Lecture 11

Image Blending, Image Pyramids

Last Time

- Mosaics and Panoramas

Projective Geometry

•Goal is to stitch together many shots to get a wide-angle panorama.

•Need to define a model for the camera and its motion

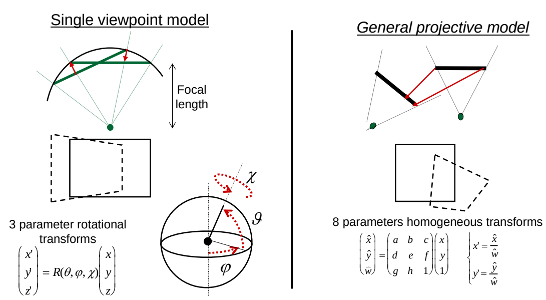
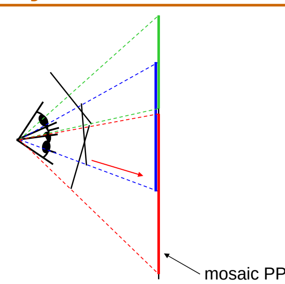


Image reprojection

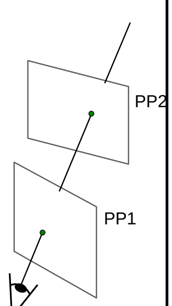


- The mosaic has a natural interpretation in 3D
  - The images are reprojected onto a common plane
  - The mosaic is formed on this plane
  - Mosaic is a *synthetic wide-angle camera*

Homography

- Projective – mapping between any two PPs with the same center of projection
  - rectangle should map to arbitrary quadrilateral
  - parallel lines aren't
  - but must preserve straight lines
  - same as: project, rotate, reproject
- called Homography

$$\underbrace{\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix}}_{\mathbf{x'}} = \underbrace{\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}}_{\mathbf{H}} \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{\mathbf{x}}$$



To apply a homography **H**

- Compute **x' = Hx** (regular matrix multiply)
- Convert **x'** from homogeneous to image coordinates

Today

- Solving homographies
- Image blending
  - Region blending
- Image pyramids
  - Gaussian pyramids
  - Laplacian pyramids

Solving for homographies

$$\mathbf{x}' = \mathbf{H}\mathbf{x}$$

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Can set scale factor h_{33}=1. So, there are 8 unkowns.
- Set up a system of linear equations (re-write **H** as a vector **h**):

$$\mathbf{A}\mathbf{h} = \mathbf{b}$$

where vector of unknowns **h** = $[h_{11},h_{12},h_{13},h_{21}, h_{22},h_{23}, h_{31},h_{32}]^T$

- Note: we do not know w, but we can compute it from x & y

$$w=h_{31}x+h_{32}y+1$$

- The equations are linear in the unknown

Method I: Solving for $\mathbf{h}$

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \qquad y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}$$

$$\begin{aligned} x'(h_{31}x + h_{32}y + h_{33}) &= h_{11}x + h_{12}y + h_{13} \\ y'(h_{31}x + h_{32}y + h_{33}) &= h_{21}x + h_{22}y + h_{23} \end{aligned}$$

- If $h_{33} = 1$

$$\begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & -y'_1y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -y'_2x_2 & -y'_2y_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -y'_3x_3 & -y'_3y_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -y'_4x_4 & -y'_4y_4 \end{pmatrix} \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{pmatrix} = \begin{pmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ x'_3 \\ y'_3 \\ x'_4 \\ y'_4 \end{pmatrix}$$

- The problem with this approach is that if the true solution is h_{33}=0 , then this cannot be reached.
- Consequently, a poor quality estimate of $\mathbf{H}$ will be obtained.

