

اصول پردازش تصویر

Principles of Image Processing

مصطفی کمالی تبریزی

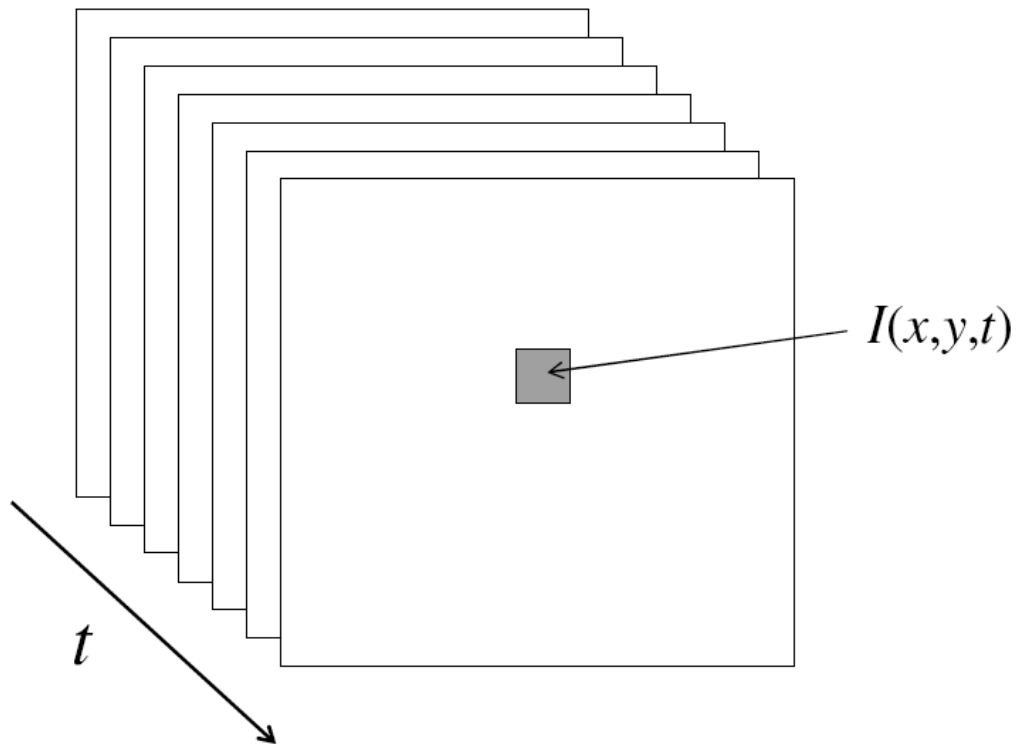
۳ و ۸ آذر ۱۳۹۹

جلسه نوزدهم و بیستم

Optical Flow and Motion

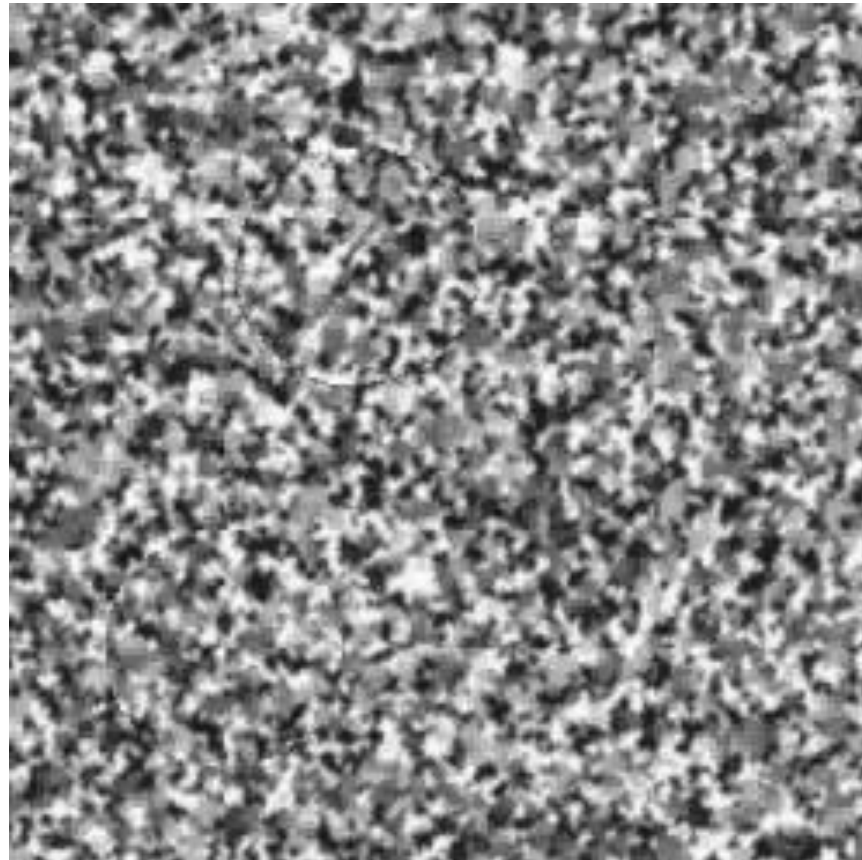
Video

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



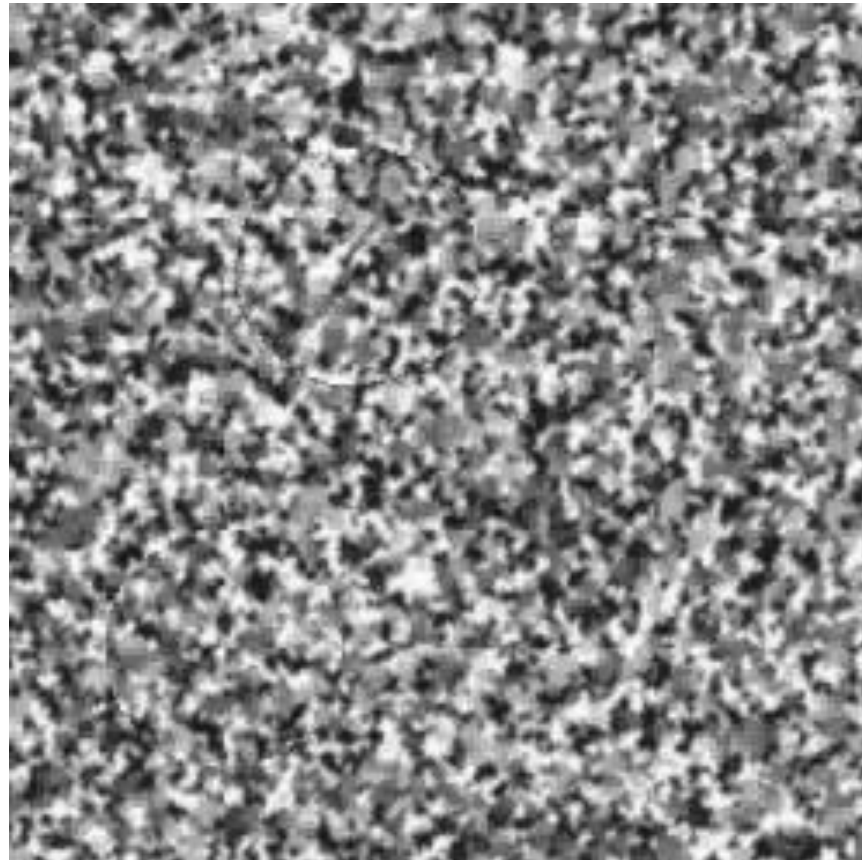
Motion and perceptual organization

- Sometimes, motion is the only cue



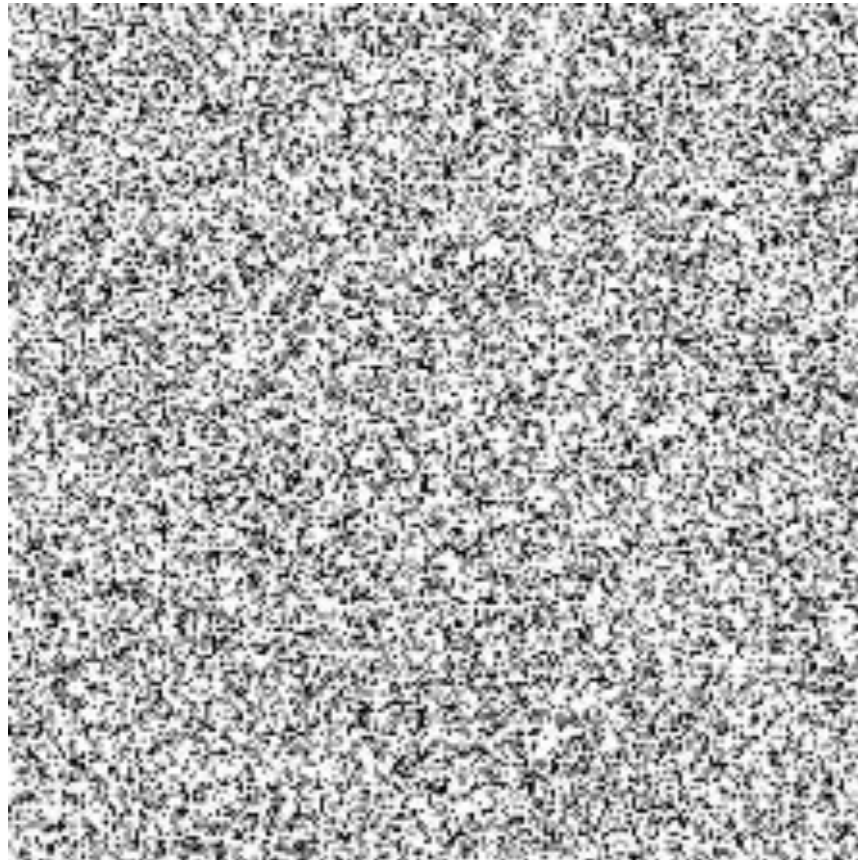
Motion and perceptual organization

- Sometimes, motion is the only cue



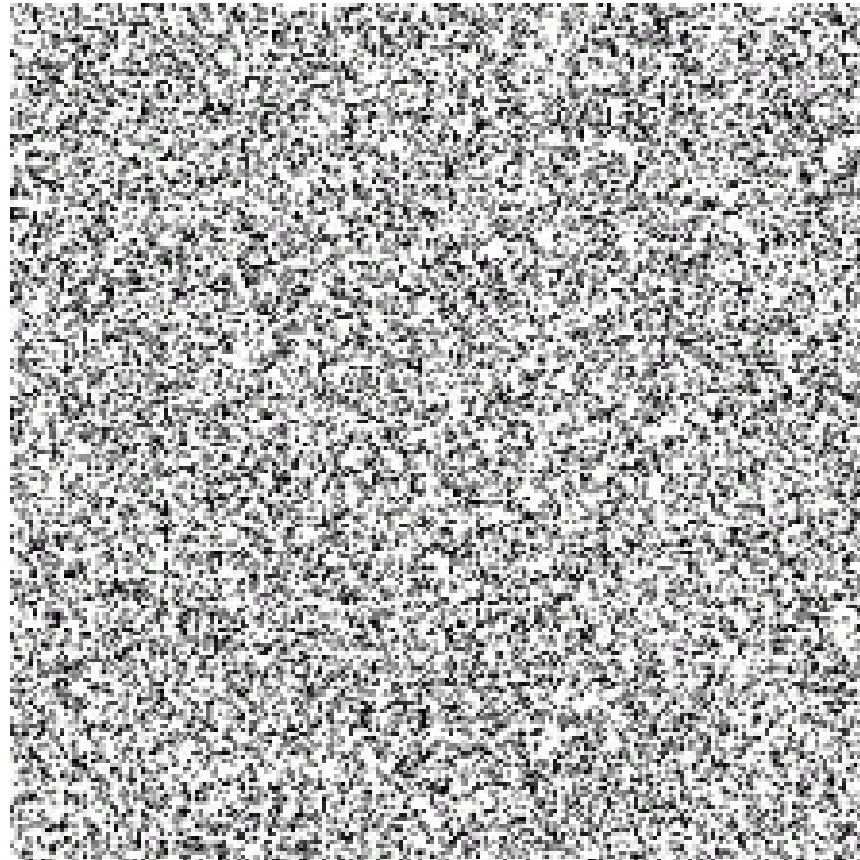
Motion and perceptual organization

- Sometimes, motion is the only cue



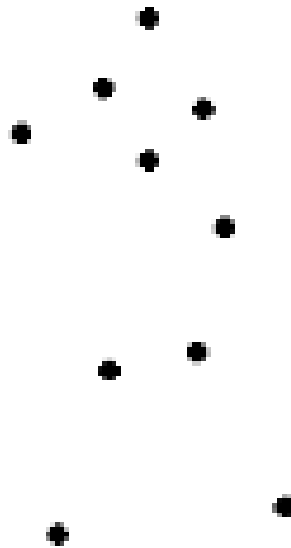
Motion and perceptual organization

- Sometimes, motion is the only cue



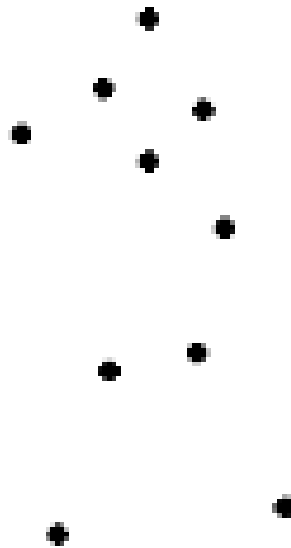
Motion and perceptual organization

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



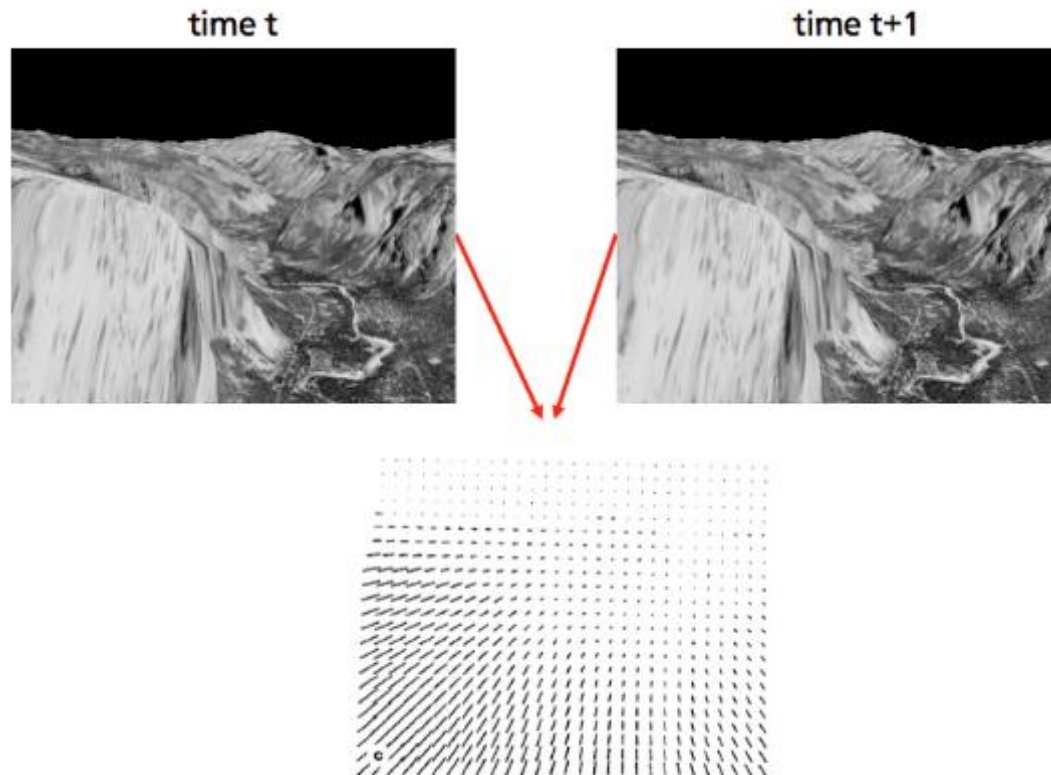
Motion and perceptual organization

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

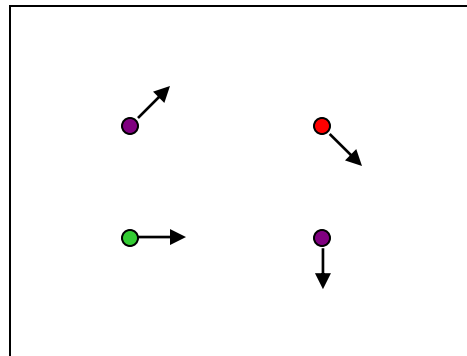


Dense Motion Estimation

