Introduction to Computer Vision

Lecture 10 - 3D Vision III and Temporal Analysis I

Prof. He Wang

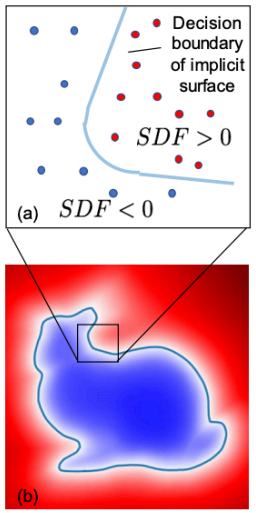
Logistics

- Assignment 3:
  - released on 4/26 evening
  - due on 5/10 11:59PM (Saturday)

- If 1 day (0 - 24 hours) past the deadline, 15% off
- If 2 day (24 - 48 hours) past the deadline, 30% off
- Zero credit if more than 2 days.

Implicit Field

Implicit Shape

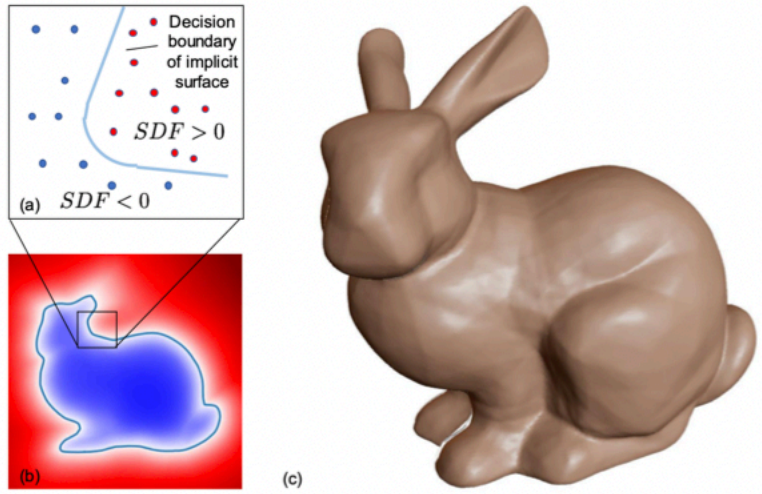


SDF

- Both an implicit geometry and surface representation
- Can convert into mesh
- Signed distance function, unsigned distance function, occupancy network

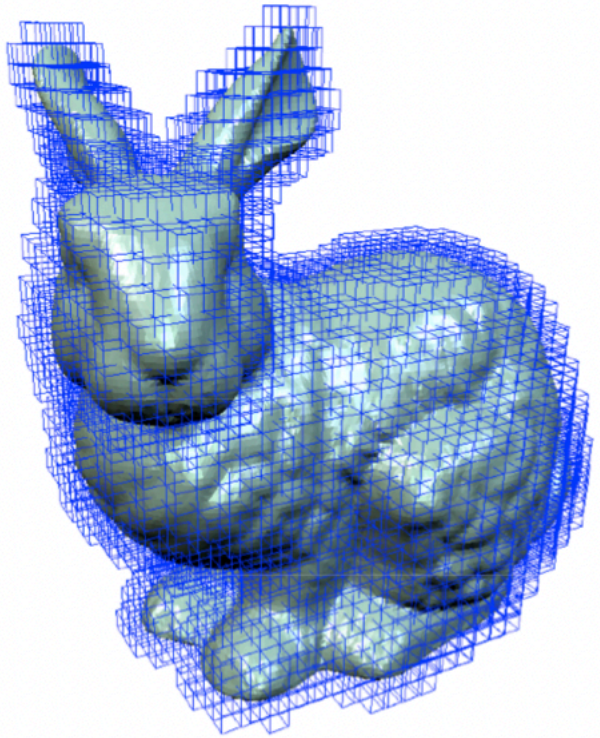
Signed Distance Function (SDF)

- Interior: $F(x, y, z) < 0$
- Exterior: $F(x, y, z) > 0$
- Surface: $F(x, y, z) = 0$ (zero set, zero iso-surface)
- Example implementation:
  - SDF: $F(x, y, z) =$ distance to the surface



How to Extract Zero Iso-Surface

Input: a signed distance field
(Implicitly assumed knowing the inside/outside of the shape, often needs to be estimated with **normal** information)



