CAP 4453
Robot Vision

Dr. Gonzalo Vaca-Castaño

gonzalo.vacacastano@ucf.edu

Administrative details

- Issues submitting homework

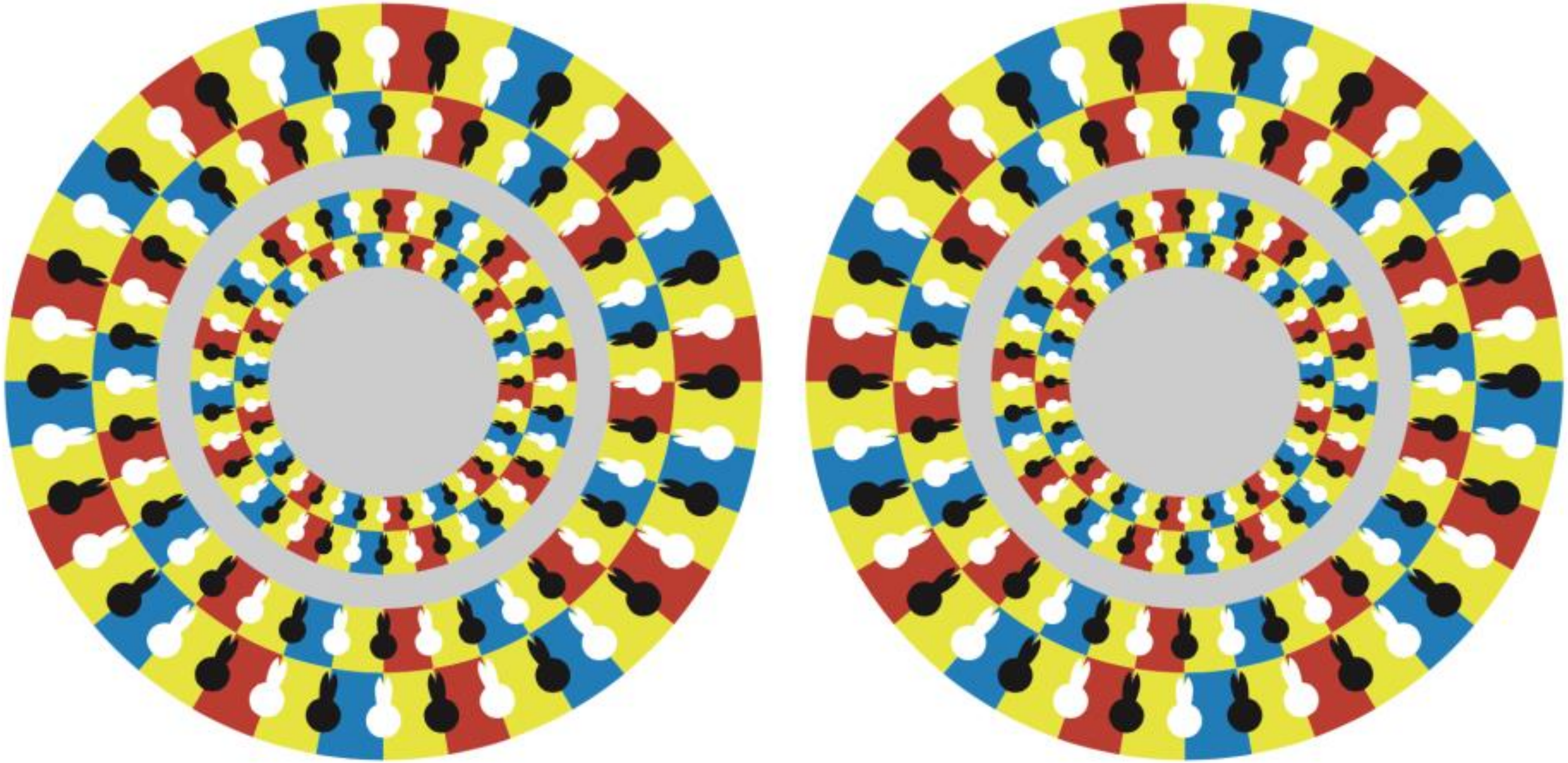
Credits

- slides comes directly from:
  - Yosesh Rawat
  - Mubarak Shah

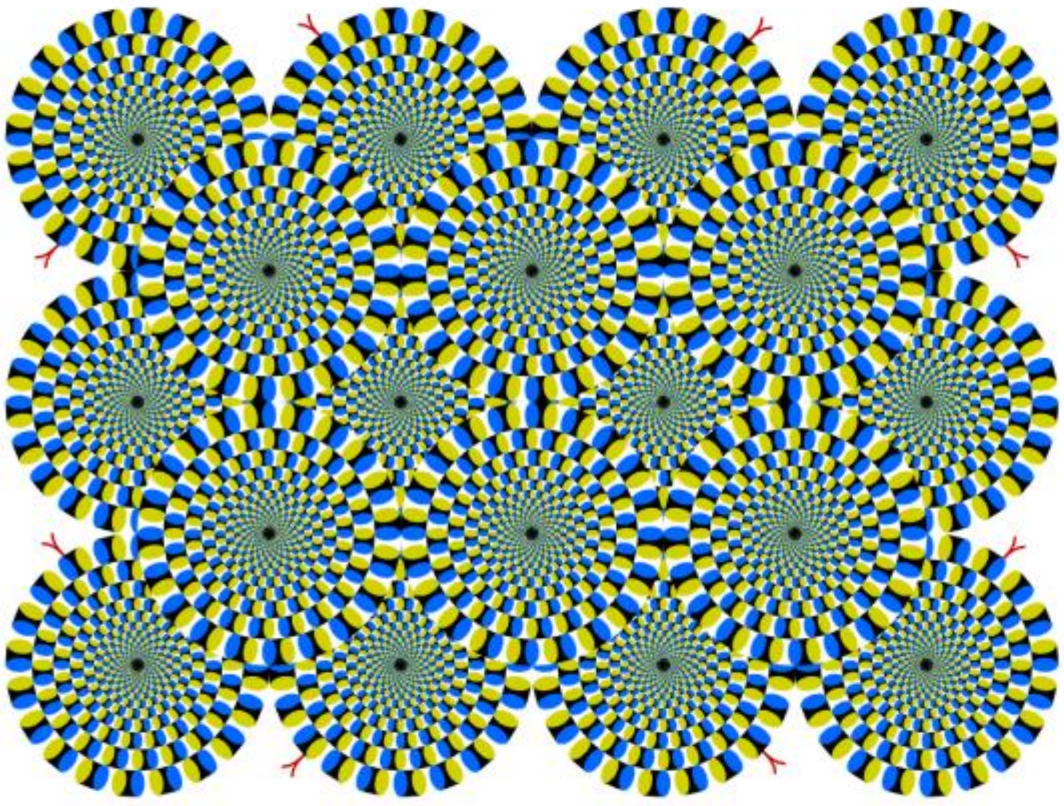
Robot Vision

15. Optical Flow

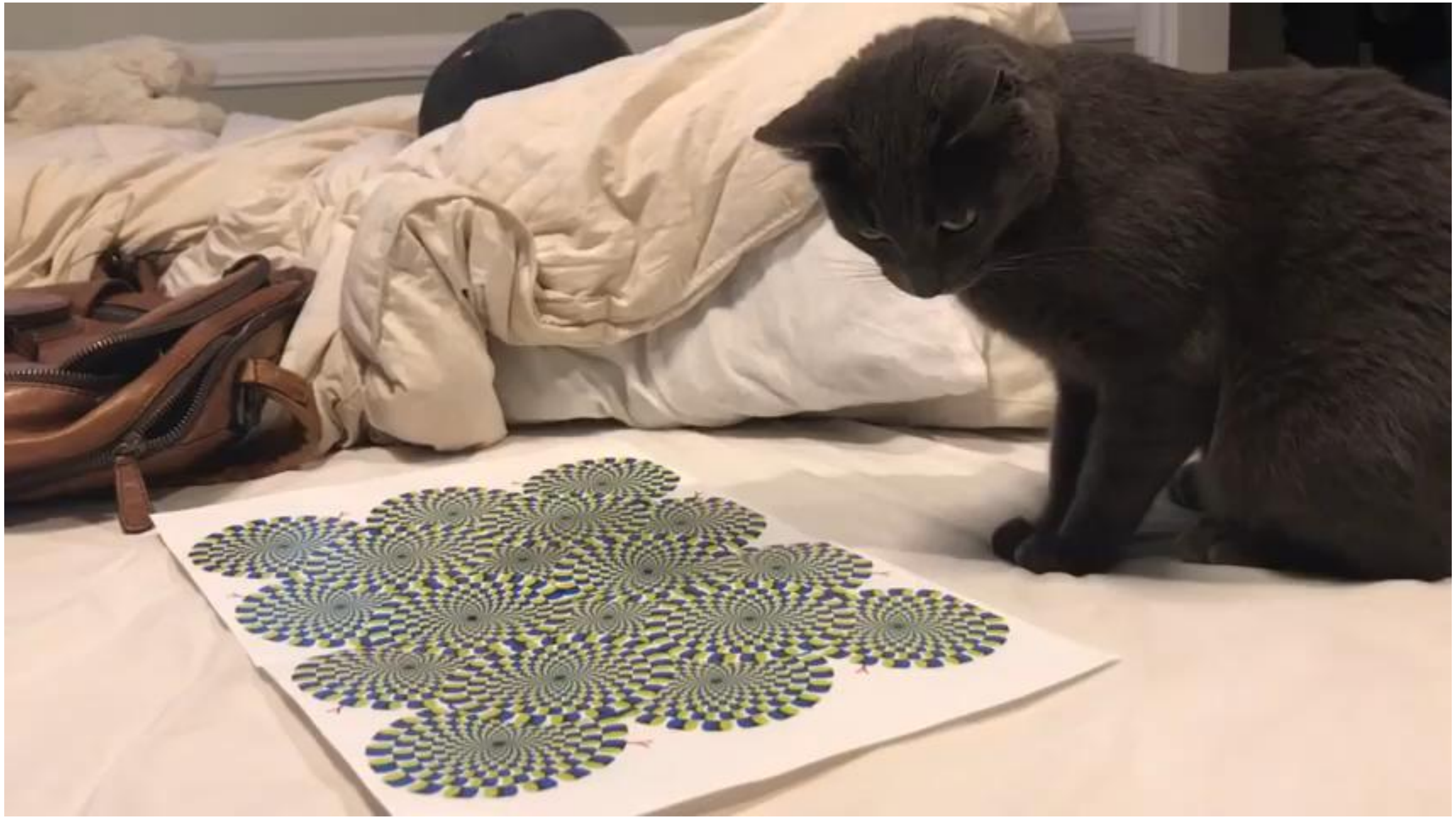
Motion



Motion ?



Not only humans see it moving



Watch: https://www.youtube.com/watch?v=S4IHB3qK1KU

Reasons?

- The patterns only move when you blink or move your eyes
