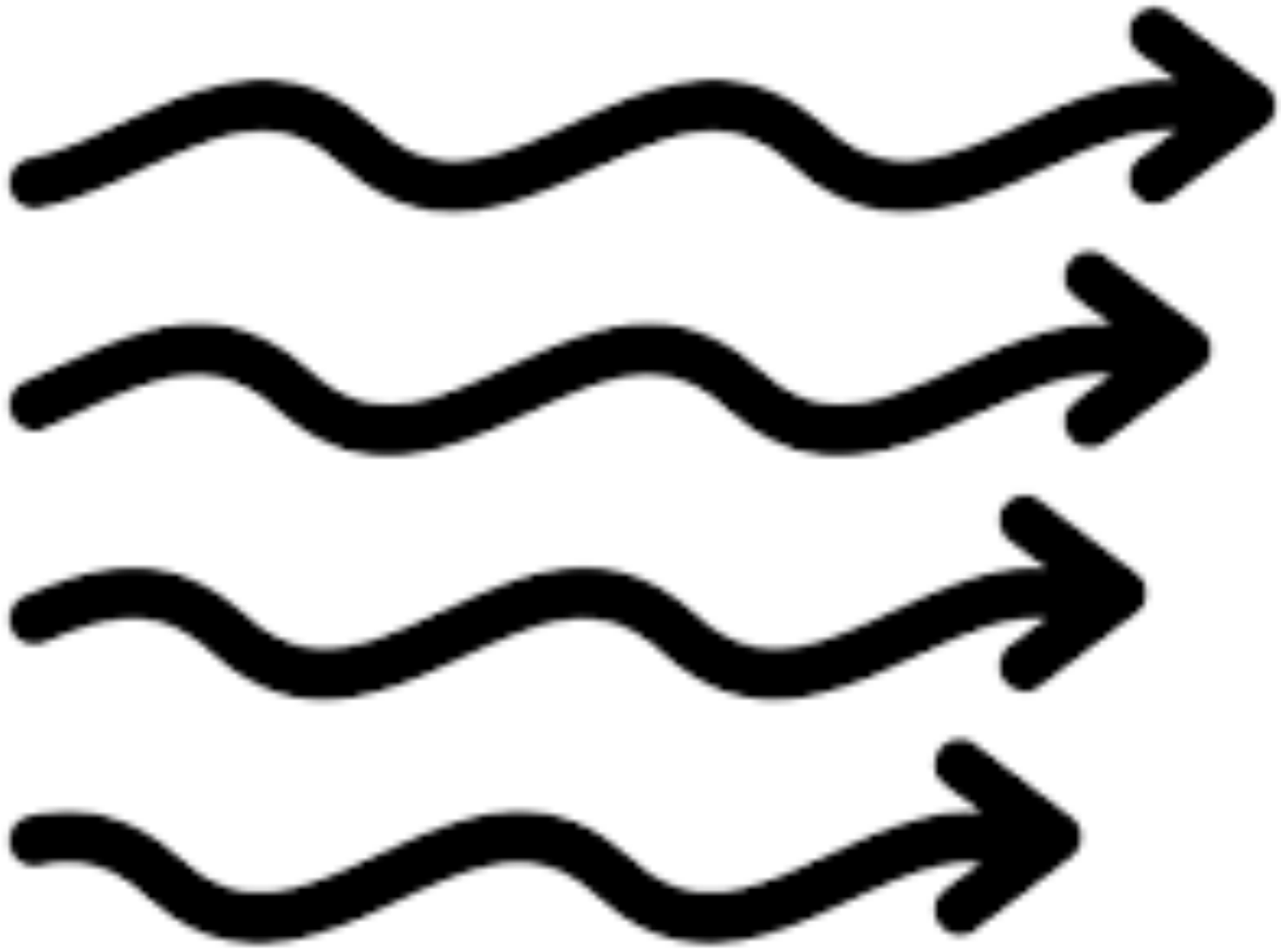
Motion and flow



Summary

- Last section: If we can compute the 2D optical flow, then we can compute the time to contact

- This section: given a sequence of video frames, how do we estimate the 2D optical flow?

- Last section: assumed stationary world, moving camera

- This section: stationary camera, moving world

Optical Flow: Outline

- Brightness constancy
- Aperture problem
- Small-motion assumption
- Sparse (KLT) vs Dense (variational)
- Optimization tools: robust statistics, coarse-to-fine, Markov Random Fields
- Motion Segmentation (dominant motion estimation, background subtraction, layered models)

General motion feilds

Simple evidence that “single image” analysis won’t always suffice

Note: Modern computer vision methods would likely fail on this example.

This is evidence that “we’re doing it wrong” somehow by not processing motion in the right way in modern computer vision systems

Moving light sequences

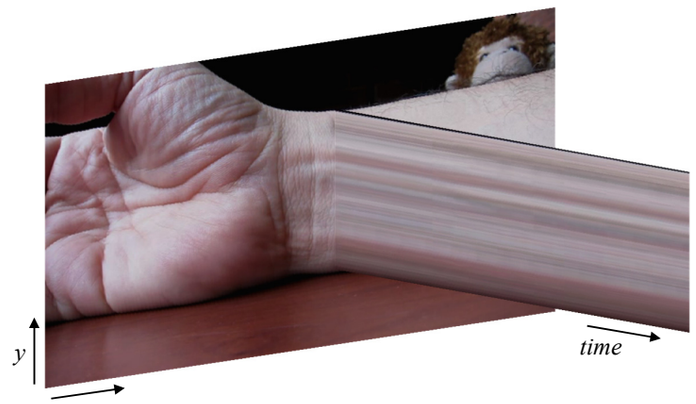
https://www.youtube.com/watch?v=1F5ICP9SYLU

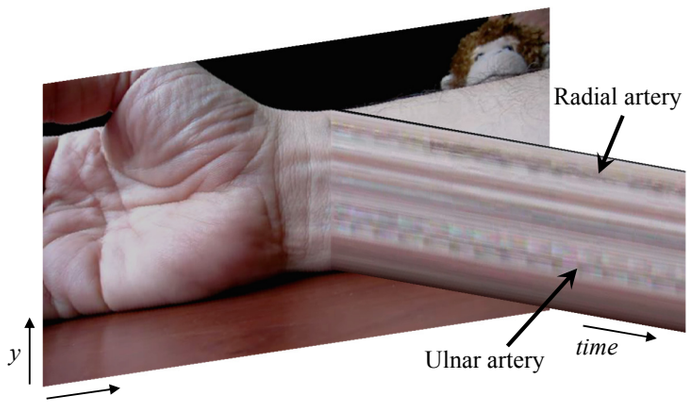
Johansson displays (1971)

As before: Modern computer vision methods would likely fail on these examples.

Amplifying temporal signals

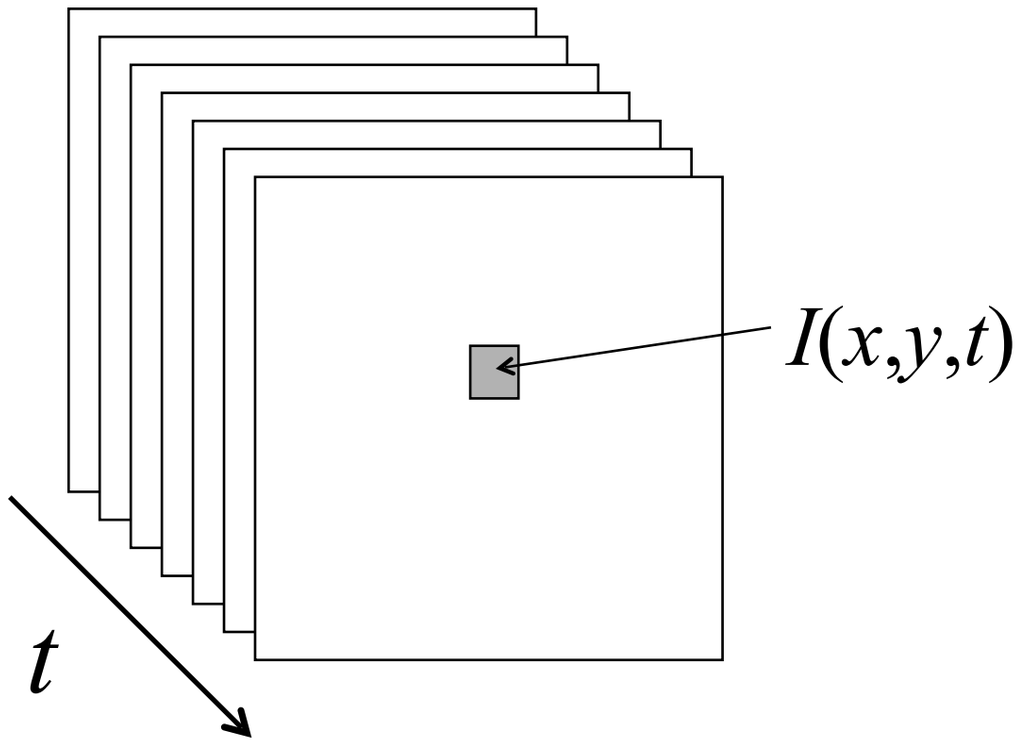
By (3D!) frequency modification or by optical flow tracking (today)





