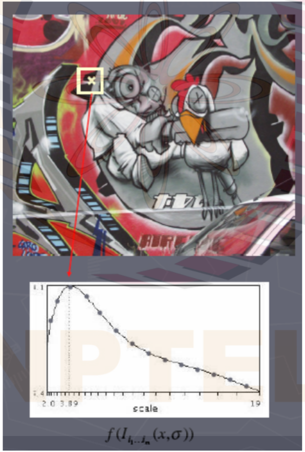
Function responses for increasing scale (scale signature).




*Credit: R Urtasun*

# Automatic Scale Selection

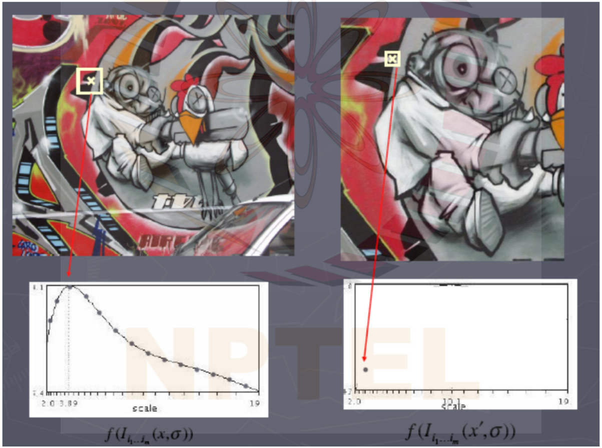
Function responses for increasing scale (scale signature).




*Credit: R Urtasun*

# Automatic Scale Selection

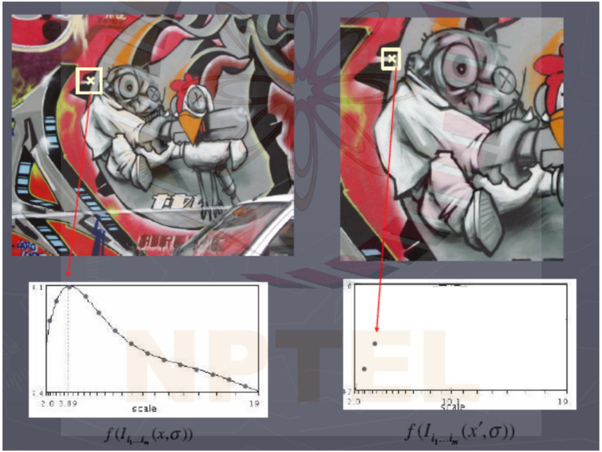
Function responses for increasing scale (scale signature).




*Credit: R Urtasun*

# Automatic Scale Selection

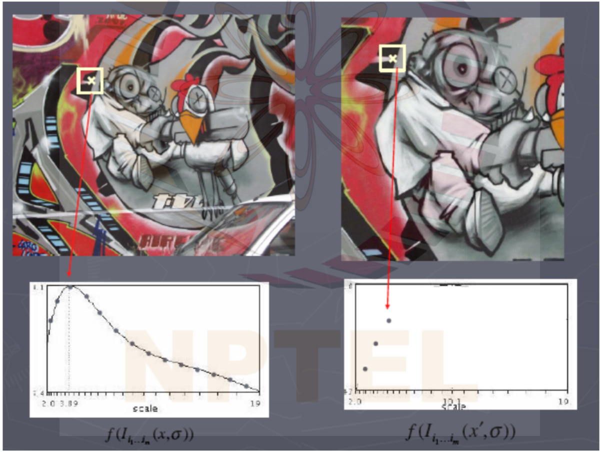
Function responses for increasing scale (scale signature).




*Credit: R Urtasun*

# Automatic Scale Selection

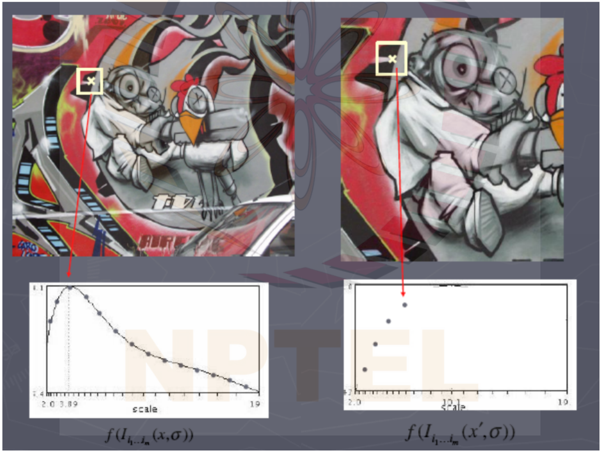
Function responses for increasing scale (scale signature).




