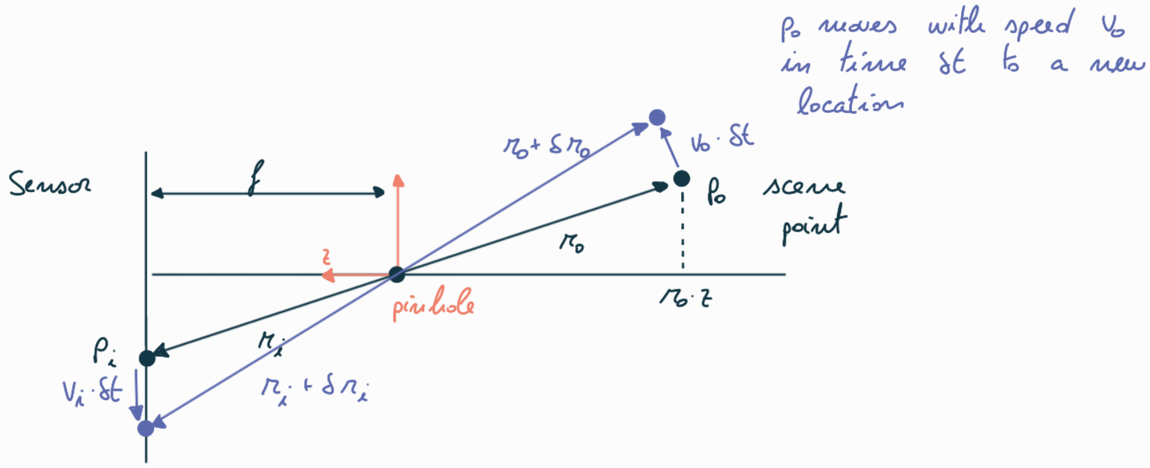
Optical flow

Before: we assumed that objects in the scene do not move (stationary scene).
Now : we are interested in measuring the motion of objects in the scene.
This is optical flow.

Goal: build a method to estimate apparent motion of scene points from a sequence of images.

A moving point in the scene will project onto the image plane and create a motion on the image plane which we will refer to as the motion field corresponding to that moving point.
We can only measure the motion of brightness patterns in the image and that's referred to as the optical flow.
When is the optical flow equal to the motion field?



Scene point velocity : $v_o = \dfrac{\partial r_o}{\partial t}$

We are interested on the projection of vector v_o onto the image plane.

Image point velocity : $v_i = \dfrac{\partial r_i}{\partial t}$

We want to relate v_i to v_o

z axis of the pinhole

perspective projection : $\dfrac{r_i}{f} = \dfrac{r_o}{r_o \cdot z} \longrightarrow r_i = f \dfrac{r_o}{r_o \cdot z}$

$$\frac{\partial r_i}{\partial t} = f \frac{(r_o \cdot z) v_o - (v_o \cdot z) r_o}{(r_o \cdot z)^2}$$

(quotient rule of derivatives)

numerator

denominator

derivative of the numerator r_o with respect to t
$\partial r_o / \partial t = v_o$

derivative of the denominator

Simplify using cross products:

$$\boxed{v_i = f \frac{(r_o \times v_o) \times z}{(r_o \cdot z)^2}}$$

INPUT: where a point is in the scene
velocity of the point (in 3D)

OUTPUT: velocity of the point in the image

v_i is the motion field corresponding to the moving point in the scene.
Even with r_o, we don't have v_o
We need to estimate optical flow

Optical flow constraint equation

Assumptions:

① As the point moves in space, the brightness of the point remains the same

$$I(x,y,t) = I(x+\delta x, y+\delta y, t+\delta t)$$ brightness constant over time

② Displacement $(\delta x, \delta y)$ and time step δt are small.
This allows us to approximate $I(x+\delta x, y+\delta y, t+\delta t)$ with the Taylor expansion.

$$f(x+\delta x) = f(x) + \frac{\partial f}{\partial x}\delta x + \underbrace{\frac{\partial^2 f}{\partial x^2}\frac{\delta x^2}{2!} + \dots + \frac{\partial^n f}{\partial x^n}\frac{\delta x^n}{n!}}$$

