

Lecture 14

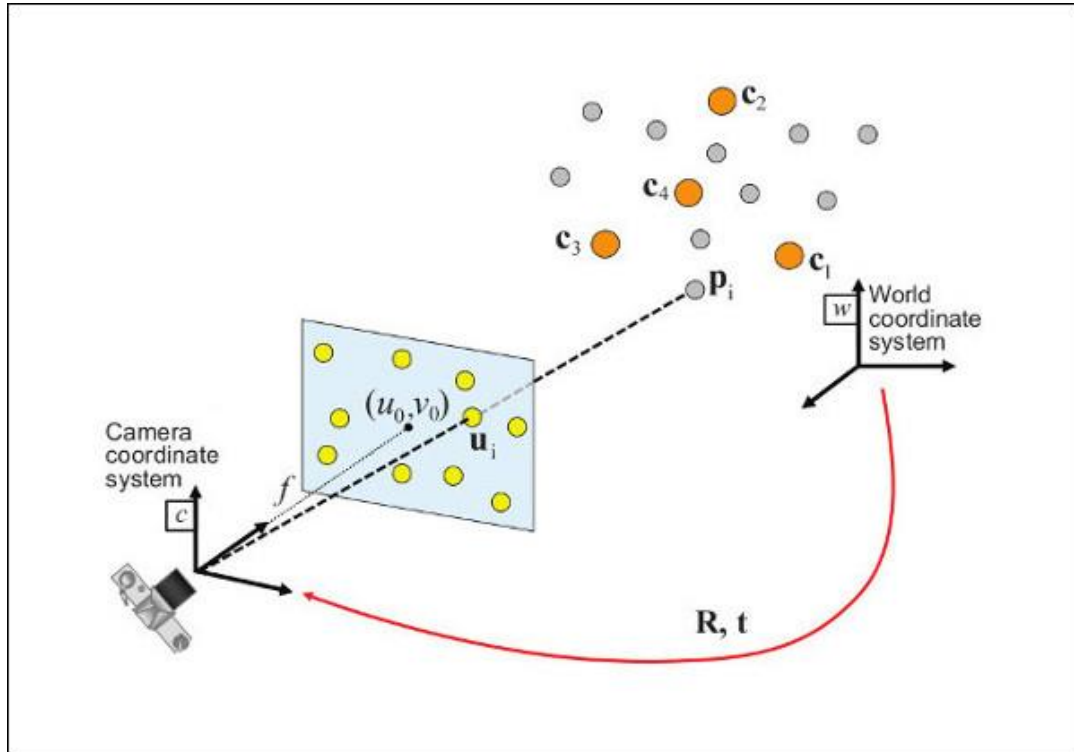
Depth and 3D Estimation

ECE 1390/2390



Image from
<https://my.clevelandclinic.org/>

Prospective-n-Point (PnP)



$$\begin{bmatrix} B_x \\ B_y \\ B_z \\ 1 \end{bmatrix} = \begin{bmatrix} r_{0,0} & r_{0,1} & r_{0,2} & \Delta X \\ r_{1,0} & r_{1,1} & r_{1,2} & \Delta Y \\ r_{2,0} & r_{2,1} & r_{2,2} & \Delta Z \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} A_x \\ A_y \\ A_z \\ 1 \end{bmatrix}$$

3D to 2D projection

$$\begin{bmatrix} u_{ideal} \\ v_{ideal} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} B_x \\ B_y \\ B_z \\ 1 \end{bmatrix}$$

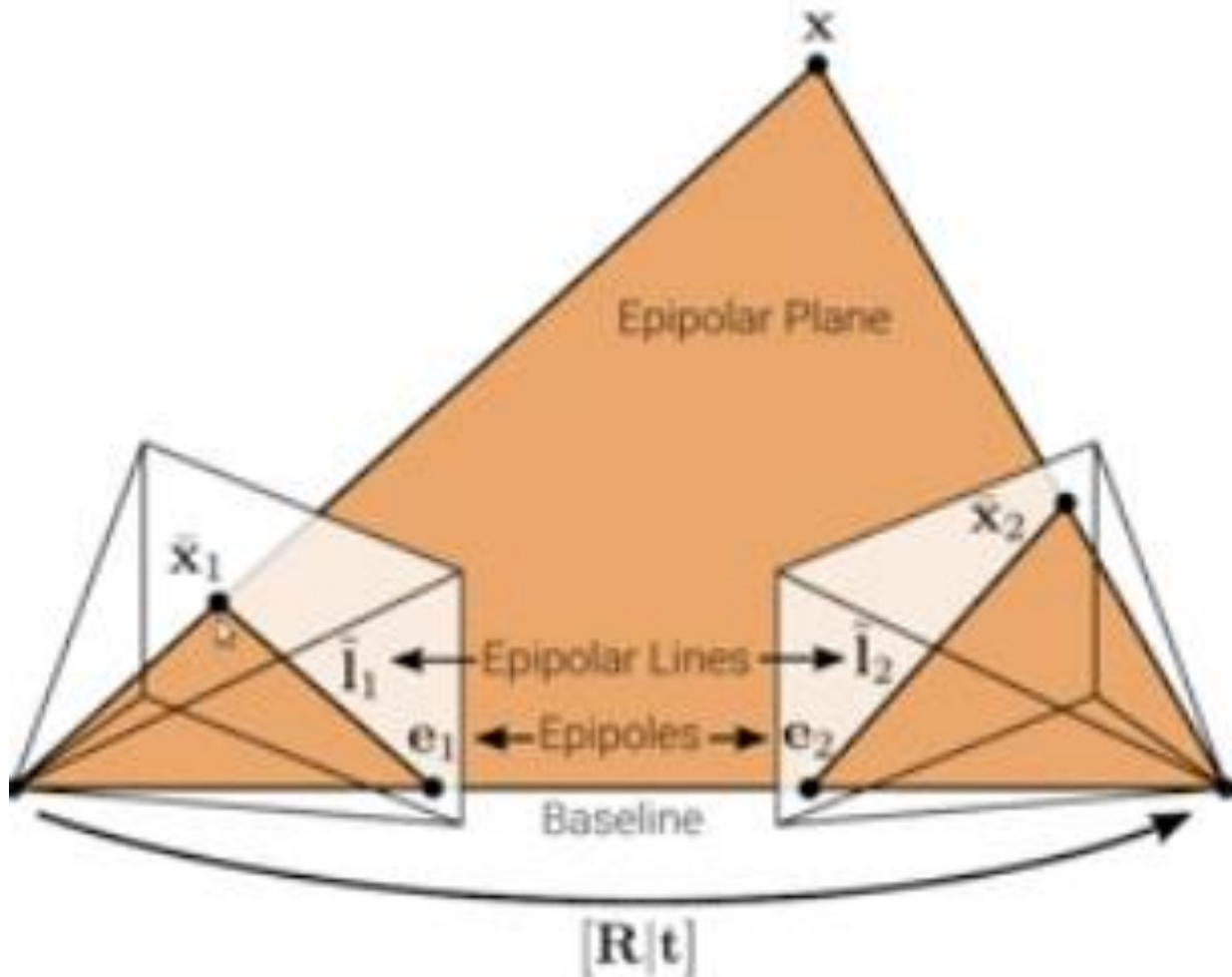
Intrinsic Camera distortion (K)

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{ideal} \\ v_{ideal} \\ 1 \end{bmatrix}$$

Epipolar coordinates

Epipolar plane is defined by the three points:

- Camera Origin 1
- Camera Origin 2
- 3D point of interest

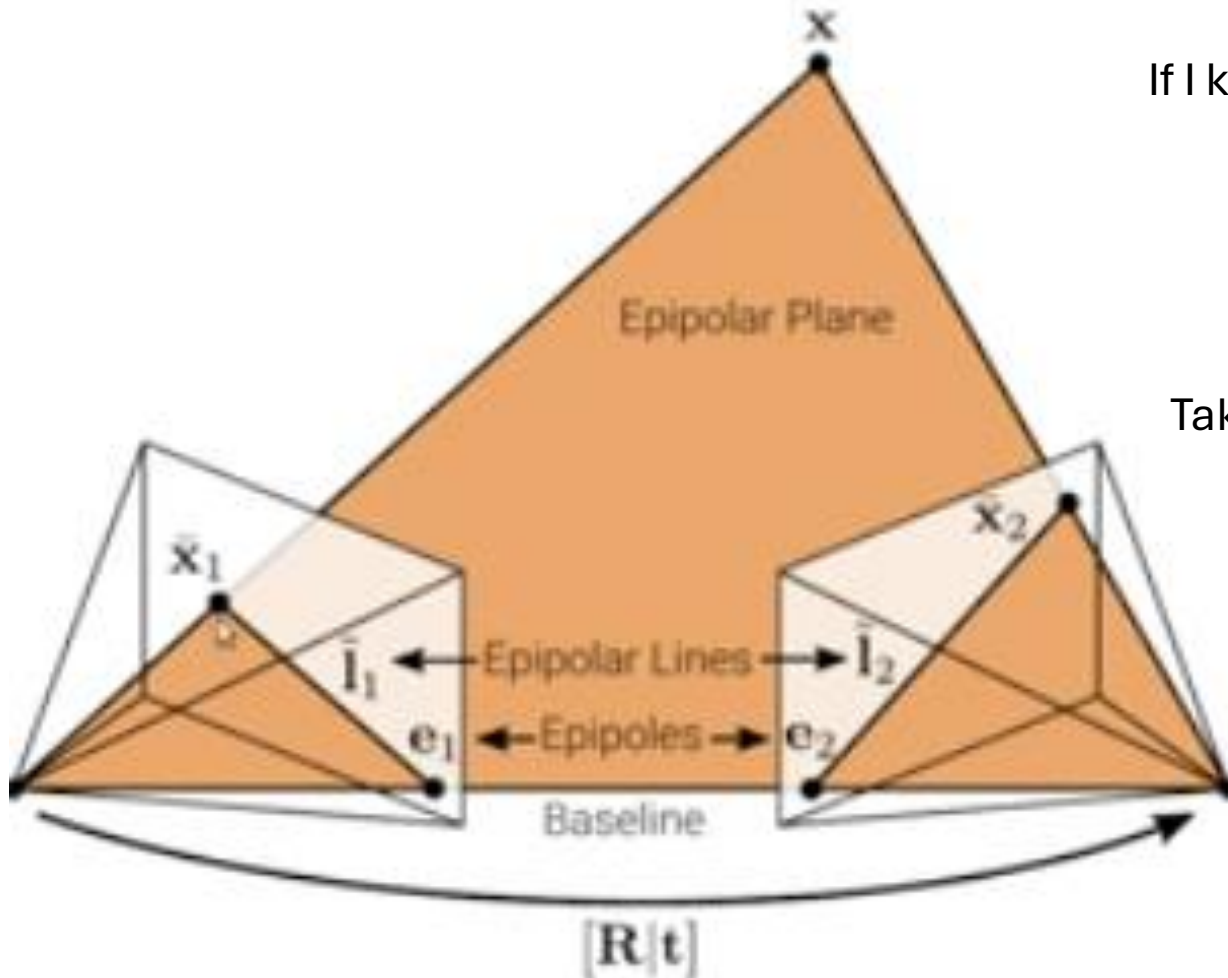


$$u_1 = K_1 * u_1'$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{ideal} \\ v_{ideal} \\ 1 \end{bmatrix}$$

$$K_1^{-1} u_1 = u_1'$$

Epipolar coordinates



$$K_1^{-1}u_1 = u_1'$$

If I know the R and T between the cameras:

$$u_2' = R * u_1' + st$$

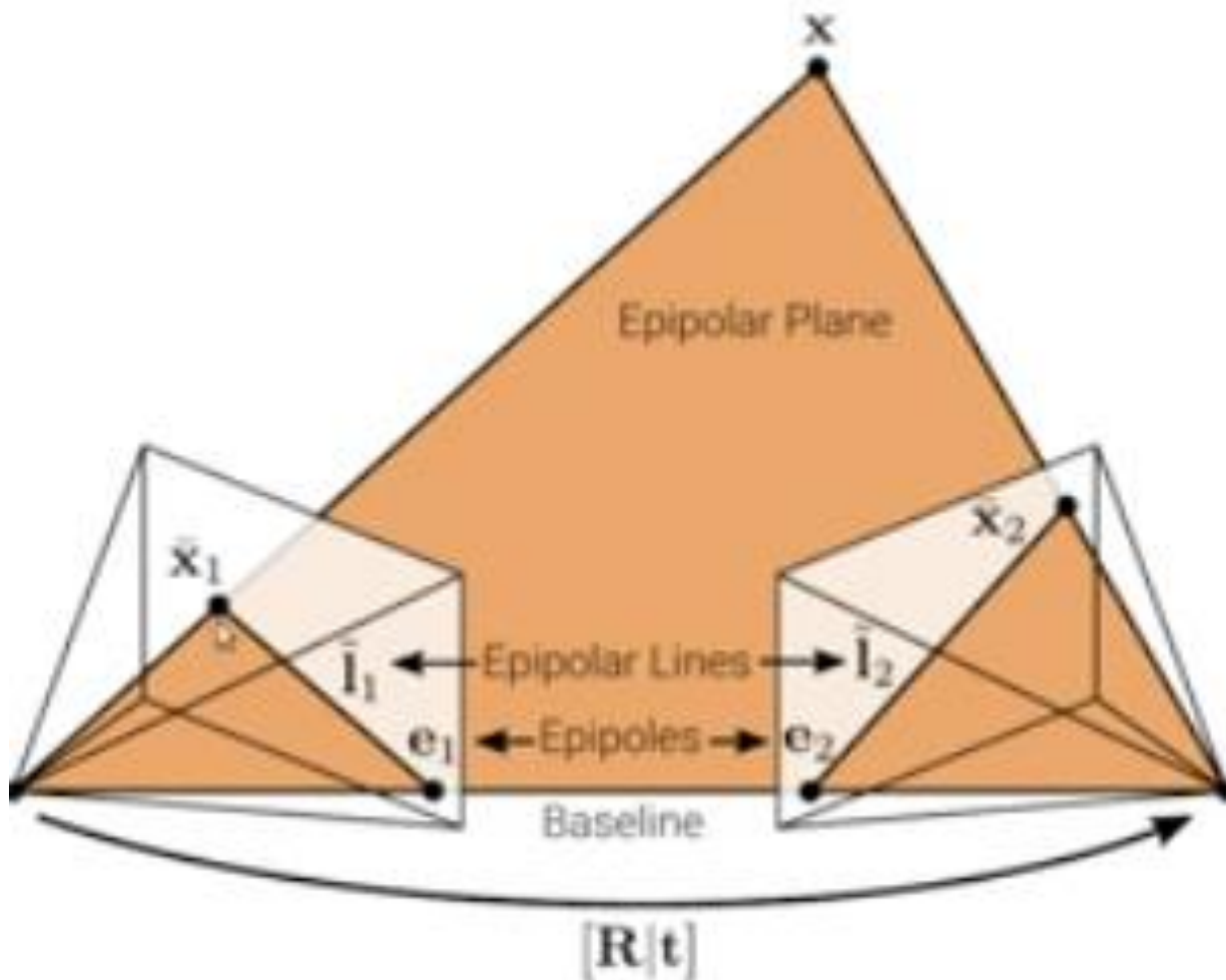
Take cross product with t on both sides

$$t \times u_2' = t \times R * u_1' + t \times st \quad 0$$

Take dot product with u_2^t on both sides

$$u_2'^T (t \times u_2') = u_2'^T (t \times R * u_1') \quad 0$$

Epipolar coordinates



$$0 = u_2'^T (t \times R * u_1')$$

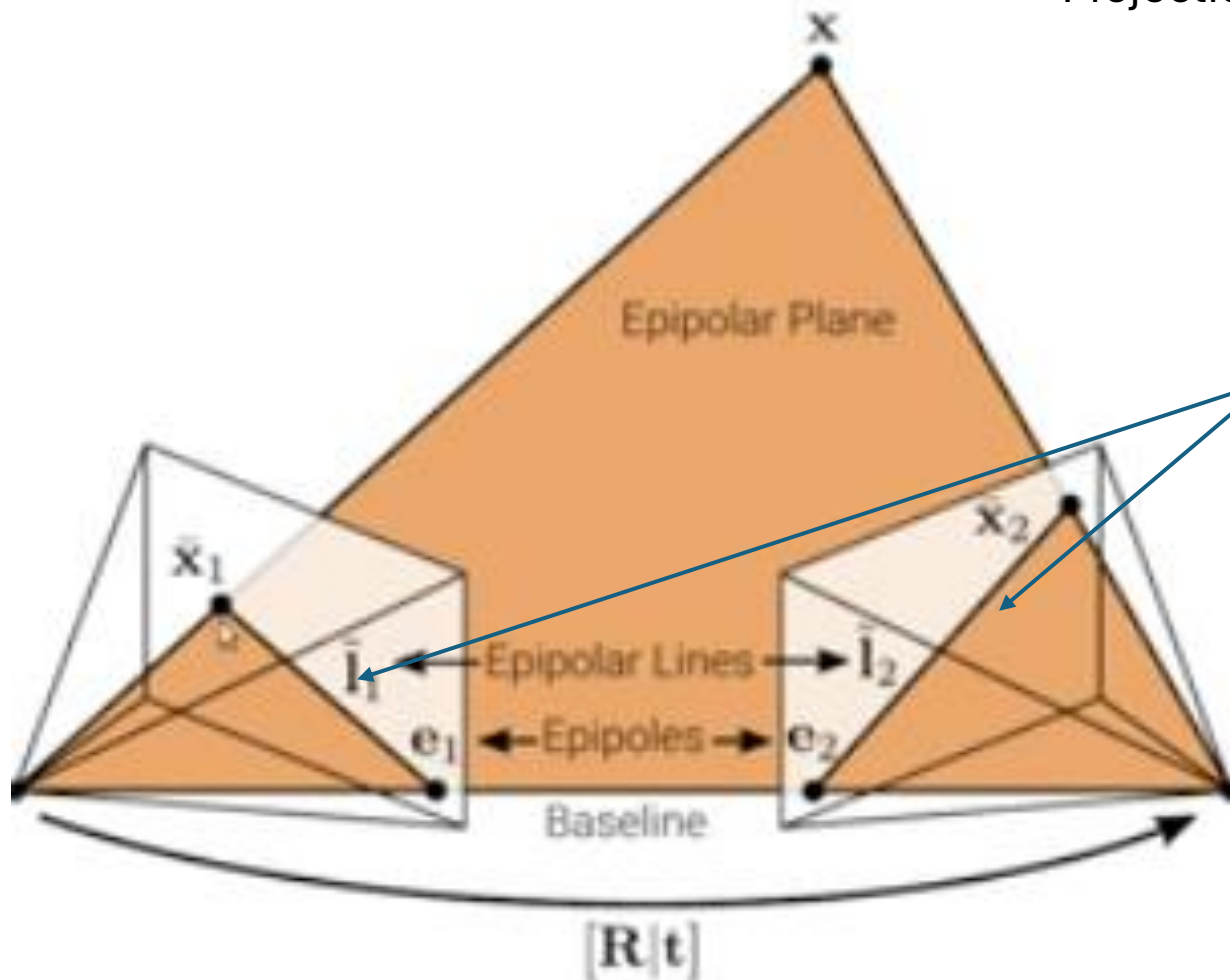
$$0 = u_2'^T * E * u_1'$$

$$E = t \times R$$

Essential matrix

Epipolar coordinates

Projection by the Essential matrix produces a (epipolar) line



$$I_2 = E * u_1'$$

$$I_1 = E^T * u_2'$$



Epipolar coordinates

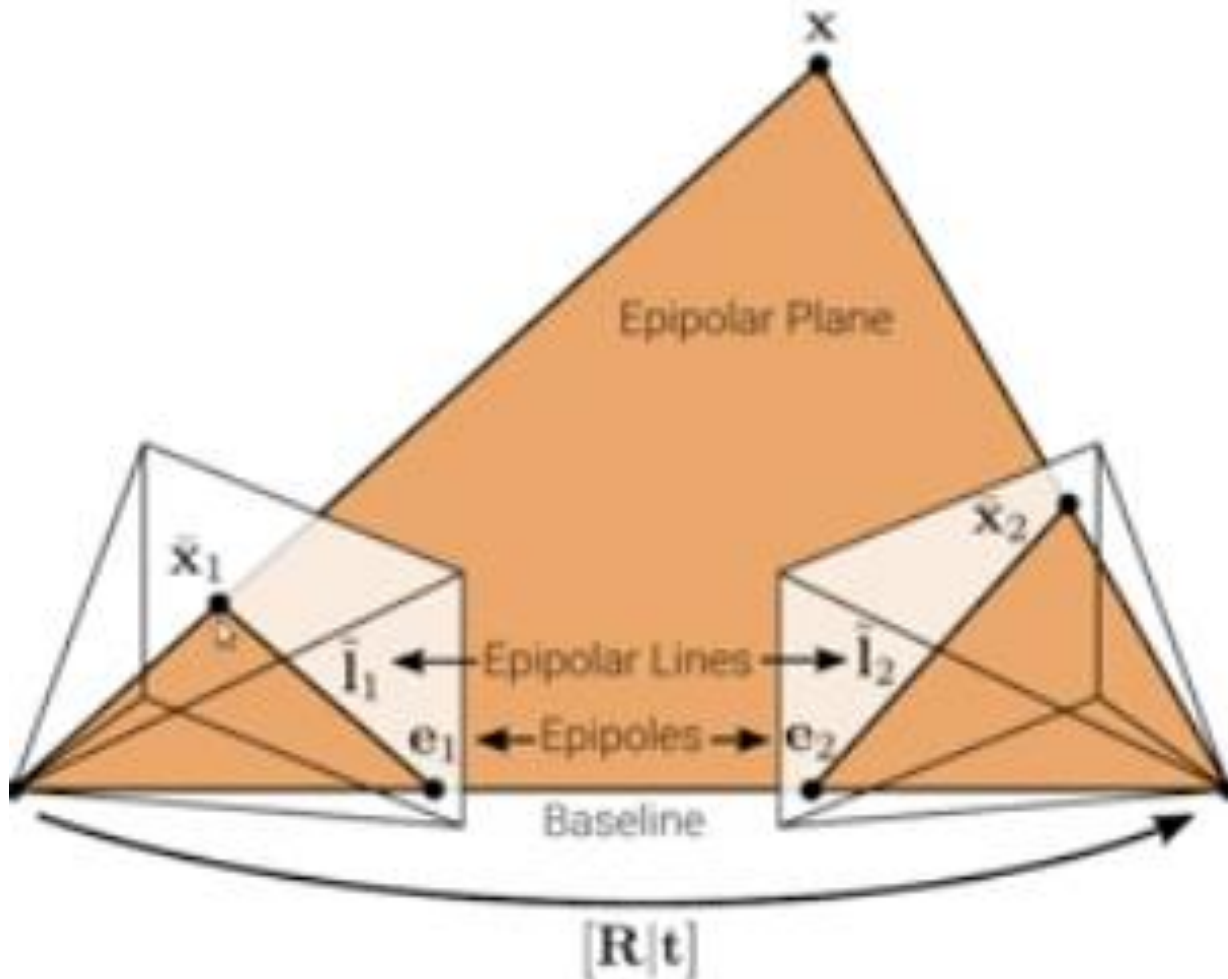
$$0 = u_2'^T * E * u_1'$$

Foil out matrix equation:

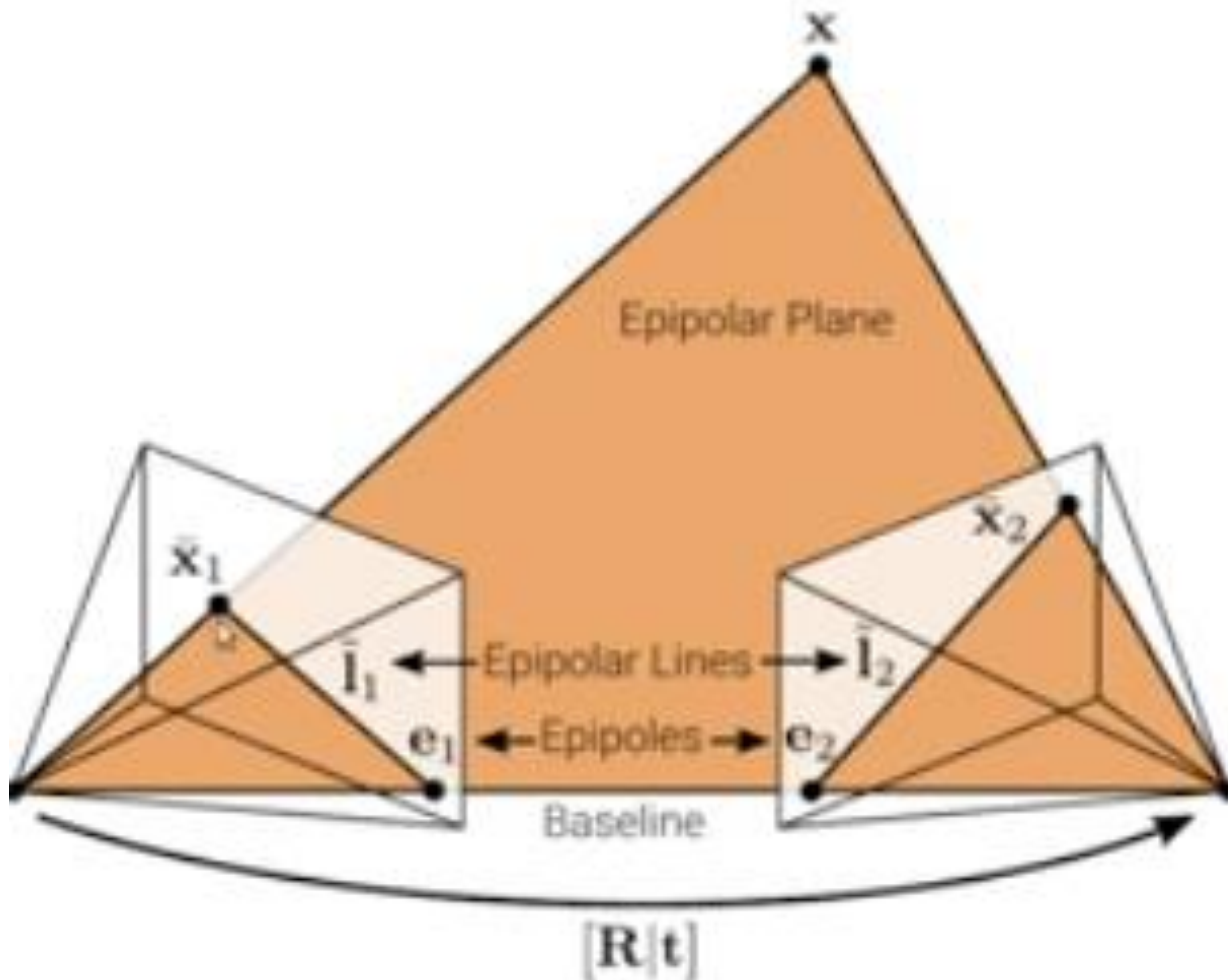
$$u_1 u_2 E_{1,1} + v_1 u_2 E_{1,2} + u_2 E_{1,3} + u_1 v_2 E_{2,1} + v_1 v_2 E_{2,2} + v_2 E_{2,3} + u_1 E_{3,1} + v_2 E_{3,2} + E_{3,3} = 0$$

8 unknowns (plus 1 scale)

Need 8+ corresponding points in the two images



Epipolar coordinates




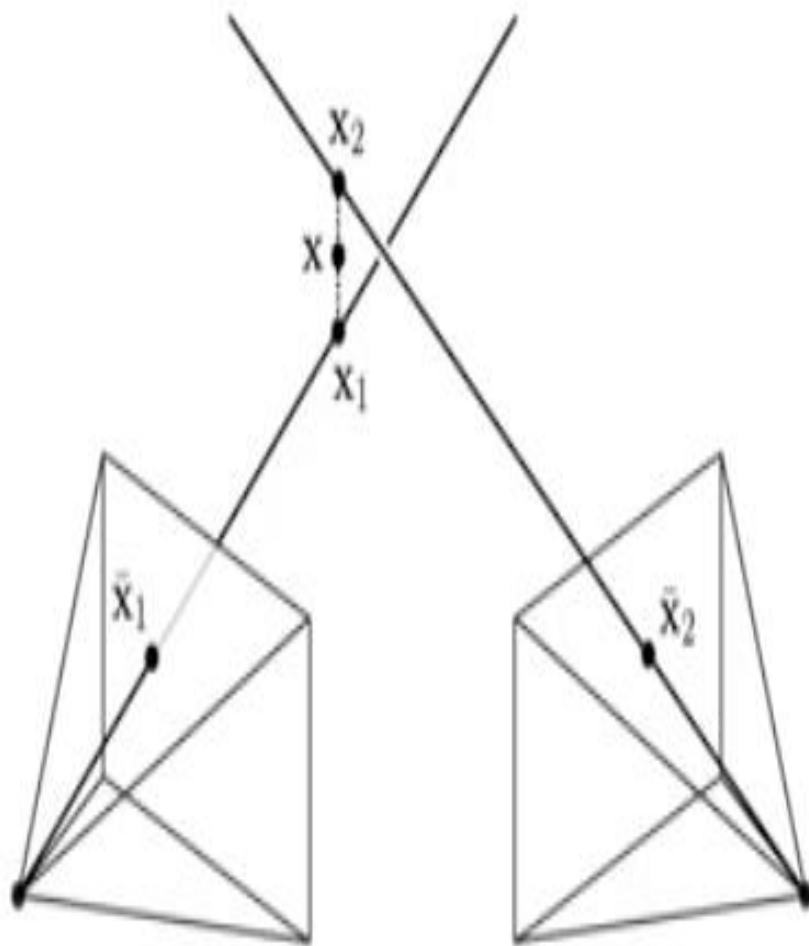
$$E = t \times R$$

$$t^T E = t^T (t \times R) = 0$$

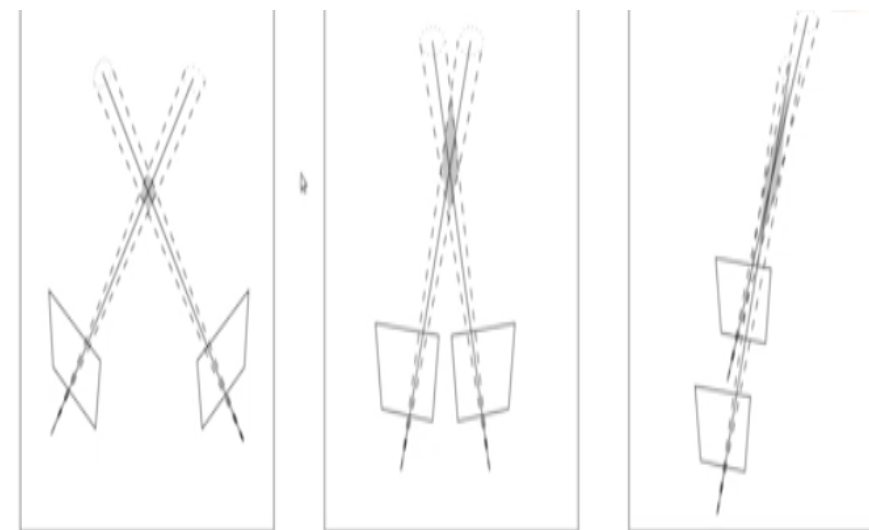
t is the null projector of the essential matrix