*Credit: R Urtasun*

# Automatic Scale Selection

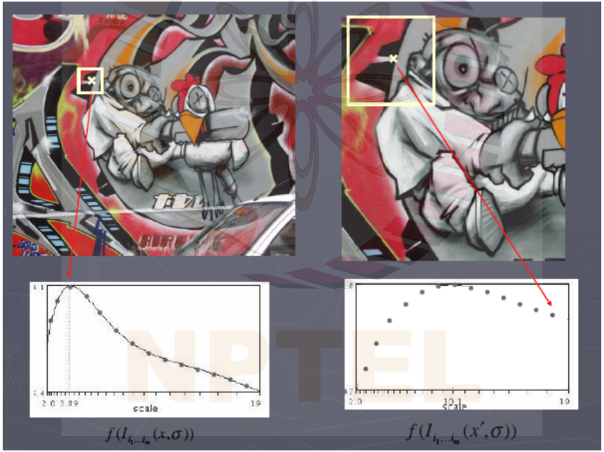
Function responses for increasing scale (scale signature).




*Credit: R Urtasun*

# Automatic Scale Selection

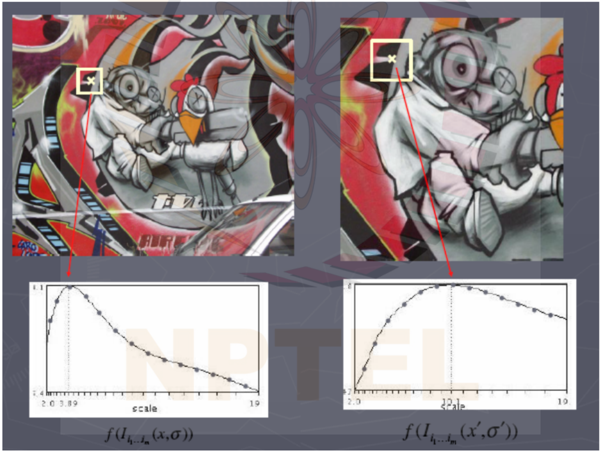
Function responses for increasing scale (scale signature).




*Credit: R Urtasun*

# Automatic Scale Selection

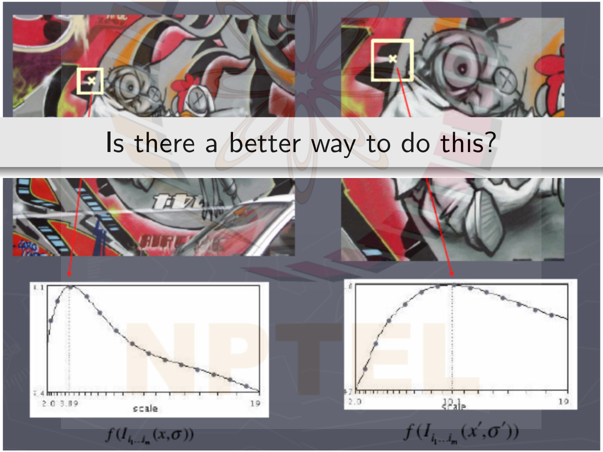
Function responses for increasing scale (scale signature).




*Credit: R Urtasun*

# Automatic Scale Selection

Function responses for increasing scale (scale signature).




Is there a better way to do this?

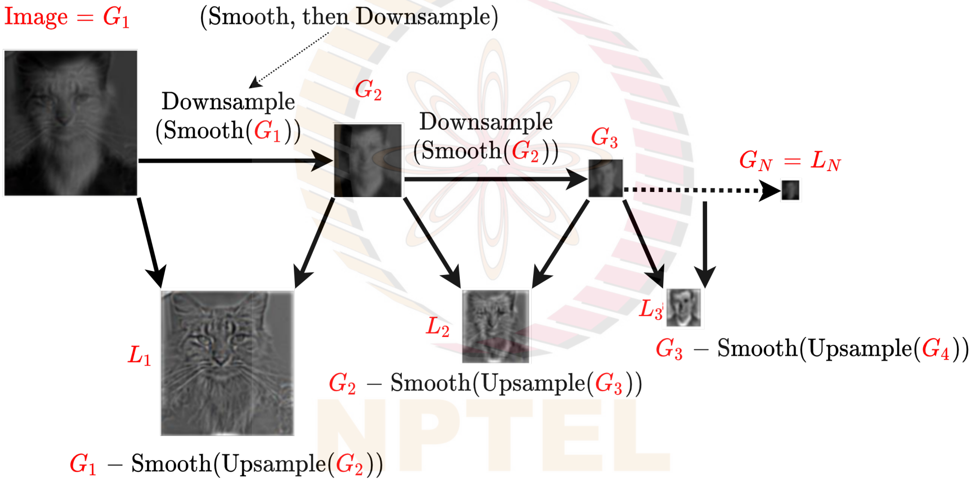
# Automatic Scale Selection: Implementation

