Computer Vision I

Optical Flow

A Stereo Vision System

Calibration
(offline)

Rectification

Stereo Correspondences

Triangulation

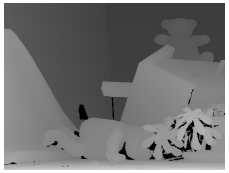
PC, FPGA

Intrinsic and extrinsic parameter

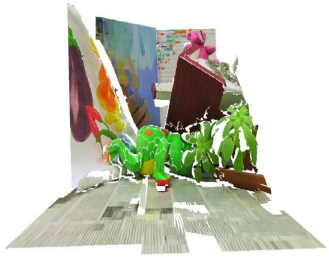
Stereo pair

Rectified Stereo pair

Disparity map

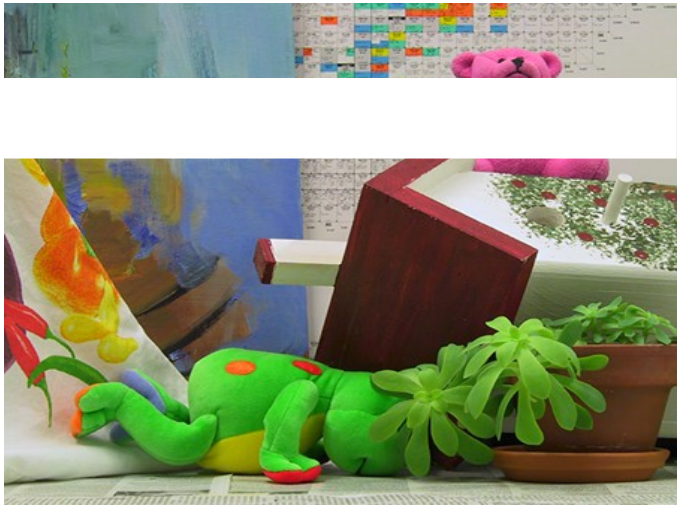


Depth map



Improvising Stereo Cameras

Improvising Stereo Cameras

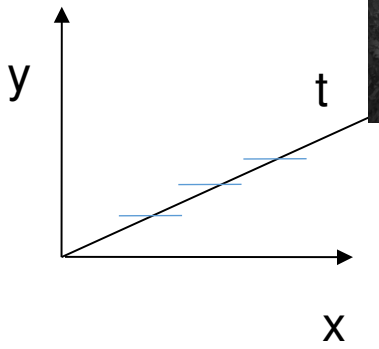


Improvising Stereo Cameras

Improvising Stereo Cameras

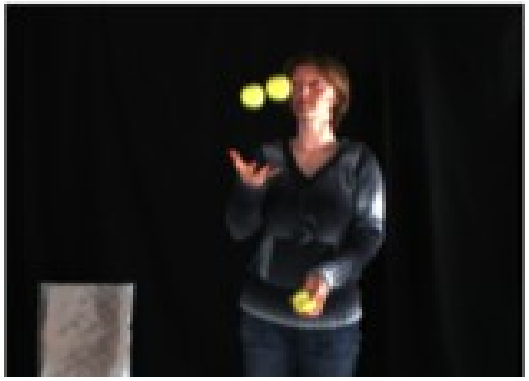
Video Cameras

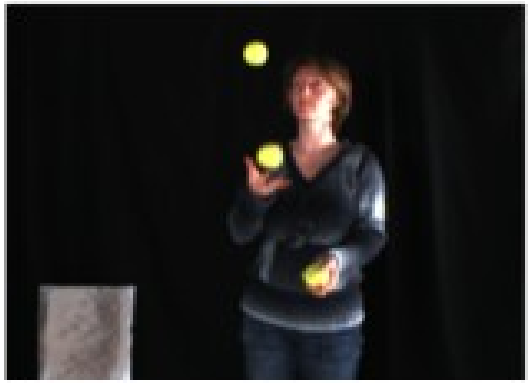
- 25-30 frames per second



Temporal Sampling

- Images



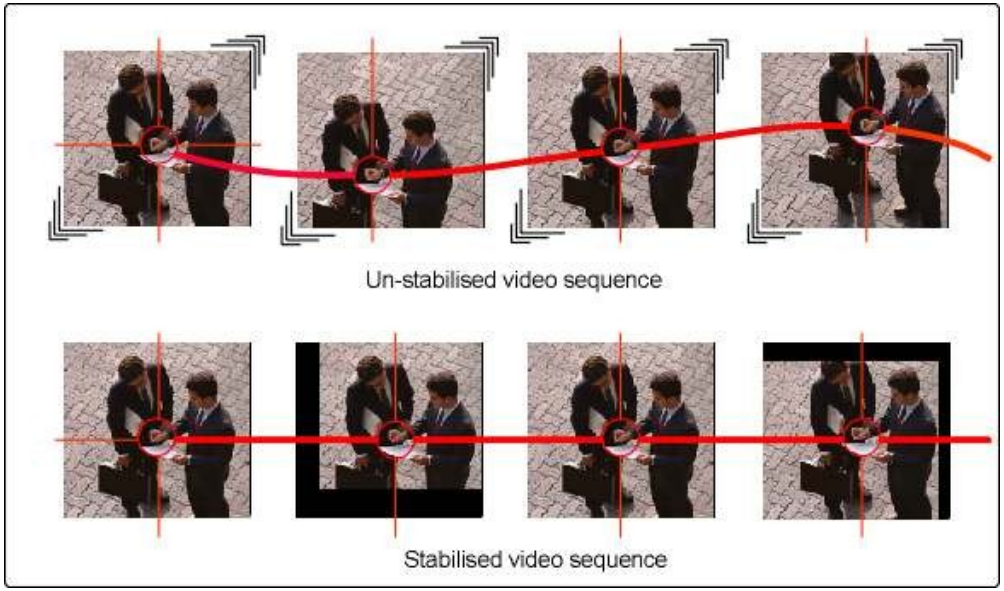


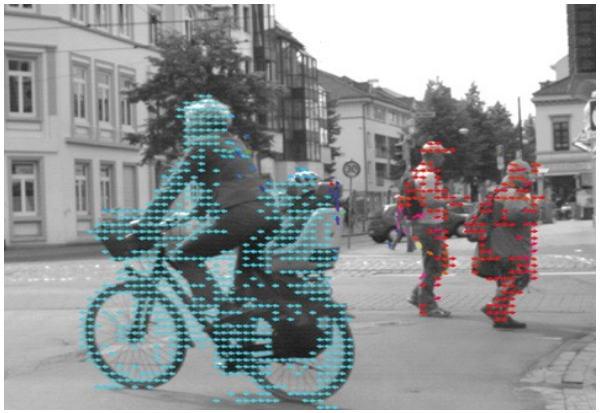
- Time

Applications

- Frame Interpolation
- Compression
- Video stabilization
- Inpainting
- Superresolution
- Pedestrian recognition
- …

?

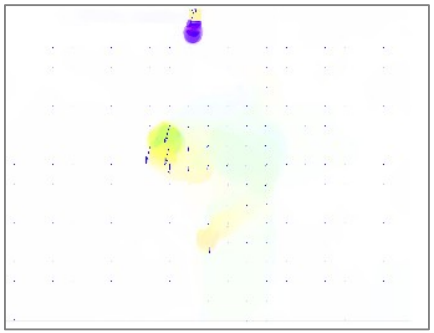




Optical Flow Estimation in Computer Vision

- Input: at least 2 frames

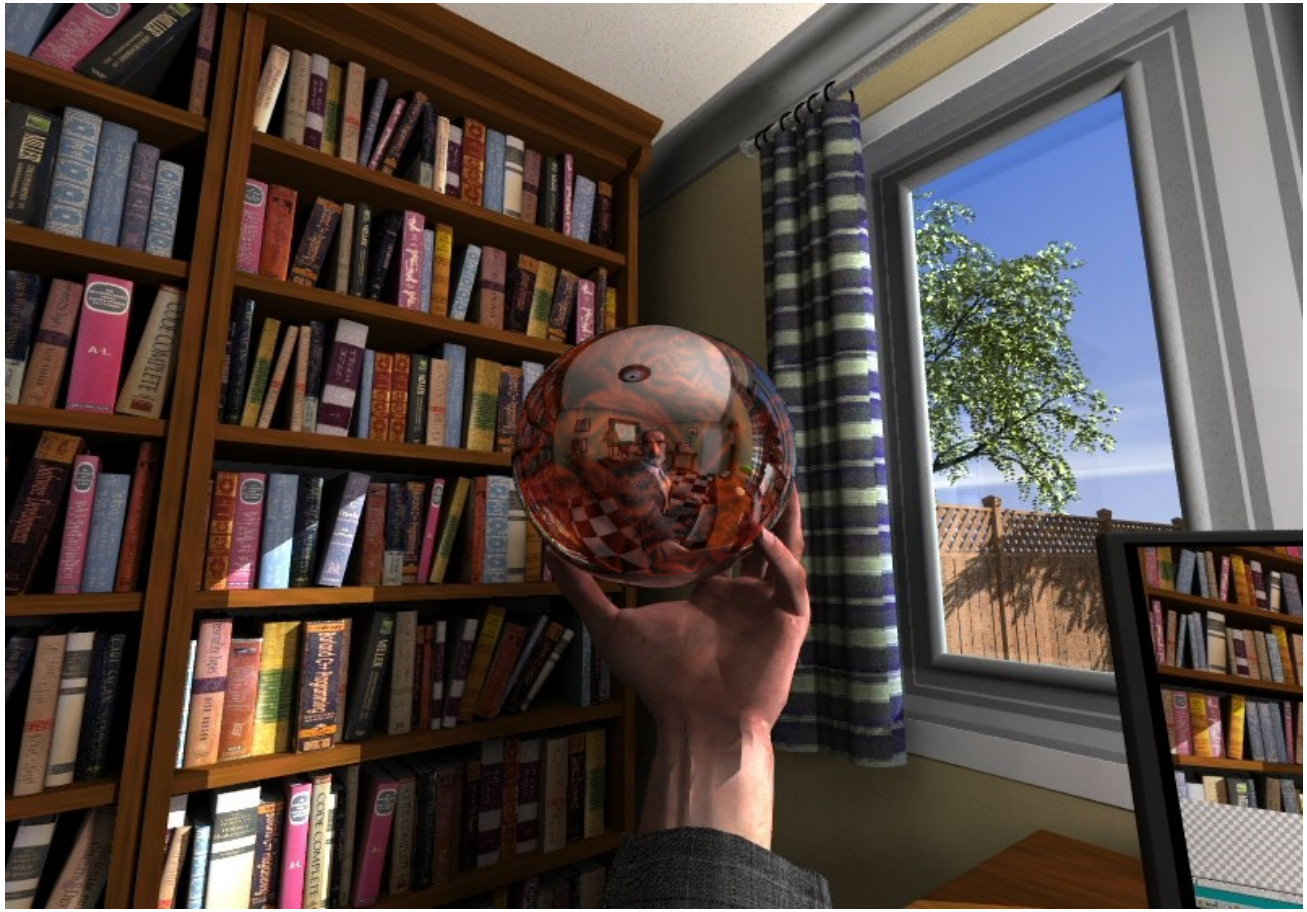
- Wanted: dense motion field



- Most general setup: multiple moving objects, unknown camera

Optical Flow Estimation: Assumption I

- The color of a pixel does not change between frames



What is the Optical Flow of this Object?

