Spherical Warping

Image warping

we did this using homographies

We can view homographies as <u>linear</u> transformations if we consider that are planes are in 3D

project one image onto another images' view plane

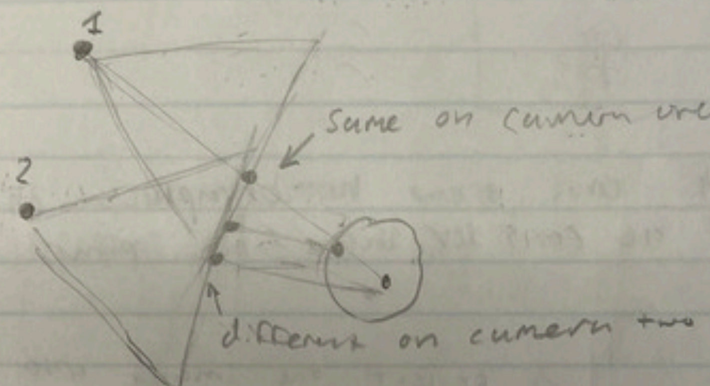
However, we <u>cannot</u> handle cases like this:

So, lets project onto a sphere

Nice and separate!

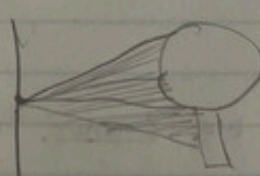
Going from (x, y) to (ϕ, θ) space.

Question: These methods hinge on a single viewpoint. What if we have multiple?



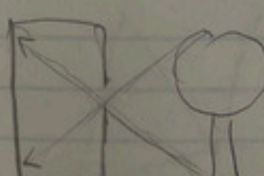
This is bad for panorama stitching, but great for depth estimation!

576 cam Mk II:



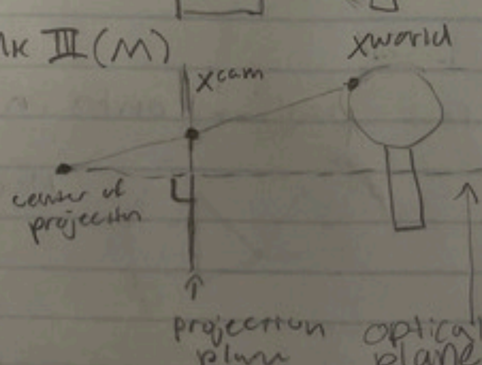
Too many locations contribute to one point on the sensor

576 cam Mk II:



Good! Realistic

576 cam Mk III (M)

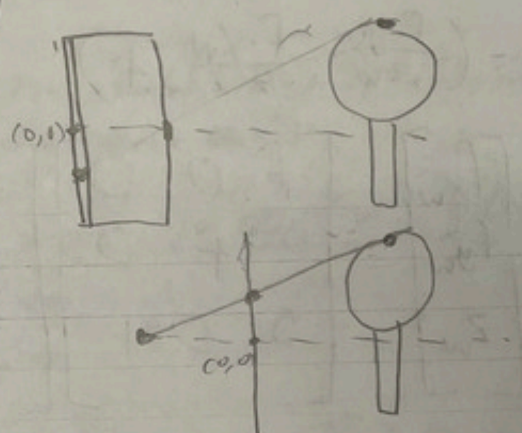


Mathematical, useful to us...