Method II: Solving for $\mathbf{h}$

$$\begin{aligned} x'(h_{31}x + h_{32}y + h_{33}) &= h_{11}x + h_{12}y + h_{13} \\ y'(h_{31}x + h_{32}y + h_{33}) &= h_{21}x + h_{22}y + h_{23} \end{aligned}$$

- Re-arrange: $$\begin{pmatrix} x & y & 1 & 0 & 0 & 0 & -x'x & -x'y & -x' \\ 0 & 0 & 0 & x & y & 1 & -y'x & -y'y & -y' \end{pmatrix}\mathbf{h} = \mathbf{0}$$

  where $\mathbf{h} = (h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32}, h_{33})^\top$

- Rewrite: $$\begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & -y'_1y_1 & -y'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 & -x'_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -y'_2x_2 & -y'_2y_2 & -y'_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 & -x'_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -y'_3x_3 & -y'_3y_3 & -y'_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 & -x'_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -y'_4x_4 & -y'_4y_4 & -y'_4 \end{pmatrix}\mathbf{h} = \mathbf{0}$$

- For **n > 4** real point correspondences, **A** is a $2n$ x 9 matrix, and there will not be an exact solution to $\mathbf{Ah}=\mathbf{0}$.
- In this case, a sensible procedure is to again minimise the residuals. It can be shown that the vector that minimises $\mathbf{h}^\top\mathbf{A}^\top\mathbf{A}\mathbf{h}$ subject to ||**h**||=1 , is the eigenvector with least eigenvalue of $\mathbf{A}^\top\mathbf{A}$.

Method III: Discrete Linear Transform

- Solve for **H**: $\mathbf{x}' = \mathbf{H}\mathbf{x}$
- $\mathbf{x}_i = \begin{bmatrix} x_i \\ y_i \\ w_i \end{bmatrix}$ and $\mathbf{x}'_i = \begin{bmatrix} x'_i \\ y'_i \\ w'_i \end{bmatrix}$ have different lengths but are collinear
- Therefore, $\mathbf{x}'_i \times \mathbf{H}\mathbf{x}_i = \mathbf{0}$

$$\mathbf{x}'_i \times \mathbf{H}\mathbf{x}_i = \begin{bmatrix} y'_i\mathbf{h}_3^T\mathbf{x}_i - w'_i\mathbf{h}_2^T\mathbf{x}_i \\ w'_i\mathbf{h}_1^T\mathbf{x}_i - x'_i\mathbf{h}_3^T\mathbf{x}_i \\ x'_i\mathbf{h}_2^T\mathbf{x}_i - y'_i\mathbf{h}_1^T\mathbf{x}_i \end{bmatrix}$$

Let $\mathbf{h}_j^T$ be row j of $\mathbf{H}$, $\mathbf{h}$ be stacked $\mathbf{h}_j$'s

Review: Cross Product

$$\mathbf{a} \times \mathbf{b} = \|\mathbf{a}\|\,\|\mathbf{b}\|\sin(\theta) \qquad \theta = \textit{angle between } \mathbf{a}, \mathbf{b}$$

- when a and b are colinear (parallel) then the cross product is 0.
- For additional properties, see:
  http://chortle.ccsu.edu/VectorLessons/index.html

Review: Matrix Form of the Cross Product

The vector cross product also acts on two vectors and returns a third vector. Geometrically, this new vector is constructed such that its projection onto either of the two input vectors is zero.

$$\vec{a} \times \vec{b} = \begin{bmatrix} a_yb_z - a_zb_y \\ a_zb_x - a_xb_z \\ a_xb_y - a_yb_x \end{bmatrix}$$

$$\vec{a} \times \vec{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = \vec{c} \qquad \begin{aligned} \vec{a} \cdot \vec{c} &= 0 \\ \vec{b} \cdot \vec{c} &= 0 \end{aligned}$$

Review: Matrix Form of the Cross Product

$$\vec{a}\times\vec{b}=\begin{bmatrix}0 & -a_z & a_y\\ a_z & 0 & -a_x\\ -a_y & a_x & 0\end{bmatrix}\begin{bmatrix}b_x\\ b_y\\ b_z\end{bmatrix}=\vec{c} \qquad \begin{matrix}\vec{a}\cdot\vec{c}=0\\ \vec{b}\cdot\vec{c}=0\end{matrix}$$

$$[a_x]=\begin{bmatrix}0 & -a_z & a_y\\ a_z & 0 & -a_x\\ -a_y & a_x & 0\end{bmatrix}$$

$$\boxed{\vec{a}\times\vec{b}=[a_x]\vec{b}}$$

DLT: *Homography Estimation*

- Since vectors are homogeneous, $\mathbf{x}'_i, \mathbf{H}\mathbf{x}_i$ are parallel, so

