

# Feature Tracking and Optical Flow

Dr. Sander Ali Khowaja,

Assistant Professor, Department of Telecommunication Engineering  
Faculty of Engineering and Technology, University of Sindh, Pakistan

Senior Member, IEEE – Member, ACM

<https://sander-ali.github.io>

## Computer Vision & Image Processing



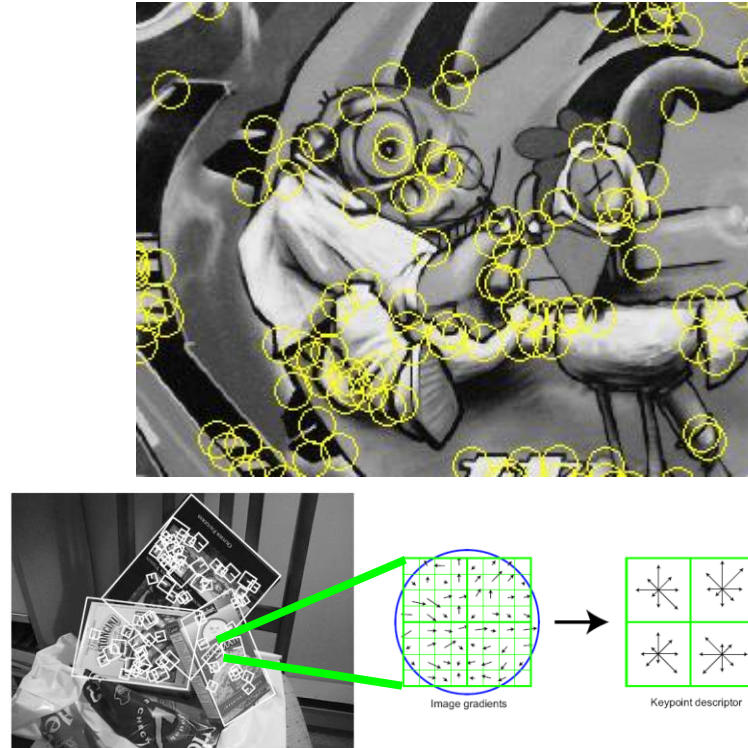
- Interest point/keypoint/feature detectors
  - Harris: detects corners
  - DoG: detects peaks/troughs
- Interest point/keypoint/feature descriptors
  - SIFT (do read the paper)

[Distinctive image features from scale-invariant keypoints](#)

[DG Lowe](#) - International journal of computer vision, 2004 - Springer

Abstract This paper presents a method for extracting distinctive invariant features from images that can be used to perform reliable matching between different views of an object or scene. The features are invariant to image scale and rotation, and are shown to provide ...

Cited by 36572 Related articles All 214 versions Web of Science: 13301 Import into BibTeX



- Feature matching
  - Ratio distance =  $\frac{||f_1 - f_2||}{||f_1 - f_2'||||}$
  - Remove 90% false matches, 5% of true matches in Lowe's study



# This class: recovering motion

- 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

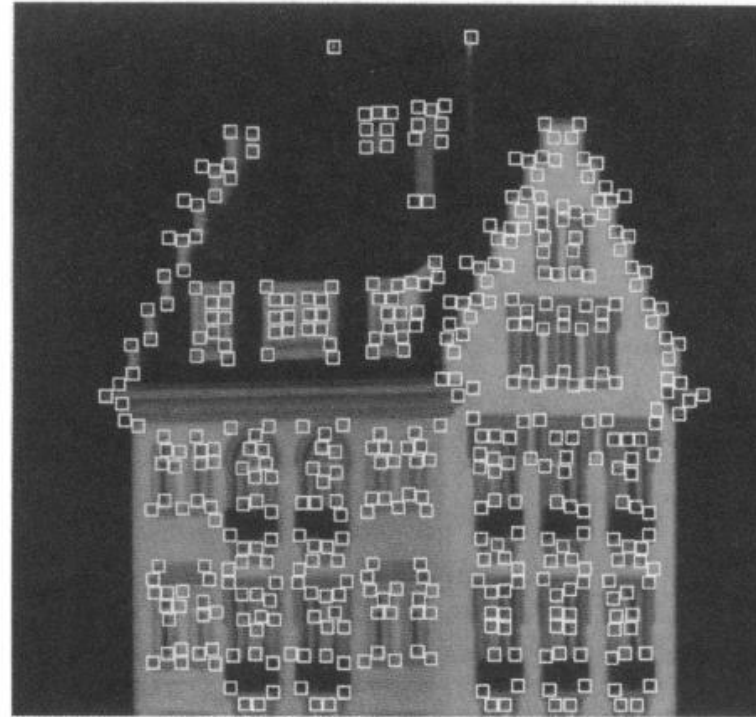
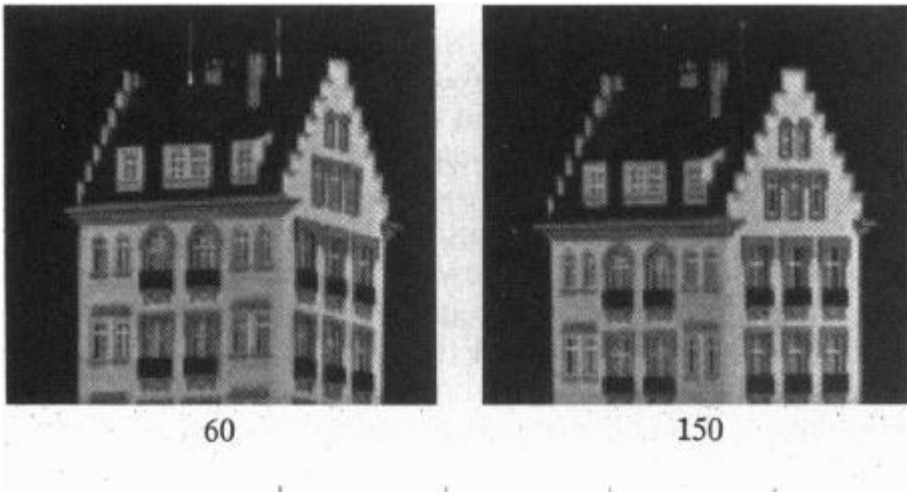
Two problems, one registration method

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*, 1981.



# Feature Tracking

- Many problems, such as structure from motion require matching points
- If motion is small, tracking is an easy way to get them



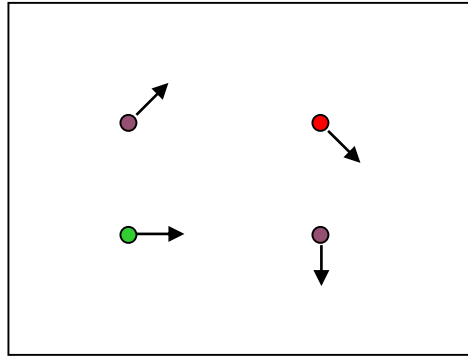
# Feature Tracking - Challenges

- 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

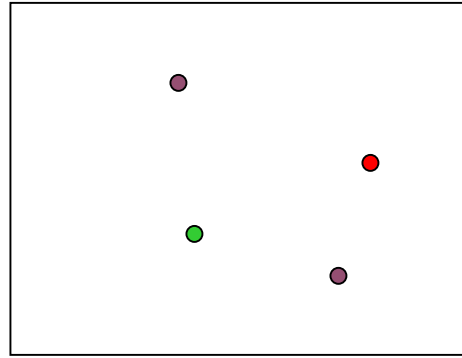




# Feature Tracking



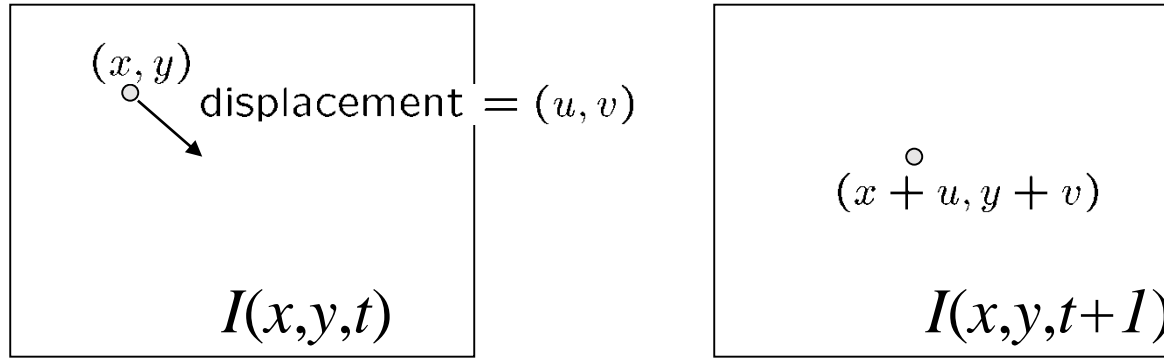
$I(x,y,t)$



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

- 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:

$$I(x + u, y + v, t + 1) \approx I(x, y, t) + \overset{\text{Image derivative along x}}{I_x} \cdot u + I_y \cdot v + \overset{\text{Difference over frames}}{I_t}$$

