Optical Flow I

Guido Gerig
CS 6643, Spring 2017

(credits: Marc Pollefeys UNC Chapel Hill, Comp 256 / K.H. Shafique, UCSF, CAP5415 / S. Narasimhan, CMU / Bahadir K. Gunturk, EE 7730 / Bradski&Thrun, Stanford CS223

Materials

- Gary Bradski & Sebastian Thrun, Stanford CS223 http://robots.stanford.edu/cs223b/index.html
- S. Narasimhan, CMU: http://www.cs.cmu.edu/afs/cs/academic/class/15385-s06/lectures/ppts/lec-16.ppt
- M. Pollefeys, ETH Zurich/UNC Chapel Hill: http://www.cs.unc.edu/Research/vision/comp256/vision10.ppt
- K.H. Shafique, UCSF: http://www.cs.ucf.edu/courses/cap6411/cap5415/
  - Lecture 18 (March 25, 2003), Slides: PDF/ PPT
- Jepson, Toronto: http://www.cs.toronto.edu/pub/jepson/teaching/vision/2503/opticalFlow.pdf
- Original paper Horn&Schunck 1981: http://www.csd.uwo.ca/faculty/beau/CS9645/PAPERS/Horn-Schunck.pdf
- MIT AI Memo Horn& Schunck 1980: http://people.csail.mit.edu/bkph/AIM/AIM-572.pdf
- Bahadir K. Gunturk, EE 7730 Image Analysis II
- Some slides and illustrations from L. Van Gool, T. Darell, B. Horn, Y. Weiss, P. Anandan, M. Black, K. Toyama

Optical Flow and Motion

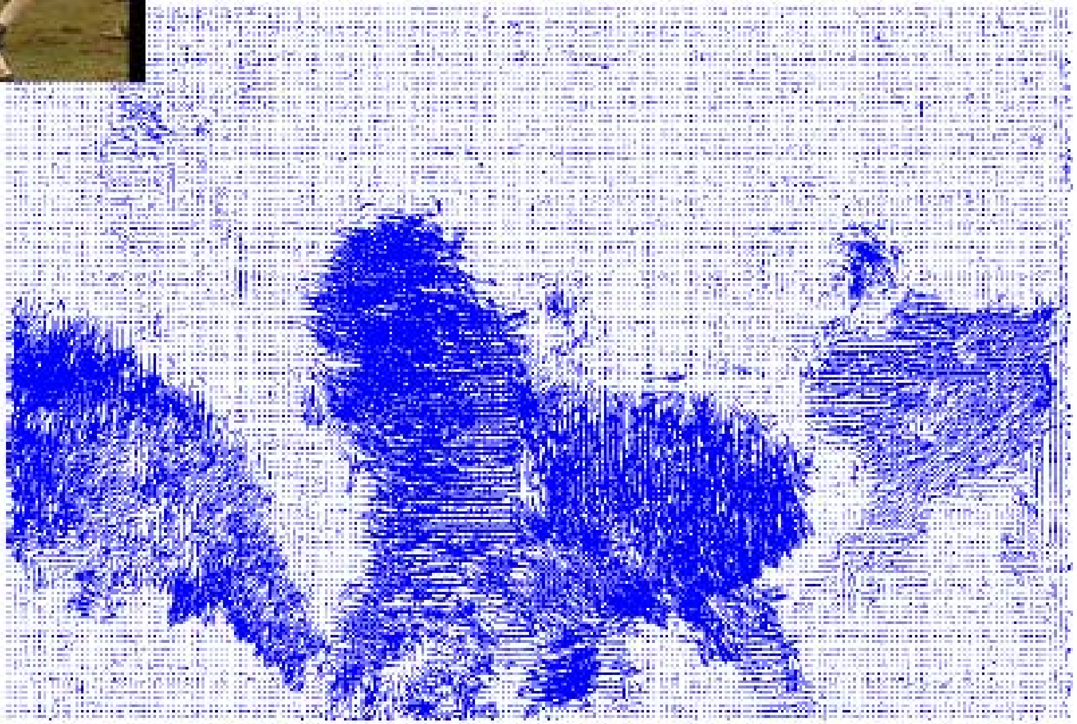
- We are interested in finding the movement of scene objects from time-varying images (videos).
- Lots of uses
  - Motion detection
  - Track objects
  - Correct for camera jitter (stabilization)
  - Align images (mosaics)
  - 3D shape reconstruction
  - Special effects
  - Games: http://www.youtube.com/watch?v=JlLkkom6tWw
  - User Interfaces: http://www.youtube.com/watch?v=Q3gT52sHDI4
  - Video compression

Tracking – Rigid Objects

Tracking – Non-rigid Objects

(Comaniciu et al, Siemens)

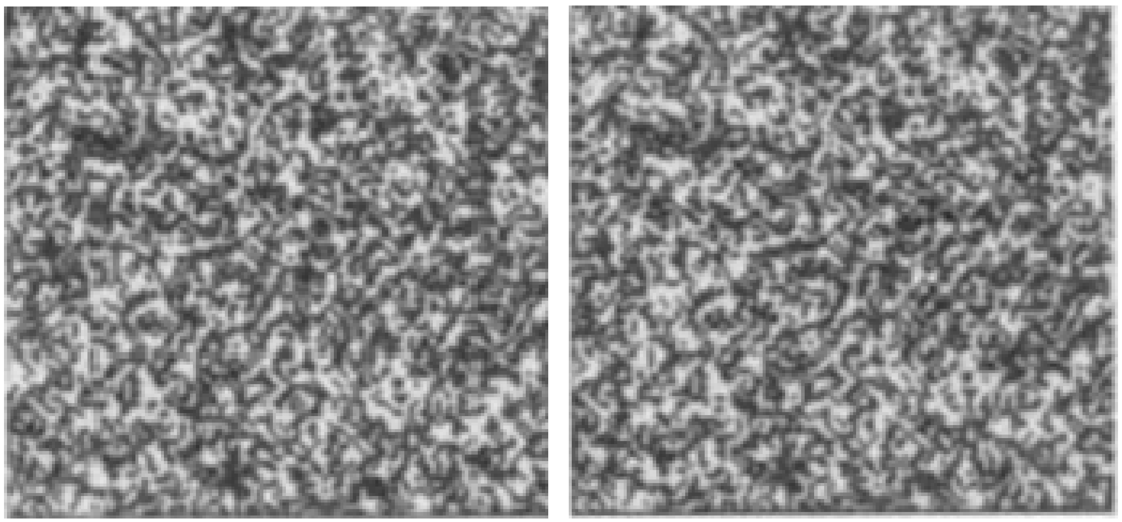
Tracking – Non-rigid Objects



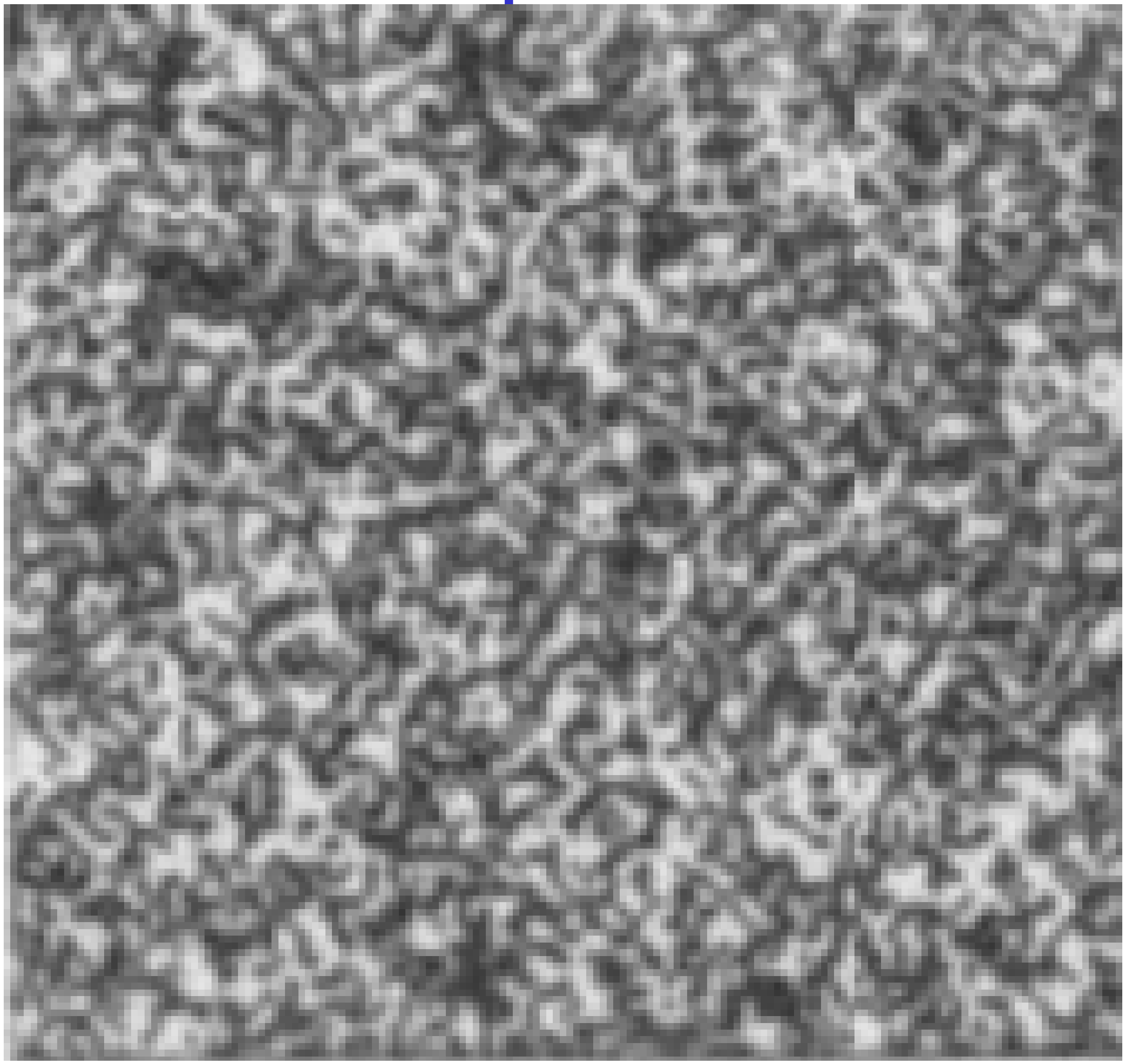
Alper Yilmaz, Fall 2005 UCF

Optical Flow:
Where do pixels move to?

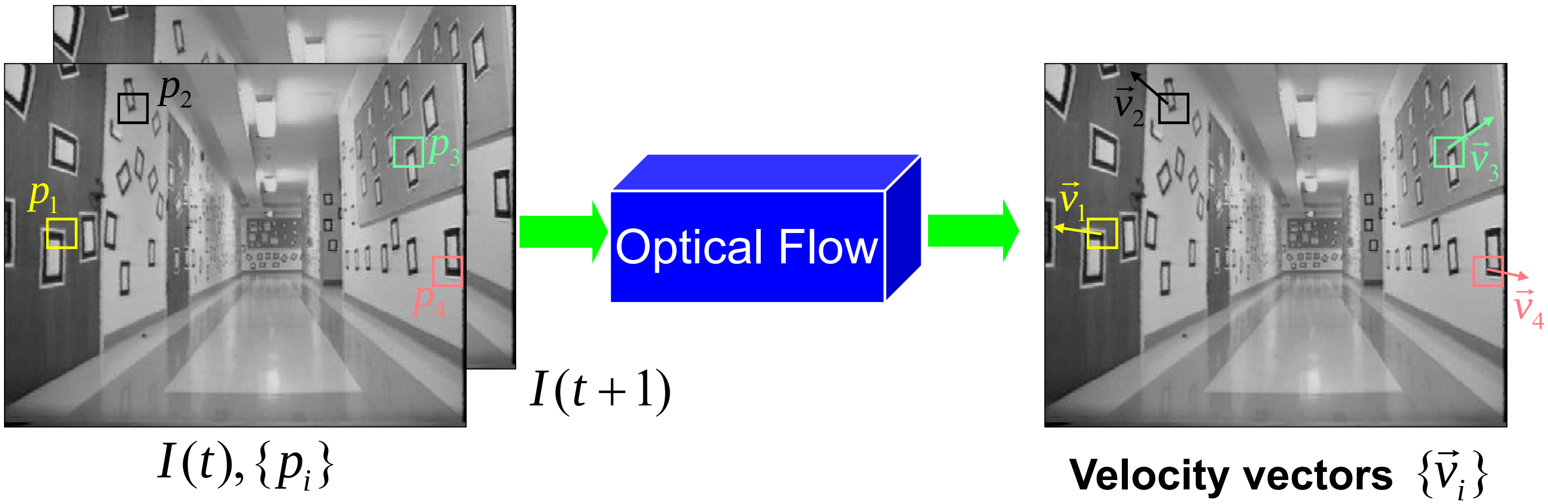
Optical Flow:
Where do pixels move to?



Optical Flow:
Where do pixels move to?



What is Optical Flow (OF)?



Optical flow is the relation of the motion field:

- *the 2D projection of the physical movement of points relative to the observer* to 2D displacement of pixel patches on the image plane.

Common assumption:

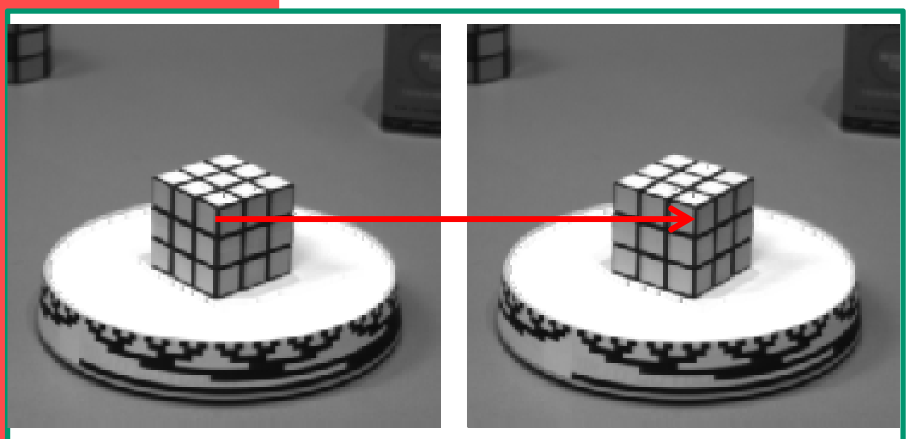
The appearance of the image patches do not change (brightness constancy)

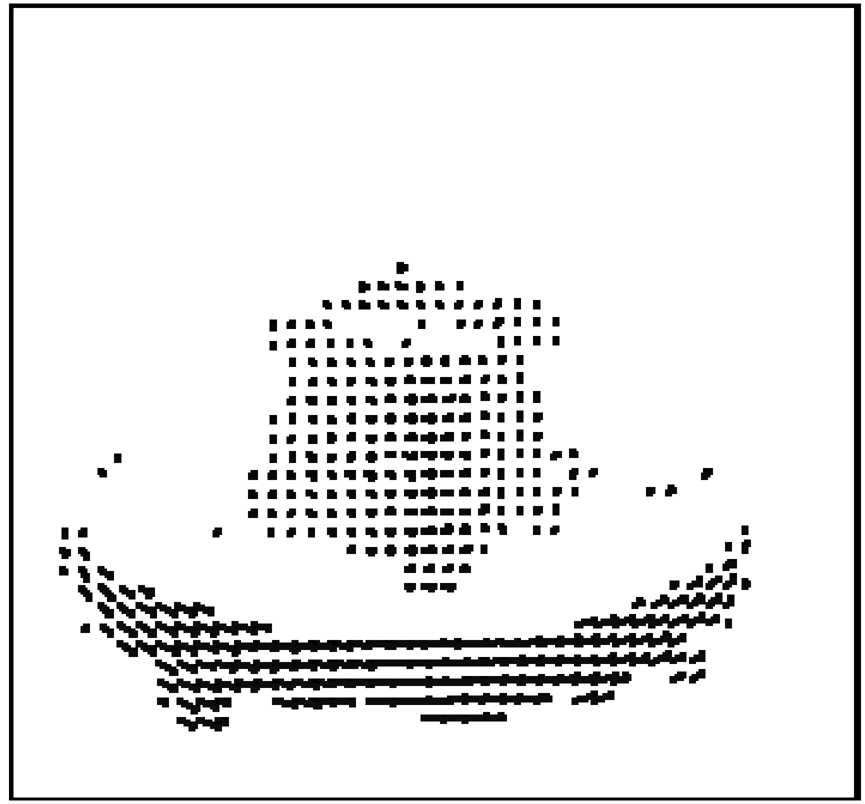
$$I(p_i, t) = I(p_i + \vec{v}_i, t+1)$$

Note: more elaborate tracking models can be adopted if more frames are process all at once

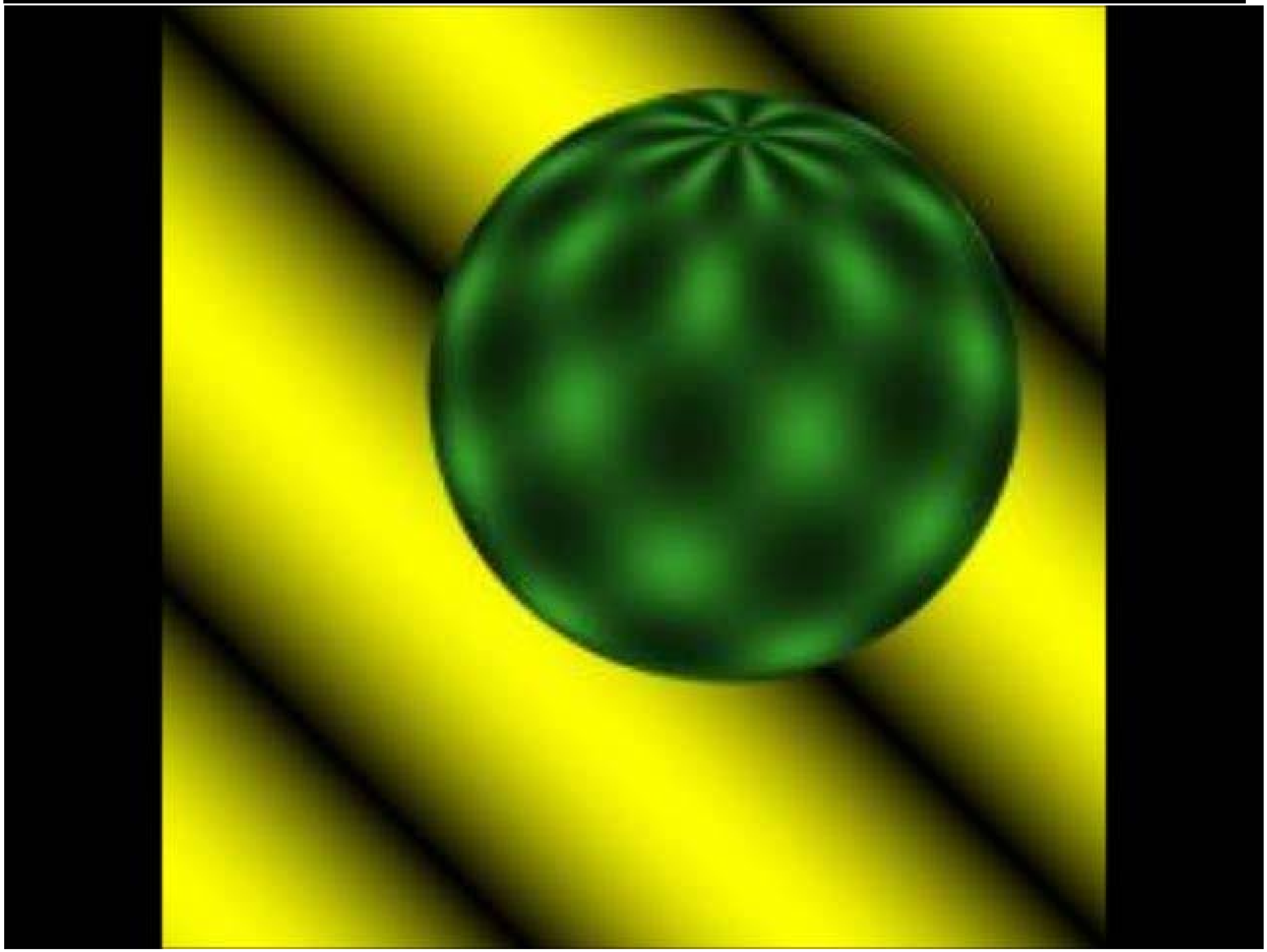
Optical Flow: Correspondence

Basic question: Which Pixel went where?

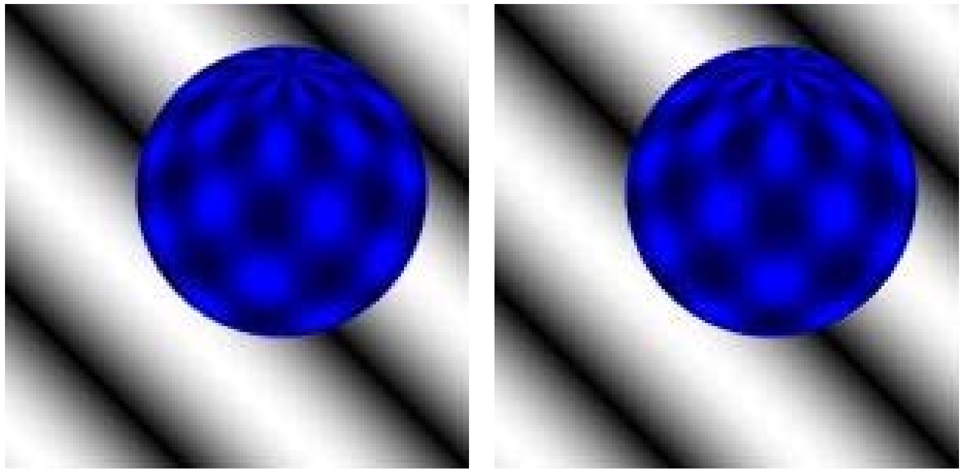


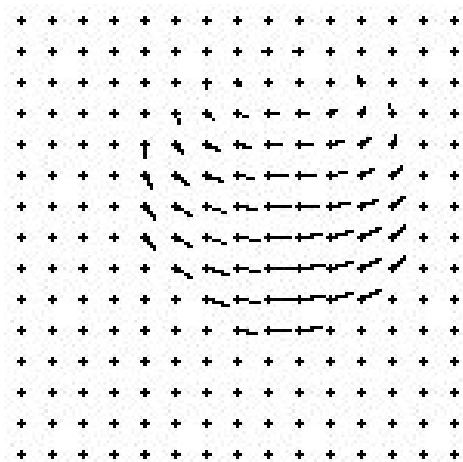


Structure from Motion?



Optical Flow is NOT 3D motion field

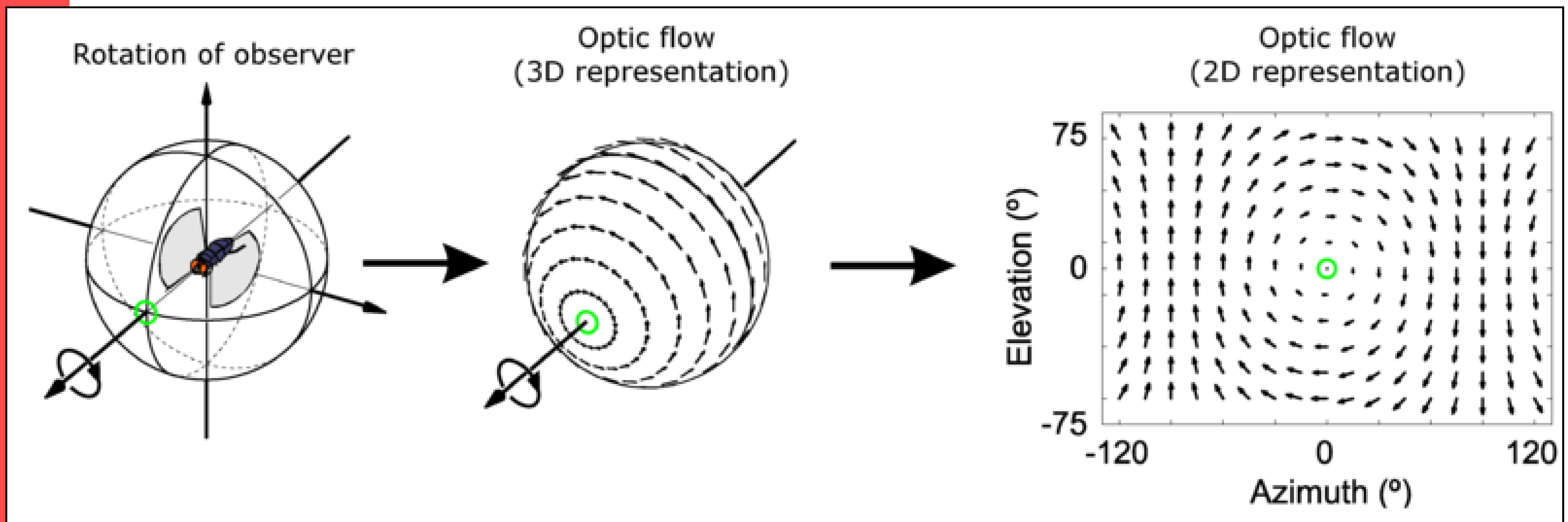




Optical flow: Pixel motion field as observed in image.

http://of-eval.sourceforge.net/

Optical Flow is NOT 3D motion field

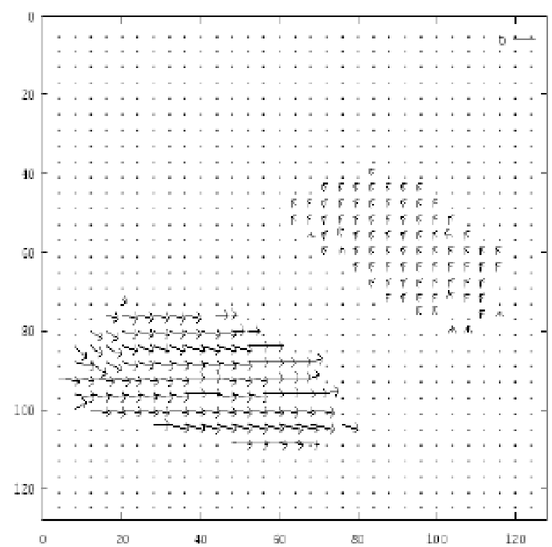


http://en.wikipedia.org/wiki/File:Opticfloweg.png

Definition of optical flow

OPTICAL FLOW = apparent motion of brightness patterns

Ideally, the optical flow is the projection of the three-dimensional velocity vectors on the image



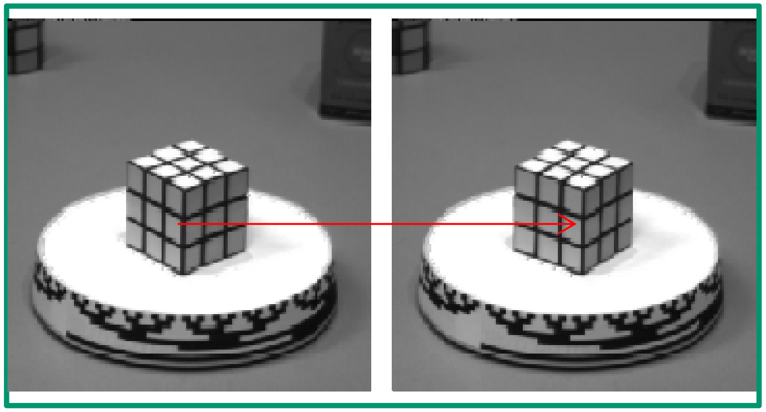
Optical Flow - Agenda

- Brightness Constancy
- The Aperture problem
- Regularization
- Lucas-Kanade
- Coarse-to-fine
- Parametric motion models
- Direct depth
- SSD tracking
- Robust flow
- Bayesian flow

Optical Flow - Agenda

- Brightness Constancy
- The Aperture problem
- Regularization
- Lucas-Kanade
- Coarse-to-fine
- Parametric motion models
- Direct depth
- SSD tracking
- Robust flow
- Bayesian flow

Start with an Equation: Brightness Constancy



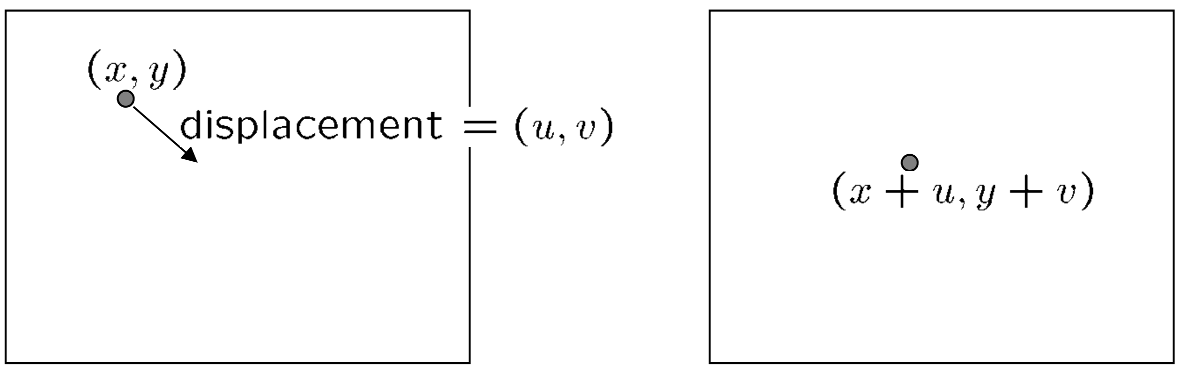
Time: t

Time: t + dt

Point moves (small), but its brightness remains constant:

$$I_{t1}(x, y) = I_{t2}(x + u, y + v)$$

$$I = constant \;\rightarrow\; \frac{dI}{dt} = 0$$



I_1 I_2

Mathematical formulation (1D)

$I\ (x(t),t)$ = brightness at (x) at time t

Brightness constancy assumption (shift of location but brightness stays same):

$$I(x + \frac{dx}{dt}\delta t, t + \delta t) = I(x, y, t)$$

Optical flow constraint equation (chain rule):

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

Optical Flow: 1D Case

Brightness Constancy Assumption:

$$f(t) \equiv I(x(t),t) = I(x(t+dt),t+dt)$$

Optical Flow: 1D Case

Brightness Constancy Assumption:

$$f(t) \equiv I(\underbrace{x(t),t}_{}) = I(x(t+dt),t+dt)$$

$$\frac{\partial f(x)}{\partial t} = 0$$ Because no change in brightness with time

Optical Flow: 1D Case

Brightness Constancy Assumption:

$$f(t) \equiv I\underbrace{(x(t),t)} = I(x(t+dt),t+dt)$$

$$\frac{\partial f(x)}{\partial t} = 0$$ Because no change in brightness with time

$$\left.\frac{\partial I}{\partial x}\right|_t \left(\frac{\partial x}{\partial t}\right) + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

Optical Flow: 1D Case

Brightness Constancy Assumption:

$$f(t) \equiv I\underbrace{(x(t),t)} = I(x(t+dt),t+dt)$$

$$\frac{\partial f(x)}{\partial t} = 0$$ Because no change in brightness with time

$$\left.\frac{\partial I}{\partial x}\right|_{t}\left(\frac{\partial x}{\partial t}\right) + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

I_x v I_t

Optical Flow: 1D Case

Brightness Constancy Assumption:

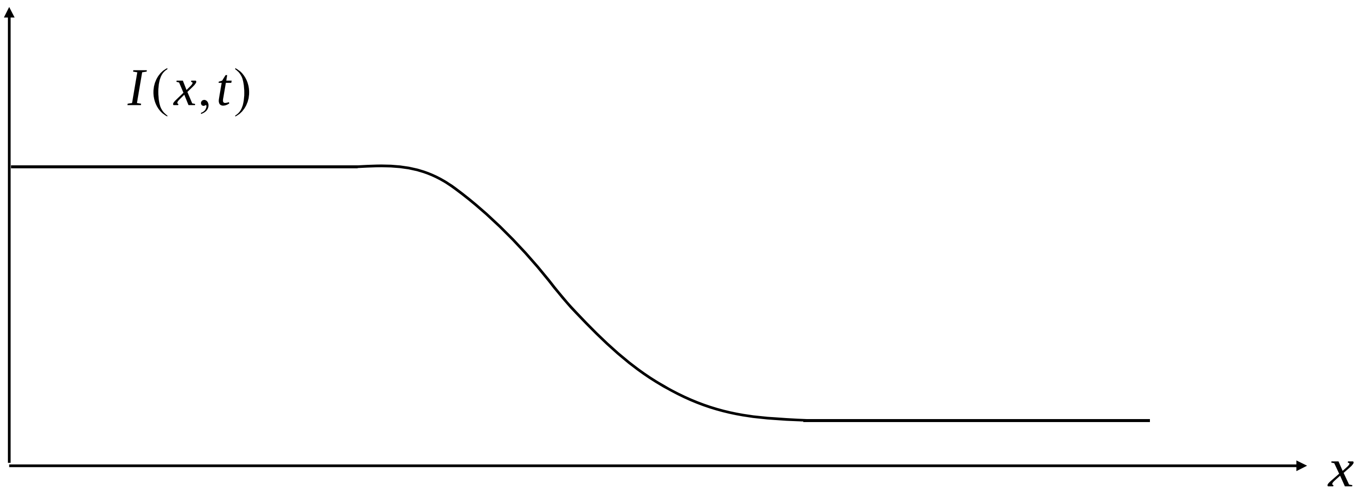
$$f(t) \equiv I(\underbrace{x(t),t}) = I(x(t+dt), t+dt)$$

$$\frac{\partial f(x)}{\partial t} = 0 \quad \text{Because no change in brightness with time}$$