$$\mathbf{x}'_i \times \mathbf{H}\mathbf{x}_i = \mathbf{0}$$

- Let $\mathbf{h}_j^T$ be row *j* of $\mathbf{H}$, $\mathbf{h}$ be stacked $\mathbf{h}_j$'s
- Expanding and rearranging cross product above, we obtain $\mathbf{A}_i\mathbf{h}=\mathbf{0}$, where

$$\mathbf{A}_i=\begin{bmatrix}\mathbf{0}^T & -w'_i\mathbf{x}_i^T & y'_i\mathbf{x}_i^T\\ w'_i\mathbf{x}_i^T & \mathbf{0}^T & -x'_i\mathbf{x}_i^T\\ -y'_i\mathbf{x}_i^T & x'_i\mathbf{x}_i^T & \mathbf{0}^T\end{bmatrix}$$

DLT Algorithm:

- Only 2 linearly independent equations in each $\mathbf{A}_i$, leave out 3rd row to make it 2 x 9
- Stack every $\mathbf{A}_i$ to get 2*n* x 9 $\mathbf{A}$
- Solve $\mathbf{A}\mathbf{h}=\mathbf{0}$ by computing singular value decomposition (SVD) $\mathbf{A}=\mathbf{U}\mathbf{D}\mathbf{V}^T$
- $\mathbf{h}$ is last column of $\mathbf{V}$
- Solution is improved by normalizing image coordinates before applying DLT

DLT: Normalization

Objective

Given $n \geq 4$ 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the 2D homography matrix $\mathtt{H}$ such that $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$.

Algorithm

(i) **Normalization of x:** Compute a similarity transformation $\mathtt{T}$, consisting of a translation and scaling, that takes points $\mathbf{x}_i$ to a new set of points $\tilde{\mathbf{x}}_i$ such that the centroid of the points $\tilde{\mathbf{x}}_i$ is the coordinate origin $(0,0)^\top$, and their average distance from the origin is $\sqrt{2}$.

(ii) **Normalization of x′:** Compute a similar transformation $\mathtt{T}'$ for the points in the second image, transforming points $\mathbf{x}'_i$ to $\tilde{\mathbf{x}}'_i$.

(iii) **DLT:** Apply algorithm 3.1(*p*73) to the correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$ to obtain a homography $\tilde{\mathtt{H}}$.

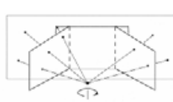
(iv) **Denormalization:** Set $\mathtt{H} = \mathtt{T}'^{-1}\tilde{\mathtt{H}}\mathtt{T}$.

Panoramic Mosaicing

Rotation about camera center: homography

- choose one image as reference
- compute homography to map neighboring image to reference image plane
- projectively warp image, add to reference plane
- repeat for all images

⇒ bow tie shape



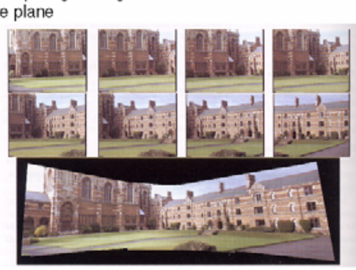
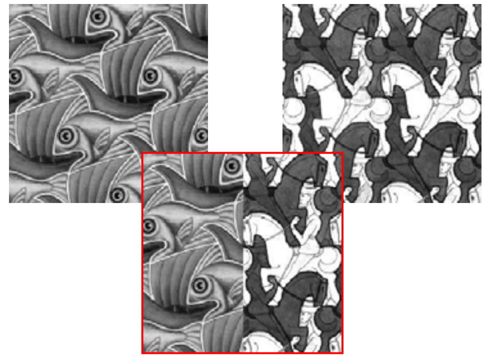


Image Blending

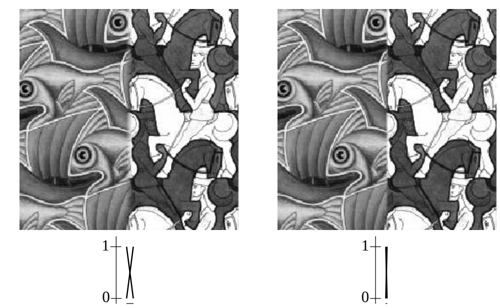
Simplification: Two-band Blending

- Brown & Lowe, 2003
  - Only use two bands: high freq. and low freq.
  - Blends low freq. smoothly

