AY 2024/2025

EE6222 Machine Vision

3D

Cheng Jun

Institute for Infocomm Research, A*STAR

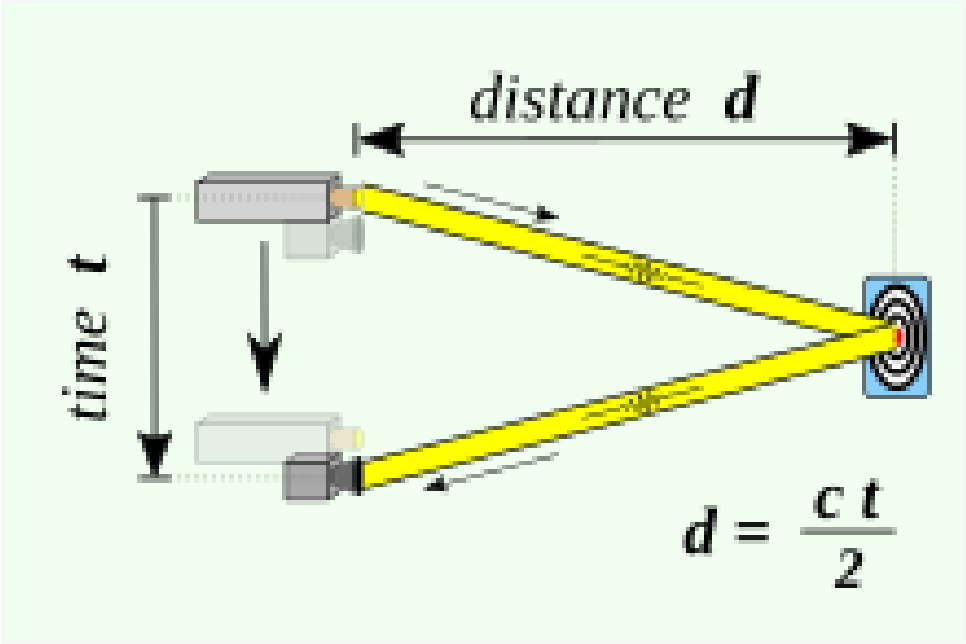
Address: 1 Fusionopolis Way, 138632

Email: cheng_jun@i2r.a-star.edu.sg; cheng.jun@ntu.edu.sg

Approaches for 3D

- Time of Flight
- LiDAR
- Structured Light
- Motion from Structure
- Stereo

Time of Flight



Weakness:

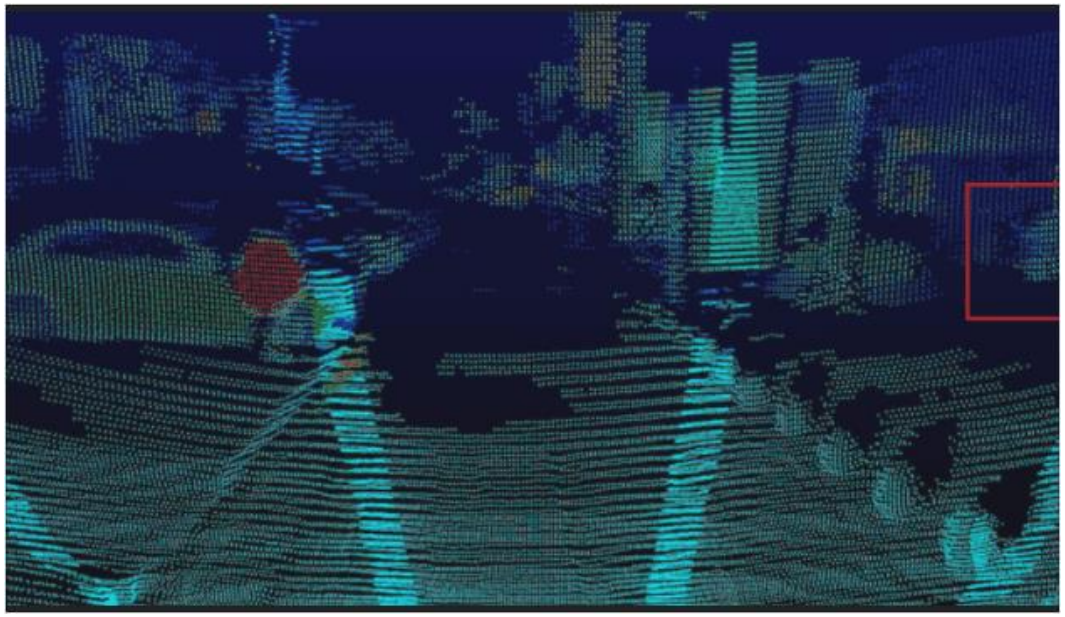
- Background light
- Multiple reflections,
- Interference….

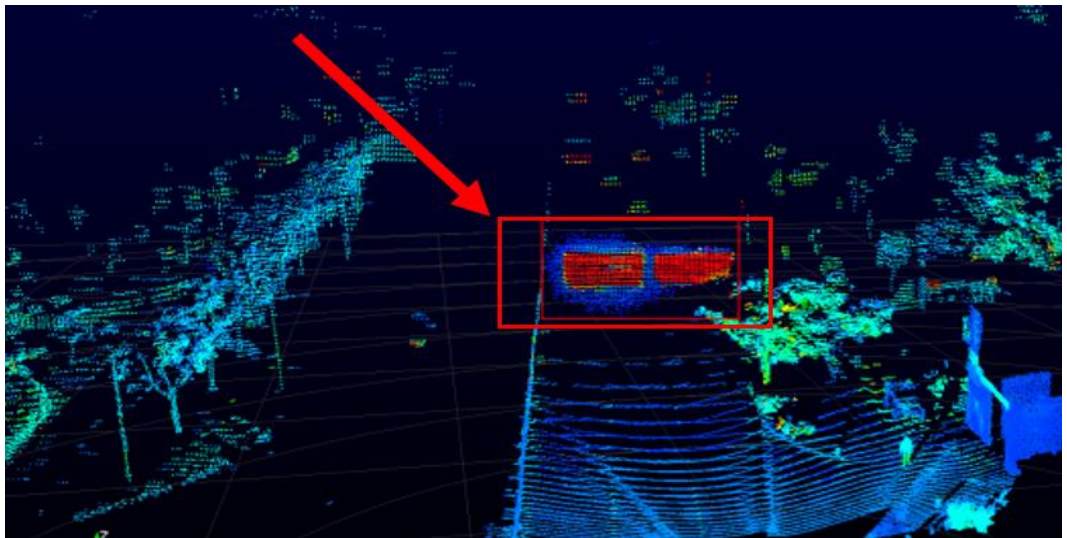
LiDAR: Light Detection and Ranging

LiDAR is a type of Time of Flight

LiDAR Ghost: false readings or artifacts that appear in lidar data, caused by reflections or scattering of the laser beams.

LiDAR Blooming: occurs when the laser beam from a lidar system is reflected or scattered back to the sensor by a particularly bright or reflective object, such as a car with a shiny paint job

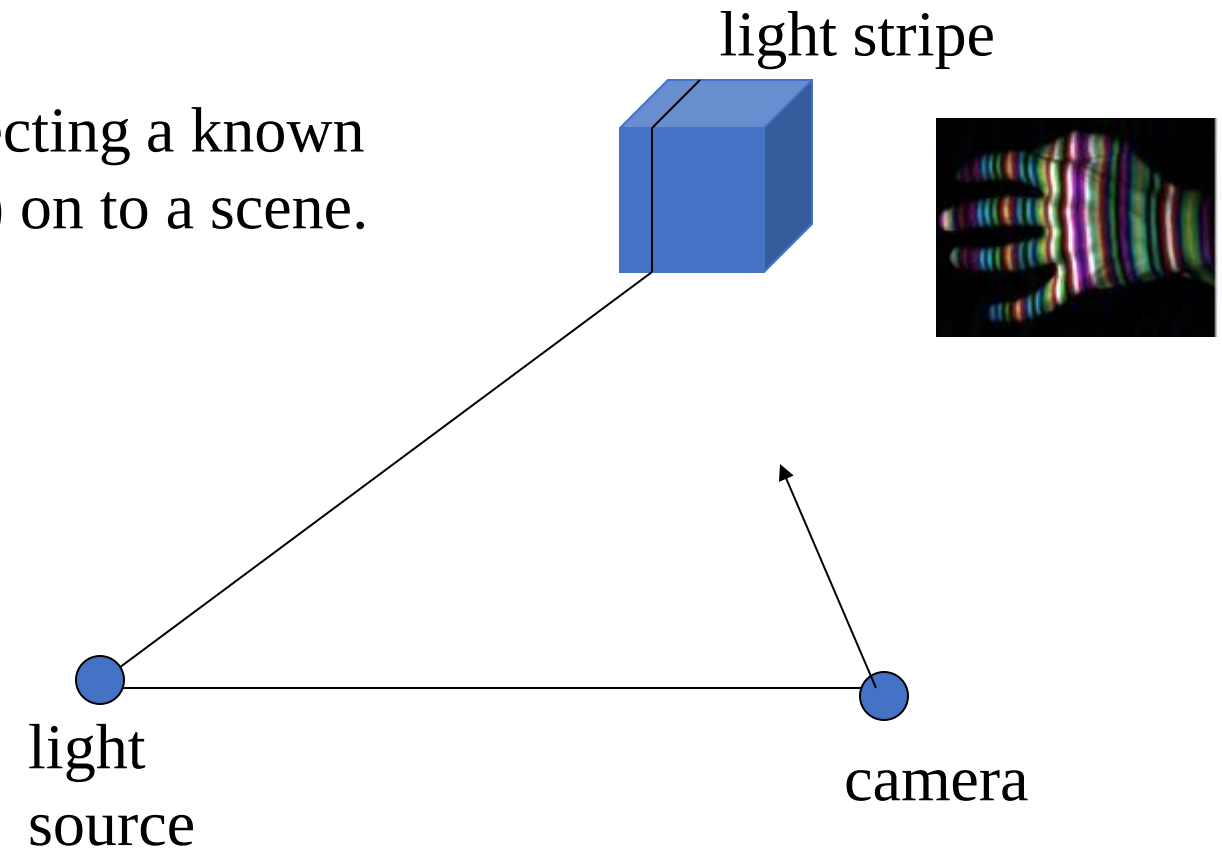




Structured Light

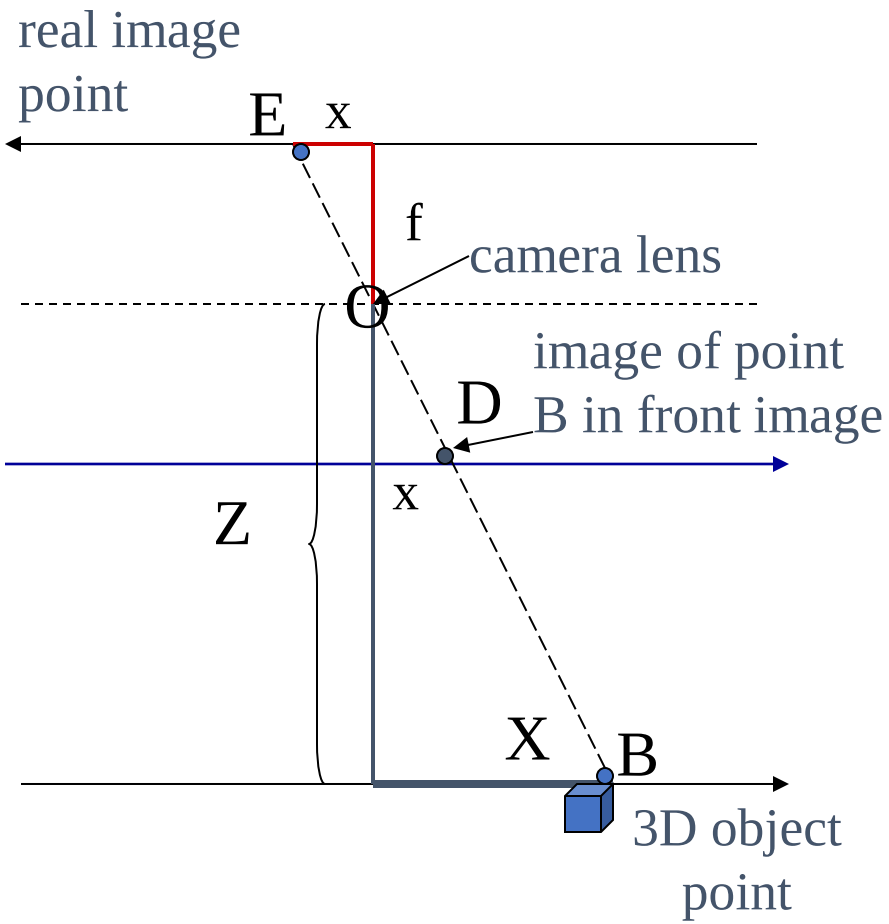
Structured light is the process of projecting a known pattern (often grids or horizontal bars) on to a scene.

Not suitable to outdoor



Structure From Motion

Perspective Imaging Model: 1D



This is the axis of the real image plane.

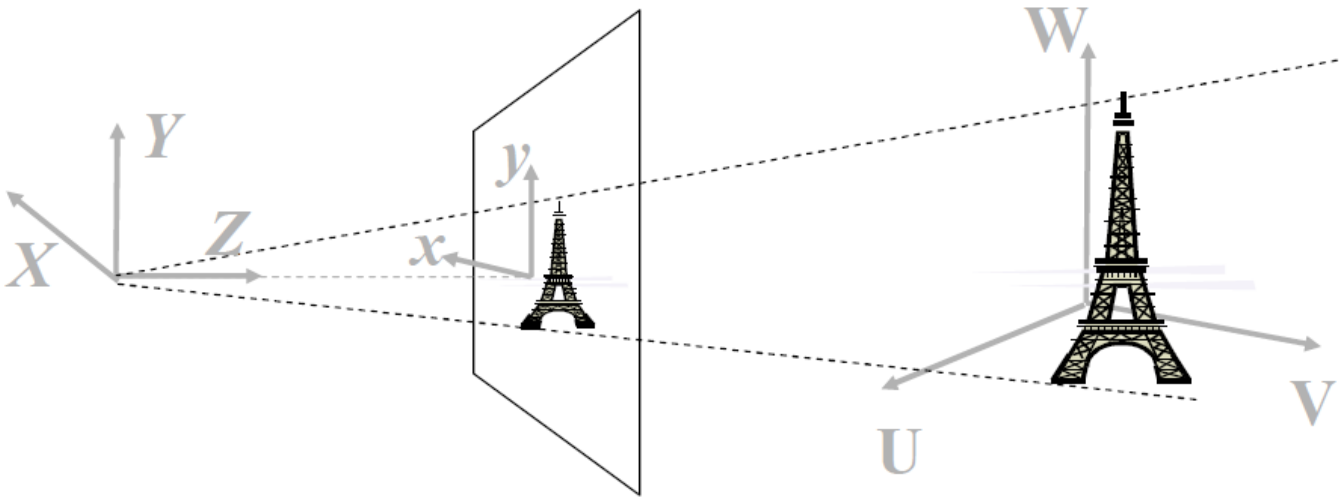
O is the center of projection.

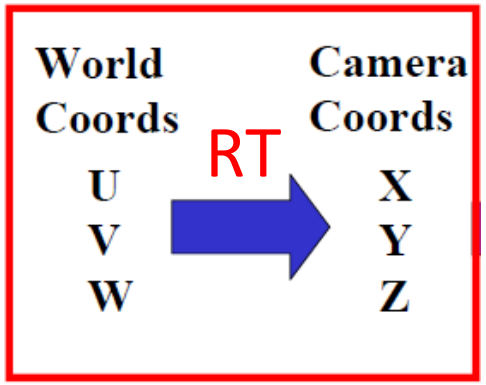
This is the axis of the front image plane, which we use.

$$\frac{x}{f} = \frac{X}{Z}$$

Camera Model

- Many vision tasks try to recover 3D scene structure from images (via stereo or motion)

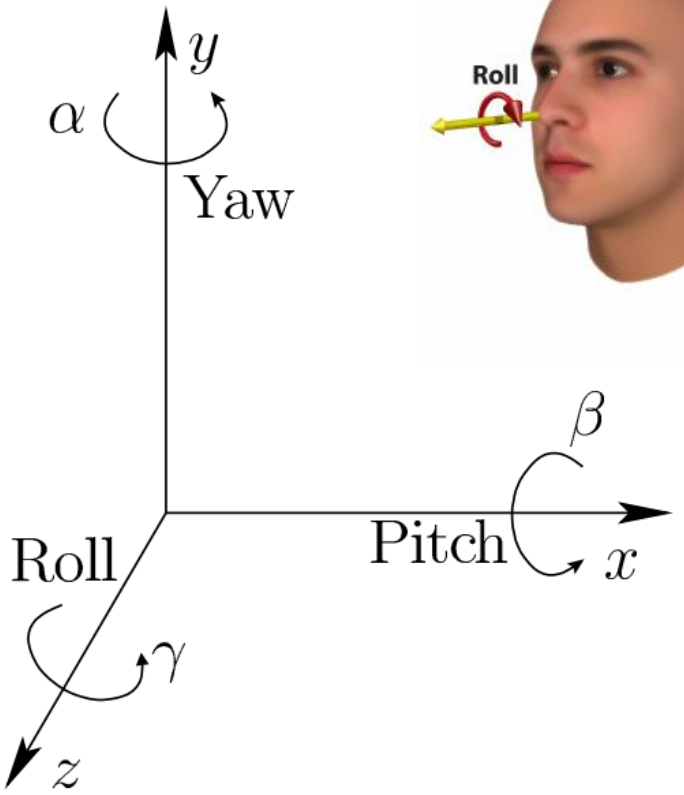


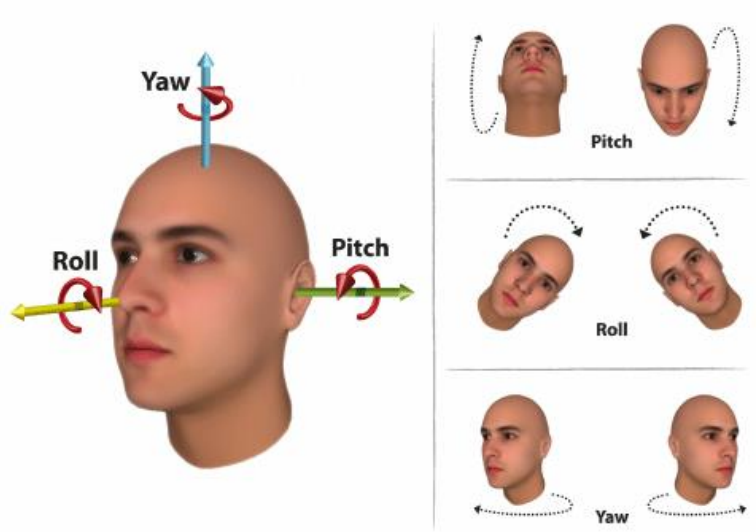


Real world to Image: 3D point in UVW projected to 2D (x,y): Rotation, Translation and Perspective

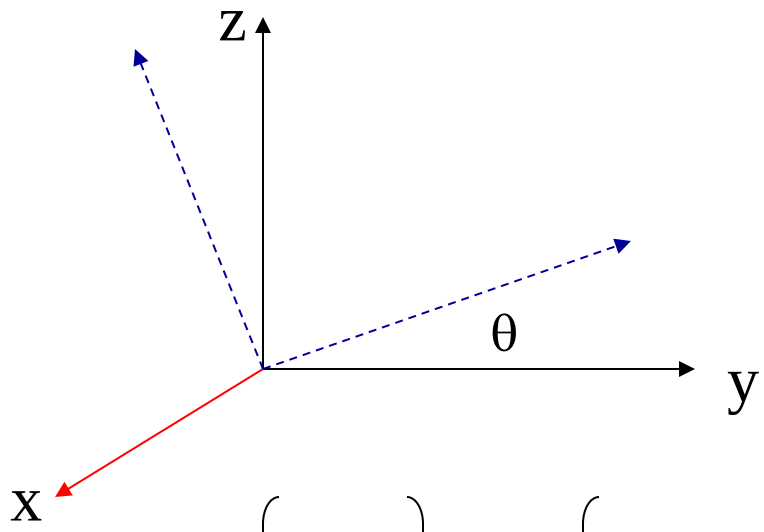
Euler Angles

- As in *Virtual Reality* by Lavalle we will define
  - Roll
    - rotation about z
  - Pitch
    - rotation about x
  - Yaw
    - rotation about y
- Orientation
  - Rz(roll) Rx(pitch) Ry(yaw)





Rotation in 3D is about an axis



rotation by angle θ
about the x axis

$$\begin{pmatrix} Px' \\ Py' \\ Pz' \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} Px \\ Py \\ Pz \end{pmatrix}$$

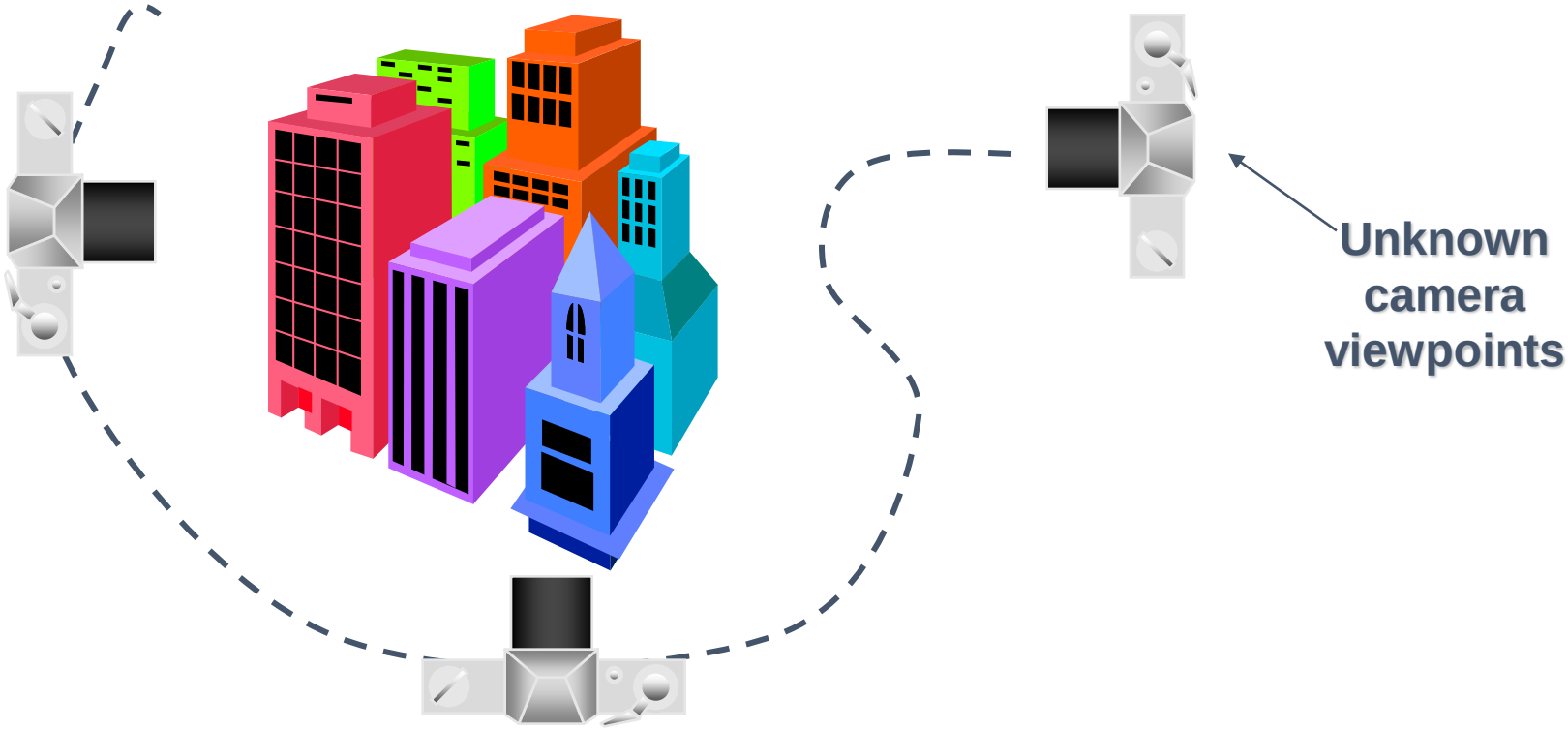
Euler Angles to Matrix Conversion

To build a matrix from a set of Euler angles
just multiply a sequence of rotation matrices together:

$$\mathbf{R}_x \times \mathbf{R}_y \times \mathbf{R}_z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_x & -s_x \\ 0 & s_x & c_x \end{bmatrix} \times \begin{bmatrix} c_y & 0 & s_y \\ 0 & 1 & 0 \\ -s_y & 0 & c_y \end{bmatrix} \times \begin{bmatrix} c_z & -s_z & 0 \\ s_z & c_z & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} c_y c_z & -c_y s_z & s_y \\ s_x s_y c_z + c_x s_z & -s_x s_y s_z + c_x c_z & -s_x c_y \\ -c_x s_y c_z + s_x s_z & c_x s_y s_z + s_x c_z & c_x c_y \end{bmatrix}$$

Structure from motion



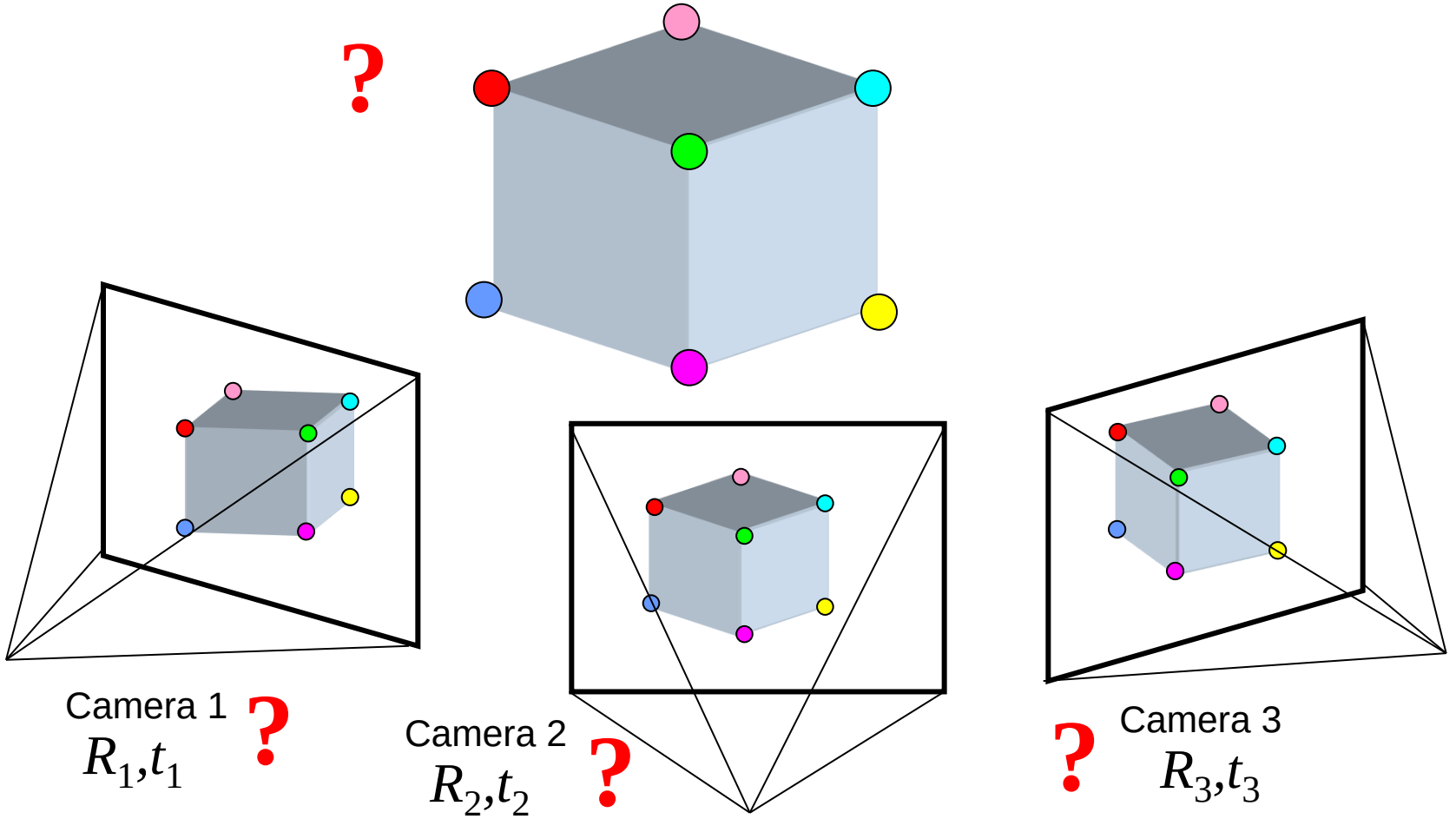
- Reconstruct
  - Scene geometry
  - Camera motion

Motion Capture and Tracking System

Demo of SfM

Structure from motion

- Find a set of corresponding points in two or more images
- Compute the camera parameters and the 3D point coordinates



Structure from motion

- The SfM Problem
  - Reconstruct scene geometry and camera positions from two or more images

- Assume
  - Pixel correspondence
    - via tracking
  - Projection model
    - classic methods are orthographic
    - newer methods use perspective
    - Recently deep learning

Tracking

- Keypoint (Feature) -tracking
  - Extract visual features (corners, textured areas) and “track” them over multiple frames

- Optical flow
  - Recover image motion at each pixel from spatio-temporal image brightness variations (optical flow)

Two problems, one registration method

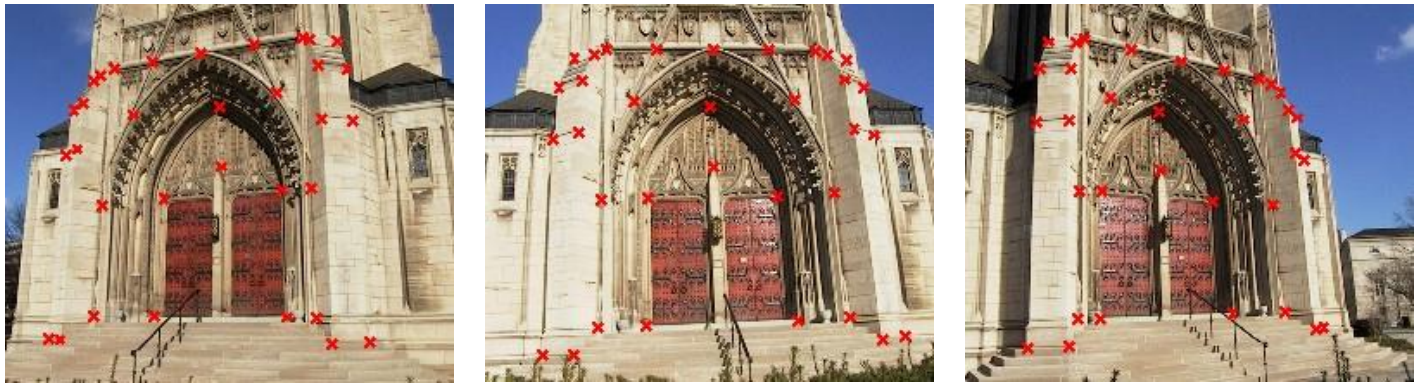
B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.