Eulerian Video Magnification for Revealing Subtle Changes in the World

Videos as spacetime cubes



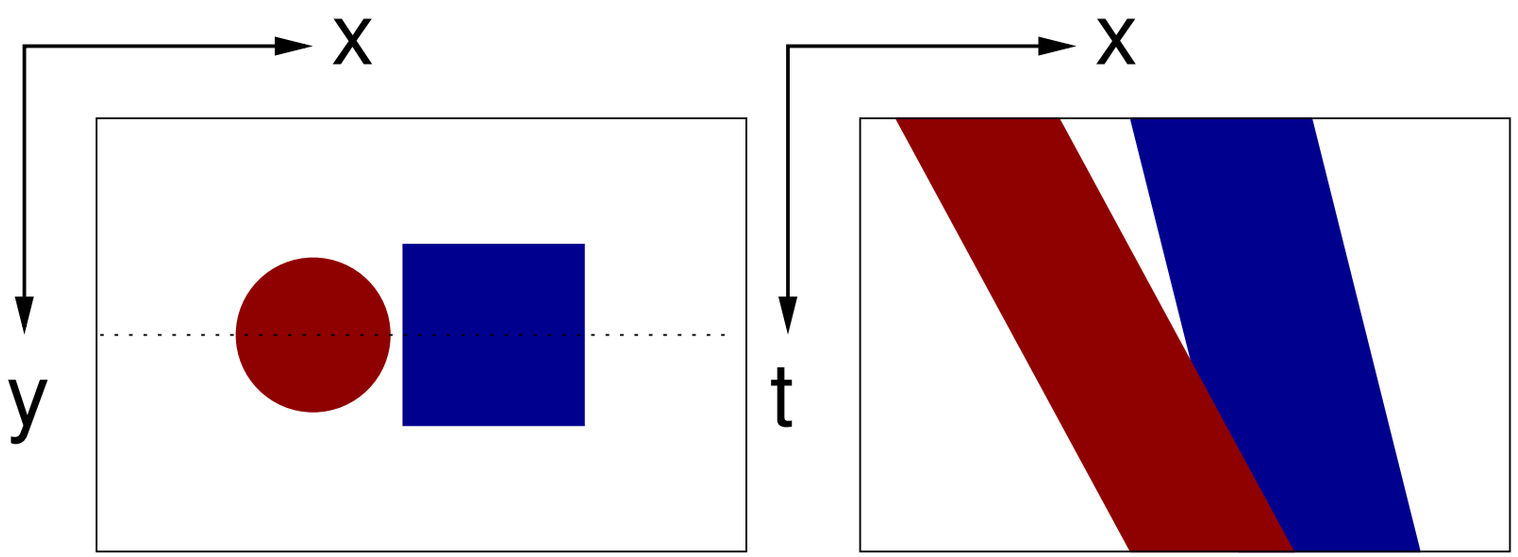
Different representations:

- “slice” this cube in different ways
- apply 3D frequency analysis
- taylor expansion in (x,y,t)

Challenges:

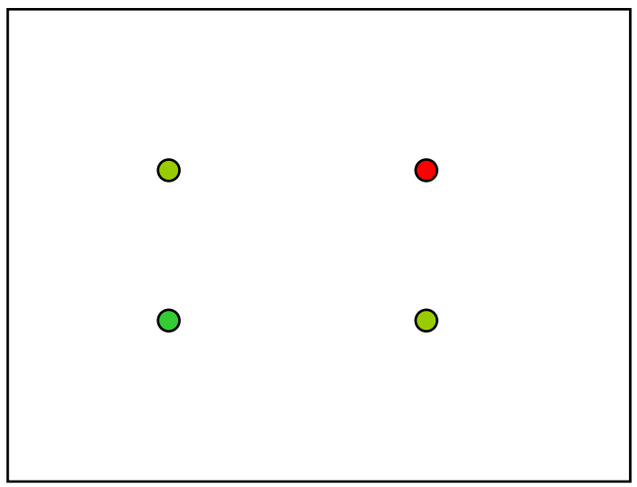
- Can’t hold large videos in (GPU) memory
- Offline vs streaming / online processing

Visualizing spacetime cubes

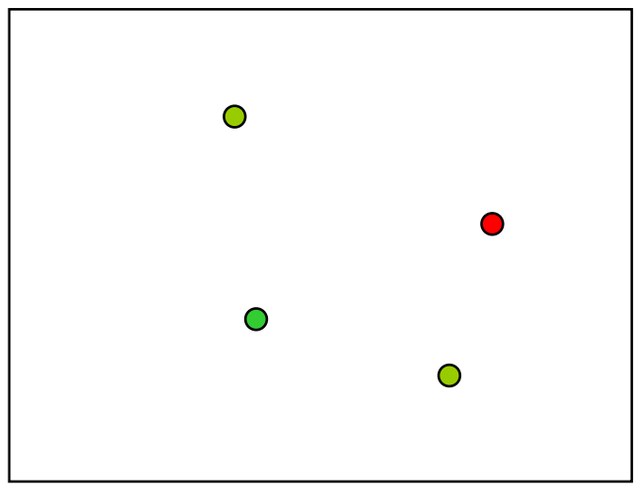


In this example, the circle is moving to the right in front of the square

Problem Definition: Optical Flow



$H(x, y)$



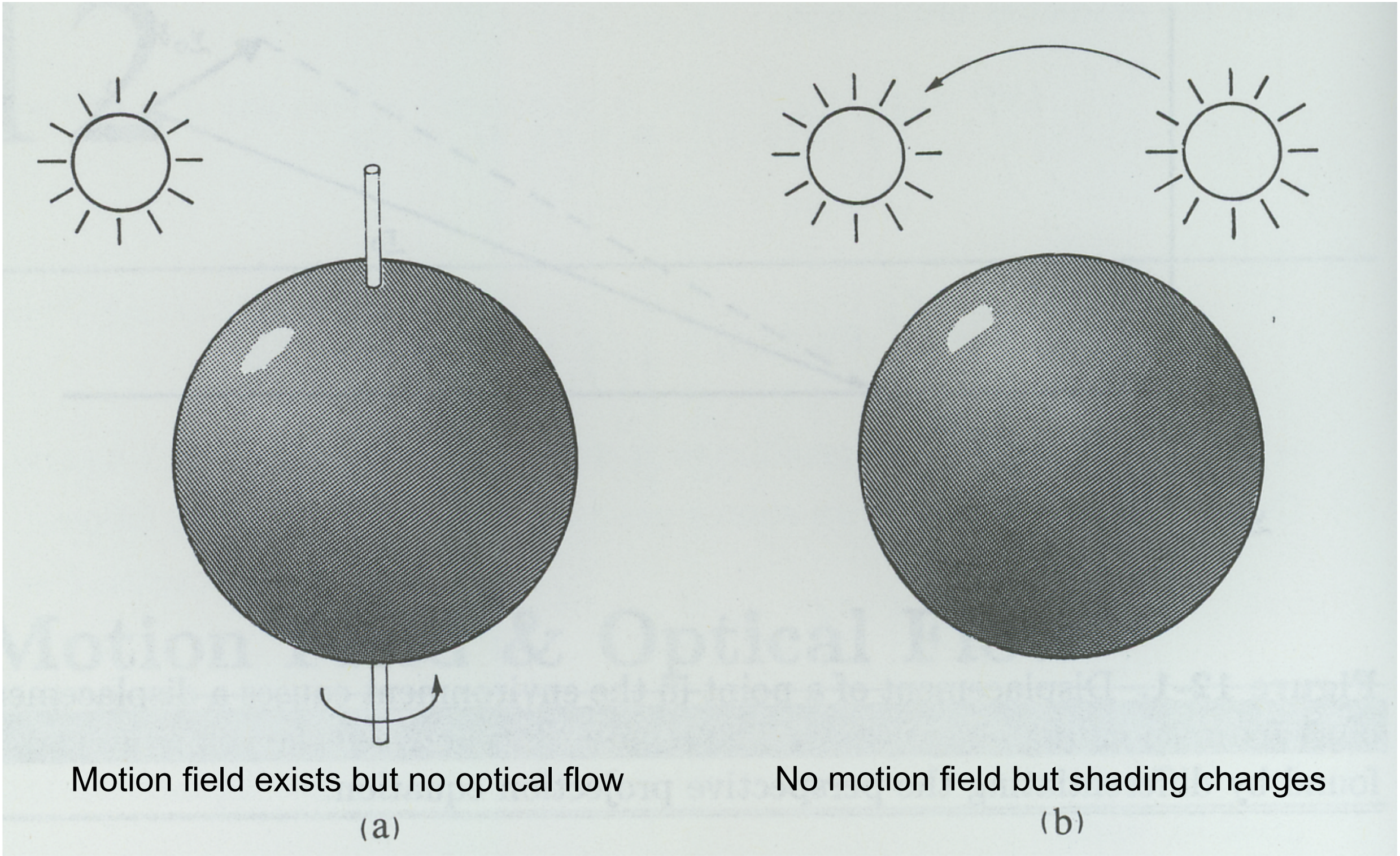
$I(x, y)$

- How to estimate pixel motion from image H to image I?
  - Find pixel correspondences
    - Given a pixel in H, look for nearby pixels of the same color in I
- Key assumption
  - Color constancy: a point in H looks "the same" in image I
  - For grayscale images, this assumption is **brightness constancy**

Where does assumption break: a point does not have the same color or brightness across frames?

Where does assumption break: a moving object does not result in any change in the image pixels?

Caution:
2D measured optical flow $\neq$ 3D scene flow



Motion field exists but no optical flow

(a)

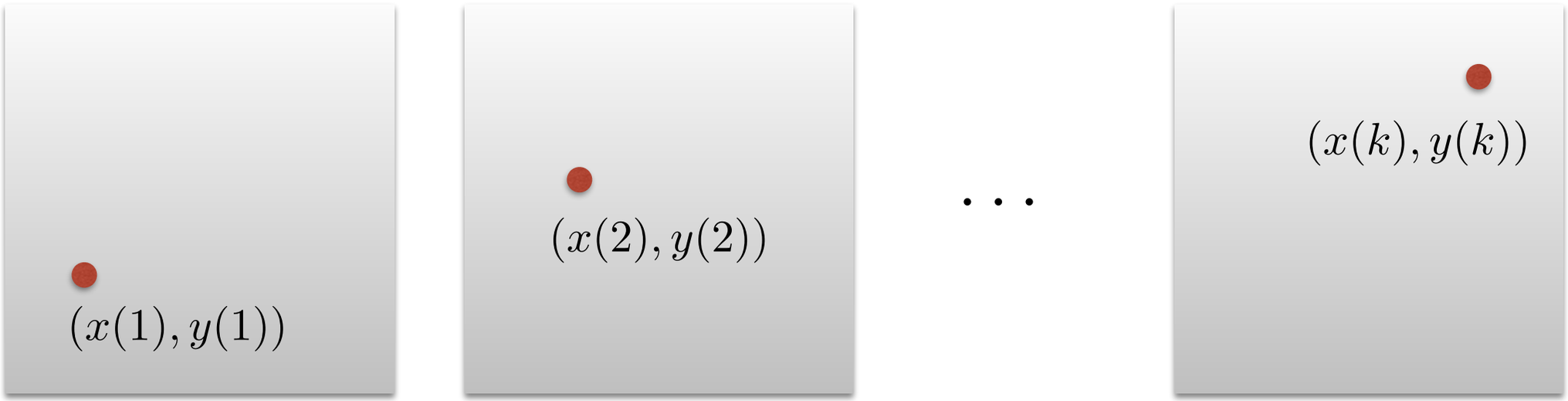
No motion field but shading changes

(b)