Instead of computing $f$ for larger and larger windows, we can implement using a fixed window size with a Gaussian pyramid.

Sometimes need to create in-between levels, e.g., a $\frac{3}{4}$ size image.

*Credit: R Urtasun*

# Gaussian and Laplacian Pyramid




*Credit: Derek Hoiem*

# Image Pyramids: Uses

- Compression

# Image Pyramids: Uses

- Compression
- Object detection

# Image Pyramids: Uses

- Compression
- Object detection
  - Scale search

# Image Pyramids: Uses

- Compression
- Object detection
    - Scale search
    - Features

# Image Pyramids: Uses

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points

# Image Pyramids: Uses

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Registration

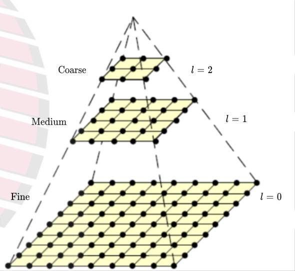
# Image Pyramids: Uses

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Registration
  - Coarse-to-fine Image Registration

# Image Pyramids: Uses

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Registration
  - Coarse-to-fine Image Registration

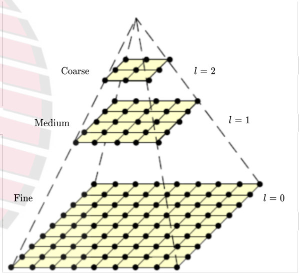
Coarse-to-fine Image Registration:



# Image Pyramids: Uses

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Registration
  - Coarse-to-fine Image Registration

Coarse-to-fine Image Registration:

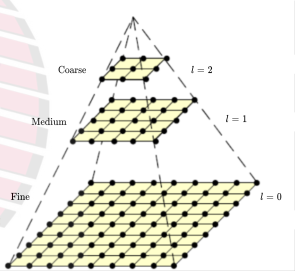



- Compute Gaussian pyramid.

# Image Pyramids: Uses

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Registration
  - Coarse-to-fine Image Registration

Coarse-to-fine Image Registration:

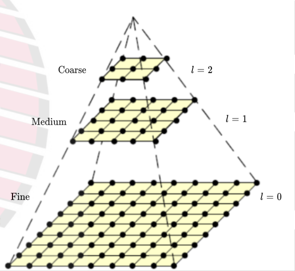



- Compute Gaussian pyramid.
- Align with coarse pyramid.

# Image Pyramids: Uses

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Registration
  - Coarse-to-fine Image Registration

Coarse-to-fine Image Registration:

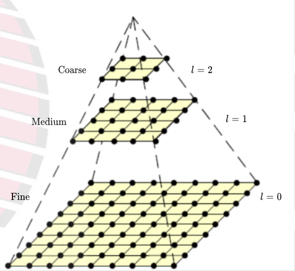



- Compute Gaussian pyramid.
- Align with coarse pyramid.
- Successively align with finer pyramids.

# Image Pyramids: Uses

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Registration
  - Coarse-to-fine Image Registration

Coarse-to-fine Image Registration:




- Compute Gaussian pyramid.
- Align with coarse pyramid.
- Successively align with finer pyramids.
  - Search smaller range.

*Credit: Derek Hoiem*

# Texture in Images

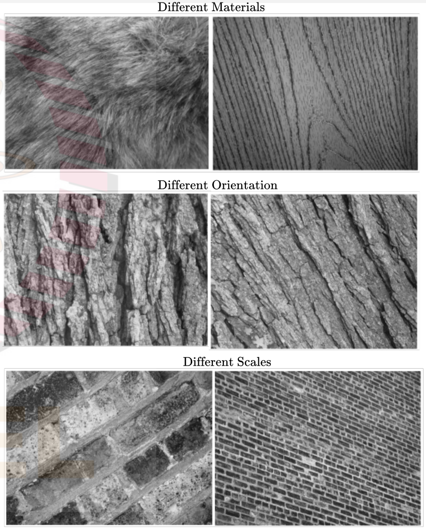
**Textures:**

- Regular or stochastic patterns caused by bumps, grooves and/or markings.

