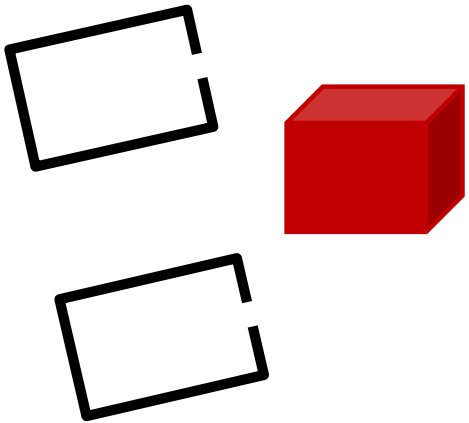
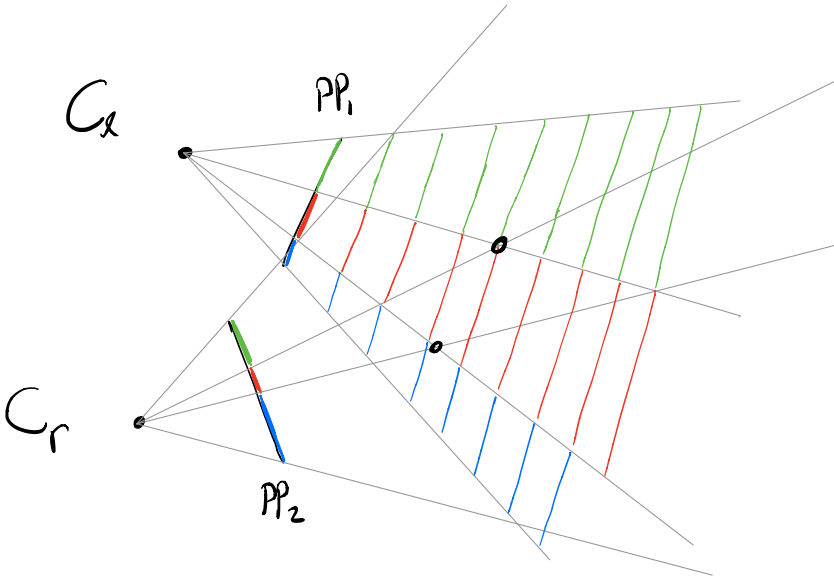
CSCI 497P/597P: Computer Vision

Scott Wehrwein

Stereo Rectification
Planesweep Stereo





Announcements

- Reminder: Exam out Friday, due Monday night.

P1 Artifacts: Hybrid

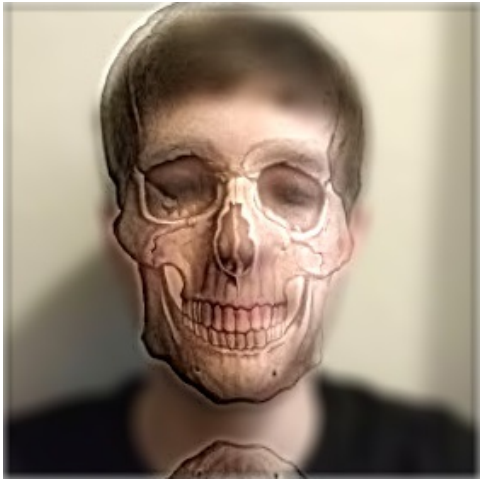
- 5-way tie (2 votes each)

