6.098 Digital and Computational Photography
6.882 Advanced Computational Photography

Panoramas

Bill Freeman

Frédo Durand

MIT - EECS

Lots of slides stolen from Alyosha Efros, who stole them from Steve Seitz and Rick Szeliski

Olivier Gondry

- **Director of music video and commercial**
- **Special effect specialist (Morphing, rotoscoping)**
- **Today at 5:40pm in 32-141**

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

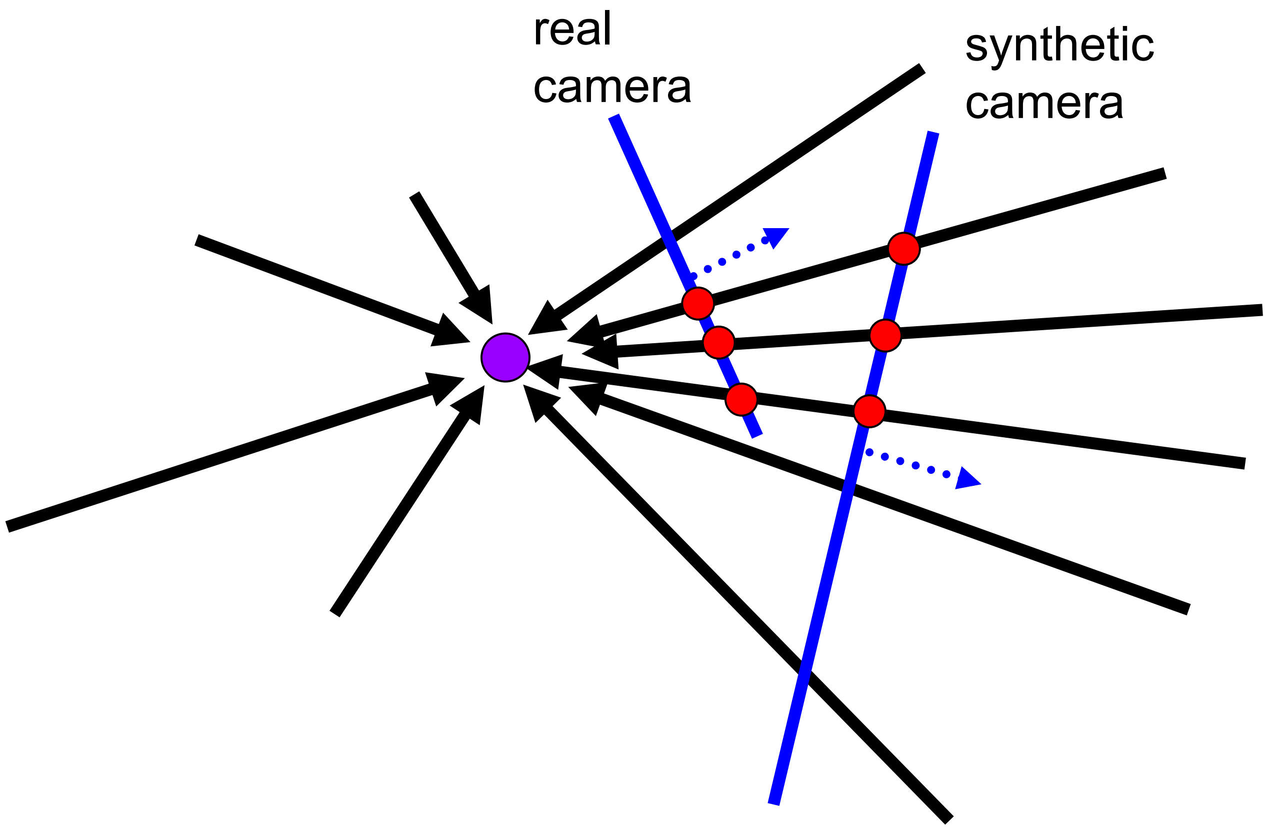
Mosaics: stitching images together

virtual wide-angle camera

How to do it?

- **Basic Procedure**
  - 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
- **…but** wait**, why should this work at all?**
  - What about the 3D geometry of the scene?
  - Why aren't we using it?

A pencil of rays contains all views



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

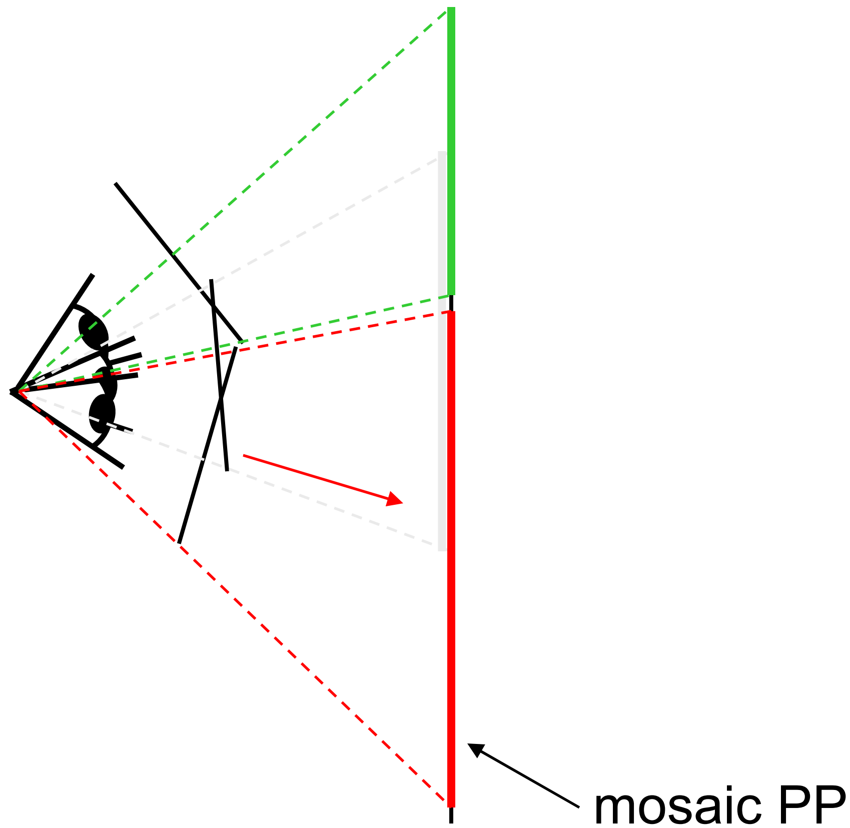
Aligning images: translation

left on top

right on top

Translations are not enough to align the images

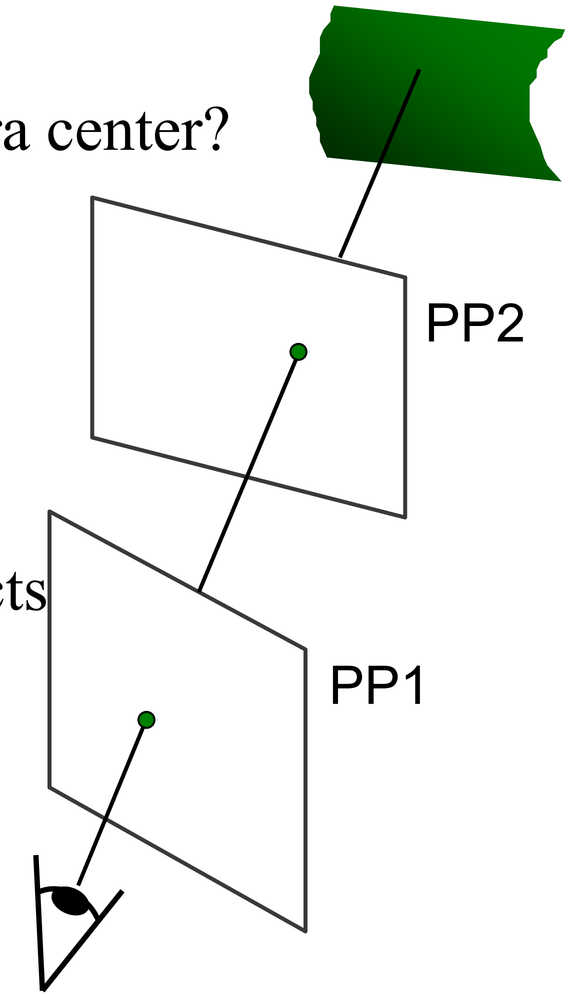
Image reprojection



- **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*

Image reprojection

- **Basic question**
  - How to relate 2 images from same camera center?
    - how to map a pixel from PP1 to PP2



- **Answer**
  - Cast a ray through each pixel in PP1
  - Draw the pixel where that ray intersects PP2

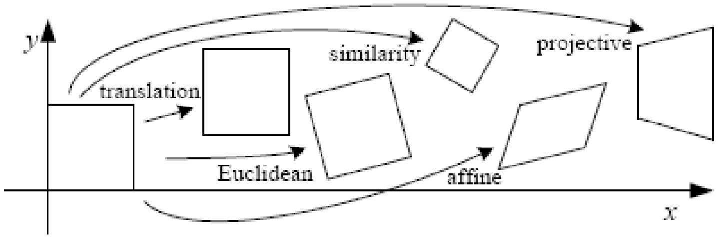
But don't we need to know the geometry of the two planes in respect to the eye?

Observation:
Rather than thinking of this as a 3D reprojection, think of it as a 2D **image warp** from one image to another

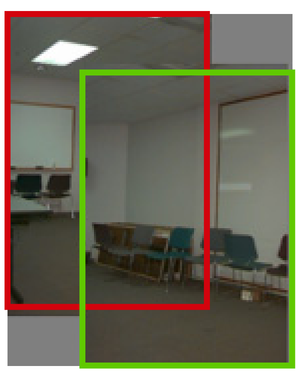
Back to Image Warping

Which t-form is the right one for warping PP1 into PP2?

