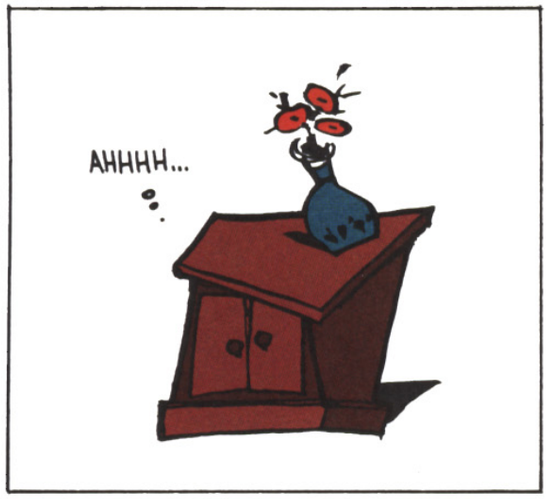
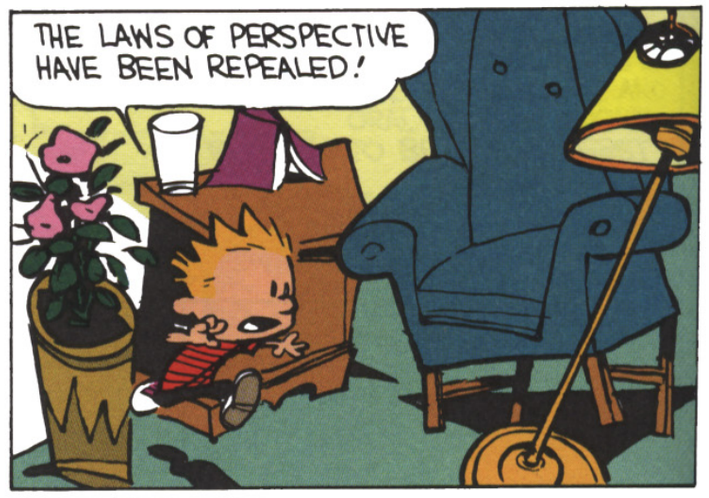
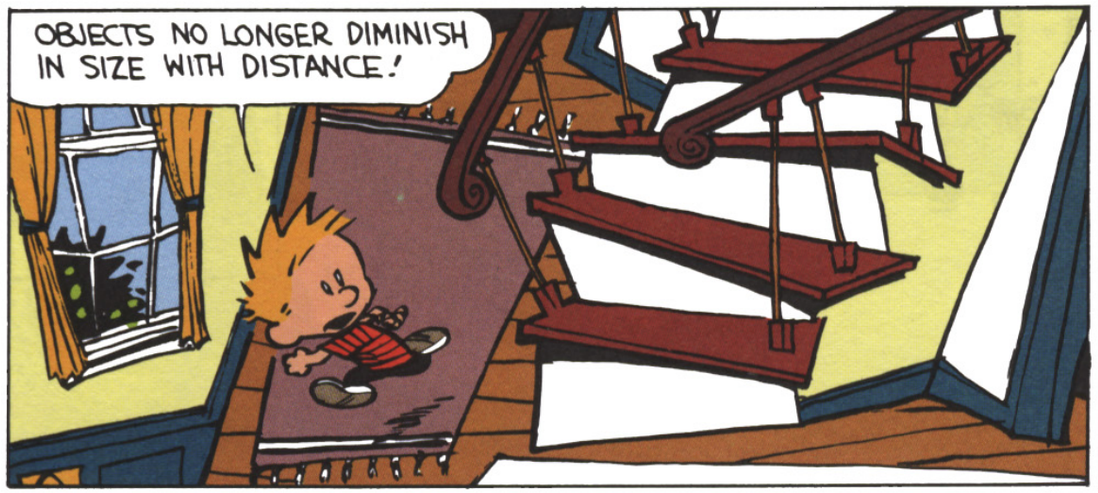
Panoramas

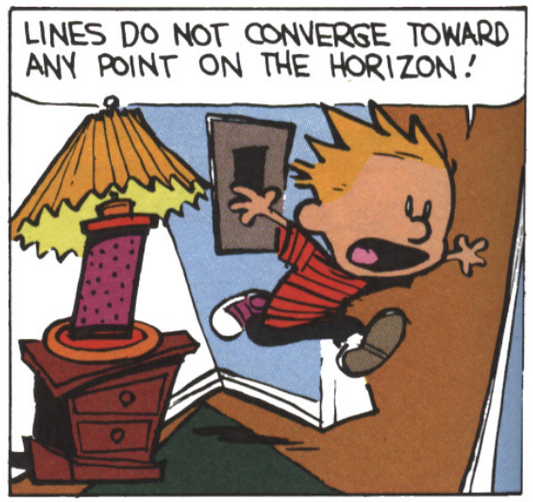

calvin
and
HOBBES
by WATTERSON

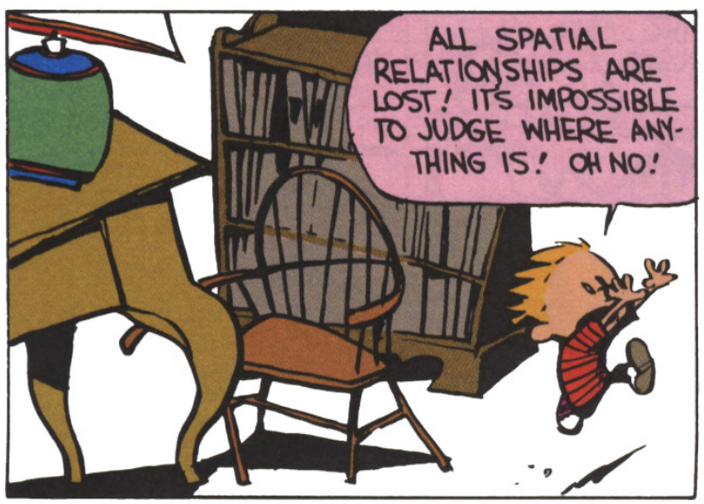

AHHHH...


UH-OH. SOMETHING IS SERIOUSLY WRONG HERE.


THE LAWS OF PERSPECTIVE HAVE BEEN REPEALED!


OBJECTS NO LONGER DIMINISH IN SIZE WITH DISTANCE!


LINES DO NOT CONVERGE TOWARD ANY POINT ON THE HORIZON!


ALL SPATIAL RELATIONSHIPS ARE LOST! IT'S IMPOSSIBLE TO JUDGE WHERE ANYTHING IS! OH NO!


CALVIN, QUIT RUNNING AROUND AND CRASHING INTO THINGS, OR I'LL SELL YOU TO THE MONKEY HOUSE!


...AND NOW SHE'S LOST PERSPECTIVE.

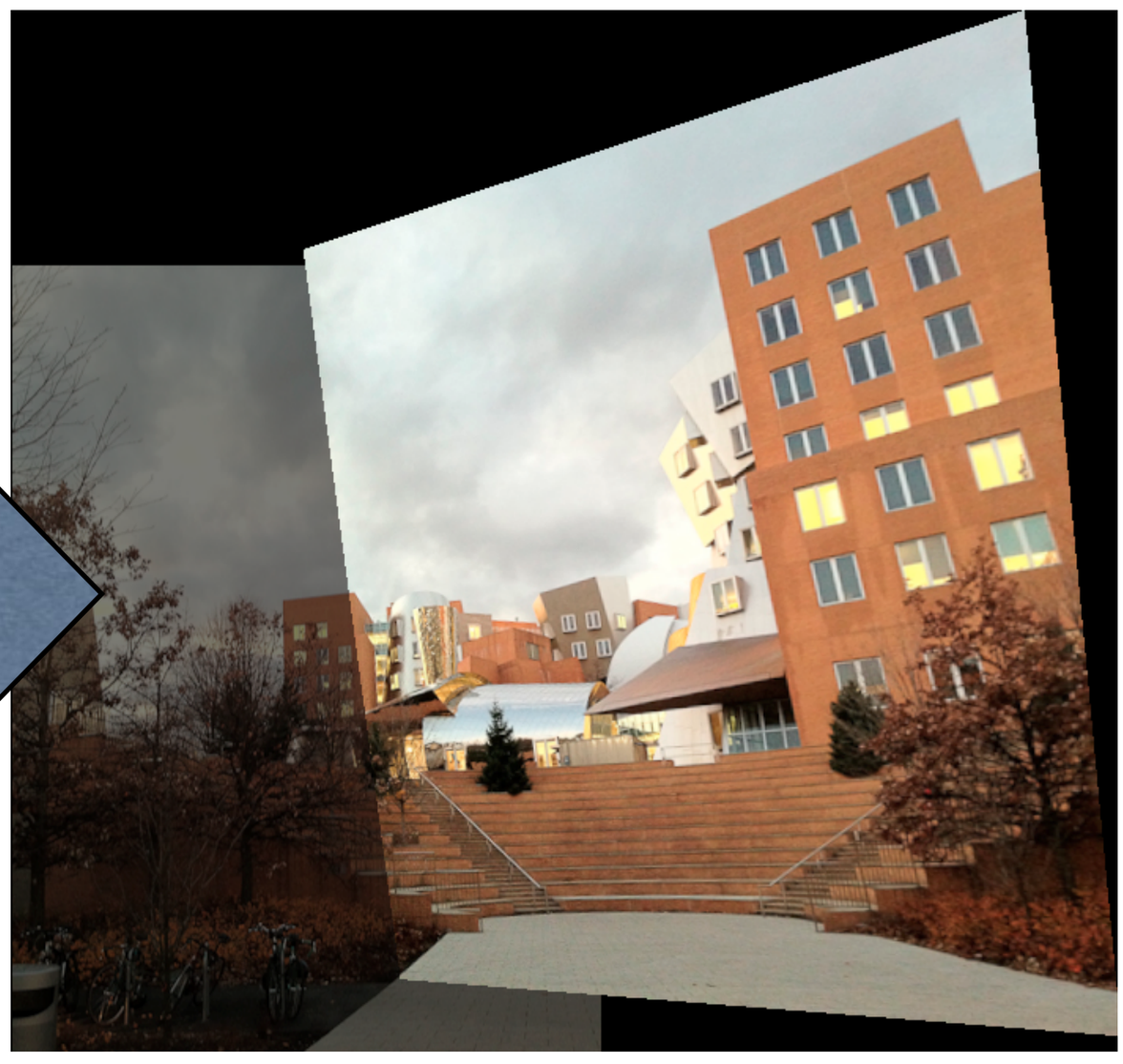
Overview & Recap

Today

- The user gives us 4 correspondences
- We reproject one image to match the other one
- Creates a wider angle view







Later

- Automagic correspondences
  - corner detection
  - patch descriptor

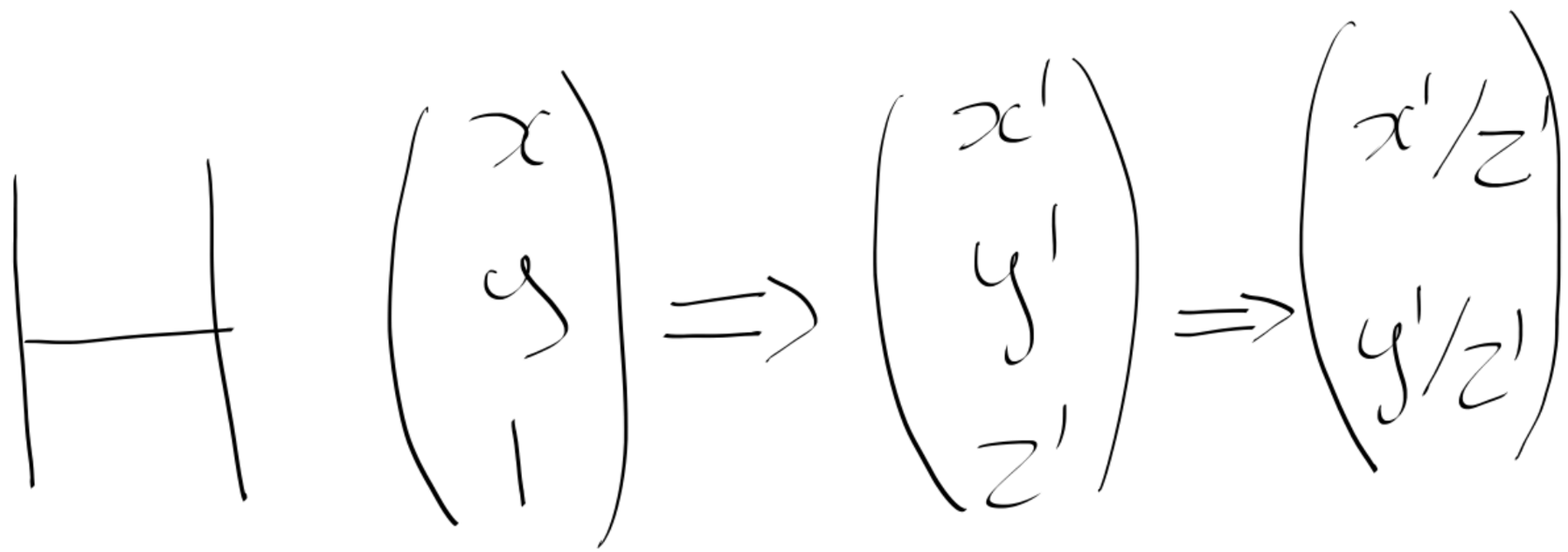


- Nice blending
  - smooth transition
  - 2 scale



From last time

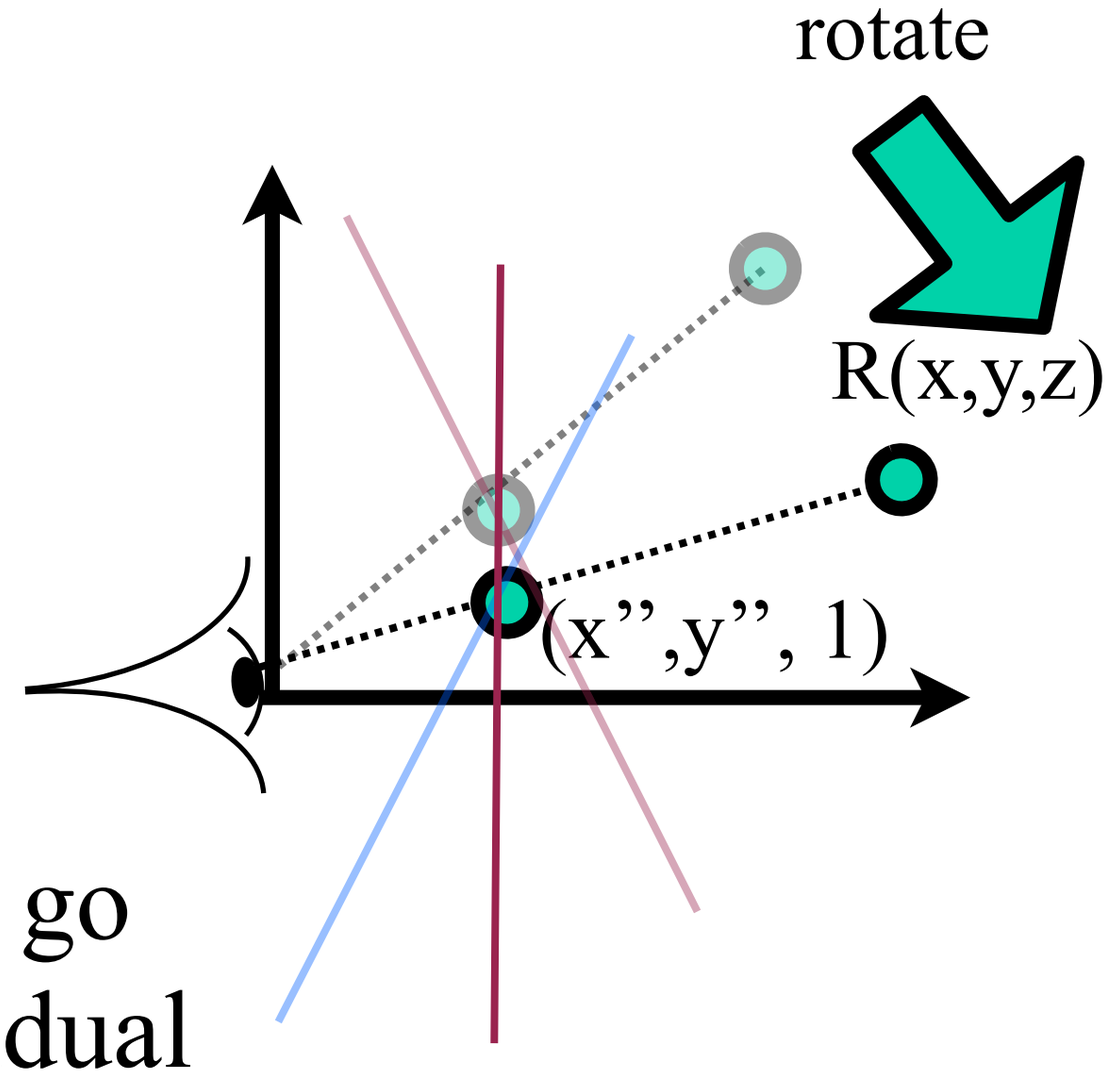
- **The transformation from one 2D image to a rotated image of the same scene is a 3x3 matrix H applied to x,y,1 followed by a division by z'**
  - called a homography



- **Now the only problem is that we usually don't know H**

3D rotation: Pano case

- Each image has a different rotation Ri
- project (x,y,z) wrt camera rotated by Ri:
  - Apply rotation Ri^{-1} to (x,y, z)
  - Apply projection division



- Now we understand how to go from the 3D world to individual pano input images
  - We now need to go directly from 2D input to another one

Recap

- canonical projection = division by z
- For other direction, just apply 3D rotation first

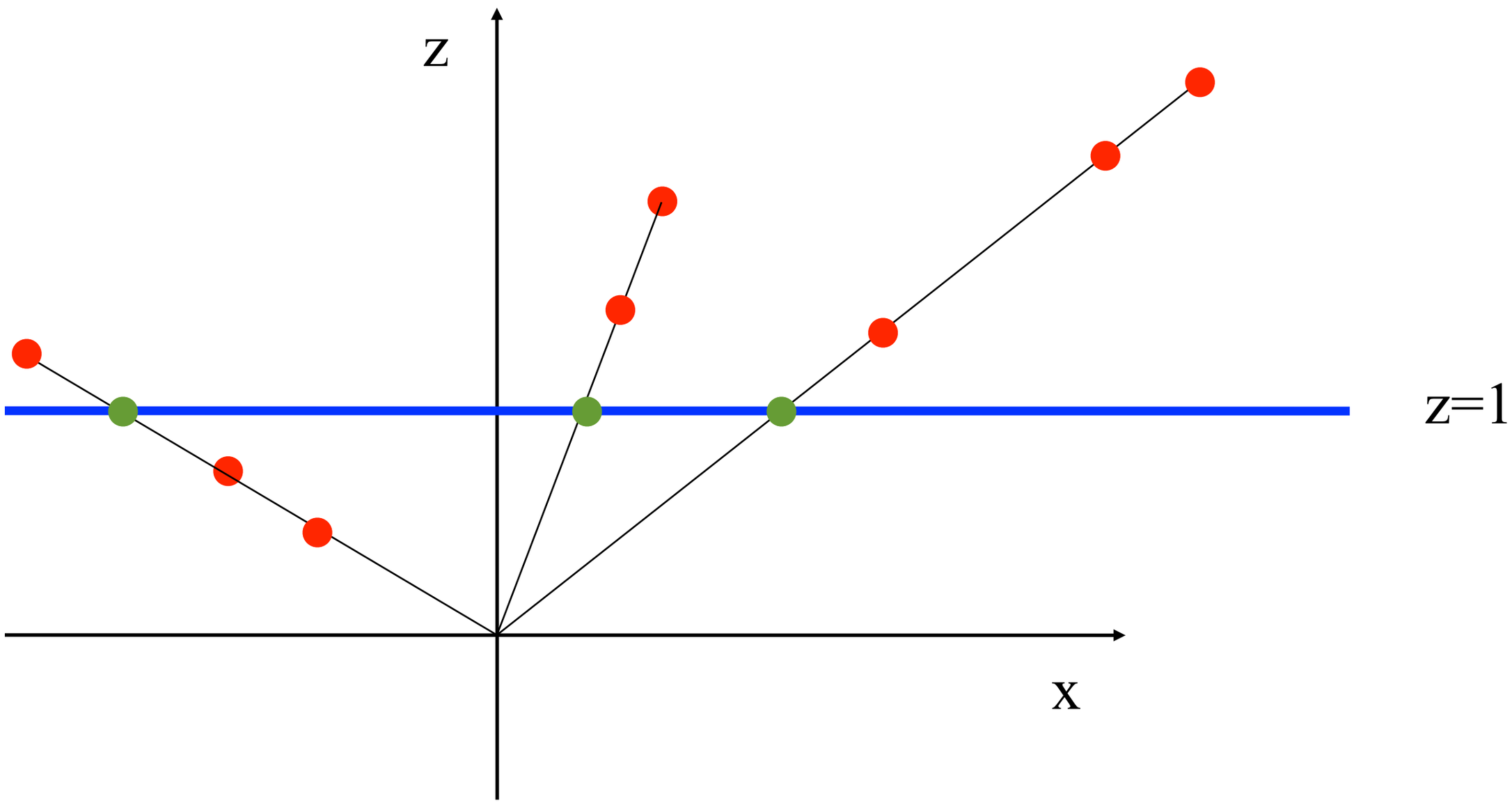
- But... this applies only when we know *z*
  - And for panorama stitching, we don't
  - What are we going to do?Are we in big trouble? Should we give up? Buy a 3D scanner? Cancel assignment 6?

Questions?

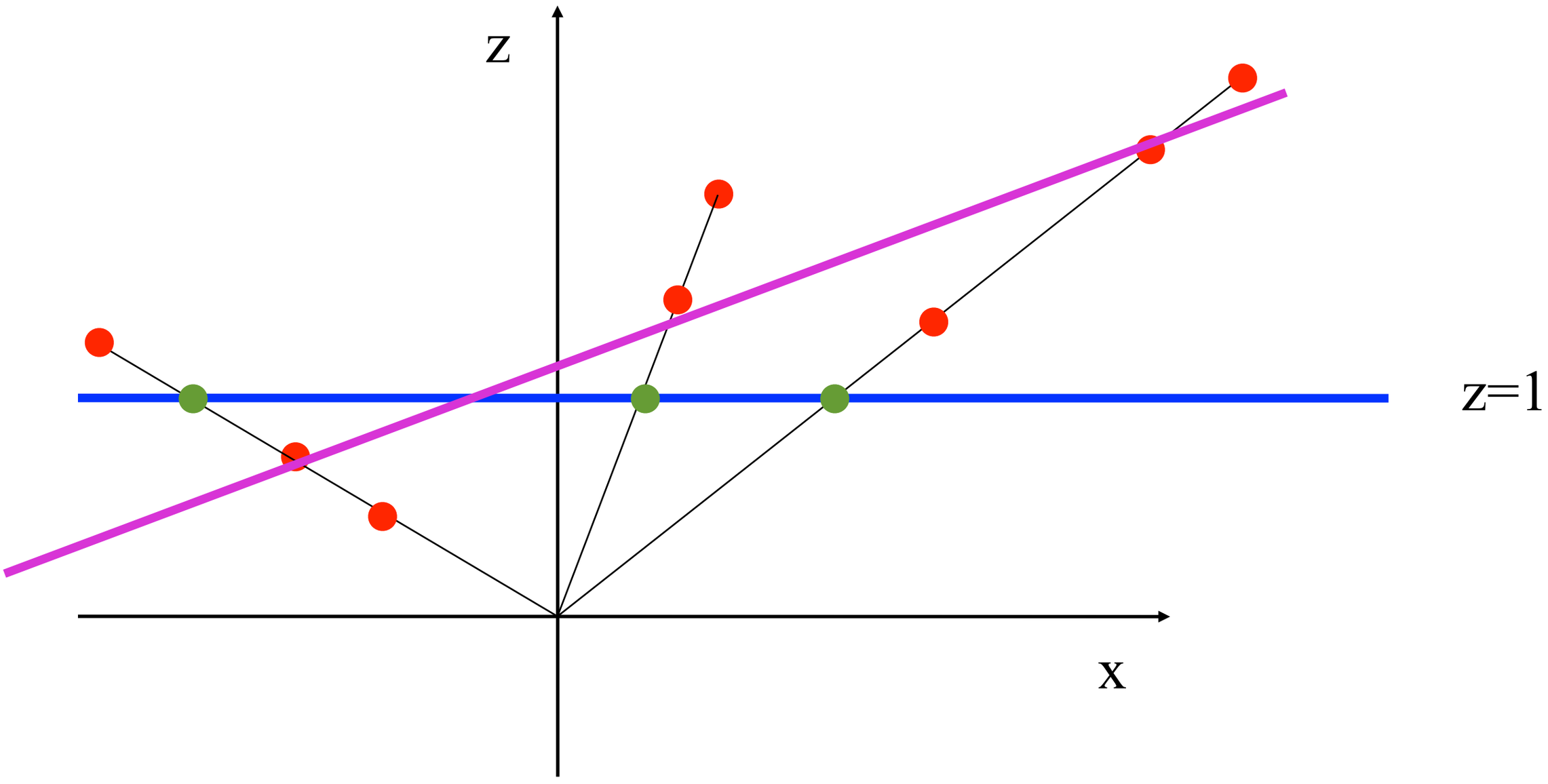


Unknown z

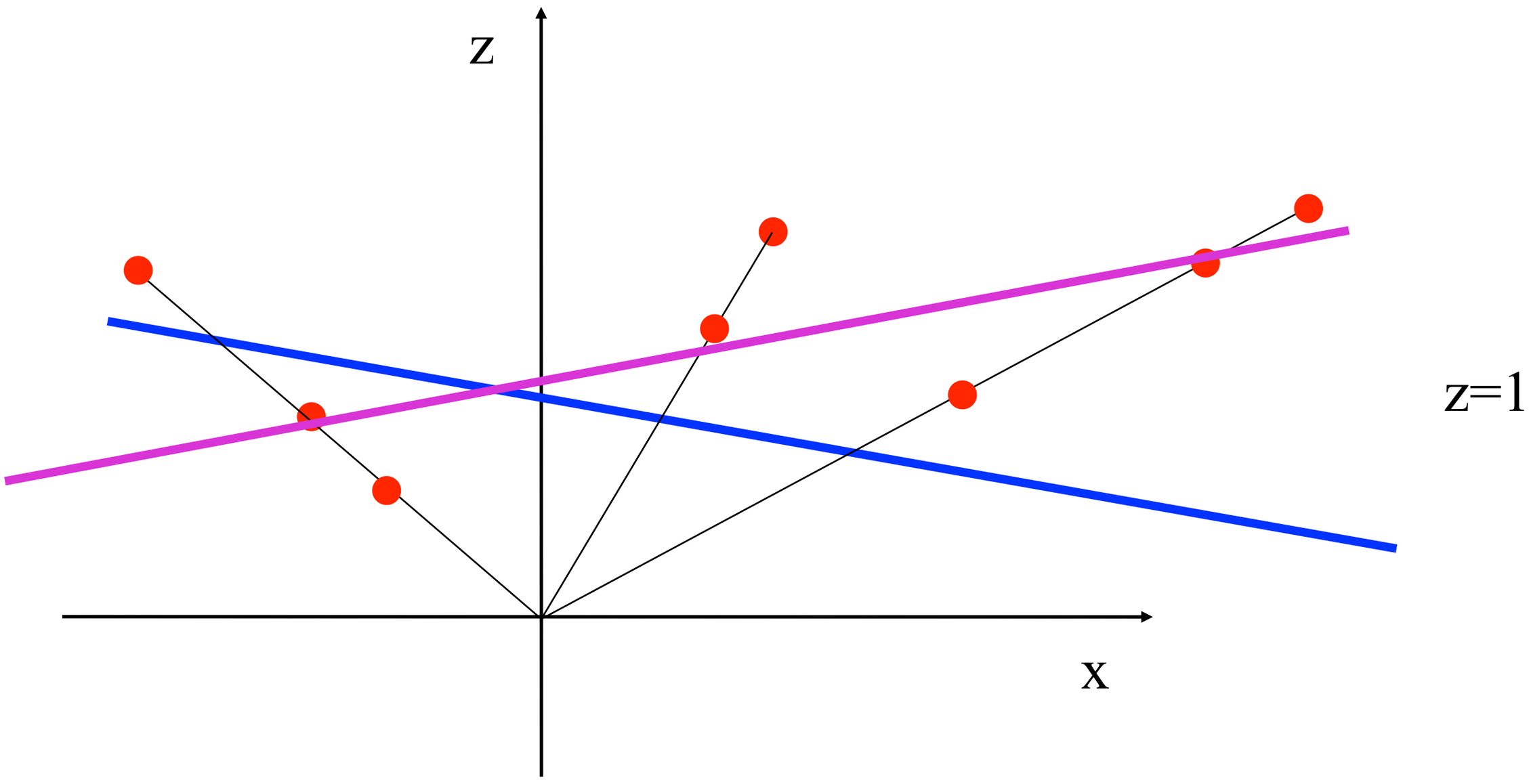
Points that project on the same x,y



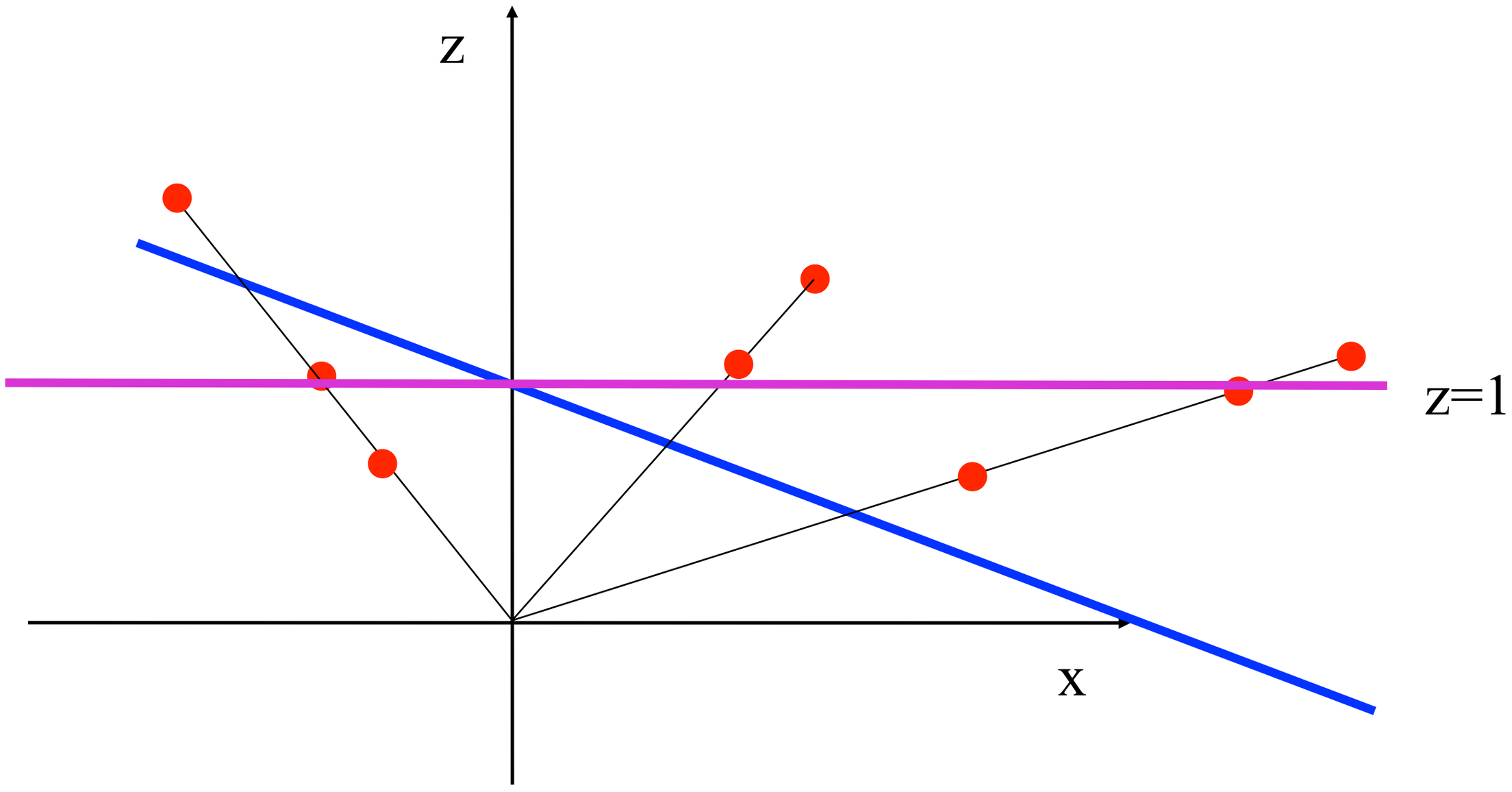
Project to different image?