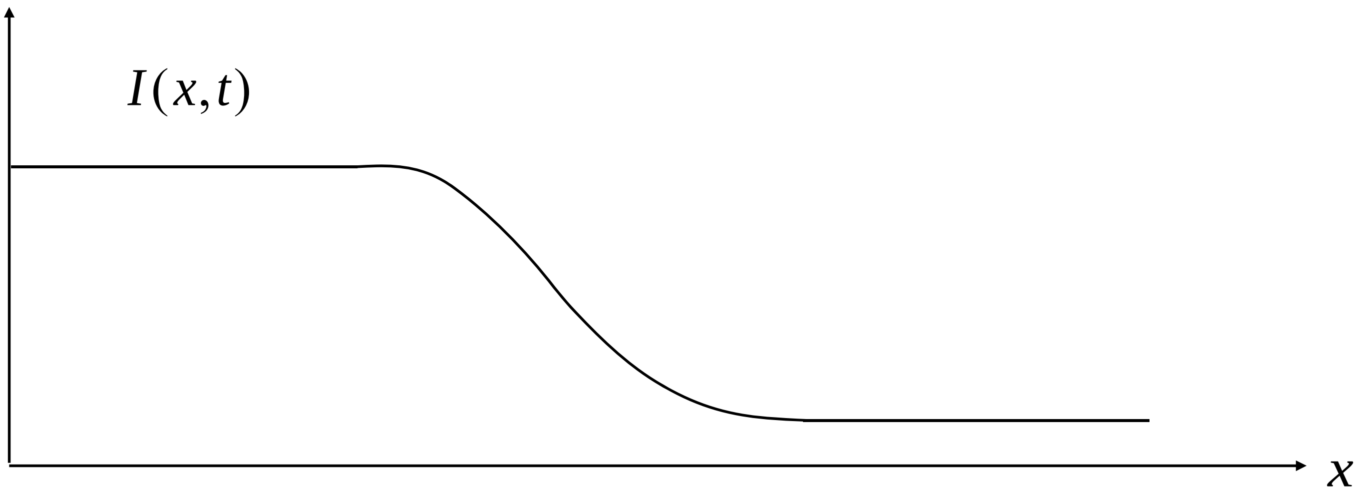
$$\underbrace{\left.\frac{\partial I}{\partial x}\right|_t}_{I_x} \underbrace{\left(\frac{\partial x}{\partial t}\right)}_{v} + \underbrace{\left.\frac{\partial I}{\partial t}\right|_{x(t)}}_{I_t} = 0$$

$$\Rightarrow \quad v = -\frac{I_t}{I_x}$$

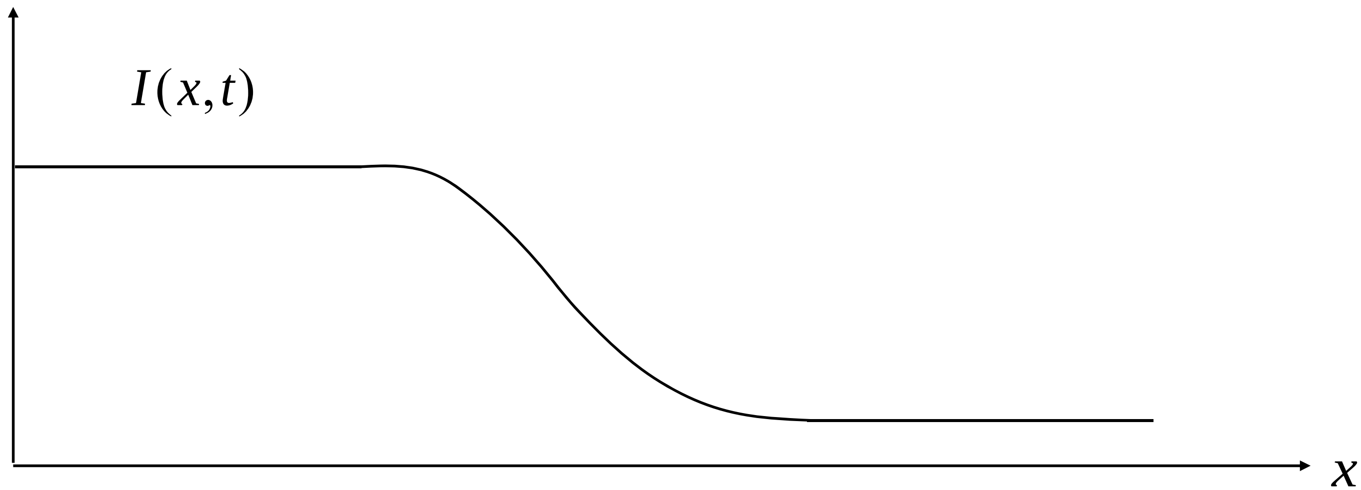
Tracking in the 1D case:



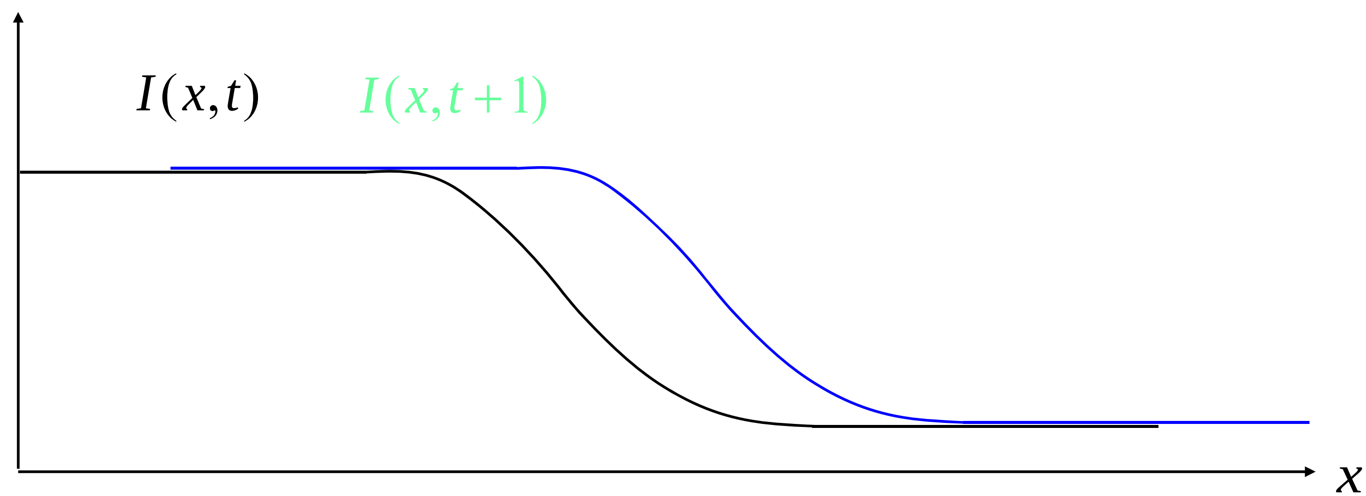
Tracking in the 1D case:



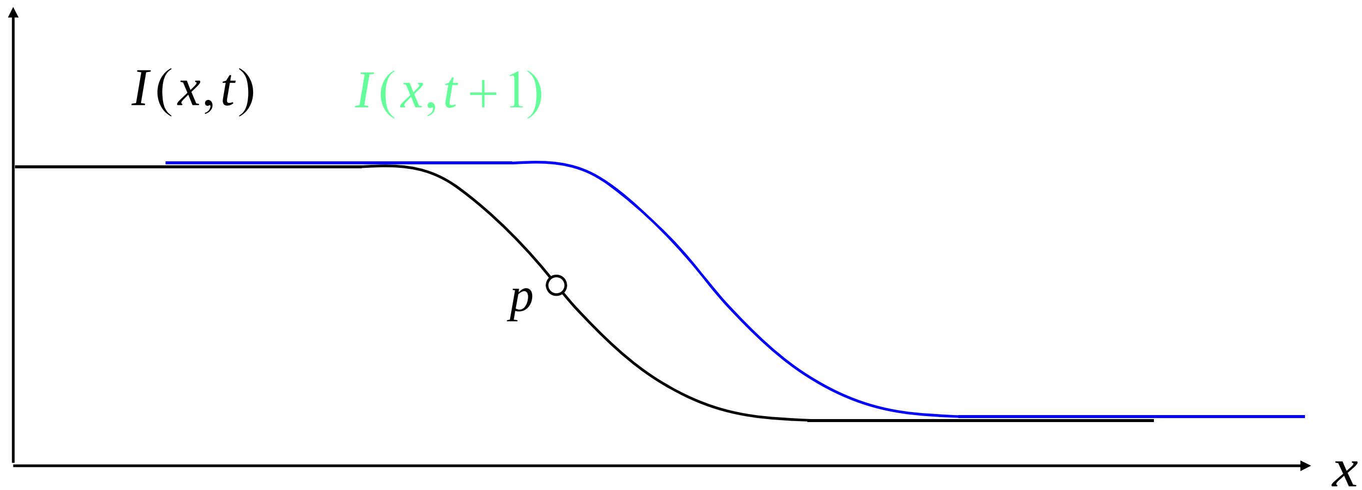
Tracking in the 1D case:



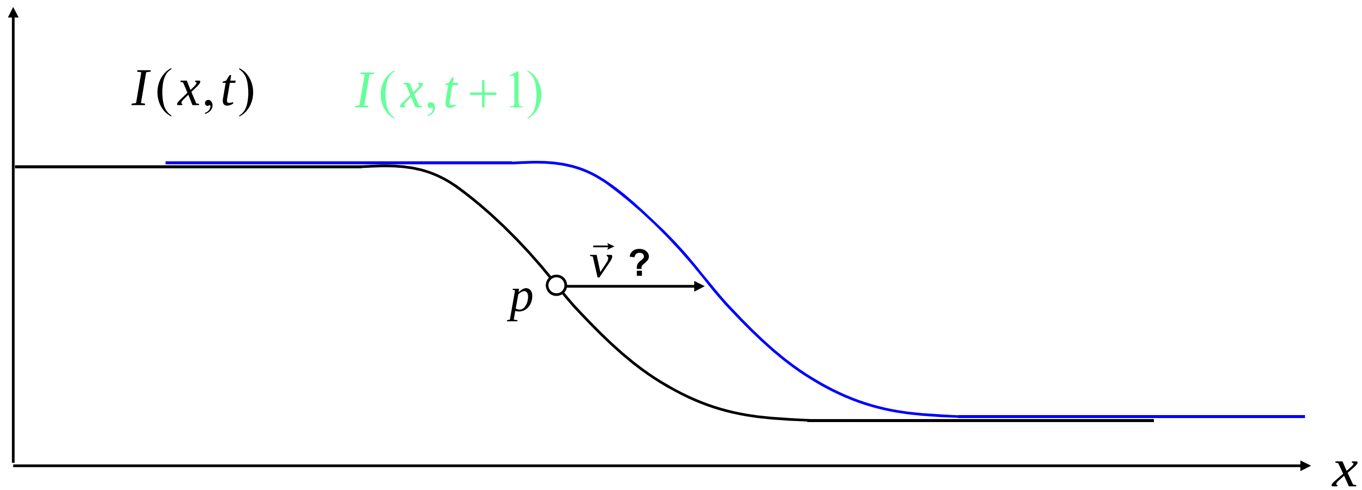
Tracking in the 1D case:



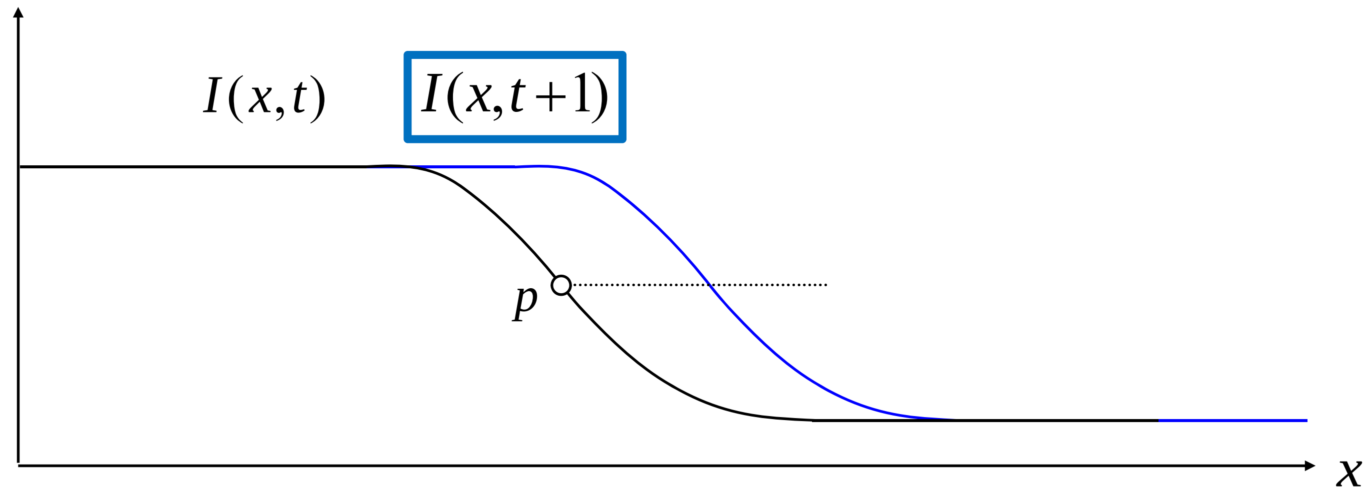
Tracking in the 1D case:



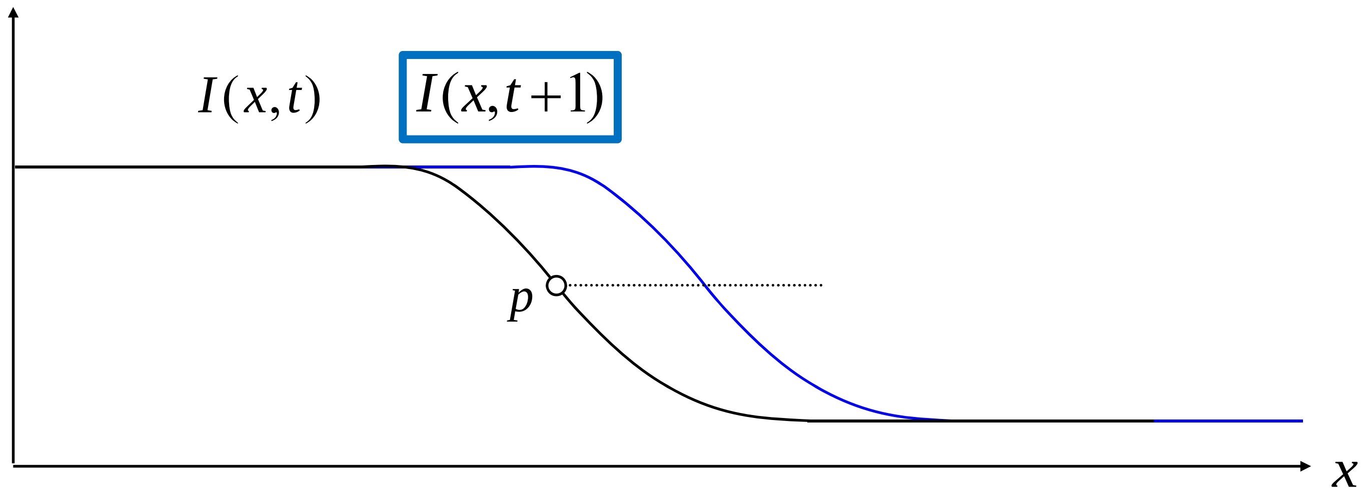
Tracking in the 1D case:



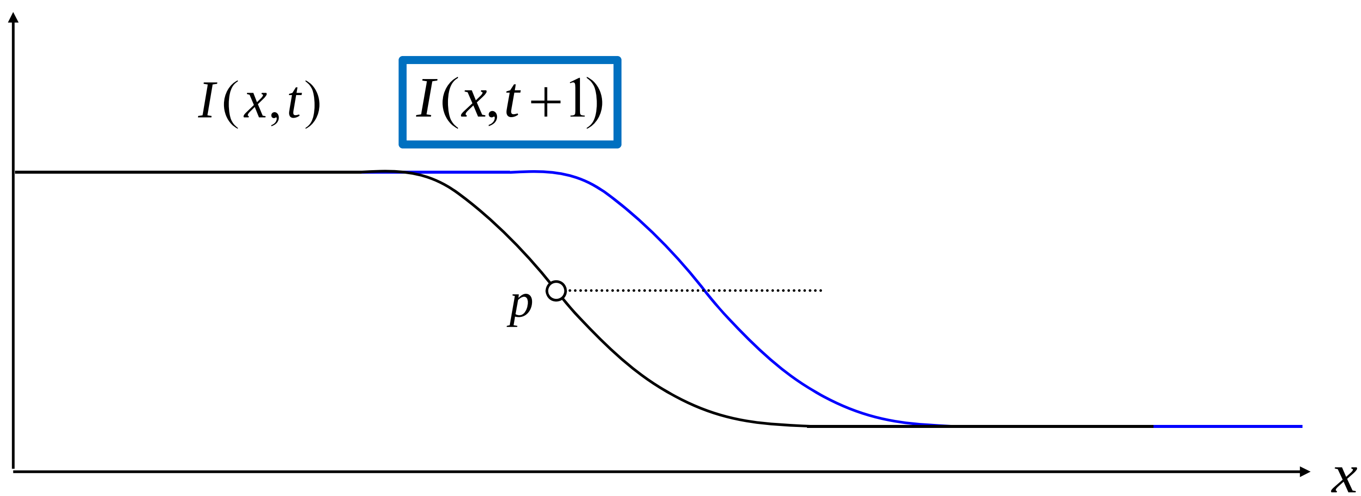
Tracking in the 1D case:



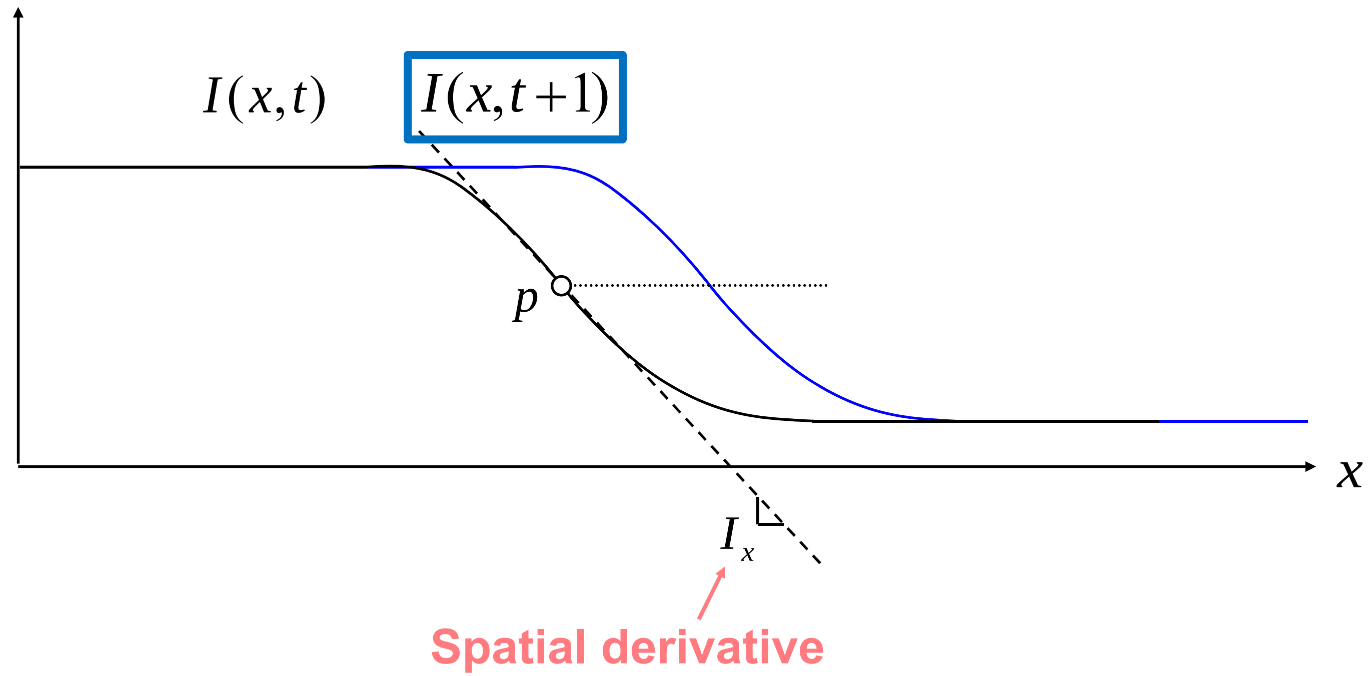
Tracking in the 1D case:



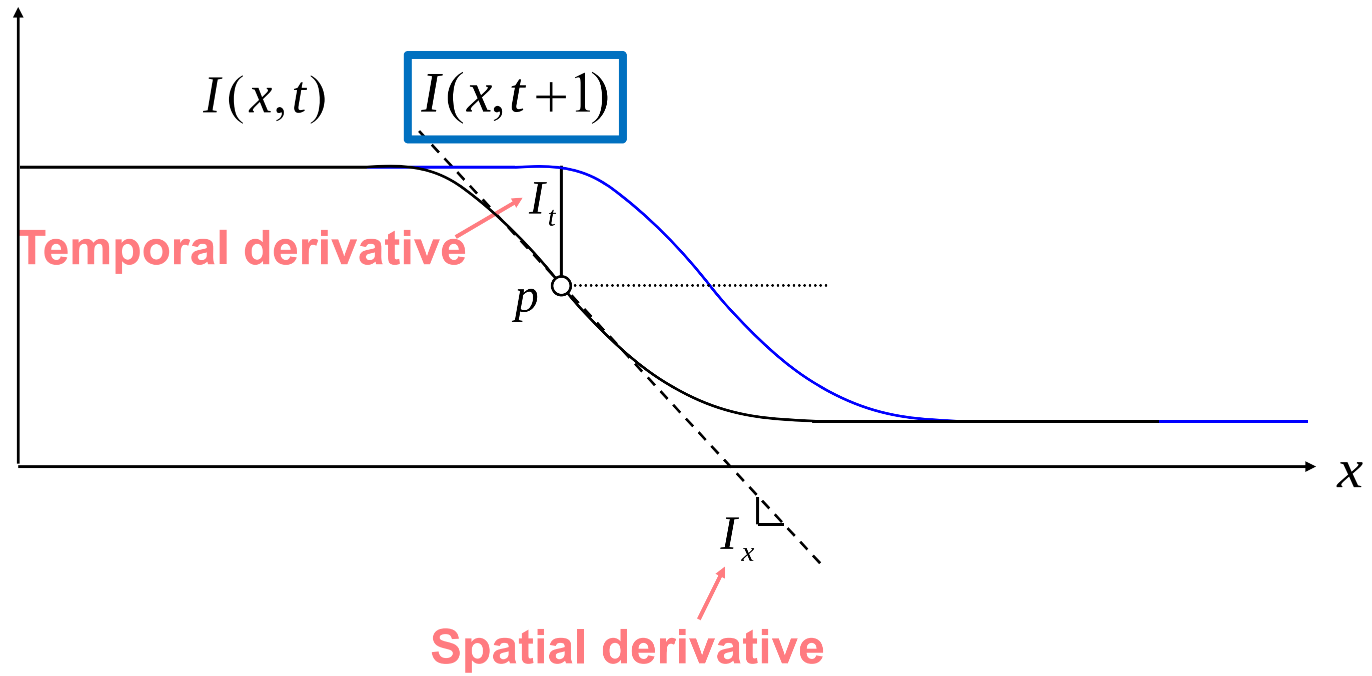
Tracking in the 1D case:



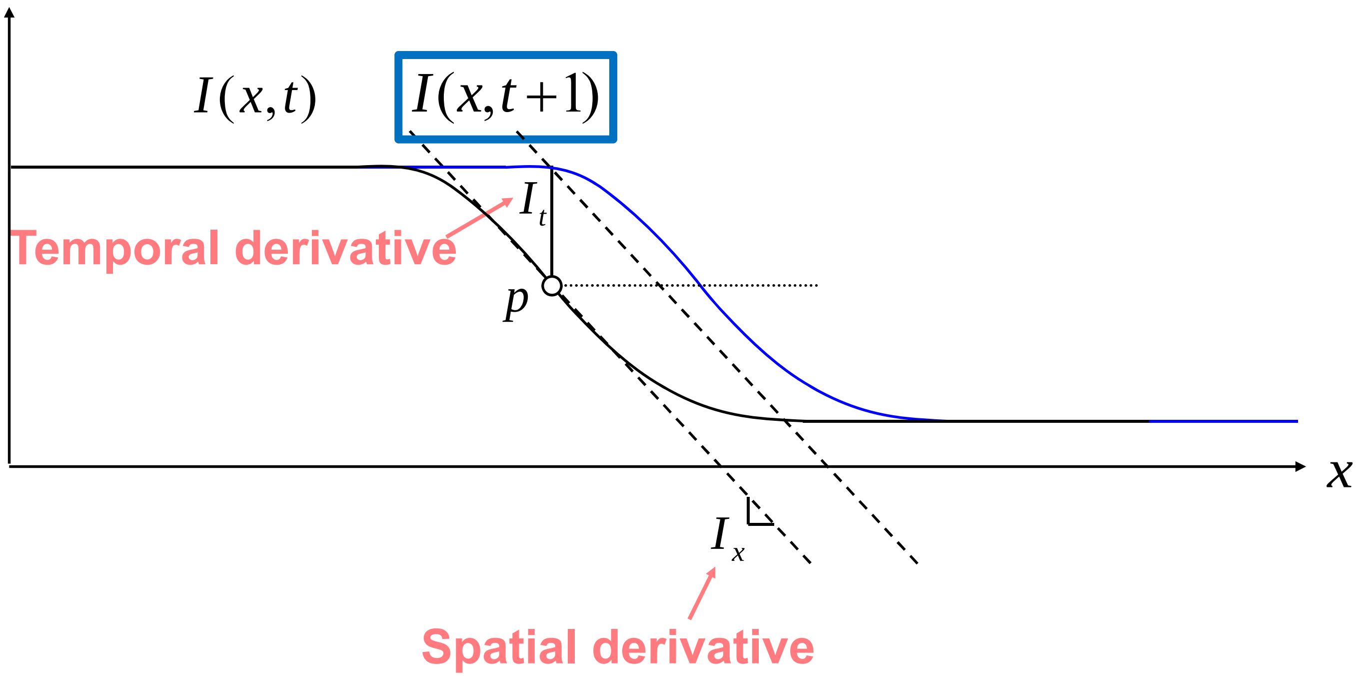
Tracking in the 1D case:



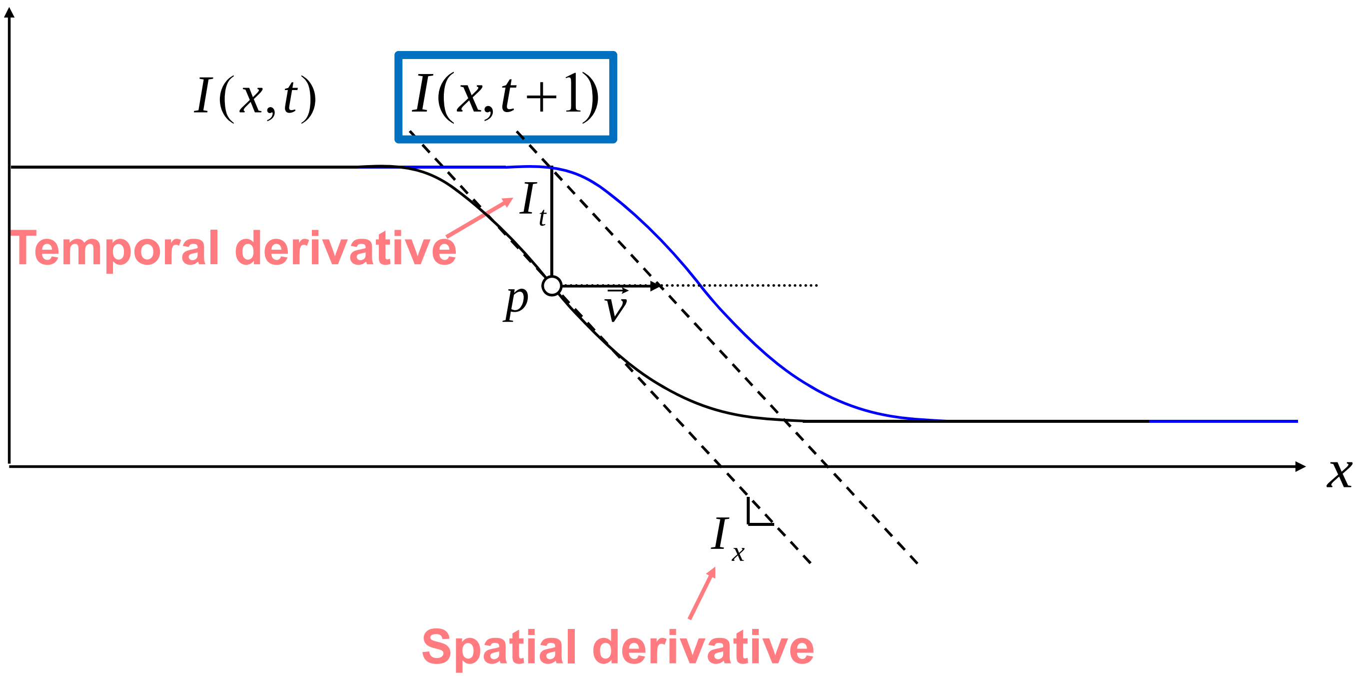
Tracking in the 1D case:



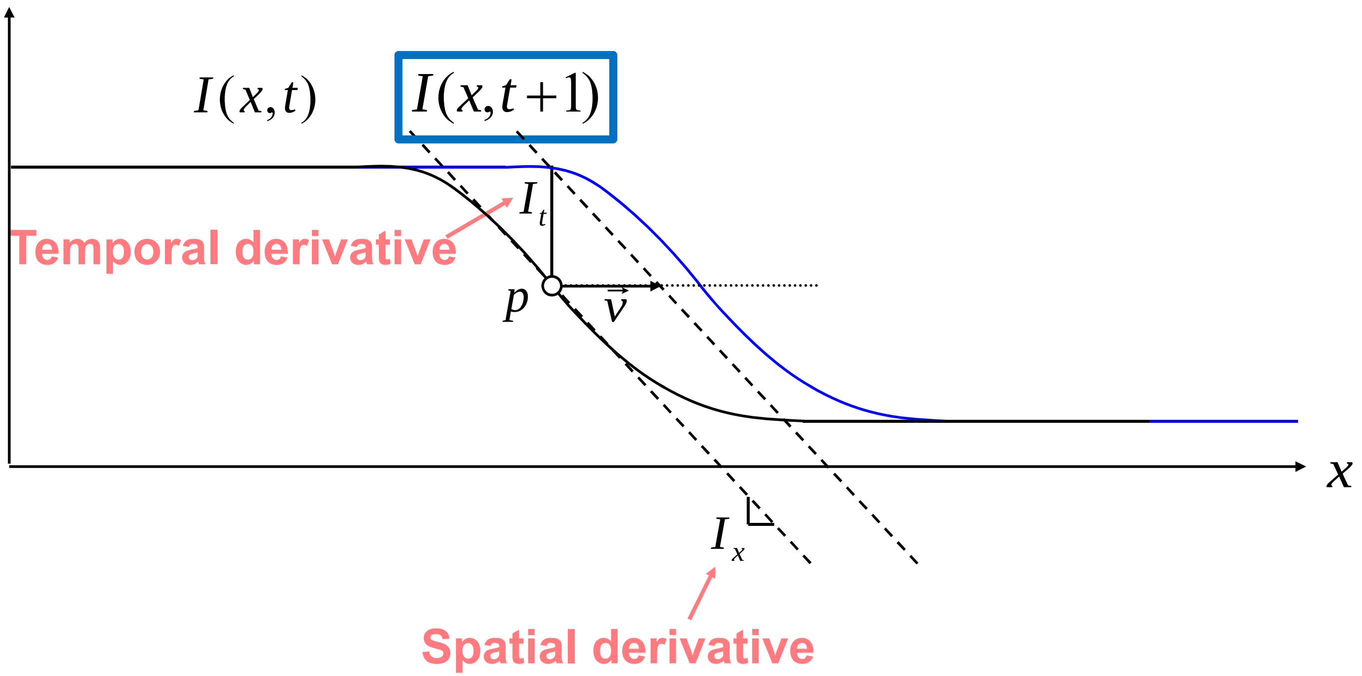
Tracking in the 1D case:



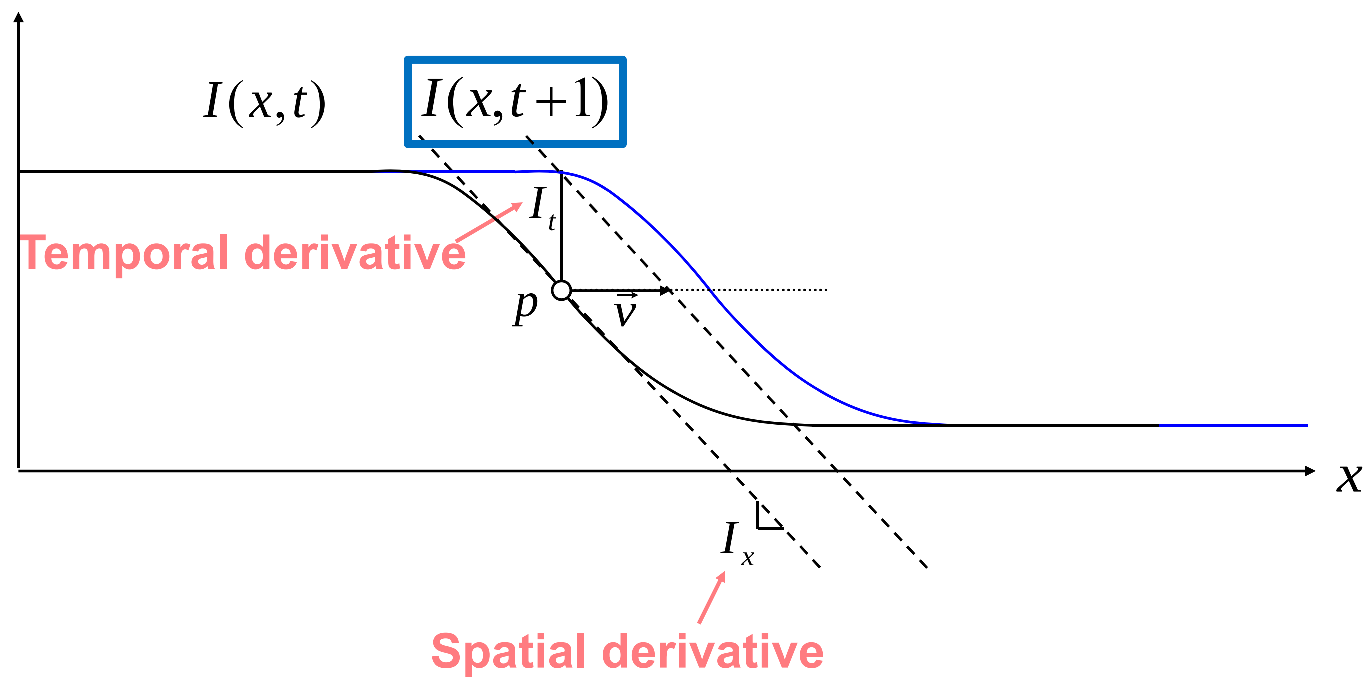
Tracking in the 1D case:



Tracking in the 1D case:



Tracking in the 1D case:



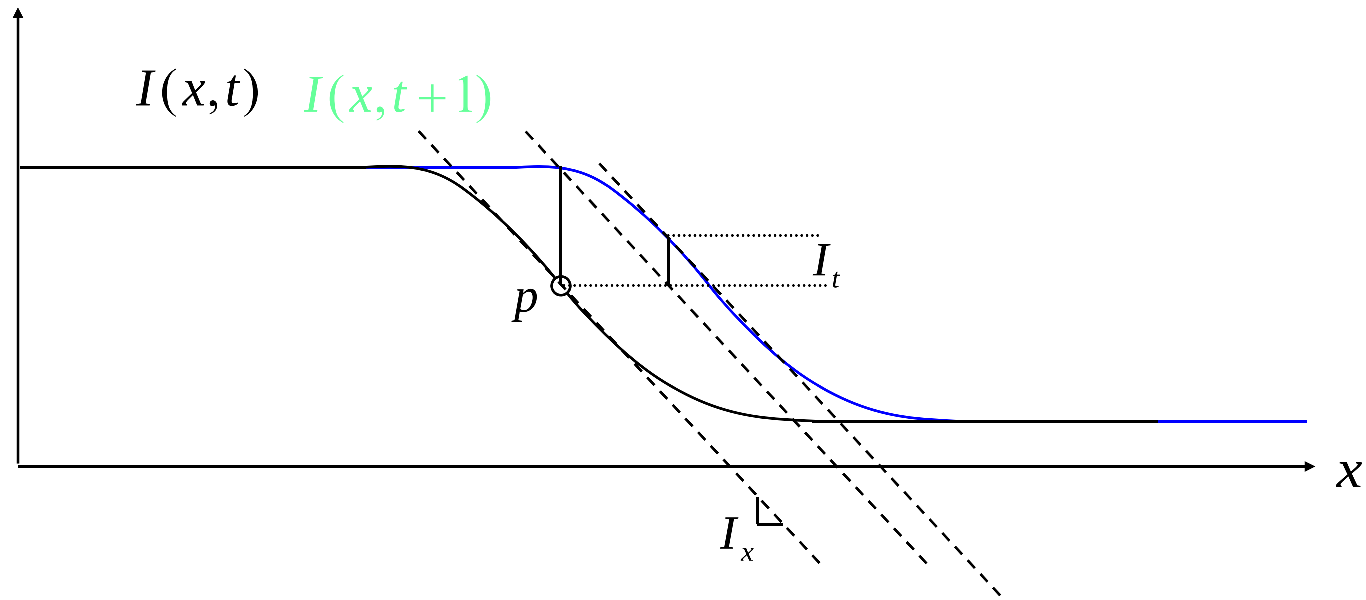
$$I_x = \frac{\partial I}{\partial x}\bigg|_t \qquad I_t = \frac{\partial I}{\partial t}\bigg|_{x=p} \qquad \Rightarrow \qquad \vec{v} \approx -\frac{I_t}{I_x}$$

Assumptions:

- **Brightness constancy**
- **Small motion**

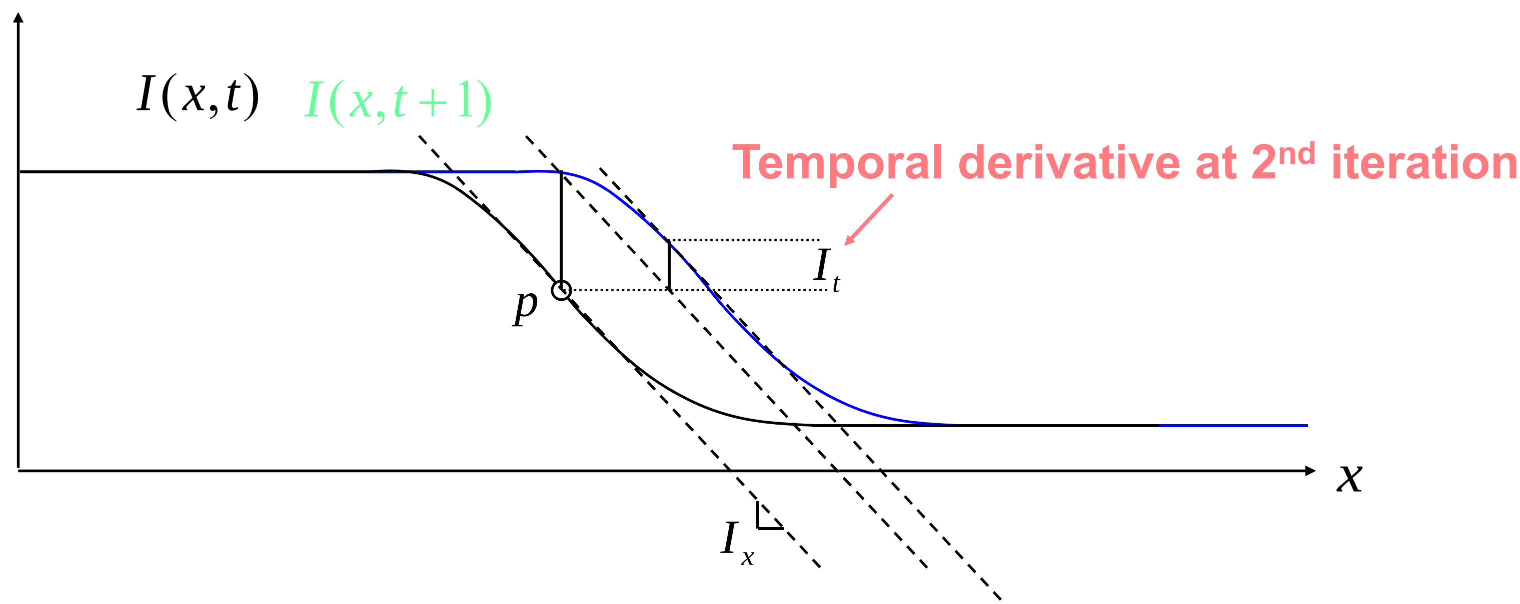
Tracking in the 1D case:

Iterating helps refining the velocity vector



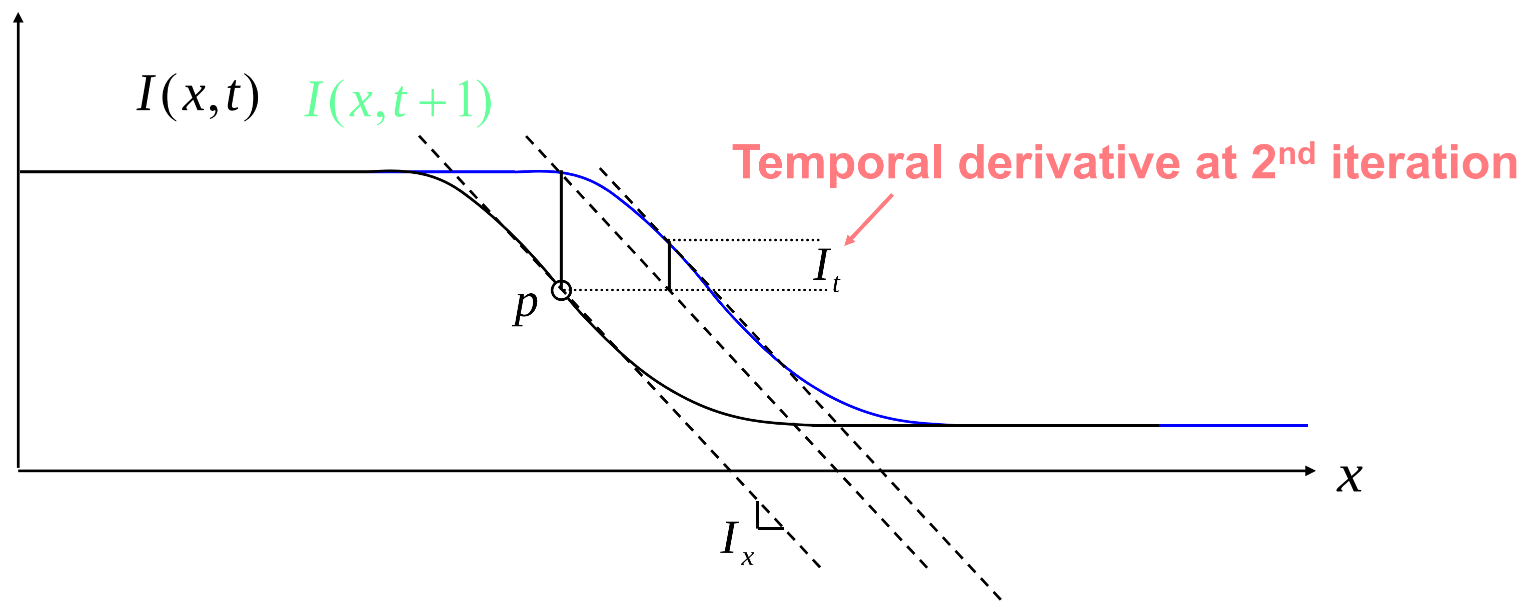
Tracking in the 1D case:

Iterating helps refining the velocity vector



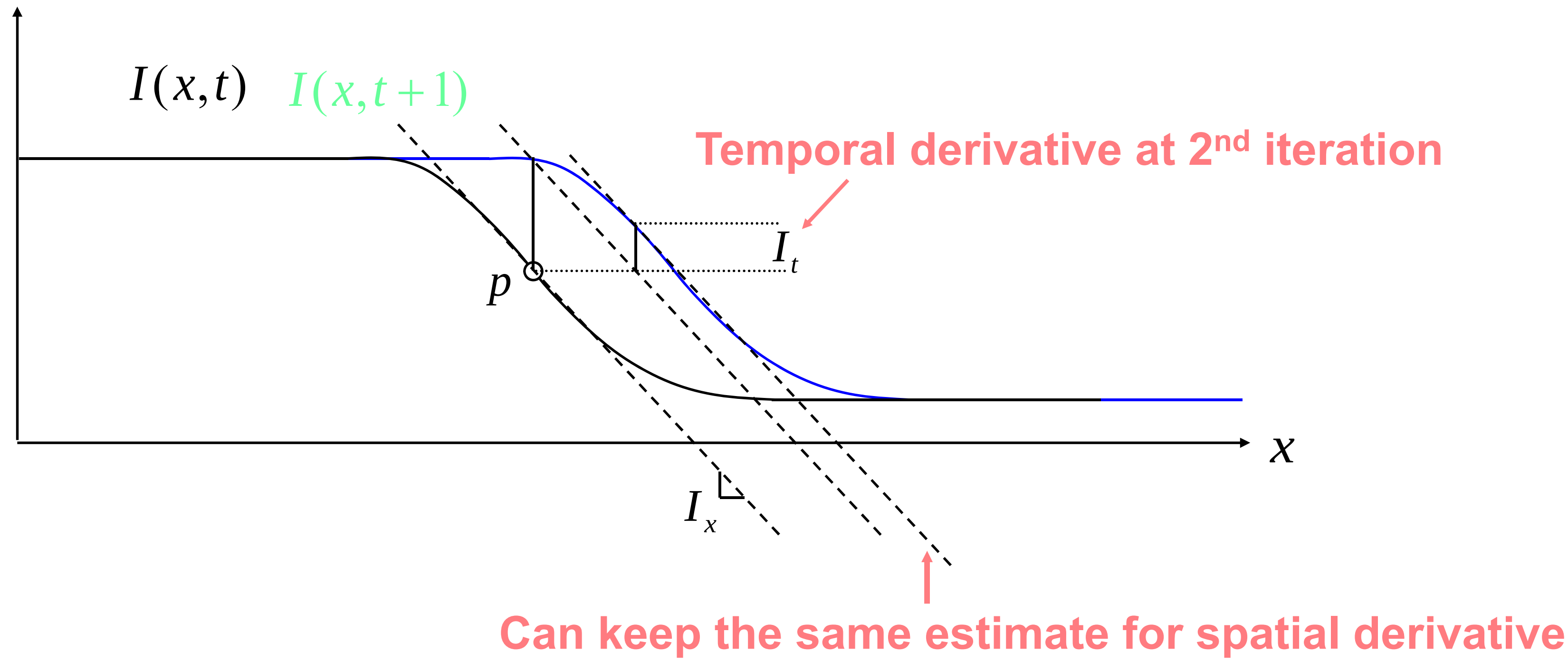
Tracking in the 1D case:

Iterating helps refining the velocity vector



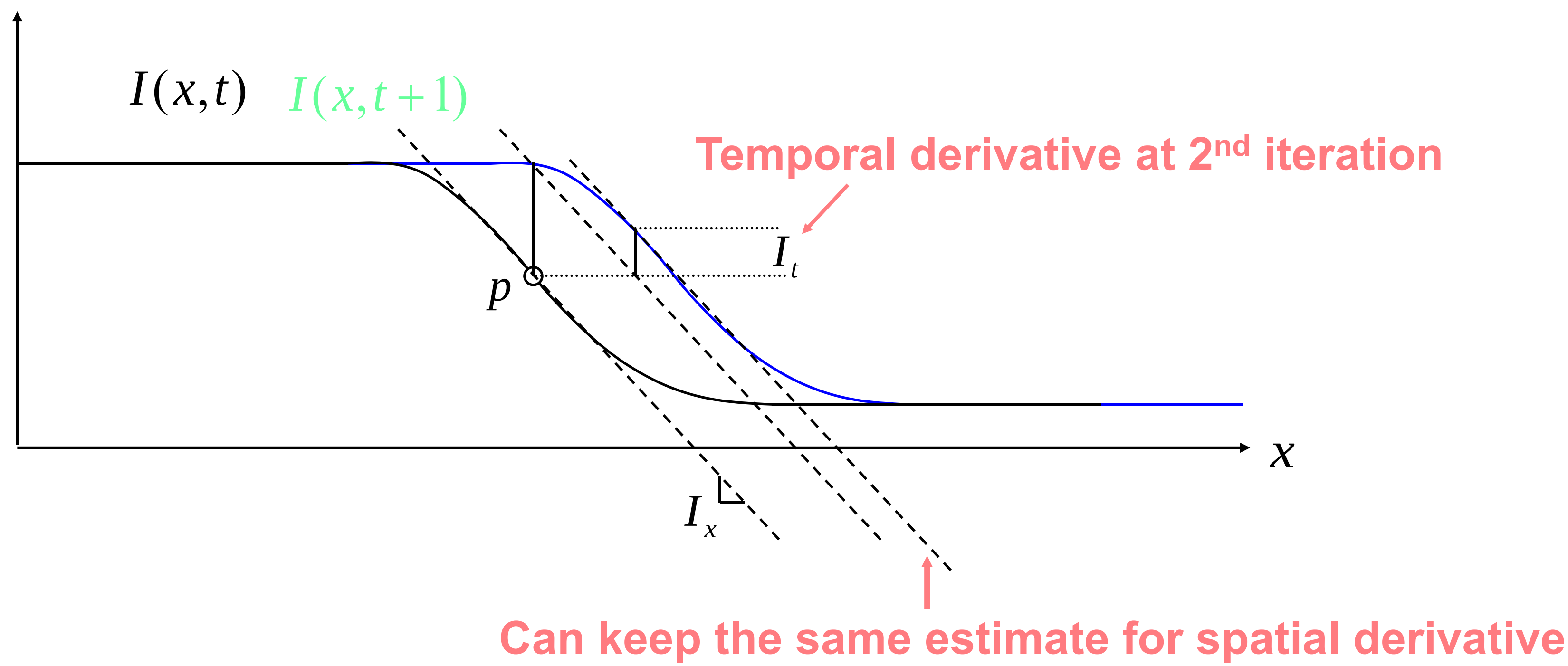
Tracking in the 1D case:

Iterating helps refining the velocity vector



Tracking in the 1D case:

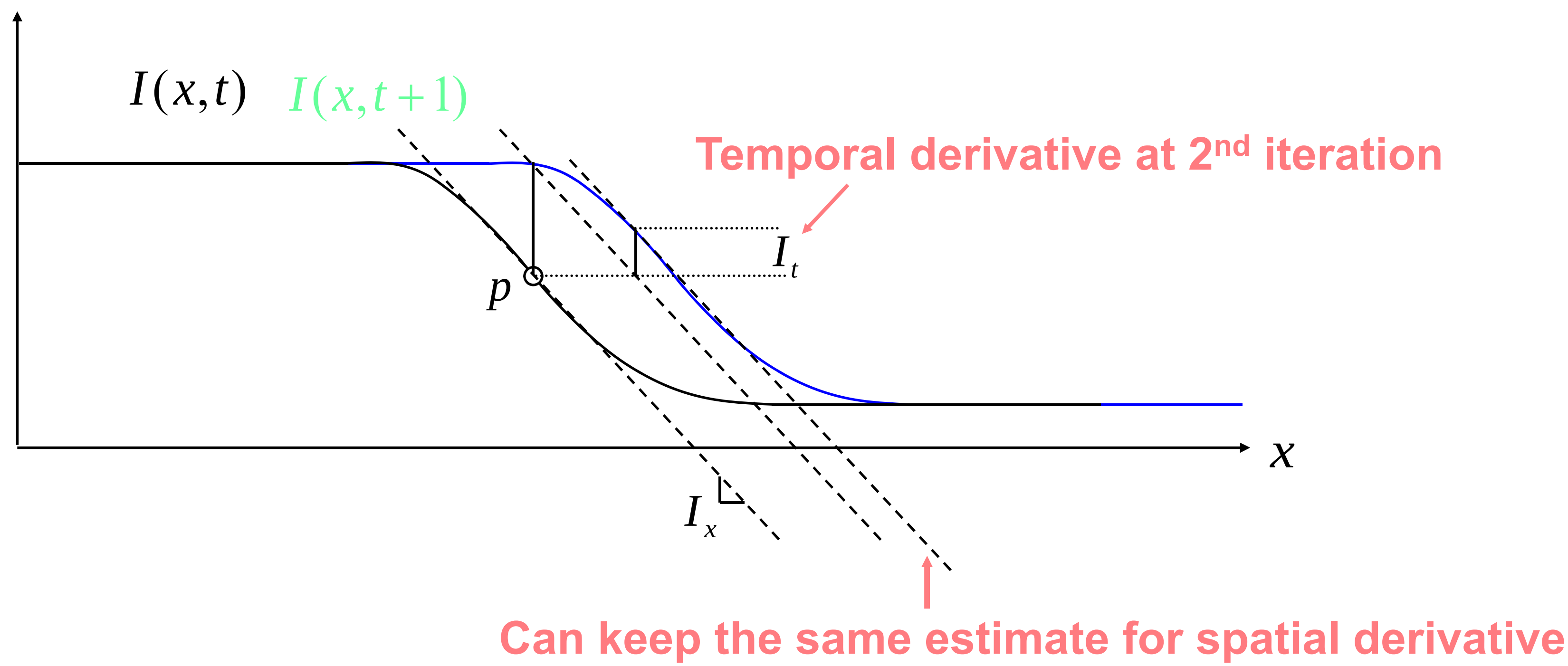
Iterating helps refining the velocity vector



$$\vec{v} \leftarrow \vec{v}_{previous} - \frac{I_t}{I_x}$$

Tracking in the 1D case:

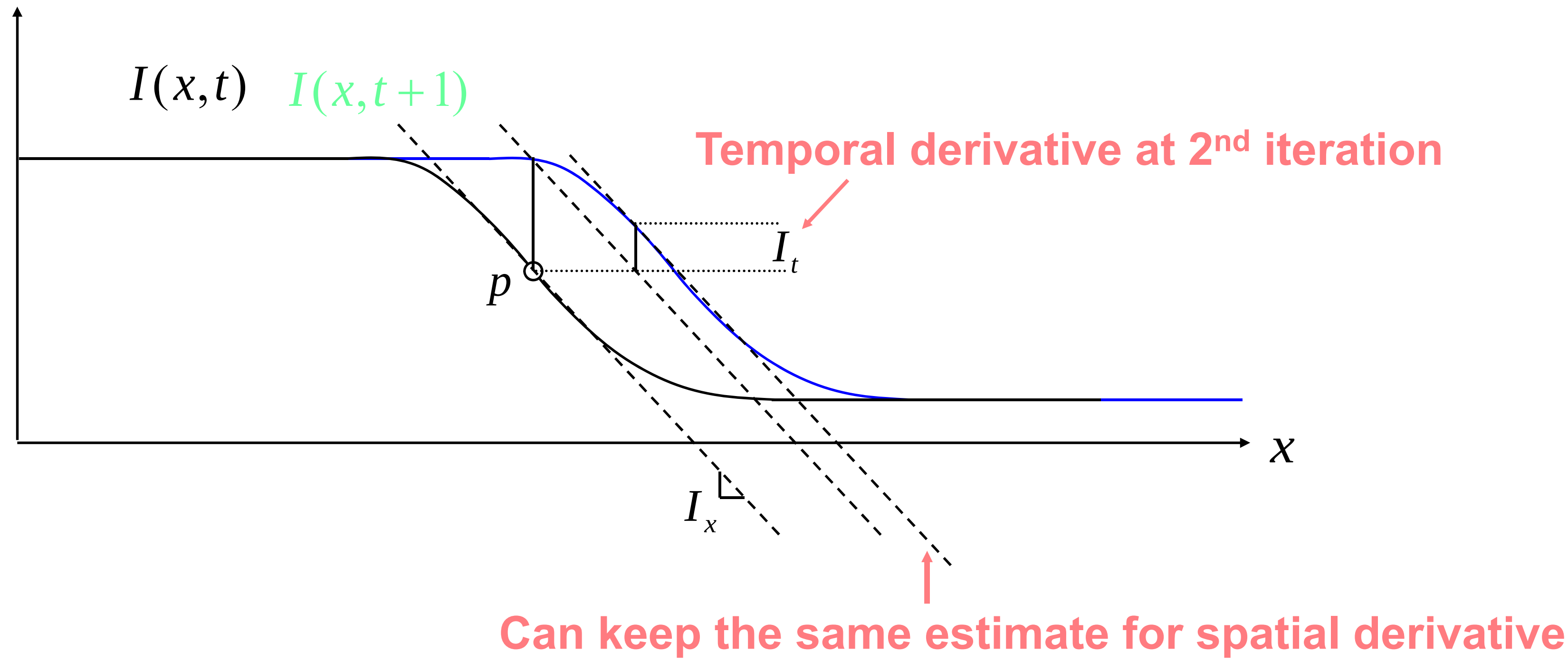
Iterating helps refining the velocity vector



$$\vec{v} \leftarrow \vec{v}_{previous} - \frac{I_t}{I_x}$$

Tracking in the 1D case:

Iterating helps refining the velocity vector



$$\vec{v} \leftarrow \vec{v}_{previous} - \frac{I_t}{I_x}$$

Converges in about 5 iterations

From 1D to 2D tracking

1D: $$\left.\frac{\partial I}{\partial x}\right|_{t}\left(\frac{\partial x}{\partial t}\right)+\left.\frac{\partial I}{\partial t}\right|_{x(t)}=0$$

From 1D to 2D tracking

1D: $$\left.\frac{\partial I}{\partial x}\right|_{t}\left(\frac{\partial x}{\partial t}\right)+\left.\frac{\partial I}{\partial t}\right|_{x(t)}=0$$

From 1D to 2D tracking

$$\text{1D: } \left.\frac{\partial I}{\partial x}\right|_{t} \left(\frac{\partial x}{\partial t}\right) + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

From 1D to 2D tracking

1D: $$\left.\frac{\partial I}{\partial x}\right|_{t}\left(\frac{\partial x}{\partial t}\right)+\left.\frac{\partial I}{\partial t}\right|_{x(t)}=0$$

2D: $$\left.\frac{\partial I}{\partial x}\right|_{t}\left(\frac{\partial x}{\partial t}\right)+\left.\frac{\partial I}{\partial y}\right|_{t}\left(\frac{\partial y}{\partial t}\right)+\left.\frac{\partial I}{\partial t}\right|_{x(t)}=0$$

From 1D to 2D tracking

1D: $$\left.\frac{\partial I}{\partial x}\right|_t \left(\frac{\partial x}{\partial t}\right) + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

2D: $$\left.\frac{\partial I}{\partial x}\right|_t \left(\frac{\partial x}{\partial t}\right) + \left.\frac{\partial I}{\partial y}\right|_t \left(\frac{\partial y}{\partial t}\right) + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

From 1D to 2D tracking

1D: $$\left.\frac{\partial I}{\partial x}\right|_t \left(\frac{\partial x}{\partial t}\right) + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

2D: $$\left.\frac{\partial I}{\partial x}\right|_t \left(\frac{\partial x}{\partial t}\right) + \left.\frac{\partial I}{\partial y}\right|_t \left(\frac{\partial y}{\partial t}\right) + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

$$\left.\frac{\partial I}{\partial x}\right|_t u + \left.\frac{\partial I}{\partial y}\right|_t v + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

From 1D to 2D tracking

1D: $$\left.\frac{\partial I}{\partial x}\right|_{t}\left(\frac{\partial x}{\partial t}\right)+\left.\frac{\partial I}{\partial t}\right|_{x(t)}=0$$

2D: $$\left.\frac{\partial I}{\partial x}\right|_{t}\left(\frac{\partial x}{\partial t}\right)+\left.\frac{\partial I}{\partial y}\right|_{t}\left(\frac{\partial y}{\partial t}\right)+\left.\frac{\partial I}{\partial t}\right|_{x(t)}=0$$

$$\left.\frac{\partial I}{\partial x}\right|_{t}u+\left.\frac{\partial I}{\partial y}\right|_{t}v+\left.\frac{\partial I}{\partial t}\right|_{x(t)}=0$$

Shoot! One equation, two velocity *(u,v)* unknowns…

The aperture problem

$$u = \frac{dx}{dt}, \qquad v = \frac{dy}{dt}$$

$$I_x = \frac{\partial I}{\partial x}, \qquad I_y = \frac{\partial I}{\partial y}, \qquad I_t = \frac{\partial I}{\partial t}$$

$$\boxed{I_x u + I_y v + I_t = 0}$$

Horn and Schunck optical flow equation

The aperture problem

$$u = \frac{dx}{dt}, \qquad v = \frac{dy}{dt}$$

$$I_x = \frac{\partial I}{\partial x}, \qquad I_y = \frac{\partial I}{\partial y}, \qquad I_t = \frac{\partial I}{\partial t}$$

$$\boxed{I_x u + I_y v + I_t = 0}$$

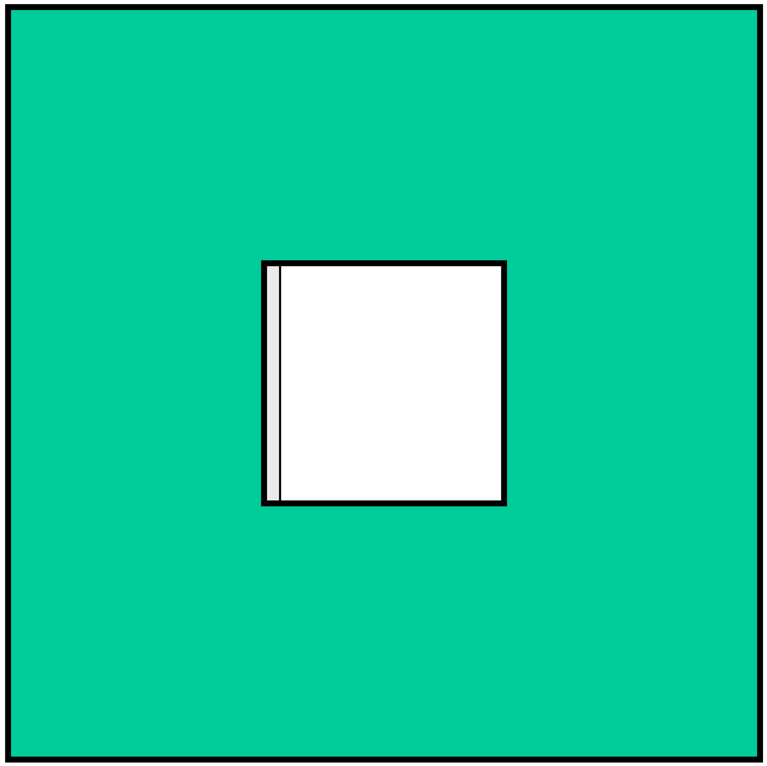
Horn and Schunck optical flow equation

1 equation in 2 unknowns

Optical Flow

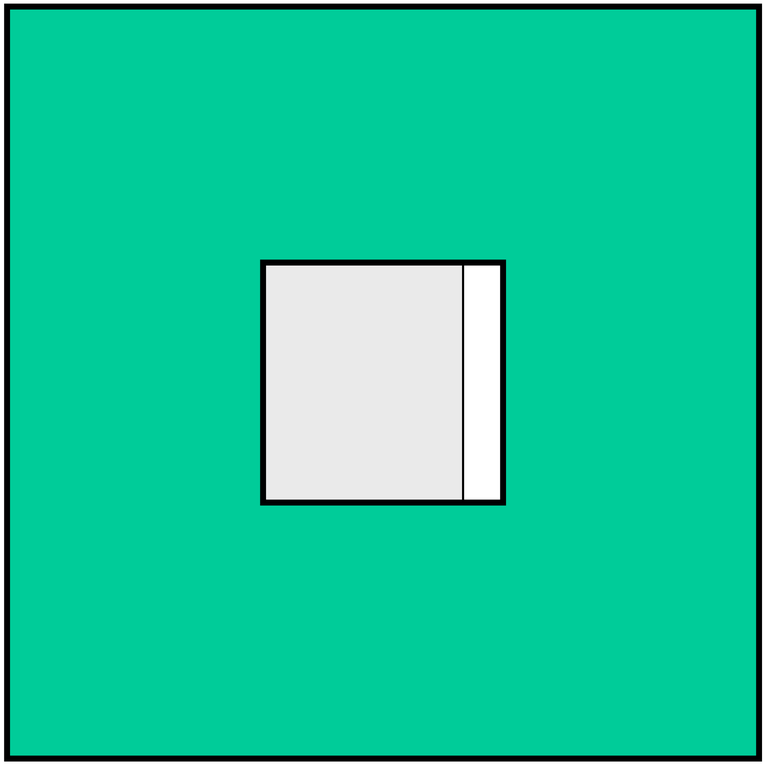
- Brightness Constancy
- The Aperture problem
- Regularization
- Lucas-Kanade
- Coarse-to-fine
- Parametric motion models
- Direct depth
- SSD tracking
- Robust flow
- Bayesian flow

How does this show up visually? Known as the “Aperture Problem”



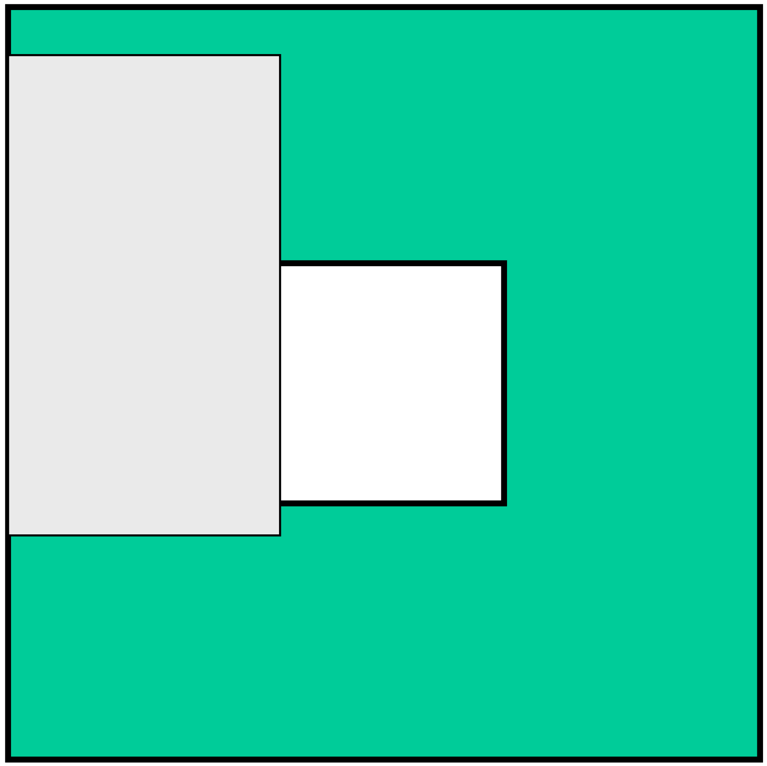
Gary Bradski & Sebastian Thrun, Stanford CS223
http://robots.stanford.edu/cs223b/index.html

How does this show up visually?
Known as the “Aperture Problem”



Gary Bradski & Sebastian Thrun, Stanford CS223
http://robots.stanford.edu/cs223b/index.html