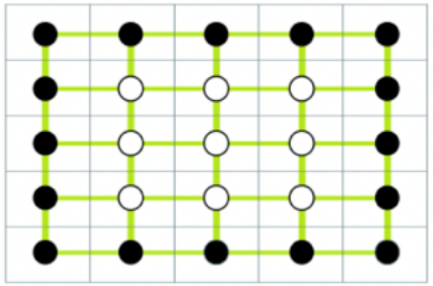
Classical Solution: 2D Marching Square

1	1	1	1	1
1	2	3	2	1
1	3	3	3	1
1	2	3	2	1
1	1	1	1	1

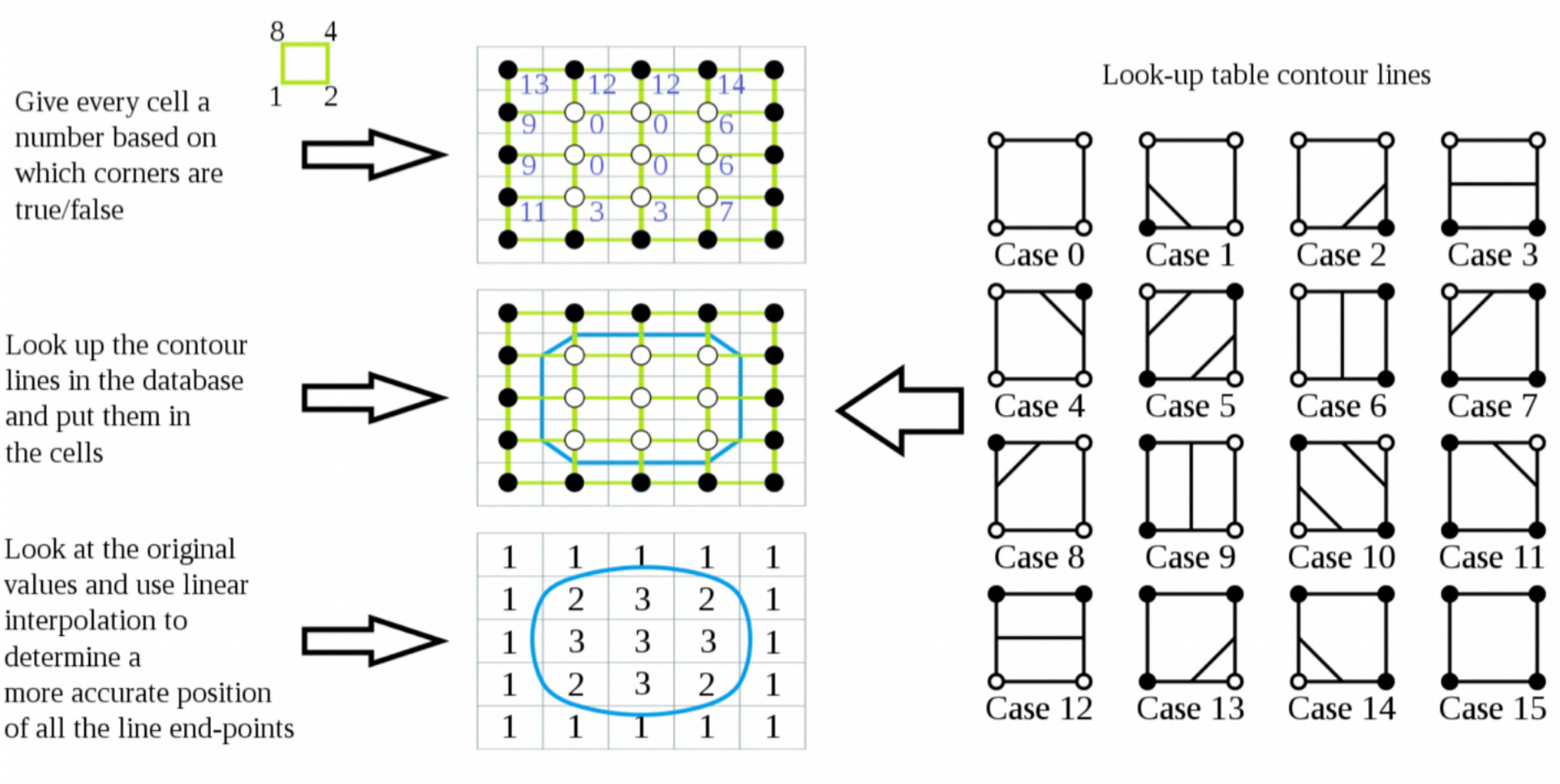
Threshold with iso-value

0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

Binary image to cells

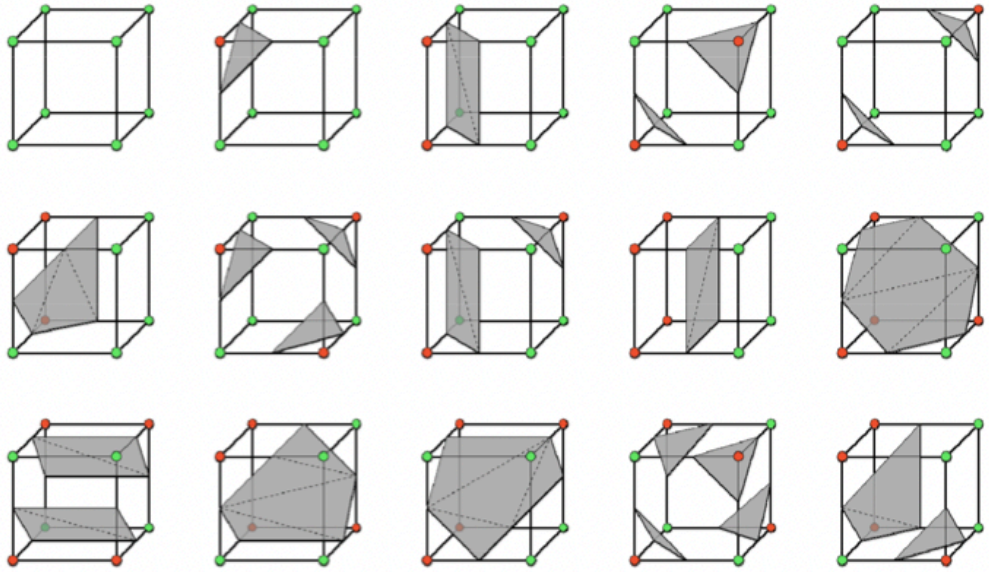


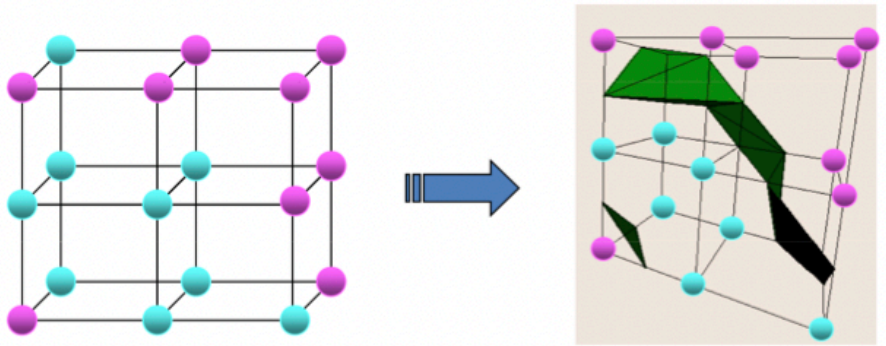
Classical Solution: 2D Marching Square



Classical Solution: 3D Marching Cube

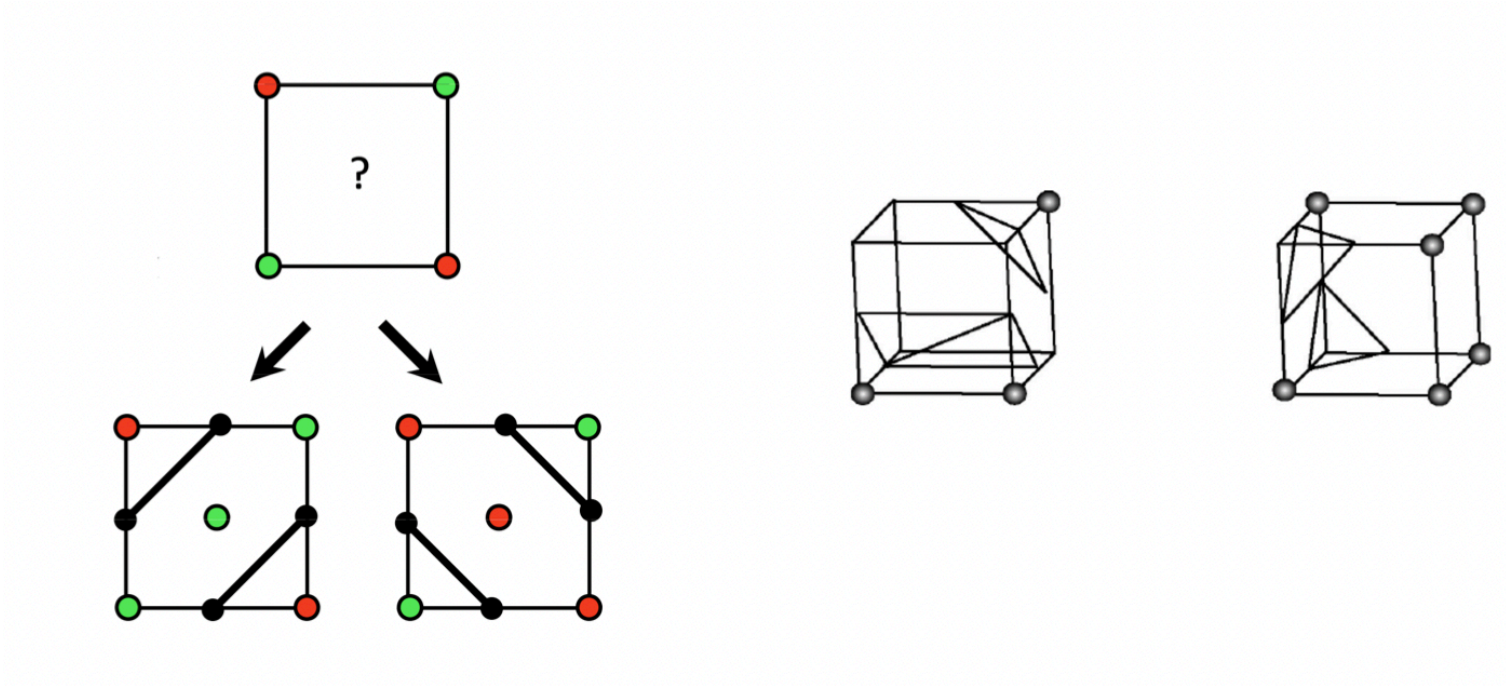
- $2^8 = 256$ cases
- The first published version exploits rotation and inversion, and only considers 15 unique cases:



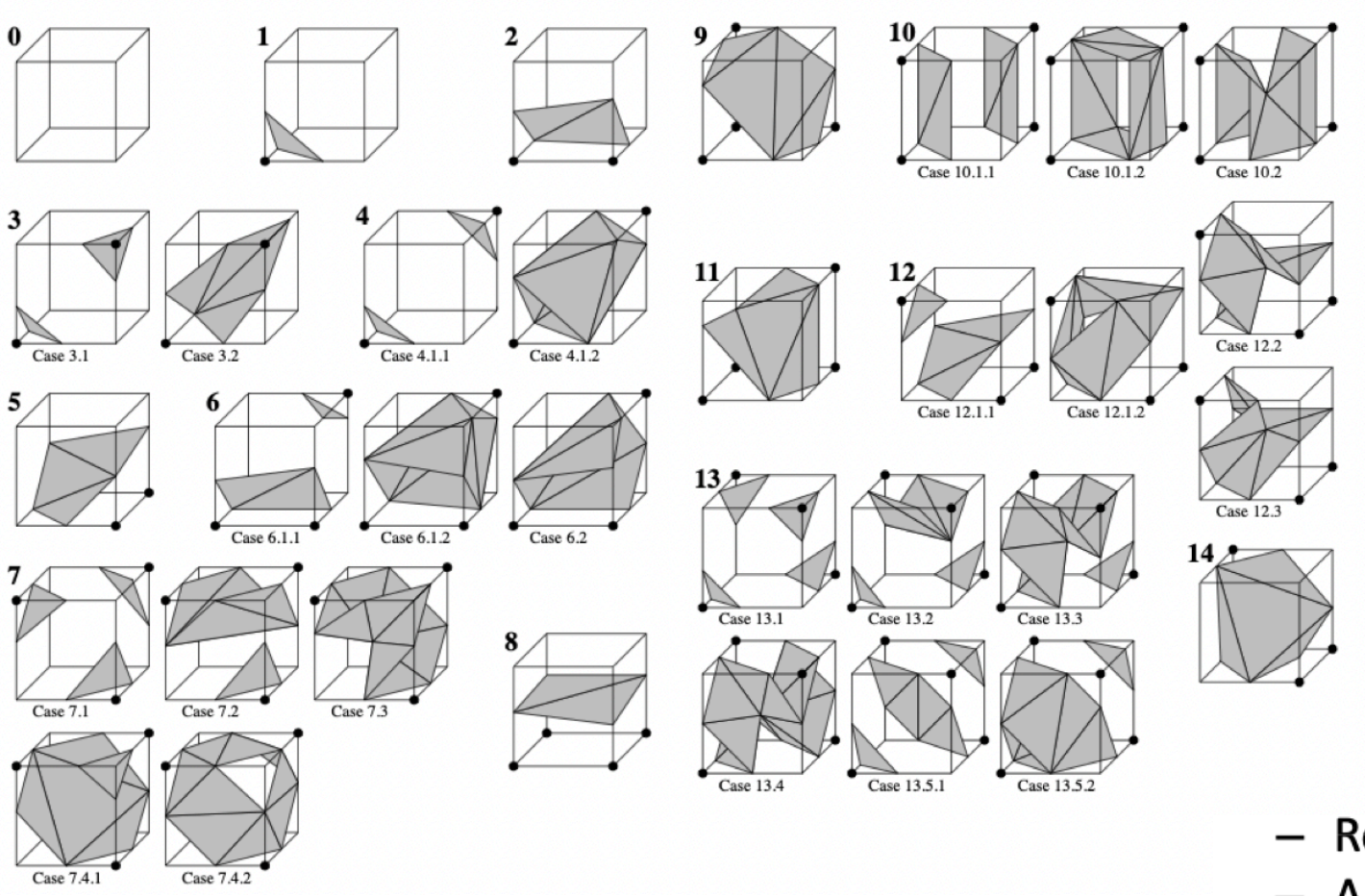


Ambiguity

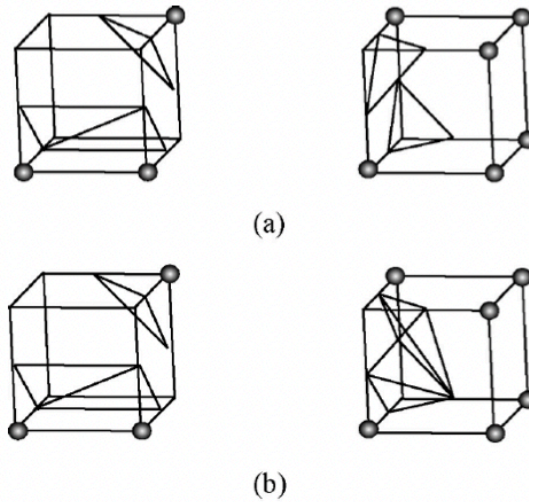
- Ambiguity leads to holes:



Solution to Ambiguity



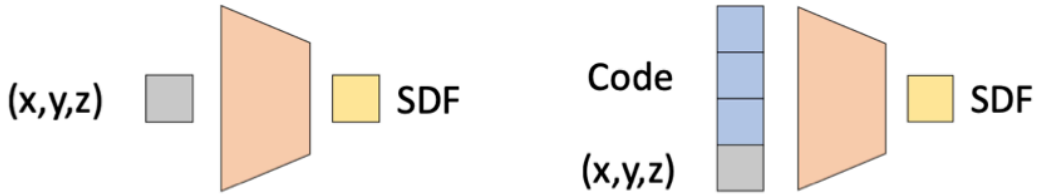
a



- Rotation only
- Always separate same “color”
- → Ambiguous faces triangulated consistently

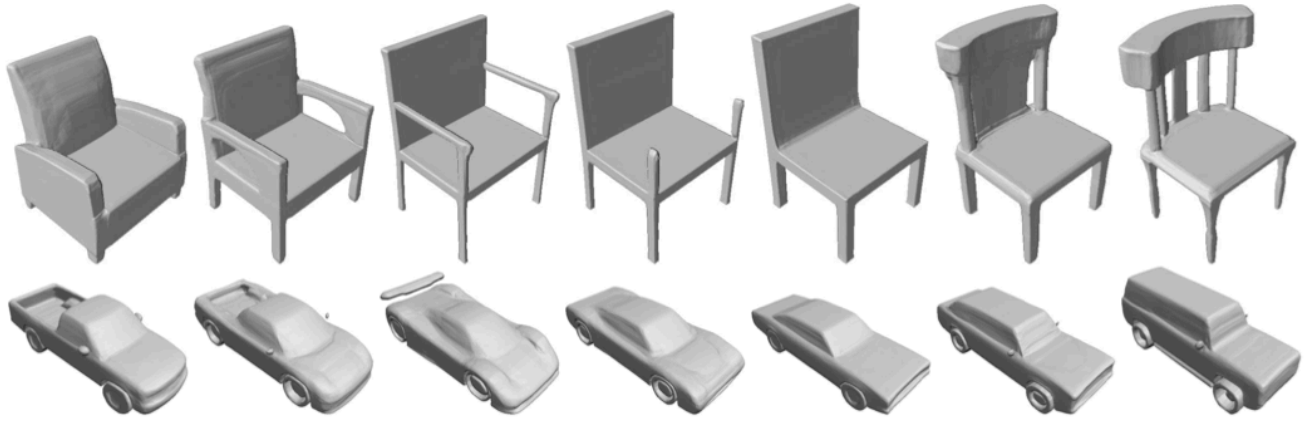
https://graphics.stanford.edu/courses/cs468-12-spring/LectureSlides/04_Surface_Reconstruction.pdf
Chernyaev et al., “Marching cube 33: Construction of topologically correct isosurfaces”, 1995