# Texture in Images

**Textures:**

- Regular or stochastic patterns caused by bumps, grooves and/or markings.
- Gives us information about spatial arrangement of colors or intensities in an image.




*Credit: Derek Hoiem*

# Texture in Images

Conveys more information that can be exploited to match regions of interest in images.

Histogram conveys 50% white pixels and 50% black pixels

(Block Pattern) (Checkerboard Pattern) (Striped Pattern)

Drastically different textures

*Credit: Linda G Shapiro*

# Texture in Images

Conveys more information that can be exploited to match regions of interest in images.

Histogram conveys 50% white pixels and 50% black pixels

(Block Pattern) (Checkerboard Pattern) (Striped Pattern)

Drastically different textures

How to represent textures?

# Texture in Images

Conveys more information that can be exploited to match regions of interest in images.

Histogram conveys 50% white pixels and 50% black pixels

**How to represent textures?**

- Compute responses of blobs and edges at various orientations and scales.

(Block Pattern) (Checkerboard Pattern) (Striped Pattern)

Drastically different textures

# Texture in Images

Conveys more information that can be exploited to match regions of interest in images.

Histogram conveys 50% white pixels and 50% black pixels

**How to represent textures?**

- Compute responses of blobs and edges at various orientations and scales.
- Ways to process:

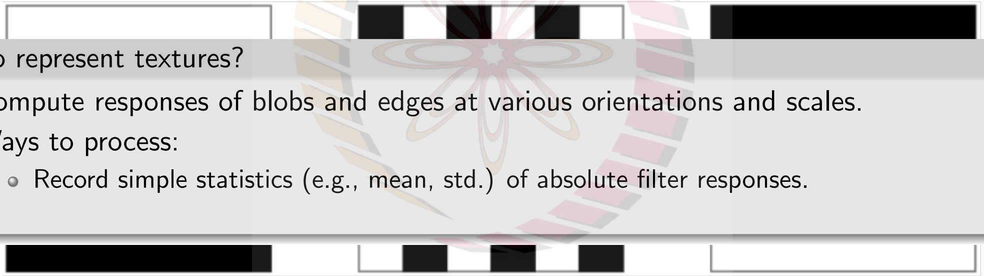
(Block Pattern) (Checkerboard Pattern) (Striped Pattern)

Drastically different textures

# Texture in Images

Conveys more information that can be exploited to match regions of interest in images.

Histogram conveys 50% white pixels and 50% black pixels



## How to represent textures?

- Compute responses of blobs and edges at various orientations and scales.
- Ways to process:
  - Record simple statistics (e.g., mean, std.) of absolute filter responses.

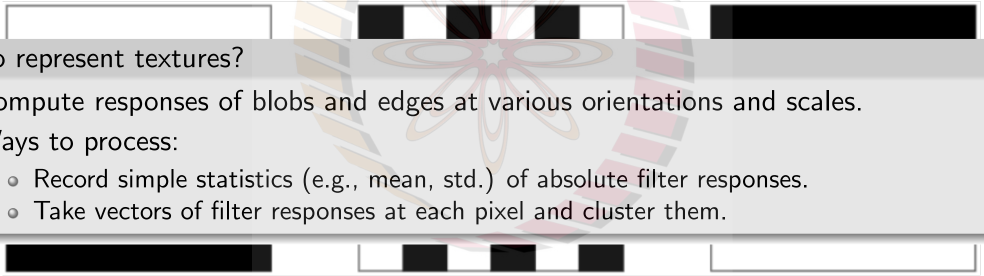
(Block Pattern) (Checkerboard Pattern) (Striped Pattern)

Drastically different textures

# Texture in Images

Conveys more information that can be exploited to match regions of interest in images.

Histogram conveys 50% white pixels and 50% black pixels



### How to represent textures?

- Compute responses of blobs and edges at various orientations and scales.
- Ways to process:
  - Record simple statistics (e.g., mean, std.) of absolute filter responses.
  - Take vectors of filter responses at each pixel and cluster them.