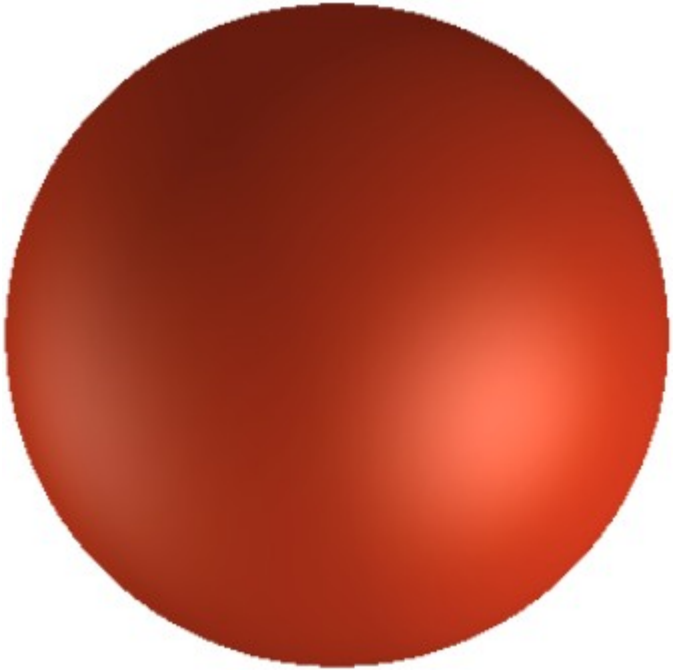
- Perfect sphere undergoing 3D rotation
- Material reflects equally in all directions (Lambertian)

1: 2D projection of the motion

2: static

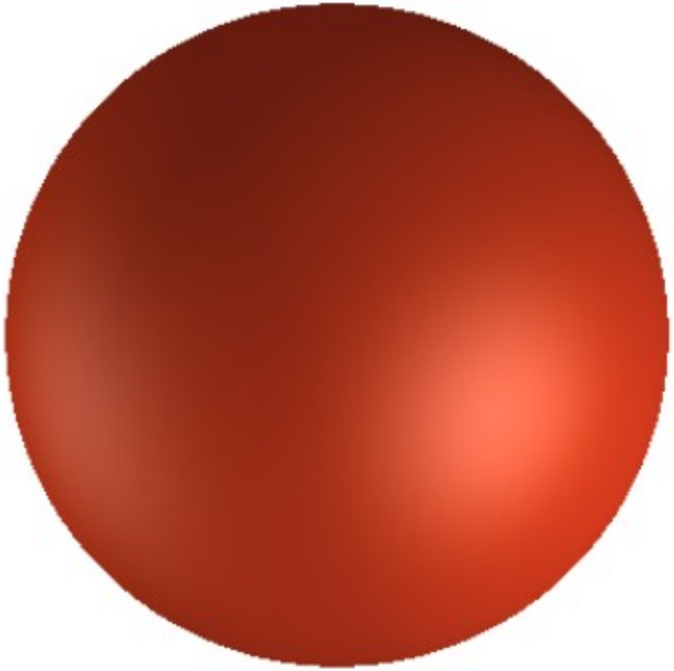
3: random

4: don't know



What is the Optical Flow of this Object?

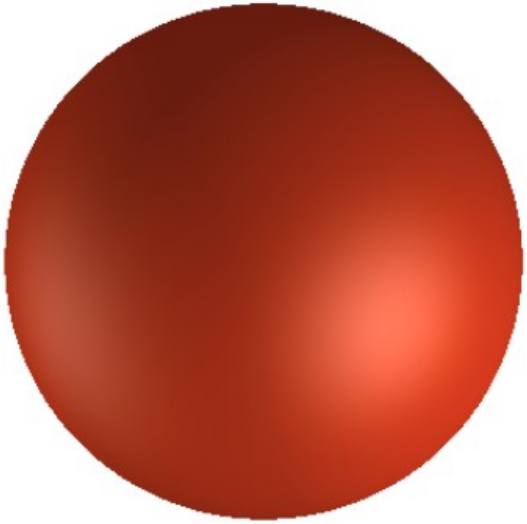
- Perfect sphere, static
- Material reflects equally in all directions (Lambertian)
- Lightsource moves



- 2D motion field
- Optical flow field

What is the Optical Flow of this Object?

- Many objects have at least some texture
- Often, texture dominates the effects of light



Optical Flow Assumption I

- The color of a pixel does not change between frames

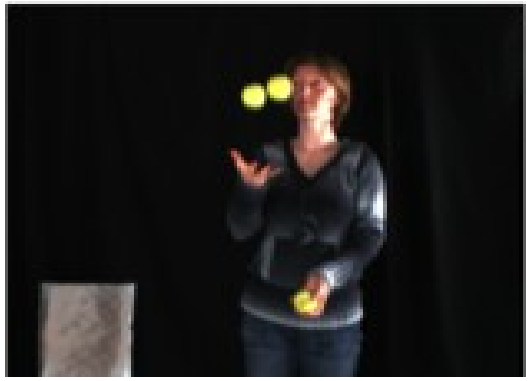


Image domain $\Omega \subset \mathbb{R}^2$

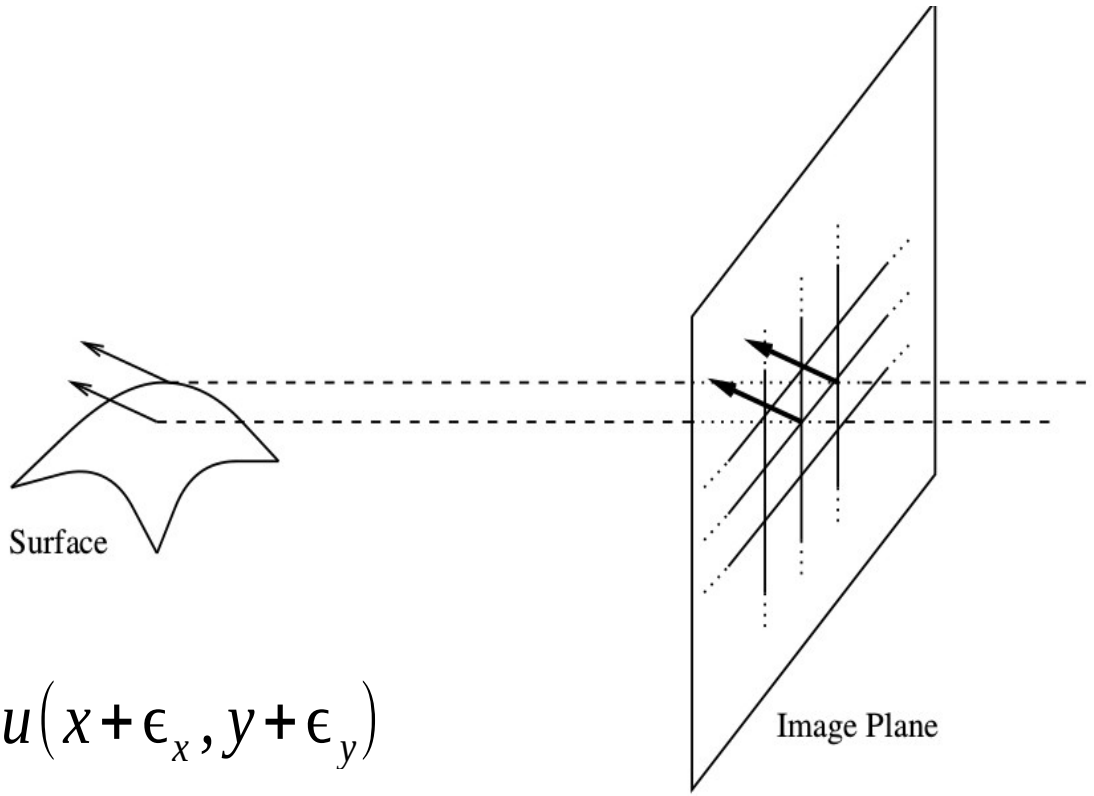
Image sequence $I : \Omega \times \mathbb{R}^+ \rightarrow \mathbb{R}^+, (x, y, t) \rightarrow I(x, y, t)$

Optical Flow $\vec{u} : \Omega \times \mathbb{R}^+ \rightarrow \mathbb{R}^2, (x, y, t) \rightarrow u(x, y, t)$

$$I(x, y, 0) \approx I(x + u(x, y, 0), y + v(x, y, 0), 1)$$

Optical Flow Assumption II(a)

