Various Perspectives
(Mosaics and Panoramas)

Projects

- List available now
- Email to me: group, topic, why it is interesting
  - until Thursday next week (24th of May)
- Project proposal (2 pages): 1st of June
- Project idea presentation: 8th of June
- Final Project presentation: 20th of July
- Project report

Mosaics and Panoramas

- basic idea
- registration
- resample
- blend

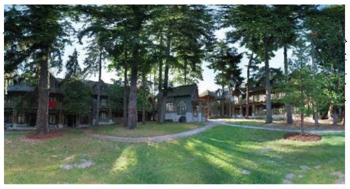
Why Mosaic?

Are you getting the whole picture?
- Compact Camera FOV = 50 x 35°

Slide from Brown & Lowe

Why Mosaic?

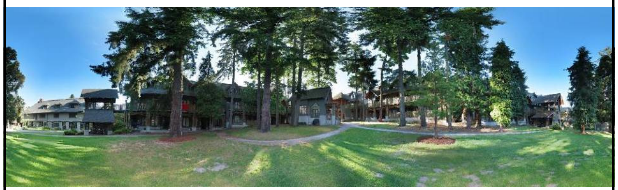
Are you getting the whole picture?
- Compact Camera FOV = 50 x 35°
- Human FOV = 200 x 135°



Slide from Brown & Lowe

Why Mosaic?

Are you getting the whole picture?
- Compact Camera FOV = 50 x 35°
- Human FOV = 200 x 135°
- Panoramic Mosaic = 360 x 180°



Slide from Brown & Lowe

Single vs. Multiple Viewpoint

Single-viewpoint

- Necessary for creating pure perspective images.
- Many vision algorithms assume pinhole cameras.
- Images that aren't perspective images look distorted.

Multi-viewpoint

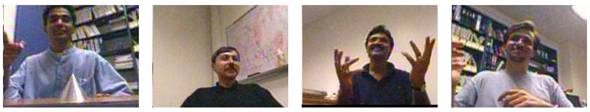
- Cross-slit panoramas, etc.
- necessary for scenes which cannot be captured from a single viewpoint

Omnidirectional (Catadioptric) Cameras

O-360

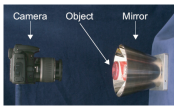
EyeSee360

Images of an Omnidirectional Camera



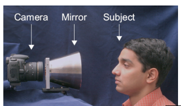
images: CAVE lab

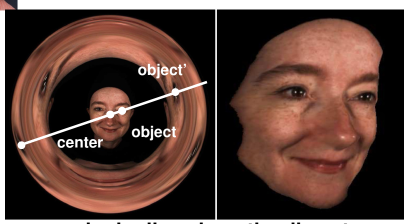
Catadioptric System – Full Texture



[Kuthirummal 2006]

Catadioptric System – Stereo

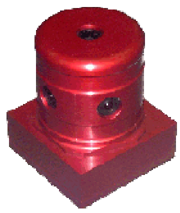




epipolar line along the diameter

Multi-camera, Single-viewpoint ?

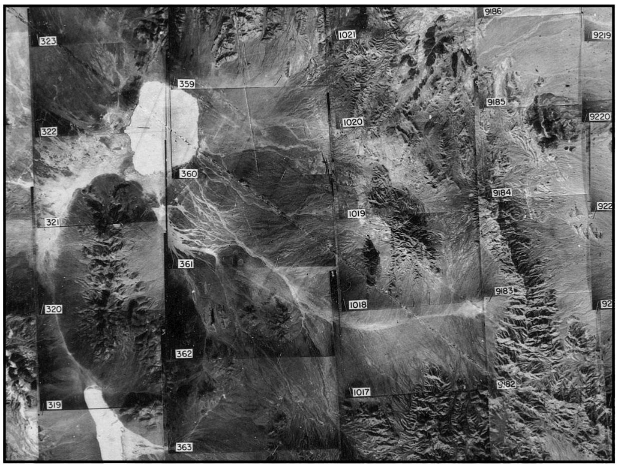
Immersive Media "Dodeca2000"



PointGrey Ladybug

Image Mosaicing

- Register multiple images
- Blend



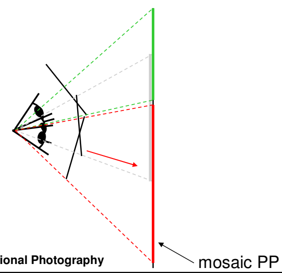
Single Center of Projection

- Take a sequence of images from the same position
  - Rotate the camera about its optical center
- Compute transformation between second image and first
- Transform the second image to overlap with the first
- Blend the two together to create a mosaic
- If there are more images, repeat

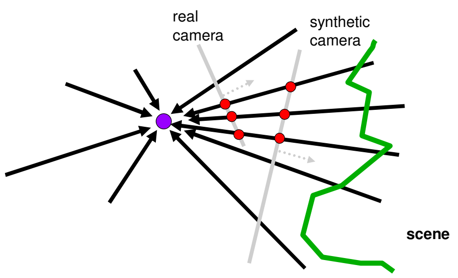
…why don't we need the 3D geometry?