If δx is small:

$$f(x+\delta x) = f(x) + \frac{\partial f}{\partial x}\delta x + O(\delta x^2)$$ almost zero

For a function of three variables with small $\delta x, \delta y, \delta t$:

$$f(x+\delta x, y+\delta y, t+\delta t) \approx f(x,y,t) + \frac{\partial f}{\partial x}\delta x + \frac{\partial f}{\partial y}\partial y + \frac{\partial f}{\partial t}\partial t$$

linear approximations

Approximate $I(x+\delta x, y+\delta y, t+\delta t)$ this way:

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

$$I(x+\delta x, y+\delta y, t+\delta t) \approx I(x,y,t) + I_x\delta x + I_y\partial y + I_t\partial t$$

Given these two approximations:

① $I(x+\delta x, y+\delta y, t+\delta t) = I(x,y,t)$

② $I(x+\delta x, y+\delta y, t+\delta t) = I(x,y,t) + I_x\delta x + I_y\partial y + I_t\partial t$

② - ① $\longrightarrow$ $I_x\delta x + I_y\partial y + I_t\partial t = 0$

Divide by ∂t and take the limit $\delta t \to 0$:

$$I_x\underbrace{\frac{\partial x}{\delta t}}_{u} + I_y\underbrace{\frac{\partial x}{\partial t}}_{v} + I_t = 0$$

we have these

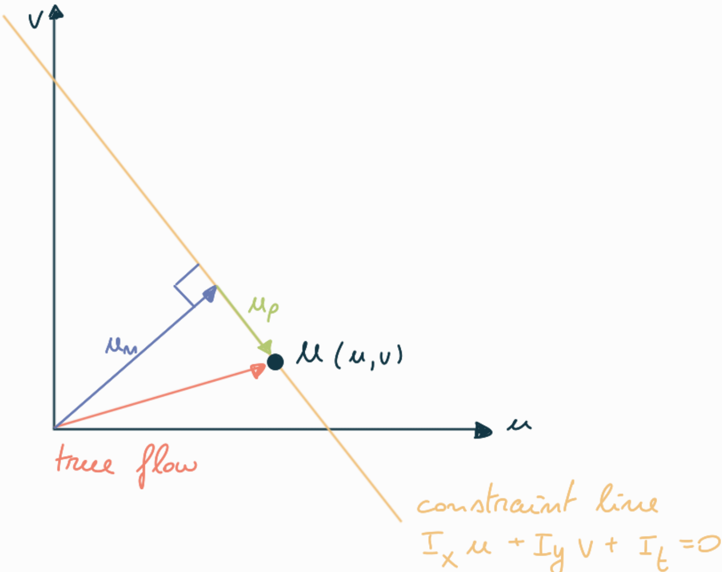
$$\boxed{I_x u + I_y v + I_t = 0}$$

(u,v) : optical flow

(I_x, I_y, I_t) can be easily computed from two frames using finite differences for any point in the image, its optical flow (u,v) lies on the line :

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

we don't know where on this line



Optical flow is a vector. We can split it up into two components :

$$u = u_n + u_p$$

normal

parallel

locally

We can compute u_n but not u_p

$$\hat{u}_n = \frac{(I_x, I_y)}{\sqrt{I_x^2 + I_y^2}}$$

direction of normal flow :
unit vector perpendicular to the constraint line

$$|u_n| = \frac{|I_t|}{\sqrt{I_x^2 + I_y^2}}$$

magnitude of normal flow :
distance of origin from the constraint line

u_n can be computed only from the constraint equation.

The underconstrained nature of the optical flow problem (that applyes to us humans as well) gives birth to the aperture problem.

We need additional constraints.

Lukas-Kanade method

The Lukas-Kanade method uses the assumption that the optical flow in a very small neighborhood in the scene is the same for all points within that neighborhood.

Assumption: for each pixel, assume motion field, and hence optical flow (u,v), is constant within a small neighborhood W.

