
Computer vision

Computer Vision

Lecture 10: Motion Estimation

DR. DINA KHATTAB

DINA.KHATTAB@CIS.ASU.EDU.EG

SCIENTIFIC COMPUTING DEPARTMENT

Instructor:	Dr. Dina Khattab
Email:	dina.khattab@cis.asu.edu.eg
Office:	Main Building – 4th floor – Room 302
Office Hours:	Monday 12:00 AM to 1:00 PM Thursday 11:00 AM to 12:00 PM

Agenda

Motion Estimation

- Optical Flow
- Lucas and Kanade (LK)
- Hierarchical LK
- Deep learning for optical flow estimation (FlowNet)

Visual Motion

Visual Motion

Video

A video is a ordered set of frames of the same resolution captured over time – usually quickly.

Taken at regular intervals (fps) – (from 3-5 to 30-50 fps)

Resolution – from 320x240 to 1920x1080 (FullHD) to 4K

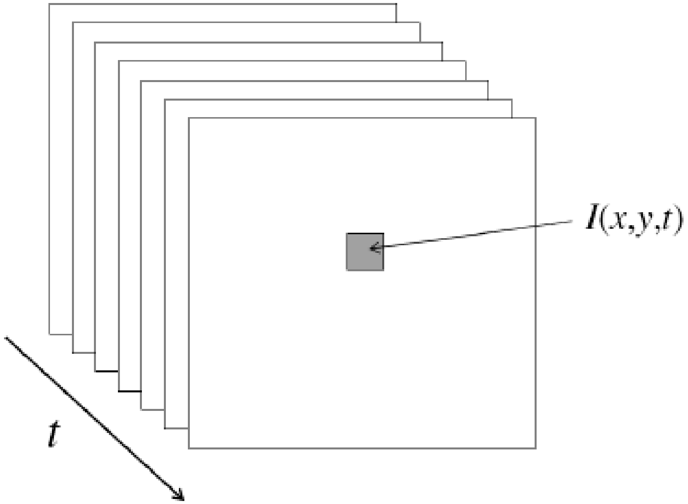
Grayscale or color (usually RGB – 3 channels)

Data Stream – 2MB (1 channel for FullHD) x 3 (RGB) x 50 fps = 300MB/sec

Now our image data is

a function of space (x, y)

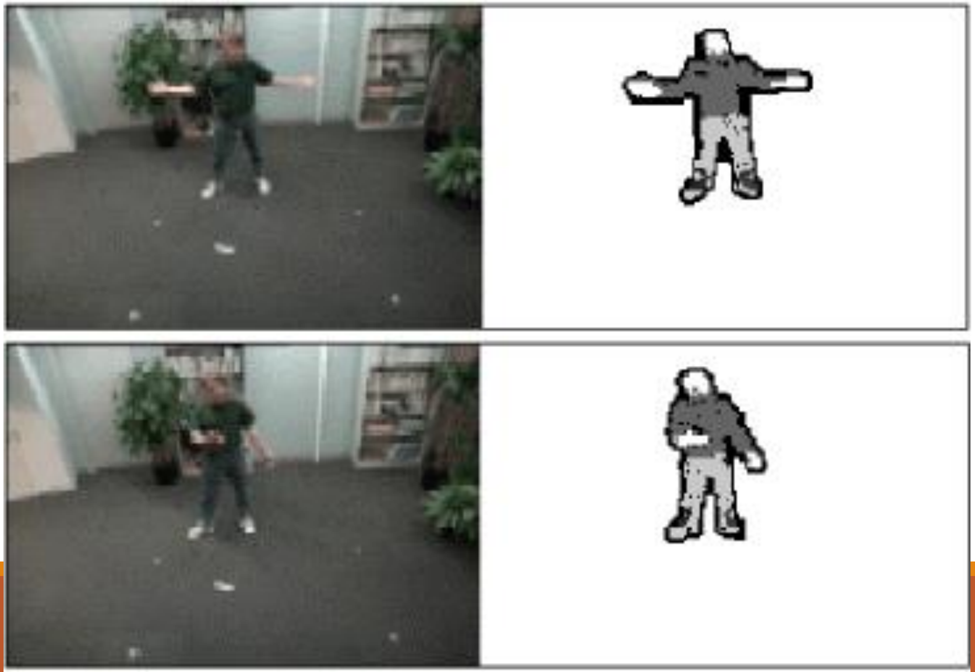
and time (t)



Motion Applications: Video Segmentation

Background subtraction

- A static camera is observing a scene
- Goal: separate the static background from the moving foreground



Motion Segmentation

Segment the video into multiple coherently moving objects

More applications of motion analysis

Segmentation of objects in space or time

Estimating 3D structure

Learning dynamical models – how things move

Recognizing events and activities

Improving video quality (motion stabilization)

Motion estimation techniques

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)

DIRECT, DENSE METHODS

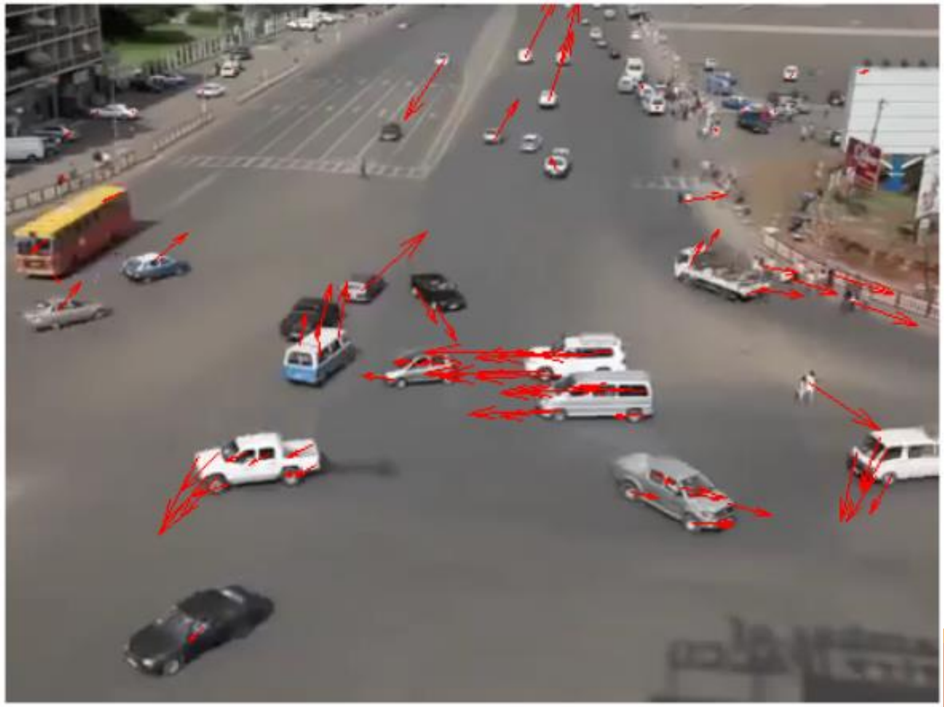
- Directly recover image motion at each pixel from spatio-temporal image brightness variations
- Dense motion fields, but sensitive to appearance variations
- Suitable for video and when image motion is small

Dense flow: Brightness constraint

Motion estimation: Optical flow

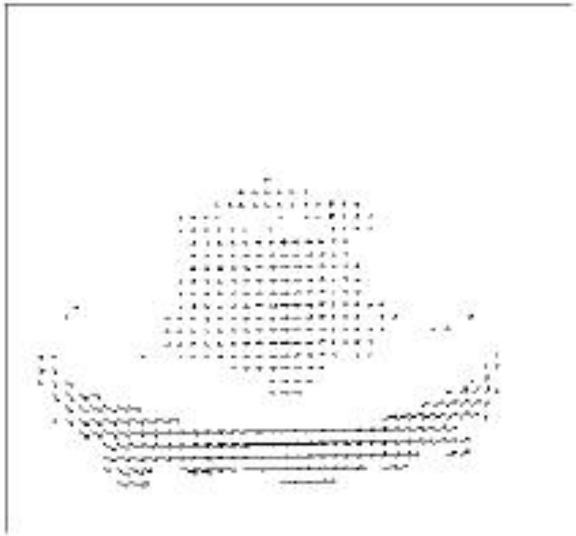
Optical flow is the movement of pixels over time.

