Mosaics (homographies and blending)

Many slides from
Yuri Boykov, Alexei Efros, Steve Seitz, Rick Szeliski

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°

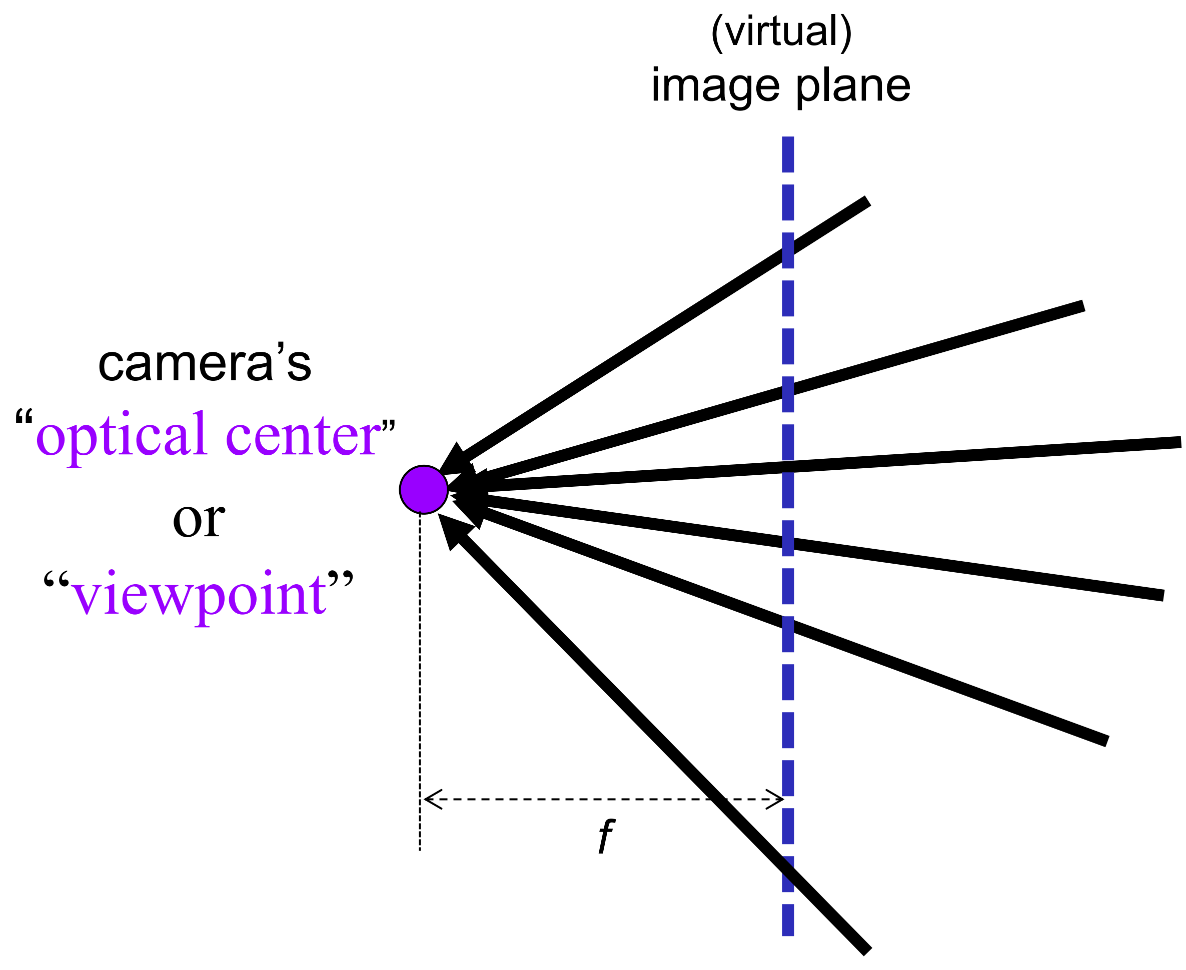
Slide from Brown & Lowe

Mosaics: stitching images together

virtual wide-angle camera

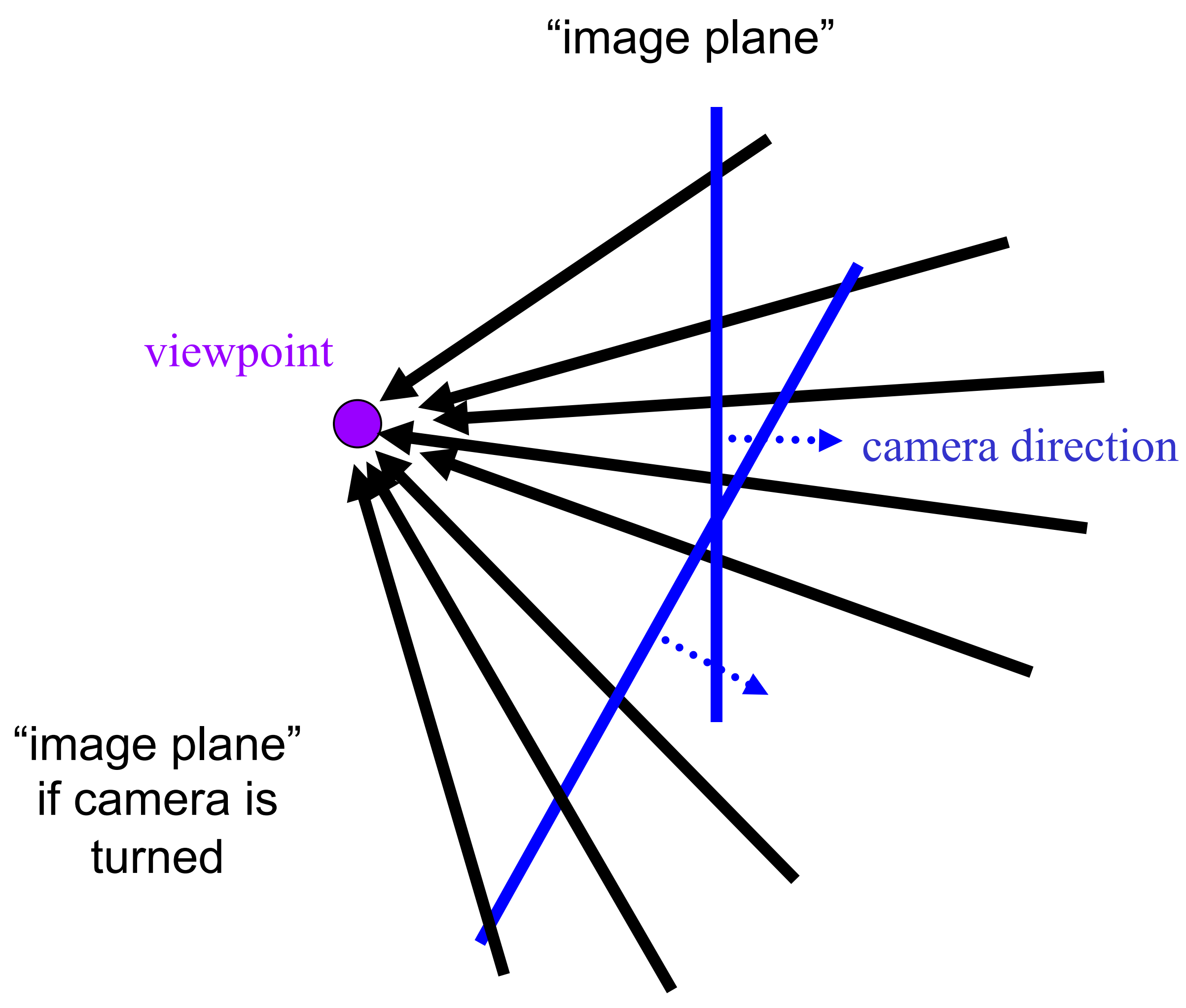
Basic camera model: “pin hole”

remember from lecture 2

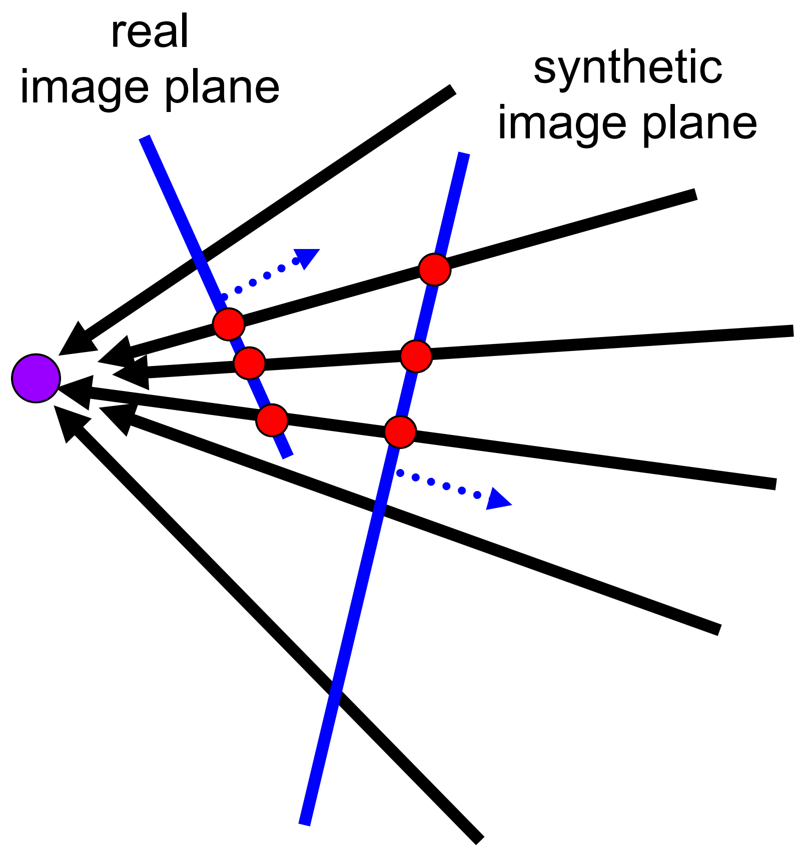


commonly used simplified representation of a pin hole camera
draws an image plane in front of the optical center