That is for all points $(k,l) \in W$:

$$I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$$

let the size of the window W be $m \times m$:

$$\underbrace{\begin{vmatrix} I_x(1,1) & I_y(1,1) \\ \vdots & \vdots \\ I_x(k,l) & I_y(k,l) \\ \vdots & \vdots \\ I_x(m,m) & I_y(m,m) \end{vmatrix}}_{\substack{A \text{ (known)} \\ m^2 \times 2}} \underbrace{\begin{vmatrix} u \\ v \end{vmatrix}}_{\substack{u \text{ (unknown)} \\ 2 \times 1}} = \underbrace{\begin{vmatrix} I_t(1,1) \\ \vdots \\ I_t(k,l) \\ \vdots \\ I_t(m,m) \end{vmatrix}}_{\substack{B \text{ (known)} \\ m^2 \times 1}}$$

We can solve this with the least Squares Solution

$$A u = B$$
$$A^T A u = A^T B$$

In matrix form:

$$\underbrace{\begin{vmatrix} \sum_W I_x I_x & \sum I_y I_x \\ \sum_W I_x I_y & \sum I_y I_y \end{vmatrix}}_{\substack{A^T A \\ \text{(known)} \\ 2\times 2}} \underbrace{\begin{vmatrix} u \\ v \end{vmatrix}}_{\substack{u \\ \text{(unknown)} \\ 2\times 1}} = \underbrace{\begin{vmatrix} -\sum_W I_x I_t \\ -\sum_W I_y I_t \end{vmatrix}}_{\substack{A^T B \\ \text{(known)} \\ 2\times 1}}$$

$$u = (A^T A)^{-1} A^T B$$ fast and easy to solve

- $A^T A$ must be invertible $\longrightarrow \det(A^T A) \neq$
- $A^T A$ must be well-conditioned

A well conditioned system has a significant change in the output if there is a change in the input. When this is the case we can take the output and estimate the input well enough.

When is a system well conditioned?

$\rightarrow$ if λ_1 and λ_2 are eigenvalues of $A^T A$, then:

$\lambda_1 > \varepsilon$ and $\lambda_2 > \varepsilon$

$\lambda_1 \geq \lambda_2$ but not $\lambda_1 \ggg \lambda_2$ significally larger

What if we have large motion? e.g. objects near the camera

① Taylor series approximation of $I(x+\delta x, y+\delta y, t+\delta t)$ is no more valid

② The simple linear constraint equation $I_x u + I_y v + I_t \neq 0$ is no more valid

We use resolution pyramids:
for each pair of images (at time t and $t+\delta t$) we compute pairs of images at lower resolution