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

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

# The brightness constancy constraint

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

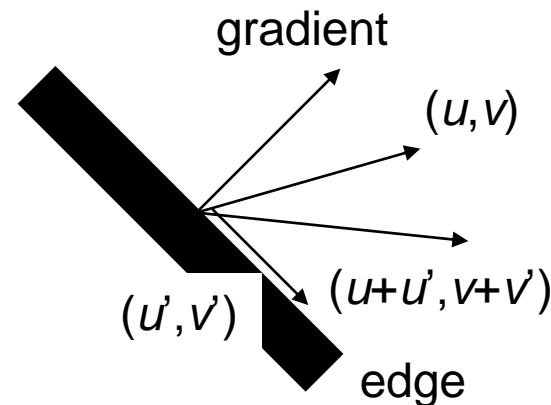
$$\nabla I \cdot [u \ v]^T + I_t = 0$$

- 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

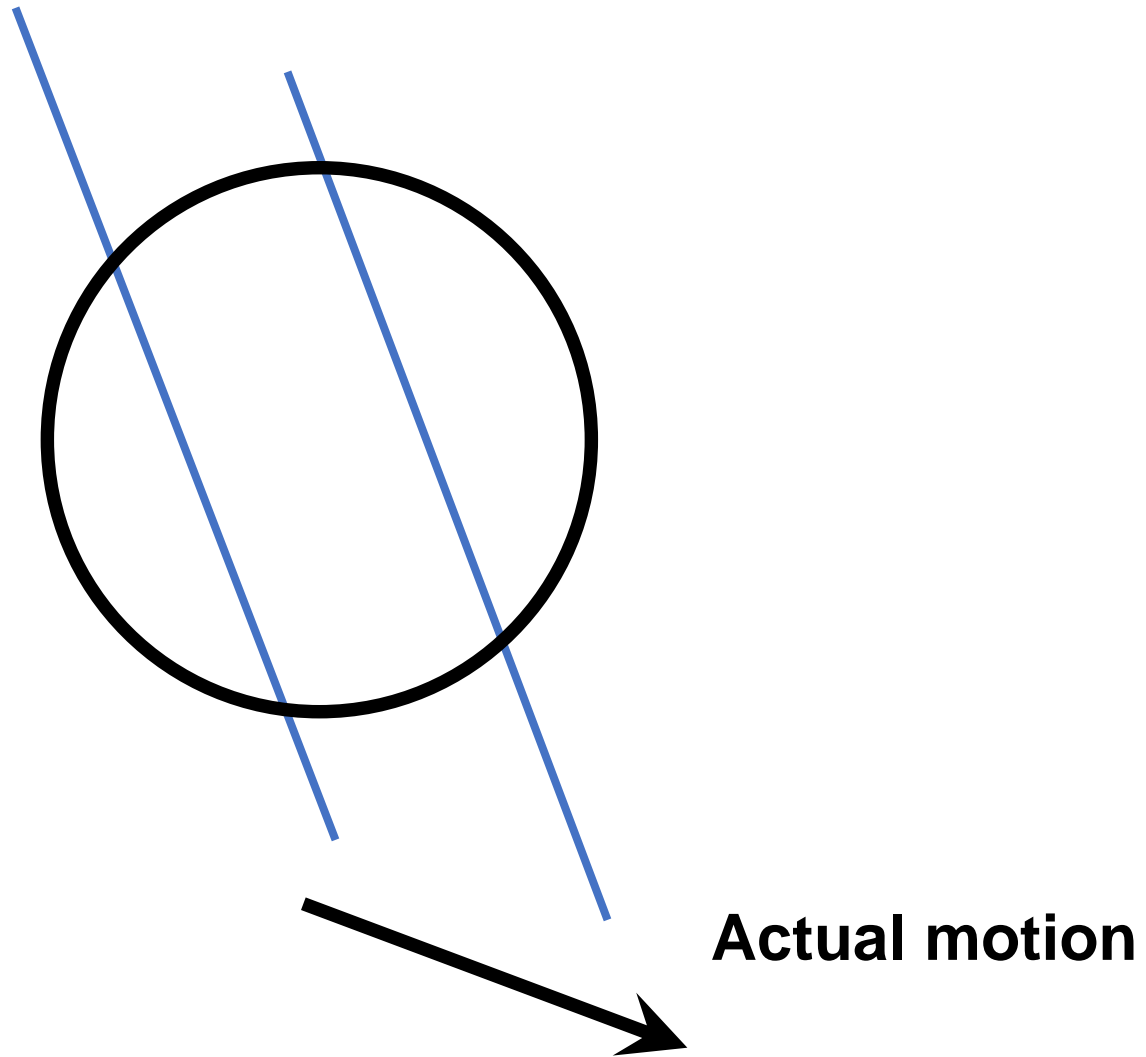
If  $(u, v)$  satisfies the equation, so does  $(u+u', v+v')$  if

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

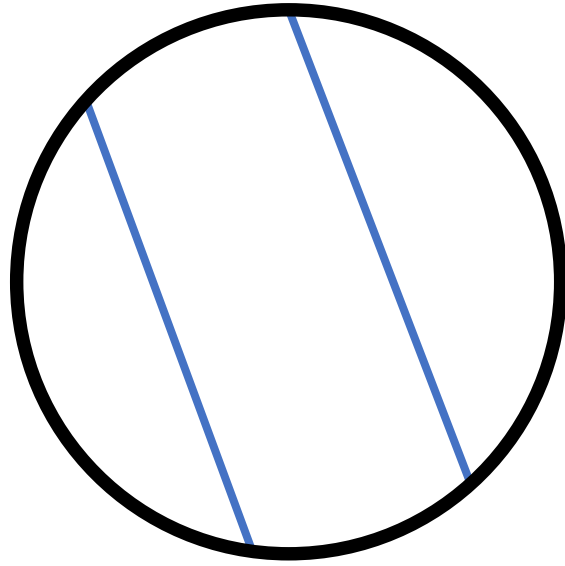




# The aperture problem

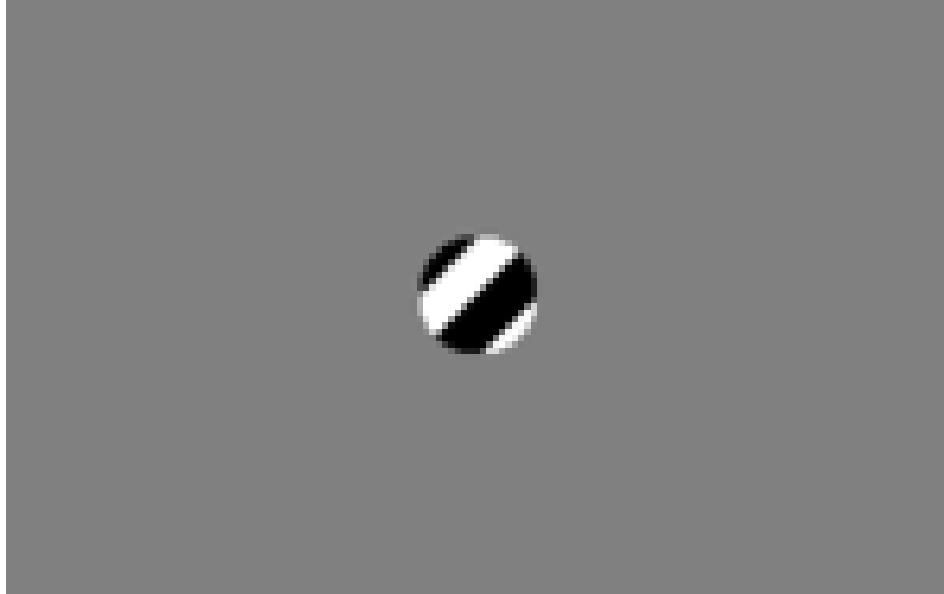


# The aperture problem



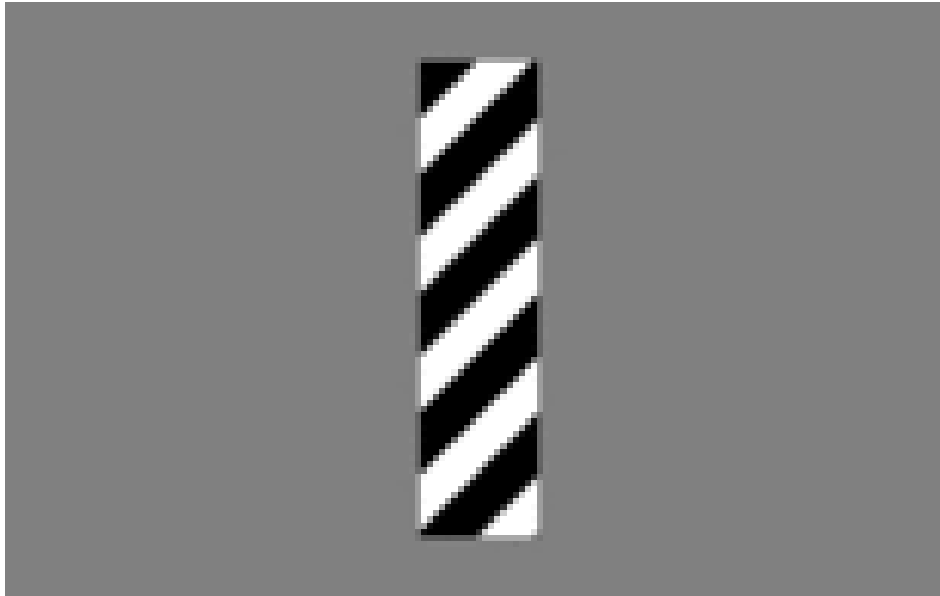
**Perceived motion**

# The barber pole illusion



[http://en.wikipedia.org/wiki/Barberpole\\_illusion](http://en.wikipedia.org/wiki/Barberpole_illusion)

# The barber pole illusion



[http://en.wikipedia.org/wiki/Barberpole\\_illusion](http://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 the 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:

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_{25}) & I_y(p_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_{25}) \end{bmatrix} \quad \begin{matrix} A & d = b \\ 25 \times 2 & 2 \times 1 & 25 \times 1 \end{matrix}$$



# Matching patches across images

- Overconstrained linear system

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_{25}) & I_y(p_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_{25}) \end{bmatrix} \quad \begin{matrix} A & d = b \\ 25 \times 2 & 2 \times 1 & 25 \times 1 \end{matrix}$$

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  $K \times K$  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?

- $A^T A$  should be invertible
- $A^T A$  should not be too small due to noise
  - eigenvalues  $\lambda_1$  and  $\lambda_2$  of  $A^T A$  should not be too small
- $A^T A$  should be well-conditioned
  - $\lambda_1 / \lambda_2$  should not be too large ( $\lambda_1$  = larger eigenvalue)

Does this remind you of anything?

