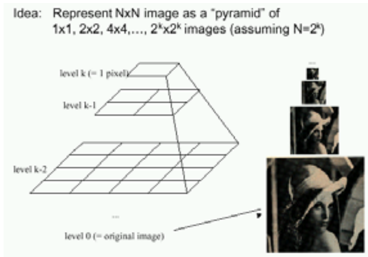
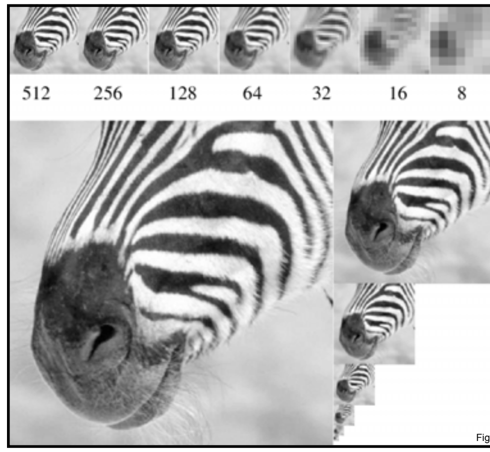
Image Pyramids



Known as a **Gaussian Pyramid** [Burt and Adelson, 1983]

- In computer graphics, a *mip map* [Williams, 1983]
- A precursor to *wavelet transform*



A bar in the big images is a line on the zebra's nose; in smaller images, a stripe; in the smallest, the animal's nose

Figure from David Forsyth

What are they good for?

Improve Search

- Search over translations
  - Classic coarse-to-fine strategy
- Search over scale
  - Template matching
  - E.g. find a face at different scales

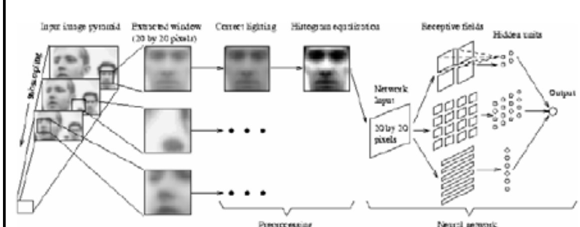
Precomputation

- Need to access image at different blur levels
- Useful for texture mapping at different resolutions (called mip-mapping)

Image Processing

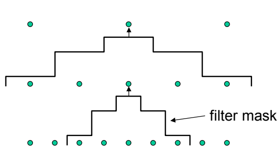
- Editing frequency bands separately
- E.g. image blending…

Example application: CMU face detector



From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

Gaussian pyramid construction



Repeat

- Filter
- Subsample

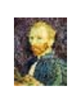
Until minimum resolution reached

- can specify desired number of levels (e.g., 3-level pyramid)

The whole pyramid is only 4/3 the size of the original image!

Image sub-sampling

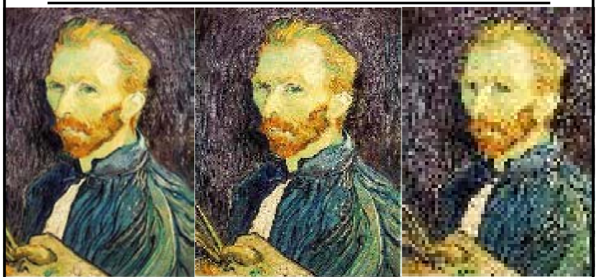
1/4



1/8

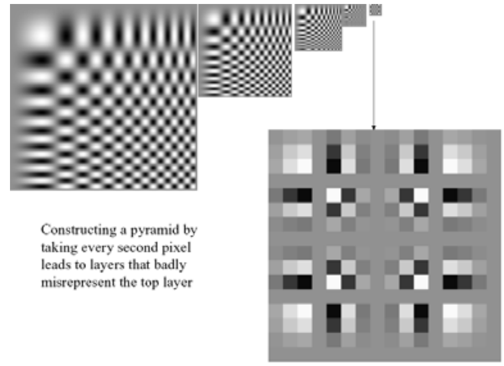
Throw away every other row and column to create a 1/2 size image
- called *image sub-sampling*