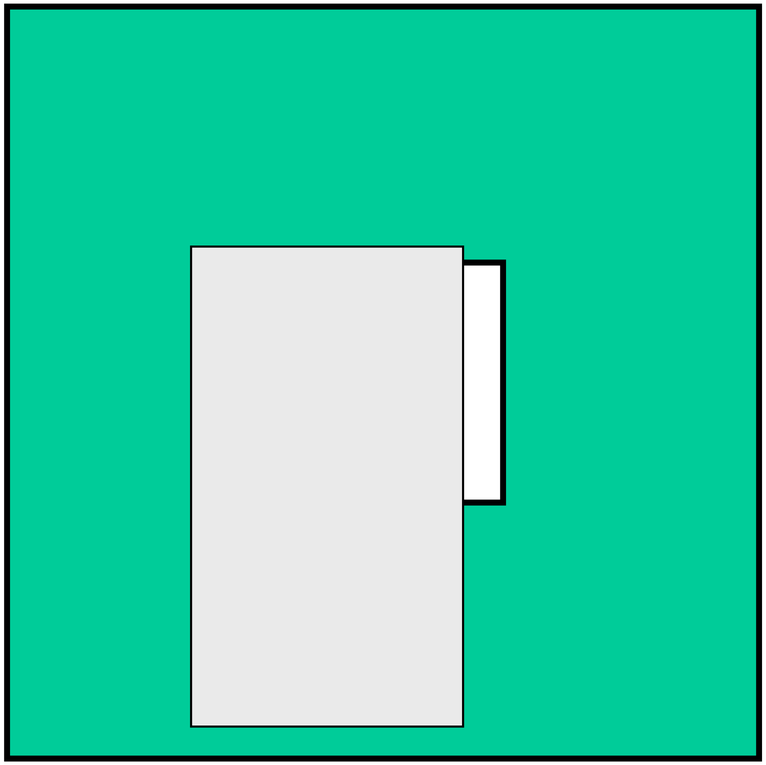
Aperture Problem Exposed



Motion along just an edge is ambiguous

Gary Bradski & Sebastian Thrun, Stanford CS223
http://robots.stanford.edu/cs223b/index.html

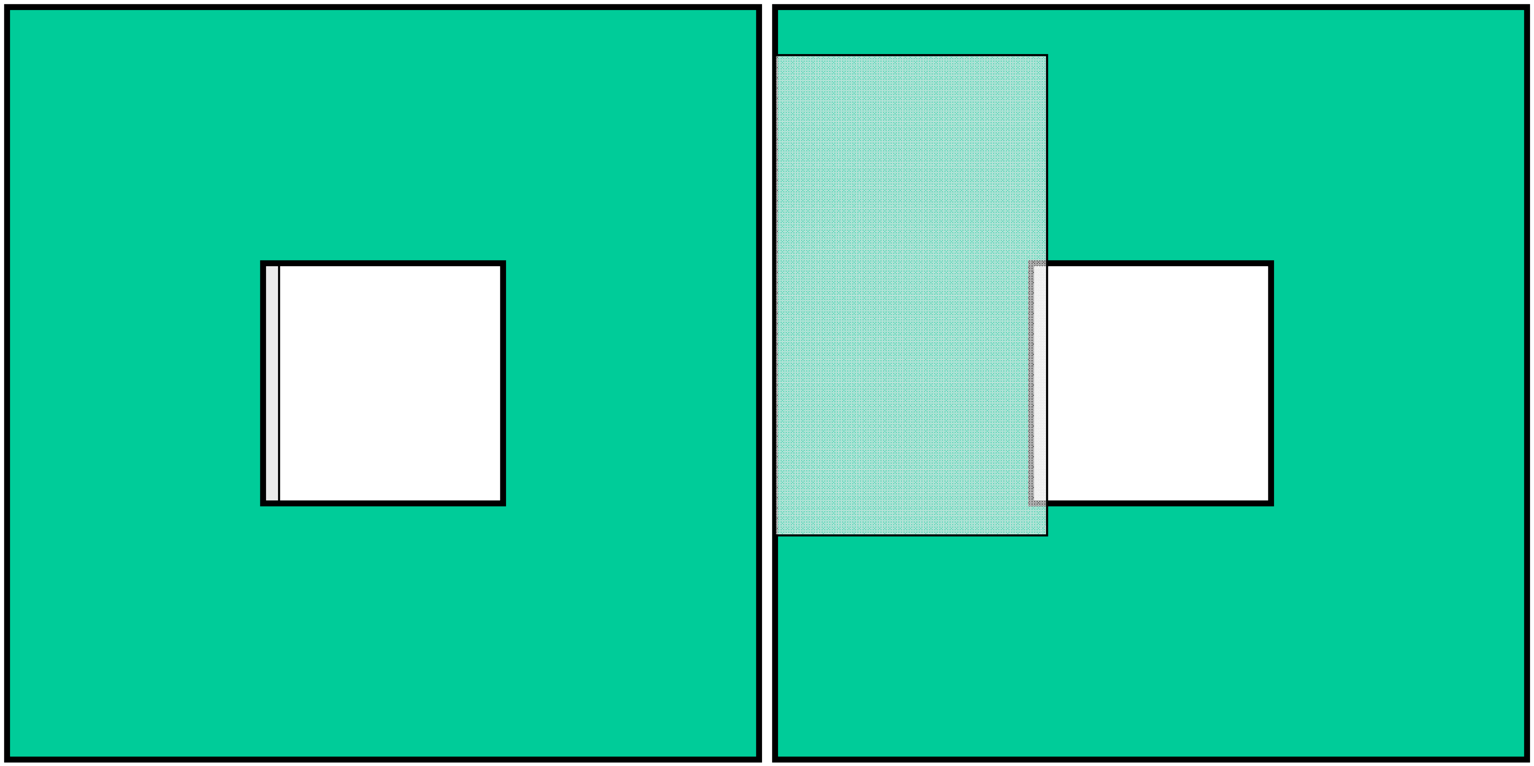
Aperture Problem Exposed



Motion along just an edge is ambiguous

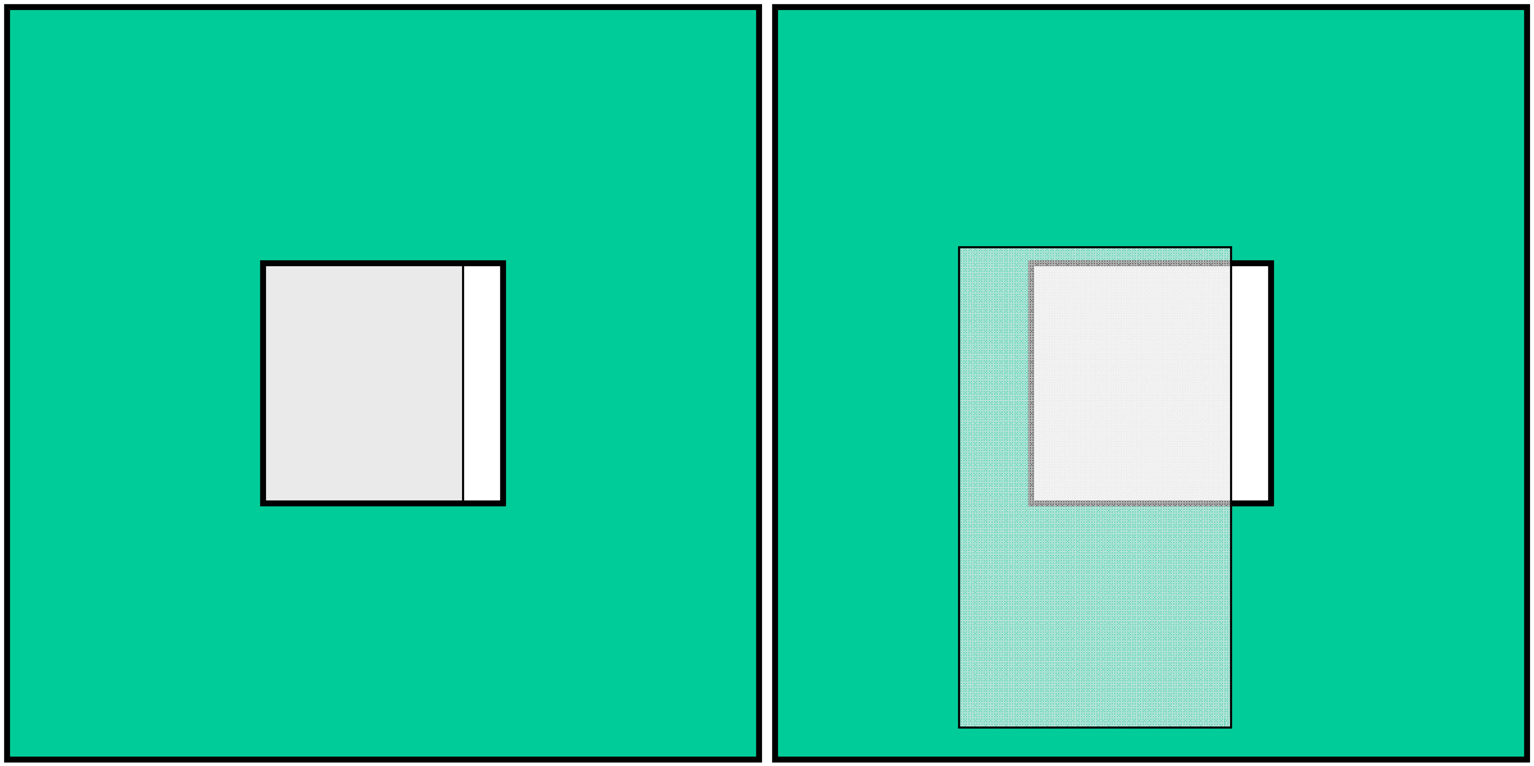
Gary Bradski & Sebastian Thrun, Stanford CS223
http://robots.stanford.edu/cs223b/index.html

How does this show up visually?
Known as the “Aperture Problem”



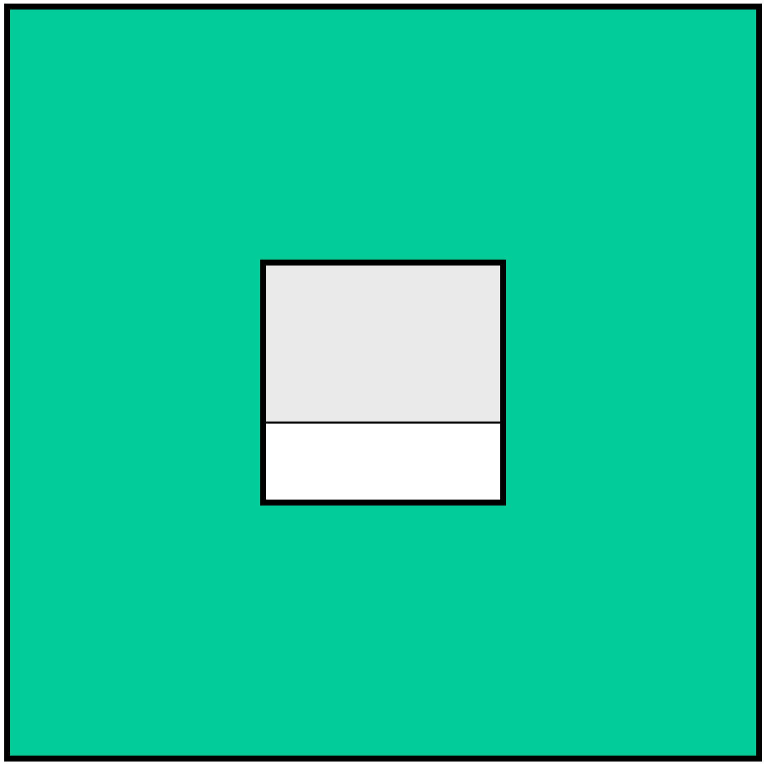
Gary Bradski & Sebastian Thrun, Stanford CS223
http://robots.stanford.edu/cs223b/index.html

How does this show up visually?
Known as the “Aperture Problem”



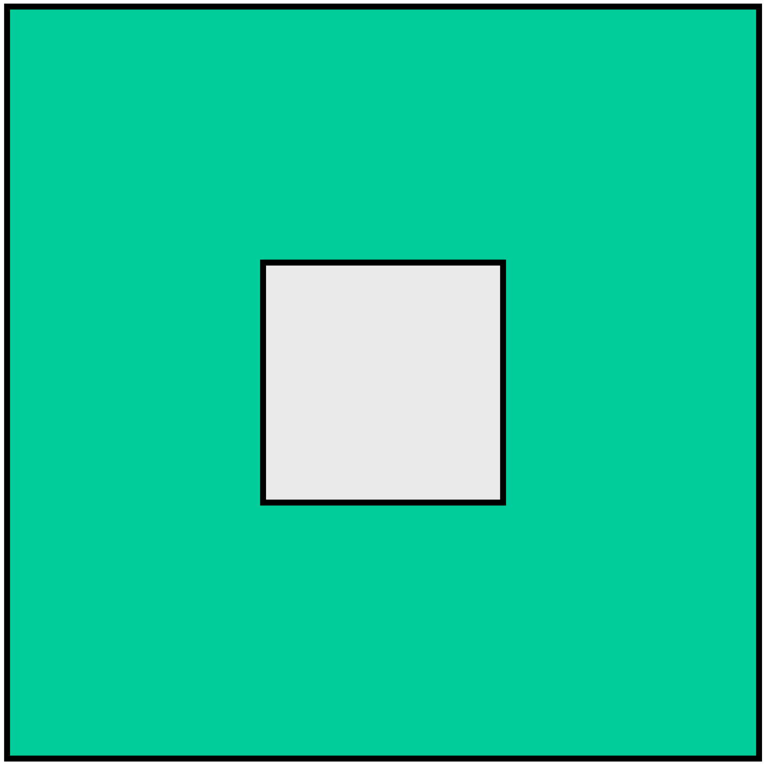
Gary Bradski & Sebastian Thrun, Stanford CS223
http://robots.stanford.edu/cs223b/index.html

How does this show up visually?
Known as the “Aperture Problem”

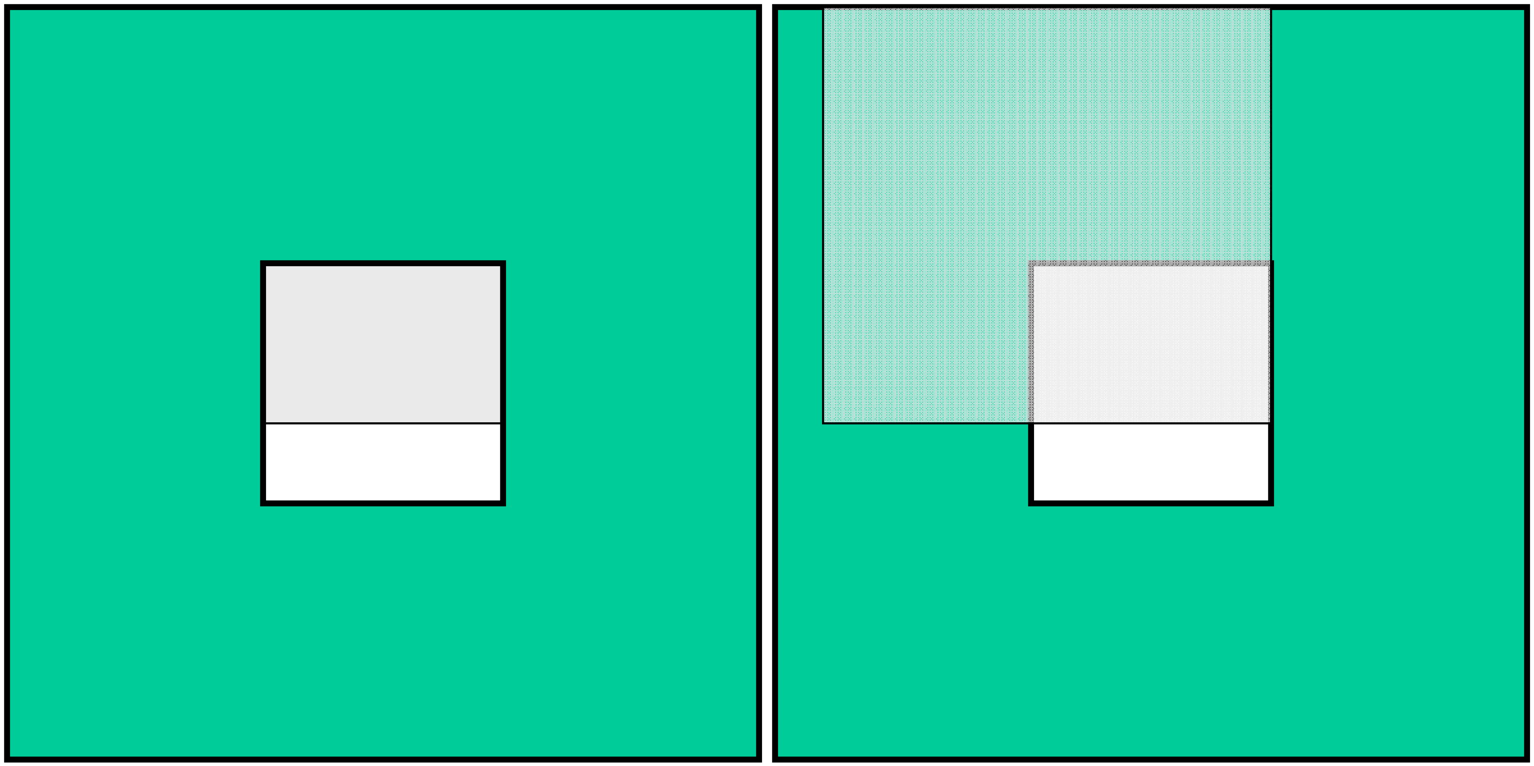


Gary Bradski & Sebastian Thrun, Stanford CS223
http://robots.stanford.edu/cs223b/index.html

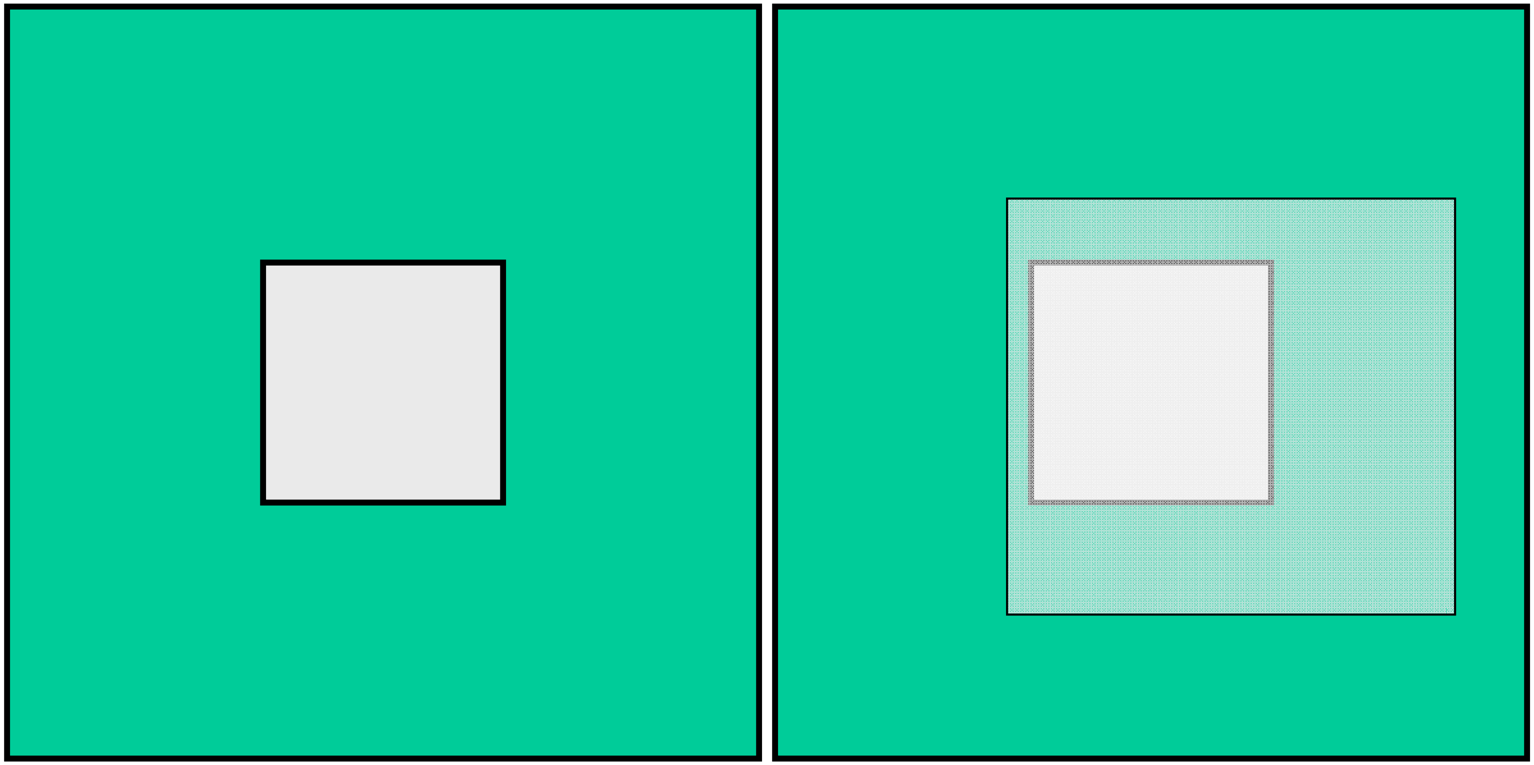
How does this show up visually? Known as the “Aperture Problem”



How does this show up visually?
Known as the “Aperture Problem”



How does this show up visually? Known as the “Aperture Problem”



Optical Flow vs. Motion: Aperture Problem

Barber shop pole:

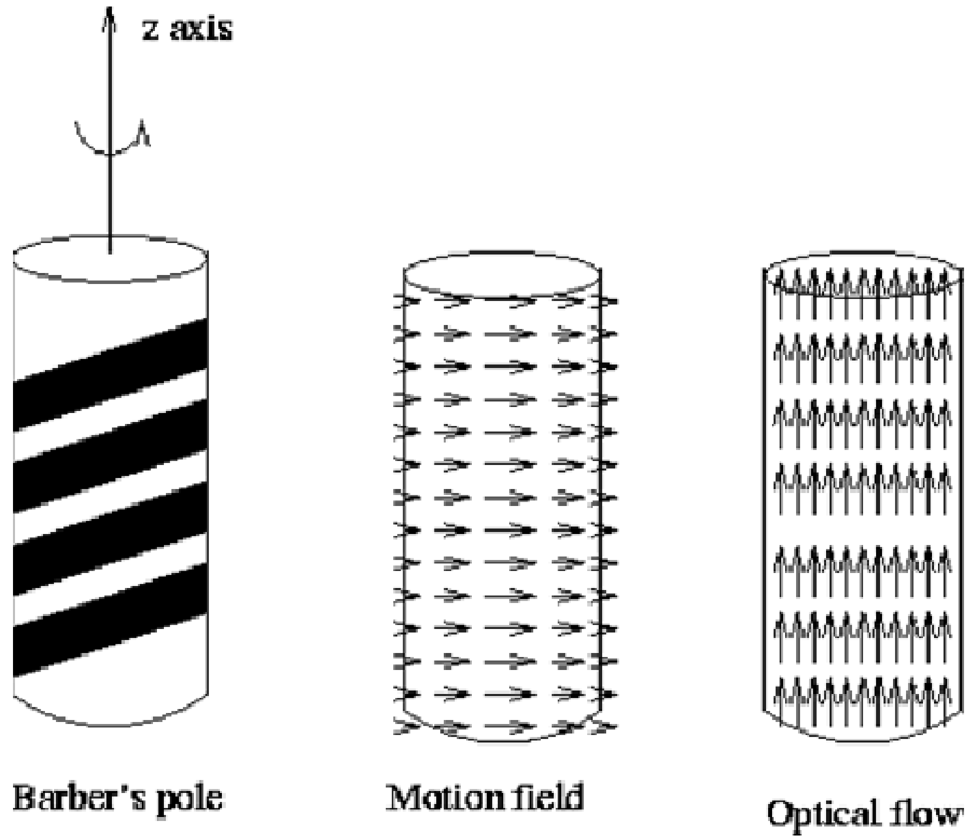
http://www.youtube.com/watch?v=VmqQs613SbE

Optical Flow vs. Motion: Aperture Problem

Barber shop pole:

http://www.youtube.com/watch?v=VmqQs613SbE

Barber pole illusion



Normal Flow
What we can get ☹!!

Notation

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

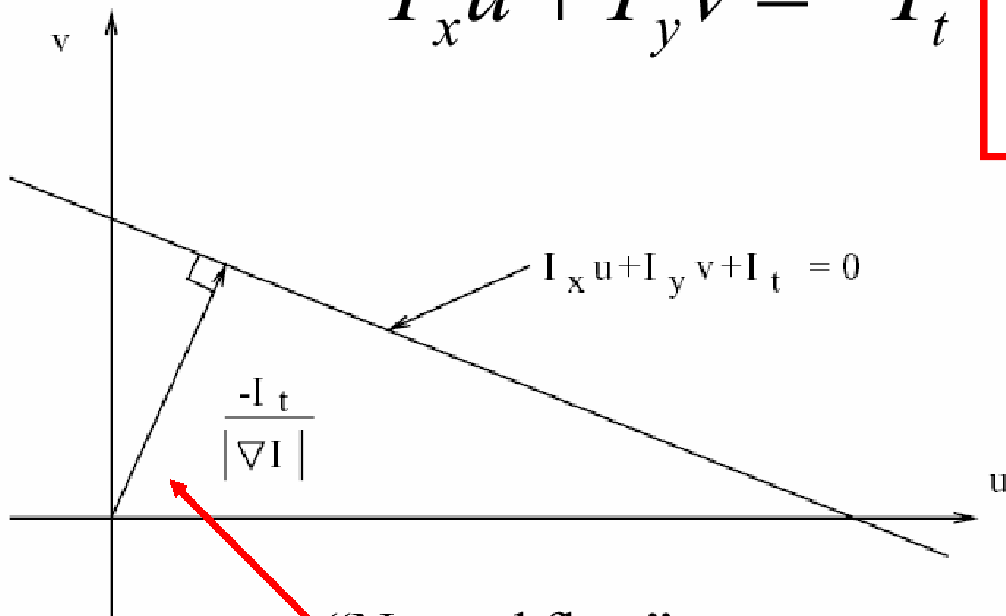
$$\nabla I^T \mathbf{u} = -I_t$$

$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} \quad \nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix}$$

Normal Flow
What we can get ☹!!

At a single image pixel, we get a line:

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



"Normal flow"

Notation

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

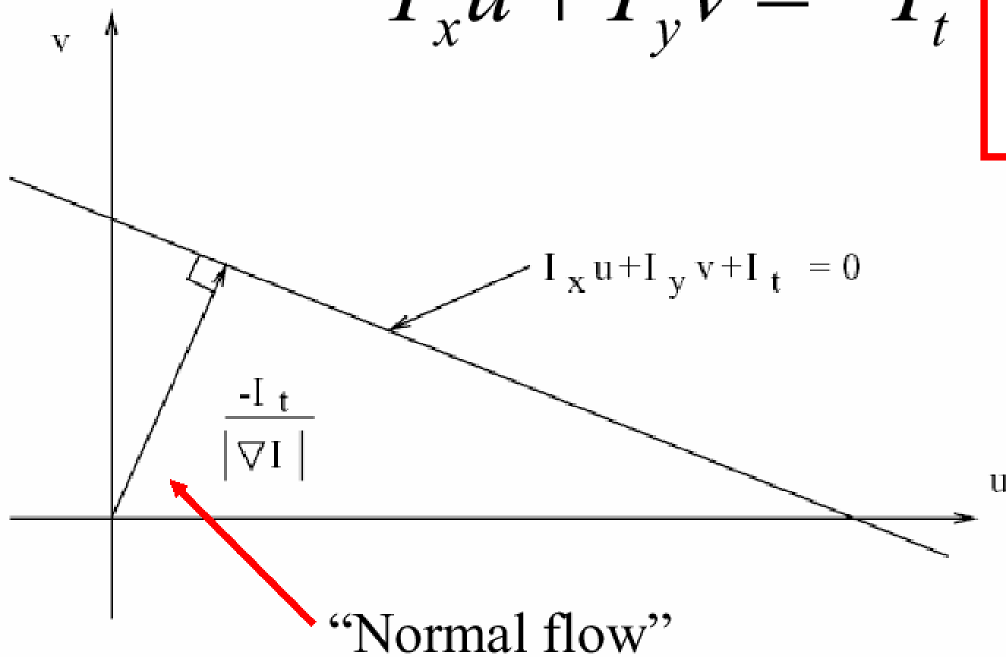
$$\nabla I^T \mathbf{u} = -I_t$$

$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} \quad \nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix}$$

Normal Flow
What we can get ☹!!

At a single image pixel, we get a line:

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



Notation

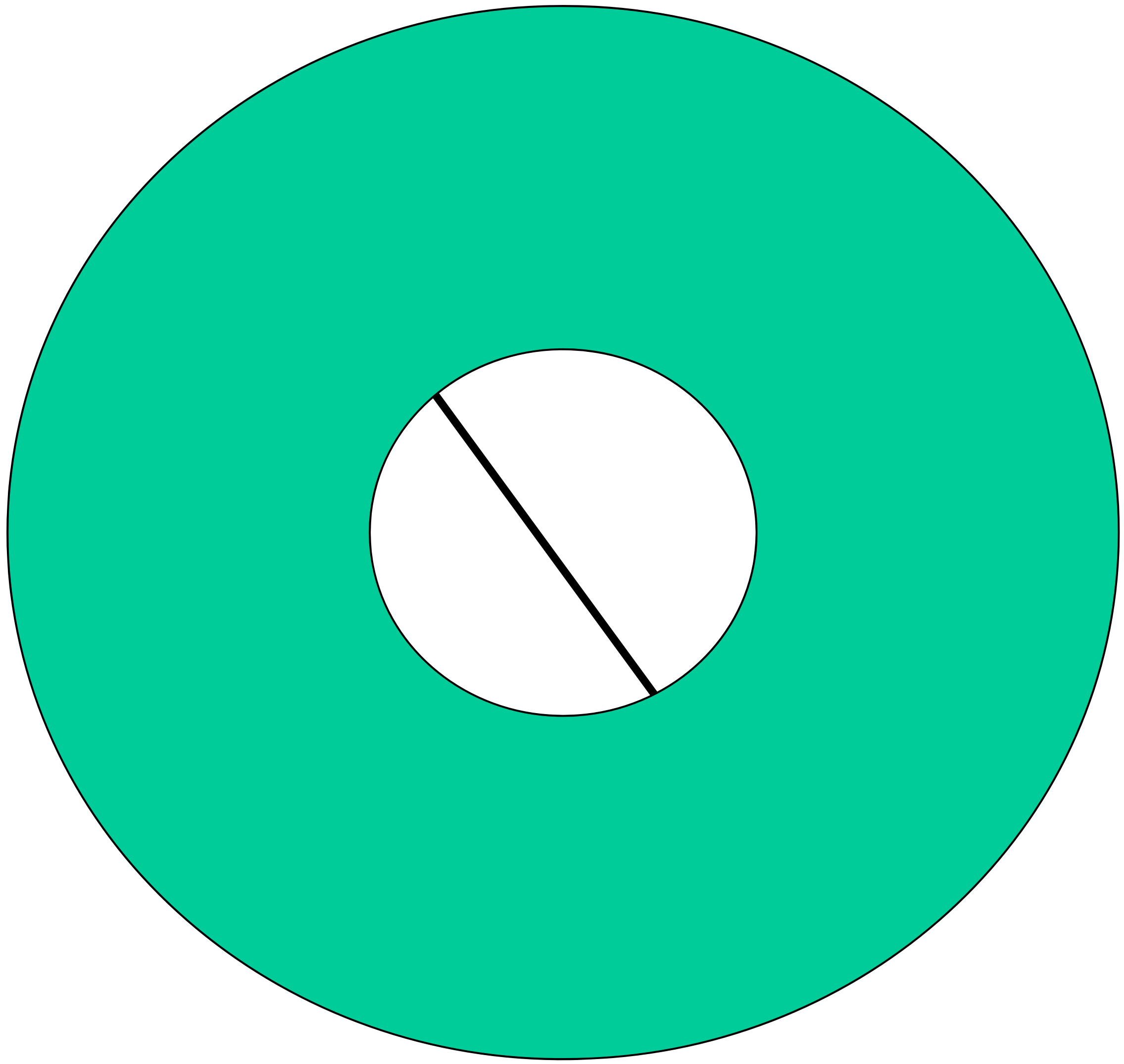
$$I_x u + I_y v + I_t = 0$$

$$\nabla I^T \mathbf{u} = -I_t$$

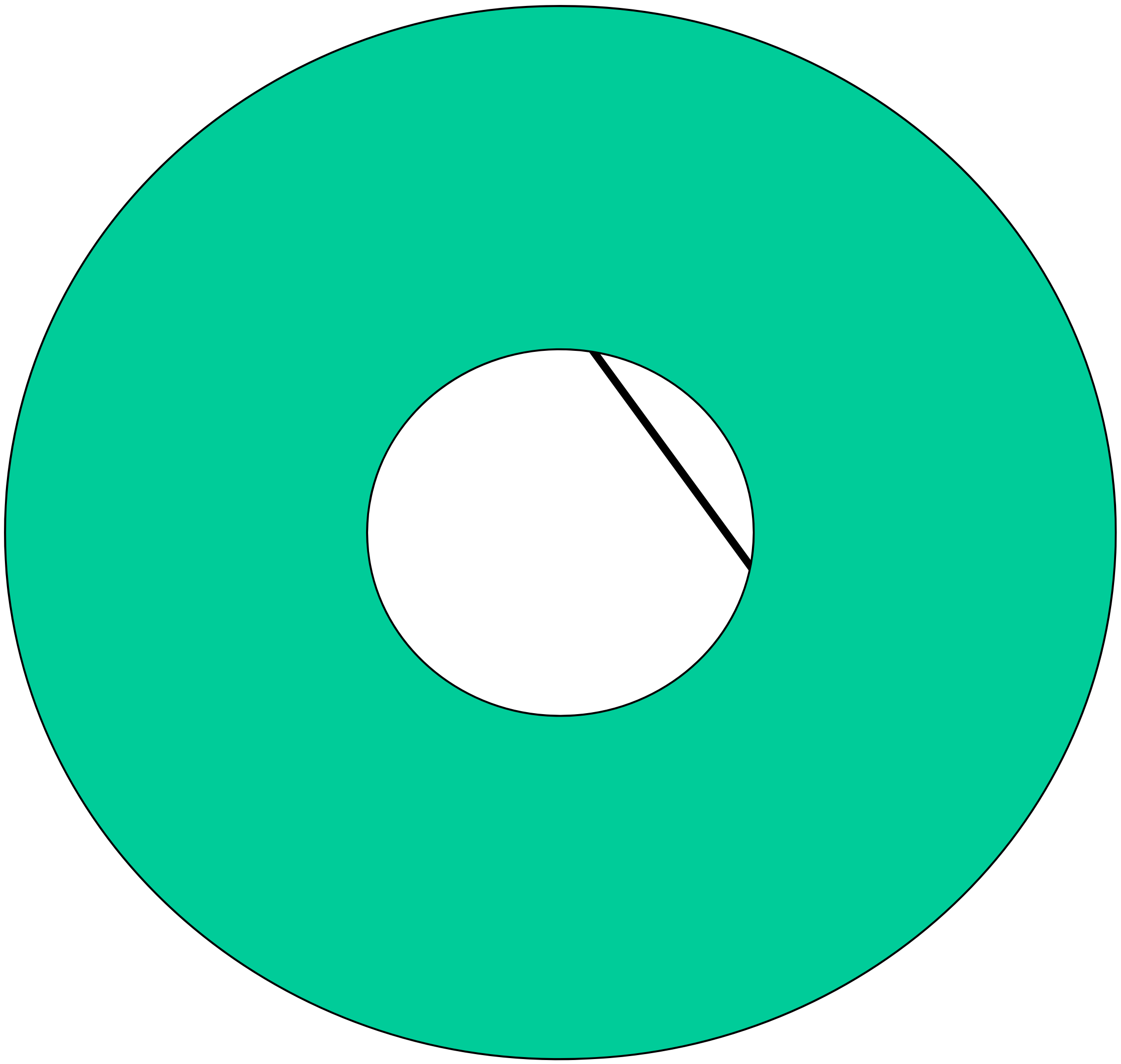
$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} \quad \nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix}$$

We get at most "Normal Flow" – with one point we can only detect movement perpendicular to the brightness gradient. Solution is to take a patch of pixels around the pixel of interest.