Rotating camera around fixed viewpoint

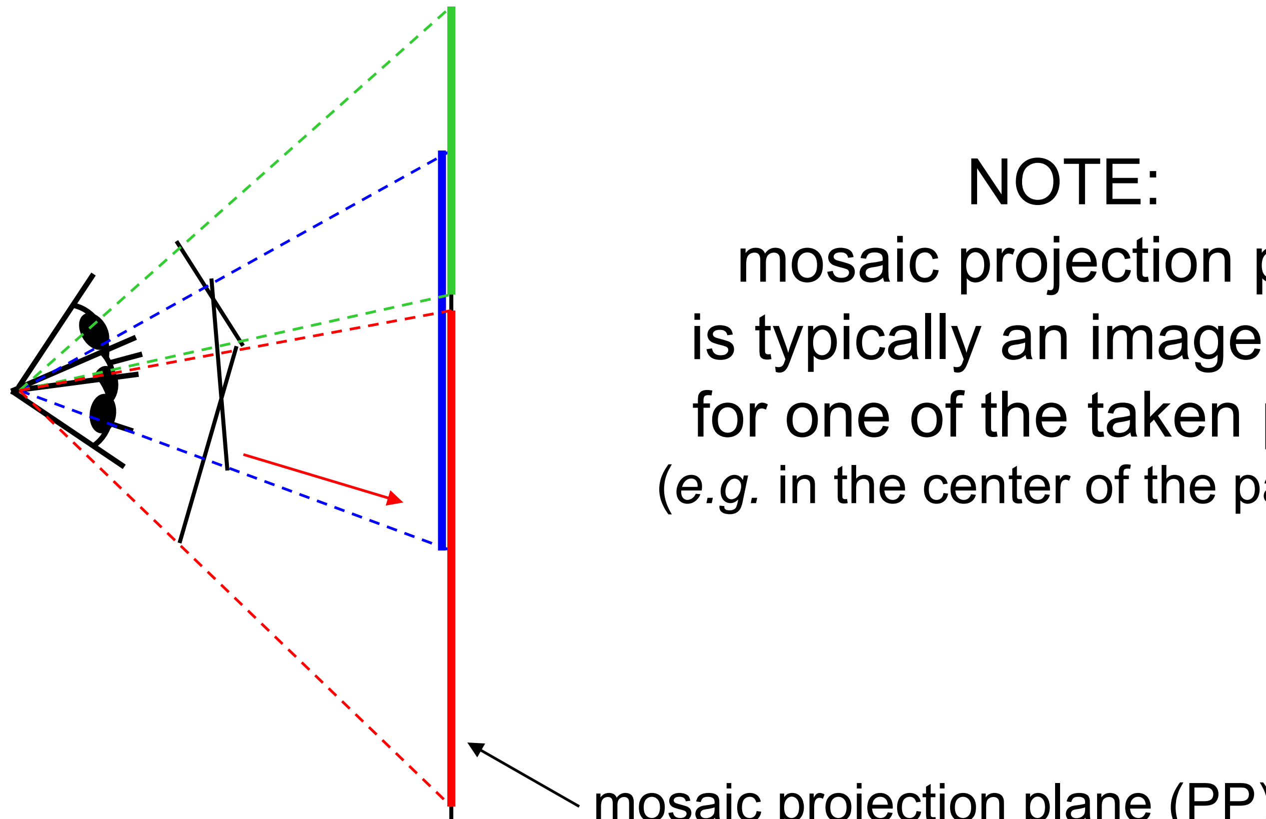


A pencil of rays contains all views



It is possible to generate any synthetic camera view as long as it has **the same center of projection**! (domain transformation defined by ray-correspondences)

Panorama: general idea (3D interpretation)



NOTE:
mosaic projection plane
is typically an image plane
for one of the taken photos
(*e.g.* in the center of the panorama)

The mosaic has a natural interpretation in 3D

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

How to build panorama mosaic?

Basic Iterative 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

NOTE: knowing scene geometry is not needed to build panoramas

However, general 3D geometric interpretation of panorama mosaicing helps to understand the type of **transformation** needed for image reprojection.

Aligning images

left on top

right on top

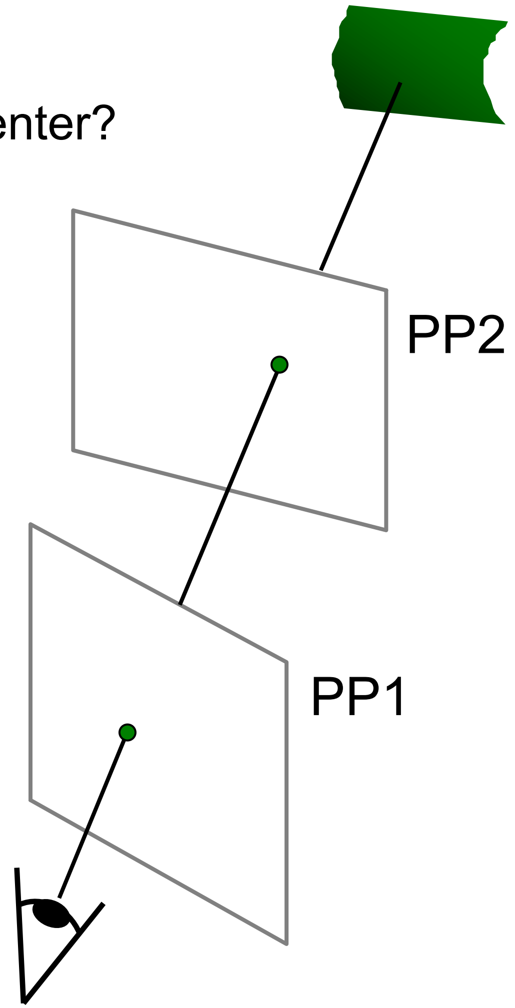
Translations are not enough to align the images

Registration via ray correspondences... How?

Image reprojection

Basic question

- How to relate two images from the same camera center?
  That is, how to map pixels from PP1 to PP2 ?



Answer 1: ray correspondence (as seen earlier)

- Cast a ray through any given pixel in PP1
- Draw the pixel where that ray intersects PP2

But don’t we need to know the positions of the two planes w.r.t. the viewpoint?

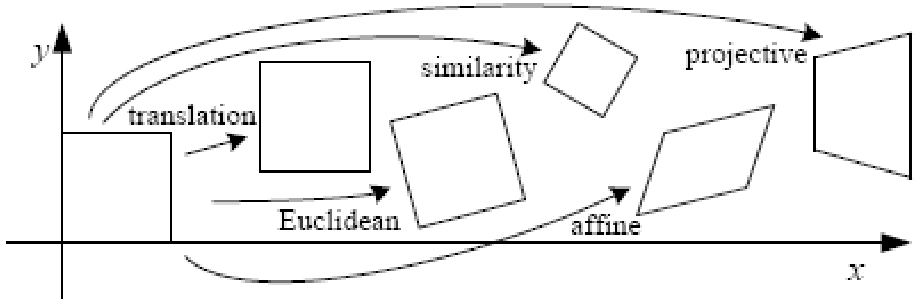
Answer 2: rather than thinking of this as a 3D reprojection, think of it as a 2D **image warp** from one image to another.

What type of 2D image warp can represent 3D reprojections?

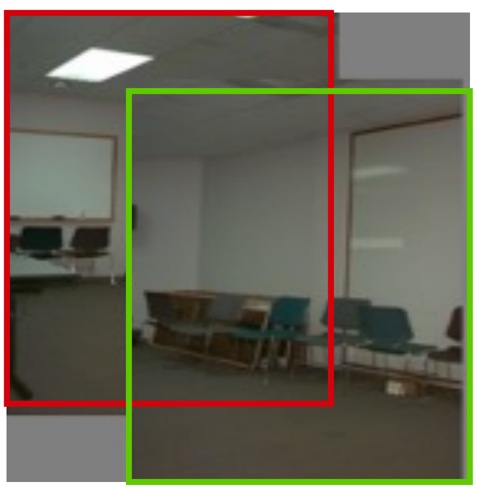
Back to Image Warping

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

e.g. translation, Euclidean, affine, projective ?

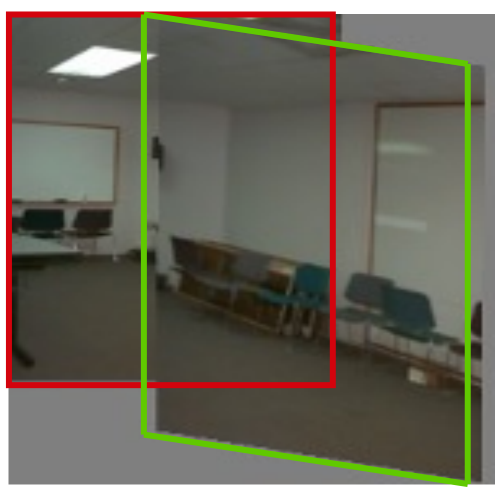


Translation



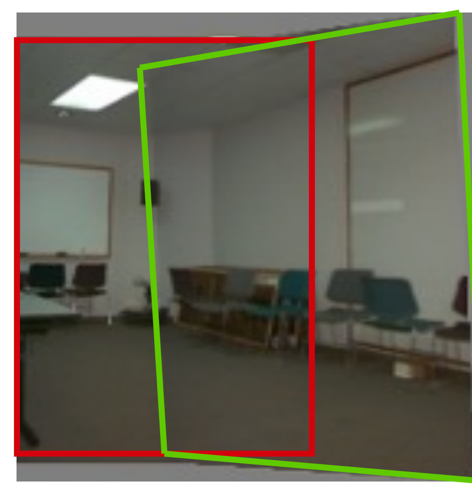
2 unknowns

Affine



6 unknowns

Perspective

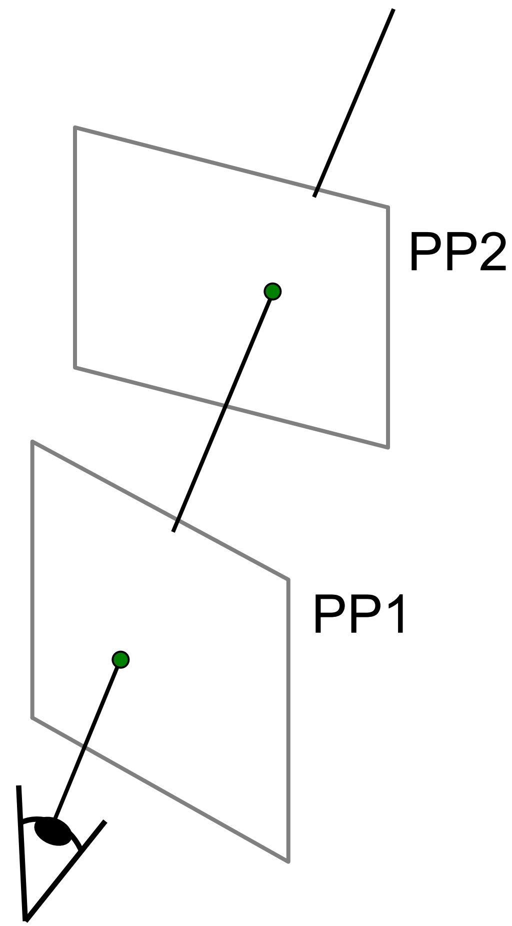


8 unknowns

Central Projection and Homographies

Central projection: mapping between any two PPs with the same center of projection based on ray-correspondences