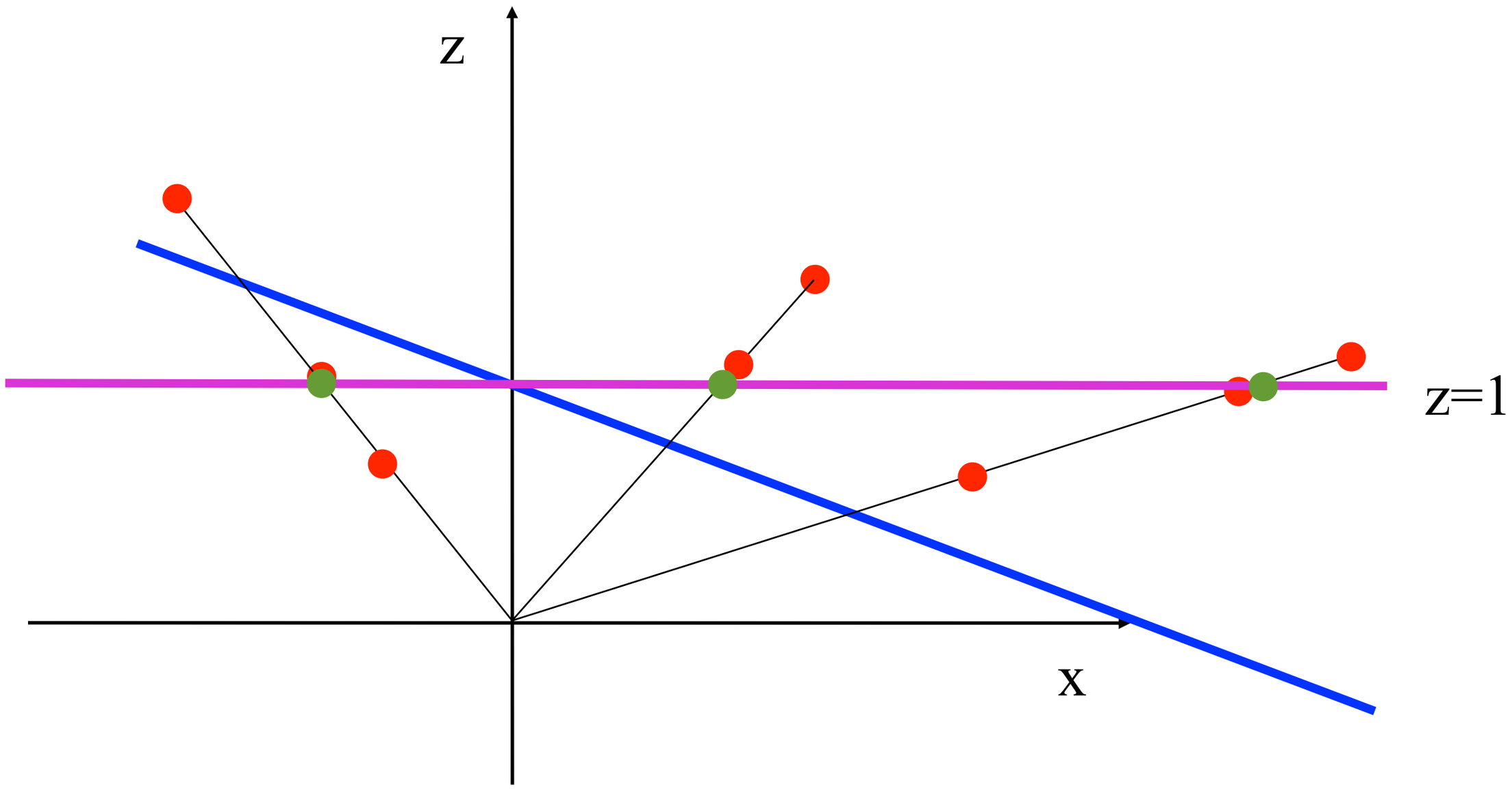
Rotate



Rotate

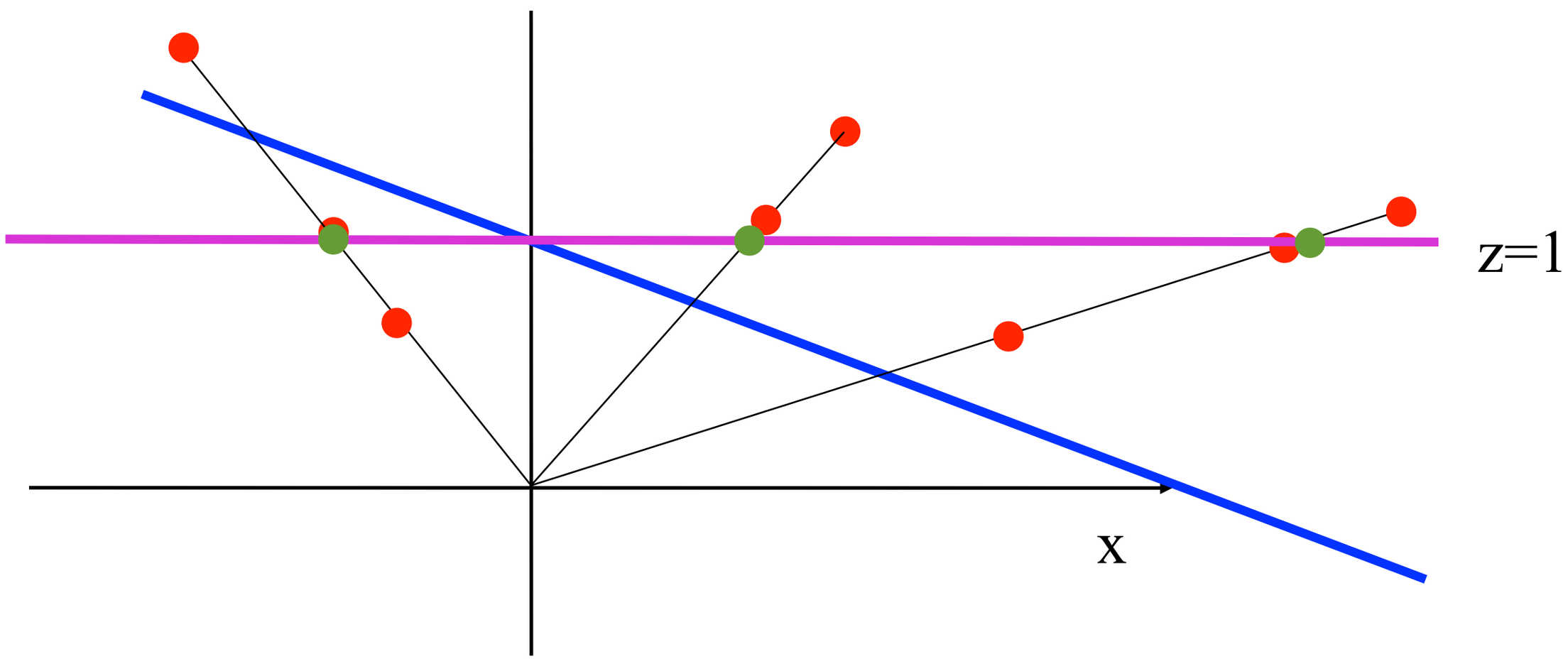


Divide by z



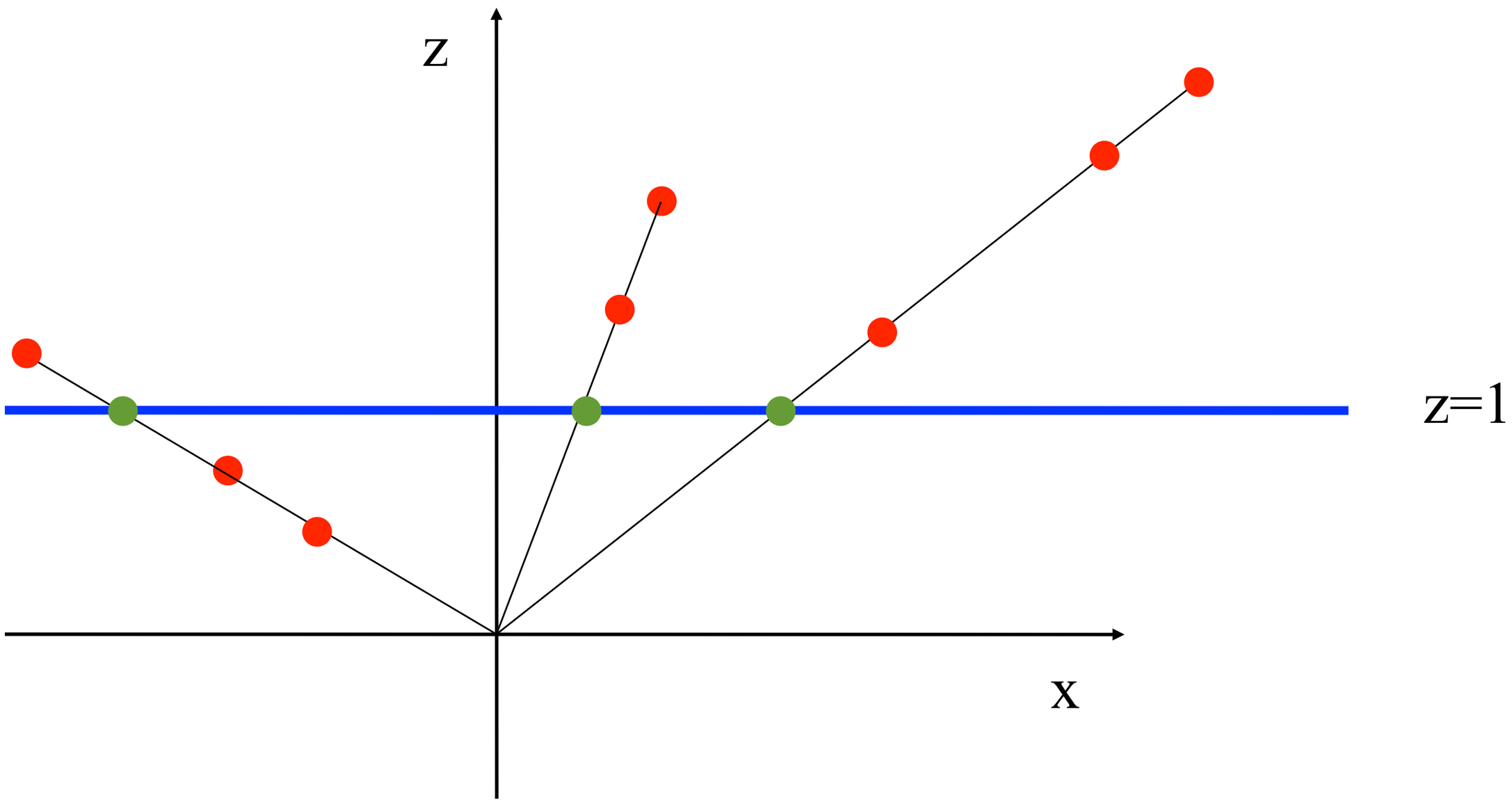
Project to different image

- **It doesn't depend on z**
- **All points that project to the same x,y reproject onto the same x',y'**
  - after perspective division



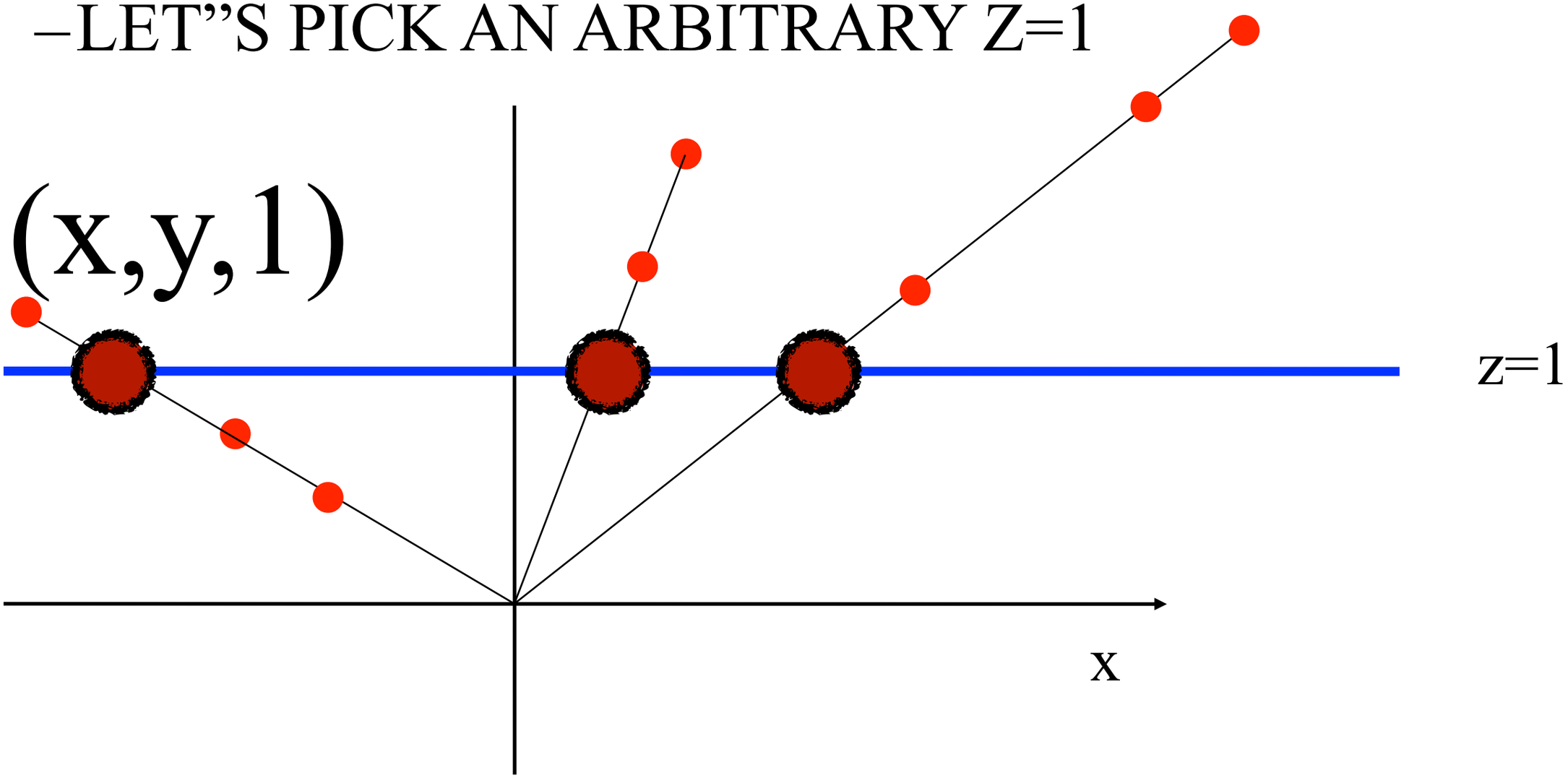
Back to pano problem

- **We want to know how points at x,y in the first image get reprojected onto the rotated second image**
  - problem: we don't know their z

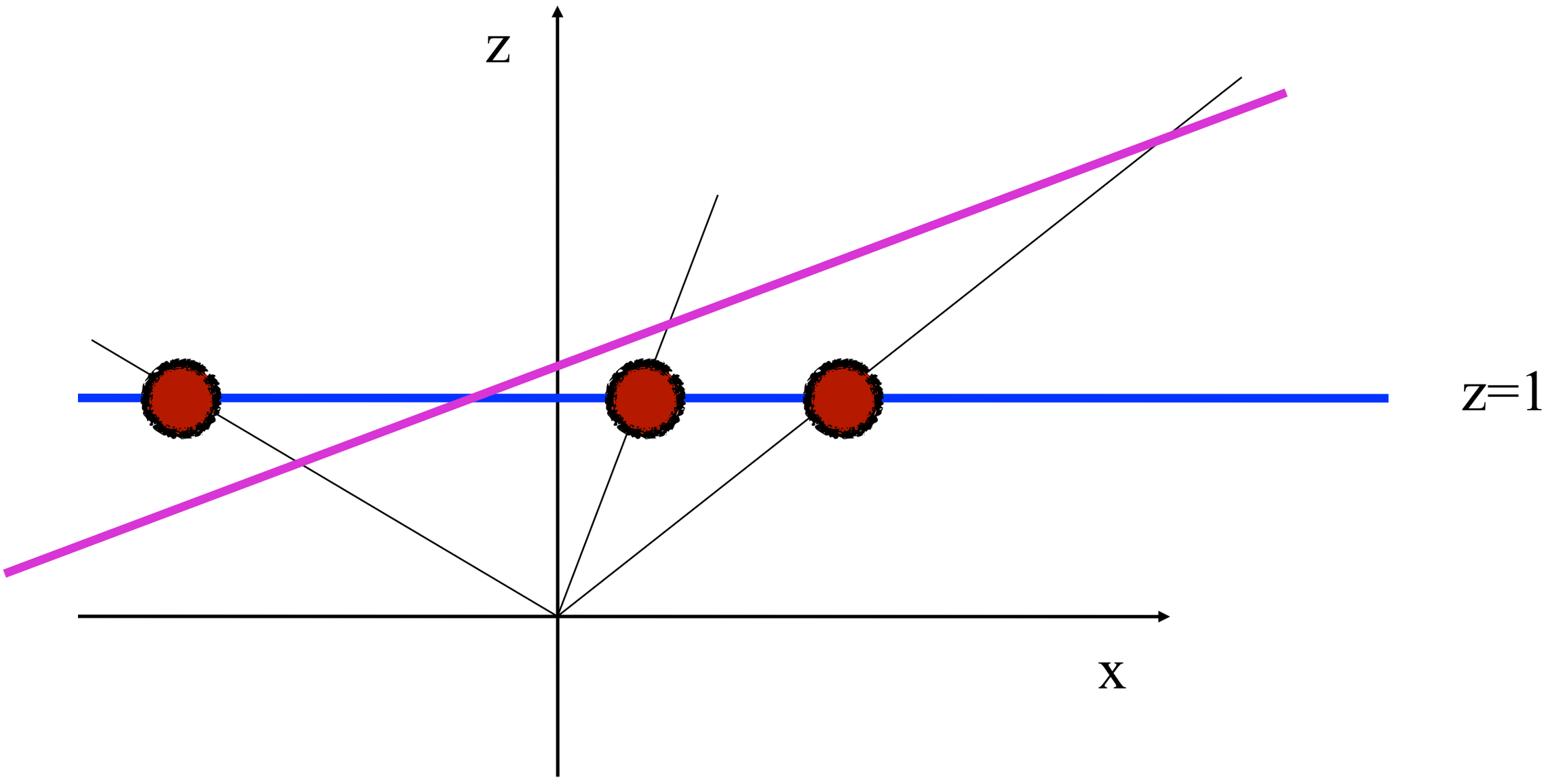


Back to pano problem

- **We want to know how points at x,y in the first image get reprojected onto the rotated second image**
  - problem: we don't know their z
  - BUT ID DOESN'T DEPEND ON Z!!!!!!
  - LET"S PICK AN ARBITRARY Z=1

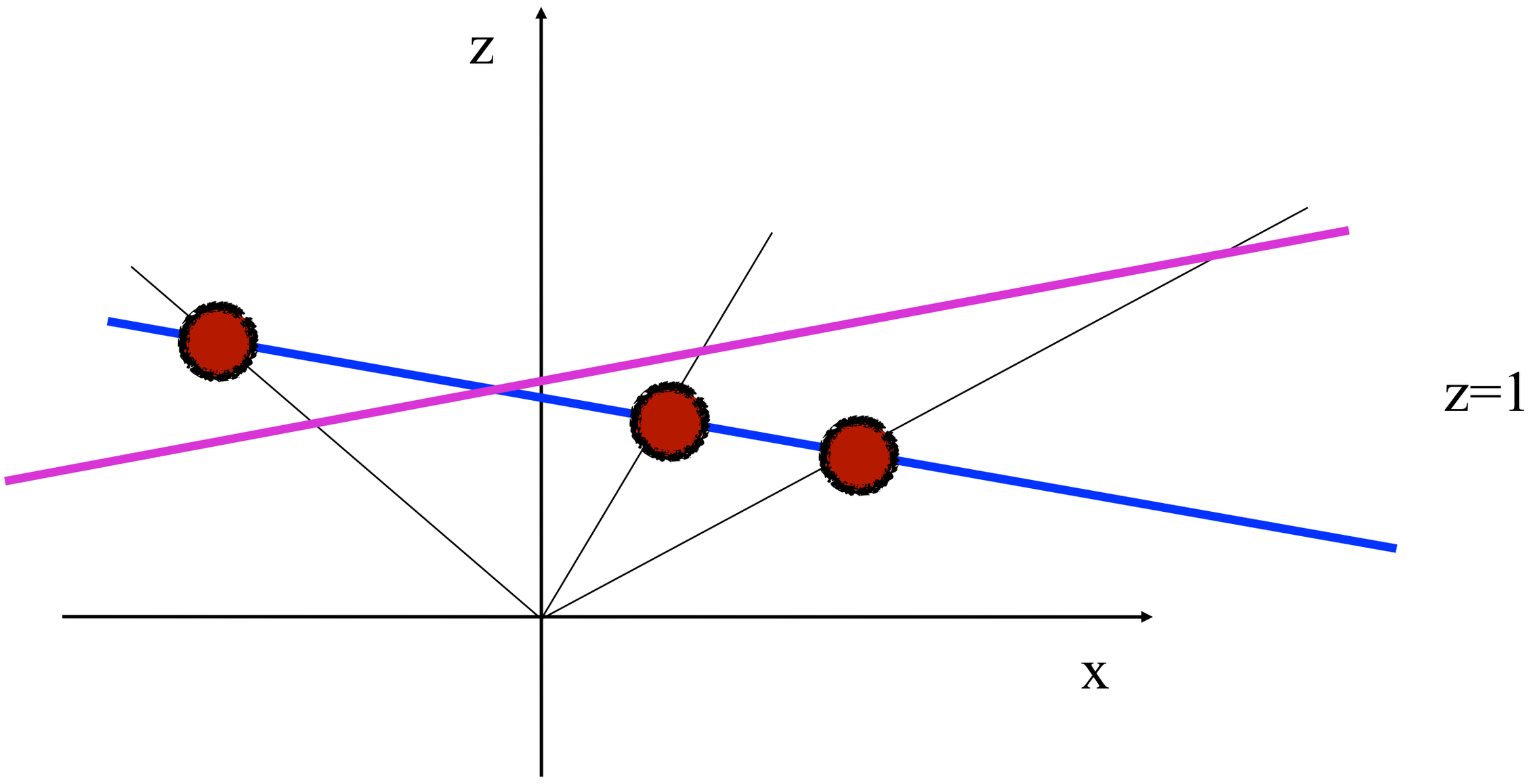


Project to different image?



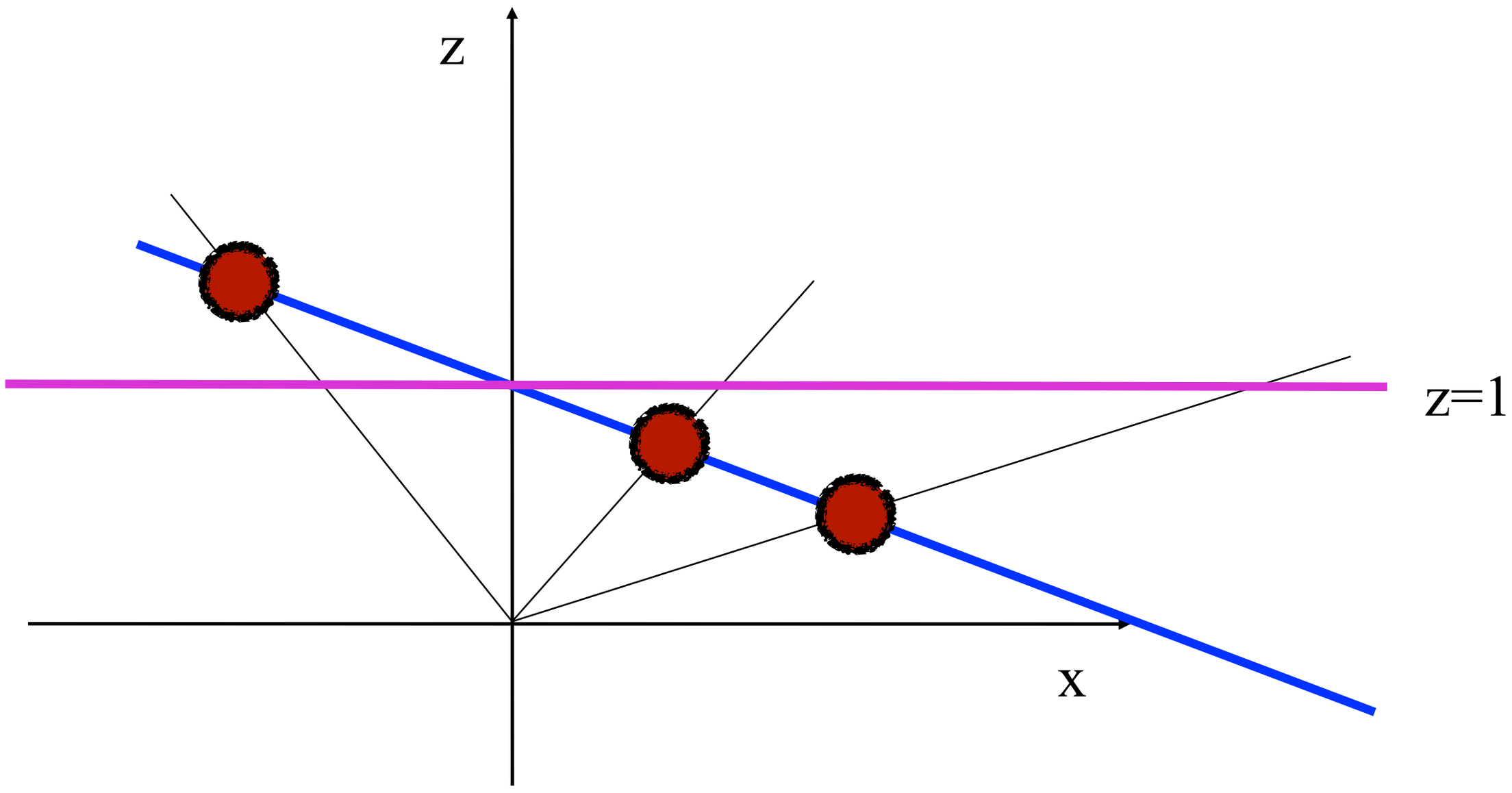
Rotate

- **Apply R to vector (x, y, 1)**



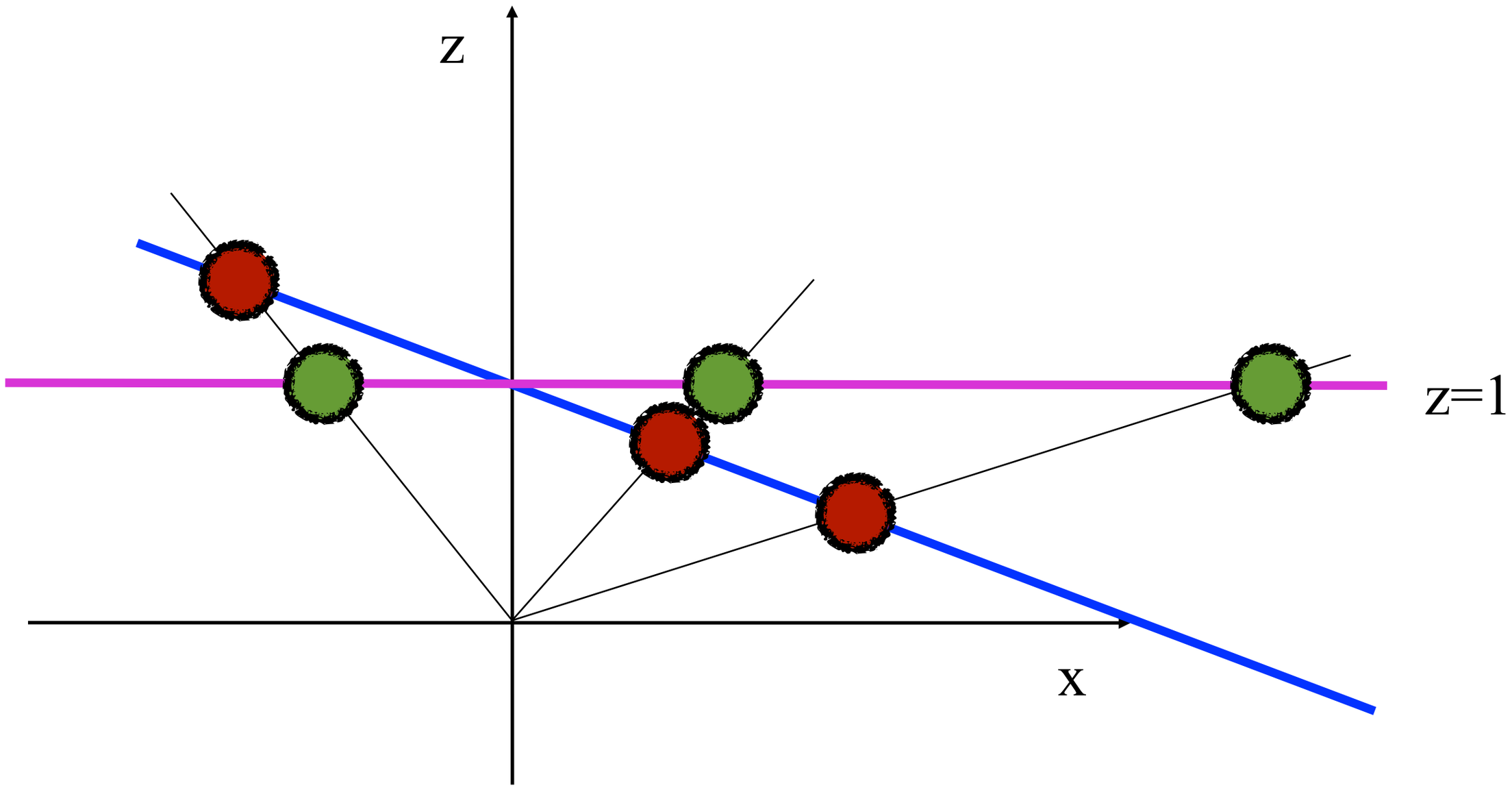
Rotate

- **Apply R to vector (x, y, 1)**



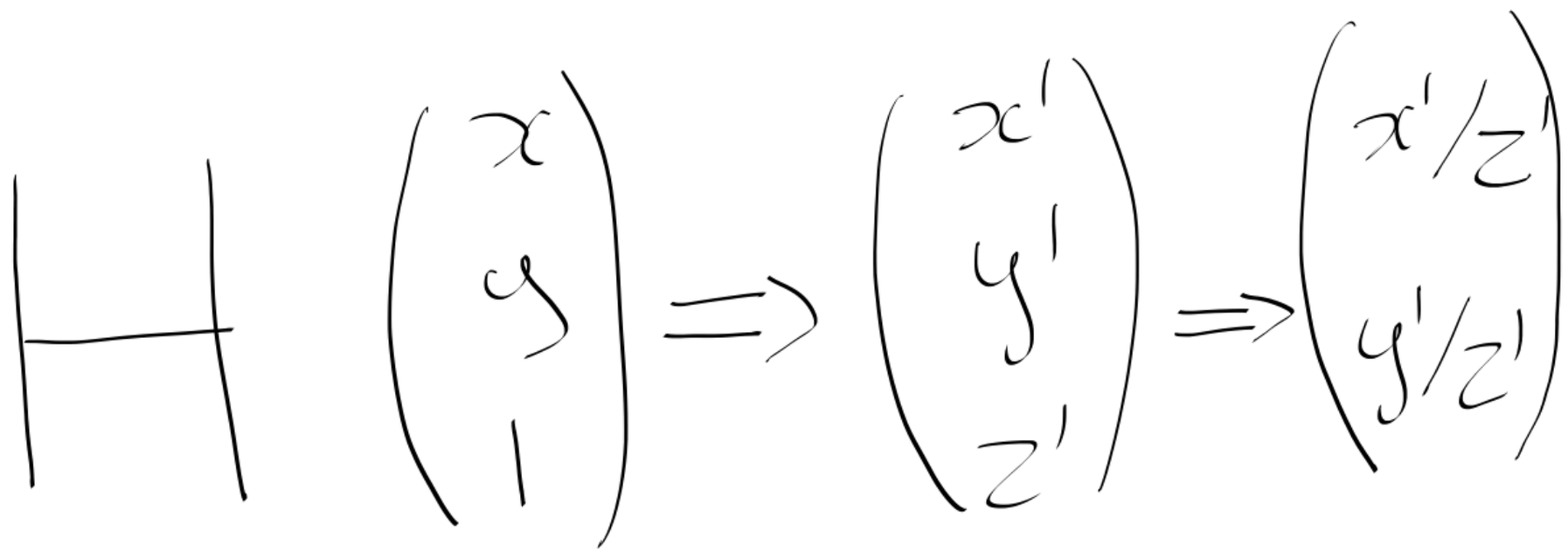
Rotate

- **Apply R to vector (x, y, 1)**
- **divide resulting x', y' by resulting z'**



Bottomline

- **The transformation from one 2D image to a rotated image of the same scene is a 3x3 matrix H applied to x,y,1 followed by a division by z'**
  - called a homography