Criteria for Harris corner detector

Dr. Sander Ali Khowaja



# Low-texture region



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

- gradients have small magnitude
- small  $\lambda_1$ , small  $\lambda_2$



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

- gradients very large or very small
- large  $\lambda_1$ , small  $\lambda_2$

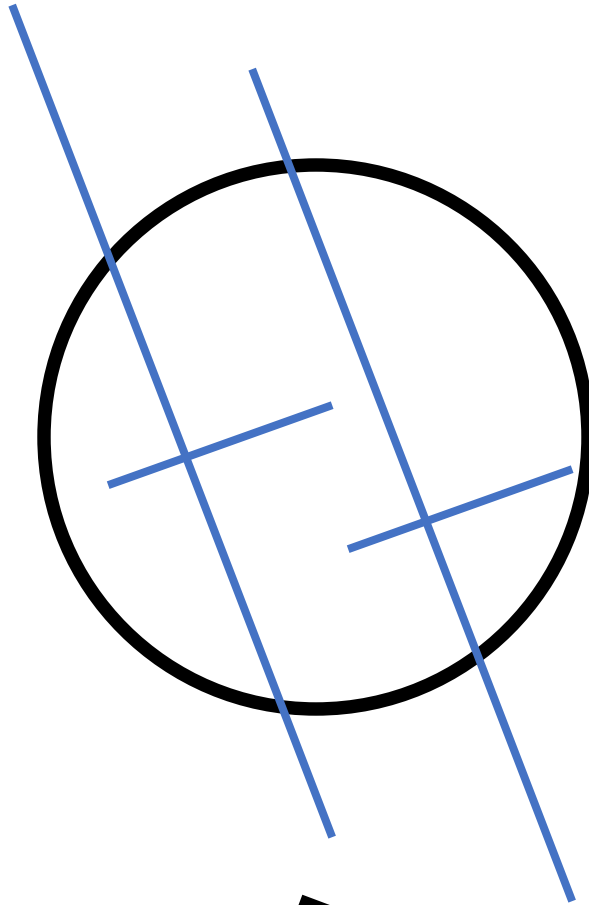
# High Texture Region



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

- gradients are different, large magnitudes
- large  $\lambda_1$ , large  $\lambda_2$

# The aperture problem resolved

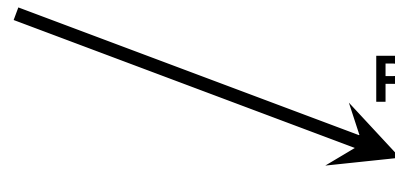
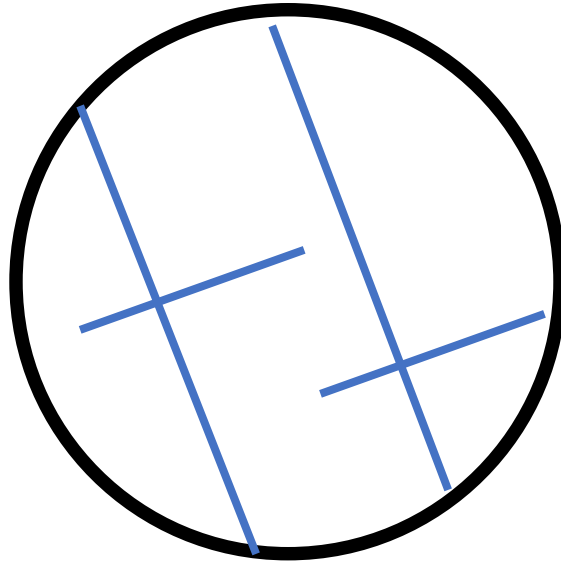


**Actual motion**

Dr. Sander Ali Khowaja



# The aperture problem resolved



**Perceived motion**

# Dealing with larger movements: Iterative Refinement

Original (x,y) position



1. Initialize  $(x', y') = (x, y)$

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



2. Compute (u,v) by

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

2<sup>nd</sup> moment matrix for feature patch in first image

displacement

3. Shift window by (u, v):  $x' = x' + u$ ;  $y' = y' + v$ ;

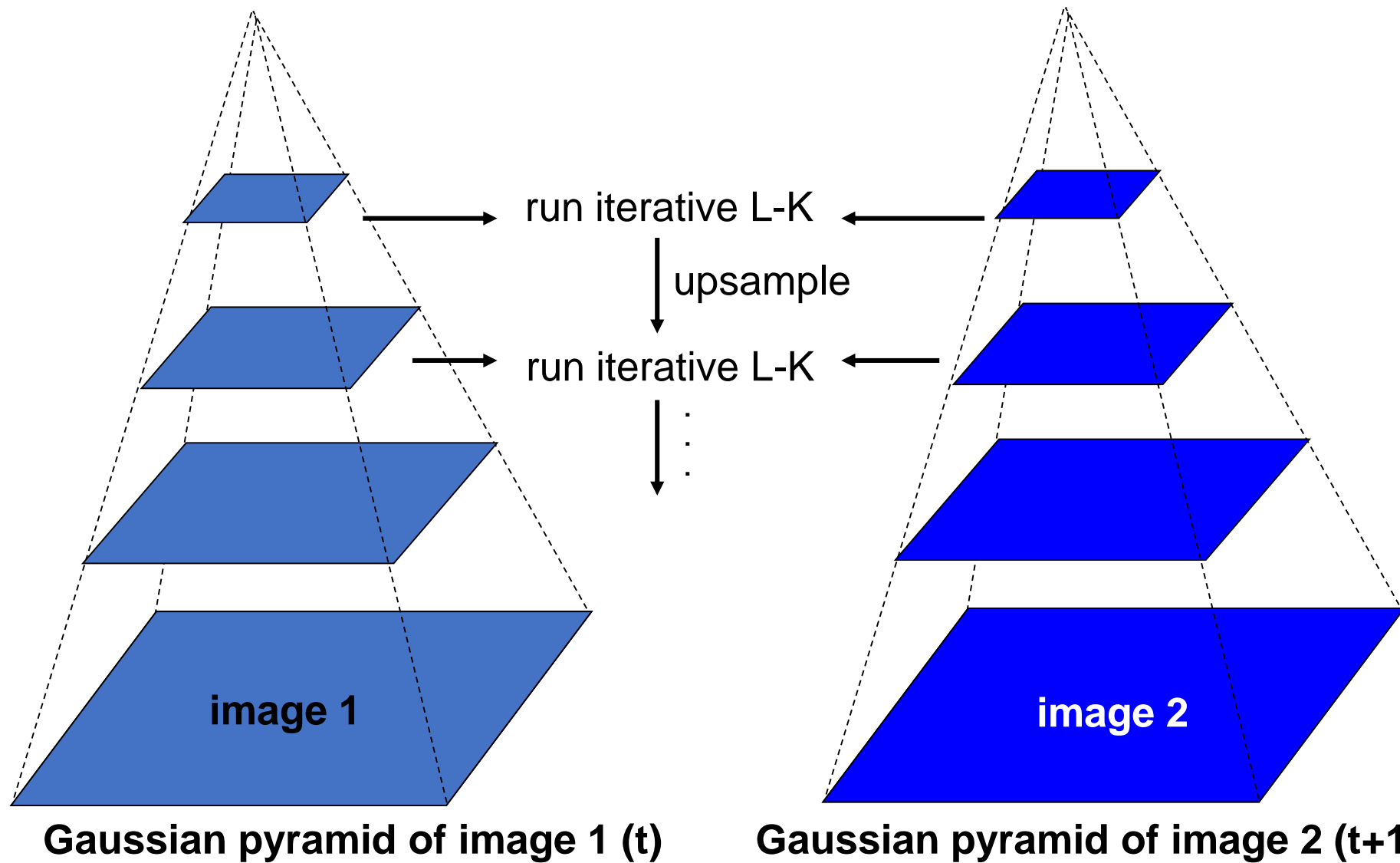
4. Recalculate  $I_t$

5. Repeat steps 2-4 until small change

- Use interpolation for subpixel values



# Dealing with larger movements: coarse-to-fine registration



# 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

J. Shi and C. Tomasi. [Good Features to Track](#). CVPR 1994.



# Tracking Example

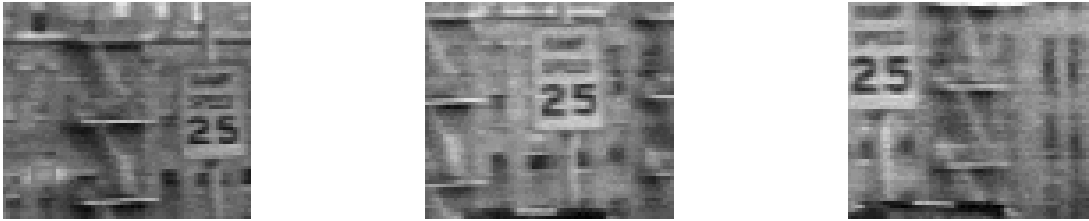


Figure 1: Three frame details from Woody Allen's *Manhattan*. The details are from the 1st, 11th, and 21st frames of a subsequence from the movie.

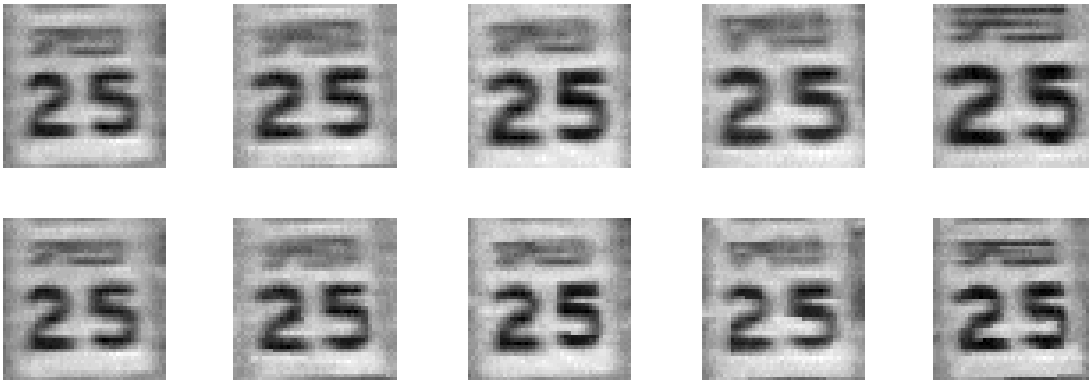


Figure 2: The traffic sign windows from frames 1,6,11,16,21 as tracked (top), and warped by the computed deformation matrices (bottom).