- preserves straight lines (**Why?**)
- parallel lines aren't (**Example?**) thus not affine
- rectangle should map to arbitrary quadrilateral



<u>Since straight lines are preserved, it can be described by a homographic transformation</u>

(remember general property of homographies from topic 4)

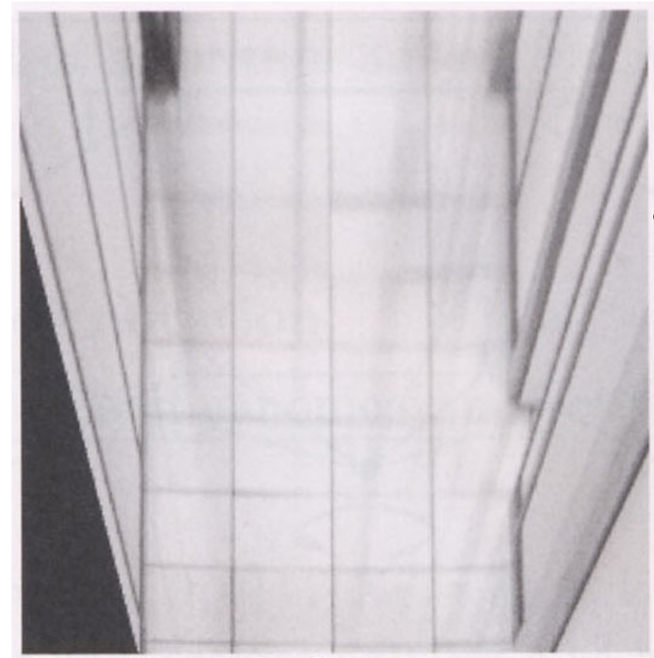
$$\underbrace{\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix}}_{\mathbf{p'}} = \underbrace{\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}}_{\mathbf{H}} \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{\mathbf{p}}$$

using homogeneous representation of 2D points

w.r.t. <u>arbitrary coordinate basis</u> in each plane

Extra assumptions (e.g. orthogonal basis) allow to express central projection as some transformation with d.o.f. < 8 [Heartlely and Zisserman, Sec. 2.3]

Image warping with homographies



“down” view

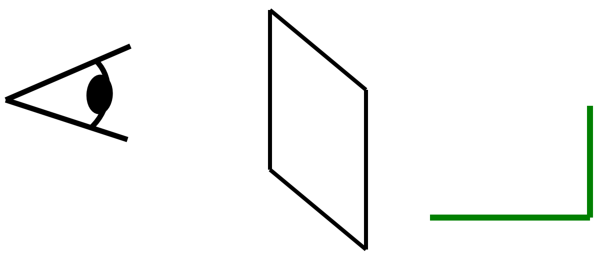


image plane in front

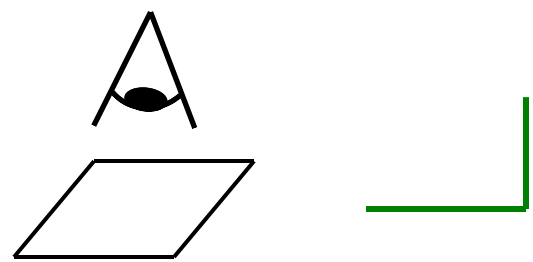
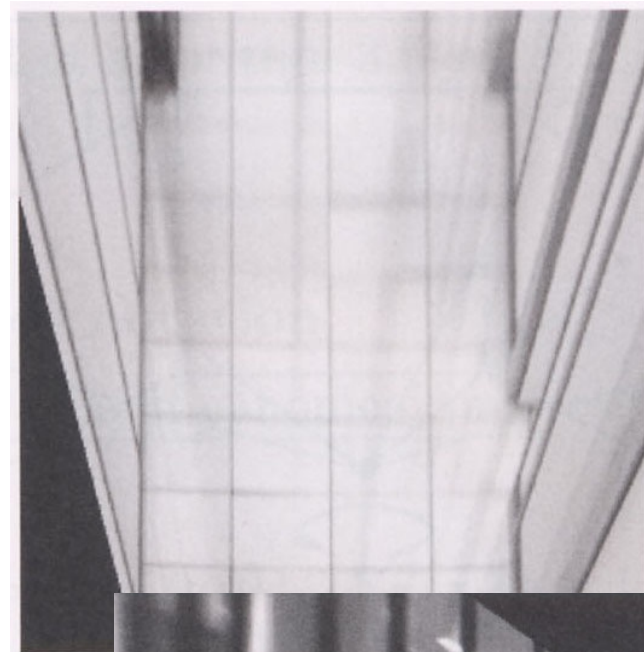


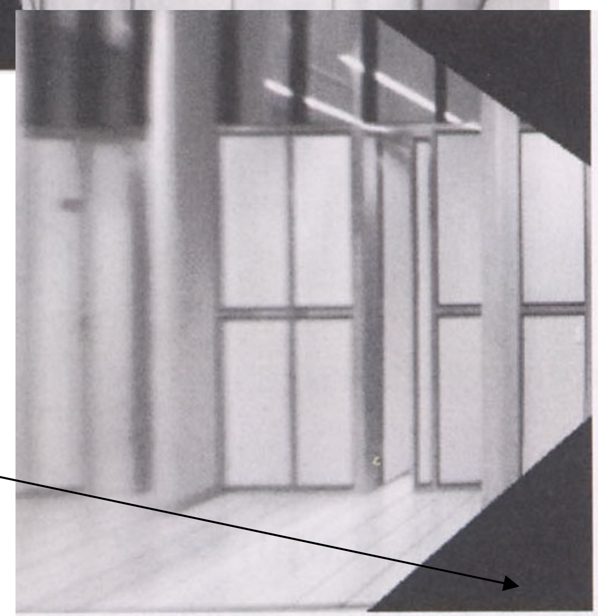
image plane below

black area
where no pixel
maps to

Image warping with homographies



“down” view



“side” view

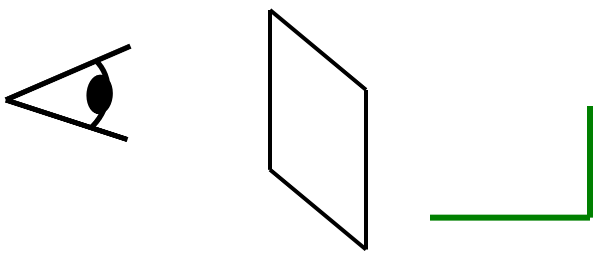
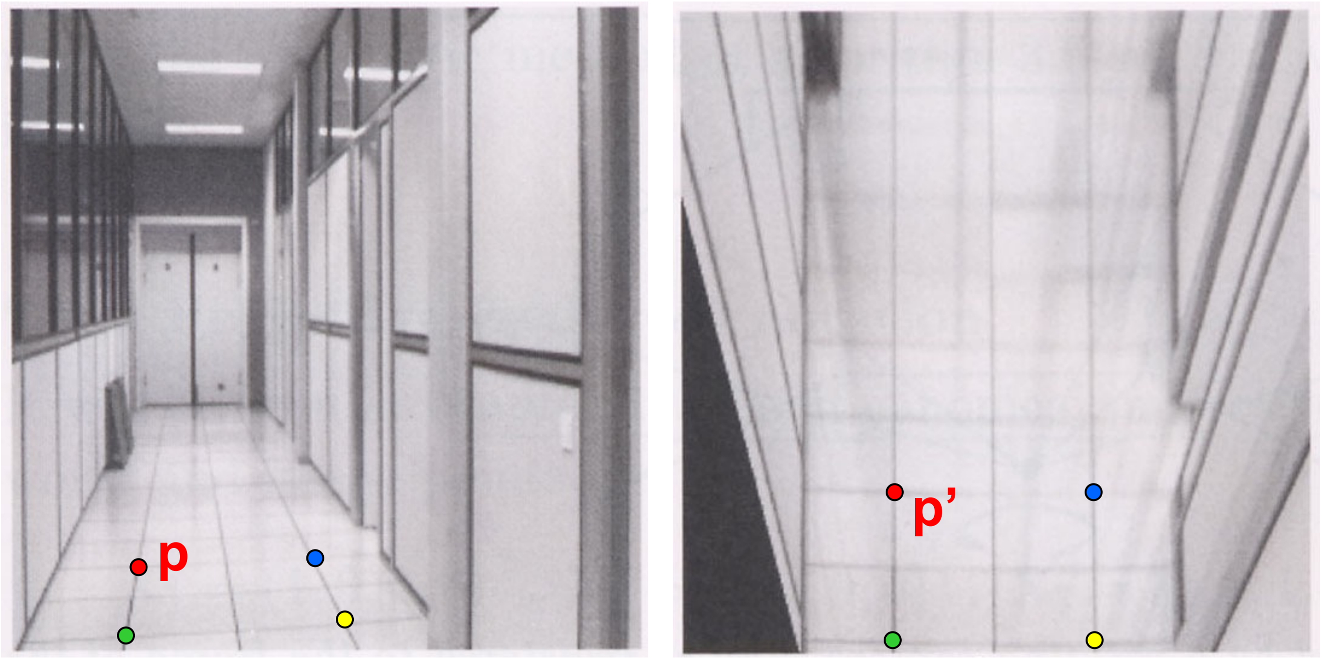


image plane in front

black area where no pixel maps 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 if more than 4 point correspondences

Fun with homographies

Original image

Virtual camera rotations

Computing Homography

Consider one point-correspondence $p = (x, y) \rightarrow p' = (x', y')$

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

3 equations, but we do not care about w

eliminating $w = gx + hy + i$:

$$\Rightarrow \quad \boxed{\begin{aligned} ax + by + c - gxx' - hyx' - ix' &= 0 \\ dx + ey + f - gxy' - hyy' - iy' &= 0 \end{aligned}}$$

Two equations linear w.r.t unknown coefficients of matrix H and quadratic w.r.t. known point coordinates (x,y,x',y')

also $\quad x' = \dfrac{ax + by + c}{gx + hy + i} \qquad y' = \dfrac{dx + ey + f}{gx + hy + i}$

See p.35 in Hartley and Zisserman

Note: nonlinear equations for x,y (but this is irrelevant here)

Computing Homography

Consider 4 point-correspondences $p_k = (x_k, y_k) \rightarrow p'_k = (x'_k, y'_k)$

$$\mathbf{p'}_k = \mathbf{H}\mathbf{p}_k \qquad \begin{bmatrix} w_k x'_k \\ w_k y'_k \\ w_k \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ 1 \end{bmatrix}$$

for k=1,2,3,4

$$\Rightarrow \quad \begin{aligned} ax_k + by_k + c - gx_k x'_k - hy_k x'_k - ix'_k = 0 \\ dx_k + ey_k + f - gx_k y'_k - hy_k y'_k - iy'_k = 0 \end{aligned}$$