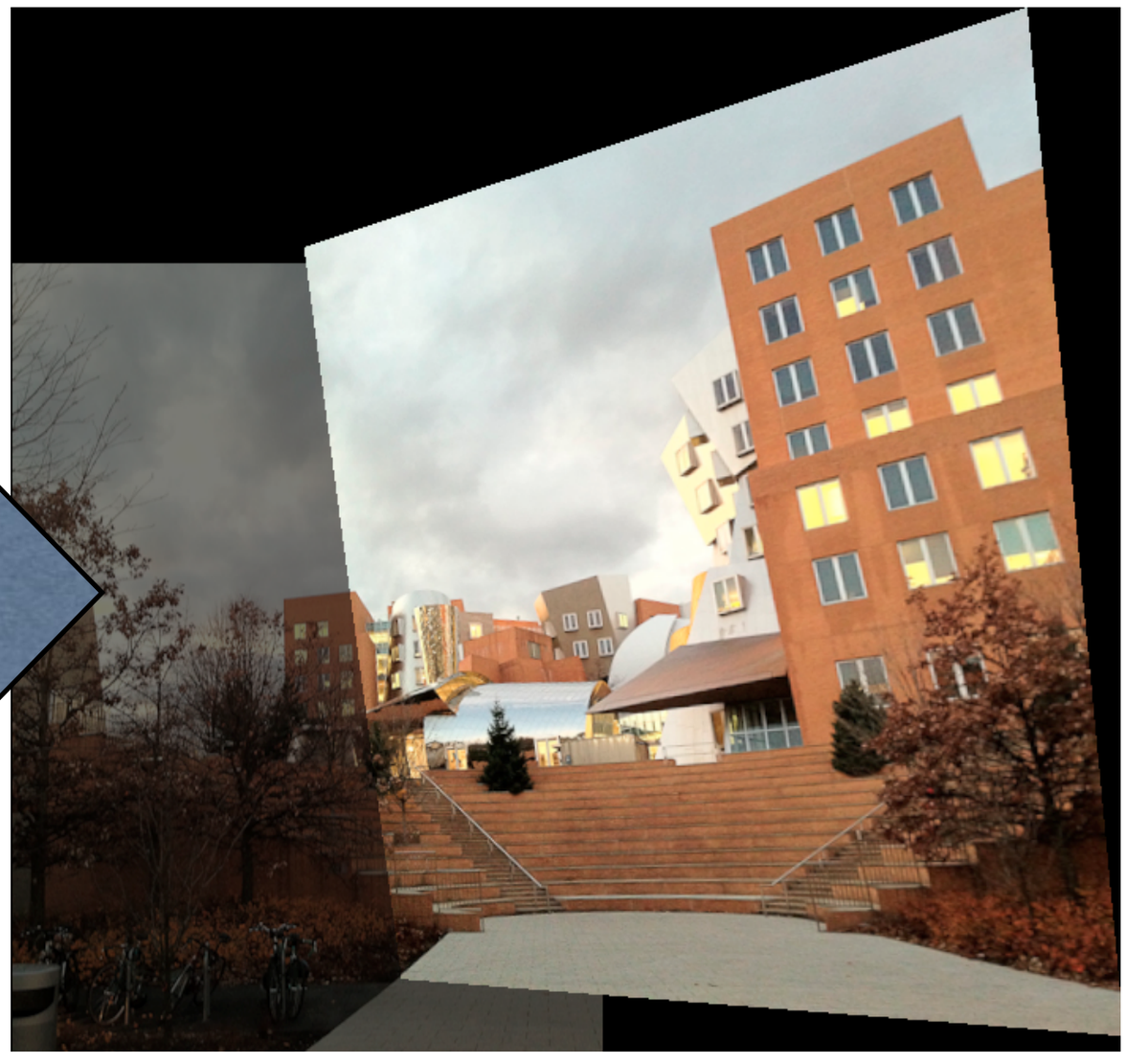


- **Now the only problem is that we usually don't know H**

Today

- The user gives us 4 correspondences
- Use regression to compute the matrix





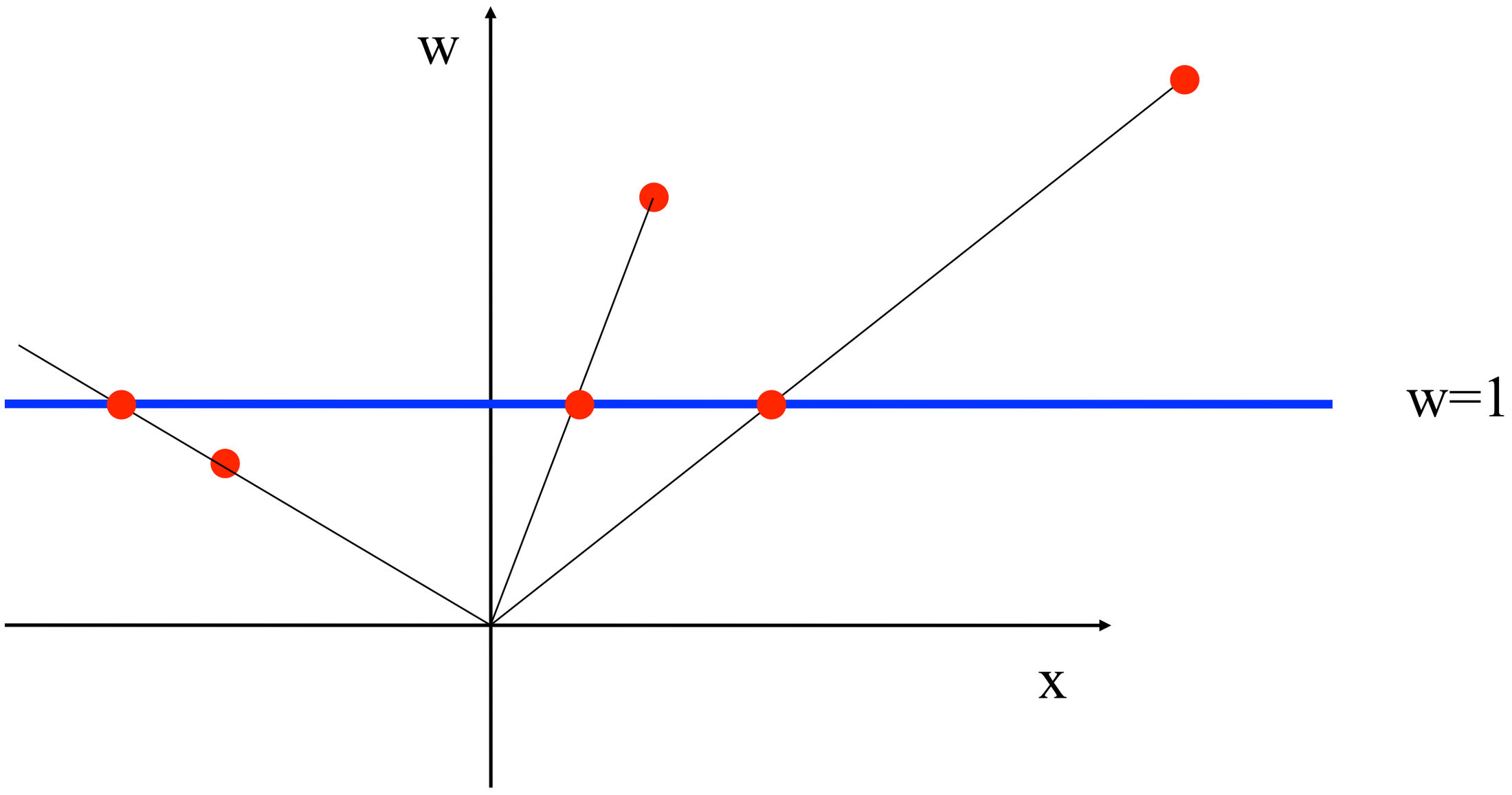
Questions?



Homogenous coordinates

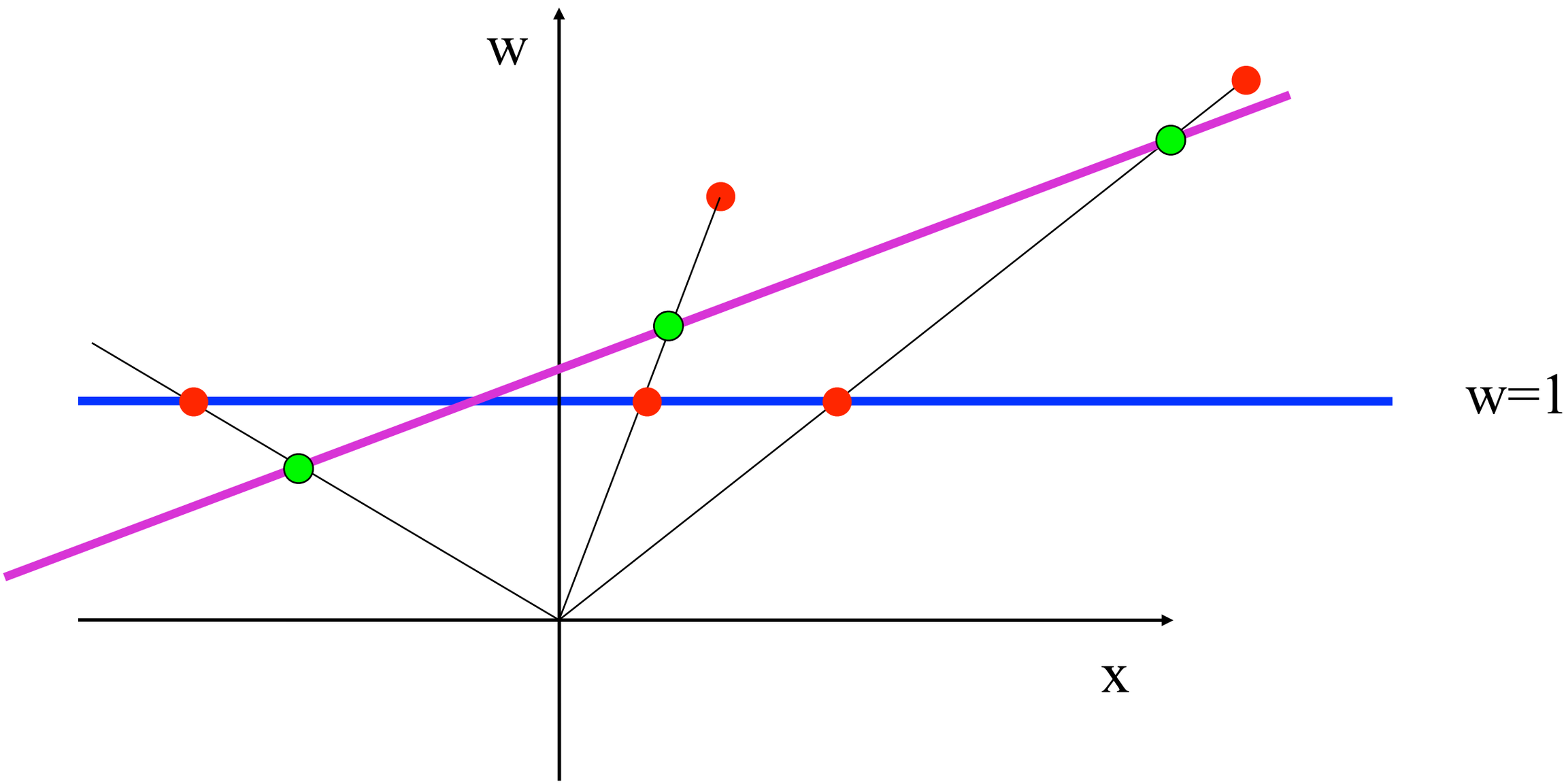
1D homogeneous coordinates

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



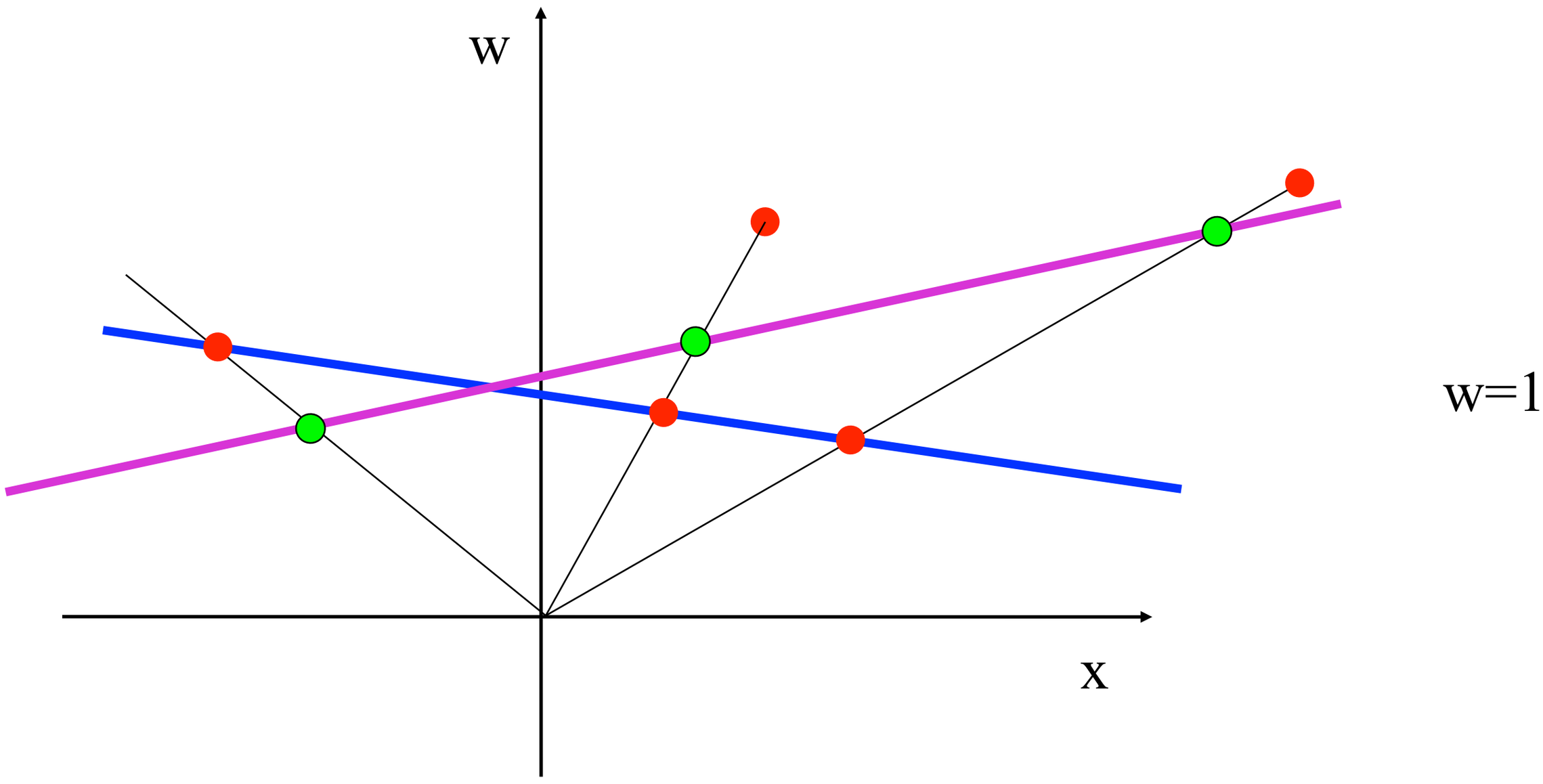
Other illustration: 1D homography

- **Reproject to different line**



1D homography

- **Reproject to different line**

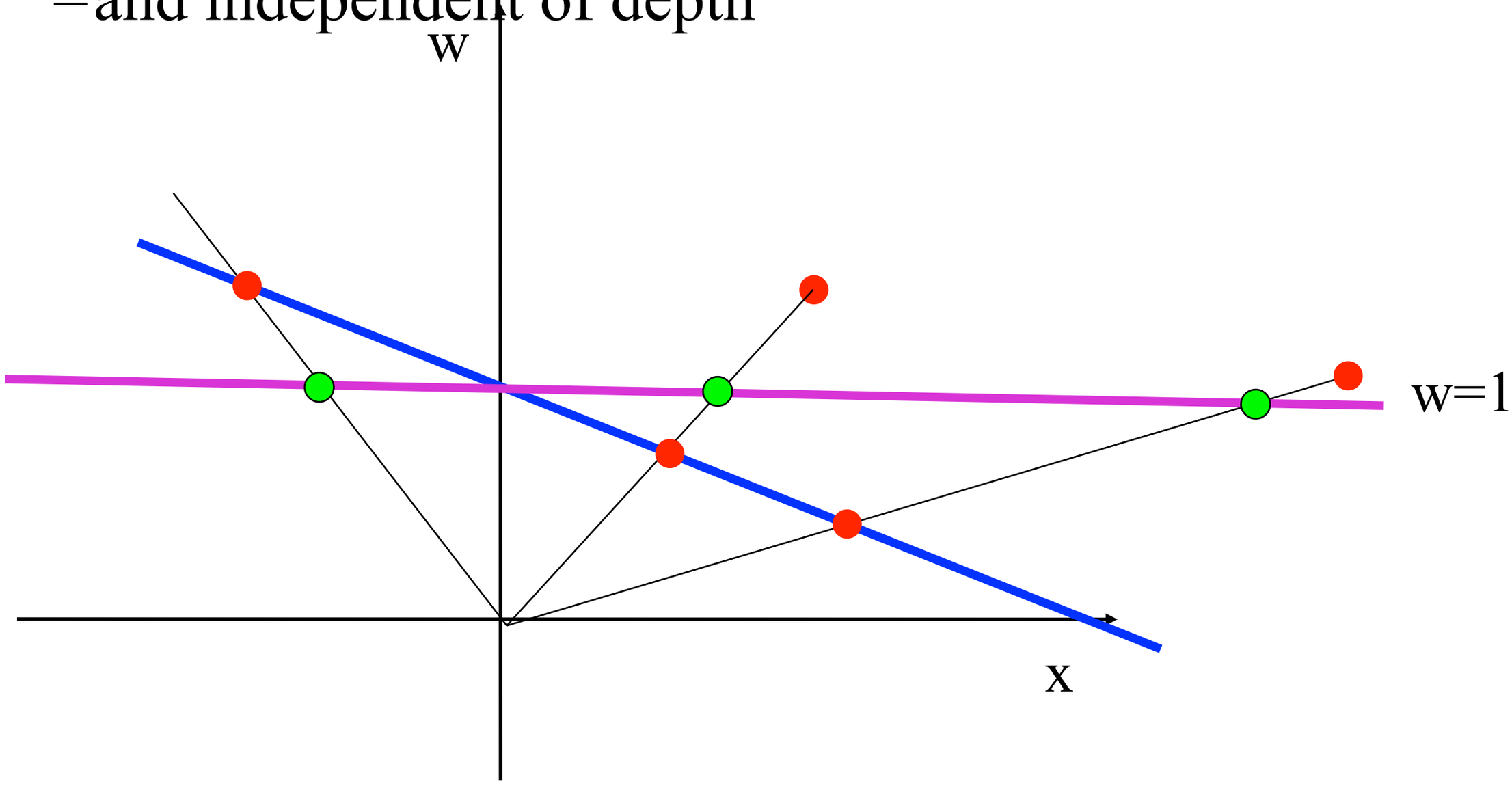


1D homography

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

  ➔ reprojection is linear in homogeneous coordinates

  –and independent of depth



Homogenous coordinates

- Representation of 2D points using 3 coordinates

- (x, y, w) represents (x/w, y/w)
  - Homogenous points are interpreted as Euclidean coordinates by dividing by the last coordinate

- **Motivations:**
  - perspective transforms
  - Make translations linear

Homogeneous coordinate

- represent 2D points with 3 numbers
- (x, y, w) represents (x/w, y/w)
- Allows us to represent projective transforms
- Nice thing: projecting onto plane $z=1$ is just the strict interpretation of homogeneous coordinates

- Yes, you can view this as a notation trick
  - –But math is all about smart notations
- Homogenous coordinates are central in computer graphics and machine vision

Scale invariant
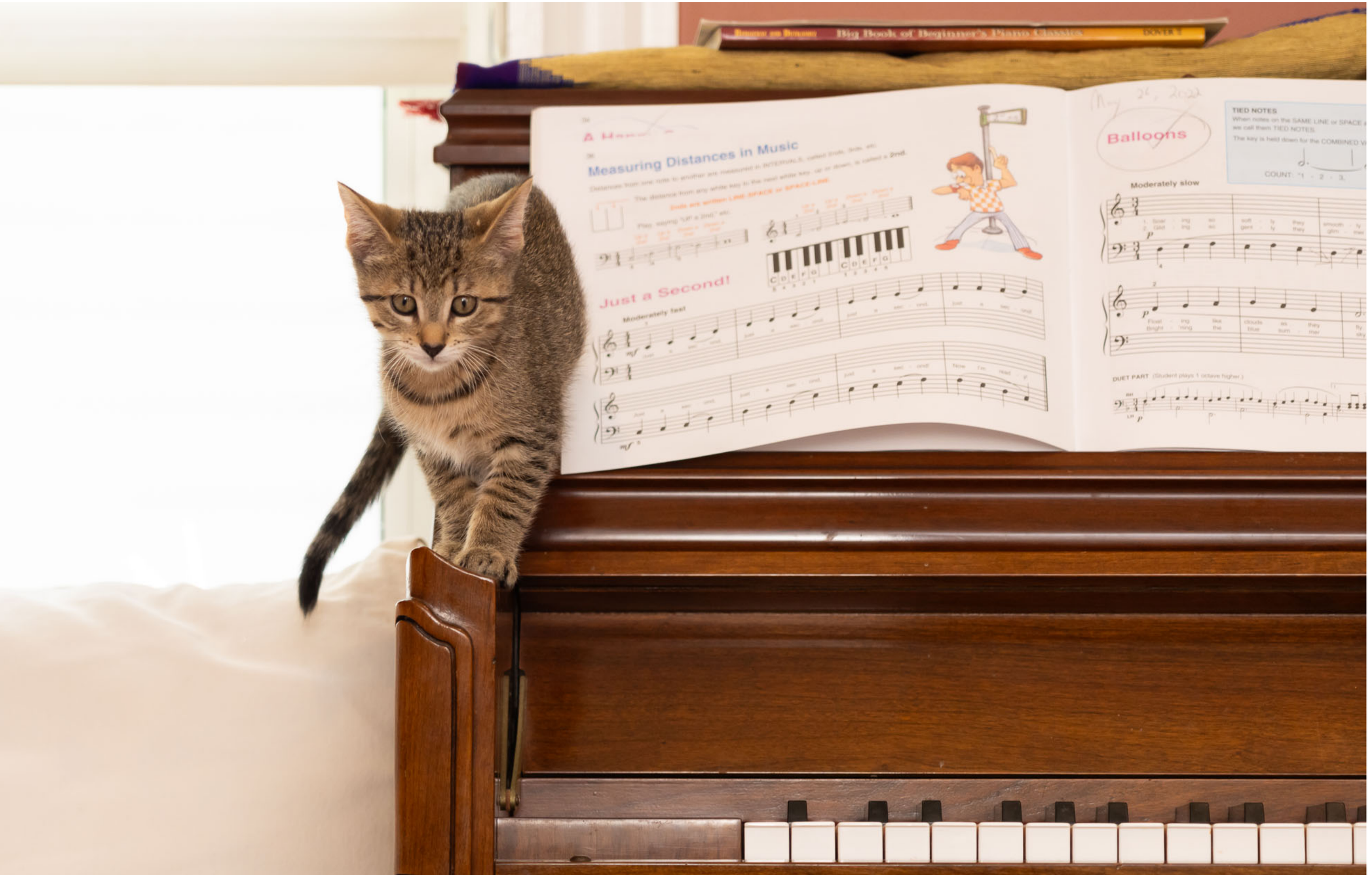
- All points (wx, wy, w) for any non-zero w represent the same Euclidean point

Extra profit

- Homogenous coordinates allow us to represent 2D translations as matrices

$$\begin{pmatrix} 1 & 0 & t_y \\ 0 & 1 & t_x \\ 0 & 0 & 1 \end{pmatrix}$$

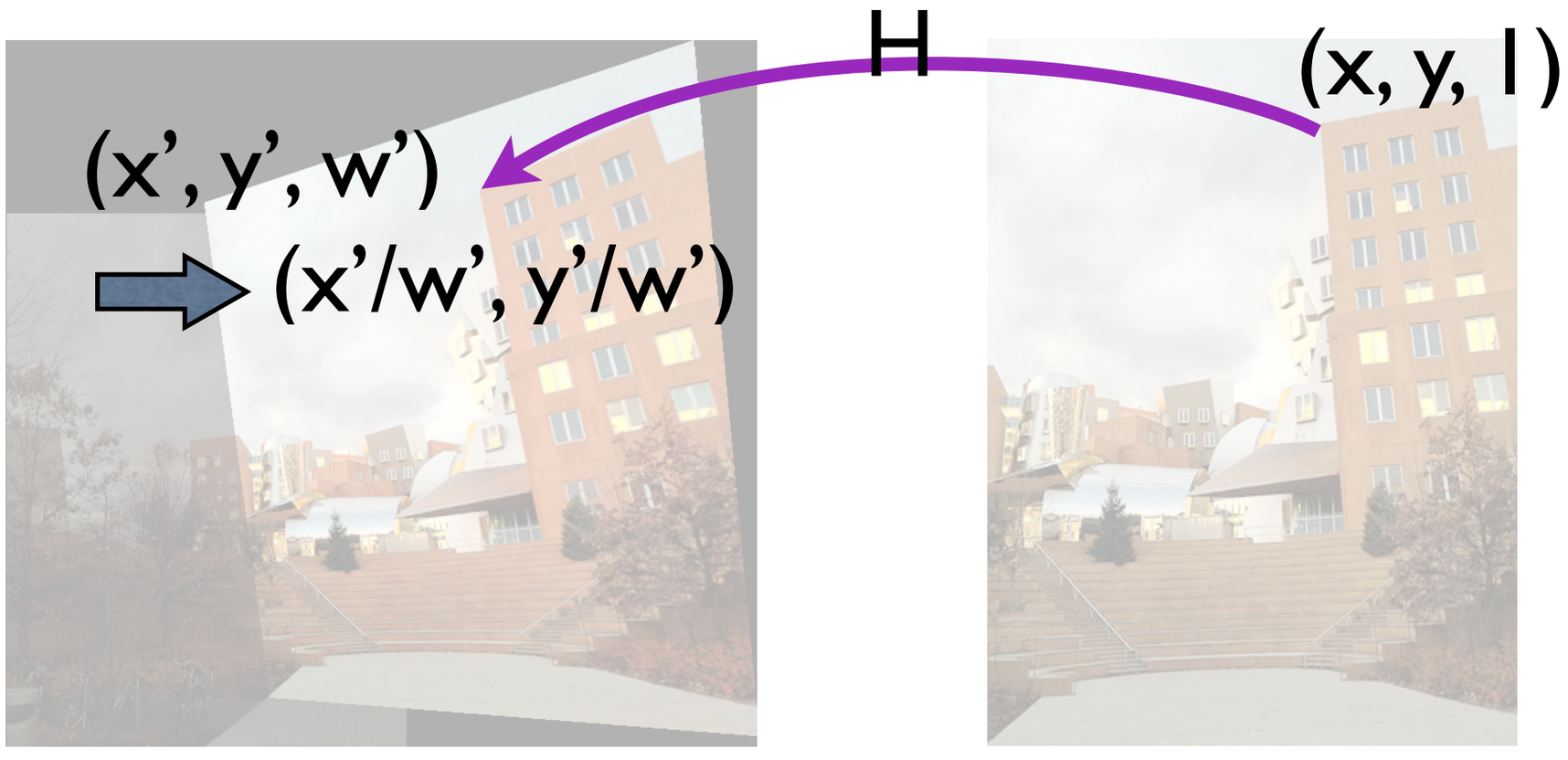
Questions


Big Book of Beginner's Piano Classics
Measuring Distances in Music
Just a Second!
Balloons
TIED NOTES
Moderately slow
DUET PART (Student plays 1 octave higher.)

Homographies

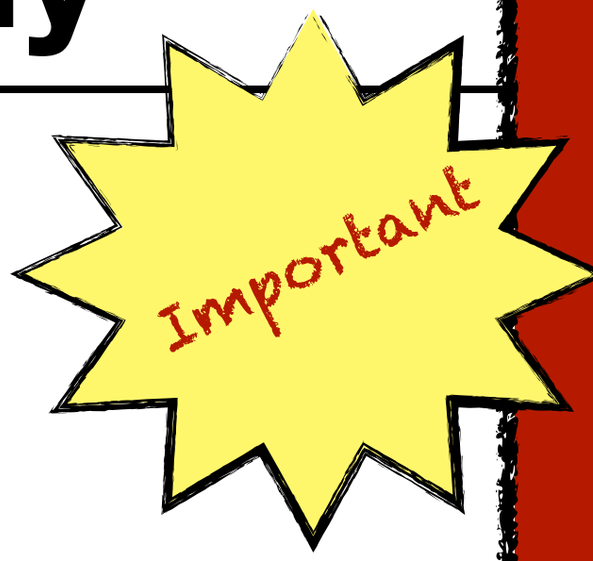
2D homogenous version

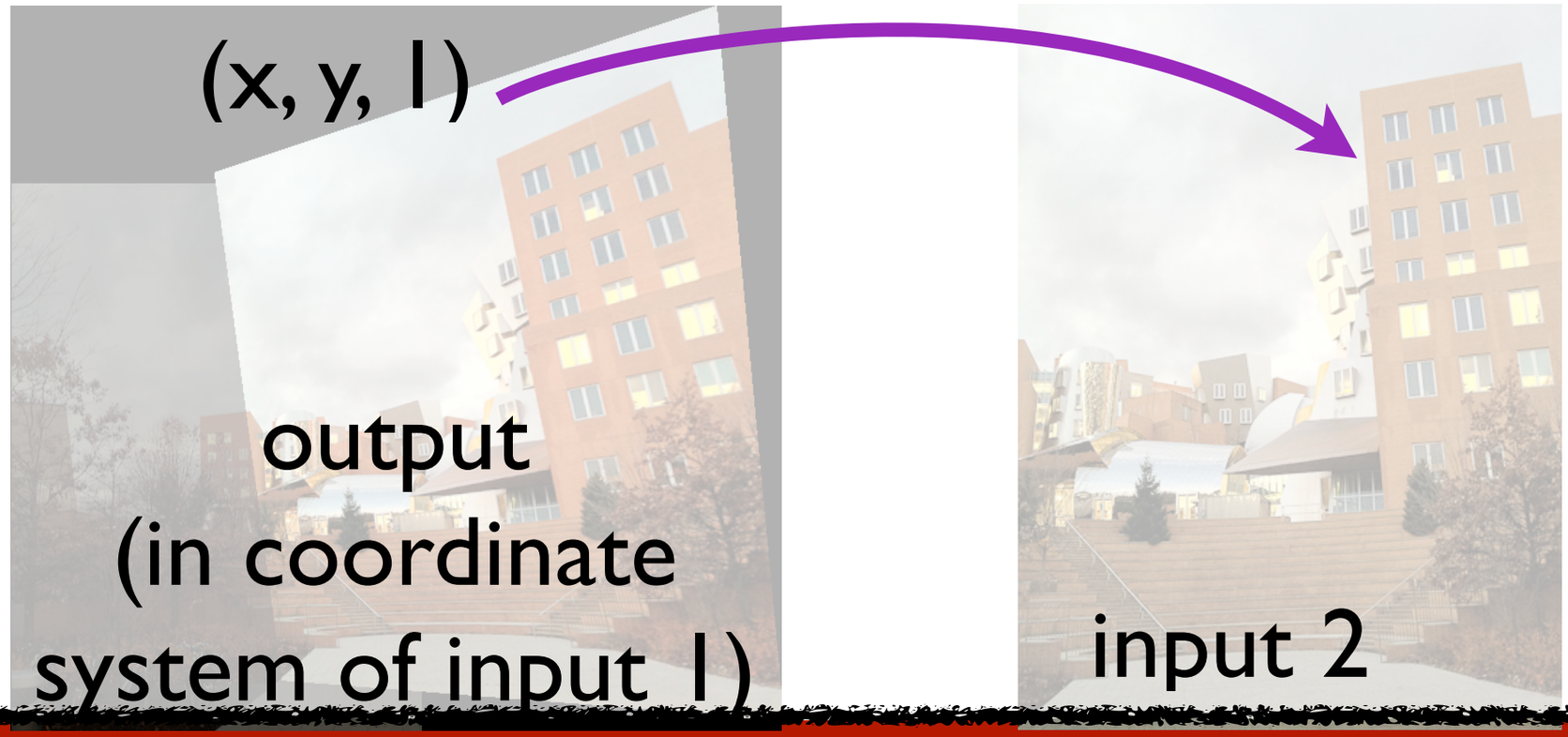
- 2D points represented as homogenous coordinates (x, y, 1)
- Mapped into the other image by a 3x3 matrix H, into homogenous coordinates (x’, y’, w’)
- get Euclidean coordinates by dividing by w’



