CS 558: Computer Vision 6th Set of Notes

Instructor: Enrique Dunn
Webpage: www.cs.stevens.edu/~edunn
E-mail: edunn@stevens.edu
Office: North Bldg 219

Overview

- Image Transformations
  - Based on slides by R. Szeliski
- Feature Tracking
  - Based on slides by D. Hoiem
- Dense Optical Flow
  - Based on slides by K. Grauman
- Grouping and Segmentation
  - Based on slides by D. Hoiem

Image Transformations

Slides by R. Szeliski

Image Transformations

image filtering: change ***range*** of image

$$g(x) = T(f(x))$$

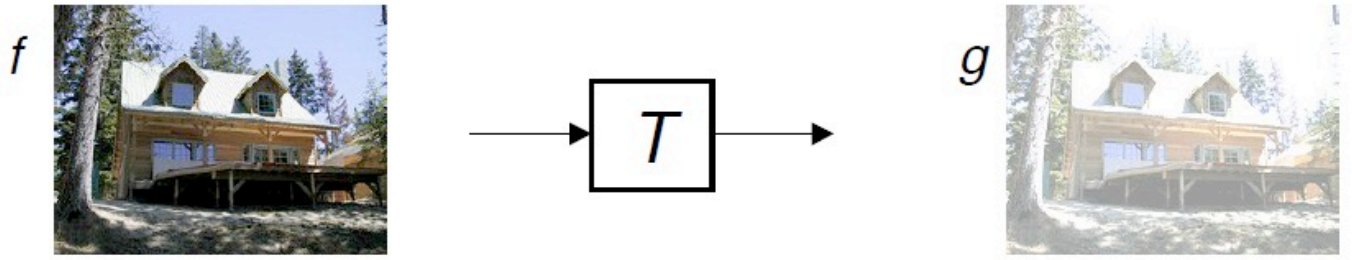
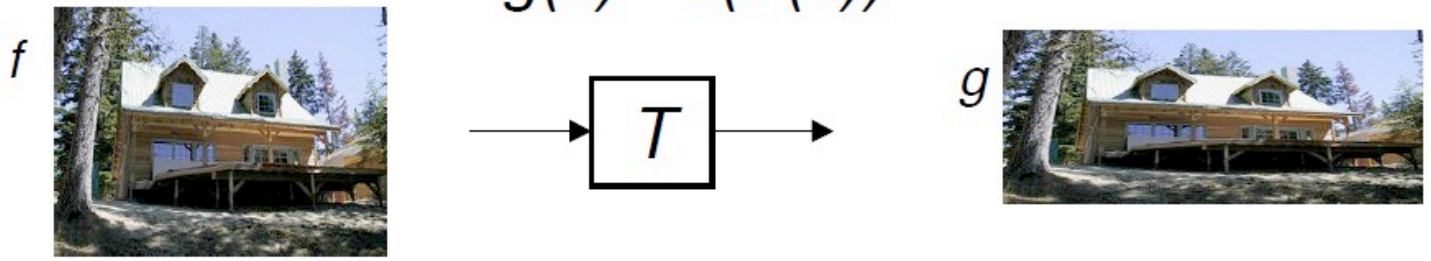
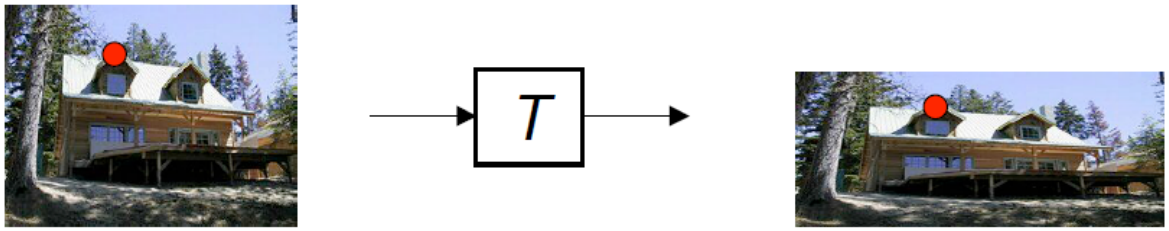


image warping: change ***domain*** of image

$$g(x) = f(T(x))$$



Parametric (Global) Warping



p = (x,y)

p' = (x',y')

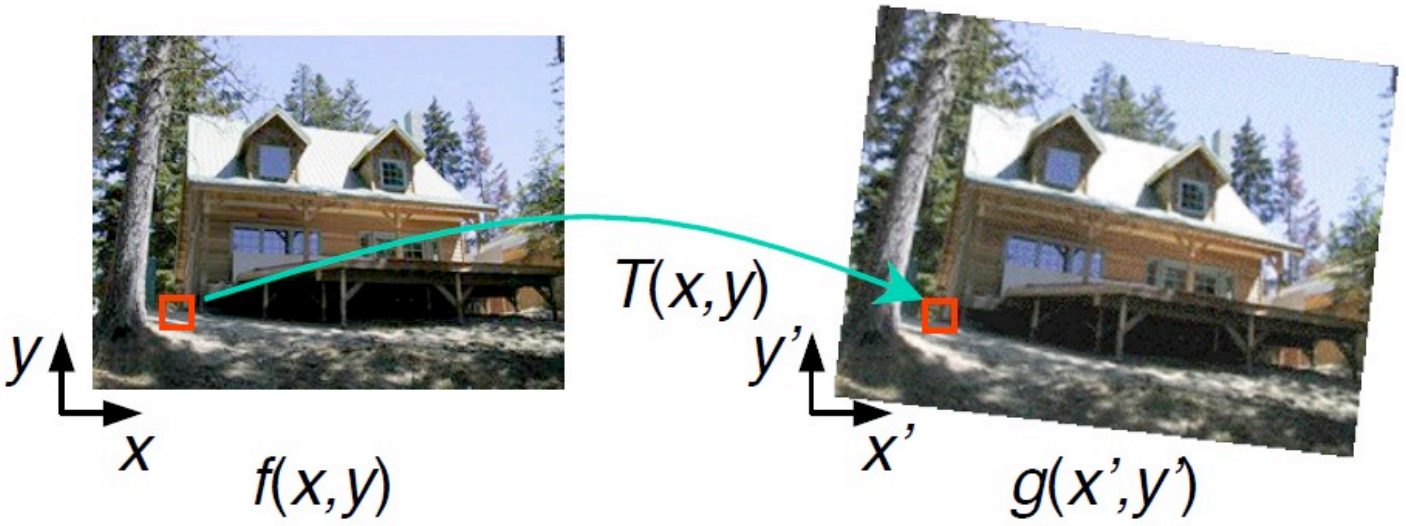
- Transformation T is a coordinate-changing machine:

  p' = T(p)

- What does it mean that T is global?
  - It is the same for any point p
  - It can be described by just a few numbers (parameters)
- T is represented as a matrix (see prev. slides):

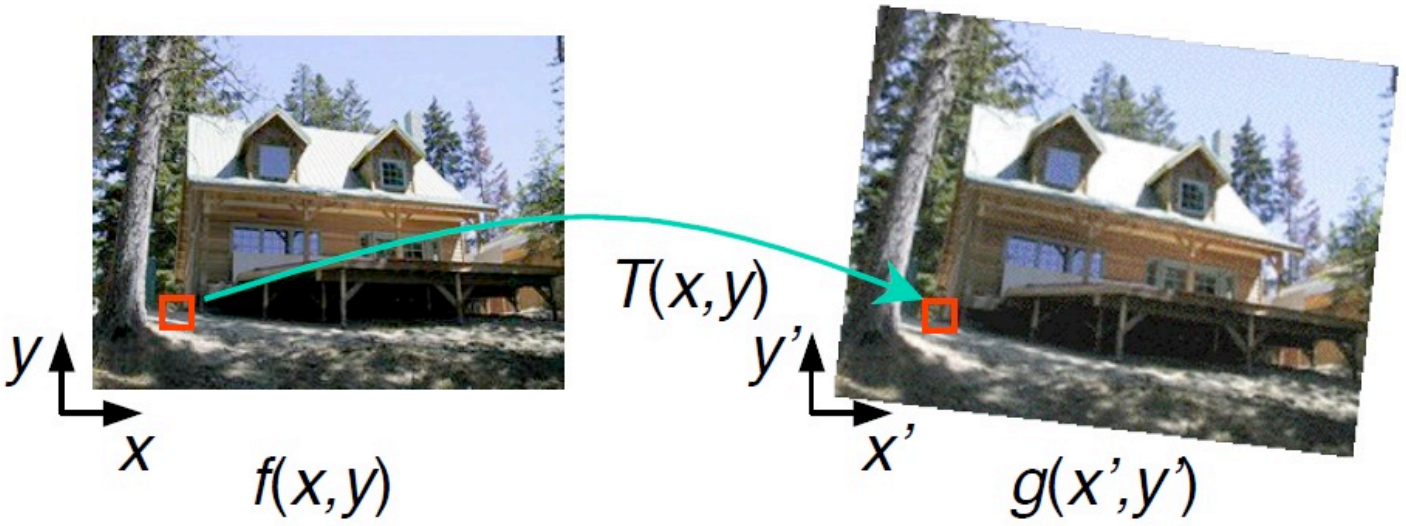
  p' = M*p

Image Warping



Given a coordinate transform (x',y') = h(x,y) and a source image f(x,y), how do we compute a transformed image g(x',y') = f(T(x,y))?

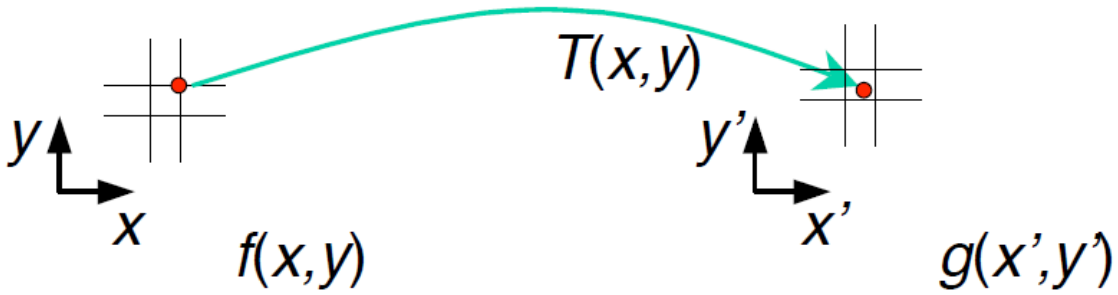
Forward Warping



Send each pixel f(x,y) to its corresponding location (x',y') = T(x,y) in the second image

Q: what if the pixel lands "between" two pixels?

Forward Warping

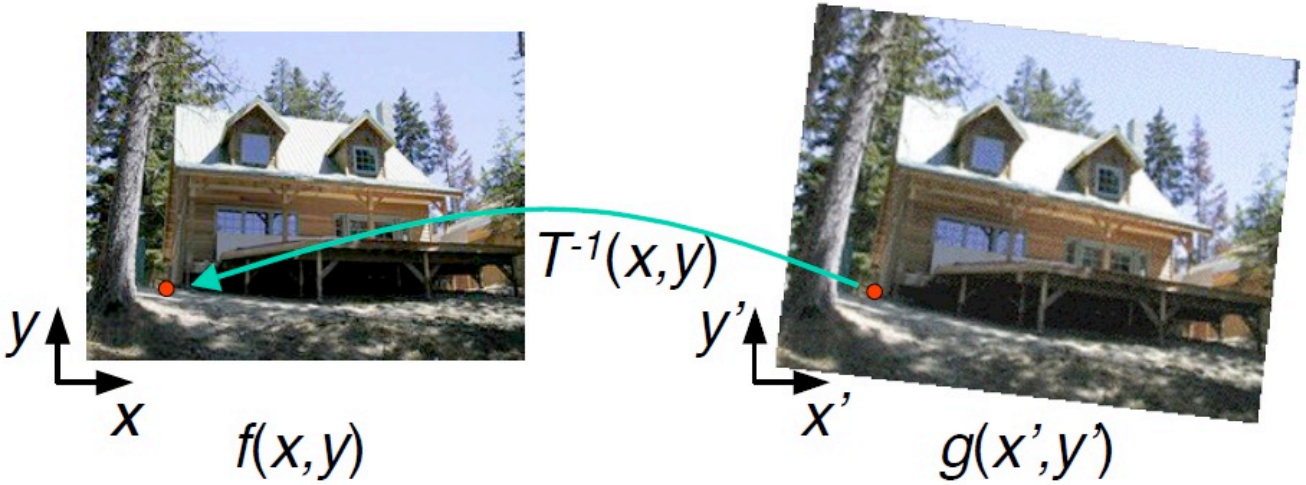


Send each pixel f(x,y) to its corresponding location (x',y') = T(x,y) in the second image

Q: what if the pixel lands "between" two pixels?