Garrett Claeys

Jake Friberg

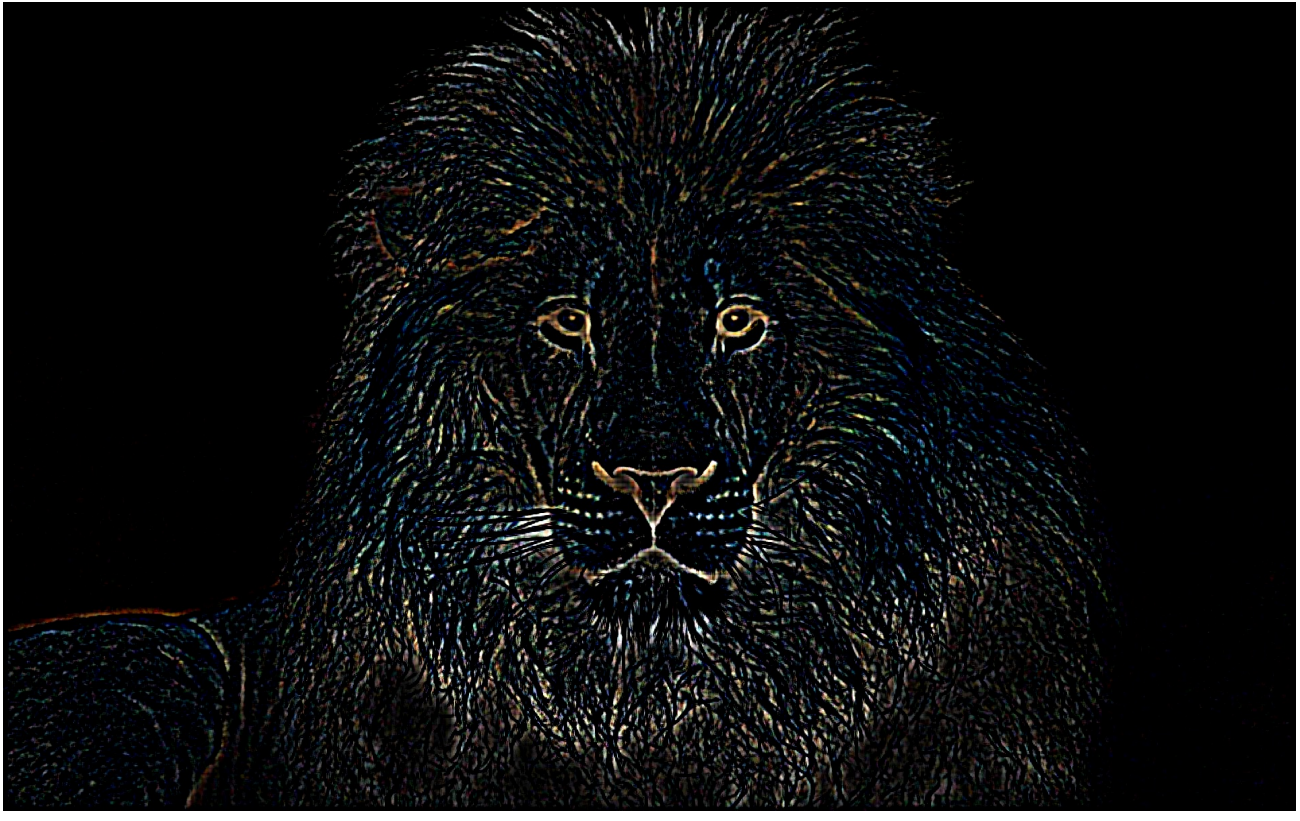
Brady Geleynse

Sean McCulloch



Melissa Swift

P1 Artifacts: Laplacian Forest Sweeney



Scott’s low-effort hybrid artifact

Goals

- Understand how to rectify a pair of stereo images given their intrinsics and extrinsics.