J. Shi and C. Tomasi. [Good Features to Track](#). CVPR 1994.

# Summary of KLT Tracking

- Find a good point to track (harris corner)
- 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



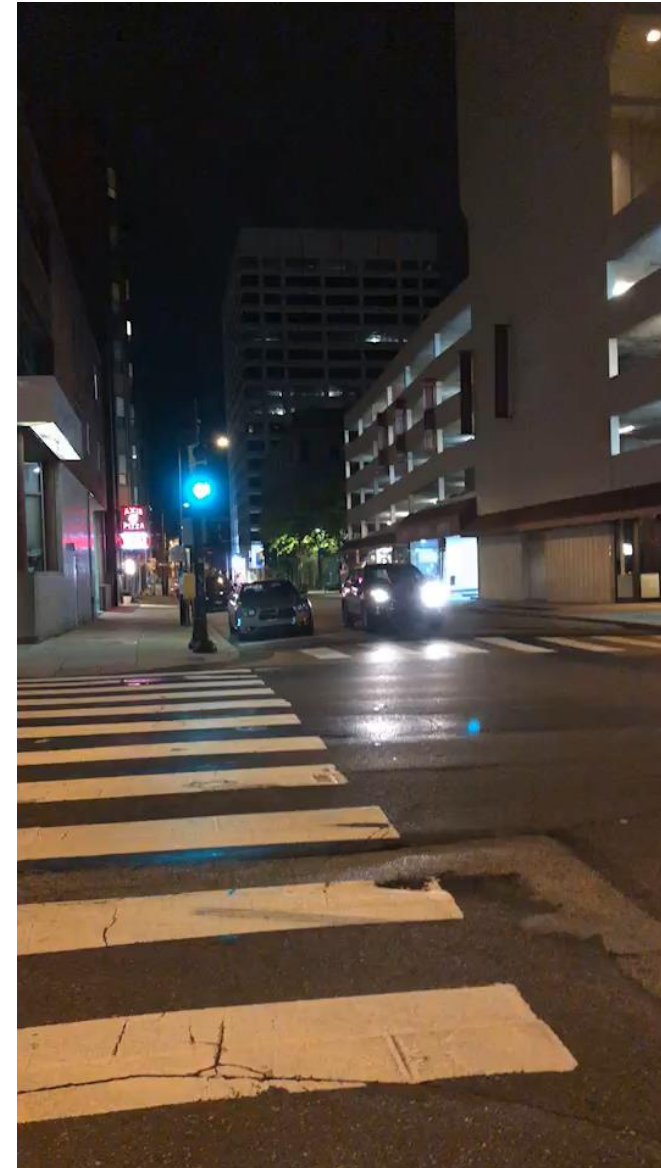


# Implementation Issues

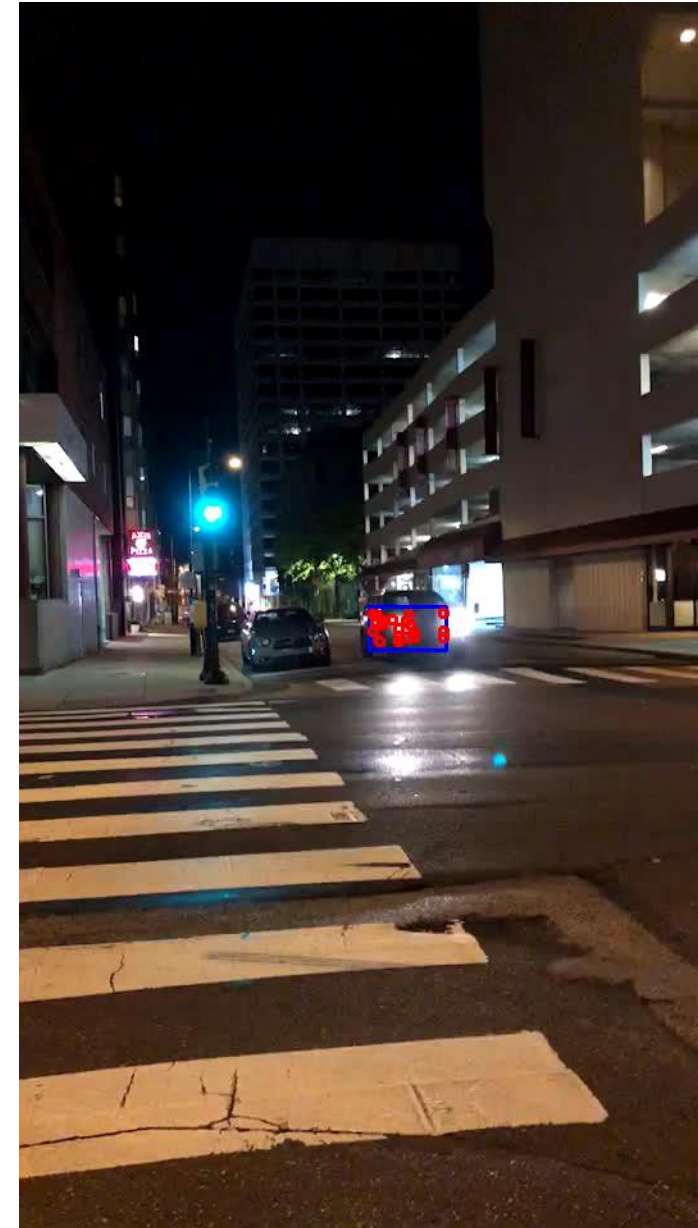
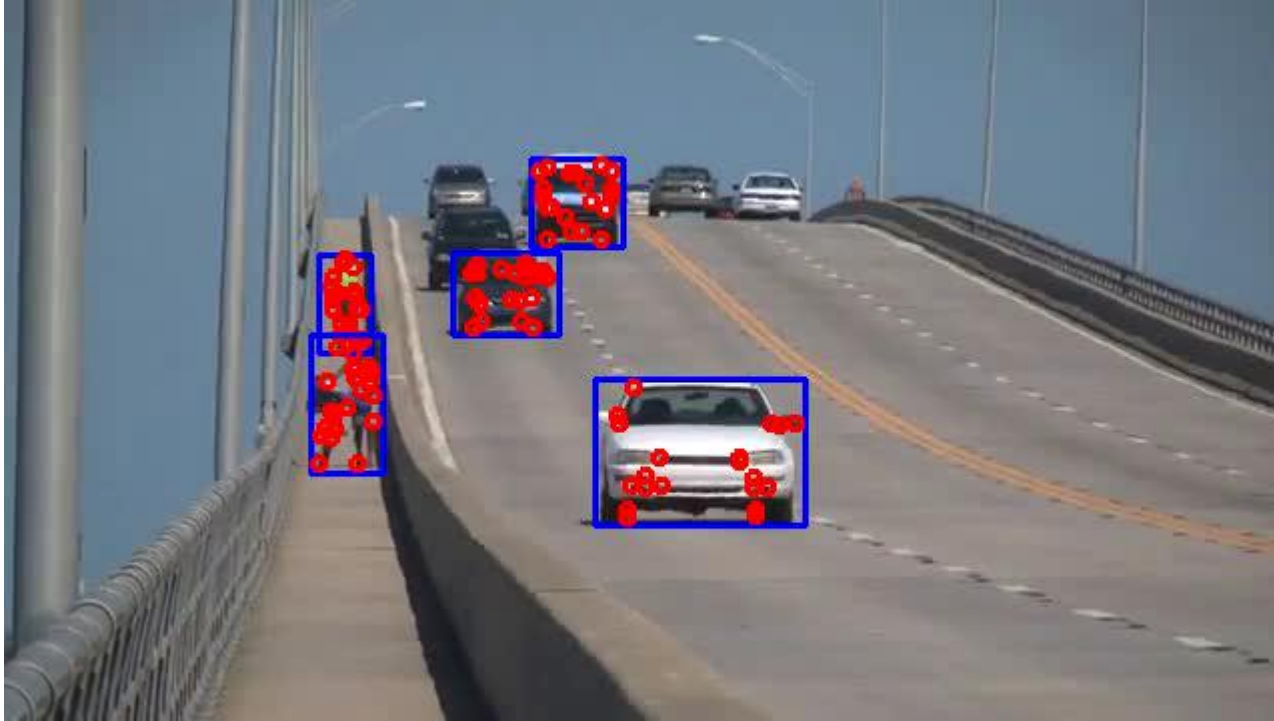
- Window size
  - Small window more sensitive to noise and may miss larger motions (without pyramid)
  - Large window more likely to cross an occlusion boundary (and it's slower)
  - 15x15 to 31x31 seems typical
- Weighting the window
  - Common to apply weights so that center matters more (e.g., with Gaussian)