The goal of optical flow is to generate a motion vector for each pixel in an image between t_0 and t_1 by looking at two images I_0 and I_1

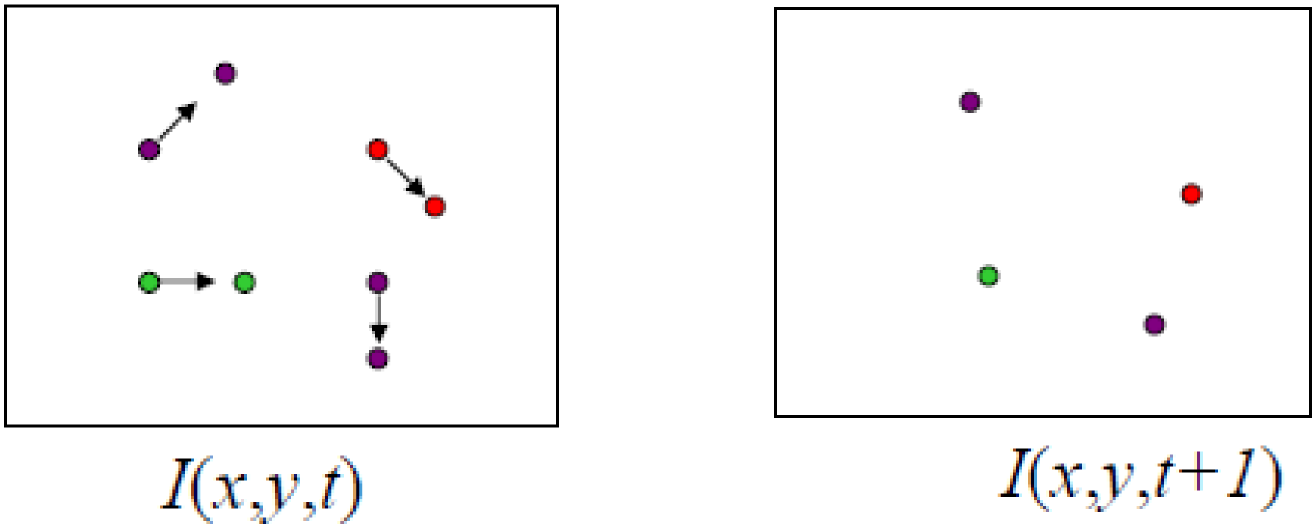


Motion estimation: Optical flow

Optical flow is limited to representing the apparent motion of objects (brightness patterns).

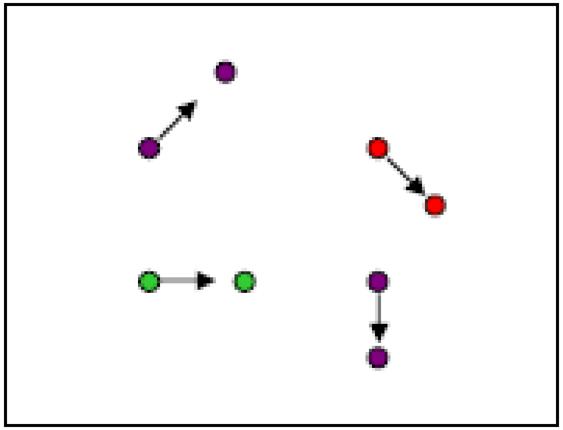


Problem definition: Optical flow

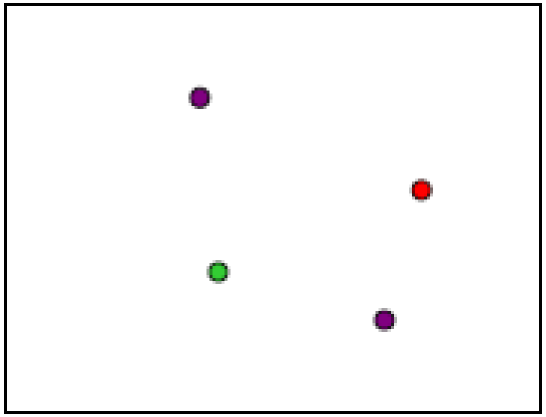


Given two subsequent frames *I(x, y, t) and I(x, y, t+1)*, how to estimate pixel motion from image?

Estimate the apparent motion field u, v between them



$I(x,y,t)$



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

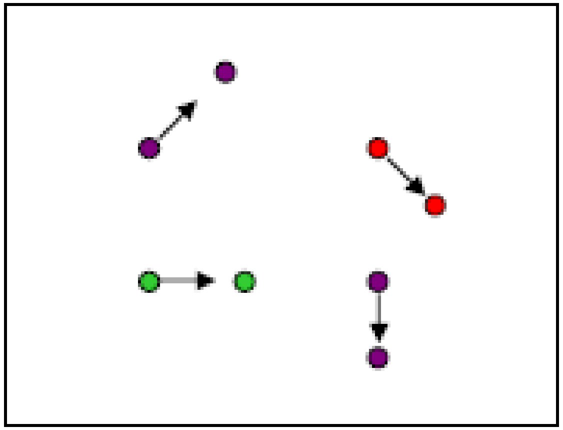
<u>How to estimate pixel motion from</u>

<u>image *I(x, y, t)* to *I(x, y, t+1)?*</u>

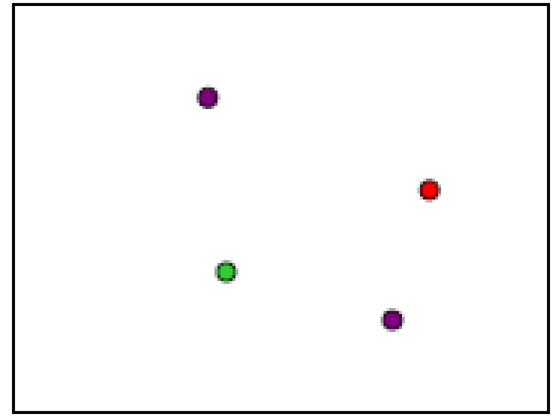
Solve pixel correspondence problem

Given a pixel in $I(x, y, t)$, look for nearby pixels of the same color in $I(x, y, t+1)$

This is the optical flow problem.