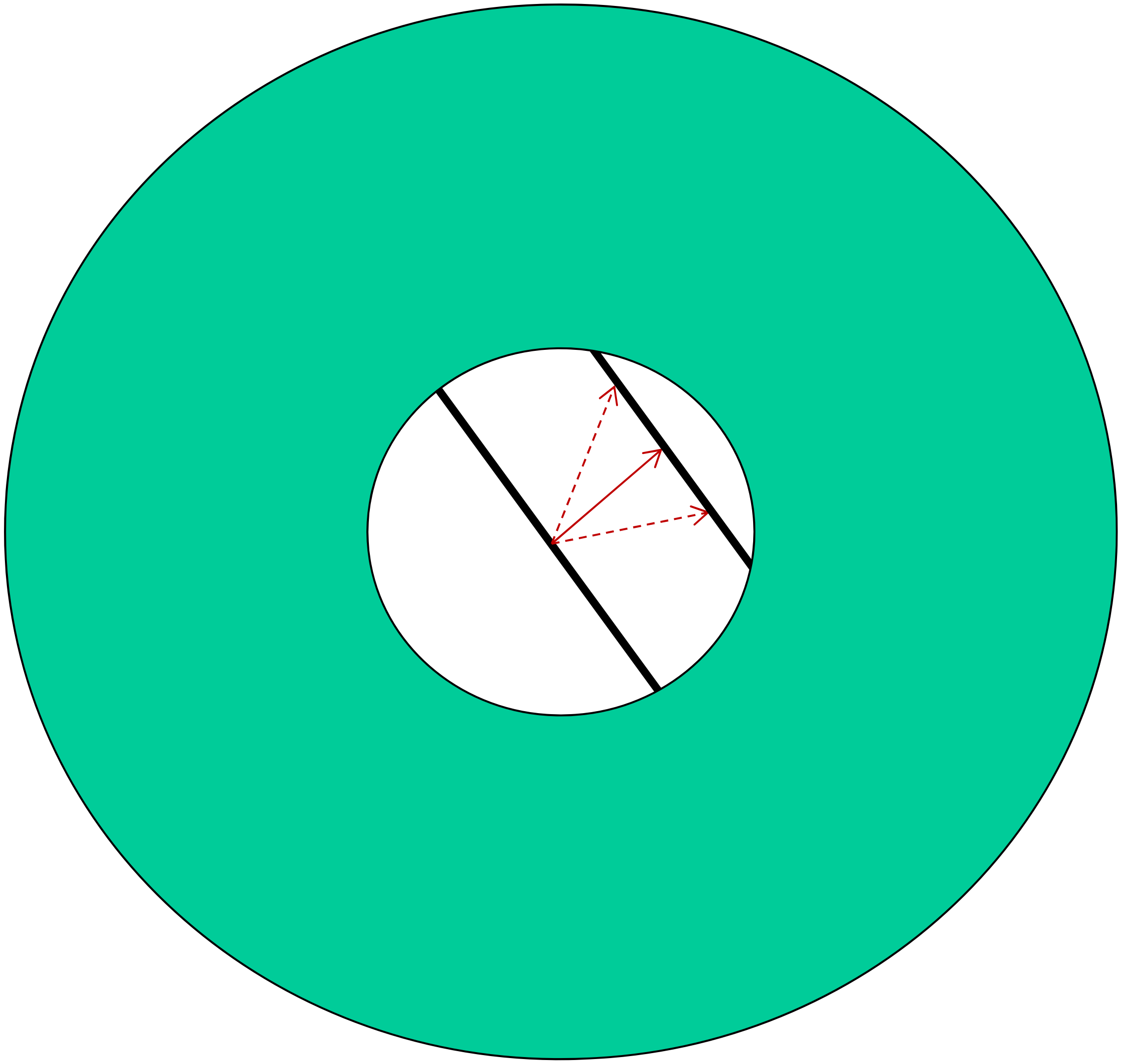
Recall: Aperture Problem



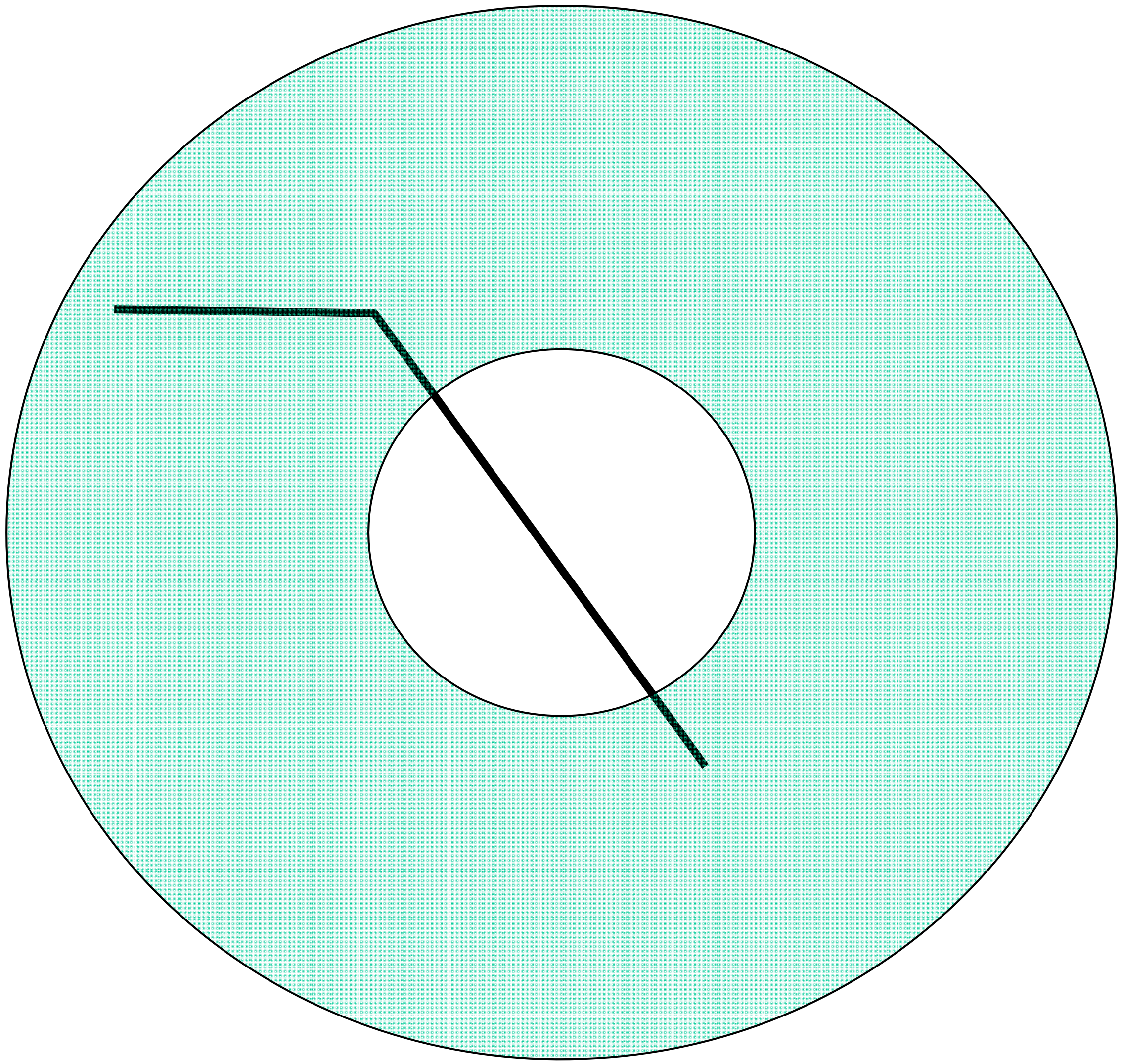
Recall: Aperture Problem



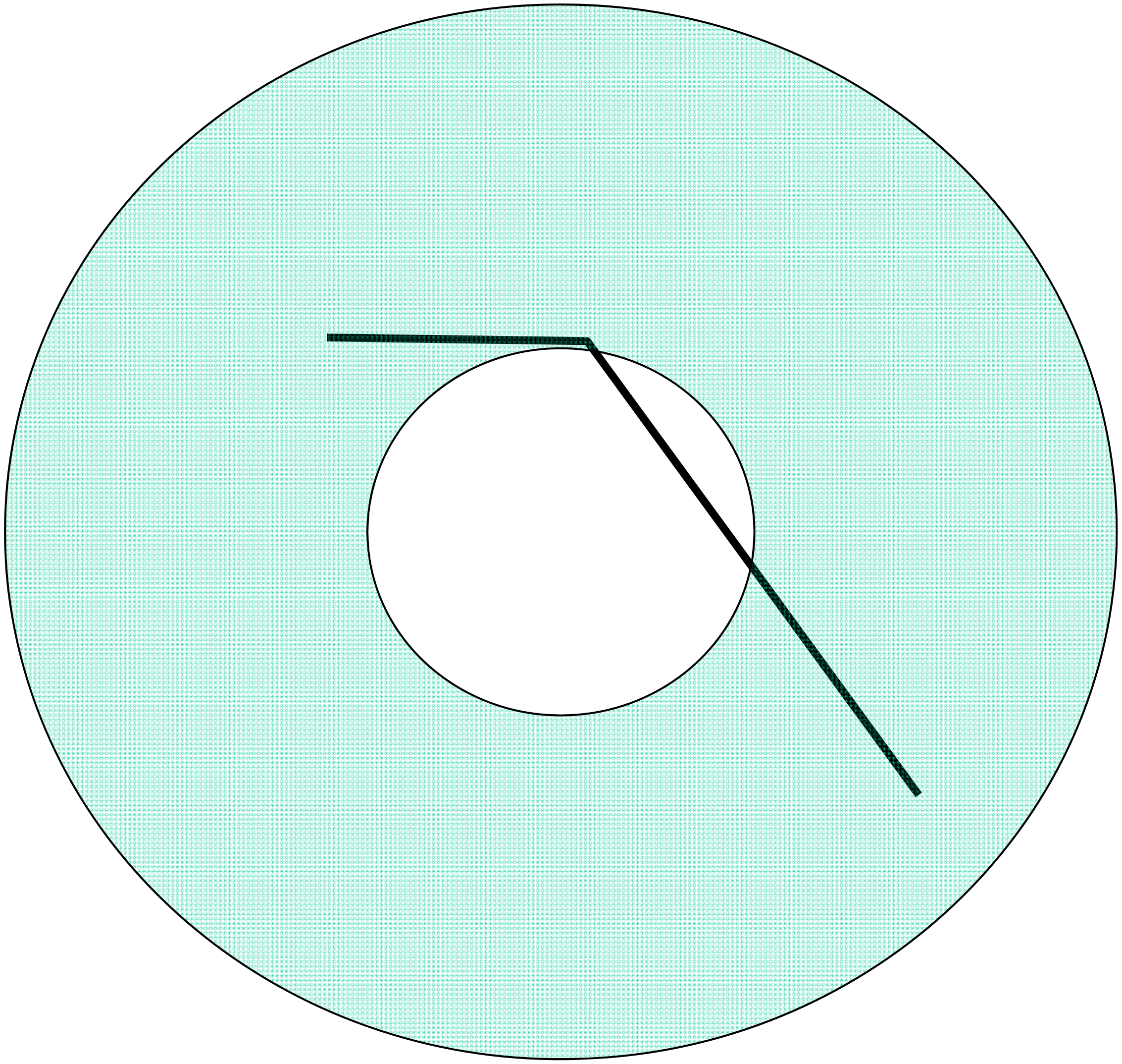
Recall: Aperture Problem



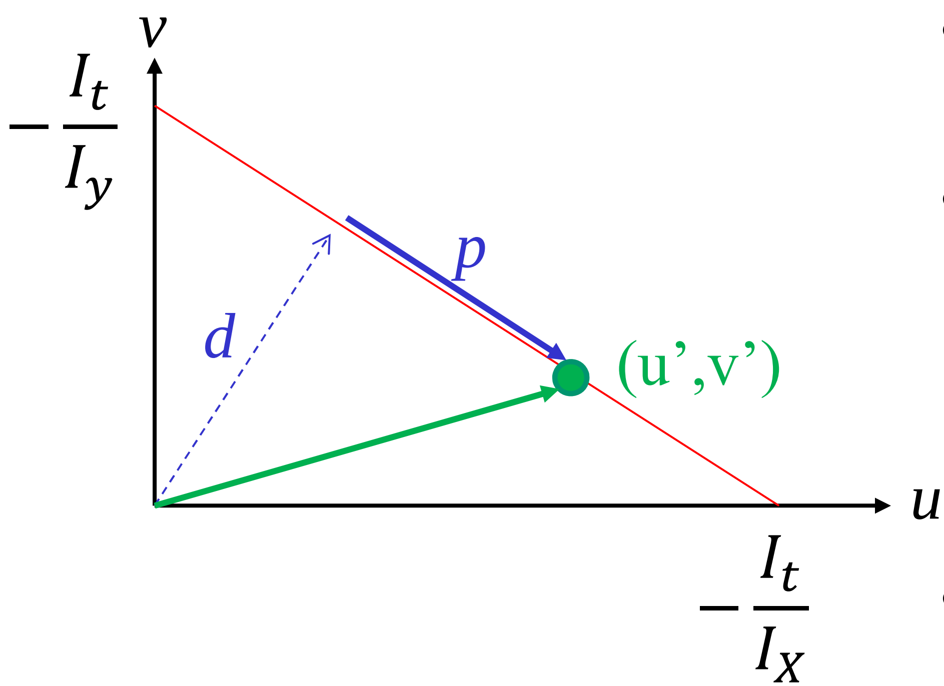
Recall: Aperture Problem



Recall: Aperture Problem

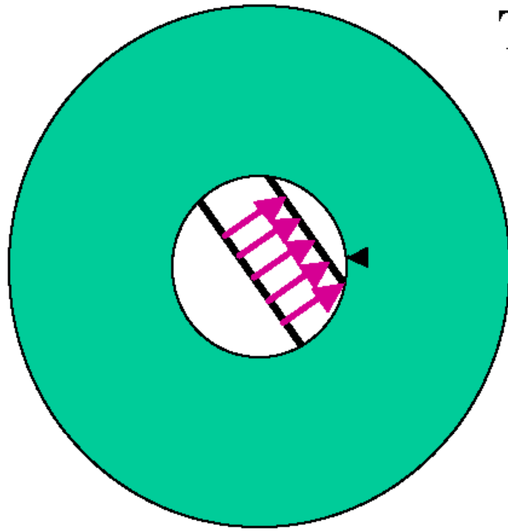


Aperture Problem and Normal Flow



- let (u', v') be true flow
- true flow has two components:
  - Normal flow: d
  - Parallel flow: p
- normal flow **can be** computed
- parallel flow **cannot**

Aperture Problem and Normal Flow

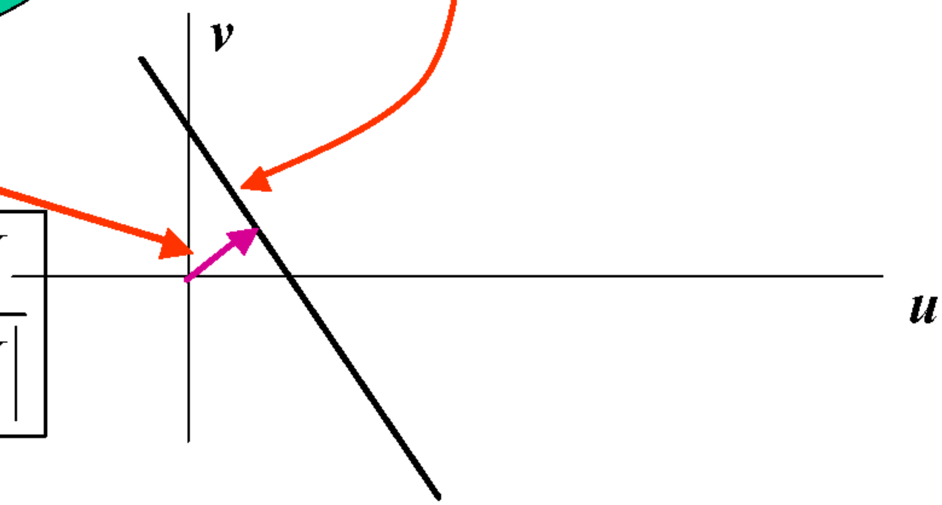


The gradient constraint:

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

$$\nabla I \bullet \vec{U} = 0$$

Defines a line in the *(u,v)* space



Normal Flow:

$$u_\perp = -\frac{I_t}{|\nabla I|}\frac{\nabla I}{|\nabla I|}$$

Computing True Flow

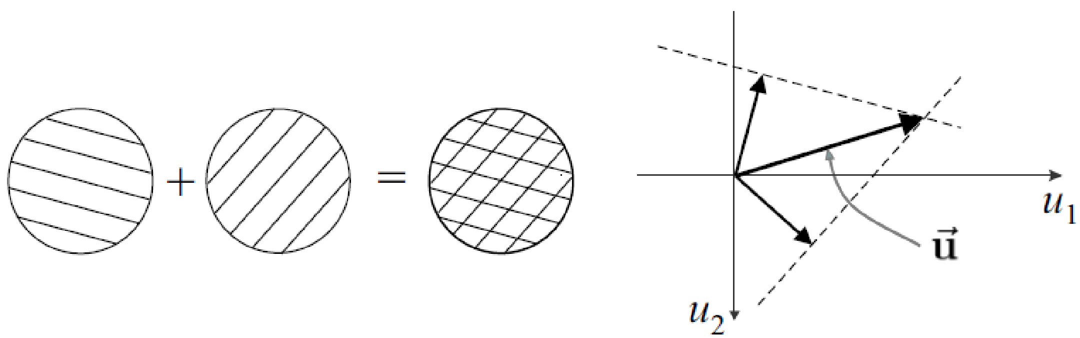
- Schunck
- Horn & Schunck
- Lukas and Kanade

Possible Solution: Neighbors

Two adjacent pixels which are part of the **same rigid object**:

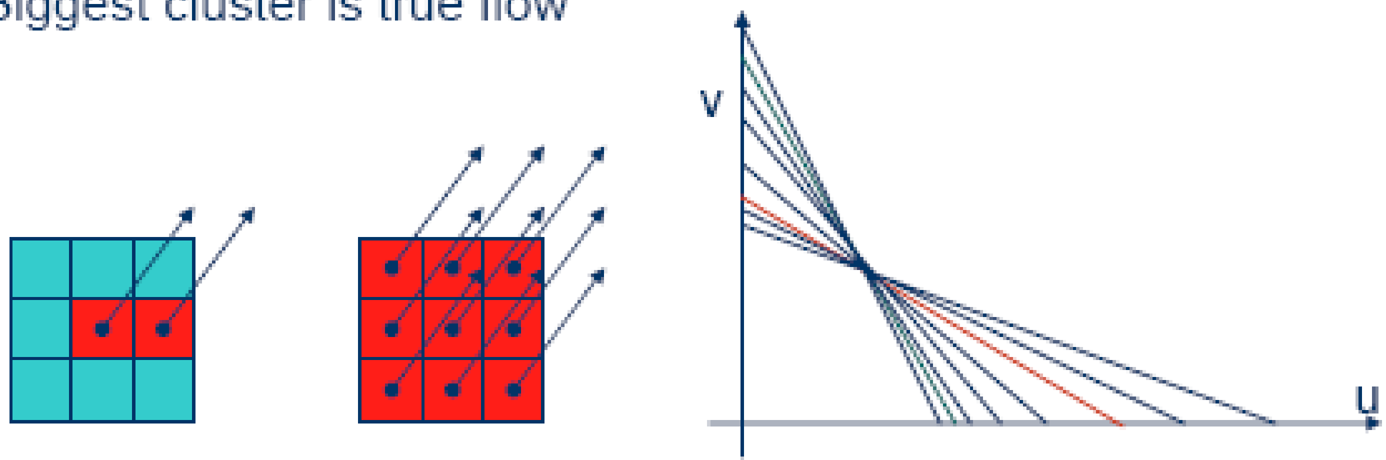
- we can calculate normal flows $\mathbf{v}_{n1}$ and $\mathbf{v}_{n2}$
- Two OF equations for 2 parameters of flow: $\bar{v} = \binom{v}{u}$

$$\nabla I_1 . \bar{v} + I_{t1} = 0$$
$$\nabla I_2 . \bar{v} + I_{t2} = 0$$



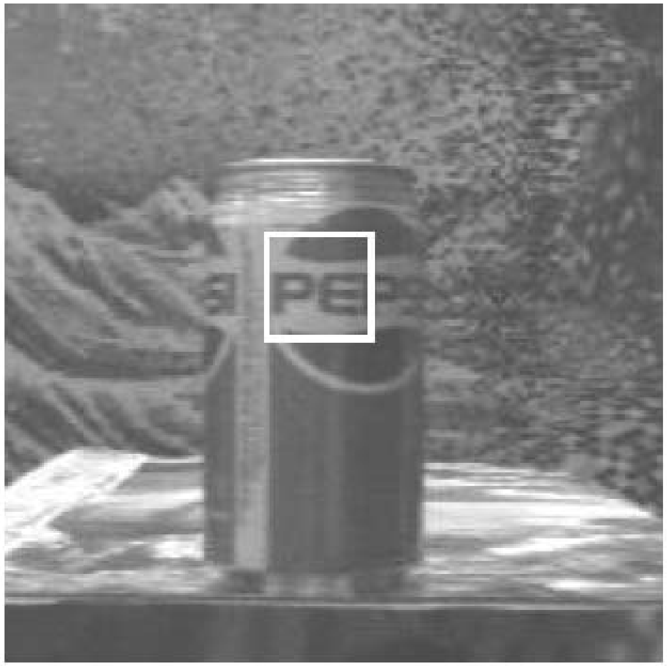
Schunck: Considering Neighbor Pixels

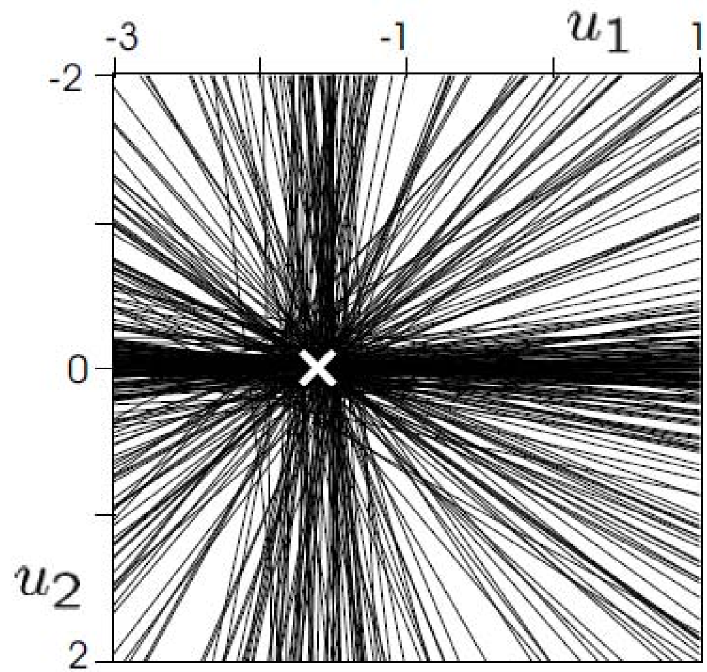
- If two neighboring pixels move with same velocity
  - Corresponding flow equations intersect at a point in (u,v) space
  - Find the intersection point of lines
  - If more than 1 intersection points find clusters
  - Biggest cluster is true flow



Alper Yilmaz, Fall 2005 UCF

Schunck: Considering Neighbor Pixels





Cluster center provides velocity vector common for all pixels in patch.

Jepson, Toronto: http://www.cs.toronto.edu/pub/jepson/teaching/vision/2503/opticalFlow.pdf

Optical Flow

- Brightness Constancy
- The Aperture problem
- Regularization: Horn & Schunck
- Lucas-Kanade
- Coarse-to-fine
- Parametric motion models
- Direct depth
- SSD tracking
- Robust flow
- Bayesian flow

Horn & Schunck algorithm

Horn and Schunck's approach — Regularization

Two terms are defined as follows:

- Departure from smoothness

$$e_s = \int\int_{\Omega}((u_x^2 + u_y^2) + (v_x^2 + v_y^2))dxdy$$

- Error in optical flow constaint equation

$$e_c = \int\int_{\Omega}(E_x u + E_y v + E_t)^2 dxdy$$

The formulation is to minimize the linear combination of e_s and e_c,

$$e_s + \lambda e_c$$

where λ is a parameter.
Note: In this formulation, u and v are functions of x and y. Physically, u is the x-component of the motion, and v is the y-component of the motion.

Horn & Schunck algorithm

Additional smoothness constraint

(usually motion field varies smoothly in the image

→ penalize departure from smoothness) :

$$e_s = \iint ((u_x^2 + u_y^2) + (v_x^2 + v_y^2))\,dxdy,$$

OF constraint equation term

(formulate error in optical flow constraint) :

$$e_c = \iint (I_x u + I_y v + I_t)^2\,dxdy,$$

Horn & Schunck algorithm

Additional smoothness constraint
(usually motion field varies smoothly in the image
→ penalize departure from smoothness) :

$$e_s = \iint ((u_x^2 + u_y^2) + (v_x^2 + v_y^2))dxdy,$$

OF constraint equation term
(formulate error in optical flow constraint) :

$$e_c = \iint (I_x u + I_y v + I_t)^2 dxdy,$$

$\boxed{\text{minimize } e_s + \lambda e_c}$

Horn & Schunck algorithm

Variational calculus: Pair of second order differential equations that can be solved iteratively.

- Define an energy function and minimize

$$E(x,y) = (uI_x + vI_y + I_t)^2 + \lambda\overbrace{(u_x^2 + u_y^2 + v_x^2 + v_y^2)}^{f}$$

- Differentiate w.r.t. unknowns *u* and *v*

$$\frac{\partial E}{\partial u} = 2I_x(uI_x + vI_y + I_t) + \frac{\partial f}{\partial u} \qquad \frac{\partial f}{\partial u} = \frac{\partial}{\partial u}\frac{\partial u}{\partial x} + \frac{\partial}{\partial u}\frac{\partial u}{\partial y} = 2\left(\boxed{u_{xx} + u_{yy}}\right)$$

laplacian of *u*

$$\frac{\partial E}{\partial v} = 2I_y(uI_x + vI_y + I_t) + 2\left(\boxed{v_{xx} + v_{yy}}\right)$$

laplacian of *v*

Horn & Schunck algorithm

$$I_x\left(I_x u + I_y v + I_t\right) + \lambda \Delta u = 0$$

$$I_y\left(I_x u + I_y v + I_t\right) + \lambda \Delta v = 0$$

Approximate Laplacian by weight averaged computed in a neighborhood around the pixel (x,y):

$$\Delta u(x,y) = u(x,y) - \bar{u}(x,y)$$

$$\Delta v(x,y) = v(x,y) - \bar{v}(x,y)$$

Rearranging terms:

$$\begin{aligned} 0 &= I_x\left(I_x u + I_y v + I_t\right) + \lambda(u - \bar{u}) \\ &= u\left(\lambda + I_x^2\right) + v I_x I_y + I_x I_t - \lambda \bar{u} \end{aligned}$$

$$\begin{aligned} 0 &= I_y\left(I_x u + I_y v + I_t\right) + \lambda(v - \bar{v}) \\ &= v\left(\lambda + I_y^2\right) + u I_x I_y + I_y I_t - \lambda \bar{v} \end{aligned}$$

2 equations in 2 unknowns, write v in terms of u and plug it in the other equation

Horn & Schunck algorithm

$$u = \frac{\lambda \bar{u} - v I_x I_y - I_x I_t}{\lambda + I_x^2}$$

$$v = \frac{\lambda \bar{v} - u I_x I_y - I_y I_t}{\lambda + I_y^2}$$

2 equations in 2 unknowns, write v in terms of u and plug it in the other equation

Horn & Schunck algorithm

$$u = \frac{\lambda \bar{u} - v I_x I_y - I_x I_t}{\lambda + I_x^2}$$

$$v = \frac{\lambda \bar{v} - u I_x I_y - I_y I_t}{\lambda + I_y^2}$$

2 equations in 2 unknowns, write v in terms of u and plug it in the other equation

$$u = u_{avg} - I_x \left(\frac{I_x u_{avg} + I_y v_{avg} + I_t}{I_x^2 + I_y^2 + \lambda} \right) \qquad v = v_{avg} - I_y \left(\frac{I_x u_{avg} + I_y v_{avg} + I_t}{I_x^2 + I_y^2 + \lambda} \right)$$

- Iteratively compute u and v
  - Assume initially u and v are 0
  - Compute u_{avg} and v_{avg} in a neighborhood

Horn & Schunck algorithm

The Euler-Lagrange equations :

$$F_u - \frac{\partial}{\partial x} F_{u_x} - \frac{\partial}{\partial y} F_{u_y} = 0$$

$$F_v - \frac{\partial}{\partial x} F_{v_x} - \frac{\partial}{\partial y} F_{v_y} = 0$$

In our case ,

$$F = (u_x^2 + u_y^2) + (v_x^2 + v_y^2) + \lambda (I_x u + I_y v + I_t)^2,$$

Horn & Schunck algorithm

The Euler-Lagrange equations :

$$F_u - \frac{\partial}{\partial x} F_{u_x} - \frac{\partial}{\partial y} F_{u_y} = 0$$

$$F_v - \frac{\partial}{\partial x} F_{v_x} - \frac{\partial}{\partial y} F_{v_y} = 0$$

In our case ,

$$F = (u_x^2 + u_y^2) + (v_x^2 + v_y^2) + \lambda (I_x u + I_y v + I_t)^2,$$

so the Euler-Lagrange equations are

$$\Delta u = \lambda (I_x u + I_y v + I_t) I_x,$$

$$\Delta v = \lambda (I_x u + I_y v + I_t) I_y,$$

$$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$ is the Laplacian operator

Horn & Schunck algorithm

Remarks :

Horn & Schunck algorithm

Remarks :

Horn & Schunck algorithm

Remarks :

1. Coupled PDEs solved using iterative methods and finite differences

$$\frac{\partial u}{\partial t} = \Delta u - \lambda(I_x u + I_y v + I_t) I_x ,$$

$$\frac{\partial v}{\partial t} = \Delta v - \lambda(I_x u + I_y v + I_t) I_y ,$$

Horn & Schunck algorithm

Remarks :

1. Coupled PDEs solved using iterative methods and finite differences

$$\frac{\partial u}{\partial t} = \Delta u - \lambda(I_x u + I_y v + I_t) I_x,$$

$$\frac{\partial v}{\partial t} = \Delta v - \lambda(I_x u + I_y v + I_t) I_y,$$

2. More than two frames allow a better estimation of I_t

Horn & Schunck algorithm

Remarks :

1. Coupled PDEs solved using iterative methods and finite differences

$$\frac{\partial u}{\partial t} = \Delta u - \lambda(I_x u + I_y v + I_t) I_x,$$

$$\frac{\partial v}{\partial t} = \Delta v - \lambda(I_x u + I_y v + I_t) I_y,$$

2. More than two frames allow a better estimation of I_t

Horn & Schunck algorithm

Remarks :

1. Coupled PDEs solved using iterative methods and finite differences

$$\frac{\partial u}{\partial t} = \Delta u - \lambda(I_x u + I_y v + I_t)I_x,$$

$$\frac{\partial v}{\partial t} = \Delta v - \lambda(I_x u + I_y v + I_t)I_y,$$

2. More than two frames allow a better estimation of I_t

Horn & Schunck algorithm

Remarks :

1. Coupled PDEs solved using iterative methods and finite differences

$$\frac{\partial u}{\partial t} = \Delta u - \lambda(I_x u + I_y v + I_t) I_x,$$

$$\frac{\partial v}{\partial t} = \Delta v - \lambda(I_x u + I_y v + I_t) I_y,$$

2. More than two frames allow a better estimation of I_t

3. Information spreads from corner-type patterns

Discrete Optical Flow Algorithm

Consider image pixel (i, j)