Warping with homography

- For each pixel y,x, in output

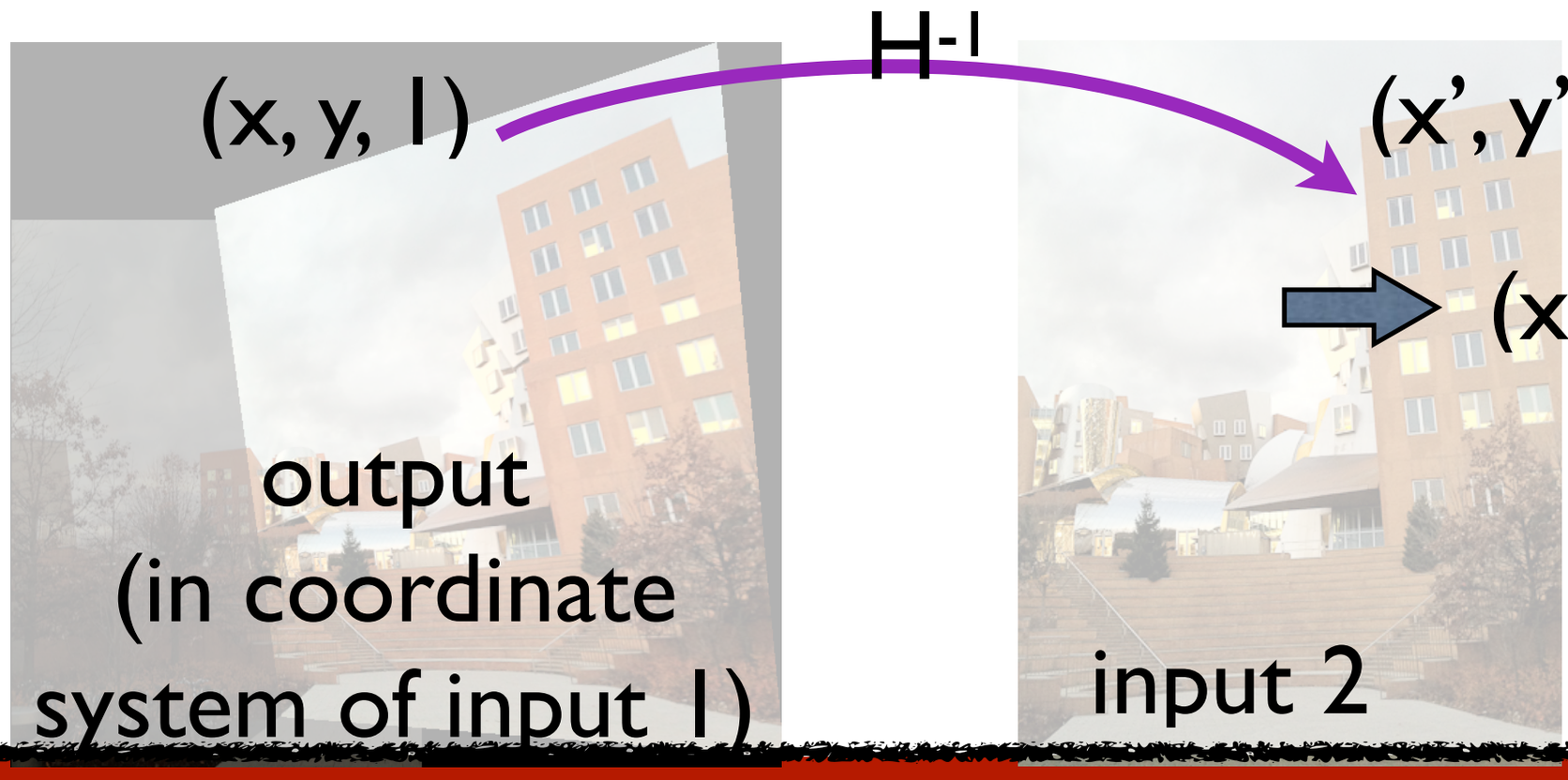




Warping with homography

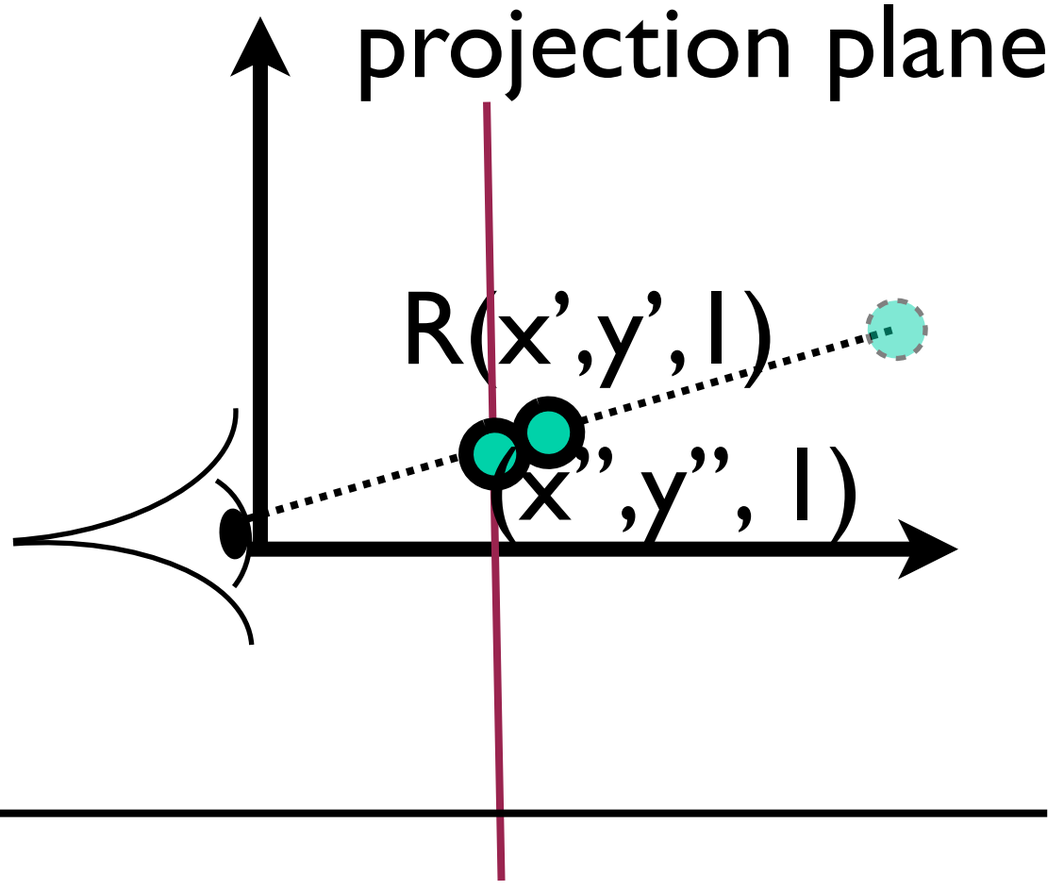
- For each pixel y,x, in output
  - $(y',x',w')=H^{-1}(y,x,1)$
  - $(y'', x'')=(y'/w' , x'/w')$
  - $out(y,x)=in(y'', x'')$



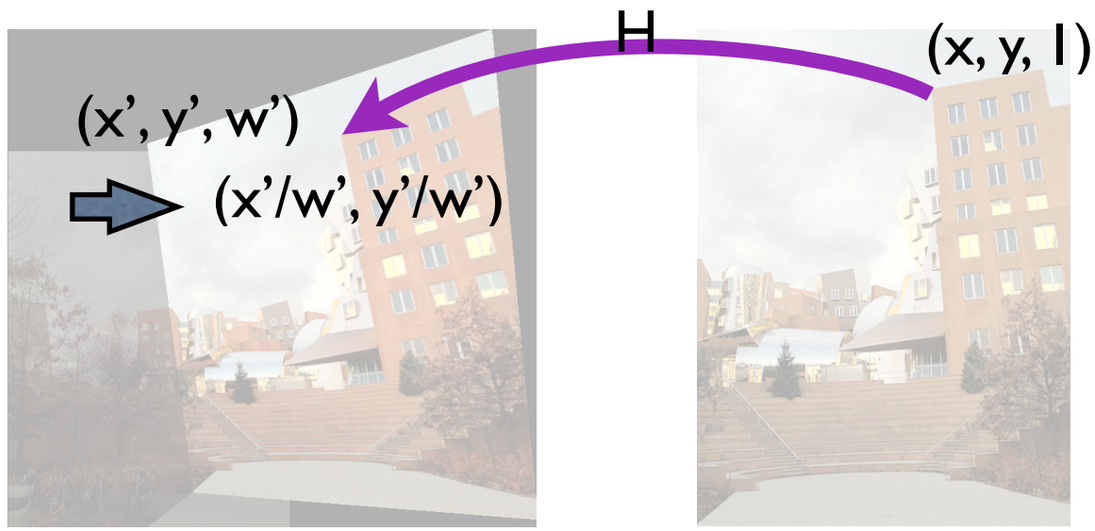


3D vs. homogenous: the same

- 3 coordinates because viewed as plane in 3D
- 3x3 matrix because rotation
- Divide because perspective



- weird extra coordinate
- 3x3 homography matrix
- divide because it's the rule

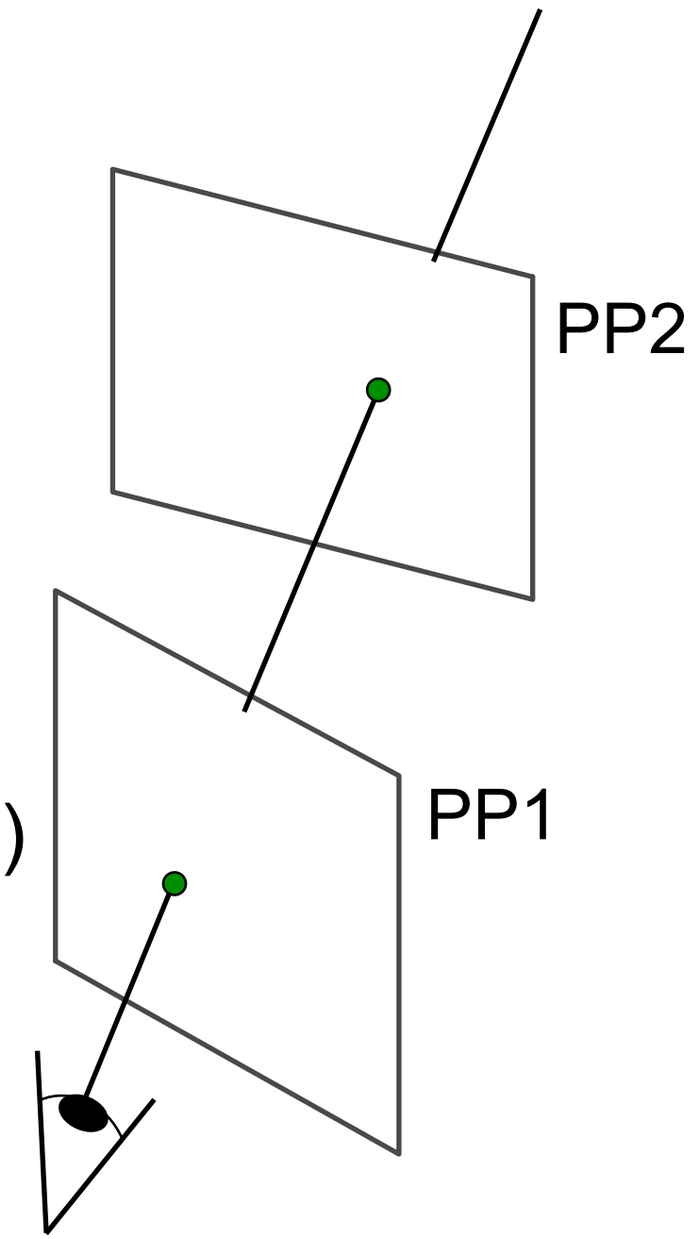


Wrapping it up: Homography

- **Projective mapping between any two planes**
- **represented as 3x3 matrix in homogenous coordinates**
  - corresponds to the 3D rotation

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

$\mathbf{p}'$ $\mathbf{H}$ $\mathbf{p}$



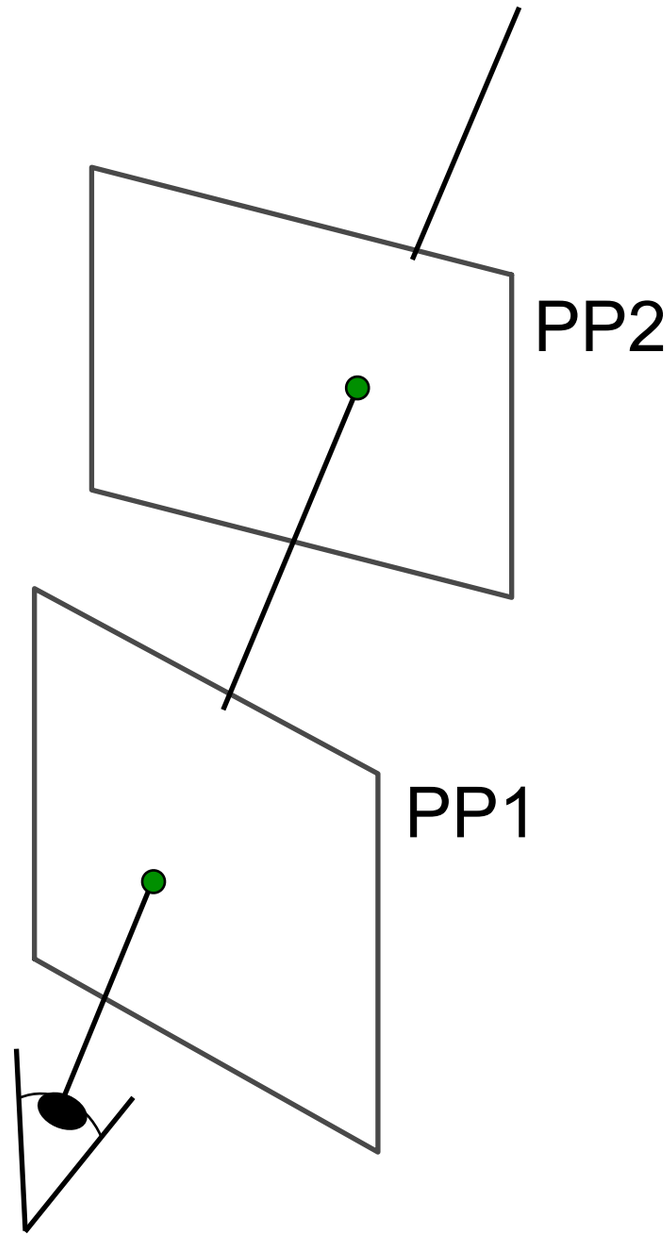
To apply a homography **H**

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

Wrapping it up: Homography

- **rectangles map to arbitrary quadrilateral**
- **parallel lines aren't parallel anymore**
- **but straight lines remain straight**

- **same as: project, rotate, reproject**

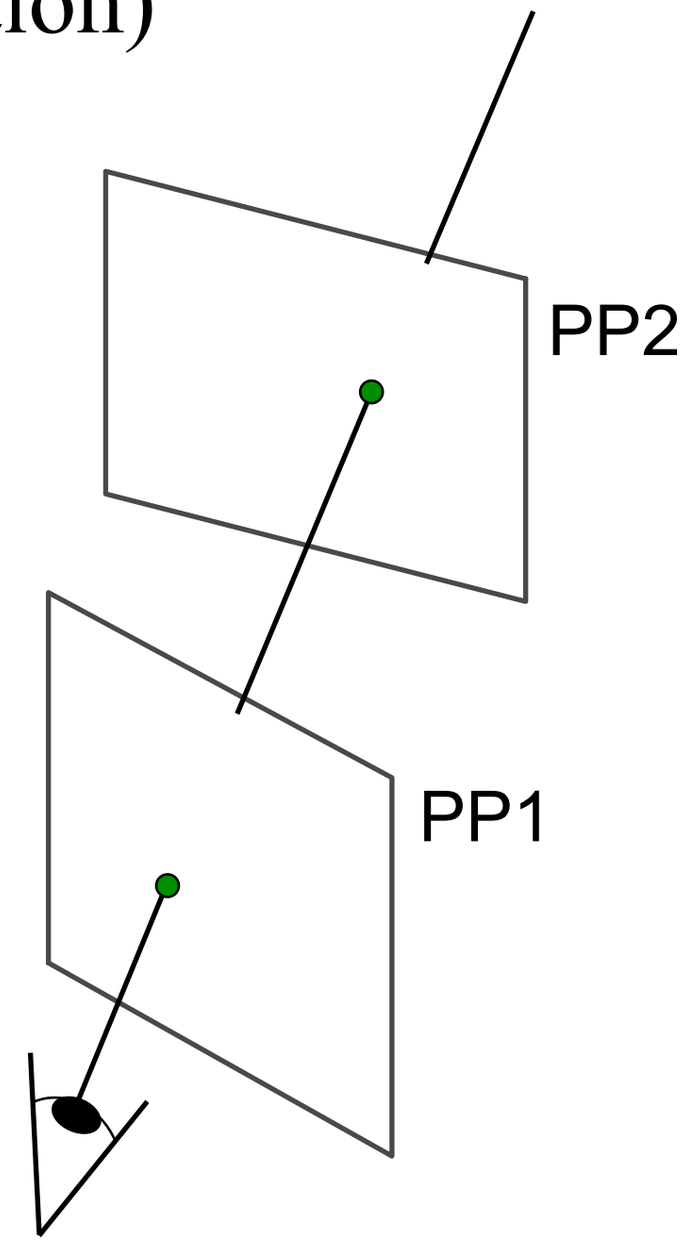


Recap

- **Reprojection = homography**
- **3x3 matrix in homogeneous coordinate**
  - (the matrix can be constrained to be a rotation)

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

$\mathbf{p}'$ $\mathbf{H}$ $\mathbf{p}$



Scale

- Homographies are defined up to a scale
- H and kH represent the same 2D transformation
- because (w'x', w'y', w') and (kw'x', kw'y', kw') represent the same point

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

$$\mathbf{p}' \qquad \mathbf{H} \qquad \mathbf{p}$$

General pairs of plane

- **Homographies can project from any plane to any plane**



+

Homework
due 9pm

→



General pairs of plane

- **Homographies can project from any plane to any plane**

- **Note: here we only warped the image rectangle, but the homography is defined for the whole plane, any x,y**



+

Homework
due 9pm

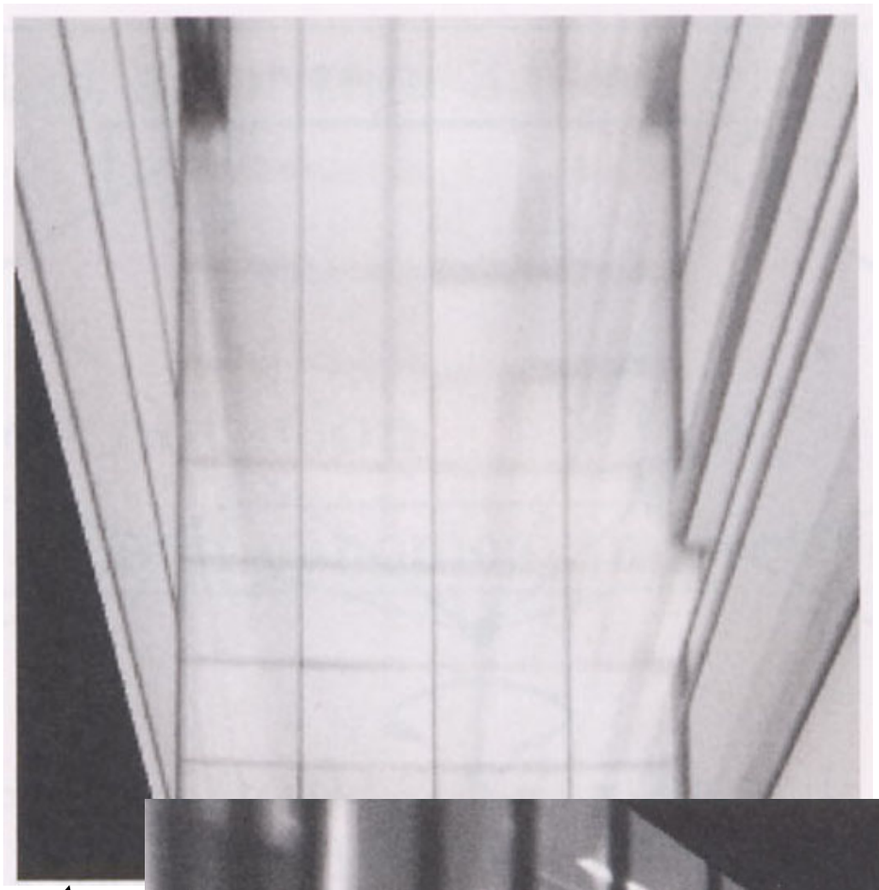
→



Image warping with homographies



homography so that image is parallel to floor



homography so that image is parallel to right wall



black area where no pixel maps to

Questions?

- http://news.nationalgeographic.com/news/2012/04/120416-nasa-space-shuttle-discovery-smithsonian-360-tour-panorama-science/?source=link_tw20120417news-discoveryshuttle



Solving for homographies

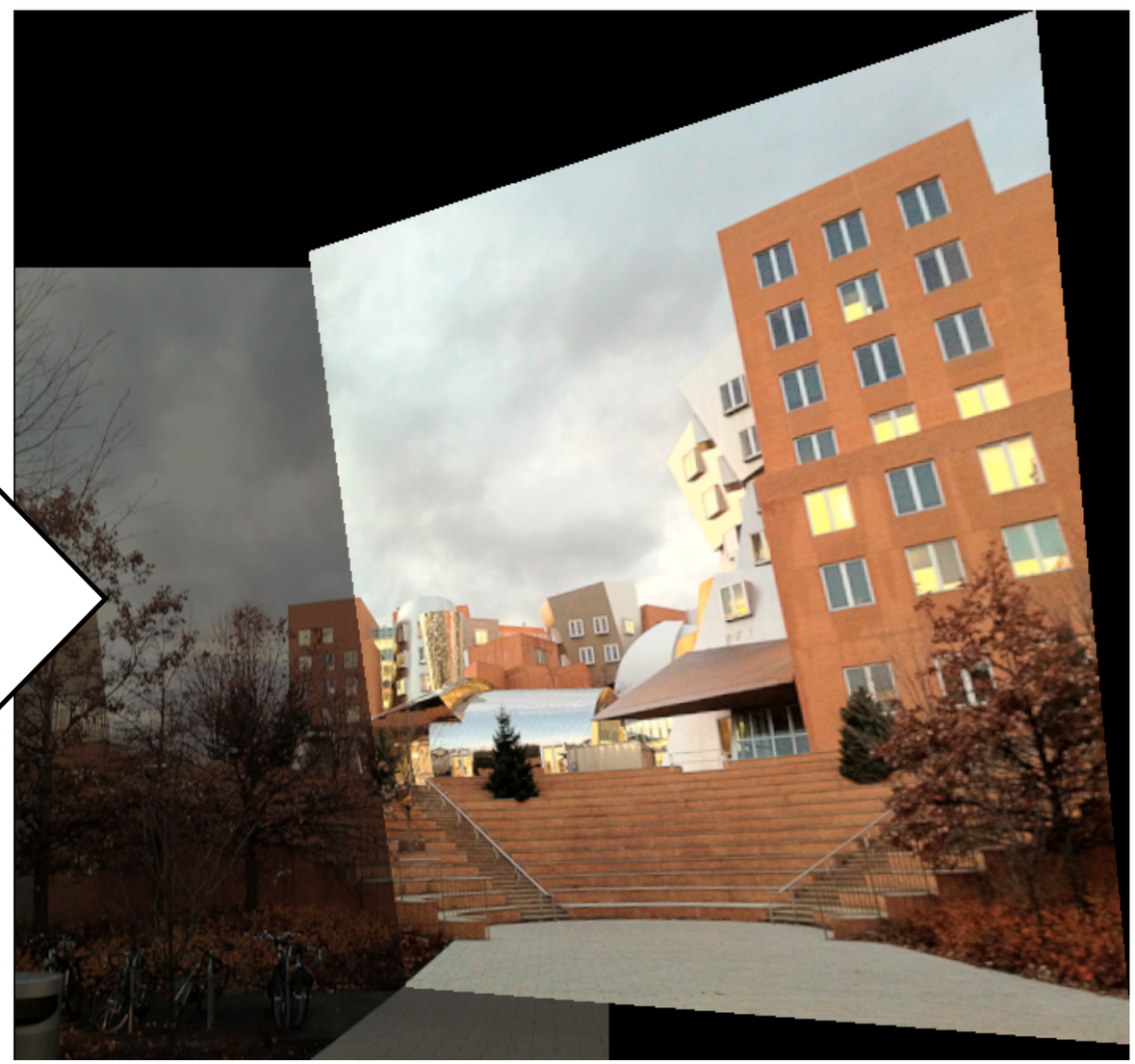
Goal

- Given correspondences
- Find homography matrix H that maps the p_i to p_i’





H



Given 4 pairs, deduce H

Warning

- In what follows I use the order y, x
  - At least I will try

Homography equation

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

- **We are given pairs of corresponding points**
  - x, y, x', y' are known
- **Unknowns: matrix coefficients and w'**

Homography equation

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

- **We are given pairs of corresponding points**
  - x, y, x', y' are known
- **Unknowns: matrix coefficients and w'**
  - but w' is easy to get:

$$w' = gy + hx + i$$

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

$$w' = gy + hx + i$$

- **For a pair of points (x, y)->(x',y') we have**

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

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

$$w' = gy + hx + i$$

- **For a pair of points (x, y)->(x',y') we have**

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

- **Unknowns: a, b, c, d, e, f, g, h, i**
  - Linear!

How many pairs?

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

- Each correspondence pair gives us two equations

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

- How many unknowns?
  - 9
  - but H is defined up to scale. Four pairs are

Forming the linear system

- We have 4x2 linear equations in our 8 unknowns
- Represent as a matrix systemAx=B:

$$\begin{pmatrix} & & \\ & A & \\ & & \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \end{pmatrix} = B$$

- Now we need to fill matrix A and vector B

Forming the matrix

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

$$\overbrace{\begin{pmatrix} a & b & c & d & e & f & g & h & i \end{pmatrix}}^{\text{}} \cdot \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \end{pmatrix} = \begin{matrix} \cdot \\ \cdot \\ \\ \cdot \\ \cdot \\ \cdot \end{matrix}$$

Forming the matrix

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

$$\overbrace{\begin{matrix} a & b & c & d & e & f & g & h & i \end{matrix}}$$

$$\begin{pmatrix} & & & & & & & & \\ & & & & \cdots & & & & \\ y & x & 1 & 0 & 0 & 0 & -yy'-xy'-y' & & \\ & & & & \cdots & & & & \\ & & & & \cdots & & & & \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \end{pmatrix} = \begin{matrix} \cdot \\ \cdot \\ 0 \\ \cdot \\ \cdot \\ \cdot \end{matrix}$$

I'll let you do the x case for pset 6

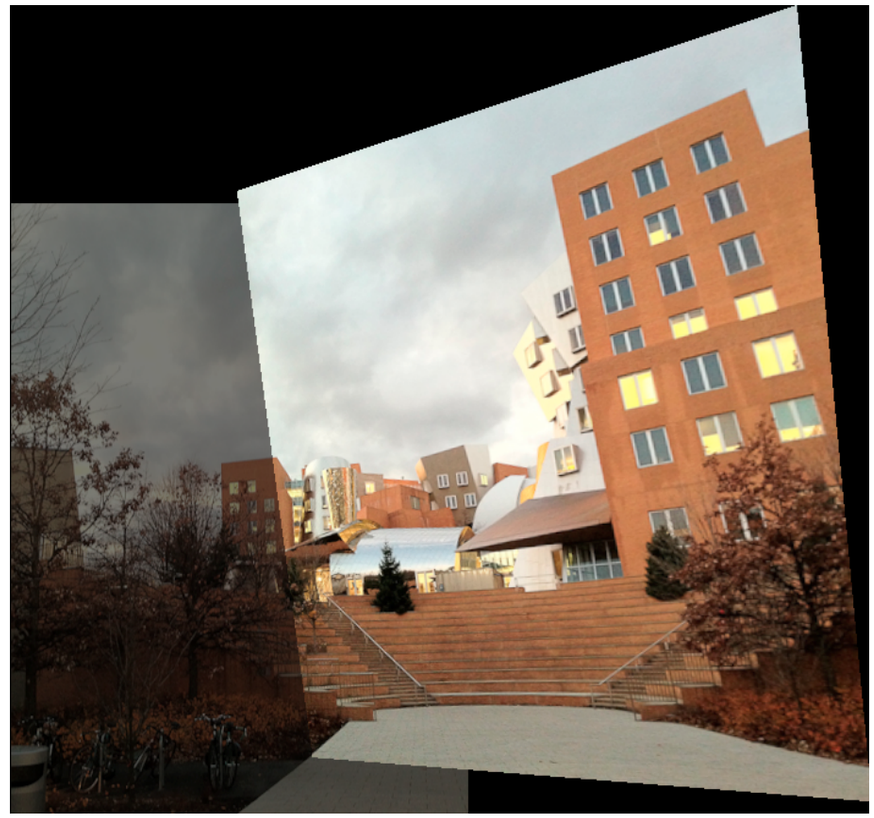
Recap

- We have four pairs of points



- Looking for homography H

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



- **Formed a big 8x9 linear system Ax=0**
  - where x is the 9 homography coefficients

$$\begin{pmatrix} & & & & & & & & \\ & \cdots\cdots & & & & & & & \\ y & x & 1 & 0 & 0 & 0 & -yy' & -xy' & -y' \\ & \cdots\cdots & & & & & & & \\ & \cdots\cdots & & & & & & & \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ f \\ g \\ h \\ i \end{pmatrix} = \begin{matrix} \cdot \\ \cdot \\ \cdot \\ 0 \\ \cdot \\ \cdot \\ \cdot \end{matrix}$$

Solve for scale invariance

- We know that there exists a full family of solutions kH, for any non-zero k
- i.e., if (a, b, c, d, e, f, g, h, i) is a solution, so is (ka, kb, kc, kd, ke, kf, kg, kh, ki)

- Adding more correspondences won’t help

Dirty solution

- Hope that *i* is not 0
- Set it arbitrarily to 1
  - either create an 8x8 matrix
  - or add a last row that says i should be 1

- In practice i is rarely zero
  - But it’s still dirty
- You can use this solution for pset 6

Cleaner solution

- Use SVD
- The singular vector with singular value 0 is a solution
- See your favorite linear algebra textbook

Dirty-clean: Three versions

- Dirty with i=1
  - 9x9:
    - RHS is zero almost everywhere, except last row
    - last row just says i=1
  - 8x8:
    - substitute i=1 from beginning
    - RHS is usually non-zero
- Clean with SVD: 8x9
  - Ax=0

Questions?

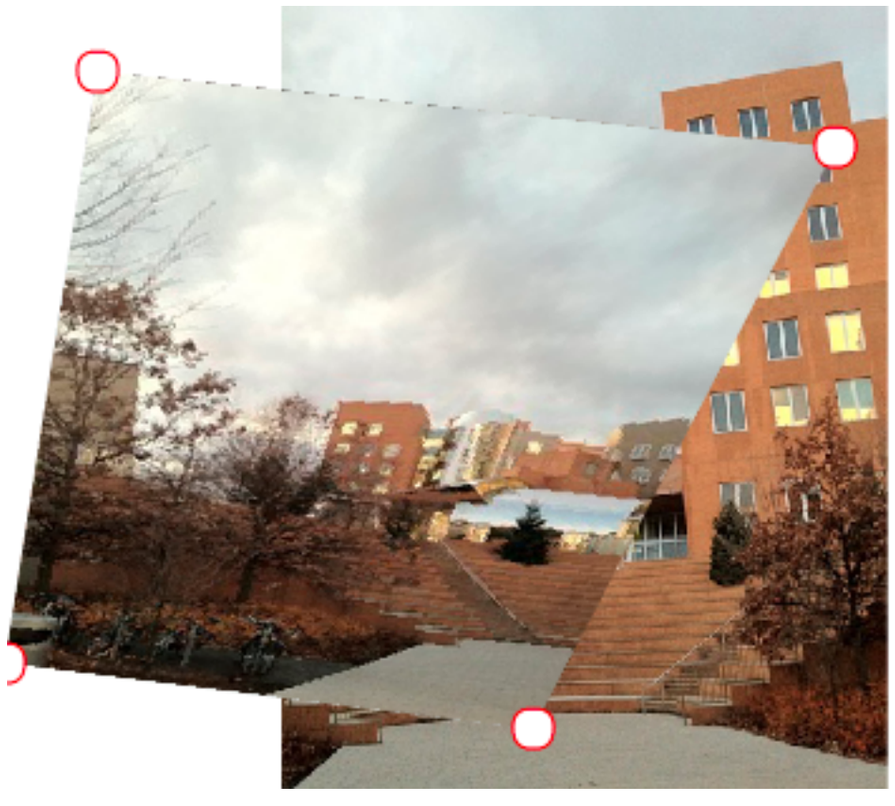
- **Julian Beever**
- **e.g. http://users.skynet.be/J.Beever/ http://www.crystalinks.com/julian_beever.html**





Demo

- **http://openphotovr.org/edit.html?id=33RwaQHm**
- **If we change the p', we get a different homography**