Deep SDF



(a) Single Shape DeepSDF **(b)** Coded Shape DeepSDF

- (a) use the network to overfit a single shape
- (b) use a latent code to represent a shape, so that the network can be used for multiple shapes



Park et al., "DeepSDF: Learning continuous signed distance functions for shape representation.", CVPR 2019

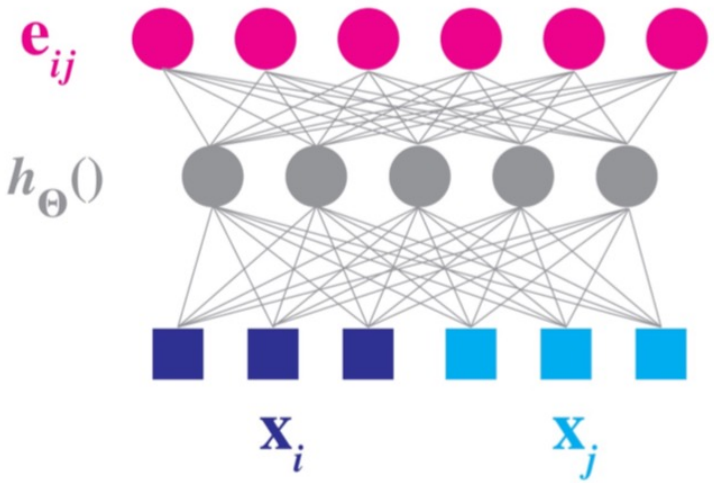
Neural Radiance Field

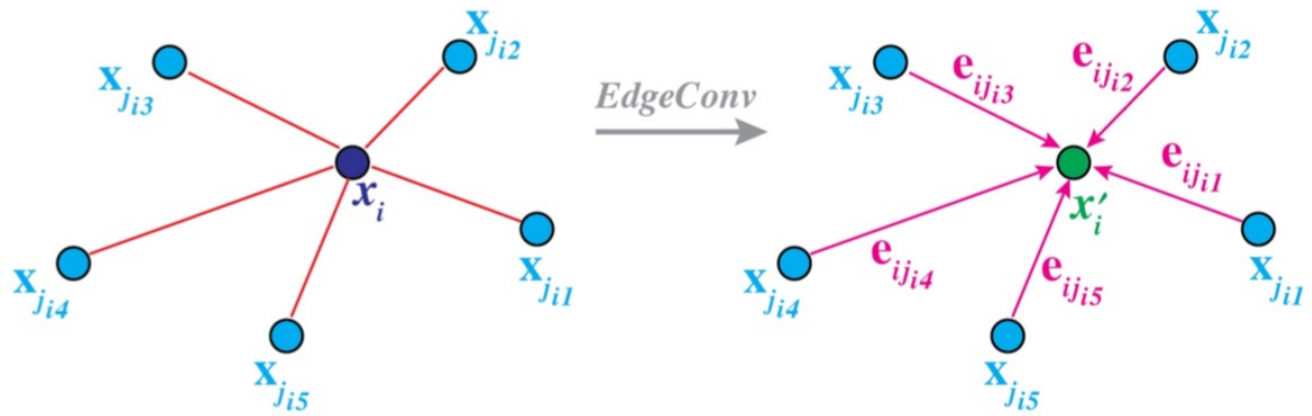
Training network to reproduce all input views of the scene

Neural Radiance Field (NeRF)

A brief intro to NeRF: https://www.youtube.com/watch?v=JuH79E8rdKc

Convolution on Mesh/Graph





Message passing: The output of EdgeConv at the i-th vertex is thus given by

$$\mathbf{x}'_i = \square_{j:(i,j)\in\mathcal{E}} h_\Theta(\mathbf{x}_i, \mathbf{x}_j). \quad (1)$$

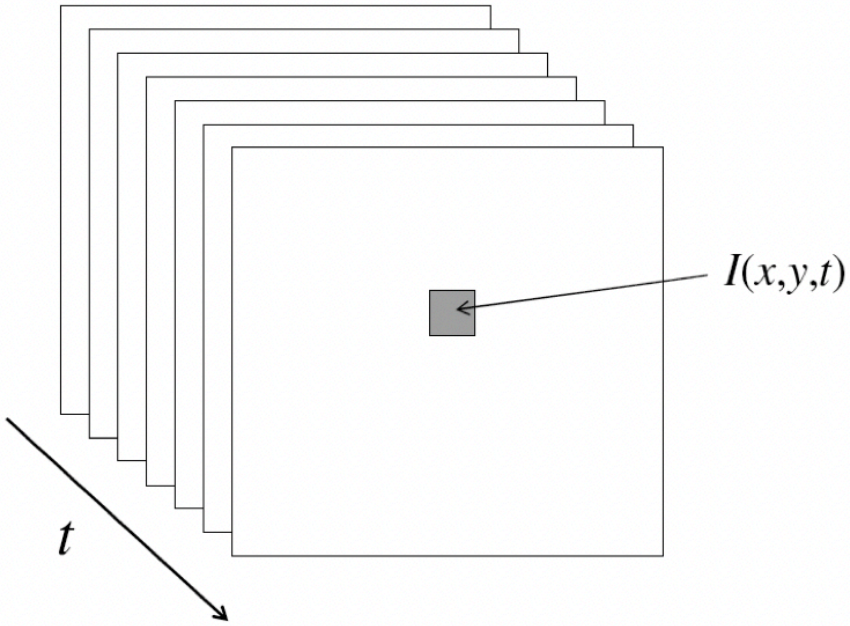
Wang, et.al., Dynamic Graph CNN for Learning on Point Clouds, ToG 2019

Motion and Optical Flow

Some slides are borrowed from Stanford CS131.

From Single Image to Video

- A video is a sequence of frames captured over time
- Now our image data is a function of space (x, y) and time (t)



Today let's focus on motions between two consecutive frames!