Keypoint tracking

- Problem
  - Find correspondence between *n* keypoints/features in *f* images

- Issues
  - What's a keypoint/feature?
  - What does it mean to "correspond"?
  - How can correspondence be reliably computed?

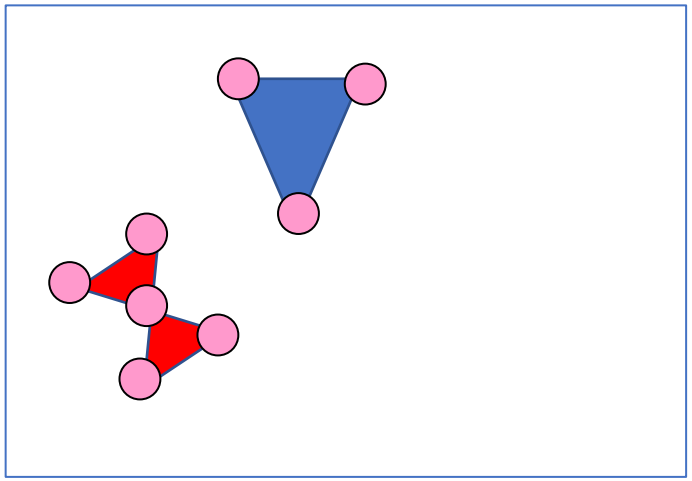
Keypoint/~~Feature~~ tracking

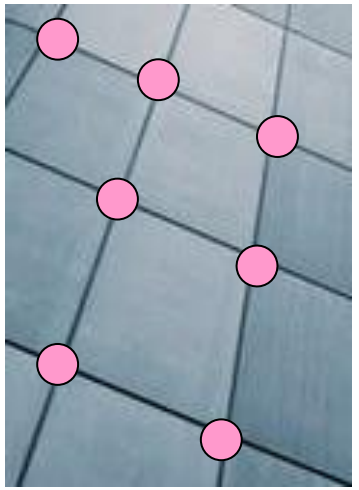


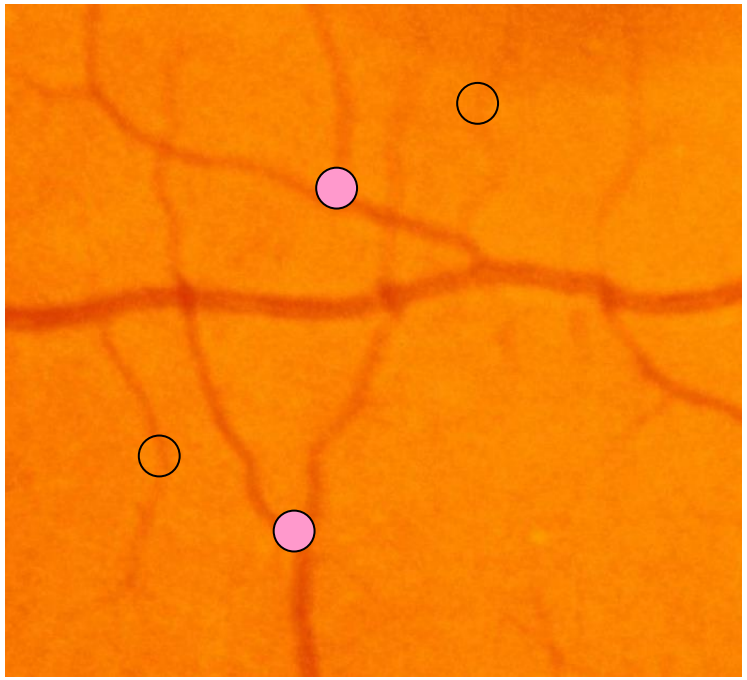
Track Keypoint

- Detect good keypoint/features
  - corners, line segments
- Find correspondences between frames
  - Lucas & Kanade-style motion estimation
  - window-based correlation

What is a good keypoint?







?

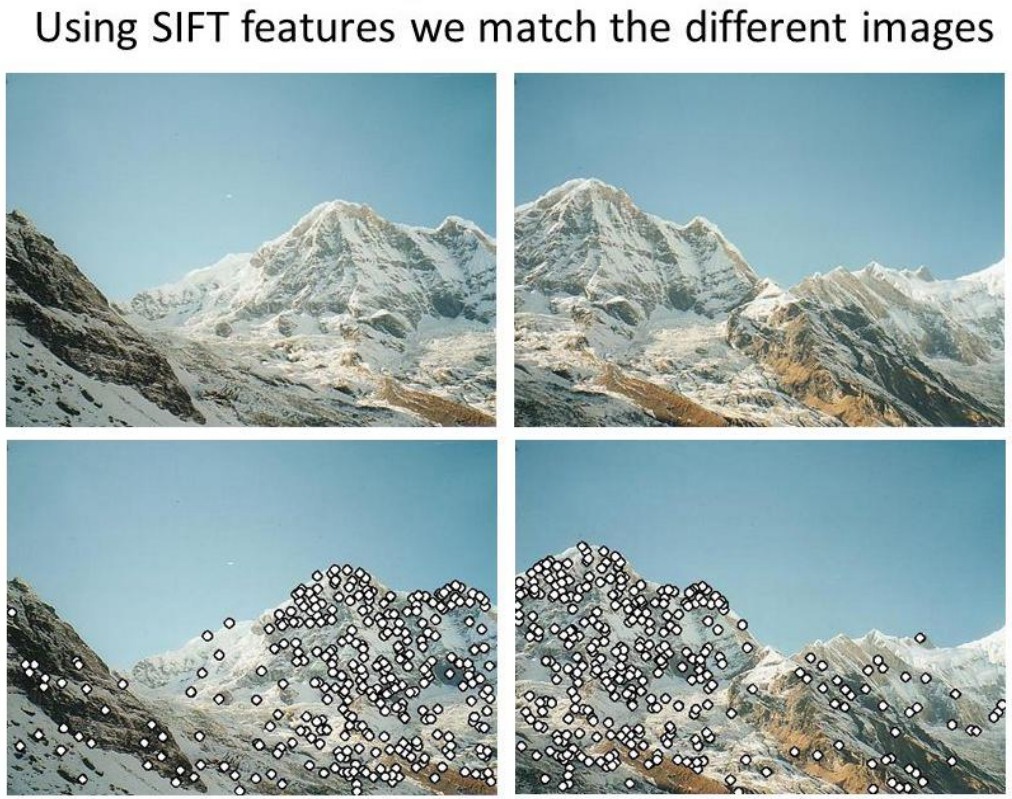
Keypoint tracking

- Challenges
  - Figure out which keypoints can be tracked
  - Efficiently track across frames
  - Some points may change appearance over time (e.g., due to rotation, moving into shadows, etc.)
  - Points may appear or disappear: need to be able to add/delete tracked points

Harris Corner

https://en.wikipedia.org/wiki/Harris_corner_detector

SIFT



*Lowe, David G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints". International Journal of Computer Vision. **60** (2): 91–110.*

Scale space

- Definition: $L(x,y,\sigma) = G(x,y,\sigma) * I(x,y)$

where $G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$

Scale space

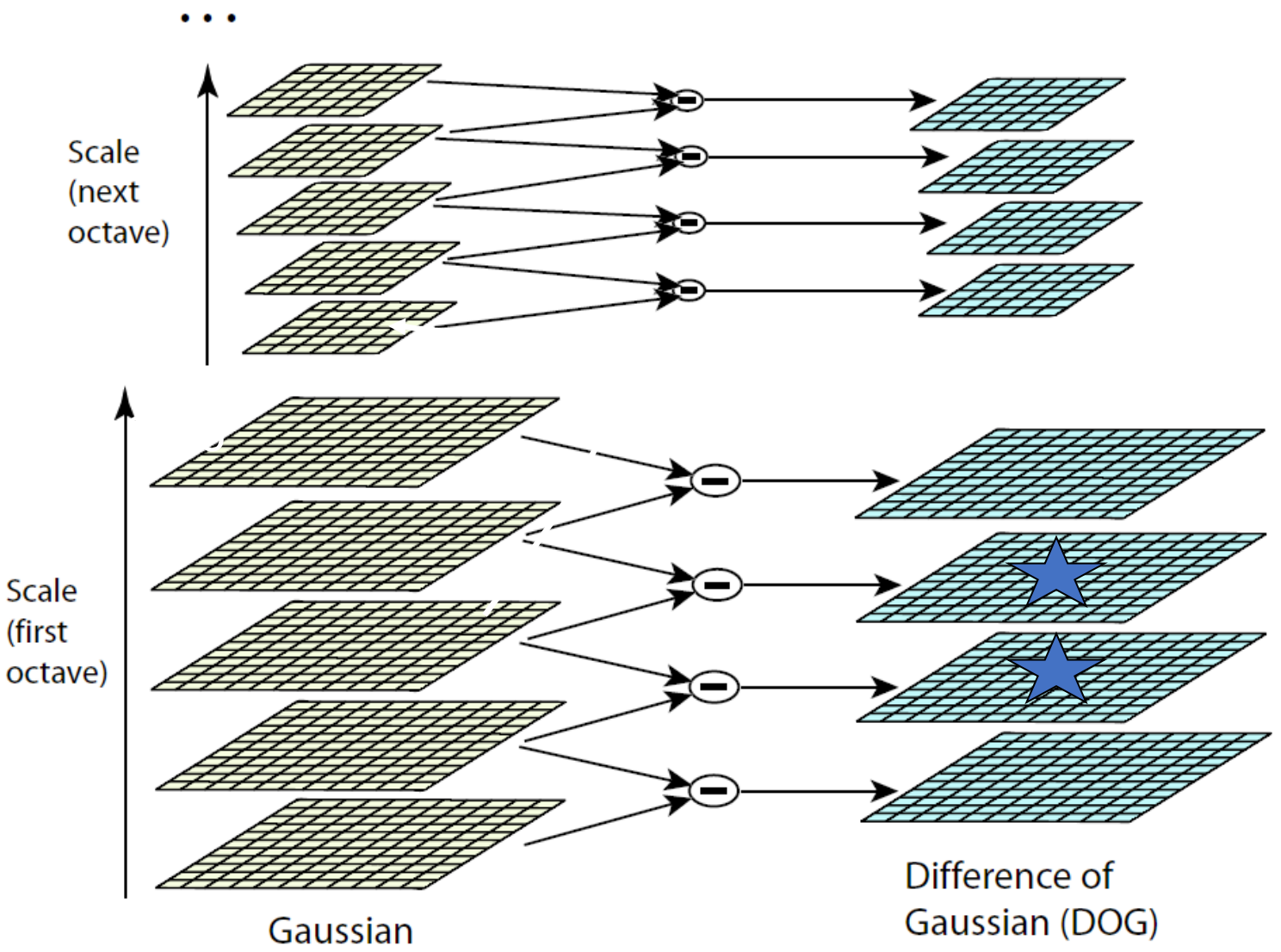
- Keypoints are detected using scale-space extrema in difference-of-Gaussian function *D*
- *D* definition:

$$D(x,y,\sigma) = (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y)$$

$$= L(x,y,k\sigma) - L(x,y,\sigma)$$

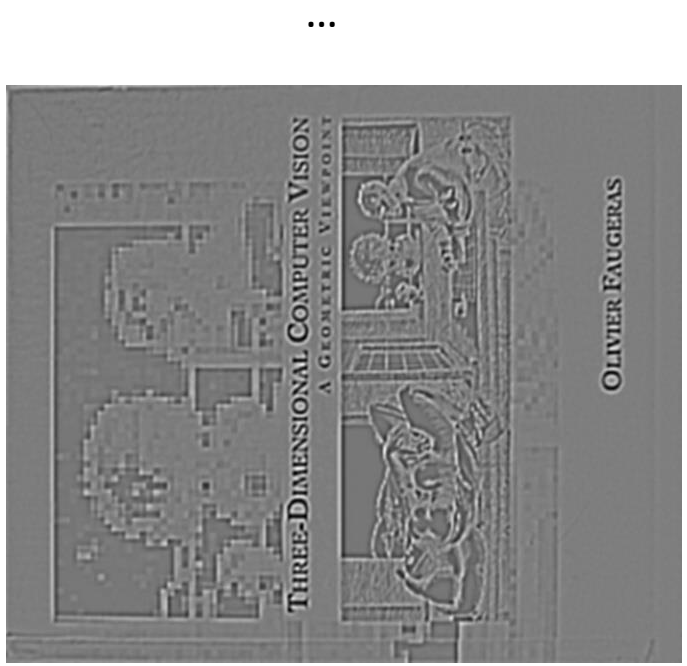
- Efficient to compute

Scale space construction



Scale space images

Difference-of-Gaussian images



first octave

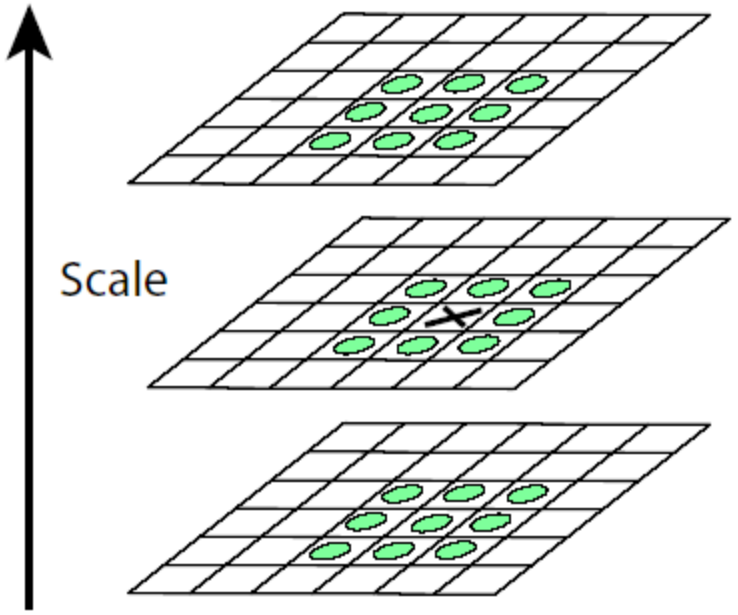
second octave

third octave

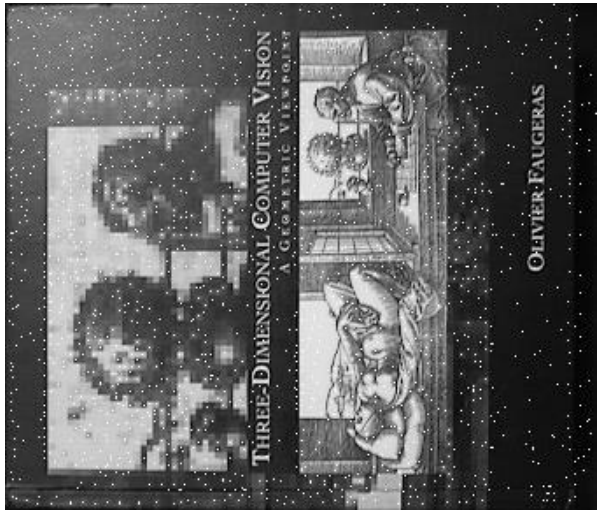
fourth octave

Finding extrema

- Sample point is selected only if it is a minimum or a maximum of these points

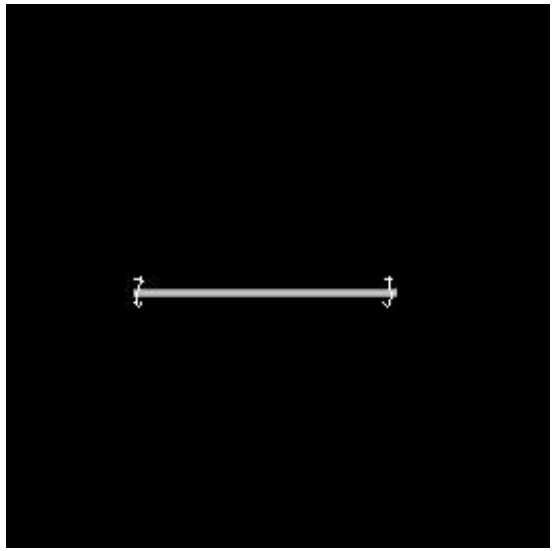


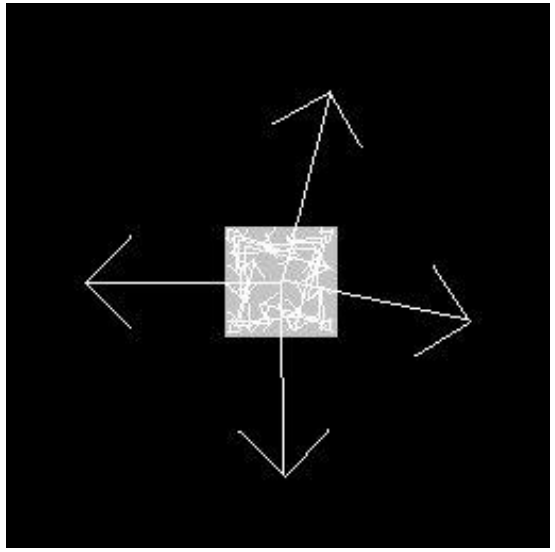
DoG scale space

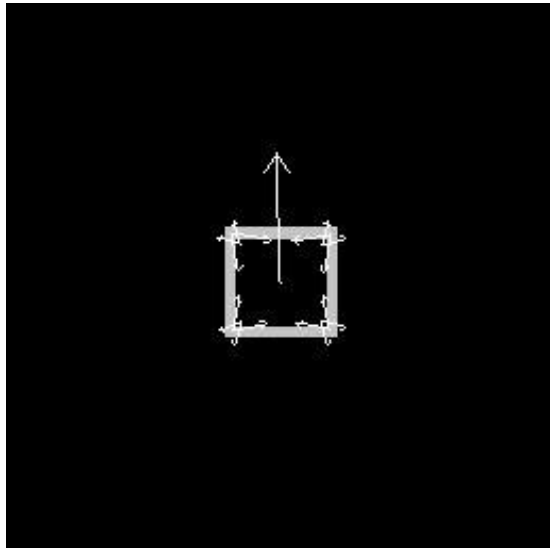


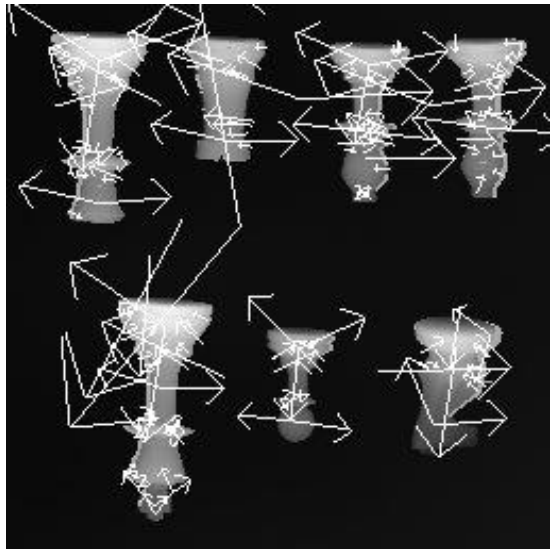
Extrema in this image

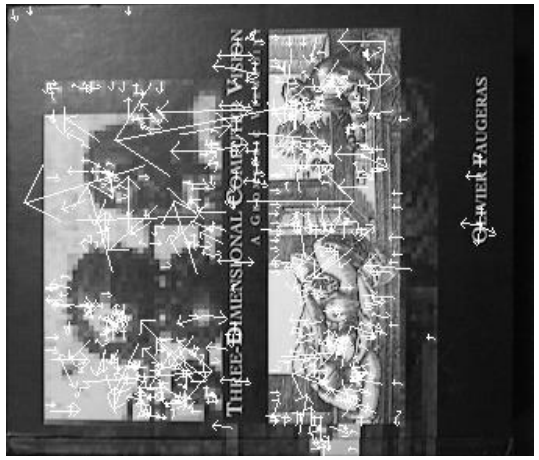
Keypoint images











Descriptor

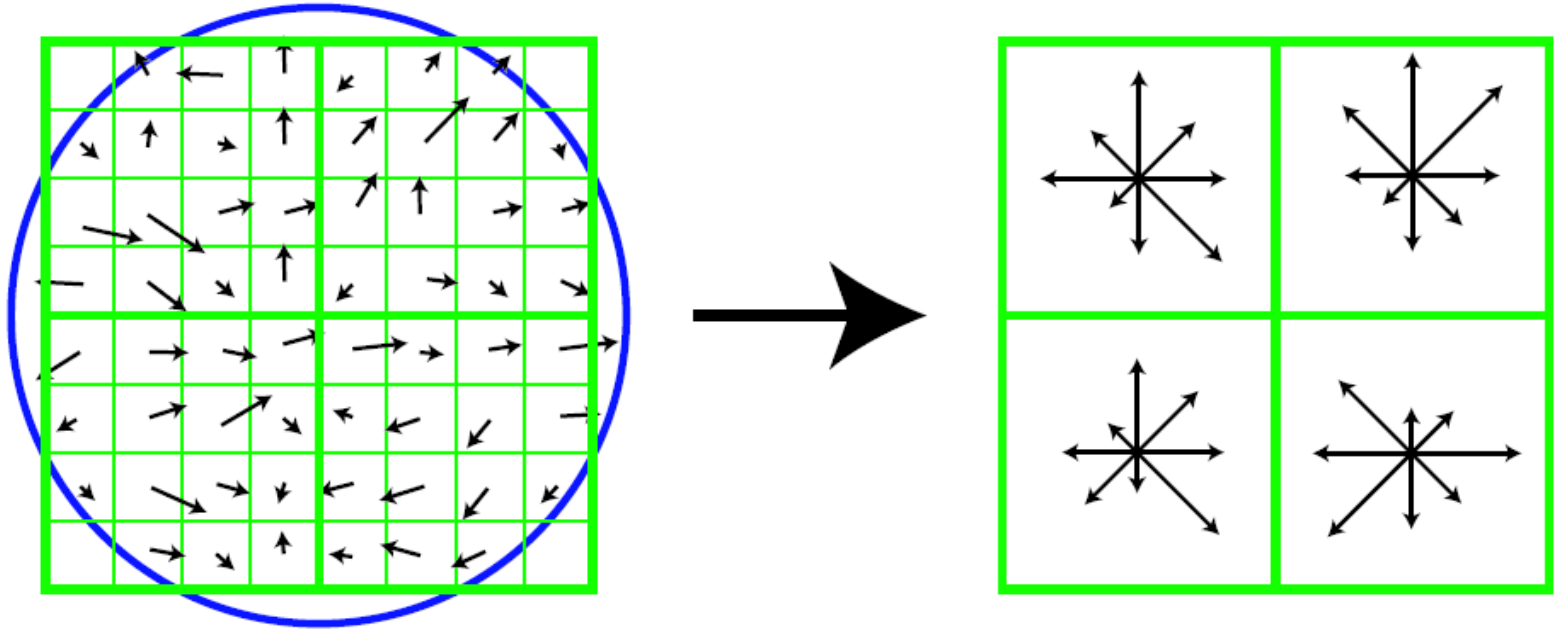


Image gradients

Keypoint descriptor

- Descriptor has 3 dimensions *(x,y,ϑ)*
- Orientation histogram of gradient magnitudes
- Position and orientation of each gradient sample rotated relative to keypoint orientation

Descriptor

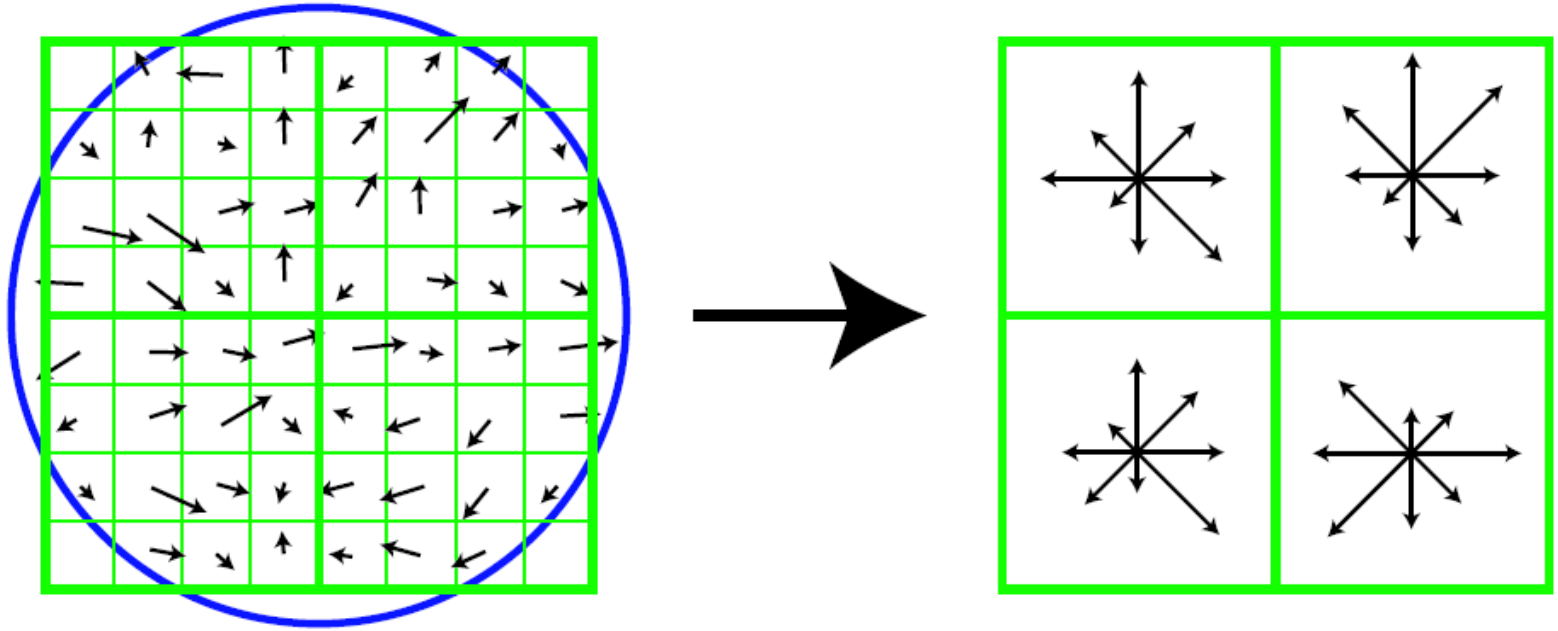


Image gradients

Keypoint descriptor

- Weight magnitude of each sample point by Gaussian weighting function
- Distribute each sample to adjacent bins by trilinear interpolation (avoids boundary effects)