Image sub-sampling



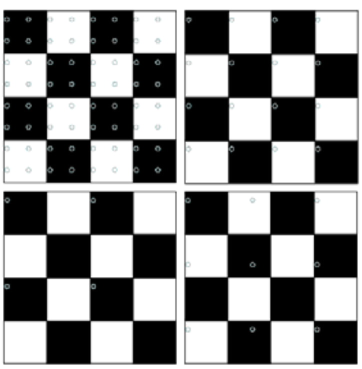
1/2   1/4 (2x zoom)   1/8 (4x zoom)

Why does this look so bad?



Constructing a pyramid by taking every second pixel leads to layers that badly misrepresent the top layer

Sampling



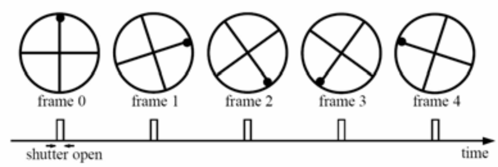
Good sampling:
•Sample often or,
•Sample wisely

Bad sampling:
•see aliasing in action!

Really bad in video

Imagine a spoked wheel moving to the right (rotating clockwise). Mark wheel with dot so we can see what's happening.

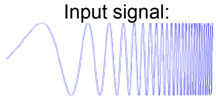
If camera shutter is only open for a fraction of a frame time (frame time = 1/30 sec. for video, 1/24 sec. for film):

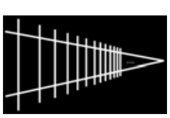


Without dot, wheel appears to be rotating slowly backwards! (counterclockwise)

Alias: n., an assumed name

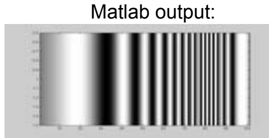
Input signal:





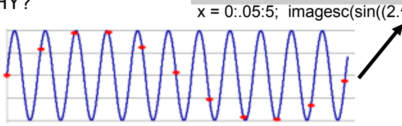
Picket fence receding Into the distance will produce aliasing…

Matlab output:



x = 0:.05:5; imagesc(sin((2.^x).*x))

WHY?

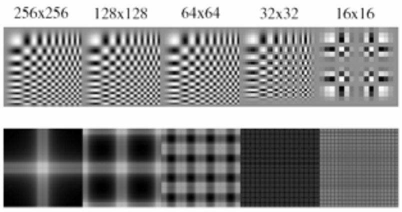


Not enough samples

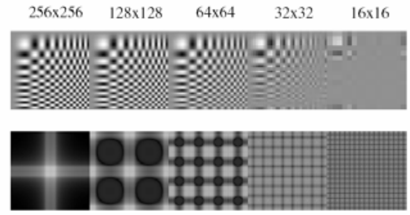
Smoothing as low-pass filtering

- The message of the FT is that high frequencies lead to trouble with sampling.
- Solution: suppress high frequencies before sampling
  - multiply the FT of the signal with something that suppresses high frequencies
  - or convolve with a low-pass filter
- A filter whose FT is a box is bad, because the filter kernel has infinite support
- Common solution: use a Gaussian
  - multiplying FT by Gaussian is equivalent to convolving image with Gaussian.

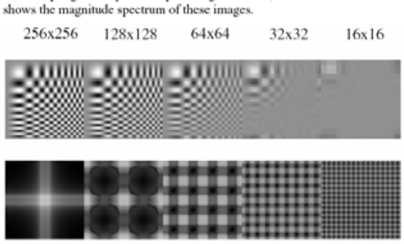
Sampling without smoothing. Top row shows the images, sampled at every second pixel to get the next; bottom row shows the magnitude spectrum of these images.



Sampling with smoothing. Top row shows the images. We get the next image by smoothing the image with a Gaussian with sigma 1 pixel, then sampling at every second pixel to get the next; bottom row shows the magnitude spectrum of these images.