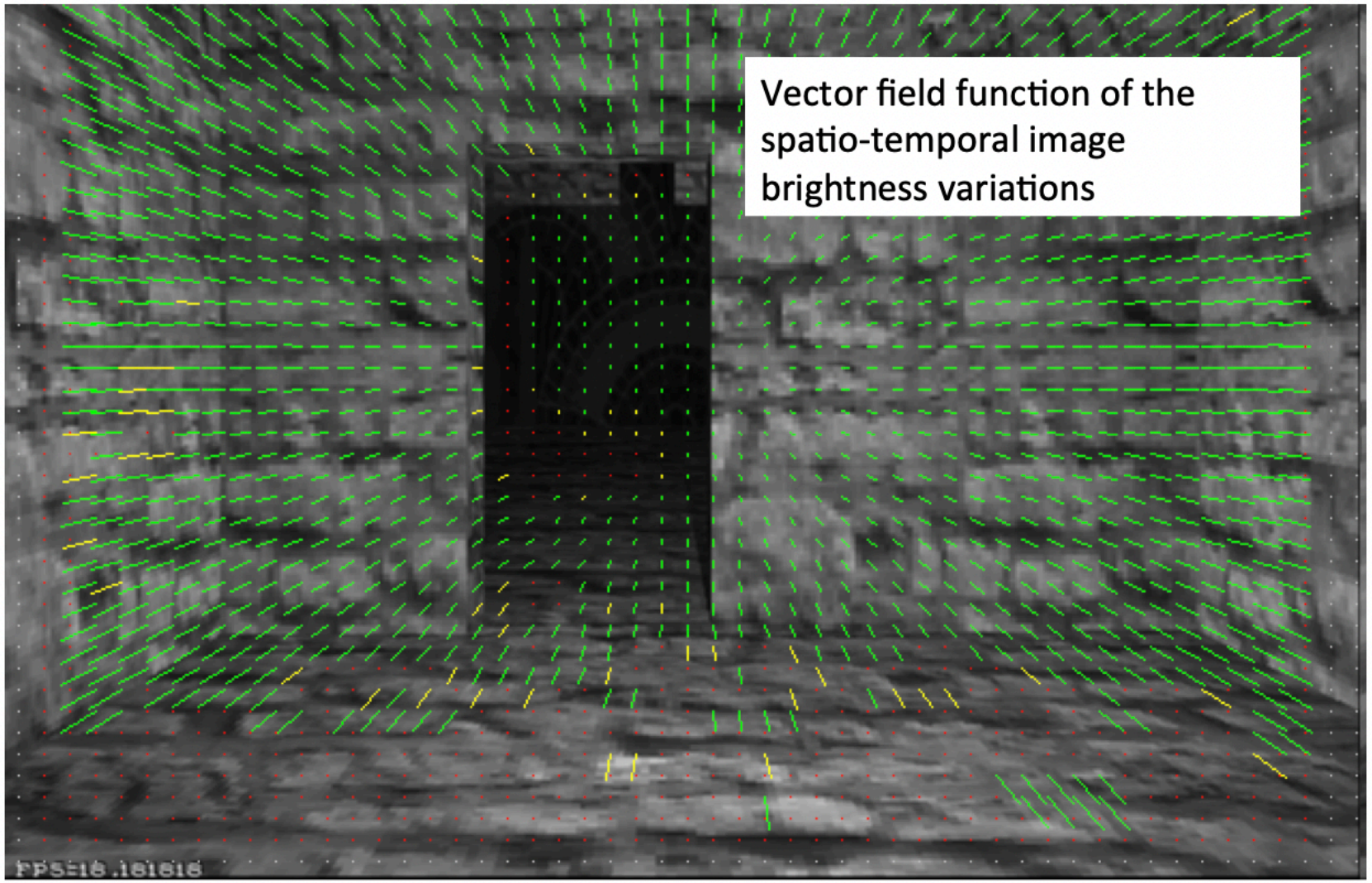
Optical Flow

- Definition: optical flow is the *apparent* motion of brightness patterns in the image

- Note: apparent motion can be caused by lighting changes without any actual motion
  - Think of a uniform rotating sphere under fixed lighting vs. a stationary sphere under moving illumination

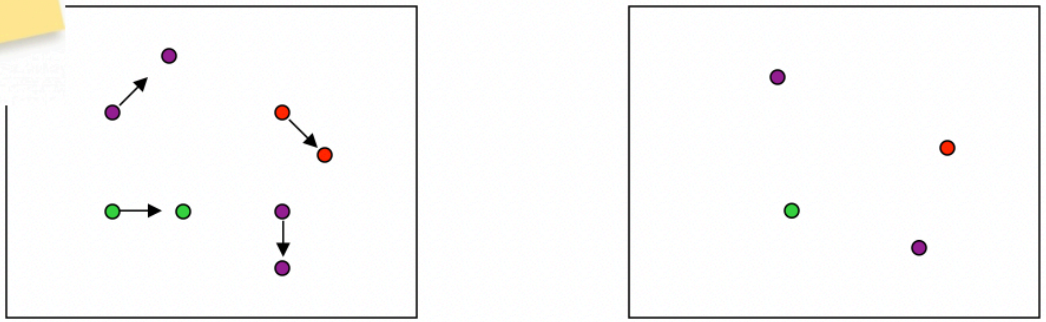
GOAL: Recover image motion at each pixel from optical flow

Optical Flow



Picture courtesy of Selim Temizer - Learning and Intelligent Systems (LIS) Group, MIT

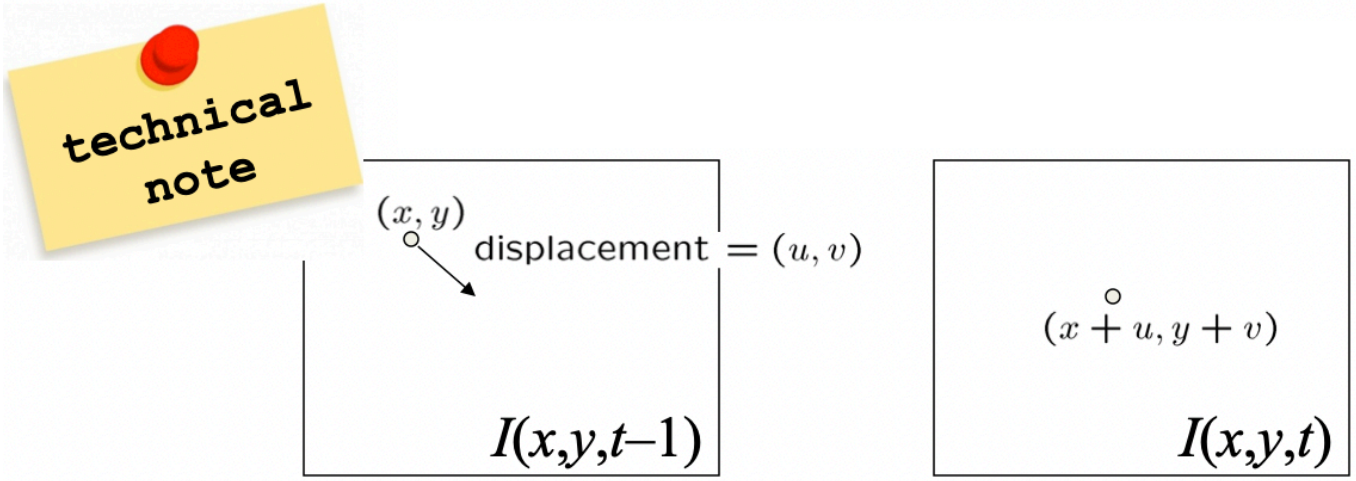
Estimating Optical Flow



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

- Given two subsequent frames, estimate the apparent motion field u(x,y), v(x,y) between them
- Key assumptions
  - **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

The Brightness Constancy Constraint



- Brightness Constancy Equation:

$$I(x,y,t-1) = I(x+u(x,y), y+v(x,y), t)$$

Linearizing the right side using Taylor expansion:

Image derivative along x

$$I(x+u,y+v,t) \approx I(x,y,t-1) + I_x \cdot u(x,y) + I_y \cdot v(x,y) + I_t$$

$$I(x+u,y+v,t) - I(x,y,t-1) = I_x \cdot u(x,y) + I_y \cdot v(x,y) + I_t$$

Hence, $I_x \cdot u + I_y \cdot v + I_t \approx 0 \quad \rightarrow \nabla I \cdot \left[u \; v\right]^T + I_t = 0$

The Brightness Constancy Constraint

Can we use this equation to recover image motion (u,v) at each pixel?