Special case of DLT method (see p.89 in Hartley and Zisserman)

Can solve for unknown Homography parameters $\{a, b, c, d, e, f, g, h, i\}$ **from 8 (=2x4) linear equations above plus some additional assumption**

For example, maybe assume $i=1$

(is it OK or not?)

assume that the "*vanishing point*" is at the center of image coordinates

the rail tracks are parallel on this image

$$p = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

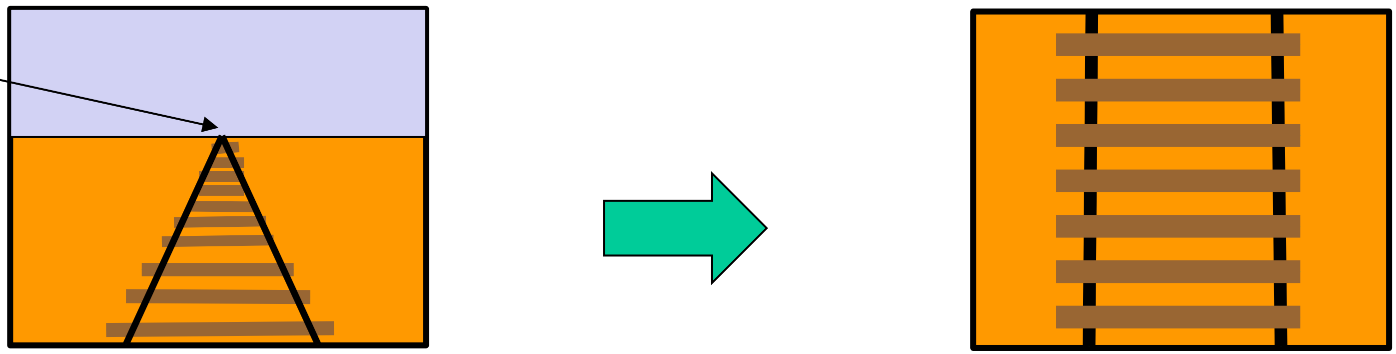


image 1

image 2
(camera looks down)

Q: select a feasible homography from image plane 1 to image plane 2

A: $\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & \color{red}{0} \end{bmatrix}$ B: $\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & \color{red}{1} \end{bmatrix}$ C: $\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & \color{red}{2} \end{bmatrix}$

Computing Homography

Conclusions:

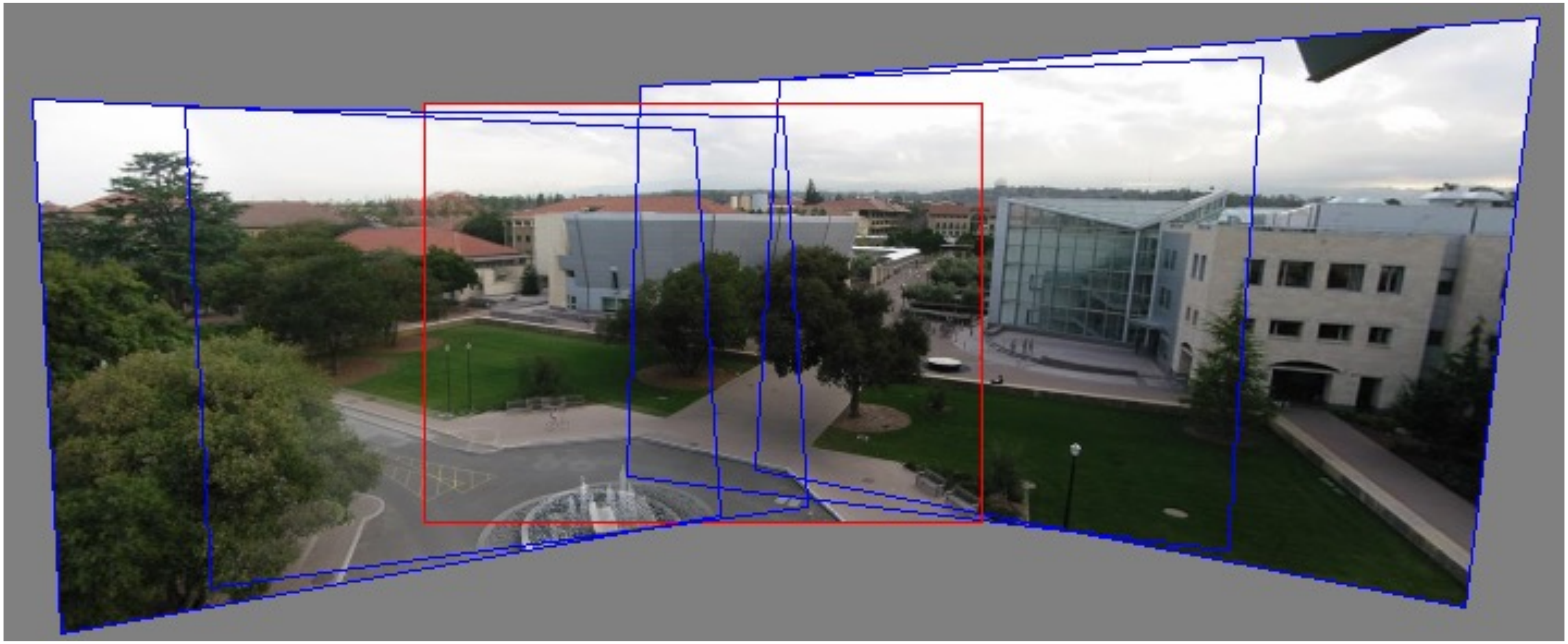
Assumption i=1 could be wrong

Assumption i=1 is equivalent to assumption $i \neq 0$
which makes it look significantly less dramatic