- The arrangement of the backgrounds of the ‘rabbits’ determines which way the patterns rotate.

Why estimate visual motion?

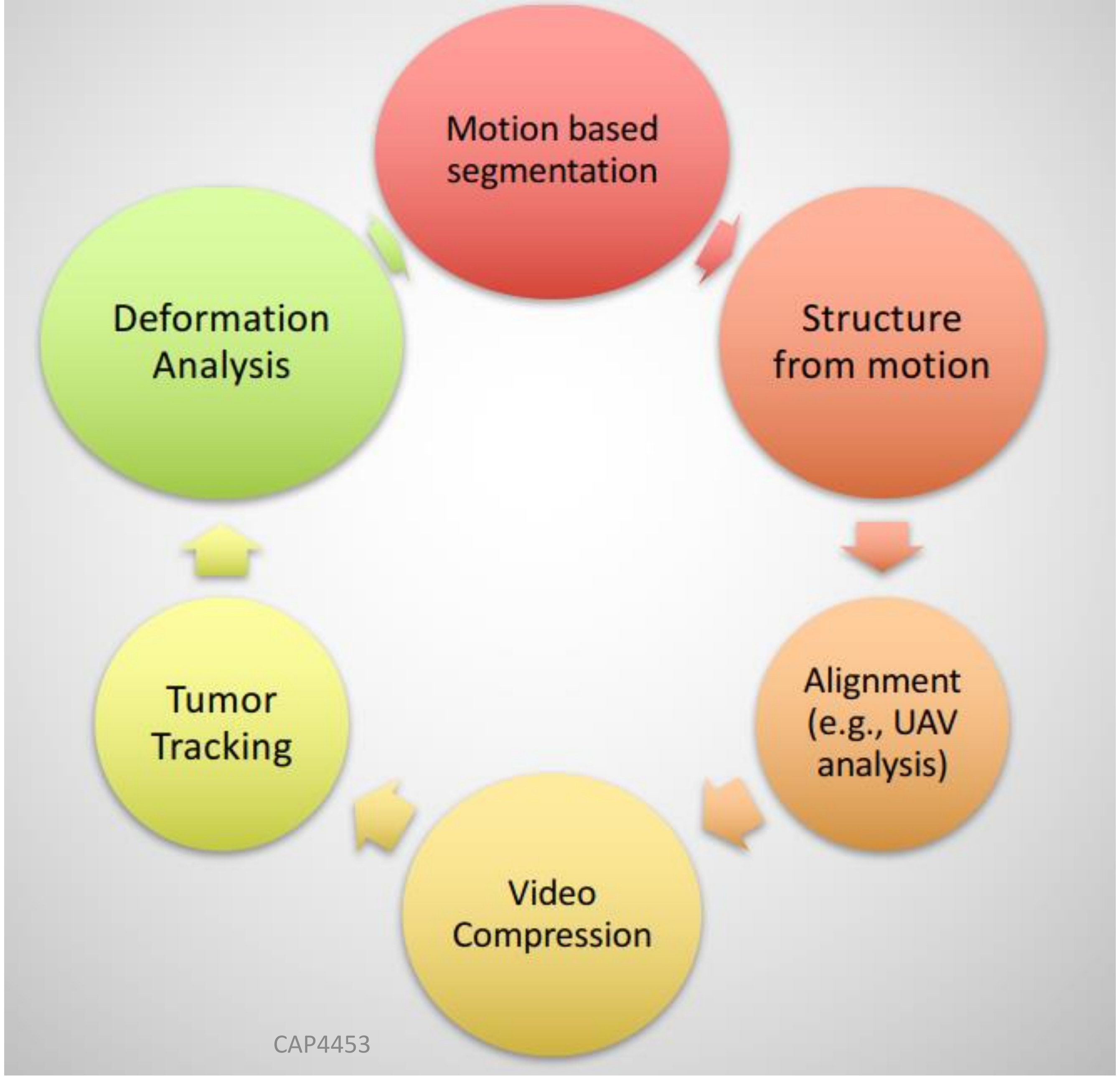
- Visual Motion can be due to problems
  - Camera instabilities, jitter
- Visual Motion Indicates dynamics in the scene
  - Moving objects, behavior, tracking objects, analyze trajectories
- Visual Motion reveals spatial layout
  - Motion parallax

Optical Flow

Visual Motion Estimation

- Patch-based motion
  - Optical Flow
    - Lucas-Kanade
    - Horn-Schunck
- NN-based approaches:
  - Example: FlowNET