# Easy to hard tracking problem



# Tracking using KLT



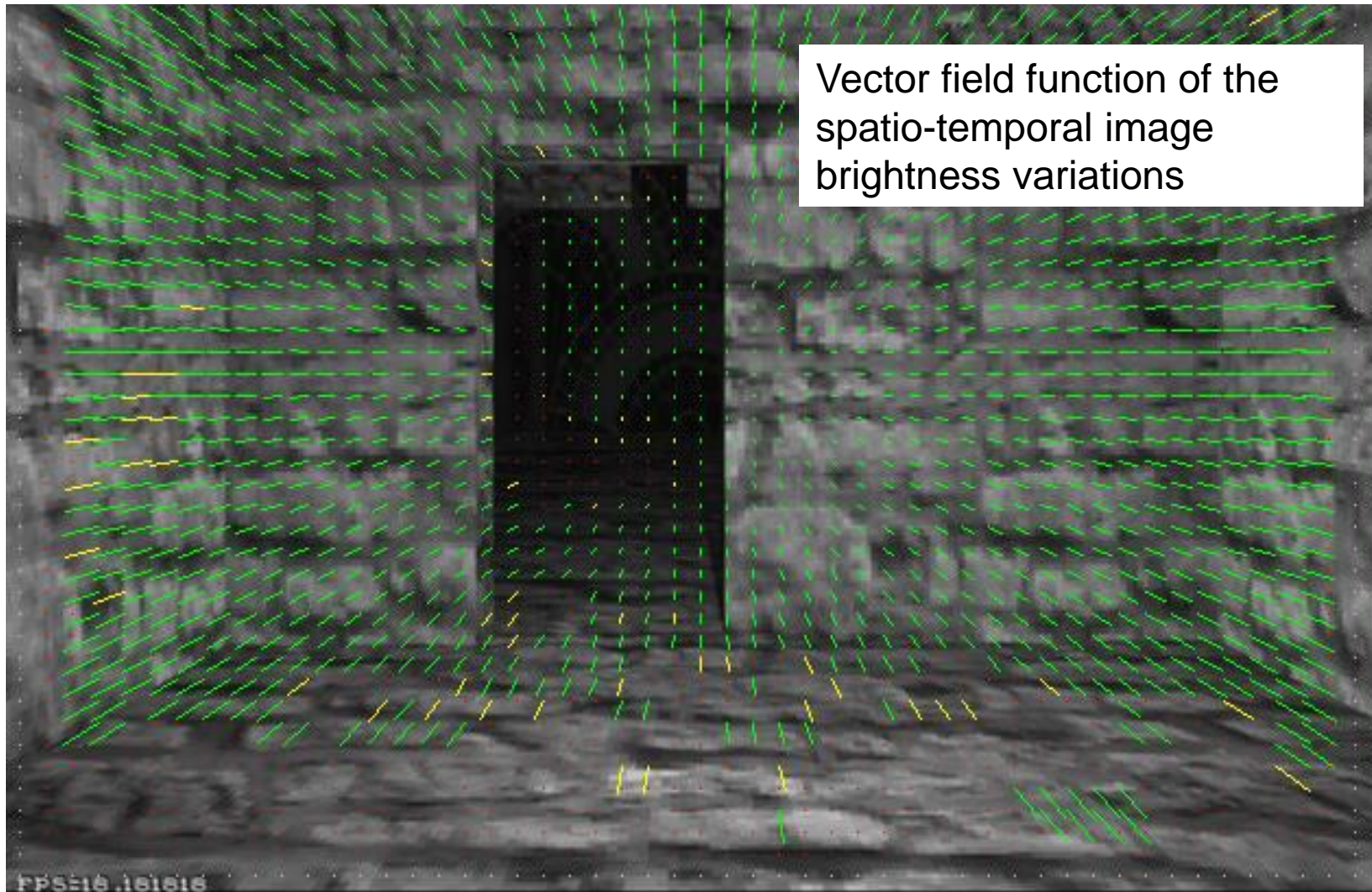
# Why not just do local template matching?

- 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 movements





# Optical Flow



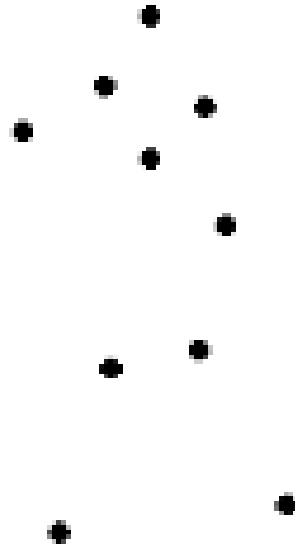
Picture courtesy of Selim Temizer - Learning and Intelligent Systems (LIS) Group, MIT

Dr. Sander Ali Khowaja



# Motion and Perceptual Organization

- Even “impoverished” motion data can evoke a strong percept

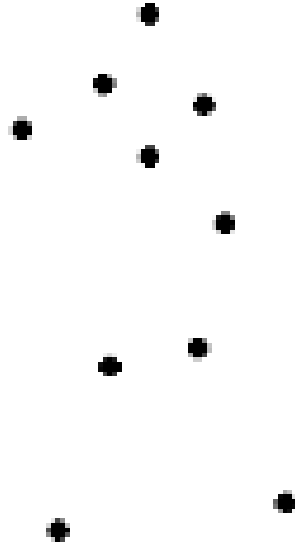


G. Johansson, “Visual Perception of Biological Motion and a Model For Its Analysis”, *Perception and Psychophysics* 14, 201-211, 1973.



# Motion and Perceptual Organization

- Even “impoverished” motion data can evoke a strong percept



G. Johansson, “Visual Perception of Biological Motion and a Model For Its Analysis”, *Perception and Psychophysics* 14, 201-211, 1973.

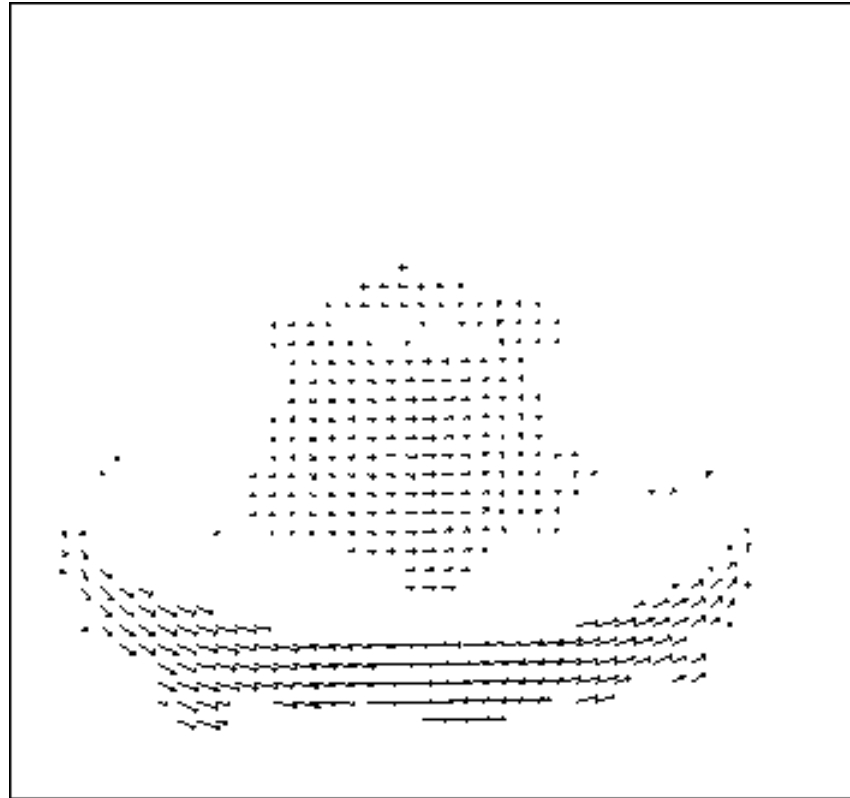
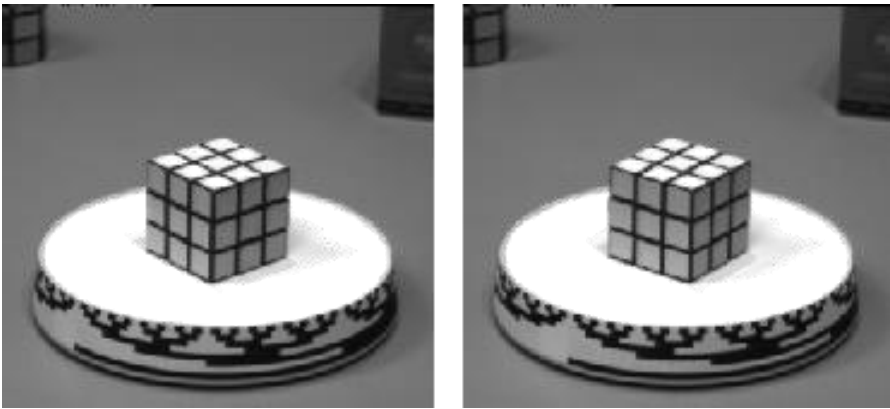
# Uses of motion

- Estimating 3D structure
- Segmenting objects based on motion cues
- Learning and tracking 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



What would the motion field of a non-rotating ball moving towards the camera look like?

# 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
  - Think of a uniform rotating sphere under fixed lighting vs. a stationary sphere under moving illumination