e.g. translation, Euclidean, affine, projective

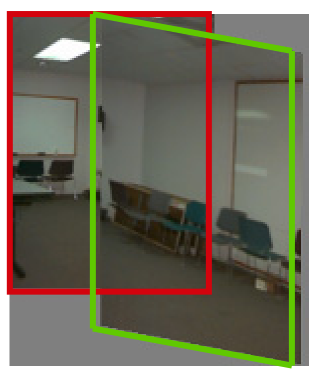


Translation



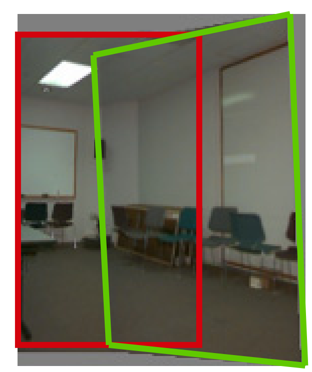
2 unknowns

Affine



6 unknowns

Perspective

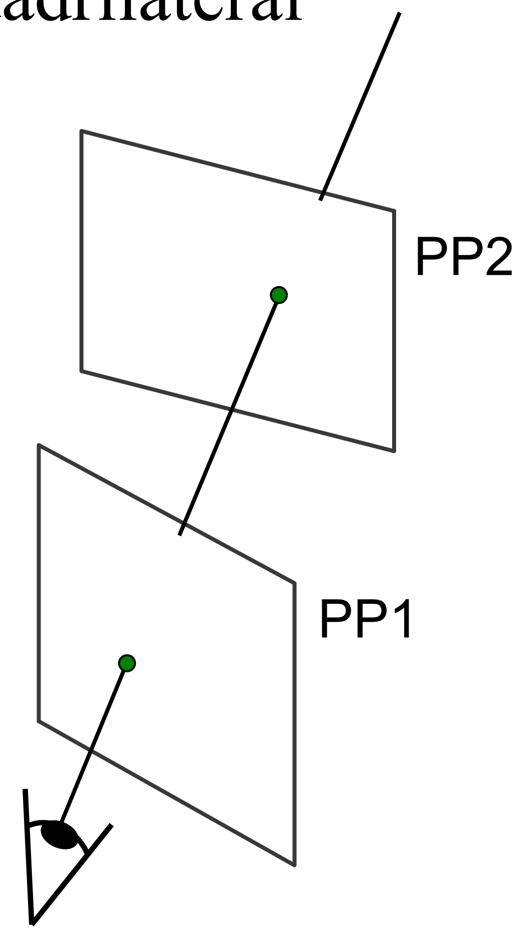


8 unknowns

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**

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

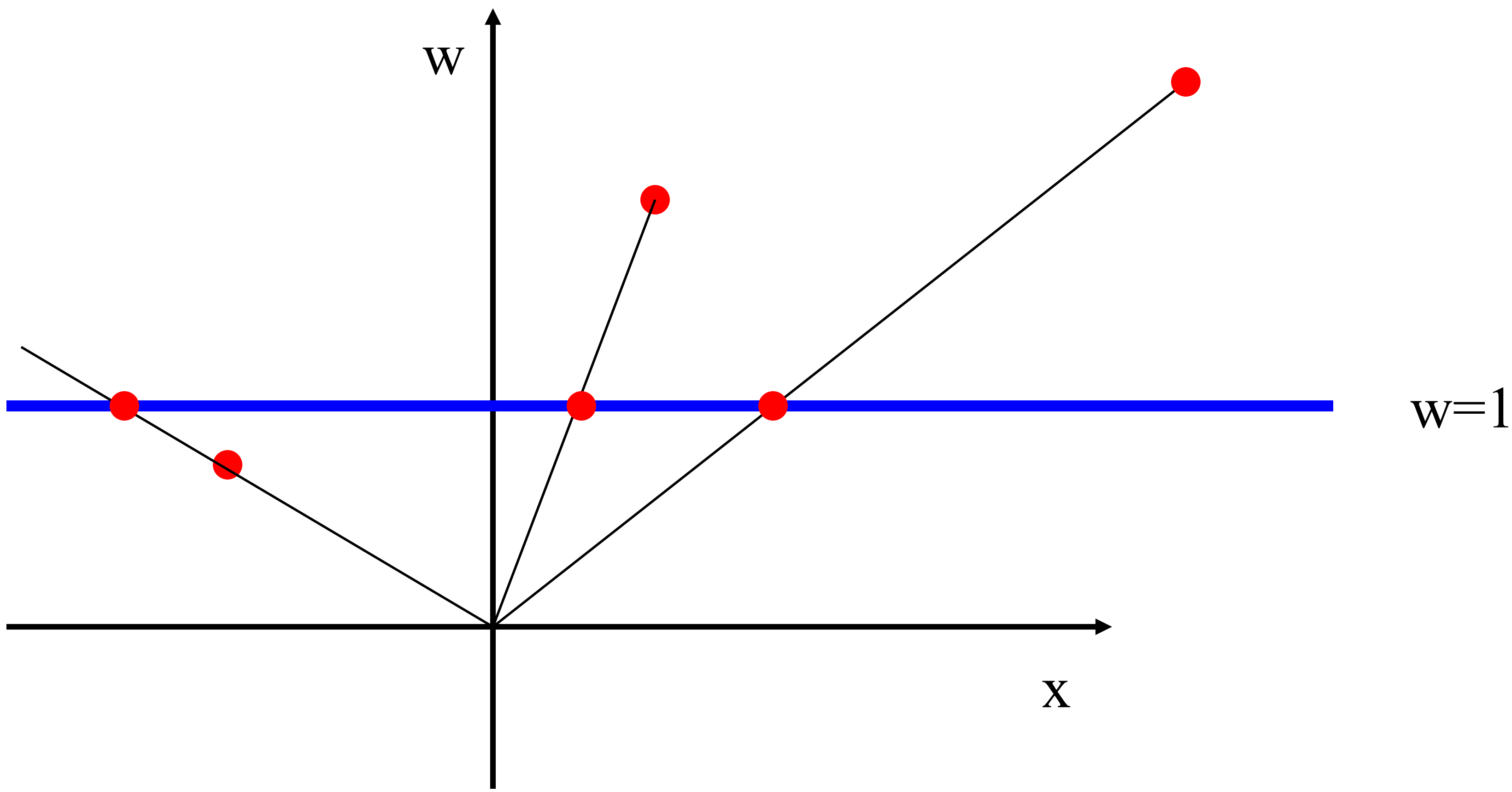


To apply a homography **H**

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

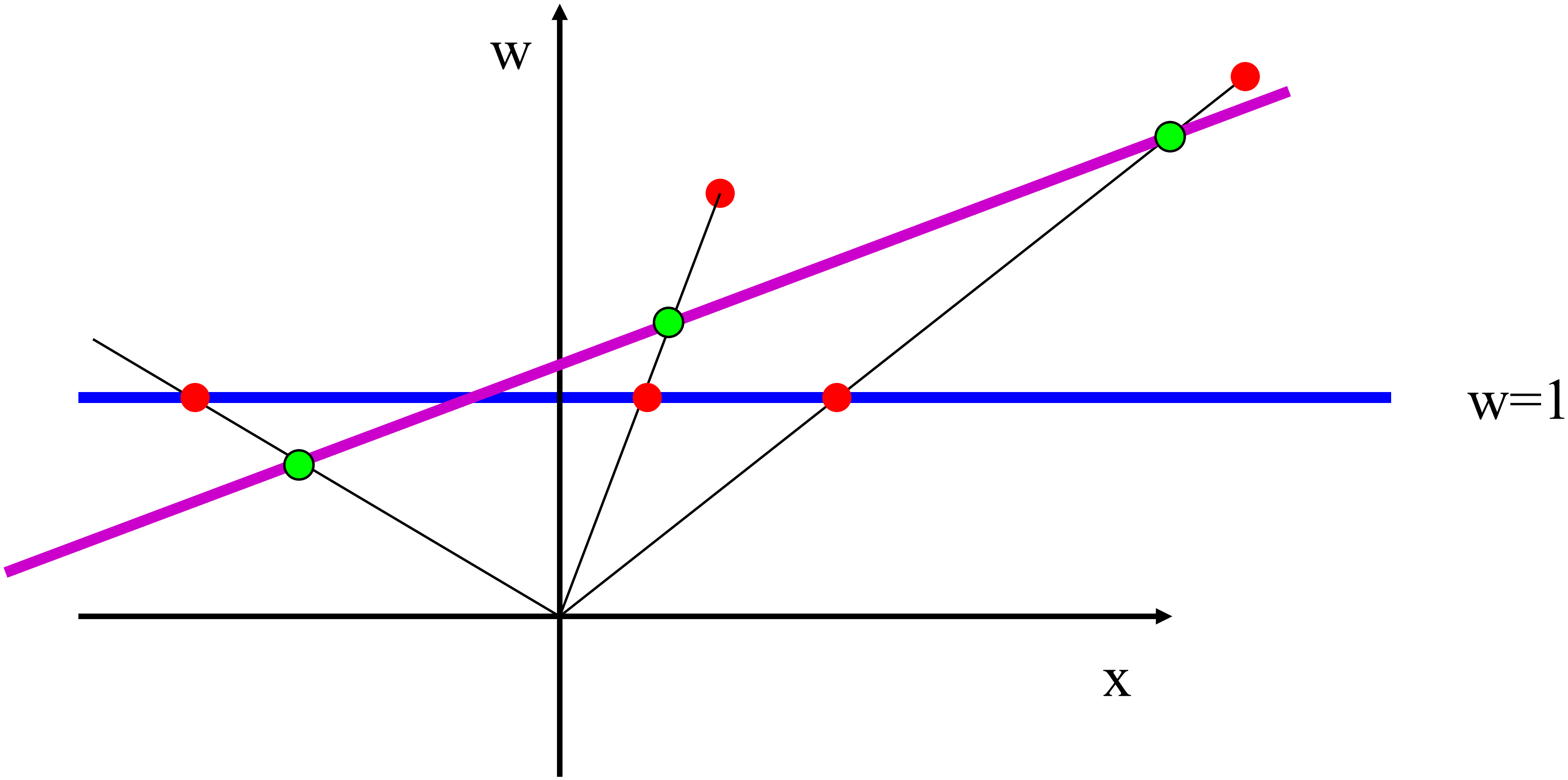
1D homogeneous coordinates

- **Add one dimension to make life simpler**
- **(x, w) represent point x/w**



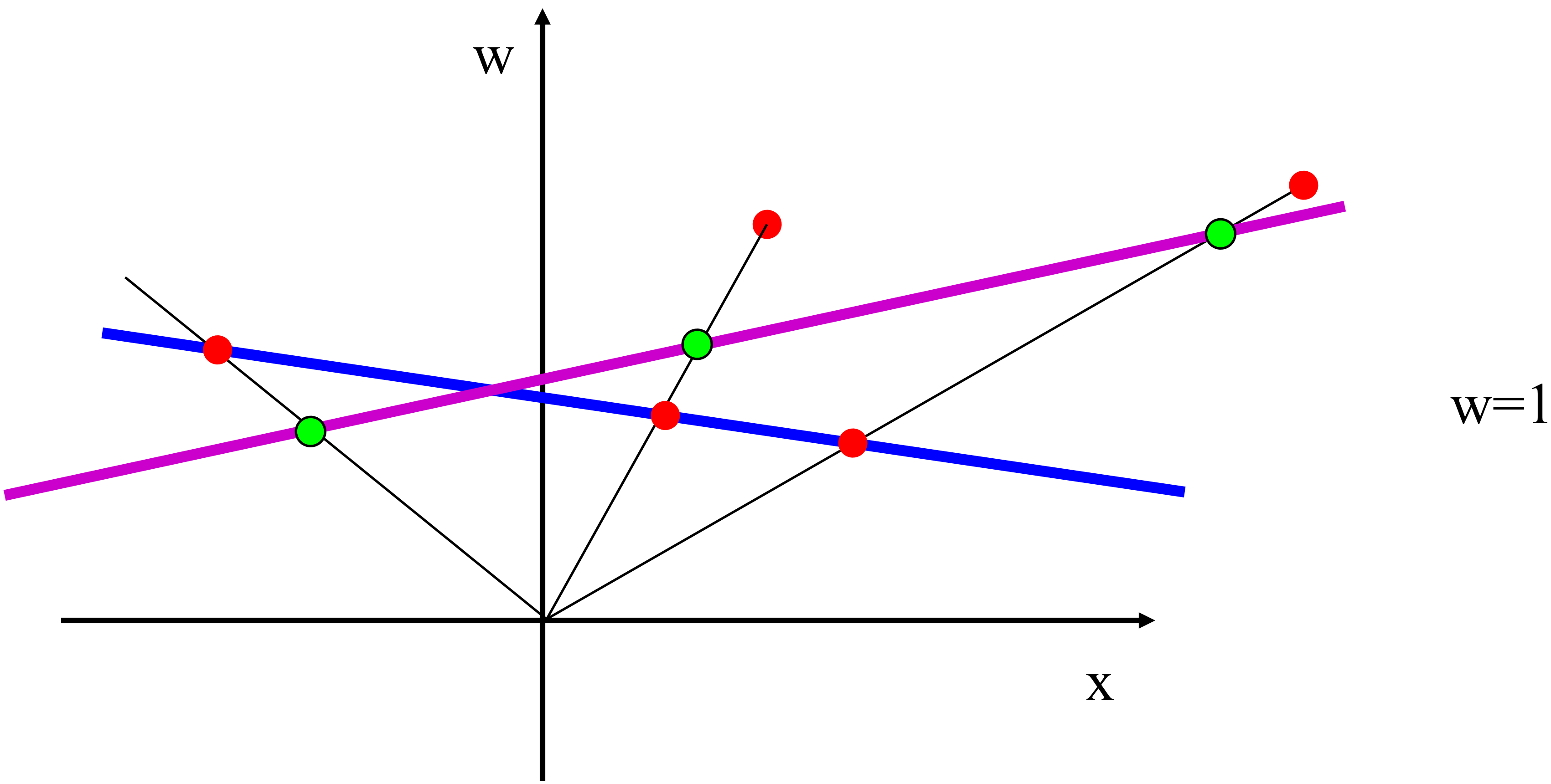
1D homography

- **Reproject to different line**



1D homography

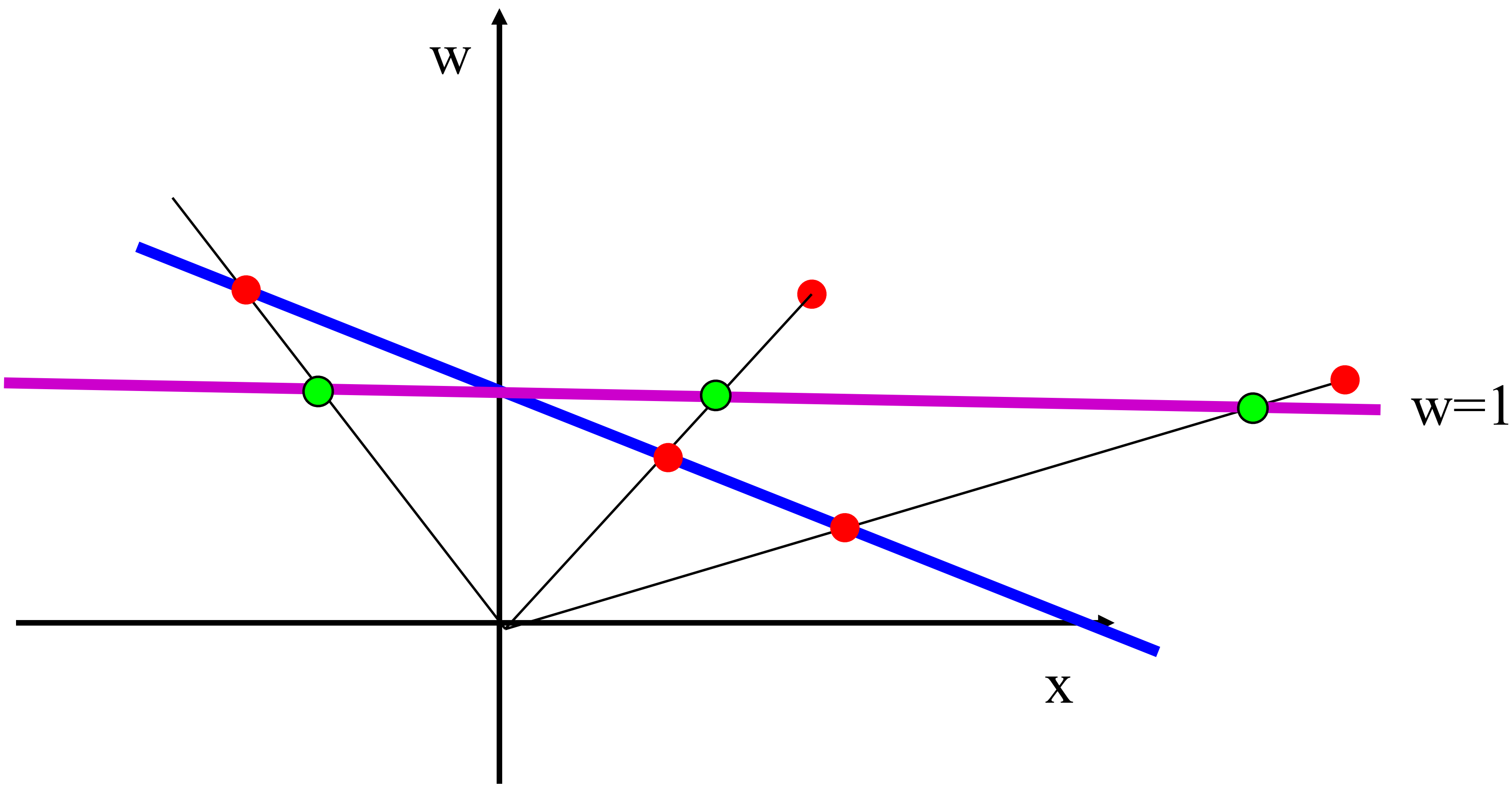
- **Reproject to different line**



1D homography

- **Reproject to different line**
- **Equivalent to rotating 2D points**

➔ reprojection is linear in homogeneous coordinates



Same in 2D

- **Reprojection = homography**
- **3x3 matrix**

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

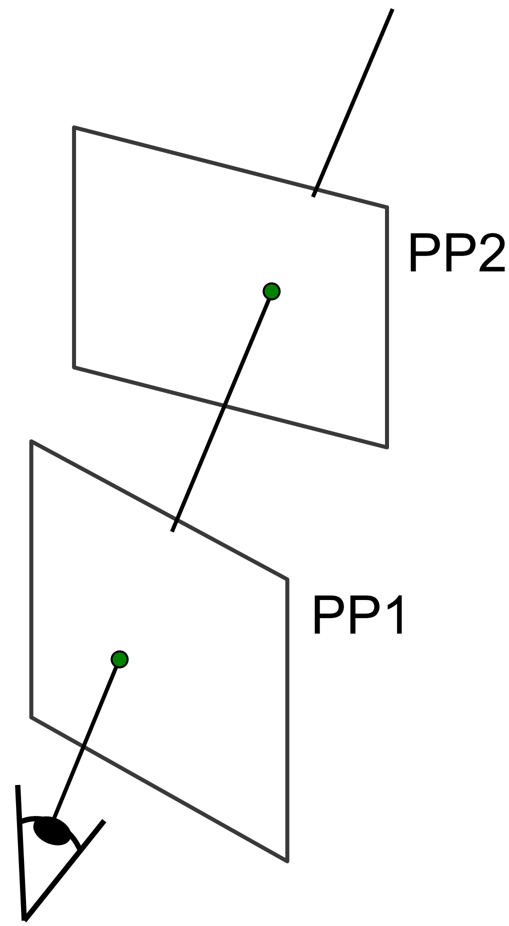
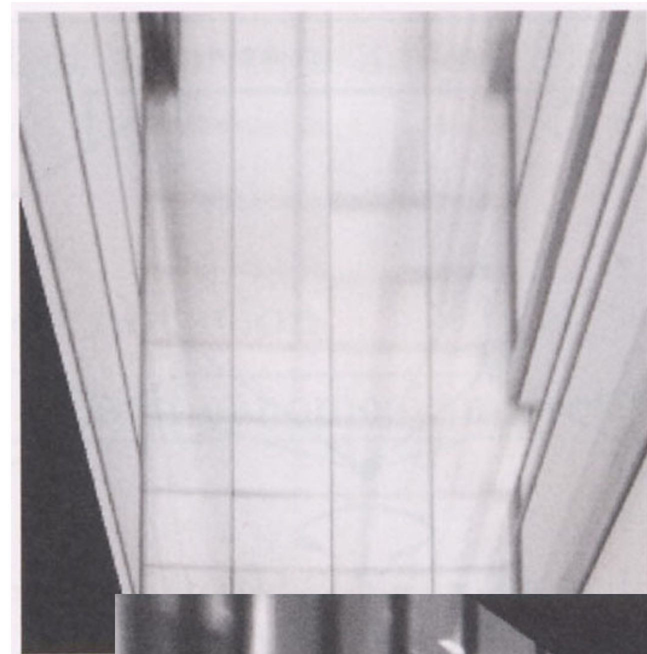


Image warping with homographies



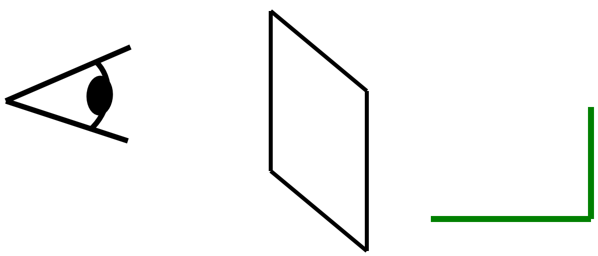
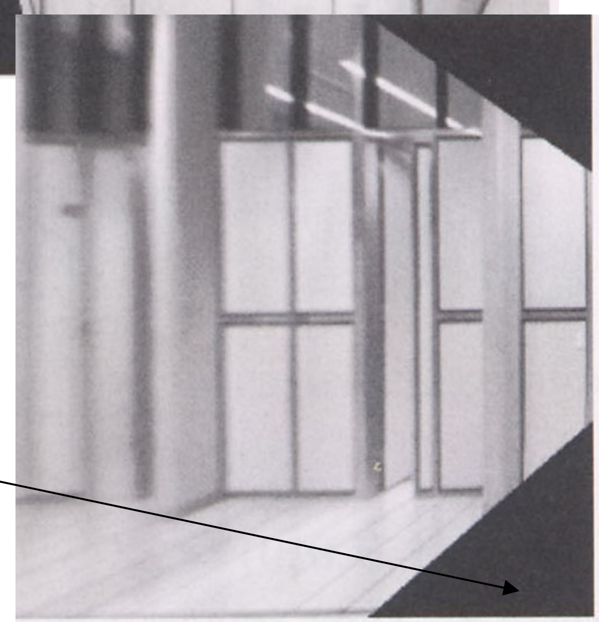
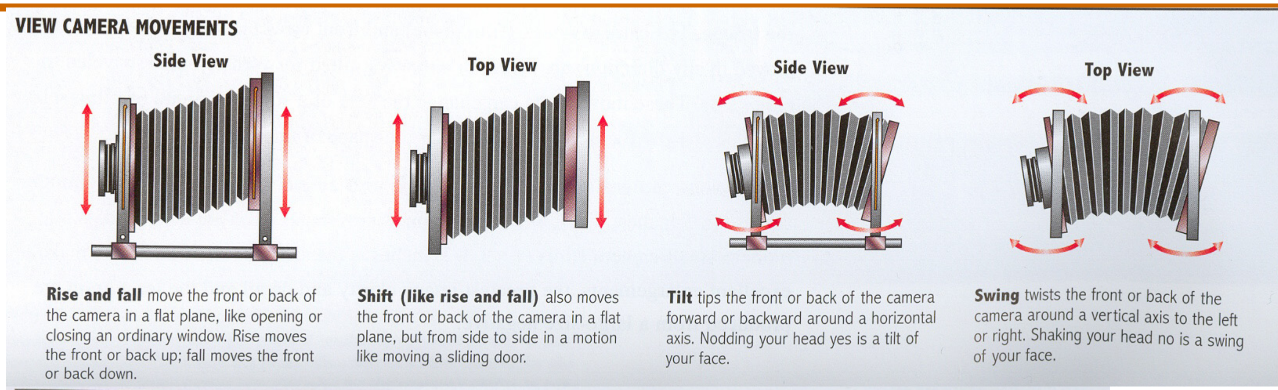