$I(x,y,t)$



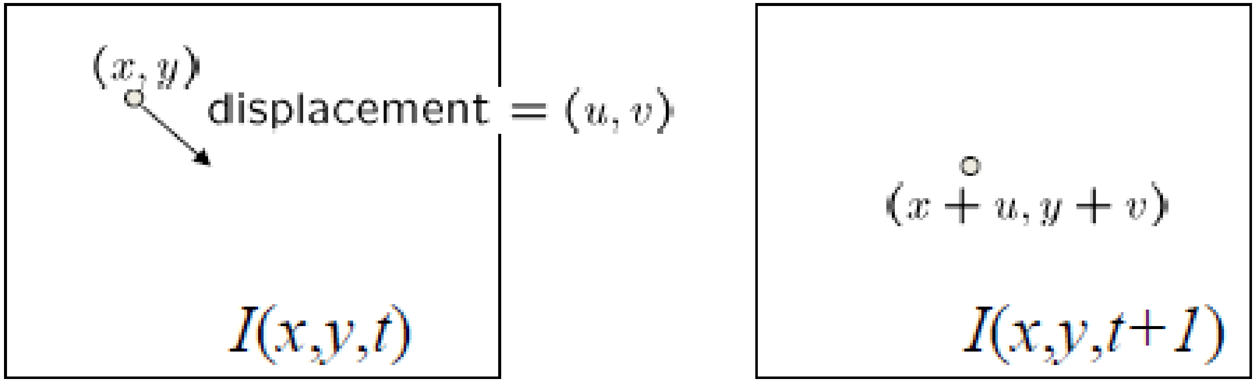
$I(x,y,t+1)$

How to estimate pixel motion from image *I(x, y, t)* to *I(x, y, t+1)?*

Key assumptions

- **Color constancy**: a point in $I(x, y, t)$ looks the same in $I(x', y', t + 1)$ For grayscale images, this is ***brightness constancy***
- **Small motion**: points do not move very far

Optical flow constraints (grayscale images)

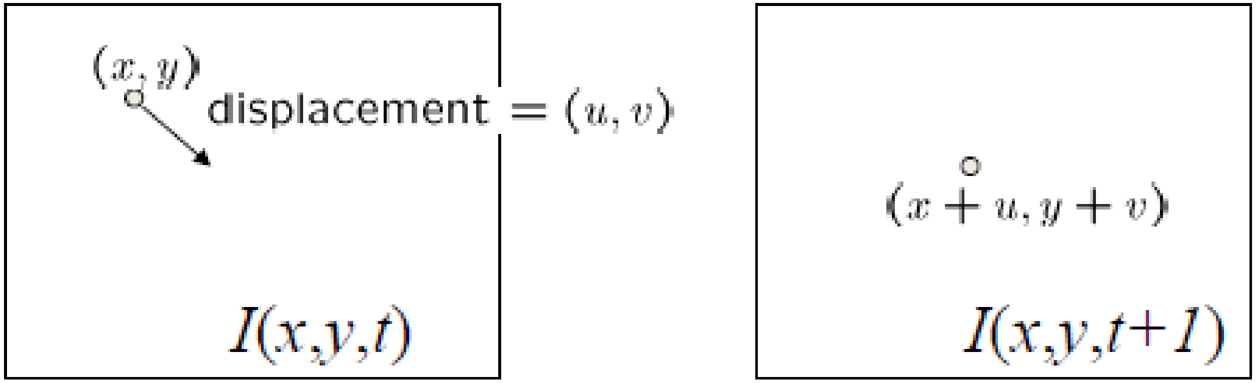


1. Brightness constancy constraint (equation)

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

$$I(x+u, y+v, t+1) - I(x, y, t) = 0$$

Optical flow constraints (grayscale images)



2. Small motion: (u and v are less than 1 pixel, or smooth)

Taylor series expansion of *I*:

$$I(x+u, y+v) \approx I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

(Short hand: $I_x = \frac{\partial I}{\partial x}$ for t or $t+1$)

Combining these two equations:

$$0 = I(x+u, y+v, t+1) - I(x,y,t)$$

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

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

$$\approx I_t + I_x u + I_y v$$

$$\approx I_t + \nabla I \cdot < u, v >$$

Combining these two equations:

In the limit as u and v approaches zero, this becomes exact:

$$0 = I_t + \nabla I \cdot < u, v >$$

$$0 \approx I_t + \nabla I \cdot < u, v >$$

Brightness constancy constraint equation

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

Gradient component of flow

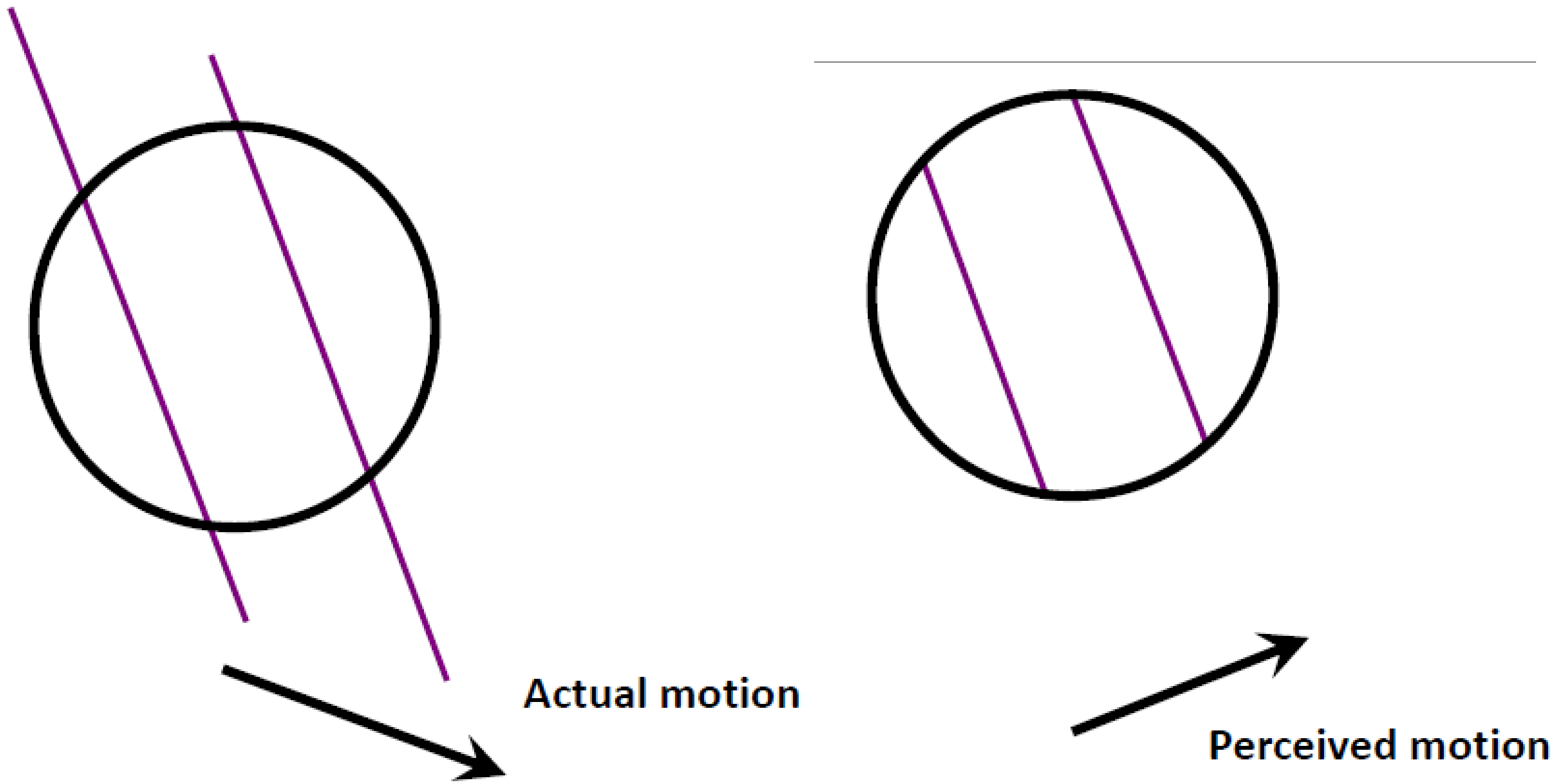
$$0 = I_t + \nabla I \cdot < u, v > \quad \text{or} \quad I_x u + I_y v + I_t = 0$$

Q: How many unknowns and equations per pixel?