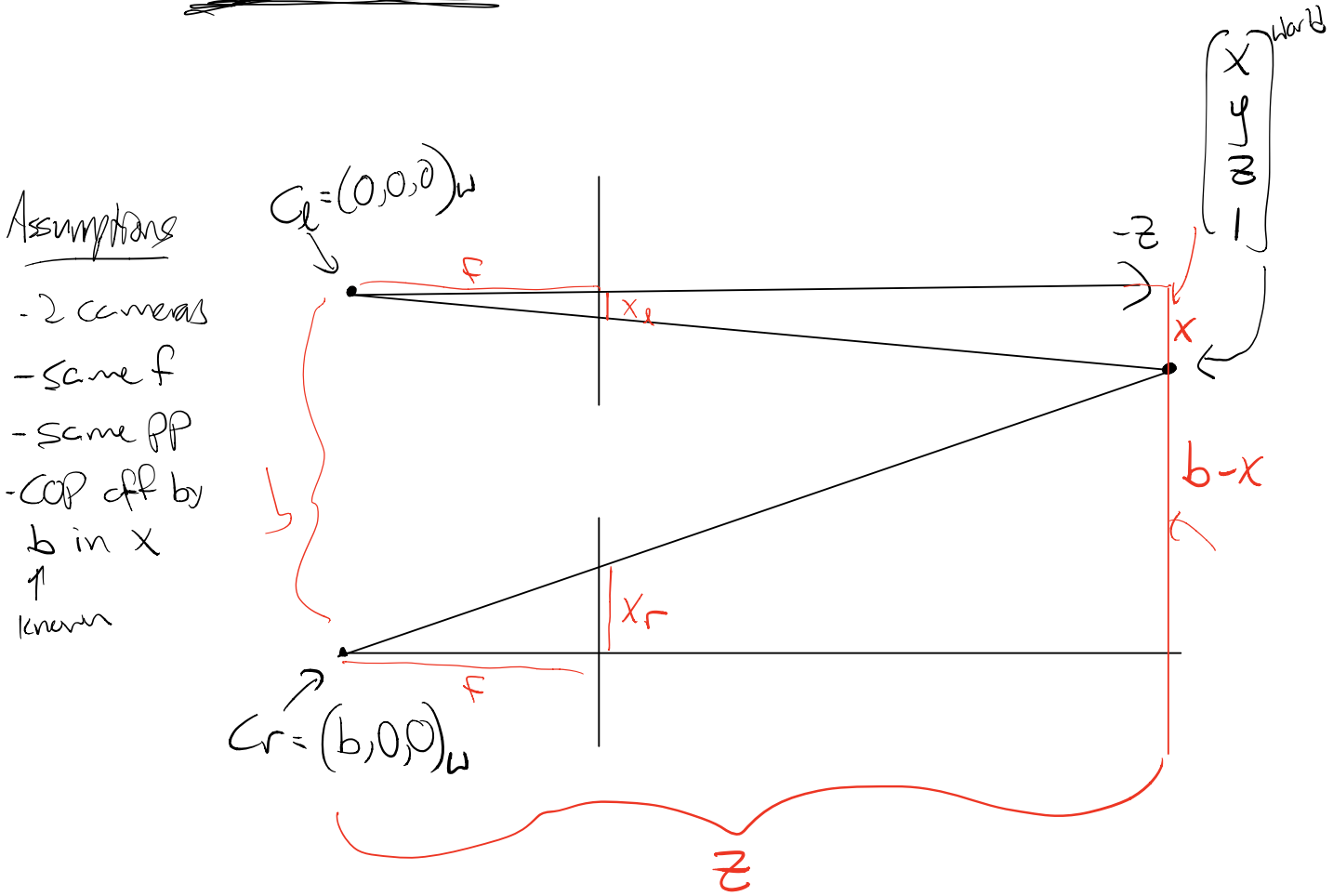
- Understand the plane sweep stereo algorithm.

A Stereo Algorithm

1. For every pixel (x, y)
    1. For every disparity d
        1. Get patch from image 1 at (x, y)
        2. Get patch from image 2 at (x + d, y)
        3. Compute cost using your metric of choice

```
C = np.array(h,w,d)
for r in range(0,h):
  for c in range(0,w):
    for d in range(-maxd, maxd):
      C[r,c,d] = metric(get_patch(im1,r,c), get_patch(im2,r,c+d)

disp = np.max(C, axis=2)
depth = f * b / disp
```

Rectified Stereo Cameras



What if the cameras aren't rectified?

- Assume cameras are **calibrated,** i.e., we know:

$K_l, K_r, \quad R_l, R_r, \quad t_l \; t_r$

$\uparrow \; \uparrow$

$C_l \quad C_r$

Projective Geometry: Homogeneous Points

Which of the following 3-vectors does not represent the same projective point as the others?

A. [12, 8, 4] (3 2 1)

B. [8, 6, 3] (8/3, 2, 1)

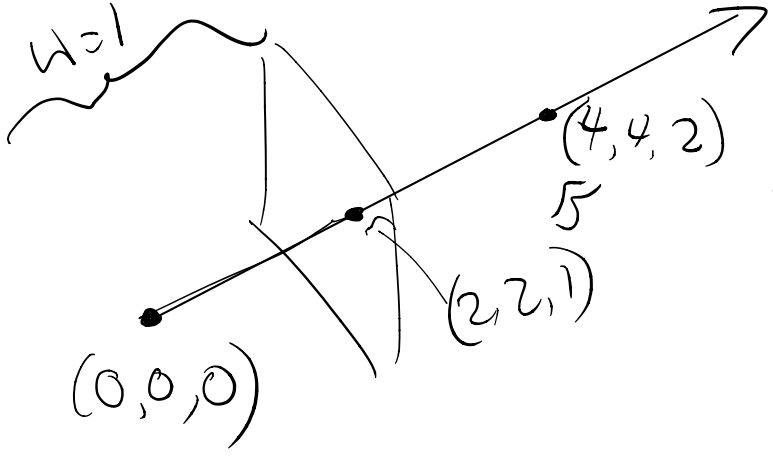
C. [24, 16, 8] (3 2 1)