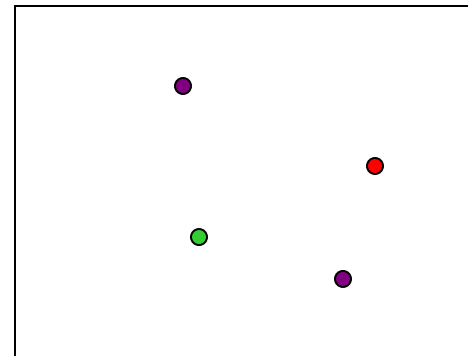
- Direct methods
 - Directly recover image motion at each pixel from spatio-temporal image brightness variations
 - Dense motion fields, but sensitive to appearance variations
 - Suitable for video and when image motion is small



Optical Flow



$I(x, y, t)$

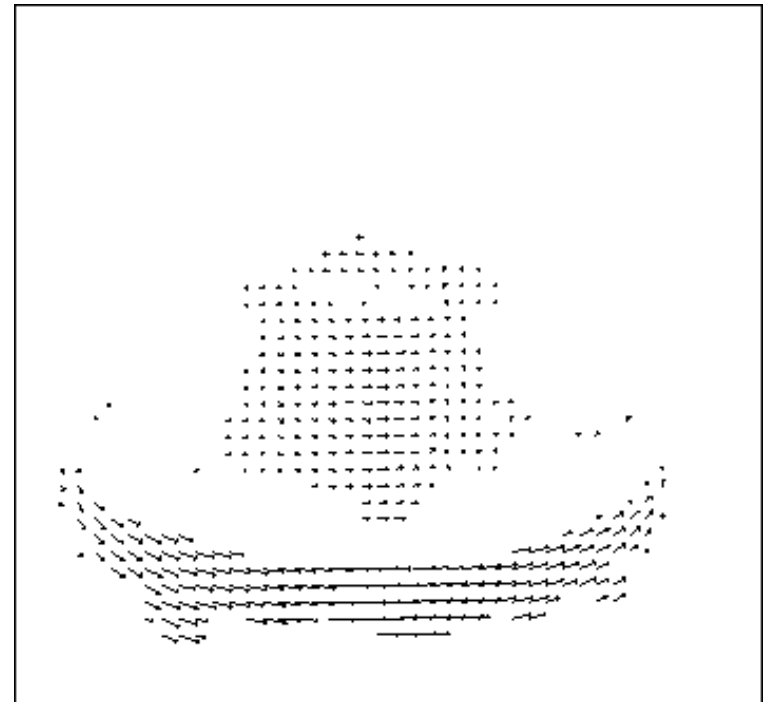
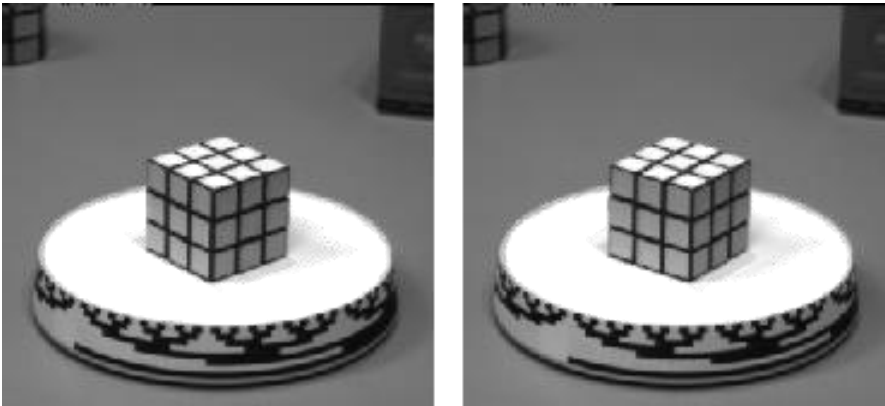


$I(x, y, t + 1)$

- Given two subsequent frames, estimate the motion vectors to move pixels from the first frame to their corresponding points in the second frame

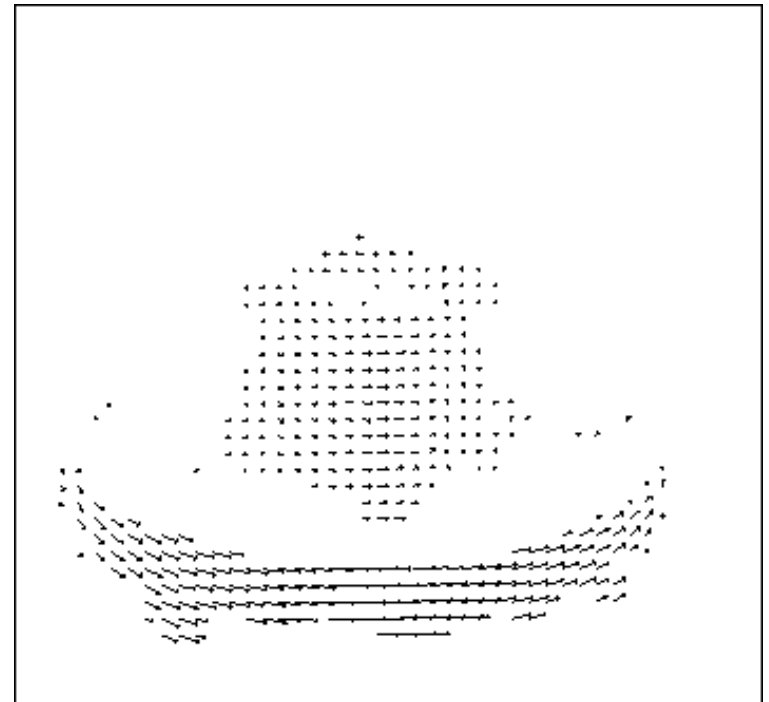
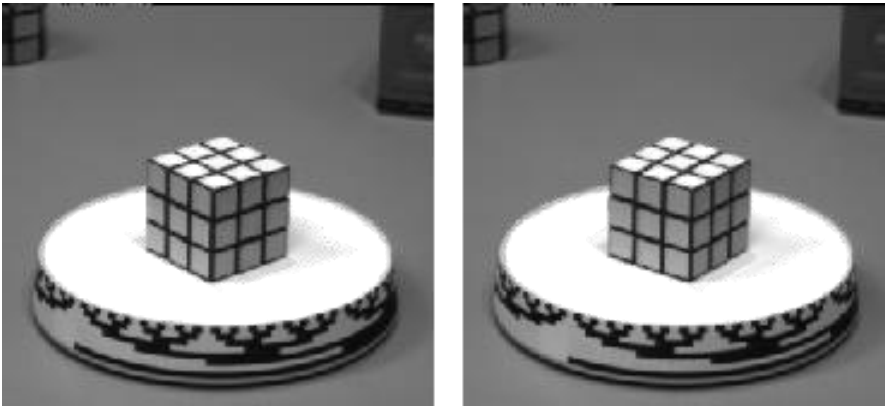
Motion Field

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



Optical Flow: Apparent Motion

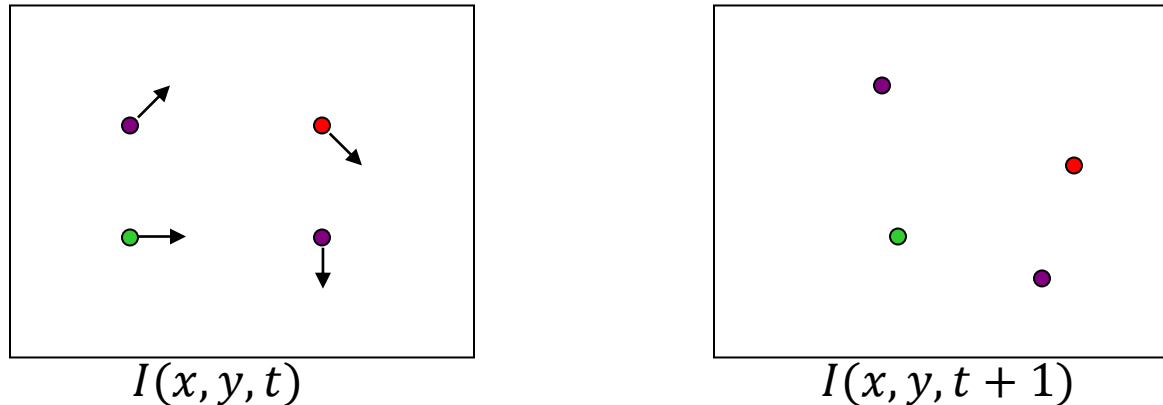
- *Optic flow* is the **apparent** motion of objects or surfaces



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

Problem definition: Optical Flow



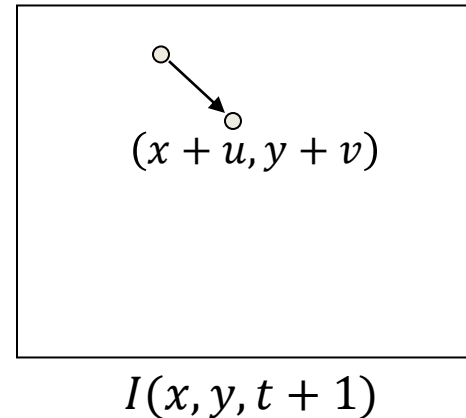
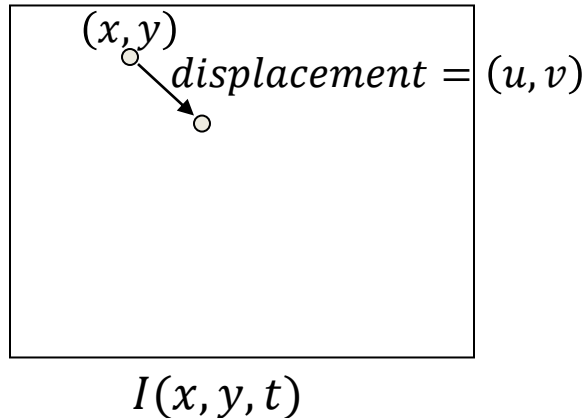
How to estimate pixel motion from image $I(x, y, t)$ to $I(x, y, t + 1)$?