A: Distribute color among neighboring pixels (splatting)

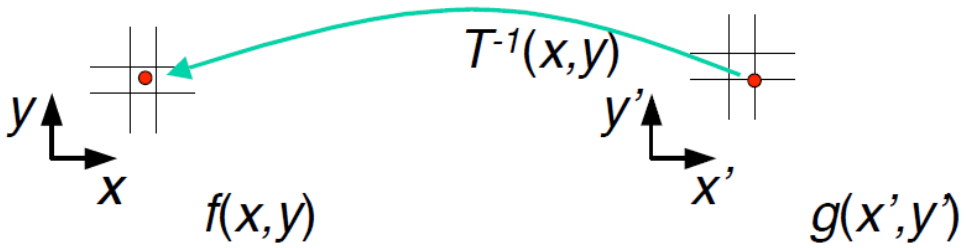
Inverse Warping



- Get each pixel g(x',y') from its corresponding location (x,y) = T^{-1}(x',y') in the first image

- Q: what if pixel comes from "between" two pixels?

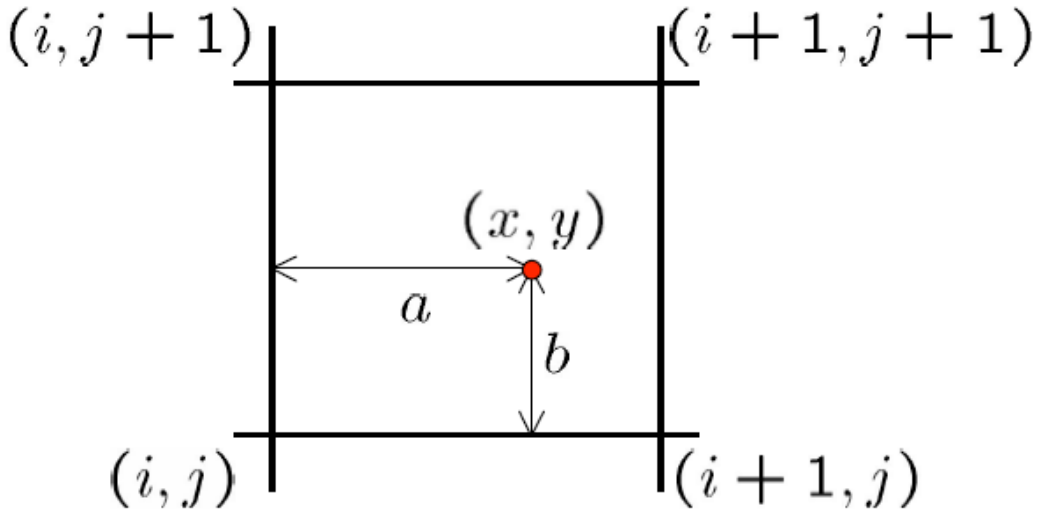
Inverse Warping



- Get each pixel g(x',y') from its corresponding location (x,y) = T^{-1}(x',y') in the first image

- Q: what if pixel comes from "between" two pixels?
- A: interpolate color value from neighbors
  - Bilinear interpolation typically used

Bilinear Interpolation



$$
\begin{aligned}
f(x,y) = \quad & (1-a)(1-b) \quad f[i,j] \\
& +a(1-b) \quad f[i+1,j] \\
& +ab \quad f[i+1,j+1] \\
& +(1-a)b \quad f[i,j+1]
\end{aligned}
$$

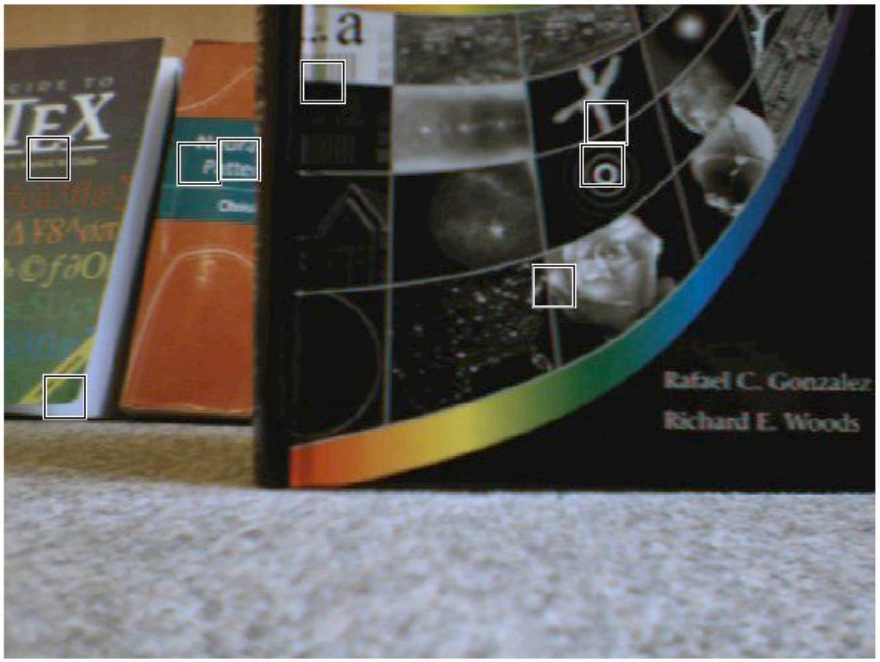
Feature Matching

- Given a feature in one image, how to find the best match in another image?
  - Assumption images are consecutive frames in video
- So far we have searched for best match
  - By testing all possible translations by integer number of pixels (template matching)
  - By comparing rotation-invariant descriptors (SIFT)

Camera Motion

I

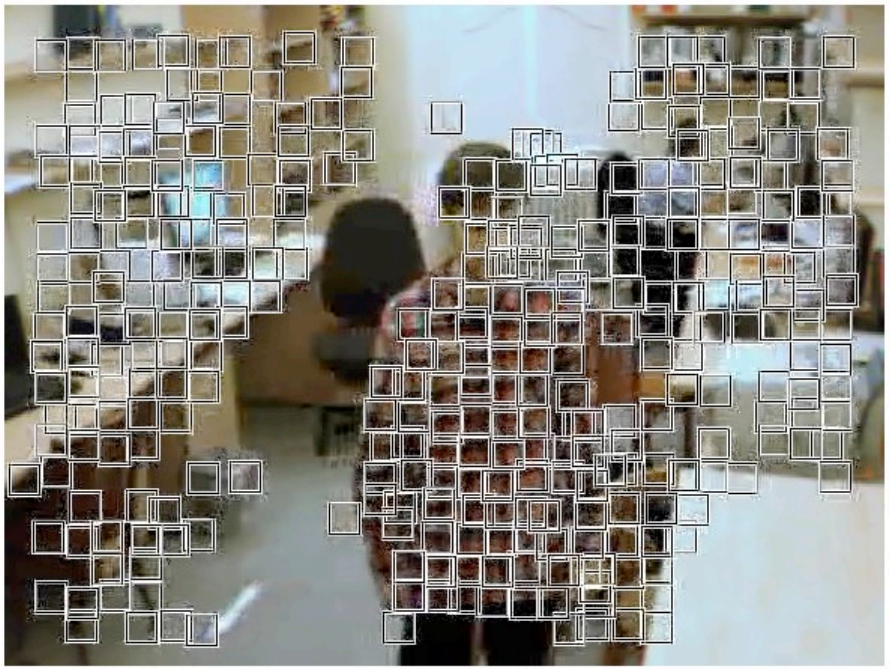
J



Object Motion

I

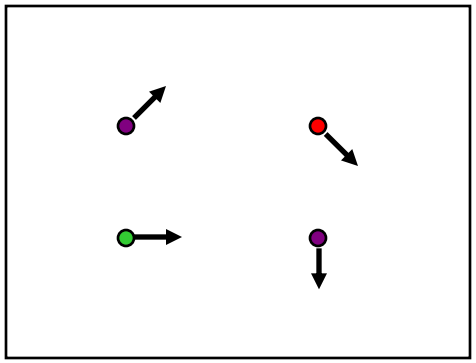
J



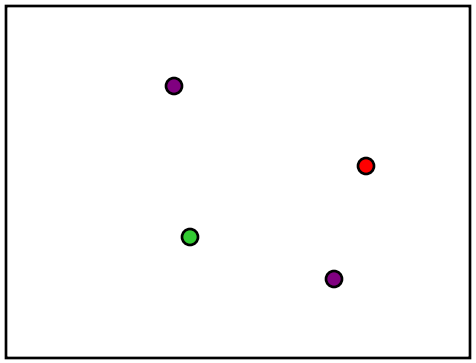
Feature Tracking

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

Feature Tracking



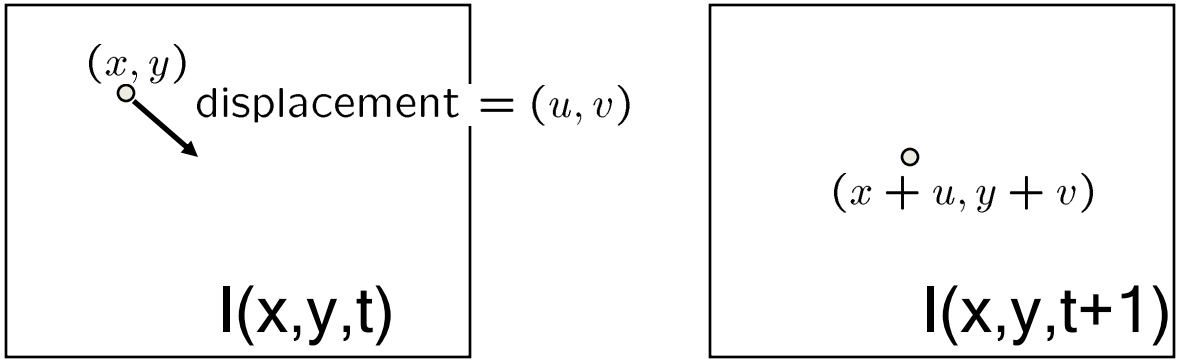
$I(x,y,t)$



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

- 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

The Brightness Constancy Constraint



- Brightness Constancy Equation:

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

Take Taylor expansion of I(x+u, y+v, t+1) at (x,y,t) to linearize the right side:

Image derivative along x Difference over frames

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

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

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

The Brightness Constancy Constraint

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

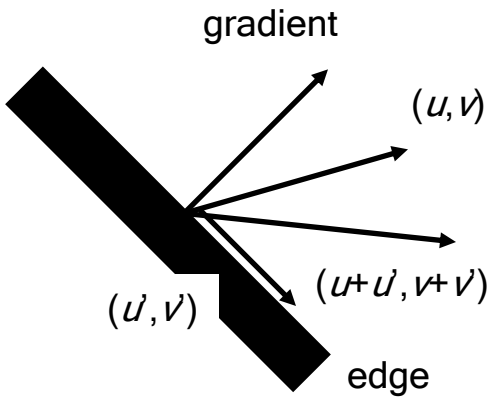
$$\nabla I \cdot \begin{bmatrix} u & v \end{bmatrix}^T + I_t = 0$$