Instead, later we will use a completely safe additional constraint

$$\|H\| = 1$$

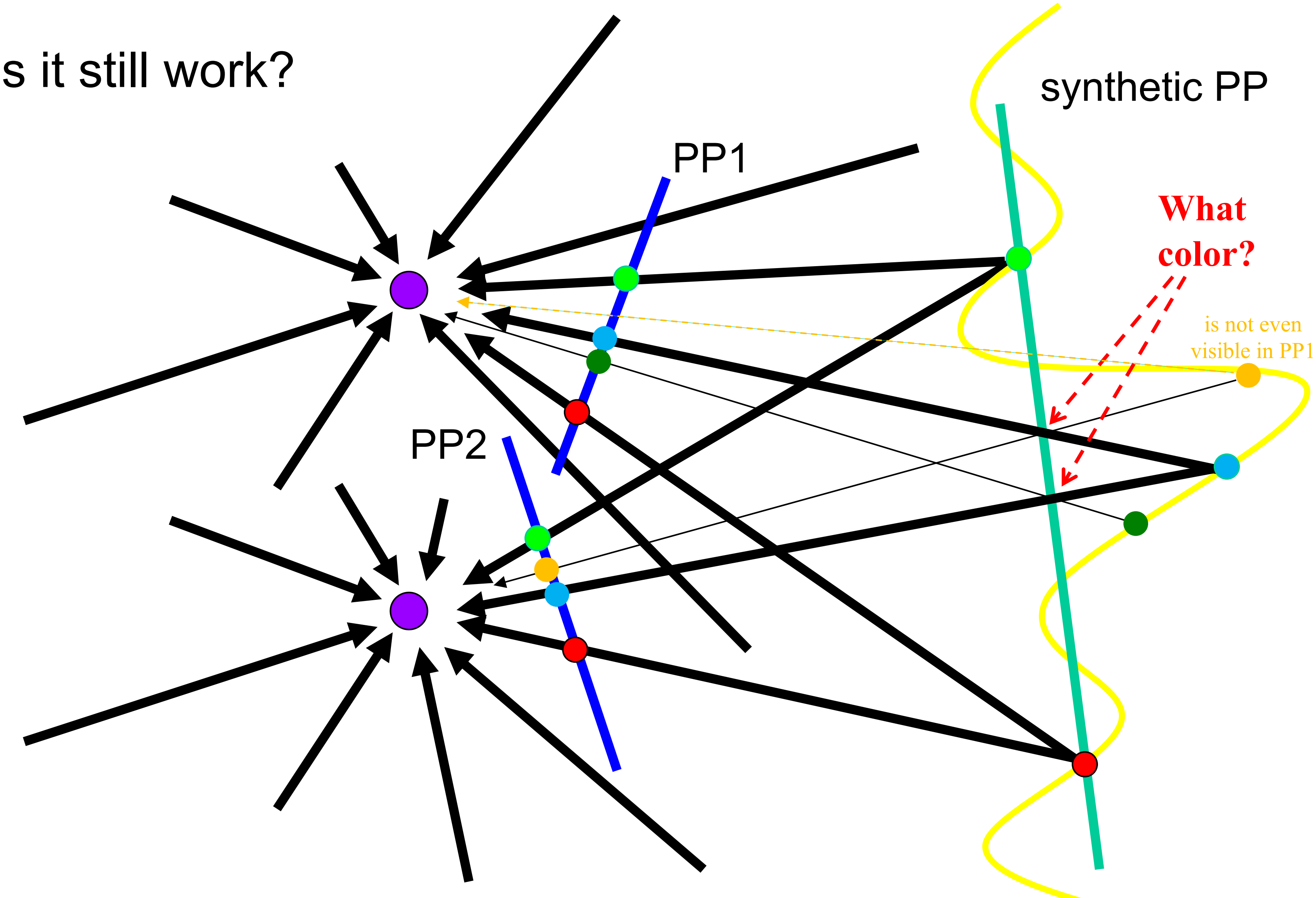
Panoramas



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

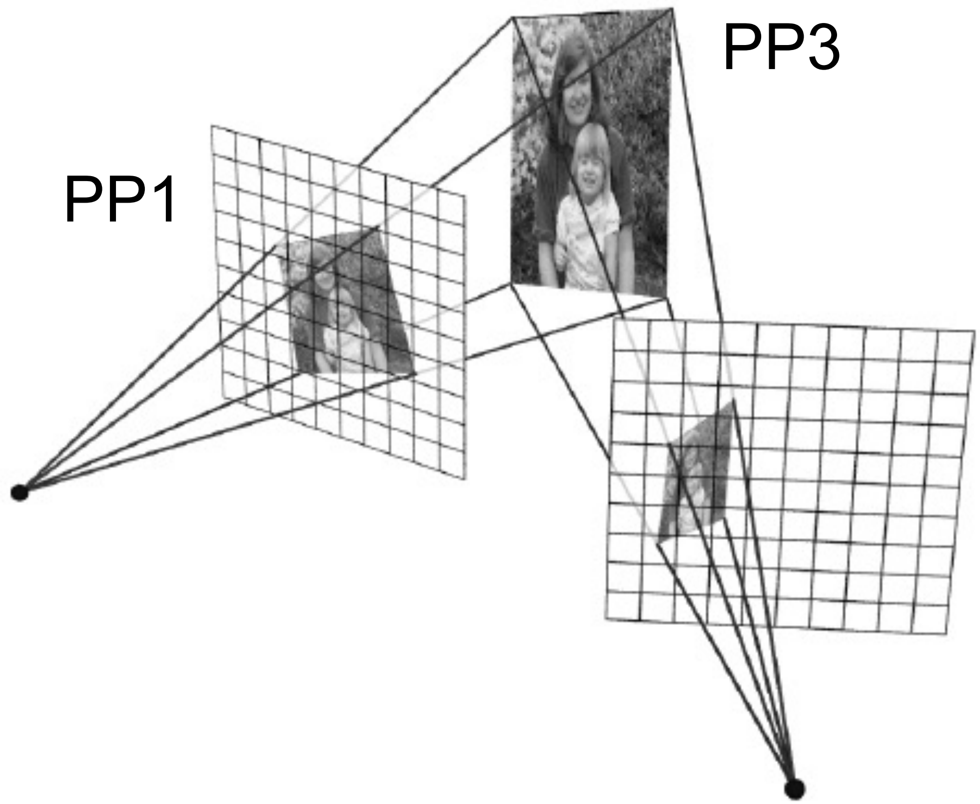
changing camera center

Does it still work?



ray correspondences no longer work for a common PP (for a **general scene**)

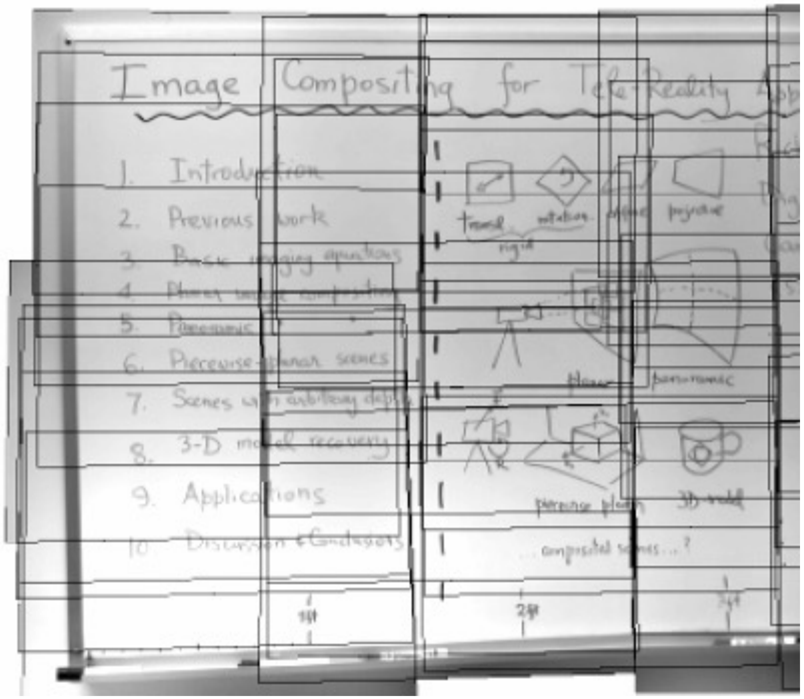
Planar scene (or far away)

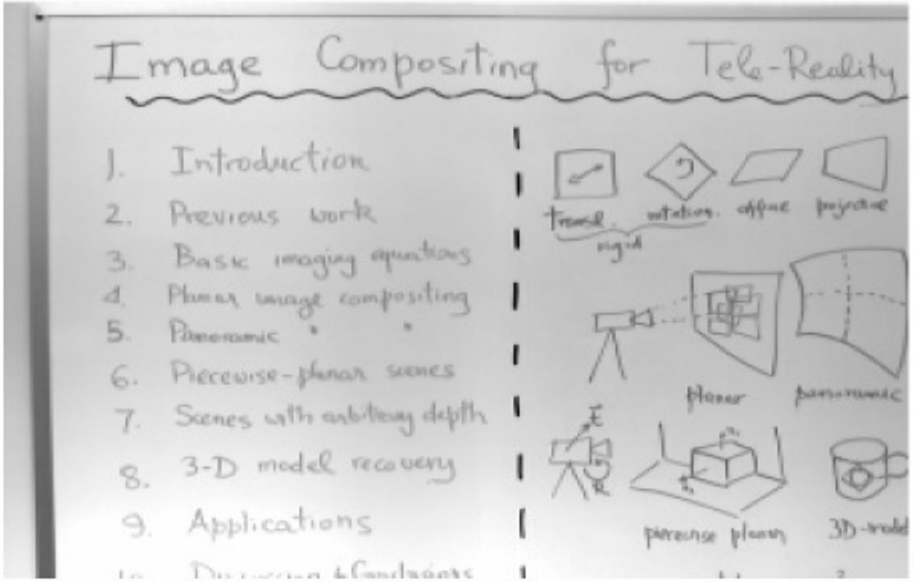


PP3 is a projection plane of both centers of projection, so we are OK!

This is how big aerial photographs are made

Planar mosaic

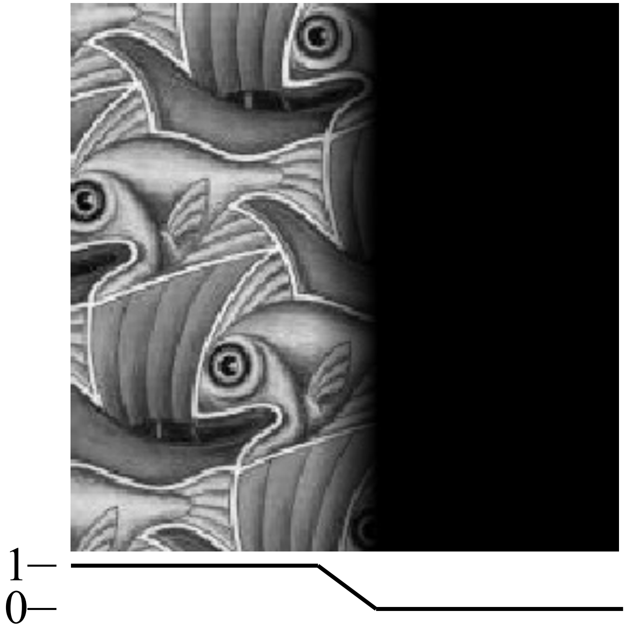




Blending the mosaic

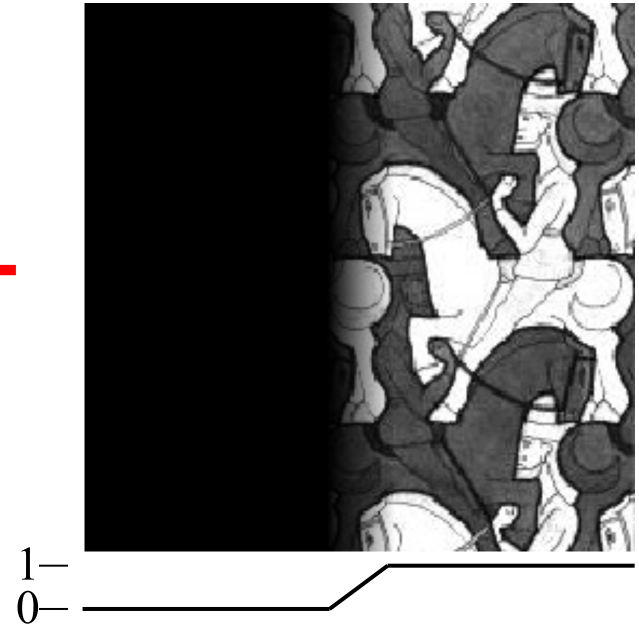
An example of image compositing:
the art (and sometime science) of
combining images together…

Feathering



α_{left}

$+$



α_{right}

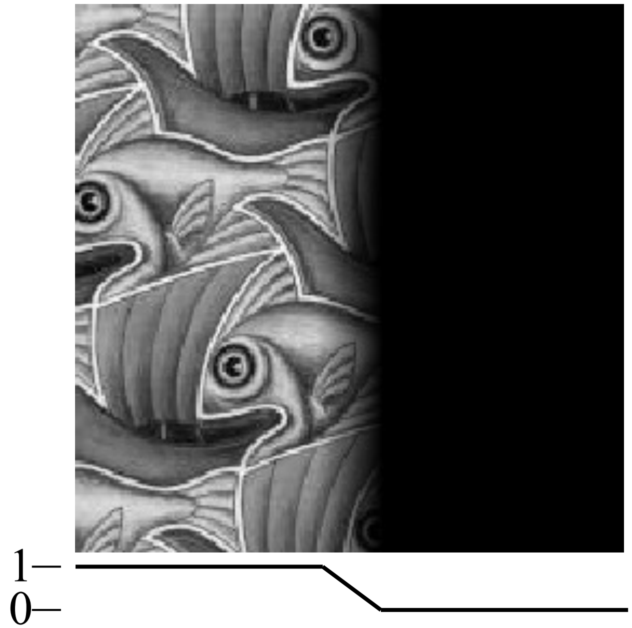
$=$

Encoding transparency via "alpha" channel

Normally, $\alpha_{\text{left}} + \alpha_{\text{right}} = 1$

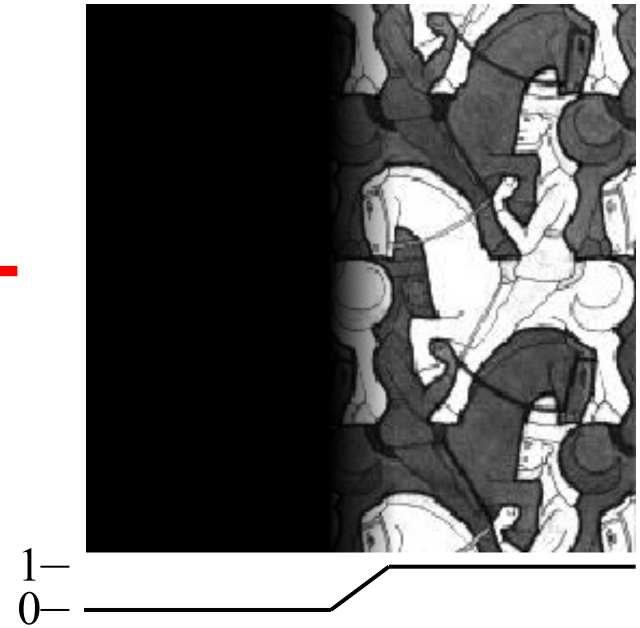
$$I_{\text{blend}} = \alpha_{\text{left}}\, I_{\text{left}} + \alpha_{\text{right}}\, I_{\text{right}}$$