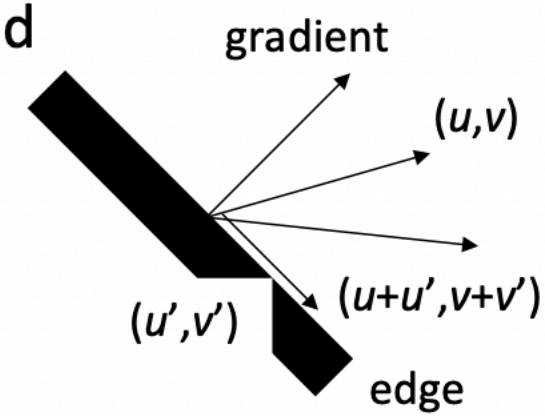
$$\nabla I \cdot \begin{bmatrix} u & v \end{bmatrix}^T + I_t = 0$$

- How many equations and unknowns per pixel?
  - One equation (this is a scalar equation!), two unknowns (u,v)

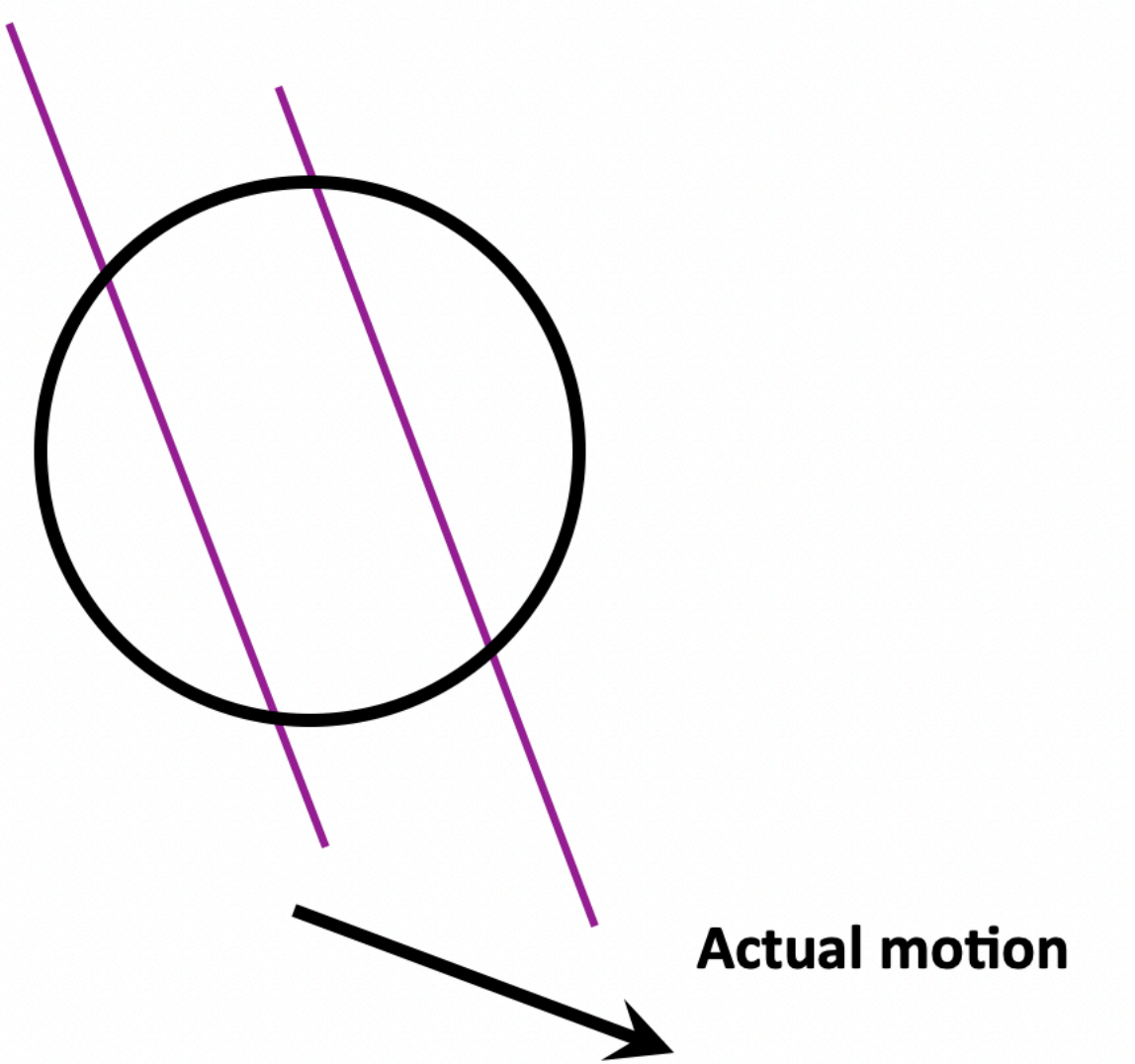
The component of the flow perpendicular to the gradient (i.e., parallel to the edge) cannot be measured

If (u, v) satisfies the equation, so does $(u+u', v+v')$ if

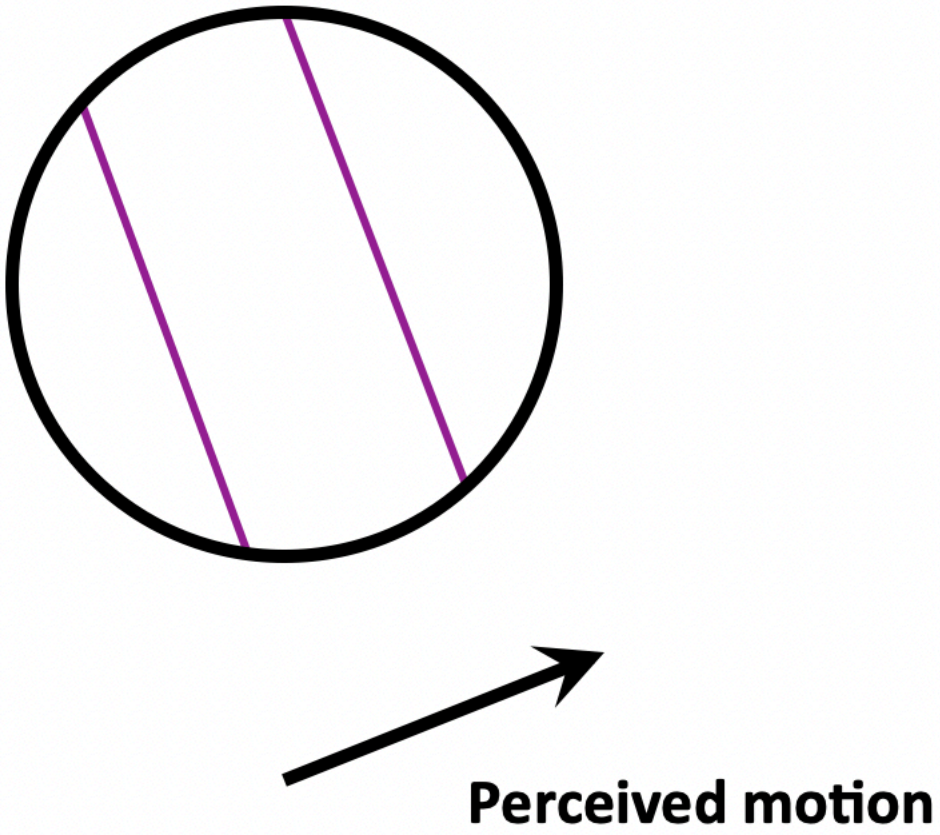
$$\nabla I \cdot \begin{bmatrix} u' & v' \end{bmatrix}^T = 0$$



The Aperture Problem



The Aperture Problem



Barberpole Illusion

- https://en.wikipedia.org/wiki/Barberpole_illusion

Solving the Ambiguity

- How to get more equations for a pixel?
- **Spatial coherence constraint:**
- Assume the pixel's neighbors have the same (u,v)
  - If we use a 5x5 window, that gives us 25 equations per pixel

$$0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u \ v]$$

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}$$

B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In Proceedings of the Interna6onal Joint Conference on Artificial Intelligence, pp. 674- 679, 1981.