Applications

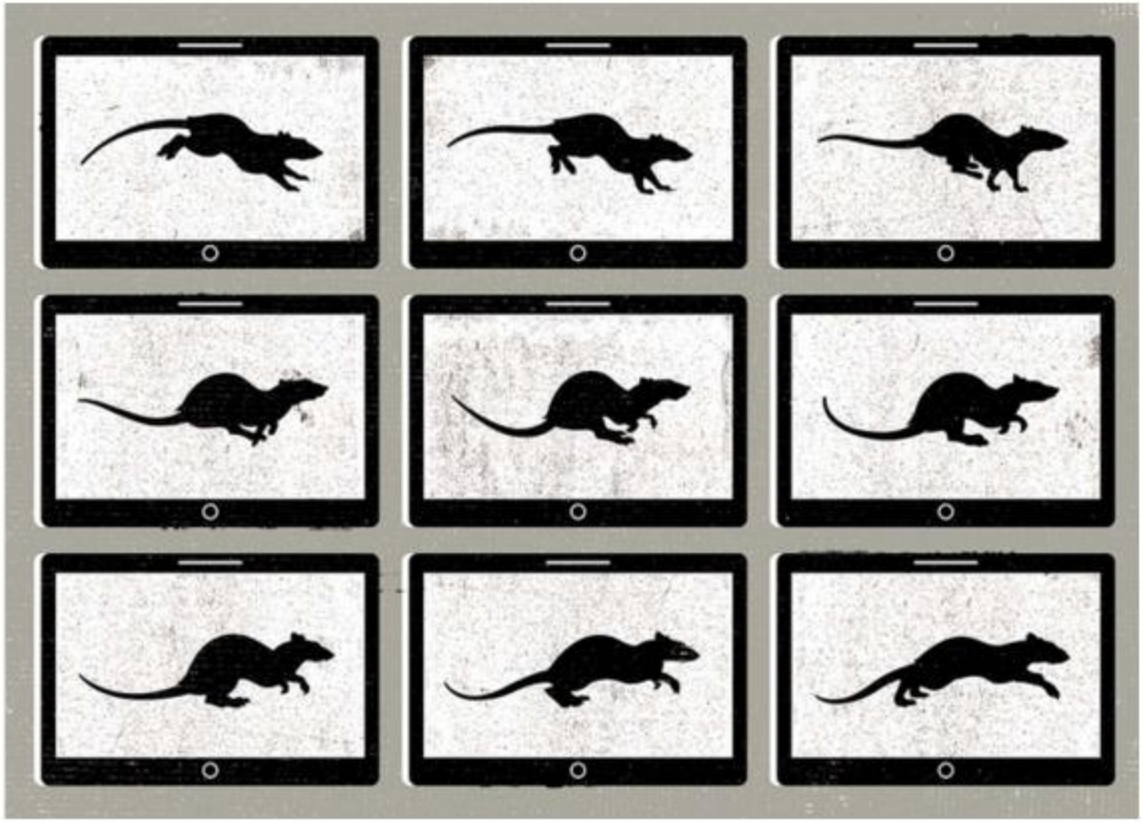


Nature News-Vol 525, Issue 7567

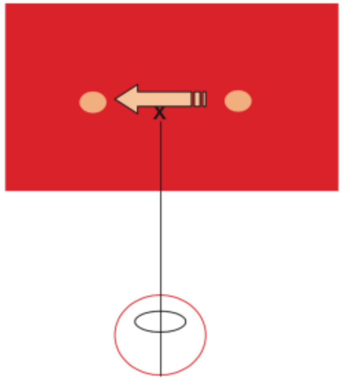
Sep 2015

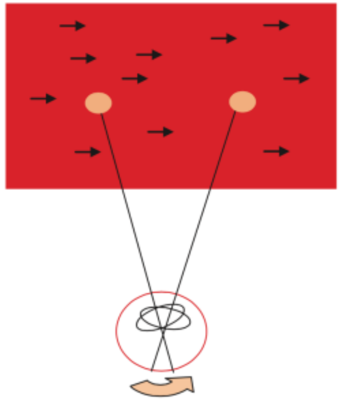
- CV Tools that track how animals move are helping researchers to do everything from diagnosing neurological conditions to illuminating evolution

Higher-speed cameras eventually improved what could be captured. But movement studies still needed a person to look through the results frame by frame, laboriously tracing the arc of each step, arm swing or wing flap to extract information about angles and forces.



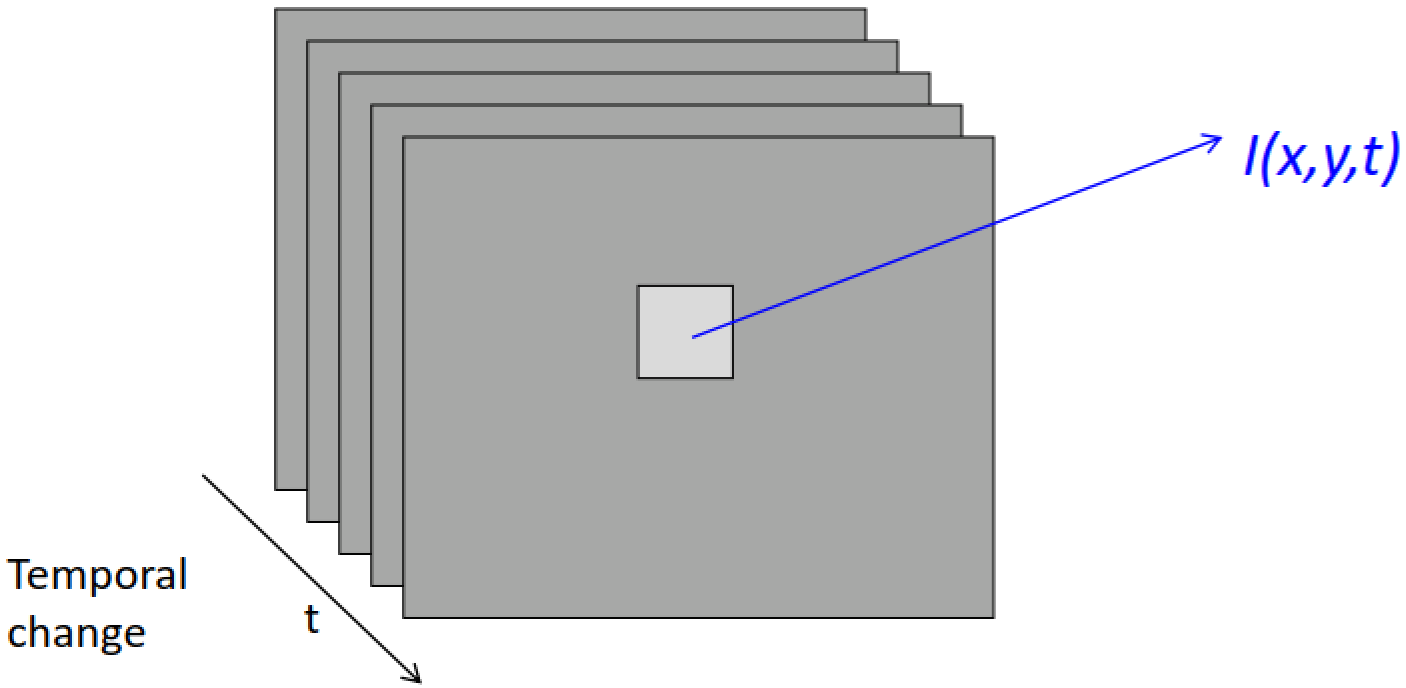
Perception of Motion



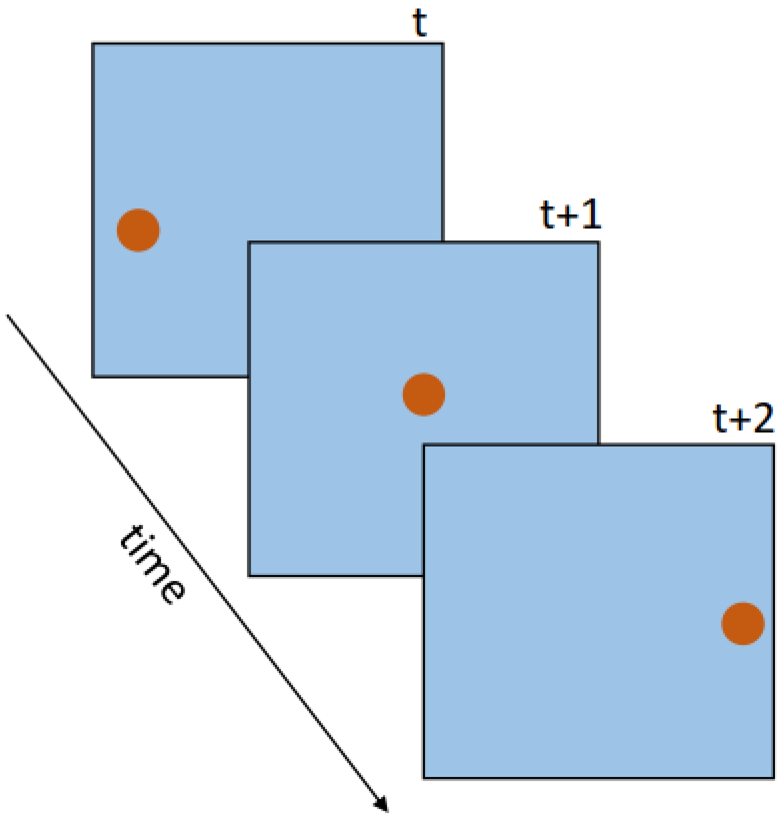


Video

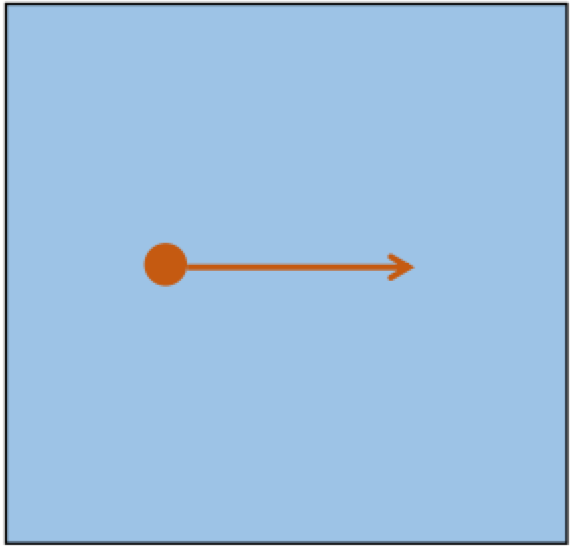
- A video is a sequence of frames captured over time
  - Image data is a function of space (x,y) and time (t)