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

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

Optical flow constraints (grayscale images)



- Let's look at these constraints more closely

- brightness constancy constraint (equation)

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

- small motion: (u and v are less than 1 pixel)

Taylor series expansion of I :

$$I(x + u, y + v, t + 1) = I(x, y, t) + I_x u + I_y v + I_t + (\text{higher order terms})$$

$$I(x + u, y + v, t + 1) - I(x, y, t) \approx I_x u + I_y v + I_t \approx 0$$

Optical Flow Equation

$$I_x u + I_y v + I_t \approx 0$$

$$\operatorname{argmin}_{u,v} E(u, v) = \operatorname{argmin}_{u,v} \|I_x u + I_y v + I_t\|^2$$

$$\frac{\partial E}{\partial u} = 2I_x(I_x u + I_y v + I_t) = 0 \Rightarrow I_x u + I_y v + I_t = 0$$

$$\frac{\partial E}{\partial v} = 2I_y(I_x u + I_y v + I_t) = 0 \Rightarrow I_x u + I_y v + I_t = 0$$

Brightness constancy constraint equation

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

- The brightness constancy constraint:

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

- How many equations and unknowns per pixel?

One equation, two unknowns

- The constraint can be written as:

$$\nabla I \cdot (u, v) + I_t = 0$$

Solving the ambiguity

- How to get more equations for a pixel?

- **Spatial coherence constraint:**

Assume the pixel's neighbors have the same (u, v)

- E.g., if we use a 5x5 window, that gives us 25 equations per pixel

$$\nabla I(p_i) \cdot (u, v) + I_t(p_i) = 0$$

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

Lucas-Kanade Algorithm

Least squares problem:

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_n) & I_y(p_n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_n) \end{bmatrix} \quad \longrightarrow \quad A_{n \times 2} d_{2 \times 1} = b_{n \times 1}$$
$$(A^t A) d = A^t b$$


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

The summations are over all pixels in the $n \times n$ window.

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.

Conditions for solvability

Optimal (u, v) satisfies Lucas-Kanade equation


$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} \quad (A^t A)d = A^t b$$

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 λ_1 and λ_2 of $A^t A$ should not be too small
- $A^t A$ should be well-conditioned
 - λ_1/λ_2 should not be too large (λ_1 = larger eigenvalue)

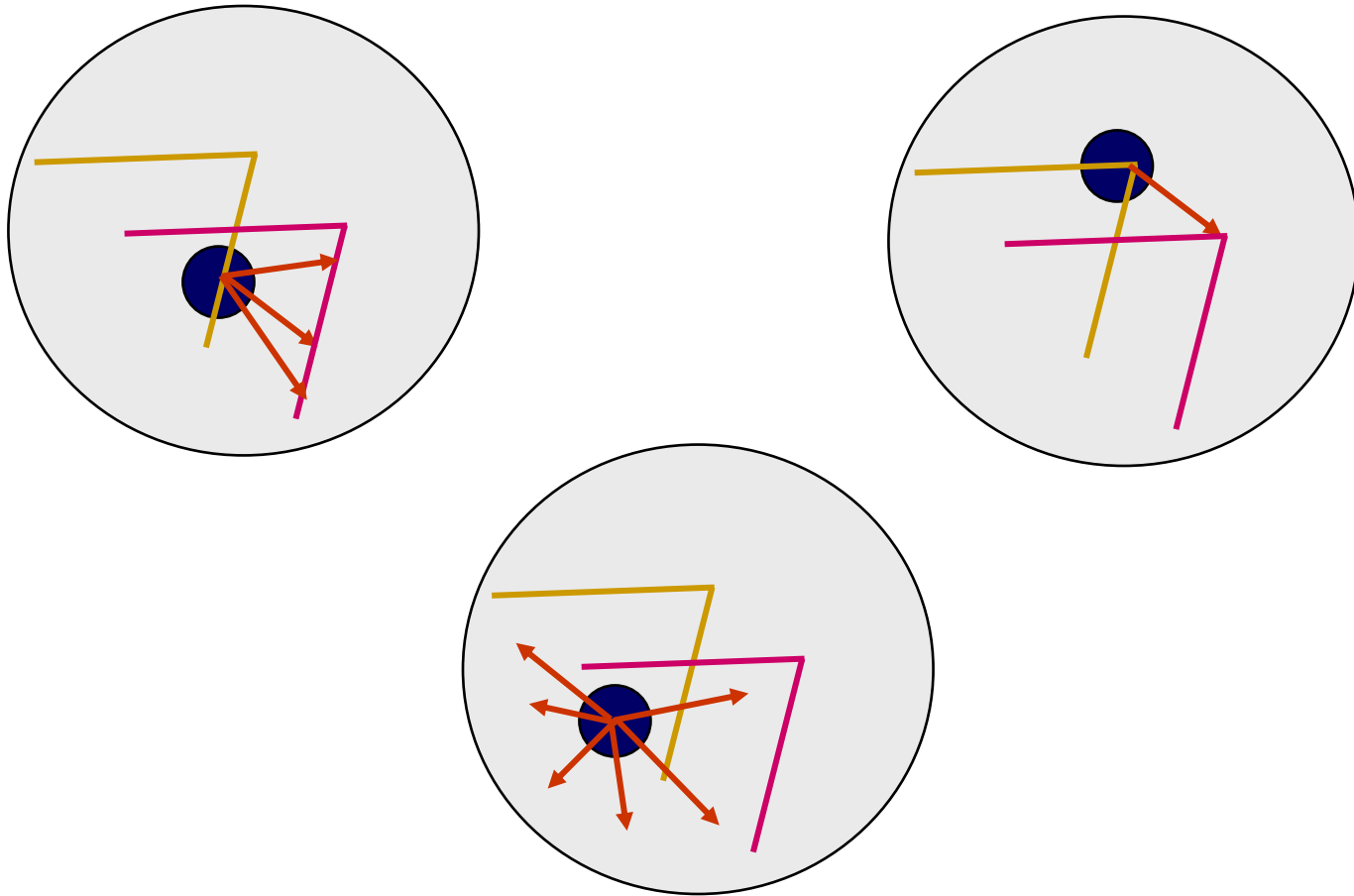
Does this remind you of anything? Criteria for Harris corner detector

$M = A^t A$ is the *second moment matrix* !
(Harris corner detector...)

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

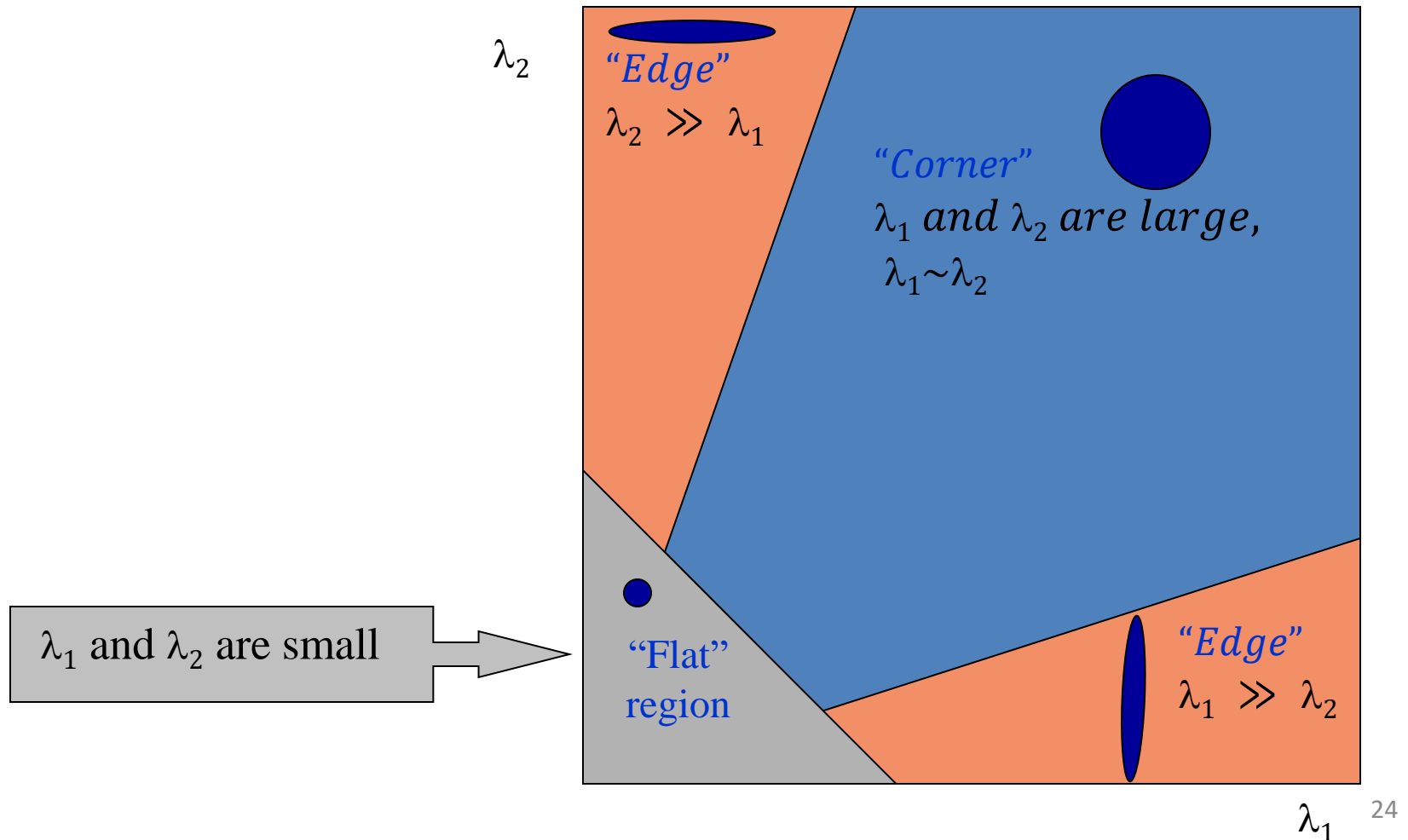
- Eigenvectors and eigenvalues of $A^t A$ relate to edge direction and magnitude
 - The eigenvector associated with the larger eigenvalue points in the direction of fastest intensity change.
 - The other eigenvector is orthogonal to it.

Local Patch Analysis

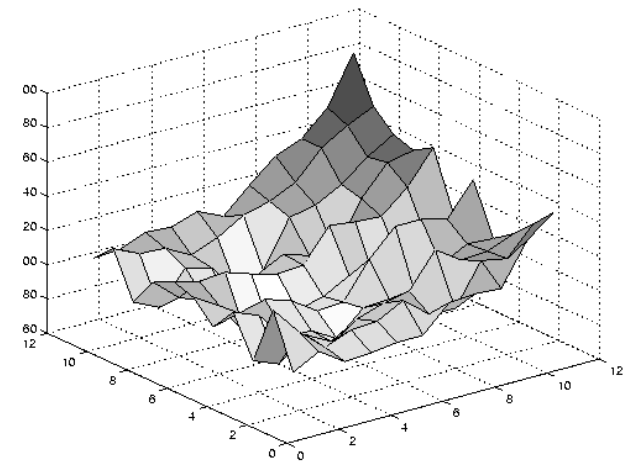
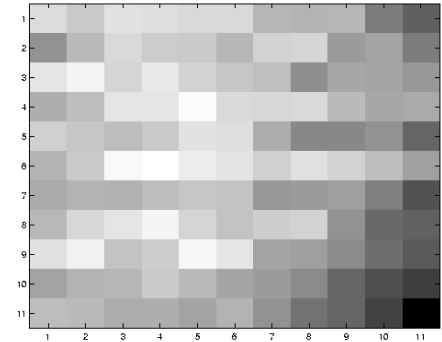


Recall: second moment matrix

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



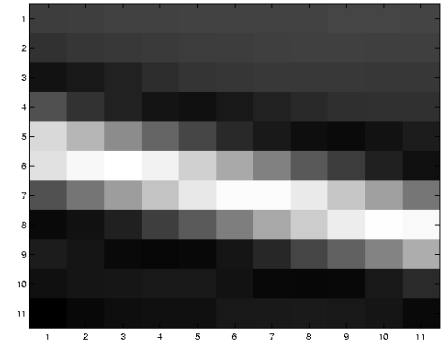
Low texture regions



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

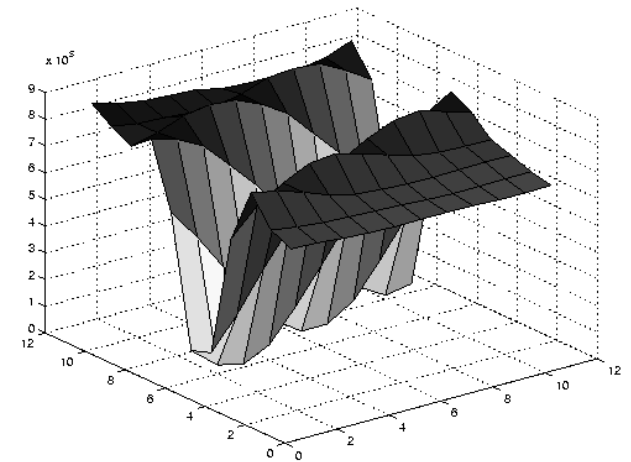
- gradients have small magnitude
- small λ_1 , small λ_2

Edge

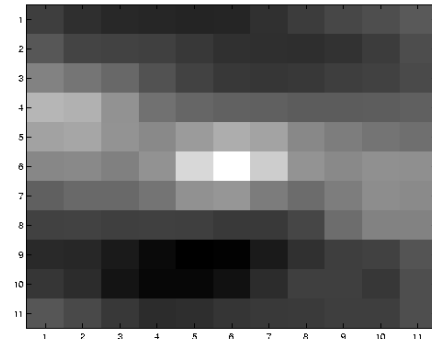


$$\sum \nabla I (\nabla I)^T$$

- large gradients, all the same
- large λ_1 , small λ_2

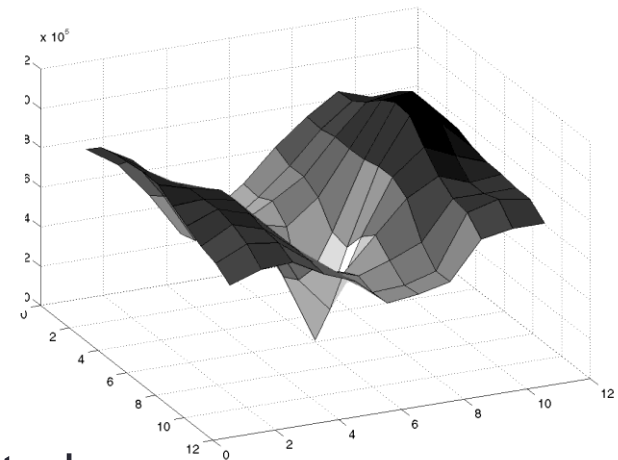


High textured region



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

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



The brightness constancy constraint

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

- What does this constraint mean?