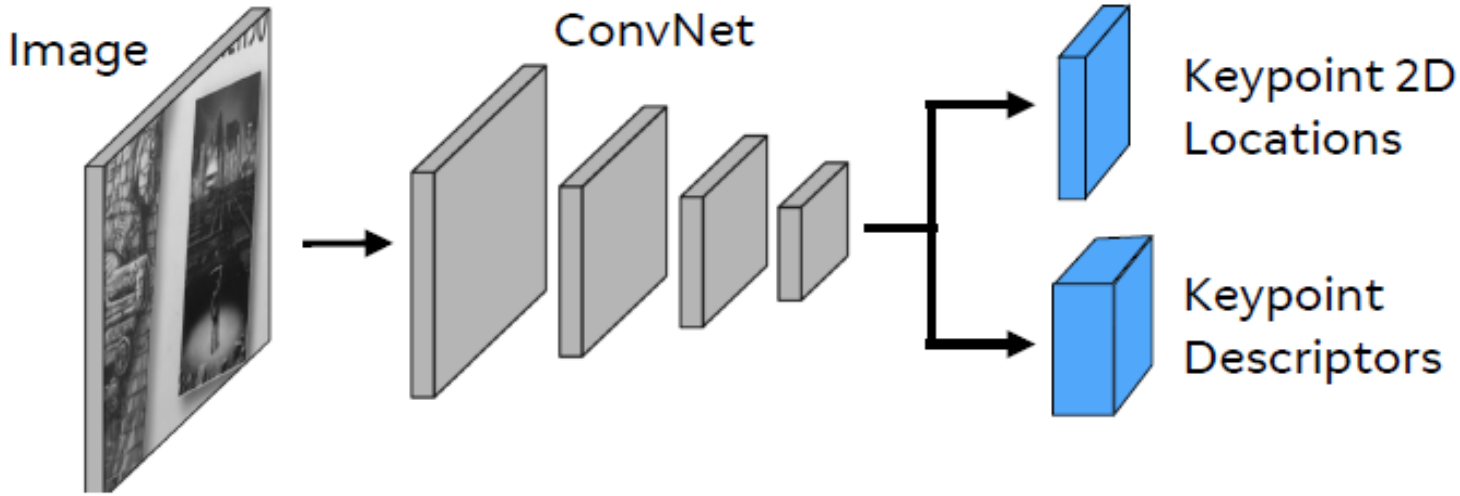
Descriptor

- Best results achieved with 4x4x8 = 128 descriptor size
- Normalize to unit length
  - Reduces effect of illumination change
- Cap each element to 0.2, normalize again
  - Reduces non-linear illumination changes
  - 0.2 determined experimentally

Object Detection

- Create a database of keypoints from training images
- Match keypoints to a database
  - Nearest neighbor search

SuperPoint

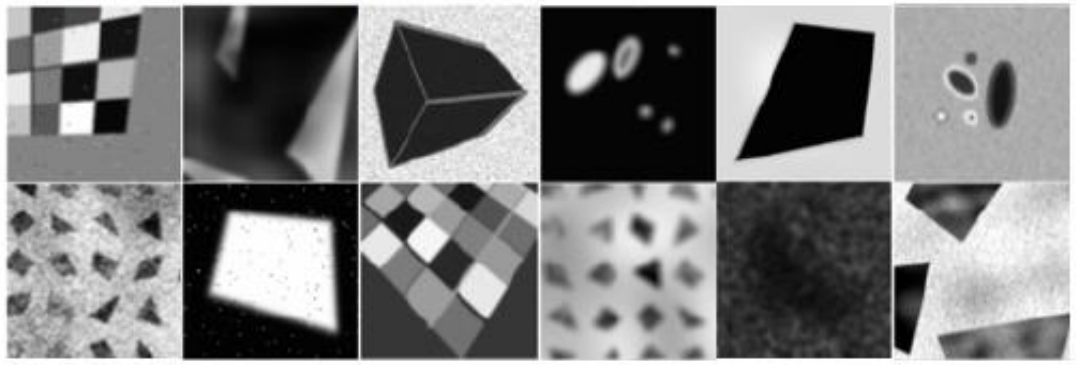


- Fully convolutional design
  - Points + descriptors computed jointly
  - Share VGG-like backbone
- Designed for real-time
  - Tasks share ~90% of compute

Daniel DeTone, et al., SuperPoint: Self-Supervised Interest Point Detection and Description , CVPR workshop 2018

Training of SuperPoint

Synthetic Shapes (has interest point labels)



First train on this

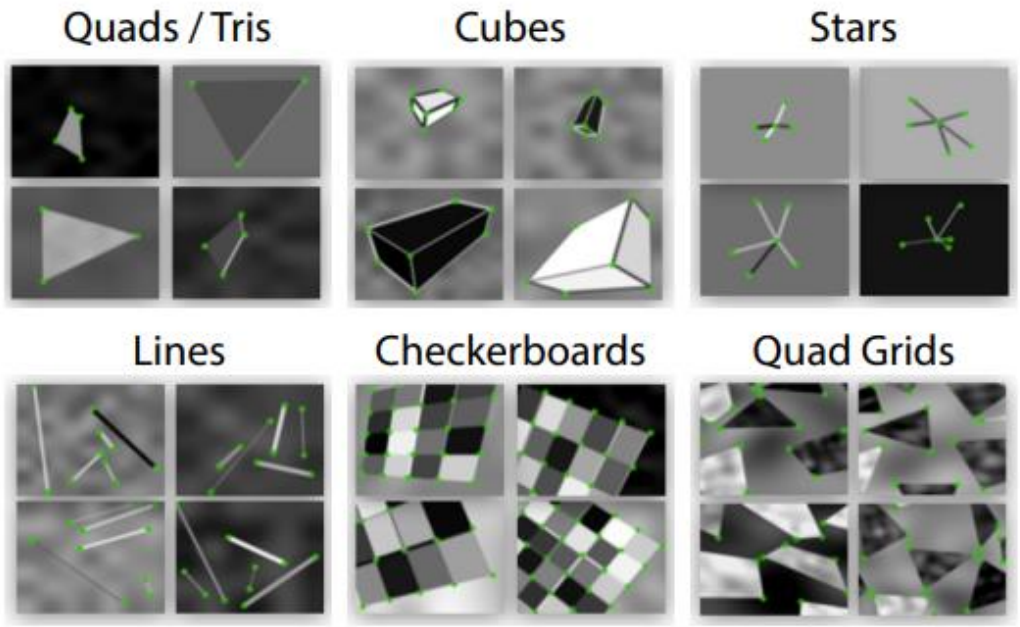
"Homographic Adaptation"

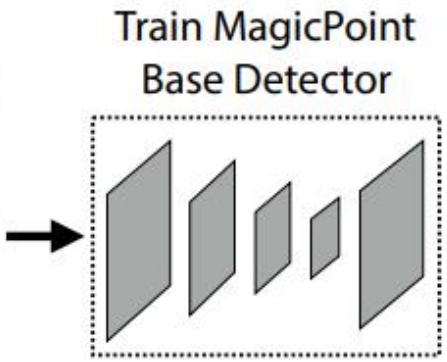
MS-COCO (no interest point labels)

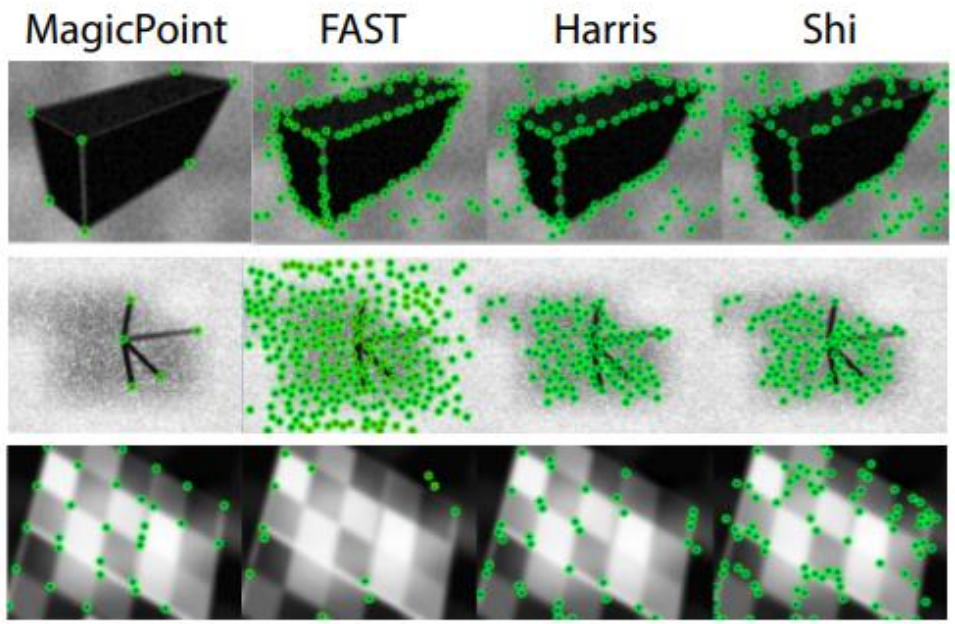
Use resulting detector to label this

Code available: github.com/MagicLeapResearch/SuperPointPretrainedNetwork

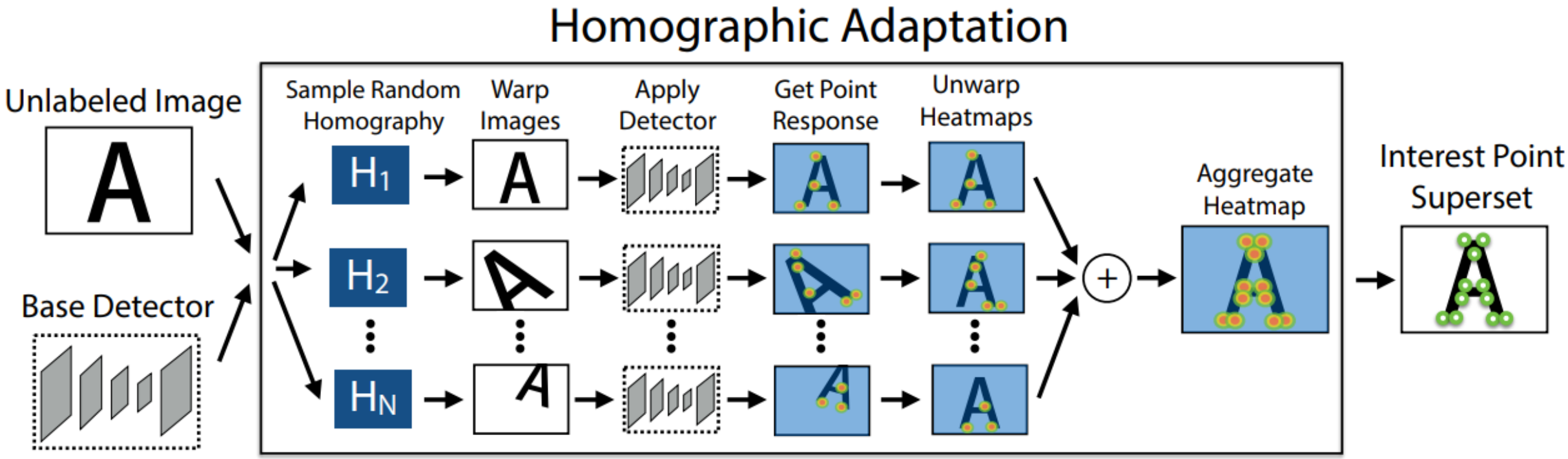
Training of SuperPoint



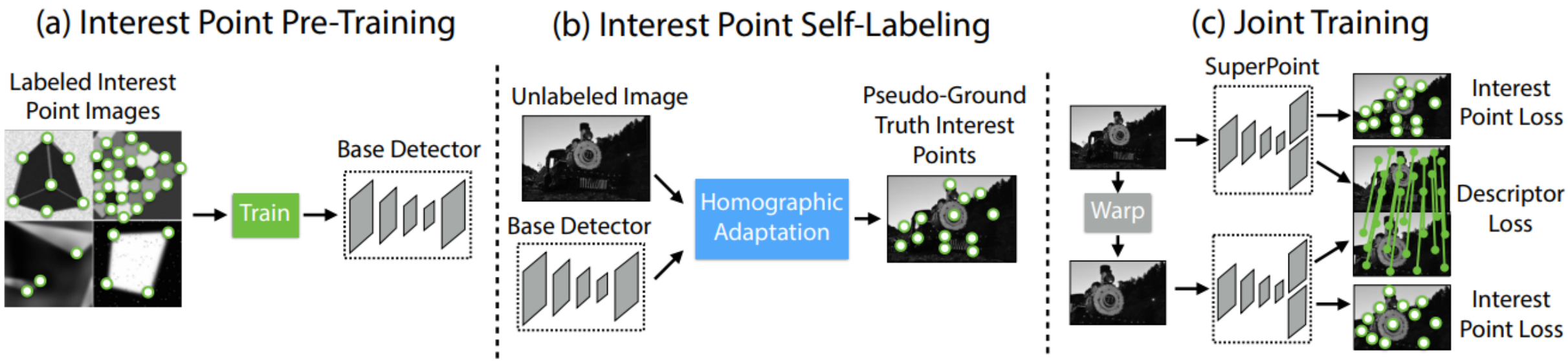




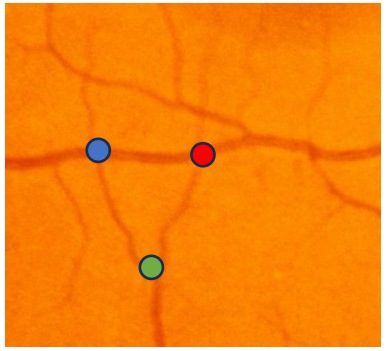
Training of SuperPoint

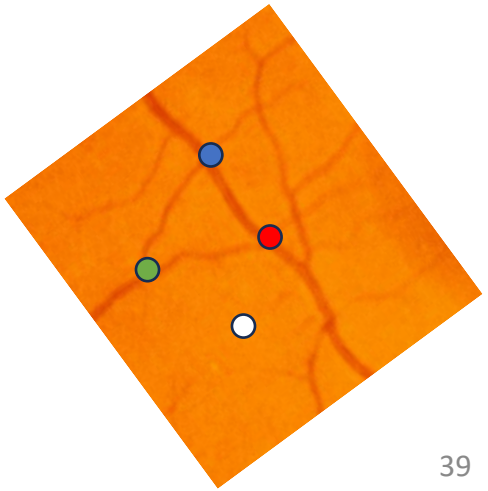


Training of SuperPoint



$$\mathcal{L}(\mathcal{X}, \mathcal{X}', \mathcal{D}, \mathcal{D}'; Y, Y', S) = \mathcal{L}_p(\mathcal{X}, Y) + \mathcal{L}_p(\mathcal{X}', Y') + \lambda \mathcal{L}_d(\mathcal{D}, \mathcal{D}', S).$$





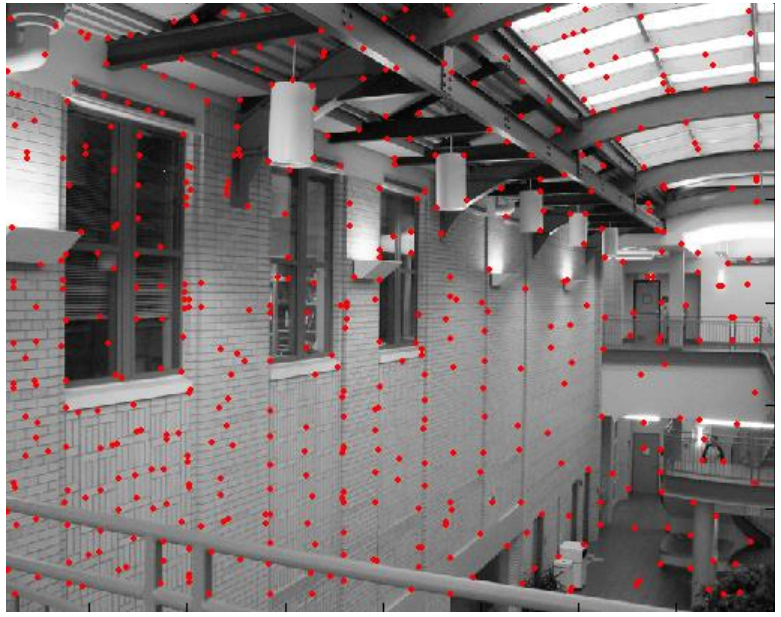
Align Two or More Images

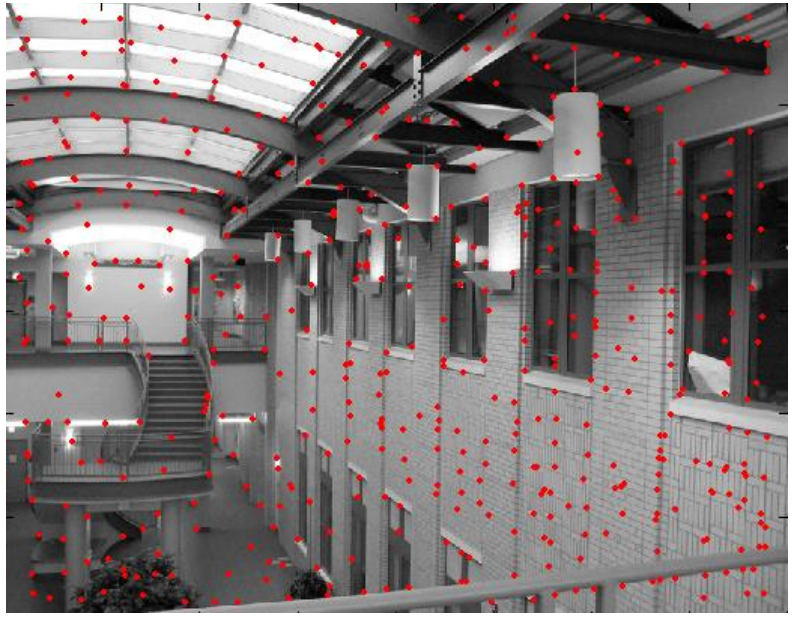
left on top

right on top

Translations are not enough to align the images

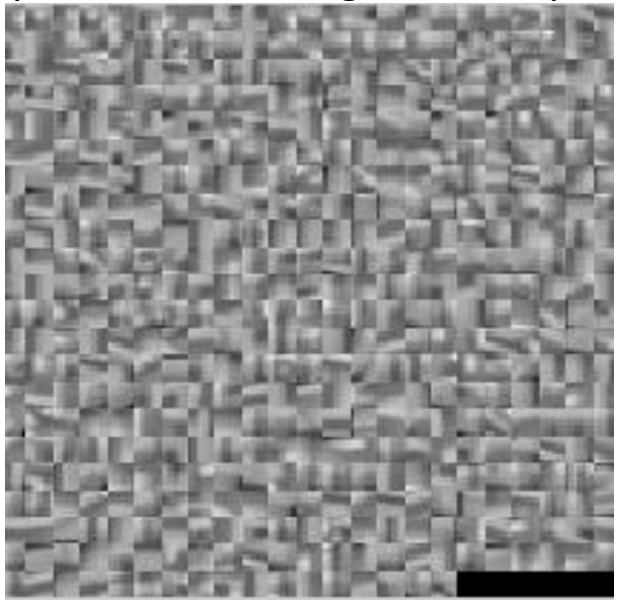
Feature matching

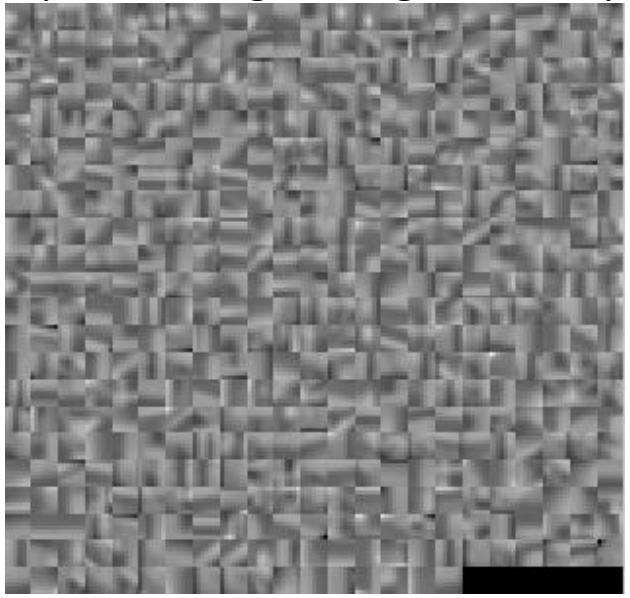




descriptors for left image feature points

descriptors for right image feature points





Strategies to match images robustly

(a) Working with individual keypoints: For each keypoint, find most similar point in other image (SIFT distance)

Reject ambiguous matches where there are too many similar points

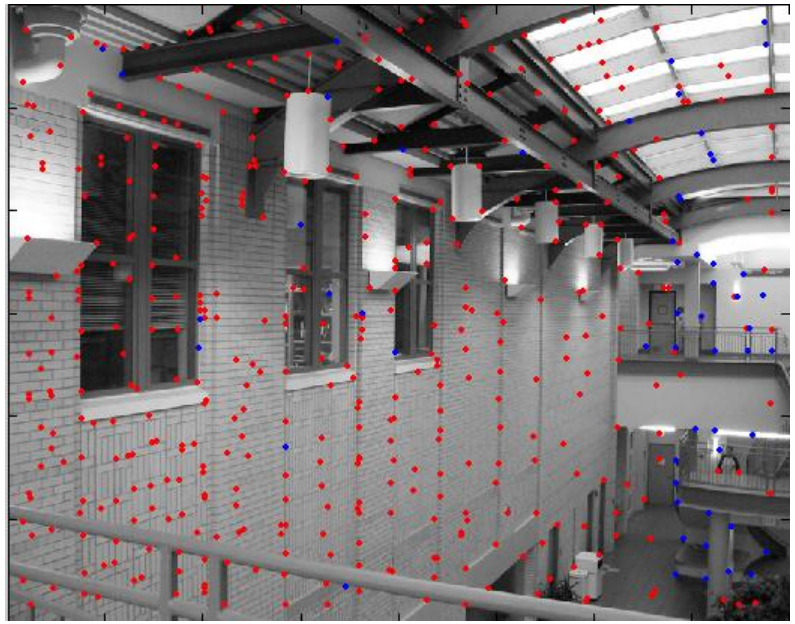
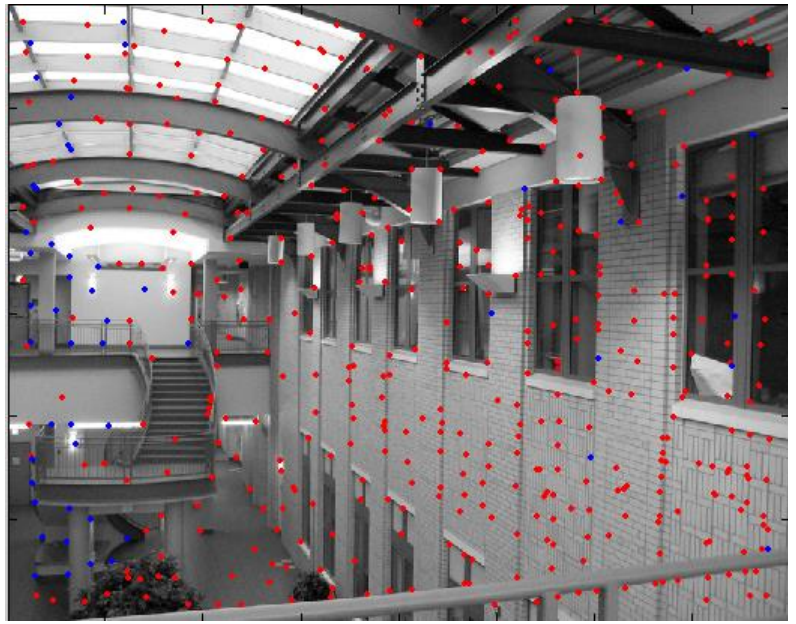
(b) Working with all the keypoints: Given some good keypoint matches, look for possible homographies relating the two images

Reject homographies that don't have many keypoint matches.

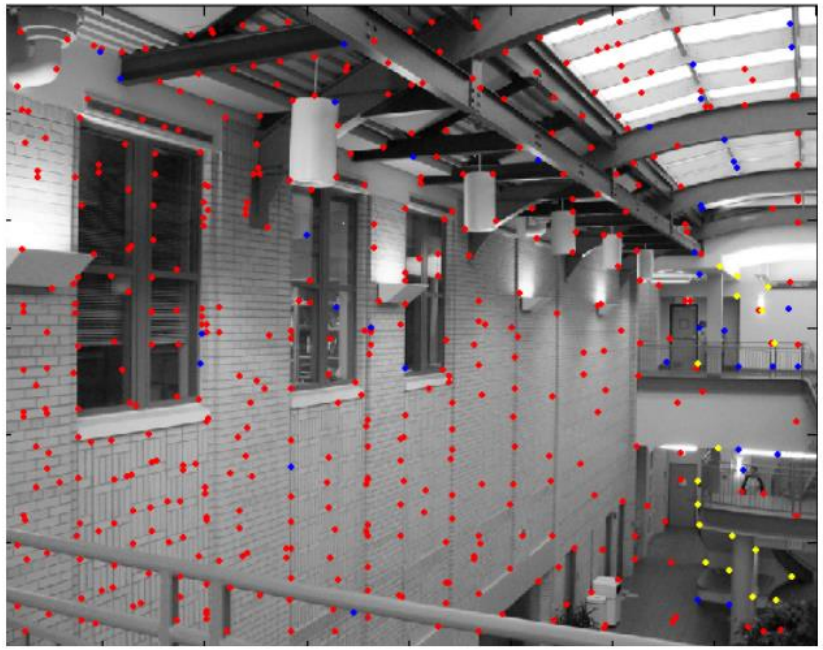
42

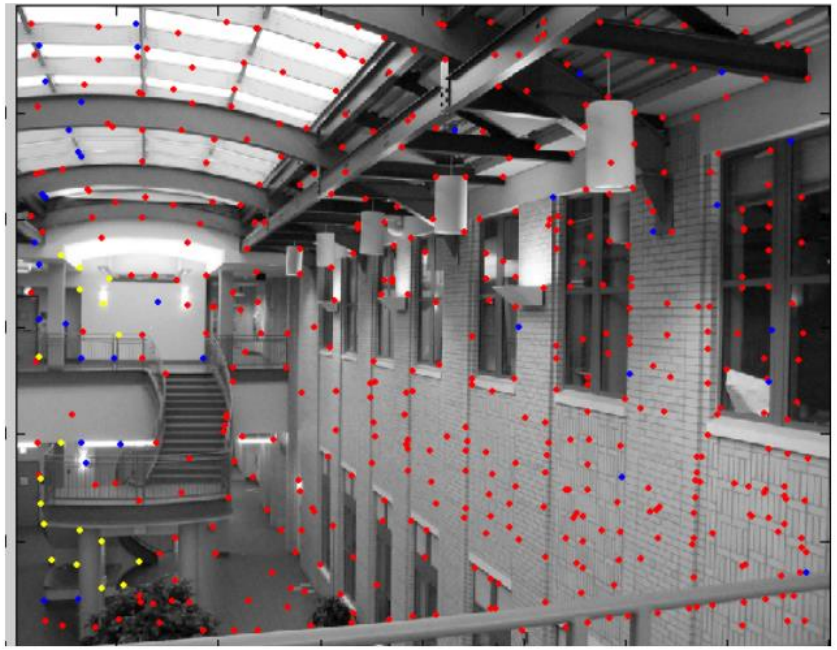
(a) Keypoint-space outlier rejection

- Let's not match all keypoints, but only these that have "similar enough" matches?
- How can we do it?
  - SSD(patch1,patch2) < threshold
  - How to set threshold?
    Not so easy.