Apparent motion



Present dots on three movie frames



See one moving dot

Describing motion

- Simplest way: Image Differencing (intensity values)

Optical Flow

- Refers to the problem of estimating a vector field of local displacement in a sequence of images.
- When we fix our attention to a single point and measure velocities flowing through that location, then the problem is called **optical flow**.
  - Stereo matching, image matching, tracking,…

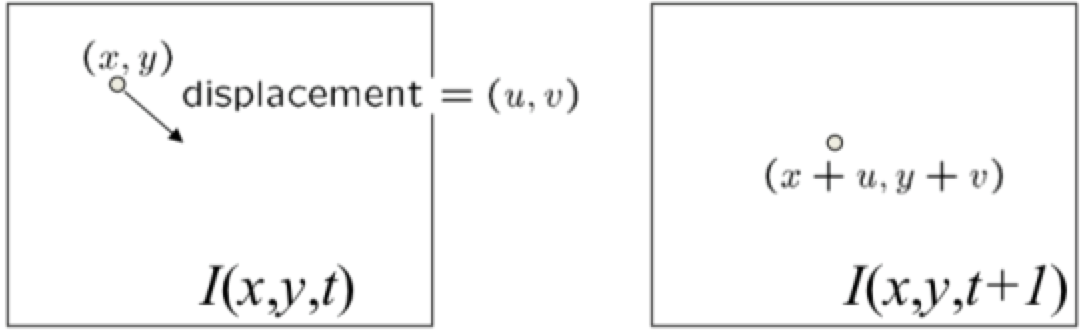
Estimating optical flow

- Assume the image intensity I is constant

Time = *t*   Time = *t+dt*

$$I(x,y,t) = I(x+dx, y+dy, t+dt)$$

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

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

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

So: $$I_x \cdot u + I_y \cdot v + I_t \approx 0 \quad \rightarrow \nabla \mathrm{I} \cdot [\mathrm{u} \ \ \mathrm{v}]^{\mathrm{T}} + \mathrm{I_t} = 0$$

Source: https://courses.cs.washington.edu/courses/cse455/16wi/notes/14_Optical_Flow.pdf

Estimating the optical flow

$$I(x, y, t) \simeq I(x + dx, y + dy, t + dt)$$

$$I(x(t) + u.\Delta t, y(t) + v.\Delta t) - I(x(t), y(t), t) \approx 0$$

Assuming I is differentiable function, and expand the first term using Taylor's series:

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0$$

Compact representation

$$I_x u + I_y v + I_t = 0$$

Brightness constancy constraint

The brightness constancy constraint

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

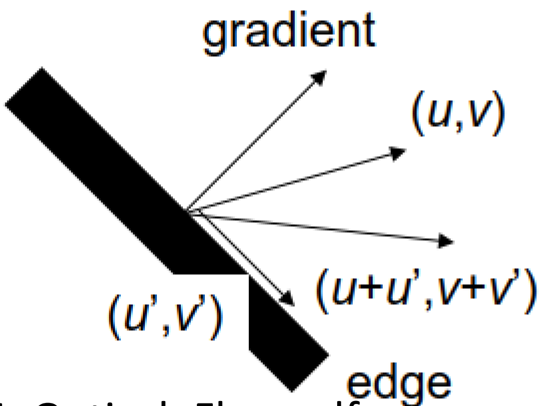
$$\nabla I \cdot [u \;\; v]^T + I_t = 0$$

- 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

$$\nabla I \cdot [u' \;\; v']^T = 0$$

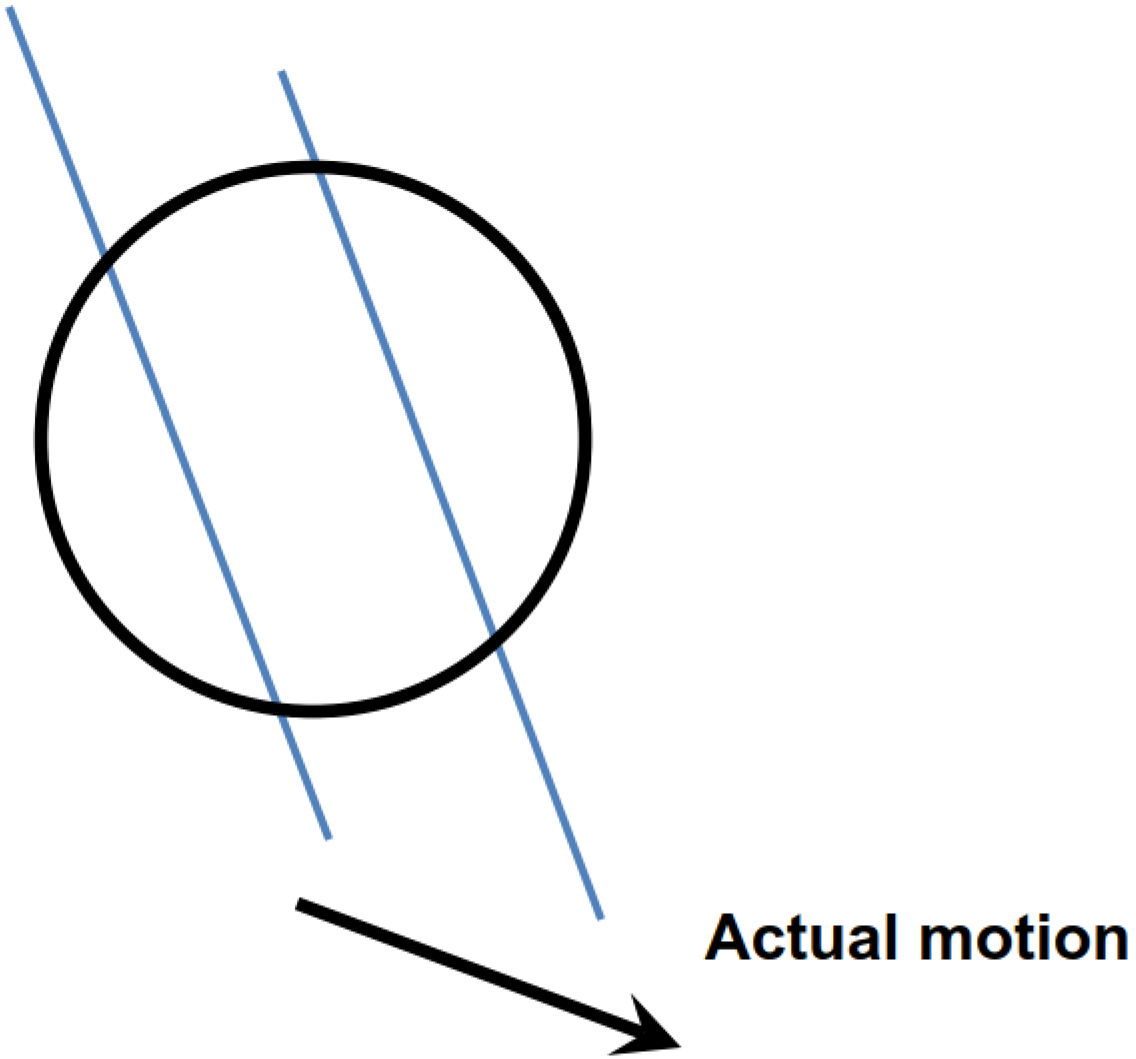


Source: https://courses.cs.washington.edu/courses/cse455/16wi/notes/14_Optical_Flow.pdf