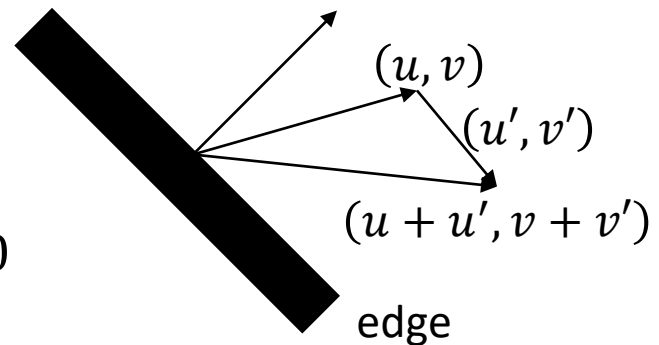
$$\nabla I \cdot (u, v) + I_t = 0$$

- The component of the flow perpendicular to the gradient (i.e., parallel to the edge) is unknown

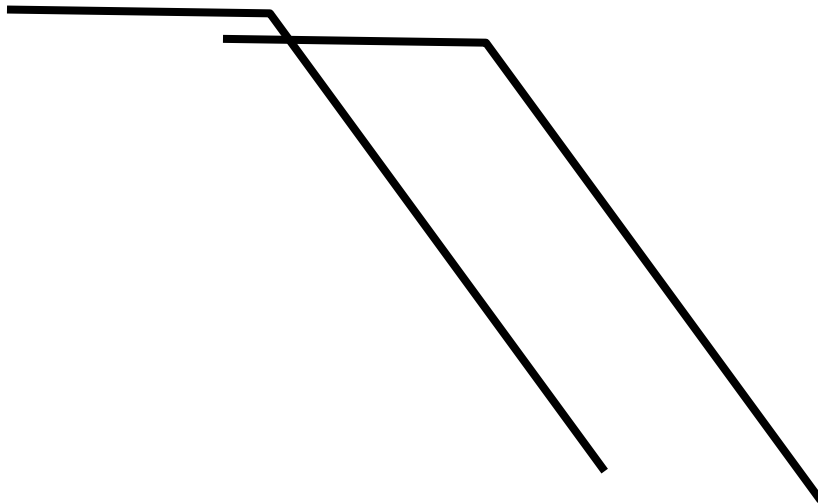
$$\nabla I \cdot (u, v) + I_t = 0$$

- The component of the flow perpendicular to the gradient (i.e., parallel to the edge) is unknown

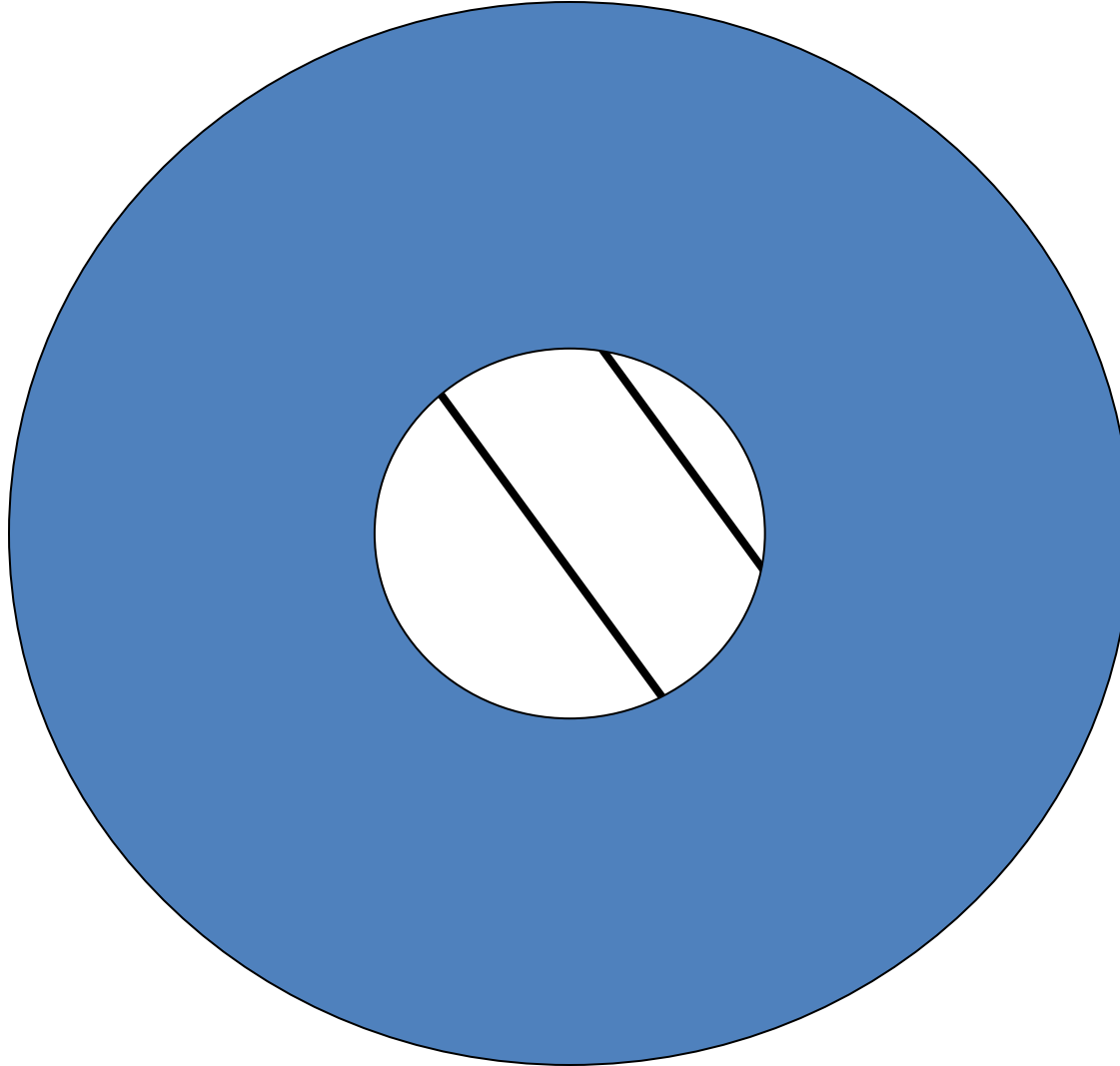
If (u, v) satisfies the equation,
so does $(u + u', v + v')$ if $\nabla I \cdot (u', v') = 0$