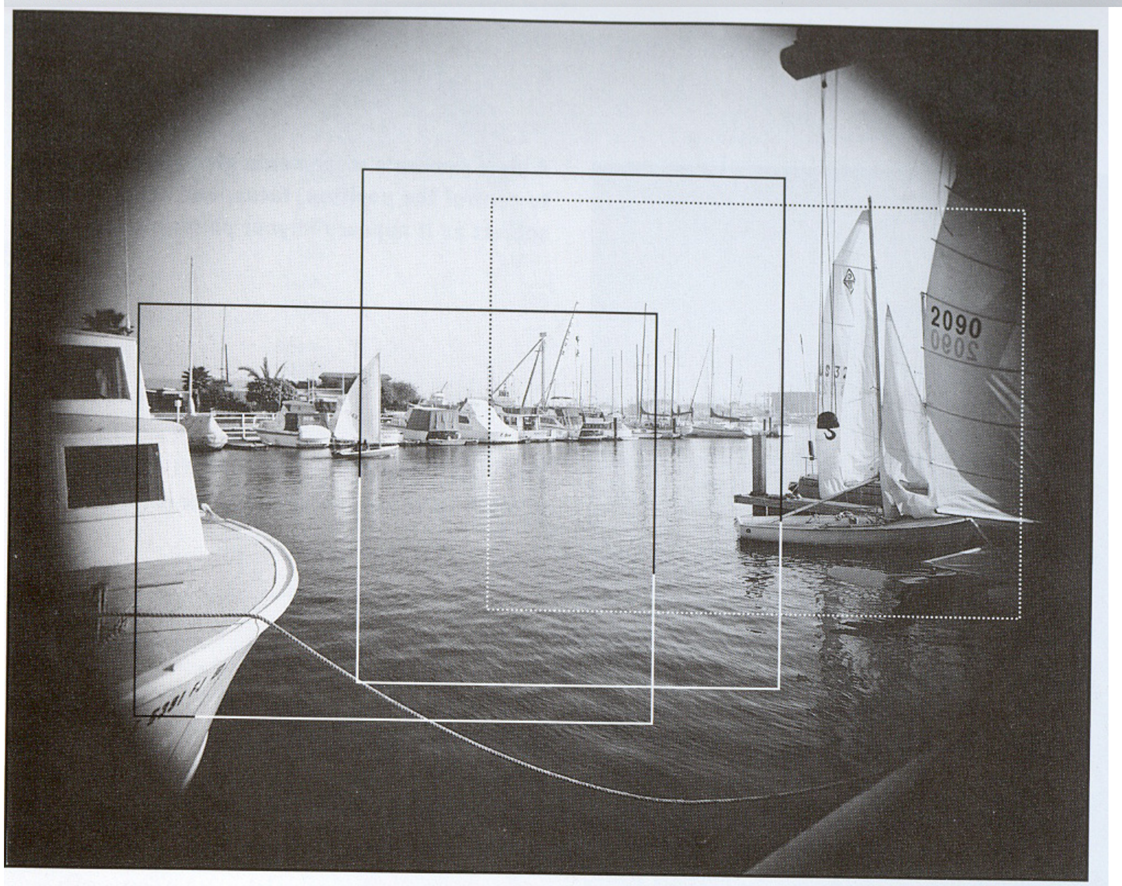
image plane in front

black area
where no pixel
maps to



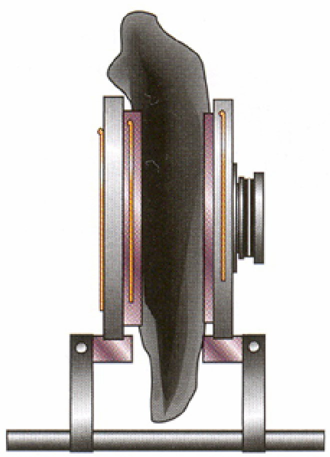
Digression: perspective correction





From Photography, London et al.

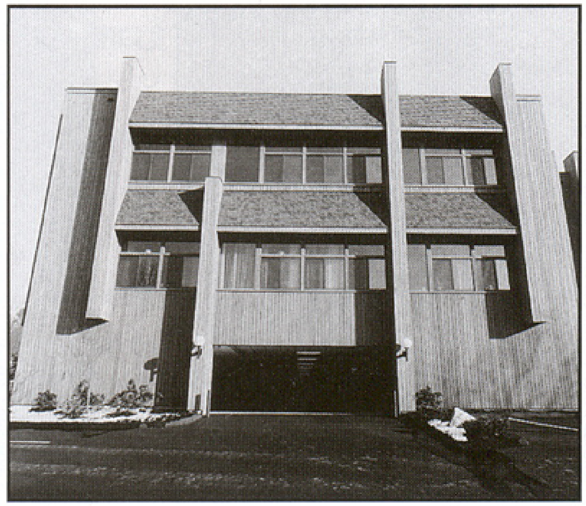
CONTROLLING CONVERGING LINES: THE KEY

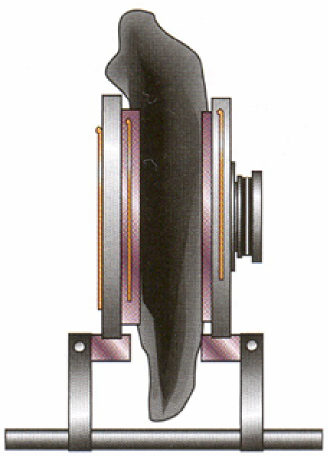


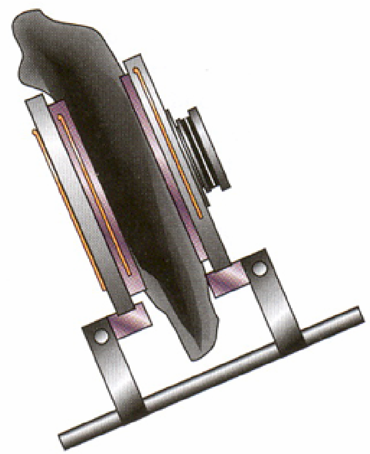
Standing at street level and shooting straight at a building produces too much street and too little building. Sometimes it is possible to move back far enough to show the entire building while keeping the camera level, but this adds even more foreground and usually something gets in the way.

From Photography, London et al.

CONTROLLING CONVERGING LINES: THE KEYSTONE EFFECT





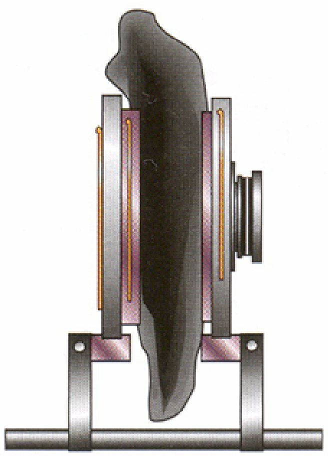


Standing at street level and shooting straight at a building produces too much street and too little building. Sometimes it is possible to move back far enough to show the entire building while keeping the camera level, but this adds even more foreground and usually something gets in the way.

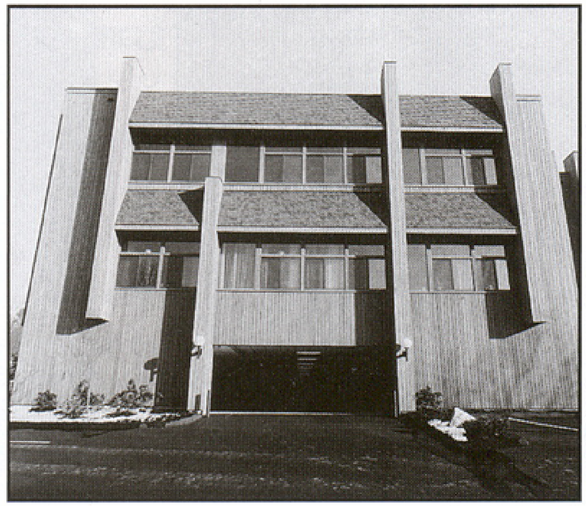
Tilting the whole camera up shows the entire building but distorts its shape. Since the top is farther from the camera than the bottom, it appears smaller; the vertical lines of the building seem to be coming closer together, or converging, near the top. This is named the keystone effect, after the wedge-shaped stone at the top of an arch. This convergence gives the illusion that the building is falling backward—an effect particularly noticeable when only one side of the building is visible.

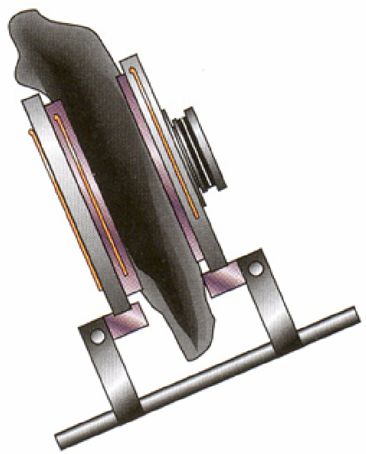
From Photography, London et al.

CONTROLLING CONVERGING LINES: THE KEYSTONE EFFECT

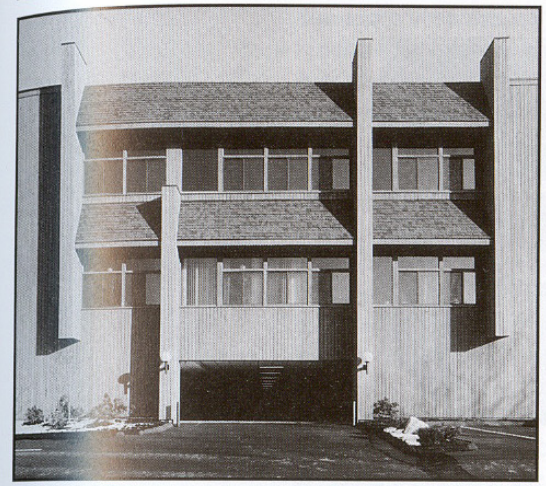


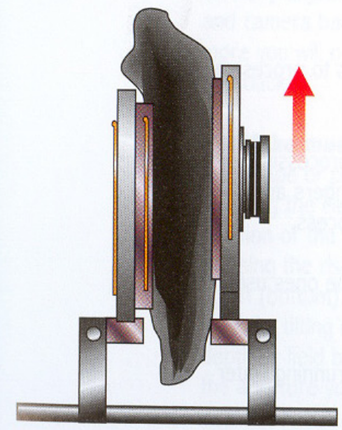
Standing at street level and shooting straight at a building produces too much street and too little building. Sometimes it is possible to move back far enough to show the entire building while keeping the camera level, but this adds even more foreground and usually something gets in the way.





Tilting the whole camera up shows the entire building but distorts its shape. Since the top is farther from the camera than the bottom, it appears smaller; the vertical lines of the building seem to be coming closer together, or converging, near the top. This is named the keystone effect, after the wedge-shaped stone at the top of an arch. This convergence gives the illusion that the building is falling backward—an effect particularly noticeable when only one side of the building is visible.





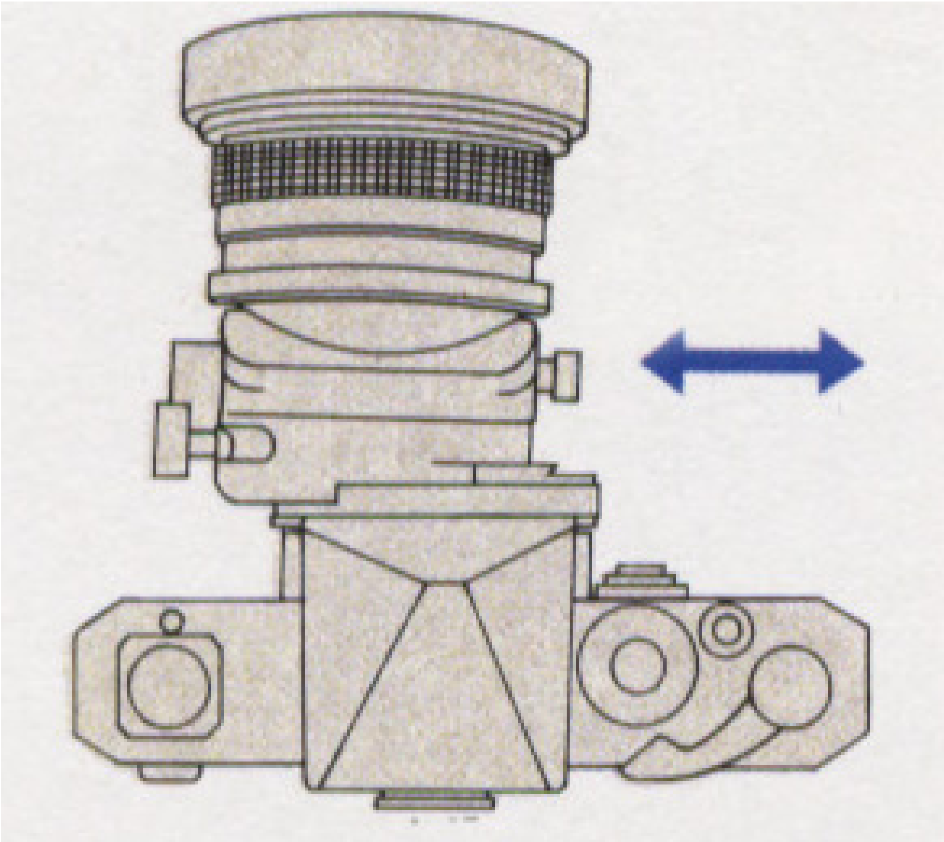
To straighten up the converging vertical lines, keep the camera back parallel to the face of the building. To keep the face of the building in focus, make sure the lens is parallel to the camera back. One way to do this is to level the camera and then use the rising front or falling back movements or both.

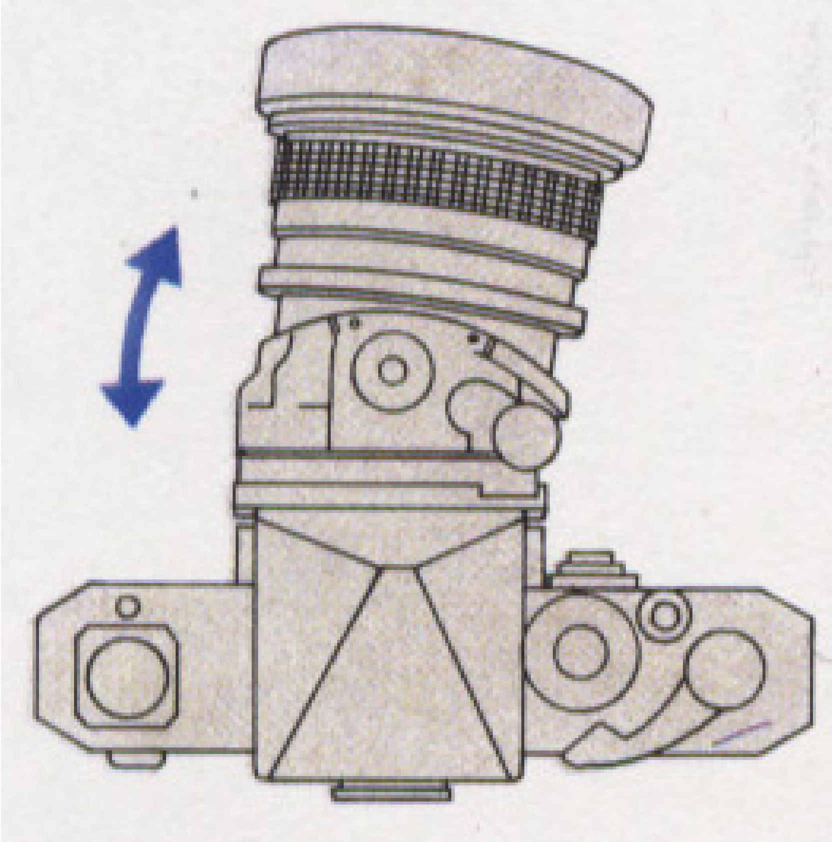
Another solution is to point the camera upward toward the top of the building, then use the tilting movements—first to tilt the back to a vertical position (which squares the shape of the building), then to tilt the lens so it is parallel to the camera back (which brings the face of the building into focus). The lens and film will end up in the same positions with both methods.

From Photography, London et al.