Image Reprojection

- The images are reprojected onto a common plane
- The mosaic is formed on this plane
- Mosaic is a *synthetic wide-angle camera*



A pencil of rays contains all views



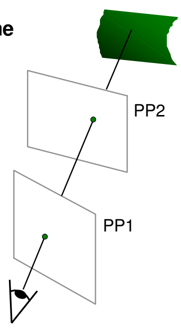
Can generate any synthetic camera view
as long as it has **the same center of projection**!

Image reprojection

How to relate two images from the same camera center?

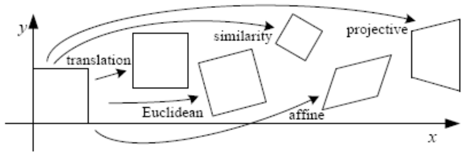
Images contain the same information along the same ray.

Use 2D image wrap instead of ray tracing.



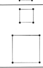
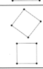
Taxonomy of Projective Transformations

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix},$$

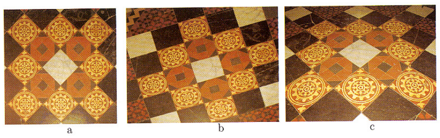


Taxonomy of Projective Transformations

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix},$$

Group	Matrix	Distortion	Invariant properties
Projective 8 dof	$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$		Concurrency, collinearity, **order of contact**: intersection (1 pt contact); tangency (2 pt contact); inflections (3 pt contact with line); tangent discontinuities and cusps. cross ratio (ratio of ratio of lengths).
Affine 6 dof	$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Parallelism, ratio of areas, ratio of lengths on collinear or parallel lines (e.g. midpoints), linear combinations of vectors (e.g. centroids). The line at infinity, $\mathbf{l}_\infty$.
Similarity 4 dof	$\begin{bmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Ratio of lengths, angle. The circular points, $\mathbf{I}, \mathbf{J}$ (see section 1.7.3).
Euclidean 3 dof	$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Length, area

Distortions under Central Projection



- **Similarity**: circle remains circle, square remains square

⇒ line orientation is preserved

- **Affine**: circle becomes ellipse, square becomes rhombus

⇒ parallel lines remain parallel

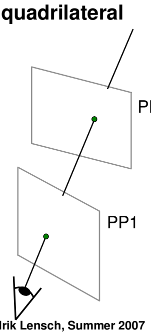
- **Projective**: imaged object size depends on distance from camera

⇒ parallel lines converge

Homography

A: 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**



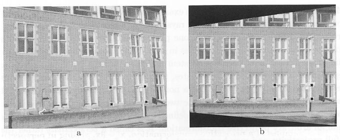
called Homography

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

To apply a homography **H**

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

Removing Projective Distortion



$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix},$$

Projective transformation in inhomogeneous form

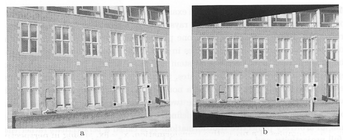
$$x' = \frac{x'_1}{x'_3} = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \qquad y' = \frac{x'_2}{x'_3} = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}.$$

4 general point correspondences (x,y ->x',y') on the planar facade lead to eight linear equations of the type

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

Sufficient to solve for **H** up to multiplicative factor

The Direct Linear Transform (DLT) Algorithm



Given: 4 2D point correspondences $\mathbf{x}_i = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \Leftrightarrow \mathbf{x}_i' = \begin{pmatrix} x_1' \\ x_2' \\ x_3' \end{pmatrix}$

Objective: estimate the projective transform matrix **H**

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

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix},$$

The DLT Algorithm II

Estimating matrix **H** from point correspondences is equivalent to

$$\mathbf{x}' = \mathtt{H}\mathbf{x}. \qquad \mathbf{x_i} = \begin{pmatrix} x_i \\ y_i \\ w_i \end{pmatrix} \Leftrightarrow \mathbf{x_i}' = \begin{pmatrix} x_i' \\ y_i' \\ w_i' \end{pmatrix} \qquad \mathbf{x}_i' \times \mathtt{H}\mathbf{x}_i = \mathbf{0}$$

might have different length but are collinear

gives

$$\mathbf{x}_i' \times \mathtt{H}\mathbf{x}_i = \begin{pmatrix} y_i'\mathbf{h}^{3\top}\mathbf{x}_i - w_i'\mathbf{h}^{2\top}\mathbf{x}_i \\ w_i'\mathbf{h}^{1\top}\mathbf{x}_i - x_i'\mathbf{h}^{3\top}\mathbf{x}_i \\ x_i'\mathbf{h}^{2\top}\mathbf{x}_i - y_i'\mathbf{h}^{1\top}\mathbf{x}_i \end{pmatrix}.$$