- Can we now compute H from the blue points?
  - No! Still too many outliers…
  - What can we do?





Red: points without a “good” match in the other image

Blue: points with a wrong “g match

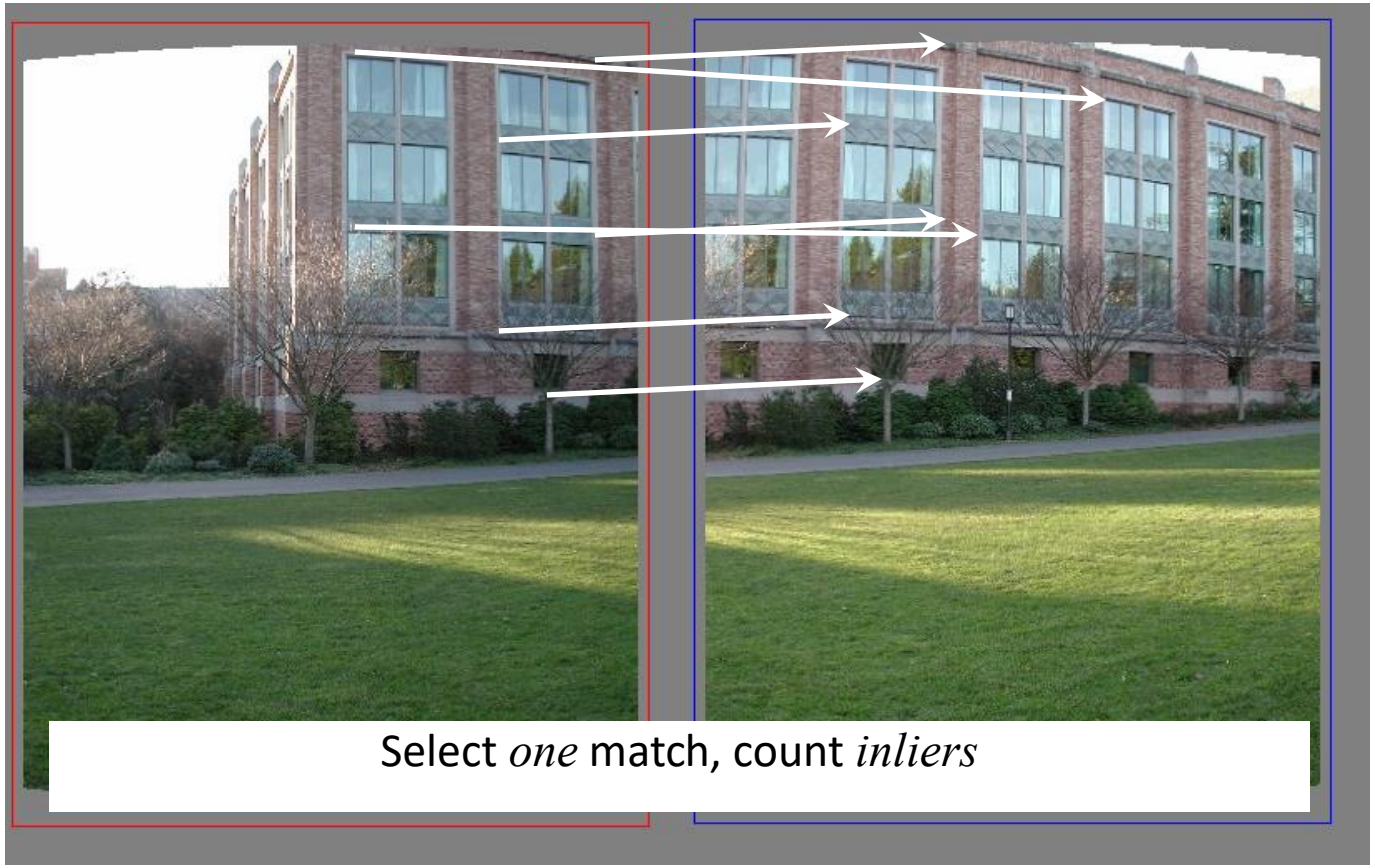
Yellow: correct matches.

(b) Matching many features--looking for a good homography

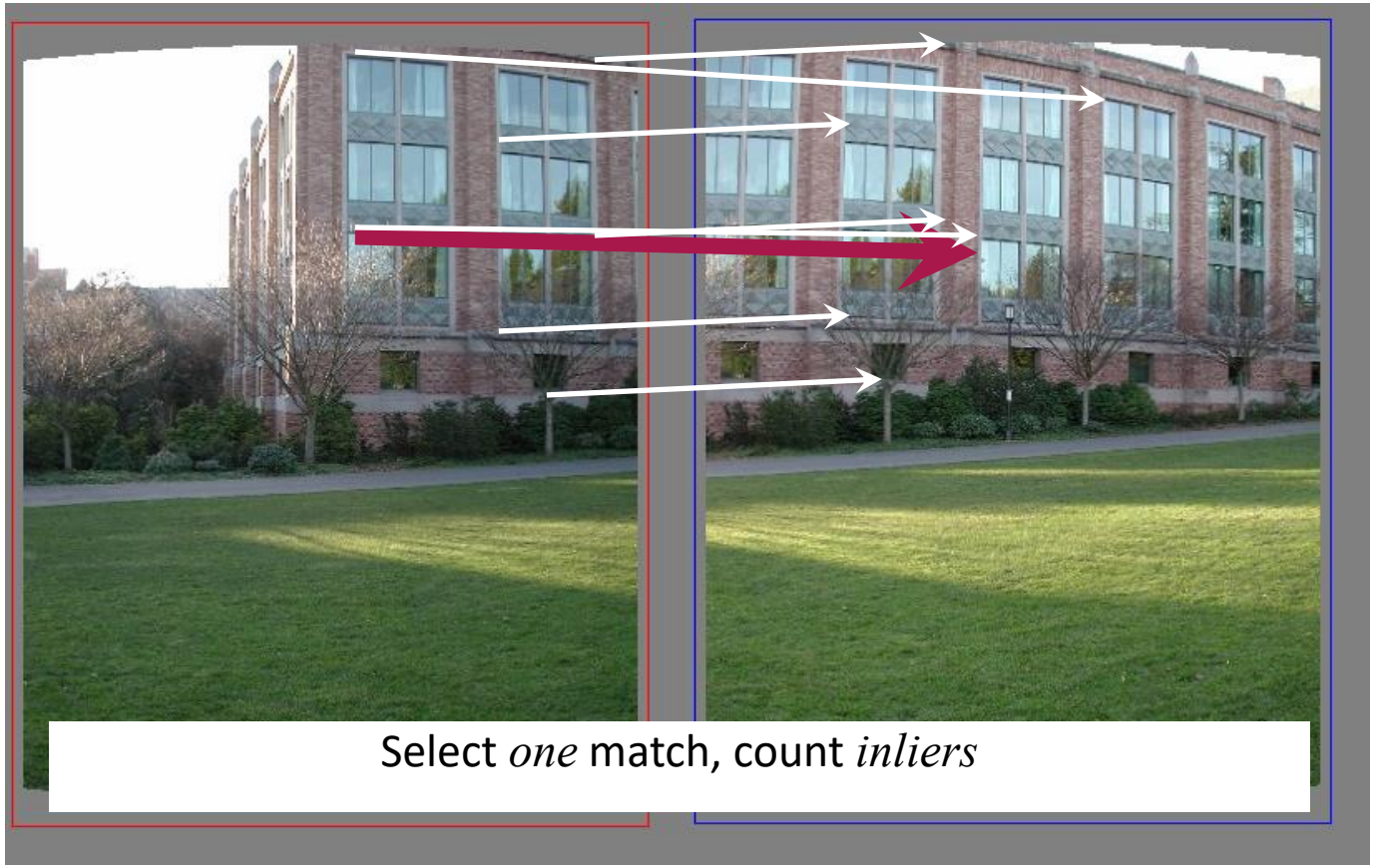
Simplified illustration with translation instead of homography

Note: at this point we don't know which ones are good/bad

RAndom SAmple Consensus

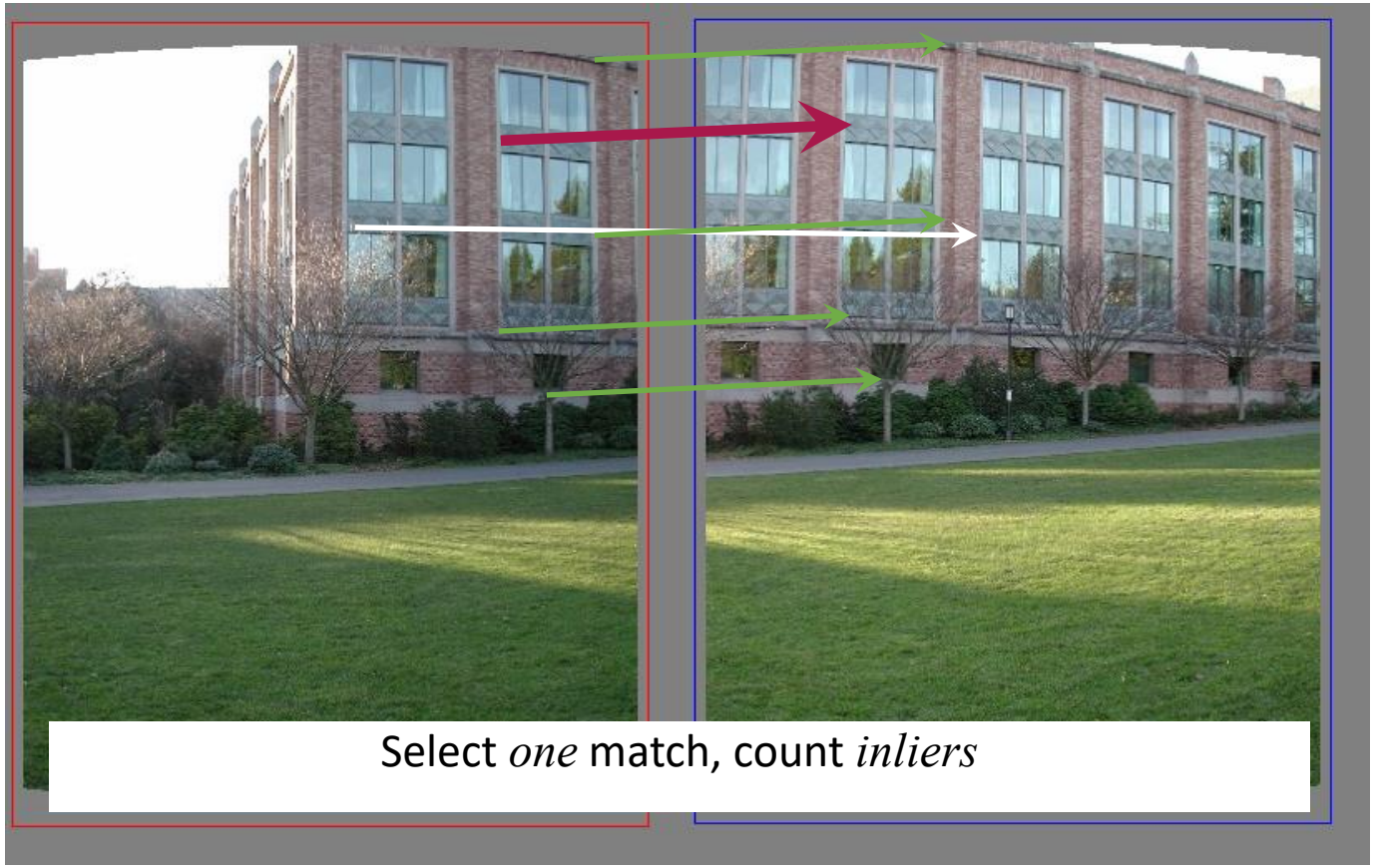


RAndom SAmple Consensus



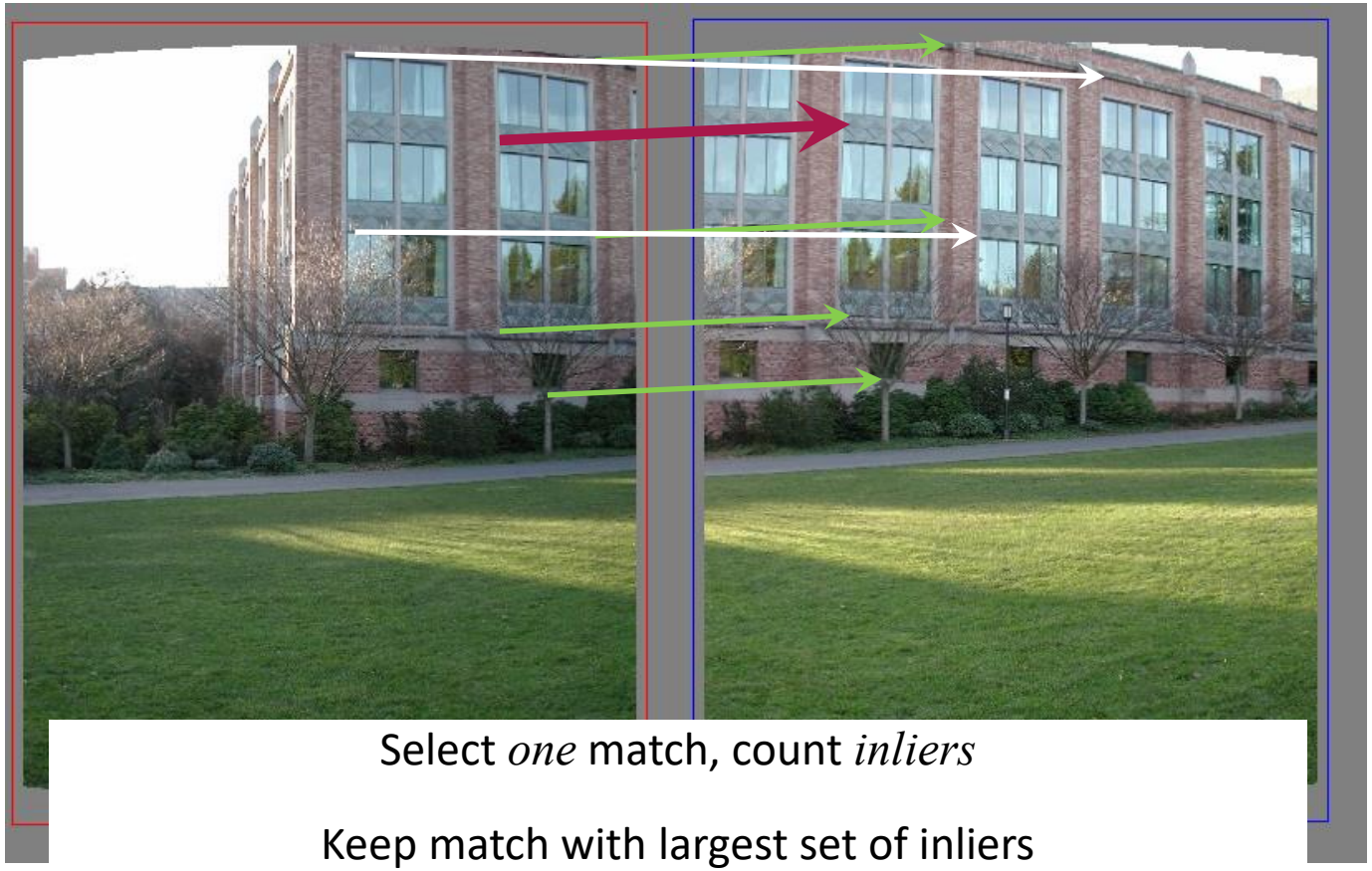
0 inliers

RAndom SAmple Consensus



4 inliers

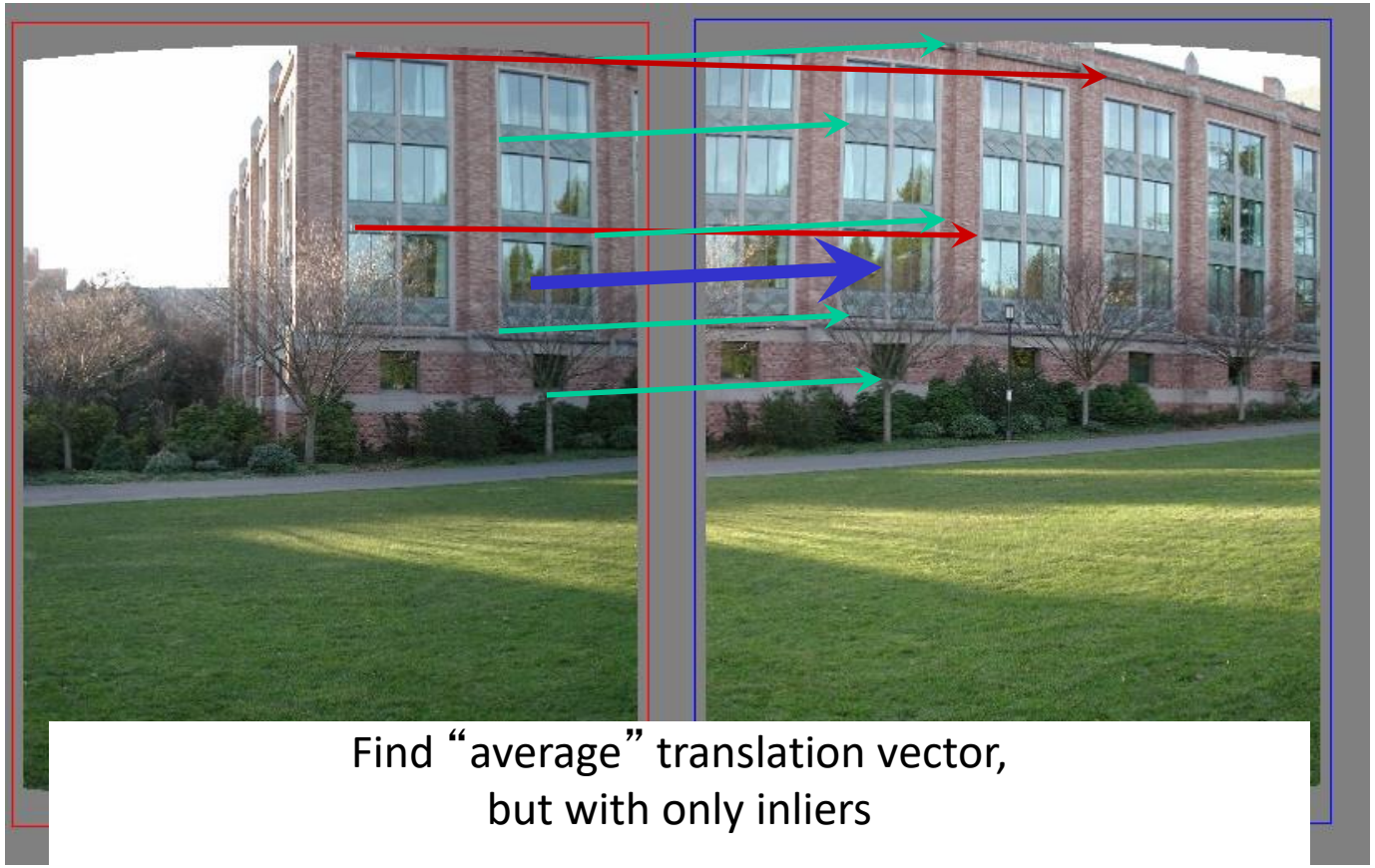
RAndom SAmple Consensus



Select *one* match, count *inliers*

Keep match with largest set of inliers

At the end: Least squares fit

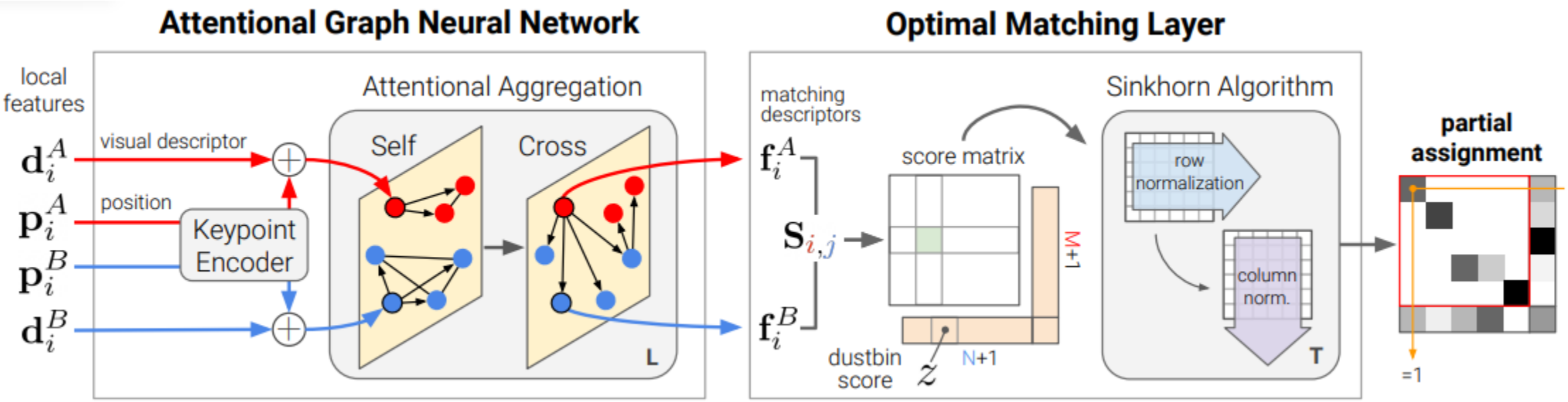


Find “average” translation vector,
but with only inliers

RANSAC Loop

- 1. Select four keypoint pairs (at random)
- 2. Compute homography H (exact)
- 3. Compute *inliers* where $SSD(p_i', \boldsymbol{H}\, p_{i)} < \varepsilon$
- 4. Keep largest set of inliers
- 5. Re-compute least-squares H estimate on all of the inliers

SuperGlue



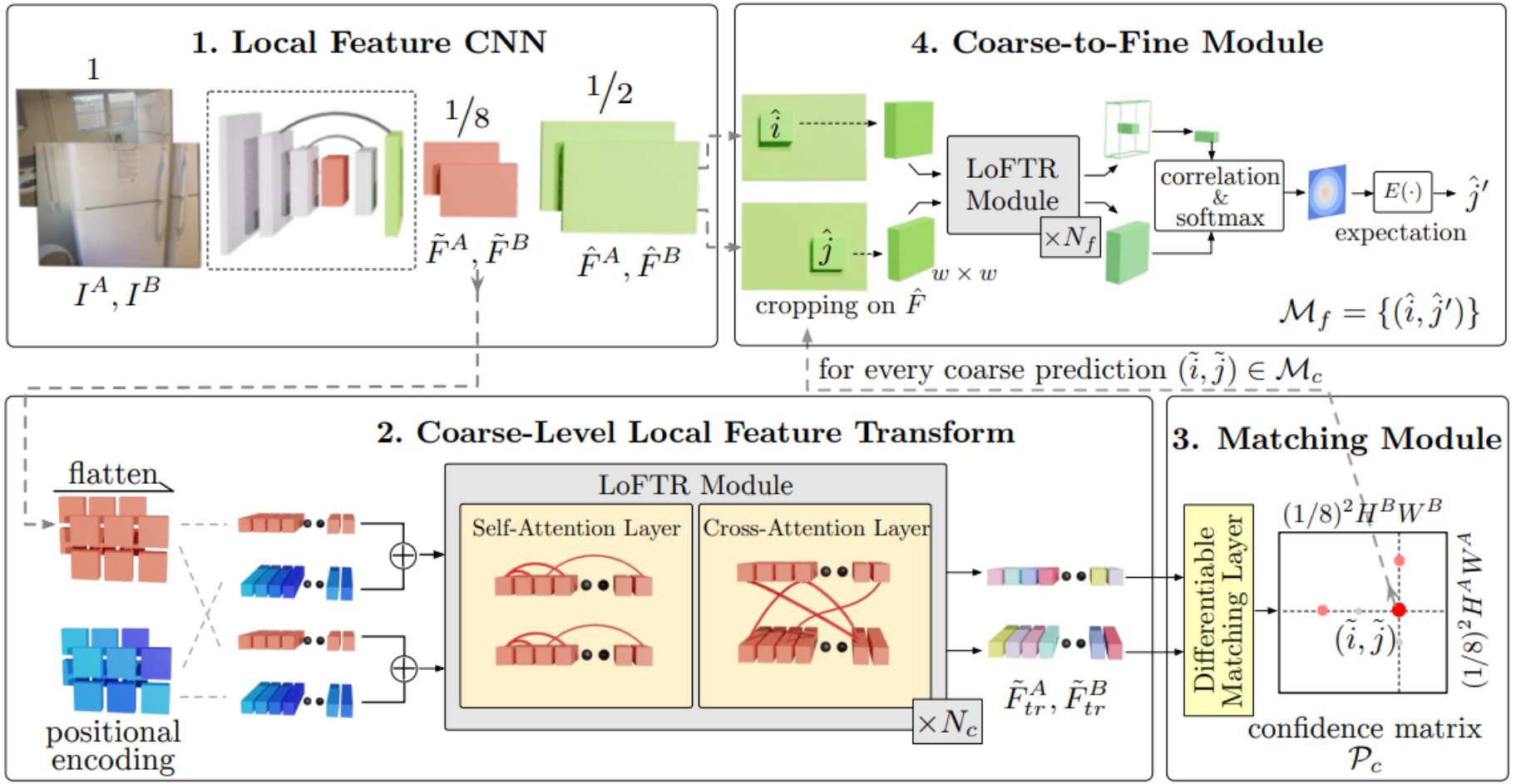
i) a keypoint can have at most a single correspondence in the other image;

ii) some keypoints will be unmatched due to occlusion and failure of the detector

P. -E. Sarlin, D. DeTone, T. Malisiewicz and A. Rabinovich, "SuperGlue: Learning Feature Matching With Graph Neural Networks," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 2020, pp. 4937-4946.