Feathering



α_{left}

+



α_{right}

=

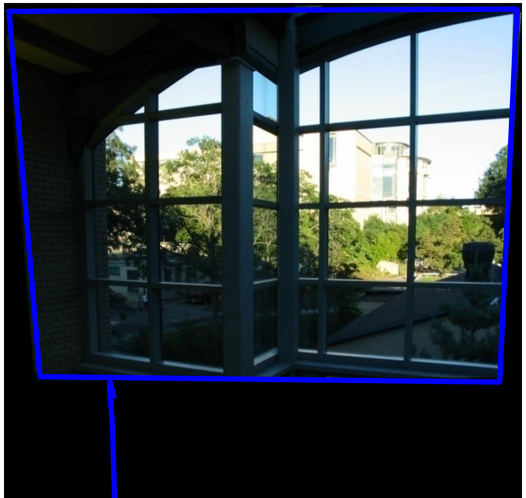
$$R_{\text{blend}} = \alpha_{\text{left}}\ R_{\text{left}} + \alpha_{\text{right}}\ R_{\text{right}}$$

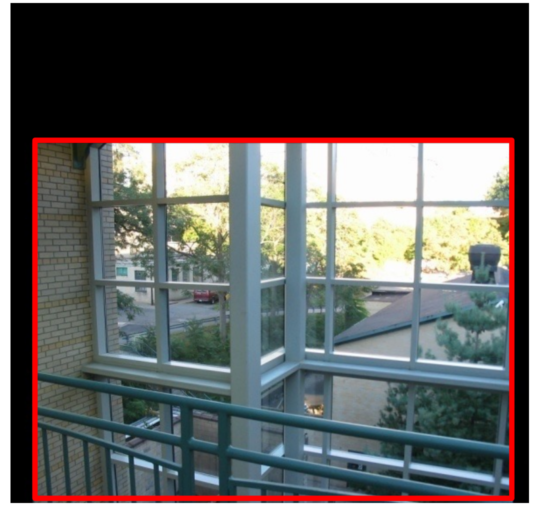
$$G_{\text{blend}} = \alpha_{\text{left}}\ G_{\text{left}} + \alpha_{\text{right}}\ G_{\text{right}}$$

$$B_{\text{blend}} = \alpha_{\text{left}}\ B_{\text{left}} + \alpha_{\text{right}}\ B_{\text{right}}$$

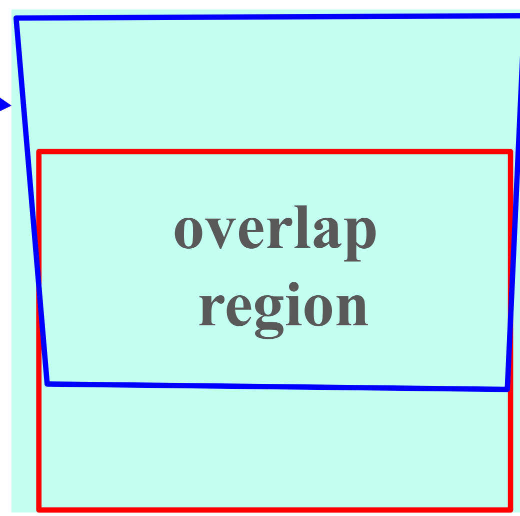
Assume images projected onto common PP

im 1 on common PP





im 2 on common PP



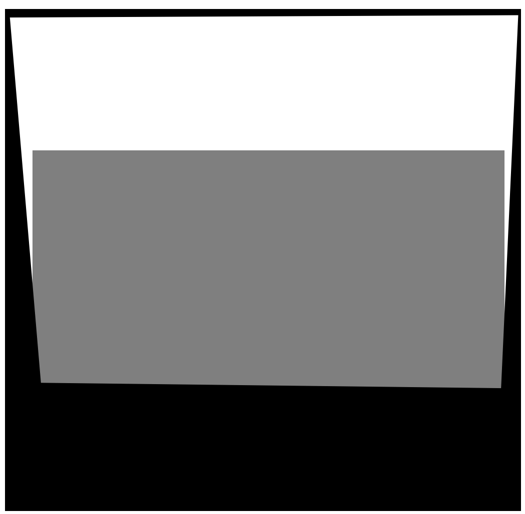
common PP

Setting alpha: simple averaging

im 1 on common PP

im 2 on common PP

alpha1 (for im 1)



alpha2 (for im 2)

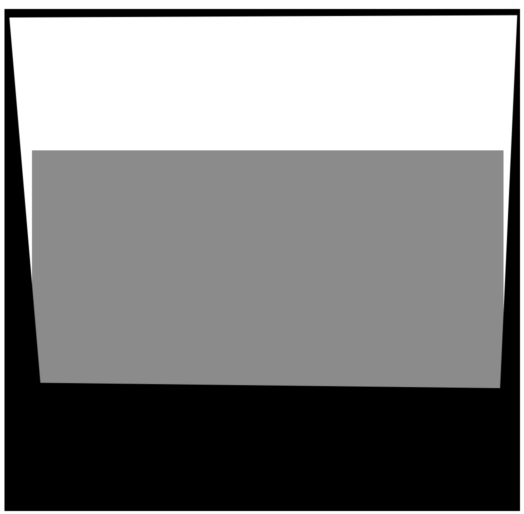
alpha = .5 in overlap region

Setting alpha: simple averaging

im 1 on common PP

im 2 on common PP

alpha1
(for im 1)

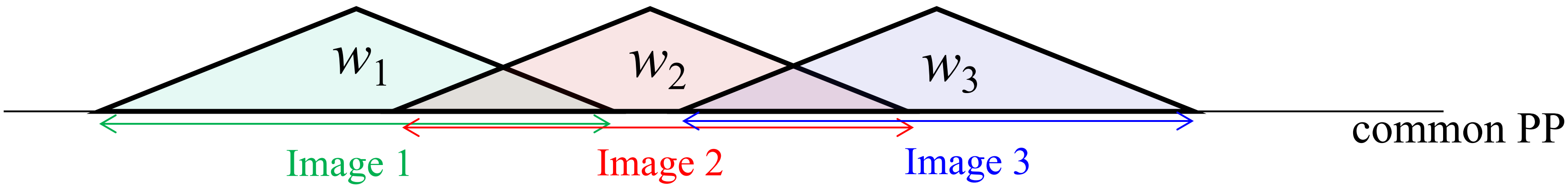


blended image

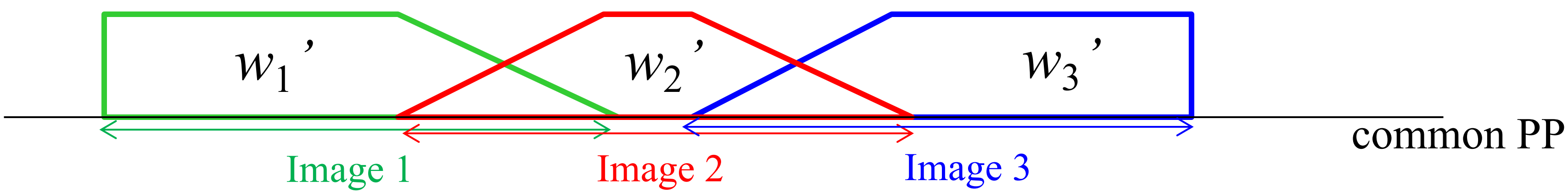
alpha = .5 in overlap region

Image feathering

Weight each image proportional to its distance from the edge (distance map [Danielsson, CVGIP 1980]



1. Generate *weight map* for each image (based on distance from edge)
2. **Normalize**: sum up all of the weights and divide by sum: weights sum up to 1: $w_i' = w_i \,/\, (\sum_i w_i)$

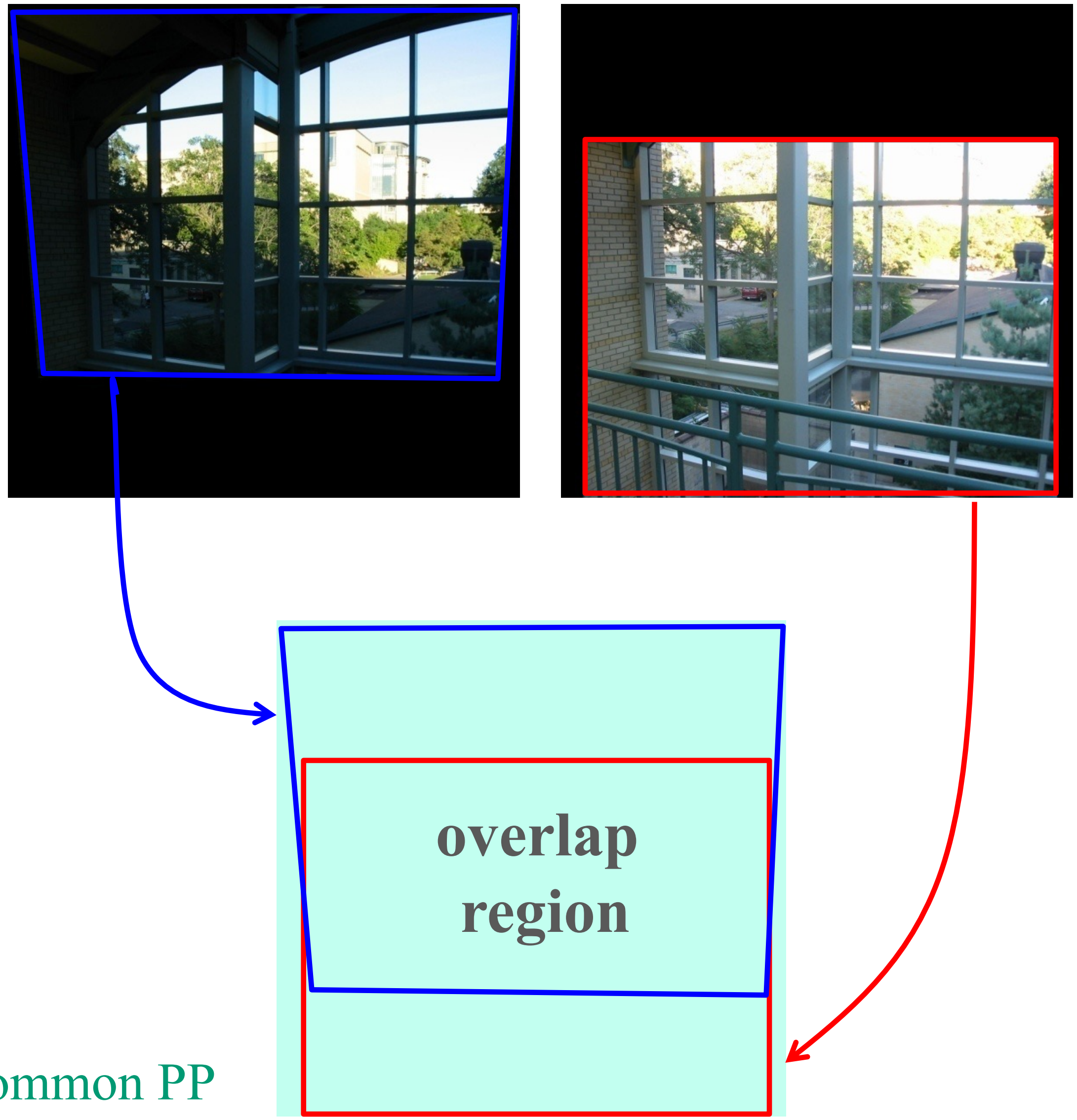


after normalization

(can be used as alphas)