Assumption 1

Brightness constancy

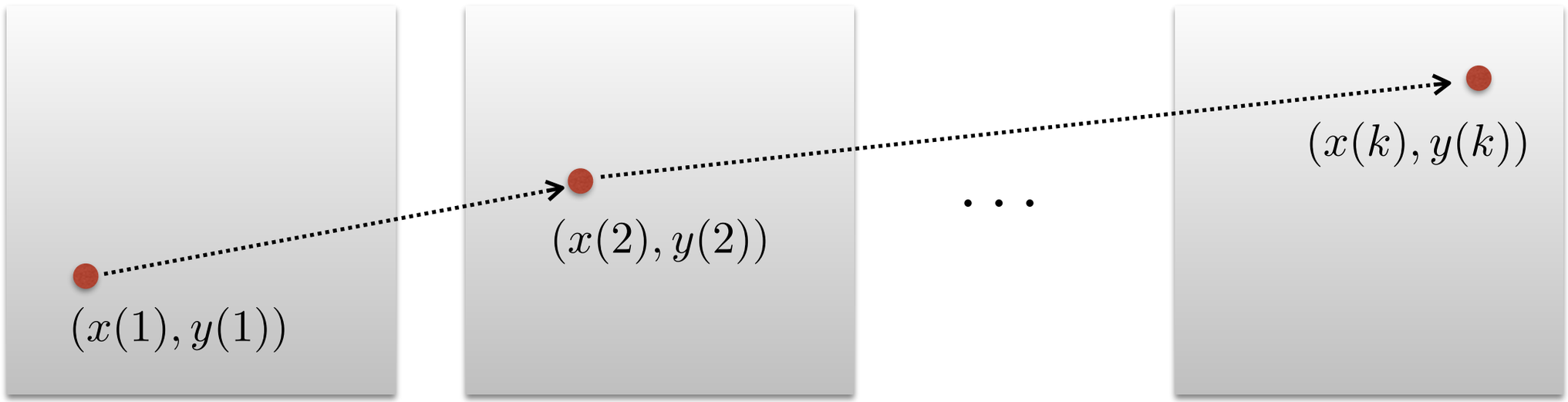
Scene point moving through image sequence



Assumption 1

Brightness constancy

Scene point moving through image sequence



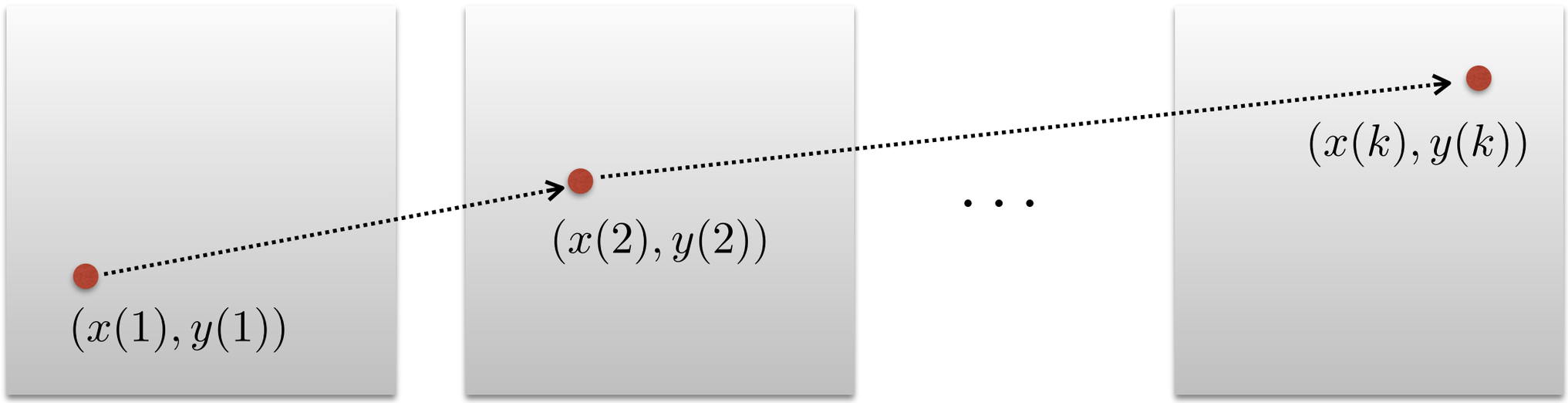
$I(x, y, 1)$ $I(x, y, 2)$ $I(x, y, k)$

Frame 1 Frame 2 Frame k

Assumption 1

Brightness constancy

Scene point moving through image sequence



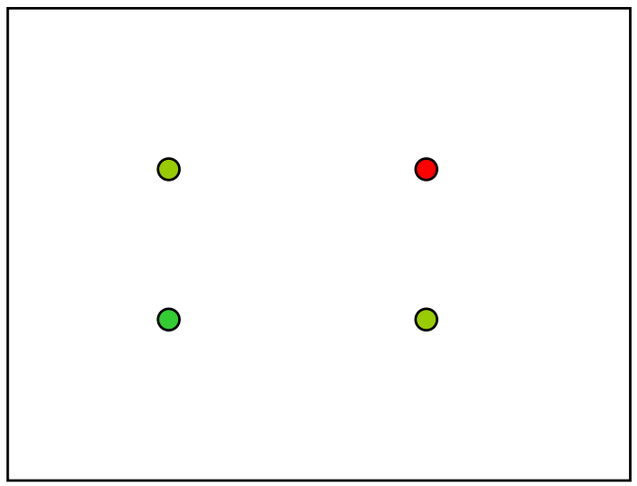
Assumption: Brightness of the point will remain the same over time

$$I(x(t), y(t), t) = C$$

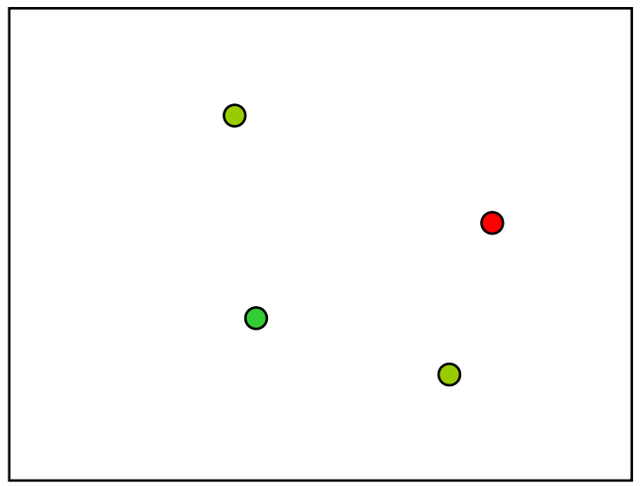
constant

Problem Definition: Optical Flow

Let's estimate the flow:



$H(x, y)$

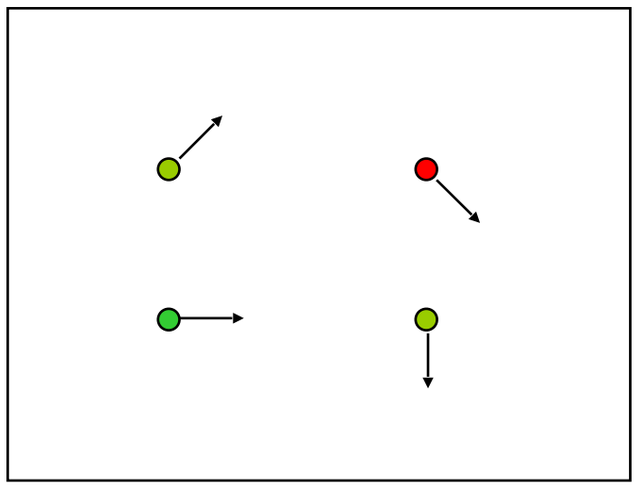


$I(x, y)$

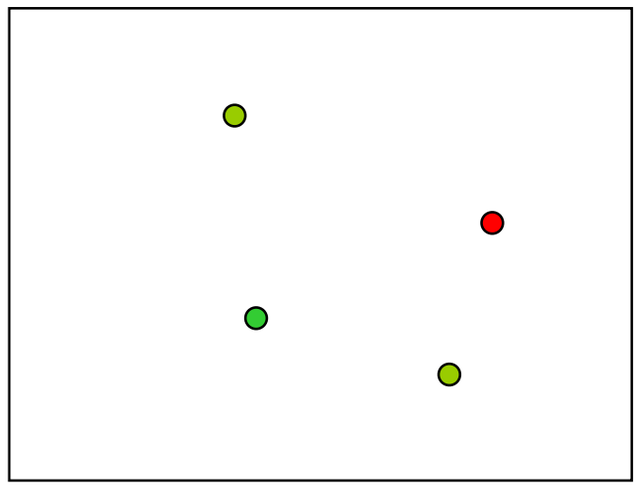
- How to estimate pixel motion from image H to image I?
  - Find pixel correspondences
    - Given a pixel in H, look for nearby pixels of the same color in I
- Key assumption
  - **color constancy**: a point in H looks "the same" in image I
    - For grayscale images, this is **brightness constancy**

Problem Definition: Optical Flow

Let's estimate the flow:



$H(x, y)$



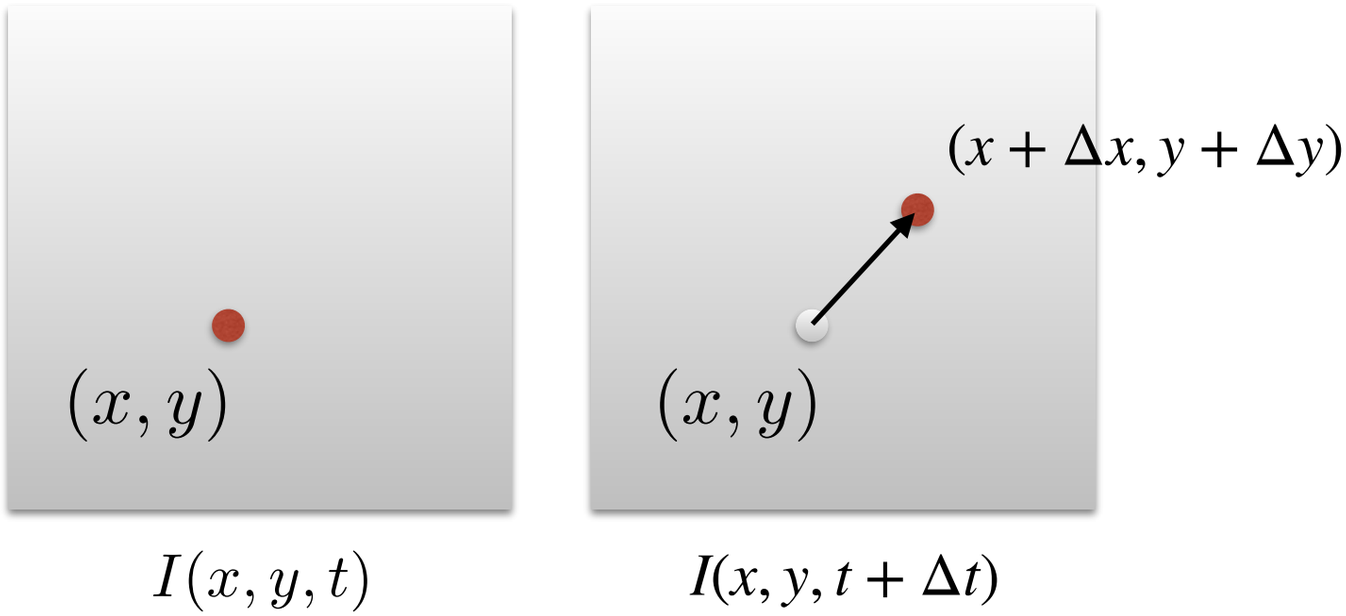
$I(x, y)$

Why did we assume this flow instead of one that rearranges the points?

- How to estimate pixel motion from image H to image I?
  - Find pixel correspondences
    - Given a pixel in H, look for nearby pixels of the same color in I
- Key assumption
  - **color constancy**: a point in H looks "the same" in image I
    - For grayscale images, this is **brightness constancy**

Assumption 2

Small motion



Assumption: Motion of point $(\Delta x, \Delta y)$ is small

Brightness constancy equations

I(x,y,t) = greyscale image intensity / brightness

$$I(x + \Delta x, y + \Delta y, t + \Delta t) - I(x, y, t) = 0$$

pixel location at time $t + \Delta t$

pixel location at time t

pixel brightness is unchanged (assumption)

Linearization (approximation):

$$\approx I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t$$

Cancel terms:

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t \approx 0$$

Define pixel velocities ("flow"):

$$u = \frac{\Delta x}{\Delta t}, v = \frac{\Delta y}{\Delta t}$$

Divide by Δt:

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0$$

Brightness constancy equation:

$$\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} + \frac{\partial I}{\partial t} = 0$$

Brightness constancy equation gives us a constraint on the flow vector (u,v); we can search for [u,v] such that this equation holds

What happens when the brightness constancy assumption does not perfectly hold?

The right hand side will be ≈ 0 but not = 0

How should we handle this? Try to find [u,v] that minimizes this expression!

Estimating Optical Flow

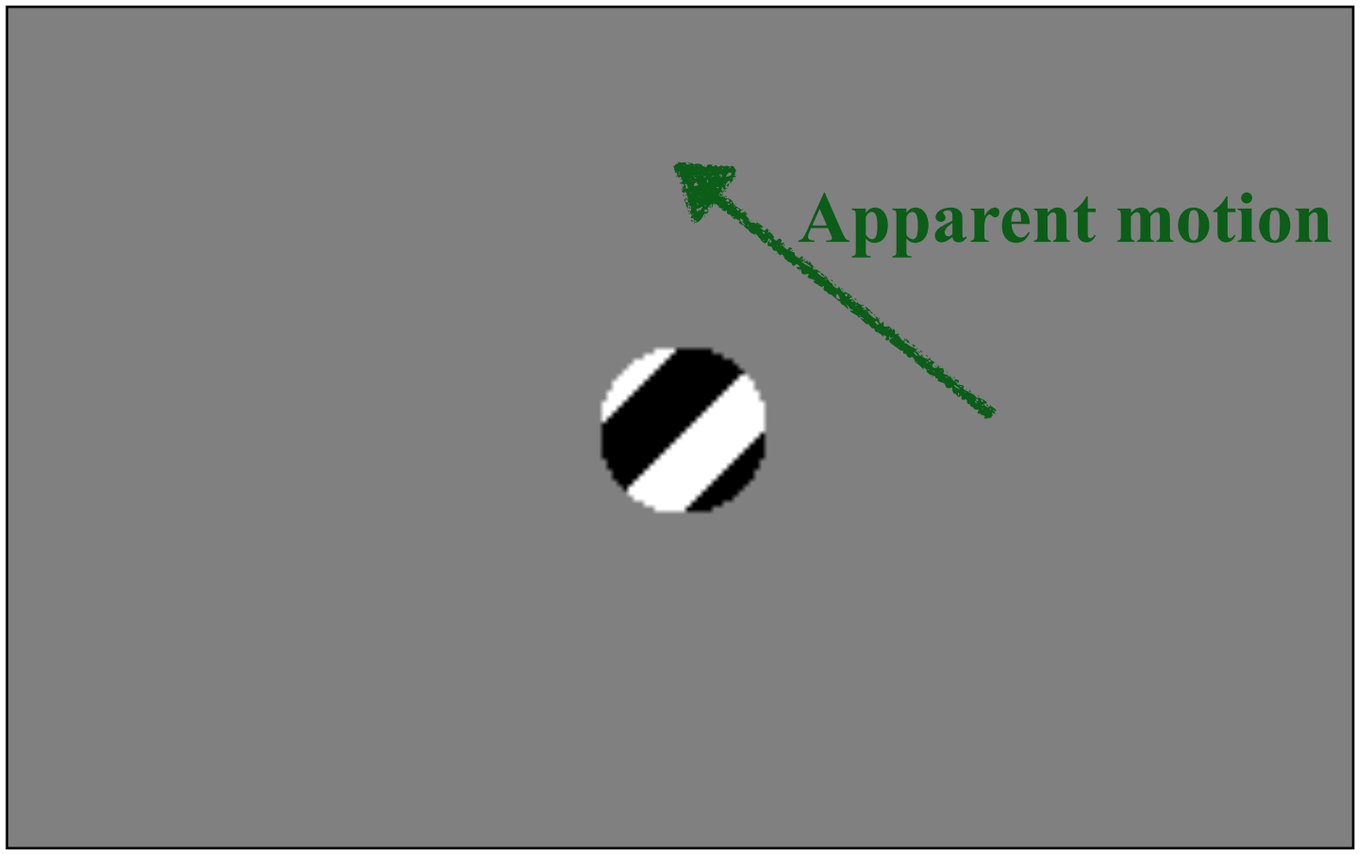
Brightness constancy equation: $\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} + \frac{\partial I}{\partial t} = 0$

image gradient / derivative

pixel velocity (“flow”)

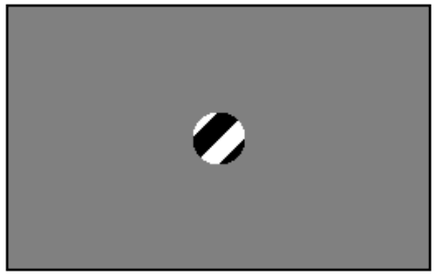
change in pixel brightness over time keeping (x,y) fixed

Aperture problem



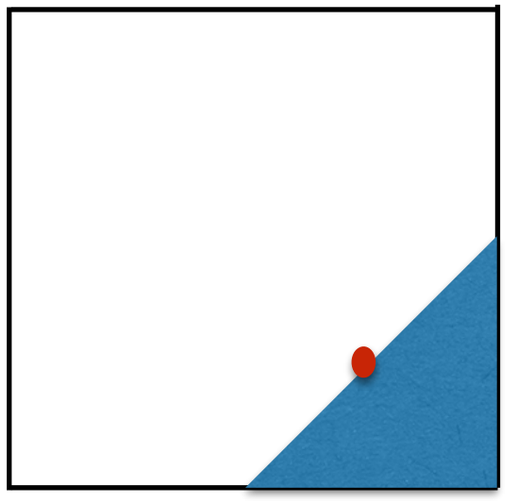
The direction of pixel motion cannot be accurately estimated from just looking at pixel values in a local window

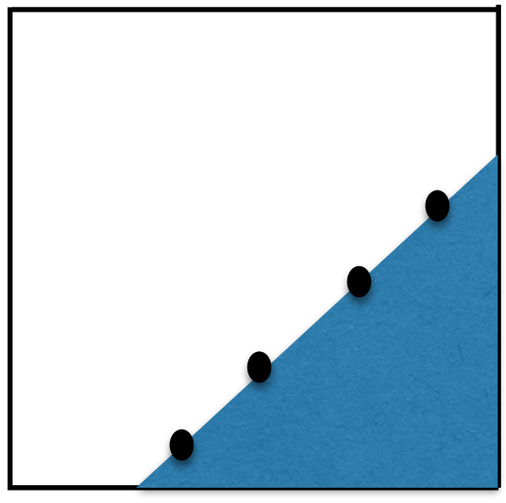
Fundamental ambiguity in optical flow estimation!



Aperture problem

A simpler example





Where did this red dot move to?

There are many possible answers that are all reasonable!