- Neighbouring scene points belong to the same surface
- Neighbouring pixels move in similar directions



$$u(x,y) \approx u(x+\epsilon_x, y+\epsilon_y)$$

Consider Image Patches

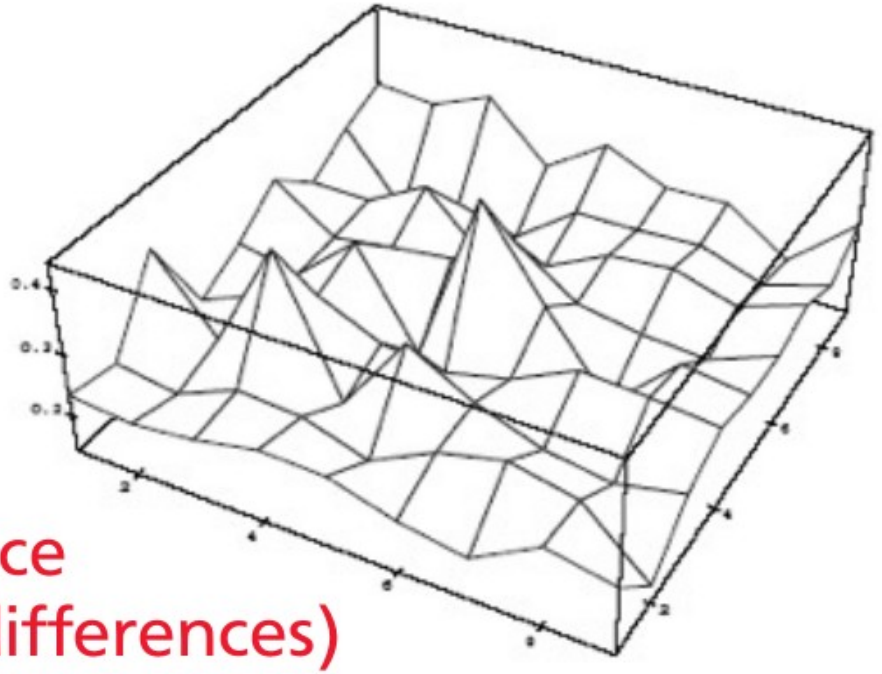
- Assume an entire image patch moves with **constant** motion
- Minimize brightness difference for a patch

$$u(x,y) = u_R$$

$$E_{SSD}(x,y) = \sum_{(x,y)\in R} (I(x+u, y+v, t+1) - I(x,y,t))^2$$

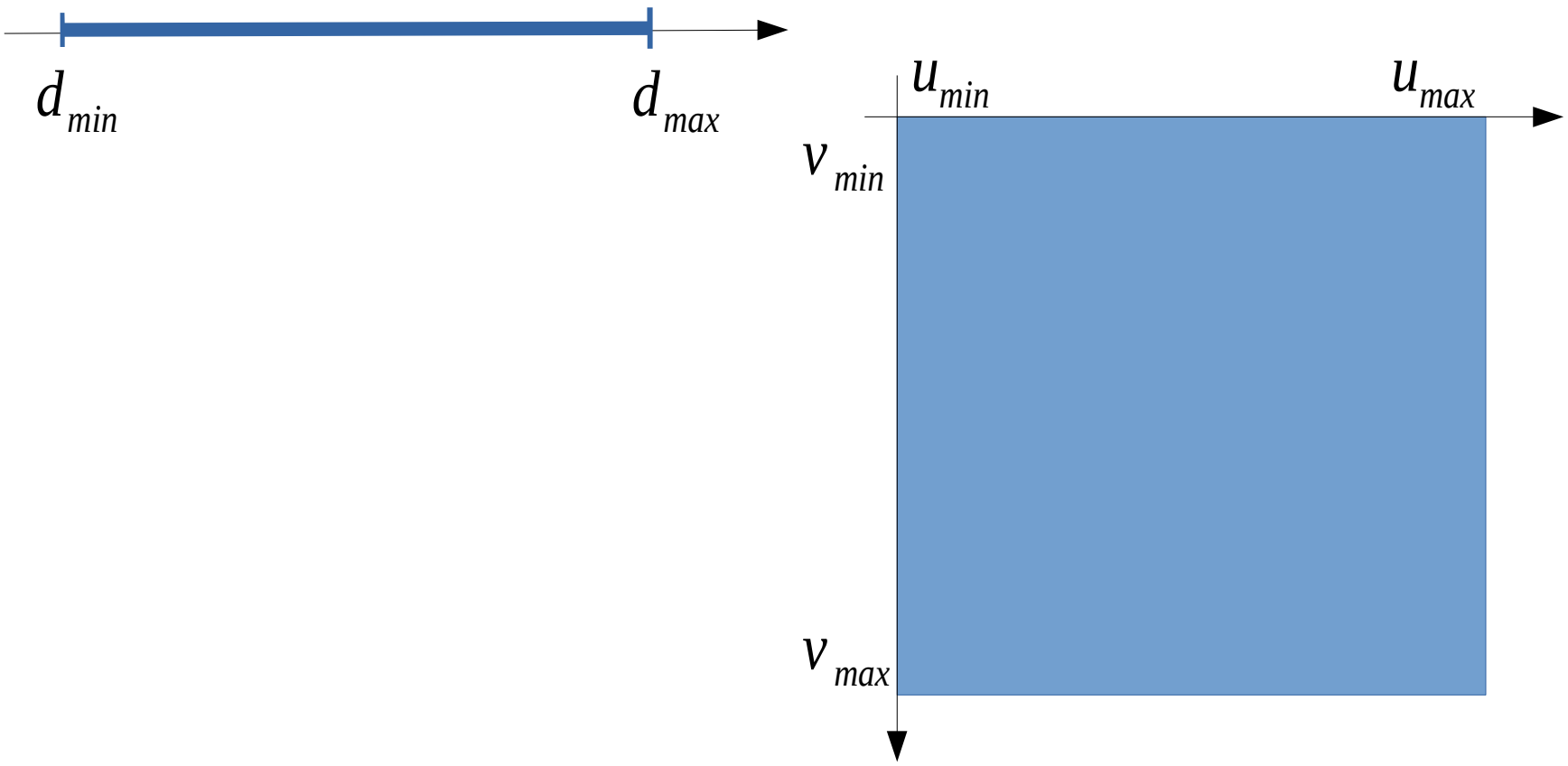
Bloch Matching in 2D

- For each pixel
  - For each possible 2D motion (in a range)
    - Compare image patches
    - Greedy pointwise optimization



SSD surface
(sum of squared differences)

Discretization in 2D



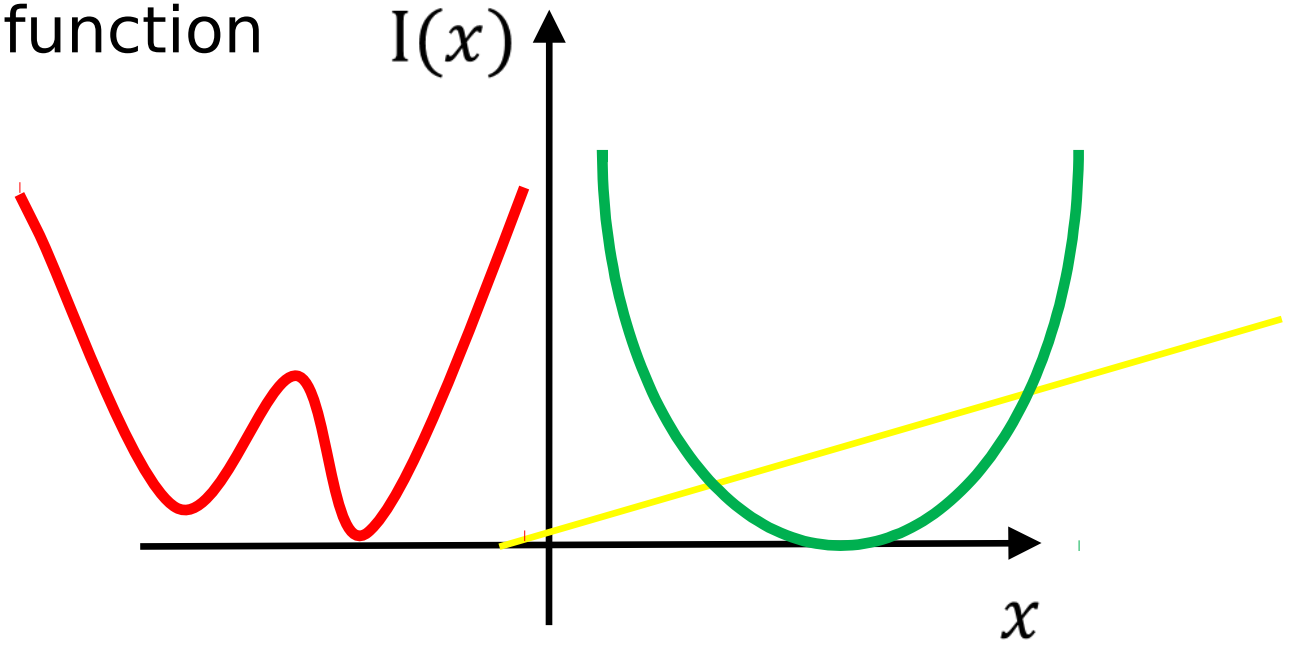
Optical Flow as Optimization Problem

$$E_{SSD}(u,v) = \sum_{(x,y)\in R} (I(x+u, y+v, t+1) - I(x,y,t))^2$$

Optical Flow as Optimization Problem

$$E_{SSD}(u,v) = \sum_{(x,y)\in R} (I(x+u, y+v, t+1) - I(x,y,t))^2$$

- Non-linear function
- Non-convex function



Linearize the Image Function

$$E_{SSD}(u,v) = \sum_{(x,y)\in R} \left(I(x+u, y+v, t+1) - I(x,y,t)\right)^2$$

Let us look at the general case:

$$I(x+\Delta_x, y+\Delta_y, t+\Delta_t)$$

Taylor series approximation:

$$I(x,y,t) + \Delta_x \frac{\partial}{\partial x} I(x,y,t) + \Delta_y \frac{\partial}{\partial y} I(x,y,t) + \Delta_t \frac{\partial}{\partial t} I(x,y,t) + \epsilon(\Delta_x^2, \Delta_y^2, \Delta_t^2)$$

Approximation error

Linearize the Image Function

$$E_{SSD}(u,v) = \sum_{(x,y)\in R} \left(I(x+u, y+v, t+1) - I(x,y,t)\right)^2$$

Assume small motion

Brightness varies smoothly

$$I(x,y,t) + u\frac{\partial}{\partial x}I(x,y,t) + v\frac{\partial}{\partial y}I(x,y,t) + \frac{\partial}{\partial t}I(x,y,t) + \epsilon(u,v,1)$$

SSD approximation:

$$\begin{aligned} E_{SSD}(u,v) &\approx \sum_{(x,y)\in R} \left(I(x,y,t) + u\frac{\partial}{\partial x}I(x,y,t) + v\frac{\partial}{\partial y}I(x,y,t) + \frac{\partial}{\partial t}I(x,y,t) - I(x,y,t)\right)^2 \\ &= \sum_{(x,y)\in R} \left(u\frac{\partial}{\partial x}I(x,y,t) + v\frac{\partial}{\partial y}I(x,y,t) + \frac{\partial}{\partial t}I(x,y,t)\right)^2 \\ &= \sum_{(x,y)\in R} \left(u\cdot I_x(x,y,t) + v\cdot I_y(x,y,t) + I_t(x,y,t)\right)^2 \end{aligned}$$

Linearize the Image Function

$$E_{SSD}(u,v) \approx \sum_{(x,y)\in R} \left(u \cdot I_x(x,y,t) + v \cdot I_y(x,y,t) + I_t(x,y,t)\right)^2$$

- The approximation is a convex function of the motion u and v. (we will see how to optimize this shortly)
- It holds only for small motions.
- Practically, image derivatives are approximated by finite differences.