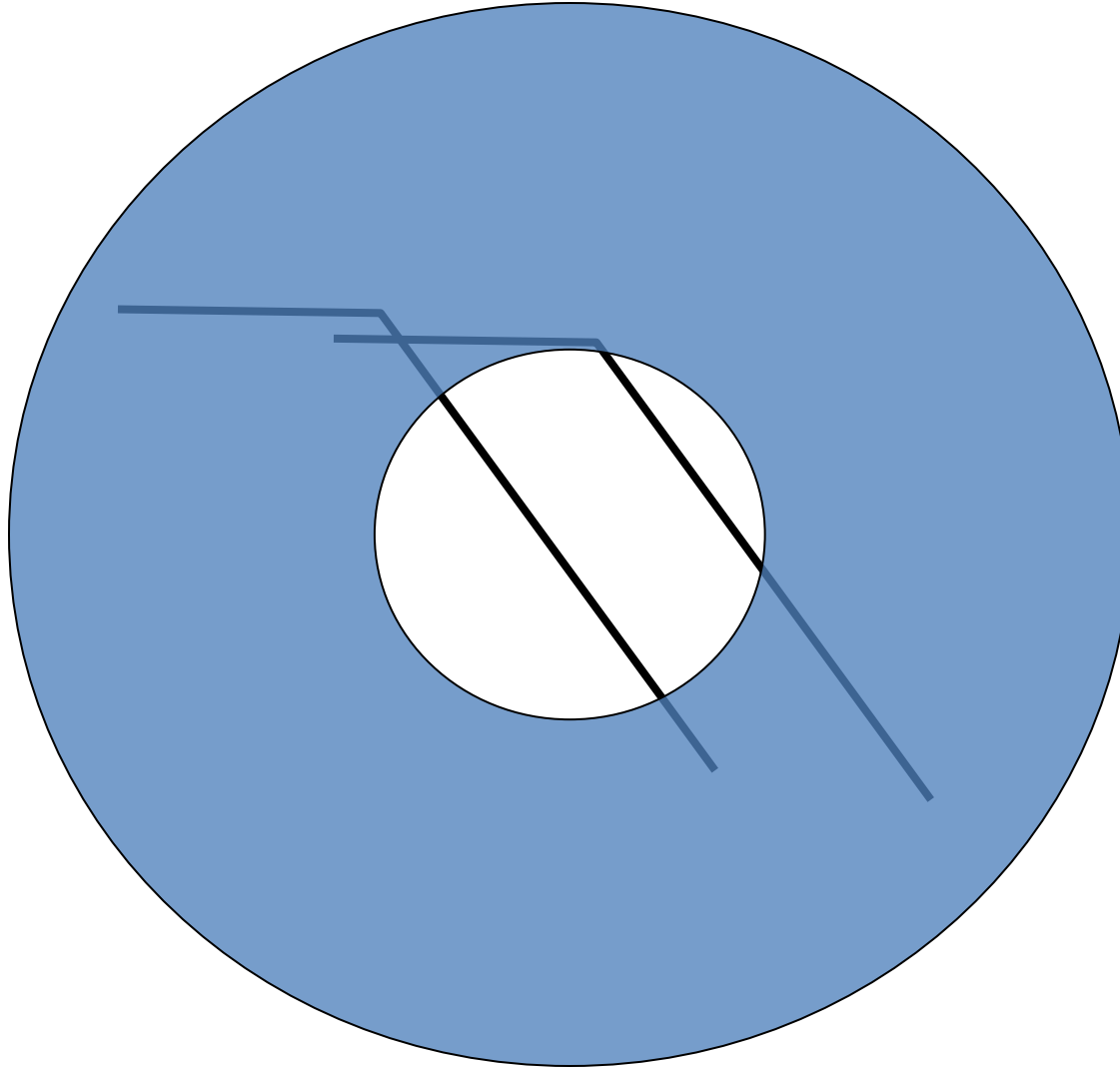
Aperture problem



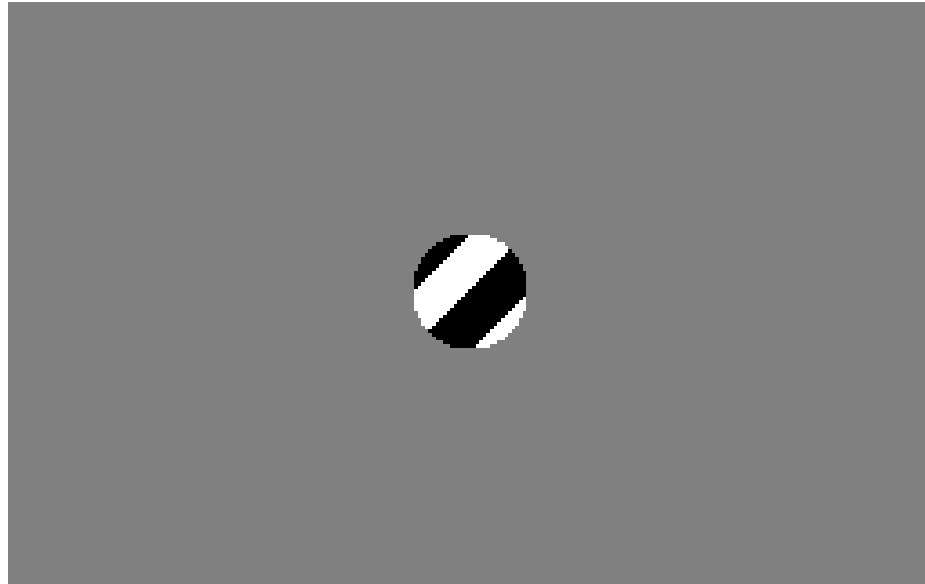
Aperture problem



Aperture problem

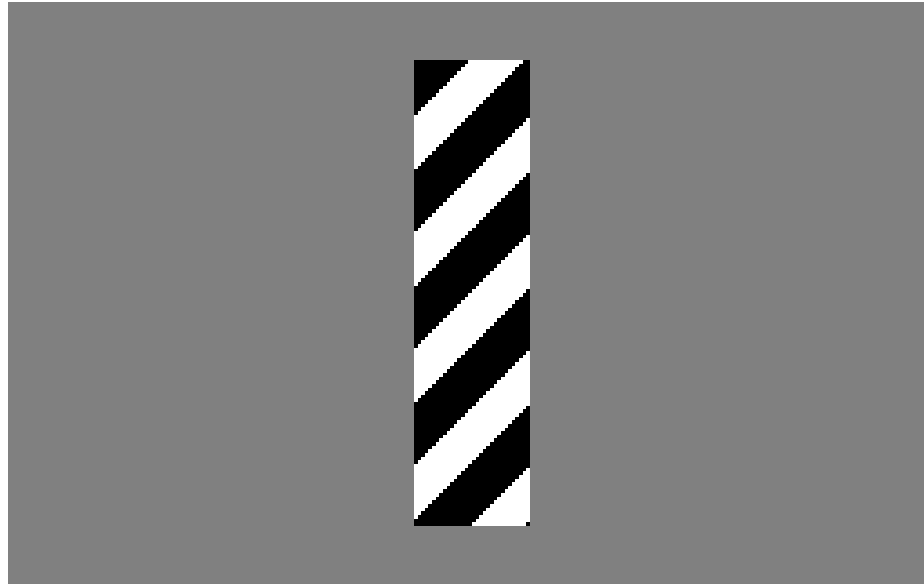


The barber pole illusion



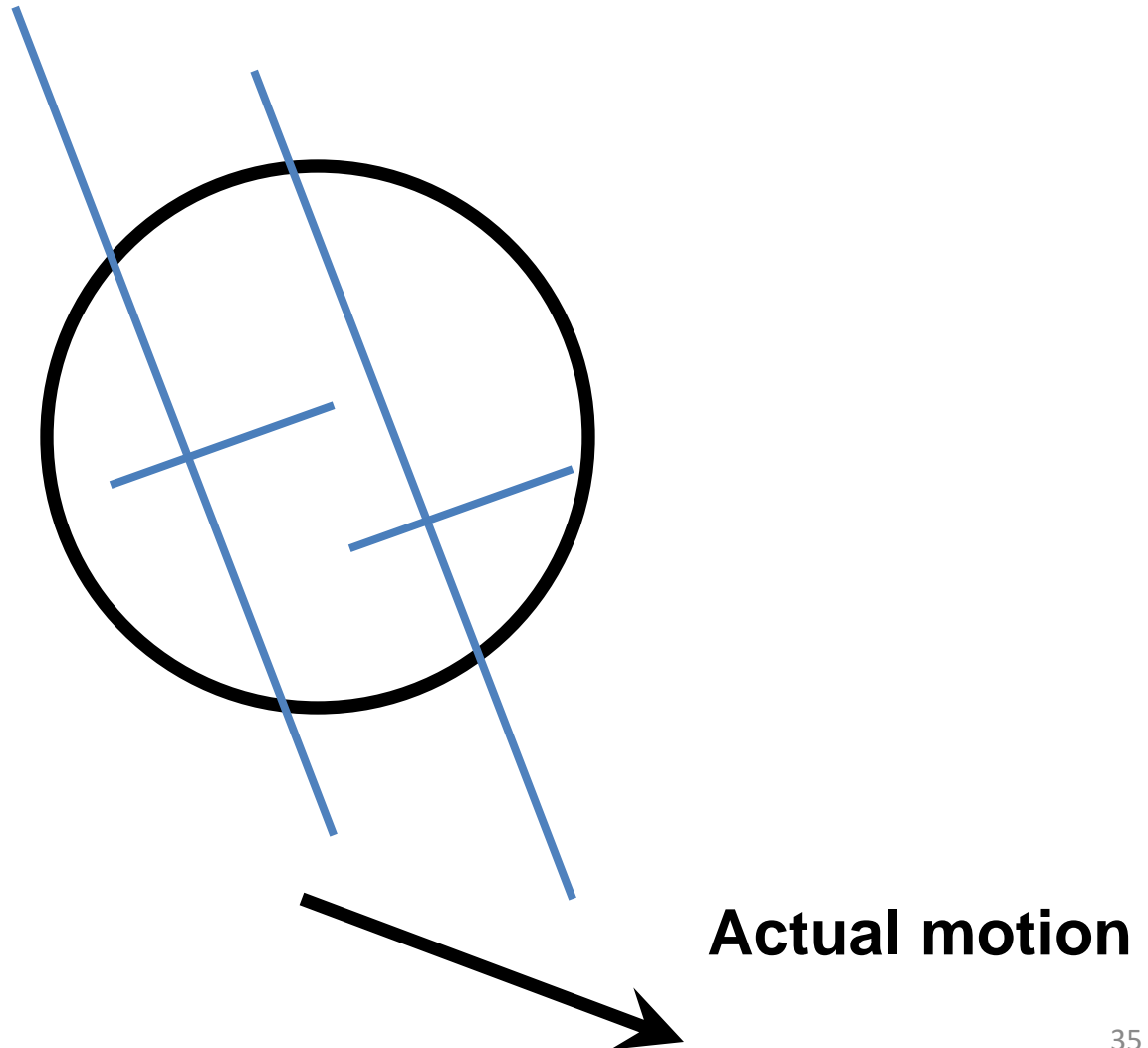
http://en.wikipedia.org/wiki/Barberpole_illusion

The barber pole illusion

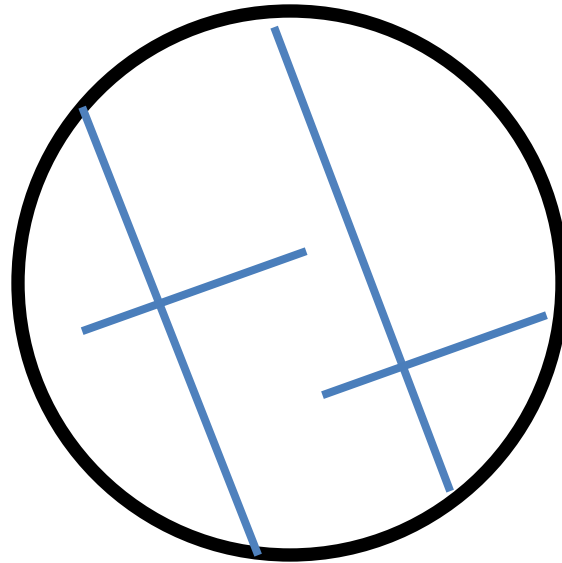


http://en.wikipedia.org/wiki/Barberpole_illusion

The aperture problem resolved



The aperture problem resolved



Perceived motion

Not tangent: Iterative Refinement

Iterative Lukas-Kanade Algorithm

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

Optical Flow: Iterative Estimation

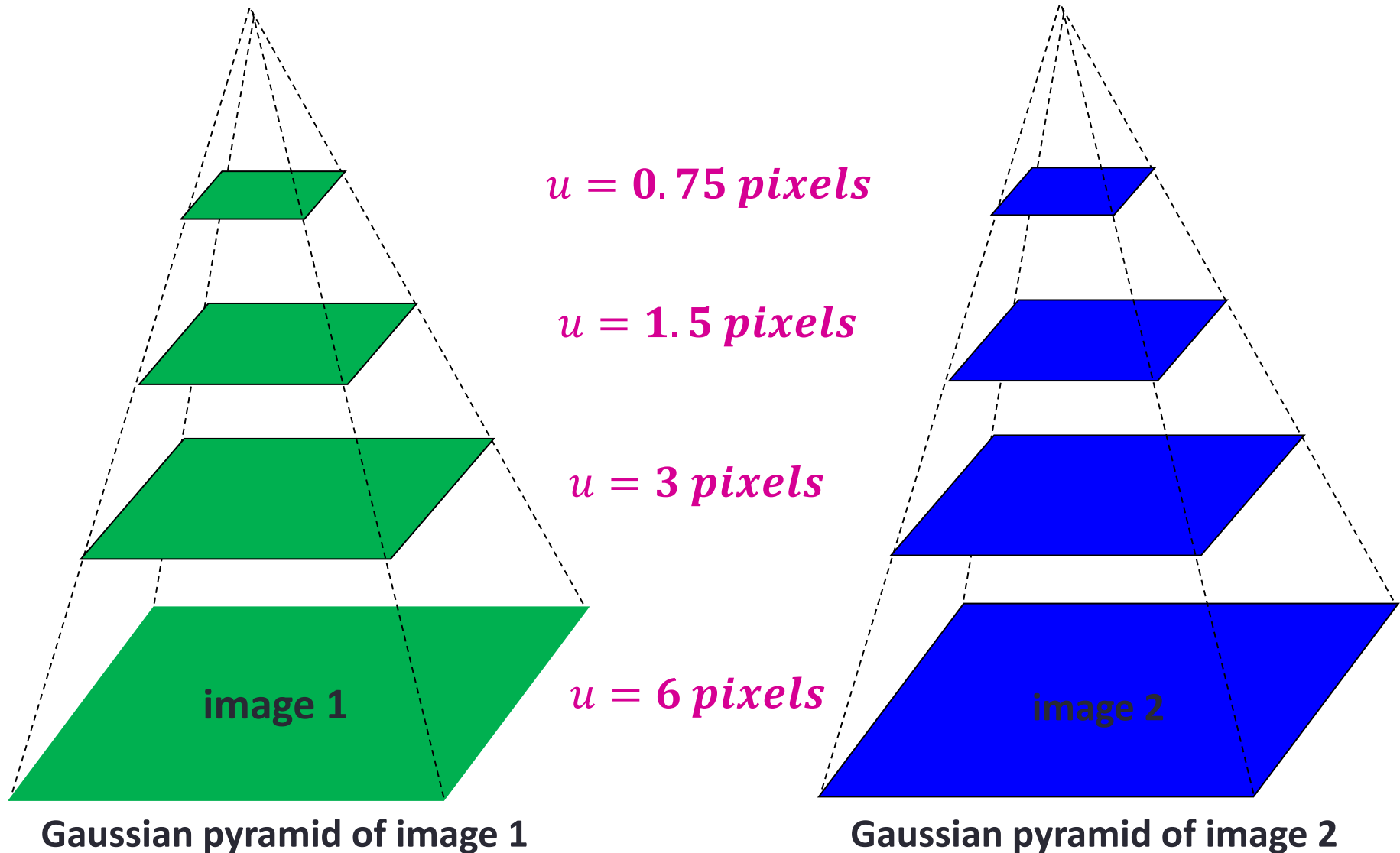
- Some Implementation Issues:
 - Warping is not easy (ensure that errors in warping are smaller than the estimate refinement) – but it is in MATLAB (Python)!
 - Often useful to low-pass filter the images before motion estimation (for better derivative estimation, and linear approximations to image intensity)

Revisiting the small motion assumption

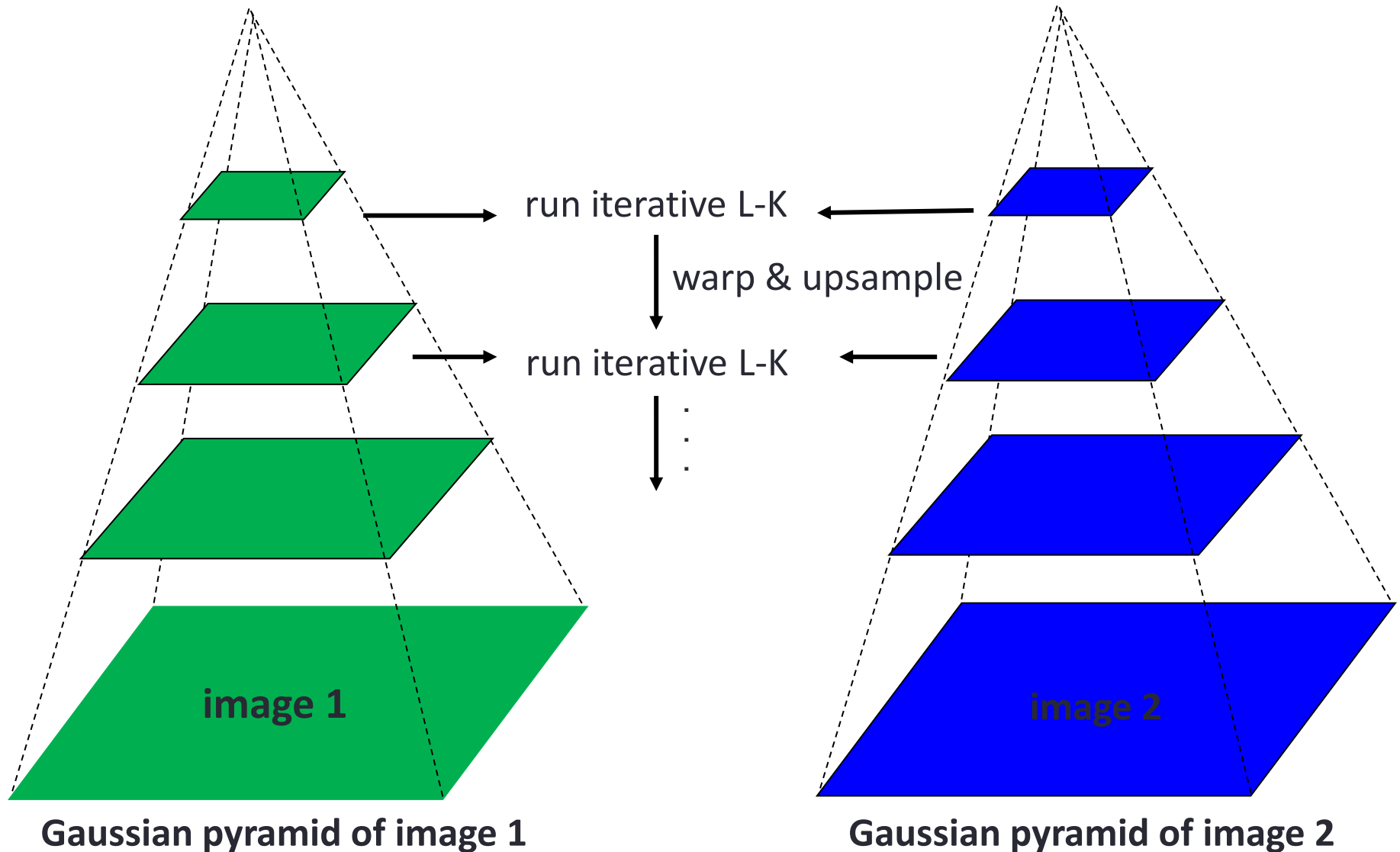


- Is this motion small enough?
 - Probably not—it's much larger than one pixel
 - How might we solve this problem?