Assumption: The brightness constant

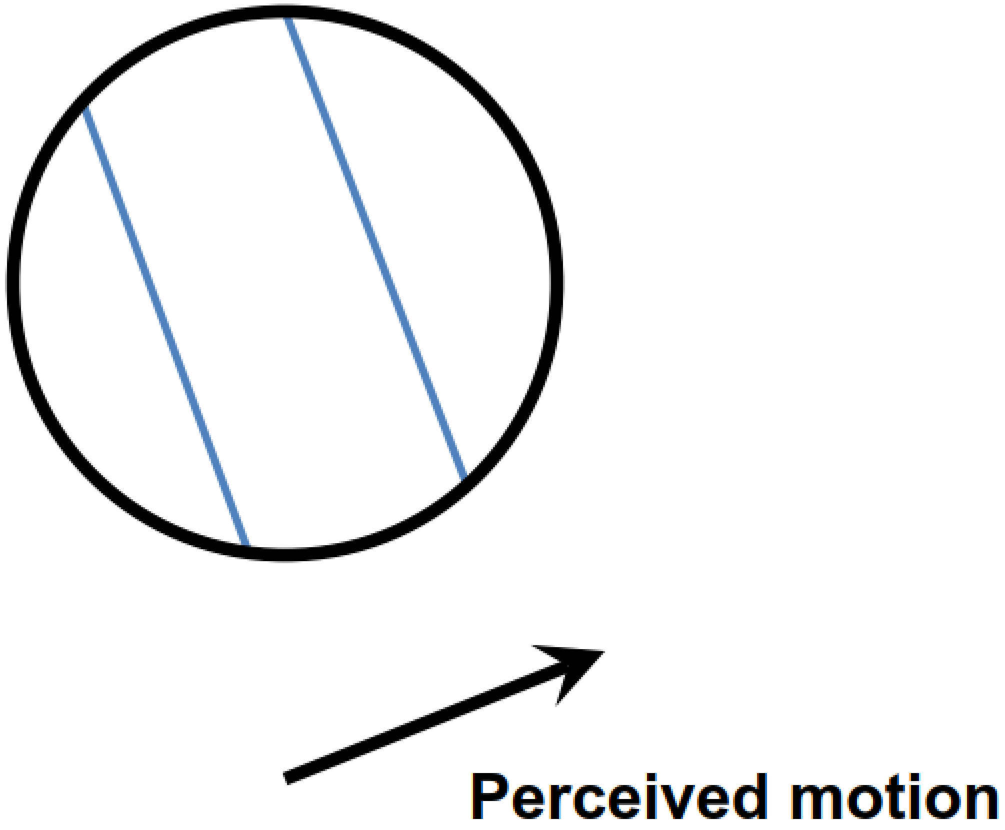
- Expresses the idea of similar brightness for the same objects observed in a sequence
- When we follow with a given location and trace their position in consecutive images of a sequence, then the problem is called “feature tracking”

The aperture problem



Source: https://courses.cs.washington.edu/courses/cse455/16wi/notes/14_Optical_Flow.pdf

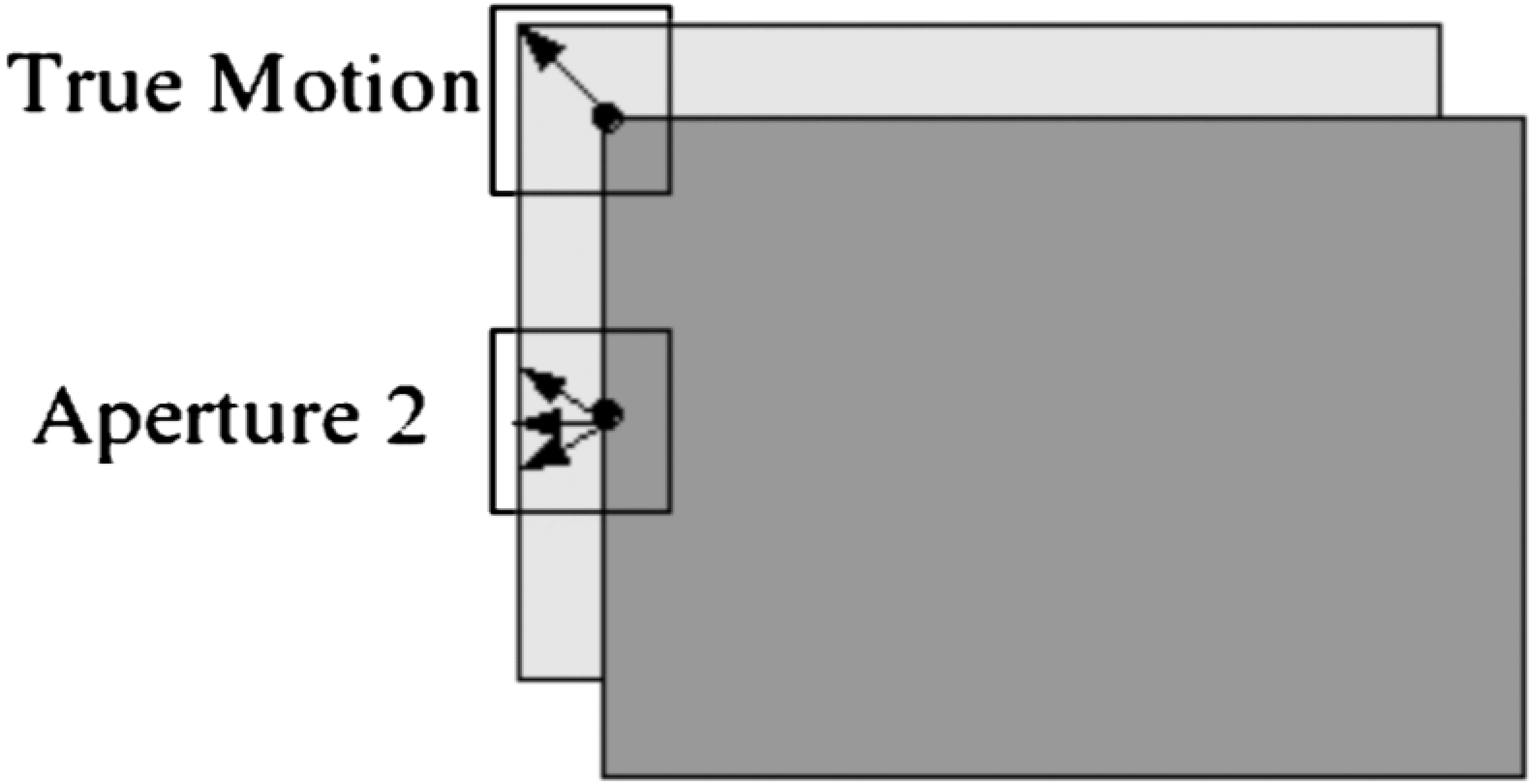
The aperture problem



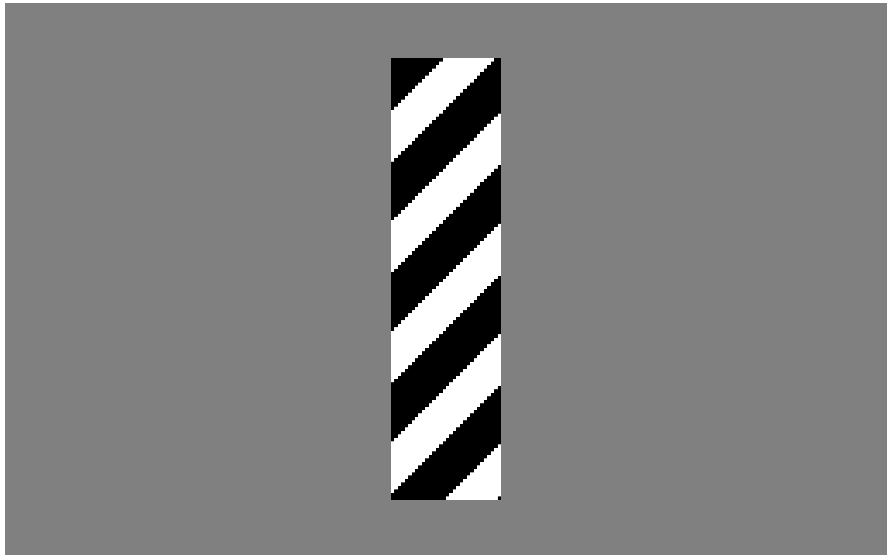
Source: https://courses.cs.washington.edu/courses/cse455/16wi/notes/14_Optical_Flow.pdf

Aperture Problem

- How certain are the motion estimates?



Barber pole ilusion



The illusion occurs when a diagonally striped pole is rotated around its vertical axis (horizontally), it appears as though the stripes are moving in the direction of its vertical axis

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

Solving the ambiguity…

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

- 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:

$$\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix} \qquad \underset{25\times 2}{A} \; \underset{2\times 1}{d} = \underset{25\times 1}{b}$$

Matching patches across images

- Overconstrained linear system

$$\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix} \qquad \underset{25\text{x}2}{A} \; \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

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

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

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

Conditions for solvability

Optimal (u, v) satisfies Lucas-Kanade equation

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

When is this solvable? I.e., what are good points to track?

- **A^TA** should be invertible
- **A^TA** should not be too small due to noise
  - eigenvalues λ_1 and λ_2 of **A^TA** should not be too small
- **A^TA** should be well-conditioned
  - λ_1 / λ_2 should not be too large (λ_1 = larger eigenvalue)

Does this remind you of anything?