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

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

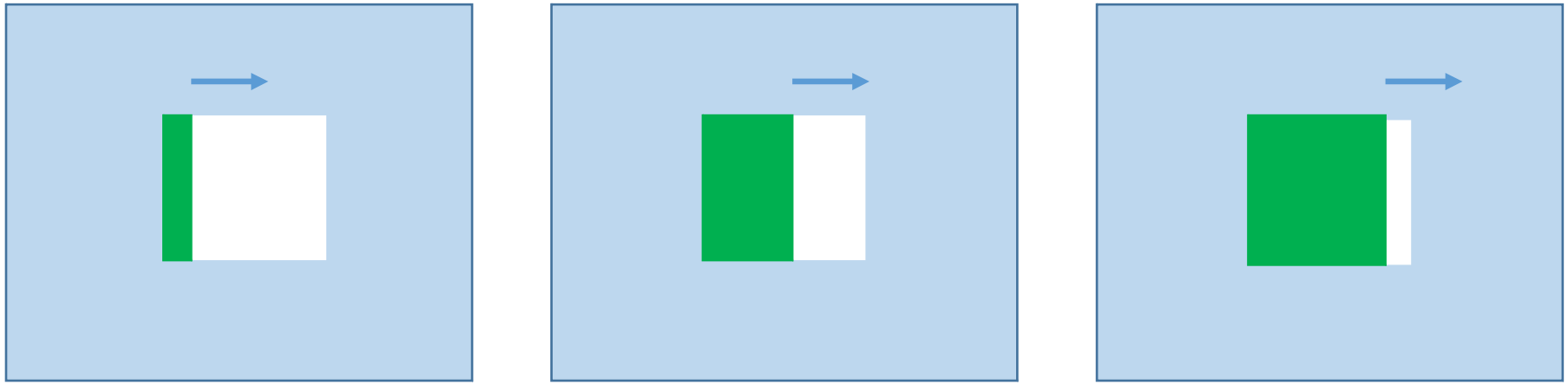
$$\mathbf{u} = \begin{pmatrix} u \\ v \end{pmatrix} \qquad \nabla I = \begin{pmatrix} I_x \\ I_y \end{pmatrix}$$

Optical Flow Constraint Equation

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

The Aperture Problem

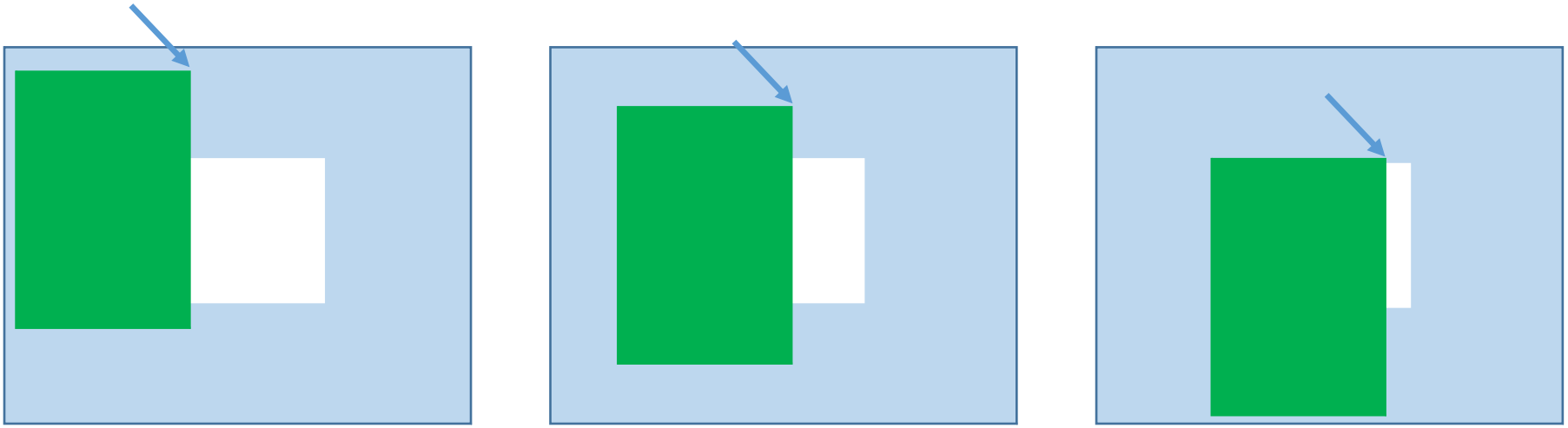
- Given a patch, the motion can only be estimated perpendicular to the image gradient



$$I_y = 0 \implies v \text{ indetermined}$$

The Aperture Problem

- Given a patch, the motion can only be estimated perpendicular to the image gradient



$$I_y = 0 \implies v \text{ indetermined}$$

Optimization

$$E_{SSD}(u,v) \approx \sum_{(x,y)\in R} \left(u \cdot I_x(x,y,t) + v \cdot I_y(x,y,t) + I_t(x,y,t)\right)^2$$

Optimization

$$E_{SSD}(u,v) \approx \sum_{(x,y)\in R} \big(u \cdot I_x(x,y,t) + v \cdot I_y(x,y,t) + I_t(x,y,t)\big)^2$$

Differentiate with respect to u and v.

Set to zero.

$$\frac{\partial}{\partial u} E_{SSD}(u,v) \approx 2 \sum_{(x,y)\in R} \big(u \cdot I_x(x,y,t) + v \cdot I_y(x,y,t) + I_t(x,y,t)\big) I_x(x,y,t) = 0$$

$$\frac{\partial}{\partial v} E_{SSD}(u,v) \approx 2 \sum_{(x,y)\in R} \big(u \cdot I_x(x,y,t) + v \cdot I_y(x,y,t) + I_t(x,y,t)\big) I_y(x,y,t) = 0$$

Optimization

$$\left[\sum_R I_x^2\right] u + \left[\sum_R I_x I_y\right] v = -\left[\sum_R I_x I_t\right]$$

$$\left[\sum_R I_x I_y\right] u + \left[\sum_R I_y^2\right] v = -\left[\sum_R I_y I_t\right]$$

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

Symmetric positive definite 2x2 matrix
“structure tensor”

Lucas Kanade Optical Flow

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

$$\mathbf{u} = -\left(\sum_R \nabla I \nabla I^{\mathrm{T}} \right)^{-1} \left(\sum_R I_t \nabla I \right)$$

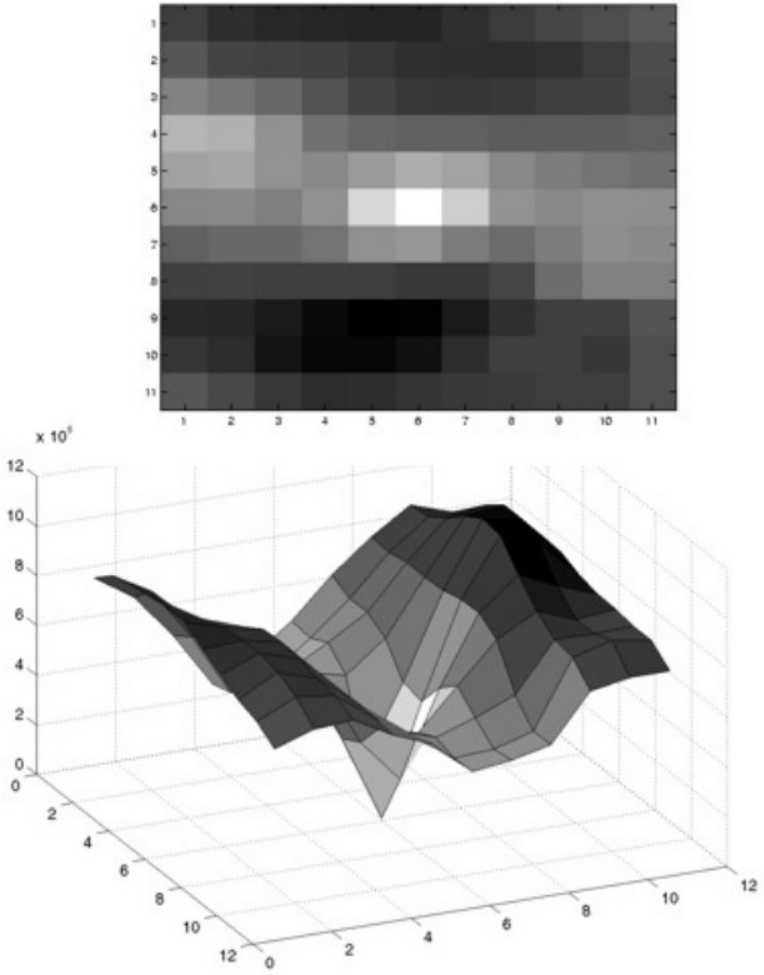
- A classic optical flow technique:

  B.D. Lucas and T. Kanade: An iterative image registration technique
  with an application to stereo vision. IJCA, pp. 674-679, 1981

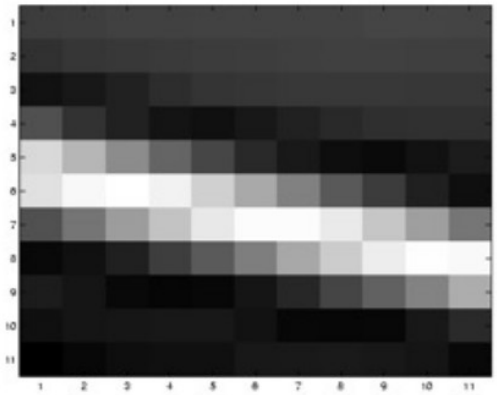
SSD Surface – Textured Area

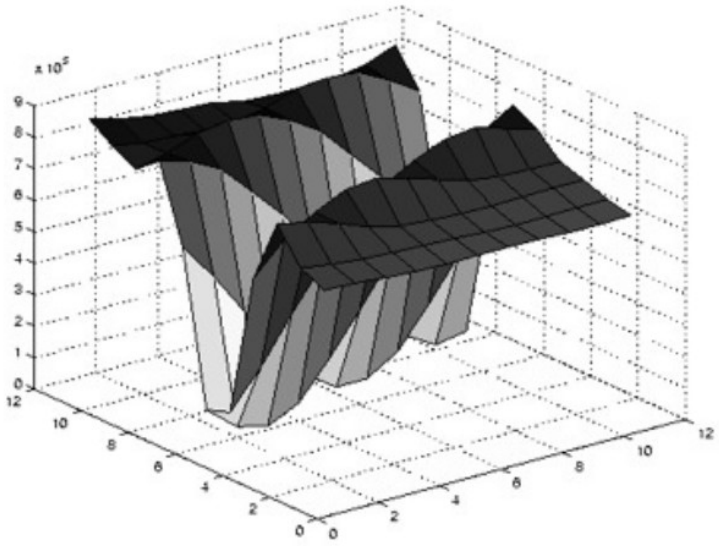
$$\begin{bmatrix} \sum_R I_x^2 & \sum_R I_x I_y \\ \sum_R I_x I_y & \sum_R I_y^2 \end{bmatrix}$$

Gradients in x and y.