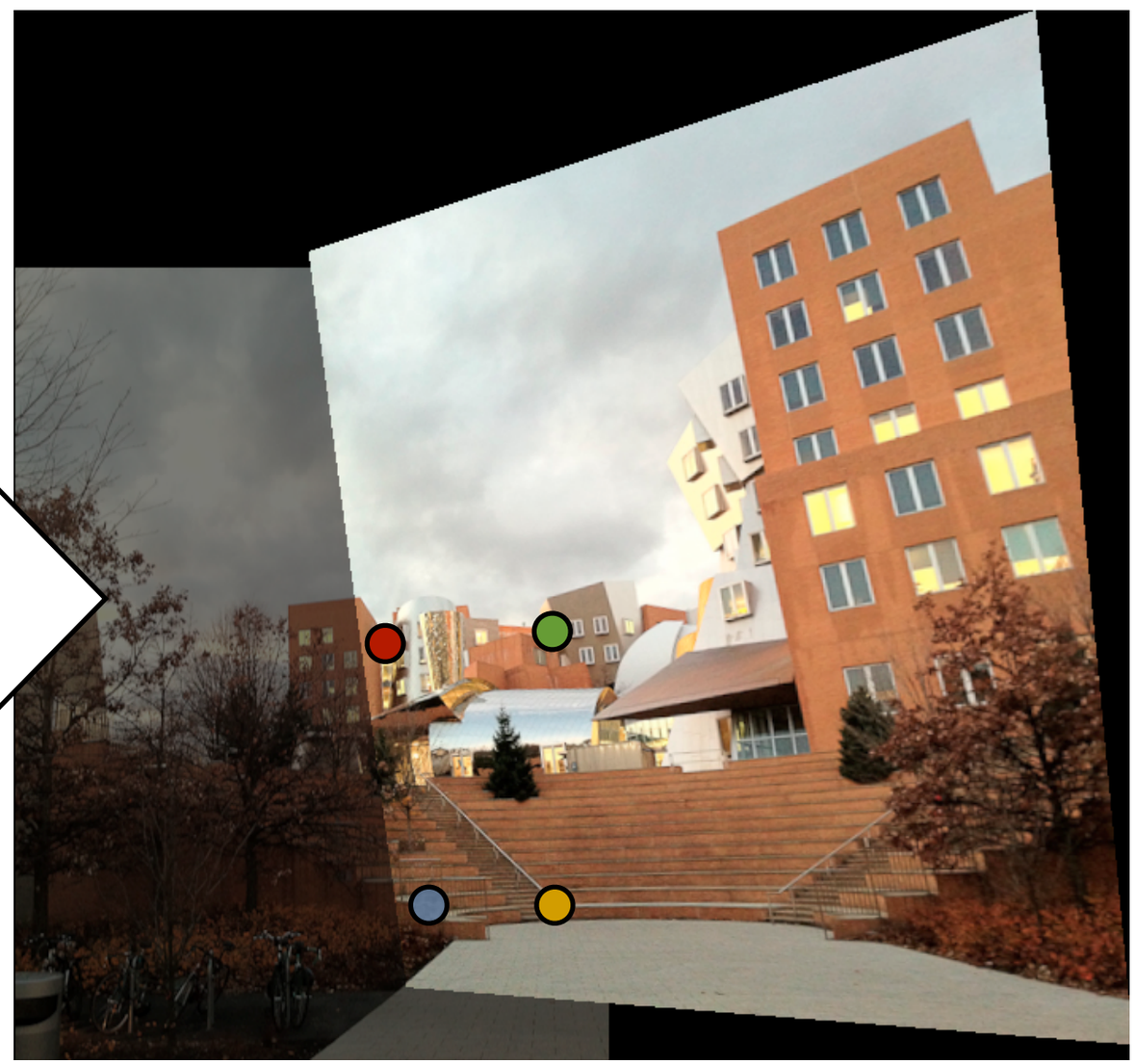
Homography

- Different sets of p, p’ correspondences can yield the same homography





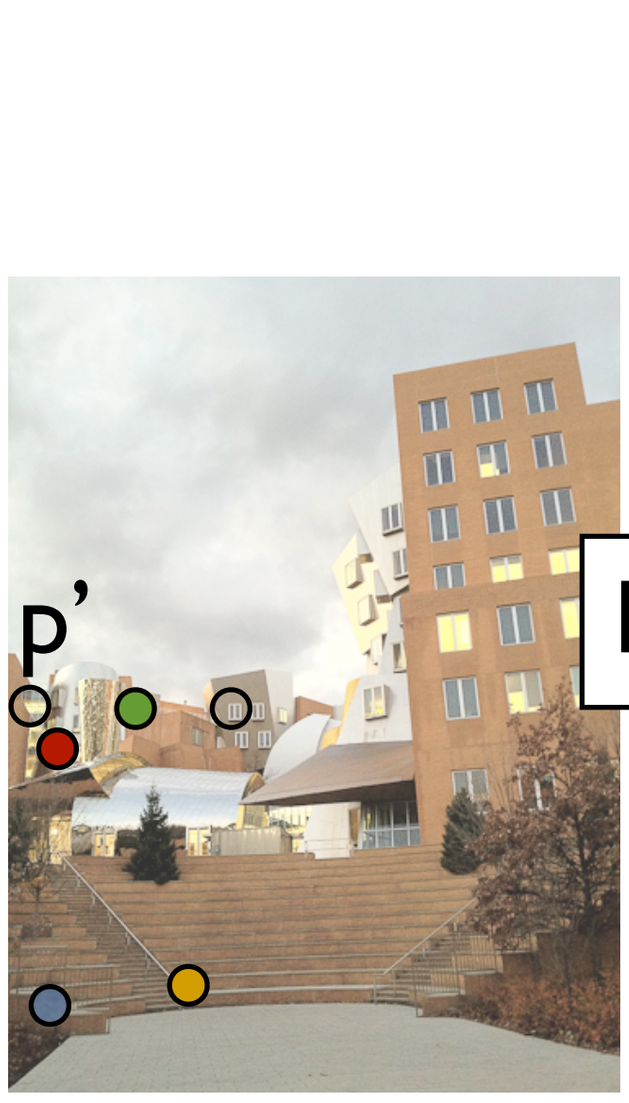
H



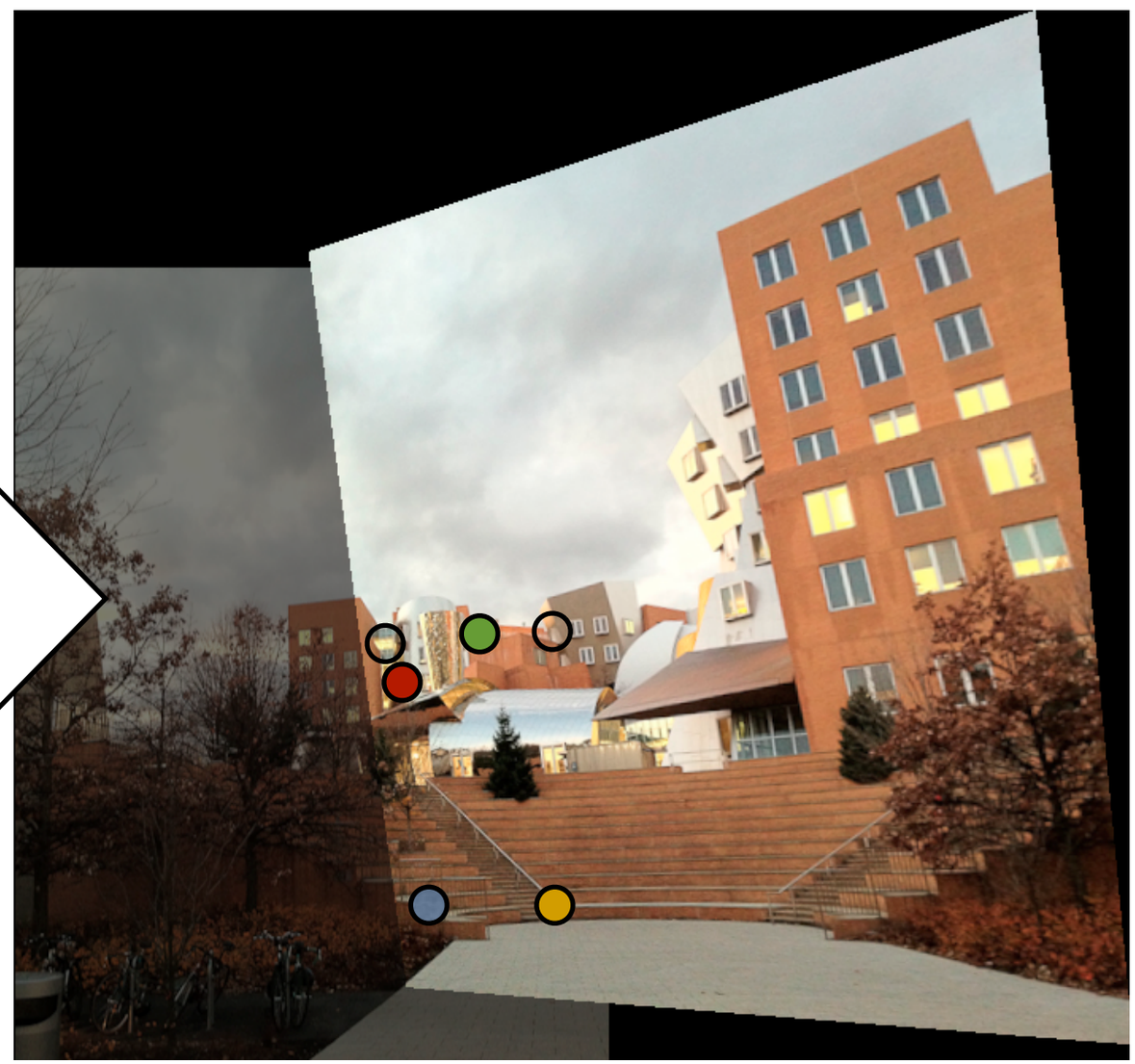
Homography

- Different sets of p, p’ correspondences can yield the same homography



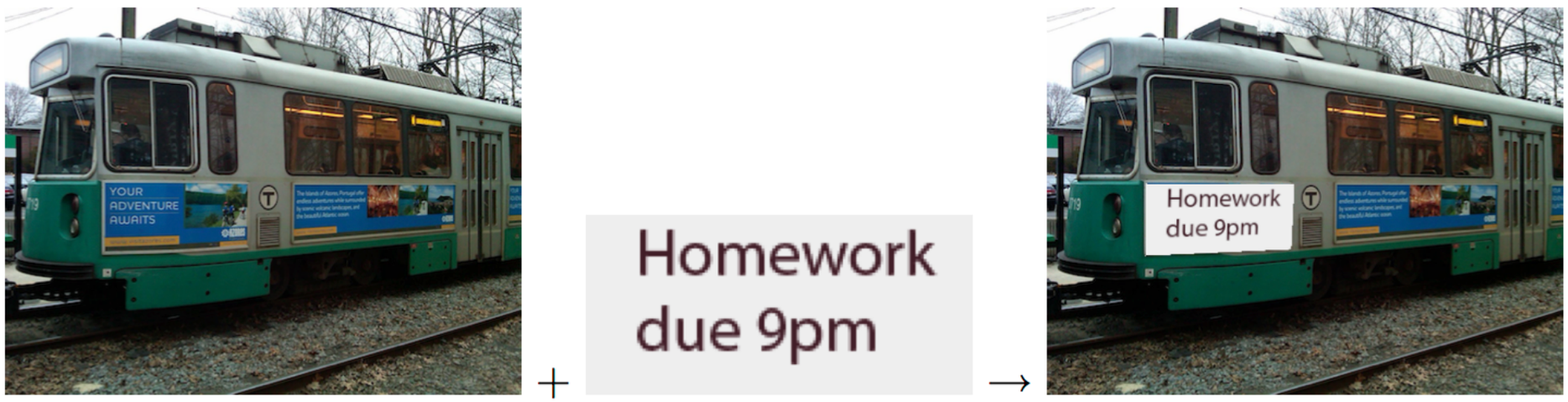


H

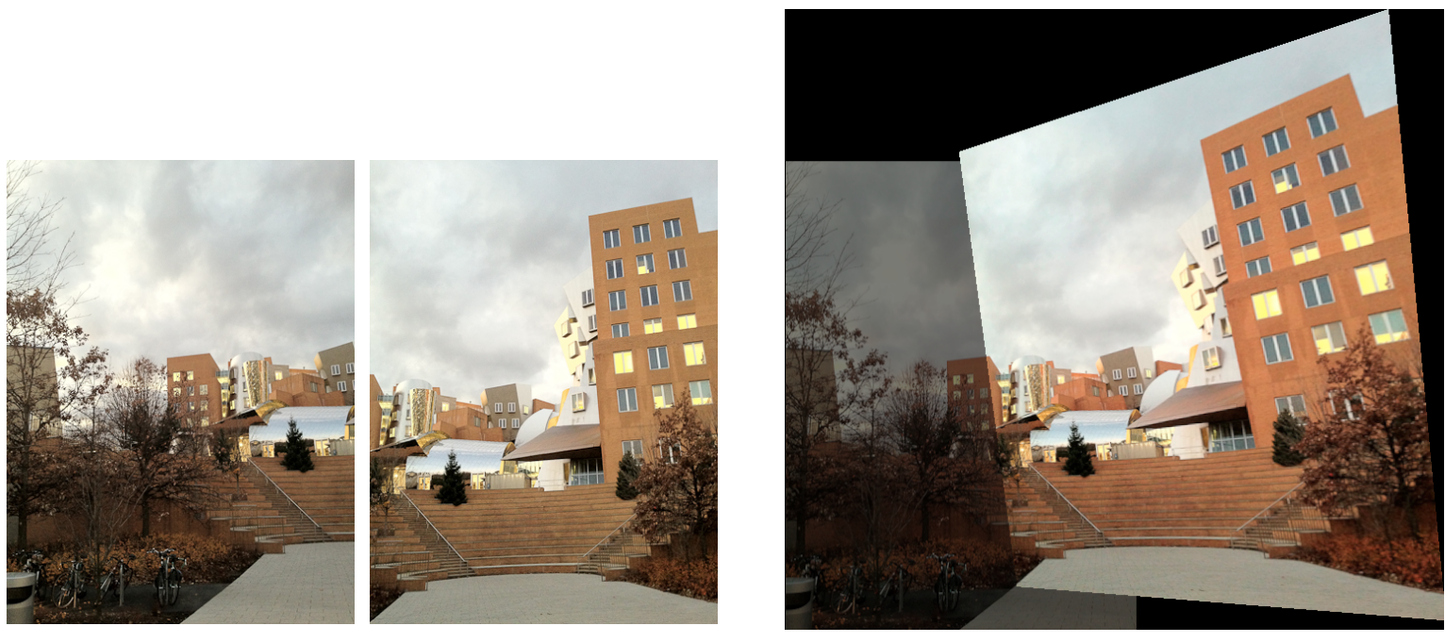


In pset 5

- First use to warp from a plane to a 3D plane

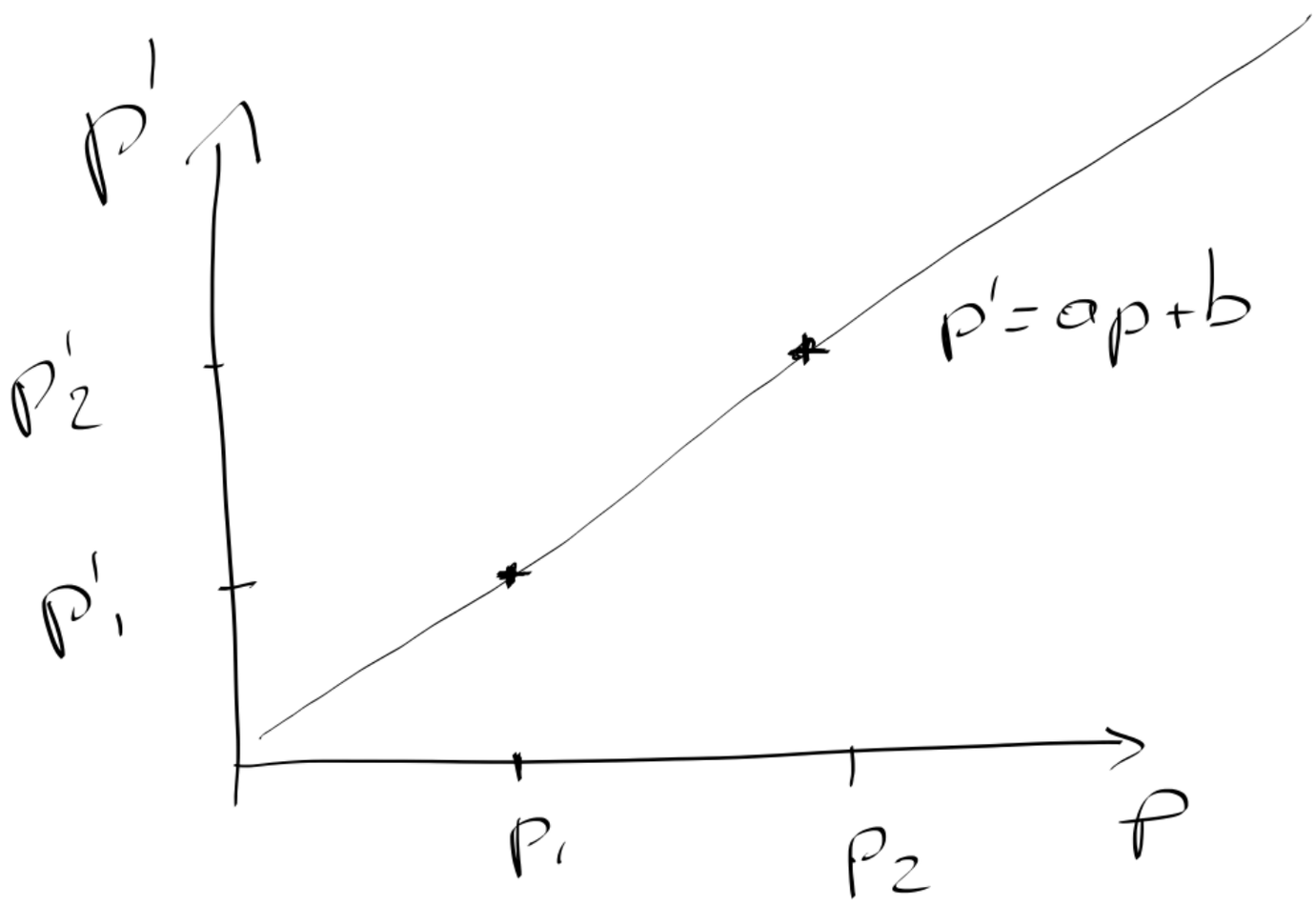


- Then use to warp pano inputs



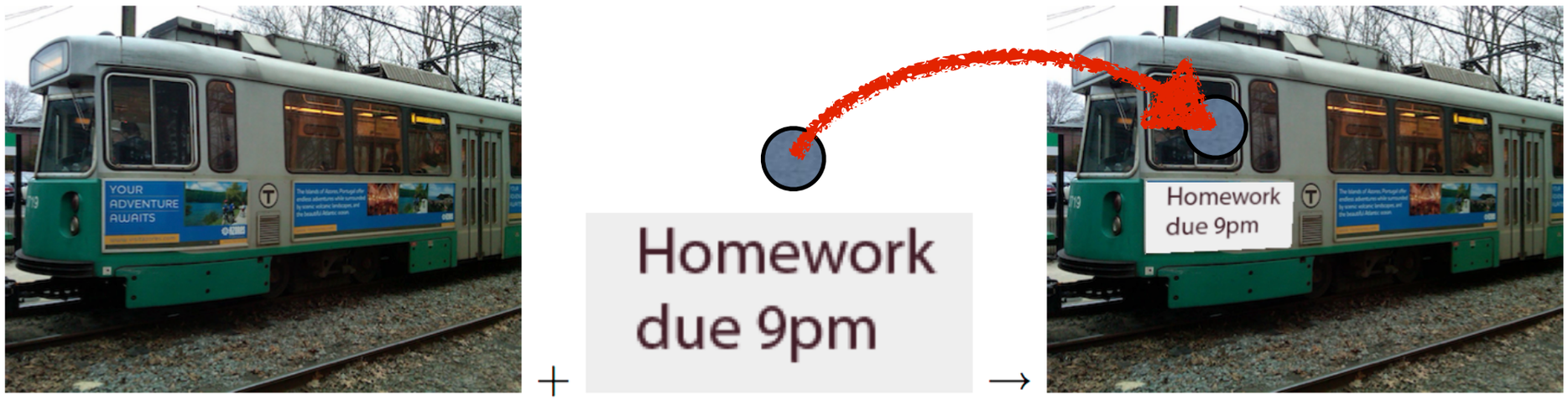
Metaphor

- Example of fitting lines, p’=ap+b, different pairs of correspondences (p1->p’1, p2->p’2) define the same line.



Beware

- Homographies are defined for the whole plane
  - just apply the formula!
- But we often only want to apply it inside the rectangle of the input



- And as usual, recall forward vs. inverse warp

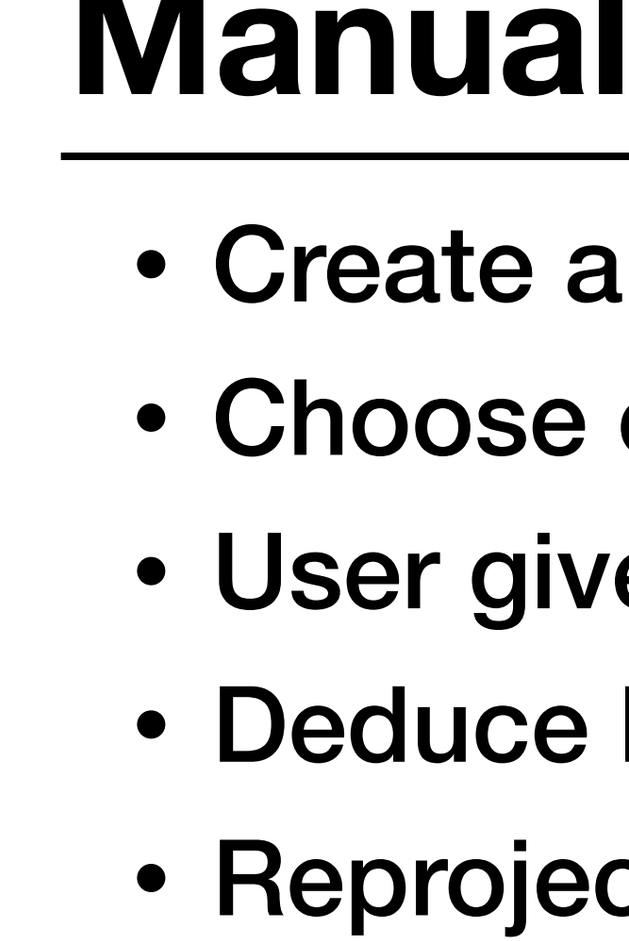
Questions?

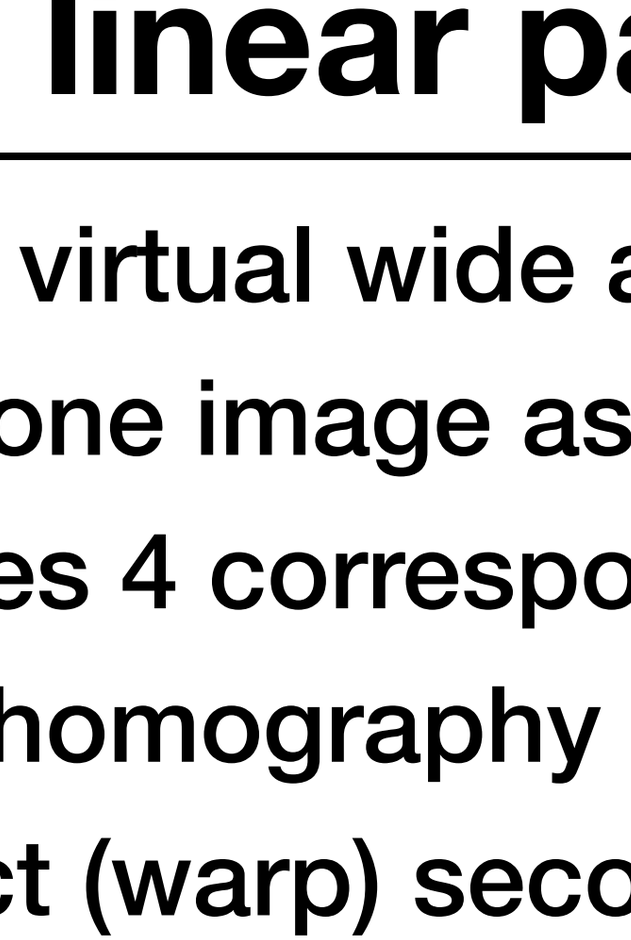


Recap

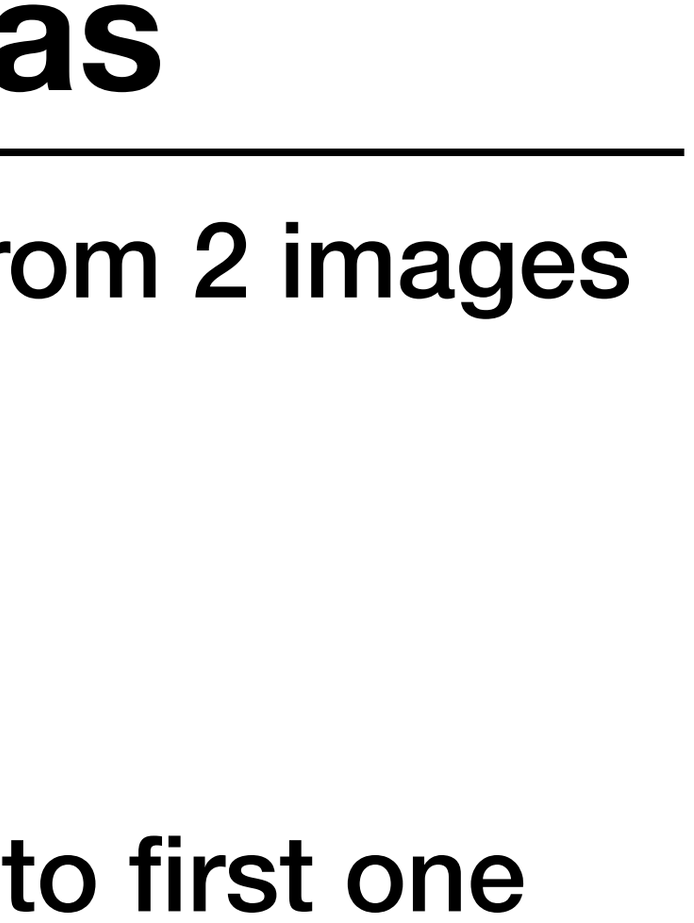
Manual linear panoramas

- Create a virtual wide angle view from 2 images
- Choose one image as reference
- User gives 4 correspondences
- Deduce homography matrix
- Reproject (warp) second image into first one





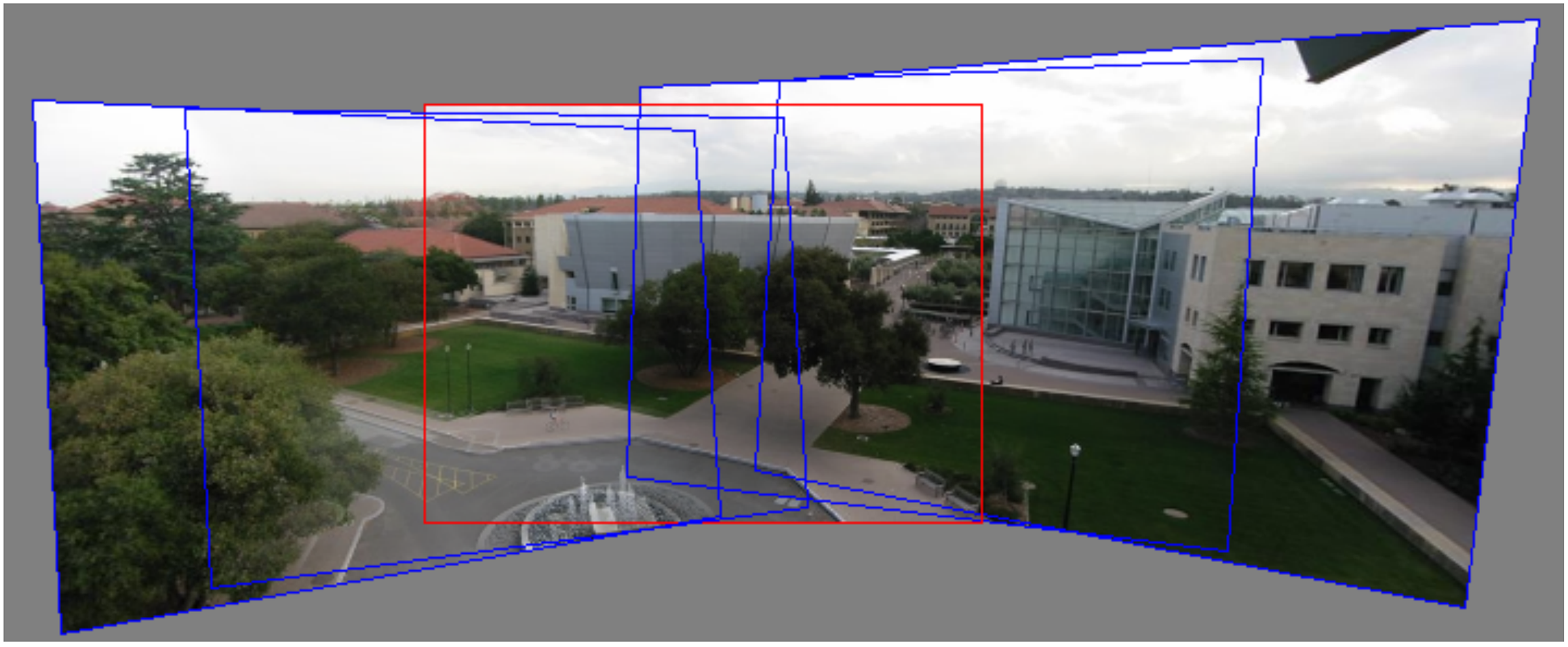




Under the hood

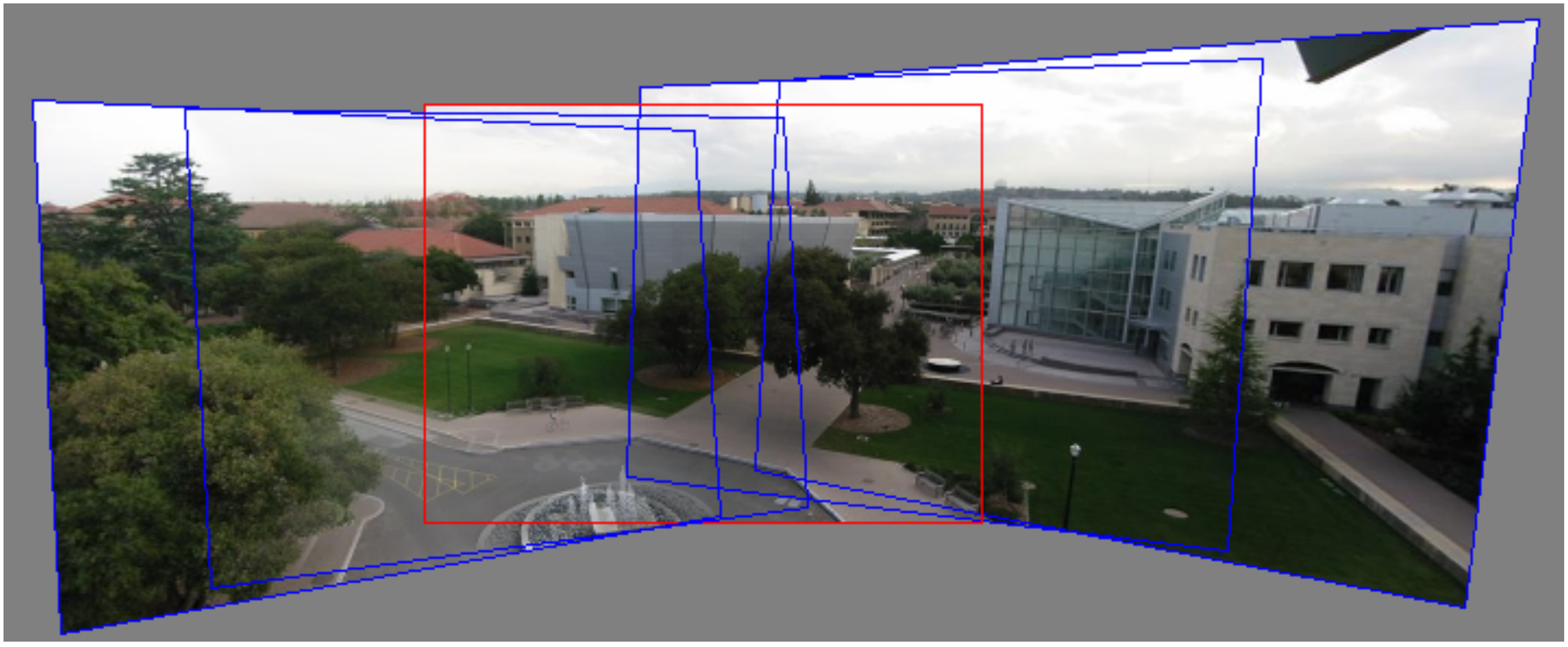
- **Homogenous coordinates**
  - encode 2D points with 3 coordinates (x, y, w)
  - represents Euclidean points (x/w, y/w)
  - make it easy to express perspective and to go between 3D and 2D
- **Homography**
  - 3x3 matrix on homogenous coordinates
  - represent any perspective mapping of a plane
  - to apply a homography to a 2D point x, y: compute $H(x, 1, y)^T$ and divide by third coordinate

Multiple images



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

Homography warp



- **For each output pixel**
- **compute input location with homography matrix**
  – p’=Hp, followed by division
- **copy pixel color (with appropriate antialiasing)**