Criteria for Harris corner detector

Lucas-Kanade

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

Lucas-Kanade

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

Structural Tensor representation

$$\begin{bmatrix} T_{xx} & T_{xy} \\ T_{xy} & T_{yy} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} T_{xt} \\ T_{yt} \end{bmatrix}$$

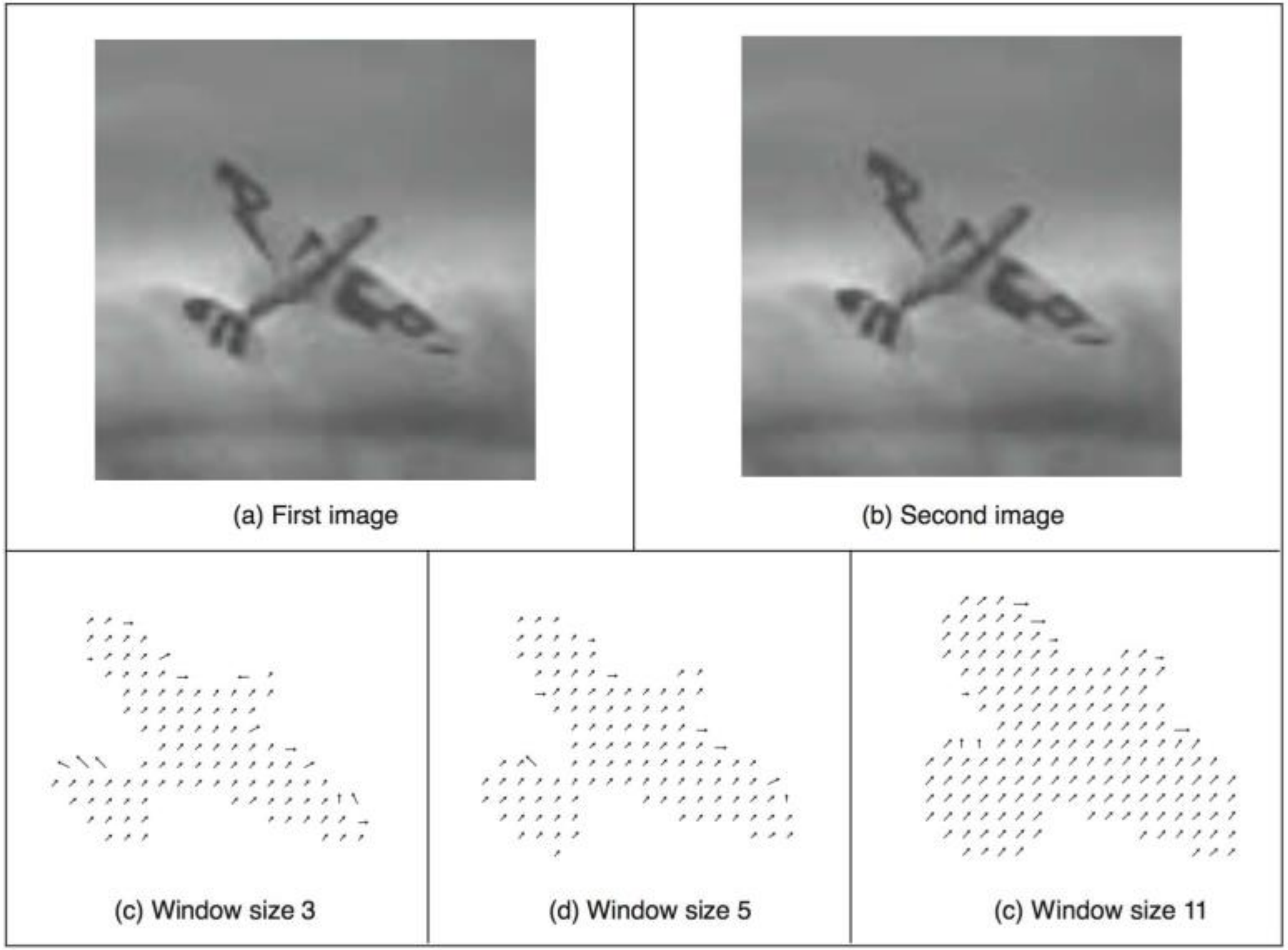
Lucas-Kanade

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

Structural Tensor representation

$$\begin{bmatrix} T_{xx} & T_{xy} \\ T_{xy} & T_{yy} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} T_{xt} \\ T_{yt} \end{bmatrix}$$

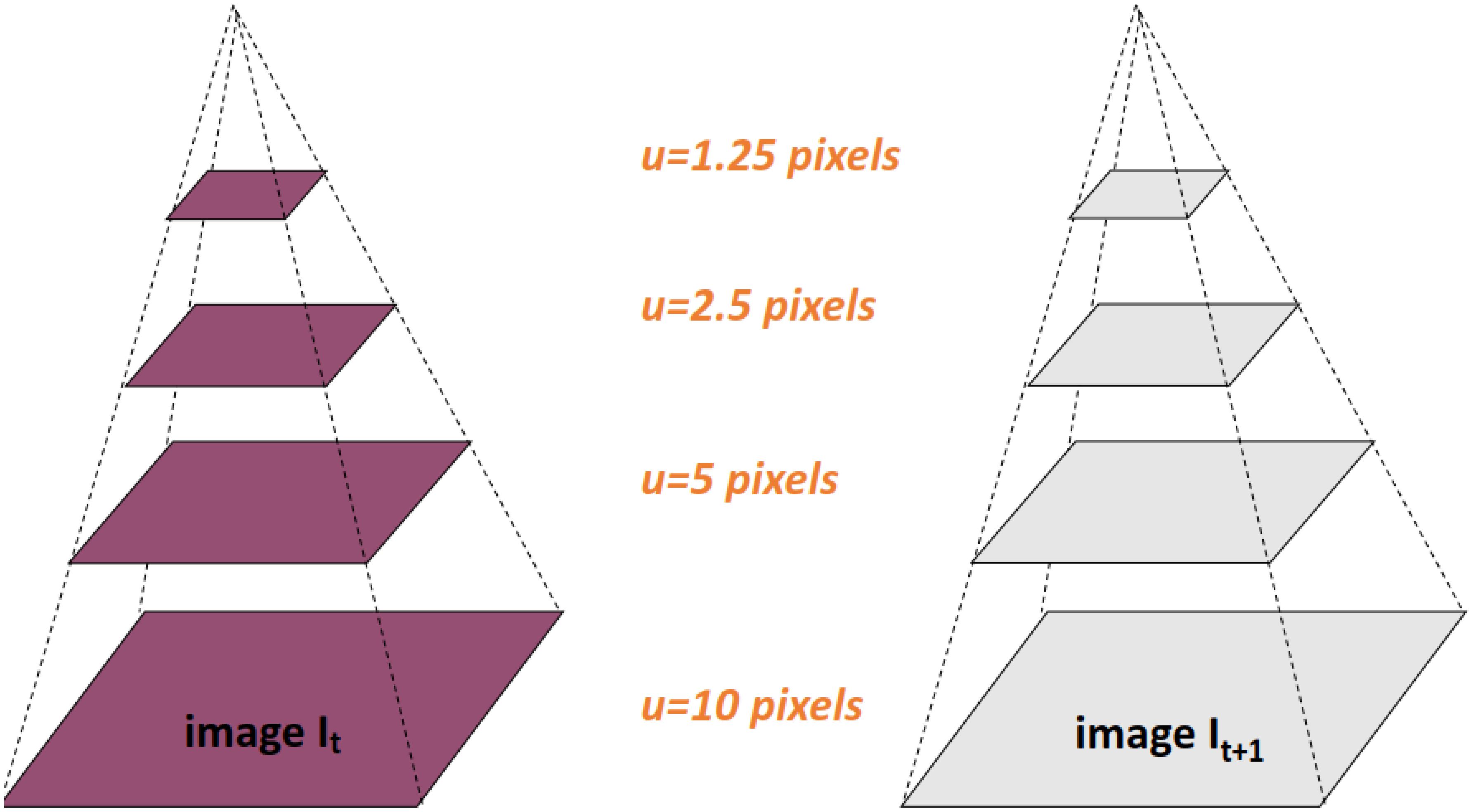
$$u = \frac{T_{yt}T_{xy} - T_{xt}T_{yy}}{T_{xx}T_{yy} - T_{xy}^2} \text{ and } v = \frac{T_{xt}T_{xy} - T_{yt}T_{xx}}{T_{xx}T_{yy} - T_{xy}^2}$$


(a) First image
(b) Second image
(c) Window size 3
(d) Window size 5
(c) Window size 11

Pitfalls and alternatives

- Brightness constancy is not satisfied
  - Correlation based method could be used
- A point may not move like its neighbors
  - Regularization based methods
- The motion may not be small (Taylor does not hold!)
  - Multi-scale estimation could be used