This is the fundamental ambiguity

Aperture problem

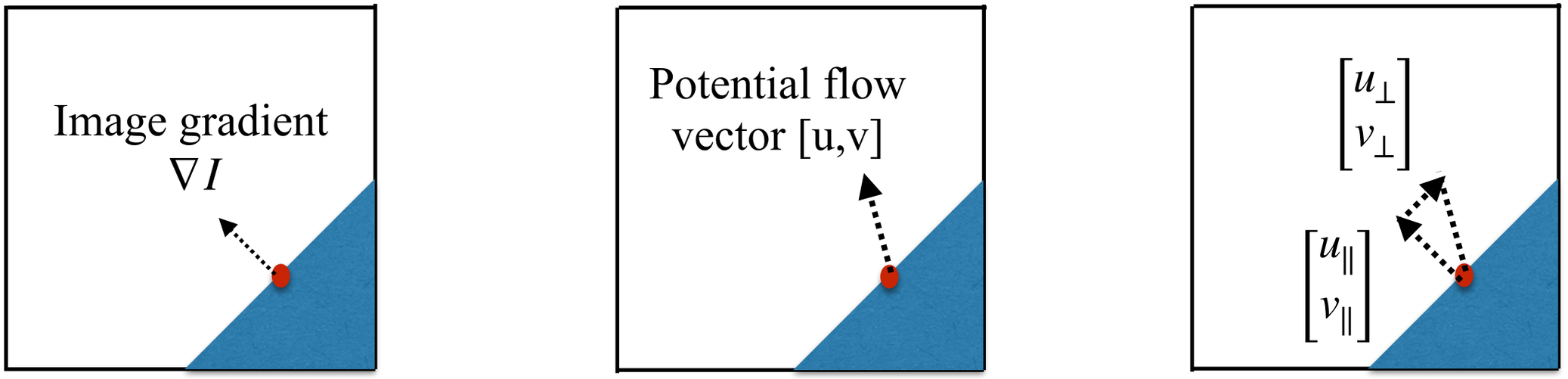
Decompose flow vector into parallel and perpendicular components:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} u_\perp \\ v_\perp \end{bmatrix} + \begin{bmatrix} u_\parallel \\ v_\parallel \end{bmatrix}$$

pixel velocity ("flow")

flow component perpendicular to image gradient

flow component parallel to image gradient



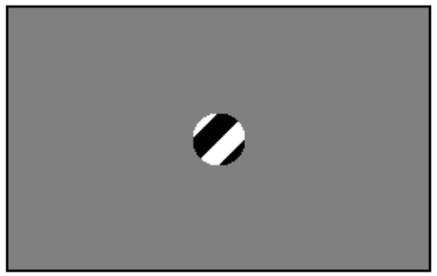
What direction of motion can we compute?

Let's plug this in to the brightness constancy equation: $\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} + \frac{\partial I}{\partial t} = 0$

$$\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} = \nabla I \cdot \begin{bmatrix} u_\perp \\ v_\perp \end{bmatrix} + \nabla I \cdot \begin{bmatrix} u_\parallel \\ v_\parallel \end{bmatrix} \qquad = \nabla I \cdot \begin{bmatrix} u_\parallel \\ v_\parallel \end{bmatrix}$$

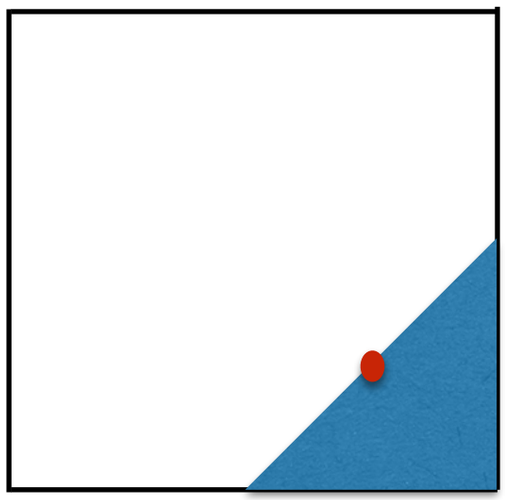
We can only estimate the component of the flow that is parallel to the image gradient! $\begin{bmatrix} u_\parallel \\ v_\parallel \end{bmatrix}$

We <u>cannot</u> estimate the component of the flow that is <u>perpendicular</u> to the image gradient

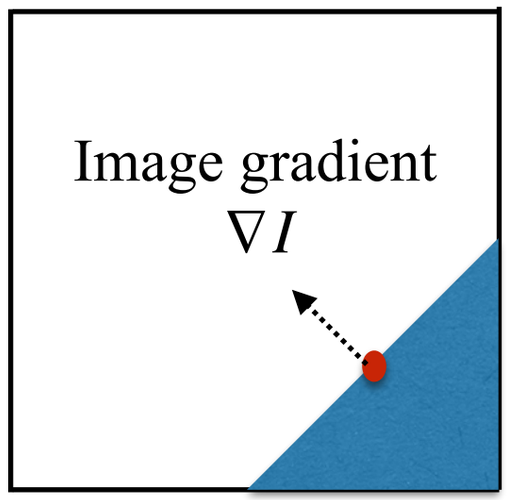


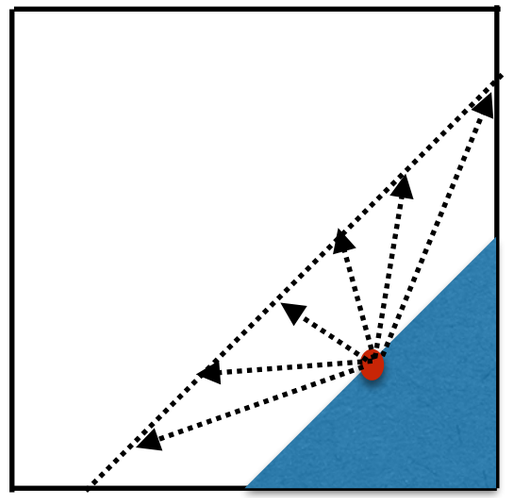
Aperture problem

Why does this occur?



Let's compute the flow of this red dot



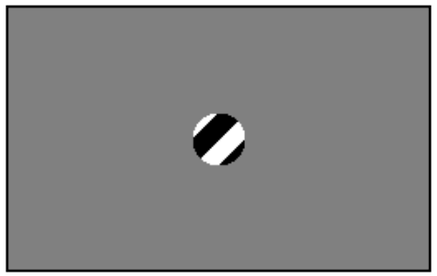


Different possible flow directions [u,v]

What do we get for $\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix}$ for these different possible values of [u,v] ? **All the same!**

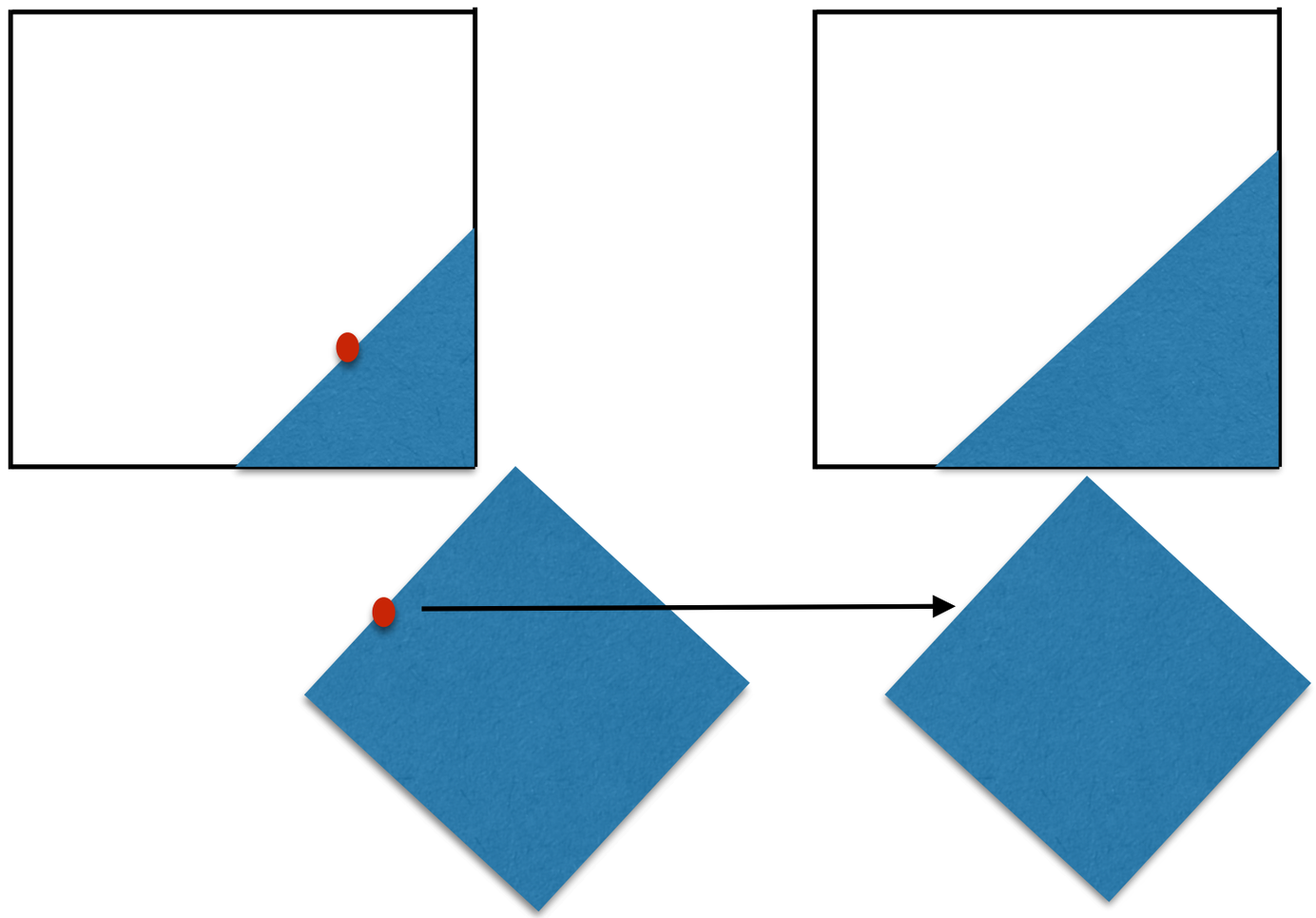
Thus we cannot determine which is these flows is correct!

Brightness constancy equation: $$\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} + \frac{\partial I}{\partial t} = 0$$



Aperture problem

Why is this called the aperture problem?



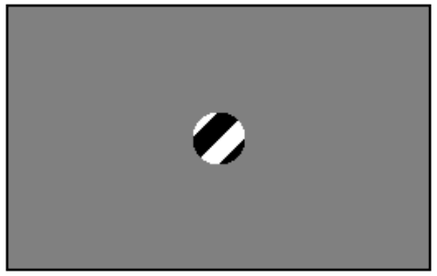
Where did this red dot move to?