Setting alpha:

im 1 on common PP



im 2 on common PP

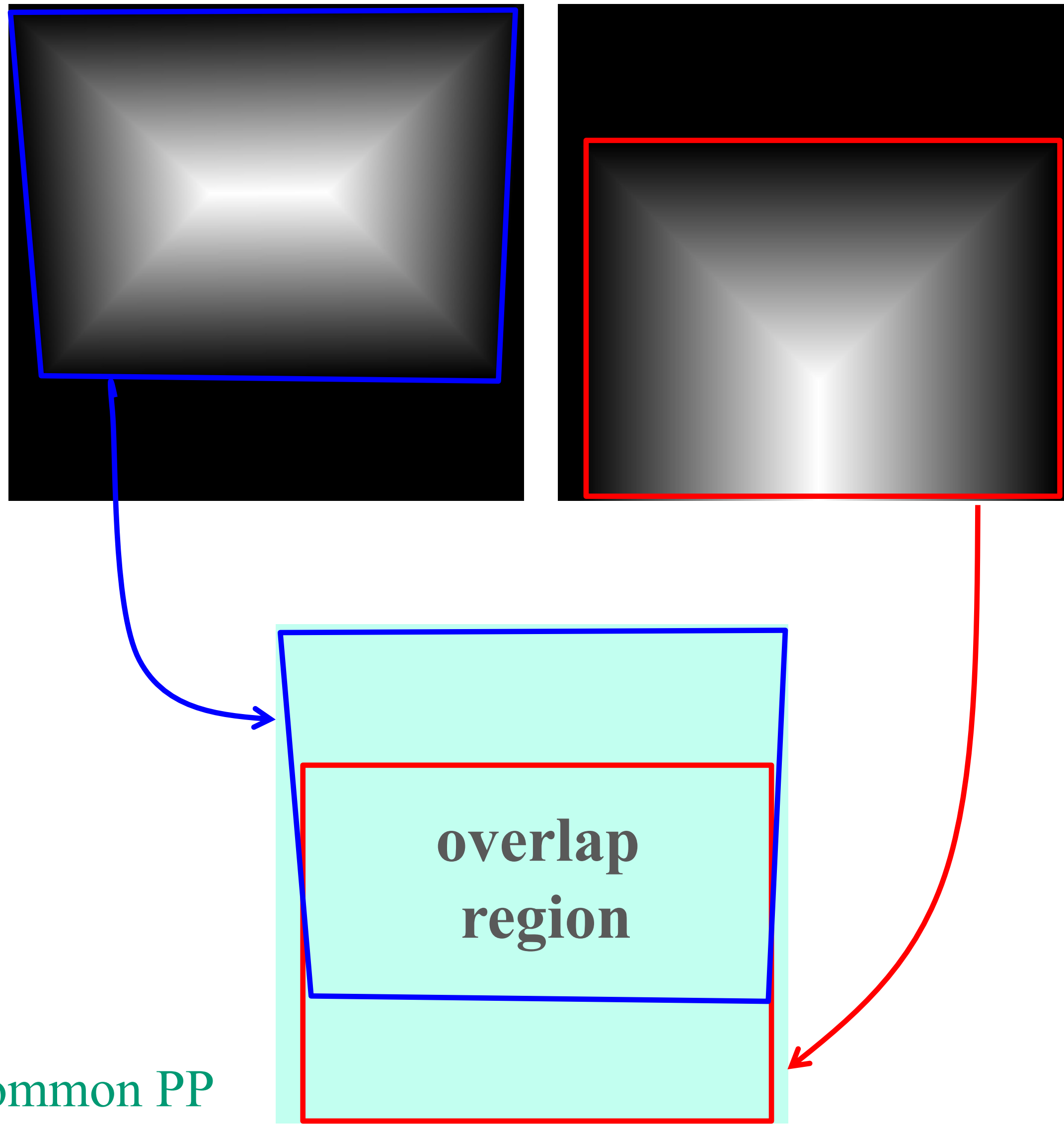
common PP

Setting alpha:

dtrans1

For simplicity, compute **distances from rectangular borders in the original image plane** (which is trivial) and project onto common PP by applying the same homography used for mapping the image

dtrans2



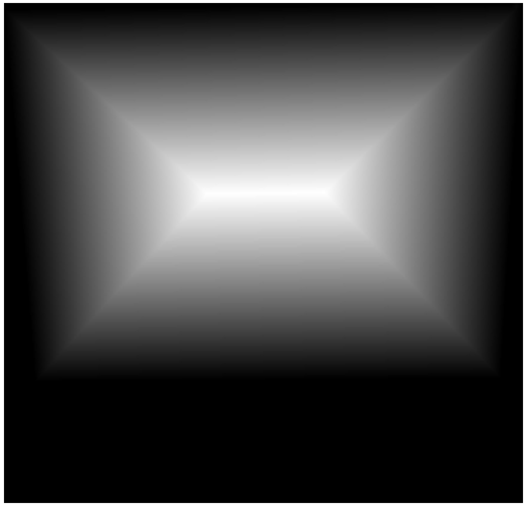
common PP

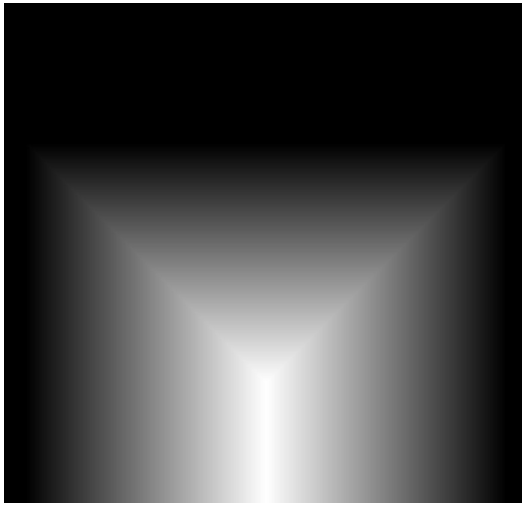
Distance Transforms

(distance from "borders")

Setting alpha: center seam

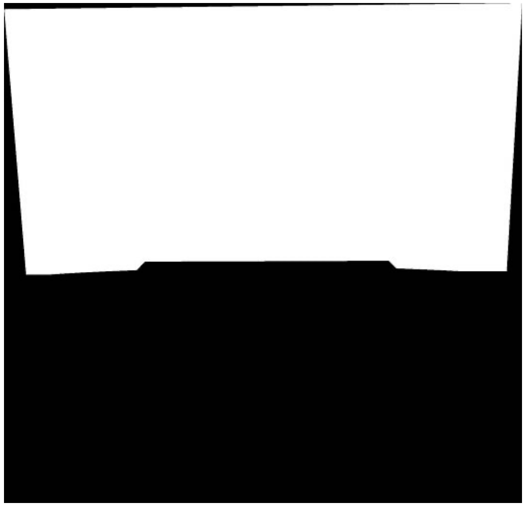
dtrans1





dtrans2

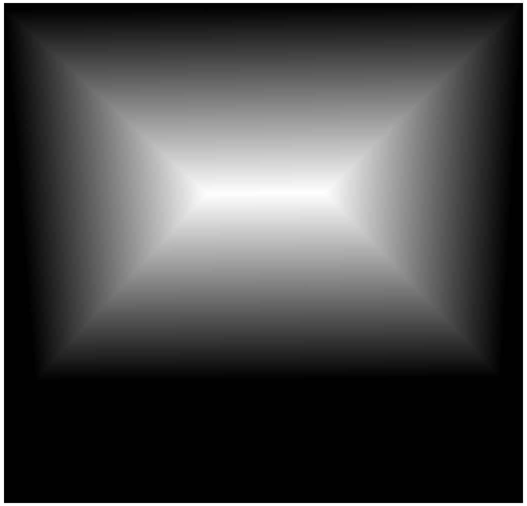
alpha1
(for im 1)

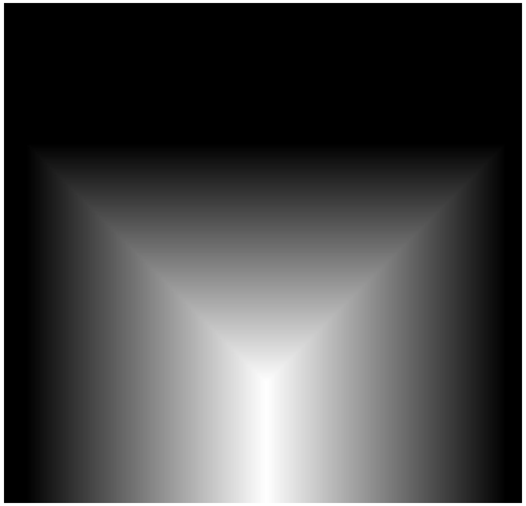


alpha1 = logical(dtrans1>dtrans2)
alpha2 = logical(dtrans2>dtrans1)