Tilt-shift lens

- **35mm SLR version**

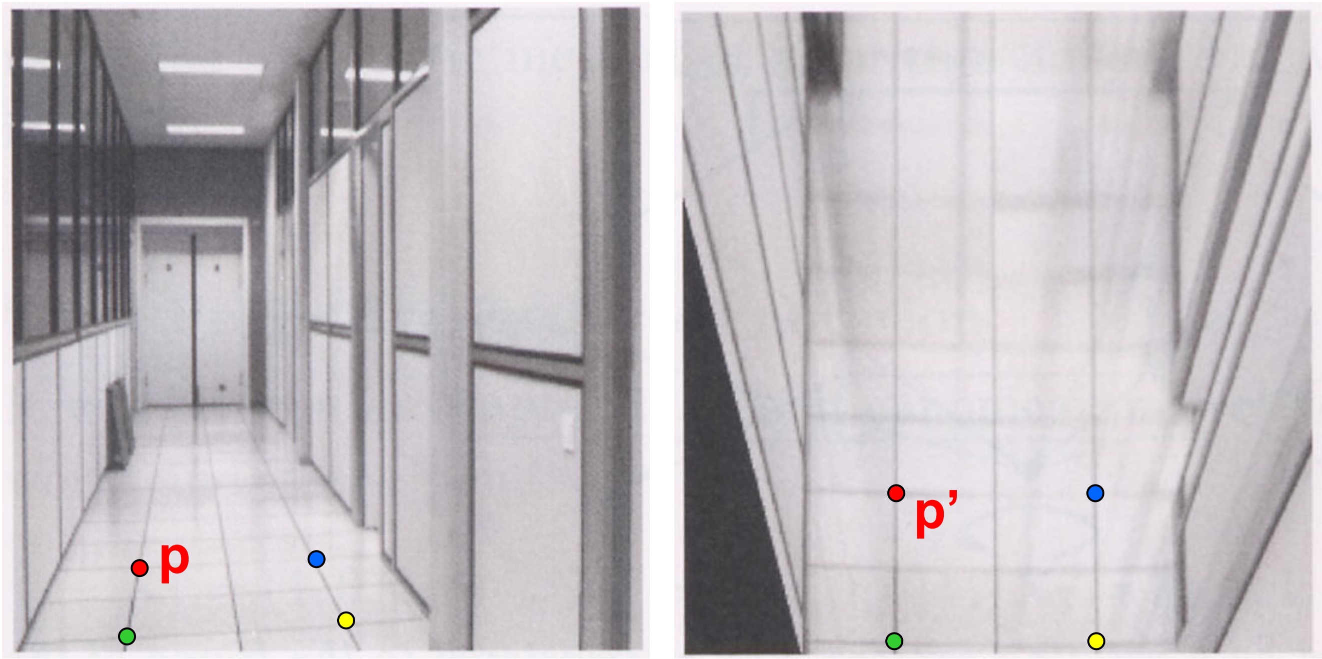




Photoshop version (perspective crop)

+ you control reflection and perspective independently

Back to Image rectification



To unwarp (rectify) an image

- Find the homography **H** given a set of **p** and **p'** pairs
- How many correspondences are needed?
- Tricky to write H analytically, but we can <u>solve</u> for it!
  - Find such H that "best" transforms points p into p'
  - Use least-squares!

Least Squares Example

- Say we have a set of data points (X1,X1'), (X2,X2'), (X3,X3'), etc. (e.g. person's height vs. weight)
- We want a nice compact formula (line) to predict X's from Xs:

  Xa + b = X'

- We want to find a and b
- How many (X,X') pairs do we need?
- $$X_1 a + b = X_1' \qquad X_2 a + b = X_2'$$

$$\begin{bmatrix} X_1 & 1 \\ X_2 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} X_1' \\ X_2' \end{bmatrix}$$

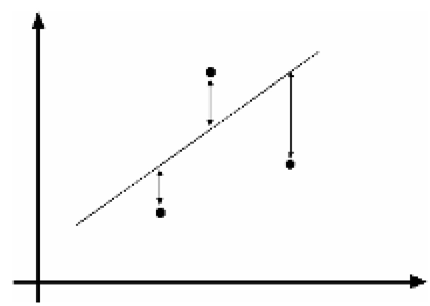
Ax=B

- What if the data is noisy?

$$\begin{bmatrix} X_1 & 1 \\ X_2 & 1 \\ X_3 & 1 \\ \dots & \dots \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} X_1' \\ X_2' \\ X_3' \\ \dots \end{bmatrix}$$

overconstrained

$$\min \|Ax - B\|^2$$



Solving for homographies

p’ = Hp

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- **Can set scale factor *i*=1. So, there are 8 unkowns.**
- **Set up a system of linear equations:**
  - Ah = b
- **where vector of unknowns h = $[a,b,c,d,e,f,g,h]^T$**

- **Note: we do not know w but we can compute it from x & y w=gx+hy+1**
- **The equations are linear in the unknown**

Solving for homographies

p' = Hp

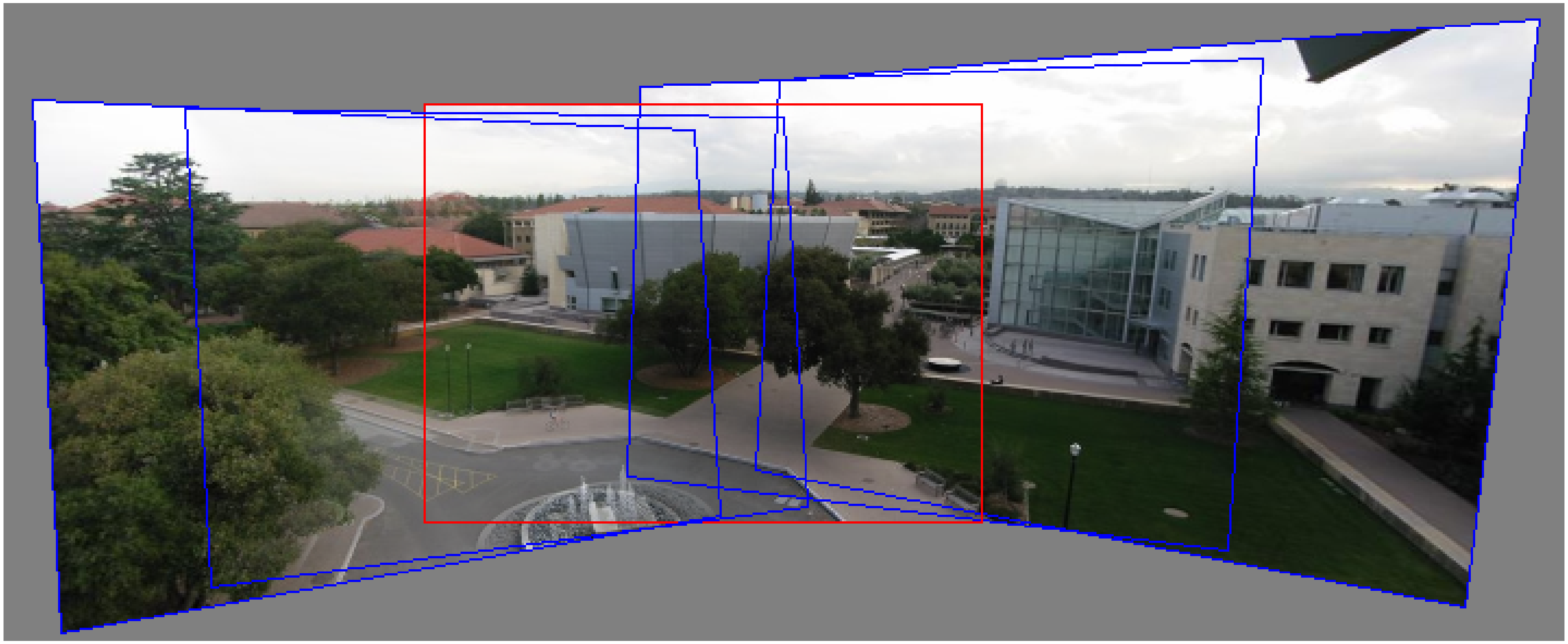
$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- **Can set scale factor *i*=1. So, there are 8 unkowns.**
- **Set up a system of linear equations:**
  - Ah = b
- **where vector of unknowns h = $[a,b,c,d,e,f,g,h]^T$**
- **Need at least 8 eqs, but the more the better…**
- **Solve for h. If overconstrained, solve using least-squares:**

$$\min \|Ah - b\|^2$$

- **Can be done in Matlab using "\" command**
  - see "help lmdivide"

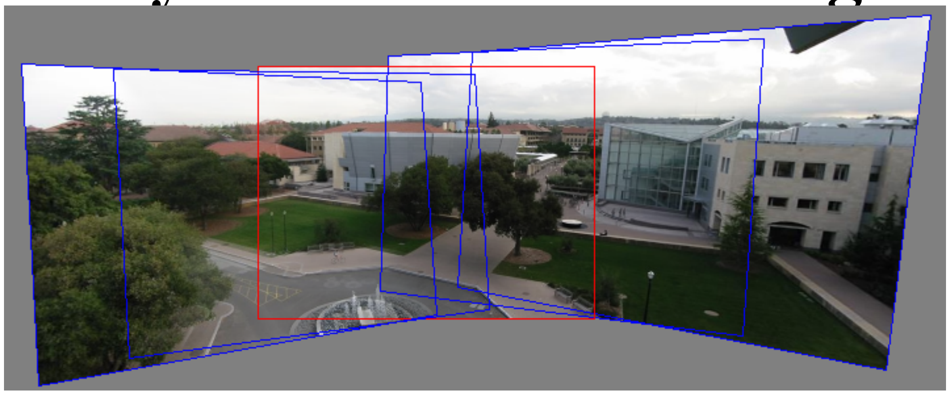
Panoramas



1. Pick one image (red)
2. Warp the other images towards it (usually, one by one)
3. blend

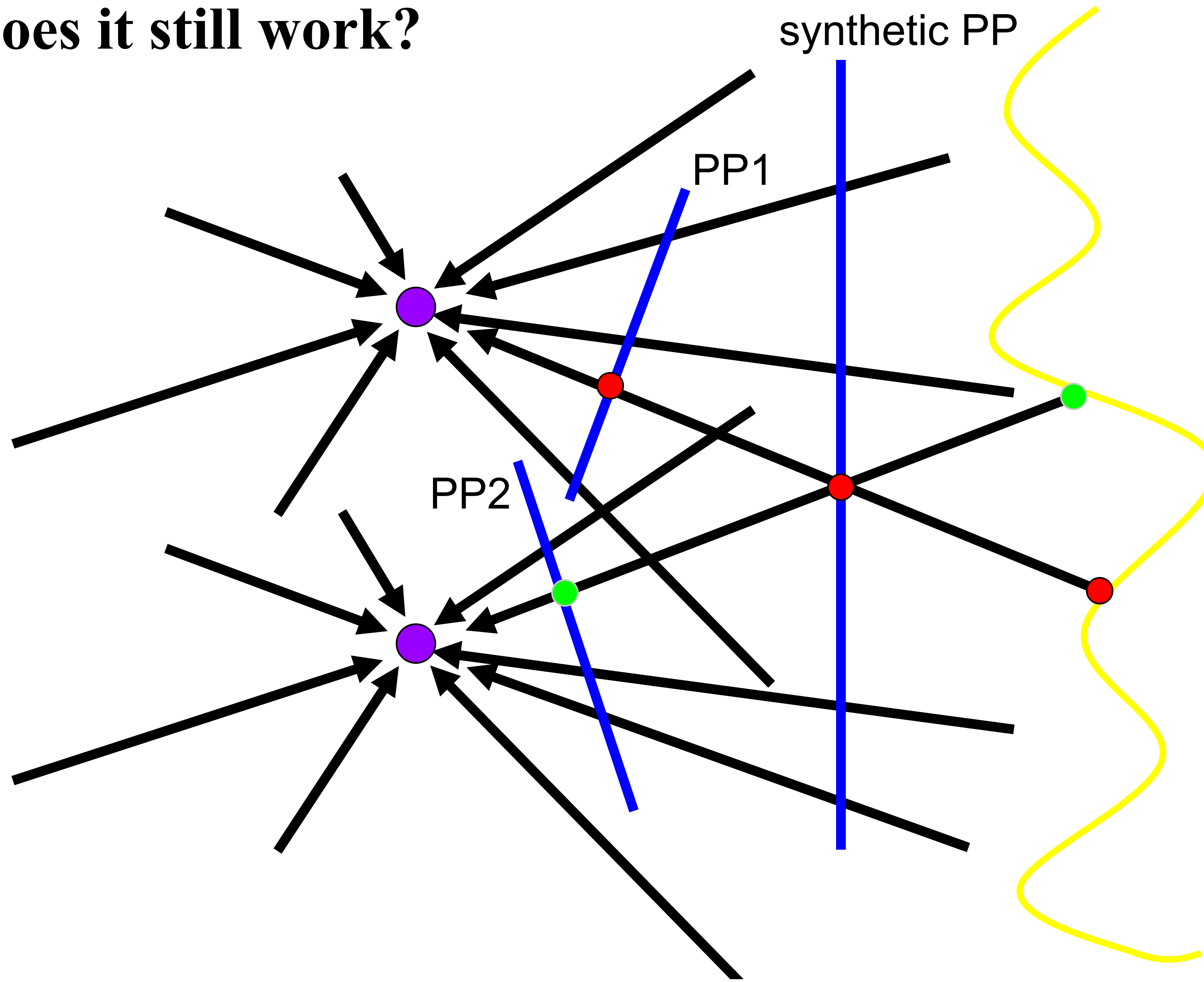
Recap

- **Panorama = reprojection**
- **3D rotation → homography**
  - Homogeneous coordinates are kewl
- **Use feature correspondence**
- **Solve least square problem**
  - Se of linear equations
- **Warp all images to a reference one**
- **Use your favorite blending**