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

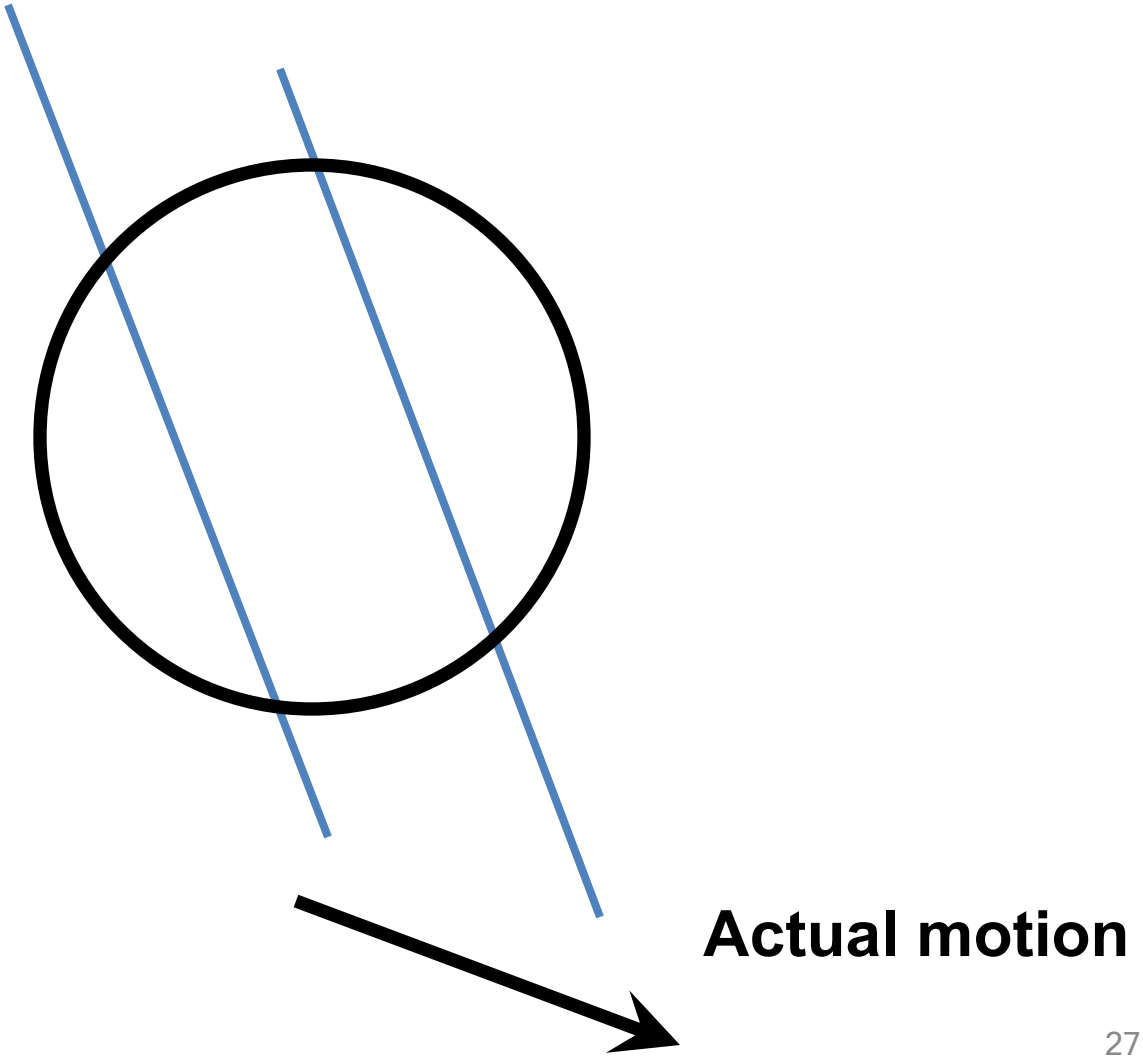
The component of the motion perpendicular to the gradient (i.e., parallel to the edge) cannot be measured

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

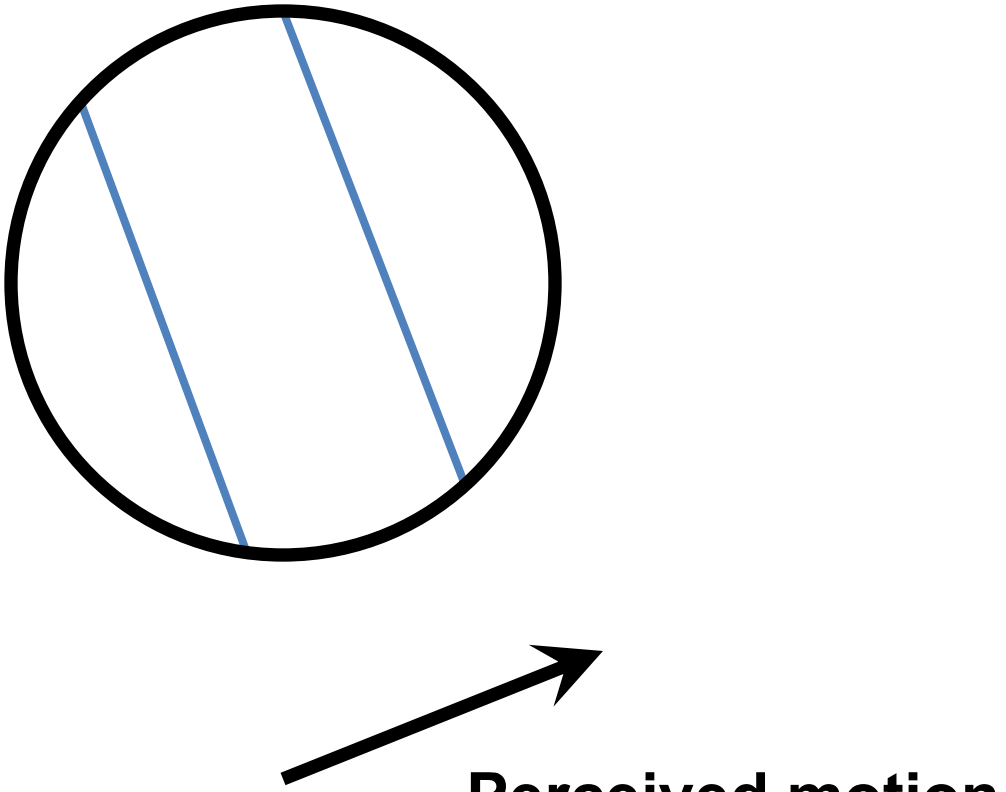
$$\nabla I \cdot \begin{bmatrix} u' & v' \end{bmatrix}^T = 0$$



The Aperture Problem



The Aperture Problem



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

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

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

Solving the Ambiguity…

- Least squares problem:

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

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

Matching Patches across Images

- 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 $(A^T A)\ d = A^T b$

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

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

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

Conditions 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? I.e., what are good points to track?

- $\mathbf{A^T A}$ 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 = largest eigenvalue)

Does this remind you of anything?

Criteria for Harris corner detector

Low-texture region

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

– gradients have small magnitude

– small λ_1, small λ_2

Edge

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

– gradients very large or very small

– large λ_1, small λ_2

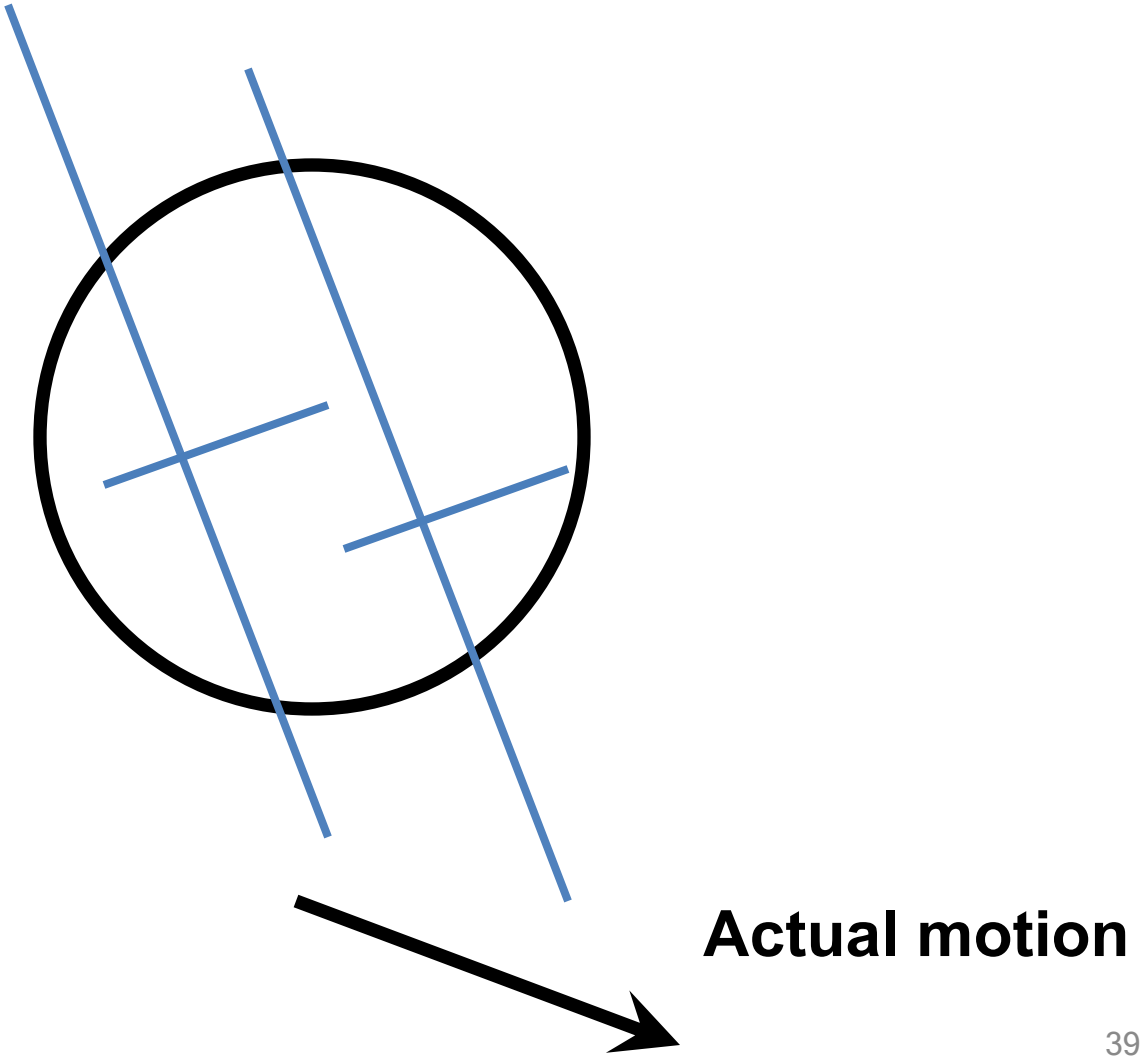
High-texture Region

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

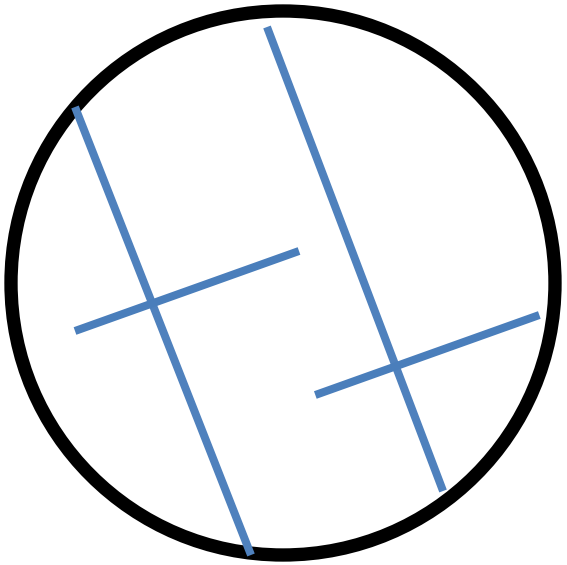
– gradients are different, large magnitudes