Coarse-to-fine optical flow estimation



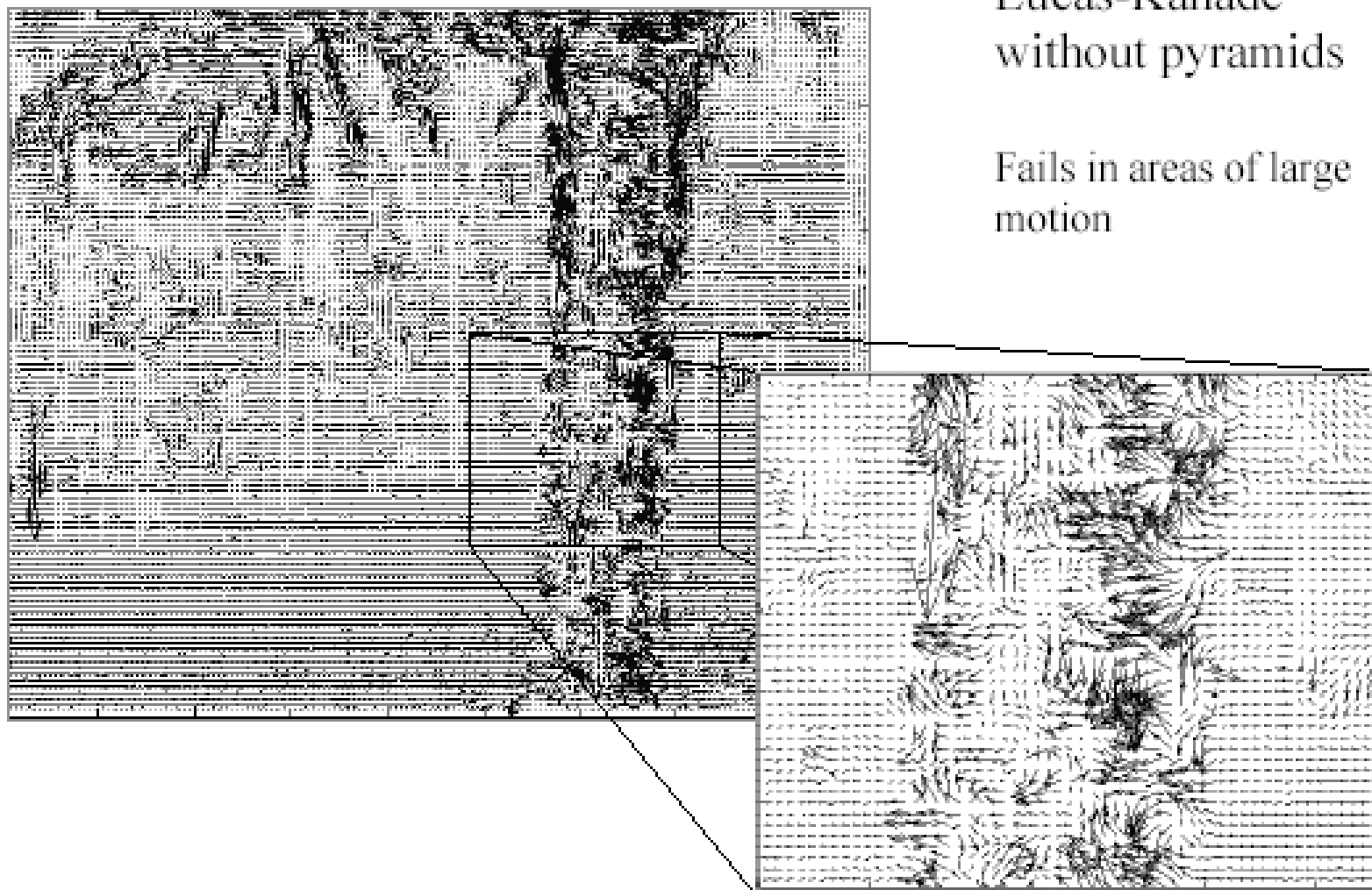
Coarse-to-fine optical flow estimation



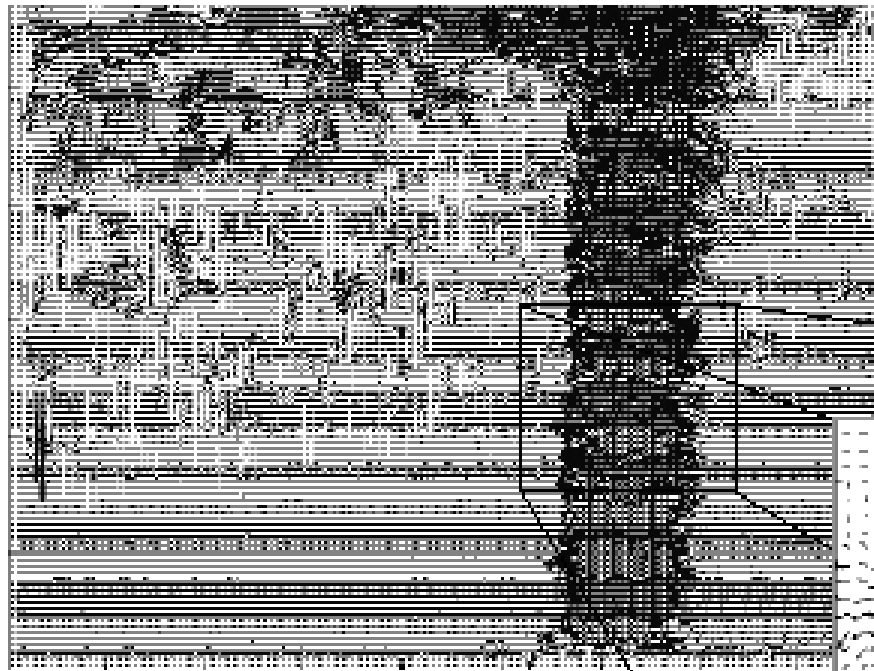
Optical Flow Results

Lucas-Kanade
without pyramids

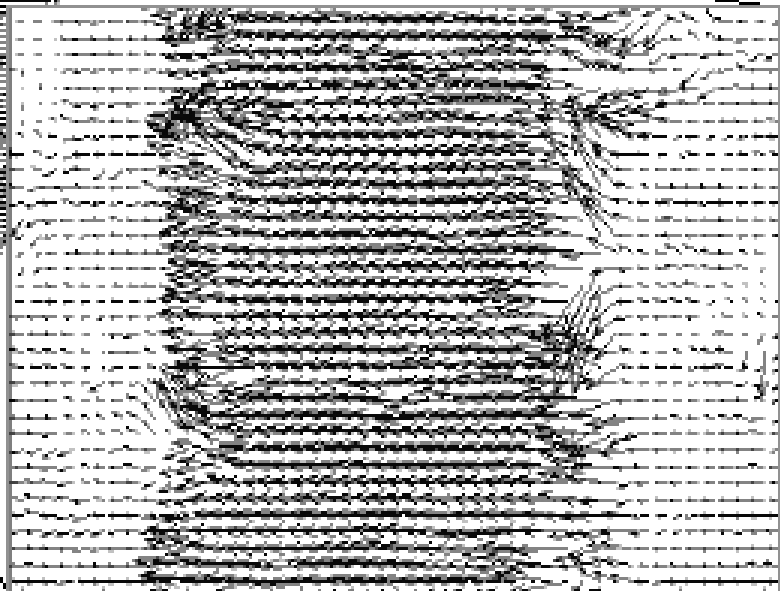
Fails in areas of large
motion



Optical Flow Results



Lucas-Kanade with Pyramids



Errors in Lucas-Kanade

- A point does not move like its neighbors
 - Possible Fix: Region-based matching
 - Motion segmentation
- Brightness constancy does not hold
 - Possible Fix: Gradient constancy
 - Do exhaustive neighborhood search with normalized correlation - tracking features – maybe SIFT – more later....
- The motion is large (larger than a pixel)
 - Possible Fix: Keypoint matching
 - Not-linear: Iterative refinement
 - Local minima: coarse-to-fine estimation

Middlebury

ICCV 2007 & IJCV 2011

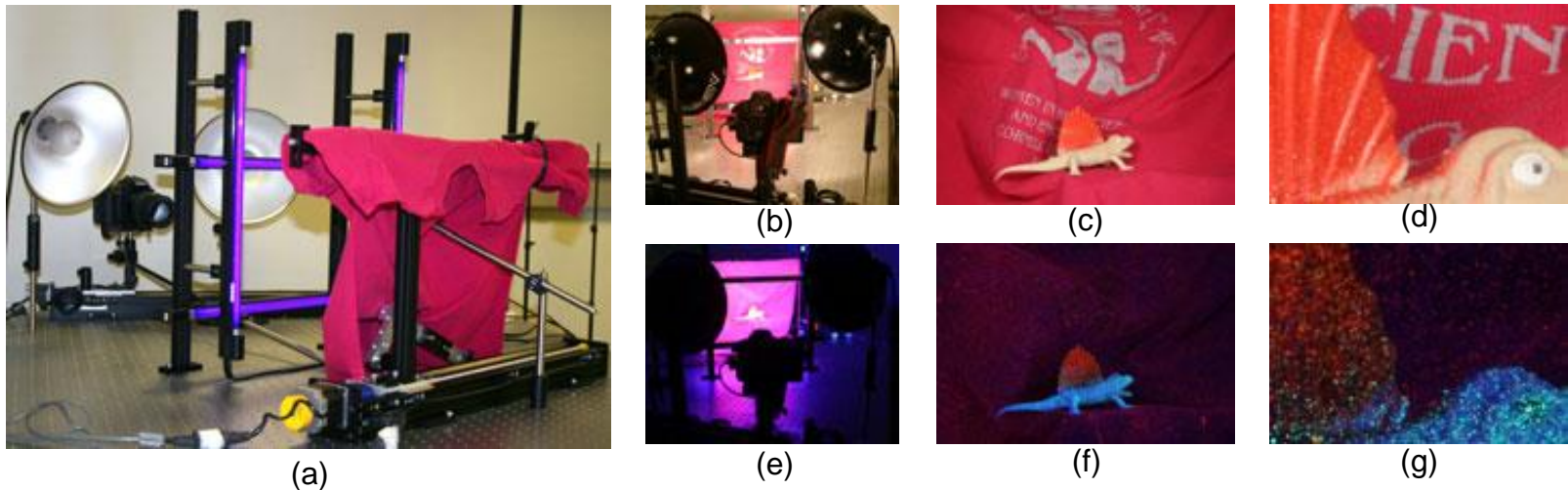


Fig. 1 (a) The setup for obtaining ground-truth flow using hidden fluorescent texture includes computer-controlled lighting to switch between the UV and visible lights. It also contains motion stages for both the camera and the scene. (b–d) The setup under the visible illumination. (e–g) The setup under the UV illumination. (c and f) Show the high-resolution images taken by the digital camera. (d and g) Show a zoomed portion of (c) and (f). The high-frequency fluorescent texture in the images taken under UV light (g) allows accurate tracking, but is largely invisible in the low-resolution test images

Middlebury

ICCV 2007 & IJCV 2011



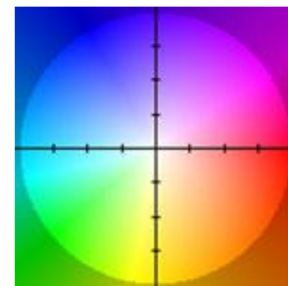
Army frame 0



Army frame 1



Army GT flow



flow color coding



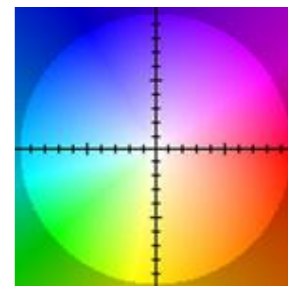
Mequon frame 0



Mequon frame 1



Mequon GT flow



flow color coding

KITTI

The KITTI Vision Benchmark Suite

A project of Karlsruhe Institute of Technology
and Toyota Technological Institute at Chicago



[home](#) [setup](#) [stereo](#) [flow](#) [scene](#)[flow](#) [depth](#) [odometry](#) [object](#) [tracking](#) [road](#) [semantics](#) [raw data](#) [submit results](#)

Andreas Geiger (MPI Tübingen) | Philip Lenz (KIT) | Christoph Stiller (KIT) | Raquel Urtasun (University of Toronto)



Welcome to the KITTI Vision Benchmark Suite!

We take advantage of our [autonomous driving platform Annleway](#) to develop novel challenging real-world computer vision benchmarks. Our tasks of interest are: stereo, optical flow, visual odometry, 3D object detection and 3D tracking. For this purpose, we equipped a standard station wagon with two high-resolution color and grayscale video cameras. Accurate ground truth is provided by a Velodyne laser scanner and a GPS localization system. Our datasets are captured by driving around the mid-size city of [Karlsruhe](#), in rural areas and on highways. Up to 15 cars and 30 pedestrians are visible per image. Besides providing all data in raw format, we extract benchmarks for each task. For each of our benchmarks, we also provide an evaluation metric and this evaluation website. Preliminary experiments show that methods ranking high on established benchmarks such as [Middlebury](#) perform below average when being moved outside the laboratory to the real world. Our goal is to reduce this bias and complement existing benchmarks by providing real-world benchmarks with novel difficulties to the community.