changing camera center

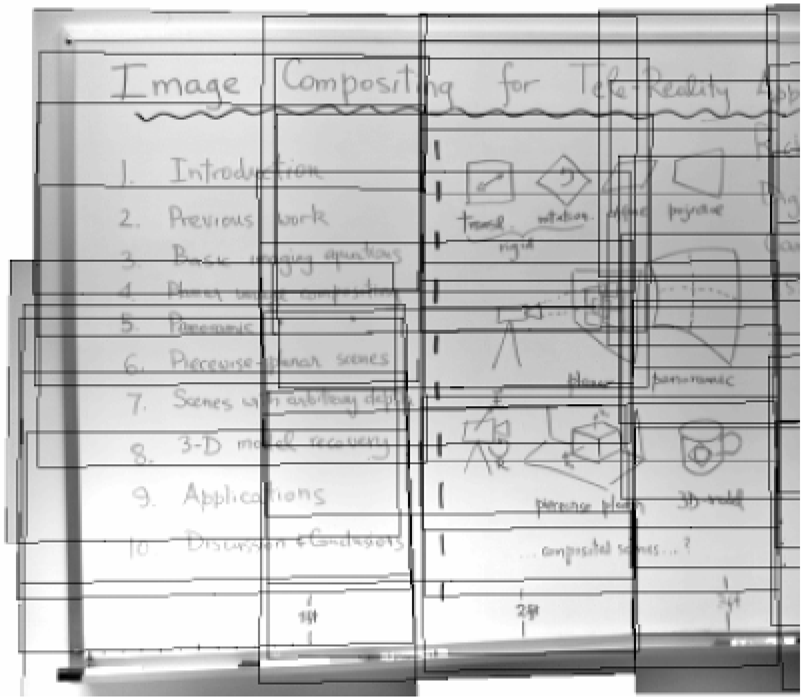
- **Does it still work?**

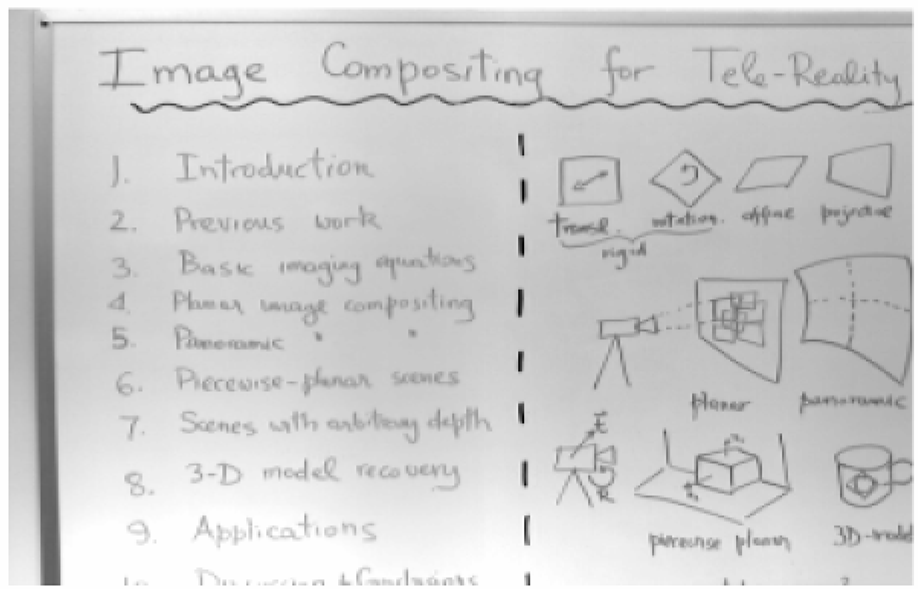


Nodal point

- **http://www.reallyrightstuff.com/pano/index.html**

Planar mosaic





Cool applications of homographies

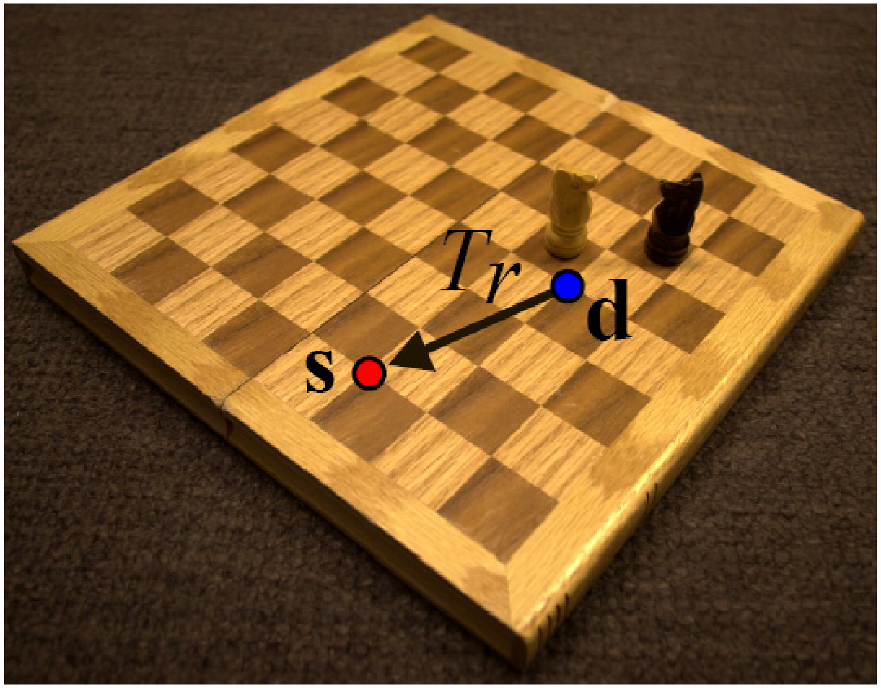
- **Oh, Durand & Dorsey**

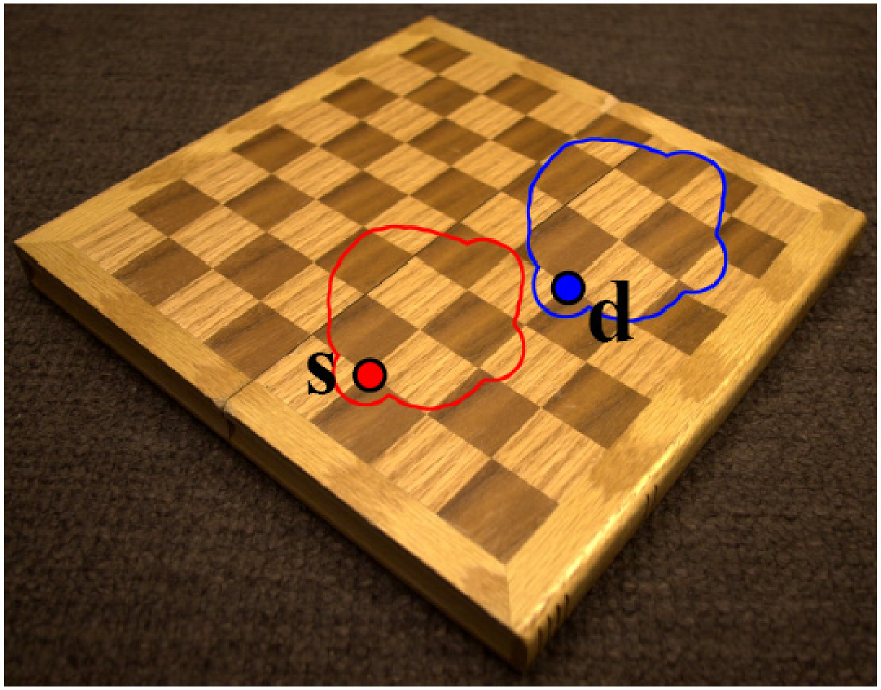
Limitations of 2D Clone Brushing

- **Distortions due to foreshortening and surface orientation**

Clone brush (Photoshop)

- Click on a reference pixel (blue)
- Then start painting somewhere else
- Copy pixel color with a translation

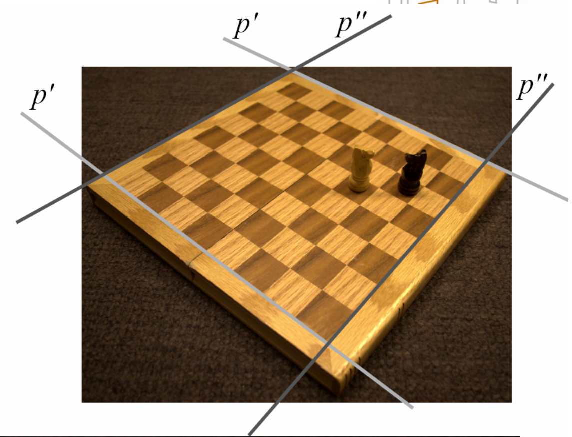




Perspective clone brush

Oh, Durand, Dorsey, unpublished

- **Correct for perspective**
- **And other tricks**



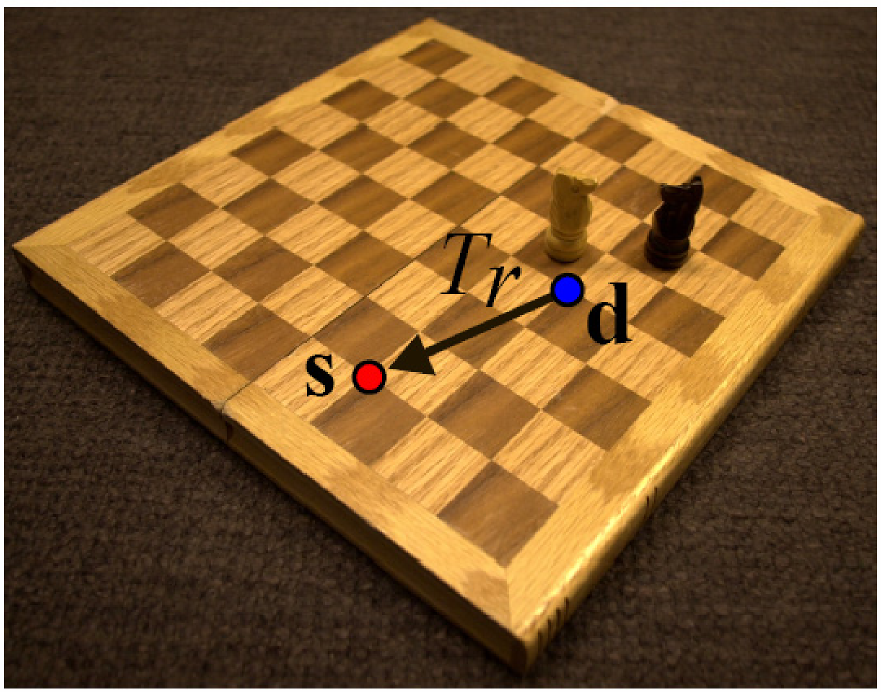
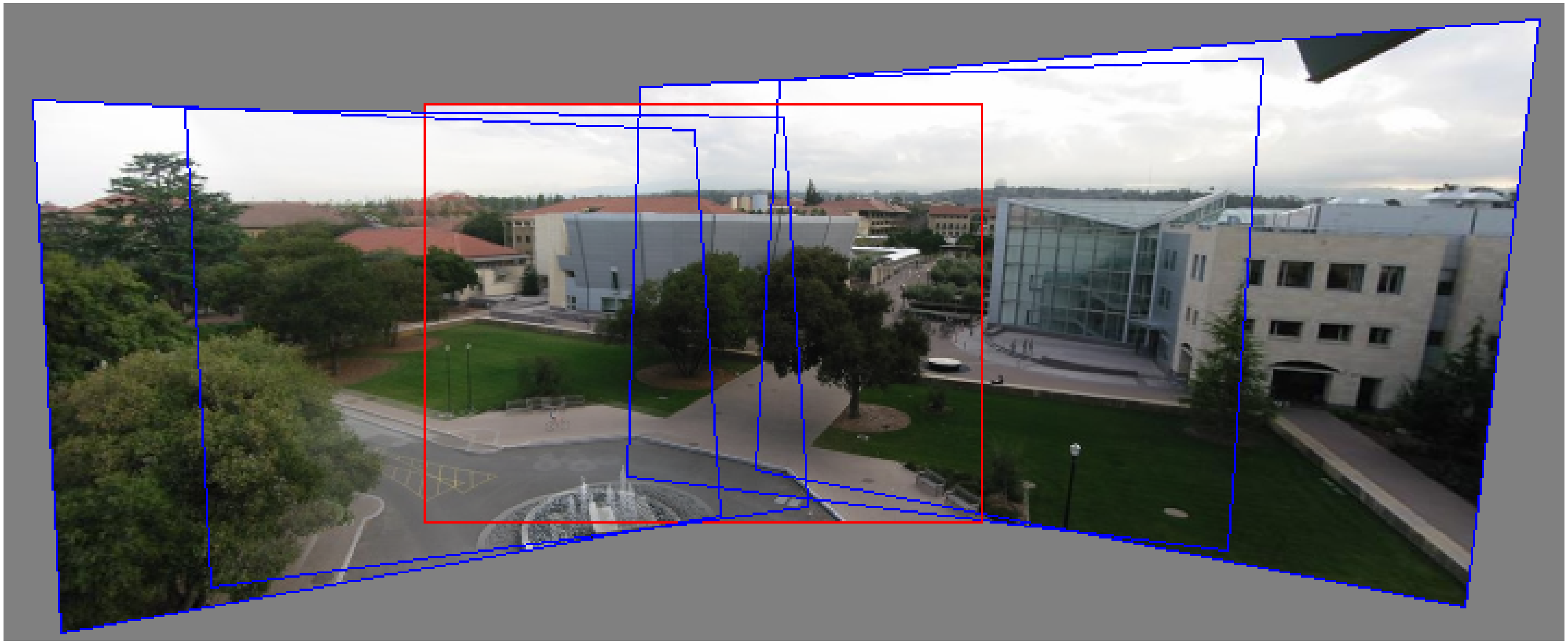


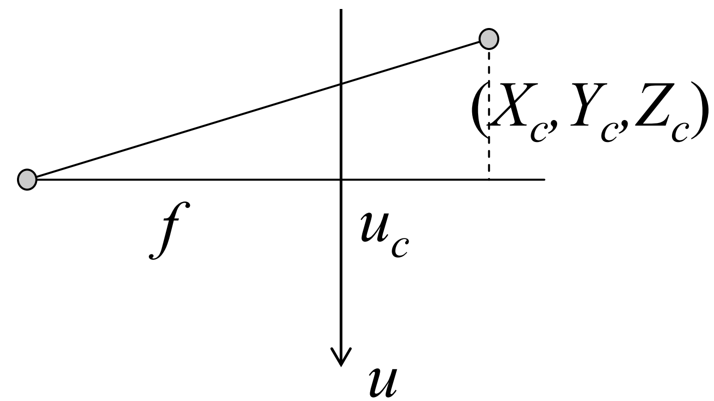
Figure 15: The cars and the street furniture have been removed. This example took less than 10 minutes.

Rotational Mosaics



- **Can we say something more about <u>rotational</u> mosaics?**
- **i.e. can we further constrain our H?**

3D → 2D Perspective Projection



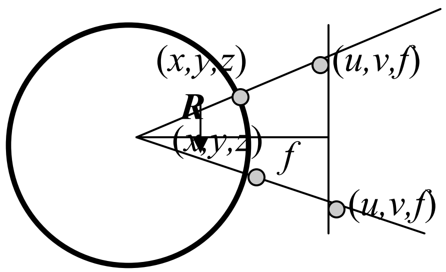
$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} f & 0 & u_c \\ 0 & f & v_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

K

3D Rotation Model

- **Projection equations**

1. **Project from image to 3D ray**
   - $(x_0,y_0,z_0) = (u_0-u_c,v_0-v_c,f)$
2. **Rotate the ray by camera motion**
   - $(x_1,y_1,z_1) = R_{01}\,(x_0,y_0,z_0)$
3. **Project back into new (source) image**
   - $(u_1,v_1) = (fx_1/z_1+u_c,fy_1/z_1+v_c)$

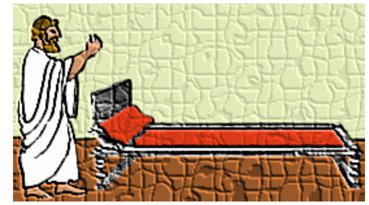


- **Therefore:**

$$\mathbf{H} = \mathbf{K}_0\mathbf{R}_{01}\mathbf{K}_1^{-1}$$

- **Our homography has only 3,4 or 5 DOF, depending if focal length is known, same, or different.**
  - This makes image registration much better behaved

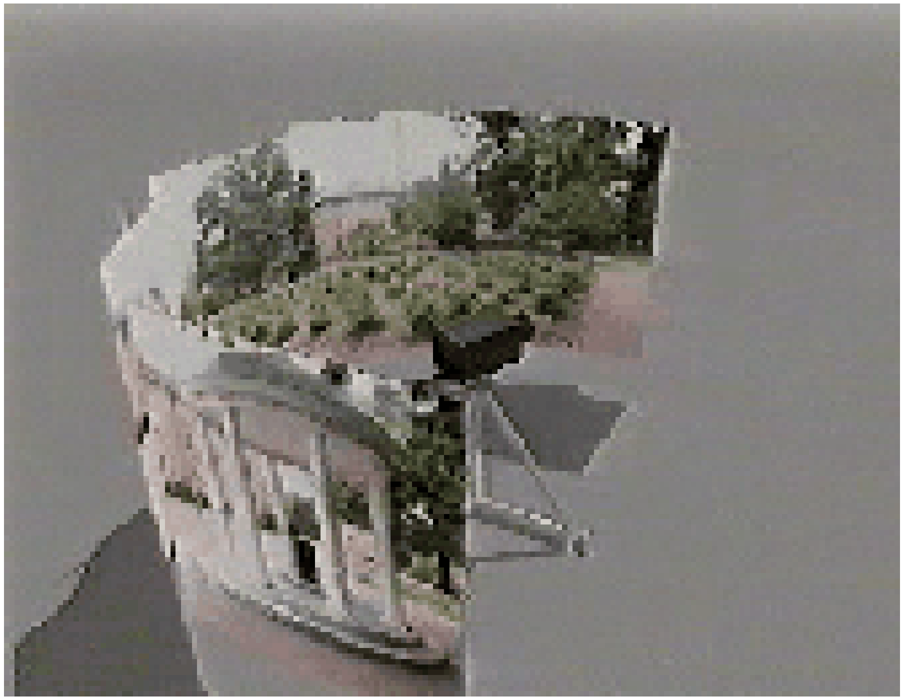
Pairwise alignment



- **Procrustes Algorithm [Golub & VanLoan]**

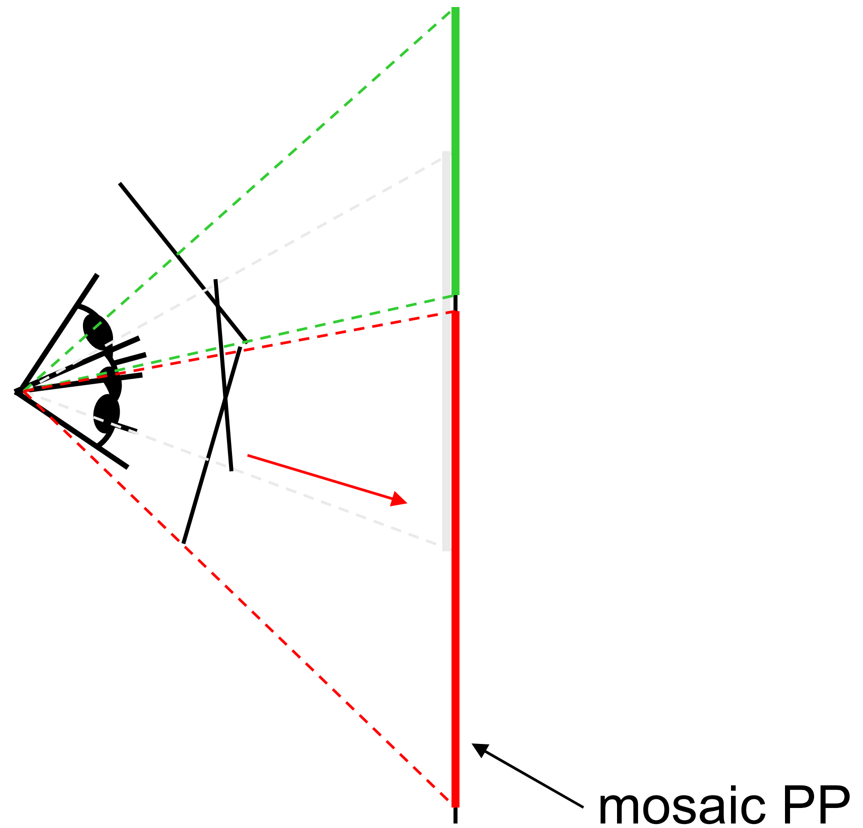
- **Given two sets of matching points, compute R**
- $p_i' = R\, p_i$ **with 3D rays**
- $p_i = N(x_i, y_i, z_i) = N(u_i - u_c, v_i - v_c, f)$
- $A = \Sigma_i\, p_i\, p_i'^T = \Sigma_i\, p_i\, p_i^T R^T = U S V^T = (U S U^T) R^T$
- $V^T = U^T R^T$
- $R = V U^T$

Rotation about vertical axis



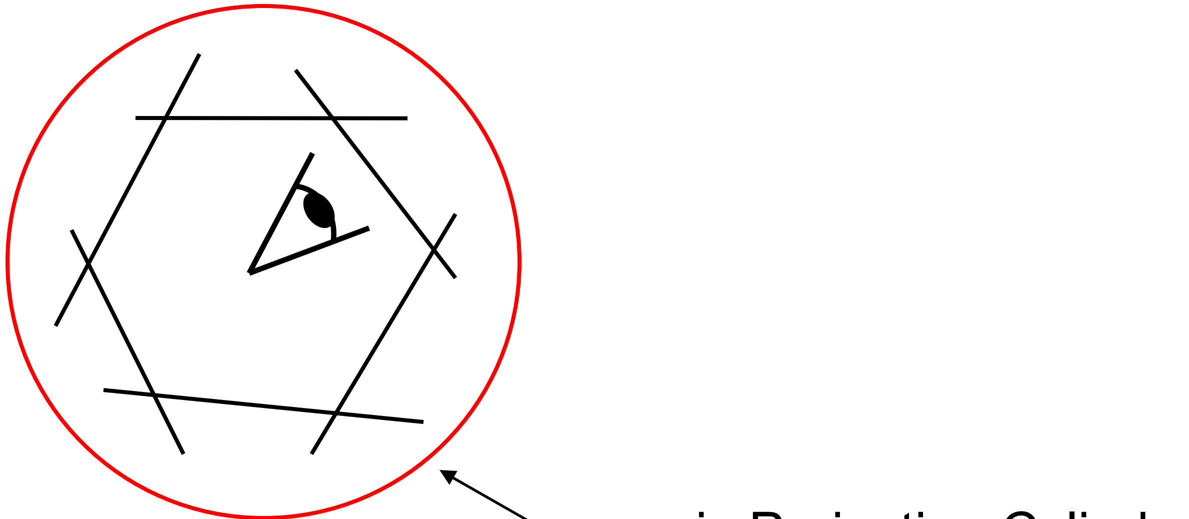
- What if our camera rotates on a tripod?
- What's the structure of H?