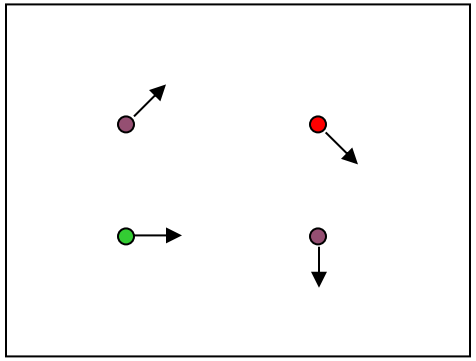
LoFTR: Detector-Free Local Feature Matching with Transformers

- SIFT, SuperPoint : Keypoint Detection, Description and Matching
- LoFTR: pixel-wise dense matching at coarse level and later refine at a finer level

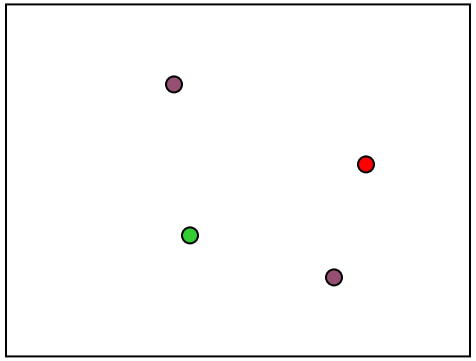


J. Sun, Z. Shen, Y. Wang, H. Bao and X. Zhou, "LoFTR: Detector-Free Local Feature Matching with Transformers," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, USA, 2021, pp. 8918-8927

Keypoint tracking



$I(x,y,t)$

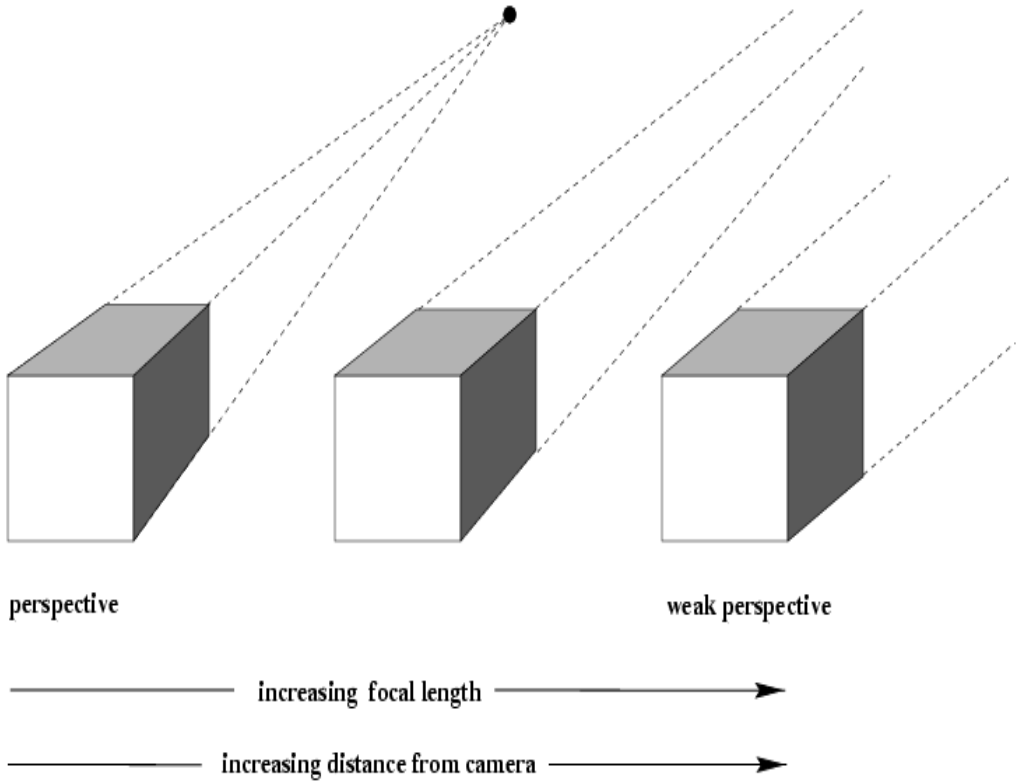


$I(x,y,t+1)$

- Given two subsequent frames, estimate the point translation
- Key assumptions of Lucas-Kanade Tracker
  - **Brightness constancy:** projection of the same point looks the same in every frame
  - **Small motion:** points do not move very far
  - **Spatial coherence:** points move like their neighbors

Structure from motion

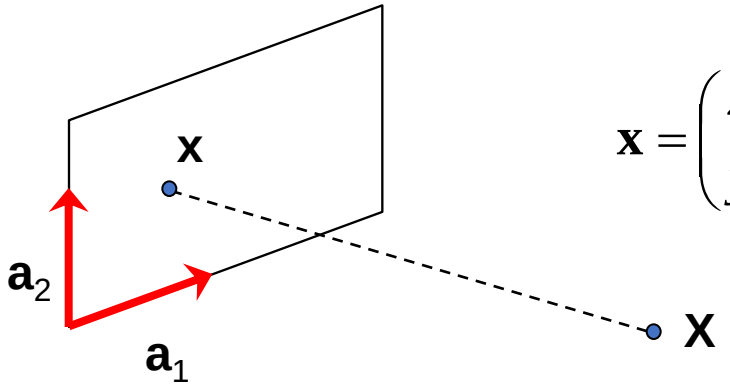
- Let's start with *affine cameras* (the math is easier)



center at infinity

Affine structure from motion

- Affine projection is a linear mapping + translation in inhomogeneous coordinates



$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \mathbf{AX} + \mathbf{t}$$

Projection of world origin

1. We are given corresponding 2D points (**x**) in several frames
2. We want to estimate the 3D points (**X**) and the affine parameters of each camera (**A**)

Step 1: Simplify by getting rid of **t**: shift to centroid of points for each camera

$$\mathbf{x} = \mathbf{A}\mathbf{X} + \mathbf{t} \qquad \hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik}$$

$$\mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik} = \mathbf{A}_i\mathbf{X}_j + \mathbf{t}_i - \frac{1}{n}\sum_{k=1}^{n}\left(\mathbf{A}_i\mathbf{X}_k + \mathbf{t}_i\right) = \mathbf{A}_i\left(\mathbf{X}_j - \frac{1}{n}\sum_{k=1}^{n}\mathbf{X}_k\right) = \mathbf{A}_i\hat{\mathbf{X}}_j$$

$$\hat{\mathbf{x}}_{ij} = \mathbf{A}_i\hat{\mathbf{X}}_j$$

2d normalized point (observed)

3d normalized point

Linear (affine) mapping

Suppose we know 3D points and affine camera parameters …

then, we can compute the observed 2d positions of each point

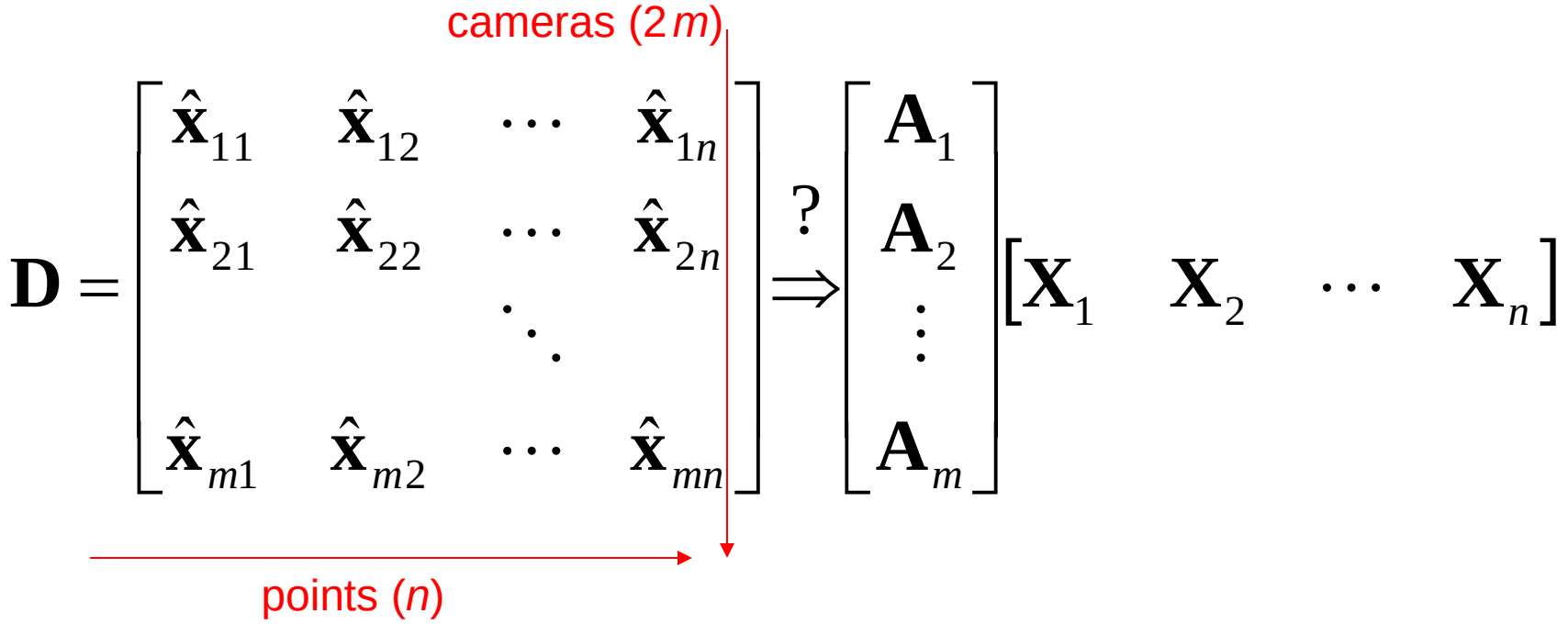
$$\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \dots & \mathbf{X}_n \end{bmatrix}$$

3D Points (3xn)

Camera Parameters (2mx3)

What if we instead observe corresponding 2d image points?

Can we recover the camera parameters and 3d points?



What rank is the matrix of 2D points?

Affine structure from motion

- Let's create a $2m \times n$ data (measurement) matrix:

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}$$

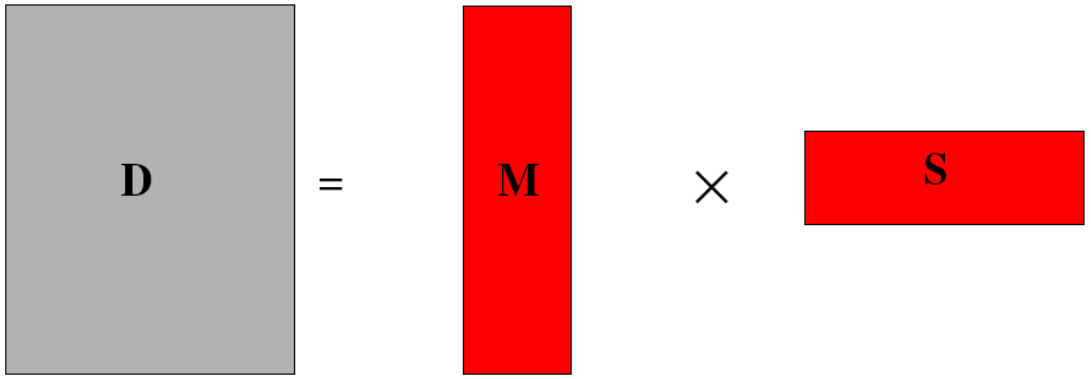
cameras ($2m \times 3$)

points ($3 \times n$)

The measurement matrix $\mathbf{D} = \mathbf{MS}$ must have rank 3!

C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

Affine ambiguity

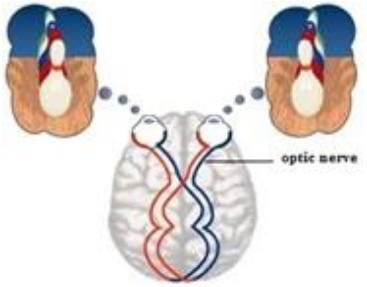


- The decomposition is not unique. We get the same **D** by using any 3×3 matrix **C** and applying the transformations **M → MC, S →C^{-1}S**
- That is because we have only an affine transformation and we have not enforced any Euclidean constraints (like forcing the image axes to be perpendicular, for example)

Stereo Imaging

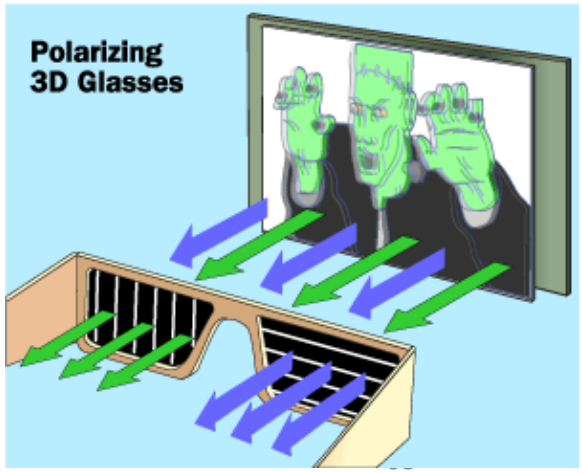
3D video principle

- Human eyes are two optical lens, two images are formed onto the human’s retina through the left and right lens. Even the two images are originated from same object, but they are viewing from different angles, which results in slightly different between image. The human brain can judge the distance of the object through the information, this is reason we can see 3D object, on nature science call "binocular disparity".



Stereoscopic TV using Polarized glasses

- Image quality is not good, if separate an image by color. Preferred method uses polarized lenses because they allow color viewing.
- Left and right images are projected superimposed onto the same screen through orthogonal polarizing filters (usually at 45 and 135 degrees).
- The viewer wears linearly polarized eyeglasses which also contain a pair of orthogonal polarizing filters oriented the same as the projector.
- As each filter only passes light which is similarly polarized and blocks the orthogonally polarized light, each eye only sees one of the projected images.

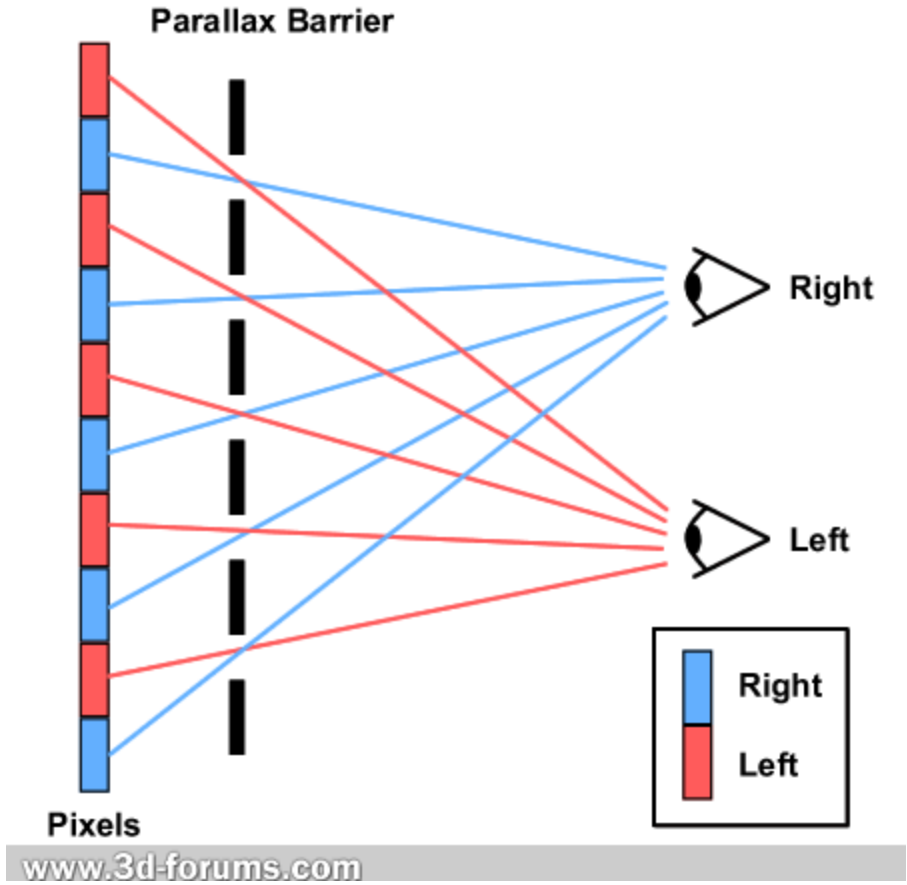


Stereoscopic TV using Shutter glasses

- Shutter glasses can be used to provide temporal filtering, which creates a stereoscopic effect by presenting different perspectives to each eye through alternate frame sequencing.
- The display alternates between left and right eye views whilst the glasses blank each eye alternately in synchronization with the screen.
- The shutter glasses are typically based on liquid crystal materials that have the property to become dark when voltage is applied, but are otherwise transparent.
- Unlike colored glasses or polarized glasses, these are active devices that require synchronization with the display through cable, wireless or infra-red communication.

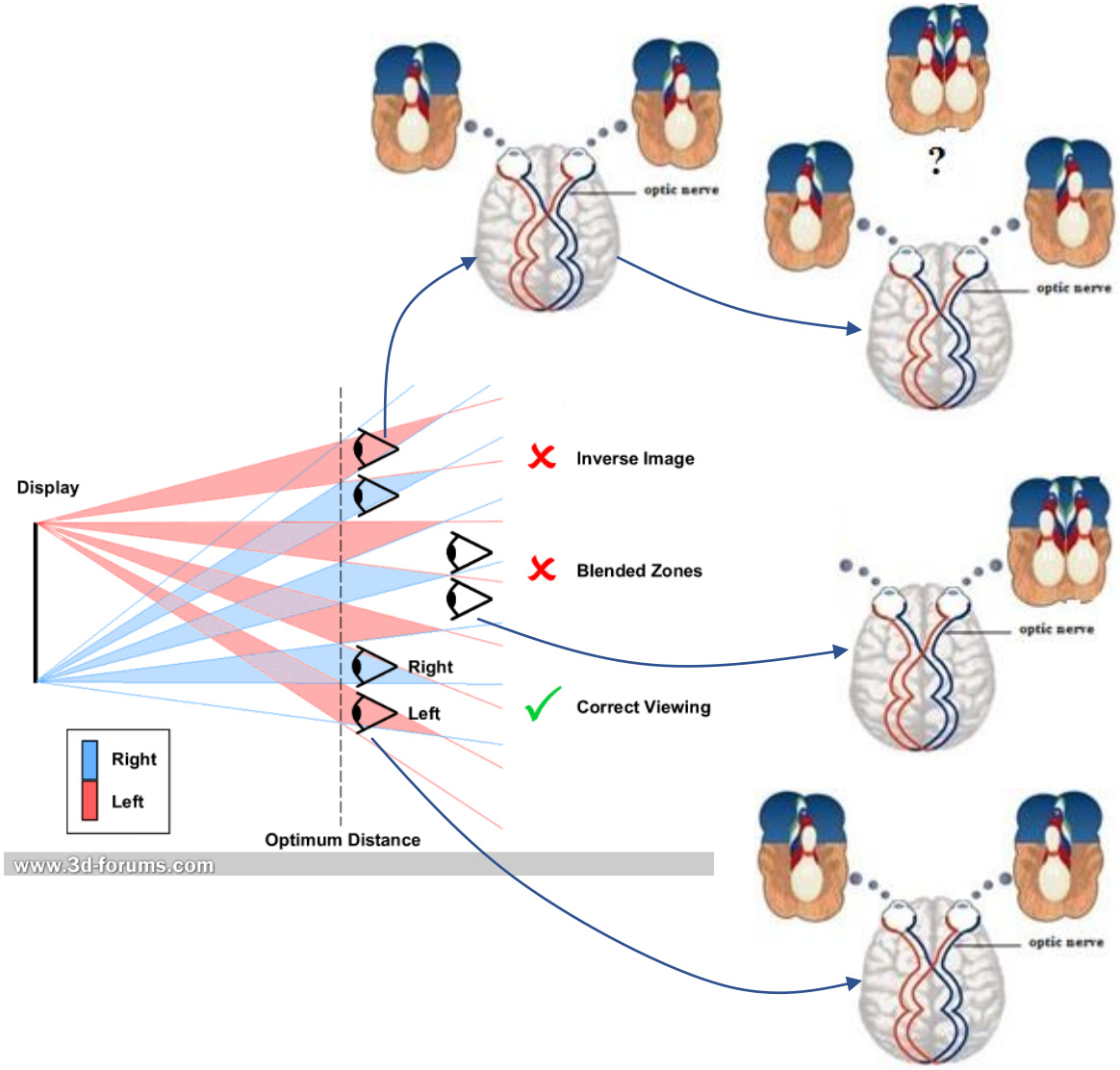
Auto-stereoscopic systems

- Auto-stereoscopic displays fool the brain so that a 2D medium can display a 3D image by providing a stereo parallax view for the user.
- This means that each eye sees a different image, having been calculated to appear from two eye positions.
- In the parallax barrier a mask is placed over the LCD display which directs light from alternate pixel columns to each eye.

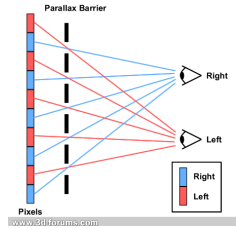


Auto-stereoscopic systems

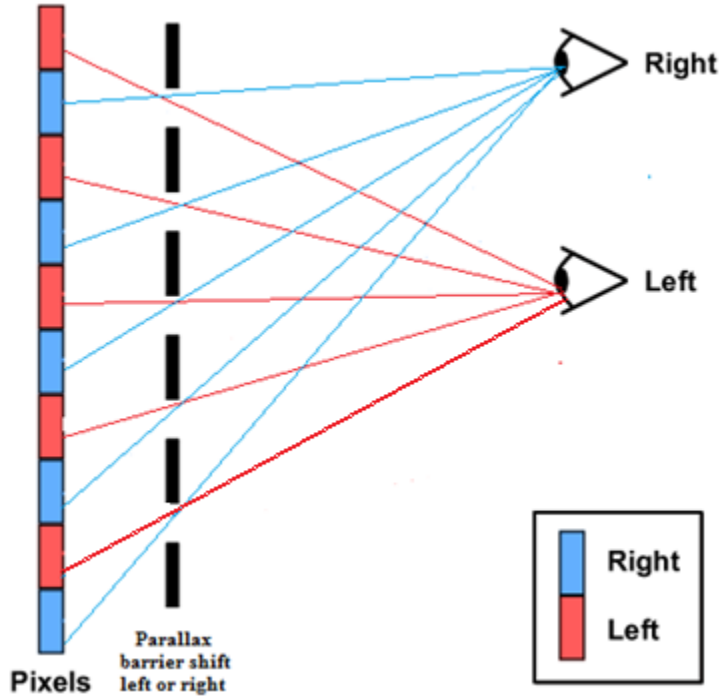
- The viewer must be at the correct viewing position.
- The user to the right (“Inverse Image”) is aligned incorrectly and the image formed at each eye is the wrong way round.
- The “blended zone” user is standing too far away from the optimal distance and is seeing both images forming in each eye, causing a blurred and confusing image.
- The user in the left is standing in the correct position, with each eye located within the correct viewing zone.

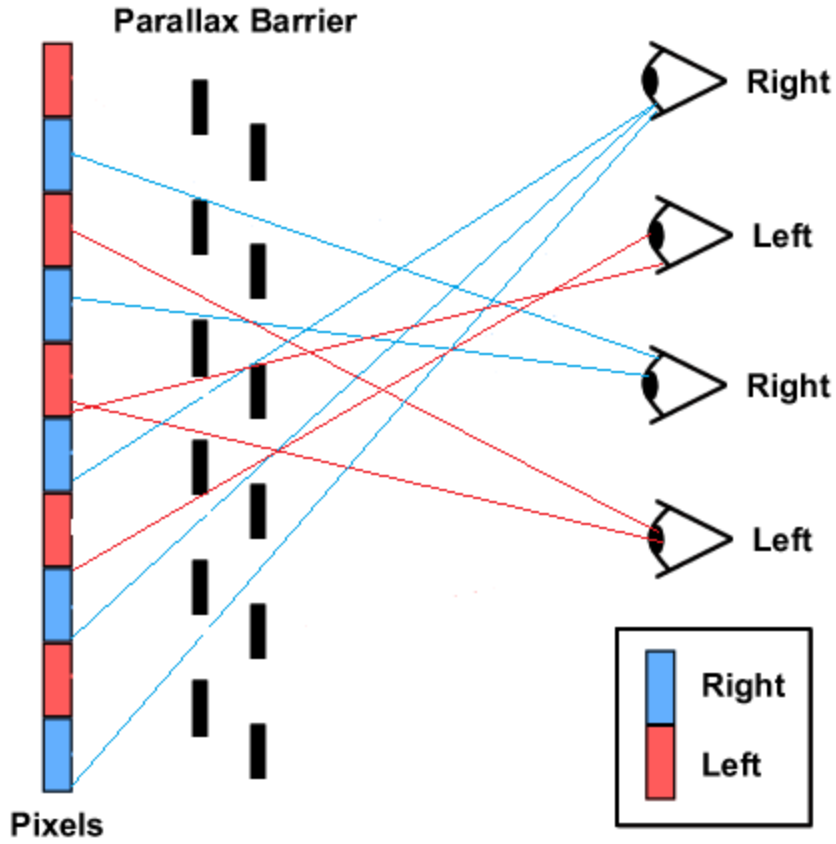


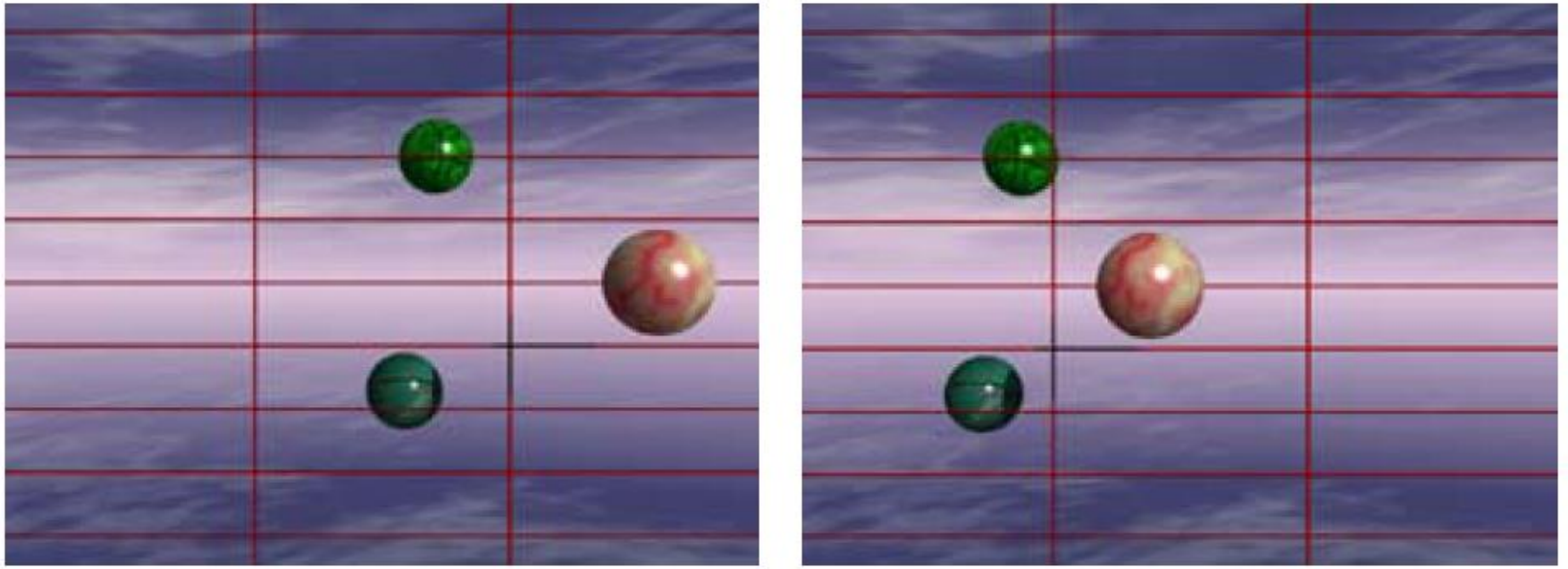
Auto-stereoscopic systems



- Viewer position shift?
- Multi-viewers?



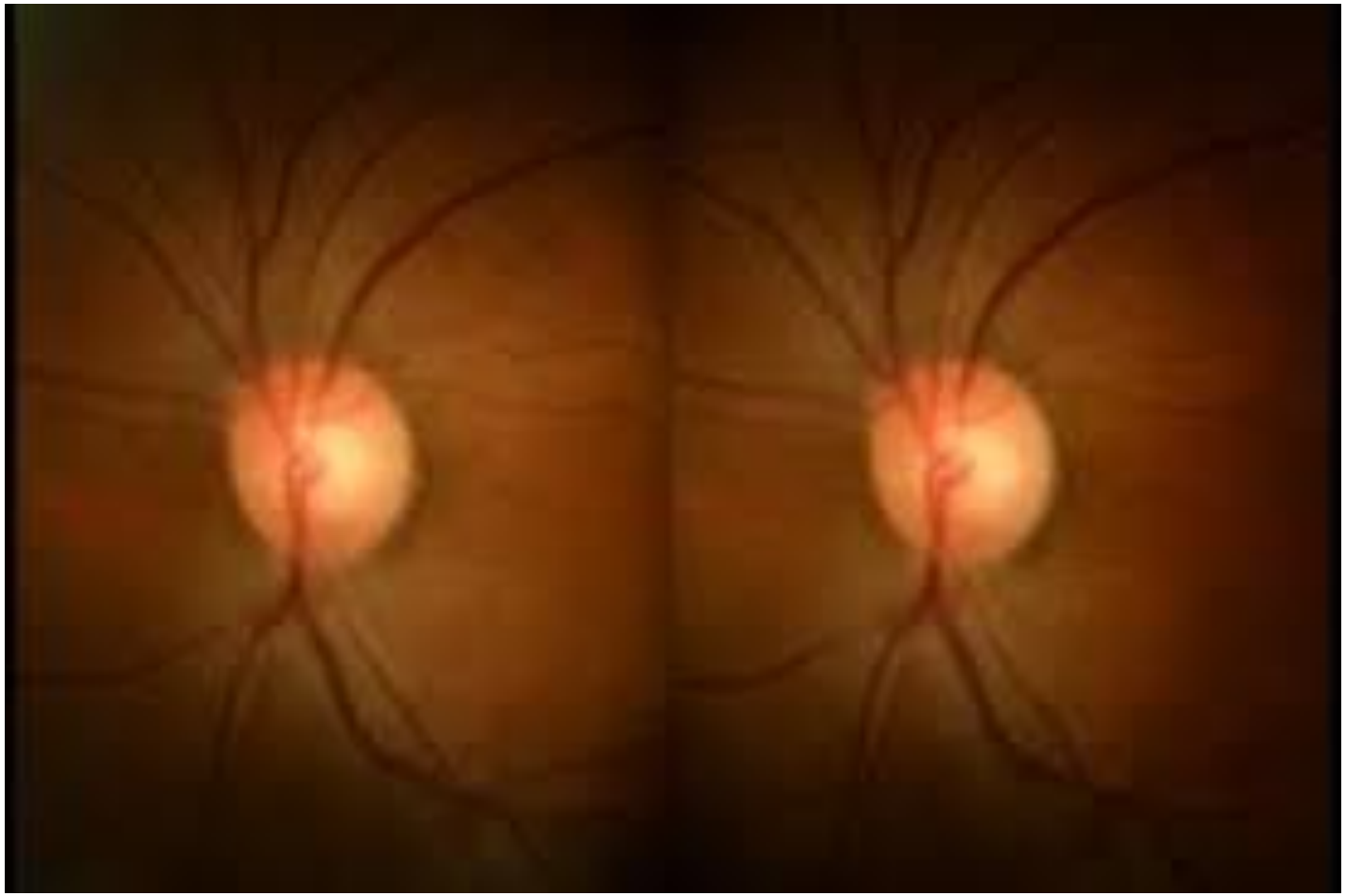




Try to look at the left and right images using your left and right eyes separately while try to merge the two images into one.
Can you tell which ball is closer?