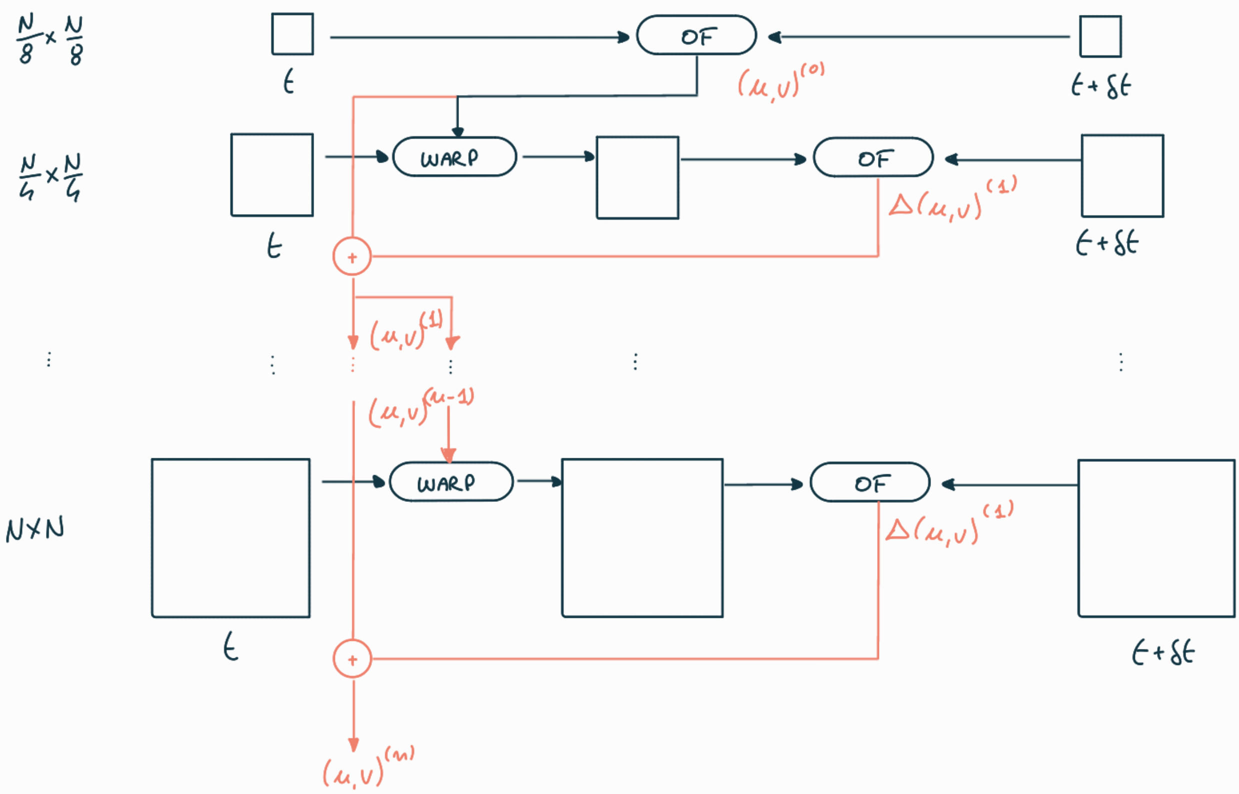
$$t: N\times N \longrightarrow \frac{N}{2}\times\frac{N}{2} \longrightarrow \frac{N}{4}\times\frac{N}{4} \cdots$$

$$t+\delta t: N\times N \longrightarrow \frac{N}{2}\times\frac{N}{2} \longrightarrow \frac{N}{4}\times\frac{N}{4} \cdots$$

At lowest resolution, all motion will be ≤ 1 pixel. At this resolution the optical flow constraint equation becomes valid again.

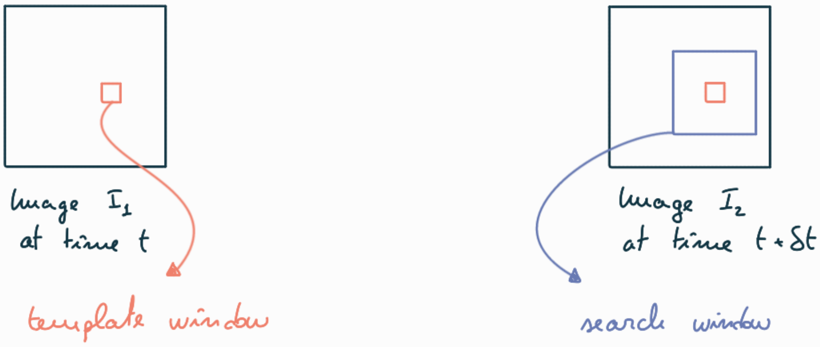
Coarse-to-Fine Estimation Algorithm

Start with the lowest resolution



Alternative approach: Template Matching

For each template window T in image I_1, find the corresponding match in image I_2.
This method uses brute force.



Template matching is slow and mismatches are possible.