- Departure from Smoothness Constraint:

$$s_{ij} = \frac{1}{4}[(u_{i+1,j} - u_{i,j})^2 + (u_{i,j+1} - u_{i,j})^2 + (v_{i+1,j} - v_{i,j})^2 + (v_{i,j+1} - v_{i,j})^2]$$

Discrete Optical Flow Algorithm

Consider image pixel (i, j)

- Departure from Smoothness Constraint:

$$s_{ij} = \frac{1}{4}[(u_{i+1,j} - u_{i,j})^2 + (u_{i,j+1} - u_{i,j})^2 + (v_{i+1,j} - v_{i,j})^2 + (v_{i,j+1} - v_{i,j})^2]$$

- Error in Optical Flow constraint equation:

$$c_{ij} = (I_x^{ij}\, u_{ij} + I_y^{ij}\, v_{ij} + I_t^{ij})^2$$

Discrete Optical Flow Algorithm

Consider image pixel (i, j)

- Departure from Smoothness Constraint:

$$s_{ij} = \frac{1}{4}[(u_{i+1,j} - u_{i,j})^2 + (u_{i,j+1} - u_{i,j})^2 + (v_{i+1,j} - v_{i,j})^2 + (v_{i,j+1} - v_{i,j})^2]$$

- Error in Optical Flow constraint equation:

$$c_{ij} = (I_x^{ij}\, u_{ij} + I_y^{ij}\, v_{ij} + I_t^{ij})^2$$

- We seek the set $\{u_{ij}\}$ & $\{v_{ij}\}$ that minimize:

$$e = \sum_i \sum_j (s_{ij} + \lambda\, c_{ij})$$

NOTE: $\{u_{ij}\}$ & $\{v_{ij}\}$ show up in more than one term

Discrete Optical Flow Algorithm

- Differentiating e w.r.t v_{kl} & u_{kl} and setting to zero:

$$\frac{\partial e}{\partial u_{kl}} = 2\,(u_{kl} - \overline{u_{kl}}) + 2\lambda\,(I_x^{kl} u_{kl} + I_y^{kl} v_{kl} + I_t^{kl}) I_x^{kl} = 0$$

$$\frac{\partial e}{\partial v_{kl}} = 2\,(v_{kl} - \overline{v_{kl}}) + 2\lambda\,(I_x^{kl} u_{kl} + I_y^{kl} v_{kl} + I_t^{kl}) I_y^{kl} = 0$$

- $\overline{v_{kl}}$ & $\overline{u_{kl}}$ are averages of (u, v) around pixel (k, l)

Discrete Optical Flow Algorithm

- Differentiating e w.r.t v_{kl} & u_{kl} and setting to zero:

$$\frac{\partial e}{\partial u_{kl}} = 2\,(u_{kl} - \overline{u_{kl}}) + 2\lambda\,(I_x^{kl} u_{kl} + I_y^{kl} v_{kl} + I_t^{kl}) I_x^{kl} = 0$$

$$\frac{\partial e}{\partial v_{kl}} = 2\,(v_{kl} - \overline{v_{kl}}) + 2\lambda\,(I_x^{kl} u_{kl} + I_y^{kl} v_{kl} + I_t^{kl}) I_y^{kl} = 0$$

- $\overline{v_{kl}}$ & $\overline{u_{kl}}$ are averages of (u, v) around pixel (k, l)

Discrete Optical Flow Algorithm

- Differentiating e w.r.t v_{kl} & u_{kl} and setting to zero:

$$\frac{\partial e}{\partial u_{kl}} = 2\,(u_{kl} - \overline{u_{kl}}) + 2\lambda\,(I_x^{kl} u_{kl} + I_y^{kl} v_{kl} + I_t^{kl}) I_x^{kl} = 0$$

$$\frac{\partial e}{\partial v_{kl}} = 2\,(v_{kl} - \overline{v_{kl}}) + 2\lambda\,(I_x^{kl} u_{kl} + I_y^{kl} v_{kl} + I_t^{kl}) I_y^{kl} = 0$$

- $\overline{v_{kl}}$ & $\overline{u_{kl}}$ are averages of (u, v) around pixel (k, l)

Update Rule:

$$u_{kl}^{n+1} = \overline{u_{kl}^{n}} - \frac{I_x^{kl}\,\overline{u_{kl}^{n}} + I_y^{kl}\,\overline{v_{kl}^{n}} + I_t^{kl}}{1 + \lambda\,[(I_x^{kl})^2 + (I_y^{kl})^2]} I_x^{kl}$$

$$v_{kl}^{n+1} = \overline{v_{kl}^{n}} - \frac{I_x^{kl}\,\overline{u_{kl}^{n}} + I_y^{kl}\,\overline{v_{kl}^{n}} + I_t^{kl}}{1 + \lambda\,[(I_x^{kl})^2 + (I_y^{kl})^2]} I_y^{kl}$$

Horn-Schunck Algorithm : Discrete Case

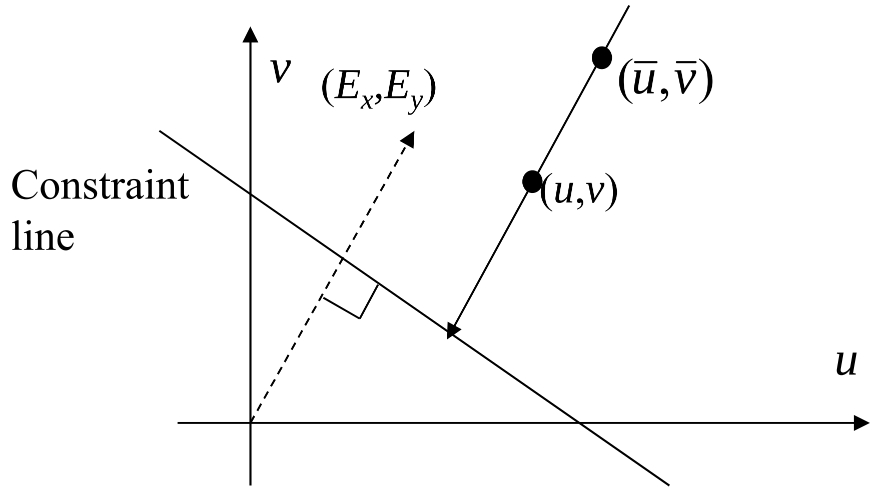
- Derivatives (and error functionals) are approximated by difference operators
- Leads to an iterative solution:

$$u_{ij}^{n+1} = \overline{u}_{ij}^{n} - \alpha I_x \qquad \alpha = \frac{I_x \overline{u}_{ij}^{n} + I_y \overline{v}_{ij}^{n} + I_t}{1 + \lambda (I_x^2 + I_y^2)}$$

$$v_{ij}^{n+1} = \overline{v}_{ij}^{n} - \alpha I_y$$

$\overline{u}, \overline{v}$ are the averages of values of neighbors

Intuition of the Iterative Scheme



The new value of (u,v) at a point is equal to the average of surrounding values minus an adjustment in the direction of the brightness gradient

Horn - Schunck Algorithm

begin
    for $j := 1$ **to** N **do** **for** $i := 1$ **to** M **do** **begin**
        calculate the values $E_x(i,j,t)$, $E_y(i,j,t)$, and $E_t(i,j,t)$ using
            a selected approximation formula;
                { special cases for image points at the image border have to be taken into account }
        initialize the values $u(i,j)$ and $v(i,j)$ with zero
    end {for};
    choose a suitable weighting value λ; { e.g $\lambda = 10$ }
    choose a suitable number $n_0 \geq 1$ of iterations; { $n_0 = 8$ }
    $n := 1$; { iteration counter }
    while $n \leq n_0$ **do begin**
        for $j := 1$ **to** N **do** **for** $i := 1$ **to** M **do** **begin**

$$\bar{u} := \tfrac{1}{4}\left(u(i-1,j) + u(i+1,j) + u(i,j-1) + u(i,j+1)\right);$$

$$\bar{v} := \tfrac{1}{4}\left(v(i-1,j) + v(i+1,j) + v(i,j-1) + v(i,j+1)\right);$$

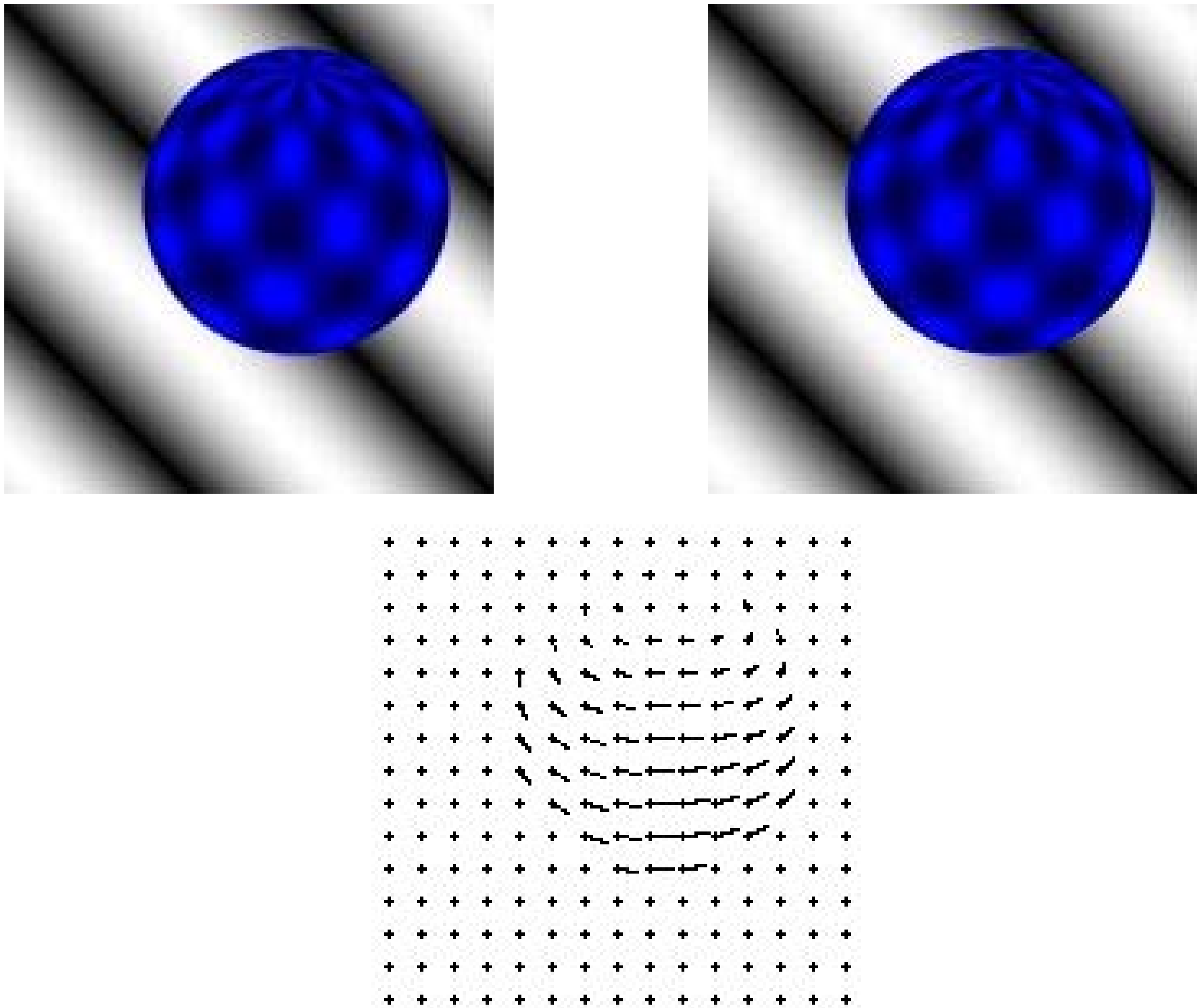
                { treat image points at the image border separately }

$$\alpha := \frac{E_x(i,j,t)\bar{u} + E_y(i,j,t)\bar{v} + E_t(i,j,t)}{1 + \lambda\left(E_x^2(i,j,t) + E_y^2(i,j,t)\right)} \cdot \lambda \;;$$

$$u(i,j) := \bar{u} - \alpha \cdot E_x(i,j,t) \;; \quad v(i,j) := \bar{v} - \alpha \cdot E_y(i,j,t)$$

        end {for};
        $n := n + 1$
    end {while}
end;

Example



http://of-eval.sourceforge.net/

Results

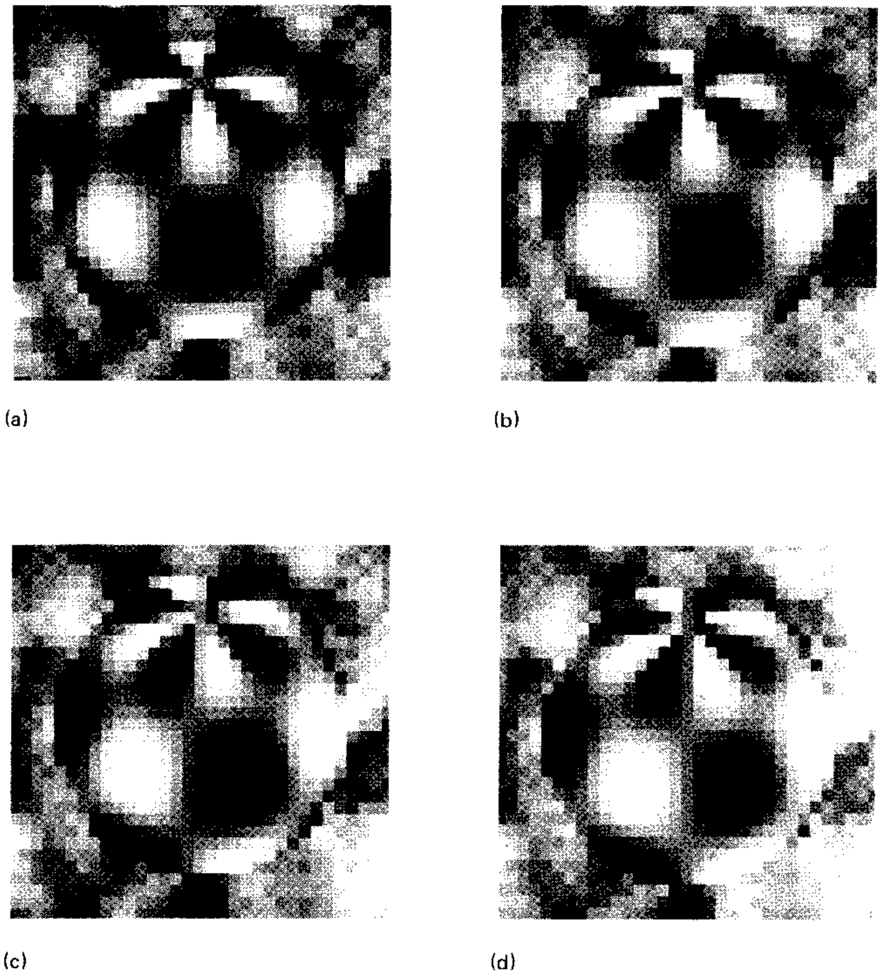


Figure 12-8. Four frames of a synthetic image sequence showing a sphere slowly rotating in front of a randomly patterned background.

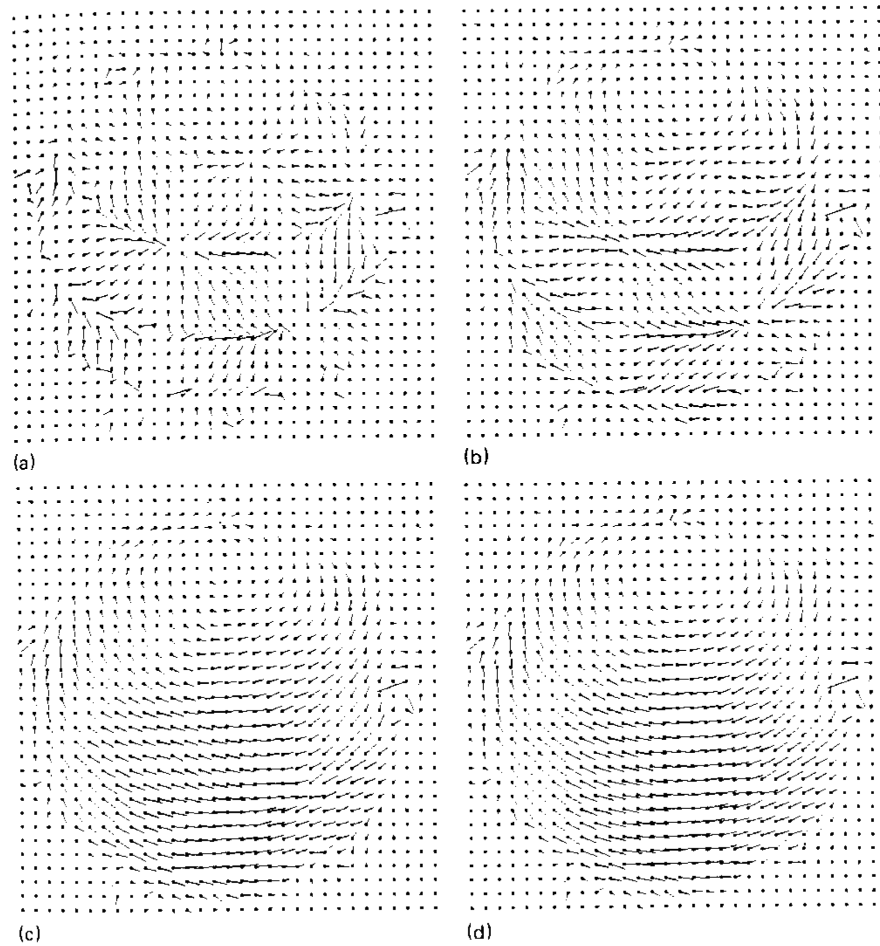
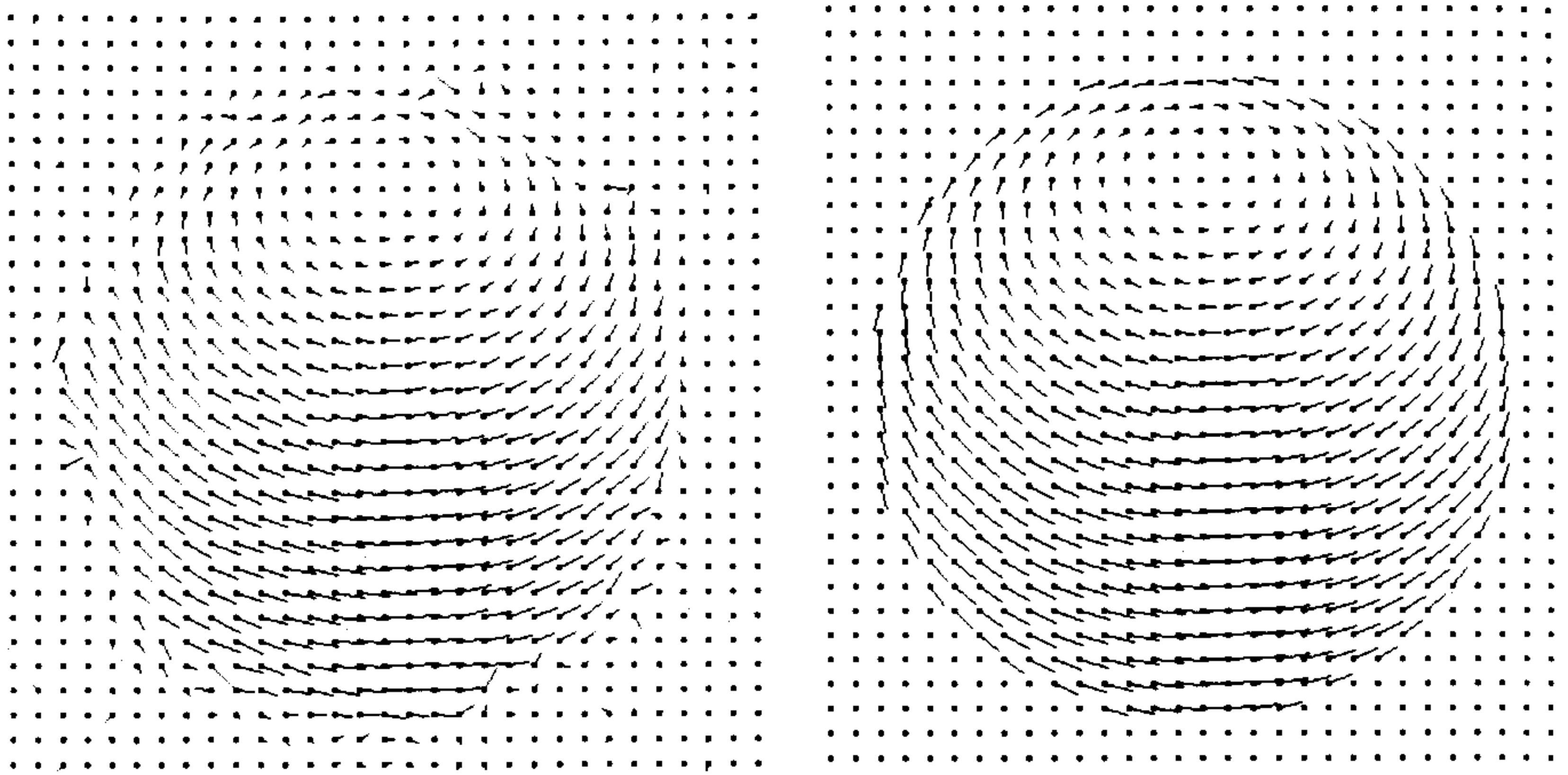


Figure 12-9. Estimates of the optical flow shown in the form of needle diagrams after 1, 4, 16, and 64 iterations of the algorithm.

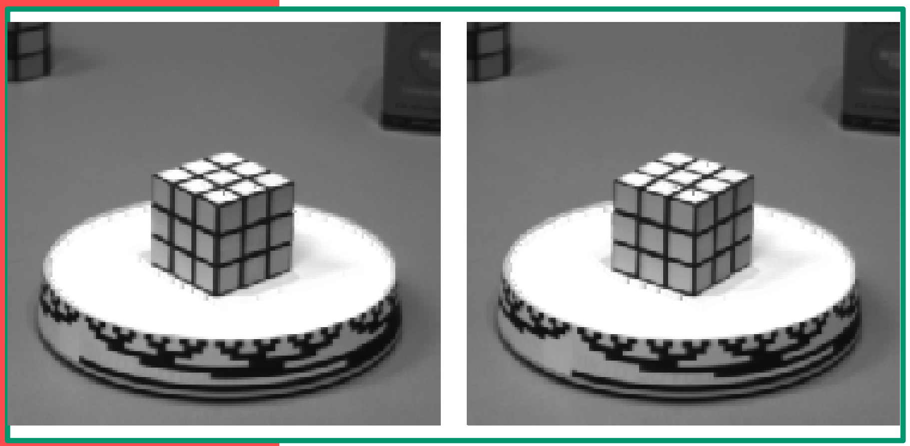
Results

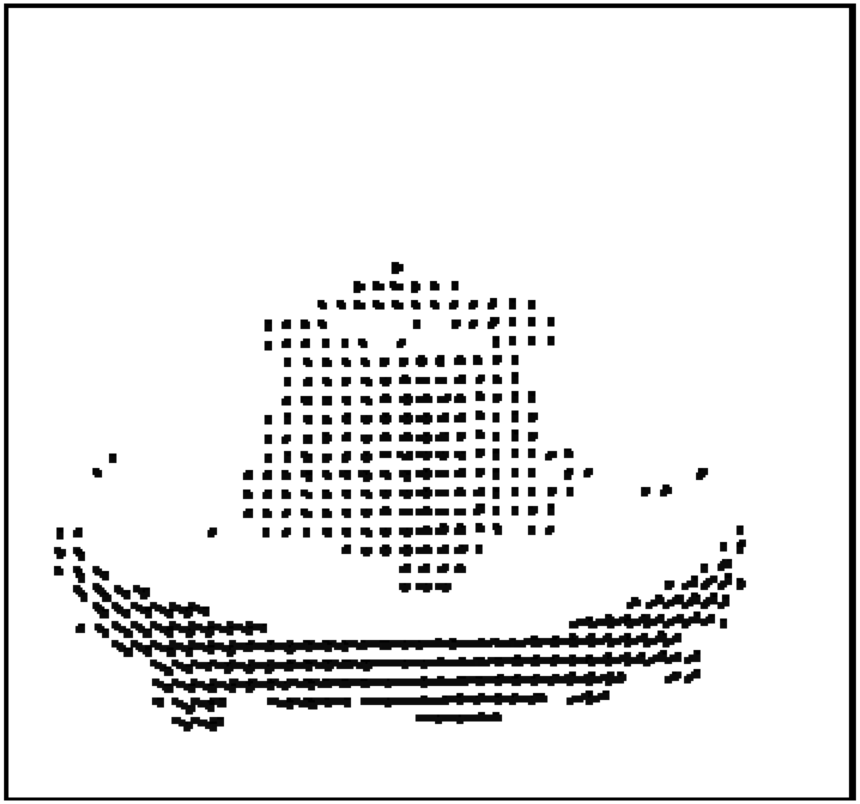


(a) (b)

Figure 12-10. (a) The estimated optical flow after several more iterations. (b) The computed motion field.

Optical Flow Result





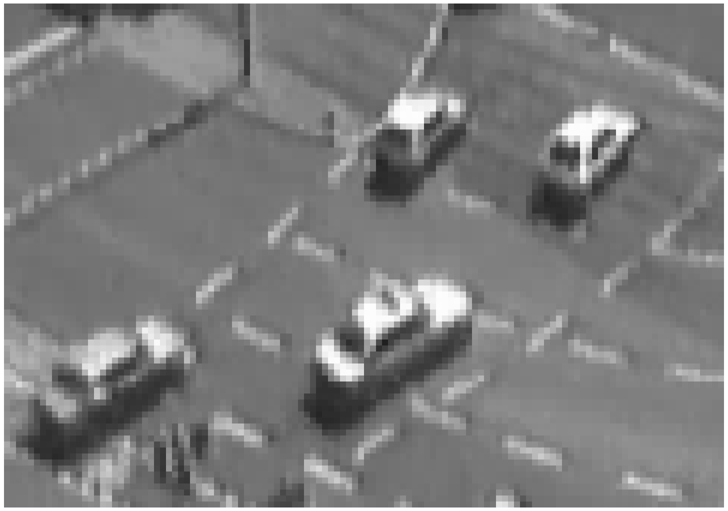
Horn & Schunck, remarks

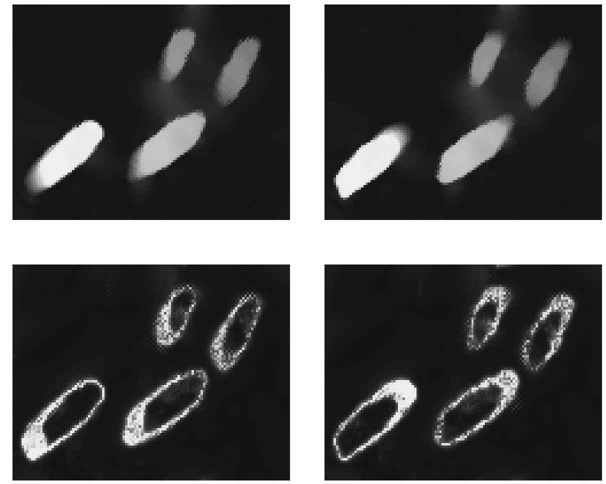
$$\int_D (\nabla I \cdot \vec{v} + I_t)^2 + \lambda^2 \left[\left(\frac{\partial v_x}{\partial x}\right)^2 + \left(\frac{\partial v_x}{\partial y}\right)^2 + \left(\frac{\partial v_y}{\partial x}\right)^2 + \left(\frac{\partial v_y}{\partial y}\right)^2 \right] dx dy$$

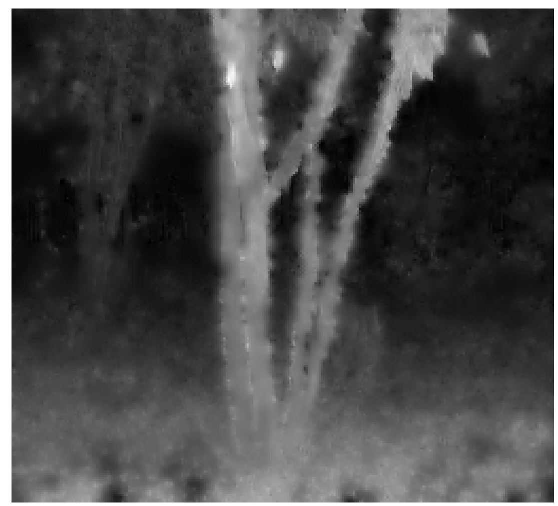
1. Errors at boundaries (smooth over)

2. Example of *regularization*
   (selection principle for the solution of ill-posed problems)

Results of an enhanced system

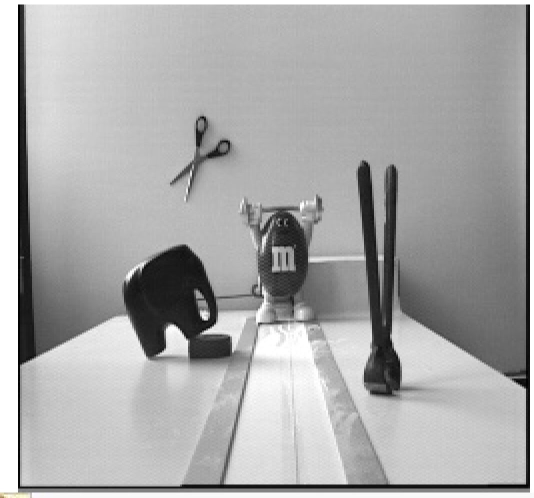


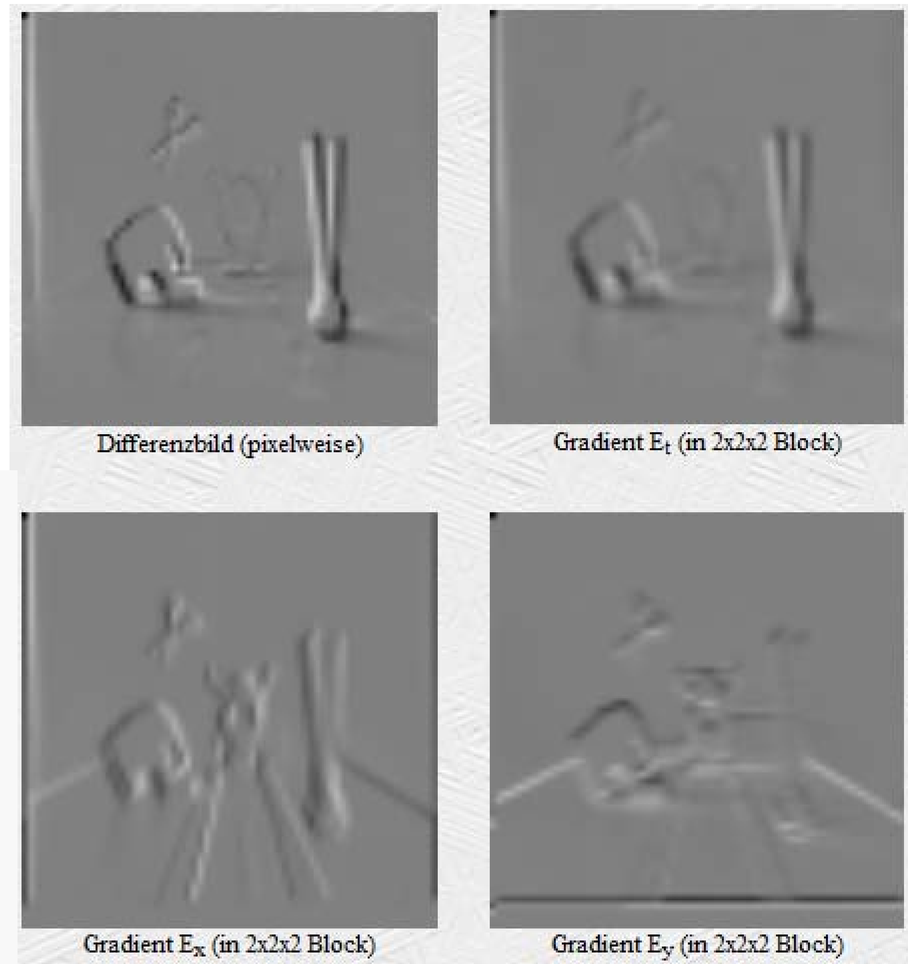




Results

http://www-student.informatik.uni-bonn.de/~gerdes/OpticalFlow/index.html



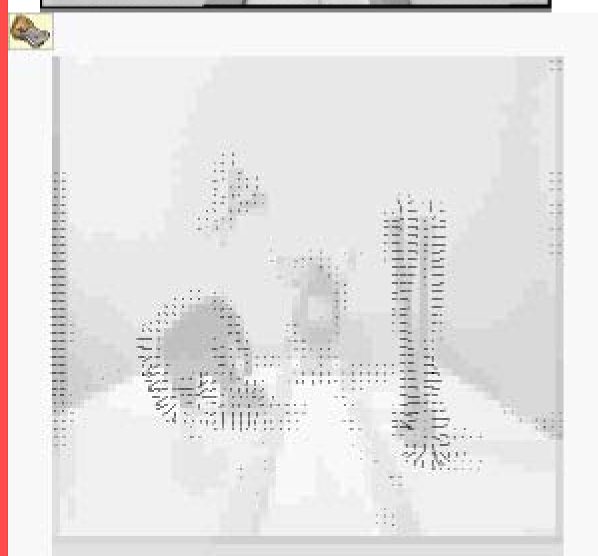


Differenzbild (pixelweise)

Gradient E_t (in 2x2x2 Block)

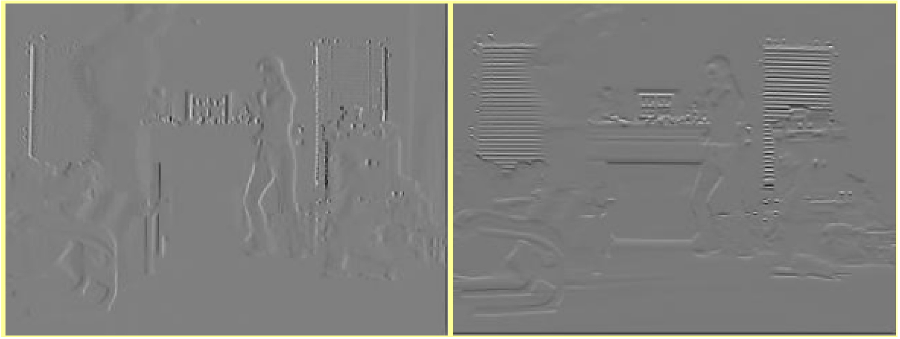
Gradient E_x (in 2x2x2 Block)

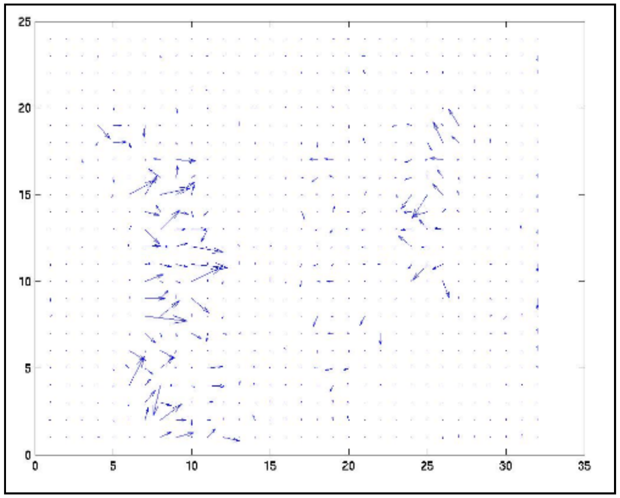
Gradient E_y (in 2x2x2 Block)



Results

http://www.cs.utexas.edu/users/jmugan/GraphicsProject/OpticalFlow/





Optical Flow

- Brightness Constancy
- The Aperture problem
- Regularization
- Lucas-Kanade
- Coarse-to-fine
- Parametric motion models
- Direct depth
- SSD tracking
- Robust flow
- Bayesian flow

Lucas & Kanade

- Assume single velocity for all pixels within a patch.
- Integrate over a patch.