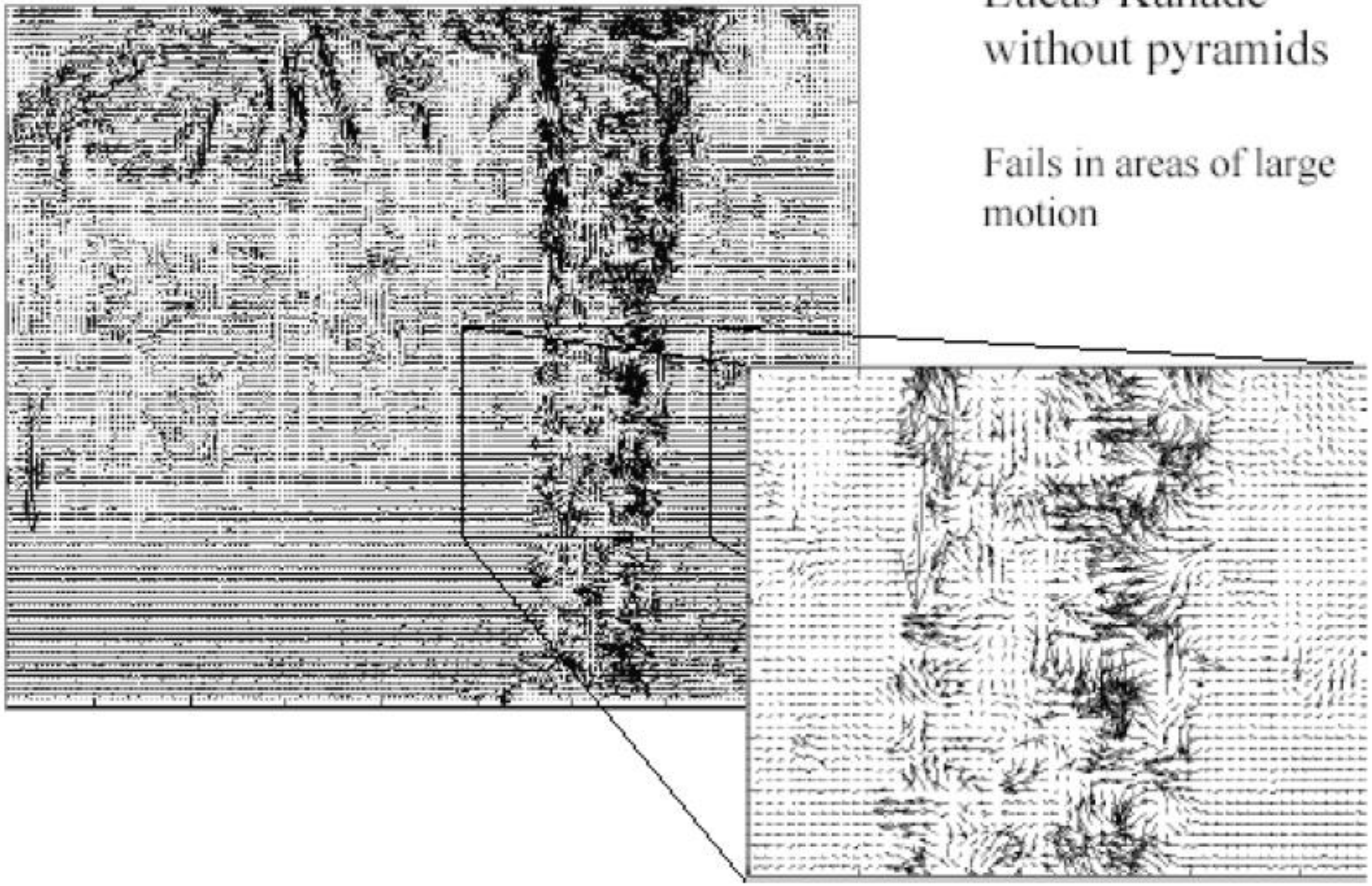
Multi-scale flow estimation



Gaussian pyramid of image I_t

Gaussian pyramid of image I_{t+1}

Optical Flow Results

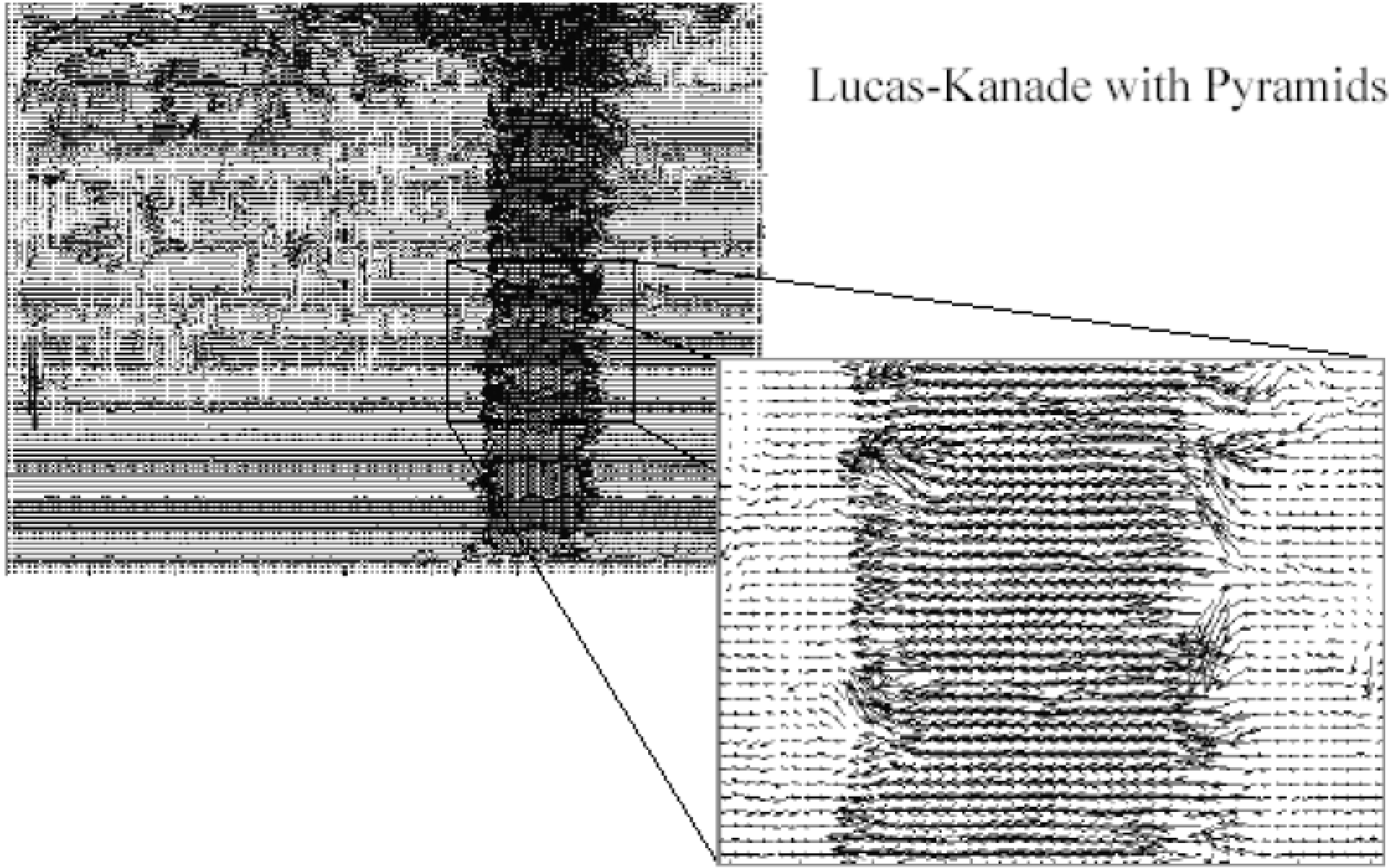


Lucas-Kanade without pyramids

Fails in areas of large motion

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Optical Flow Results

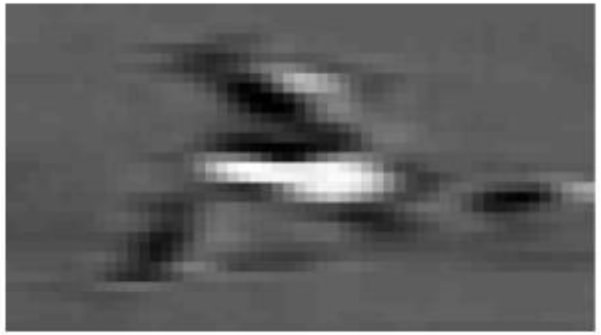


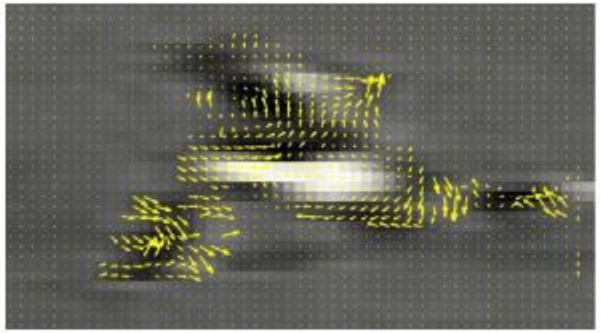
Lucas-Kanade with Pyramids

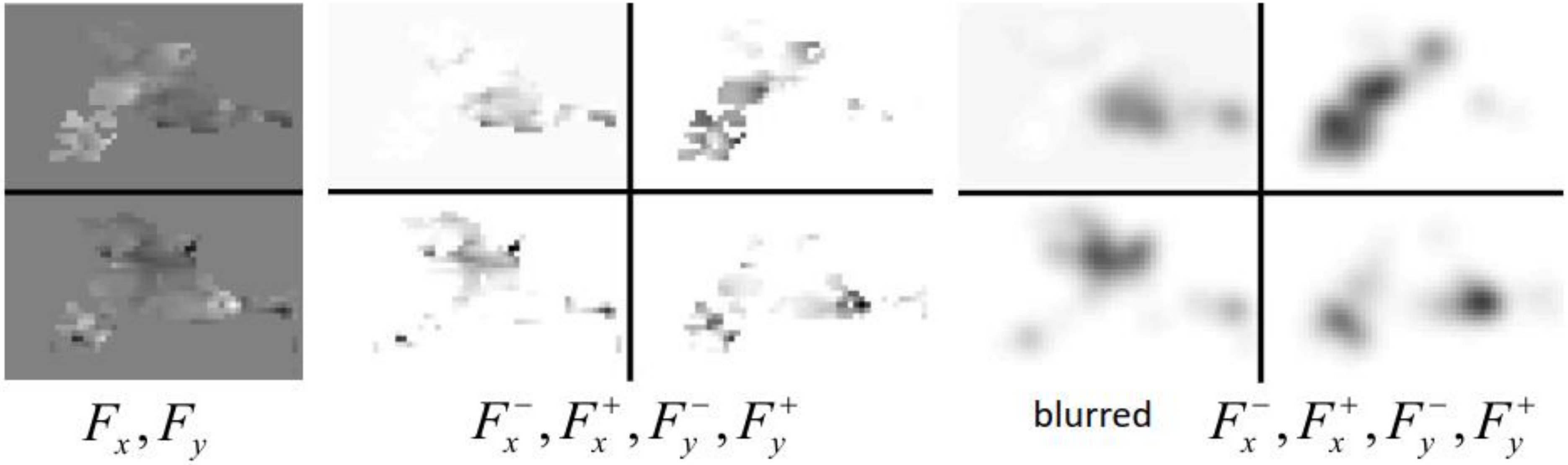
* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Applications: target tracking

Applications: action recognition







Recognition actions at a distance, Efros et al.

Applications: motion modeling

Flipping between image 1 and 2.

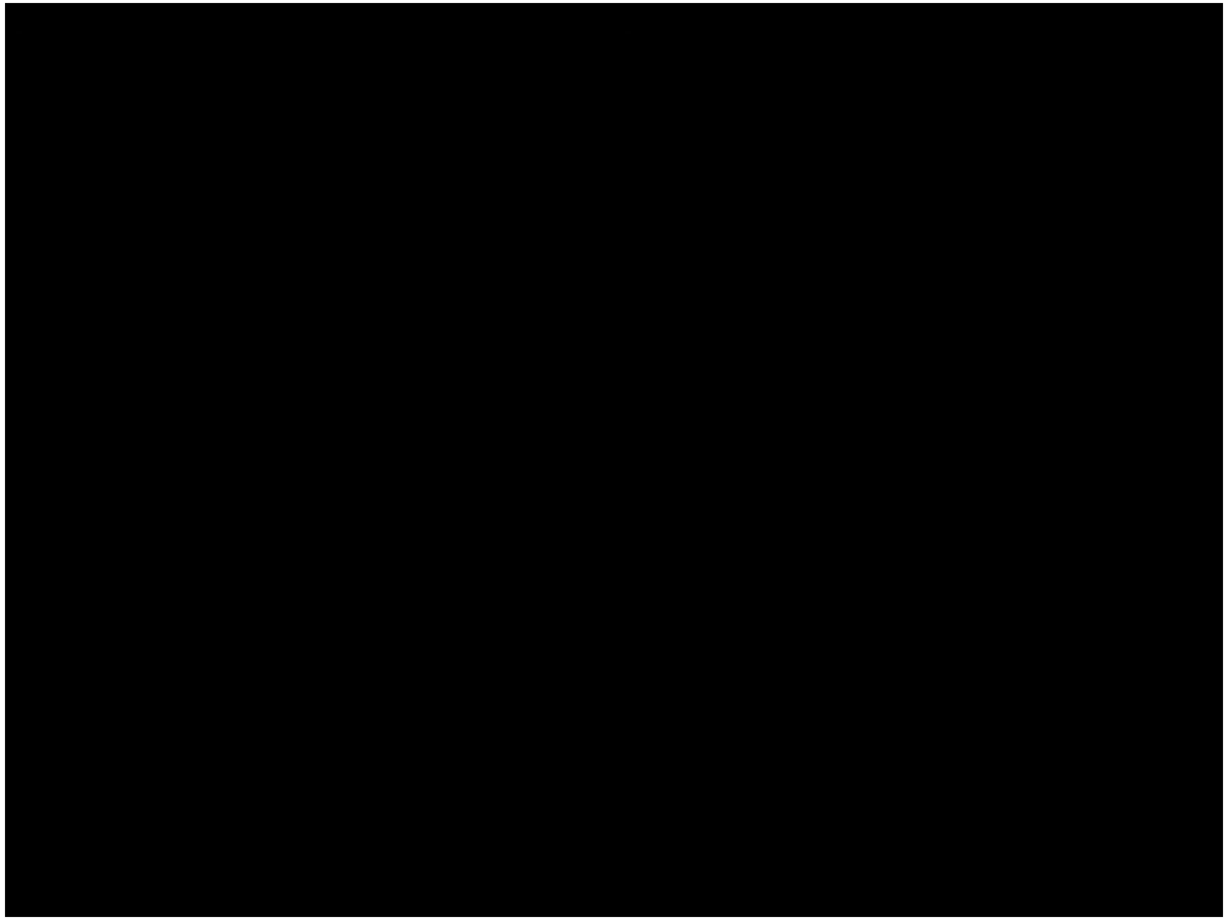
Estimated flow field

Second image is warped!
(hue indicates orientation
and saturation indicates magnitude)

Slide credit to C.Liu

Applications: Motion segmentation

Applications: FlowCap 2D Human Pose



Example:

- Python code

```python
import numpy as np
import cv2 as cv

video = 'C:/Users/gonza/Downloads/videoplayback.mp4'
cap = cv.VideoCapture(video)
# Parameters for lucas kanade optical flow
lk_params = dict( winSize = (15,15), maxLevel = 2)
# Take first frame and find corners in it

# Create some random colors
color = np.random.randint(0,255,(100,3))