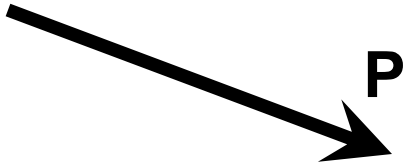
– large λ_1, large λ_2

The Aperture Problem Resolved



The Aperture Problem Resolved





Perceived motion

Dealing with Larger Motion: Iterative Refinement

Original (x,y) position

$I_t = I(x', y', t+1) - I(x, y, t)$

1. Initialize (x',y') = (x,y)
2. Compute (u,v) by

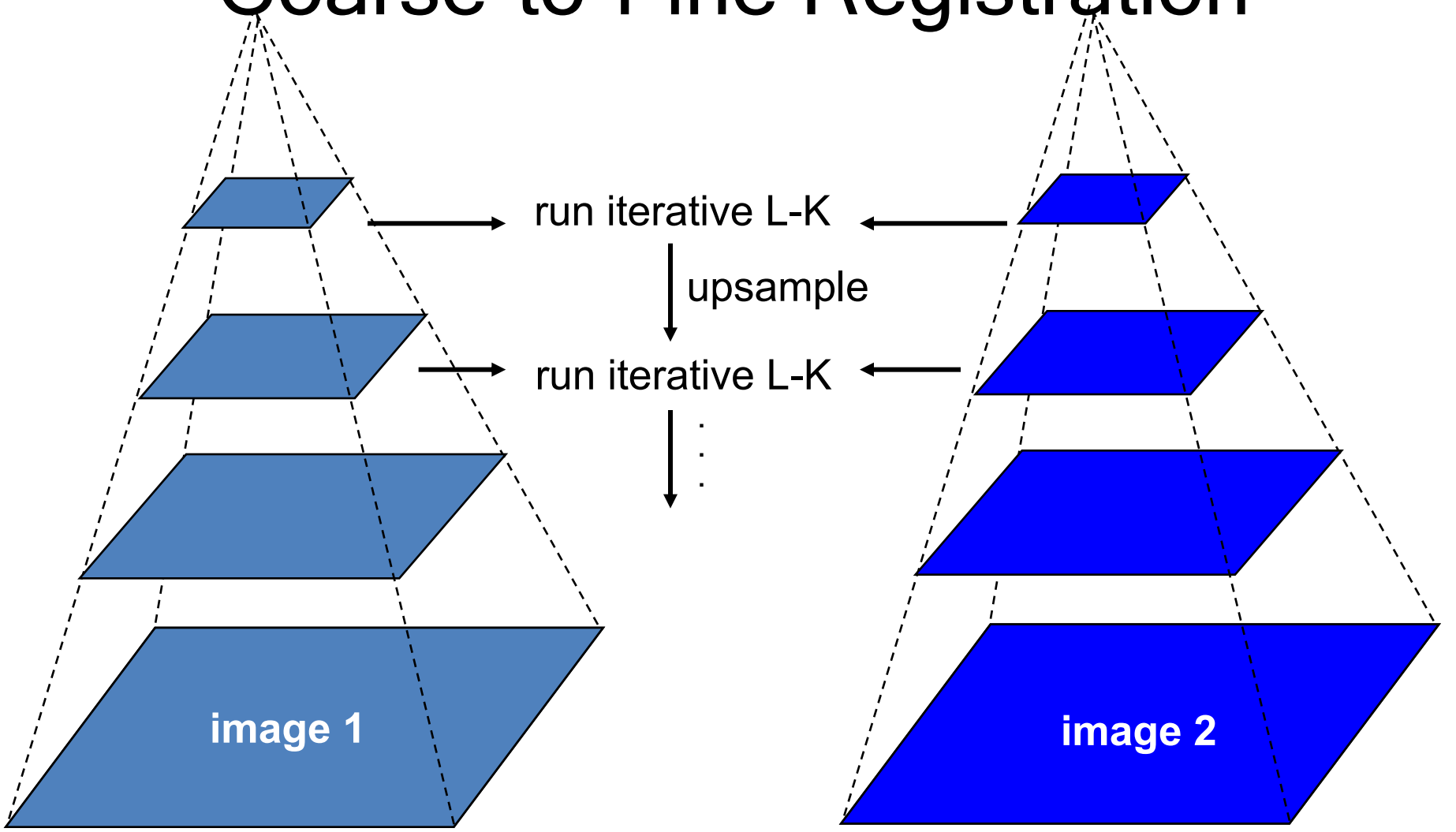
$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

2nd moment matrix for feature patch in first image

displacement

3. Shift window by (u, v): `x'=x'+u; y'=y'+v;`
4. Recalculate I_t
5. Repeat steps 2-4 until change is small
   - Use interpolation for subpixel values

Dealing with Larger Motion: Coarse-to-Fine Registration



Gaussian pyramid of image 1 (t) **Gaussian pyramid of image 2 (t+1)**

Shi-Tomasi Feature Tracker

- Find good features using eigenvalues of second-moment matrix (e.g., Harris detector or threshold on the smallest eigenvalue)
  - Key idea: "good" features to track are the ones whose motion can be estimated reliably

- Track from frame to frame with Lucas-Kanade
  - This amounts to assuming a translation model for frame-to-frame feature movement

- Check consistency of tracks by *affine* registration to the first observed instance of the feature
  - Affine model is more accurate for larger displacements
  - Comparing to the first frame helps to minimize drift

Summary of KLT tracking

- Find a good point to track (Harris corners)
- Use intensity second moment matrix and difference across frames to find displacement
- Iterate and use coarse-to-fine search to deal with larger movements
- When creating long tracks, check appearance of registered patch against appearance of initial patch to find points that have drifted

Implementation issues

- Window size
  - Small window more sensitive to noise and may miss larger motions (without pyramid)
  - Large window more likely to cross an occlusion boundary (and it's slower)

- Weighting the window
  - Common to apply weights so that center matters more (e.g., with Gaussian)

Why not just do local template matching?

- Slow (need to check more locations)
- Does not give subpixel alignment (or becomes much slower)
  - Even pixel alignment may not be good enough to prevent drift
- May be useful as a step in tracking if there are large motions

Kanade-Lucas-Tomasi Tracking

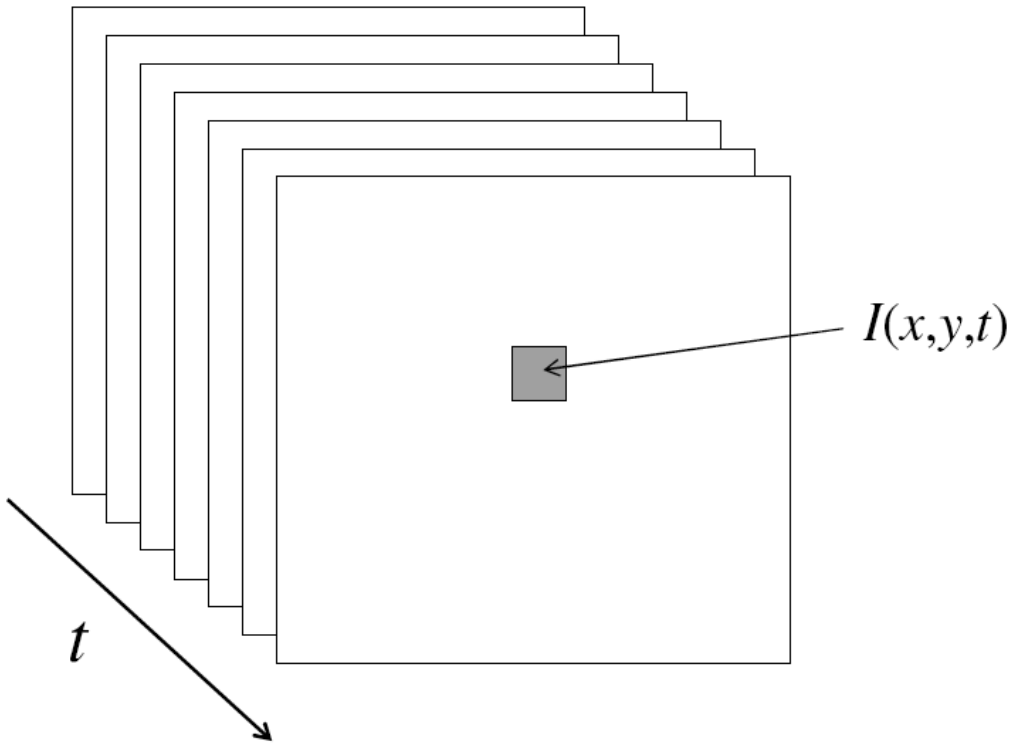
- Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In Proceedings of the 7th International Conference on Artificial Intelligence, pages 674–679, August 1981.
- Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, April 1991.
- Jianbo Shi and Carlo Tomasi. Good features to track. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 593–600, 1994.
- Code: http://www.ces.clemson.edu/~stb/klt/

Dense Motion Estimation

Based on slides by K. Grauman

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)

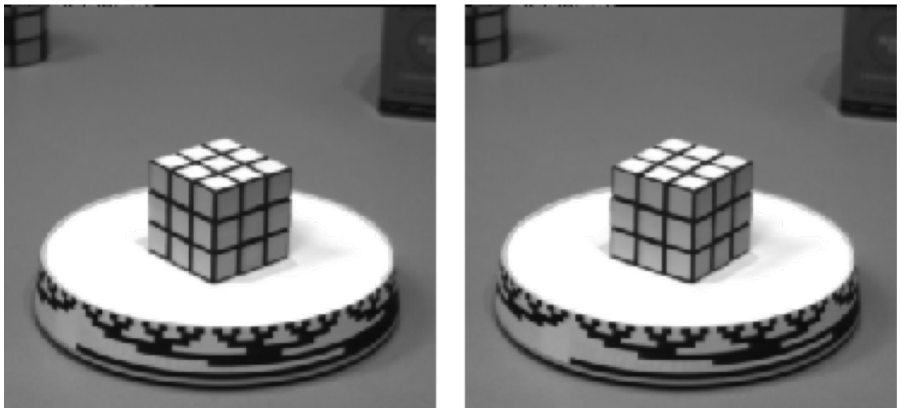


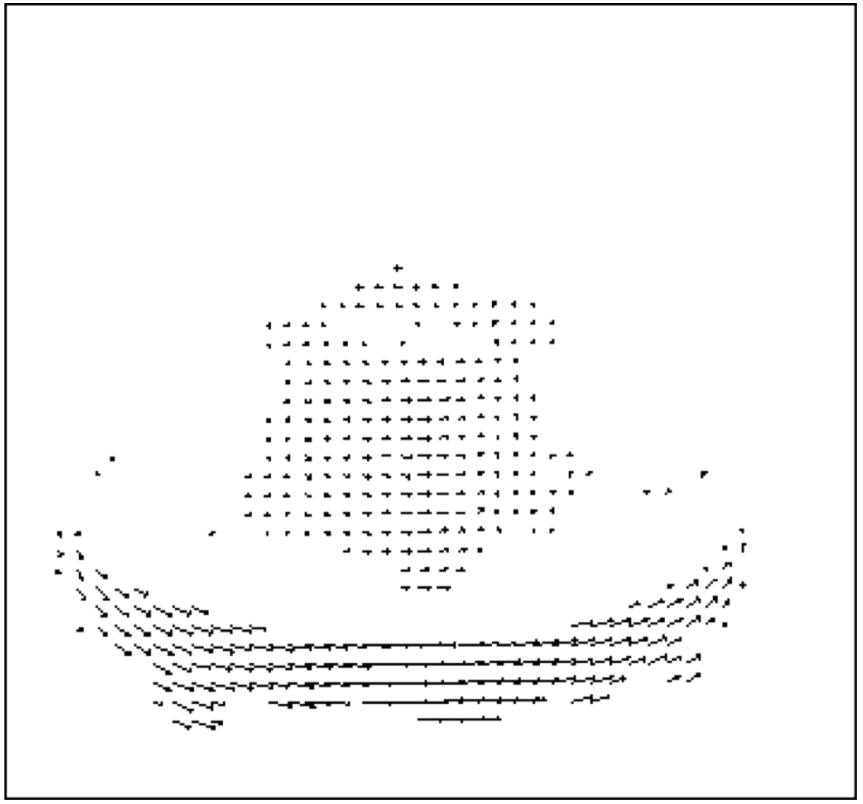
Uses of motion

- Estimating 3D structure
- Segmenting objects based on motion cues
- Learning dynamical models
- Recognizing events and activities
- Improving video quality (motion stabilization)