SSD Surface – Single Edge

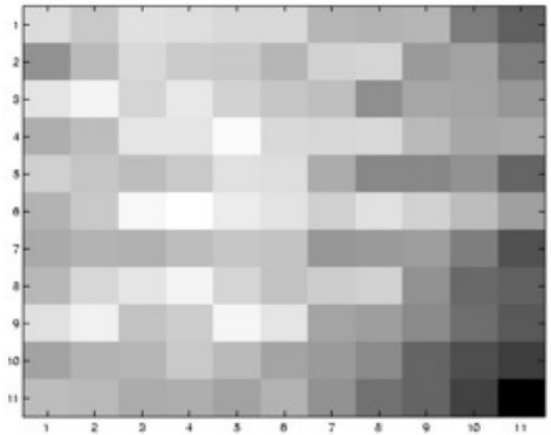


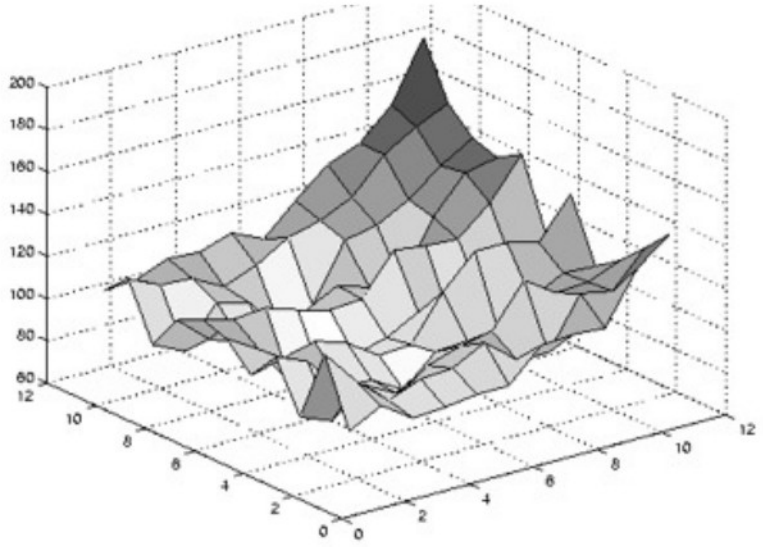


$$\begin{bmatrix} \sum_R I_x^2 & \sum_R I_x I_y \\ \sum_R I_x I_y & \sum_R I_y^2 \end{bmatrix}$$

Gradients oriented in one direction.

SSD Surface – Homogeneous Areas





$$\begin{bmatrix} \sum_R I_x^2 & \sum_R I_x I_y \\ \sum_R I_x I_y & \sum_R I_y^2 \end{bmatrix}$$

Weak gradients everywhere.

Dealing with Large Motions

- The linearization with Taylor approximation is only valid for small motions!
- Remember the title of the Lucas-Kanade paper:

  B.D. Lucas and T. Kanade: An iterative image registration technique
  with an application to stereo vision. IJCA, pp. 674-679, 1981

- Two workarounds (use both):
  - Iterative estimation
  - Coarse-to-fine estimation

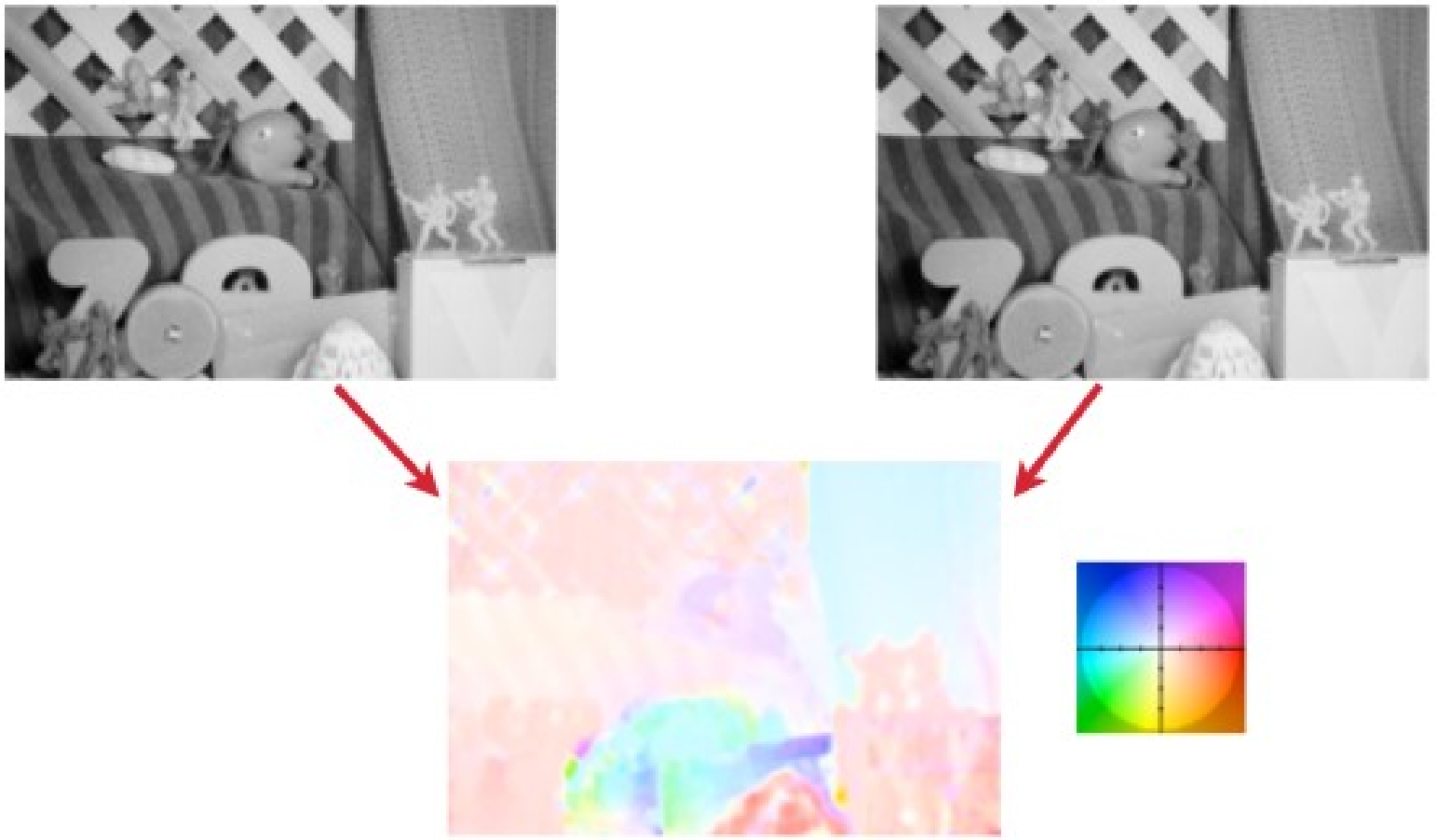
$$E_{SSD}(u,v) \approx \sum_{(x,y)\in R} \left(u \cdot I_x(x,y,t) + v \cdot I_y(x,y,t) + I_t(x,y,t)\right)^2$$