SVD(E) \rightarrow 2 nonzero and 1 zero eigenvalue

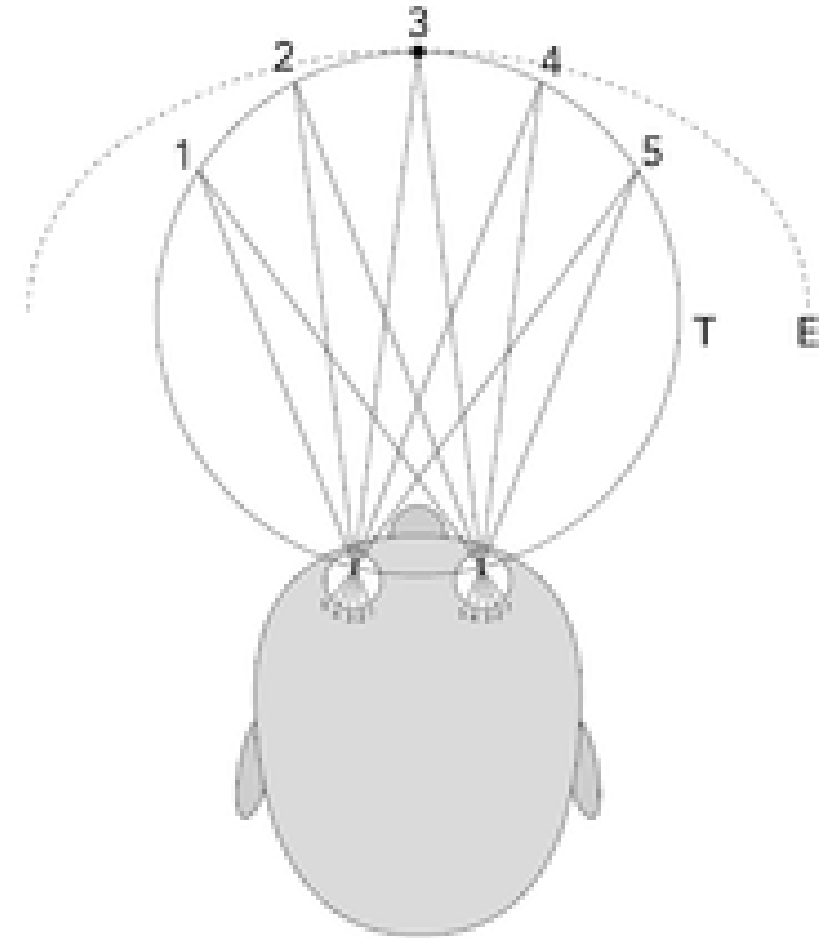
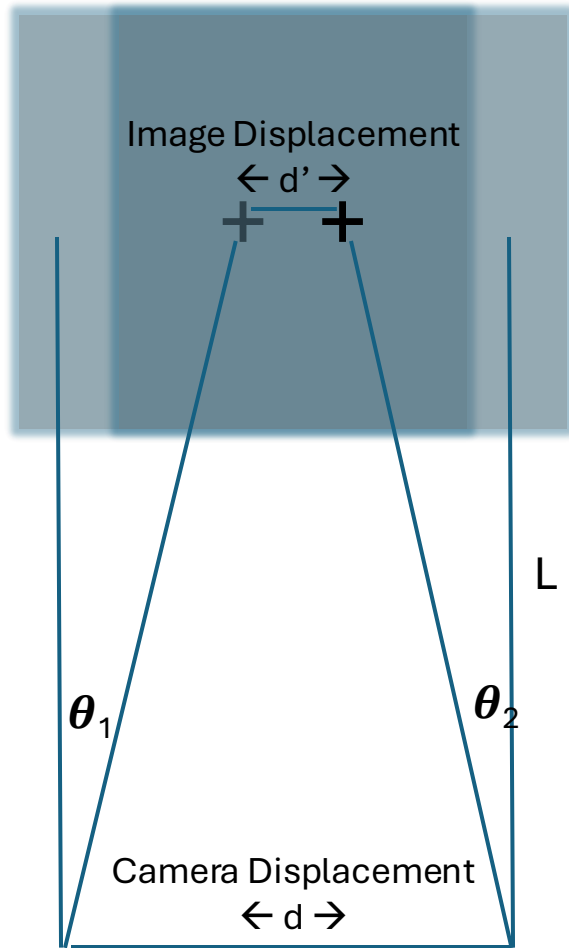
t 



Once you know the relationship between the two cameras, 3D points are easy to find as the intersection of projection rays



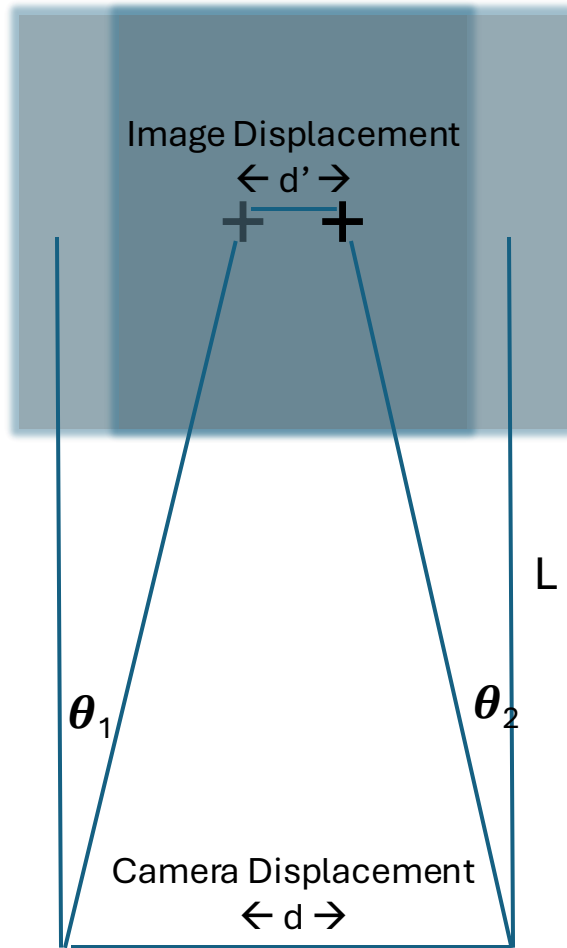
Stereo Vision



$$d - d' = L \cdot (\tan \theta_1 + \tan \theta_2)$$

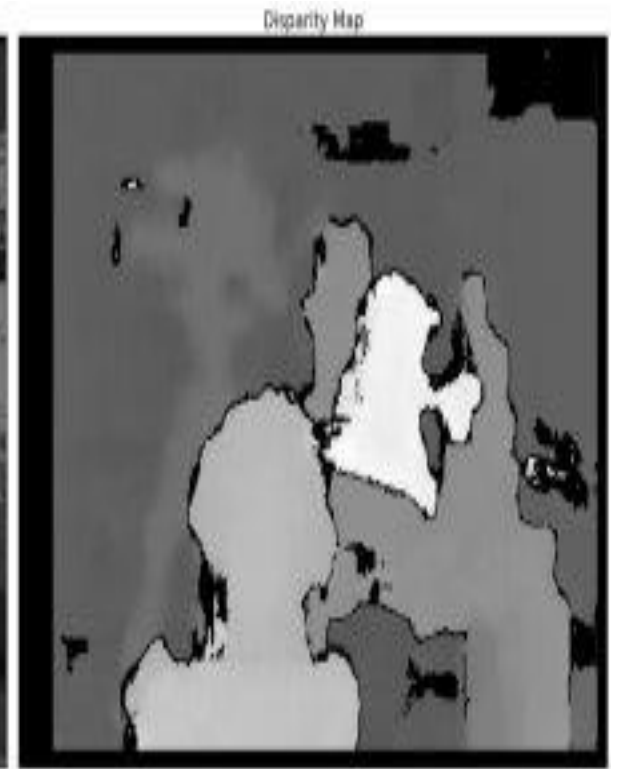
$$\frac{d - d'}{(\tan \theta_1 + \tan \theta_2)} = L$$