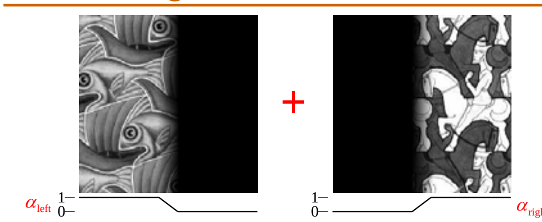
Image Blending



Affect of Window Size



Feathering

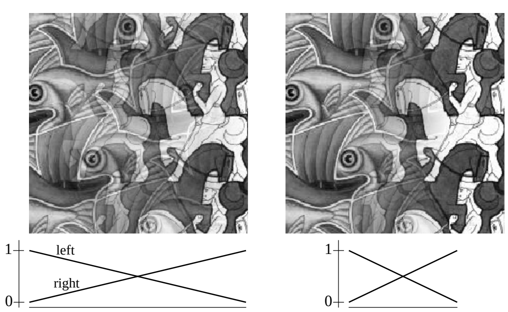


Encoding transparency

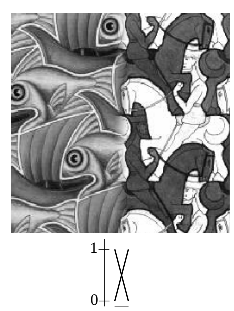
$I_{left}(x,y) = (\alpha_{left}R, \alpha_{left}G, \alpha_{left}B, \alpha_{left})$

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

Affect of Window Size



Good Window Size



“Optimal” Window: smooth but not ghosted

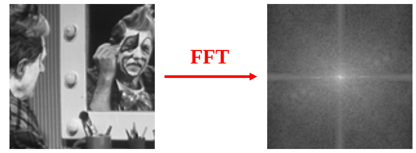
What is the Optimal Window?

- 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

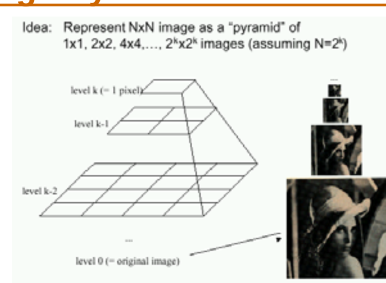
What does blurring take away?

smoothed (5x5 Gaussian)

High-Pass filter

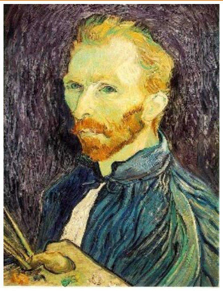
smoothed – original

Image Pyramids



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

Image sub-sampling

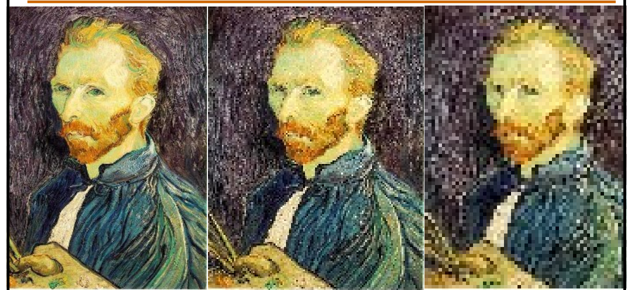


1/4

1/8

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

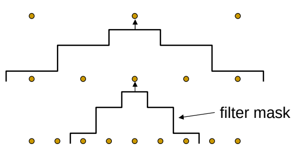
Image sub-sampling



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

Why does this look so bad?

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!