Iterative Optical Flow Estimation

- Warp the image with the previous estimate
- Use e.g. bilinear interpolation

Applying Previous Flow Estimates

- Compute initial flow estimate



The First Frame

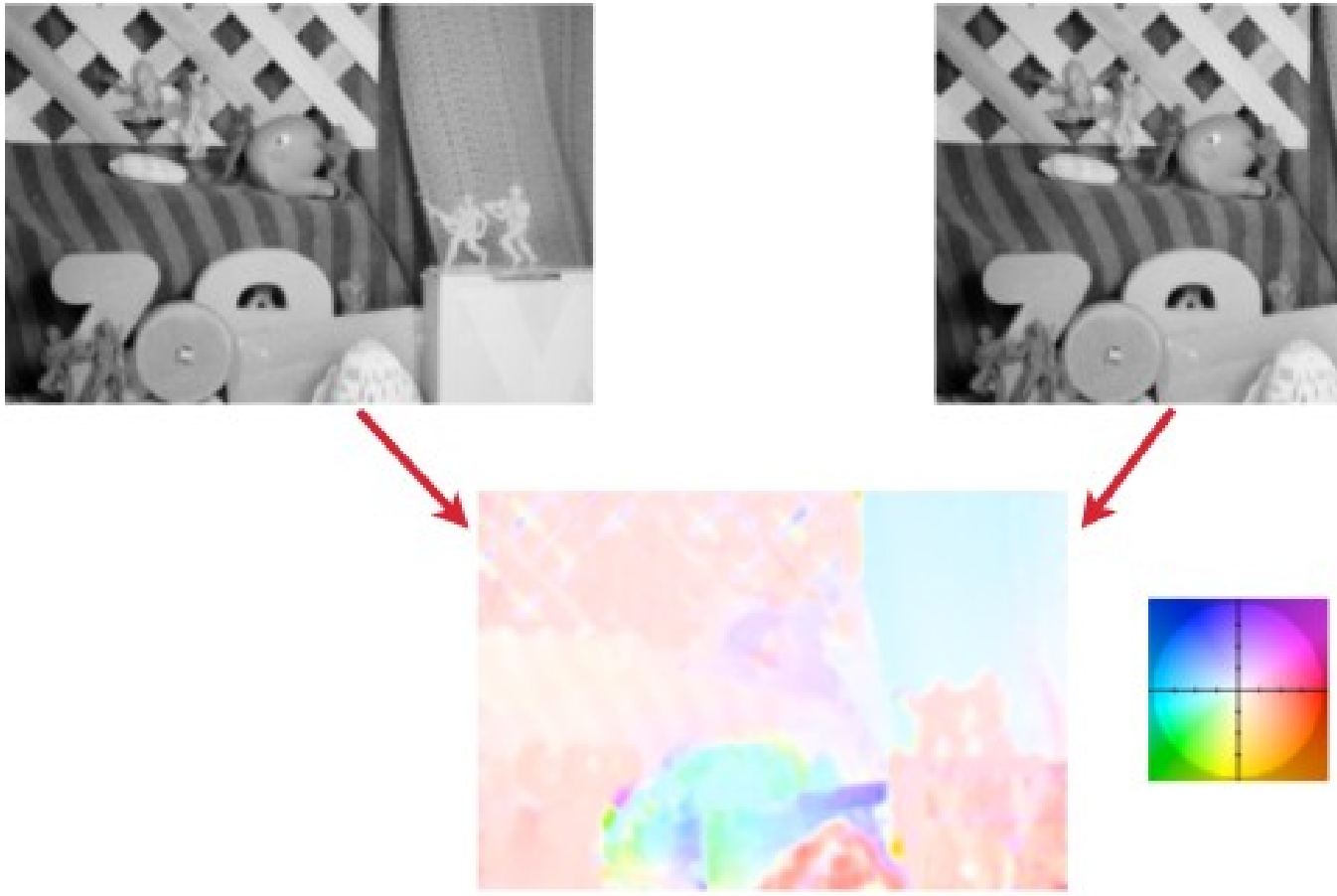
The Second Frame

The Second Frame Warped

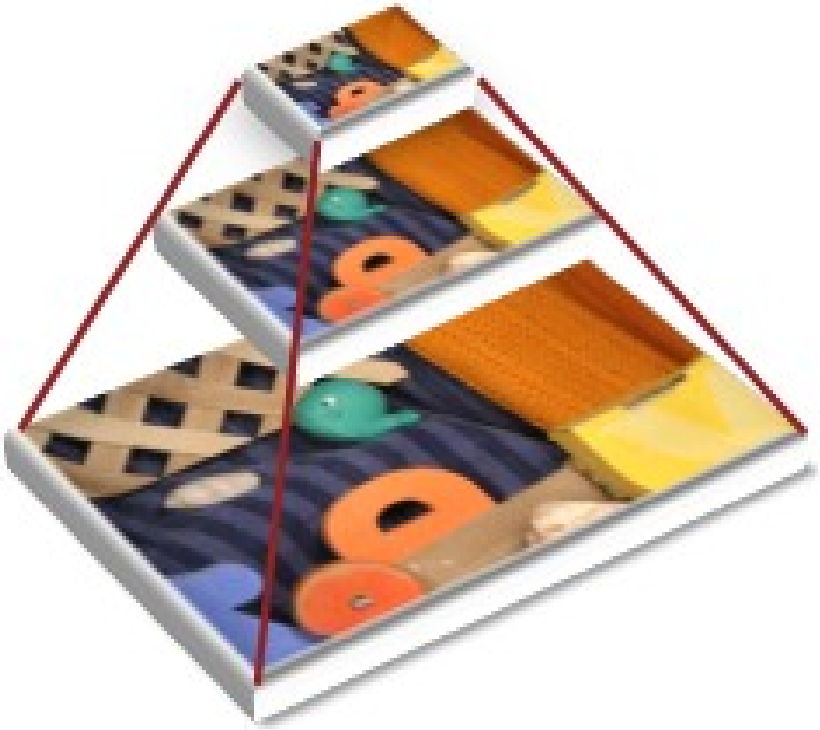
The First Frame

Incremental Flow From Warped Image Pair

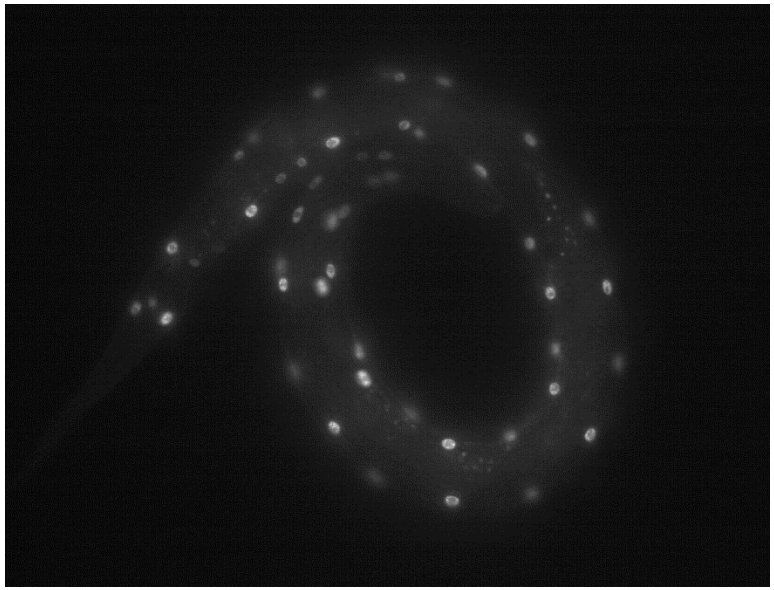
Always warp the original image with the sum of the computed flows!



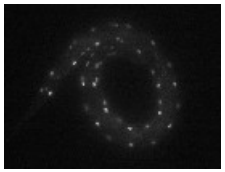
The Gaussian Image Pyramid



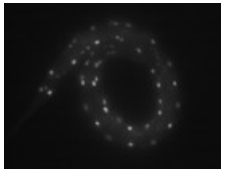
Example: Downscaling



Original Size:
1344x1024 pixel

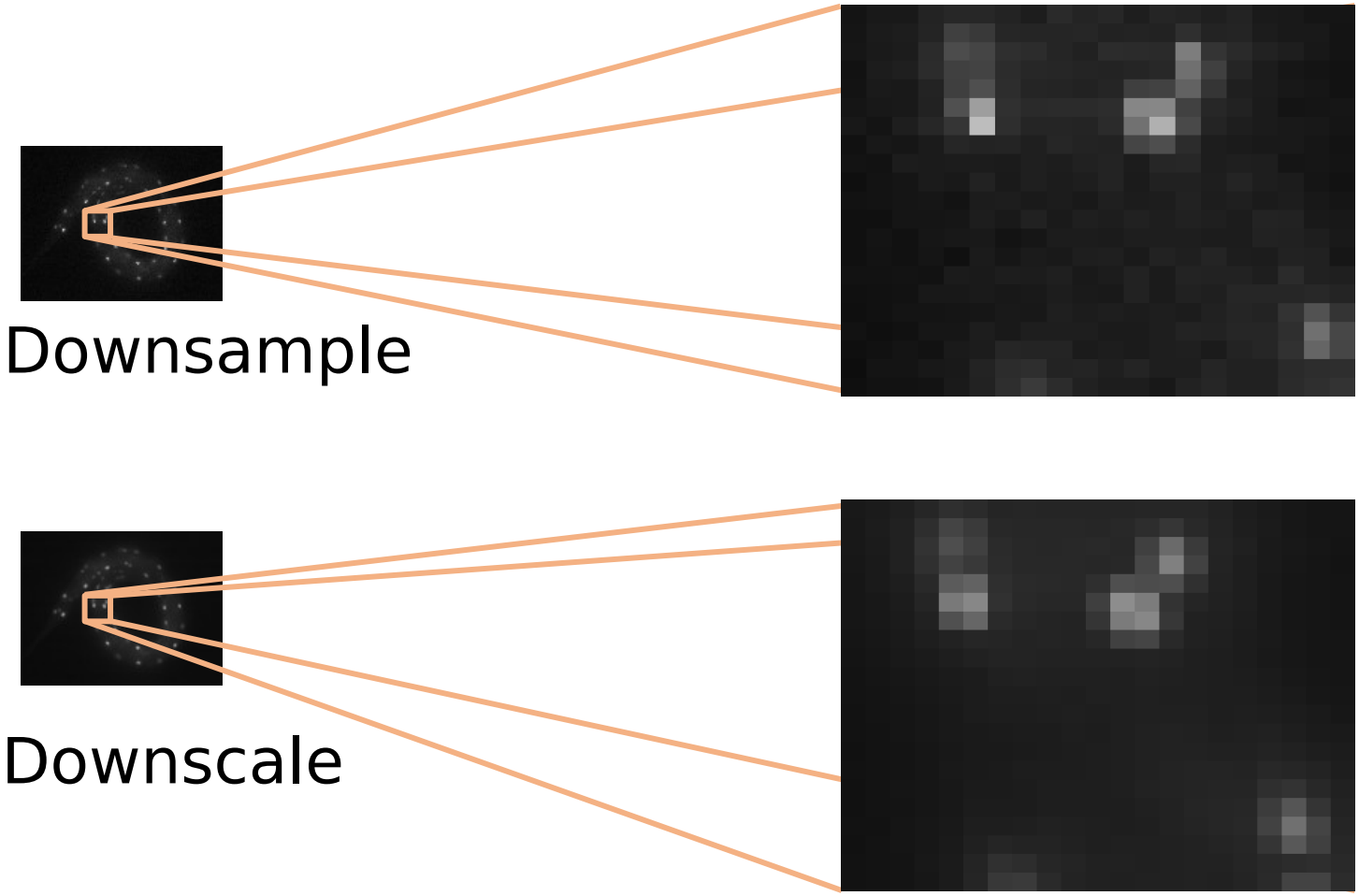


Downsample (factor 10

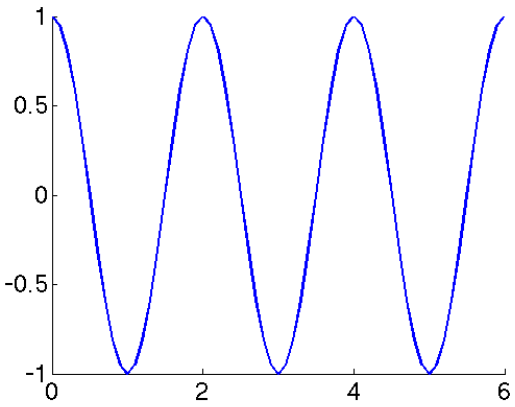


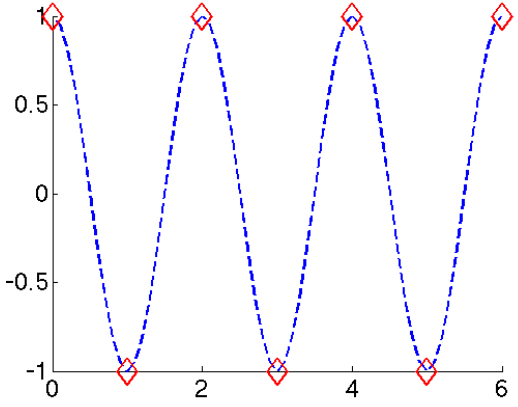
Downscale (factor 10)