Motion field

- The motion field is the projection of the 3D scene motion into the image





Motion field + camera motion

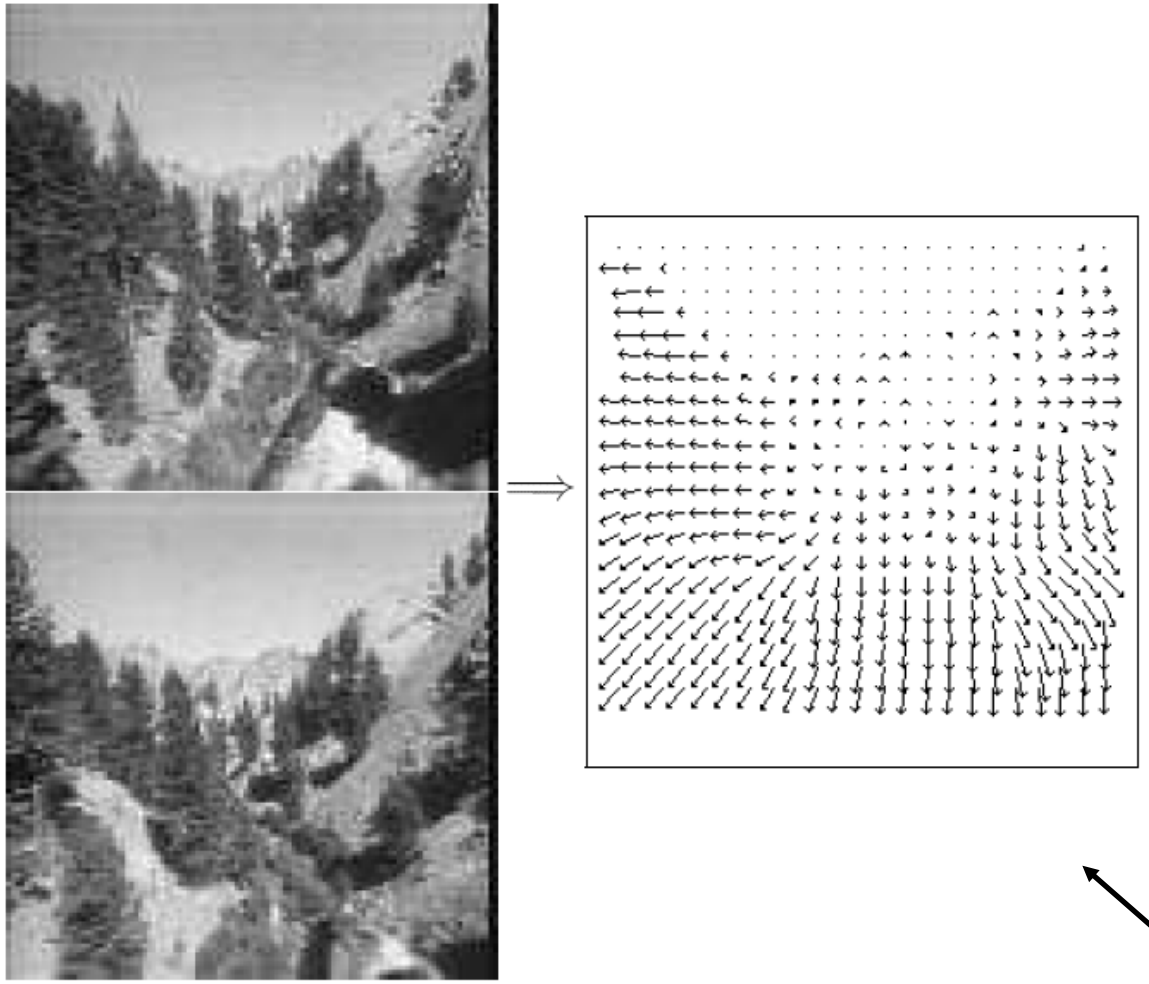


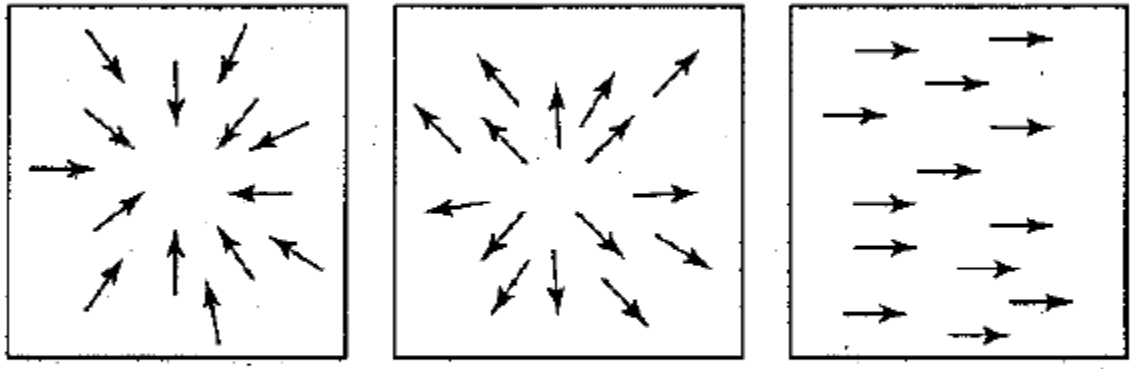
Figure 1.2: Two images taken from a helicopter flying through a canyon and the computed optical flow field.

Figure from Michael Black, Ph.D. Thesis

Length of flow vectors inversely proportional to depth Z of 3d point

points closer to the camera move more quickly across the image plane

Motion field + camera motion



Zoom out **Zoom in** **Pan right to left**

Motion estimation techniques

- Direct methods
  - Directly recover image motion at each pixel from spatio-temporal image brightness variations
  - Dense motion fields, but sensitive to appearance variations
  - Suitable for video and when image motion is small
- Feature-based methods
  - Extract visual features (corners, textured areas) and track them over multiple frames
  - Sparse motion fields, but more robust tracking
  - Suitable when image motion is large (10s of pixels)

Optical Flow

- Definition: optical flow is the *apparent* motion of brightness patterns in the image
- Ideally, optical flow would be the same as the motion field
- Have to be careful: apparent motion can be caused by lighting changes without any actual motion

Apparent motion != motion field

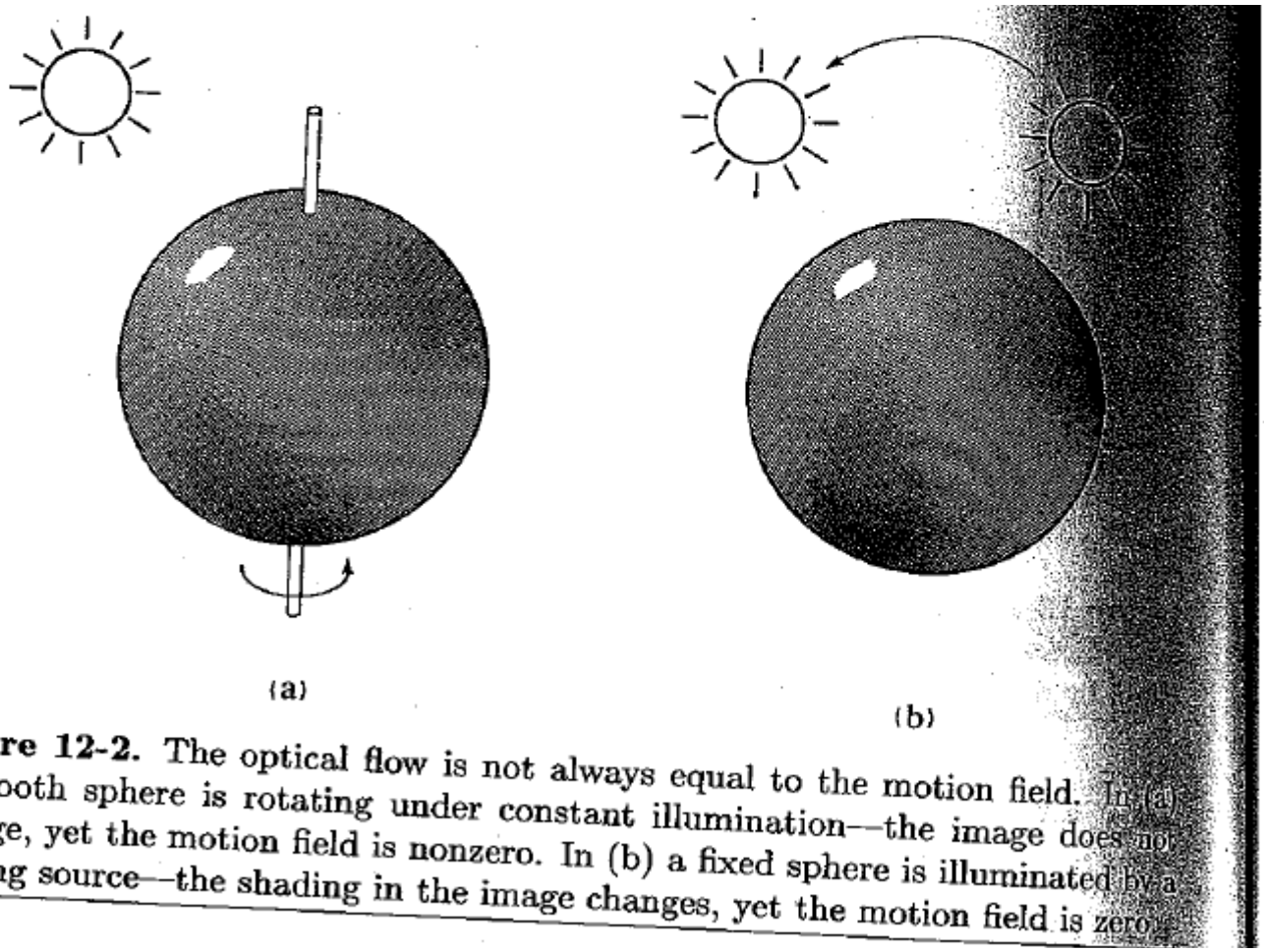


Figure 12-2. The optical flow is not always equal to the motion field. In (a) a smooth sphere is rotating under constant illumination—the image does not change, yet the motion field is nonzero. In (b) a fixed sphere is illuminated by a moving source—the shading in the image changes, yet the motion field is zero.

