**Image Pyramids**

- Represent an image as a “pyramid” of 1x1, 2x2, 4x4, ..., $2^k$ size images (assuming $N=2^k$)

Known as a **Gaussian Pyramid** [Burt and Adelson, 1983]
- In computer graphics, a mip map [Williams, 1983]
- A precursor to wavelet transform

---

**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…

---

**Gaussian pyramid construction**

- Filter
- Subsample

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**

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

---

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

---

Example application: CMU face detector

Image sub-sampling

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

Why does this look so bad?

Sampling

Good sampling:
- Sample often or,
- Sample wisely

Bad sampling:
- see aliasing in action!

Really bad in video

Imagine a spokeed 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! (counter-clockwise)

Alias: n., an assumed name

Picket fence receding into the distance will produce aliasing…

WHY?

Input signal:

Matlab output:

x = 0.05:5; imagesc(sin((2.^x).*x))

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.
Gaussian pre-filtering

Solution: filter the image, then subsample

Subsampling with Gaussian pre-filtering

Solution: filter the image, then subsample
- Filter size should double for each \( \frac{1}{2} \) size reduction. Why?
- How can we speed this up?
Image Blending

Feathering

Encoding transparency
\[ I(x,y) = (\alpha R, \alpha G, \alpha B, \alpha) \]

\[ I_{\text{blend}} = I_{\text{left}} + I_{\text{right}} \]

Affect of Window Size

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)

“Optimal” Window: smooth but not ghosted
What if the Frequency Spread is Wide

Idea (Burt and Adelson)
- Compute $F_{\text{left}} = \text{FFT}(I_{\text{left}})$, $F_{\text{right}} = \text{FFT}(I_{\text{right}})$
- Decompose Fourier image into octaves (bands)
  - $F_{\text{left}} = F_{\text{left}_1} + F_{\text{left}_2} + \ldots$
- Feather corresponding octaves $F_{\text{left}_i}$ with $F_{\text{right}_i}$
  - 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

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

General Approach:
1. Build Laplacian pyramids $L_A$ and $L_B$ from images $A$ and $B$
2. Build a Gaussian pyramid $G_R$ from selected region $R$
3. Form a combined pyramid $L_S$ from $L_A$ and $L_B$ using nodes of $G_R$ as weights:
   - $L_S(i,j) = G_R(i,j) \cdot L_A(i,j) + (1 - G_R(i,j)) \cdot L_B(i,j)$
4. Collapse the $L_S$ pyramid to get the final blended image