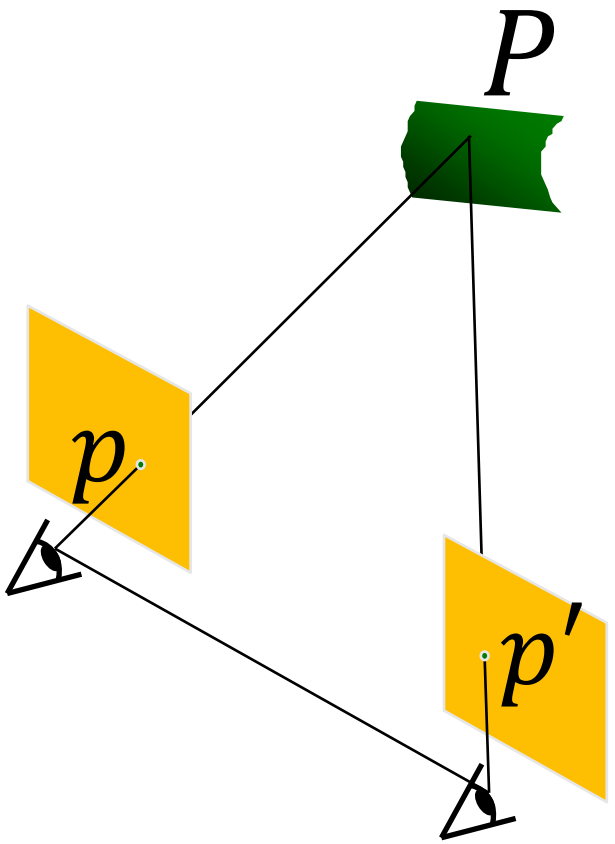
Stereo Color Fundus

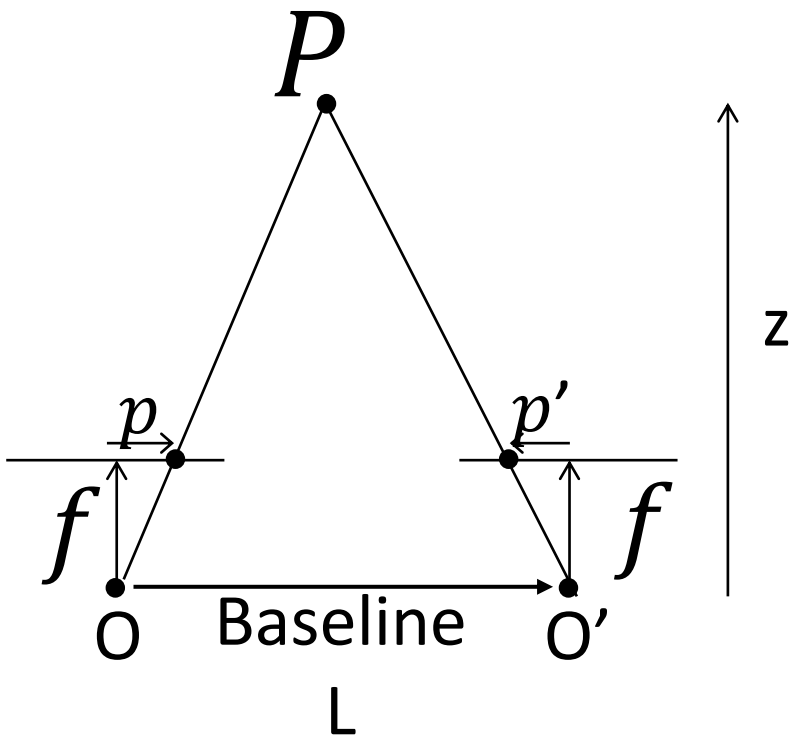
Application: glaucoma detection



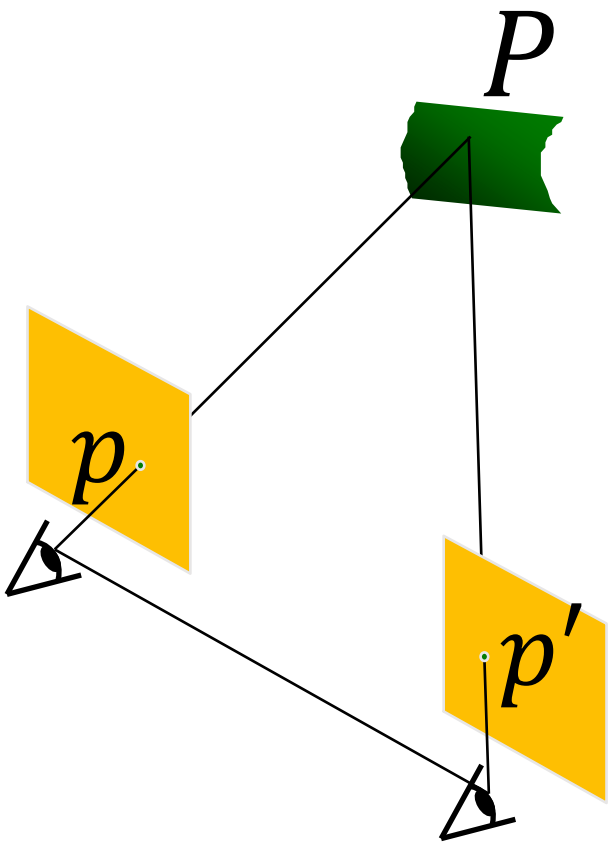
Depth from Stereo

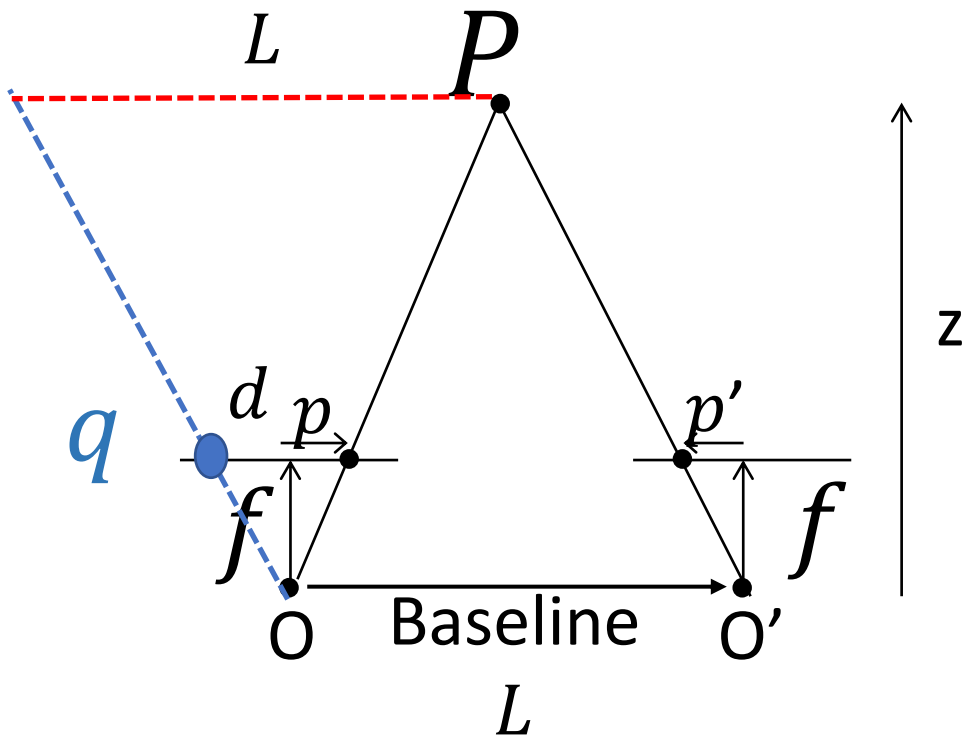
- Goal: recover depth by finding image coordinate p' that corresponds to p





Depth from Stereo





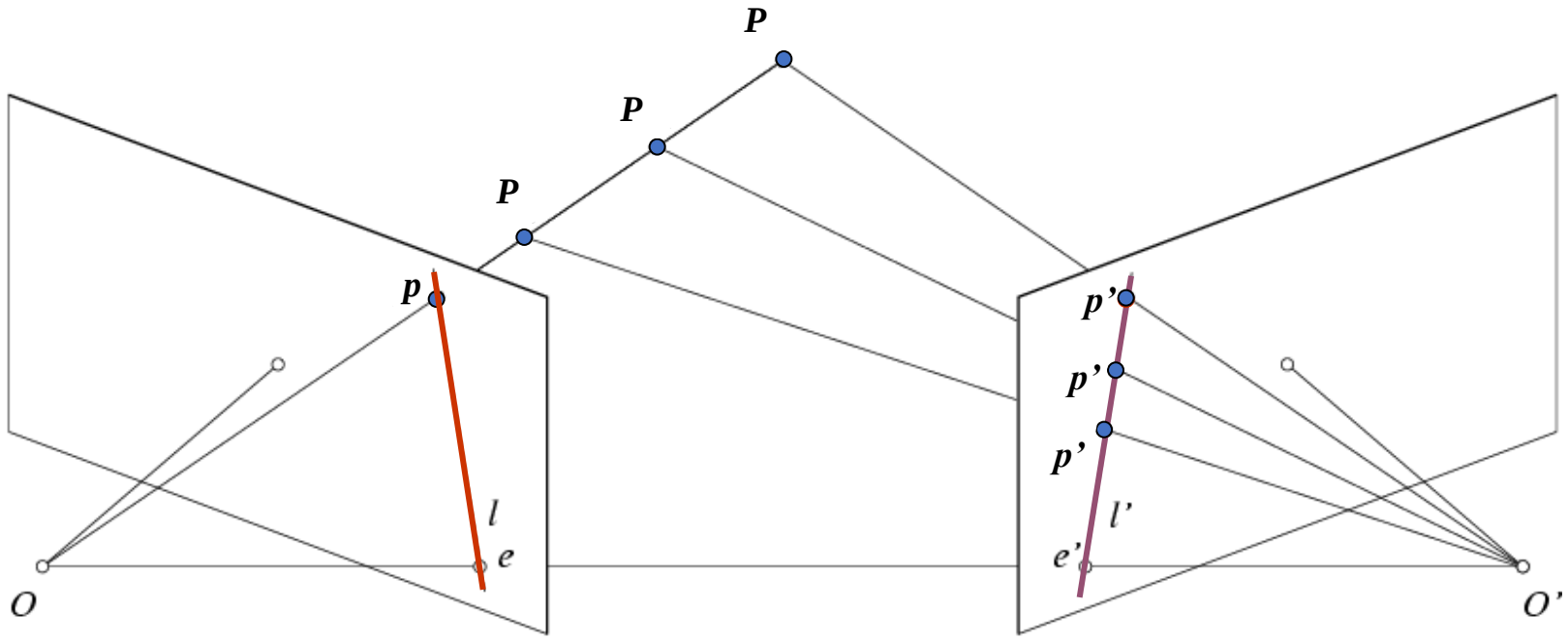
$$\frac{f}{d} = \frac{z}{L} \Rightarrow z = \frac{f * L}{d}$$

d is called the disparity.

Depth from Stereo

- **Correspondence geometry:** Given an image point p in the first view, how does this constrain the position of the corresponding point p' in the second image?
- **Camera geometry (motion):** Given a set of corresponding image points $\{p_i \leftrightarrow p'_i\}$, i=1,…,n, what are the cameras O and O' for the two views? Or what is the geometric transformation between the views?
- **Scene geometry (structure):** Given corresponding image points $p_i \leftrightarrow p'_i$ and cameras O, O', what is the position of the point P in space?

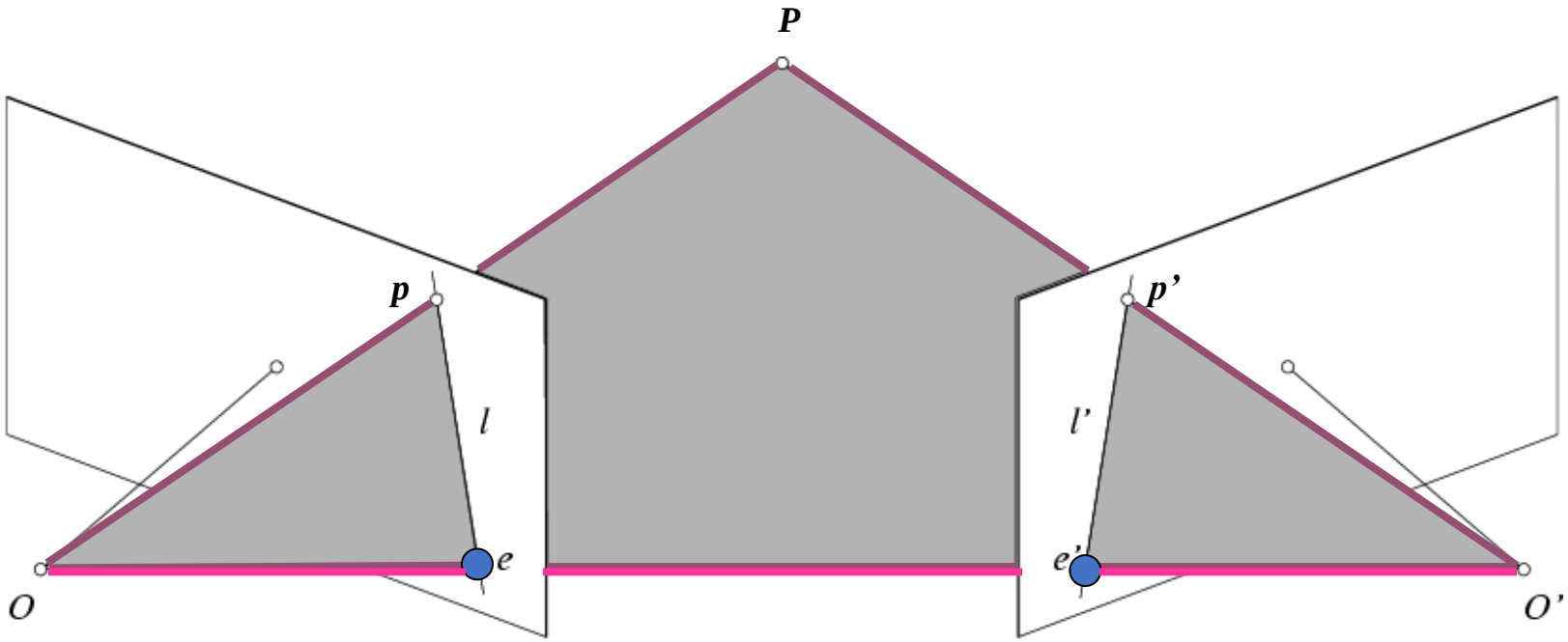
Epipolar Constraint



Potential matches for *p* have to lie on the corresponding line *l'*.
We call line *l'* as *epipolar line for p*

Similarly, line l as epipolar line for p'

Epipolar geometry: notation



- **Baseline** – line connecting the two camera centers
- **Epipolar Plane – the points P, O & O' form the epipolar plane.**
- **Epipoles**

= intersections of baseline with image planes

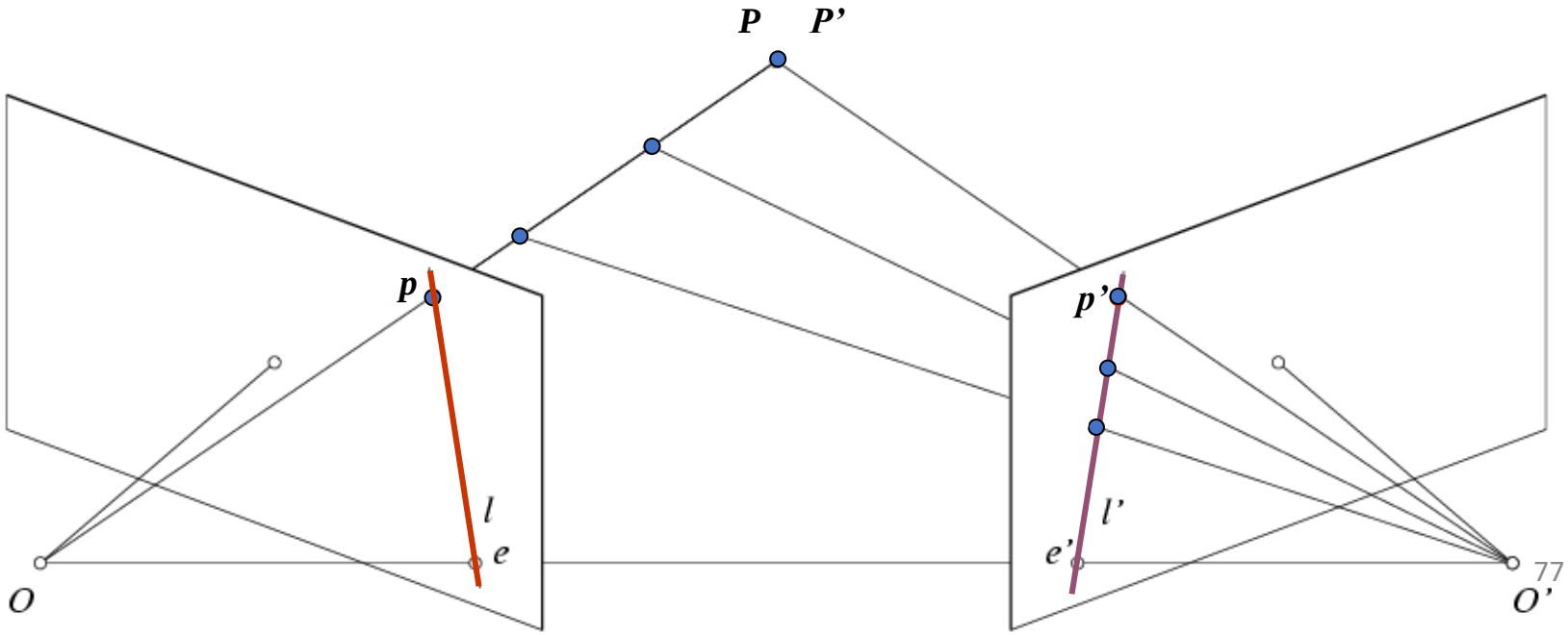
= projections of the other camera center

Epipolar geometry

O and O’: $\boldsymbol{R}$ and $\boldsymbol{t}$

What is the relationship between $\boldsymbol{P}$ and $\boldsymbol{P}$’?

$$\boldsymbol{P}' = \boldsymbol{R}(\boldsymbol{P} - \boldsymbol{t})$$



Essential Matrix

$$\boldsymbol{P' = R(P - t)}$$

$$\boldsymbol{(P')^T = \left(R(P - t)\right)^T = (P - t)^T R^T}$$

$$\boldsymbol{(P')^T R = (P - t)^T R^T R = (P - t)^T}$$

$$\boldsymbol{(P')^T R\,(t \times P) = (P - t)^T\,(t \times P)}$$

$$\boldsymbol{(P')^T R\,(t \times P) = 0}$$

$$\boldsymbol{(P')^T R t_\times P = 0}$$

$$a \times b = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} b = \quad [a_\times] b$$

$$\boldsymbol{E = R t_\times}$$

$$\boldsymbol{(P')^T E P = 0}$$

Fundamental Matrix

- Suppose the cameras are uncalibrated. Then the matrices K_1 and K_2 containing the internal parameters of the two cameras are needed to transform the normalized coordinates into pixel coordinates:

$$p = K_1 \boldsymbol{P} \quad \text{and} \quad p' = K_2 \boldsymbol{P'}$$

We have $(p')^T (K_2^{-1})^T \mathrm{E}\, K_1^{-1}\, \mathrm{p} = 0$

$$F = (K_2^{-1})^T \mathrm{E}\, K_1^{-1}$$

$$(p')^T F p = 0$$

Estimating the Fundamental Matrix

$$(p')^T F p = 0 \quad => \quad (p_i')^T F\, p_i = 0$$

$$[x_i', y_i', 1] \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = 0$$

$$x_i' x_i\, f_{11} + x_i' y_i\, f_{12} + x_i'\, f_{13} + x_i y_i'\, f_{21} + y_i' y_i f_{22} + y_i'\, f_{23} + x_i\, f_{31} + y_i\, f_{32} + f_{33} = 0$$

$$\mathrm{A}\boldsymbol{f} = \begin{bmatrix} x_1' x_1 & y_1 x_1' & x_1' & x_1 y_1' & y_1' y_1 & y_1' & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n' x_n & y_n x_n' & x_n' & x_n y_n' & y_n' y_n & y_n' & x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ \vdots \\ f_{33} \end{bmatrix} = \mathbf{0}$$

Estimating the Fundamental Matrix

$$\mathrm{A}\boldsymbol{f} = 0$$

Total least square

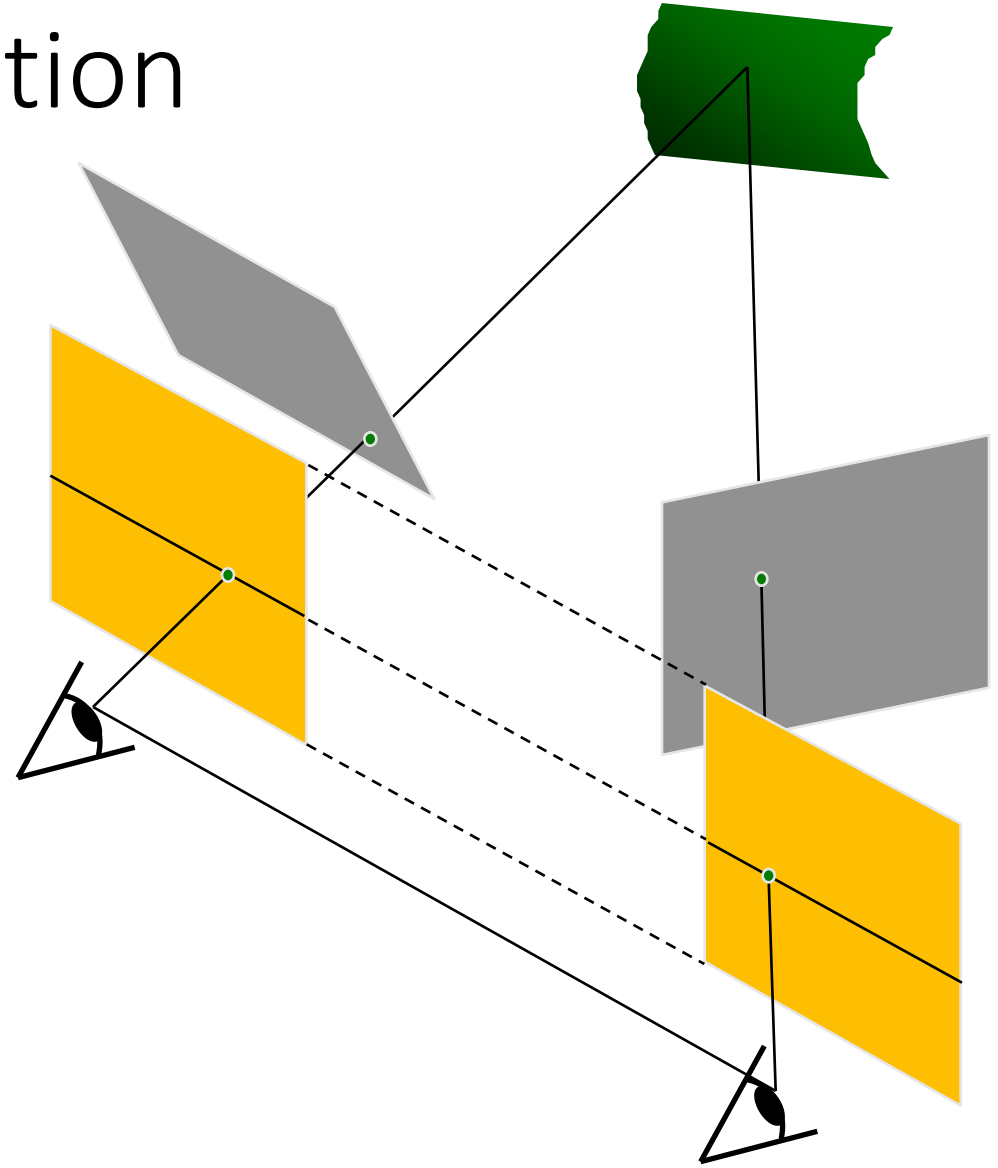
$$\min \|\mathrm{A}\boldsymbol{f}\|_2$$

Subject $\|\boldsymbol{f}\|_2=1$

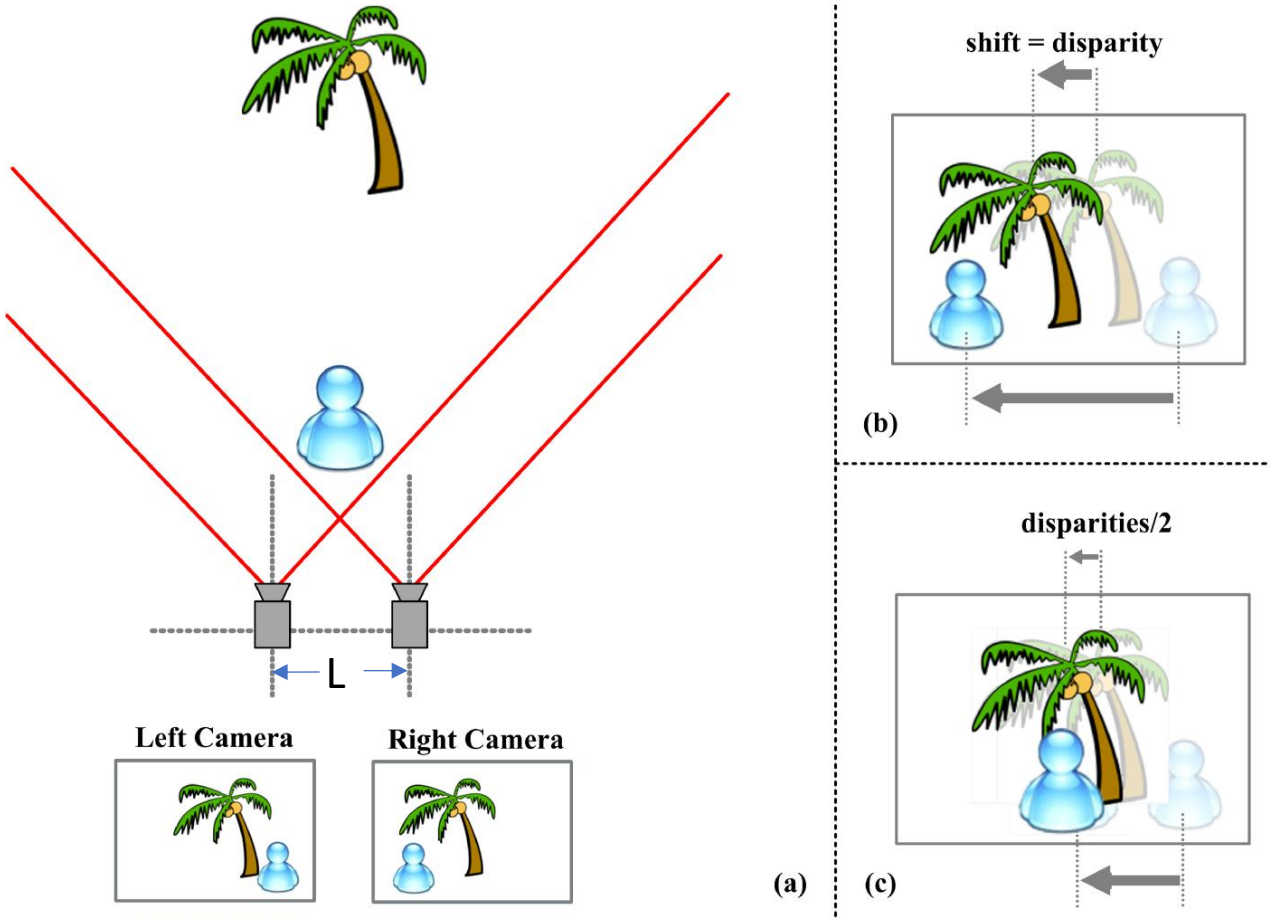
How to solve?