ret, old_frame = cap.read()
old_gray = cv.cvtColor(old_frame, cv.COLOR_BGR2GRAY)
# **feature_params
p0 = cv.goodFeaturesToTrack(old_gray, mask = None, maxCorners=100, qualityLevel=0.5, minDistance=5)

# Create a mask image for drawing purposes
mask = np.zeros_like(old_frame)
while(1):
    ret,frame = cap.read()
    frame_gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)
    # calculate optical flow
    p1, st, err = cv.calcOpticalFlowPyrLK(old_gray, frame_gray, p0, None, **lk_params)
    # Select good points
    if p1 is not None:
        good_new = p1[st==1]
        good_old = p0[st==1]

            # draw the tracks
    for i,(new,old) in enumerate(zip(good_new,good_old)):
        a,b = new.ravel()
        c,d = old.ravel()
        mask = cv.line(mask, (a,b),(c,d), color[i].tolist(), 2)
        frame = cv.circle(frame,(a,b),5,color[i].tolist(),-1)
    img = cv.add(frame,mask)
    cv.imshow('frame',img)
    k = cv.waitKey(30) & 0xff
    if k == 27:
        break

    # Now update the previous frame and previous points
    old_gray = frame_gray.copy()
    p0 = good_new.reshape(-1,1,2)
cv.destroyAllWindows()
cap.release()
```

Further readings

- Szeliski, R. Ch. 7
- Bergen et al. ECCV 92, pp. 237-252.
- Shi, J. and Tomasi, C. CVPR 94, pp.593-600.
- Baker, S. and Matthews, I. IJCV 2004, pp. 221-255.

Questions?