2 unknowns (u,v) but 1 equation!

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

Aperture problem



The aperture problem refers to the fact that motion estimation is highly ambiguous when the observation window is very small.

The optical flow in such case cannot be reliably estimated along the edges.

No. of unknowns vs equations (pixels)

So if the brightness constraint equation gives us more unknowns than pixels, how do we recover motion?

Additional Flow constraints:

? Nearby pixels move together (Spatial Coherence)

Local constraint

? Motion must be consistent over the entire image

Global constraint

Dense flow: Lucas and Kanade

Solving the aperture problem

Basic idea: Impose *local* constraints to get more equations for a pixel

E.g., assume that the flow field is smooth locally

Solving the aperture problem

One method: Pretend the pixel's neighbors have the ***same*** (u,v)

If we use a 5x5 window, that gives us 25 equations per pixel!

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

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

Lukas-Kanade flow (1981)

Problem: We have more equations than unknowns

($d = u\ v$)

$$\underset{25\text{x}2}{A}\ \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b} \longrightarrow \text{minimize } \|Ad - b\|^2$$

$$\underset{2\text{x}2}{(A^T A)}\ \underset{2\text{x}1}{d} = \underset{2\text{x}1}{A^T b}$$

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

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

Eigenvectors of $A^T A$

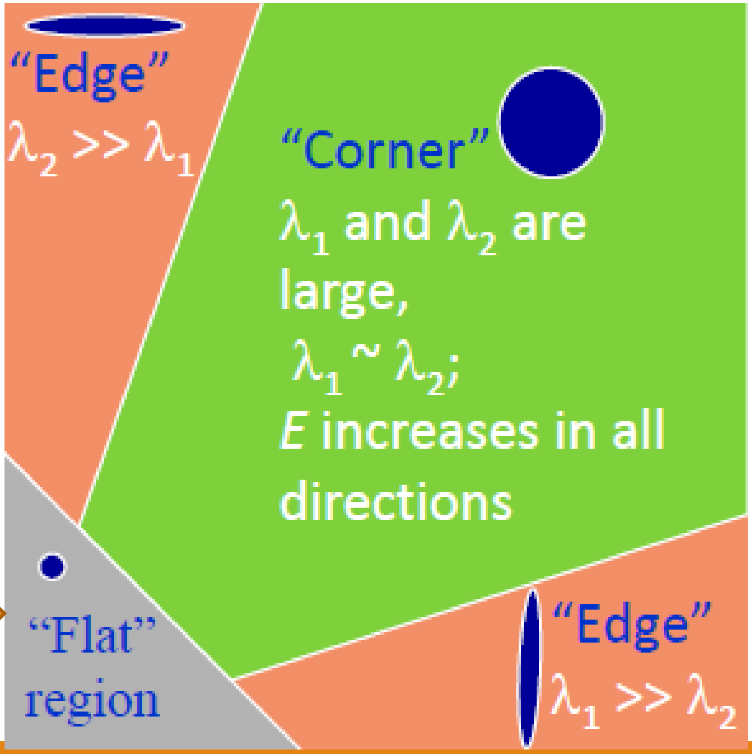
$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}$$

Recall the Harris corner detector: $M = A^T A$ is the second moment matrix

The eigenvectors and eigenvalues of M relate to edge direction and magnitude

Interpreting the eigenvalues

Classification of image points using eigenvalues of *M*:



λ_1 and λ_2 are small; *E* is almost constant in all directions

RGB version

One method: pretend the pixel's neighbors have the ***same*** (u,v)

If we use a **5x5x3** window, that gives us 75 equations per pixel!

$$0 = I_t(\mathrm{p_i})[0,1,2] + \nabla I(\mathrm{p_i})[0,1,2] \cdot [u \;\; v]$$

$$\underbrace{\begin{bmatrix} I_x(\mathrm{p}_1)[0] & I_y(\mathrm{p}_1)[0] \\ I_x(\mathrm{p}_1)[1] & I_y(\mathrm{p}_1)[1] \\ I_x(\mathrm{p}_1)[2] & I_y(\mathrm{p}_1)[2] \\ \vdots & \vdots \\ I_x(\mathrm{p}_{25})[0] & I_y(\mathrm{p}_{25})[0] \\ I_x(\mathrm{p}_{25})[1] & I_y(\mathrm{p}_{25})[1] \\ I_x(\mathrm{p}_{25})[2] & I_y(\mathrm{p}_{25})[2] \end{bmatrix}}_{\substack{A \\ 75\times 2}} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\substack{d \\ 2\times 1}} = - \underbrace{\begin{bmatrix} I_t(\mathrm{p}_1)[0] \\ I_t(\mathrm{p}_1)[1] \\ I_t(\mathrm{p}_1)[2] \\ \vdots \\ I_t(\mathrm{p}_{25})[0] \\ I_t(\mathrm{p}_{25})[1] \\ I_t(\mathrm{p}_{25})[2] \end{bmatrix}}_{\substack{b \\ 75\times 1}}$$

Errors in Lucas-Kanade

When our assumptions are violated

- Brightness constancy is **not** satisfied
- The motion is **not** small
- A point does **not** move like its neighbors
  - window size is too large

Revisiting the small motion assumption

Motion is much larger than one pixel

Revisiting the small motion assumption

Motion is much larger than one pixel

Improving accuracy

In case of the motion is large (larger than a pixel) – Taylor doesn’t hold

Solutions:

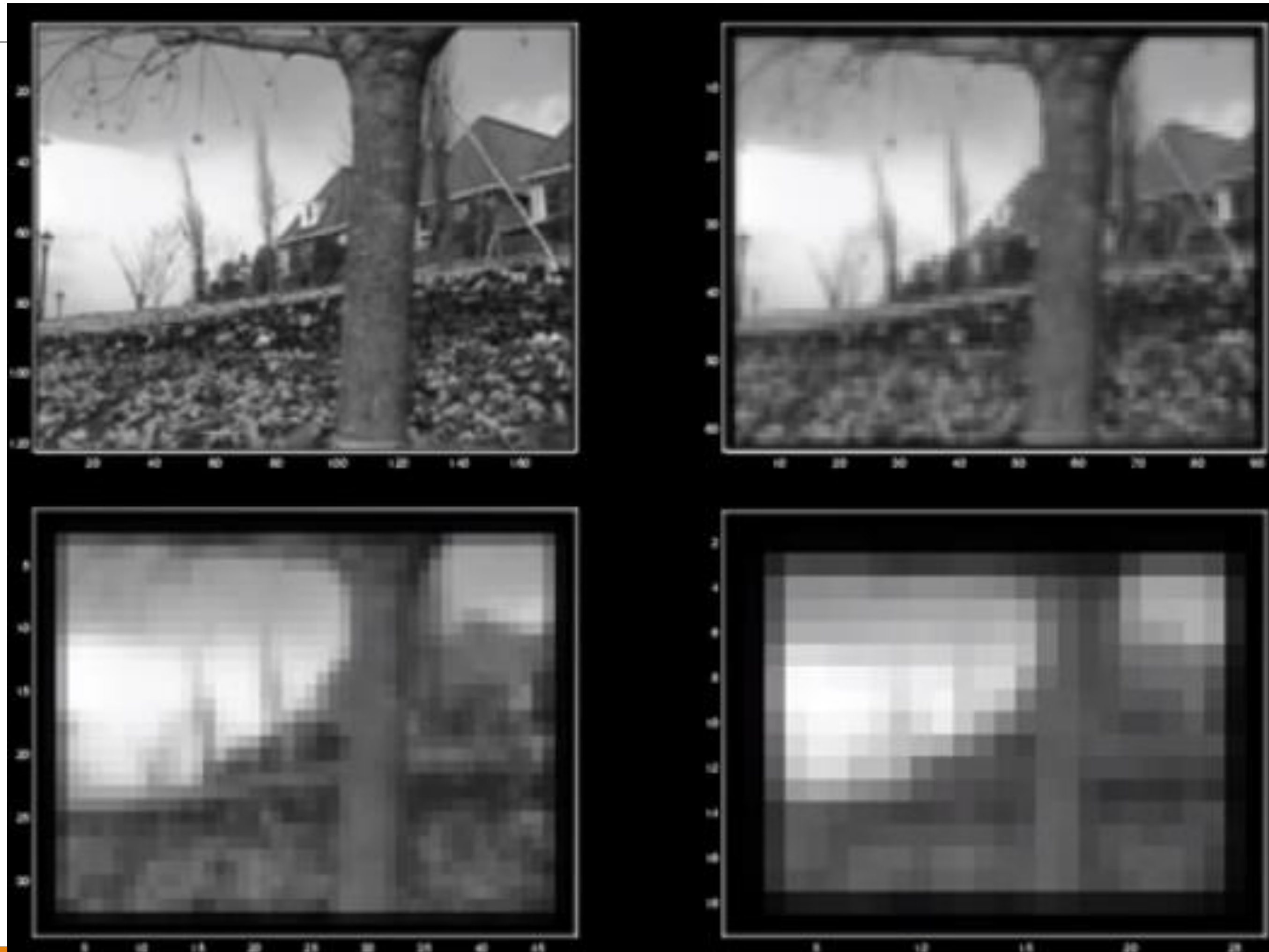
- Iterative refinement
- Coarse-to-fine estimation (Hierarchical LK)