To get started, grab a cup of your favorite beverage and watch our video trailer (5 minutes):

KITTI

The KITTI Vision Benchmark Suite

A project of Karlsruhe Institute of Technology
and Toyota Technological Institute at Chicago



home setup stereo **flow** sceneflow depth odometry object tracking road semantics raw data submit results

Andreas Geiger (MPI Tübingen) | Philip Lenz (KIT) | Christoph Stiller (KIT) | Raquel Urtasun (University of Toronto)

Optical Flow Evaluation

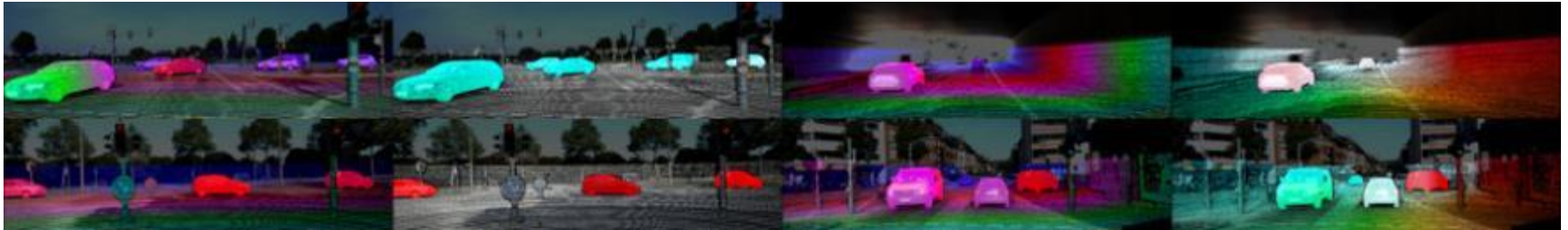


This is our original optical flow evaluation referred to as "**KITTI Flow**" or "**KITTI Flow 2012**" and published in [Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite \(CVPR 2012\)](#). It consists of 194 training and 195 test scenes of a static environment captured by a moving camera.



This is our new optical flow evaluation referred to as "**KITTI Flow 2015**" which has been derived from the scene flow dataset published in [Object Scene Flow for Autonomous Vehicles \(CVPR 2015\)](#). It consists of 200 training and 200 test scenes with moving camera and moving objects.

KITTI



	Method	Setting	Code	Fl-bg	Fl-fg	Fl-all	Density	Runtime	Environment	Compare
1	RigidMask+ISF			2.63 %	7.85 %	3.50 %	100.00 %	3.3 s	GPU @ 2.5 Ghz (Python)	<input type="checkbox"/>
2	Dahua_SF			2.86 %	8.44 %	3.79 %	100.00 %	0.5 s	1 core @ 2.5 Ghz (Python)	<input type="checkbox"/>
3	MaskRCNN+ISF			2.86 %	9.05 %	3.89 %	100.00 %	3.3 s	GPU @ 2.5 Ghz (Python)	<input type="checkbox"/>
4	RME			3.39 %	8.79 %	4.29 %	100.00 %	2 s	GPU @ 2.5 Ghz (Python + C/C++)	<input type="checkbox"/>
5	UberATG-DRISE			3.59 %	10.40 %	4.73 %	100.00 %	0.75 s	CPU+GPU @ 2.5 Ghz (Python)	<input type="checkbox"/>

ERROR: Wrong syntax in BIBTEX file.

W. Ma, S. Wang, R. Hu, Y. Xiong and R. Urtasun: [Deep Rigid Instance Scene Flow](#). CVPR 2019.

190	TVL1_RVC		code	51.80 %	53.17 %	52.03 %	100.00 %	1 s	2 cores @ 2.5 Ghz (C/C++)	<input type="checkbox"/>
191	PolyExpand			52.00 %	58.56 %	53.09 %	100.00 %	1 s	1 core @ 2.5 Ghz (C/C++)	<input type="checkbox"/>
G. Farneback: Two-Frame Motion Estimation Based on Polynomial Expansion . SCIA 2003.										
192	H+S_RVC		code	58.29 %	63.66 %	59.18 %	100.00 %	4 s	2 cores @ 2.5 Ghz (C/C++)	<input type="checkbox"/>
193	H+S_ROB		code	68.22 %	76.49 %	69.60 %	100.00 %	8 s	4 cores @ 2.5 Ghz (C/C++)	<input type="checkbox"/>
E. Meinhardt-Llopis, J. Sánchez Pérez and D. Kondermann: Horn-Schunck Optical Flow with a Multi-Scale Strategy . Image Processing On Line 2013.										
194	Pyramid-LK		code	71.84 %	76.82 %	72.67 %	100.00 %	1.5 min	1 core @ 2.5 Ghz (Matlab)	<input type="checkbox"/>
J. Bouguet: Pyramidal implementation of the Lucas Kanade feature tracker . Intel 2000.										
195	MEDIAN			87.37 %	92.80 %	88.27 %	99.86 %	0.01 s	1 core @ 2.5 Ghz (C/C++)	<input type="checkbox"/>
196	AVERAGE			88.47 %	92.08 %	89.07 %	99.86 %	0.01 s	1 core @ 2.5 Ghz (C/C++)	<input type="checkbox"/>

Table as LaTeX | [Only published Methods](#)

State-of-the-art optical flow, 2009

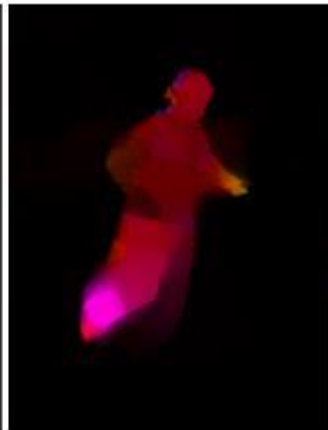
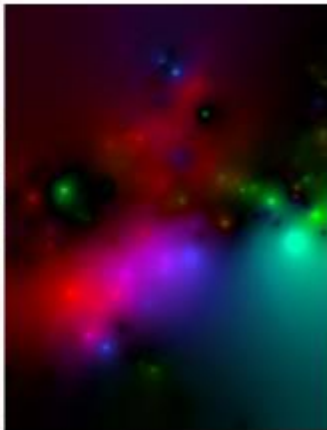
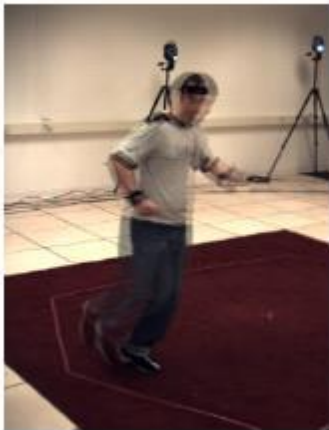
Start with something similar to Lucas-Kanade

+ gradient constancy

+ energy minimization with smoothing term

+ region matching

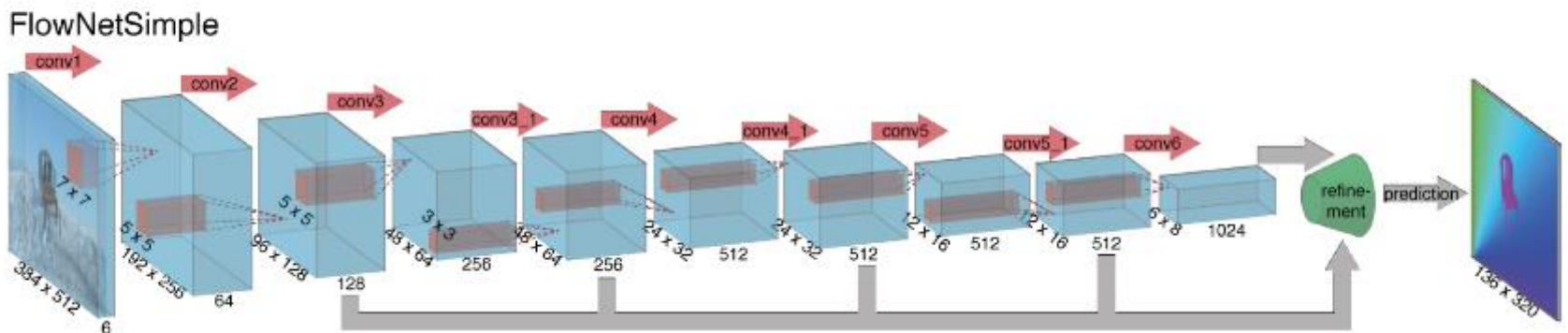
+ keypoint matching (long-range)



Region-based +Pixel-based +Keypoint-based

State-of-the-art optical flow, 2015

- Deep convolutional network
- Accepts a pair of input frames
- Upsamples the estimated flow back to input resolution
- Near the state-of-the-art in terms of end-point-error