Re-ording into **h** vector

$$\begin{bmatrix} \mathbf{0}^\top & -w_i'\mathbf{x}_i^\top & y_i'\mathbf{x}_i^\top \\ w_i'\mathbf{x}_i^\top & \mathbf{0}^\top & -x_i'\mathbf{x}_i^\top \\ -y_i'\mathbf{x}_i^\top & x_i'\mathbf{x}_i^\top & \mathbf{0}^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}.$$

The DLT Algorithm III

$$\begin{bmatrix} \mathbf{0}^\top & -w_i'\mathbf{x}_i^\top & y_i'\mathbf{x}_i^\top \\ w_i'\mathbf{x}_i^\top & \mathbf{0}^\top & -x_i'\mathbf{x}_i^\top \\ -y_i'\mathbf{x}_i^\top & x_i'\mathbf{x}_i^\top & \mathbf{0}^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}. \qquad \mathbf{x_i} = \begin{pmatrix} x_i \\ y_i \\ w_i \end{pmatrix} \Leftrightarrow \mathbf{x_i}' = \begin{pmatrix} x_i' \\ y_i' \\ w_i' \end{pmatrix}$$

Only rows 1 and 2 are linearly independent ⇒ omit row 3

$$\begin{bmatrix} \mathbf{0}^\top & -w_i'\mathbf{x}_i^\top & y_i'\mathbf{x}_i^\top \\ w_i'\mathbf{x}_i^\top & \mathbf{0}^\top & -x_i'\mathbf{x}_i^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}. \qquad \mathtt{A}_i\mathbf{h} = \mathbf{0}$$

Inhomogeneous solution: set one matrix entry equal to 1 (e.g. h33)

$$\begin{bmatrix} 0 & 0 & 0 & -x_iw_i' & -y_iw_i' & -w_iw_i' & x_iy_i' & y_iy_i' \\ x_iw_i' & y_iw_i' & w_iw_i' & 0 & 0 & 0 & -x_ix_i' & -y_ix_i' \end{bmatrix} \tilde{\mathbf{h}} = \begin{pmatrix} -w_iy_i' \\ w_ix_i' \end{pmatrix}$$

Solve by Gaussian elimination or least-squares techniques

Estimating Homographies

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 $\mathbf{x}$:** 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 $\sqrt{2}$.

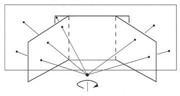
(ii) **Normalization of $\mathbf{x}'$:** Compute a similar transformation $\mathtt{T}'$ for the points in the second image, transforming points $\mathbf{x}_i'$ to $\tilde{\mathbf{x}}_i'$.

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

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

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

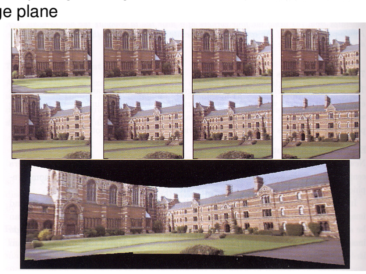
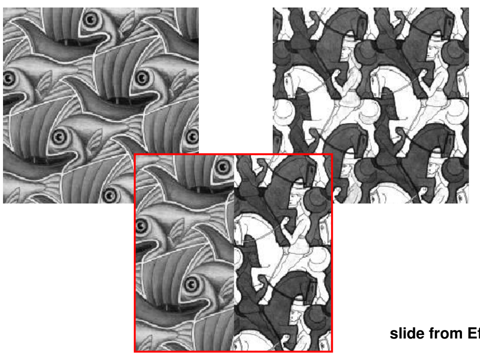
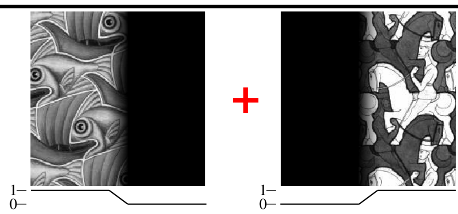


Image Blending

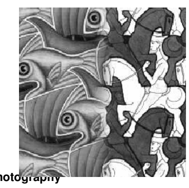


slide from Efros

Feathering



=



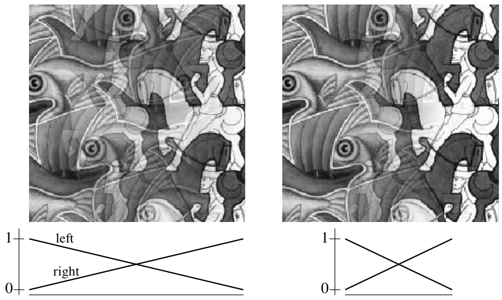
Encoding transparency

I(x,y) = (αR, αG, αB, α)

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

slide from Efros

Affect of Window Size

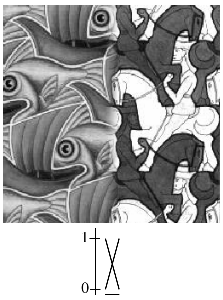


slide from Efros

Affect of Window Size

slide from Efros

Good Window Size



"Optimal" Window: smooth but not ghosted

slide from Efros

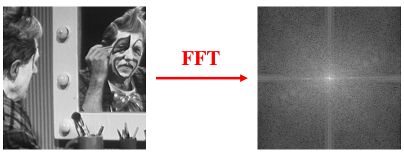
What is the Optimal Window?