Iterative Refinement

Iterative Lukas-Kanade Algorithm

1. Estimate motion at each pixel by solving Lucas-Kanade equations
2. Warp I_t towards I_{t+1} using the estimated flow field
3. Repeat until convergence

Reduce the resolution



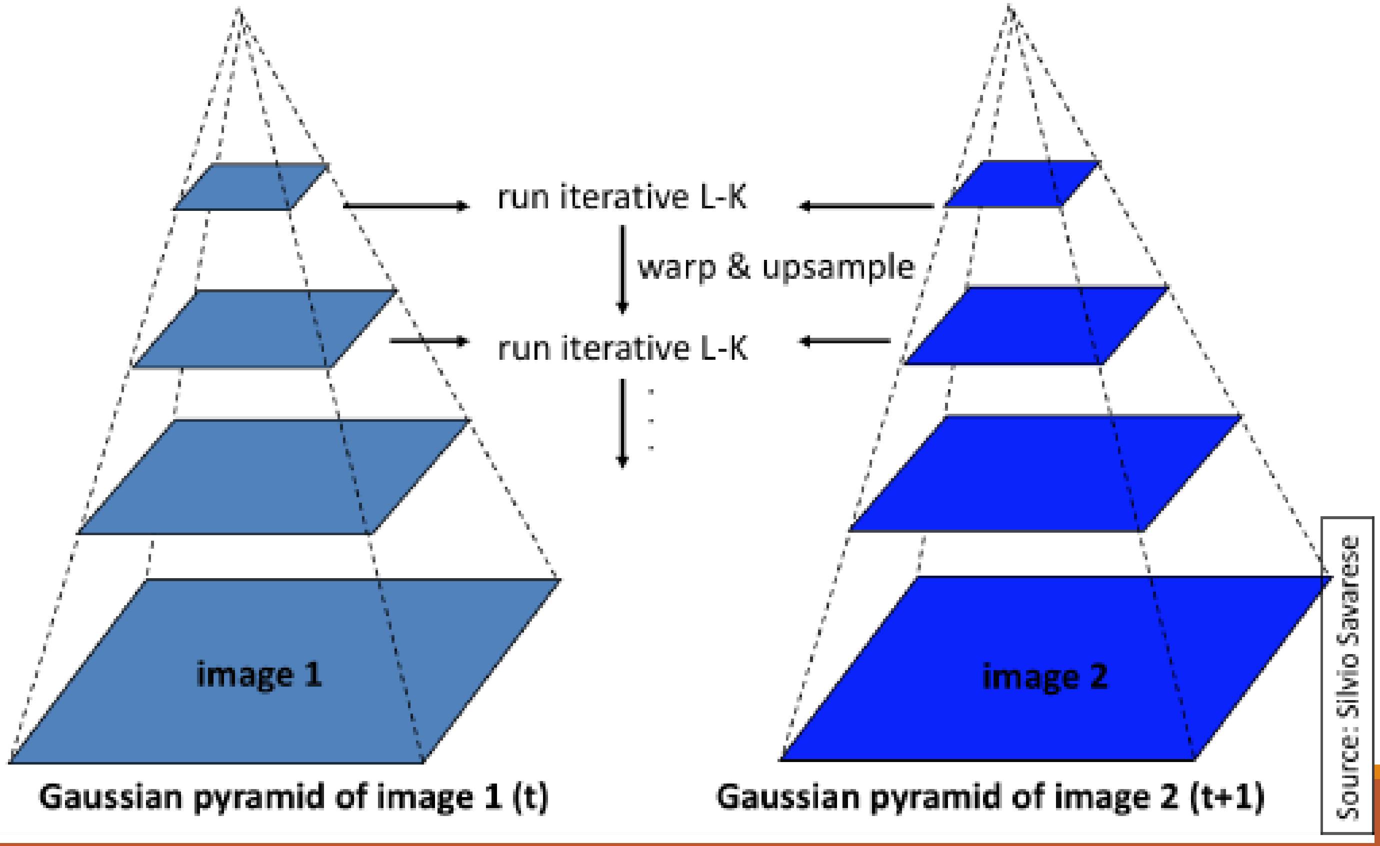
Reduce the resolution

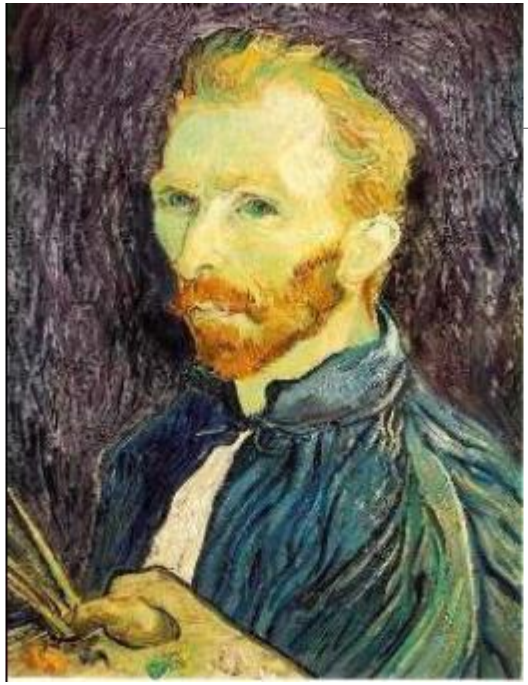
Hierarchical LK

MULTI-SCALE ANALYSIS
IMAGE PYRAMIDS

Hierarchical LK

apply Lucas-Kanade iteratively to a lower resolution version of the image





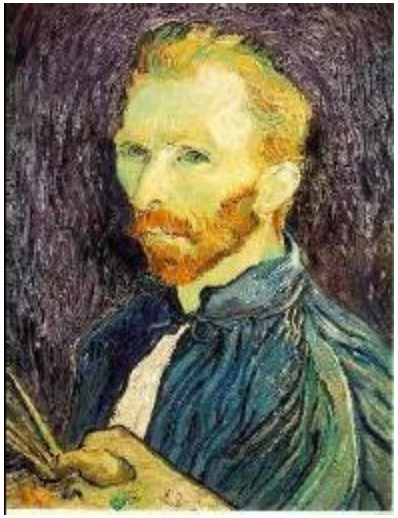
1/2

1/4

1/8

Throw away every other row and column to create a 1/2 size image: *image sub-sampling*