Stereo Vision



$$d - d' = L \cdot (\tan \theta_1 + \tan \theta_2)$$

$$\frac{d - d'}{(\tan \theta_1 + \tan \theta_2)} = L$$



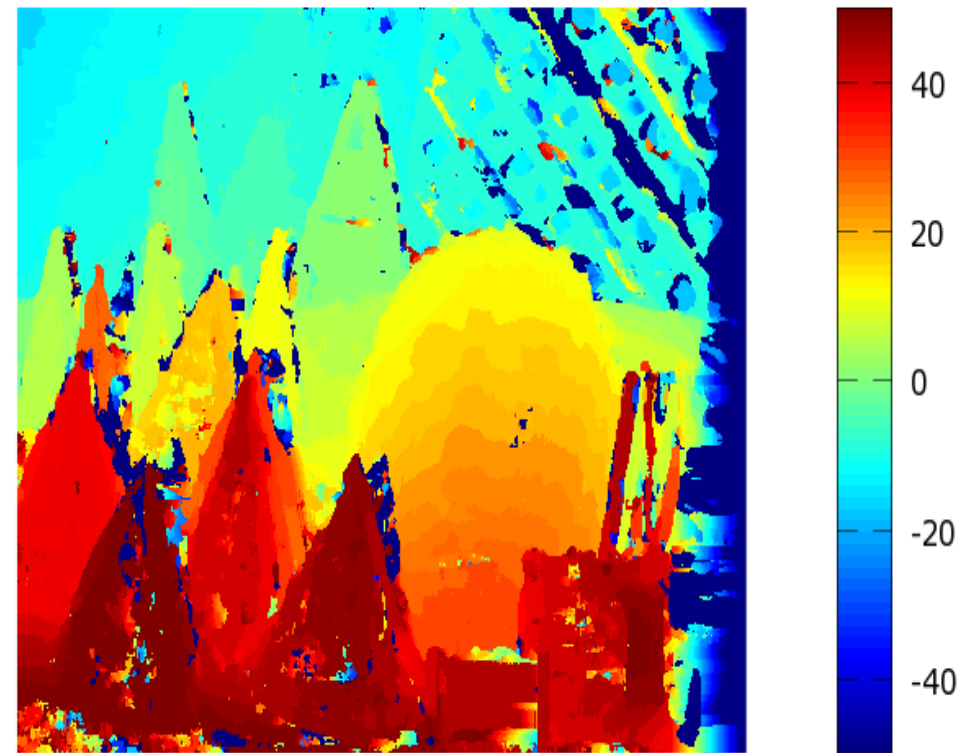
Stereo Vision

$$SAD[i,j] = \sum |B_1 - B_2[i]|$$





Depth map from basic block matching



- minDisparity: Minimum disparity value. Normally we expect 0 here but it's sometimes required when the rectification algorithm shifts the image.
- numDisparities: Max disparity value. Has to be greater than zero. Defines the disparity boundary.
- blockSize: Block size for the matching blocks. Recommended is [3–11] and an odd number is recommended since odd-sized blocks have a center.
- P1 & P2: Responsible for smoother images. The rule is $P2 > P1$.
 - P1 – penalty for neighboring pixels
 - P2 – penalty for further neighbors
- disp12MaxDiff: Maximum pixel difference (in pixels) for the disparity calculation.
- preFilterCap: A value to be used before the filtering. Before the block matching, a derivative of the image x-axis is calculated and used to check the boundary $[-\text{prefiltercap}, \text{prefiltercap}]$.
- uniquenessRatio: After the cost function calculation, this value used to compare. Suggested value range [5–15].
- speckleWindowSize: Filter to remove big values and get a smoother image. Suggested value range [50–200].
- speckleRange: Used to check disparity differences with the neighbors to get a smooth image.

Stereo Vision

Camera 1 (left)

f_x : focal length in x
 f_y : focal length in y

O_x : Origin of image in x
 O_y : Origin of image in y

$[U,V]$: Point in image

Camera 2 (right)

$f_{x'}$: focal length in x
 $f_{y'}$: focal length in y

$O_{x'}$: Origin of image in x
 $O_{y'}$: Origin of image in y

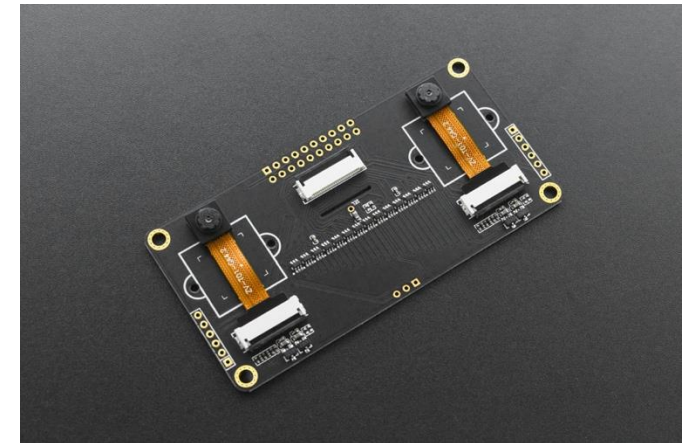
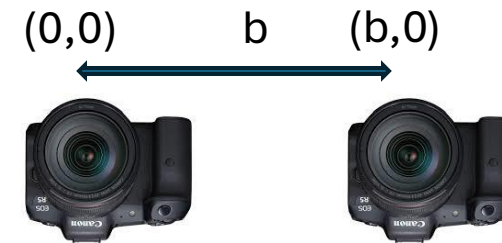
$[U',V']$: Point in image

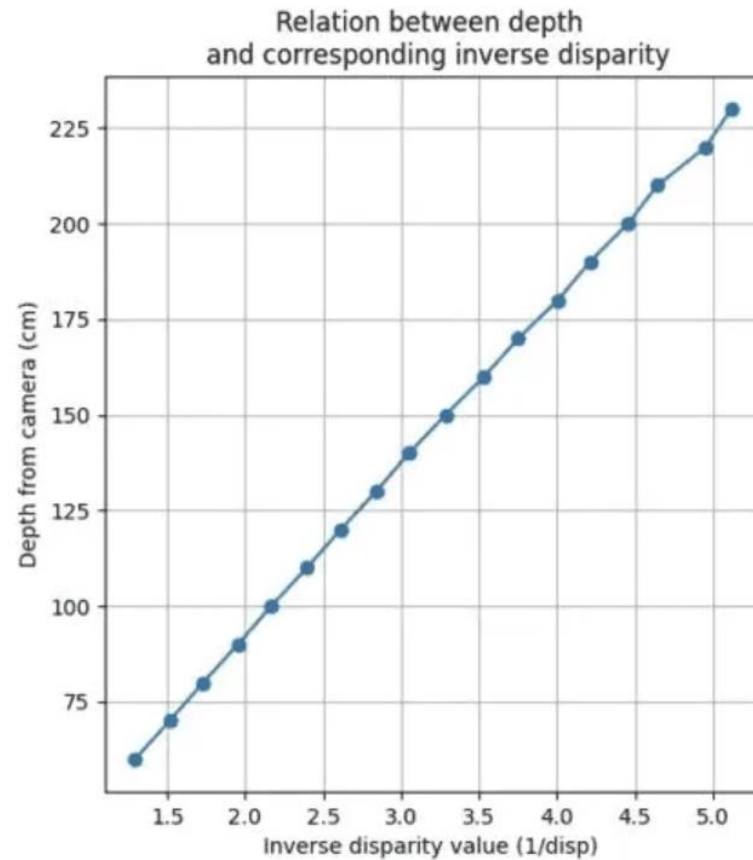
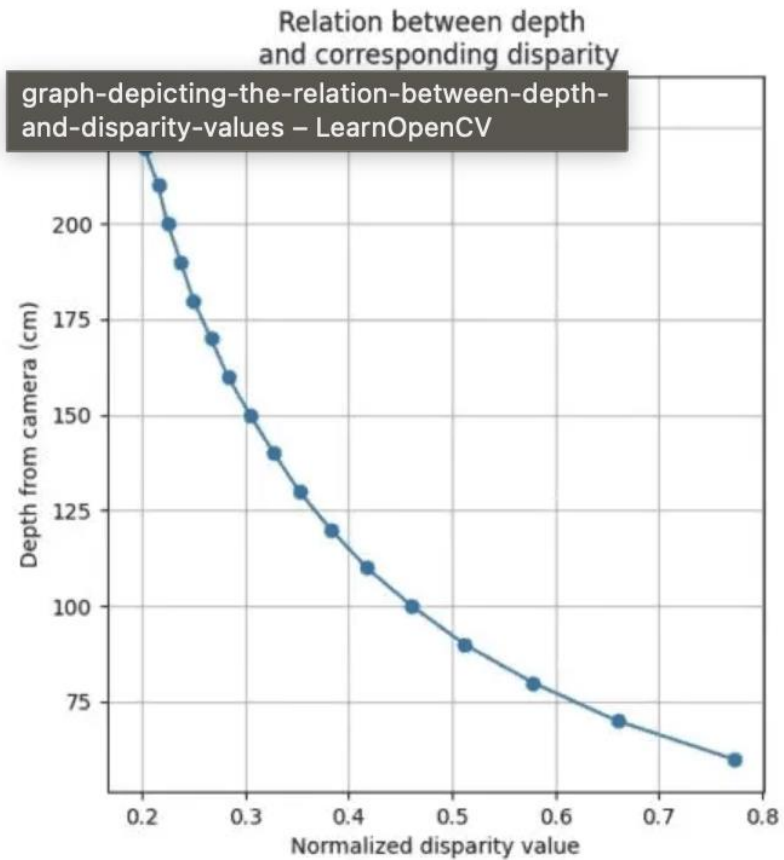
b - displacement of cameras (in x plane)

$$U = f_x * \frac{x}{z} + O_x$$
$$V = f_y * \frac{y}{z} + O_y$$

$$U' = f_x * \frac{(x - b)}{z} + O_{x'}$$
$$V' = f_{y'} * \frac{y}{z} + O_{y'}$$

$$Z = \frac{b}{\frac{U - O_x}{f_x} - \frac{U' - O_{x'}}{f_{x'}}}$$





$$Z = M * \left(\frac{1}{\frac{U - Ox}{Fx} - \frac{U' - Ox'}{Fx'}} \right)$$

<https://learnopencv.com/depth-perception-using-stereo-camera-python-c/>

Structured Illumination

Option 1. Stereo vision

- If you use two cameras, you can compute the distance/disparity between corresponding dots in the two image