Figure from Horn book

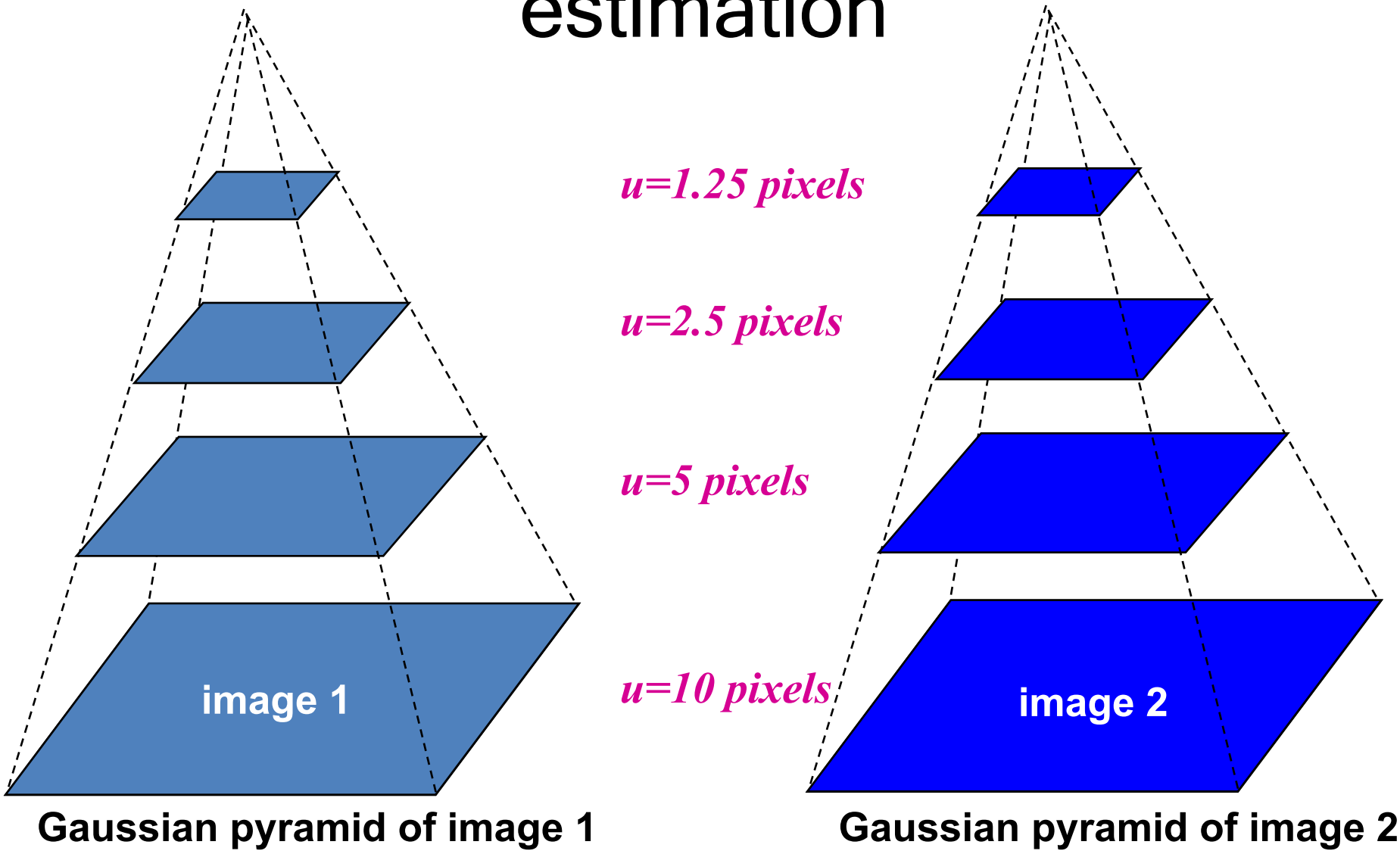
Lucas-Kanade Optical Flow

- Same as Lucas-Kanade feature tracking, but for each pixel
  - As we saw, works better for textured pixels
- Operations can be done one frame at a time, rather than pixel by pixel
  - Efficient

Iterative Refinement

- Iterative Lukas-Kanade Algorithm
  1. Estimate displacement at each pixel by solving Lucas-Kanade equations
  2. Warp I(t) towards I(t+1) using the estimated flow field
     - Basically, just interpolation
  3. Repeat until convergence

Coarse-to-fine optical flow estimation



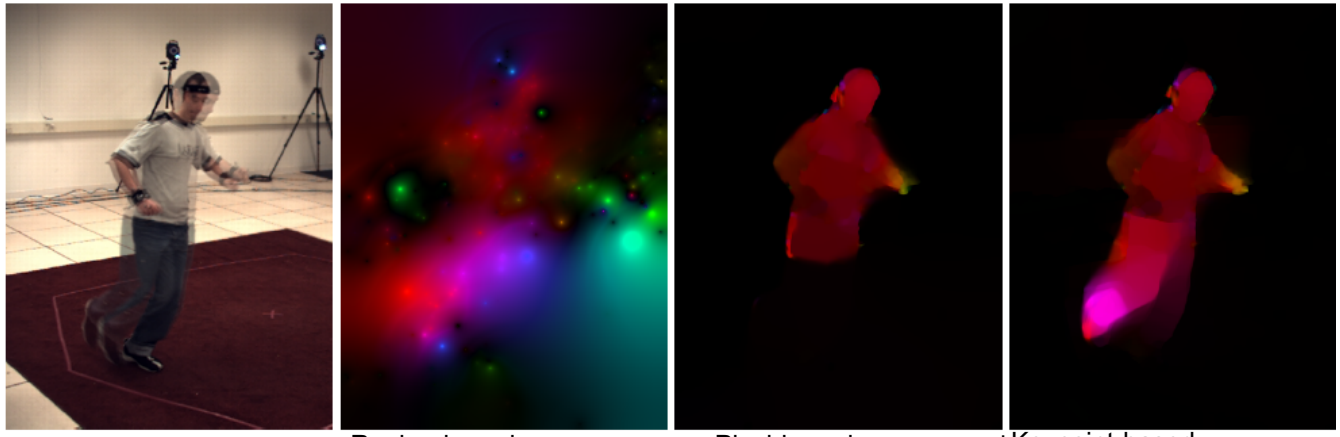
Errors in Lucas-Kanade

- The motion is large
  - Possible Fix: Keypoint matching
- A point does not move like its neighbors
  - Possible Fix: Region-based matching
- Brightness constancy does not hold
  - Possible Fix: Gradient constancy

State-of-the-art optical flow

Start with something similar to Lucas-Kanade

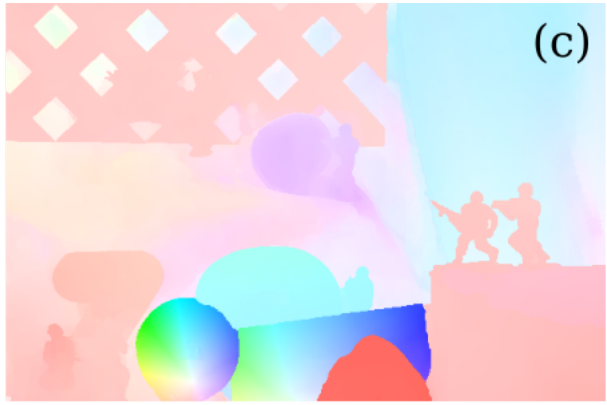
+ gradient constancy
+ energy minimization with smoothing term
+ region matching
+ keypoint matching (long-range)

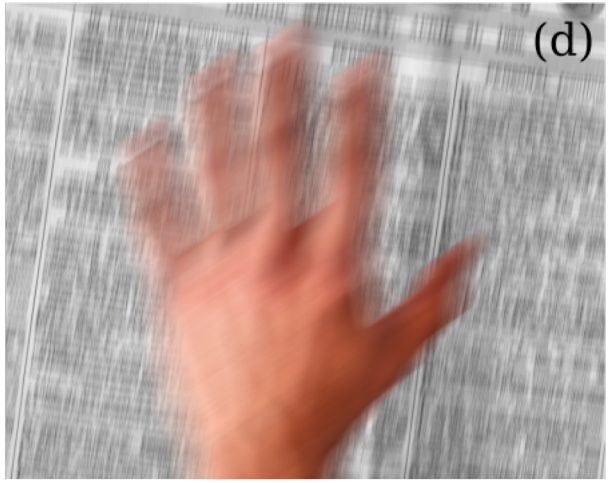


Region-based +Pixel-based +Keypoint-based

Large displacement optical flow, Brox et al., CVPR 2009

State-of-the-art optical flow





Two examples of applications of layered models. Top row: From a sequence of images (a), a layered model can extract the layer assignments (b) and compute highly accurate flow (c), especially at motion boundaries. Bottom row: Using a layered model, a motion blurred sequence (d) can be decomposed into foreground (e) and background (f), which can then be separately deblurred.

Image by J. Wulff, D. Sun and M. Black

Grouping and Segmentation

Based on slides by D. Hoiem

Gestalt psychology or gestaltism

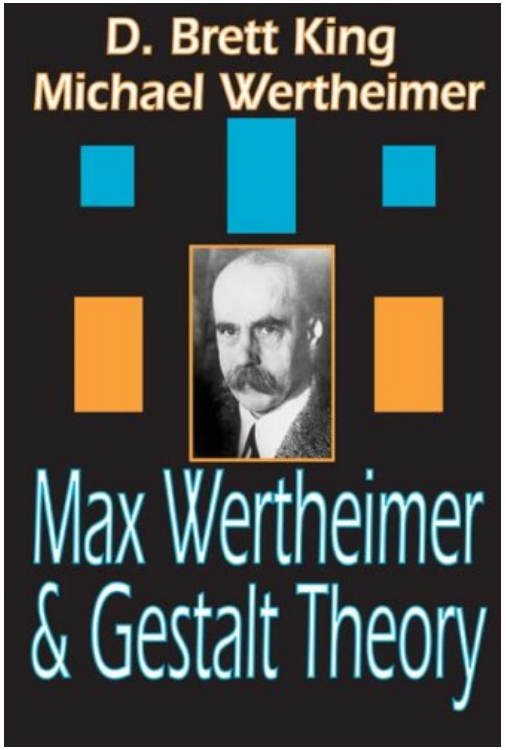
German: *Gestalt* - "form" or "whole"

Berlin School, early 20th century

Kurt Koffka, Max Wertheimer, and Wolfgang Köhler

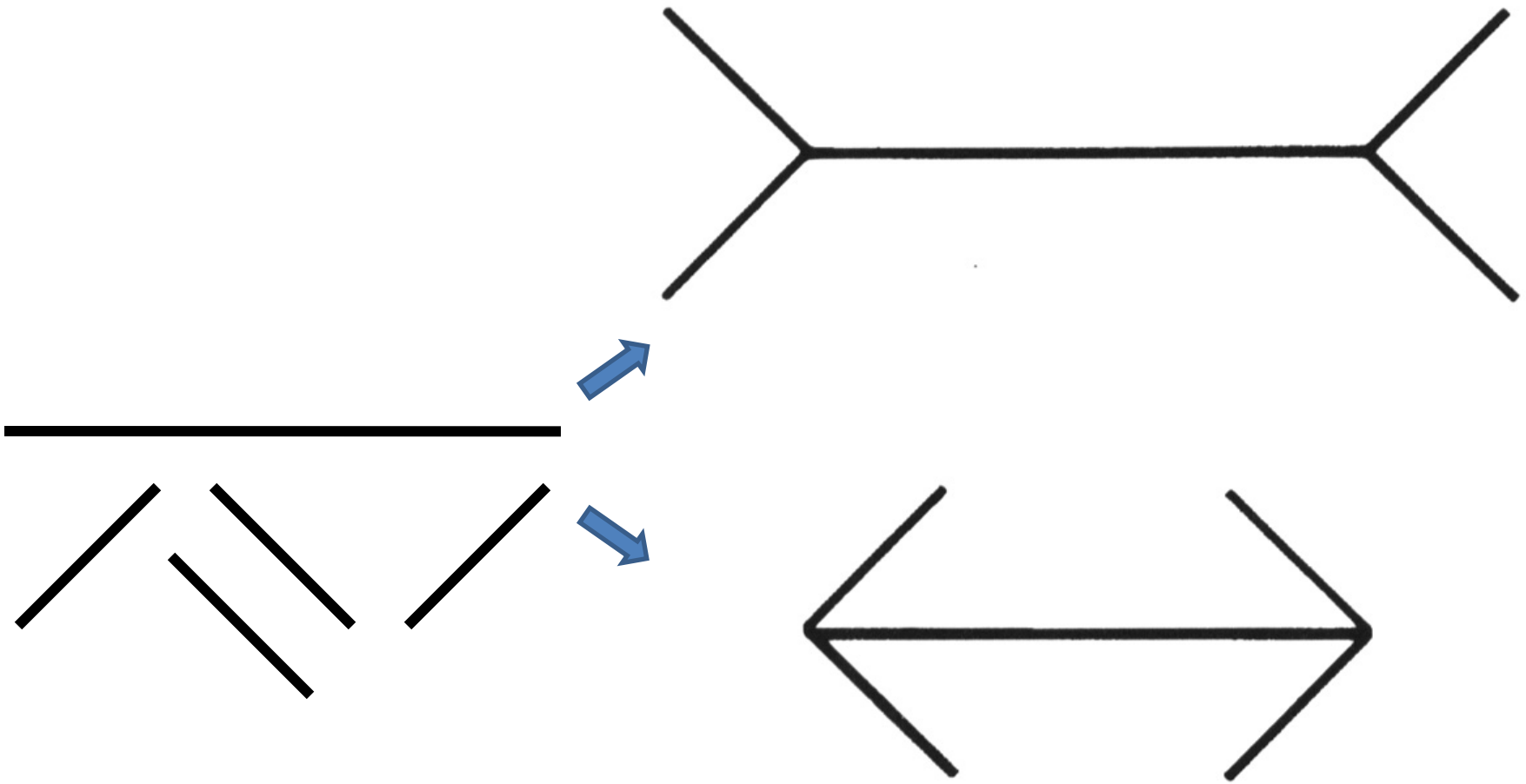
View of brain:

- whole is more than the sum of its parts
- holistic
- parallel
- analog
- self-organizing tendencies

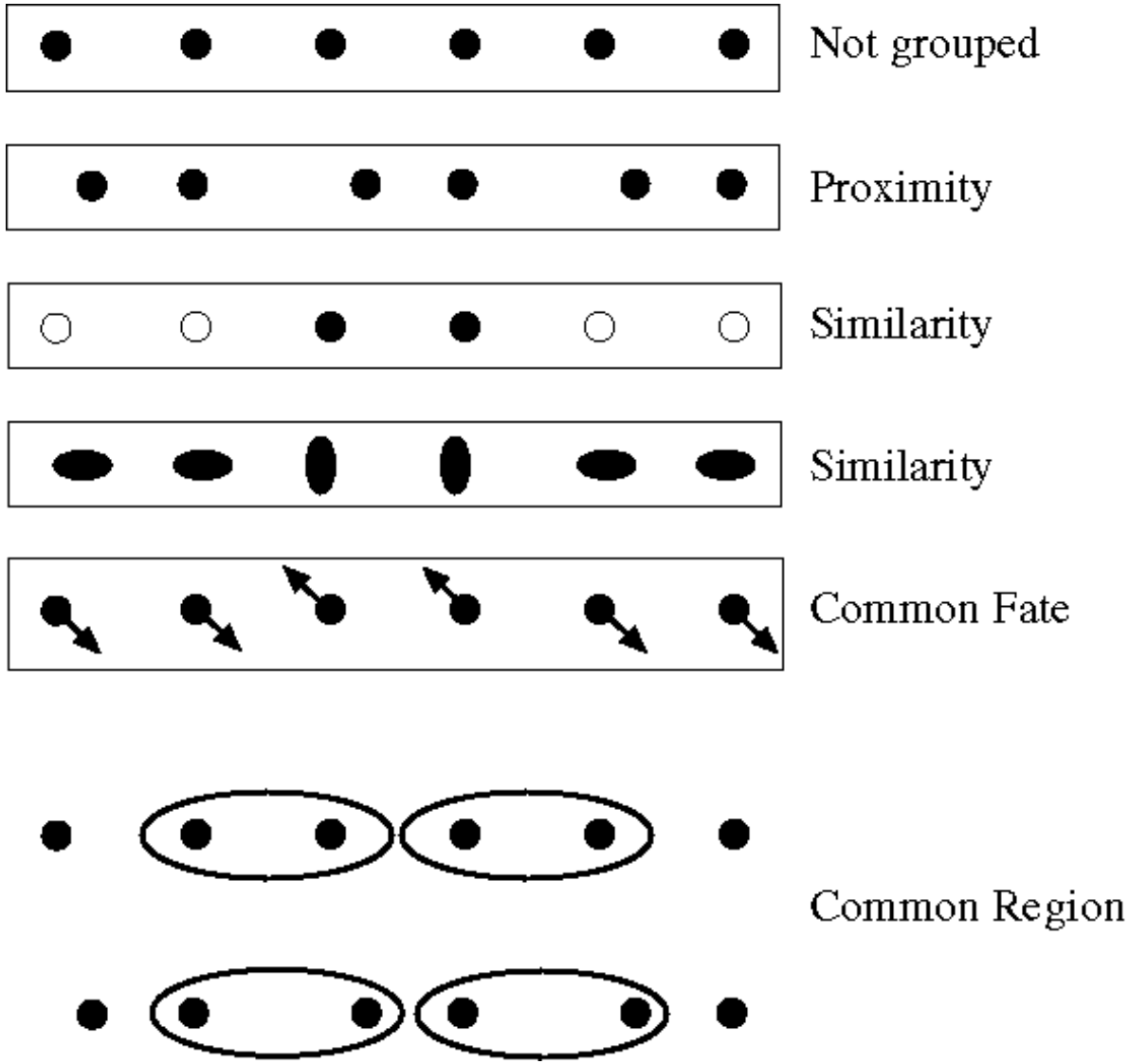


Slide from S. Saverese

Muller-Lyer Illusion

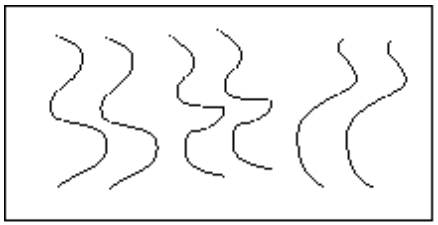


Principles of perceptual organization

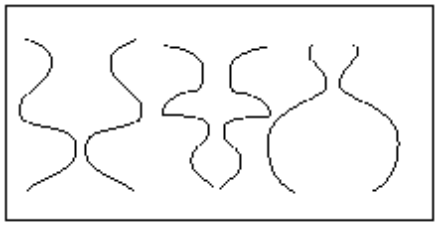


From Steve Lehar: The Constructive Aspect of Visual Perception

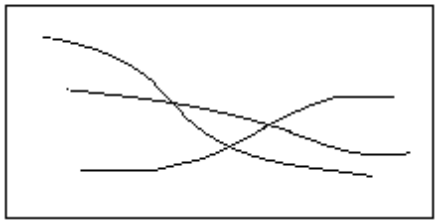
Principles of perceptual organization



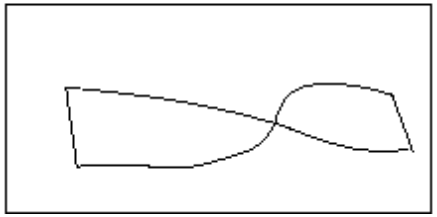
Parallelism



Symmetry

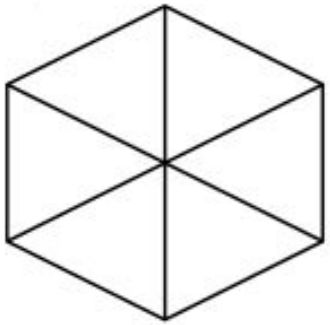


Continuity

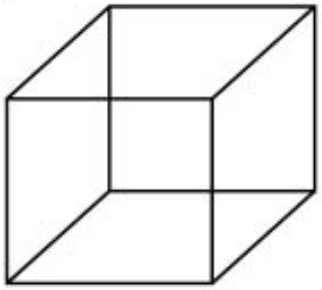


Closure

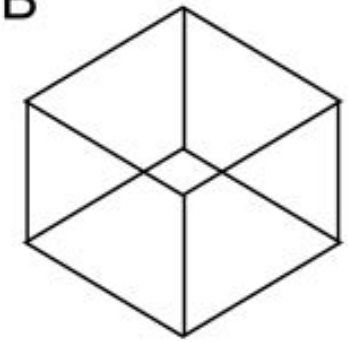
Gestaltists do not believe in coincidence



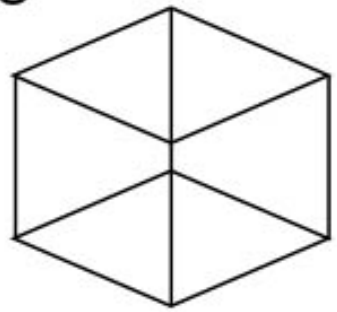
A



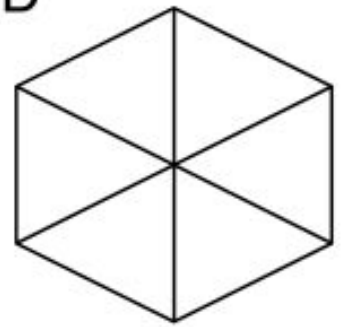
B



C

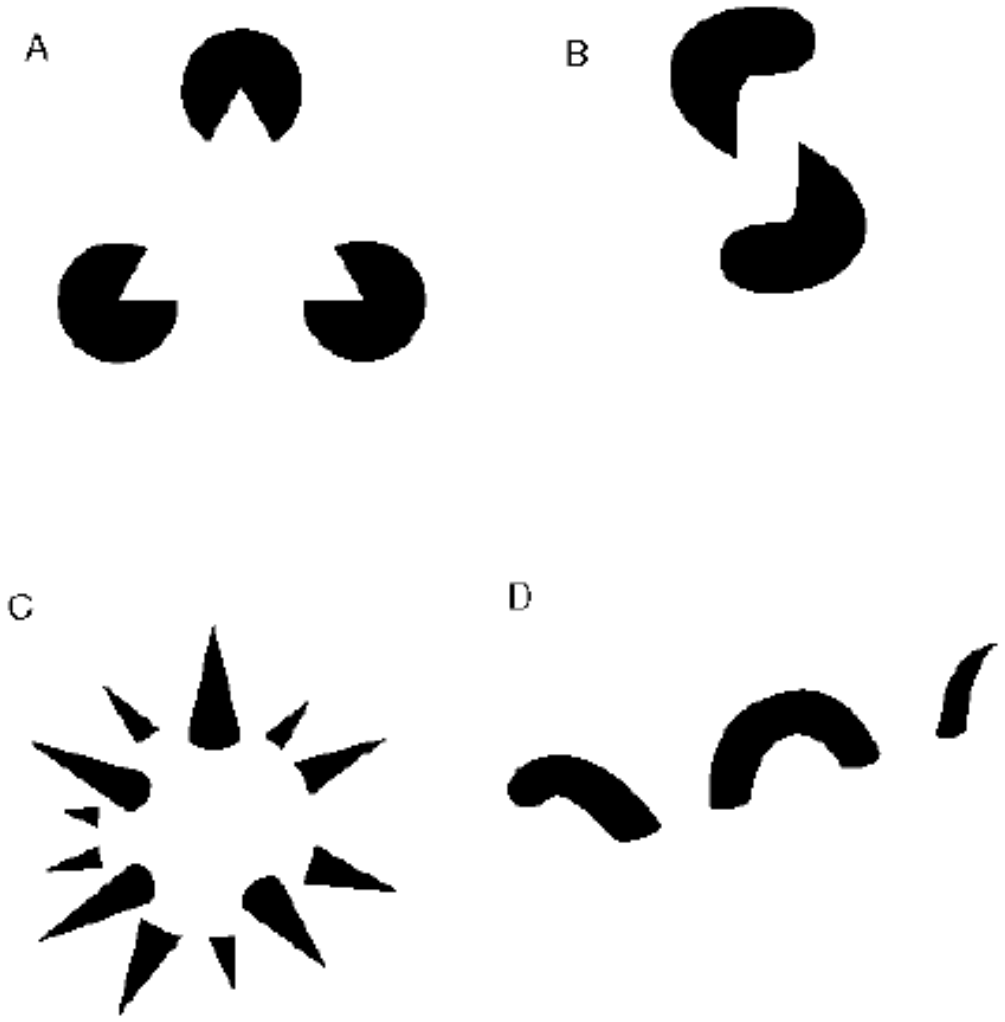


D



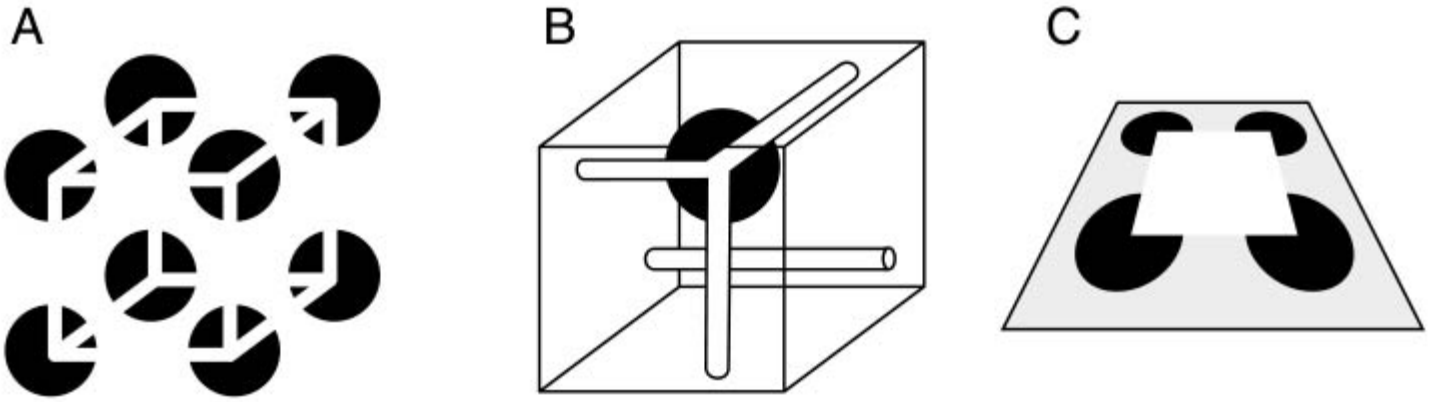
Emergence

Grouping by invisible completion



From Steve Lehar: The Constructive Aspect of Visual Perception

Grouping involves global interpretation



From Steve Lehar: The Constructive Aspect of Visual Perception

Gestalt cues

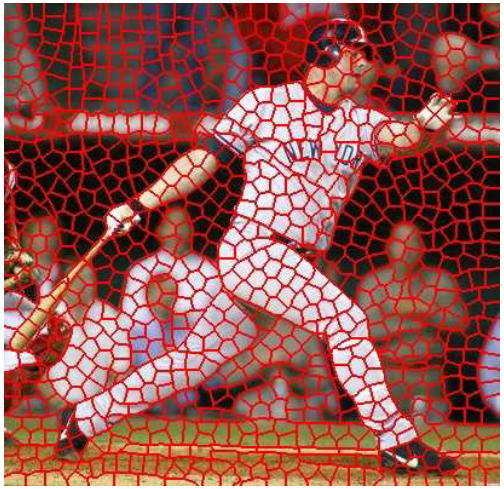
- Good intuition and basic principles for grouping
- Basis for many ideas in segmentation and occlusion reasoning
- Some (e.g., symmetry) are difficult to implement in practice