Setting alpha: blurred seam

dtrans1





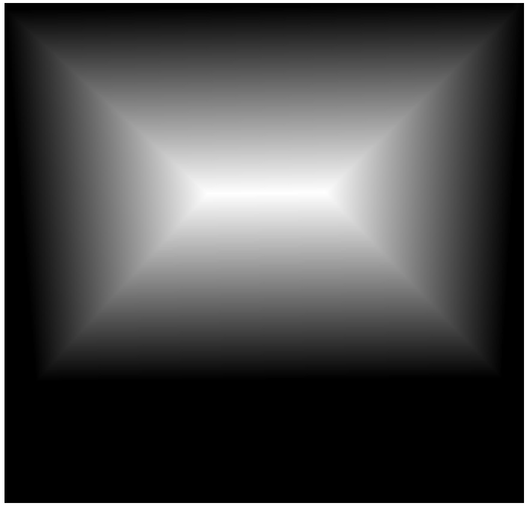
dtrans2

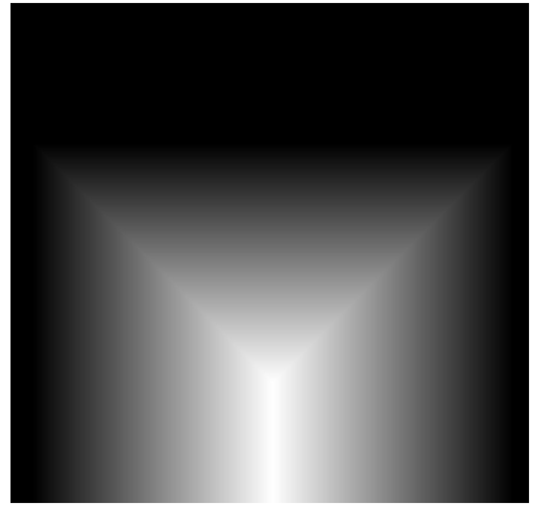
alpha1
(for im 1)

alpha = blurred

Setting alpha: center weighting

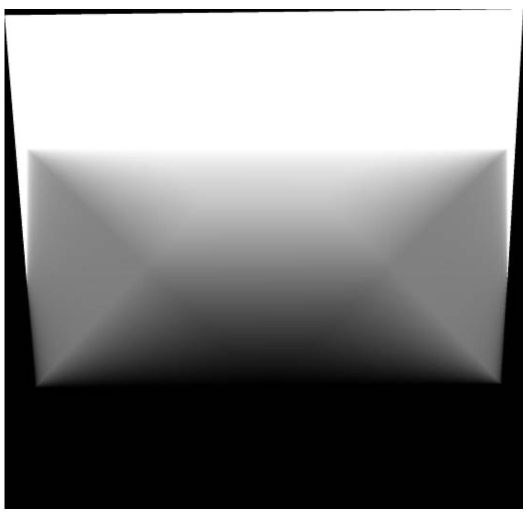
dtrans1

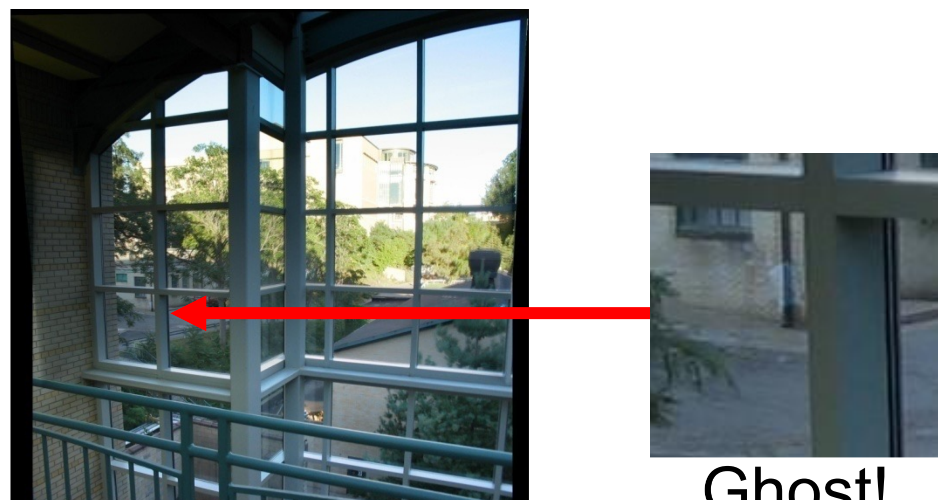




dtrans2

alpha1
(for im 1)





Ghost!

alpha = dtrans1 / (dtrans1+dtrans2)

Assignment 2

Homographies and Panoramic Mosaics

- Compute homographies (define correspondences)
  - The next topic shows how to match points automatically while estimating a homography (RANSAC)
- Warp images projecting onto common PP
- Produce panoramic mosaic on common PP via blending

Fun with Homographies

Blending and Compositing

- use homographies to combine images or video and images together in an interesting (fun) way. E.g.
  - put fake graffiti on buildings or chalk drawings on the ground
  - replace a road sign with your own poster
  - project a movie onto a building wall
  - etc.

3D Sidewalk Art by Julian Beever (see links for more)

Fun with Homographies

3D Sidewalk Art by Edgar Müller

Fun with Homographies

3D Sidewalk Art by Edgar Müller

360 panorama

a bit trickier… projecting all images onto a common "reference" cylinder or sphere, rather than a plane

NOTE: ray correspondences define **image warps onto a common "projection cylinder" or common "projection sphere"**, but these warps are not homographies (lines are not preserved)

Video Panorama

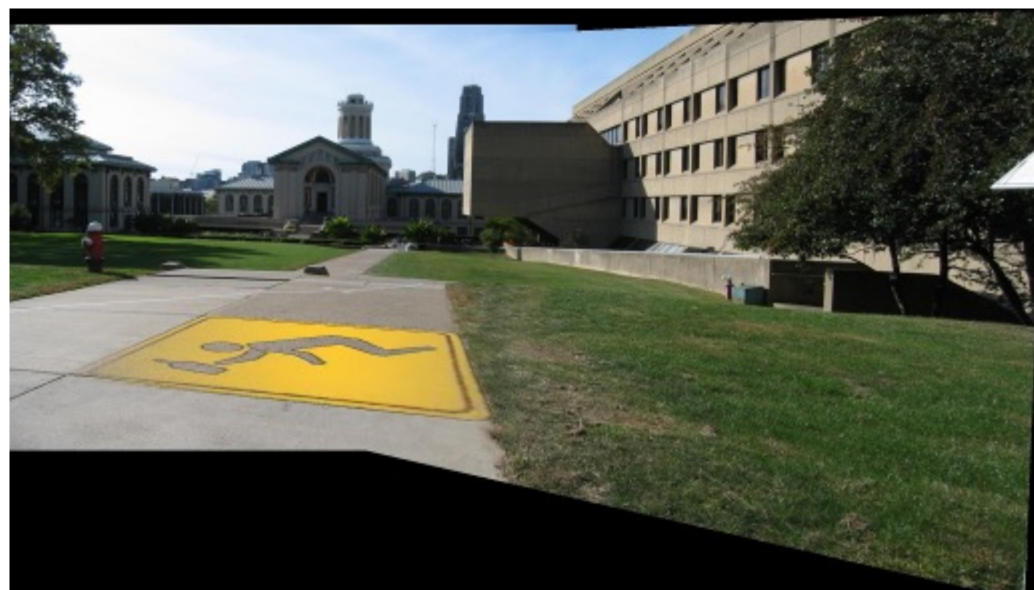
- Capture two (or more) stationary videos (either from the same point, or of a planar/far-away scene). Compute homography and produce a video mosaic. Need to worry about synchronization (not too hard).
- e.g. capturing a football game from the sides of the stadium

From CMU students' projects

Ben Hollis, 2004

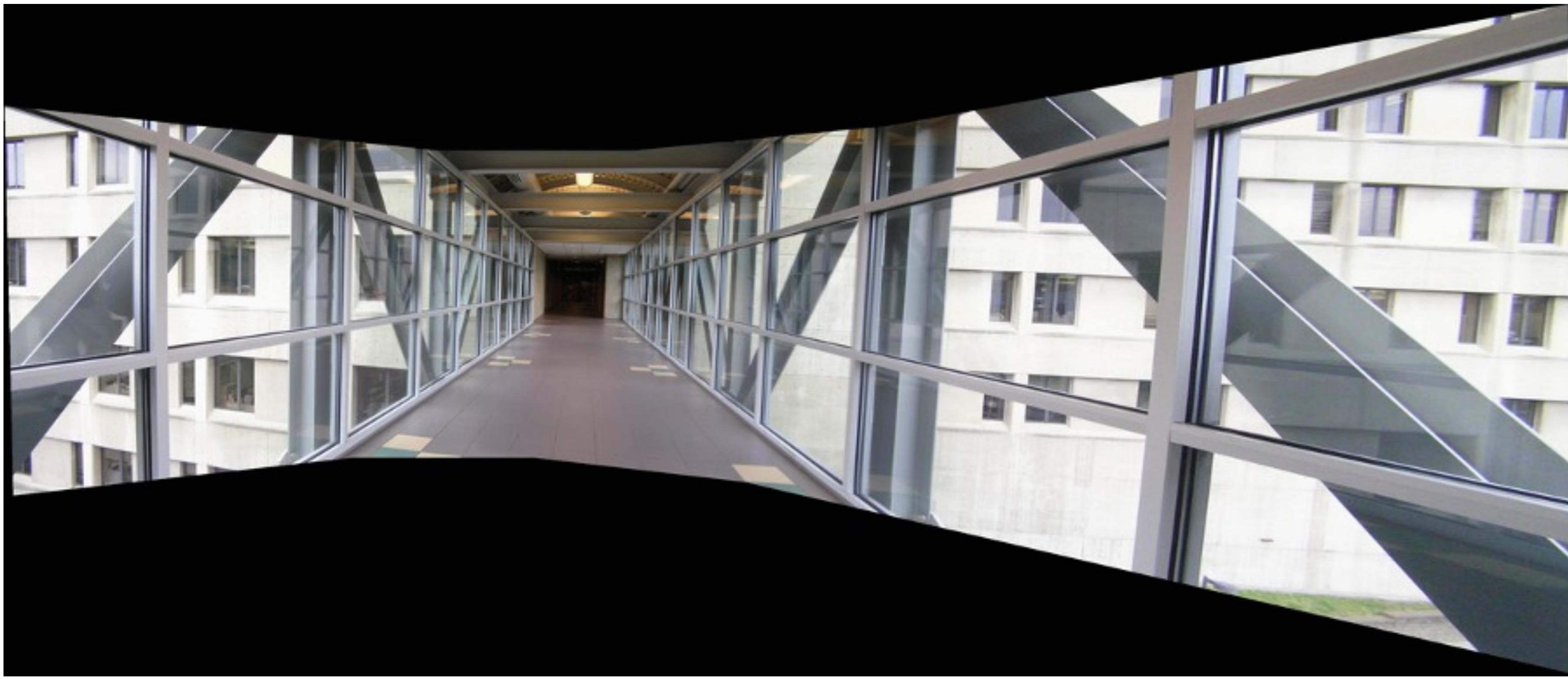
Ben Hollis, 2004

Matt Pucevich , 2004



Eunjeong Ryu (E.J), 2004

From CMU students' projects



Ken Chu, 2004