Bad image sub-sampling



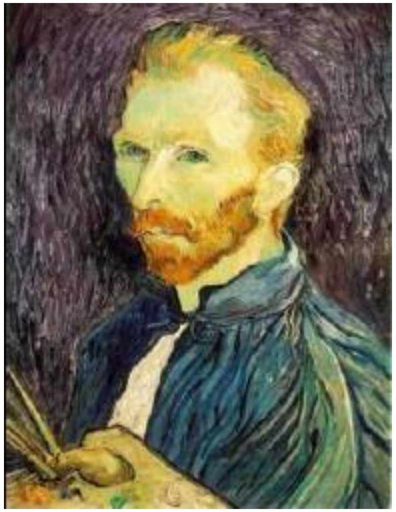
¼ 2x zoom

1/8 4x zoom

Aliasing! What do we do?

Solution: Filter the image, *then* subsample

Subsampling with Gaussian pre-filtering

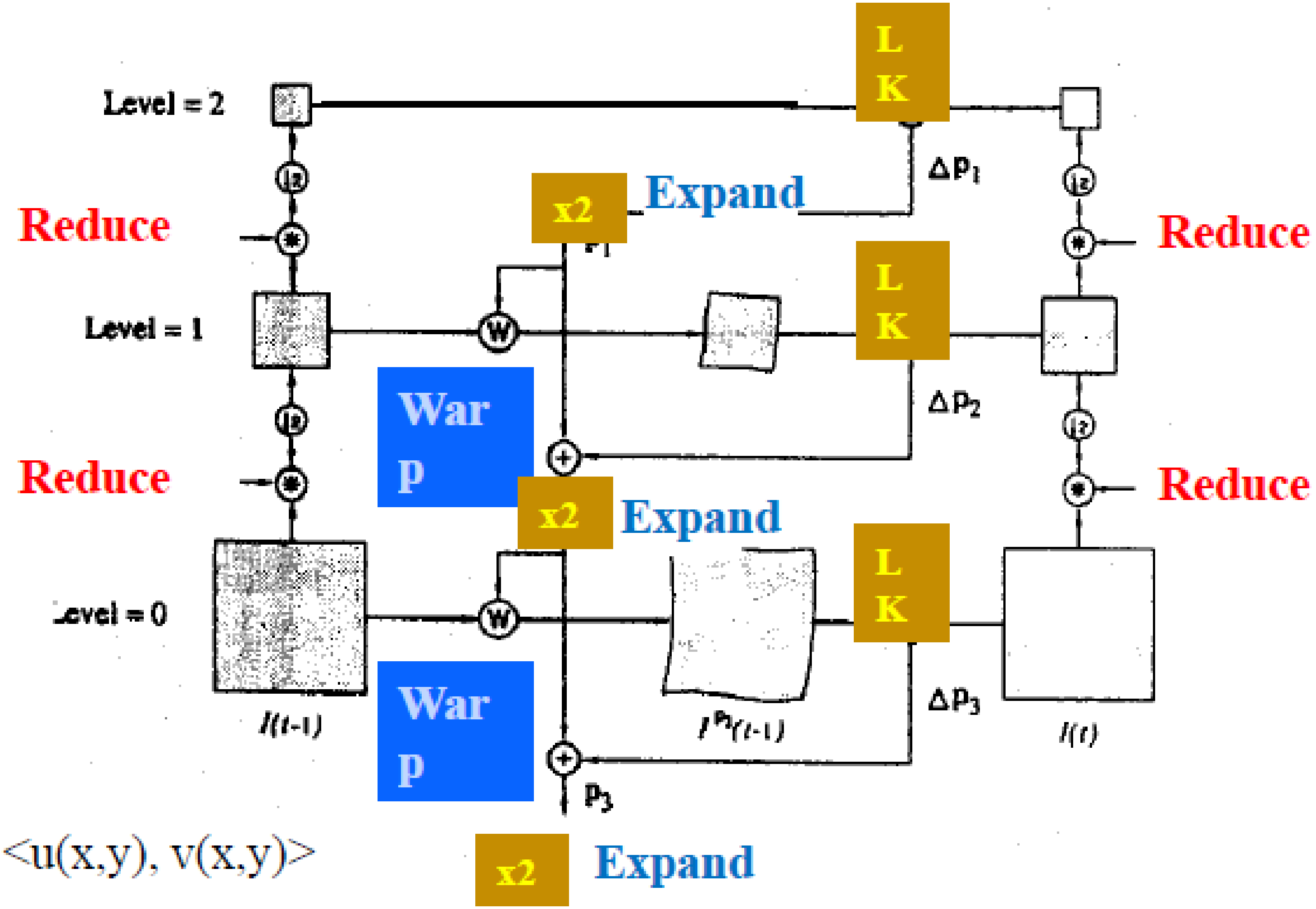


G 1/2

G 1/4

G 1/8

Applying pyramids to LK



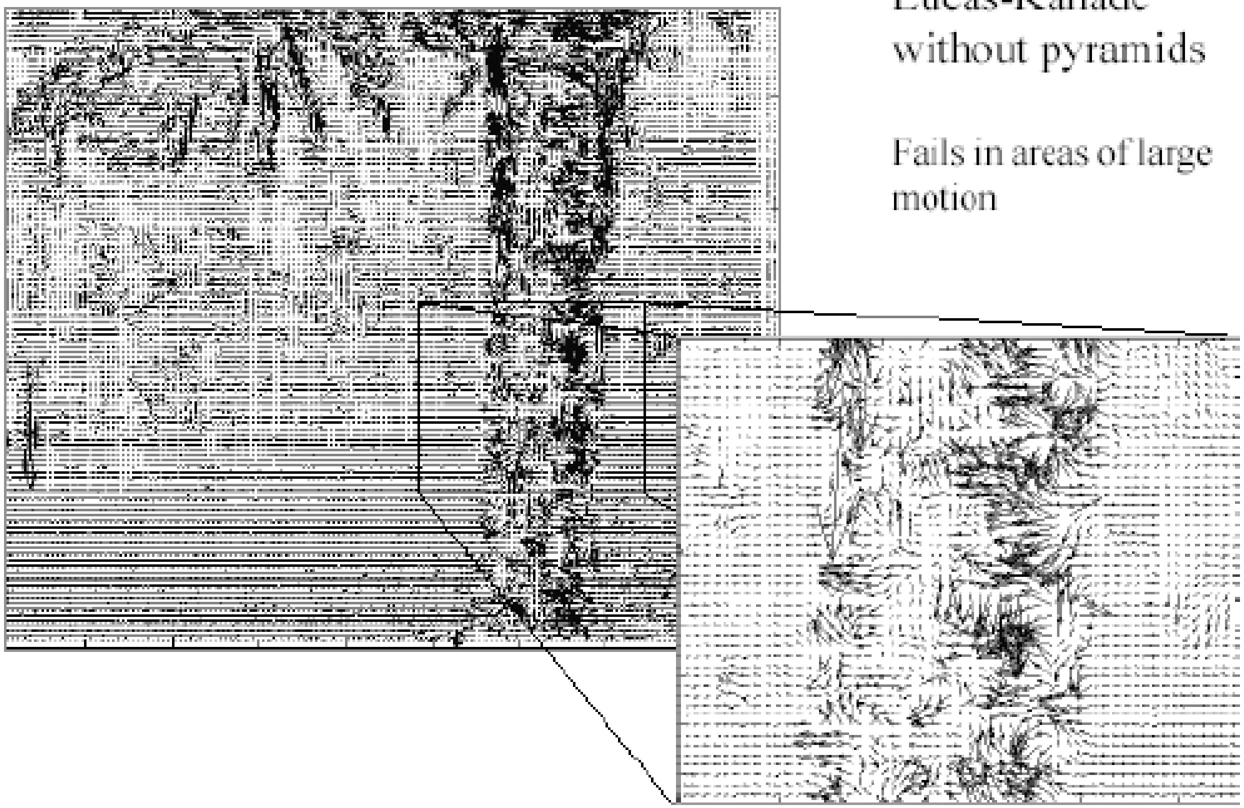
Final <u(x,y), v(x,y)>

Hierarchical LK

1. Apply reduction with Gaussian filtering of both images at time t and t+1 from 0 to K levels
2. Compute Iterative LK at level K and get u_k and v_k
3. For Each Level *i* from *K* to 0
   a) Multiply u_i and v_i by 2 to get predicted flow