Horn-Schunck Optical Flow

We have two main methods to compute optical flow:

- **Lukas-Kanade Optical Flow** is a local method (sparse) that computes the optical flow only for some key features.
- **Horn-Schunck Optical Flow** is a global method (dense) that computes the optical flow globally, for the entire image

Lukas-Kanade	Horn-Schunck
• local method (sparse)	• global method (dense)
• constant flow: flow is constant for all pixels in neighborhood	• smooth flow: flow can smoothingly vary from pixel to pixel
• brightness constancy	• brightness constancy
• small motion	• small motion

Horn-Schunck method enforces:

- brightness constancy: brightness (intensity of a pixel) in an image sequence remains constant over time except for motion-induced changes
- smooth flow field

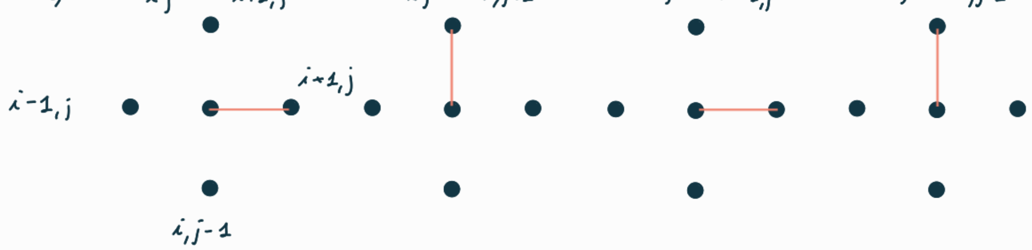
The Horn-Schunck method formulates optical flow estimation as an optimization problem. It assumes that the displacement vectors of neighboring pixels in an image sequence are similar and should be smooth. The goal is to find the optical flow vectors that minimize an energy function. Which energy function?

Enforce brightness consistency : E_d

Ensure smooth flow field : E_s

$$E_d(i,j) = \left[I_x u_{ij} + I_y v_{ij} + I_t \right]^2$$

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

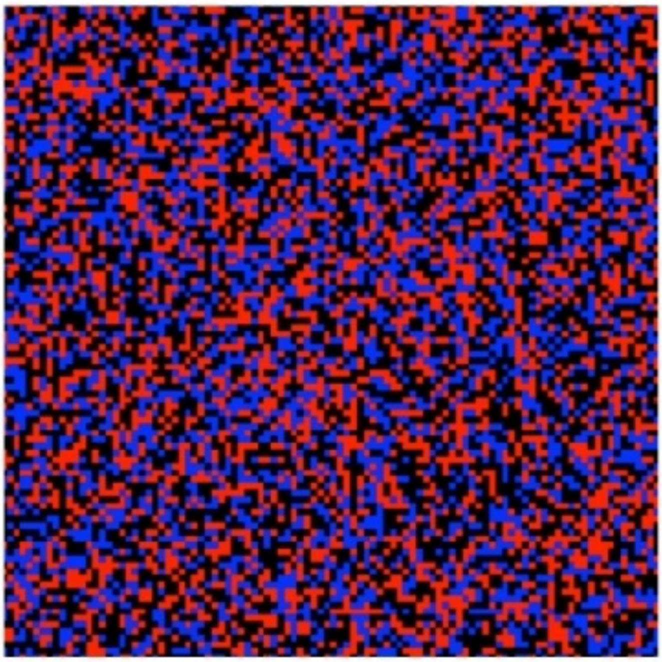


$$\min_{u,v} (E_d(i,j)) = \min_{u,v} \left[I_x u_{ij} + I_y v_{ij} + I_t \right]^2$$

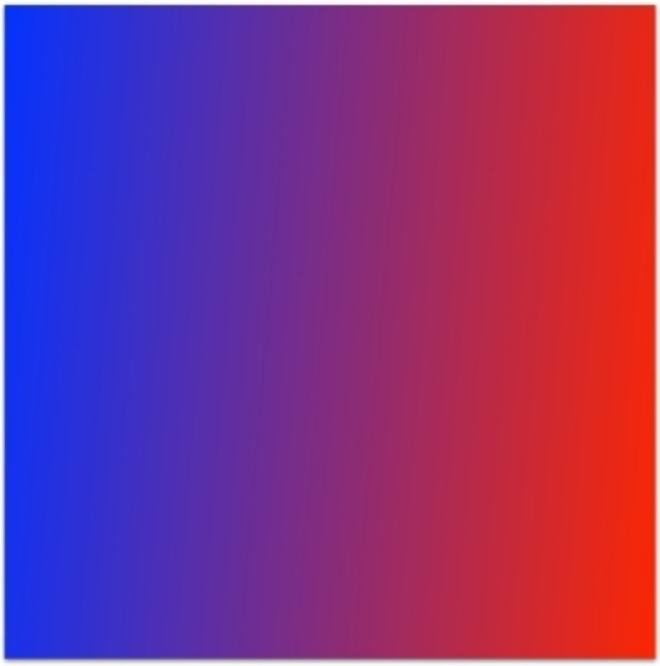
This because $I_x u + I_y v + I_t = 0$ means no motion (no change in brightness)