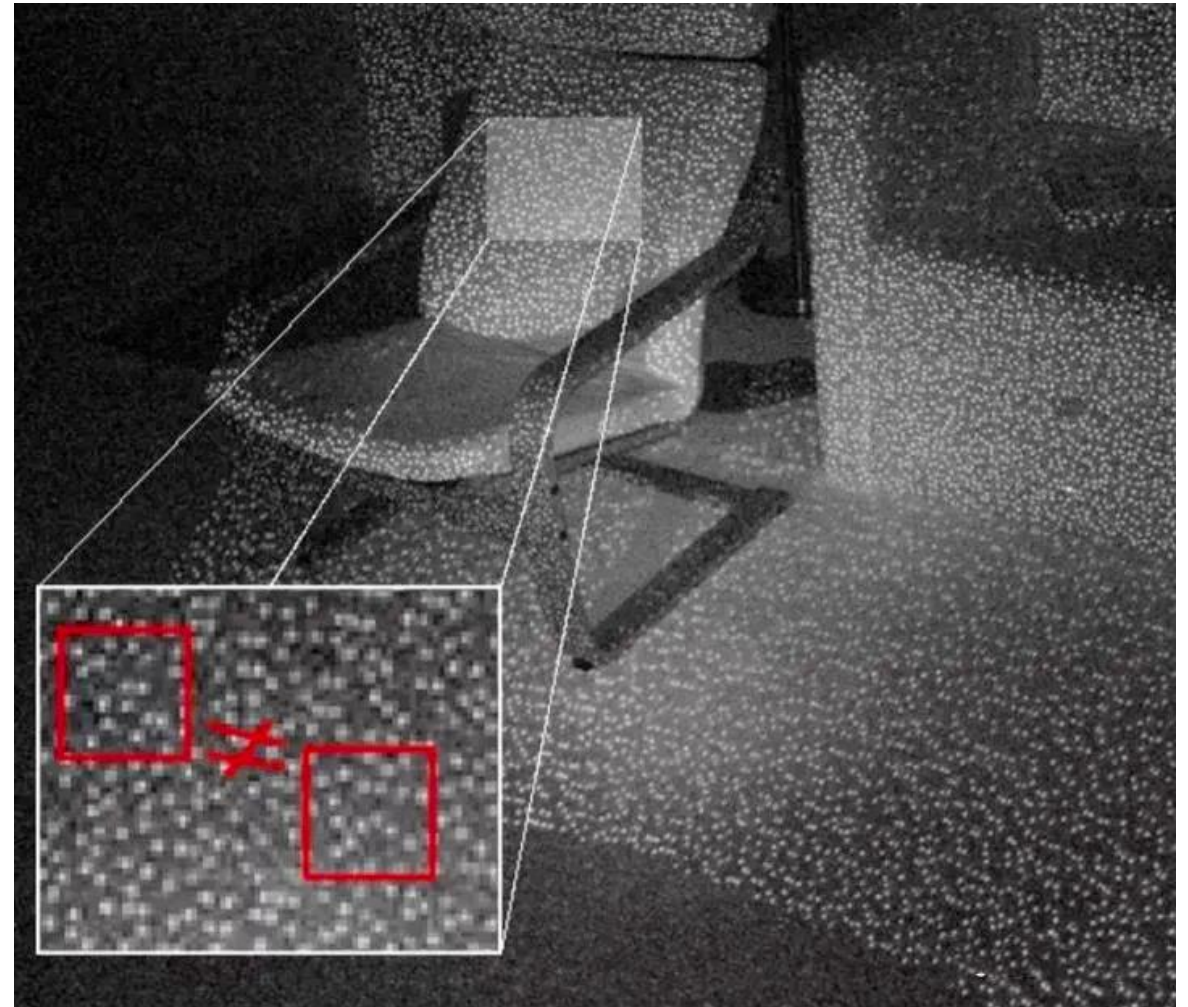
Option 2. One Camera/Shift phase of structure

- Changing the spatial phase of the illumination is the same as moving the source of the illumination. Thus, creating virtual offset patterns.

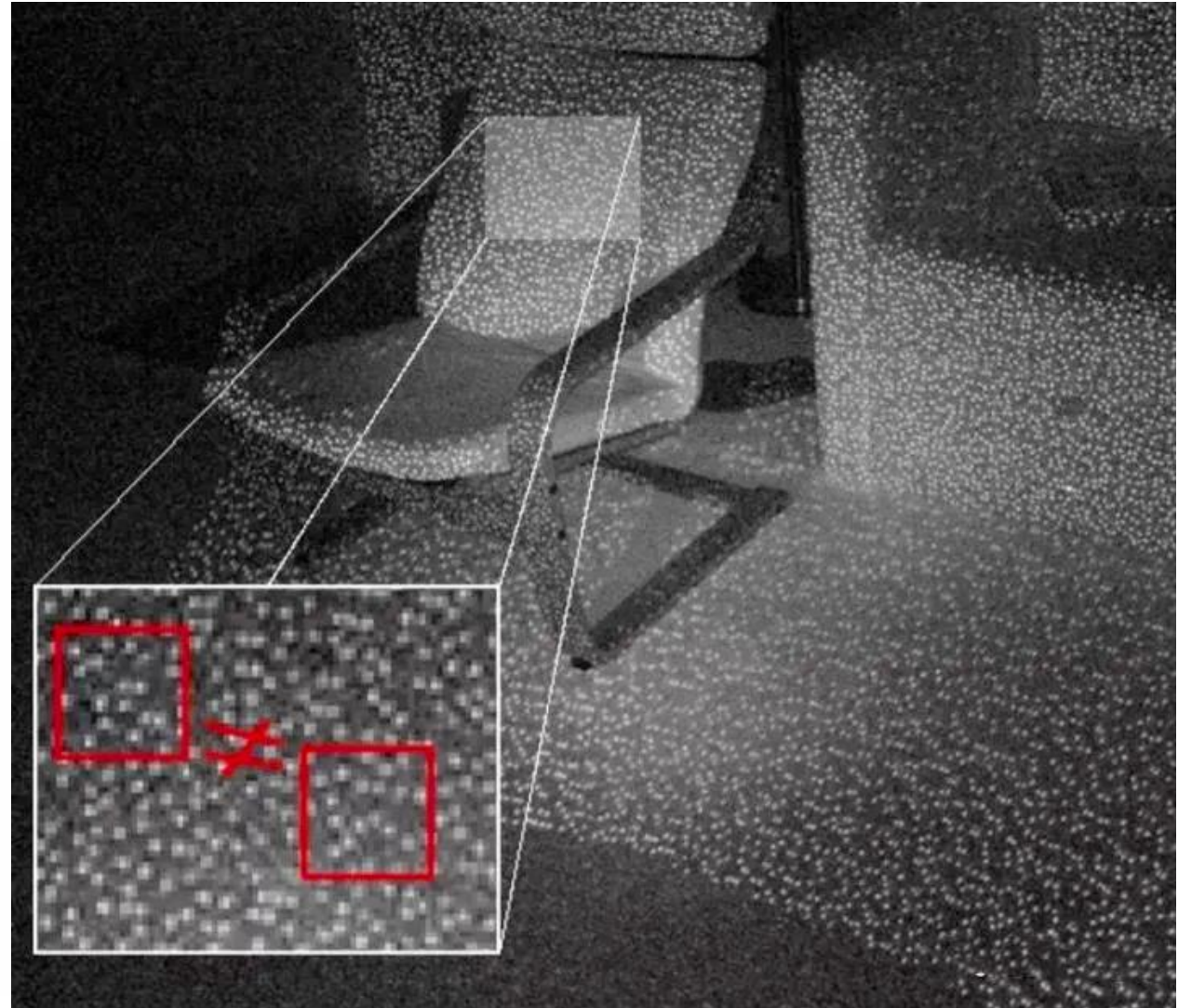
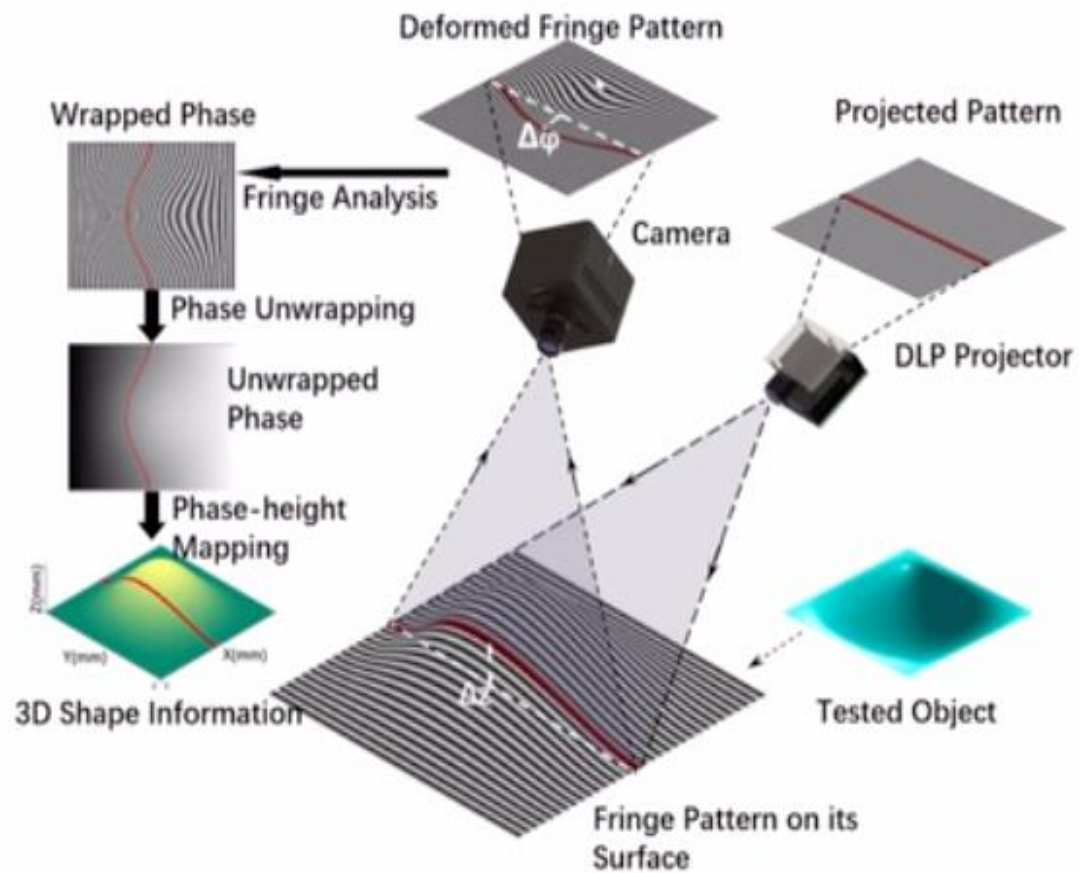
Option 3. Both