   b) Wrap level *i-1* Gaussian version image *I* at *t* according to the predicted flow to create *I'*
   c) Apply iterative LK at level *i-1* between images *I'* and *I* at *t+1* to get u_{i-1} and v_{i-1}
   d) Apply corrections to u_{i-1} and v_{i-1}

$$u_{i-1} = u_{i-1} + u_i$$
$$v_{i-1} = v_{i-1} + v_i$$

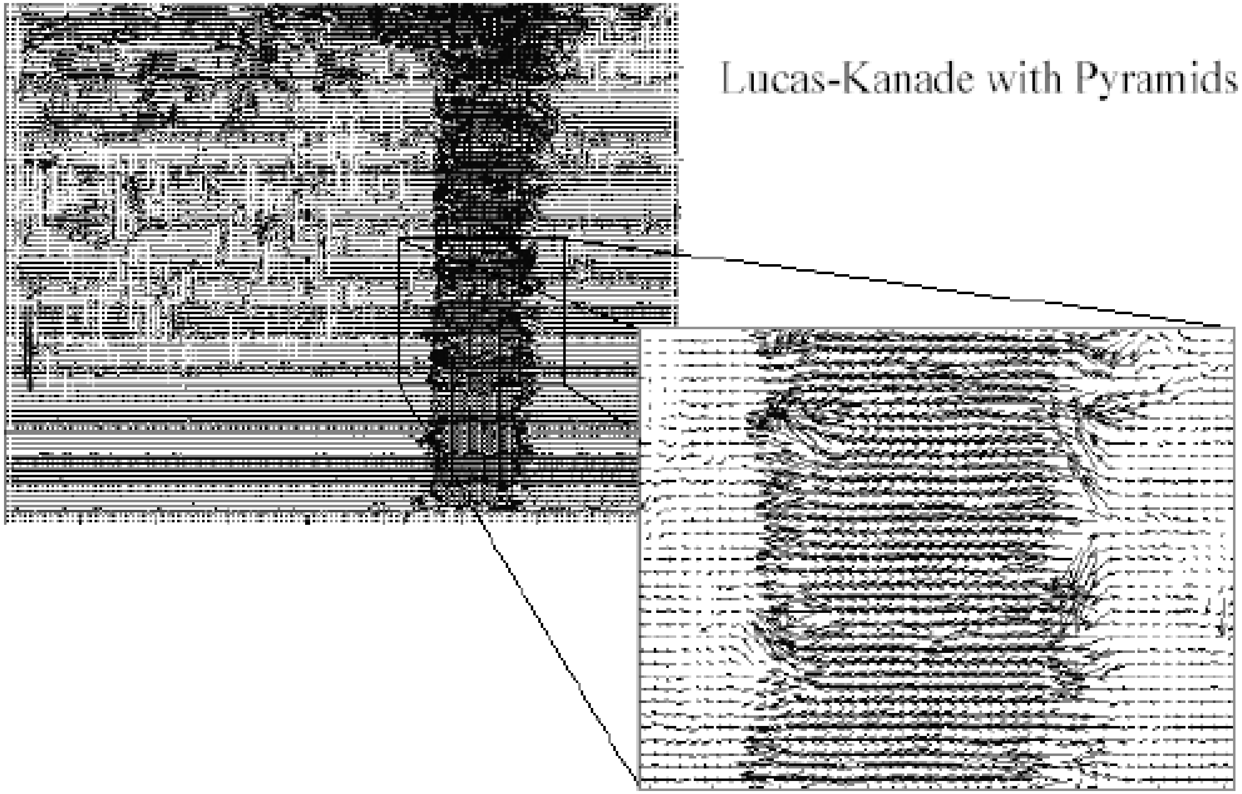
Optical Flow Results



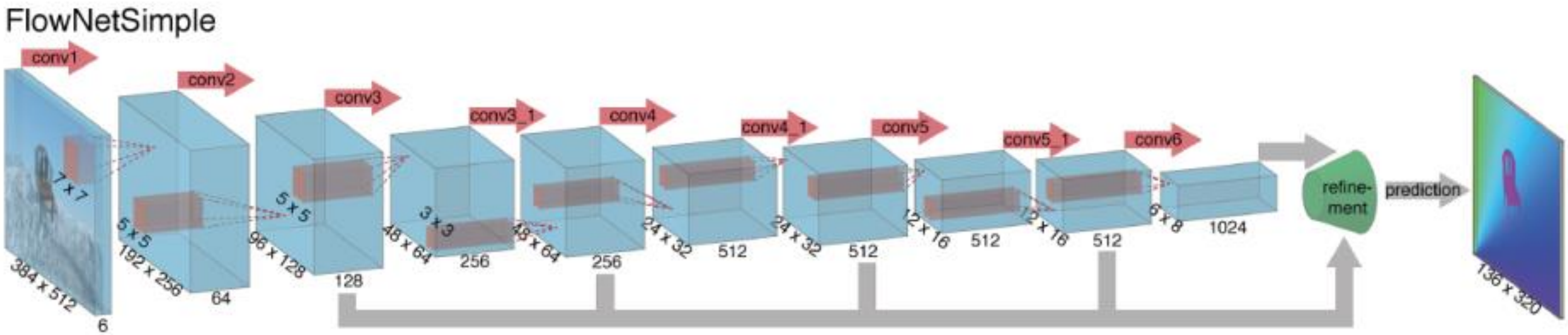
Lucas-Kanade without pyramids

Fails in areas of large motion

Optical Flow Results



Deep learning to optical flow estimation



FlowNet, 2015

FlowNetSimple (FlowNetS)