Lucas-Kanade Flow

- **Overconstrained linear system:**

$$\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix}$$

$$\underset{25\text{x}2}{A} \quad \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

Lucas-Kanade Flow

- Overconstrained linear system:

$$\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix}$$

$$\underset{25\text{x}2}{A} \quad \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

Least squares solution for *d* given by $(A^T A)\ d = A^T b$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$A^T A$ $\qquad$ $A^T b$

The summations are over all pixels in the K x K window

Condition for Solvability

– Optimal (u, v) satisfies Lucas-Kanade equation

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$A^T A$ $\qquad$ $A^T b$

When is This Solvable?

- $\mathbf{A^T A}$ should be invertible
- $\mathbf{A^T A}$ should not be too small due to noise
  - eigenvalues λ_1 and λ_2 of $\mathbf{A^T A}$ should not be too small
- $\mathbf{A^T A}$ should be well-conditioned
  - λ_1 / λ_2 should not be too large (λ_1 = larger eigenvalue)

Does this remind anything to you?

Condition for Solvability

M = A^TA is the *second moment matrix* !

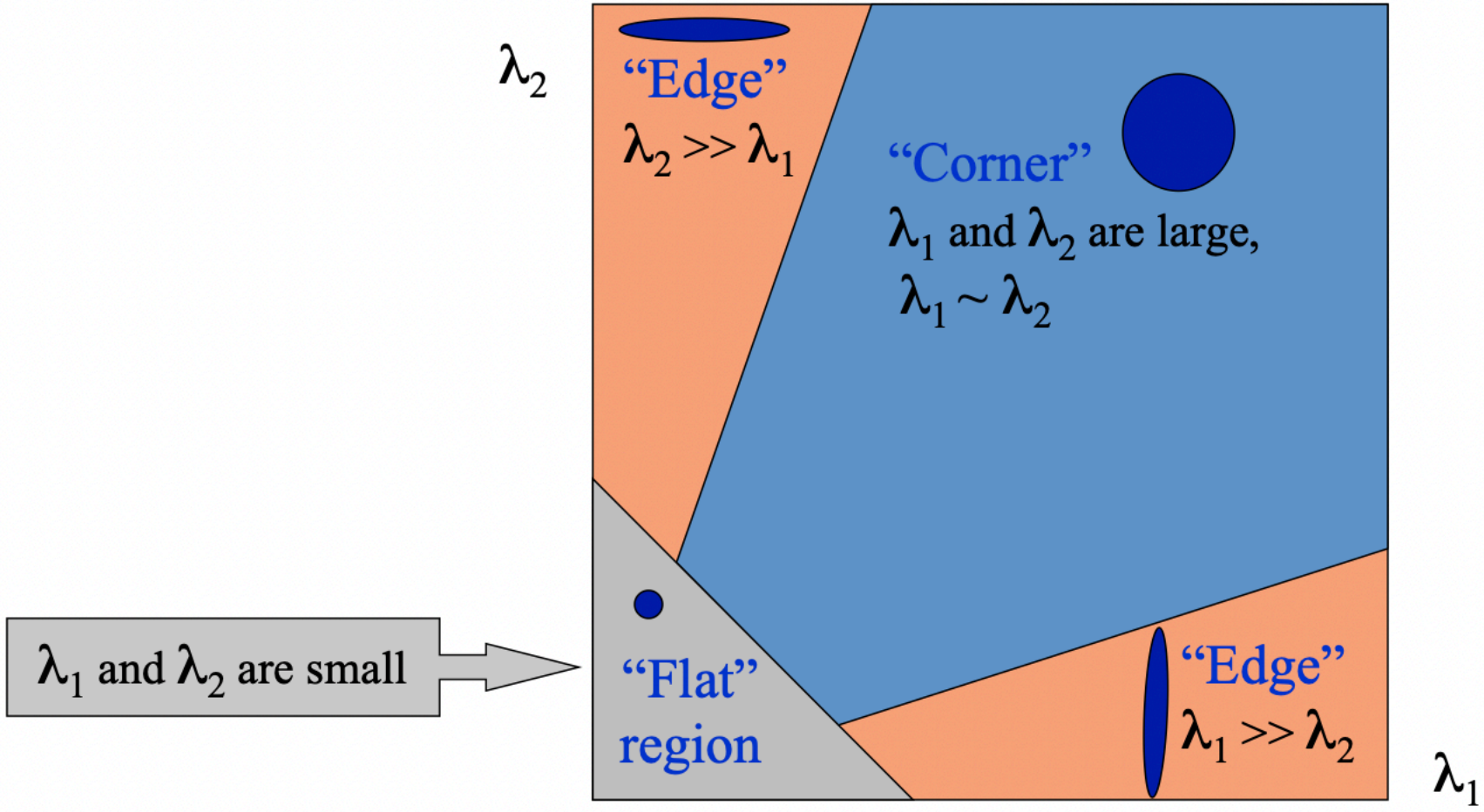
(Harris corner detector…)

$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y] = \sum \nabla I (\nabla I)^T$$

- Eigenvectors and eigenvalues of A^TA relate to edge direction and magnitude
  - The eigenvector associated with the larger eigenvalue points in the direction of fastest intensity change
  - The other eigenvector is orthogonal to it

Interpreting the Eigenvalues

Classification of image points using eigenvalues of the second moment matrix:



Edge

$$\sum \nabla I (\nabla I)^T$$

– gradients very large or very small

– large λ_1, small λ_2

Low-Texture Region

$$\sum \nabla I (\nabla I)^T$$

- gradients have small magnitude
- small λ_1, small λ_2

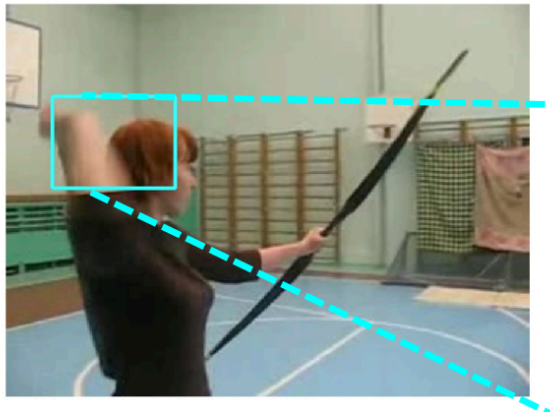
High Texture Region

$$\sum \nabla I (\nabla I)^T$$

- gradients are different, large magnitudes
- large λ_1, large λ_2

Measuring Motion

Image at frame t



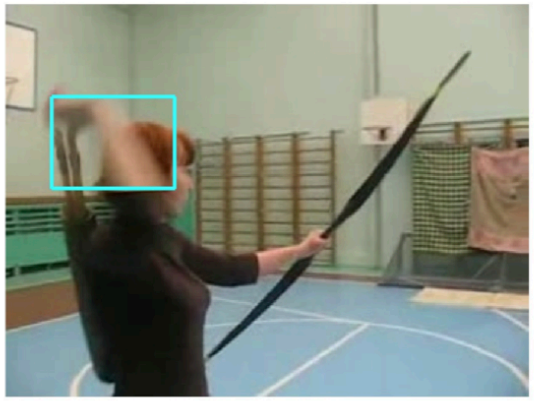
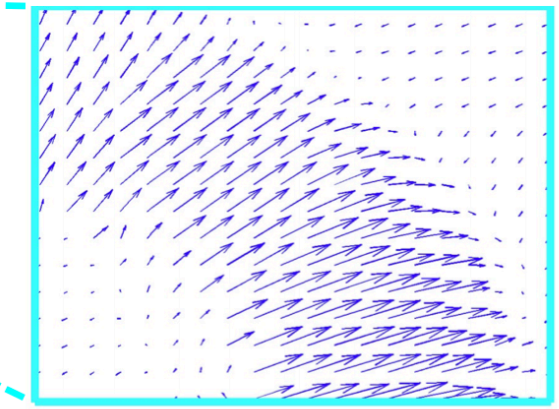