(Block Pattern) (Checkerboard Pattern) (Striped Pattern)

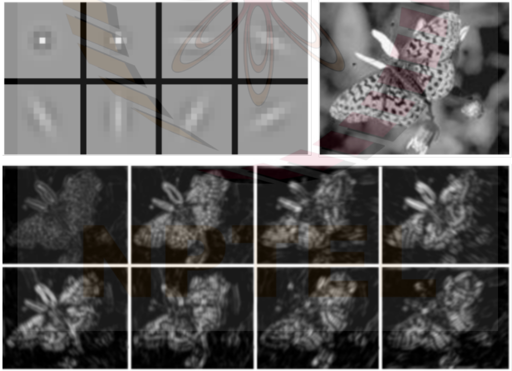
Drastically different textures

# Filter Banks

- An array of bandpass filters that separates the input signal into multiple components, each one carrying a single frequency sub-band of the original signal.

# Filter Banks

- An array of bandpass filters that separates the input signal into multiple components, each one carrying a single frequency sub-band of the original signal.
- Process image with each filter and keep responses (or squared/abs responses).



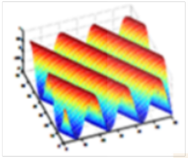
*Credit: Derek Hoiem*

## Gabor Filters

- Special classes of bandpass filters (i.e., they allow a certain 'band' of frequencies and reject the others).

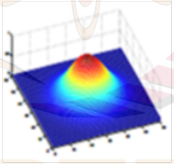
# Gabor Filters

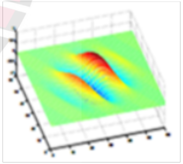
- Special classes of bandpass filters (i.e., they allow a certain 'band' of frequencies and reject the others).
- A Gabor filter can be viewed as a sinusoidal signal of particular frequency and orientation, modulated by a Gaussian wave.



A sinusoid oriented $30^\circ$ with x-axis

A 2-D Gaussian





A corresponding 2-D Gabor Filter

A 2-D Gabor filter obtained by modulating the sine wave with a Gaussian

# 2-D Gabor Filter

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = e^{-\left(\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right)} e^{i\left(2\pi \frac{x'}{\lambda} + \psi\right)}$$

where:

$x' = x \cos\theta + y \sin\theta$

$y' = -x \sin\theta + y \cos\theta$

$\theta$ – Orientation of the normal to the parallel stripes of Gabor function.

# 2-D Gabor Filter

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = e^{-\left(\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right)} e^{i\left(2\pi \frac{x'}{\lambda} + \psi\right)}$$

where:

$x' = x \cos\theta + y \sin\theta$

$y' = -x \sin\theta + y \cos\theta$

$\theta$ – Orientation of the normal to the parallel stripes of Gabor function.

$\lambda$ – Wavelength of the sinusoidal component.

# 2-D Gabor Filter

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = e^{-\left(\frac{x'^2+\gamma^2 y'^2}{2\sigma^2}\right)} e^{i\left(2\pi\frac{x'}{\lambda}+\psi\right)}$$

where:

$x' = x\cos\theta + y\sin\theta$

$y' = -x\sin\theta + y\cos\theta$

$\theta$ – Orientation of the normal to the parallel stripes of Gabor function.

$\lambda$ – Wavelength of the sinusoidal component.

$\psi$ – Phase offset of the sinusoidal function.

# 2-D Gabor Filter

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = e^{-\left(\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right)} e^{i\left(2\pi \frac{x'}{\lambda} + \psi\right)}$$

where:

$x' = x \cos\theta + y \sin\theta$

$y' = -x \sin\theta + y \cos\theta$

$\theta$ – Orientation of the normal to the parallel stripes of Gabor function.

$\lambda$ – Wavelength of the sinusoidal component.

$\psi$ – Phase offset of the sinusoidal function.

$\sigma$ – Standard deviation of the Gaussian envelope.

# 2-D Gabor Filter

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = e^{-\left(\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right)} e^{i\left(2\pi \frac{x'}{\lambda} + \psi\right)}$$

where:

$x' = x \cos\theta + y \sin\theta$

$y' = -x \sin\theta + y \cos\theta$

$\theta$ – Orientation of the normal to the parallel stripes of Gabor function.

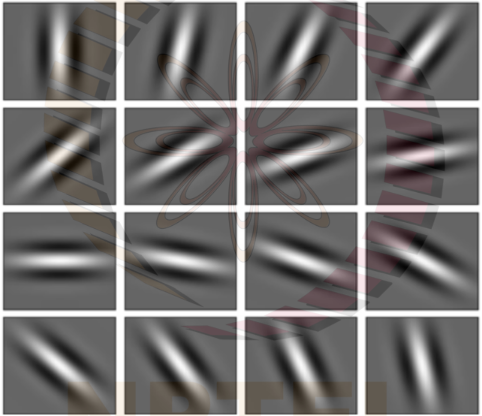
$\lambda$ – Wavelength of the sinusoidal component.