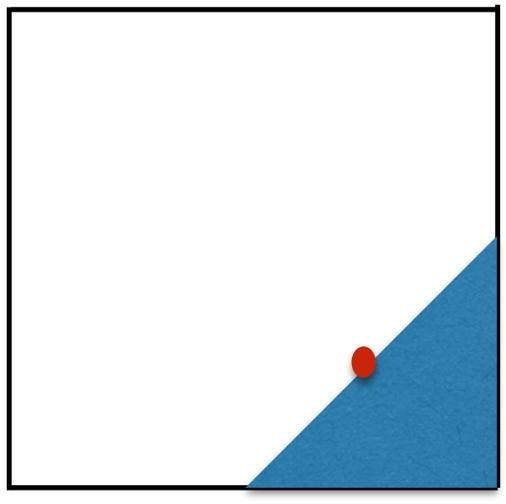
Ambiguous

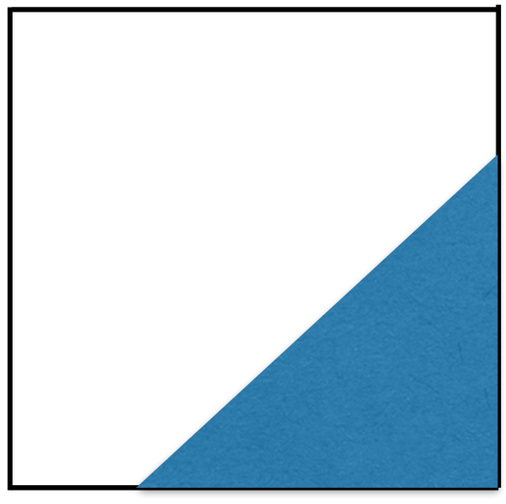
Where did this red dot move to?

Camera apertures:

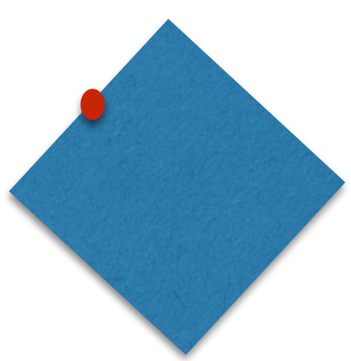
Aperture problem

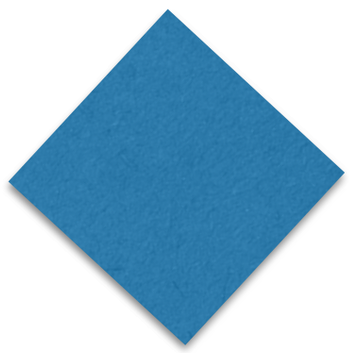






Where did this red dot move to?





Where did this red dot move to?

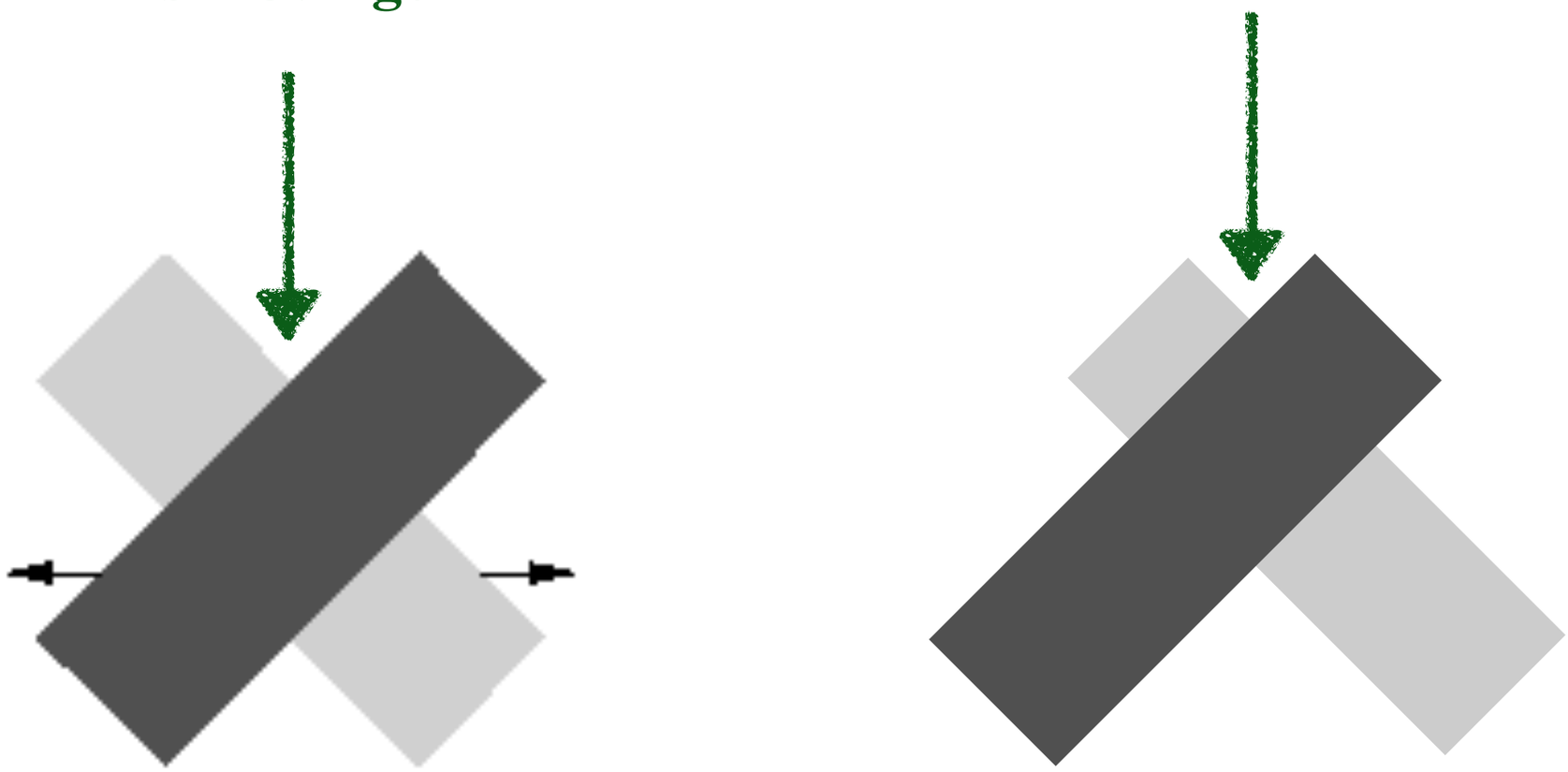
This tells us that we cannot compute flow by looking at individual points alone!

We need to look at the **larger context** of the objects in the scene

If the borders of some objects are “cut off” by the side of the image, occlusions, etc, this will create a flow ambiguity

Local motion estimation is hard

What direction does it appear that this point is moving?



We'd like to integrate motion signals *globally* instead of locally

Solutions for the aperture problem

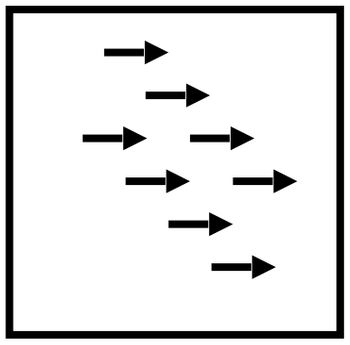
1. Assume neighboring flow vectors are similar
   - enforce *spatial smoothness* in the dense flow field ("regularization")
   - **Why is this a reasonable assumption?**

   - Camera motion causes a smooth flow field
   - The world is made of objects that move coherently

How to enforce spatial smoothness
assume flow is constant over a local patch

$$\min_{\Delta x,\Delta y} \sum_{x,y\in W} \Big(\underbrace{I(x+\Delta x, y+\Delta y, t+\Delta t)}_{\text{pixel location at time } t+\Delta t} - \underbrace{I(x,y,t)}_{\text{pixel location at time } t} \Big)^2$$

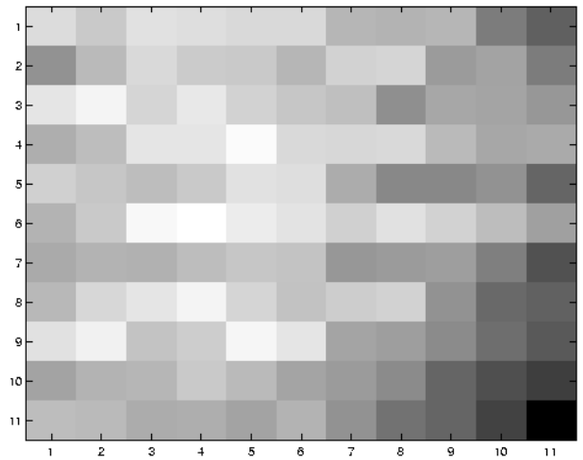
sum over pixels in a patch

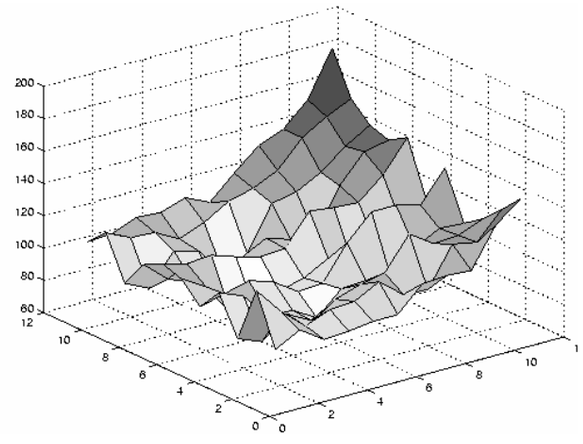


KLT (Kanade Lucas Tomasi) tracker

Does this fix the aperture problem?