Stereo image rectification

- Reproject image planes onto a common plane parallel to the line between camera centers
- Pixel motion is horizontal after this transformation
- Two homographies (3x3 transform), one for each input image reprojection



Stereo Matching



$$depth\ z = \frac{f * L}{disparity}$$

f denotes the focal length

Basic stereo algorithm

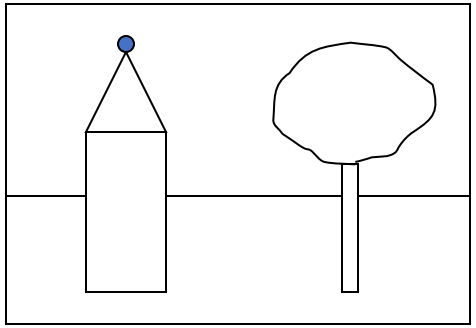
For each epipolar line

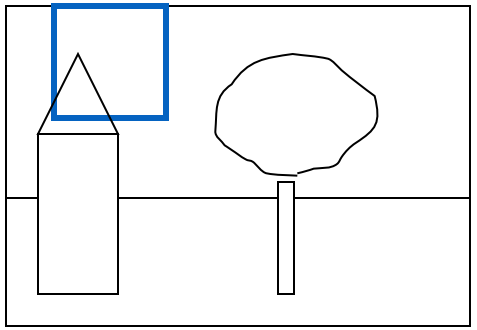
For each pixel in the left image,

- compare with every pixel on same epipolar line in right image
- pick pixel with minimum match cost

Major matching methods

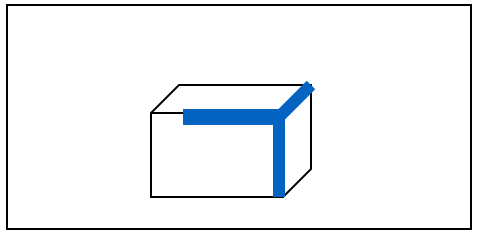
1. Cross correlation using small windows.

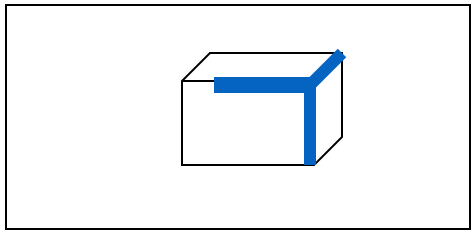




dense

2. Symbolic feature matching, usually using segments/corners.





sparse

3. Use interest operators, ie. SIFT.

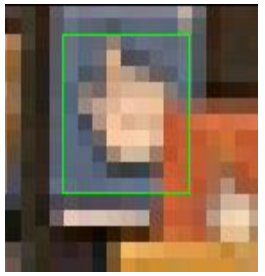
sparse

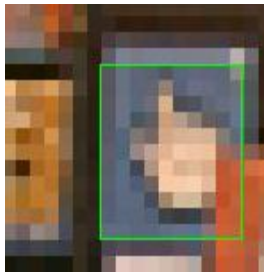
Image registration

- How do we determine correspondences?
  - *block matching* or *SSD* (sum squared differences)

$$E(x, y; d) = \sum_{(x',y')\in N(x,y)} [I_L(x'+d, y') - I_R(x', y')]^2$$

d is the *disparity* (horizontal motion)





- How big should the neighborhood be?

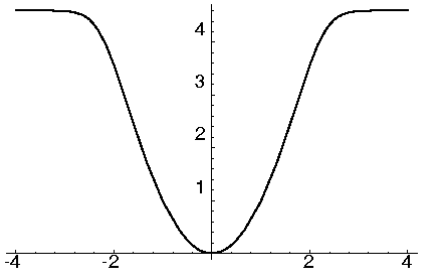
Stereo matching framework

1. For every disparity, compute *raw* matching costs

$$E_0(x, y; d) = \rho(I_L(x' + d, y') - I_R(x', y'))$$

   occlusions, other outliers

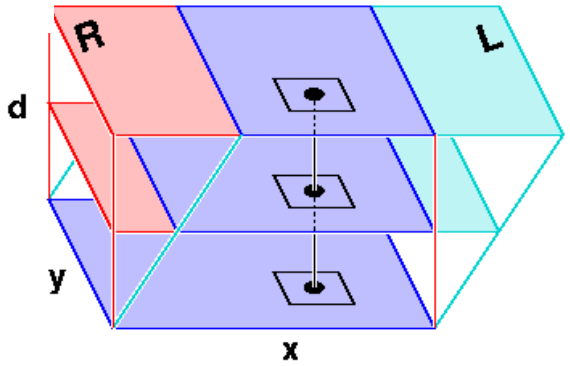
- Can also use alternative match criteria



Stereo matching framework

2. Aggregate costs spatially

$$E(x, y; d) = \sum_{(x',y') \in N(x,y)} E_0(x', y', d)$$



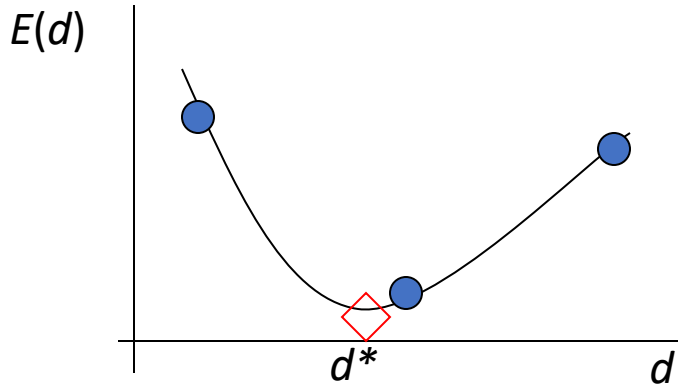
- Here, we are using a *box filter* (efficient moving average implementation)
- Can also use weighted average, [non-linear] diffusion…

Stereo matching framework

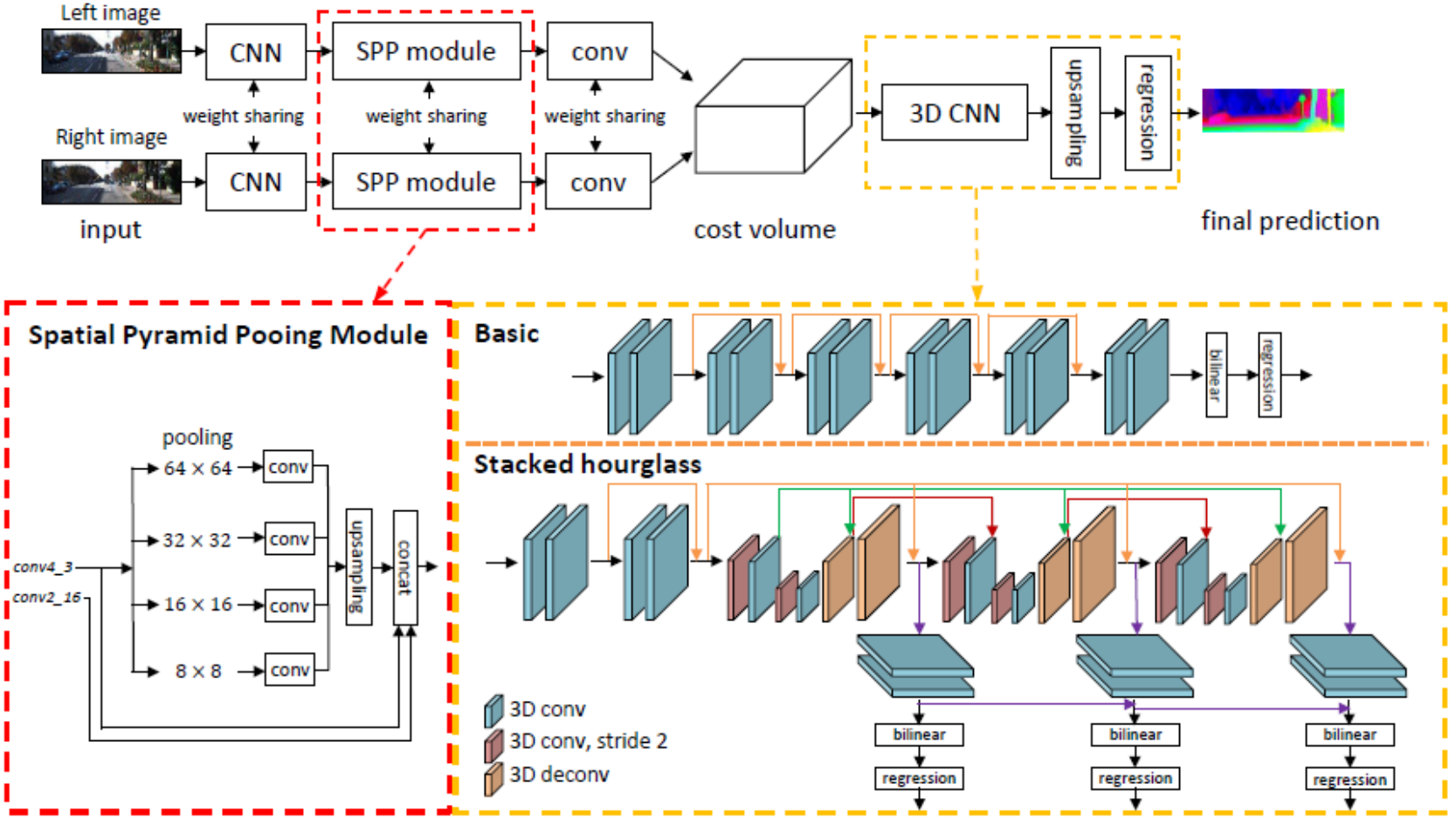
3. Choose winning disparity at each pixel

$$d(x, y) = \arg\min_{d} E(x, y; d)$$

4. Interpolate to *sub-pixel* accuracy



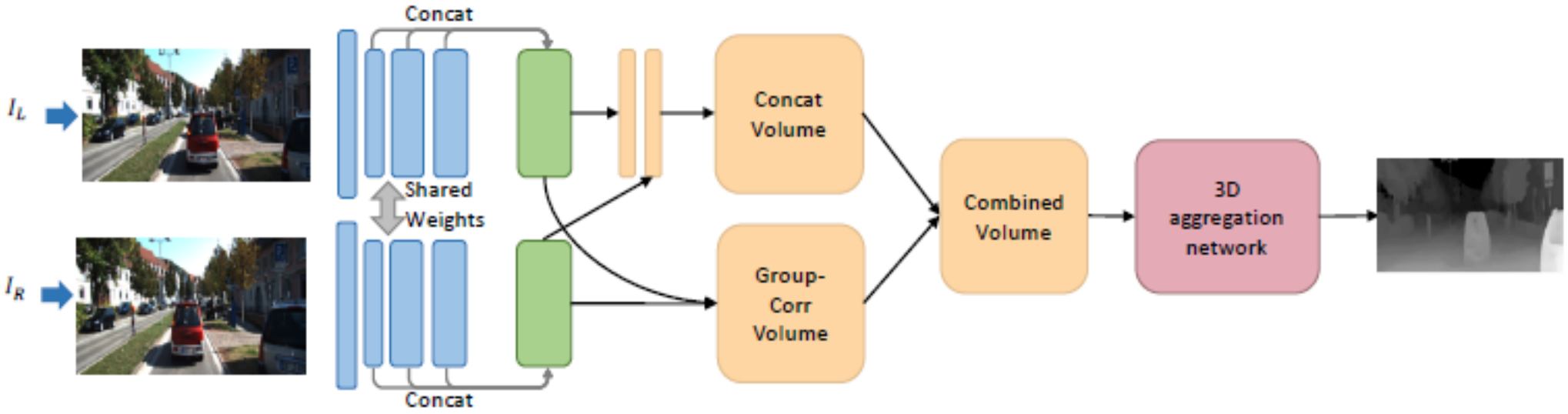
PSMNet: Pyramid Stereo Matching Network



Read this paper

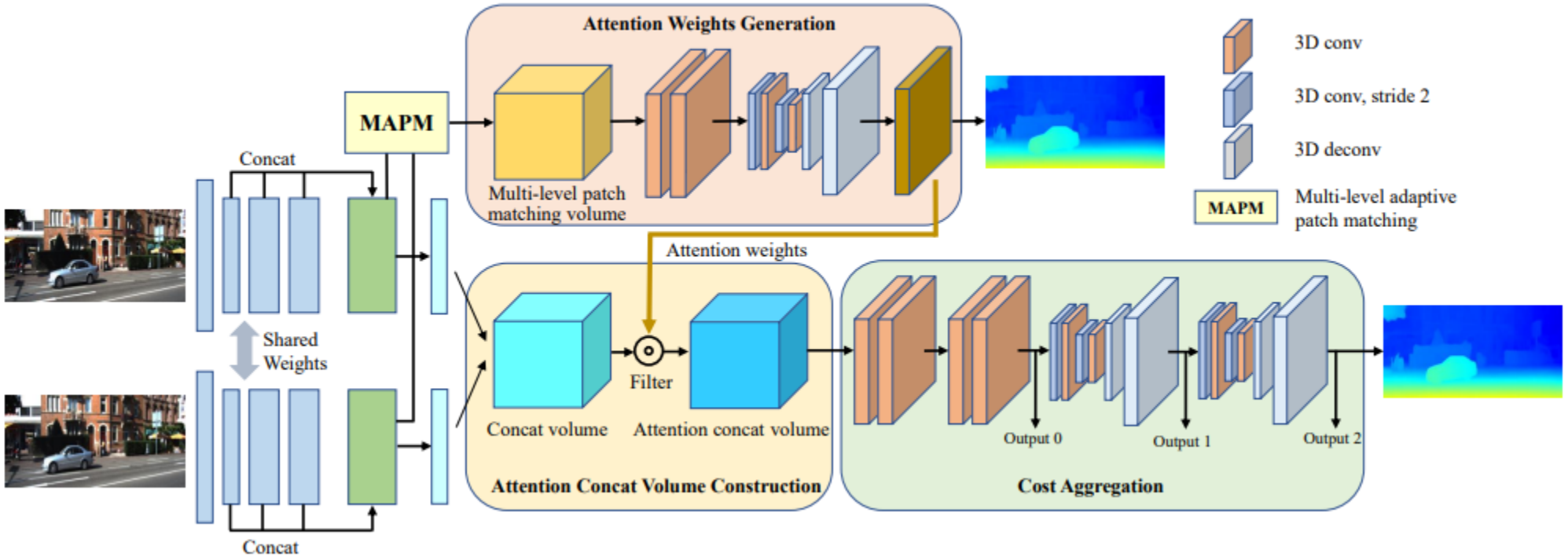
J. -R. Chang and Y. -S. Chen, "Pyramid Stereo Matching Network," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 5410-5418, doi: 10.1109/CVPR.2018.00567. https://github.com/JiaRenChang/PSMNet

GwcNet: Group-wise Correlation Stereo Network



X. Guo, K. Yang, W. Yang, X. Wang and H. Li, "Group-Wise Correlation Stereo Network," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 3268-3277

ACVNet: Attention Concatenation Volume for Accurate and Efficient Stereo Matching



G. Xu, et al., "Attention Concatenation Volume for Accurate and Efficient Stereo Matching," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, LA, USA, 2022, pp. 12971-12980

Deep Learning for Stereo Matching

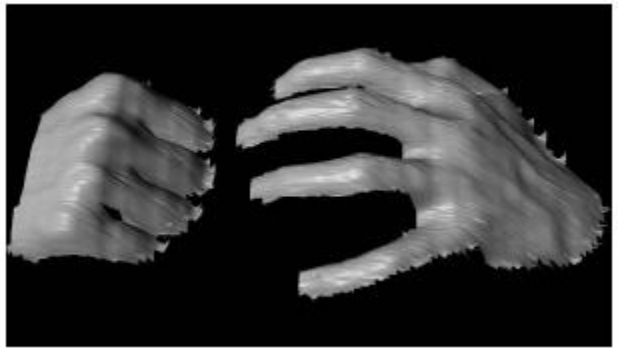
What the common steps:

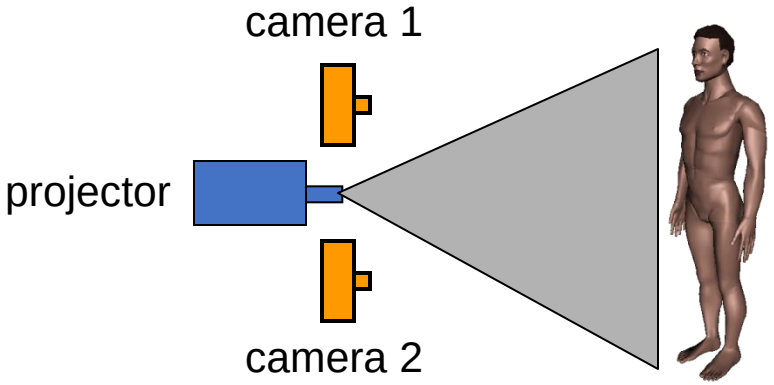
- feature extraction
- cost volume construction
- cost aggregation
- disparity regression

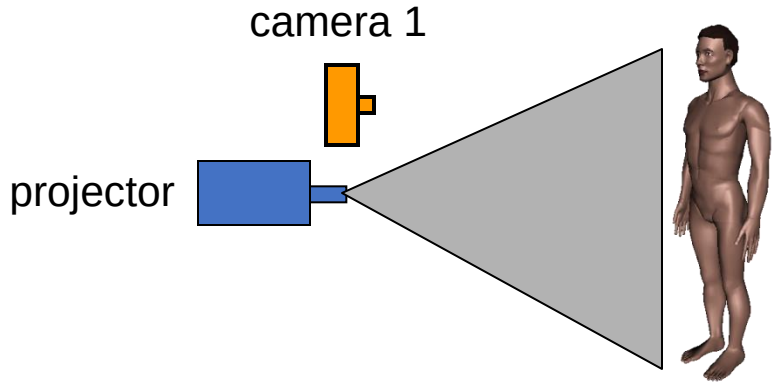
Question: What are the objectives in each step?

Active stereo with structured light









- Project “structured” light patterns onto the object
  - simplifies the correspondence problem

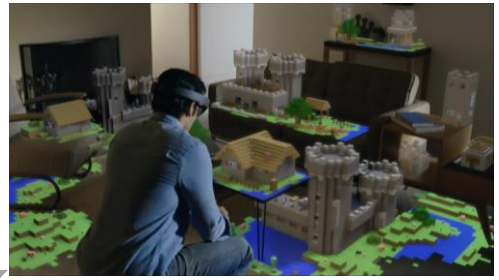
3D Meet Deep Learning

Broad Applications of 3D data

Robotics

Broad Applications of 3D data

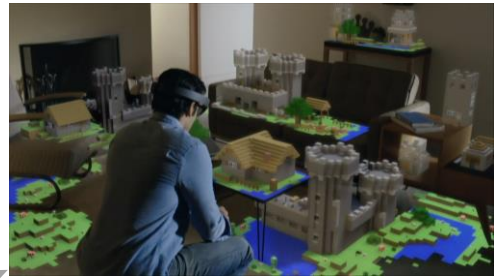
Robotics



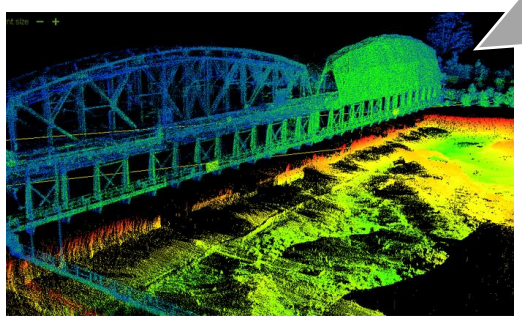
Augmented Reality

Broad Applications of 3D data

Robotics



Augmented Reality

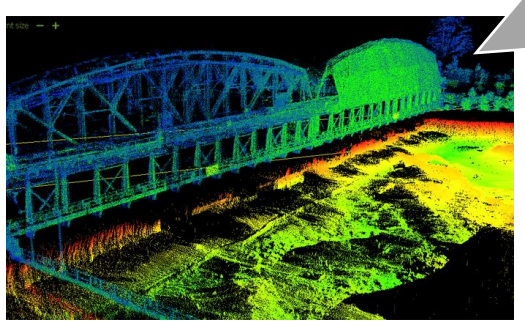


Autonomous driving

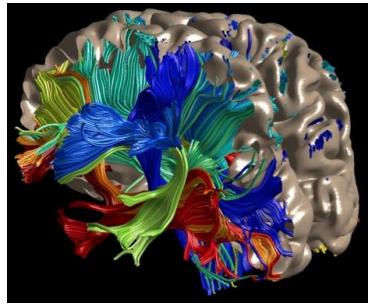
Broad Applications of 3D data

Robotics

Augmented Reality



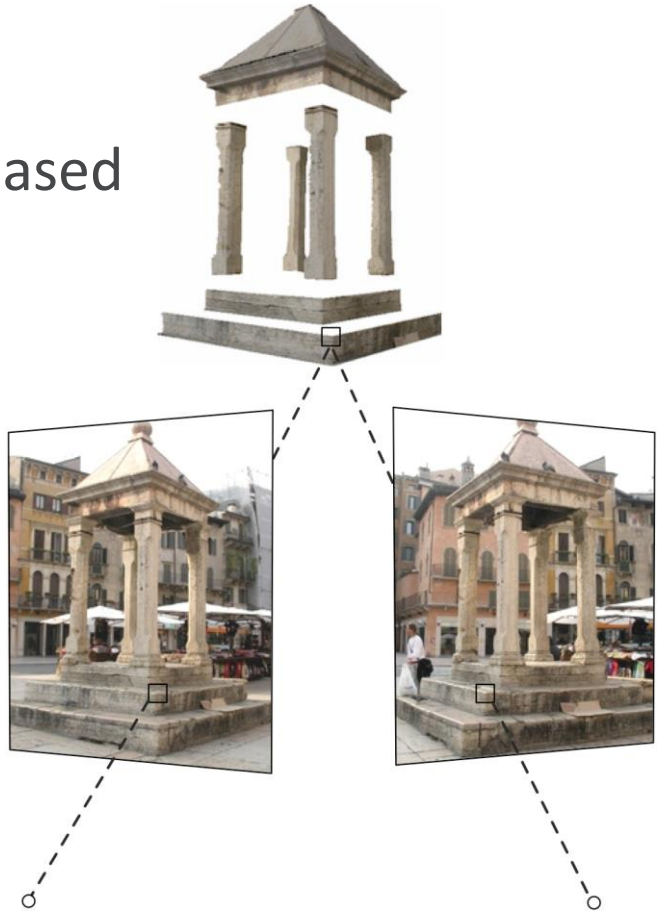
Autonomous driving



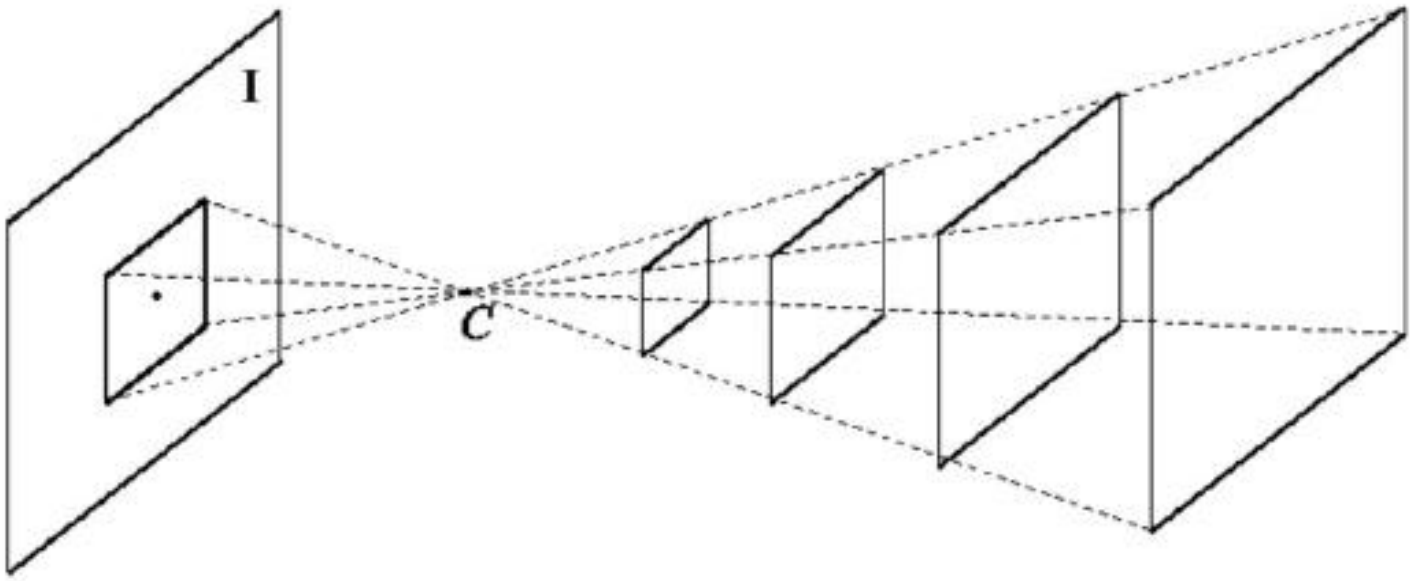
Medical Image Processing

Traditional 3D Vision

Multi-view Geometry: Physics based



3D from single image is ill-posed



Depth from 2D

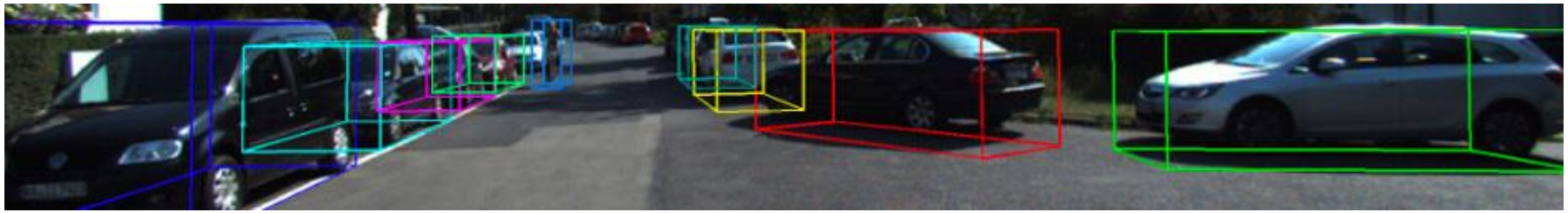
- Meaningful monocular cues:
- ❑ Linear Perspective
- ❑ Relative Size
- ❑ Superimposition
- ❑ Texture Gradient
- ❑ Height in plane