# 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

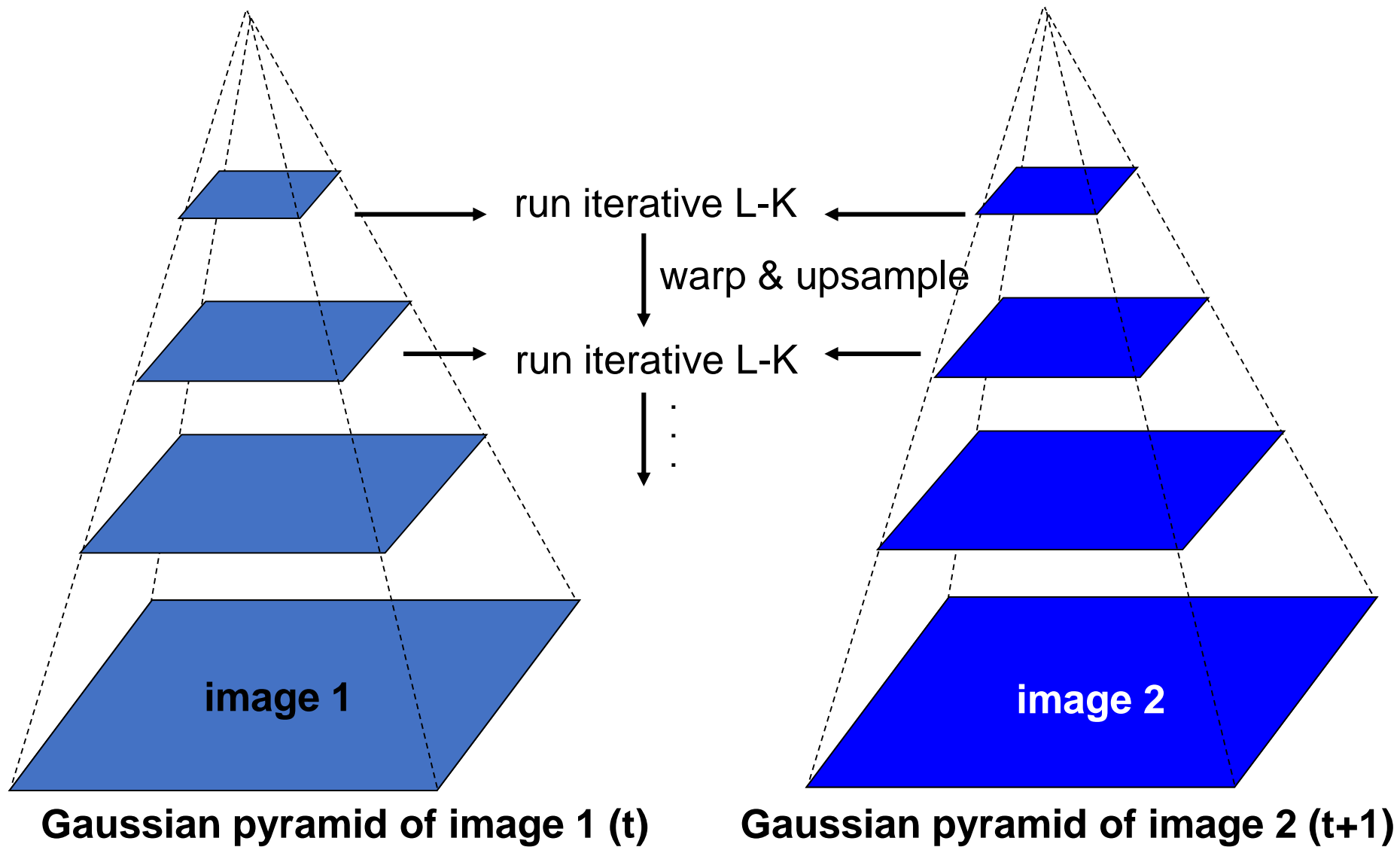


- Iterative Lukas-Kanade Algorithm
  1. Estimate displacement at each pixel by solving Lucas-Kanade equations
  2. Warp  $I(t)$  towards  $I(t+1)$  using the estimated flow field
    - Basically, just interpolation
  3. Repeat until convergence

\* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003



# Coarse-to-fine Optical Flow Estimation

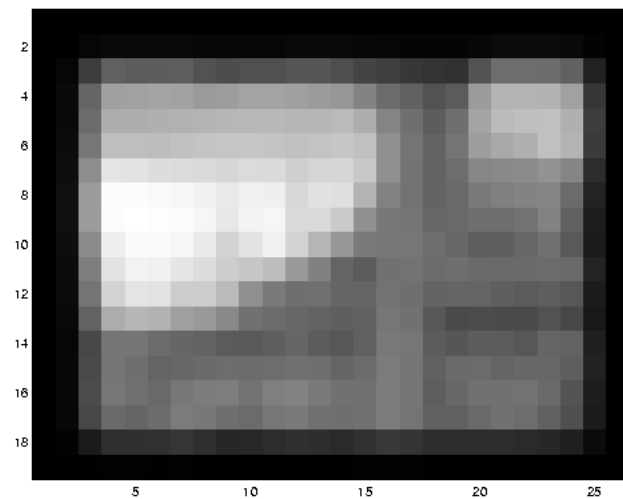
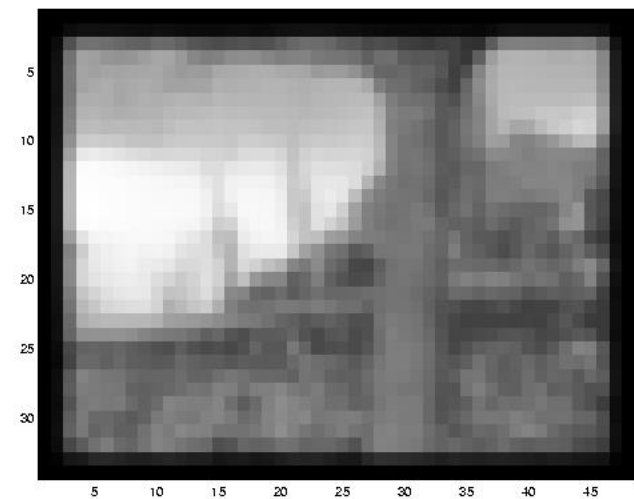
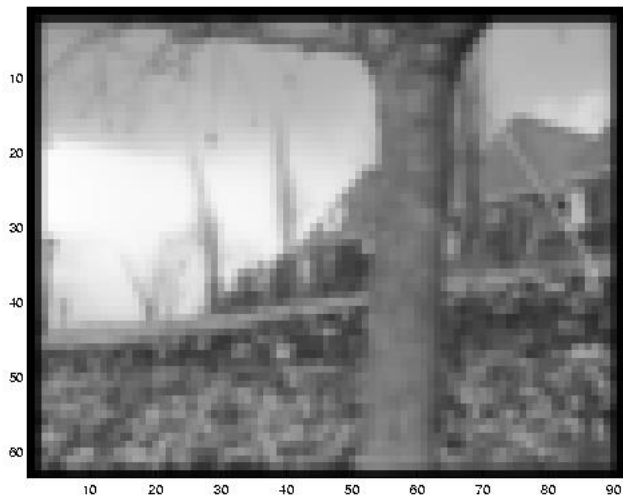