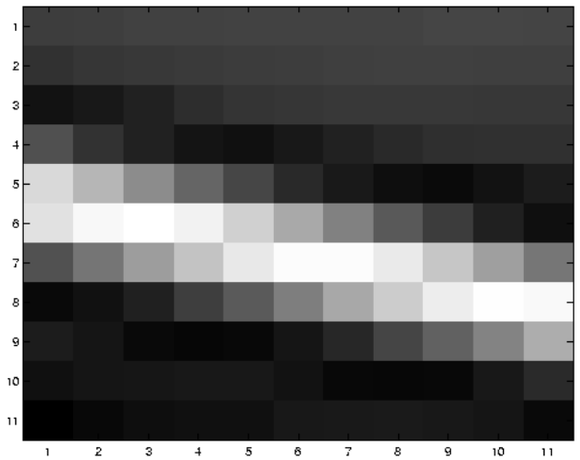
Low Texture Region - Bad

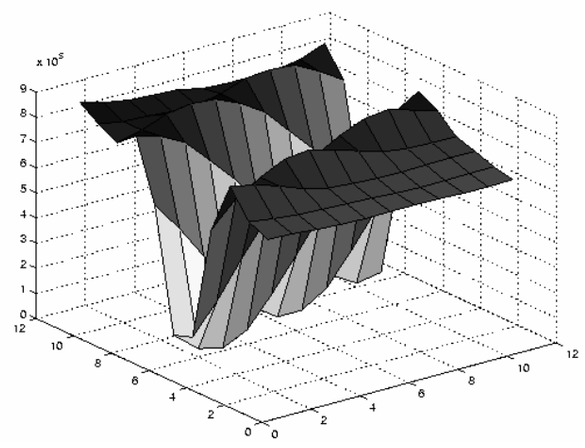




SSD surface

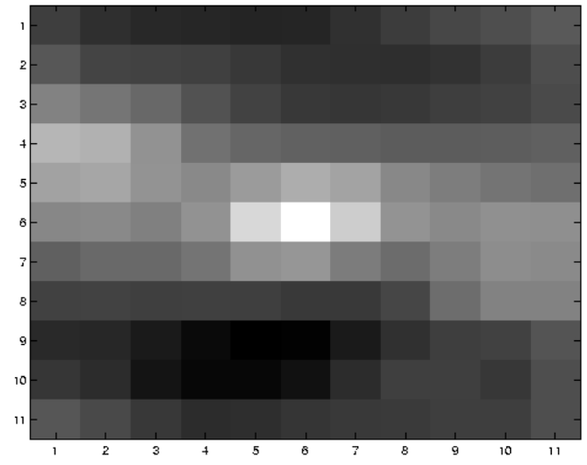
Edges - Bad (aperture problem)

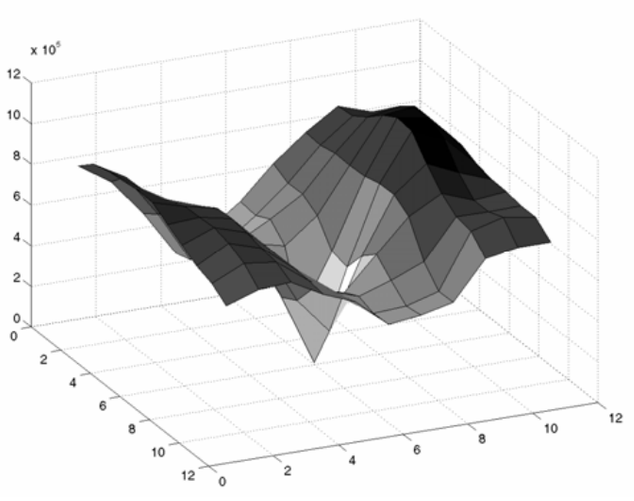




SSD surface

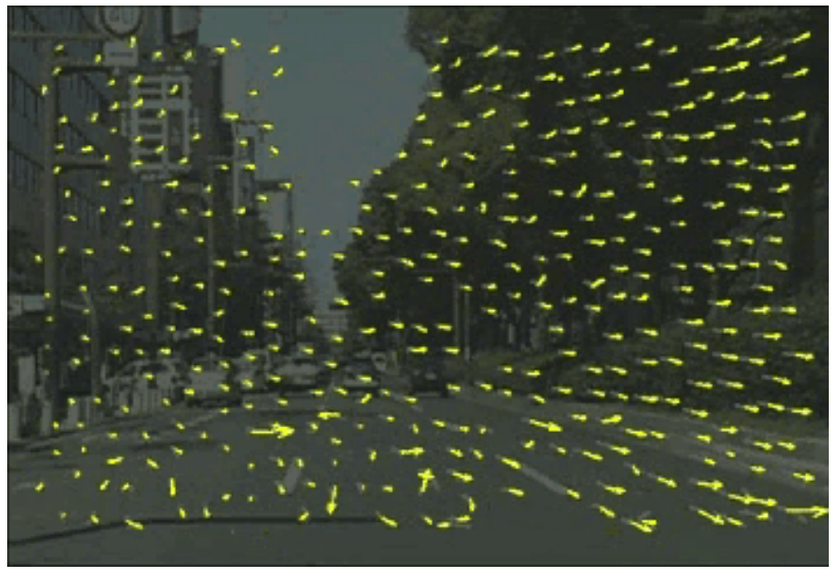
High Textured Region - Good





SSD surface

Solutions for the aperture problem



1. Assume neighboring flow vectors are similar
   - enforce *spatial smoothness* in the dense flow field ("regularization")

2. Don't try to estimate flow at unreliable points
   - only compute sparse flow
   - similar to feature point alignment of corners!

Sparse flow estimation (feature tracking)

1. User Harris corner detector to find trackable patches

$$I_2(x+u, y+v) - I_1(x,y) \approx \nabla I(x,y) \begin{bmatrix} u \\ v \end{bmatrix} + I_t(x,y)$$

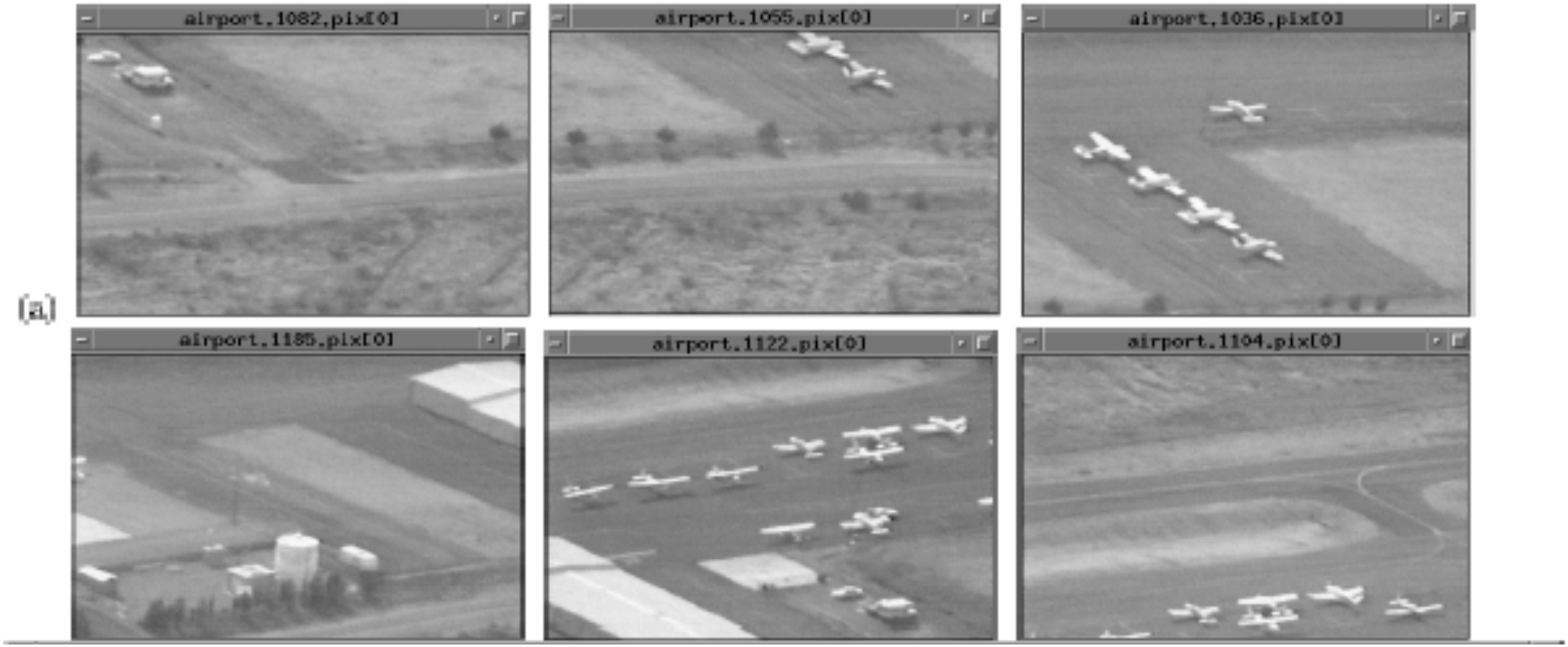
2. Apply Lucas Kanade tracking on those patches

Good Features to Track

Jianbo Shi
Computer Science Department
Cornell University
Ithaca, NY 14853

Carlo Tomasi
Computer Science Department
Stanford University
Stanford, CA 94305

Dense optical flow



Estimate warp parameters over successive frames of a video sequence (with Lucas Kanade optimization)

Applications: mosaicing

Homography warp works for some cases (rotations, planar scenes)

Examples: aerial video (e.g. from a plane or drone)
The world is mostly planar from these distances

Dense flow

Instead of estimating warp parameters, let's compute a dense flow vector for every point!

Dense flow

Instead of estimating warp parameters, let's compute a dense flow vector for every point!

sum over all points in the image

Brightness constancy: the change of intensity of each point should be as small as possible

$$\min_{\substack{u(x,y)\\v(x,y)}} \sum_{x,y} [I_2(x+u(x,y), y+v(x,y)) - I_1(x,y)]^2$$

pixel location of the same point at time 2

pixel location of a point at time 1

flow for each point

Solve for the global flow field:

Dense flow

Instead of estimating warp parameters, let's compute a dense flow vector for every point!

sum over all points in the image

Brightness constancy: the change of intensity of each point should be as small as possible

$$\min_{\substack{u(x,y)\\v(x,y)}} \sum_{x,y} [I_2(x+u(x,y), y+v(x,y)) - I_1(x,y)]^2$$

pixel location of the same point at time 2

pixel location of a point at time 1

flow for each point

Aside: continuous flow fields
(instead of computing the flow per-pixel)

$$\min_{u,v} \int\int \big(I_2(x+u, y+v) - I_1(x,y)\big)^2 dx dy$$

Dense *variational* flow

If we assume small motions….

$$I_2(x+u, y+v) - I_1(x,y) \approx \nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} + I_t$$