Acquire Knowledge of 3D World by Learning

A priori knowledge of the 3D world

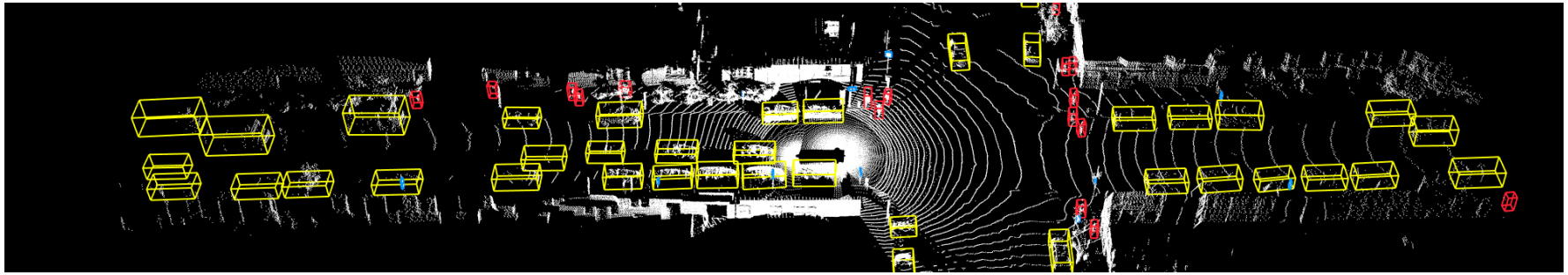
Datasets for Outdoor 3D Scenes

KITTI: LiDAR data, labeled by 3D b.boxes



Semantic KITTI: LiDAR data, labeled per point

Waymo Open Dataset: LiDAR data, labeled by 3D b.boxes



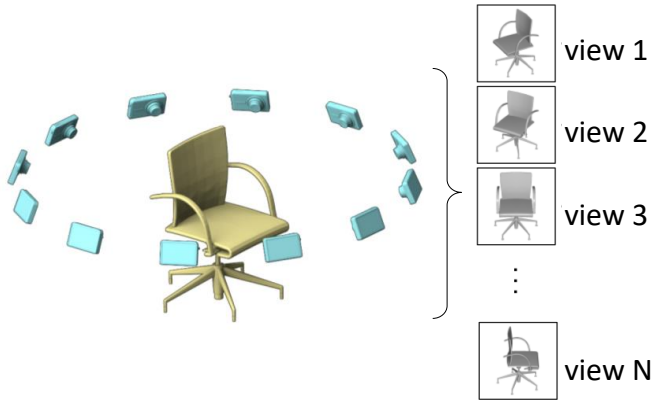
3D by Learning: Knowledge Based

Multi-View CNN

Given an Input Shape

Su et al., "**Multi-view Convolutional Neural Networks for 3D Shape Recognition**", *ICCV 2015*

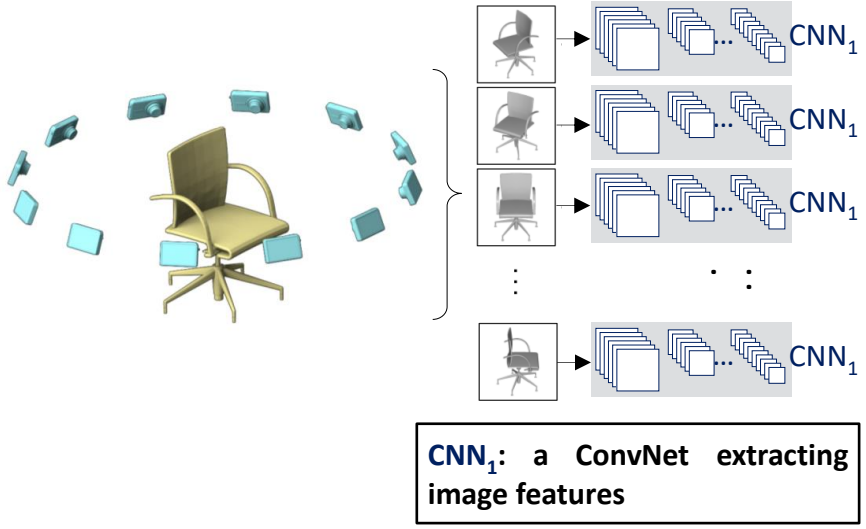
Render with Multiple Virtual Cameras



Su et al., "**Multi-view Convolutional Neural Networks for 3D Shape Recognition**", *ICCV 2015*

Render with Multiple Virtual Cameras

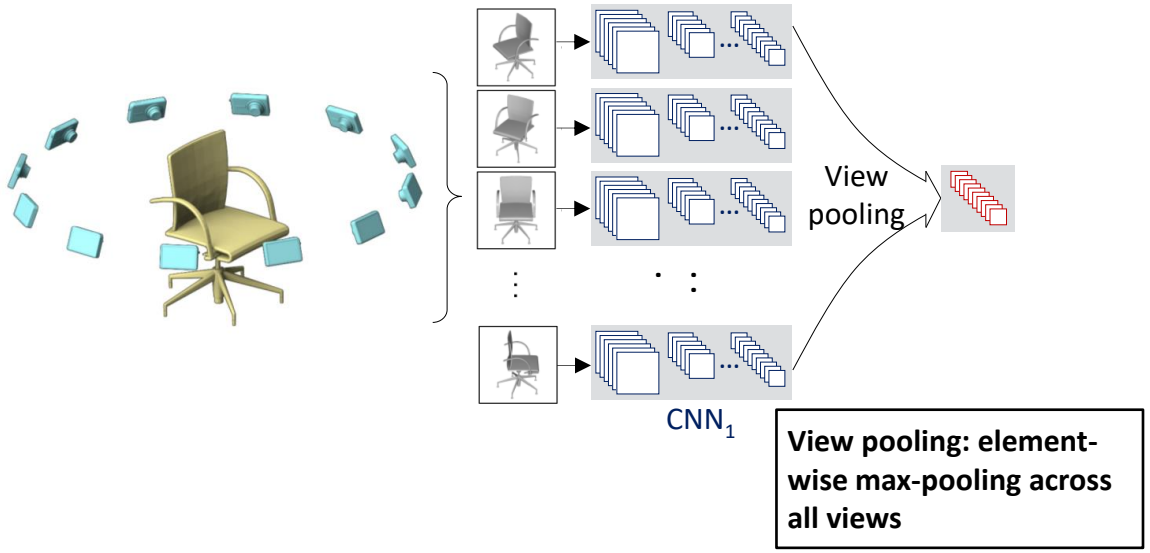
The Rendered Images are Passed through CNN_1 for Image Features



Su et al., "**Multi-view Convolutional Neural Networks for 3D Shape Recognition**", *ICCV 2015*

Render with Multiple Virtual Cameras

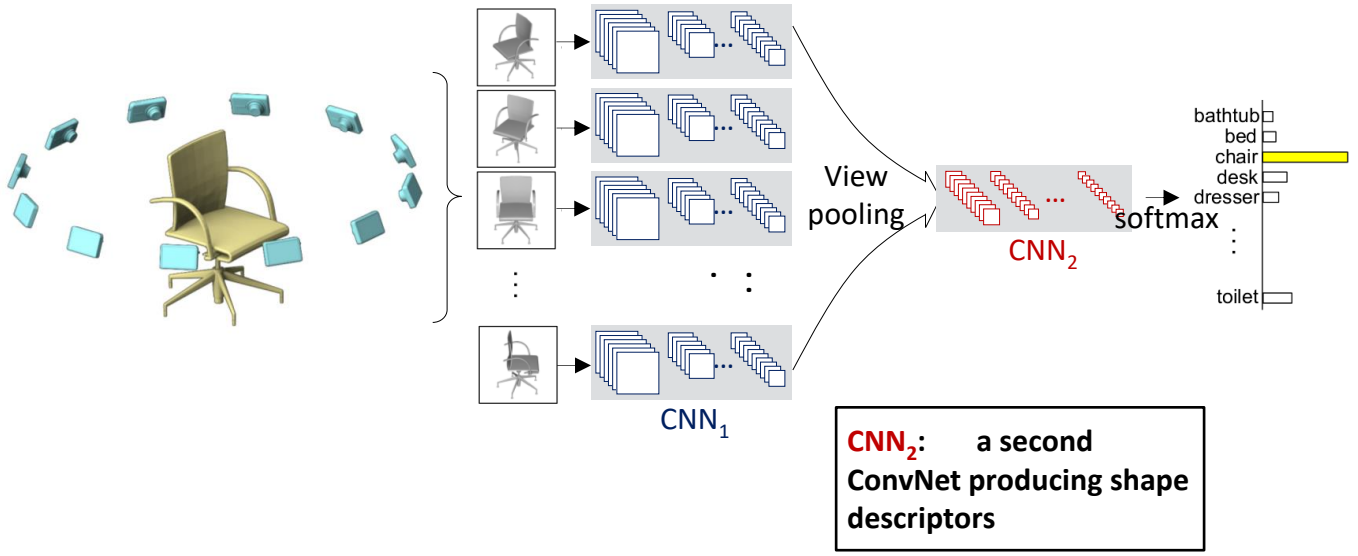
All Image Features are Combined by View Pooling



Su et al., "**Multi-view Convolutional Neural Networks for 3D Shape Recognition**", *ICCV 2015*

Render with Multiple Virtual Cameras

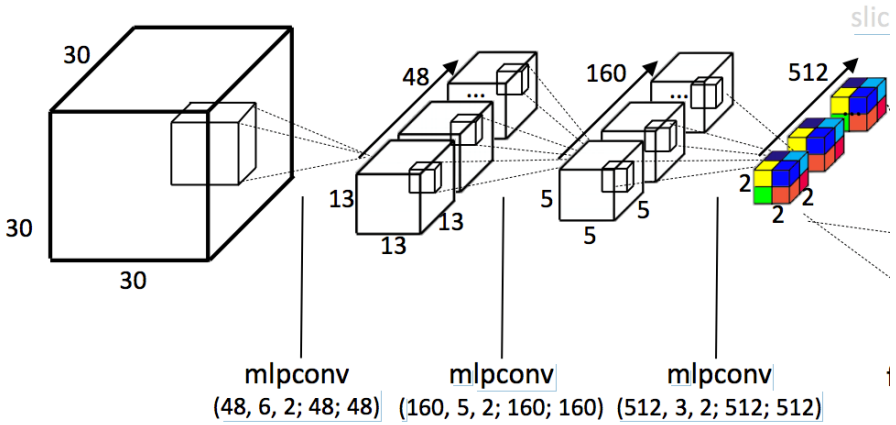
… and then Passed through CNN_2 and to Generate Final Predictions



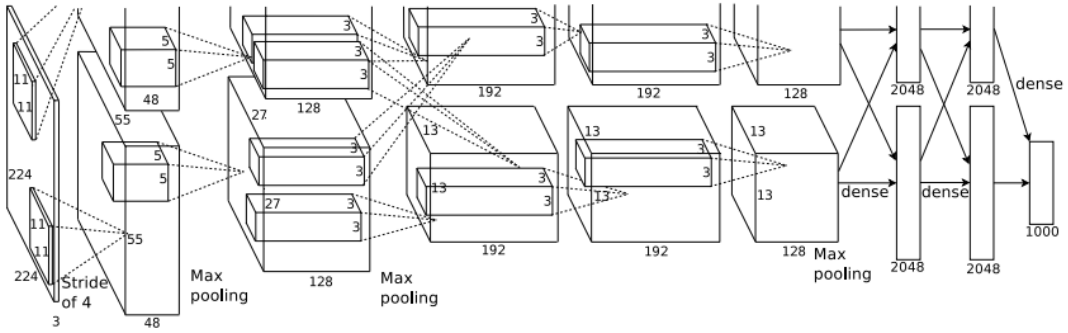
Su et al., "**Multi-view Convolutional Neural Networks for 3D Shape Recognition**", *ICCV 2015*

3D CNN on Volumetric Data

3D convolution uses 4D kernels

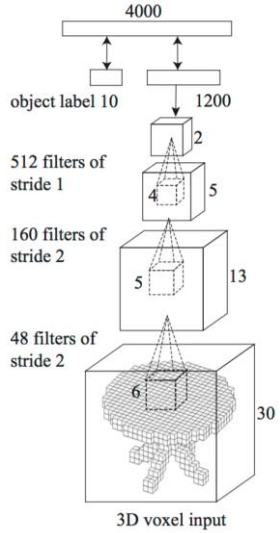


Complexity Issue



AlexNet, 2012

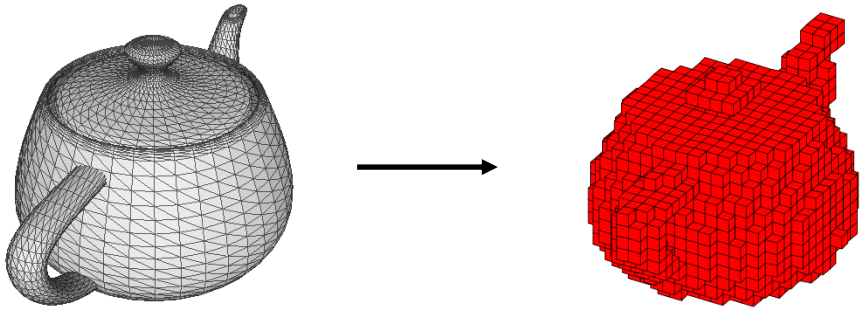
Input resolution: 224x224



3DShapeNets, 2015

Input resolution: 30x30x30

Complexity Issue



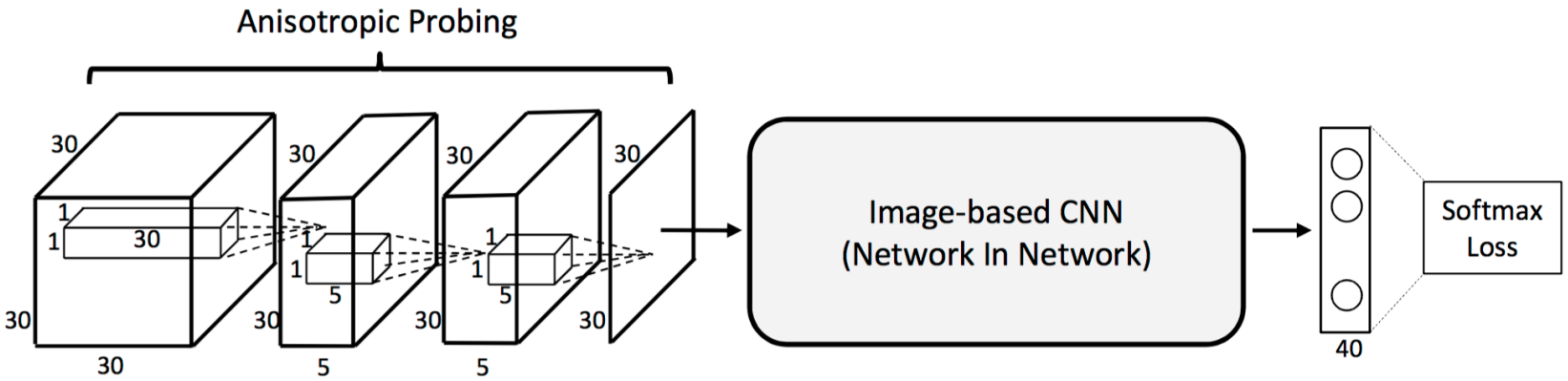
Polygon Mesh

Occupancy Grid
30x30x30

Information loss in voxelization

Idea 1: Learn to Project

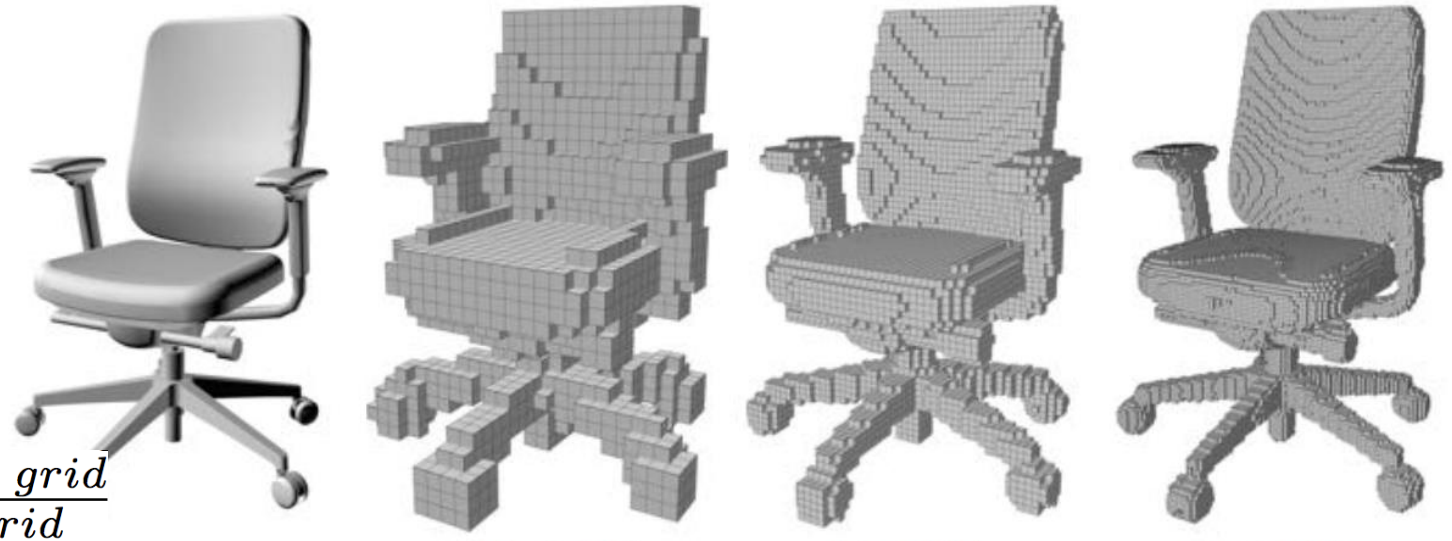
Idea: "X-ray" rendering + Image (2D) CNNs
very low #param, very low computation



Su et al., "**Volumetric and Multi-View CNNs for Object Classification on 3D Data**", *CVPR 2016*

Many other works in autonomous driving that uses bird's eye view for object detection

More Principled: Leverage Sparsity of 3D Surface Data



$$\frac{\#occupied\ grid}{\#total\ grid}$$

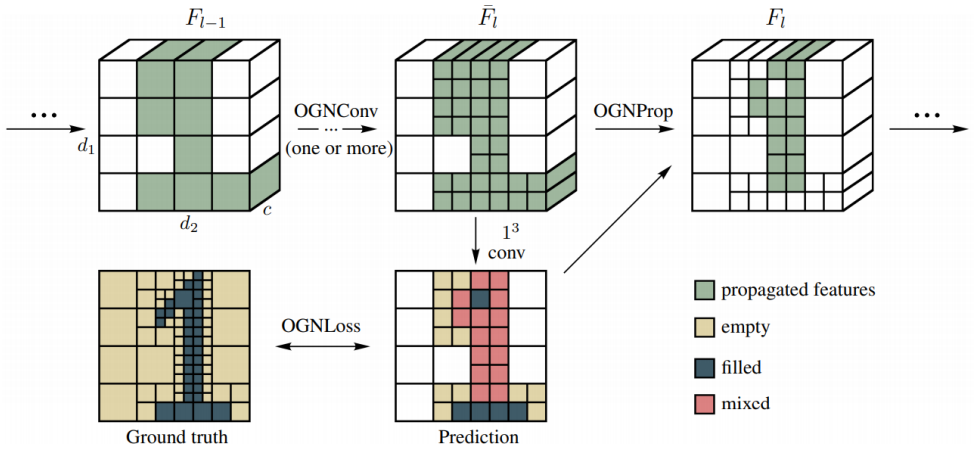
Occupancy:	10.41%	5.09%	2.41%
Resolution:	32	64	128

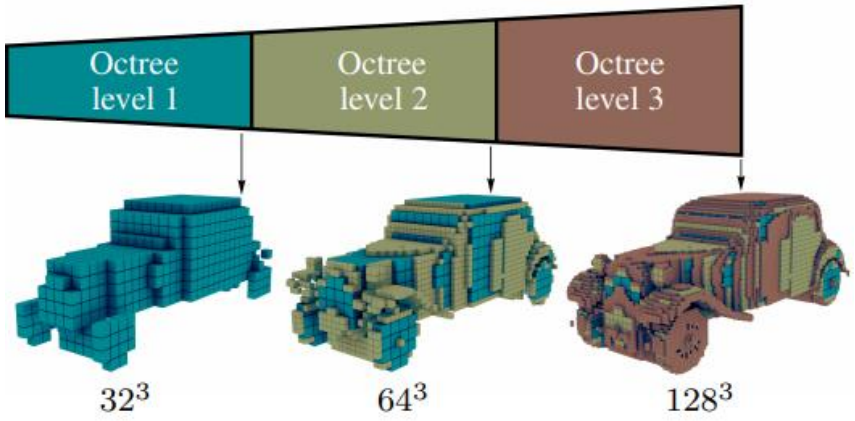
GAN for 3D

From Single Image to Volume

Avoid $\mathcal{O}(n^3)$ reconstruction

- Octree representation of shapes
- Generate the octree layer by layer

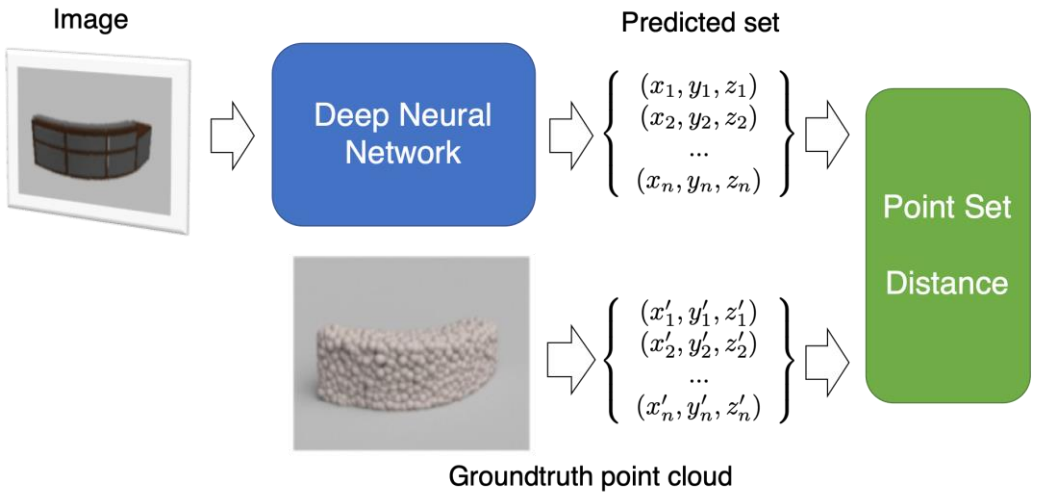




Tatarchenko et al., "**Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs**", *ICCV 2017*

From Single Image to Point Cloud

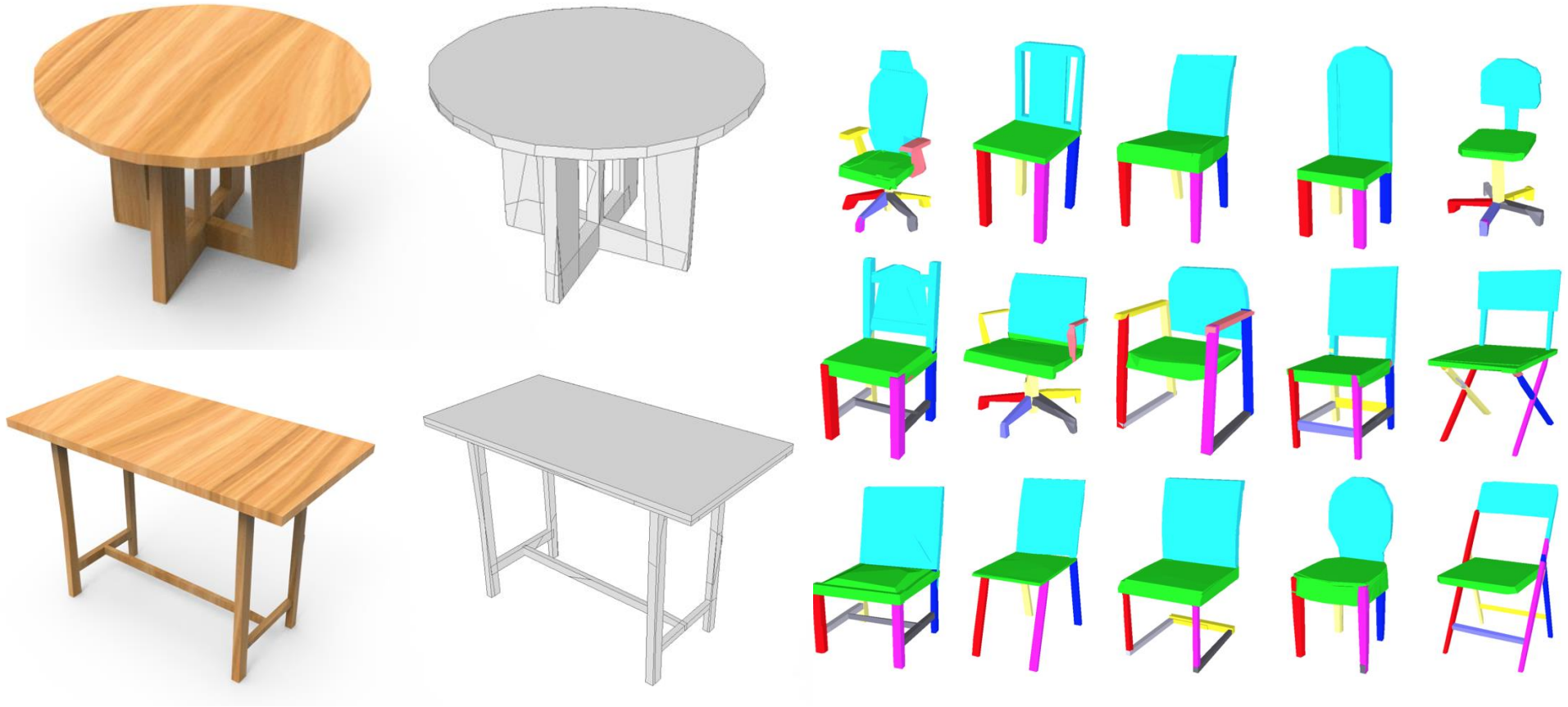
- It is possible to generate a **set** (permutation invariant)



Fan et al., "**A Point Set Generation Network for 3D Object Reconstruction from a Single Image**", *CVPR 2017*

From Image to Shape

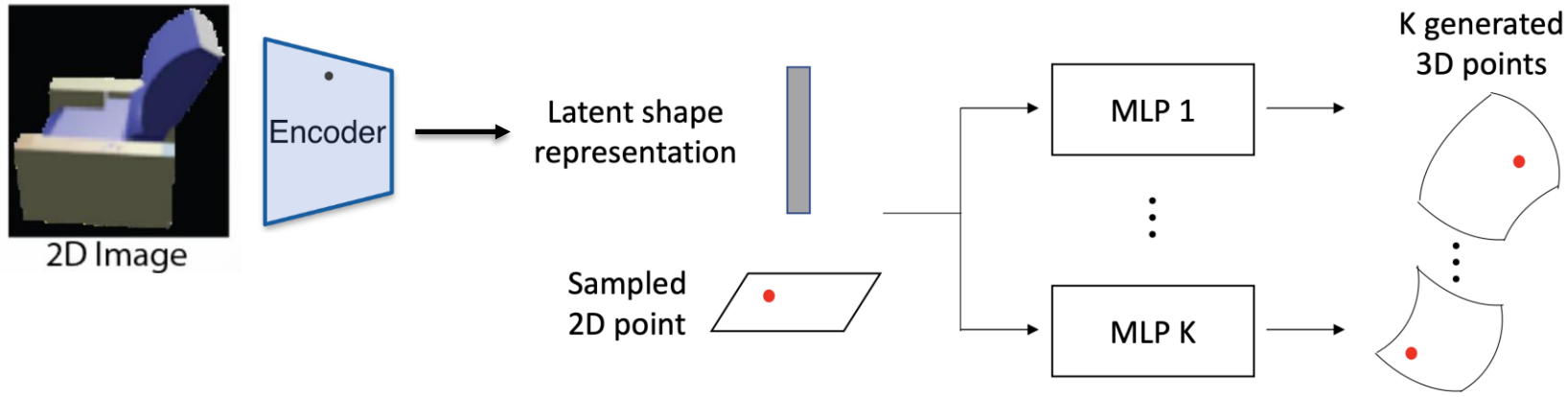
- Planes -> convex polytopes -> shape



Chen et al., "**BSP-Net: Generating Compact Meshes via Binary Space Partitioning**", *CVPR 2020*

From Image to Surface

- Learn to warp a plane to surface



Groueix et al., "**AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation**", *CVPR 2018*
Yang et al., "**Foldingnet: Point cloud auto-encoder via deep grid deformation**", *CVPR 2018*

Review

- 1 Video: Generation

  Popular GAN and diffusion models (AIGC is popular now)
- 2 Video: Analysis/Enhancement

  How a computer understands a video? (Commercial use)
- 3 3D: Keypoint and Structure from Motion

  Keypoint and keypoint-free matching is the key (SLAM, etc)
- 4 3D: Stereo Matching and Others.

  Stereo matching (3D imaging)