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
- do blending in different frequency bands

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

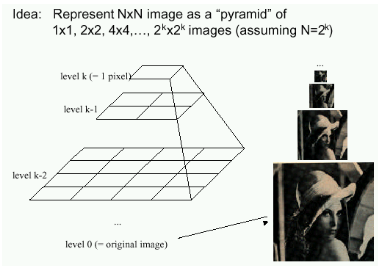
What does blurring take away?

smoothed (5x5 Gaussian)

High-Pass Filter

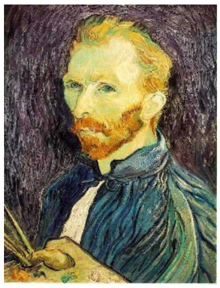
smoothed – original

Image Pyramids



mipmap or precursor of wavelets

Image Sub-sampling

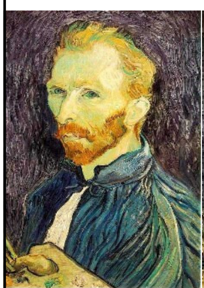


1/4

1/8

Throw away every other row and
column to create a 1/2 size image

Image Sub-sampling



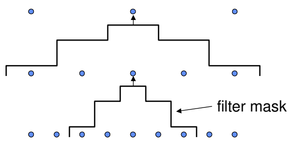
1/2

1/4 (2x zoom)

1/8 (4x zoom)

Why does this look so bad?

Gaussian Pyramid Construction



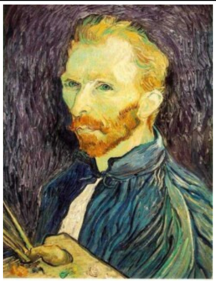
Repeat

- **Filter**
- **Subsample**

Until minimum resolution reached

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

Gaussian pre-filtering



G 1/4

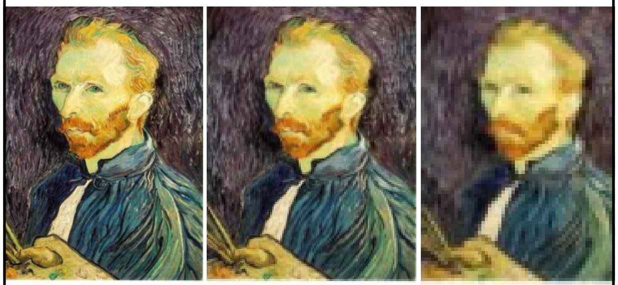
G 1/8

Gaussian 1/2

Solution: filter the image, *then* subsample

- **Filter size should double for each ½ size reduction.**

Subsampling with Gaussian pre-filtering

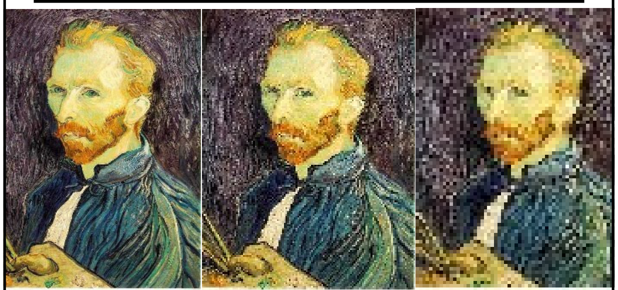


Gaussian 1/2 G 1/4 G 1/8

Solution: filter the image, *then* subsample

- **Filter size should double for each ½ size reduction.**

Compare with...

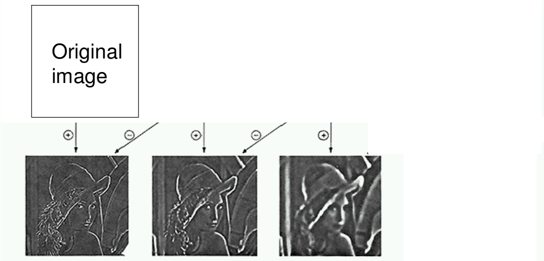


1/2 1/4 (2x zoom) 1/8 (4x zoom)

Band-pass filtering

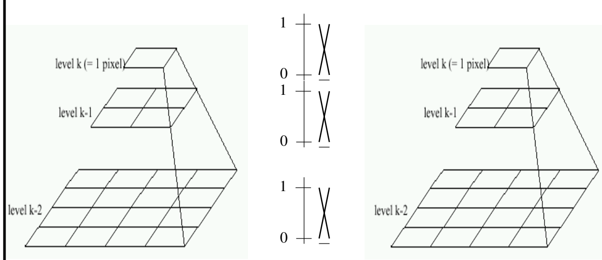
Gaussian Pyramid (low-pass images)

Laplacian Pyramid



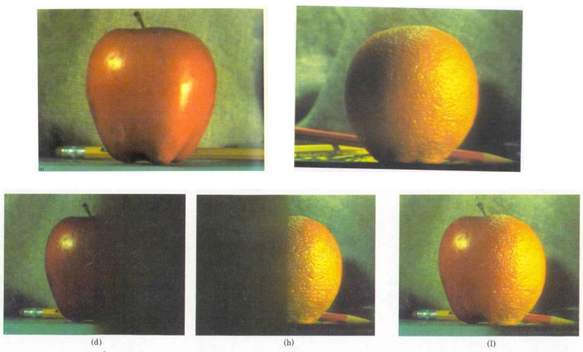
How can we reconstruct (collapse) this pyramid into the original image?