Change in brightness

linearization

image gradient

flow

$\frac{\partial I}{\partial t}$

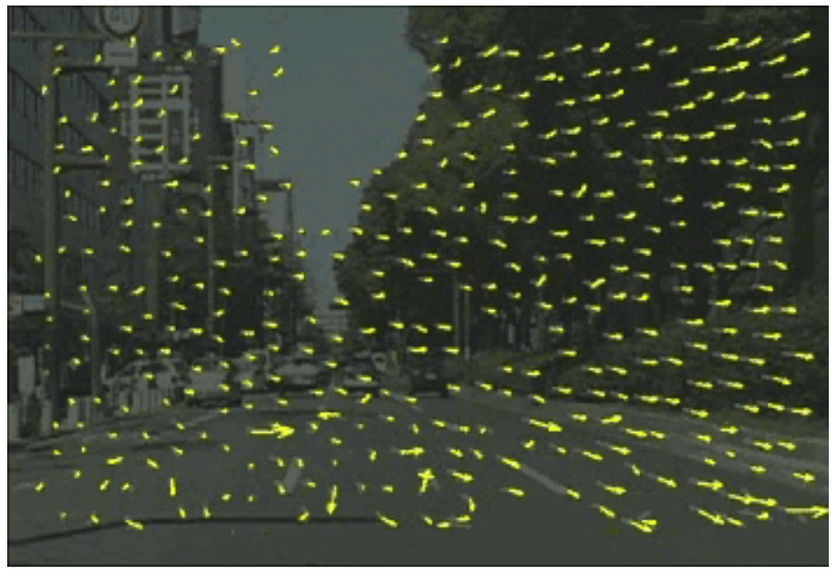
Discrete version:

$$\min_{\substack{u(x,y) \\ v(x,y)}} \sum_{x,y} \left[\nabla I(x,y) \cdot \begin{bmatrix} u(x,y) \\ v(x,y) \end{bmatrix} + I_t(x,y) \right]^2$$

Continuous version:

$$\min_{u,v} \int \int (\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} + I_t)^2 dx dy$$

Solutions for the aperture problem



1. Assume neighboring flow vectors are similar
   - enforce *spatial smoothness* in the dense flow field ("regularization")

2. Don't try to estimate flow at unreliable points
   - only compute sparse flow
   - similar to feature point alignment of corners!

Spatial regularization

Optimizing for each flow vector independently will lead to very noisy flow estimates

Penalize differences in nearby flow vectors

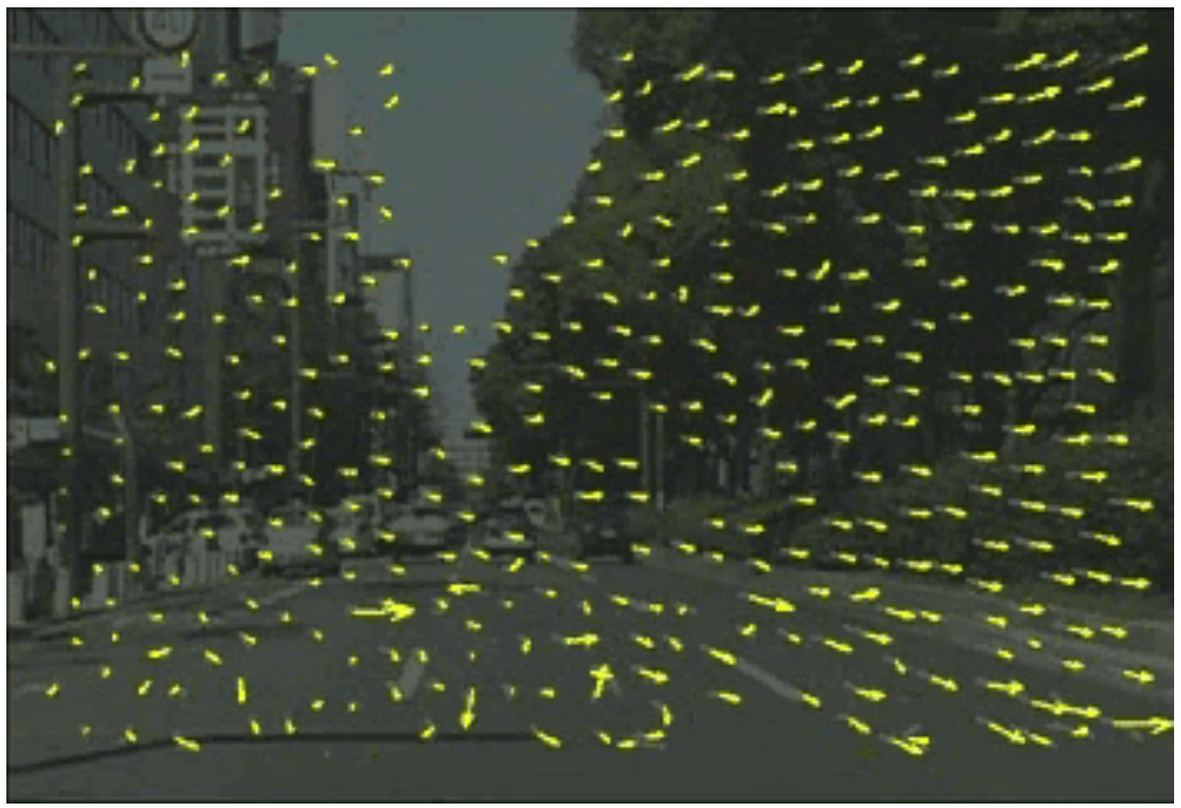
$$\min_{u,v} E_{\text{intensity}} + E_{\text{smooth}}$$

$$E_{intensity}(u,v) = \int\int (\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} + I_t)^2 dxdy$$

$$E_{smooth}(u,v) = \int\int ||\nabla u||^2 + ||\nabla v||^2 dxdy$$

gradient of the flow field

Change in flow across neighboring pixels should be small



Spatial regularization

Optimizing for each flow vector independently will lead to very noisy flow estimates

Penalize differences in nearby flow vectors

$$\min_{u,v} E_{\text{intensity}} + E_{\text{smooth}}$$

$$E_{intensity}(u,v) = \int\int (\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} + I_t)^2 dxdy$$

$$E_{smooth}(u,v) = \int\int ||\nabla u||^2 + ||\nabla v||^2 dxdy$$

gradient of the flow field

Change in flow across neighboring pixels should be small

Unknowns (u,v) appear quadratically in above expression

Solve for (u,v) by taking the derivative and setting equal to 0
This gives us a big linear system of equations!
Solve as a giant least-squares problem, solve with $x = A^{-1}b$

Challenges with regularization

https://en.wikipedia.org/wiki/Horn-Schunck_method

$$\min_{u,v} E_{intensity} + E_{smooth}$$

estimated flow ground-truth flow

Challenge: We don't actually expect smooth flow everywhere in the image

Why not?

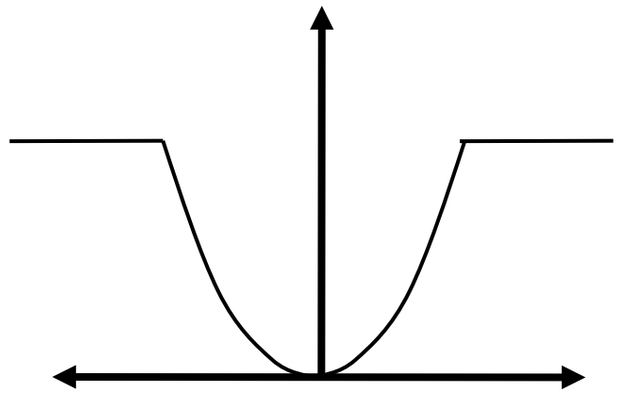
There should be flow boundaries at object boundaries

Result of naively using the above method: flow is overly smooth

Let's treat object boundaries as "outliers"

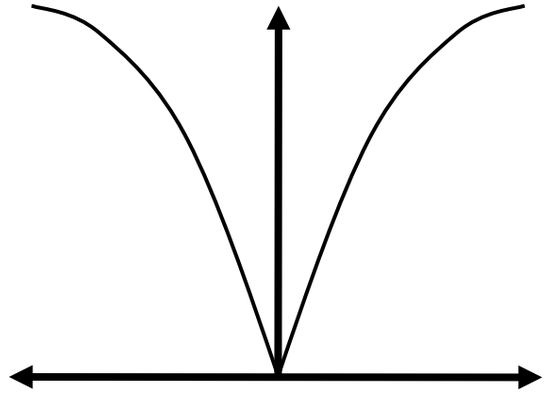
In statistics, what are some classic ways to handle outliers?

Robust error functions to handle “outliers”



truncated quadratic

$$\rho_{\alpha,\lambda}(x) = \begin{cases} \lambda x^2 & \text{if } |x| < \frac{\sqrt{\alpha}}{\sqrt{\lambda}} \\ \alpha & \text{otherwise} \end{cases}$$



lorentzian

$$\rho_\sigma(x) = log\left(1 + \frac{1}{2}(\frac{x}{\sigma})^2\right)$$

Robust variational optical flow

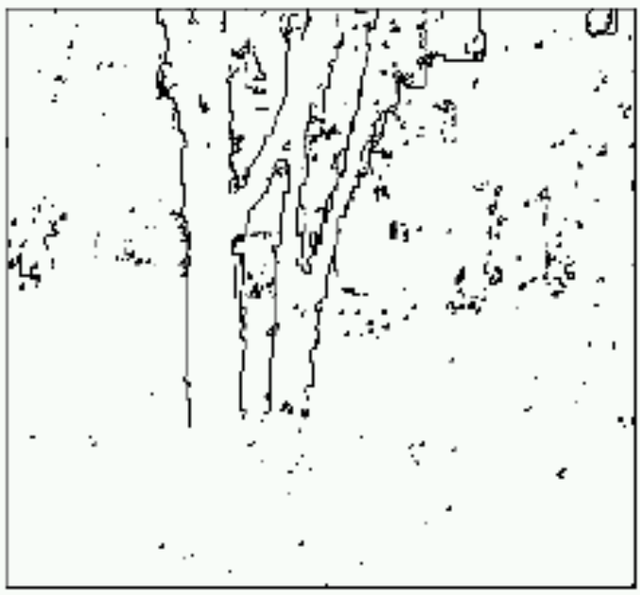
$$\min_{u,v} \int\int \rho(I_2(x+u, y+v) - I_1(x,y)) + \rho(||\nabla u||) + \rho(||\nabla v||) dx dy$$

where ρ = robust error function (instead of quadratic error)

first image

quadratic error

lorentzian error



detected outliers

Problem: not squared error so this will not be just $x = A^{-1}b$

How do we optimize this?

Typical approach: initialize solution with solution to quadratic error and then locally optimize the robust error function ρ with gradient descent

- Black, M. J. and Anandan, P., A framework for the robust estimation of optical flow, *Fourth International Conf. on Computer Vision* (ICCV), 1993, pp. 231-236 http://www.cs.washington.edu/education/courses/576/03sp/readings/black93.pdf