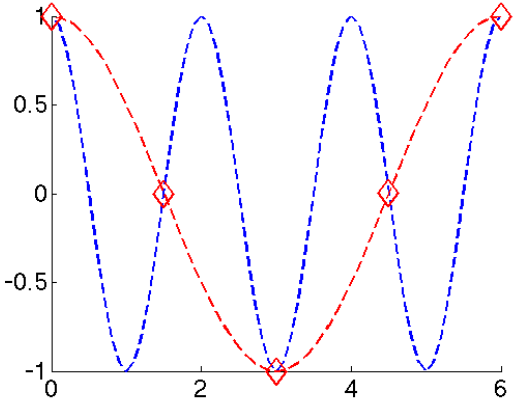
Example: Downscaling



Discrete Sampling







Input Image:

Sufficiently resolved by optics

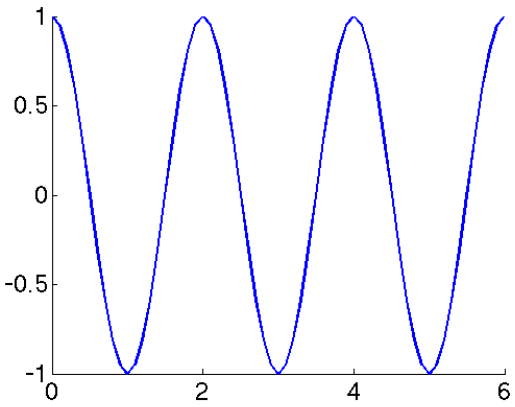
Nyquist Sampling:

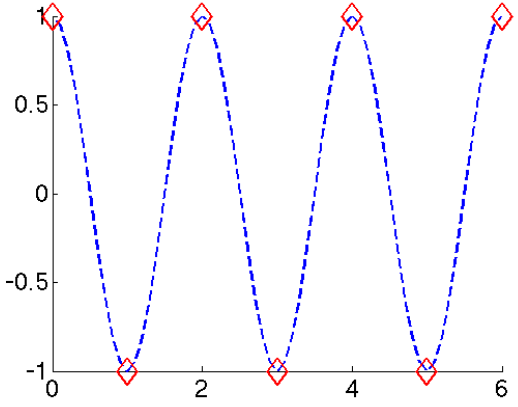
At least 2 samples per cycle

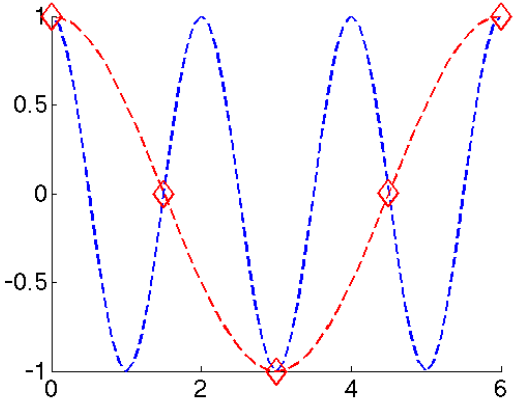
Aliasing:

Too little samples for correct reconstruction !

Discrete Sampling







Input Image:

Sufficiently resolved by optics

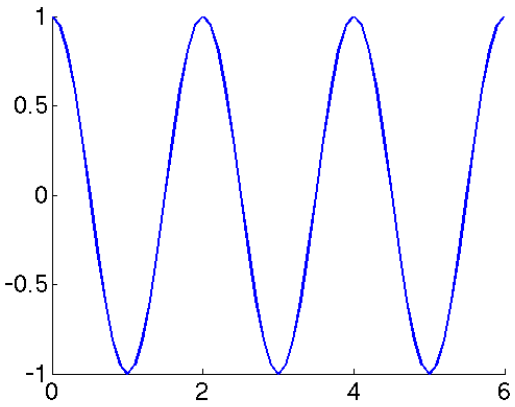
Nyquist Sampling:

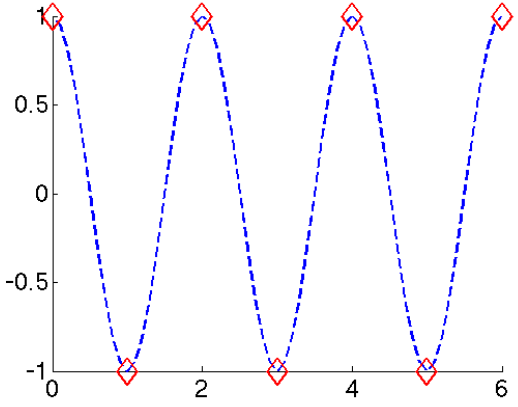
At least 2 samples per cycle

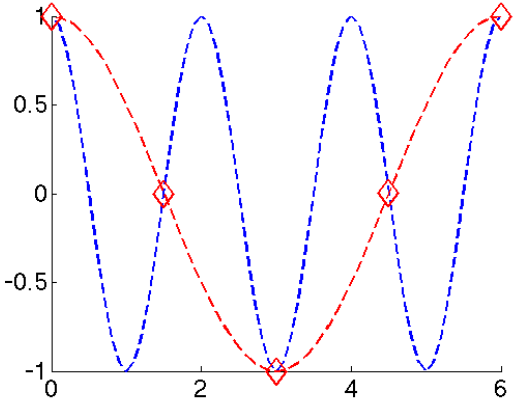
Aliasing:

Too little samples for correct reconstruction !

Discrete Sampling







Input Image:

Sufficiently resolved by optics

Nyquist Sampling:

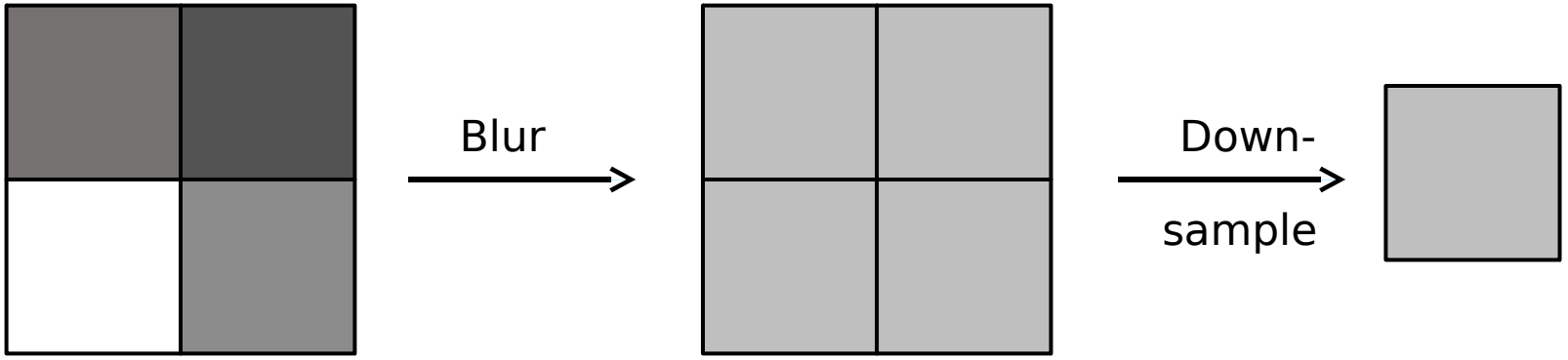
At least 2 samples per cycle

Aliasing:

Too little samples for correct reconstruction!

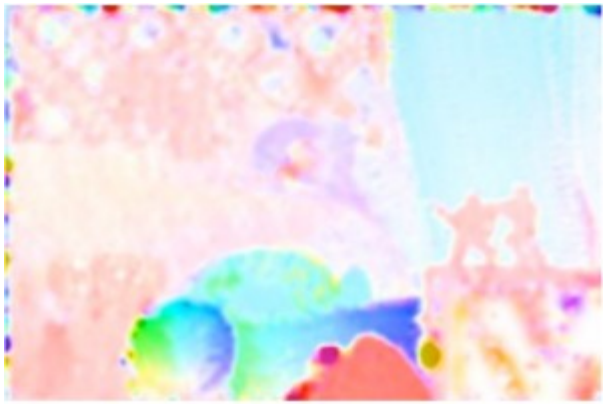
Anti-Aliasing

- Remove fine structures via blurring



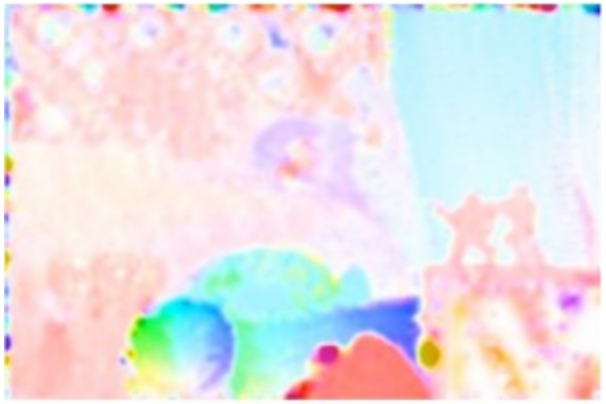
Pyramid Lucas-Kanade Algorithm

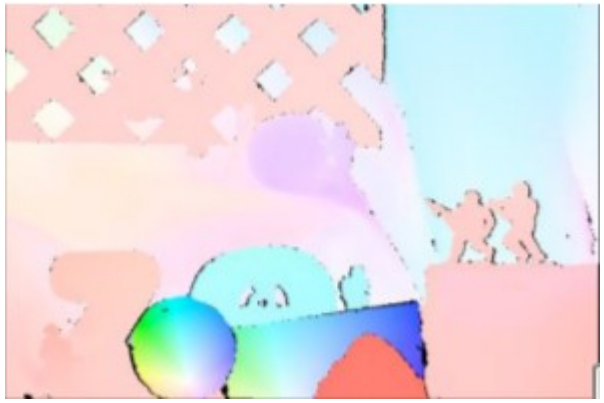
- For each level of the image pyramid
  - Generate image of appropriate resolution
  - For a number of iterations
    - Warp second image with current flow estimate backwards
    - For each pixel
      - Assume constant motion on region
      - Assume small motion update for linearization
      - Solve 2x2 equation system for motion vector



Is that a good result?