Do we have to project onto a plane?

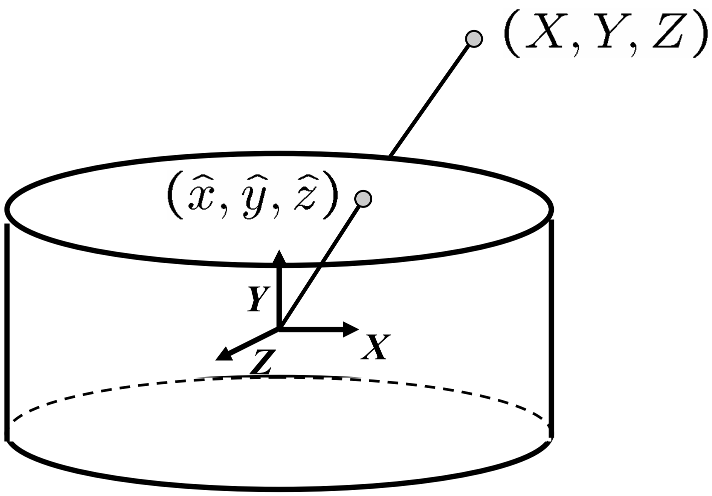


Full Panoramas

- **What if you want a 360° field of view?**



Cylindrical projection



unit cylinder

- Map 3D point (X,Y,Z) onto cylinder

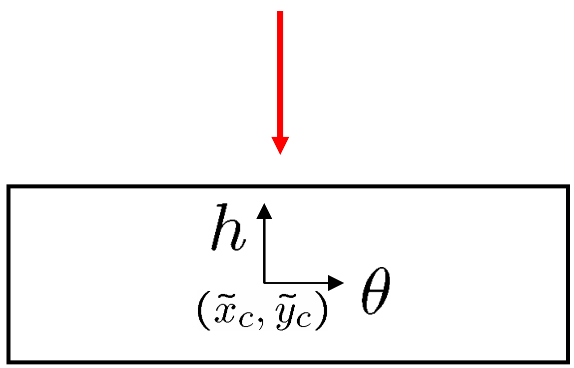
$$(\hat{x}, \hat{y}, \hat{z}) = \frac{1}{\sqrt{X^2+Z^2}}(X, Y, Z)$$

- Convert to cylindrical coordinates

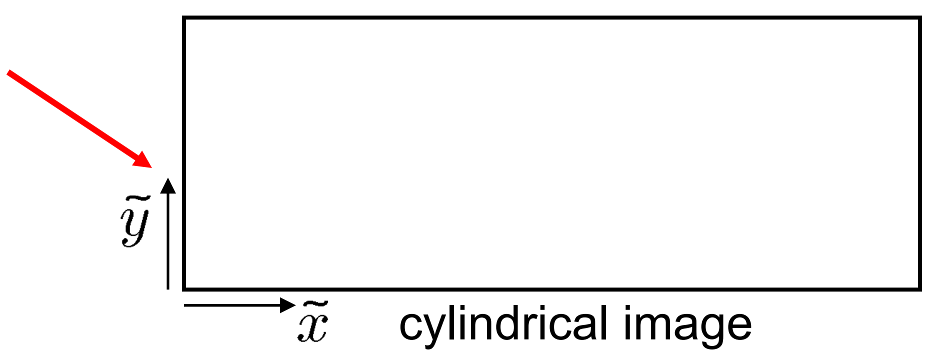
$$(sin\theta, h, cos\theta) = (\hat{x}, \hat{y}, \hat{z})$$

- Convert to cylindrical image coordinates

$$(\tilde{x}, \tilde{y}) = (f\theta, fh) + (\tilde{x}_c, \tilde{y}_c)$$

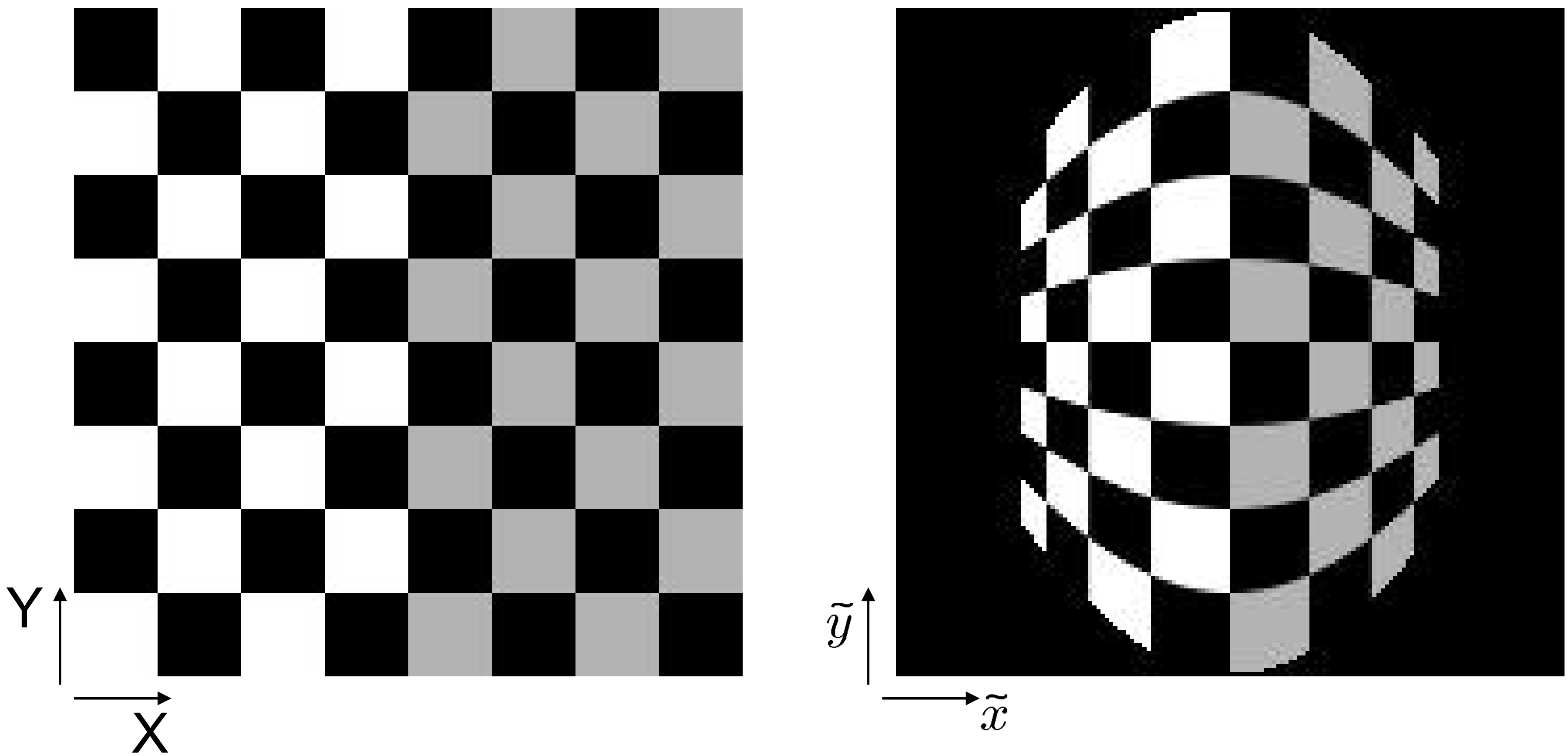


unwrapped cylinder

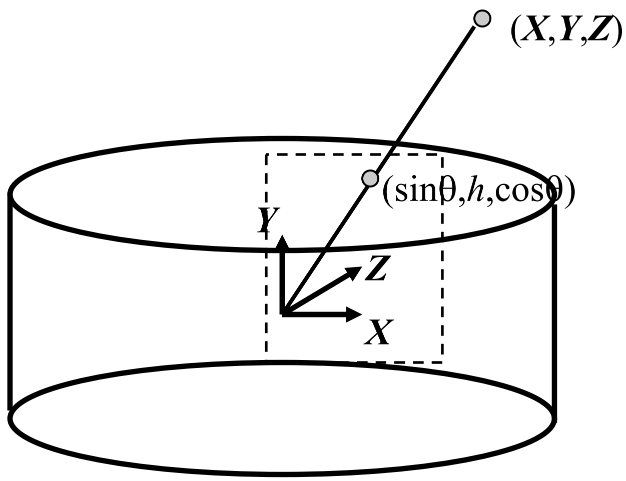


cylindrical image

Cylindrical Projection

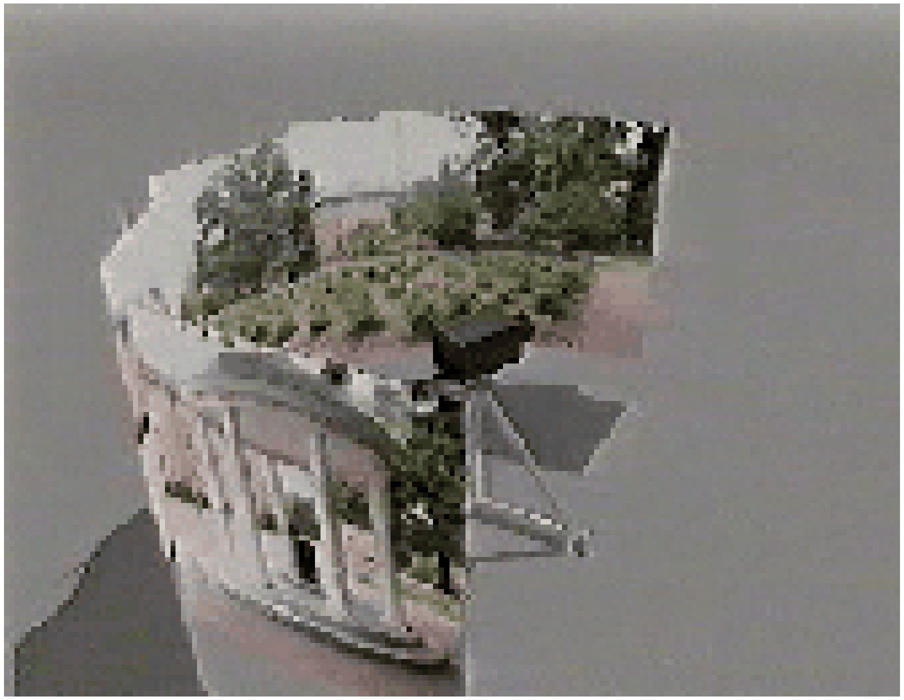


Inverse Cylindrical projection



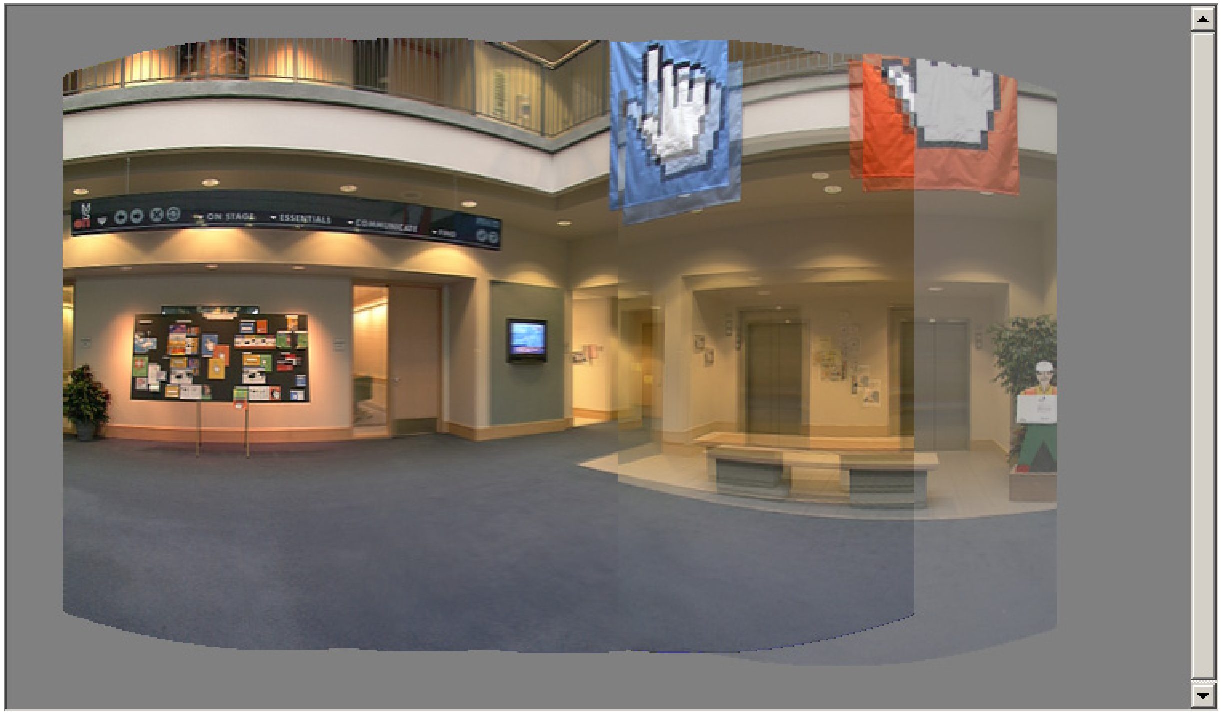
$$
\begin{aligned}
\theta &= (x_{cyl} - x_c)/f \\
h &= (y_{cyl} - y_c)/f \\
\hat{x} &= \sin\theta \\
\hat{y} &= h \\
\hat{z} &= \cos\theta \\
x &= f\hat{x}/\hat{z} + x_c \\
y &= f\hat{y}/\hat{z} + y_c
\end{aligned}
$$

Cylindrical panoramas



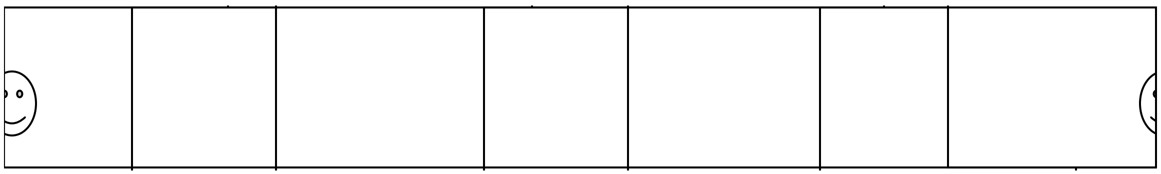
- **Steps**
  - Reproject each image onto a cylinder
  - Blend
  - Output the resulting mosaic
- **What are the assumptions here?**

Cylindrical image stitching



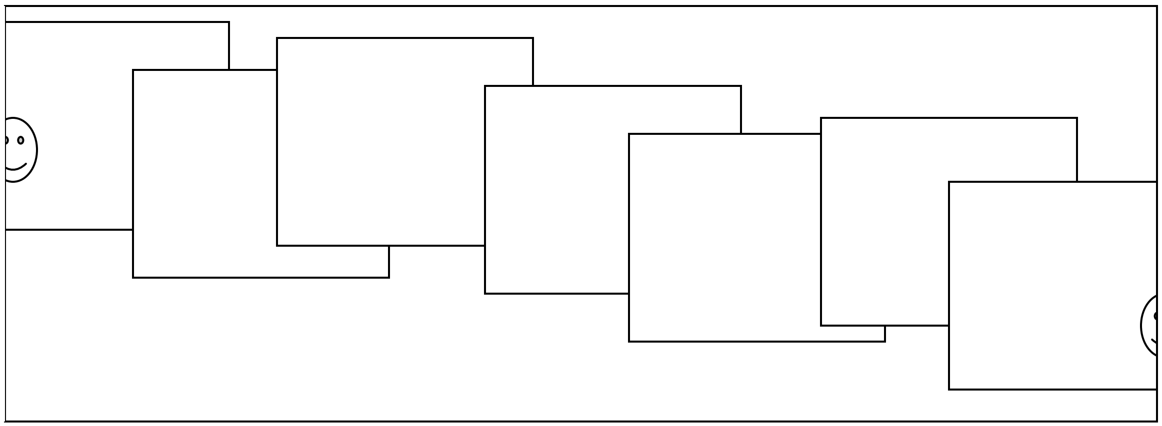
- **What if you don't know the camera rotation?**
  - Solve for the camera rotations
    - Note that a rotation of the camera is a **translation** of the cylinder!