D. [15, 10, 5] (3 2 1)

socrative.com

CSCI 497P

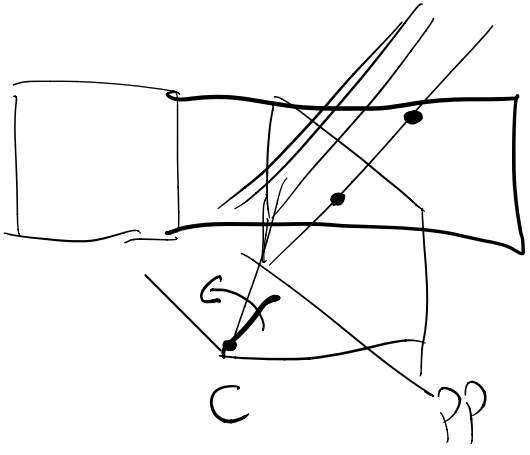
Geometric Interpretation



$\mathbb{RP}^3$ visualization of $\mathbb{P}^2$

Homogeneous coords live in $\mathbb{P}^2$

2D Projective Space



3×3 3×3 3×3

$$p_1' = \left(K R K^{-1}\right) p_1$$

project back

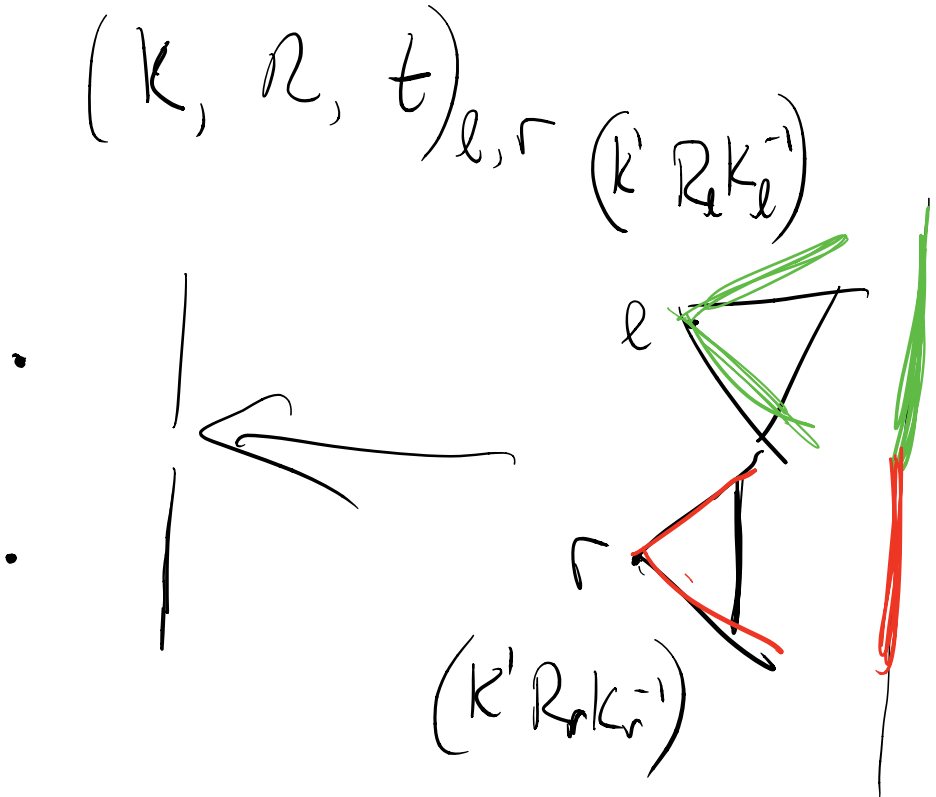
rotate

unproject into camera

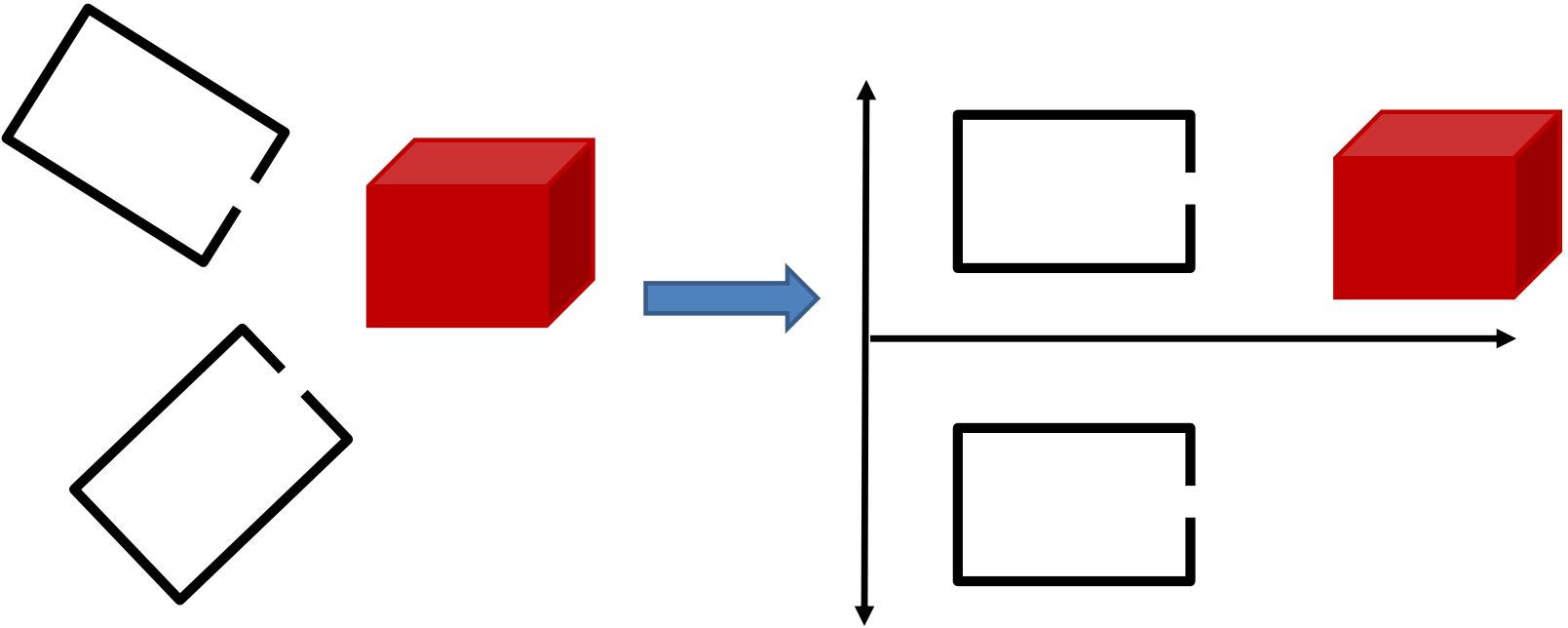
H

What if the cameras aren't rectified?

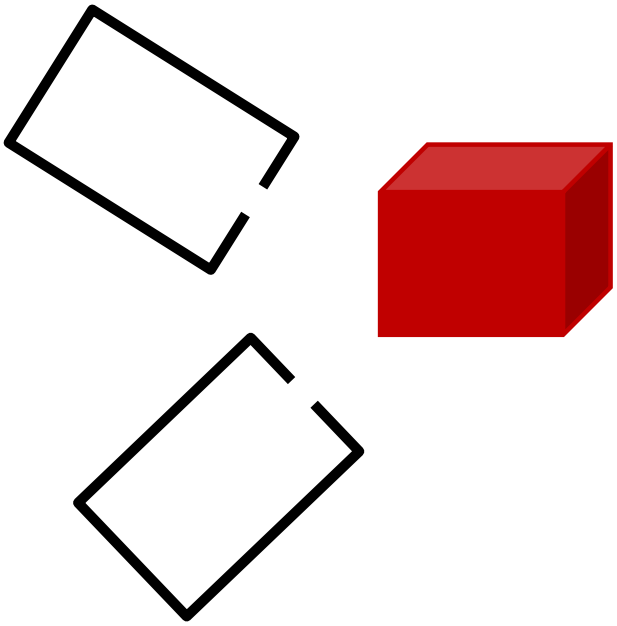
- Assume cameras are **calibrated**, i.e., we know:



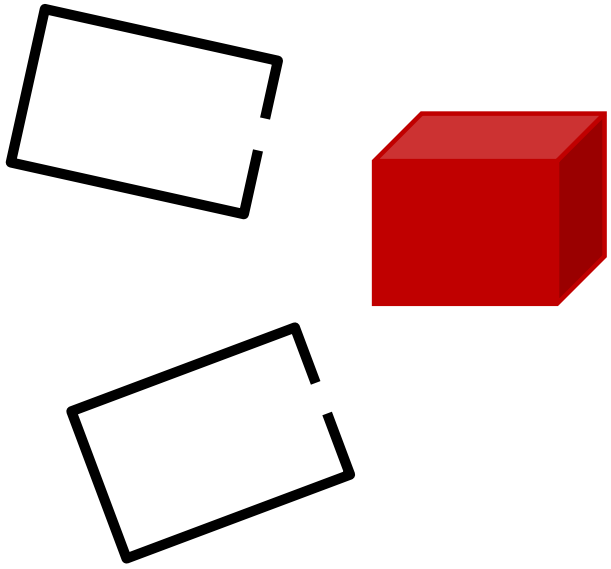
Rectifying cameras



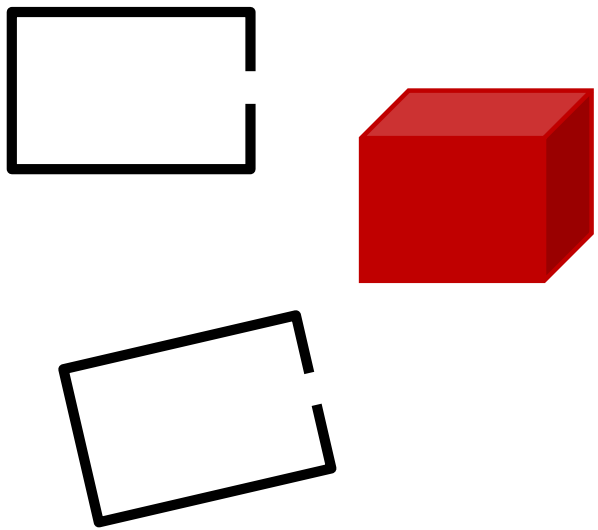
Rectifying cameras



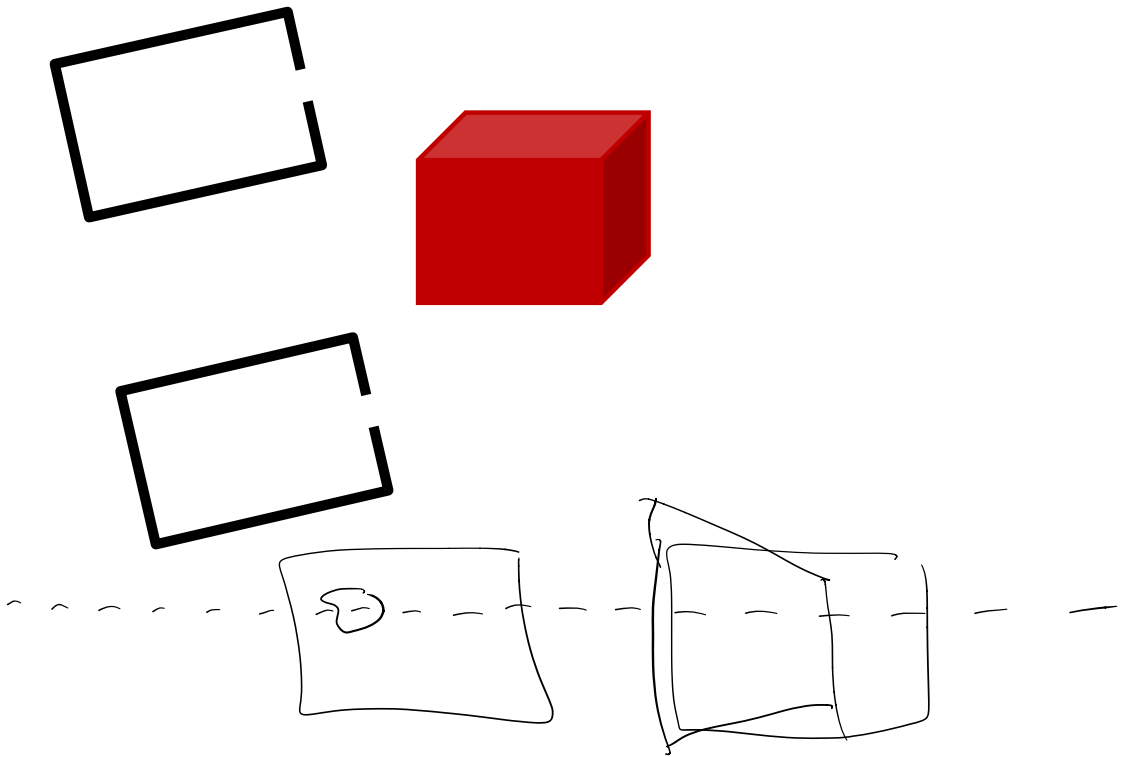
Rectifying cameras



Rectifying cameras



Rectifying cameras



Plane Sweep Stereo

A Stereo Algorithm

1. For every pixel (x, y)
    1. For every disparity d
        1. Get patch from image 1 at (x, y)
        2. Get patch from image 2 at (x + d, y)
        3. Compute cost using your metric of choice

```
C = np.array(h,w,d)
for r in range(0,h):
  for c in range(0,w):
    for d in range(-maxd, maxd):