$$\min_{u,v} (E_s(i,j)) = \min_{u,v} \left[(u_{ij} - u_{i+1,j})^2 + (u_{ij} - u_{i,j+1})^2 + (v_{ij} - v_{i+1,j})^2 + (v_{ij} - v_{i,j+1})^2 \right]$$

We need to penalize large spatial derivatives that correspond to big change in brightness and result in noise in optical flow



big difference between neighbors correspond to noise



small difference between neighbors correspond to uniform optical flow

Putting the two energy functions together we get:

$$\min_{u,v} \sum_{i,j} \{ \underbrace{E_s(i,j)}_{\text{smoothness}} + \lambda \underbrace{E_d(i,j)}_{\text{brightness}} \}$$

Horn - Schunck Optical Flow Algorithm

1. Precompute image gradients I_x and I_y
2. Precompute temporal gradients I_t
3. Initialize flow field $u = 0, v = 0$
4. While not converged :

compute flow field update for each pixel

$$\hat{u}_{kl} = \bar{u}_{kl} - \frac{I_x \bar{u}_{kl} + I_y \bar{v}_{kl} + I_t}{\lambda^{-1} + I_x^2 + I_y^2} I_x$$

$$\hat{v}_{kl} = \bar{v}_{kl} - \frac{I_x \bar{u}_{kl} + I_y \bar{v}_{kl} + I_t}{\lambda^{-1} + I_x^2 + I_y^2} I_y$$

Motion segmentation

What can we do if we have multiple moving objects in the scene? : motion segmentation
Motion segmentation is the process of partitioning an image sequence into regions based on their motion pattern.

We break the image sequence into layers, each of which has a coherent (affine) motion
An affine motion is typically represented as a linear function of the form:

$$x' = a_1 + a_2 x + a_3 y$$
$$y' = b_1 + b_2 x + b_3 y$$

original point

when we talk about 2D transformations

coordinates of the point after the transformation

$a_1, a_2, a_3, b_1, b_2, b_3$ are the parameters of the affine transformation.
The affine transformation represents a combination of scaling, translation, rotation and shearing.

$a_1, b_1 \longrightarrow$ translation
$a_2, b_3 \longrightarrow$ scaling
$a_3, b_2 \longrightarrow$ shearing
the absence of cross products determines that there is no rotation in this model.

Thus:

$$u(x,y) = a_1 + a_2 x + a_3 y$$
$$v(x,y) = b_1 + b_2 x + b_3 y$$

Brightness constancy equation:

$$I_x u + I_y v + I_t = 0$$
$$I_x (a_1 + a_2 x + a_3 y) + I_y (b_1 + b_2 x + b_3 y) + I_t = 0$$

Each pixel (x,y) provides 1 linear constraint with 6 unknowns
If we have at least 6 pixels in a neighborhood, $a_1 \dots b_3$ can be found by least squares minimization:

$$Err(a) = \sum [I_x (a_1 + a_2 x + a_3 y) + I_y (b_1 + b_2 x + b_3 y) + I_t]^2$$

residual error

How do we estimate layers ?

1. Divide the image into blocks and estimate affine motion parameters in each block by least squares
2. Eliminate hypotheses with high residual error
3. Map into motion parameter space
4. Perform K-means clustering on affine motion parameters
5. Merge clusters that are close and retain a smaller set of hypotheses ; each hypothesis represents one object
6. Assign to each pixel the best hypothesis