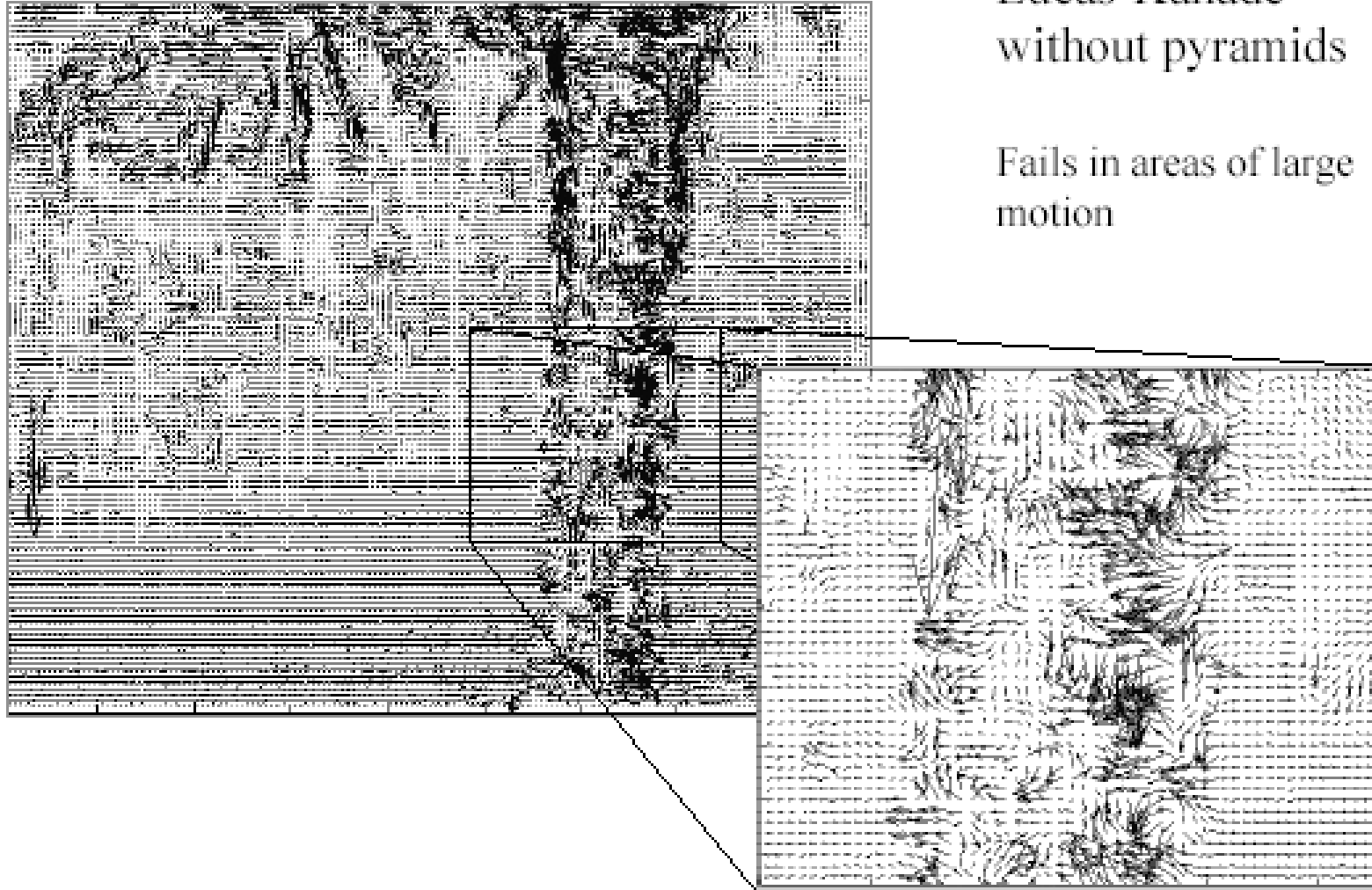
# Example



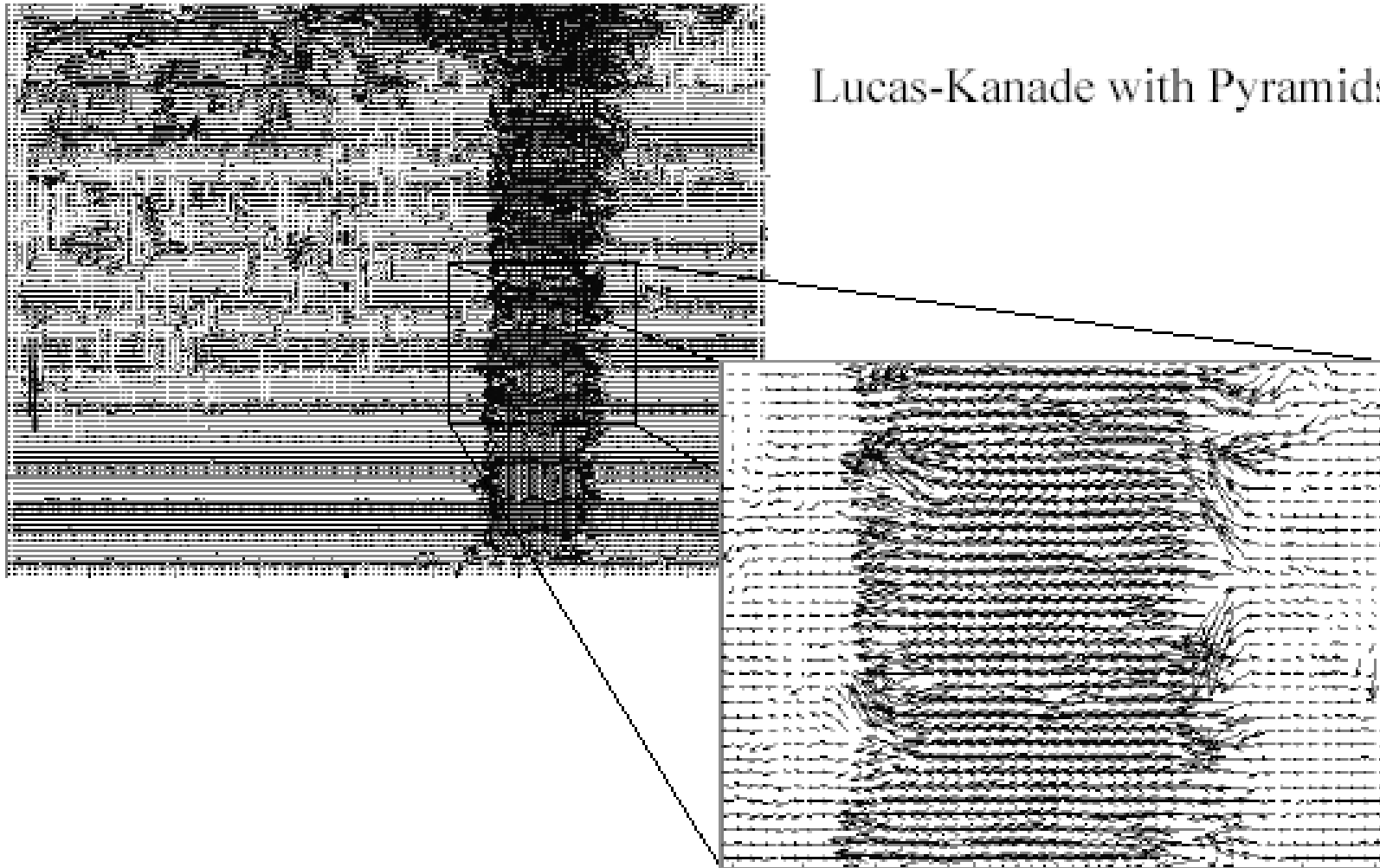
# Multi-resolution registration



# Optical Flow results

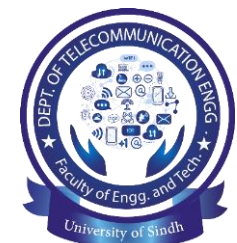


# Optical Flow Results



\* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Dr. Sander Ali Khowaja



# 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



# Python code

 optical flow

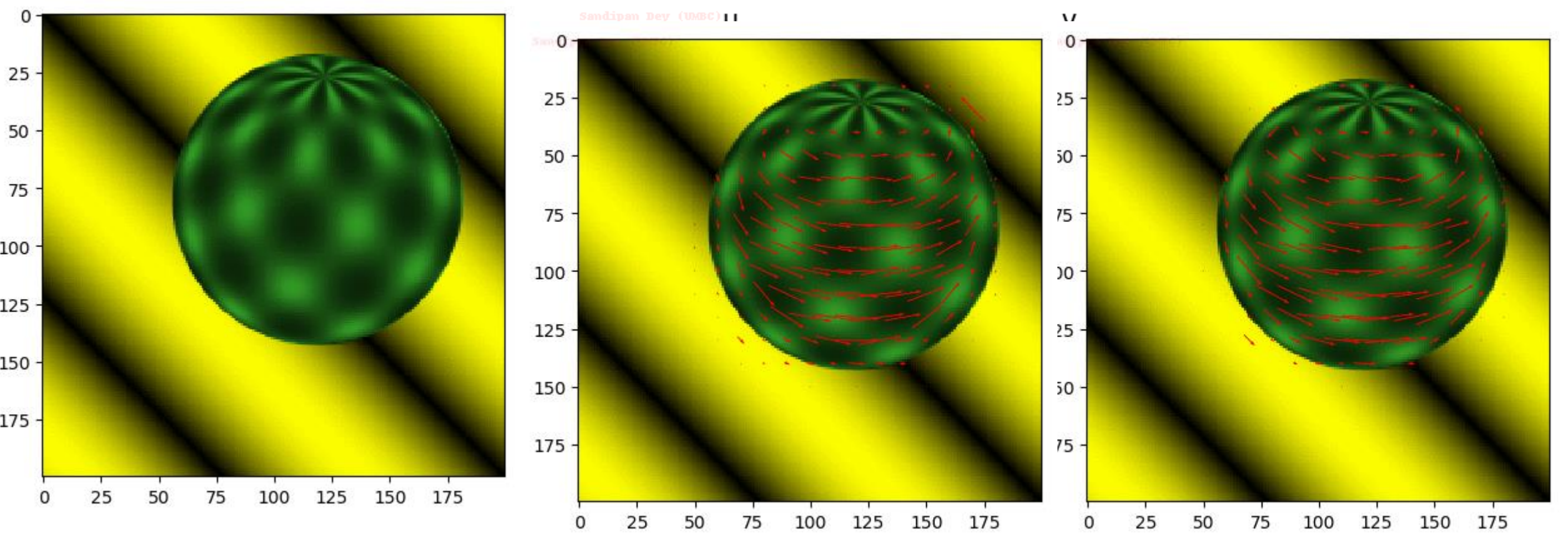


Dr. Sander Ali Khowaja

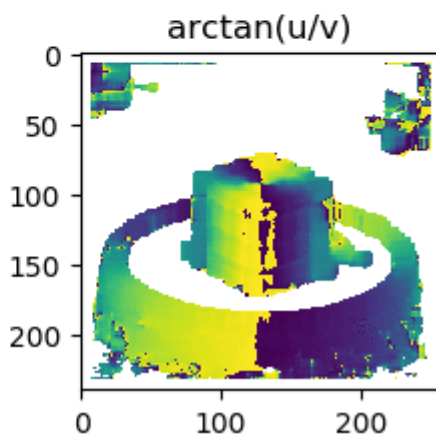
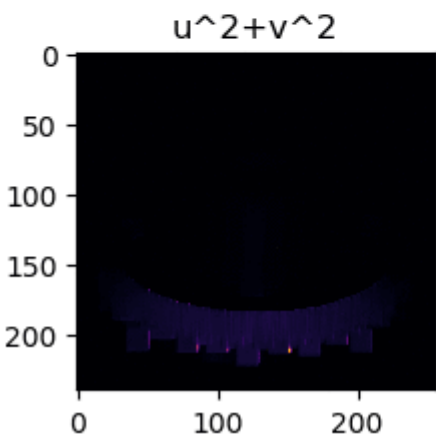
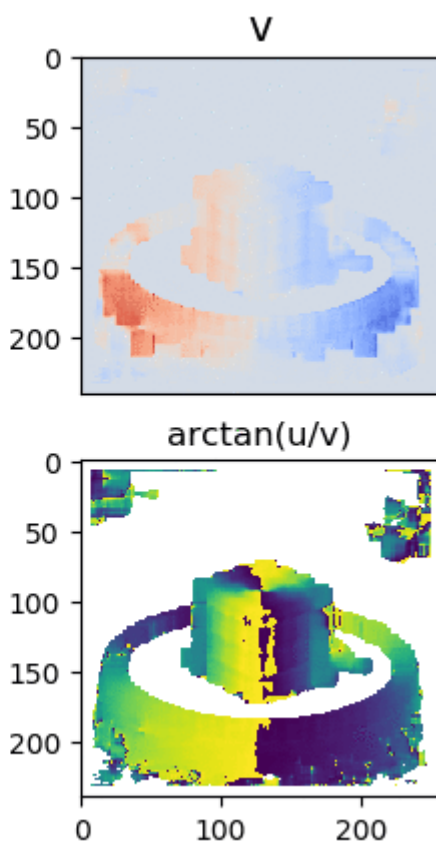
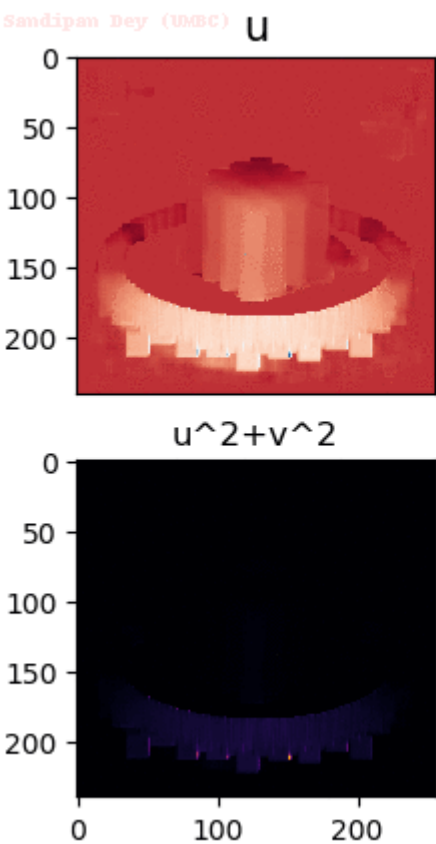
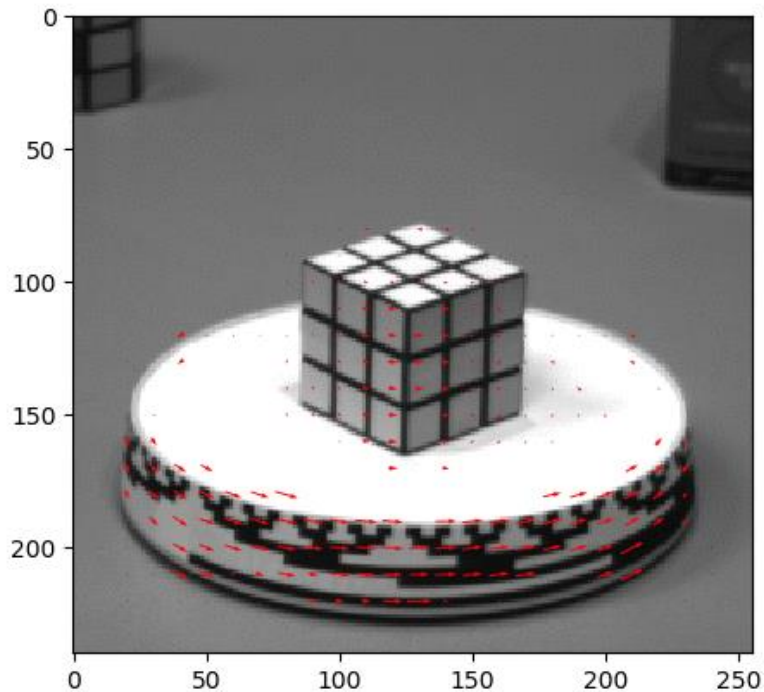
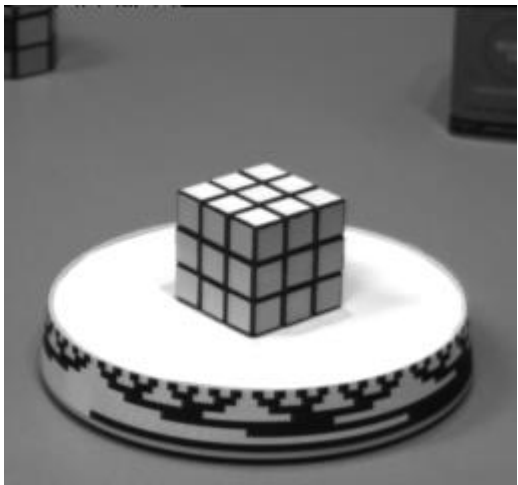