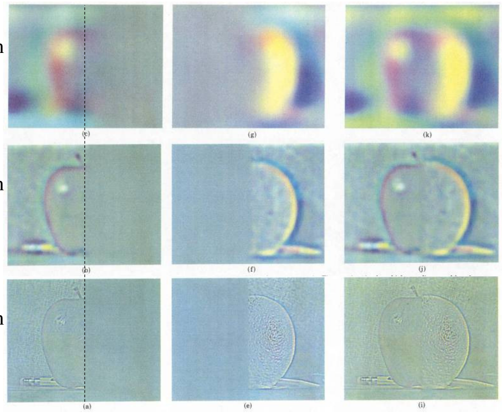
Pyramid Blending



Left pyramid blend Right pyramid

Pyramid Blending





Simplification: Two-band Blending

Brown & Lowe, 2003

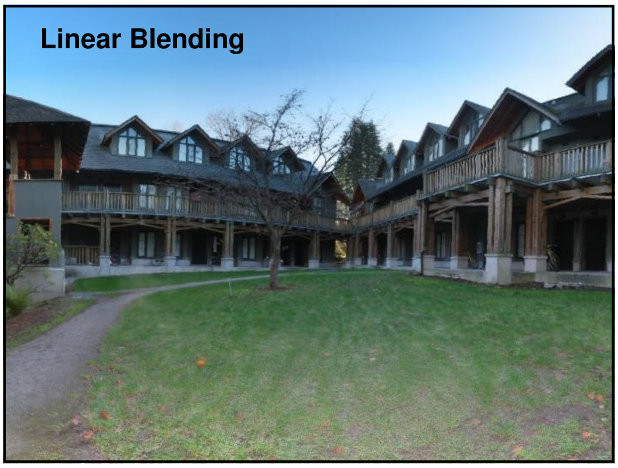
- Only use two bands: high freq. and low freq.
- Blends low freq. smoothly
- Blend high freq. with no smoothing: use binary mask

2-band Blending

Low frequency (λ > 2 pixels)

High frequency (λ < 2 pixels)

Linear Blending



2-band Blending

Still Some Artifacts Left…

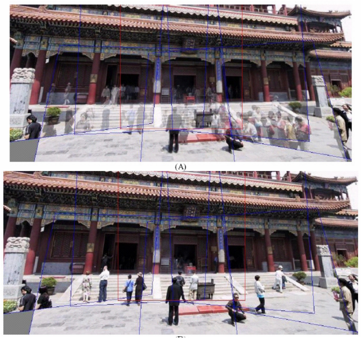
Ghosting—objects move in the scene.

Differing exposures between images.

- Pyramid blending does not solve this.

De-Ghosting

In regions with differences don't blend - crop.



[Uyttendaele et al. 2001]

Gradient Domain Blending

In Pyramid Blending, we decomposed our image into 2nd derivatives (Laplacian) and a low-res image

Let us now look at 1st derivatives (gradients):

No need for low-res image

- captures everything (up to a constant)
- easy to deal with low-frequency differences

Idea:

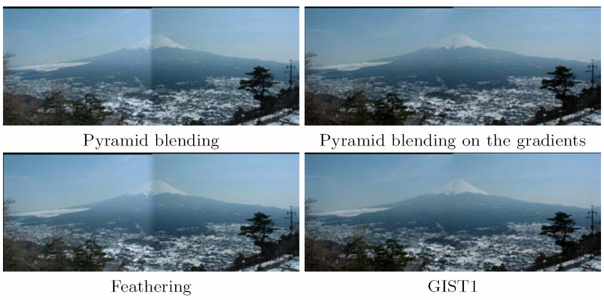
- Differentiate
- Blend
- Reintegrate

Gradient Domain Blending (2D)

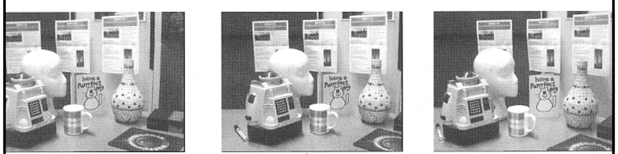
Trickier in 2D:

- Take partial derivatives dx and dy (the gradient field)
- Fiddle around with them (smooth, blend, feather, etc)
- Reintegrate
  - But now integral(dx) might not equal integral(dy)
- Find the most agreeable solution
  - Equivalent to solving Poisson equation
  - Can use FFT, deconvolution, multigrid solvers, etc.