$\psi$ – Phase offset of the sinusoidal function.

$\sigma$ – Standard deviation of the Gaussian envelope.

$\gamma$ – Spatial aspect ratio and specifies the ellipticity of the support of Gabor function.

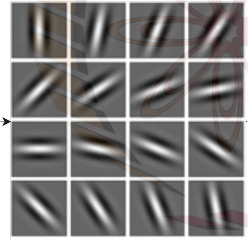
# Gabor Filter Banks



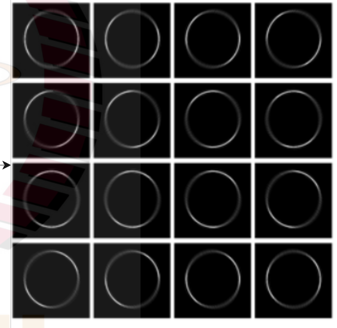
Bank of 16 Gabor filters at an orientation of 11.250 (i.e. if the first filter is at 00.00, then the second will be at 11.25, the third will be at 22.50, and so on.)

# Gabor Filter Banks

Input Image of a Circle



A bank of 16 Gabor filters



Output Image of the circle after passed through indiviual Gabor filters

# Steerable Filter Banks

Steerable Filters are a class of oriented filters that can be expressed as a linear combination of a set of basis filters.

# Steerable Filter Banks

Steerable Filters are a class of oriented filters that can be expressed as a linear combination of a set of basis filters.

- For an isotropic Gaussian filter, $G(x, y) = e^{-(x^2+y^2)}$,

$$G_1^{\theta^\circ} = G_1^{0^\circ} \cos(\theta) + G_1^{90^\circ} \sin(\theta)$$

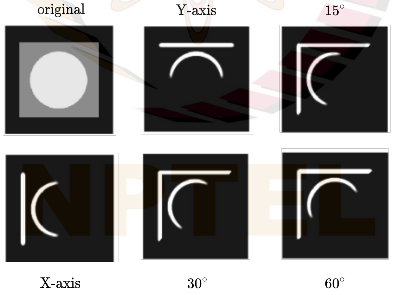
# Steerable Filter Banks

Steerable Filters are a class of oriented filters that can be expressed as a linear combination of a set of basis filters.

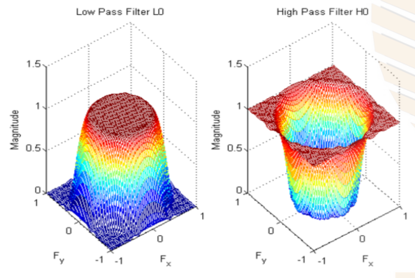
- For an isotropic Gaussian filter, $G(x, y) = e^{-(x^2+y^2)}$,

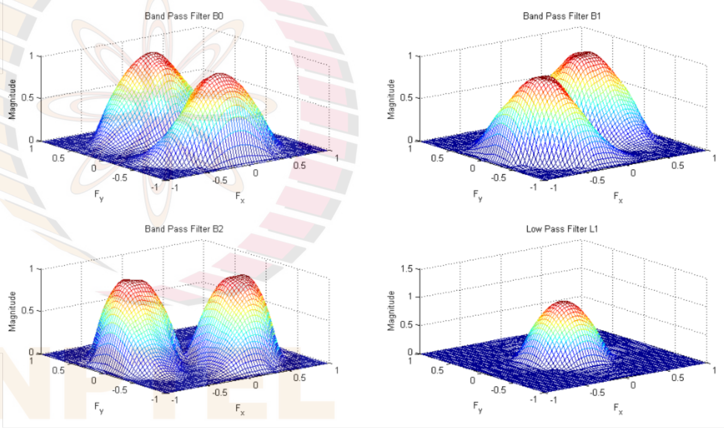
$$G_1^{\theta^\circ} = G_1^{0^\circ} \cos(\theta) + G_1^{90^\circ} \sin(\theta)$$

  where $G_1^{\theta^\circ}$ is the first derivative of $G$ at angle $\theta$.



# Steerable Filter Banks



# Homework

## Readings

- Chapter 2, Szeliski, *Computer Vision: Algorithms and Applications*

## Questions

- Why is camouflage attire effective? How?
- How is texture different from noise?
- Will scale-invariant filters be effective in matching pictures containing Matryoshka (or Russian nesting) dolls?