Lucas & Kanade

•Assume single velocity for all pixels within a patch.
•Integrate over a patch.

- Similar to line fitting we have seen
  - Define an energy functional
    - Take derivatives equate it to 0
    - Rearrange and construct an observation matrix

$$E = \sum (uI_x + vI_y + I_t)^2$$

$$\frac{\partial E}{\partial u} = \sum 2I_x(uI_x + vI_y + I_t) = 0$$

$$\frac{\partial E}{\partial v} = \sum 2I_y(uI_x + vI_y + I_t) = 0$$

Lucas & Kanade

$$\boxed{\frac{\partial E}{\partial u} = \sum 2I_x(uI_x + vI_y + I_t) = 0}$$

$$\boxed{\frac{\partial E}{\partial v} = \sum 2I_y(uI_x + vI_y + I_t) = 0}$$

$$\sum uI_x^2 + \sum vI_xI_y + \sum I_xI_t = 0$$

$$u\sum I_x^2 + v\sum I_xI_y = -\sum I_xI_t$$

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

$$\sum uI_xI_y + \sum vI_y^2 + \sum I_yI_t = 0$$

$$u\sum I_xI_y + v\sum I_y^2 = -\sum I_yI_t$$

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

$$\boxed{\overbrace{\begin{bmatrix} \sum I_x^2 & \sum I_xI_y \\ \sum I_xI_y & \sum I_y^2 \end{bmatrix}}^{A} \overbrace{\begin{bmatrix} u \\ v \end{bmatrix}}^{u} = \overbrace{\begin{bmatrix} -\sum I_xI_t \\ -\sum I_yI_t \end{bmatrix}}^{B}}$$

Lucas & Kanade

$$Au = B \qquad A^{-1}Au = A^{-1}B \qquad Iu = A^{-1}B \qquad u = A^{-1}B$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum I_x I_t \\ -\sum I_y I_t \end{bmatrix}$$

$$\boxed{\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{\sum I_x^2 \sum I_y^2 - \left(\sum I_x I_y\right)^2} \begin{bmatrix} \sum I_y^2 & -\sum I_x I_y \\ -\sum I_x I_y & \sum I_x^2 \end{bmatrix} \begin{bmatrix} -\sum I_x I_t \\ -\sum I_y I_t \end{bmatrix}}$$

Lucas-Kanade: Integrate over a Patch

Assume a single velocity for all pixels within an image patch

$$E(u,v) = \sum_{x,y\in\Omega} \left(I_x(x,y)u + I_y(x,y)v + I_t\right)^2$$

$$\frac{dE(u,v)}{du} = \sum 2I_x\left(I_x u + I_y v + I_t\right) = 0$$

Solve with:

$$\frac{dE(u,v)}{dv} = \sum 2I_y\left(I_x u + I_y v + I_t\right) = 0$$

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

On the LHS: sum of the 2x2 outer product tensor of the gradient vector

$$\left(\sum \nabla I \nabla I^T\right)\vec{U} = -\sum \nabla I I_t$$

Lucas-Kanade: Singularities and the Aperture Problem

Let $M = \sum (\nabla I)(\nabla I)^T$ and $b = \begin{bmatrix} -\sum I_x I_t \\ -\sum I_y I_t \end{bmatrix}$

- Algorithm: At each pixel compute U by solving $MU = b$

- M is singular if all gradient vectors point in the same direction
  - -- e.g., along an edge
  - -- of course, trivially singular if the summation is over a single pixel
  - -- i.e., only *normal flow* is available (aperture problem)

- Corners and textured areas are OK

Discussion

- Horn-Schunck: Add smoothness constraint.

$$\int_D (\nabla I \cdot \vec{v} + I_t)^2 + \lambda^2 \left[\left(\frac{\partial v_x}{\partial x}\right)^2 + \left(\frac{\partial v_x}{\partial y}\right)^2 + \left(\frac{\partial v_y}{\partial x}\right)^2 + \left(\frac{\partial v_y}{\partial y}\right)^2 \right] dx dy$$

- Lucas-Kanade: Provide constraint by minimizing over local neighborhood:

$$\sum_{x,y \in \Omega} W^2(x,y)[\nabla I(x,y,t) \cdot \vec{v} + I_t(x,y,t)]^2$$

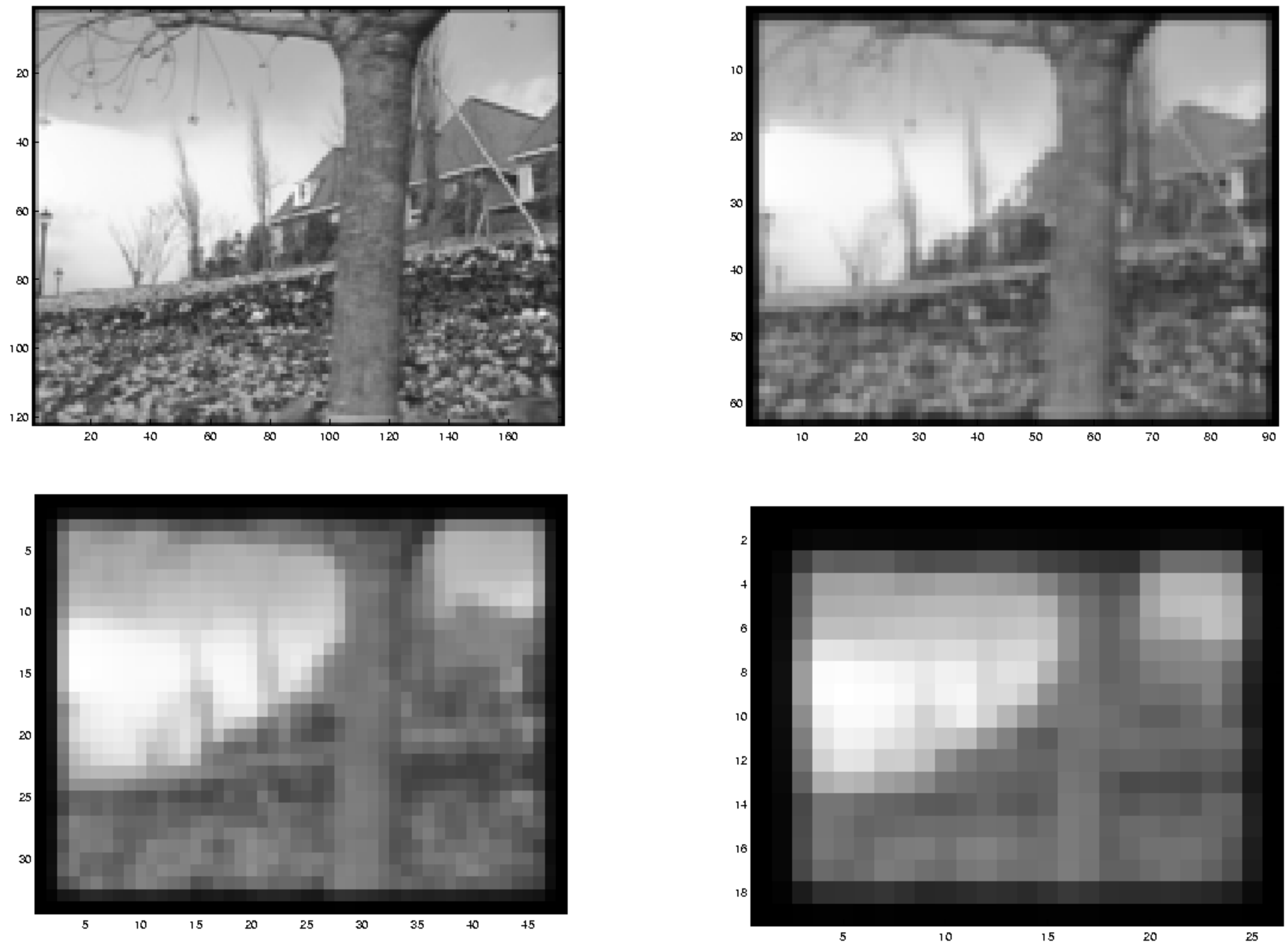
- Horn-Schunck and Lucas-Kanade optical methods work only for small motion.
- If object moves faster, the brightness changes rapidly, derivative masks fail to estimate spatiotemporal derivatives.
- Pyramids can be used to compute large optical flow vectors.

Iterative Refinement (Iterative Lucas-Kanade)

- Estimate velocity at each pixel using one iteration of Lucas and Kanade estimation
- Warp one image toward the other using the estimated flow field

  (easier said than done)
- Refine estimate by repeating the process

Reduce the Resolution!



Optical Flow

- Brightness Constancy
- The Aperture problem
- Regularization
- Lucas-Kanade
- Coarse-to-fine
- Parametric motion models
- Direct depth
- SSD tracking
- Robust flow
- Bayesian flow

Limits of the (local) gradient method

1. Fails when intensity structure within window is poor
2. Fails when the displacement is large (typical operating range is motion of 1 pixel per iteration!)
   – *Linearization of brightness is suitable only for small displacements*

Also, brightness is not strictly constant in images

– *actually less problematic than it appears, since we can pre-filter images to make them look similar*

Revisiting the Small Motion Assumption

- Is this motion small enough?
  - Probably not—it's much larger than one pixel (2^{nd} order terms dominate)
  - How might we solve this problem?

Revisiting the Small Motion Assumption

- Is this motion small enough?
  - Probably not—it's much larger than one pixel (2nd order terms dominate)
  - How might we solve this problem?

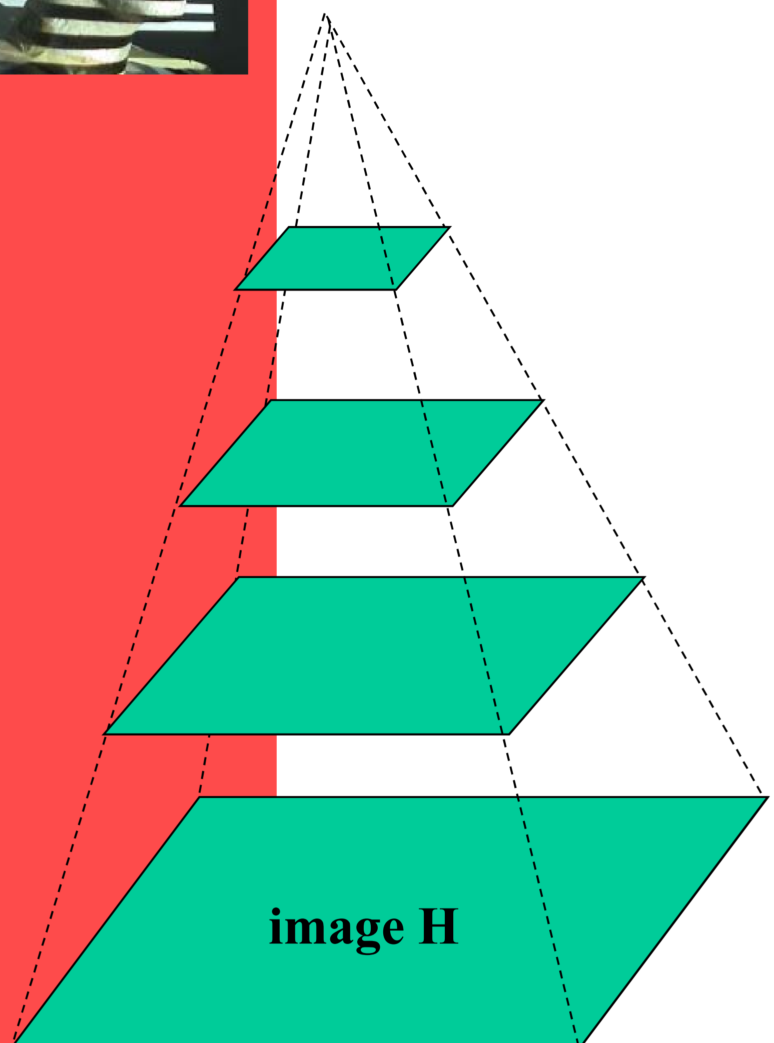
Revisiting the Small Motion Assumption

- Is this motion small enough?
  - Probably not—it's much larger than one pixel (2nd order terms dominate)
  - How might we solve this problem?