Image segmentation

Goal: Group pixels into meaningful or perceptually similar regions

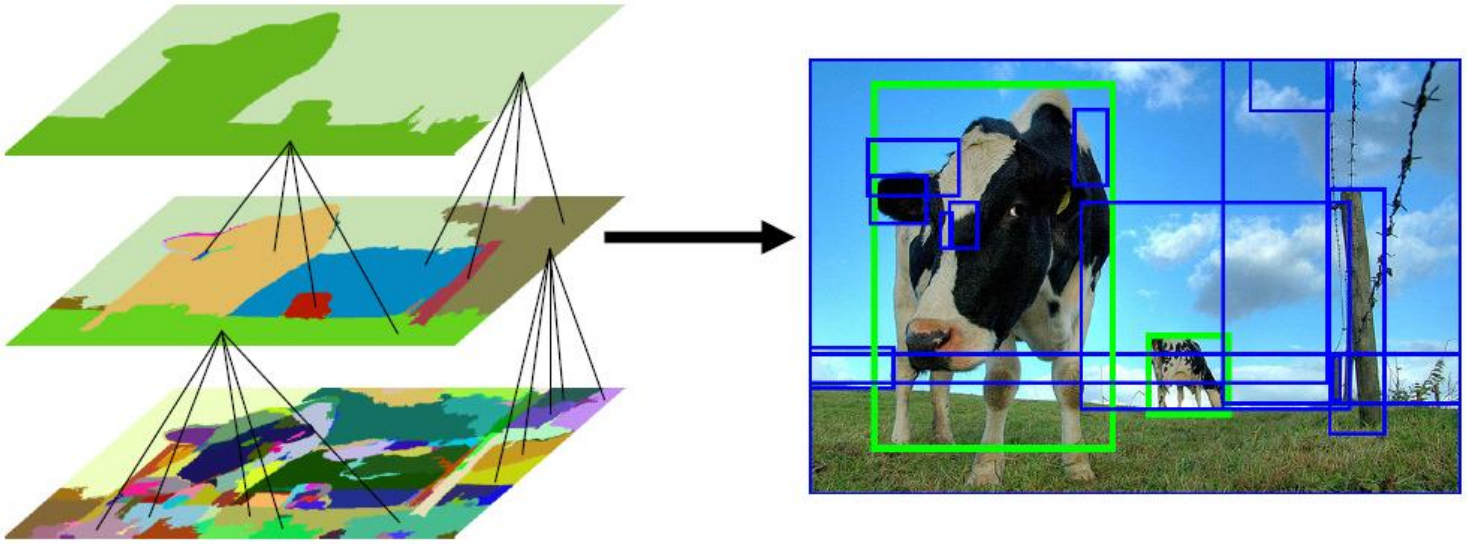
Segmentation for efficiency: “superpixels”

[Felzenszwalb and Huttenlocher 2004]

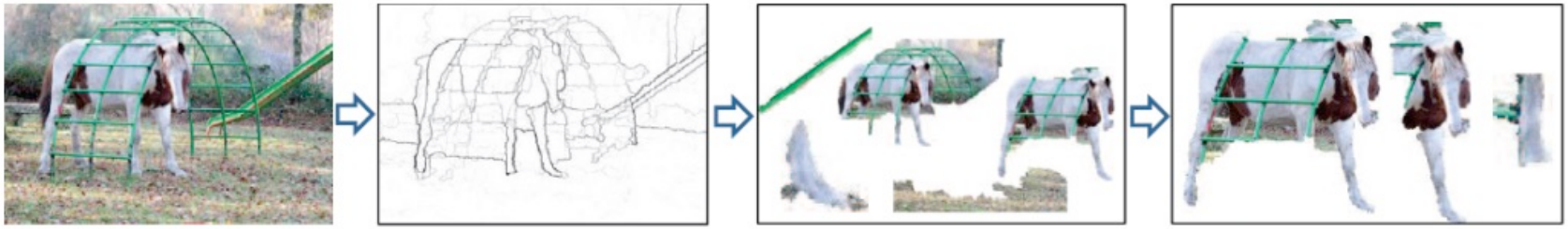


Segmentation for feature support

Segmentation for object proposals



"Selective Search" [Sande, Uijlings et al. ICCV 2011, IJCV 2013]

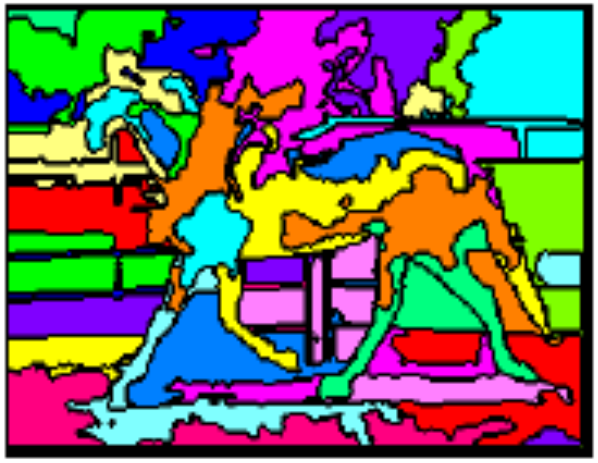


[Endres Hoiem ECCV 2010, IJCV 2014]

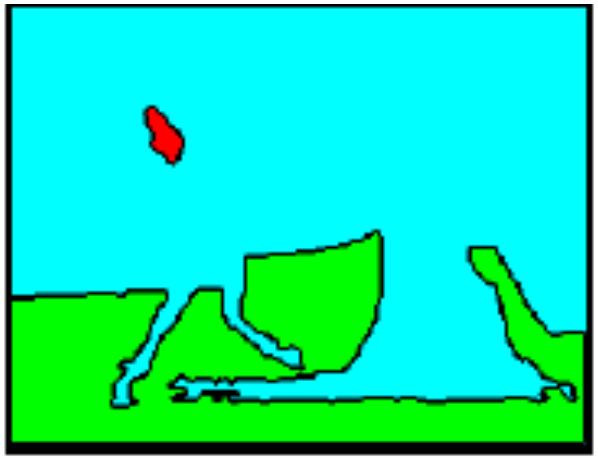
Segmentation as the Endgoal

Rother et al. 2004

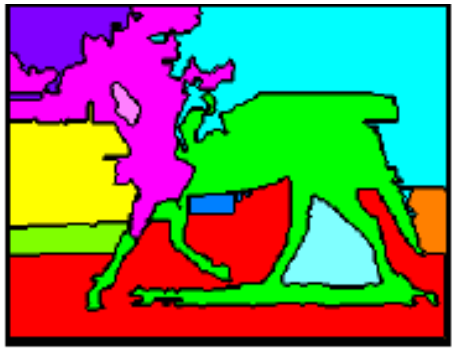
Types of segmentations

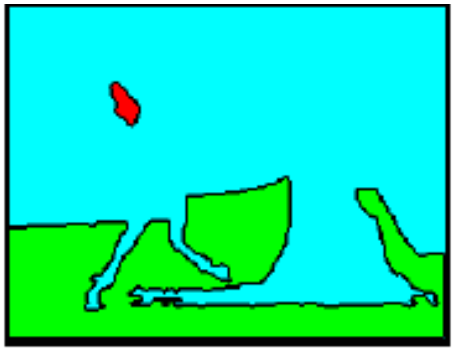


Oversegmentation



Undersegmentation

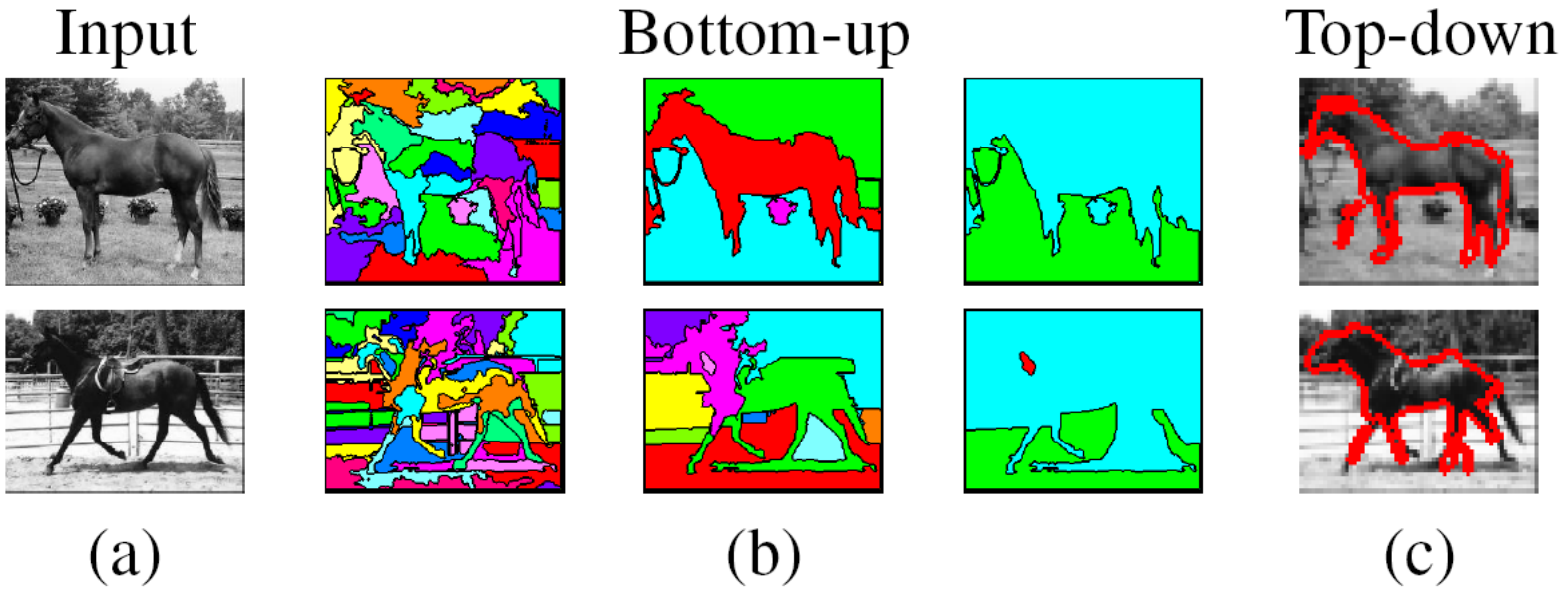




Multiple Segmentations

Major processes for segmentation

- Bottom-up: group tokens with similar features
- Top-down: group tokens that likely belong to the same object



[Levin and Weiss 2006]