Image at frame t+1

Optical flow gives a displacement field F between images I_t and I_{t+1}



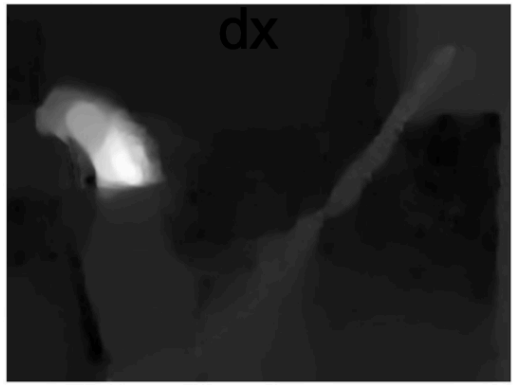
Tells where each pixel will move in the next frame:

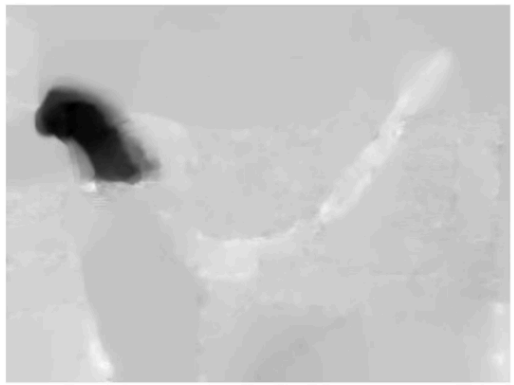
$F(x, y) = (dx, dy)$

$I_{t+1}(x+dx, y+dy) = I_t(x, y)$

Optical Flow highlights **local motion**

Horizontal flow



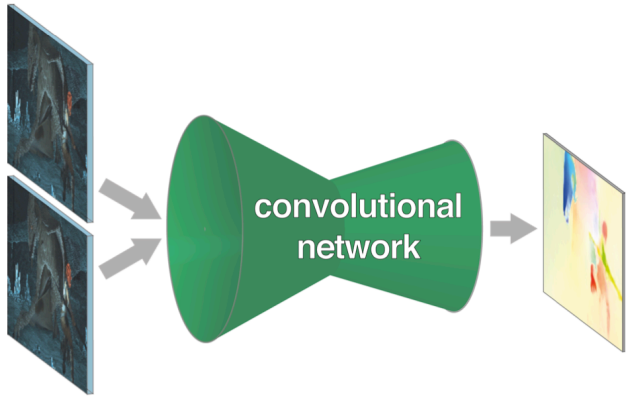


Vertical Flow dy

Slide credit: Justin Johnson

Simonyan and Zisserman, "Two-stream convolutional networks for action recognition in videos", NeurIPS 2014

FlowNet: Learning Optical Flow with Convolutional Networks



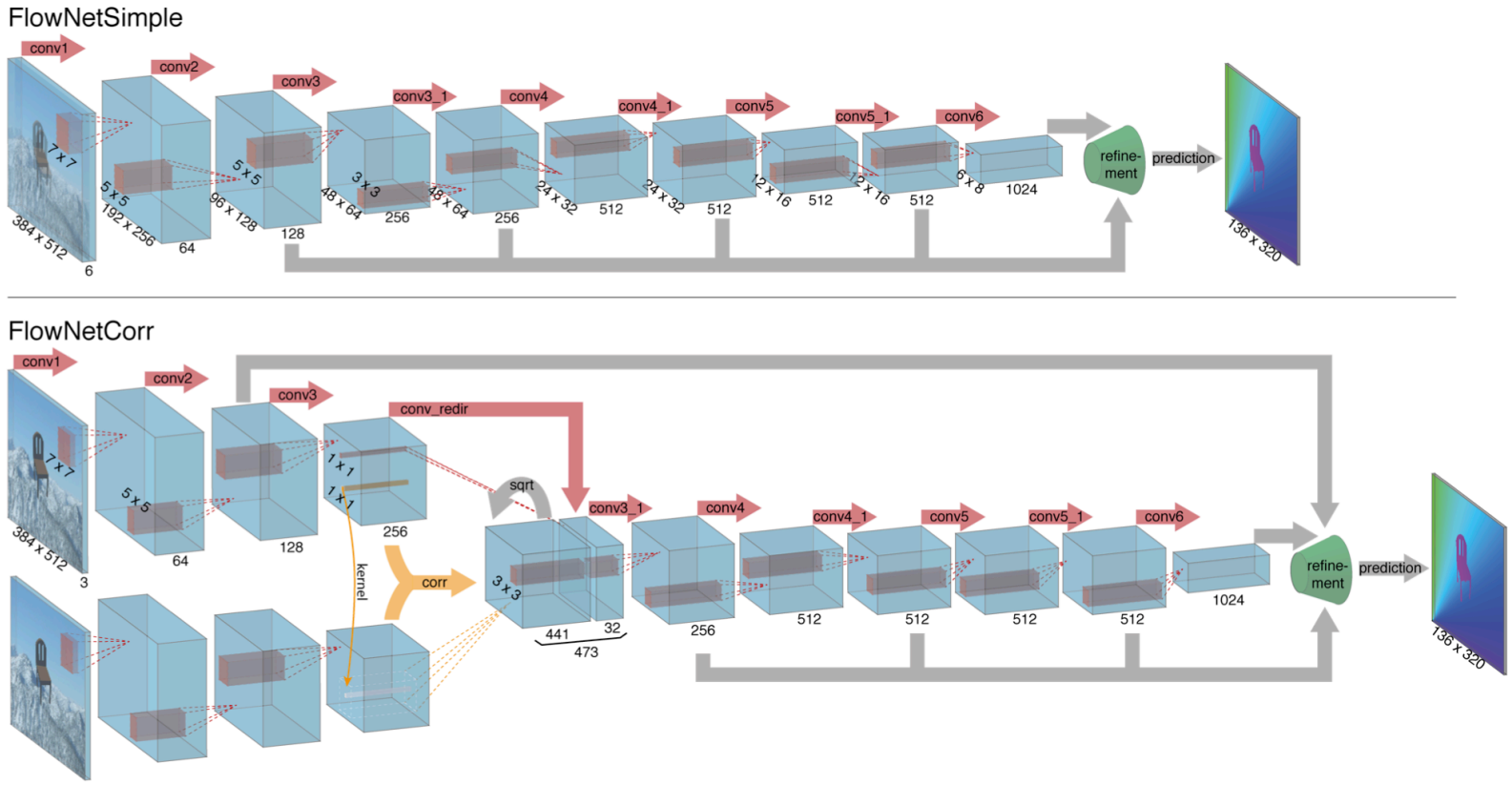


Figure 2. The two network architectures: FlowNetSimple (top) and FlowNetCorr (bottom).

Dosovitskiy, Alexey, et al. "Flownet: Learning optical flow with convolutional networks." *CVPR* 2015.

Next week:
Lecture 11 Temporal Analysis II