      C[r,c,d] = metric(get_patch(im1,r,c), get_patch(im2,r,c+d)

disp = np.max(C, axis=2)
depth = f * b / disp
```

Plane Sweep Stereo Algorithm

1. For every disparity d
    1. For every pixel (x, y)
        1. Get patch from image 1 at (x, y)
        2. Get patch from image 2 at (x + d, y)
        3. Compute cost using your metric of choice

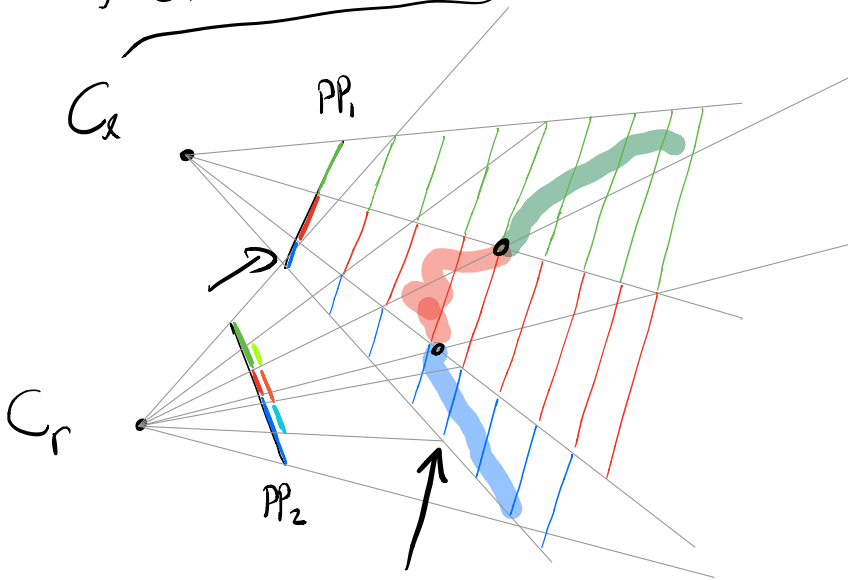
```
C = np.array(h,w,d)
for d in range(-maxd, maxd):
    for r in range(0,h):
        for c in range(0,w):
            C[r,c,d] = metric(get_patch(im1,r,c), get_patch(im2,r,c+d)

disp = np.max(C, axis=2)
depth = f * b / disp
```