Assembling the panorama



- **Stitch pairs together, blend, then crop**

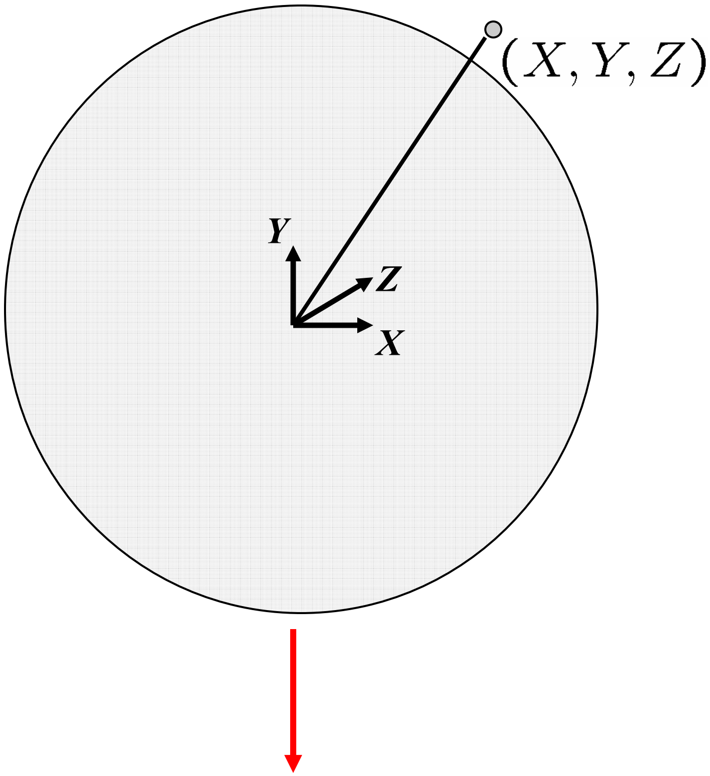
Problem: Drift



- **Vertical Error accumulation**
  - small (vertical) errors accumulate over time
  - apply correction so that sum = 0 (for 360° pan.)
- **Horizontal Error accumulation**
  - can reuse first/last image to find the right panorama radius

Full-view (360°) panoramas

Spherical projection



- Map 3D point (X,Y,Z) onto sphere

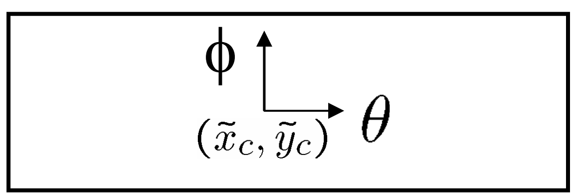
$$(\hat{x},\hat{y},\hat{z}) = \frac{1}{\sqrt{X^2+Y^2+Z^2}}(X,Y,Z)$$

- Convert to spherical coordinates

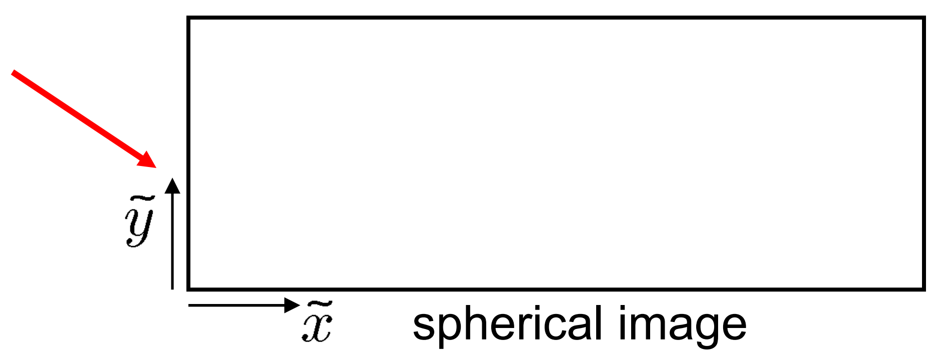
$$(\sin\theta\cos\phi, \sin\phi, \cos\theta\cos\phi) = (\hat{x},\hat{y},\hat{z})$$

- Convert to spherical image coordinates

$$(\tilde{x},\tilde{y}) = (f\theta, fh) + (\tilde{x}_c,\tilde{y}_c)$$

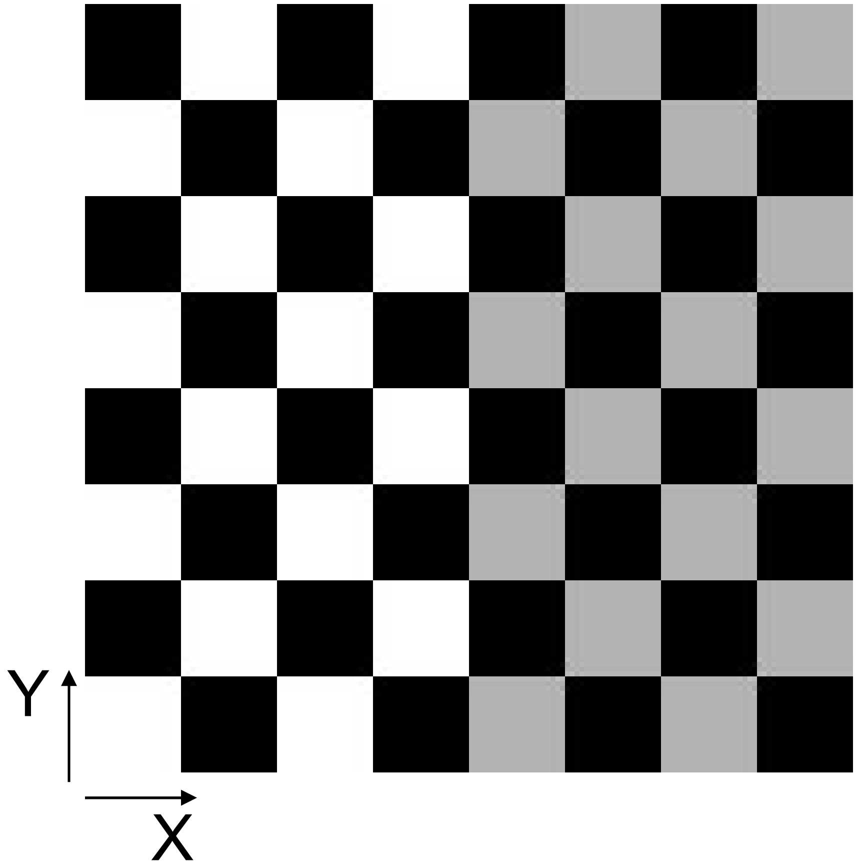


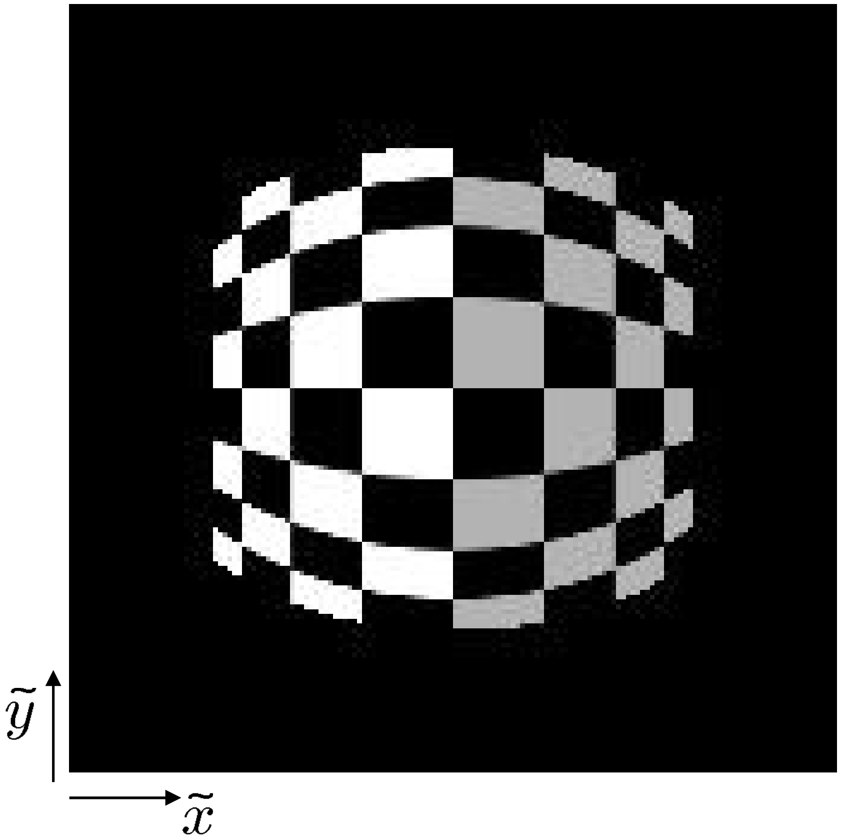
unwrapped sphere



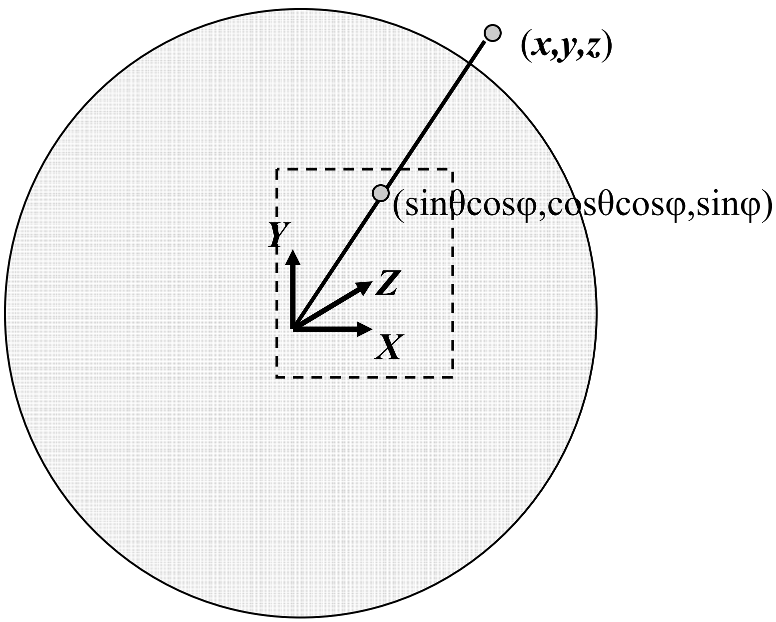
spherical image

Spherical Projection





Inverse Spherical projection



$$\theta = (x_{sph} - x_c)/f$$

$$\varphi = (y_{sph} - y_c)/f$$

$$\hat{x} = \sin\theta \cos\varphi$$

$$\hat{y} = \sin\varphi$$

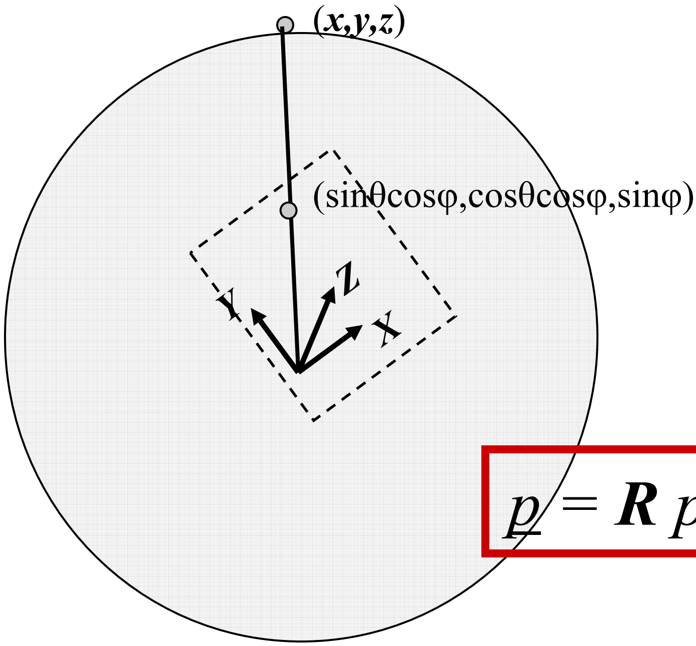
$$\hat{z} = \cos\theta\cos\varphi$$

$$x = f\hat{x}/\hat{z} + x_c$$

$$y = f\hat{y}/\hat{z} + y_c$$

3D rotation

- **Rotate image before placing on unrolled sphere**



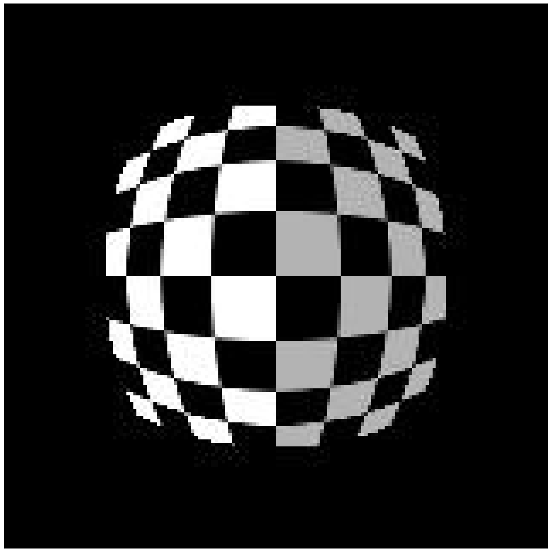
$$\underline{p} = \boldsymbol{R}\, p$$

$$\begin{aligned}
\theta &= (x_{sph} - x_c)/f \\
\varphi &= (y_{sph} - y_c)/f \\
\hat{x} &= \sin\theta\ \cos\varphi \\
\hat{y} &= \sin\varphi \\
\hat{z} &= \cos\theta\cos\varphi \\
x &= f\hat{x}/\hat{z} + x_c \\
y &= f\hat{y}/\hat{z} + y_c
\end{aligned}$$

Full-view Panorama

Polar Projection

- **Extreme "bending" in ultra-wide fields of view**



$$\hat{r}^2 = \hat{x}^2 + \hat{y}^2$$

$$(\cos\theta\sin\phi, \sin\theta\sin\phi, \cos\phi) = s\,(x, y, z)$$

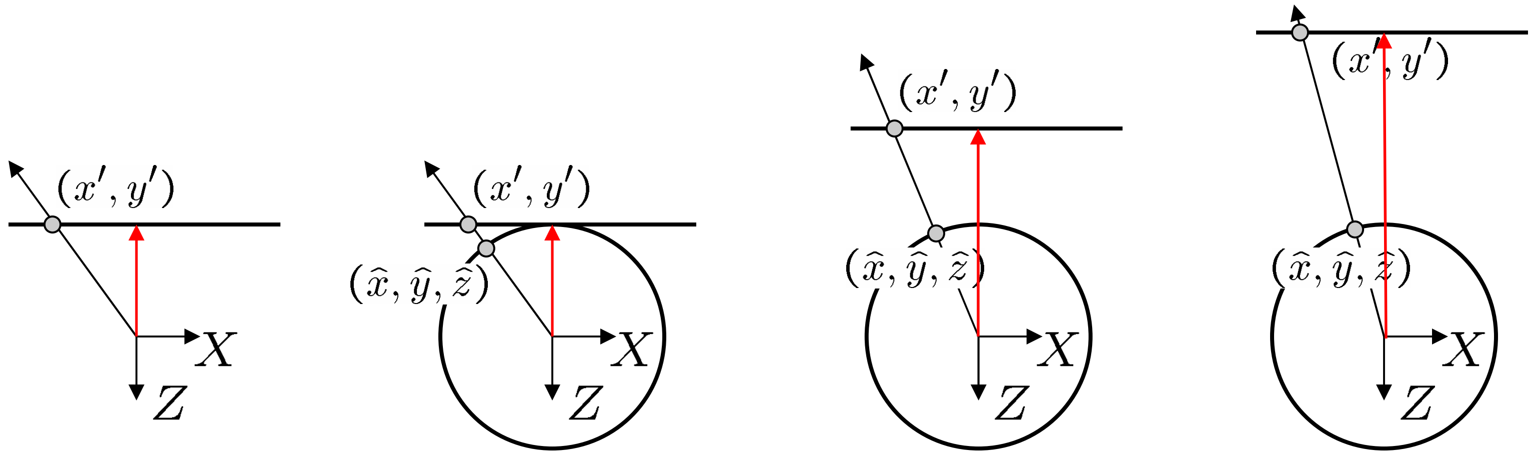
uations become

$$\begin{aligned} x' &= s\phi\cos\theta = s\frac{x}{r}\tan^{-1}\frac{r}{z}, \\ y' &= s\phi\sin\theta = s\frac{y}{r}\tan^{-1}\frac{r}{z}, \end{aligned}$$