# Some examples



# Some examples

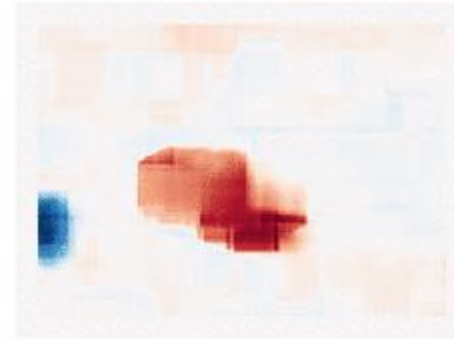


# Some examples

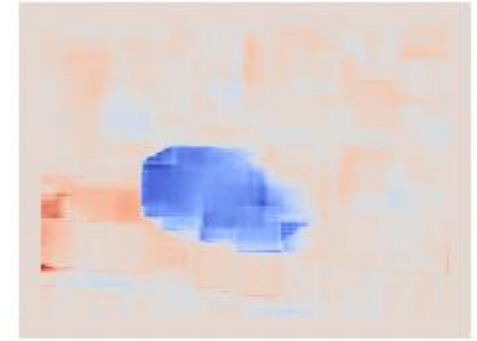


Sandipan Dey (USMC)

u



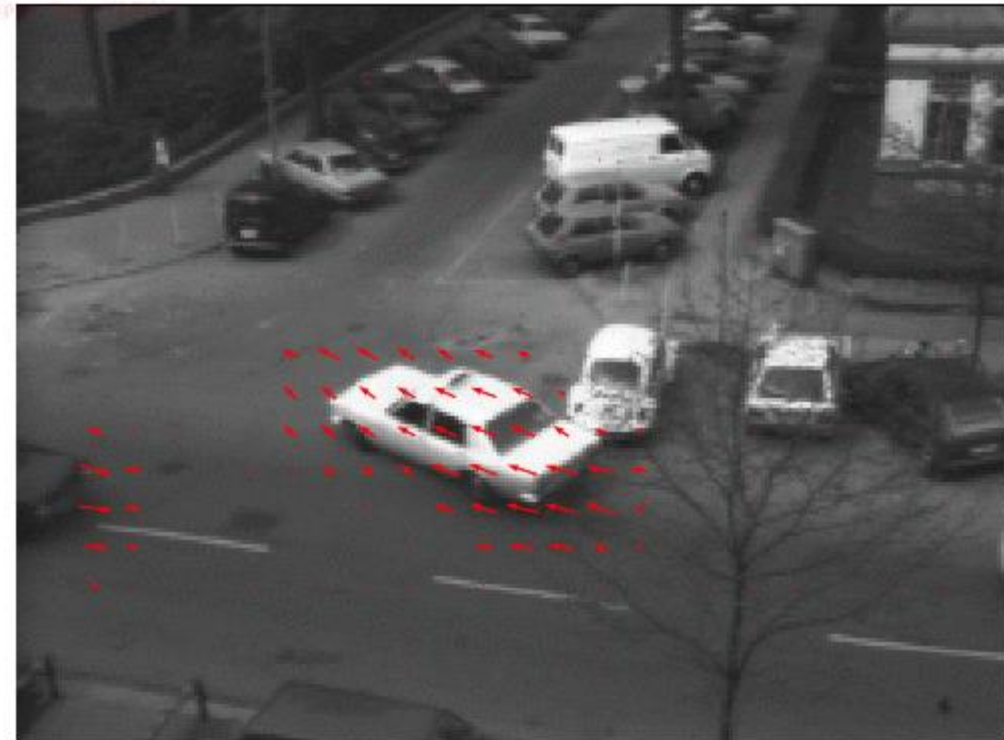
v



$u^2 + v^2$

$\arctan(u/v)$

Sandip



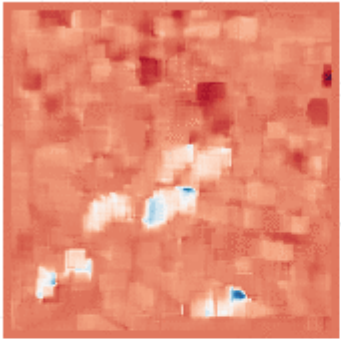


# Some examples

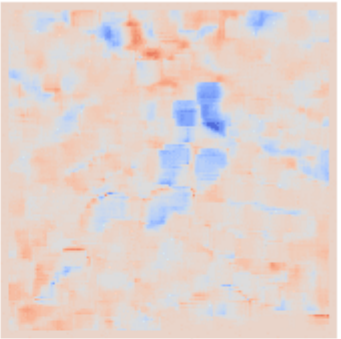


Sandipan Dey (UMBC)

u



v



$u^2+v^2$



$\arctan(u/v)$



# Some examples



Sandipan Dey (UMBC)

u



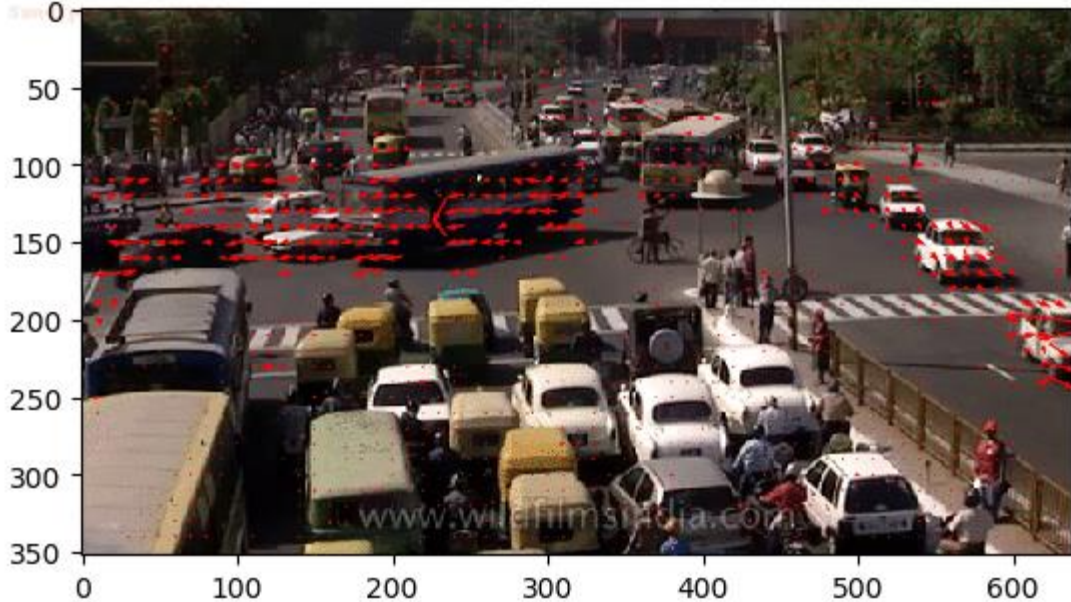
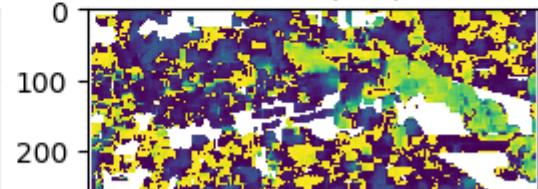
v



$u^2+v^2$



$\arctan(u/v)$





# Things to remember

- Major contributions from Lucas, Tomasi, Kanade
  - Tracking feature points
  - Optical flow
  - Stereo (later)
  - Structure from motion (later)
- Key ideas
  - By assuming brightness constancy, truncated Taylor expansion leads to simple and fast patch matching across frames
  - Coarse-to-fine registration