Gaussian pre-filtering

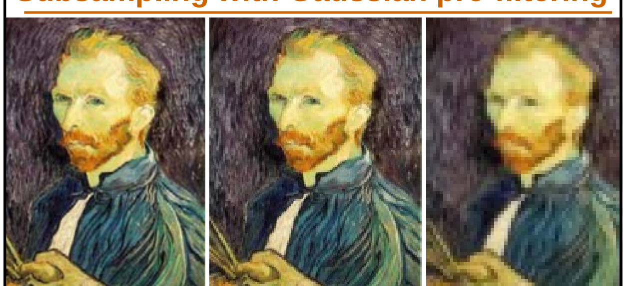
Gaussian 1/2

G 1/4

G 1/8

- Solution: filter the image, *then* subsample

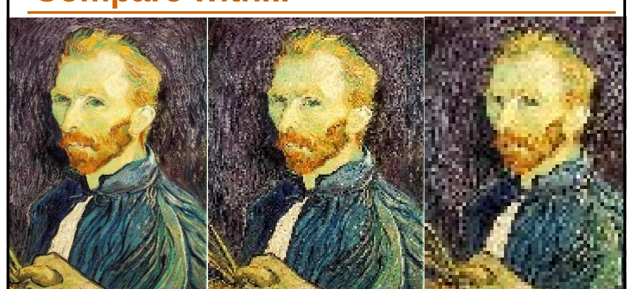
Subsampling with Gaussian pre-filtering



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

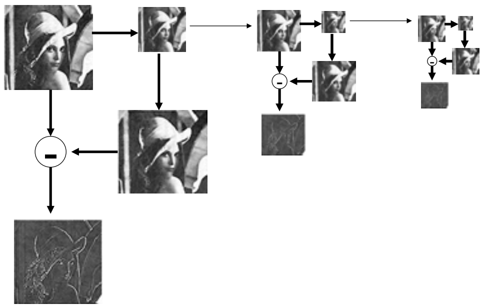
- Solution: filter the image, *then* subsample
  - Filter size should double for each ½ size reduction. Why?
  - How can we speed this up?

Compare with...

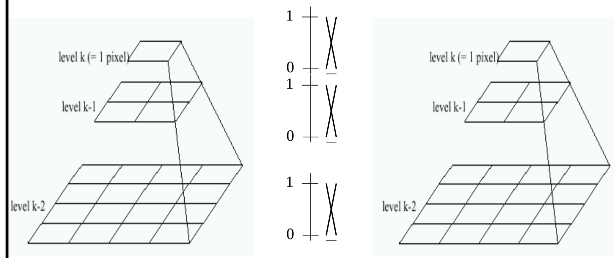


1/2 | 1/4 (2x zoom) | 1/8 (4x zoom)

Laplacian pyramid algorithm

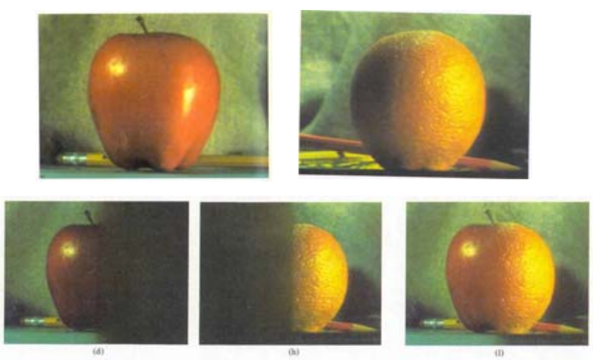


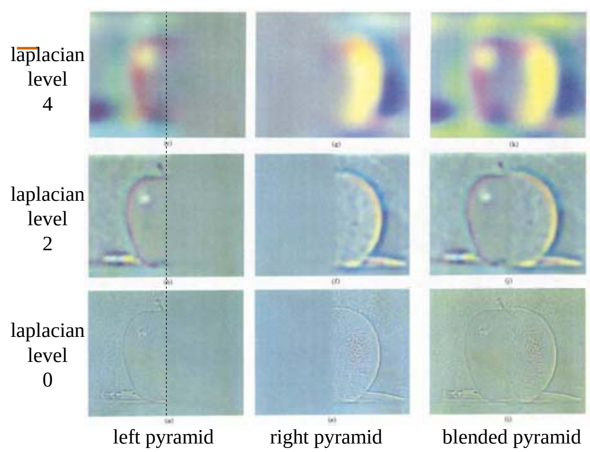
Pyramid Blending



Left pyramid blend Right pyramid

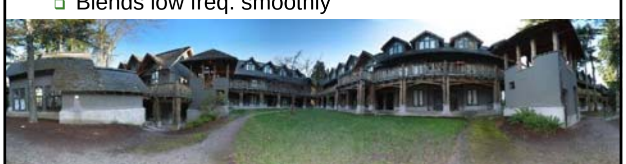
Pyramid Blending





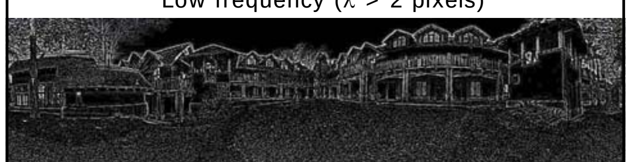
Simplification: Two-band Blending

- Brown & Lowe, 2003
  - Only use two bands: high freq. and low freq.
  - Blends low freq. smoothly