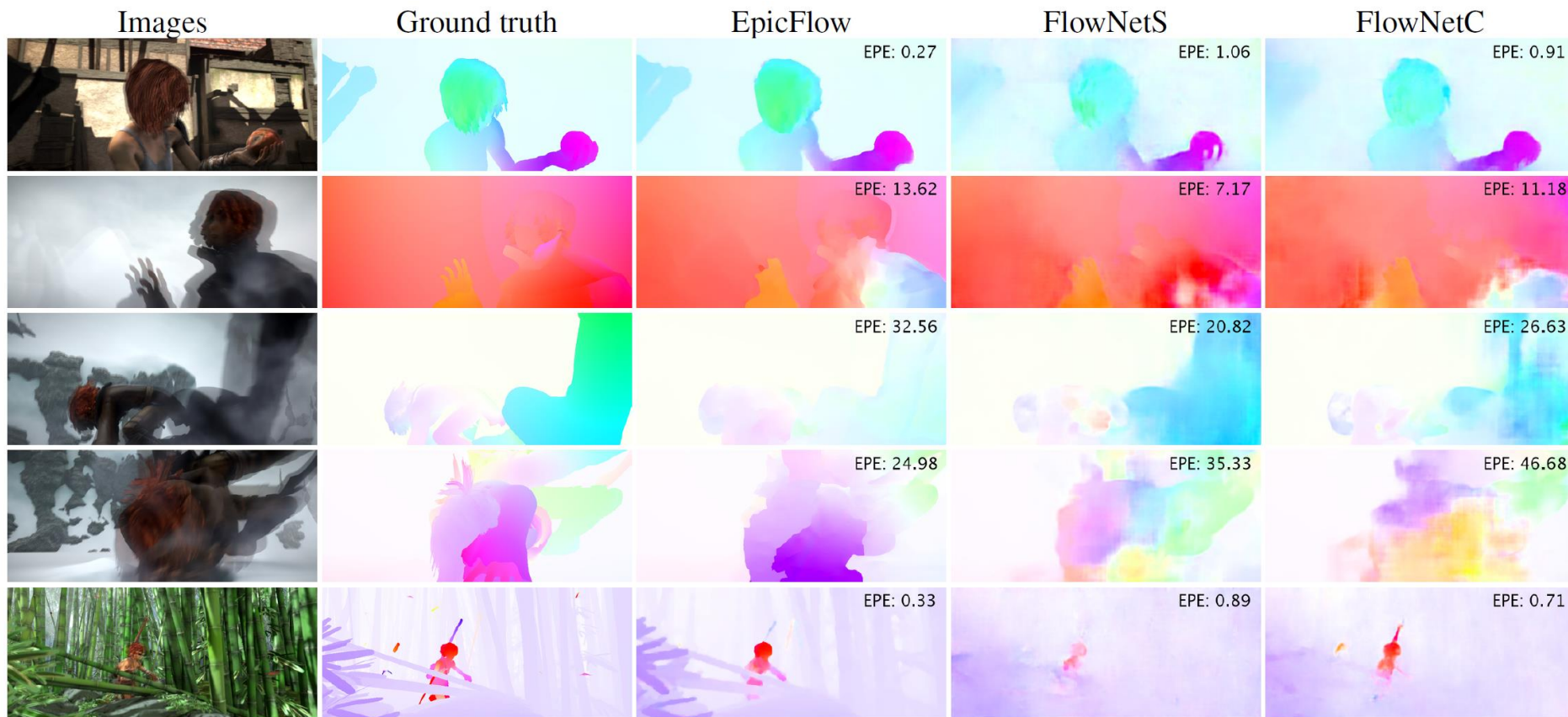
Deep optical flow, 2015

Synthetic Training data



Deep optical flow, 2015

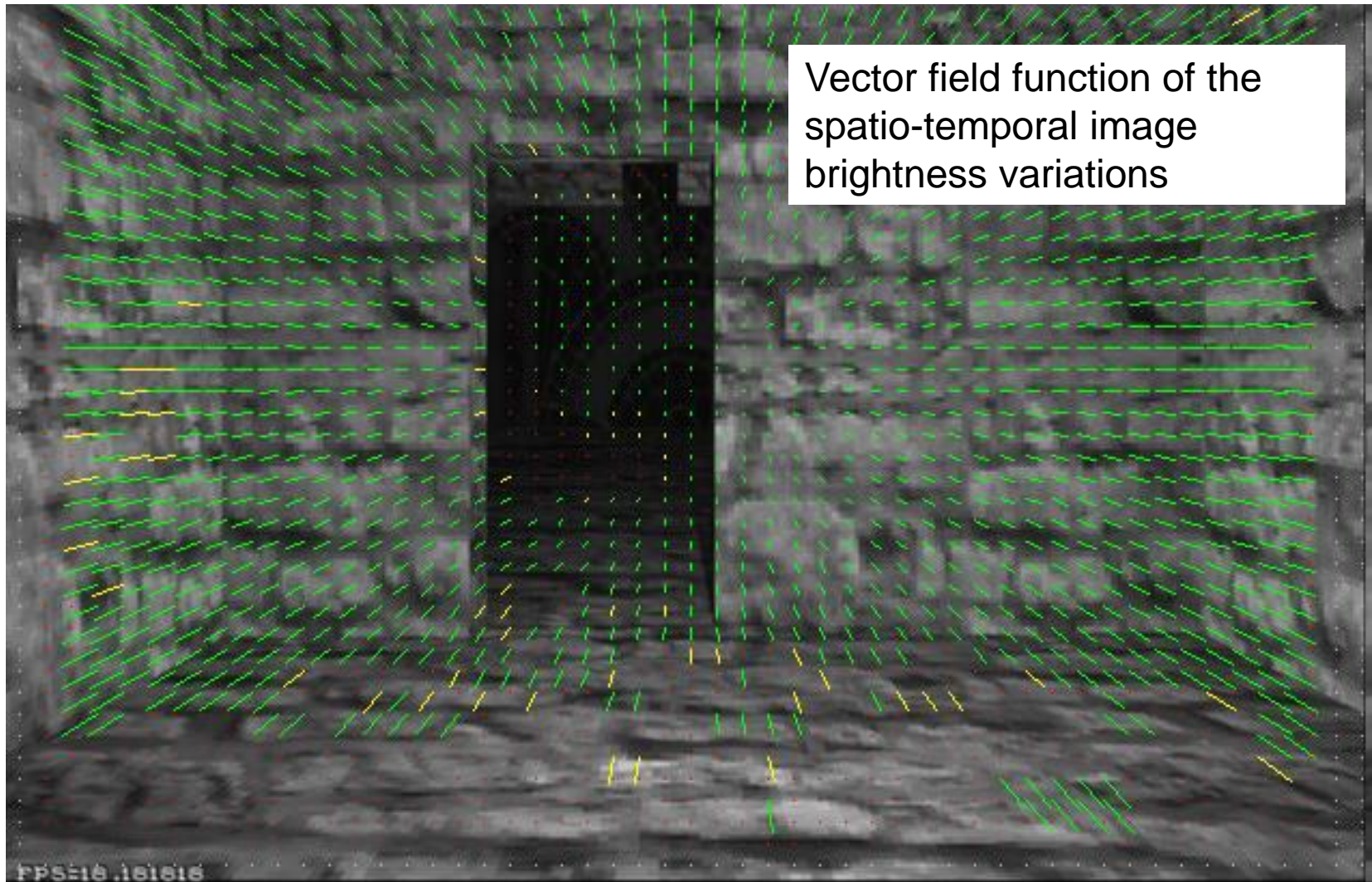
Results on Sintel



Uses of motion in computer vision

- 3D shape reconstruction
- Object segmentation
- Learning and tracking of dynamical models
- Event and activity recognition

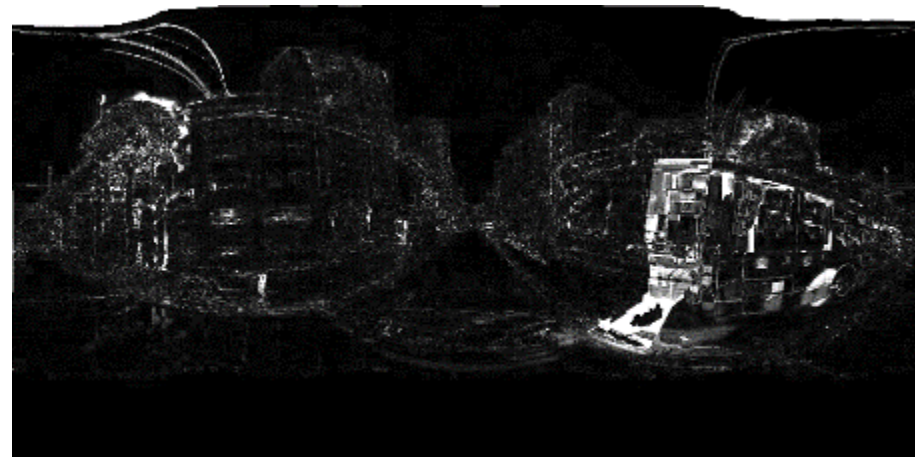
Optical Flow



Optical Flow Result

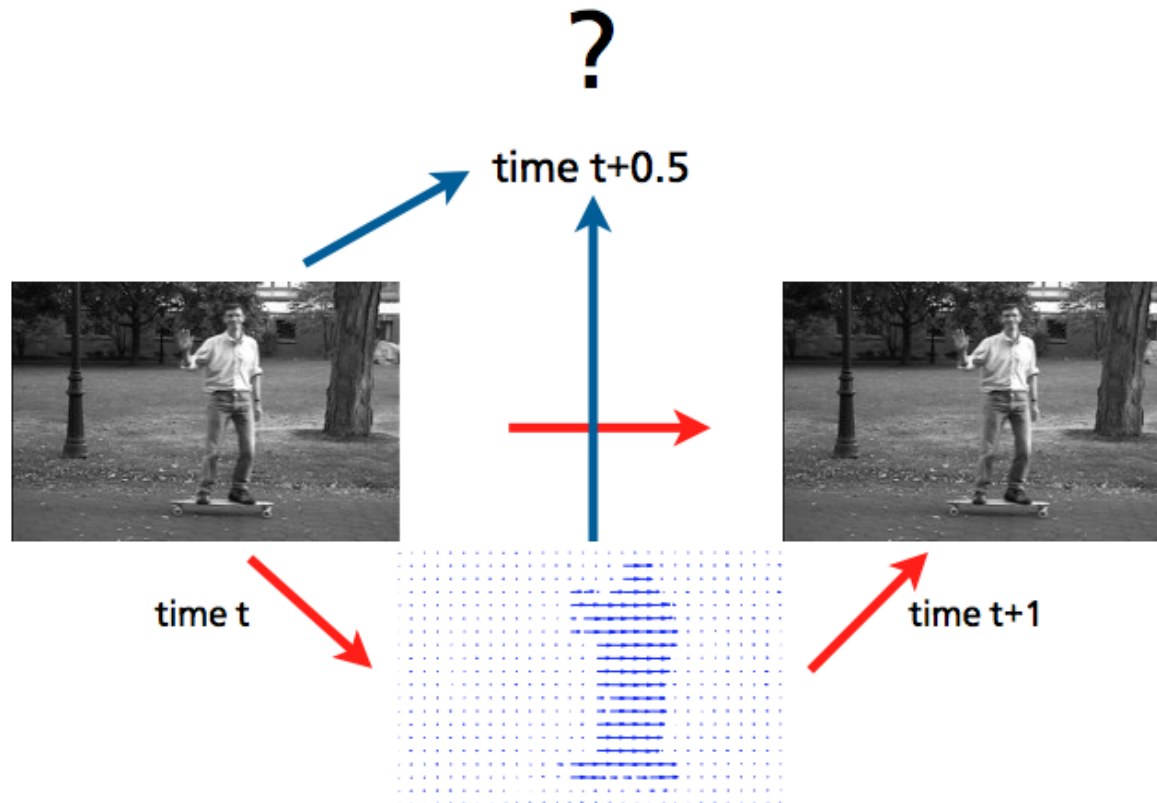


Challenge: Occlusion



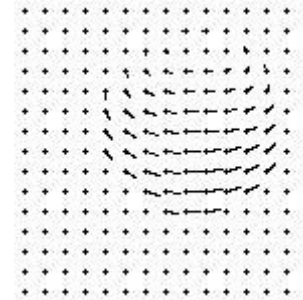
Optical Flow Applications, Frame Interpolation

- If we know the image motion, we can compute images at **intermediate time steps**:



Optical Flow Applications

Tracking objects
video analysis



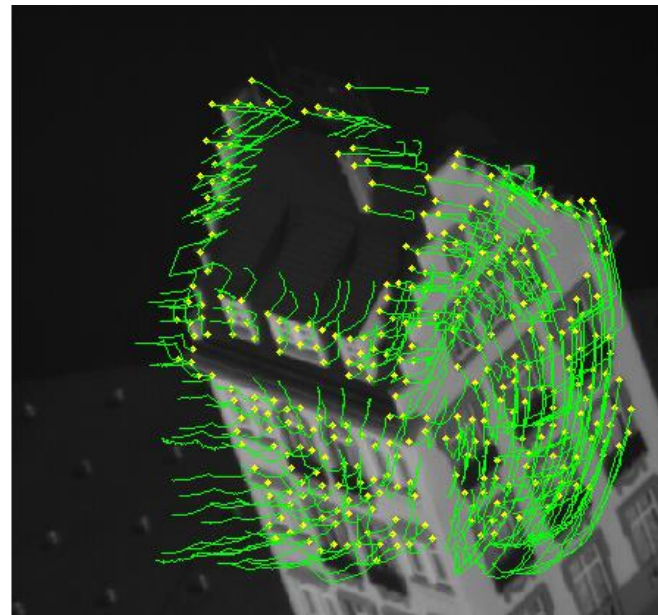
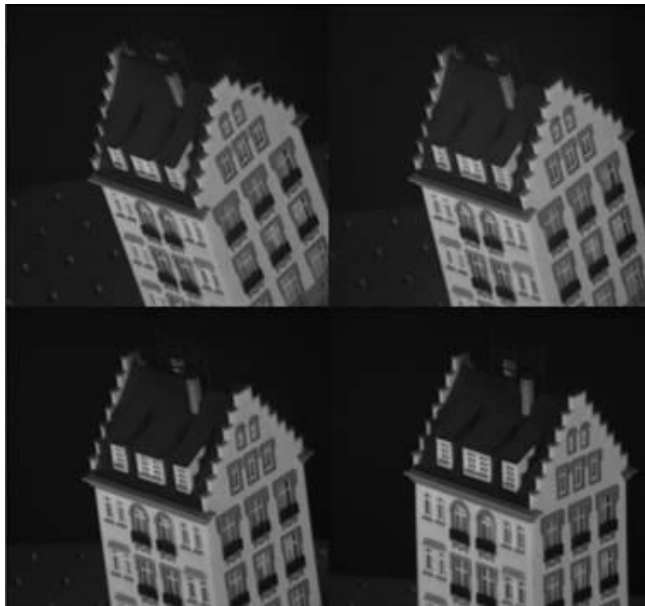
Tomas Izo

Example use of optical flow: facial animation

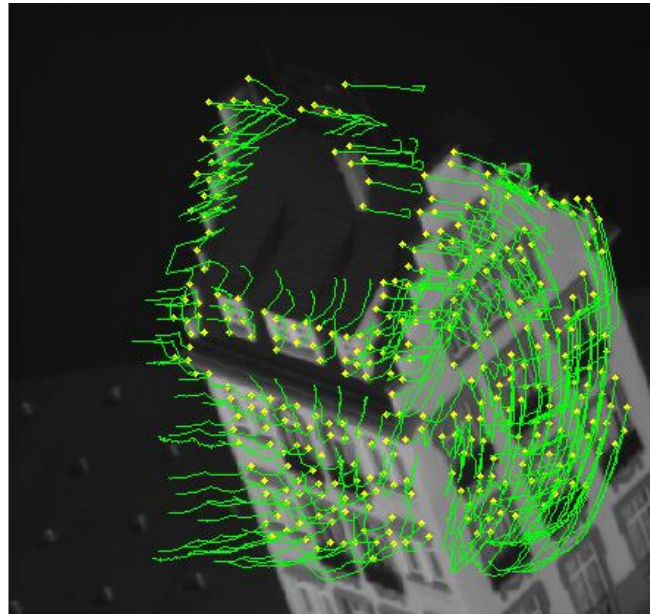
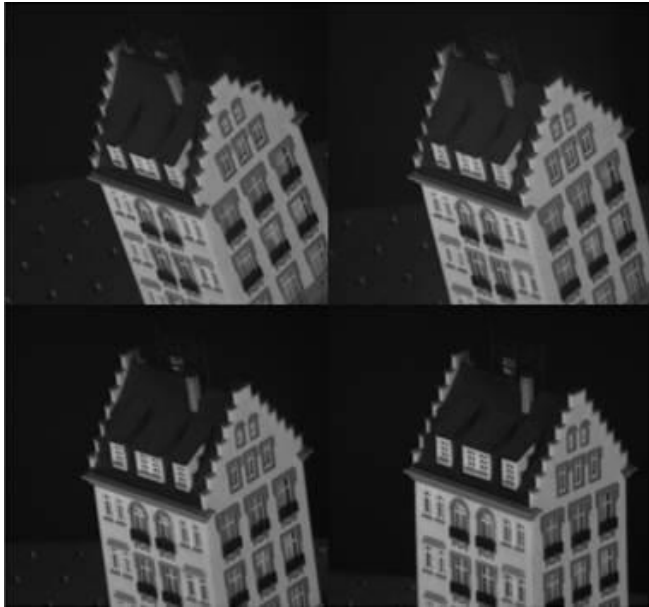


Feature tracking

- If we have more than two images, we can track a feature from one frame to the next by following the optical flow
- Challenges
 - Finding good features to track
 - Adding and deleting tracks



Keypoint Tracking



C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization method." *IJCV*, 9(2):137-154, November 1992.

Summary of KLT (Kanade-Lucas-Tomasi) 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)

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

Sparse Motion Estimation

- 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)



Summary

- Major contributions from Lucas, Tomasi, and 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

Reference

- Szeliski
Section 8.4, Optical Flow