Later

- Automagic correspondences
  - corner detection
  - patch descriptor

- Nice blending
  - smooth transition
  - 2 scale

- Other projections
  - spherical, cylindrical, miniplanets

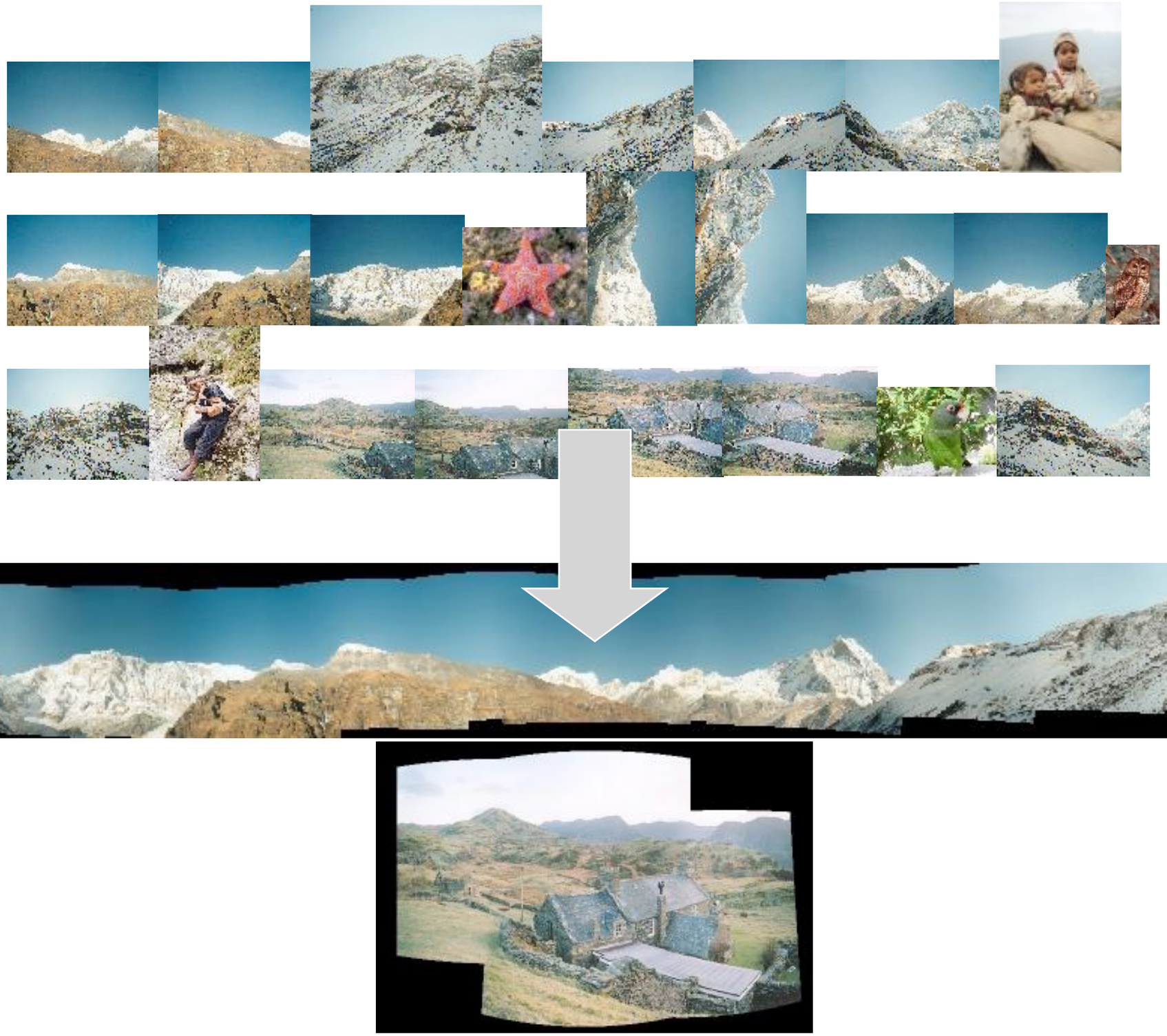






http://designedbynatalie.com/tag/panoramas/

Magic: automatic panos

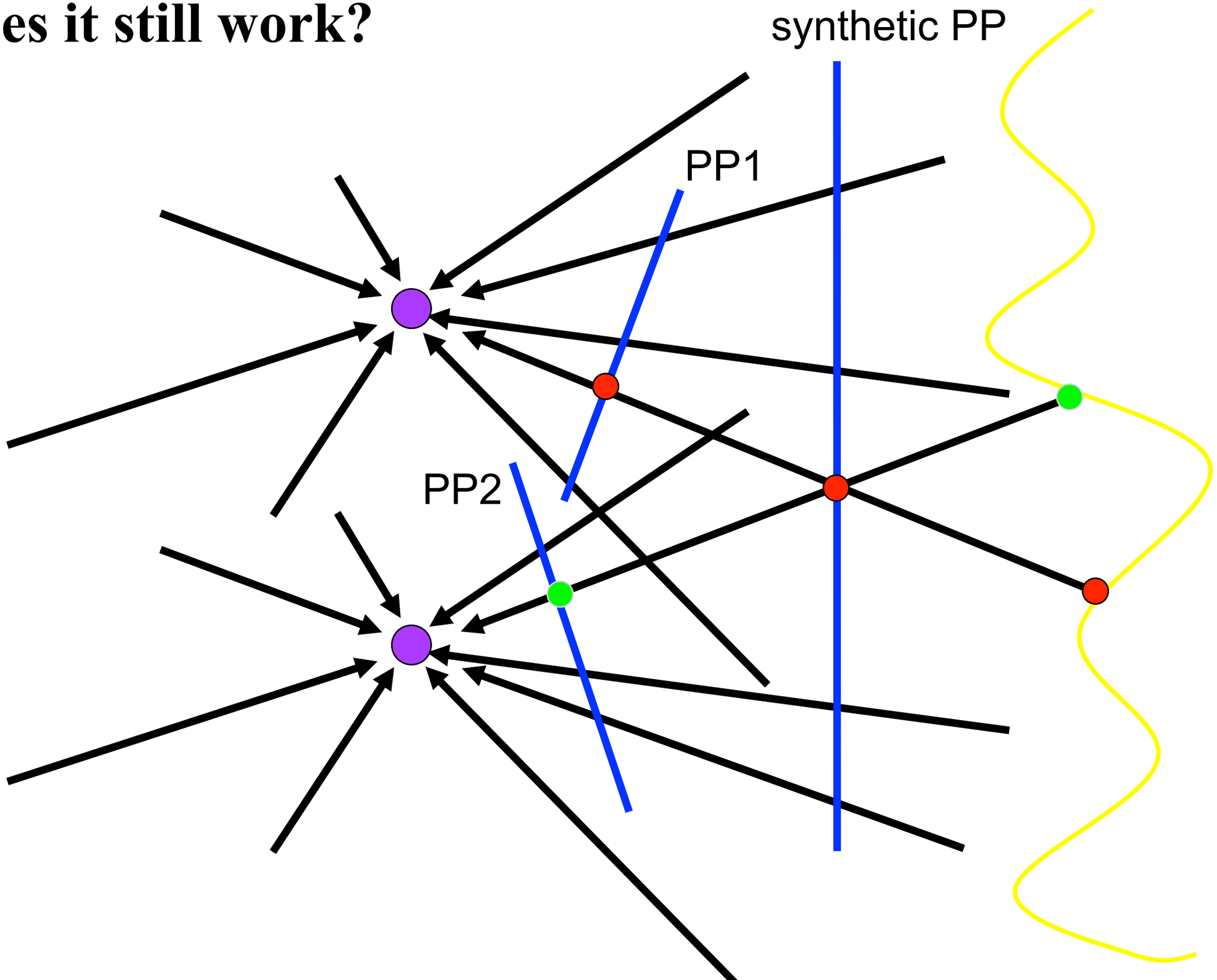


http://research.microsoft.com/~brown/papers/iccv2003.pdf

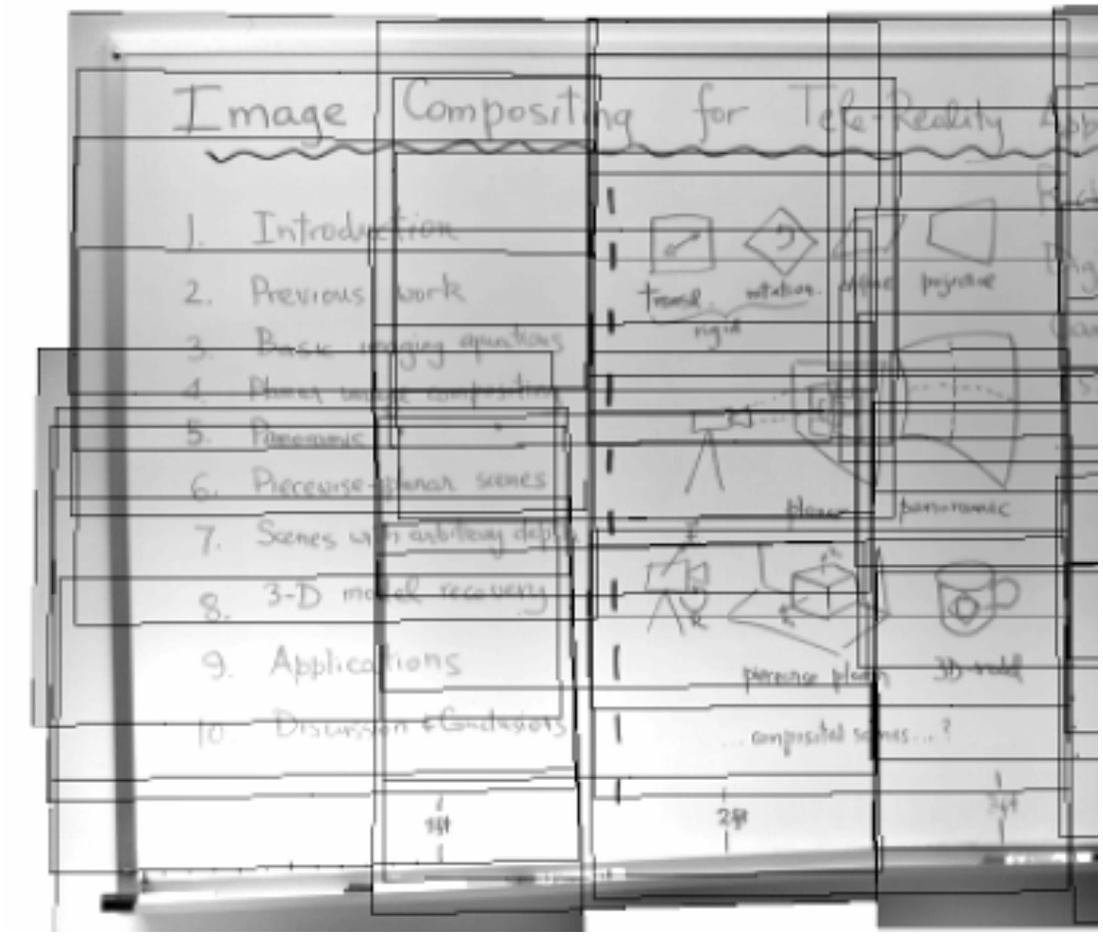
Questions?

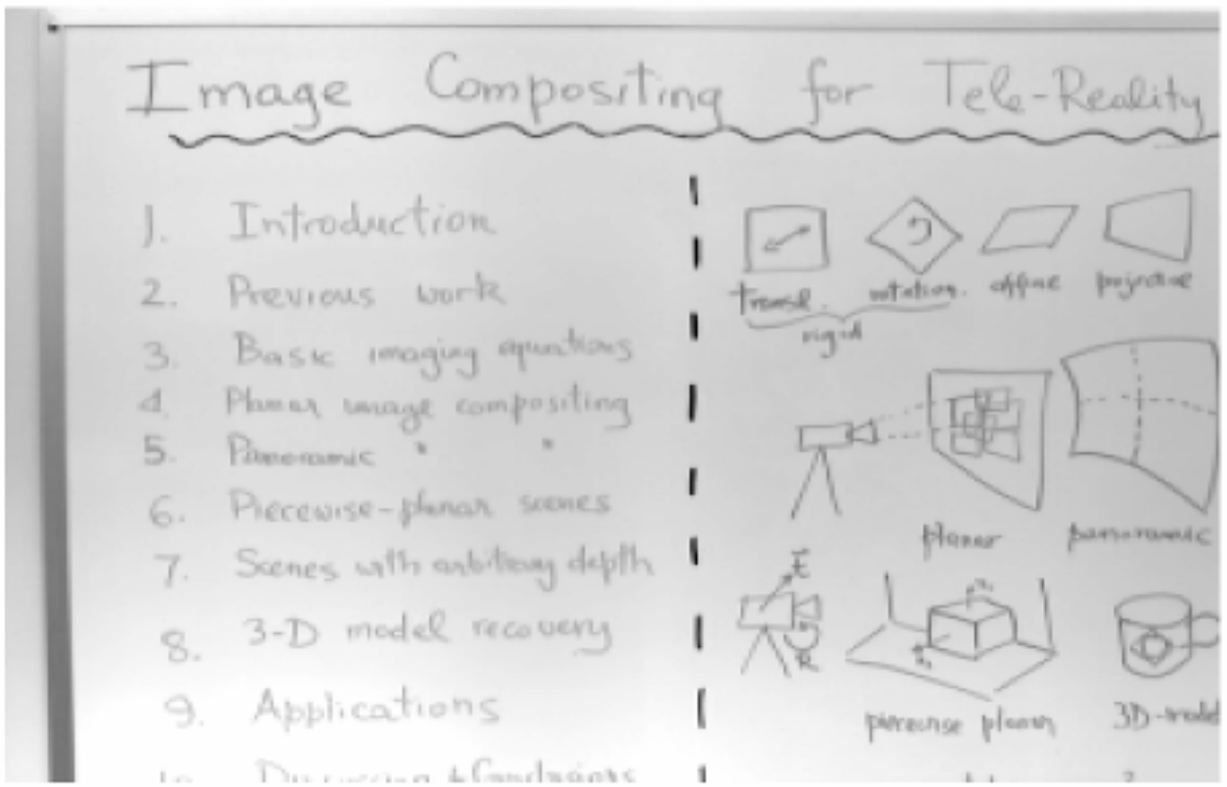
changing camera center

- **Does it still work?**



Only for Planar mosaic





Questions?

Applied homographies

- **https://www.youtube.com/watch?feature=player_embedded&v=tBNHPk-Lnkk#!**



Perspective correction

Digression: perspective correction

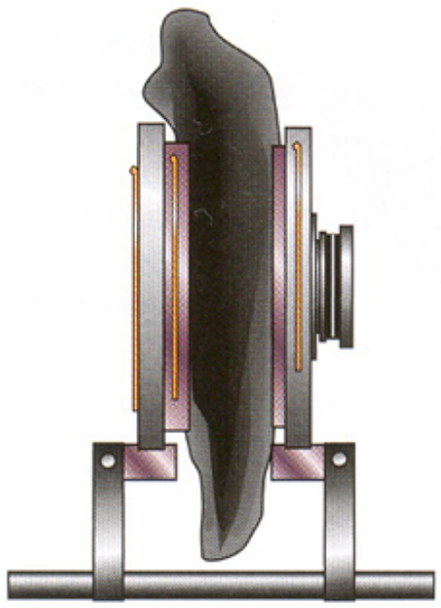
- **Perspective makes parallel lines converge**
- **Sometimes objectionable**
- **In particular, architecture photography**
  - photo looks more formal if verticals stay vertical



- **But if lines are parallel to the image plane, parallel lines don't converge**

Perspective correction



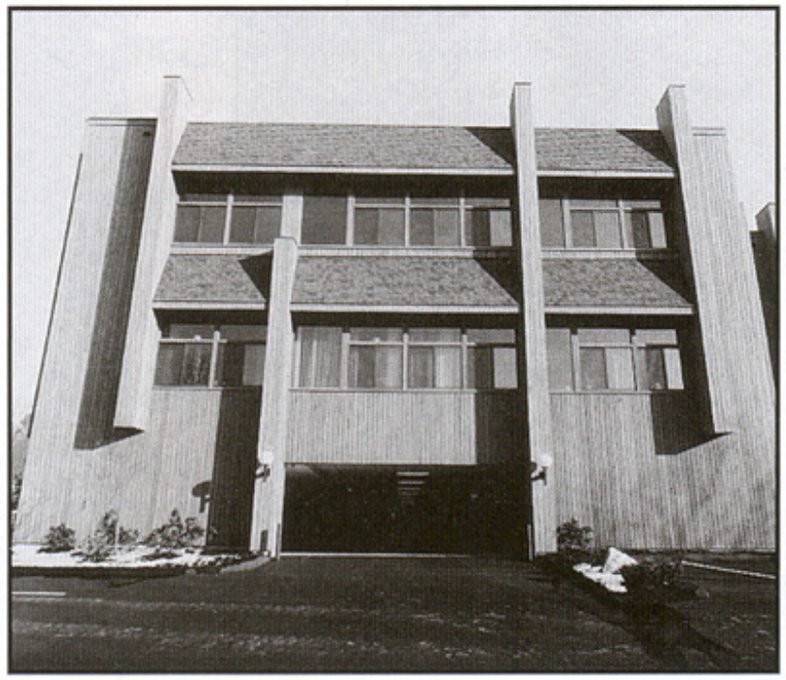


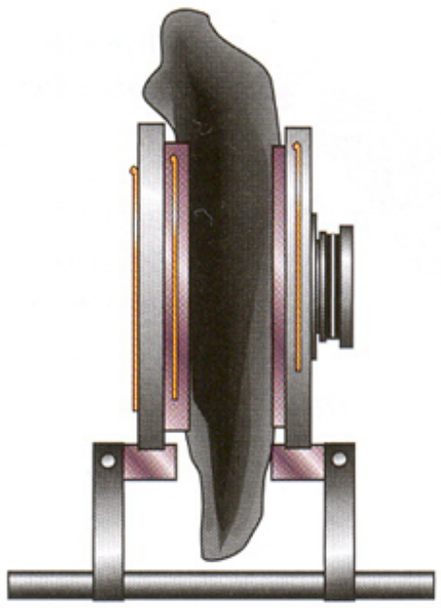
Standing at street level and shooting straight at a building produces too much street and too little building. Sometimes it is possible to move back far enough to show the entire building while keeping the camera level, but this adds even more foreground and usually something gets in the way.

From Photography, London et al.

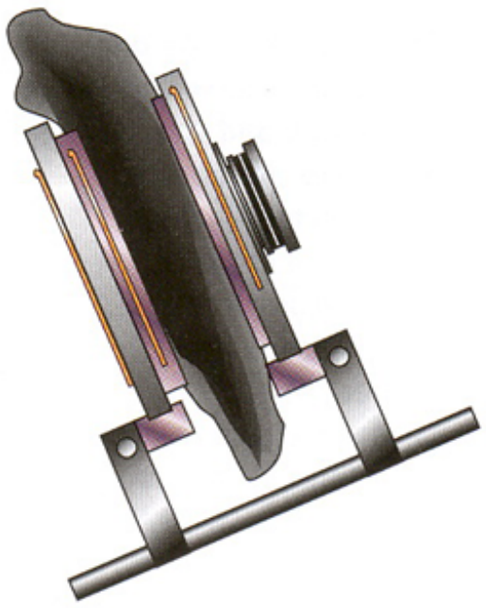
CONTROLLING CONVERGING LINES: THE KEYSTONE EFFECT







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.

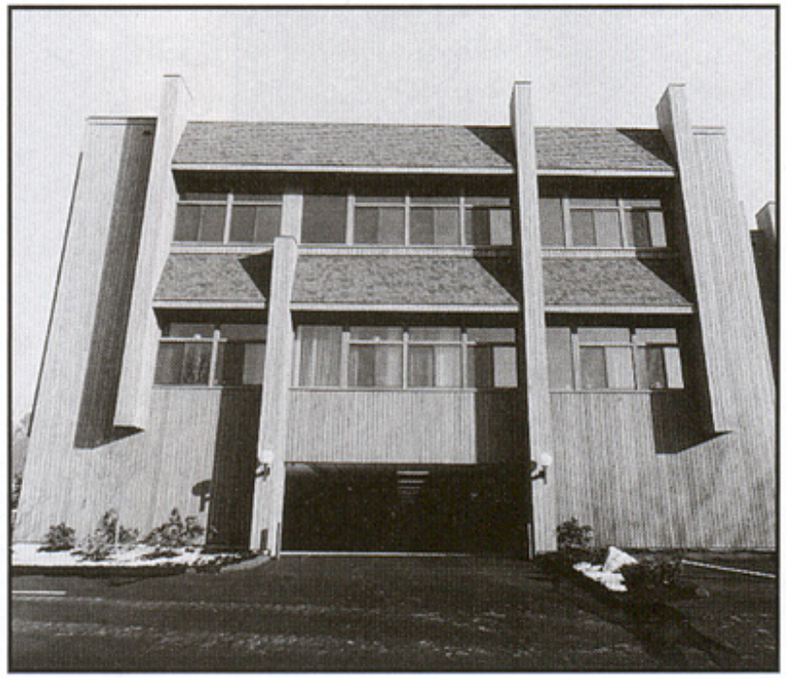


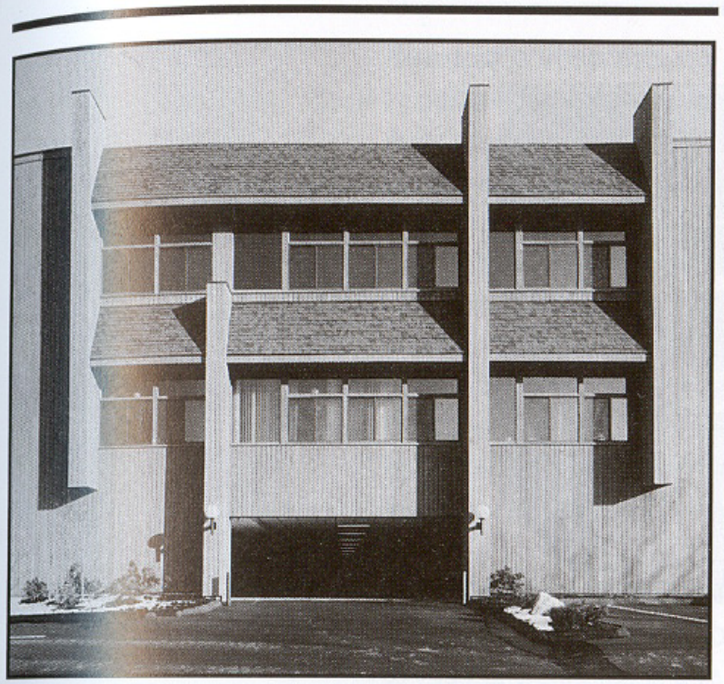
Tilting the whole camera up shows the entire building but distorts its shape. Since the top is farther from the camera than the bottom, it appears smaller; the vertical lines of the building seem to be coming closer together, or converging, near the top. This is named the keystone effect, after the wedge-shaped 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.

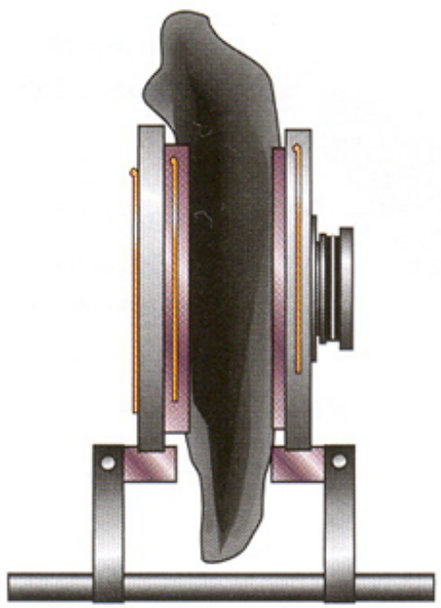
From Photography, London et al.

CONTROLLING CONVERGING LINES: THE KEYSTONE EFFECT

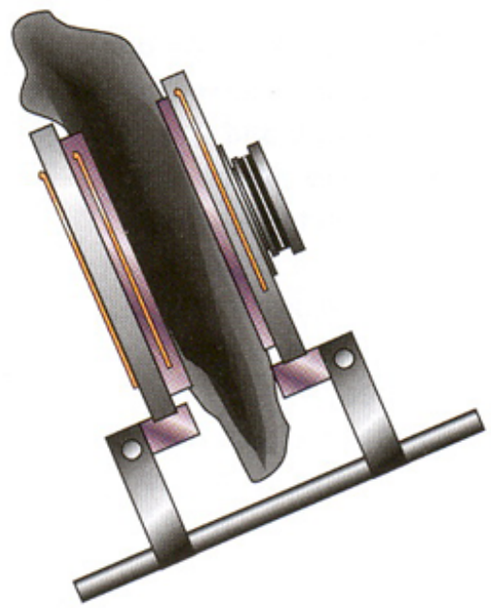




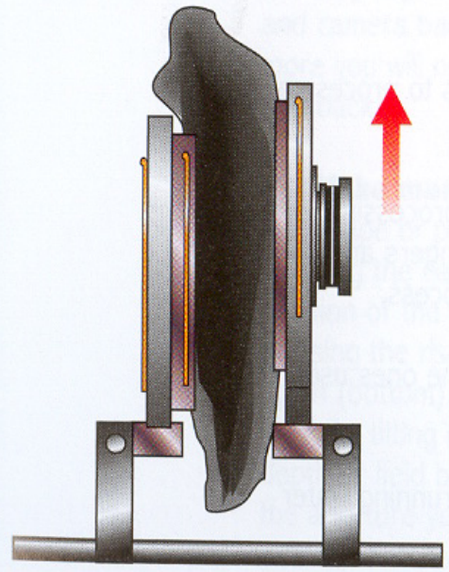




Standing at street level and shooting straight at a building produces too much street and too little building. Sometimes it is possible to move back far enough to show the entire building while keeping the camera level, but this adds even more foreground and usually something gets in the way.



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

- **Equivalent to using a wider-angle view and cropping**



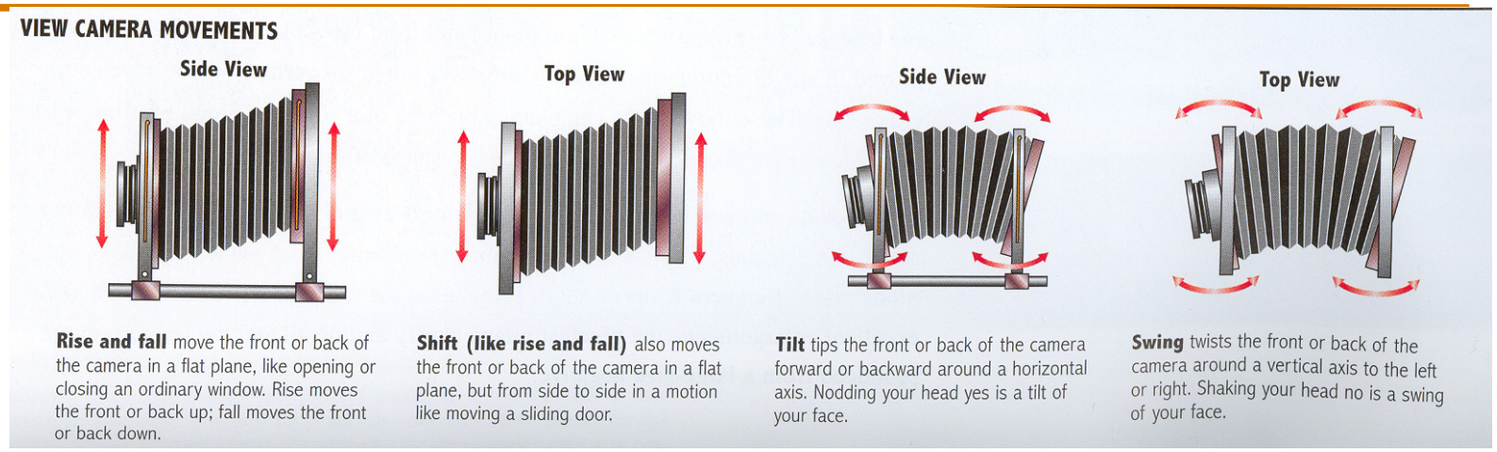
To straighten up the converging vertical lines, keep the camera back parallel to the face of the building. To keep the face of the building in focus, make sure the lens is parallel to the camera back. One way to do this is to level the camera and then use the rising front or falling back movements or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

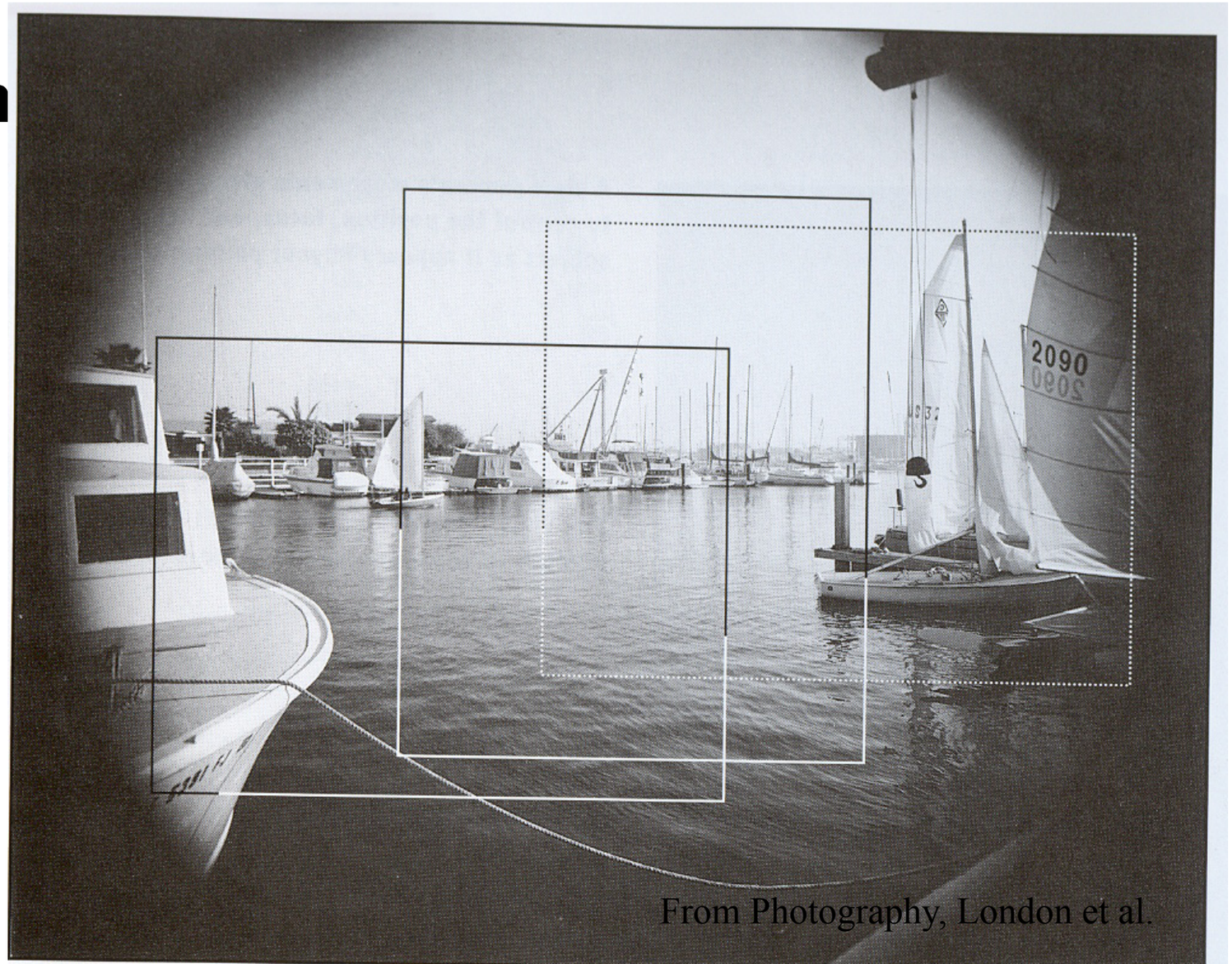
Another solution is to point the camera upward toward the top of the building, then use the tilting movements—first to tilt the back to a vertical position (which squares the shape of the building), then to tilt the lens so it is parallel to the camera back (which brings the face of the building into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

From Photography, London et al.

Perspective correction

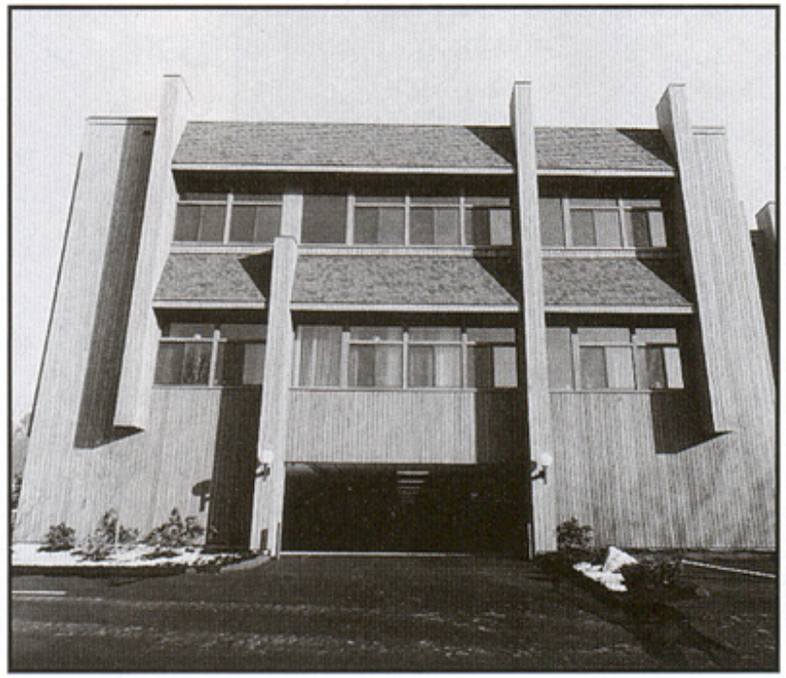
- **The lens creates an image bigger than the sensor**
- **Tilting and shifting aligns the sensor with the part we want**

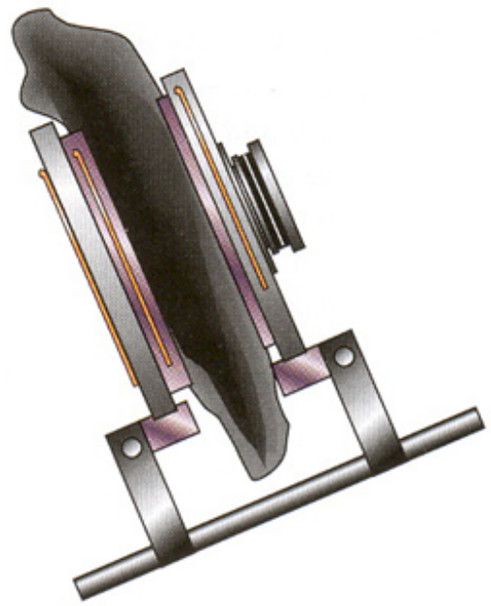




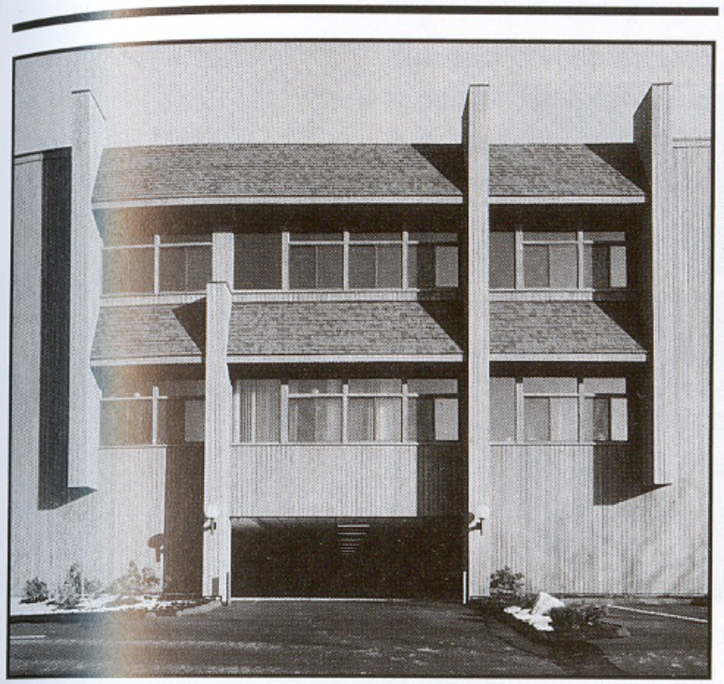
From Photography, London et al.

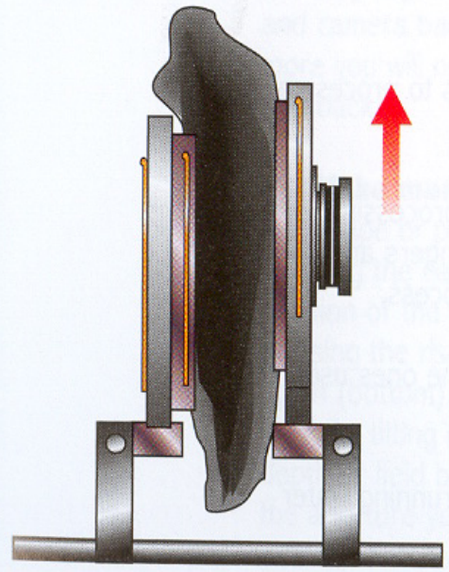
- **The two images are related by a homography**





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.