Other projections are possible

- **You can stitch on the plane and then warp the resulting panorama**
  - What's the limitation here?
- **Or, you can use these as stitching surfaces**
  - But there is a catch…

Cylindrical reprojection



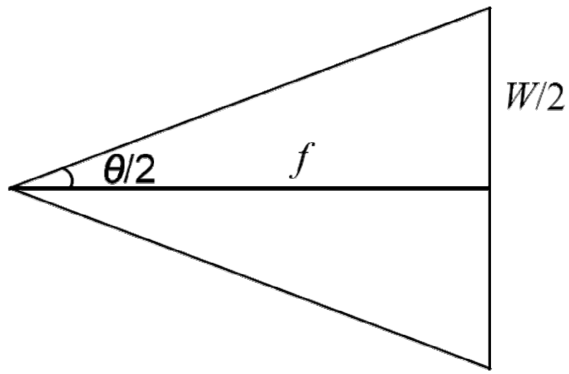
top-down view

Focal length – the dirty secret…

Image 384x300 **f = 180 (pixels)** **f = 280** **f = 380**

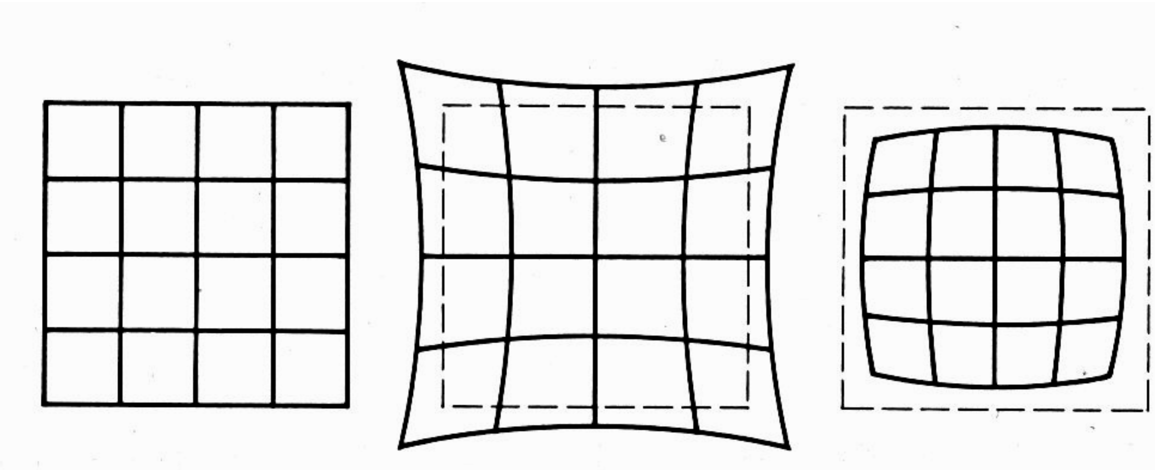
What's your focal length, buddy?

- **Focal length is (highly!) camera dependant**
  - Can get a rough estimate by measuring FOV:



  - Can use the EXIF data tag (might not give the right thing)
  - Can use several images together and try to find f that would make them match
  - Can use a known 3D object and its projection to solve for f
  - Etc.
- **There are other camera parameters too:**
  - Optical center, non-square pixels, lens distortion, etc.

Distortion

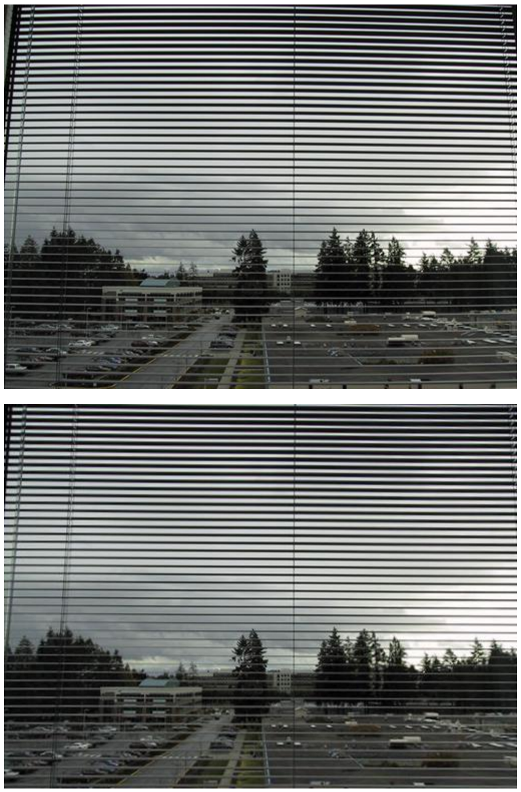


No distortion Pin cushion Barrel

- **Radial distortion of the image**
  - Caused by imperfect lenses
  - Deviations are most noticeable for rays that pass through the edge of the lens

Radial distortion

- **Correct for "bending" in wide field of view lenses**



$$
\begin{aligned}
\hat{r}^2 &= \hat{x}^2 + \hat{y}^2 \\
\hat{x}' &= \hat{x}/(1 + \kappa_1 \hat{r}^2 + \kappa_2 \hat{r}^4) \\
\hat{y}' &= \hat{y}/(1 + \kappa_1 \hat{r}^2 + \kappa_2 \hat{r}^4) \\
x &= f\hat{x}'/\hat{z} + x_c \\
y &= f\hat{y}'/\hat{z} + y_c
\end{aligned}
$$

Use this instead of normal projection

Blending the mosaic

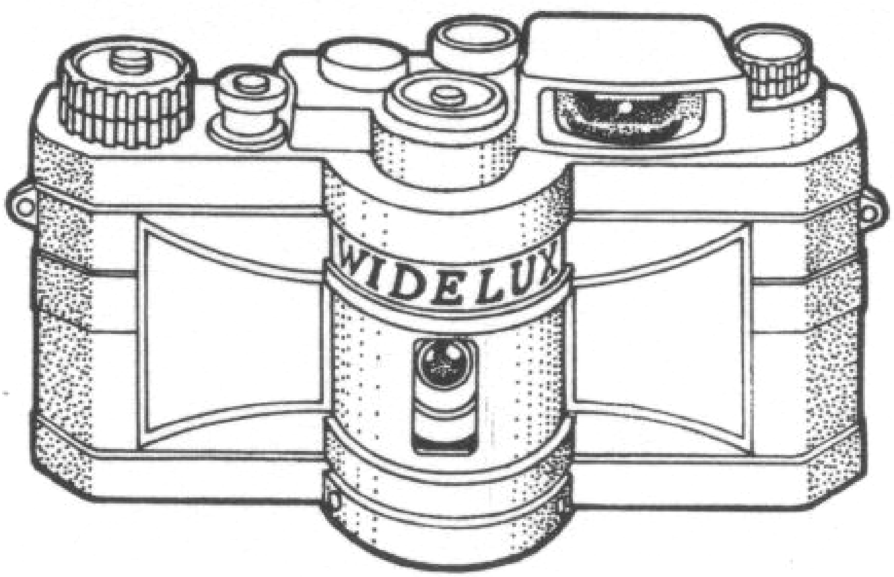
An example of image compositing: the art (and sometime science) of combining images together…

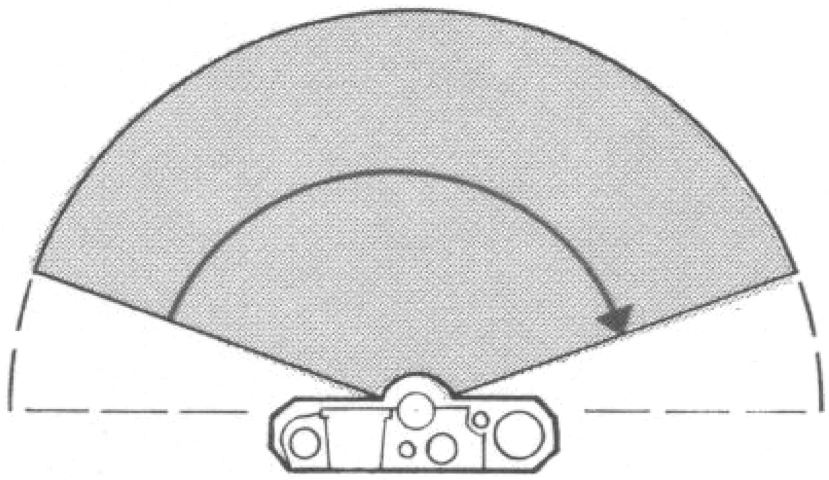
Multi-band Blending

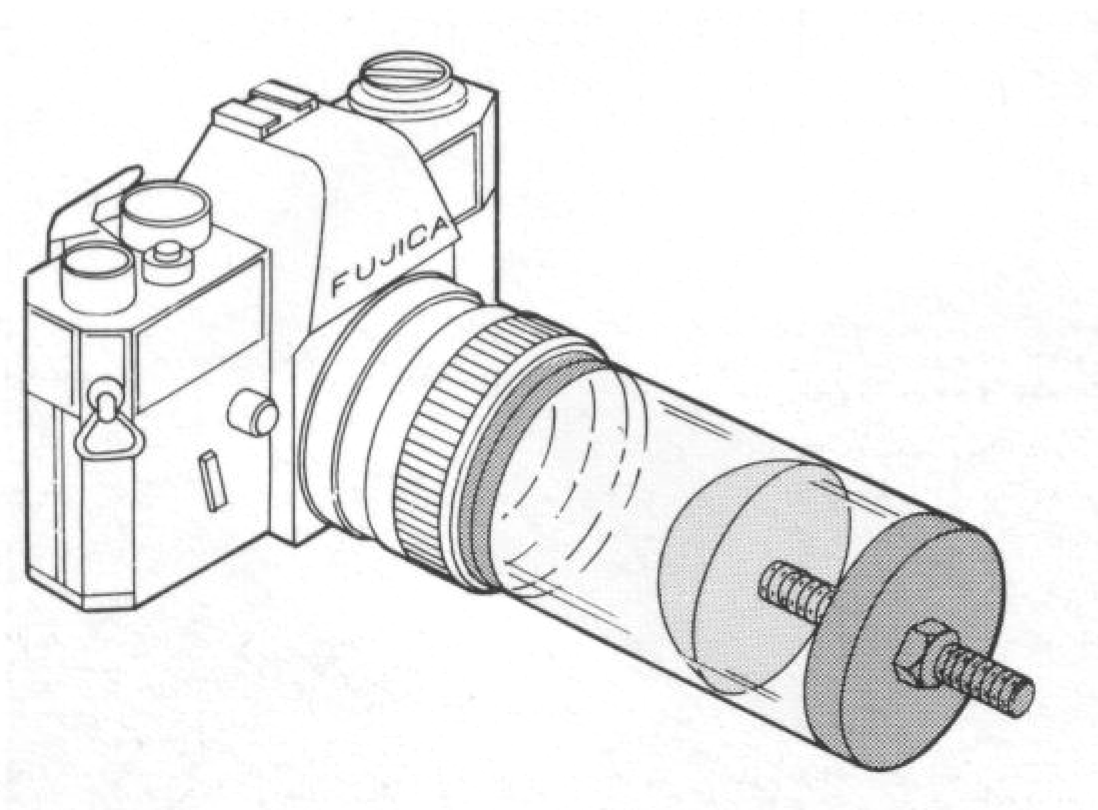
Multi-band Blending

- **Burt & Adelson 1983**
  - Blend frequency bands over range $\propto \lambda$

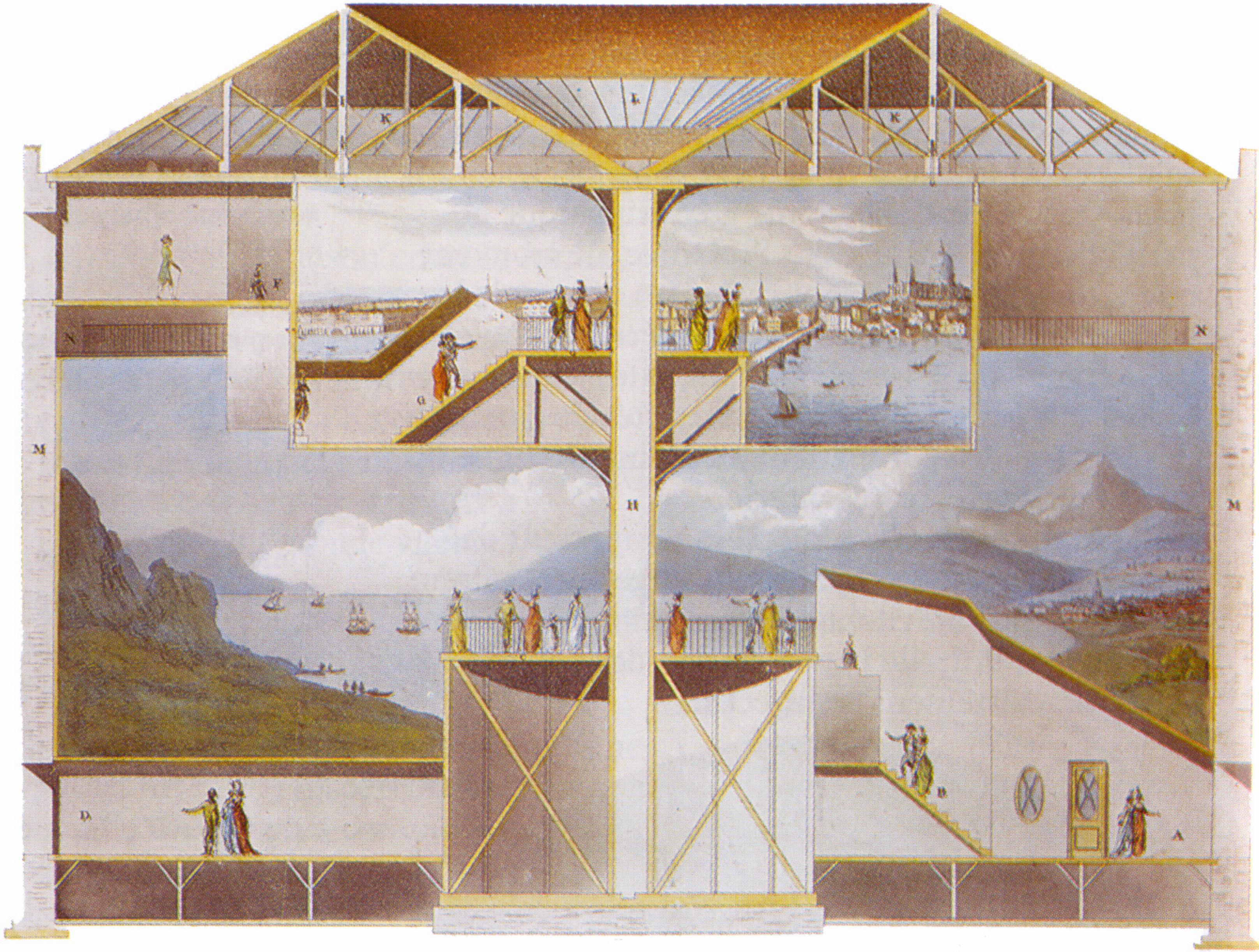
Traditional panoramas






FUJICA

19^{th} century panorama



Chinese scroll

Magic: automatic panos

http://research.microsoft.com/~brown/papers/iccv2003.pdf

Magic: ghost removal

- **See also HDR lecture**

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

Magic: ghost removal

- **See also HDR lecture**

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

Extensions

- **Video**
- **Additional objects**
- **Mok's panomorph**
- **http://www.sarnoff.com/products_services/vision/tech_papers/kumarvb.pdf**
- **http://www.cs.huji.ac.il/~peleg/papers/pami00-manifold.pdf**
- **http://www.cs.huji.ac.il/~peleg/papers/cvpr00-rectified.pdf**
- **http://www.cs.huji.ac.il/~peleg/papers/cvpr05-dynmos.pdf**
- **http://citeseer.ist.psu.edu/cache/papers/cs/20590/http:zSzzSzwww.sarnoff.comzSzcareer_movezSztech_paperszSzpdfzSzvisrep95.pdf/kumar95representation.pdf**
- **http://www.robots.ox.ac.uk/~vgg/publications/papers/schaffalitzky02.pdf**

Software

- http://photocreations.ca/collage/circle.jpg
  http://webuser.fh-furtwangen.de/%7Edersch/
  http://www.ptgui.com/
  http://hugin.sourceforge.net/
  http://epaperpress.com/ptlens/
  http://www.panotools.info/mediawiki/index.php?title=Tutorials

  http://www.fdrtools.com/front_e.php

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/