Comparisons [Levin et al 2004]



Homography or not ?



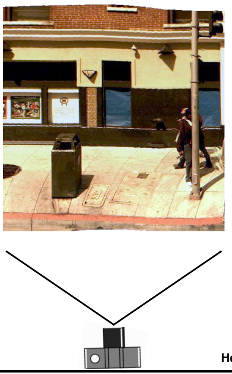
- Coincidences between 3D points at different depths are preserved
- Pure camera rotation about camera center

⇒ 2D Homography

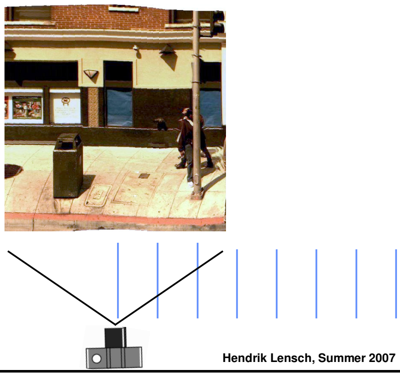
- Different depths are imaged to different image positions
- Camera rotates and translates

⇒ Motion Parallax, no Homography

Mulitperspective Panoramas



Mulitperspective Panoramas



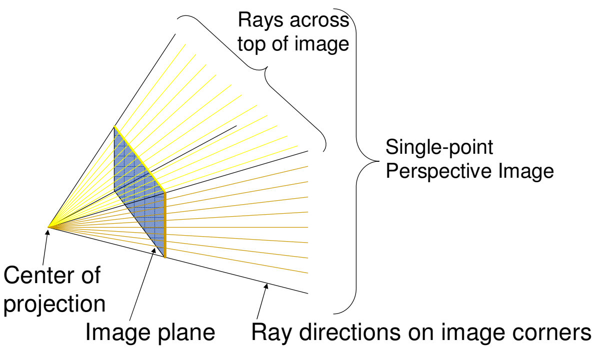
Aspect Ratio Distortion [Roman 2005]

Close and distant objects get distorted due to different perspective in x and y.

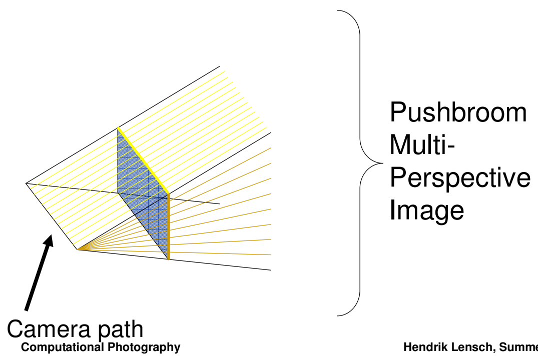
Aspect Ratio Distortion

Images with the original perspective don't suffer from this issue.
How to seamlessly combine multiple perspective images?

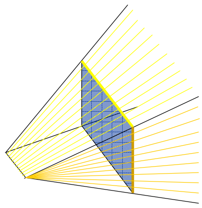
Background: Perspective Images



Background: Pushbroom Images

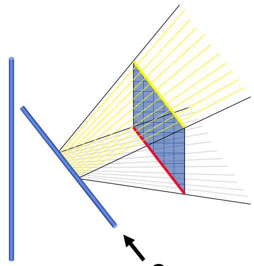


Cross-Slits Images



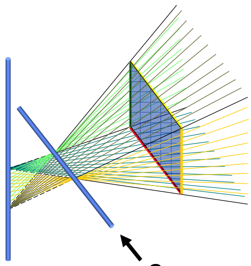
Cross-slits Multi-Perspective Image

Cross-Slits Images



Cross-slits Multi-Perspective Image

Cross-Slits Images



Cross-slits Multi-Perspective Image

Automatic Construction

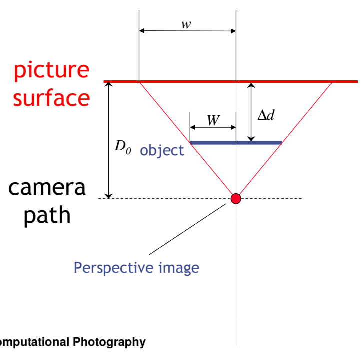
What causes the distortion?

- **difference between vertical and horizontal perspectives**
- **changes aspect ratio**

How can it be reduced?