How do we get X^{img} from X^{world}?

$$X^{img} = \underbrace{\left[\text{Intrinsics}\right]}_{\text{focal length, pixel coord. system}} \underbrace{\left[\text{Projection}\right]}_{\text{3D to 2D}} \overbrace{\underbrace{\left[\text{Extrinsics}\right]}_{\text{camera pose (in world)}} X^{world}}^{X^{cam}}$$

Activity: HM Problem 1



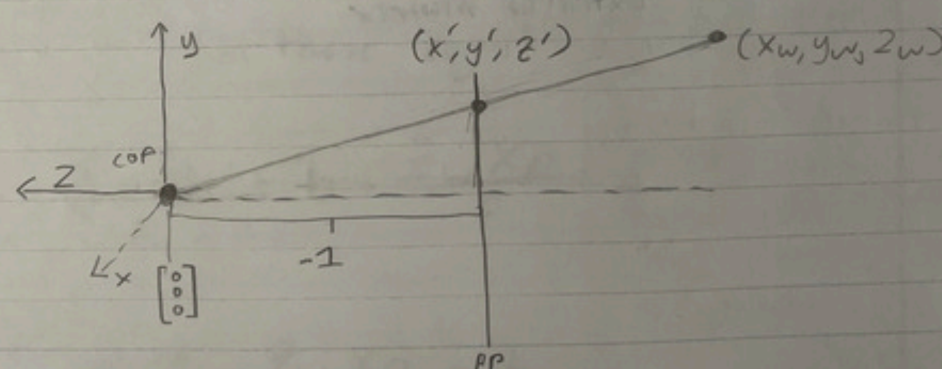
y is flipped

$$(x^{MkIII}, y^{MkIII}) = (-x^{MkI}, -y^{MkI})$$

HM Problem 2

COP
center of projection

PP
projection plane



Find (x', y', z')!

Homogeneous looking!

$$(x', y', z') = \left(\frac{x_w}{-z_w}, \frac{y_w}{-z_w}, -1\right)$$

Lets express this using linear algebra

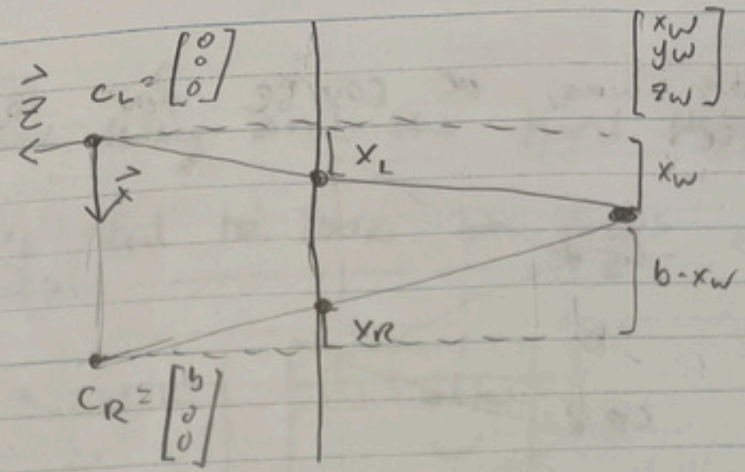
$$\begin{bmatrix} \frac{x_w}{z_w} \\ \frac{y_w}{z_w} \\ 1 \end{bmatrix} \sim \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$$

True! But not at all useful... Boring!

$$(x', y', z') = \left(-\frac{f \cdot y_w}{z_w}, -\frac{f \cdot x_w}{z_w}, f\right)$$

$$\begin{bmatrix} \frac{fx_w}{z_w} \\ \frac{fy_w}{z_w} \\ 1 \end{bmatrix} \sim \begin{bmatrix} fx_w \\ fy_w \\ z_w \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$$

Better! Version 0.1 of our projection is
Extrinsics matrix



Assume (for now) x_W is known (we don't, but things will work out)

$$\frac{z_W}{f} = \frac{x_W}{x_L}, \qquad \frac{b - x_W}{x_R} = \frac{z_W}{f}$$

$$x_W = \frac{z_W x_L}{f} \qquad x_W = b - \frac{z_W x_R}{f}$$

Set these equal! Goodbye x_W!