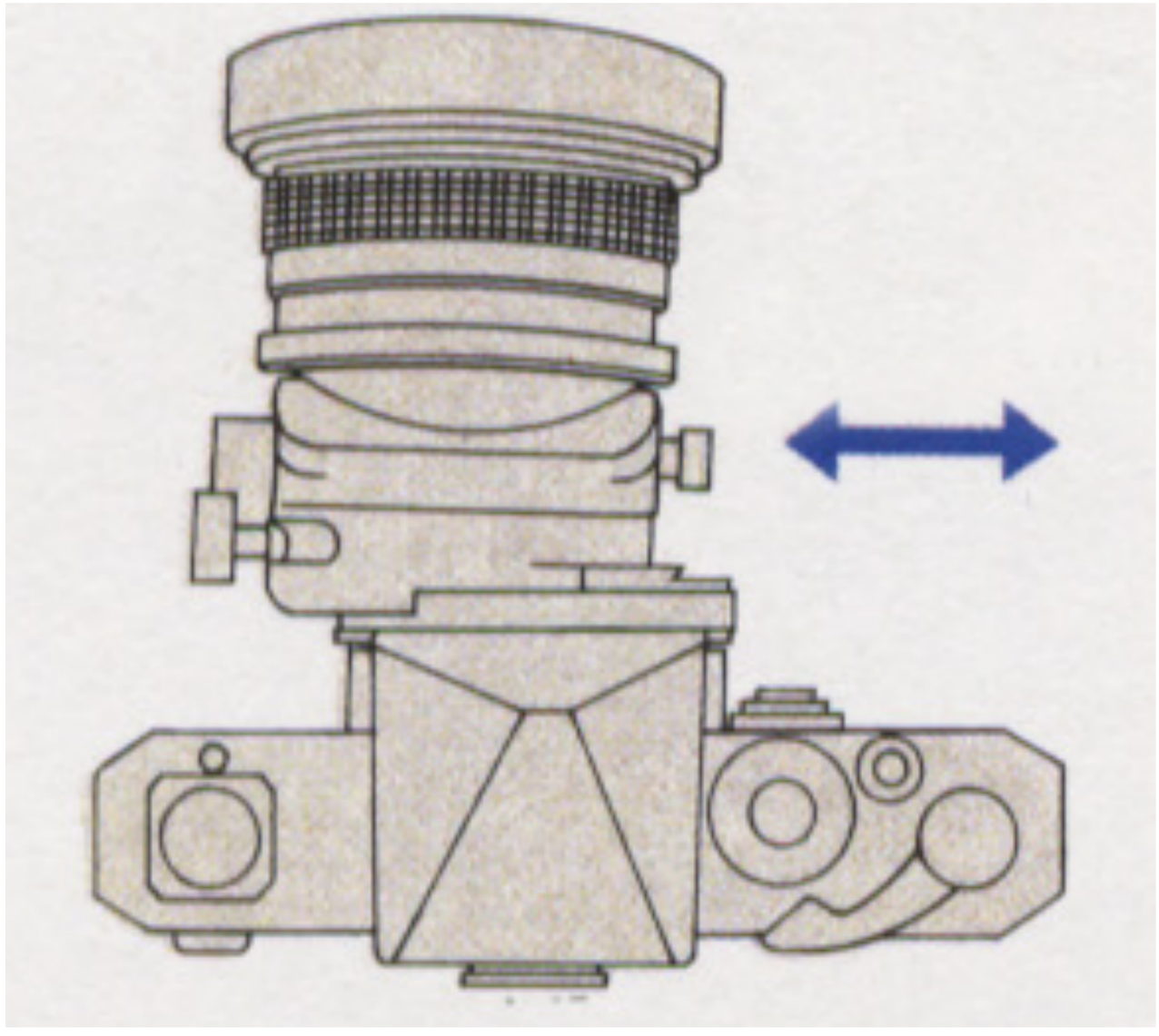
To straighten up the converging vertical lines, keep the camera back parallel to the face of the building. To keep the face of the building in focus, make sure the lens is parallel to the camera back. One way to do this is to level the camera and then use the rising front or falling back movements or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

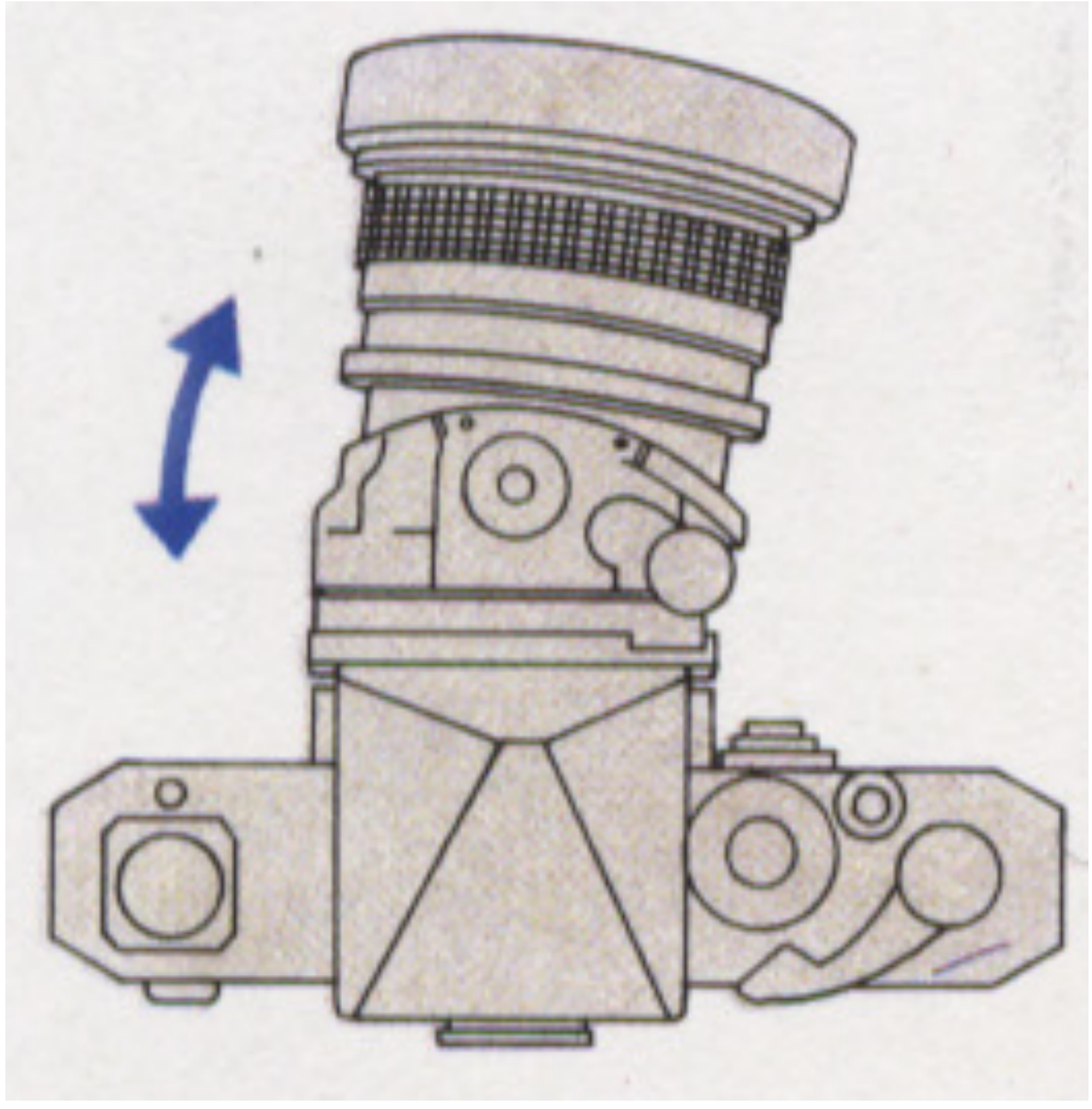
Another solution is to point the camera upward toward the top of the building, then use the tilting movements—first to tilt the back to a vertical position (which squares the shape of the building), then to tilt the lens so it is parallel to the camera back (which brings the face of the building into 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





Question?

- **The Ambassadors, Hans Holbein the Younger, 1533**



Question?

- **The Ambassadors, Hans Holbein the Younger, 1533**





Cool applications of homographies

- **With Mok Oh.**

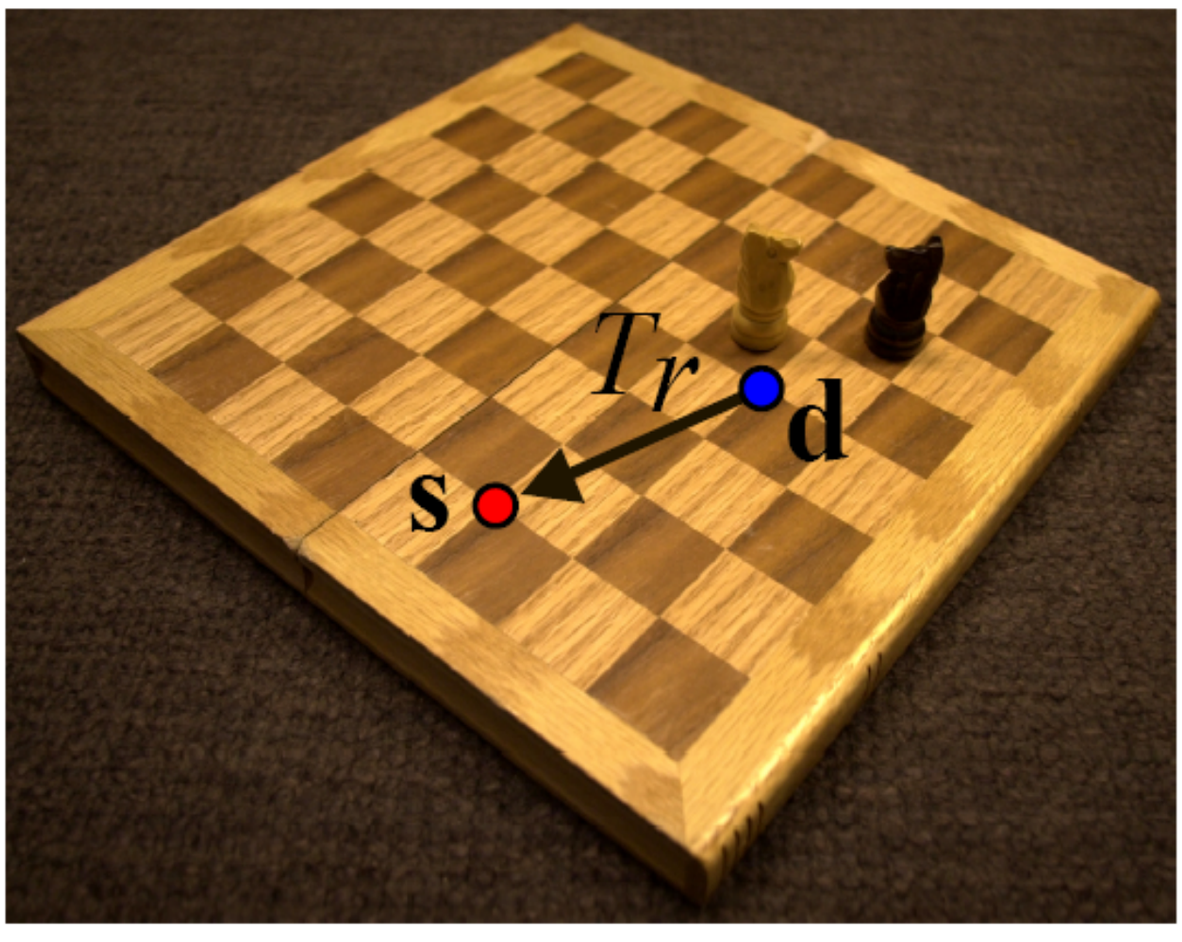
Limitations of 2D Clone Brushing

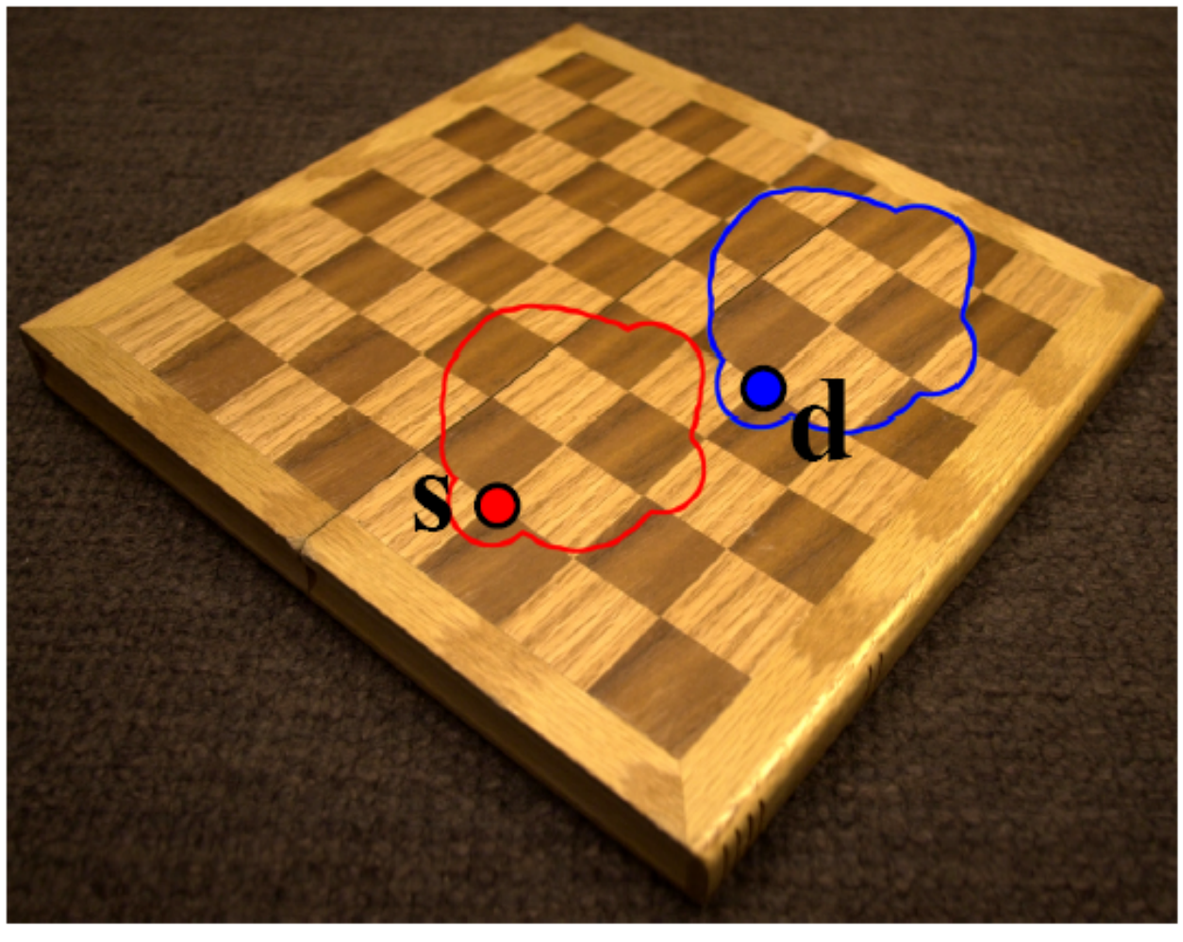
- **Distortions due to foreshortening and surface orientation**



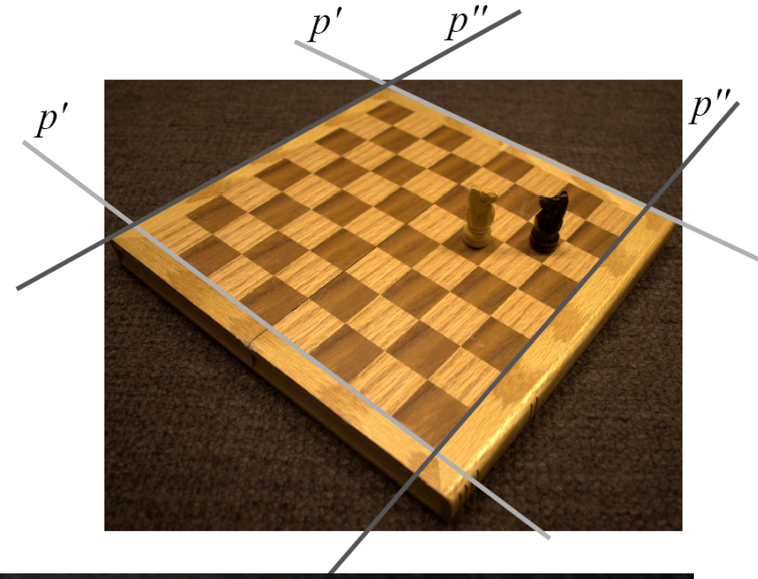
Clone brush (Photoshop)

- **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**

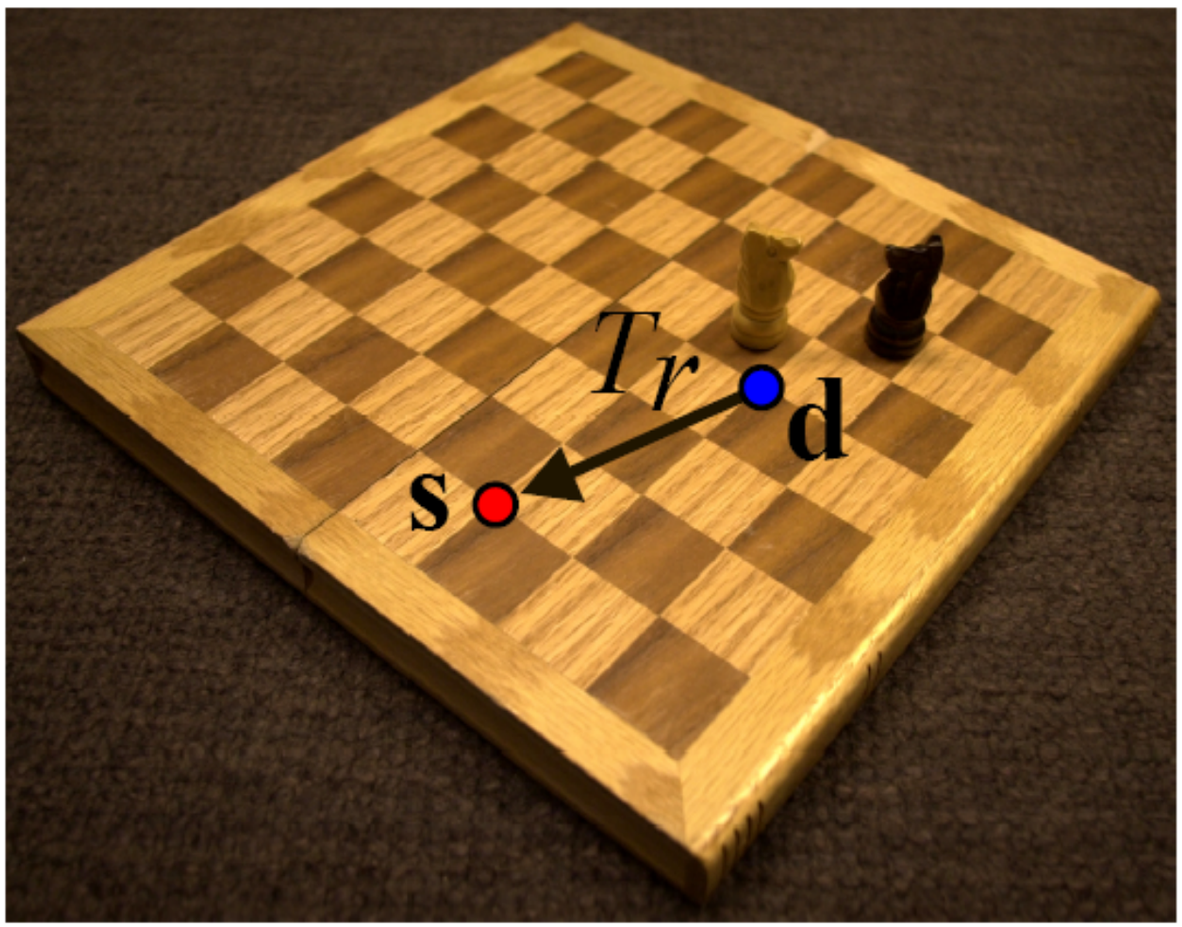










Figure 15: The cars and the street furniture have been removed. This example took less than 10 minutes.

Questions?

Other application: View morphing

- **We want to morph between two views of the same object**
- **Standard morphing won't give a realistic result**
  - Because it uses linear interpolation of image locations
  - With perspective & 3D, points don't move linearly

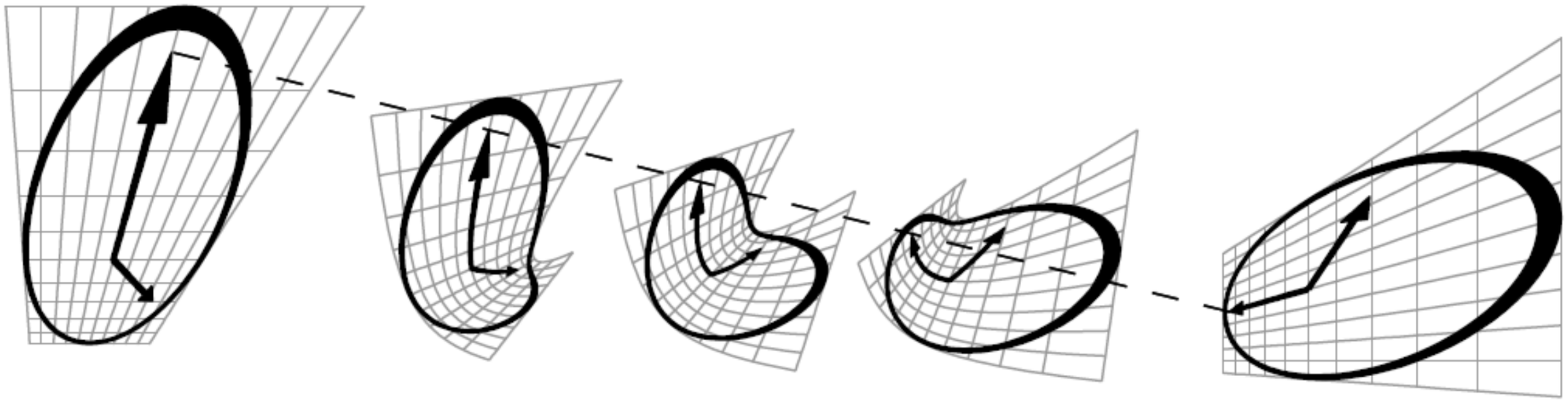


Figure 2: A Shape-Distorting Morph. Linearly interpolating two perspective views of a clock (far left and far right) causes a geometric bending effect in the in-between images. The dashed line shows the linear path of one feature during the course of the transformation. This example is indicative of the types of distortions that can arise with image morphing techniques.

View morphing

- **Seitz & Dyer**

  http://www.cs.washington.edu/homes/seitz/vmorph/vmorph.htm

- **Interpolation consistent with 3D view interpolation**

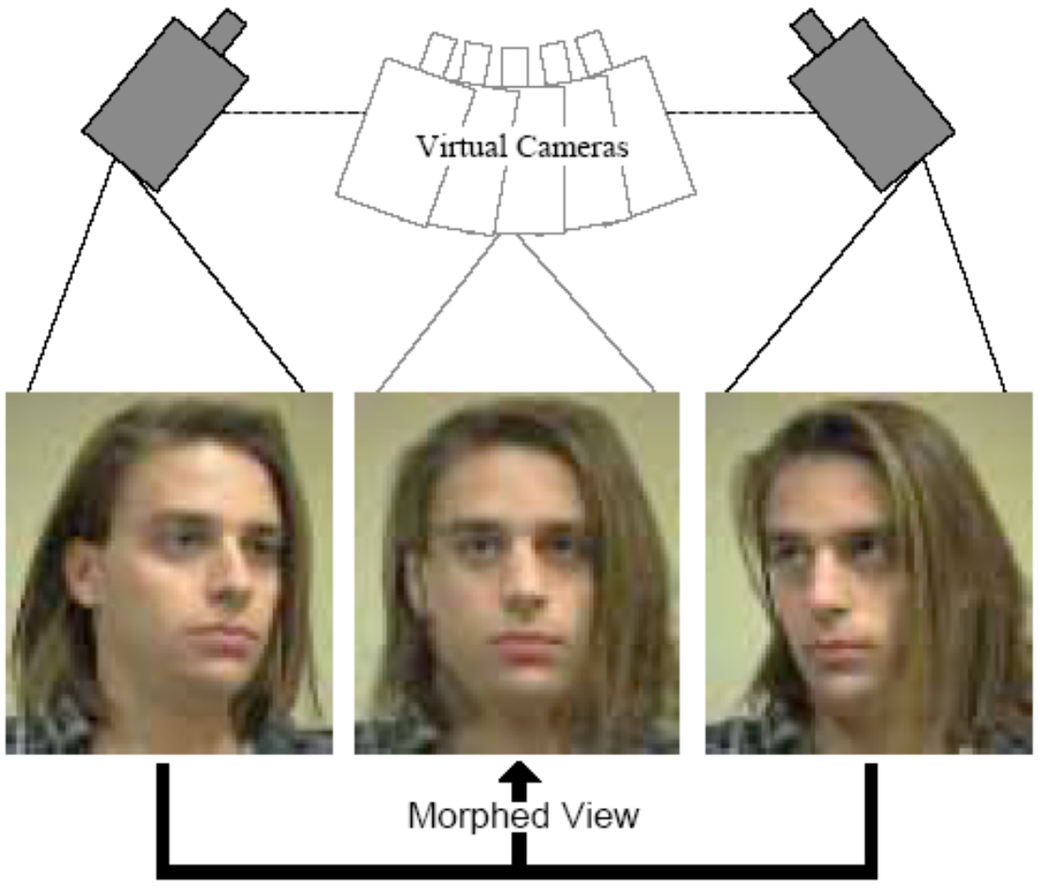
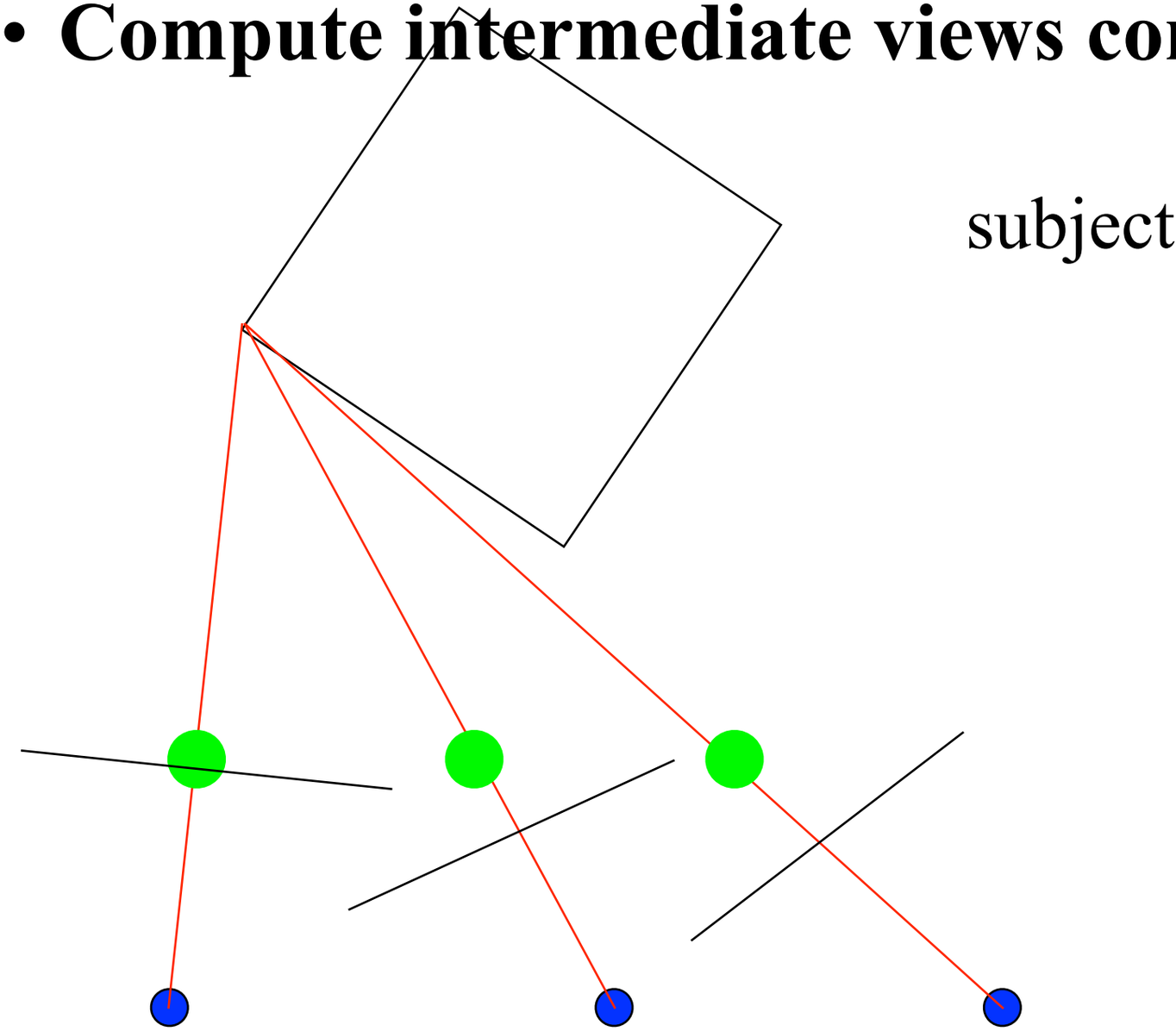


Figure 1: View morphing between two images of an object taken from two different viewpoints produces the illusion of physically moving a virtual camera.