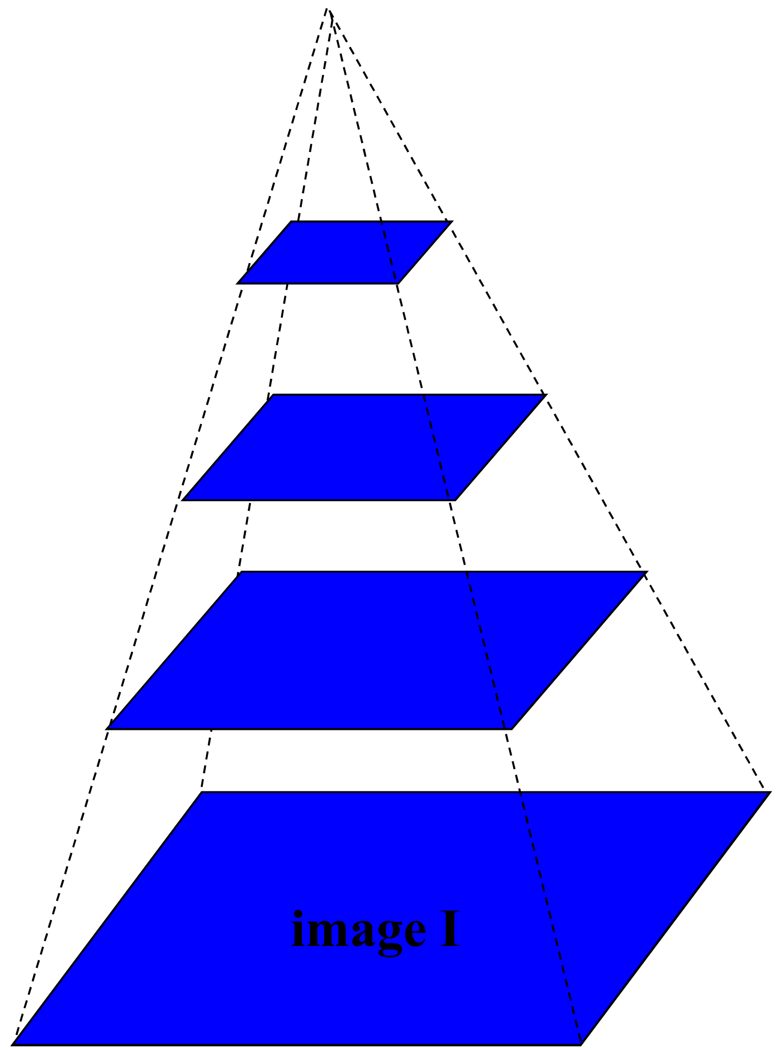
Coarse-to-fine Optical Flow Estimation

Coarse-to-fine Optical Flow Estimation

Coarse-to-fine Optical Flow Estimation

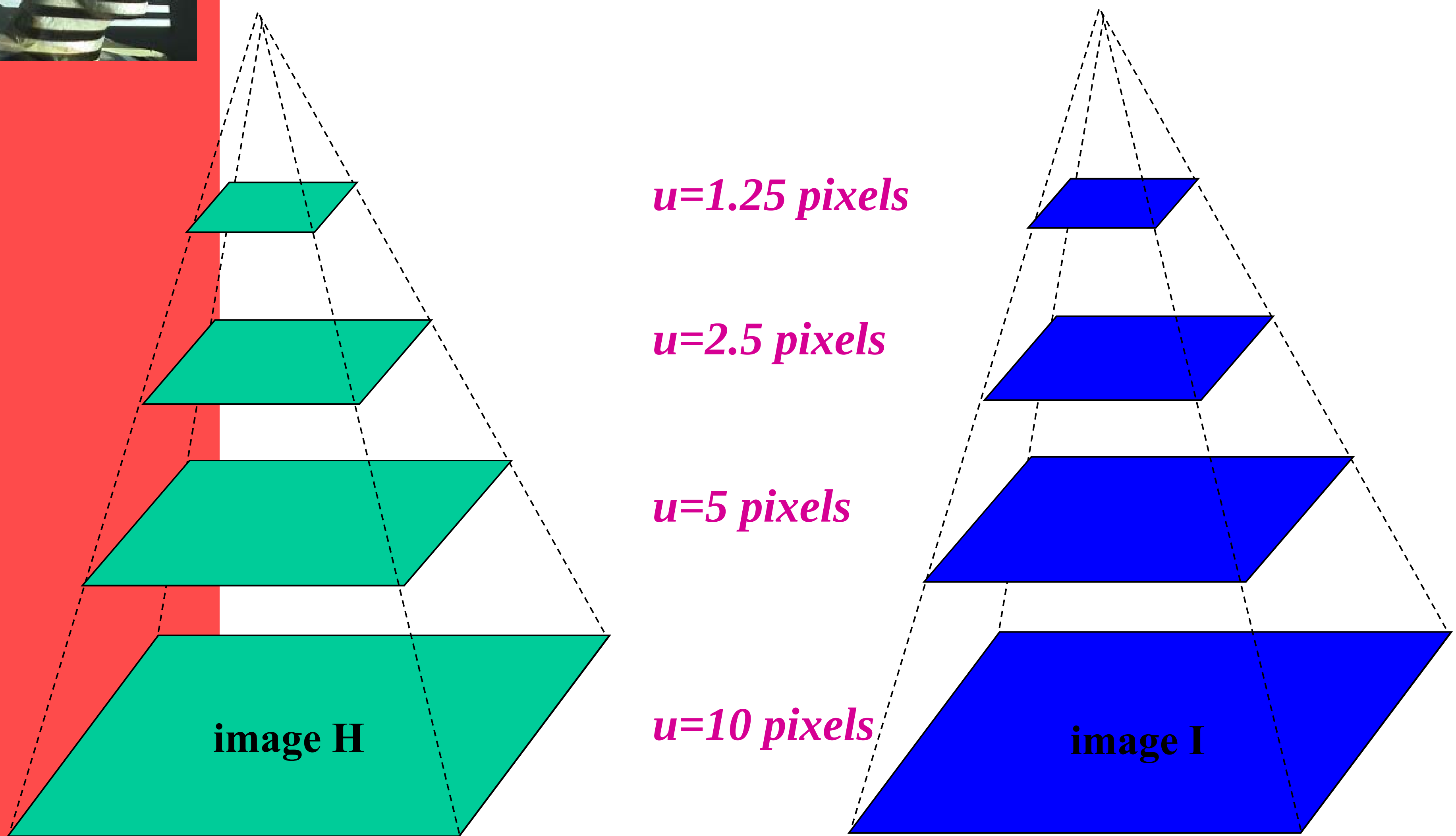


Gaussian pyramid of image *H*



Gaussian pyramid of image *I*

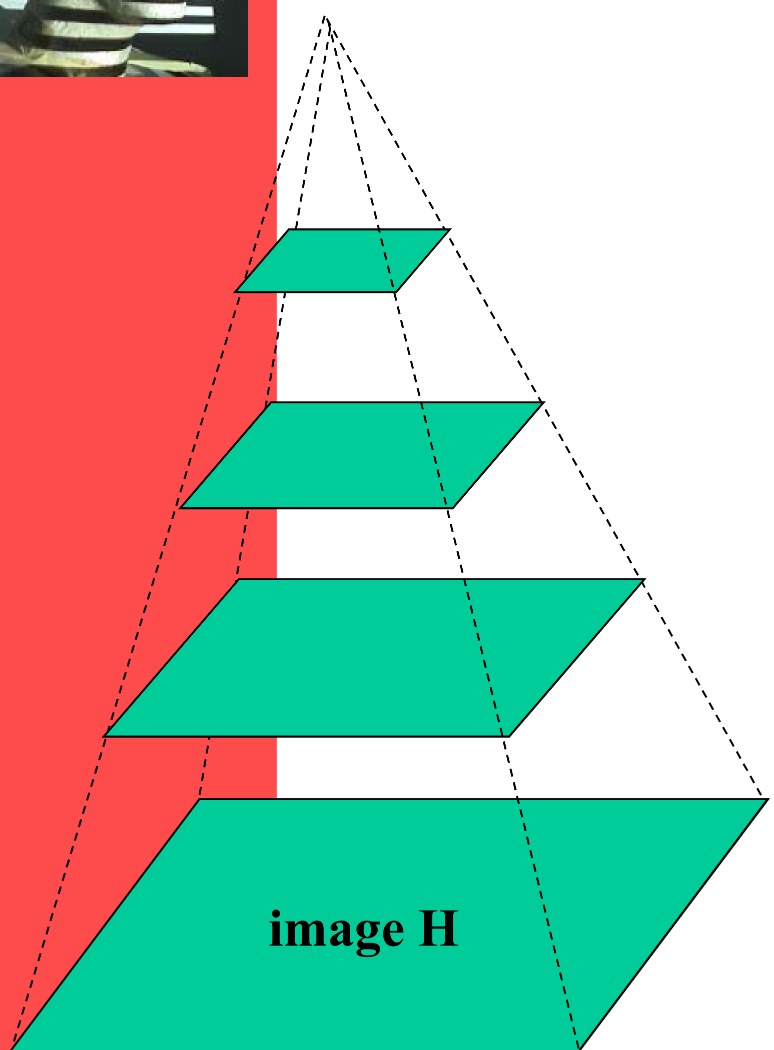
Coarse-to-fine Optical Flow Estimation



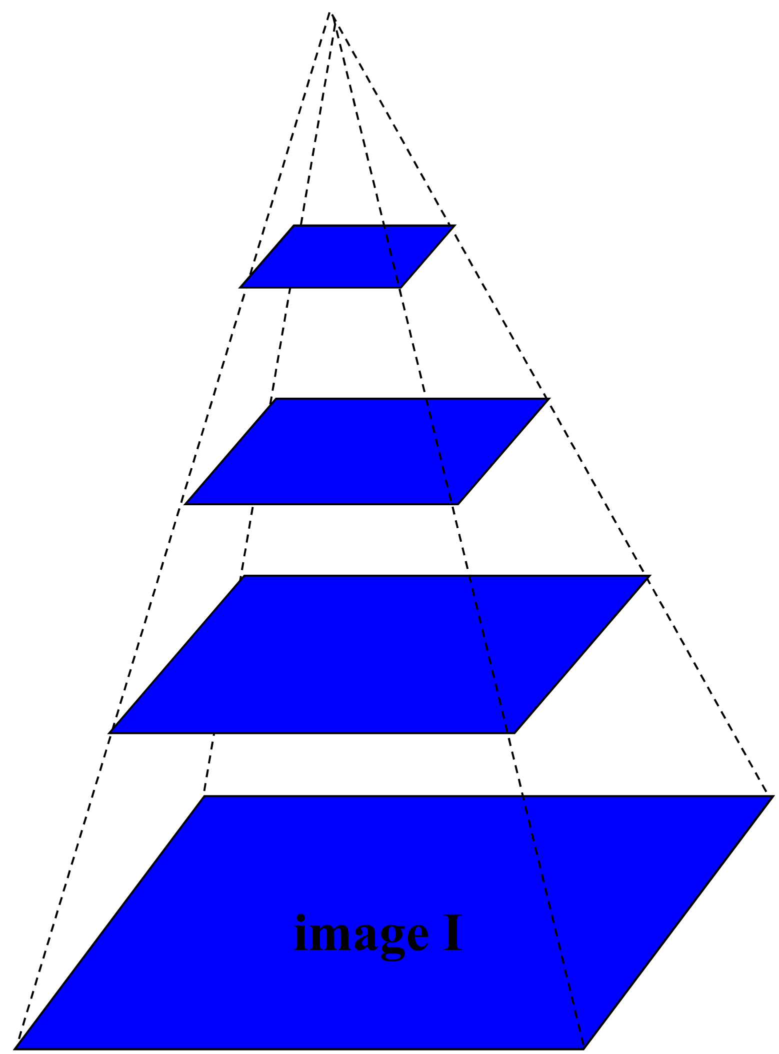
Gaussian pyramid of image *H*

Gaussian pyramid of image *I*

Coarse-to-fine Optical Flow Estimation

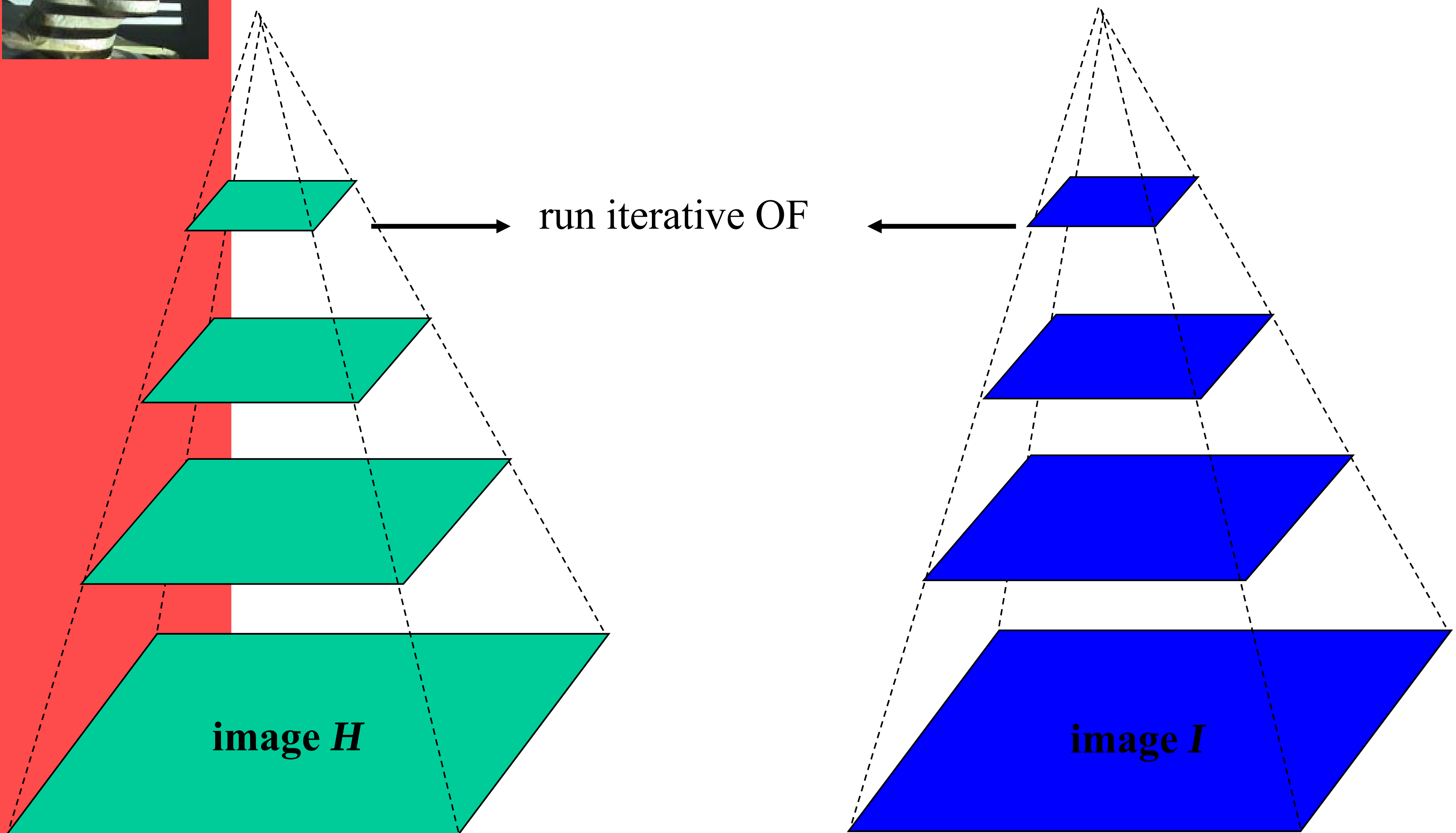


Gaussian pyramid of image *H*



Gaussian pyramid of image *I*

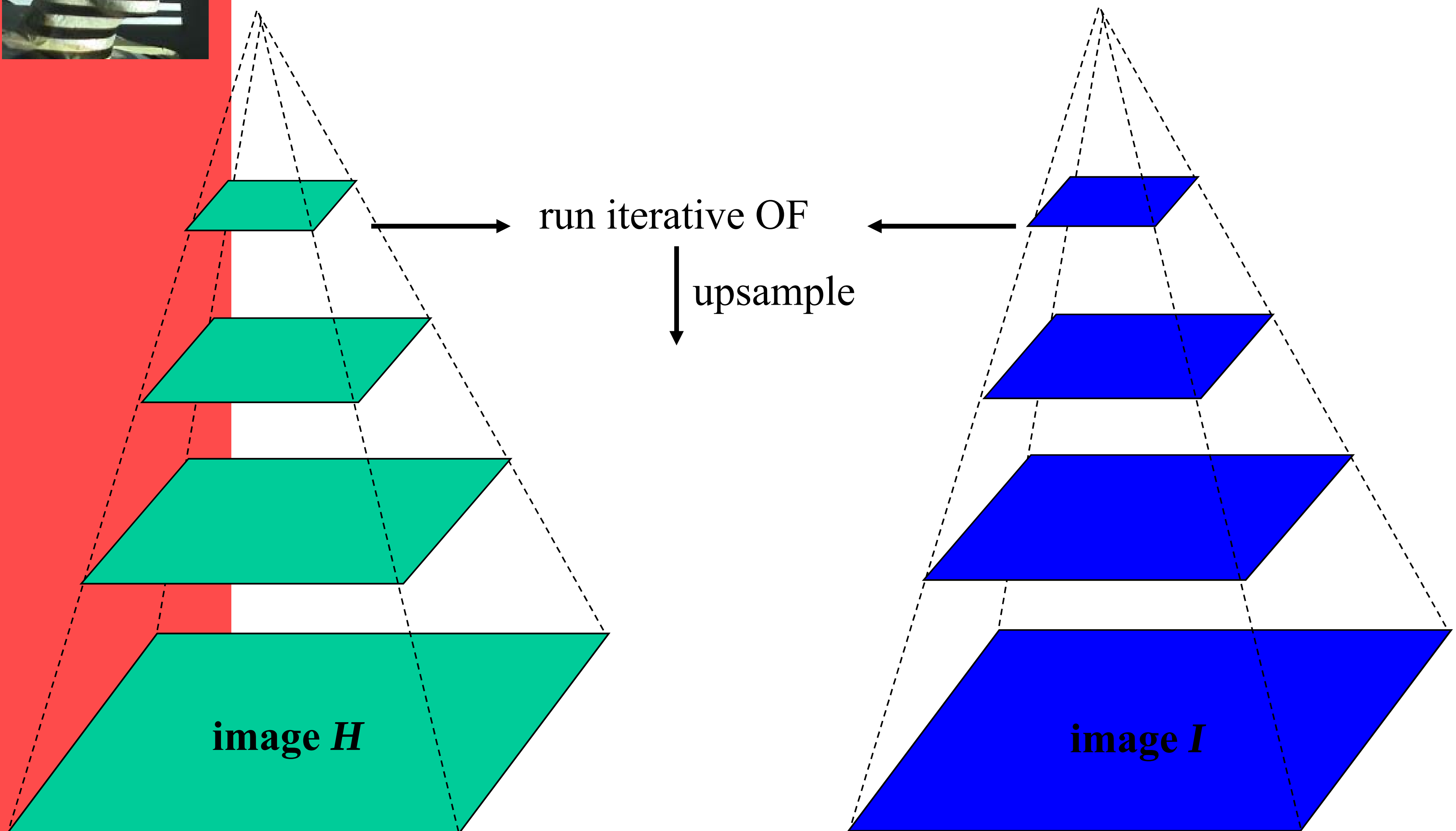
Coarse-to-fine Optical Flow Estimation



Gaussian pyramid of image *H*

Gaussian pyramid of image *I*

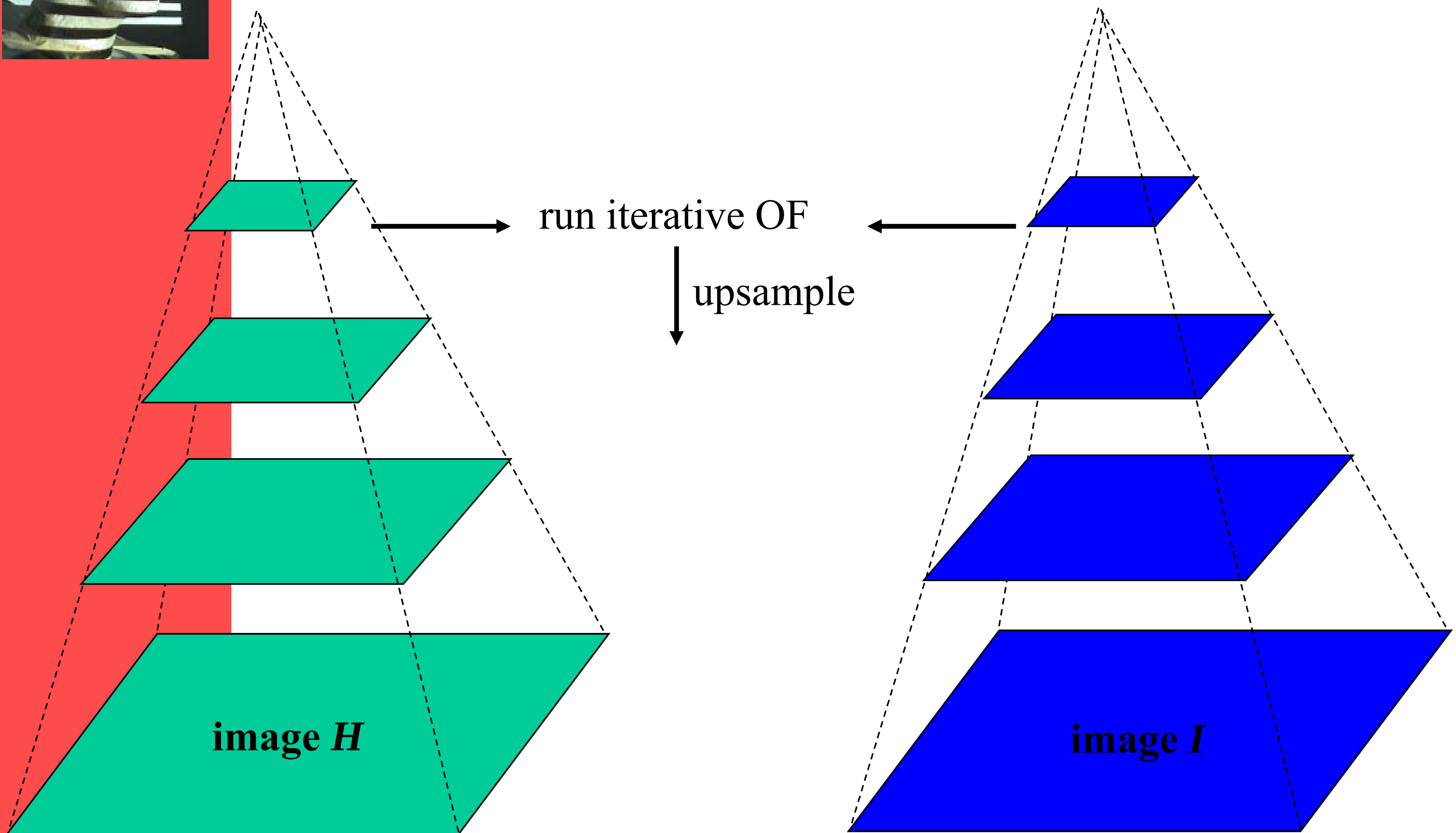
Coarse-to-fine Optical Flow Estimation



Gaussian pyramid of image *H*

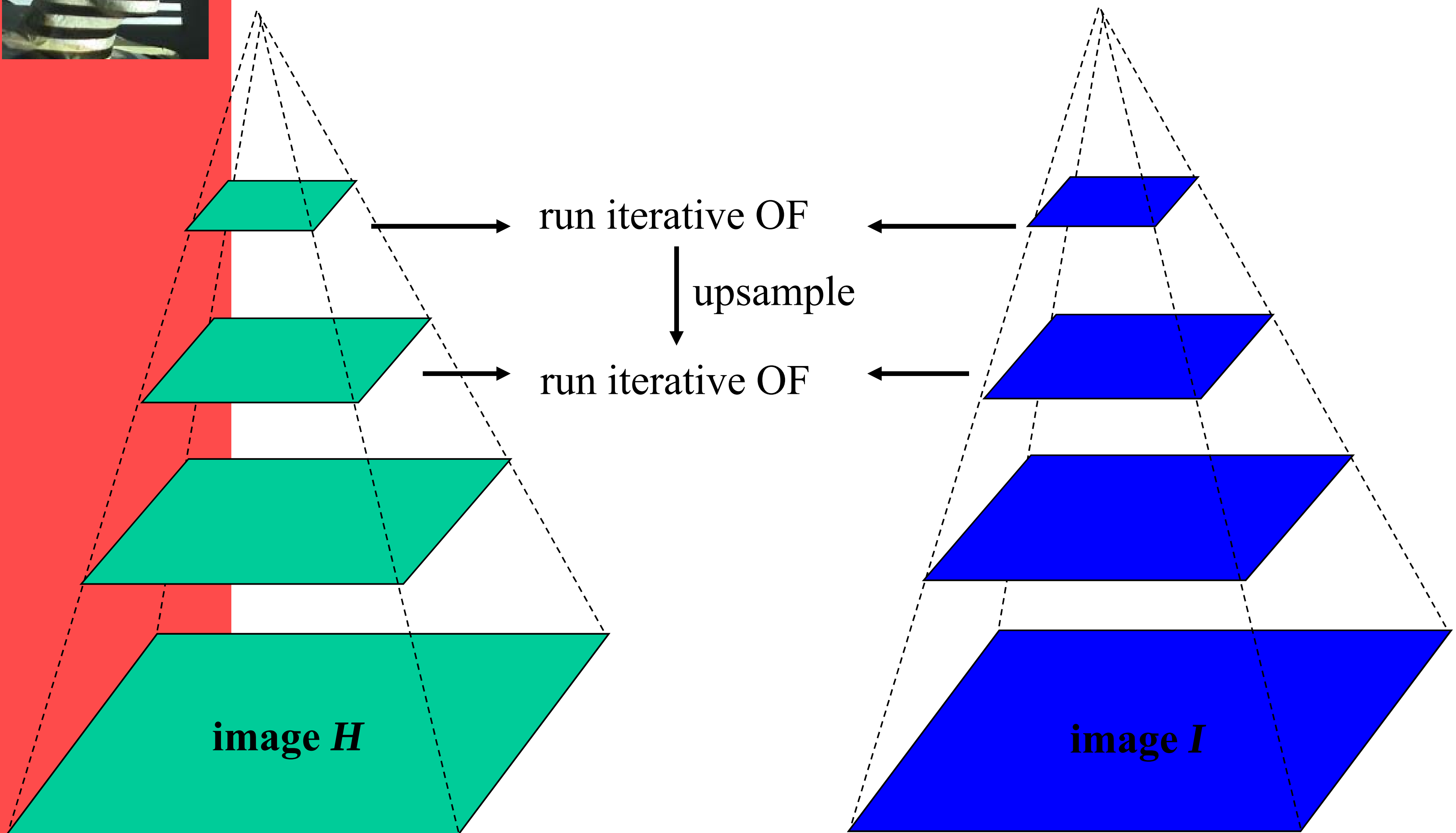
Gaussian pyramid of image *I*

Coarse-to-fine Optical Flow Estimation



Gaussian pyramid of image *H* **Gaussian pyramid of image *I***

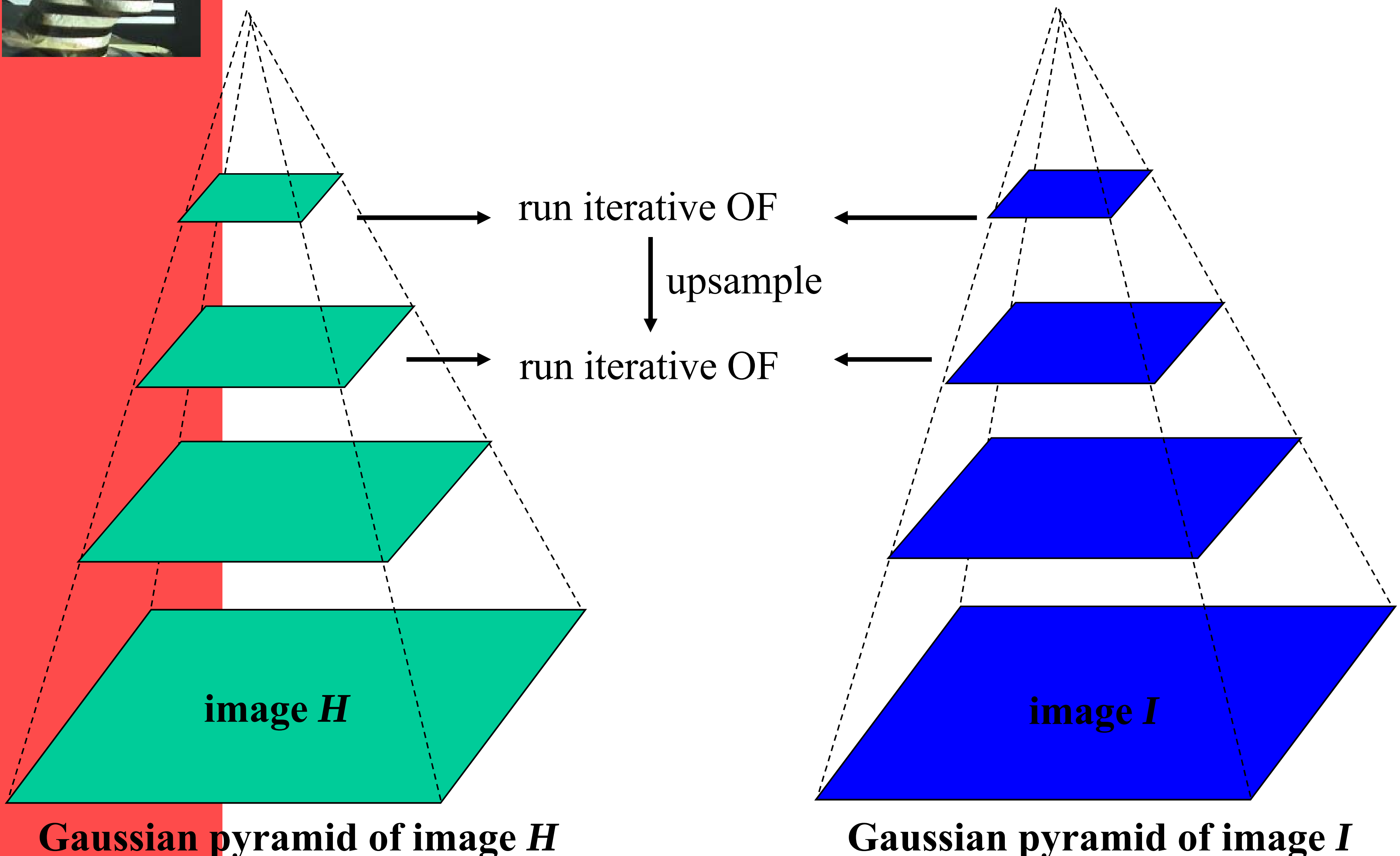
Coarse-to-fine Optical Flow Estimation



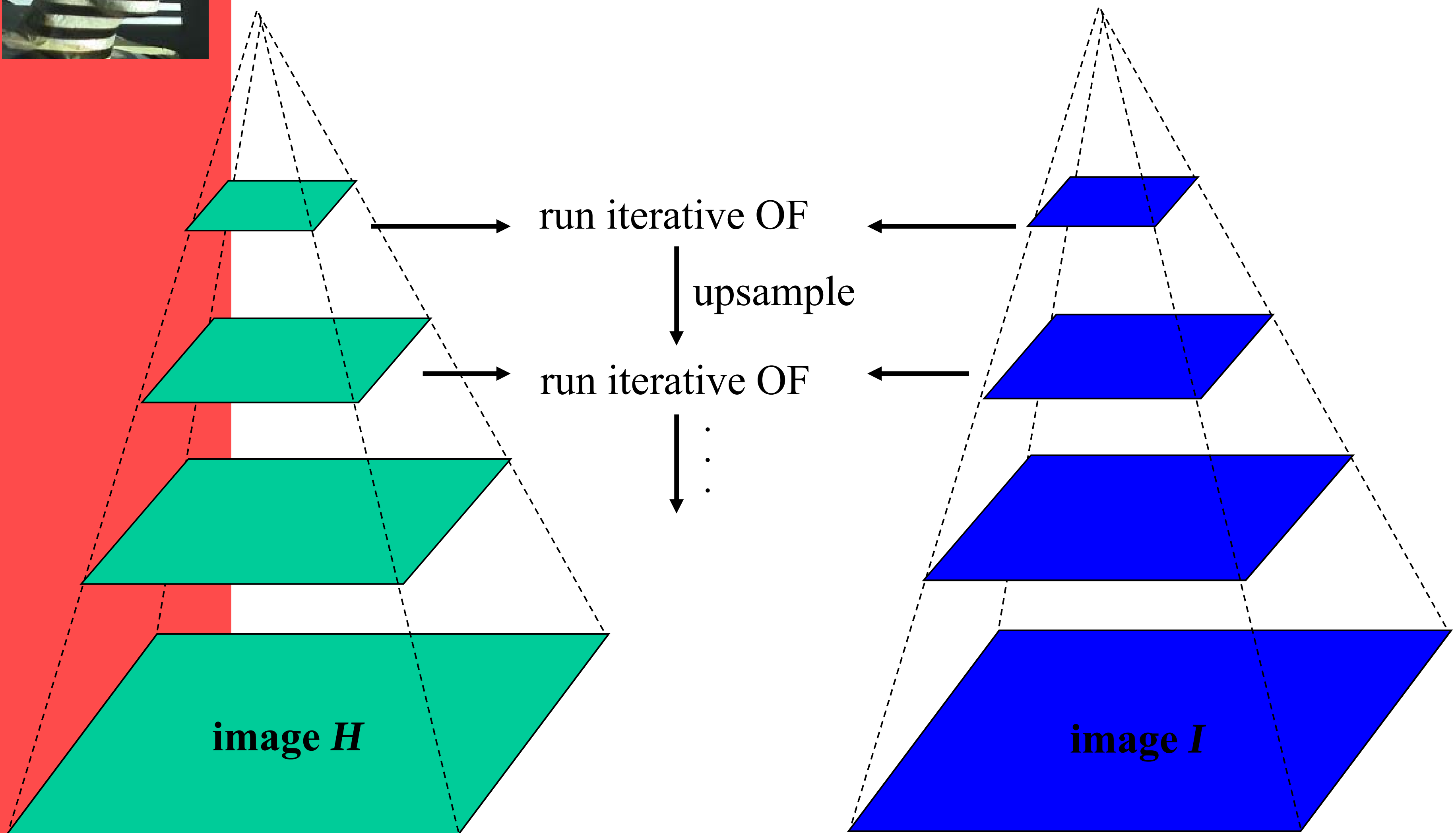
Gaussian pyramid of image *H*

Gaussian pyramid of image *I*

Coarse-to-fine Optical Flow Estimation



Coarse-to-fine Optical Flow Estimation

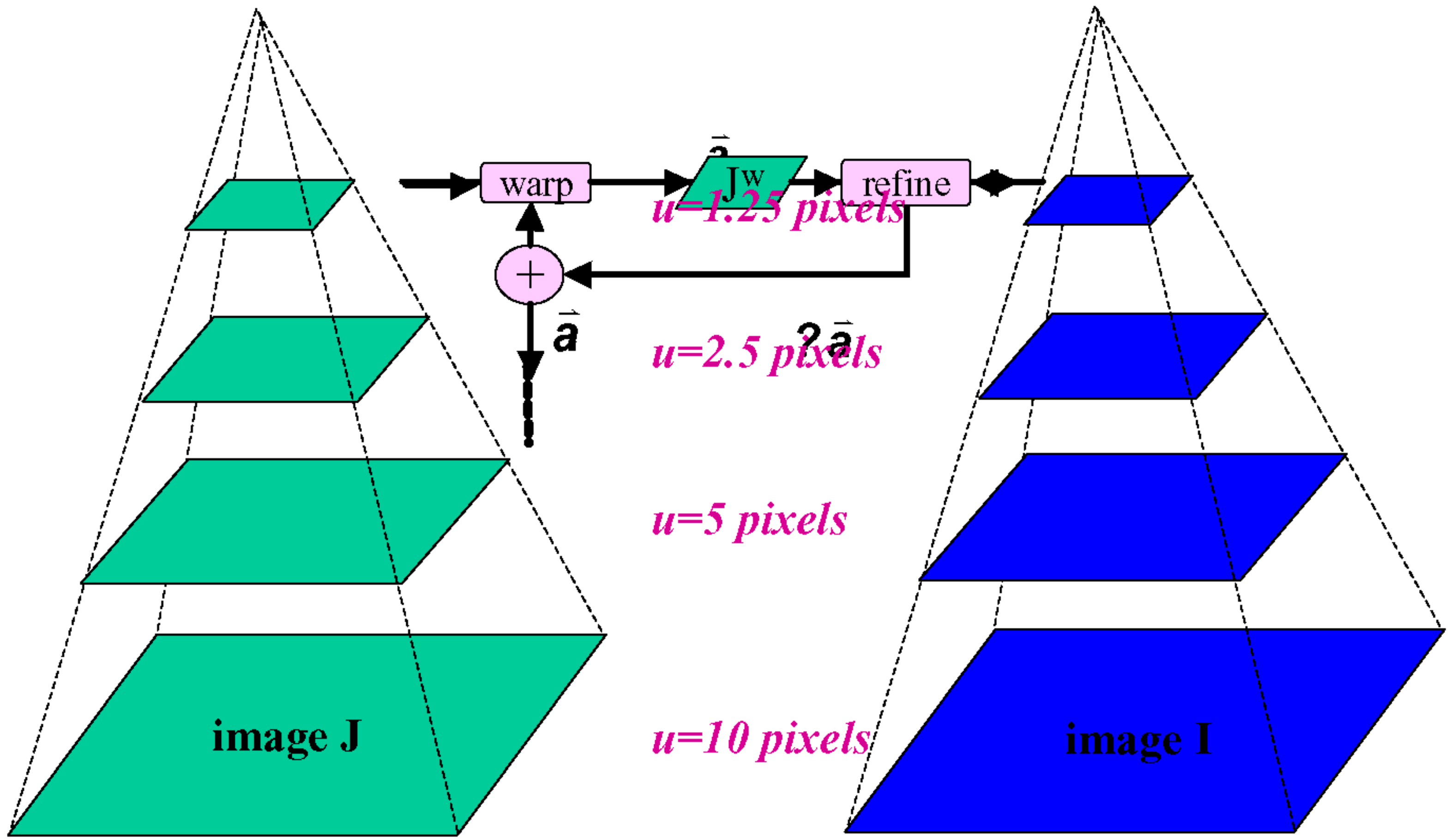


Gaussian pyramid of image *H*

Gaussian pyramid of image *I*

Coarse-to-Fine Estimation

$\boxed{I_x \cdot u + I_y \cdot v + I_t \approx 0}$ ==> small ***u*** and ***v*** ...



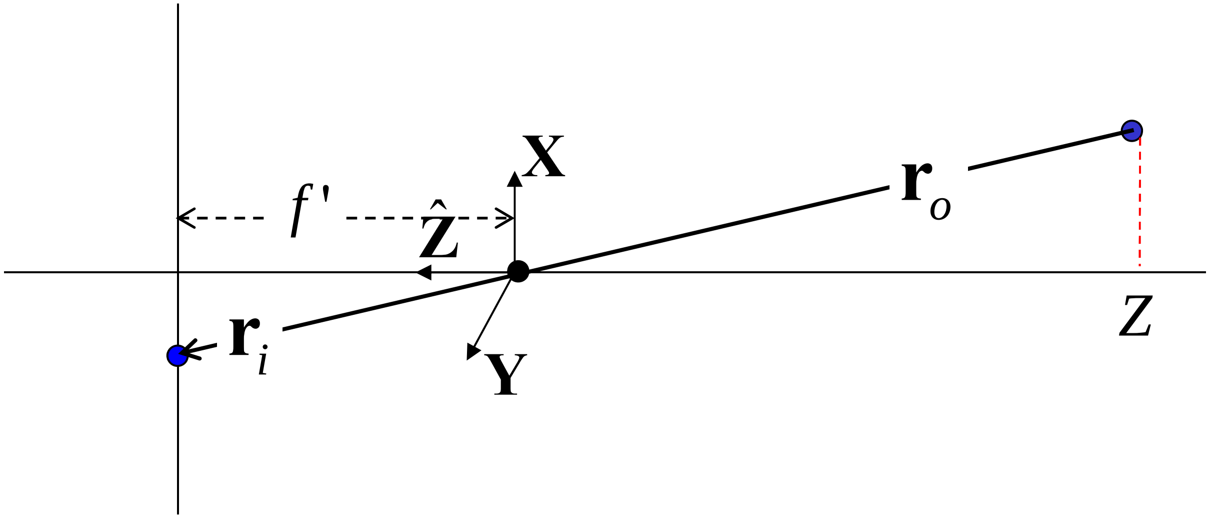
Pyramid of image J **Pyramid of image I**

Video Segmentation

Next:
Motion Field
Structure from Motion

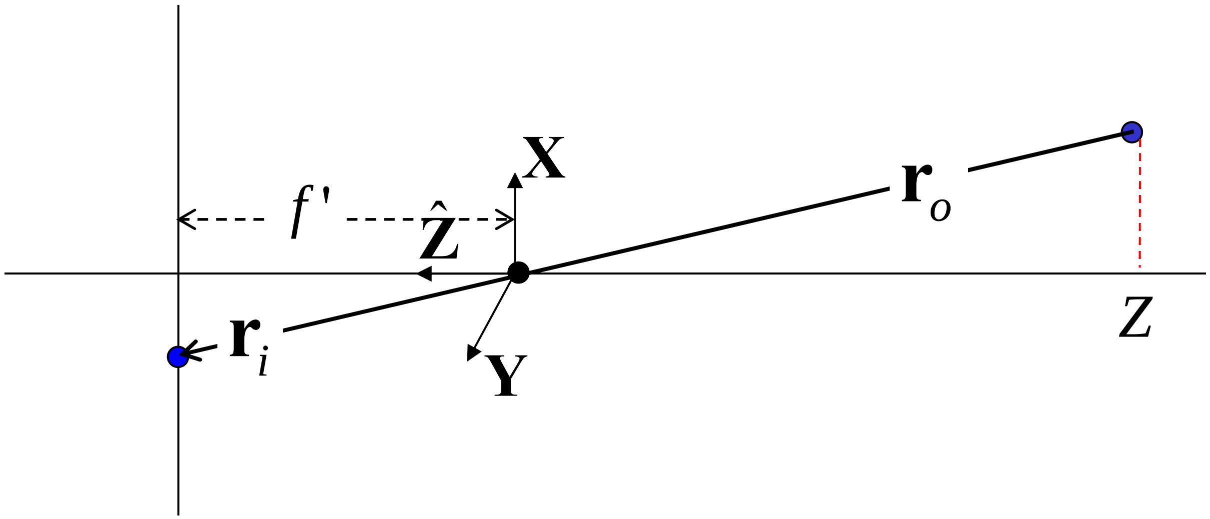
Motion Field

- Image velocity of a point moving in the scene



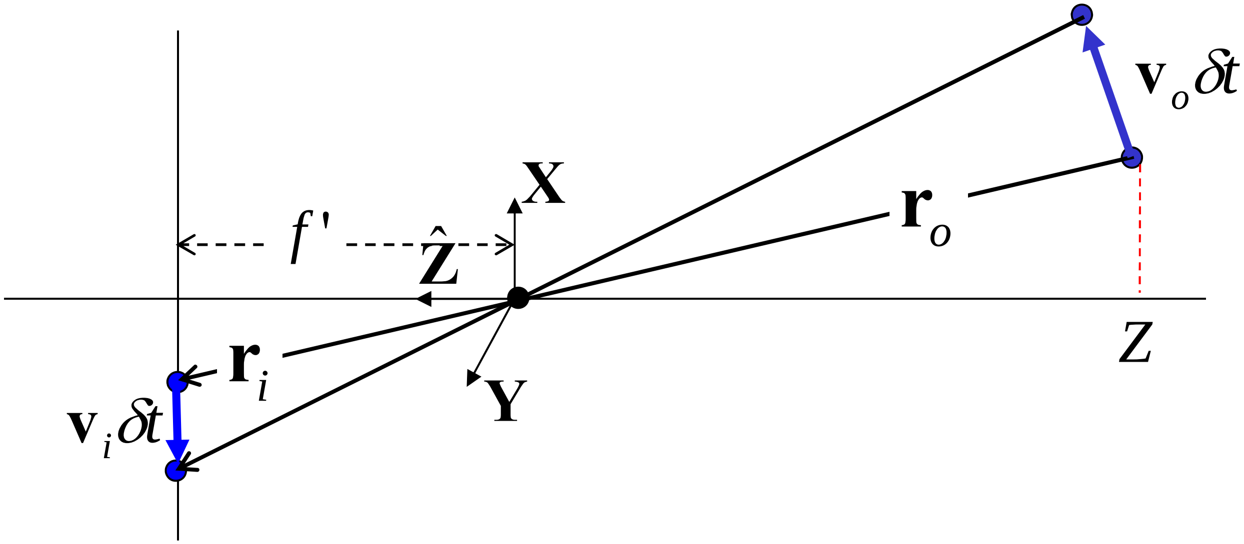
Motion Field

- Image velocity of a point moving in the scene



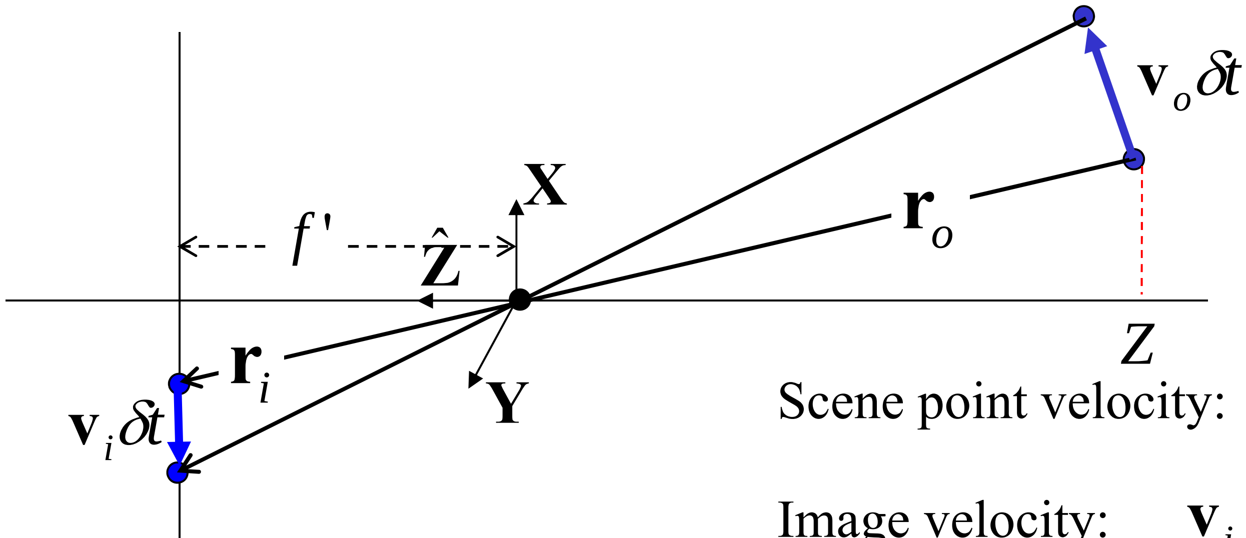
Motion Field

- Image velocity of a point moving in the scene



Motion Field

- Image velocity of a point moving in the scene

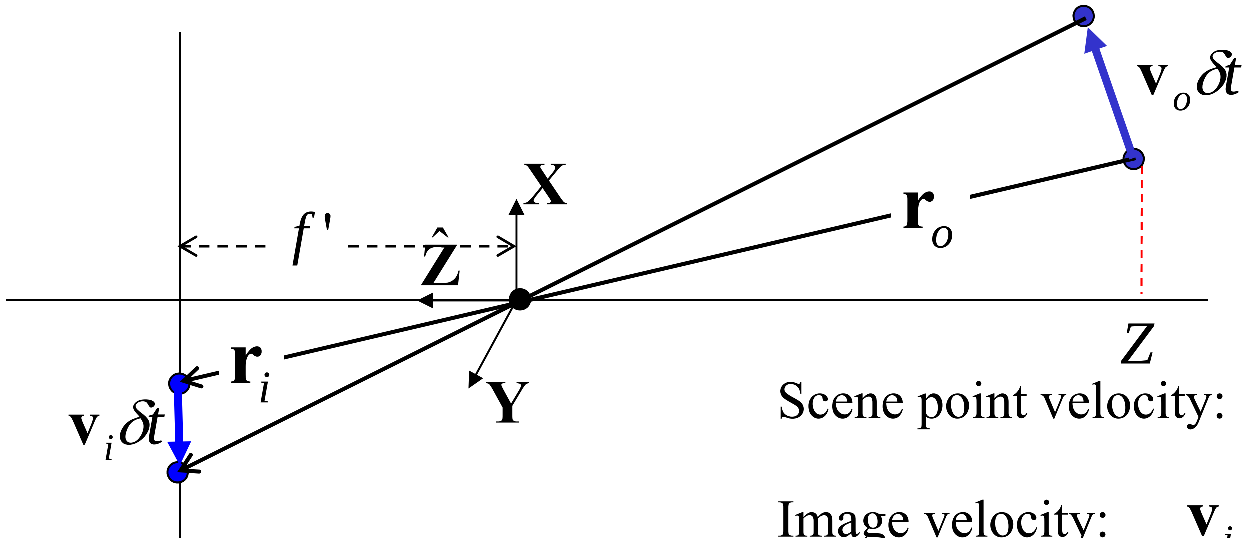


Scene point velocity: $\mathbf{v}_o = \dfrac{d\mathbf{r}_o}{dt}$

Image velocity: $\mathbf{v}_i = \dfrac{d\mathbf{r}_i}{dt}$

Motion Field

- Image velocity of a point moving in the scene



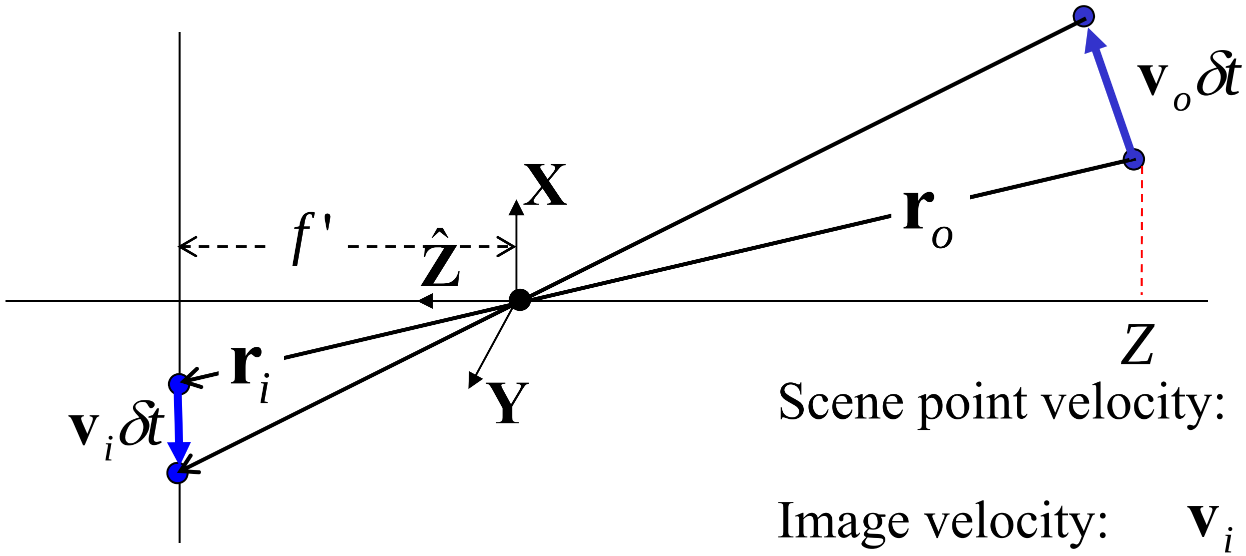
Scene point velocity: $\mathbf{v}_o = \dfrac{d\mathbf{r}_o}{dt}$

Image velocity: $\mathbf{v}_i = \dfrac{d\mathbf{r}_i}{dt}$

Perspective projection: $\dfrac{1}{f'}\mathbf{r}_i = \dfrac{\mathbf{r}_o}{\mathbf{r}_o \cdot \hat{\mathbf{Z}}} = \dfrac{\mathbf{r}_o}{Z}$

Motion Field

- Image velocity of a point moving in the scene



Scene point velocity: $\mathbf{v}_o = \dfrac{d\mathbf{r}_o}{dt}$

Image velocity: $\mathbf{v}_i = \dfrac{d\mathbf{r}_i}{dt}$

Perspective projection: $\dfrac{1}{f'}\mathbf{r}_i = \dfrac{\mathbf{r}_o}{\mathbf{r}_o \cdot \hat{\mathbf{Z}}} = \dfrac{\mathbf{r}_o}{Z}$

Motion field

$$\mathbf{v}_i = \frac{d\mathbf{r}_i}{dt} = f'\frac{(\mathbf{r}_o \cdot \mathbf{Z})\mathbf{v}_o - (\mathbf{v}_o \cdot \mathbf{Z})\mathbf{r}_o}{(\mathbf{r}_o \cdot \mathbf{Z})^2} = f'\frac{(\mathbf{r}_o \times \mathbf{v}_o) \times \mathbf{Z}}{(\mathbf{r}_o \cdot \mathbf{Z})^2}$$