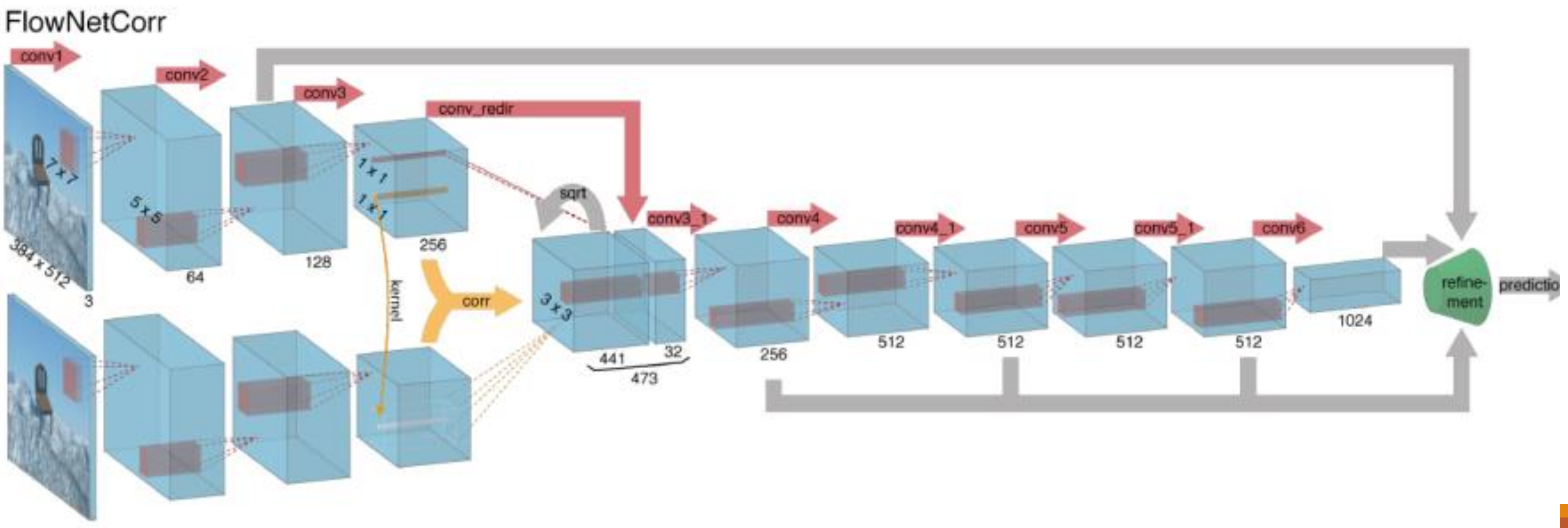
Stack two sequentially adjacent input color images together into 6-channel image and feed them through the network to CNN

Dosovitskiy et al. Flownet: Learning optical flow with convolutional networks, 2015.

FlowNet, 2015

FlowNetCorrelation (FlowNetCorr)

Produce representations of the two images separately, and then combines them together in the 'correlation layer', and learn the higher representation together.

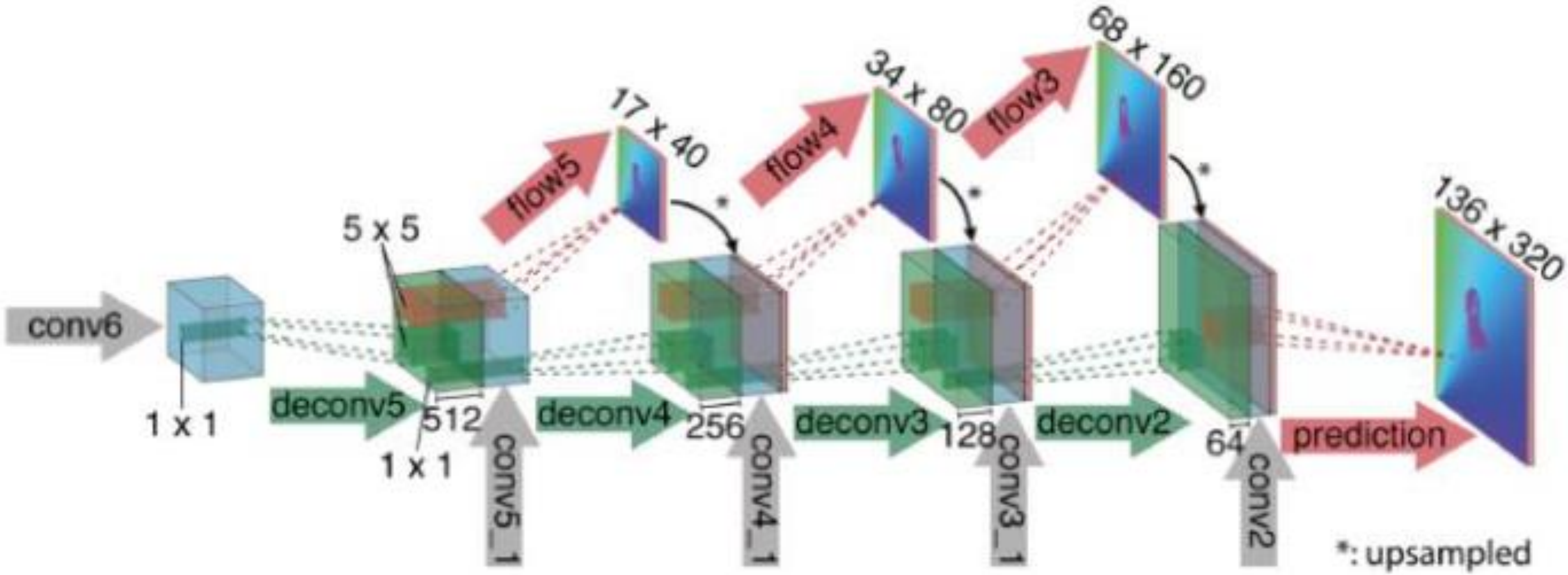


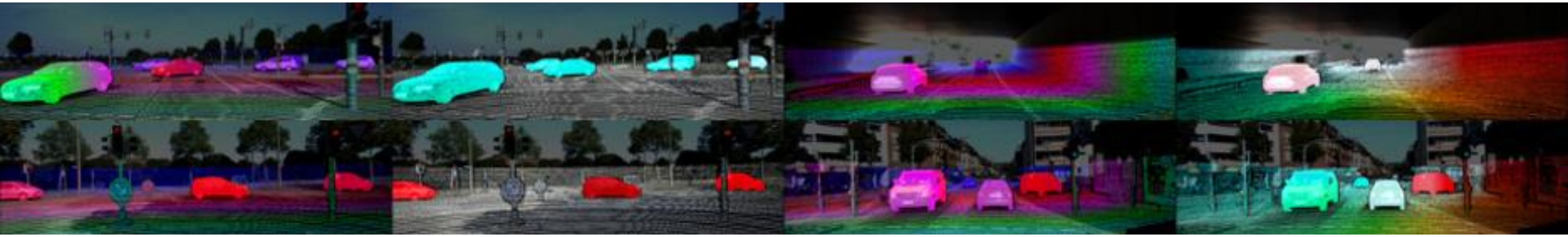
The Correlation layer is used to perform multiplicative patch comparisons between two feature maps.

Given two multi-channel feature maps f_1, f_2, with w, h, and c being their width, height and number of channels. The 'correlation' of two patches centered at x_1 in the first map and x_2 in the second map is then defined as:

$$c(\mathbf{x}_1, \mathbf{x}_2) = \sum_{\mathbf{o} \in [-k,k] \times [-k,k]} \langle \mathbf{f}_1(\mathbf{x}_1 + \mathbf{o}), \mathbf{f}_2(\mathbf{x}_2 + \mathbf{o}) \rangle$$

FlowNET: Refinement





KITTI Vision Benchmark

http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=flow

3D laser scanning for urban environments from a car.

Contains 194 image pairs.

Optical flow dataset is one dataset from that collection, others for tracking and detection.

3D models are fitted to the ground truth and each of the moving objects to reconstruct correct optical flow between frames.

Credit for

CS 4495 Computer Vision (Spring 2015)

A. Bob - College of Computing, Georgia Tech.

Lessons: 6A-L1 , 6B-L1, 6B-L2, 6B-L3

Advanced Machine Learning Specialization" by National Research University Higher School of Economics, Russia – (Coursera)

Week 4: object Tracking & Action Recognition