- **quantify the distortion**
- **place picture surface**
- **select ray angles that minimize overall distortion**

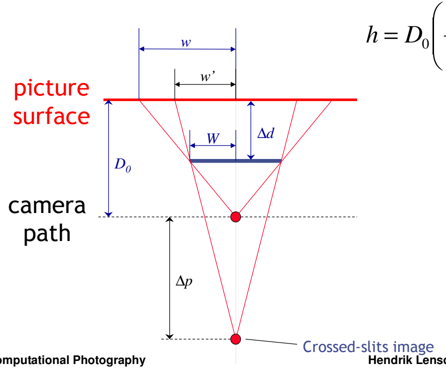
Aspect Ratio – Perspective



$$h = D_0\left(\frac{H}{D_0 - \Delta d}\right)$$

$$w = D_0\left(\frac{W}{D_0 - \Delta d}\right)$$

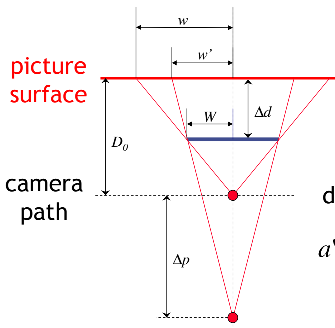
Aspect Ratio – Cross-slits



$$h = D_0\left(\frac{H}{D_0 - \Delta d}\right)$$

Aspect Ratio – Crossed-slits

[Zomet 03]



$$h = D_0\left(\frac{H}{D_0 - \Delta d}\right)$$

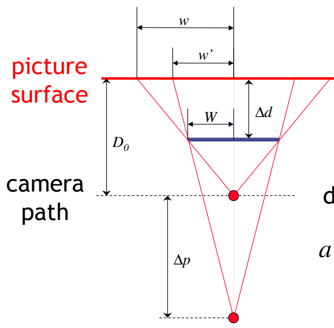
$$w' = \frac{W(D_0 + \Delta p)}{D_0 - \Delta d + \Delta p}$$

distortion:

$$a' = \frac{(D_0 + \Delta p)(D_0 - \Delta d)}{D_0(D_0 - \Delta d + \Delta p)}$$

Aspect Ratio – Crossed-slits

[Zomet 03]



$$h = D_0\left(\frac{H}{D_0 - \Delta d}\right)$$

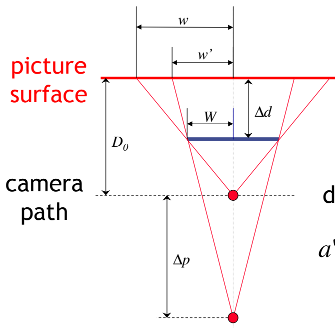
$$w' = \frac{W(D_0 + \Delta p)}{D_0 - \Delta d + \Delta p}$$

distortion:

$$a' = \frac{(D_0 + \Delta p)(D_0 - \Delta d)}{D_0(D_0 - \Delta d + \Delta p)}$$

Computational Photography
Hendrik Lensch, Summer 2007

Special Case: $\Delta p = 0$



$$h = D_0\left(\frac{H}{D_0 - \Delta d}\right)$$

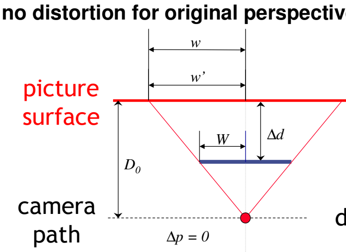
$$w' = \frac{W(D_0 + \Delta p)}{D_0 - \Delta d + \Delta p}$$

distortion:

$$a' = \frac{(D_0 + \Delta p)(D_0 - \Delta d)}{D_0(D_0 - \Delta d + \Delta p)}$$

Special Case: $\Delta p = 0$

no distortion for original perspective



$$h = D_0\left(\frac{H}{D_0 - \Delta d}\right)$$

$$w' = w = h$$

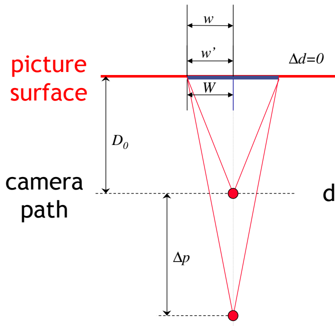
distortion:

$$a' = 1$$

Special Case: $\Delta d = 0$

no perspective effects at the picture surface

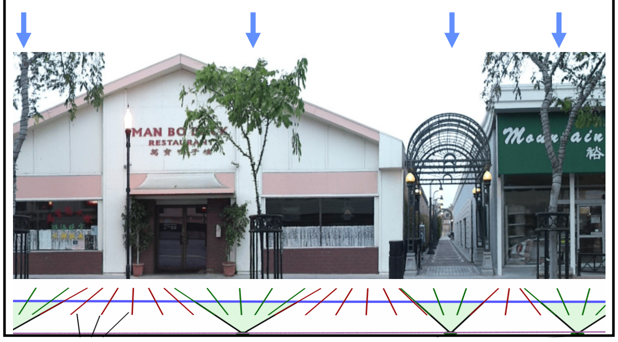
$$h = H$$



$$w' = W$$

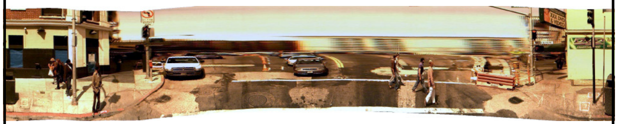
distortion:

$$a' = 1$$



Automatic Multiperspective Panormamas

pushbroom

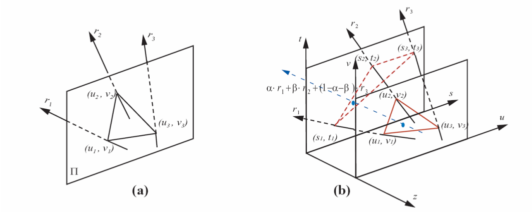


[Roman 2006]