$$\frac{z_W x_L}{f} = b + \frac{z_W x_R}{f}$$

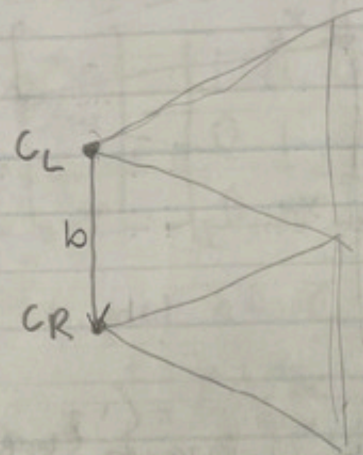
Aside!
Depth is inversely proportional to the disparity.

$$\frac{z_W x_L}{f} - \frac{z_W x_R}{f} = b$$

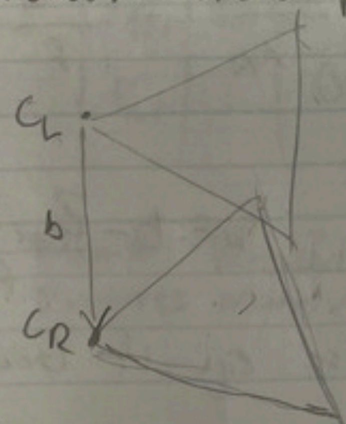
$$z_W \left(\frac{x_L - x_R}{f} \right) = b$$

$$z_W = \frac{fb}{x_L - x_R}$$

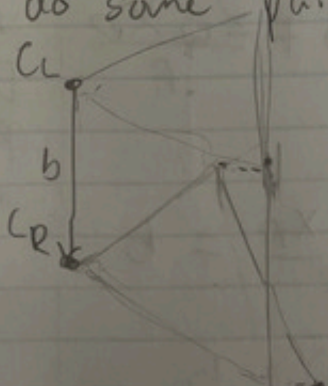
This was, of course for the simple case:



But what if we have cameras with different view planes?



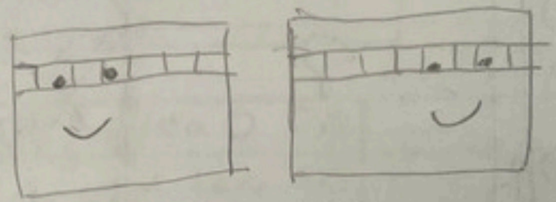
We <u>could</u> do some painful 3D math, or...



We can project C_R's view plane onto C_L:

The nice thing and the hard thing:

We only need to look at rows to find matches:



Less work (good!)

But more chances to make mistakes (not good!)

So we need some ability to find good matches...

The correspondence problem:

We need a way to assign a cost to some matching between two rows!

SSD - Sum. Squared Difference
SAD - Sum. Abs. Difference
CC - cross correlation...

If we have some pattern we want to find, if we use it as a filter in cross correlation it should create "bright spots" (large numbers) in areas that are similar to it

However, we want these detections to be consistent and comparable from one another... so we normalized

Camera matrix

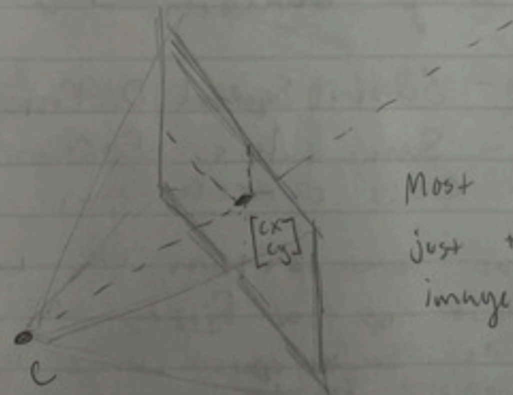
From last time!

$$\underbrace{\begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{Intrinsic},\ 3\times 4} \underbrace{\begin{bmatrix} & & \\ & ? & \\ & & \end{bmatrix}}_{\text{Extrinsics},\ 4\times 4} \vec{X}_{world} \leftarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