Rectified

C_l •

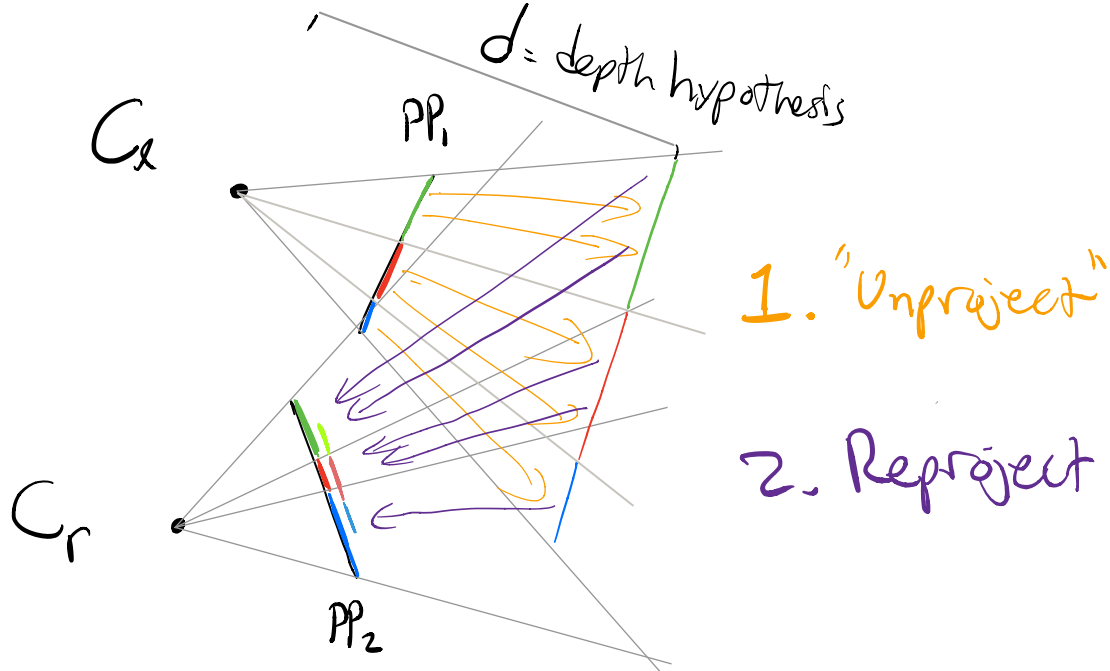
C_r •

Not Rectified



1. "Unproject":

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \leftarrow \text{add} = \text{world} \quad -t_l \quad \underbrace{R_l^T}_{\text{cam @ } l} \quad d \quad \underbrace{K_l^{-1}}_{\text{cam}} \quad \underbrace{\begin{bmatrix} x_l \\ y_l \\ 1 \end{bmatrix}}_{\text{im}}$$



1. "Unproject"

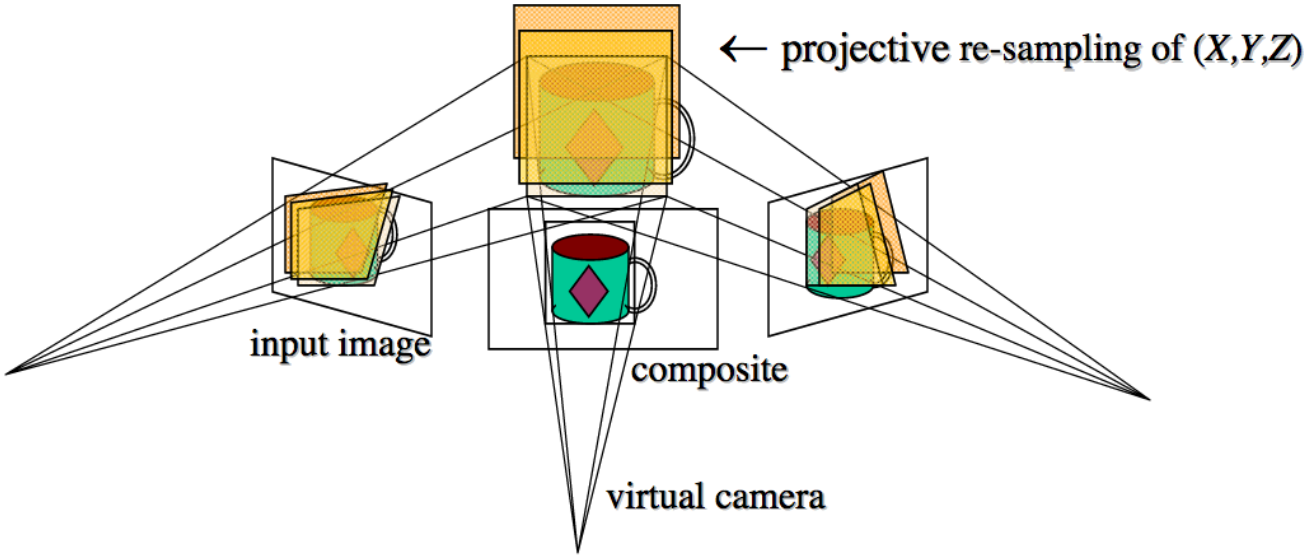
2. Reproject

2. "Reproject":

$$\begin{bmatrix} x_r \\ y_r \\ 1 \end{bmatrix} = K_r \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{bmatrix} R_r & | & t_r \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

- $\begin{bmatrix} x_r \\ y_r \\ 1 \end{bmatrix}$: normalized
- K_r: img_r (2D homog)
- $\begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix}$: cam_r (2D homog)
- $\begin{bmatrix} R_r & | & t_r \end{bmatrix}$: cam_r (3D homog)
- $\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$: World

Plane Sweep Stereo



– each plane defines an image ⇒ composite homography