Sampling with smoothing. Top row shows the images. We get the next image by smoothing the image with a Gaussian with sigma 1.4 pixels, then sampling at every second pixel to get the next; bottom row shows the magnitude spectrum of these images.



Gaussian pre-filtering

Gaussian 1/2   G 1/4   G 1/8

Solution: filter the image, *then* subsample

Subsampling with Gaussian pre-filtering

Gaussian 1/2   G 1/4   G 1/8

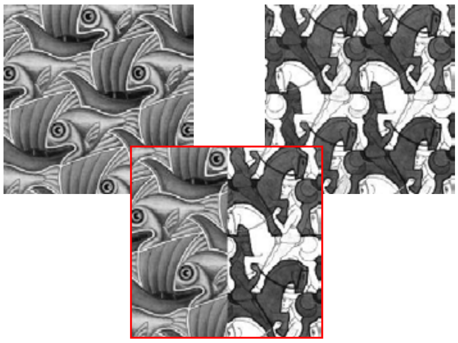
Solution: filter the image, *then* subsample

- Filter size should double for each ½ size reduction. Why?
- How can we speed this up?

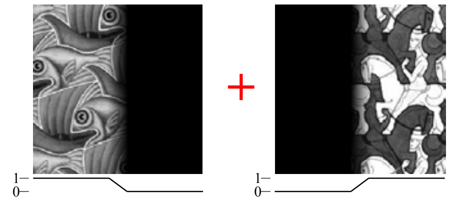
Compare with...

1/2   1/4 (2x zoom)   1/8 (4x zoom)

Image Blending



Feathering

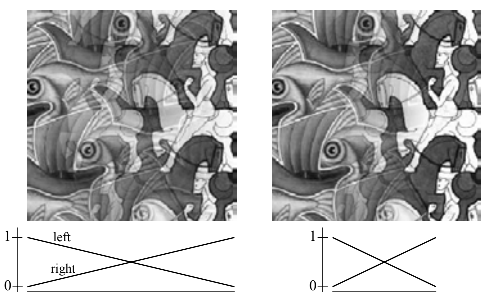


Encoding transparency

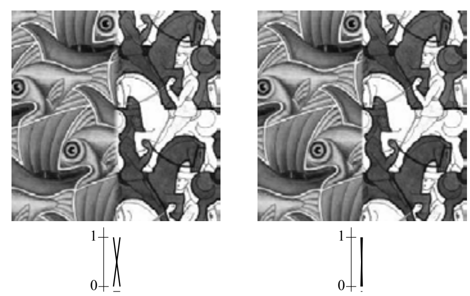
$I(x,y) = (\alpha R, \alpha G, \alpha B, \alpha)$

$I_{blend} = I_{left} + I_{right}$

Affect of Window Size



Affect of Window Size



Good Window Size

“Optimal” Window: smooth but not ghosted

What is the Optimal Window?

To avoid seams

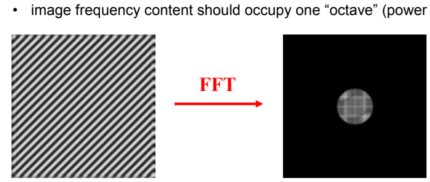
- window >= size of largest prominent feature

To avoid ghosting

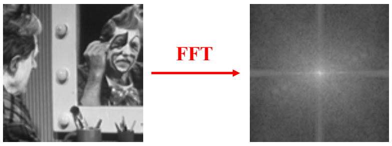
- window <= 2*size of smallest prominent feature

Natural to cast this in the *Fourier domain*

- largest frequency <= 2*size of smallest frequency
- image frequency content should occupy one “octave” (power of two)



What if the Frequency Spread is Wide



Idea (Burt and Adelson)

- Compute $F_{left} = FFT(I_{left})$, $F_{right} = FFT(I_{right})$
- Decompose Fourier image into octaves (bands)
  - $F_{left} = F_{left}^1 + F_{left}^2 + \ldots$
- Feather corresponding octaves F_{left}^i with F_{right}^i
  - Can compute inverse FFT and feather in spatial domain
- Sum feathered octave images in frequency domain

Better implemented in *spatial domain*

What does blurring take away?

original

What does blurring take away?