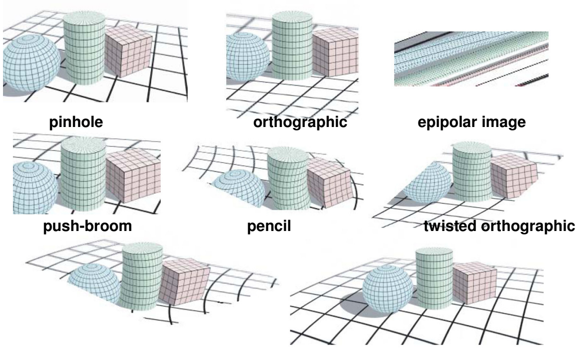
automatic perspective

General Linear Cameras

[Yu and McMillan 2004]

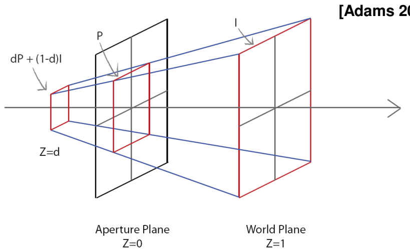


General Linear Cameras



General Linear Cameras - Classification

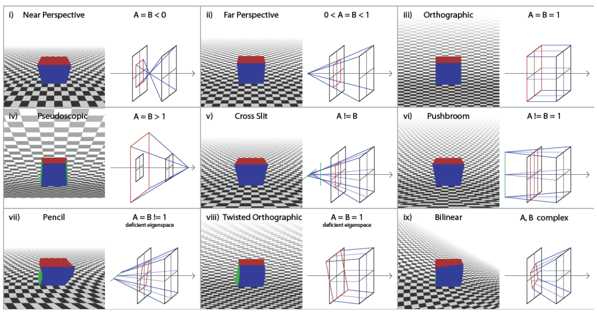
[Adams 2007]



- rays described by: $(P(x,y),(x,y))$
- characteristic matrix: $P'_d = (1-d)P + dI$

$$P''_\lambda = P + \lambda I \quad \text{with} \quad \lambda = \frac{d}{d-1}$$

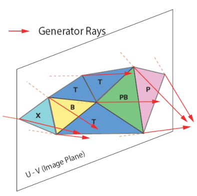
General Linear Cameras



analyze the eigenvalues of P

Multiperspective – Rendering Framework

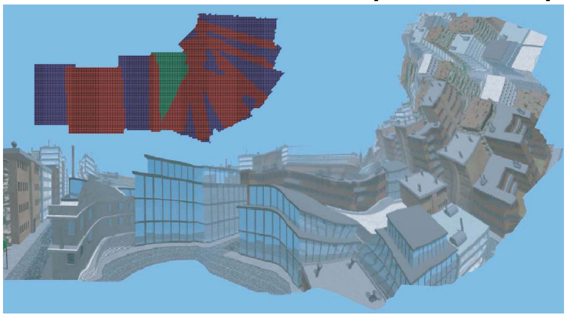
[Yu and McMillan 2004b]



- specify perspective per triangle
- blend between neighboring triangles

Multiperspective – Rendering Framework

[Yu and McMillan 2004b]



Panoramas

- A multiresolution spline with application to image mosaics
  P. J. Burt, E. H. Adelson.
  ACM Transactions on Graphics. *2(4)*, pp. 217-236, 1983.
- Recognising Panoramas. M. Brown and D. G. Lowe. In Proceedings of the *9th International Conference on Computer Vision (ICCV2003)*, pages 1218-1225, Nice, France, 2003.
- Seamless Image Stitching in the Gradient Domain. A. Levin, A. Zomet, S. Peleg and Y. Weiss, In Proc. ECCV 2004.
- Interactive Design of Multi-Perspective Images for Visualizing Urban Landscapes. Augusto Roman, Gaurav Garg, Marc Levoy. IEEE Visualization 2004.
- Automatic Multiperspective Images. Augusto Roman, Hendrik Lensch, In Proc. EGSR 2006, pages 161-171.
- Multiview Radial Catadioptric Imaging for Scene Capture. S. Kuthirummal, S. Nayar, ACM TOG (Proc. SIGGRAPH), pages 916-923, 2006.

General Linear Cameras

- General Linear Cameras. J. Yu and L. McMillan. In Proc. ECCV 2004, pages 14-27.
- A Framework for Multiperspective Rendering, J. Yu and L. McMillan. In Proc. EGSR 2004, pages 61-68.
- General Linear Cameras with Finite Aperture. A. Adams and M. Levoy, In Proc. EGSR 2007.