- Maybe not…





- The window is too big
  - Discontinuities are smoothed over
- The window is too small
  - In some areas the flow is bad because there is not enough image information in the window

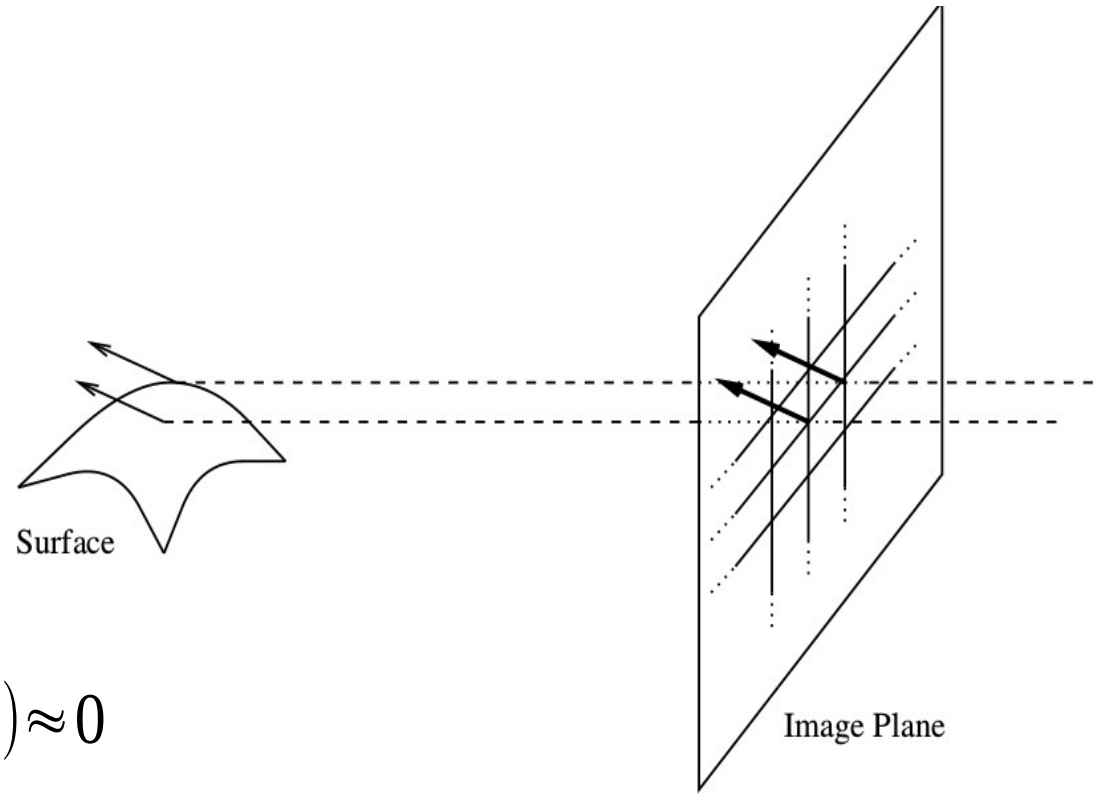
Revisiting the Assumptions

- Brightness constancy assumption
  - SSD error as objective
- The motion within a patch is constant

$$u(x, y) = u_R$$

Optical Flow Assumption II(b)

- Neighbouring scene points belong to the same surface
- Neighbouring pixels move in similar directions



$$\nabla u(x, y) \approx 0$$

Optical Flow as Optimization Problem II

- Horn and Schunck formulated the problem as a global energy term
- B.K.P. Horn and B.G. Schunck. Determining optical flow. Artificial Intelligence, 17(1-3):185-203, 1981

$$E_{HS}(u,v) = \int_{(x,y)\in\Omega} (I(x+u(x,y), y+v(x,y), t+1) - I(x,y,t))^2 + \lambda(\|\nabla u(x,y)\|^2 + \|\nabla v(x,y)\|^2)\, dx\, dy$$

The brightness difference between pixels

Regularization parameter

Gradient magnitude of the horizontal flow

Variational Calculus

A functional of the form

$$S = \int L\left(x, y, f, f_x, f_y\right) dx dy$$

Is minimized if f satisfies the partial differential equation
(Euler-Lagrange Equations)

$$\frac{\partial L}{\partial f} - \frac{\partial}{\partial x}\frac{\partial L}{\partial f_x} - \frac{\partial}{\partial y}\frac{\partial L}{\partial f_y} = 0$$

Variational Calculus: Our Case

- Applied to our case we have

$$L(x, y, \mathbf{u}, \mathbf{u_x}, \mathbf{u_y}) = (I_x u + I_y v + I_t)^2 + \lambda(u_x^2 + u_y^2 + v_x^2 + v_y^2)$$

For the PDE we thus can determine

$$\frac{\partial L}{\partial u} = 2(I_x u + I_y v + I_t) I_x \qquad \frac{\partial L}{\partial v} = 2(I_x u + I_y v + I_t) I_y$$

$$\frac{\partial L}{\partial u_x} = 2\lambda u_x \qquad \ldots$$

$$\frac{\partial}{\partial x}\frac{\partial L}{\partial u_x} = 2\lambda u_{xx} \qquad \ldots$$

Horn-Schunck Iterations

- Need to solve large, but sparse system of equations

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

- Use your favourite solver for this system (e.g. Jacobi-solver)

$$Ax = b$$

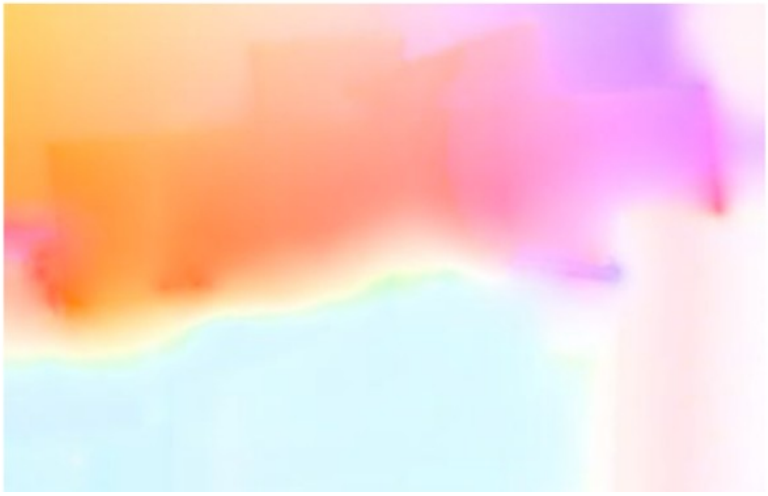
$$A = D + R$$

$$x^{k+1} = D^{-1}\left(b - Rx^k\right)$$

Pyramid Horn Schunck Algorithm

- For each level of the image pyramid
  - Generate image of appropriate resolution
  - For a number of iterations
    - Warp second image with current flow estimate backwards
    - Assume small motion update for linearization
    - For a number of iterations
      - For each pixel
        - For each pixel
          - Compute flow update

Horn Schunck Results

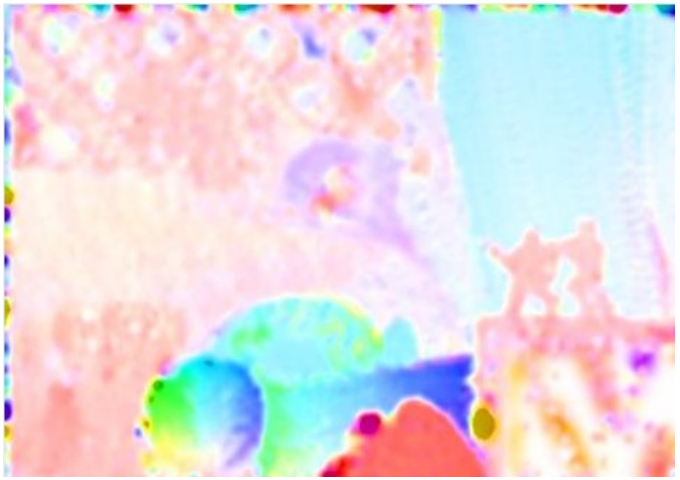


Optical Flow Algorithms

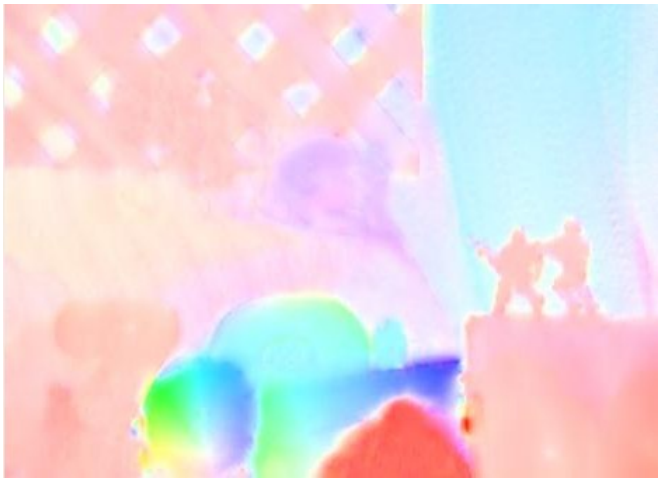
- Lucas Kanade algorithm
  - Sum of squared differences of brightness
  - Motion in a region is constant
  - Linearization of the target function (pyramid, warping)
  - Pointwise solution
- Horn Schunck algorithm
  - Sum of squared differences of brightness
  - Gradient-magnitude of the motion is small
  - Linearization of the target function (pyramid, warping)
  - Global solution

Optical Flow Algorithms: Results

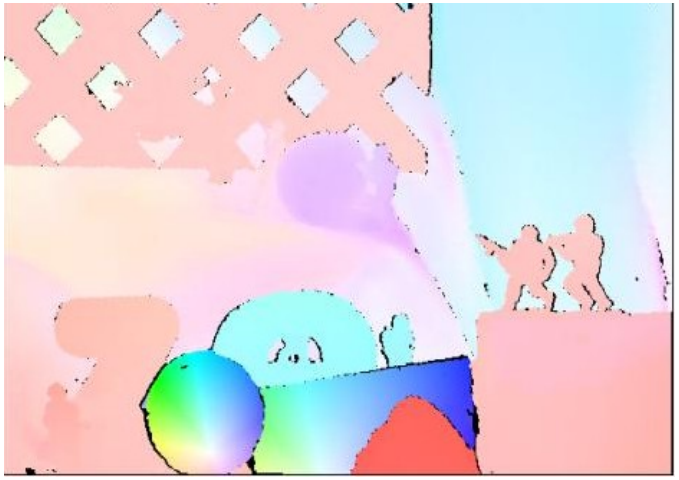
Lucas Kanade

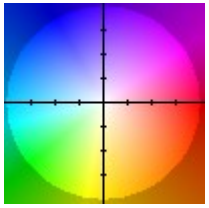


Horn Schunck



Ground Truth





Ground Truth Generation

- Use UV paint!

Known Optical Flow Fields

- Synthetic data sets
  - Sintel data set (http://sintel.is.tue.mpg.de)

- Real data sets
  - Middlebury data set (http://vision.middlebury.edu/flow)
  - Kitti data set (www.cvlibs.net/dataset/kitti)