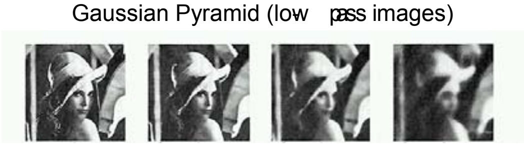
smoothed (5x5 Gaussian)

High-Pass filter

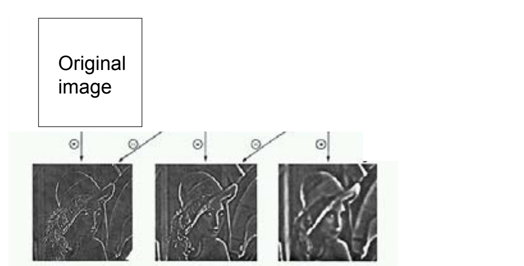
smoothed – original

Band-pass filtering

Gaussian Pyramid (low-pass images)

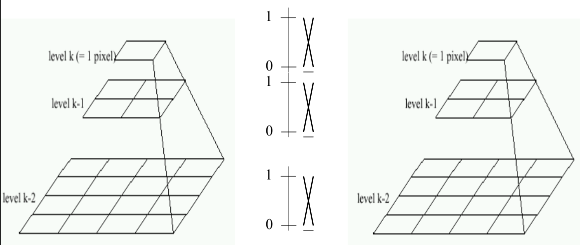


Laplacian Pyramid



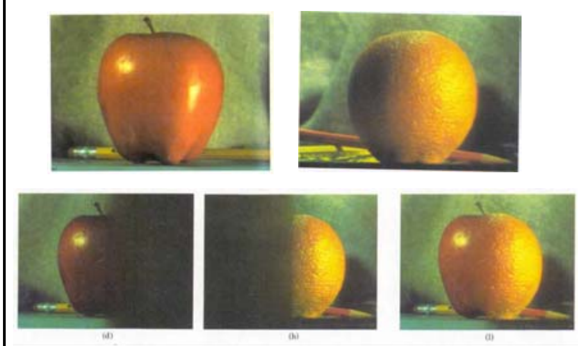
How can we reconstruct (collapse) this pyramid into the original image?

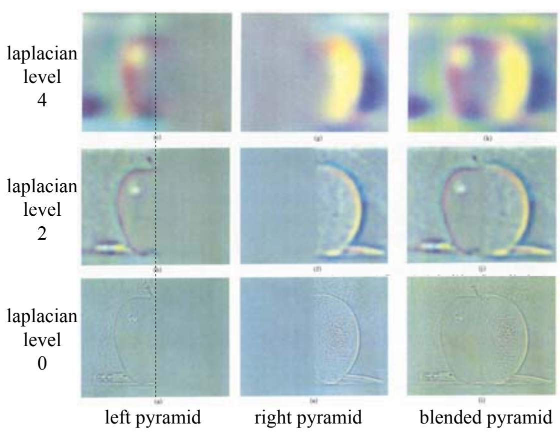
Pyramid Blending



Left pyramid blend Right pyramid

Pyramid Blending





Laplacian Pyramid: Blending

General Approach:

1. Build Laplacian pyramids *LA* and *LB* from images *A* and *B*
2. Build a Gaussian pyramid *GR* from selected region *R*
3. Form a combined pyramid *LS* from *LA* and *LB* using nodes of *GR* as weights:
   - *LS(i,j) = GR(I,j,)*LA(I,j) + (1-GR(I,j))*LB(I,j)*
4. Collapse the *LS* pyramid to get the final blended image

Blending Regions