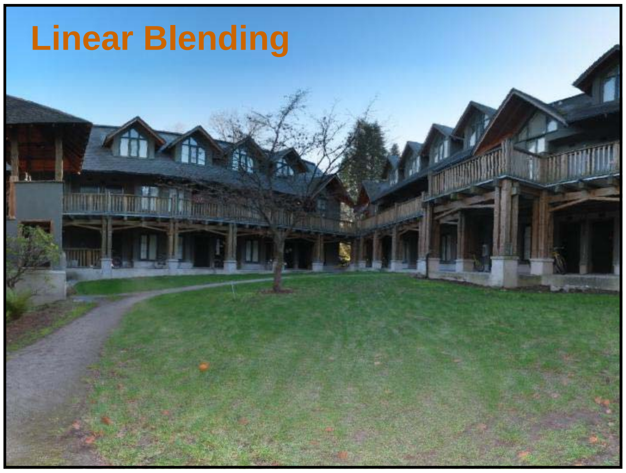
2-band Blending

Low frequency (λ > 2 pixels)



High frequency (λ < 2 pixels)

Linear Blending



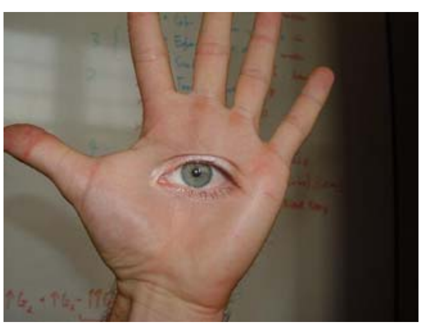
2-band Blending

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

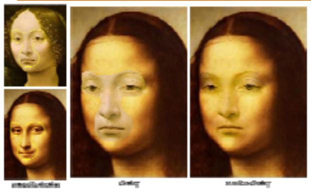
Horror Photo



© prof. dmartin

Perez et al., 2003

Perez et al, 2003



editing

- Limitations:
  - Can’t do contrast reversal (gray on black -> gray on white)
  - Colored backgrounds “bleed through”
  - Images need to be very well aligned

Some Artfacts Left ….

- **Pyramid blending does not solve this**
  - **Ghosting—objects move in the scene.**
  - **Differing exposures between images.**

Ghost removal

- **In regions with differences don’t blend - crop.**

M. Uyttendaele, A. Eden, and R. Szeliski.
Eliminating ghosting and exposure artifacts in image mosaics.
In Proceedings of the International Conference on Computer Vision and Pattern Recognition, volume 2, pages 509--516, Kauai, Hawaii, December 2001.

Magic: ghost removal

M. Uyttendaele, A. Eden, and R. Szeliski.
Eliminating ghosting and exposure artifacts in image mosaics.
In Proceedings of the International Conference on Computer Vision and Pattern Recognition, volume 2, pages 509--516, Kauai, Hawaii, December 2001.

Refs

- http://graphics.cs.cmu.edu/courses/15-463/2004_fall/www/Papers/MSR-TR-2004-92-Sep27.pdf
- http://www.cs.ubc.ca/~mbrown/papers/iccv2003.pdf
- http://www.cs.washington.edu/education/courses/csep576/05wi/readings/szeliskiShum97.pdf
- http://portal.acm.org/citation.cfm?id=218395&dl=ACM&coll=portal
- http://research.microsoft.com/~brown/papers/cvpr05.pdf
- http://citeseer.ist.psu.edu/mann94virtual.html
- http://grail.cs.washington.edu/projects/panovidtex/
- http://research.microsoft.com/users/mattu/pubs/Deghosting.pdf
- http://research.microsoft.com/vision/visionbasedmodeling/publications/Baudisch-OZCHI05.pdf
- http://www.vision.caltech.edu/lihi/Demos/SquarePanorama.html
- http://graphics.stanford.edu/papers/multi-cross-slits/