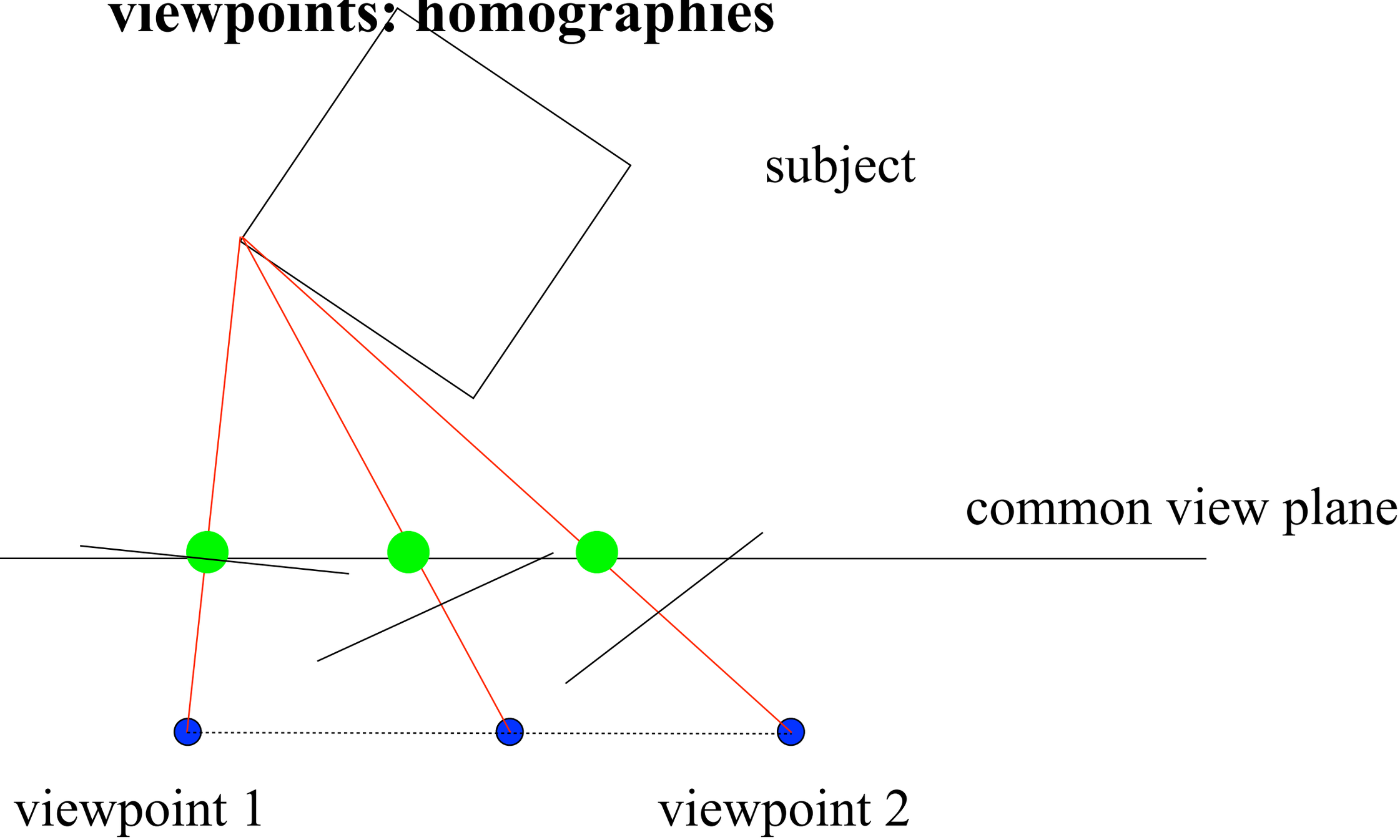
View morphing

- **Given view 1 and view 2 (and no depth!)**
- **Compute intermediate views consistent with 3D**



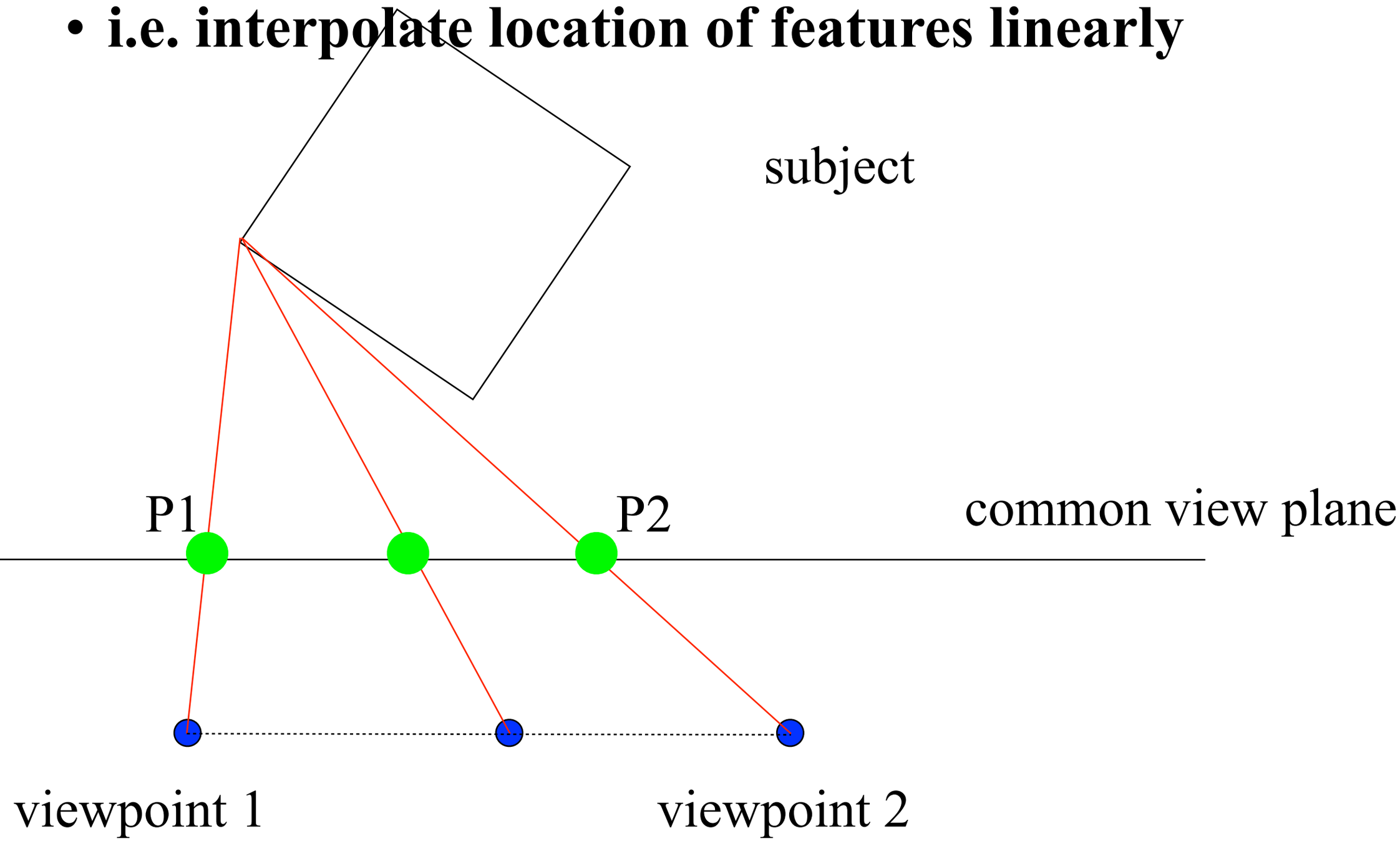
View morphing

- **Reproject to a common view plane parallel to the 2 viewpoints: homographies**



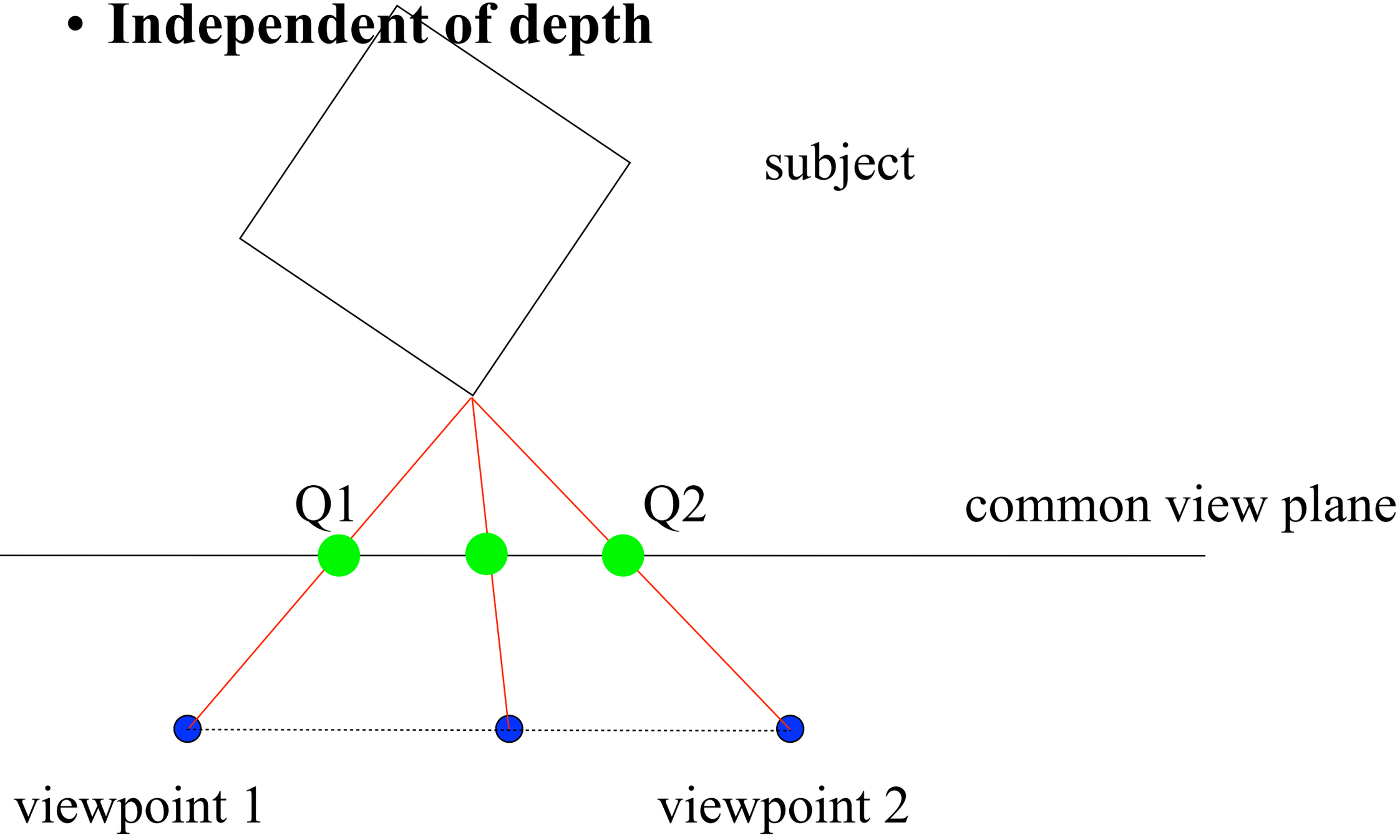
View morphing

- **Similar triangles: everything becomes linear**
- **i.e. interpolate location of features linearly**



View morphing

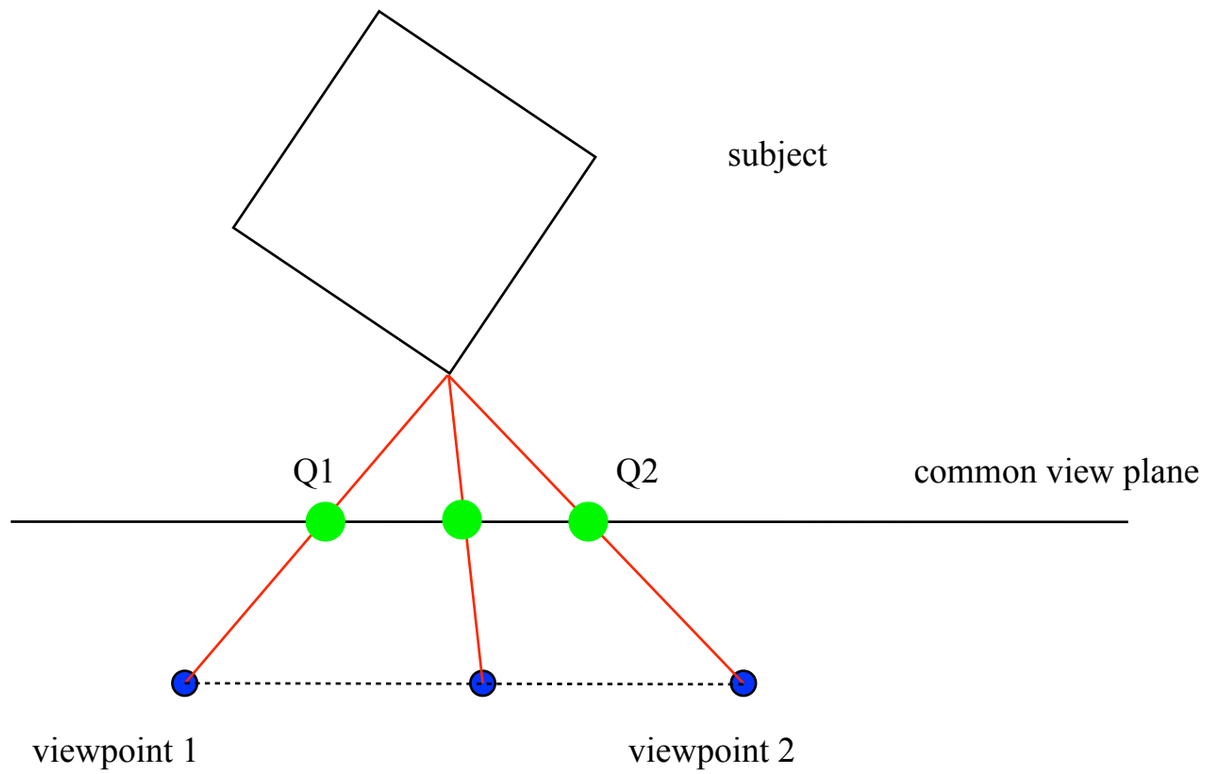
- **Similar triangles: everything becomes linear**
- **Independent of depth**



View morphing recap

- **When views are reprojected onto a common plane parallel to the line between the viewpoints: Interpolating the viewpoint results in linear interpolation of feature locations**
- **i.e.**
  - for viewpoint (1-t)V1+tV2
  - a 3D point that was at P1 from V1 and P2 from V2 is now at (1-t)P1+tP2

- **That is, a simplistic warp works**



View morphing

- **Prewarp with a homography to "pre-align" images**
- **So that the two views are parallel**
  - Because linear interpolation works when views are parallel

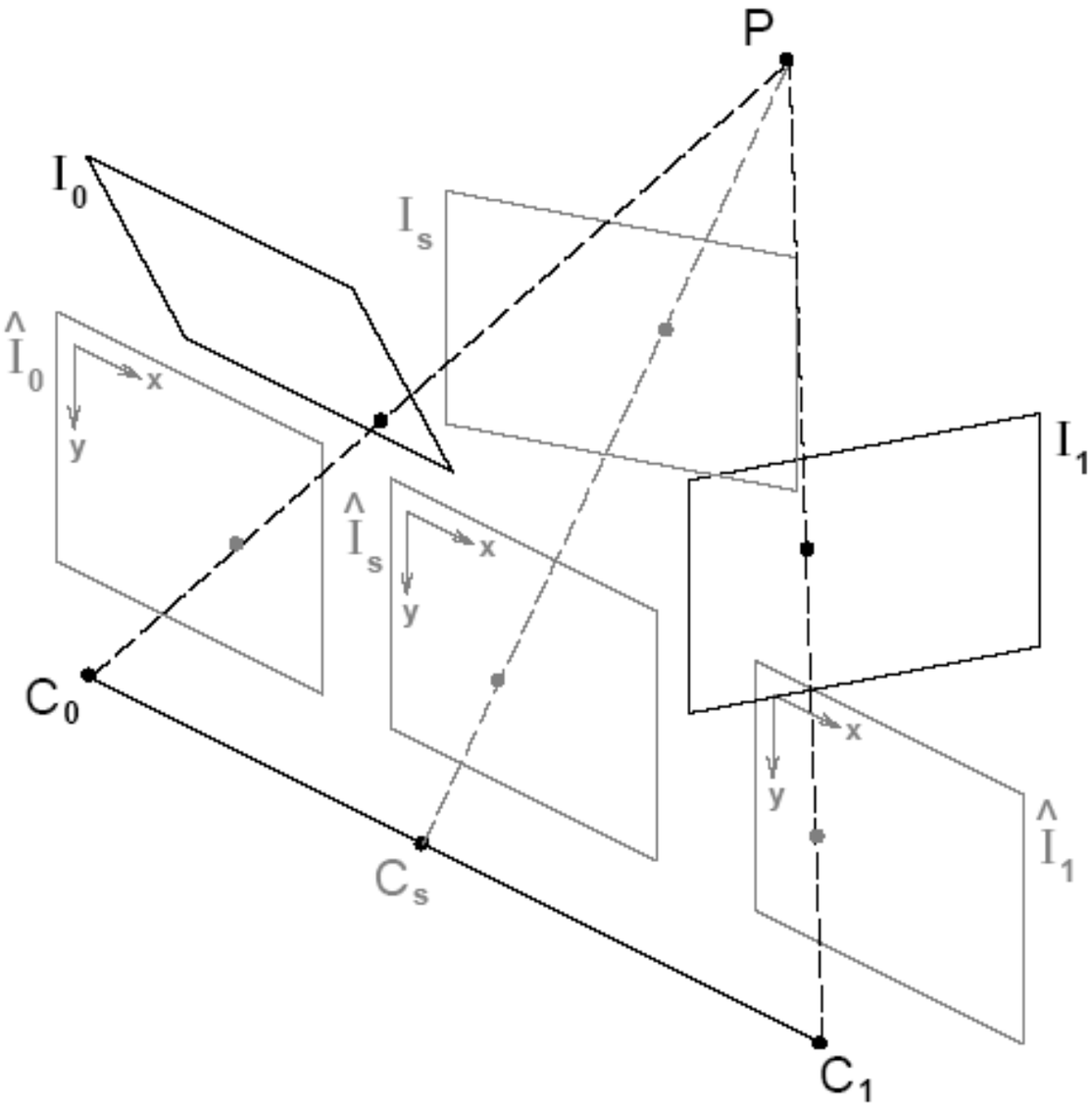


Figure 4: View Morphing in Three Steps. (1) Original images $\mathcal{I}_0$ and $\mathcal{I}_1$ are prewarped to form parallel views $\hat{\mathcal{I}}_0$ and $\hat{\mathcal{I}}_1$. (2) $\hat{\mathcal{I}}_s$ is produced by morphing (interpolating) the prewarped images. (3) $\hat{\mathcal{I}}_s$ is postwarped to form $\mathcal{I}_s$.

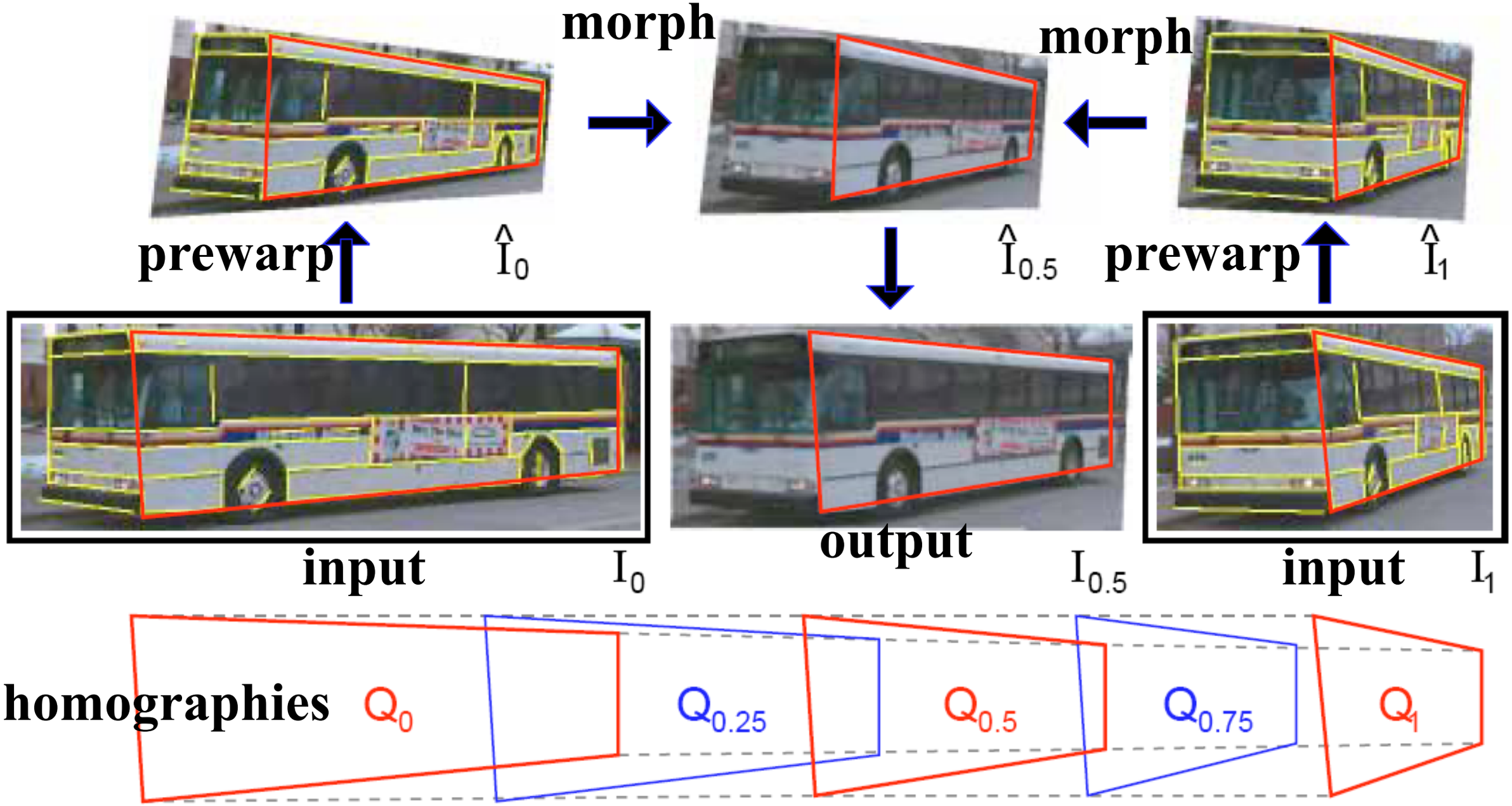


Figure 6: View Morphing Procedure: A set of features (yellow lines) is selected in original images $\mathcal{I}_0$ and $\mathcal{I}_1$. Using these features, the images are automatically prewarped to produce $\hat{\mathcal{I}}_0$ and $\hat{\mathcal{I}}_1$. The prewarped images are morphed to create a sequence of in-between images, the middle of which, $\hat{\mathcal{I}}_{0.5}$, is shown at top-center. $\hat{\mathcal{I}}_{0.5}$ is interactively postwarped by selecting a quadrilateral region (marked red) and specifying its desired configuration, $Q_{0.5}$, in $\mathcal{I}_{0.5}$. The postwarps for other in-between images are determined by interpolating the quadrilaterals (bottom).

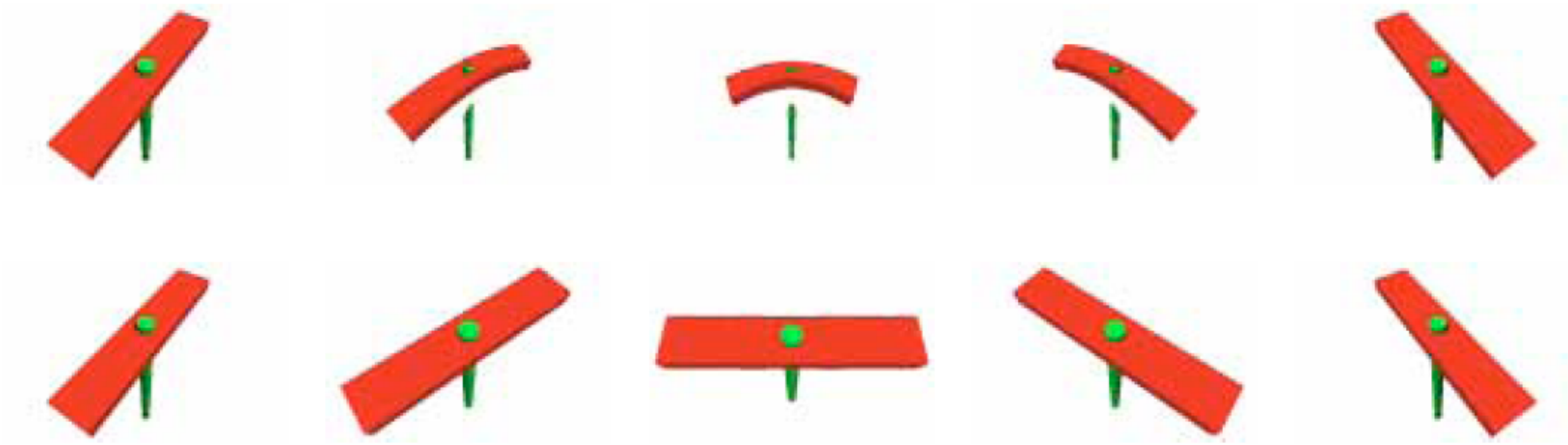
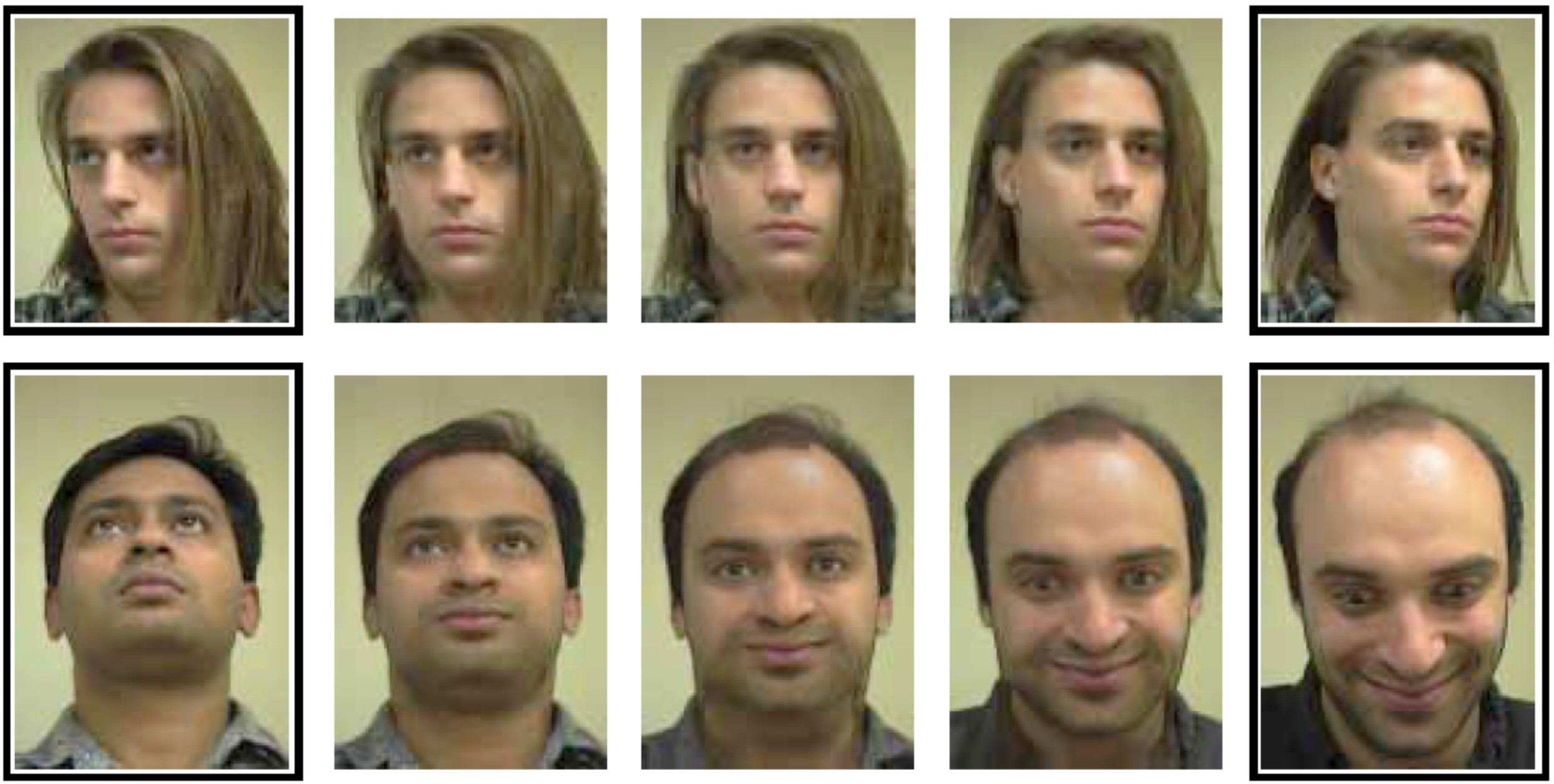


Figure 10: Image Morphing Versus View Morphing. Top: image morph between two views of a helicopter toy causes the in-between images to contract and bend. Bottom: view morph between the same two views results in a physically consistent morph. In this example the image morph also results in an extraneous hole between the blade and the stick. Holes can appear in view morphs as well.



$\mathcal{I}_0$ $\mathcal{I}_{0.25}$ $\mathcal{I}_{0.5}$ $\mathcal{I}_{0.75}$ $\mathcal{I}_1$

Figure 7: Facial View Morphs. Top: morph between two views of the same person. Bottom: morph between views of two different people. In each case, view morphing captures the change in facial pose between original images $\mathcal{I}_0$ and $\mathcal{I}_1$, conveying a natural 3D rotation.

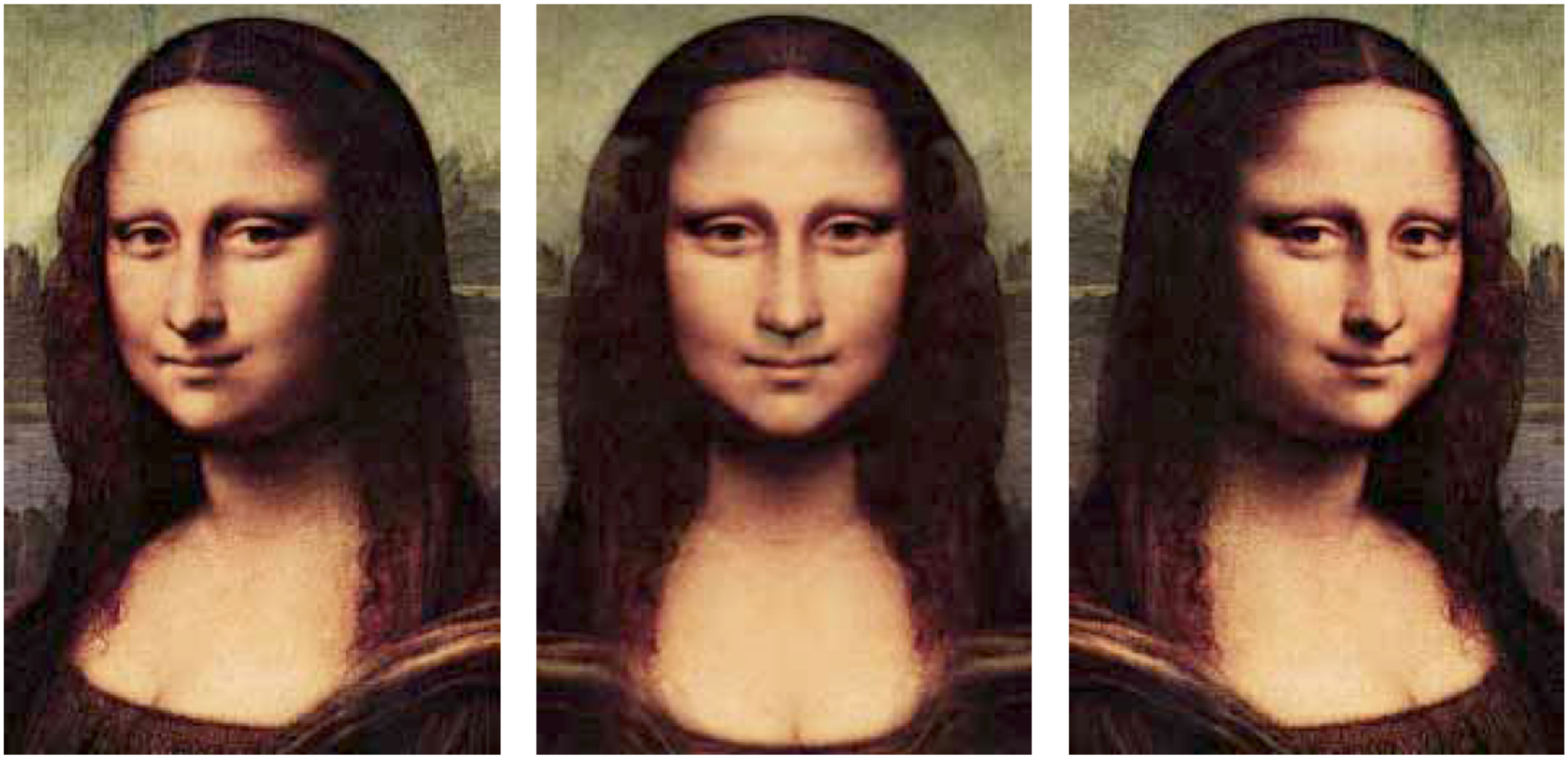
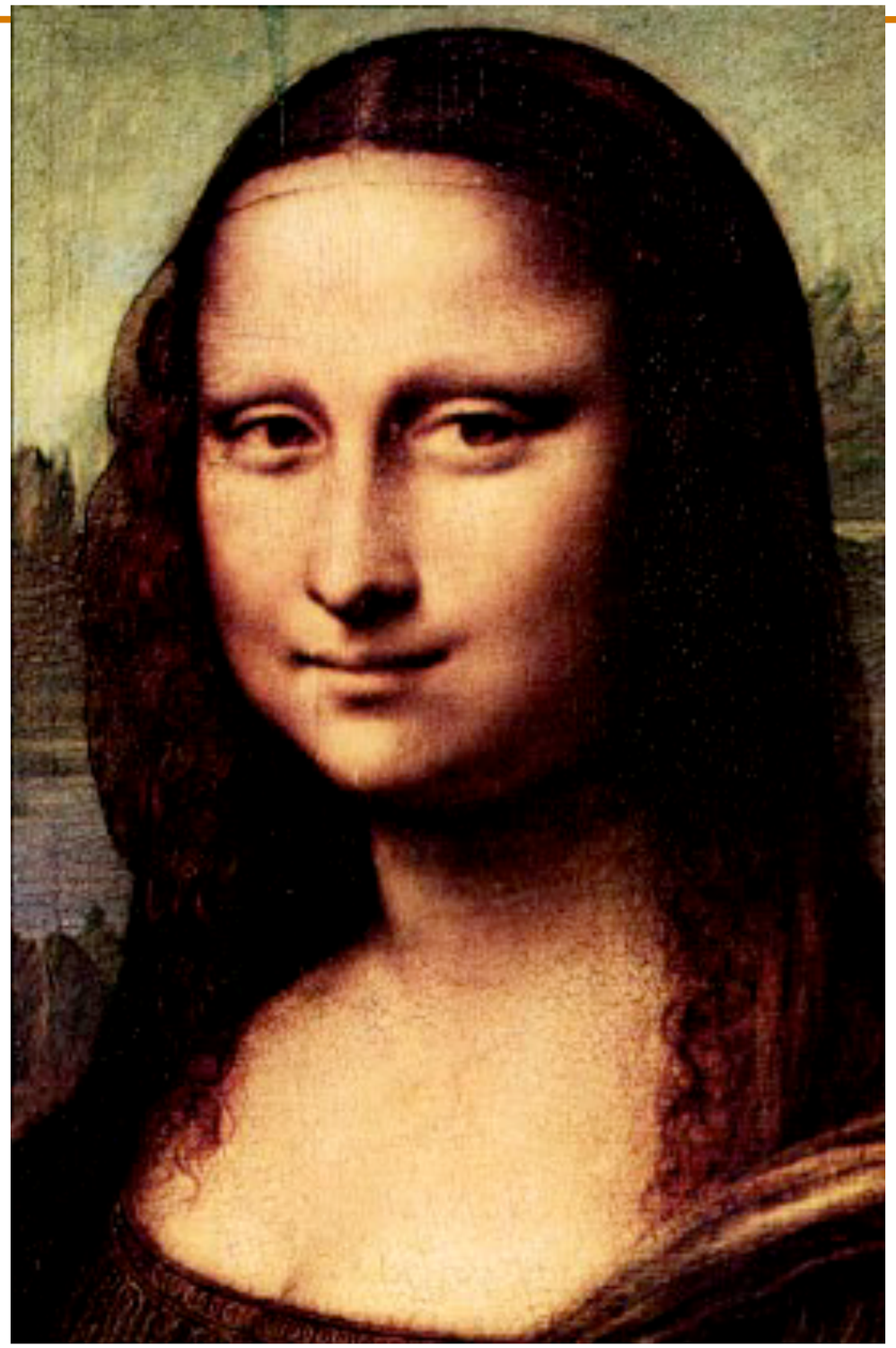


Figure 9: Mona Lisa View Morph. Morphed view (center) is halfway between original image (left) and it's reflection (right).



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**

- **Dealing with parallax**
- **http://research.microsoft.com/en-us/UM/people/yasumat/papers/cvpr2010_Takamatsu.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/

- http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=4EF222075D195D4927EAB26FC444C025?doi=10.1.1.182.8540&rep=rep1&type=pdf

Projective Geometry Considered Harmful

Berthold K.P. Horn

Copright © 1999