- Here is an example (with code) for doing this yourself

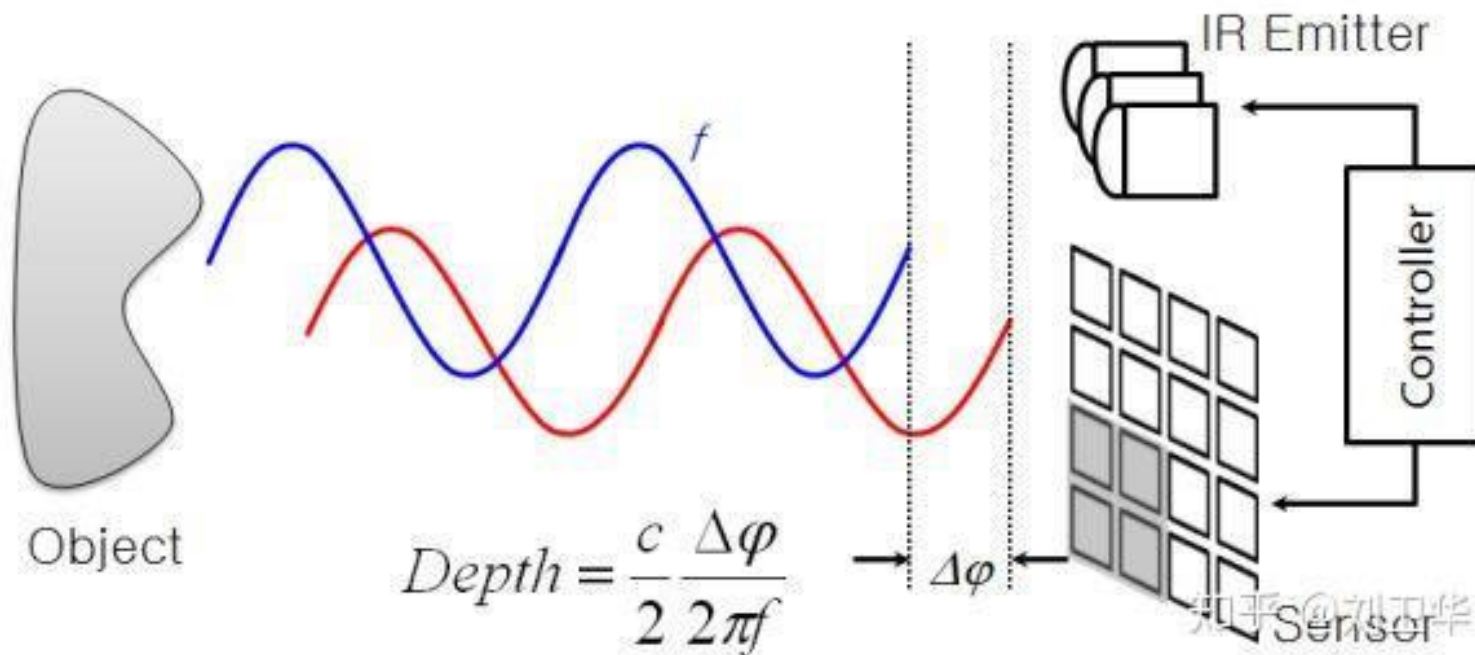
<https://www.instructables.com/DIY-3D-scanner-based-on-structured-light-and-stereo/>



Structured Illumination



LIDAR (ToF)



Monocular depth estimation

- MiDAS (<https://github.com/isl-org/MiDaS>)
 - Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., & Koltun, V. (2020). Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, 44(3), 1623-1637.
 - MIT License
 - Fastest
 - Qualitative. Depth order (but not quantitative distances)
- ZoeDepth (<https://github.com/isl-org/ZoeDepth>)
 - Bhat, S. F., Birkel, R., Wofk, D., Wonka, P., & Müller, M. (2023). Zoedepth: Zero-shot transfer by combining relative and metric depth. *arXiv preprint arXiv:2302.12288*.
 - MIT License
 - ~ 4x slower than MiDAS
 - Depth estimation in quantitative
- Marigold (<https://github.com/prs-eth/Marigold>)
 - Ke, B., Obukhov, A., Huang, S., Metzger, N., Daut, R. C., & Schindler, K. (2024). Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 9492-9502).
 - Apache License
 - ~ 200x slower than MiDAS
- PatchFusion (<https://github.com/zhyever/PatchFusion>)
 - Li, Z., Bhat, S. F., & Wonka, P. (2024). Patchfusion: An end-to-end tile-based framework for high-resolution monocular metric depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10016-10025).
 - MIT License
 - ~ 400x slower than MiDAS

• MiDAS

- Deep Learning based Model
- ResNET based architecture
- Trained on movies available in both 3D and standard format
-



TABLE 2
List of films and the number of extracted frames in the 3D Movies dataset after automatic processing.

Movie title	# frames
Training set	75074
Battle of the Year (2013)	4821
Billy Lynn's Long Halftime Walk (2016)	4178
Drive Angry (2011)	328
Exodus: Gods and Kings (2014)	8063
Final Destination 5 (2011)	1437
A very Harold & Kumar 3D Christmas (2011)	3690
Hellbenders (2012)	120
The Hobbit: An Unexpected Journey (2012)	8874
Hugo (2011)	3189
The Three Musketeers (2011)	5028
Nurse 3D (2013)	492
Pina (2011)	1215
Dawn of the Planet of the Apes (2014)	5571
The Amazing Spider-Man (2012)	5618
Step Up 3D (2010)	509
Step Up: All In (2014)	2187
Transformers: Age of Extinction (2014)	8740
Le Dernier Loup / Wolf Totem (2015)	4843
X-Men: Days of Future Past (2014)	6171
Validation set	3058
The Great Gatsby (2013)	1815
Step Up: Miami Heat / Revolution (2012)	1243
Test set	788
Doctor Who - The Day of the Doctor (2013)	508
StreetDance 2 (2012)	280