X_{cam}

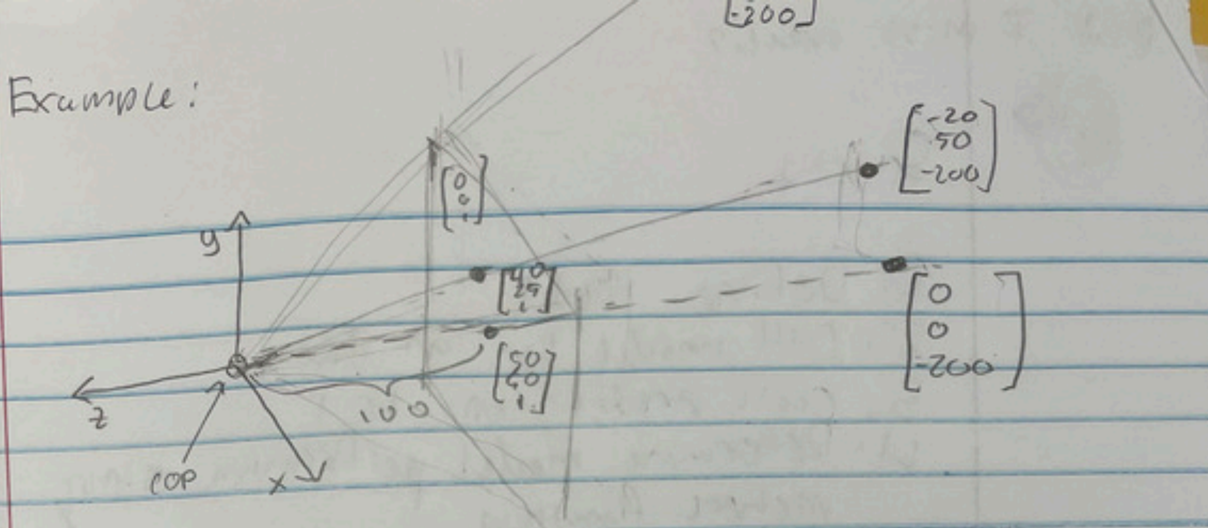
Why the extra column? 3D Homogeneous coordinates!

$\begin{bmatrix} c_x \\ c_y \end{bmatrix}$ The principal point: The location of the optical axis in pixel coordinates.



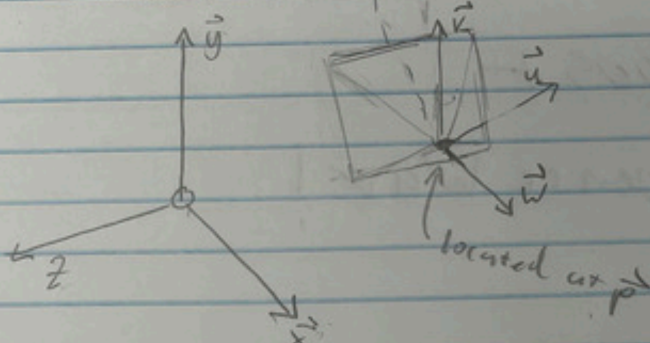
Most of the time, this is just the center of the image $\left(\frac{W}{2}, \frac{H}{2}\right)$

Example:



Intrinsic: $\begin{bmatrix} -100 & 0 & 50 \\ 0 & 100 & 50 \\ 0 & 0 & 1 \end{bmatrix}$

? Extrinsics: Our camera isn't nicely placed:



We want to be able to work relative to the camera... so we want M s.t.

$$\begin{bmatrix} & & \\ & M & \\ & & \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

$$\begin{bmatrix} | & | & | & | \\ \vec{u} & \vec{v} & \vec{w} & \vec{p} \\ | & | & | & | \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} \approx \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \text{So, } M = \begin{bmatrix} | & | & | & | \\ \vec{u} & \vec{v} & \vec{w} & \vec{p} \\ | & | & | & | \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}$$