optical flow field:

velocity field ~~for~~ defined on image domain that transforms one image to another

→ not uniquely determined (projection equations one problem)

motion field:

projection onto the image of three-dimensional vectors

other ~~problem~~ sources of error:

specular effects
shadows
insufficient texturing
occlusion

define a vector field?

$$\underset{\substack{\text{at every point}\\ \text{in image}}}{X : \mathbb{R}^2} \rightarrow \mathbb{R}^2$$

$\mathbb{R}^2$ — defines a vector describing particle motion

if we integrate the vector field

$$\dot{x} = X(x) \quad , \quad x(0) = x_0$$

then we get particle trajectories for various x_0. Call them $\Phi^X_{0,t}(x_0)$

Suppose an imaged particle is moving, then it satisfies the following equation

$$I(\Phi^X_{0,t}(x), t) = I(x, 0)$$

assuming nothing else funny is going on

called "Brightness Constancy Assumption"

$\Rightarrow$

time derivative vanishes

$$\frac{d}{dt} I(\Phi^X_{0,t}(x), t) = \frac{\partial}{\partial t} I(x,0).$$

$\Rightarrow$

$$\nabla I \cdot \frac{\partial}{\partial t}\Phi^X_{0,t} + \frac{\partial I}{\partial t} = 0$$

$\Rightarrow$

$$\nabla I \cdot X + \frac{\partial I}{\partial t} = 0$$

$\Rightarrow$

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

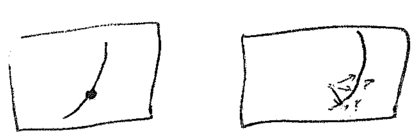
optical flow constraint equation (OFC)

- PROBLEMS one constraint (OFC) versus two unknowns (u,v).

$\Rightarrow$

multiple solutions possible

↳ equicontours



if intensity is uniform in a region, then vector field is ambiguous
but should also be "uniform" in some sense.

but, at an edge where there is more image variation, the situation is less ambiguous.

furthermore, all particles at an edge move with ~~__~~ locally consistent velocities.

what's one way to impose local consistency ?
e.g. restrict local variation

isotropic smoothing e.g. reg smoothness

sources of optical flow (relevant):

- relative motion between object & viewer
  - ↳ allows to infer spatial arrangements
- discontinuities in flow field can be used for segmentation
- motion recovery (objects xor self)
- shape

violation: uniform sphere w/shading ① rotate sphere ② move light source.

Horn & Schunck version:

$$\nabla I \cdot X = -I_t$$

component/magnitude of movement in the direction of the brightness gradient ∇I equals

$$-\frac{I_t}{\|\nabla I\|_2}$$

cannot determine component in direction of iso-brightness contours.

add smoothness constraint $\min \lambda(\|\nabla u\|^2 + \|\nabla v\|^2)$

$\|\nabla u\|^2$
$u_x^2 + u_y^2$

$\Rightarrow$

$\|\nabla(u+\delta u)\|^2$
↳ $2\nabla u \cdot \nabla \delta u$

$$\mathcal{L}(X) = \iint \left[(\nabla I \cdot X + I_t)^2 + \alpha^2 (\|\nabla u\|^2 + \|\nabla v\|^2) \right] dx\,dy$$

$\Rightarrow$ compute 1st variation:

$$\frac{\delta \mathcal{L}}{\delta X} \cdot \delta X = \iint \left[2(\nabla I \cdot X + I_t)^{\nabla I}\delta X + \alpha^2 (2\nabla u \cdot \nabla \delta u + 2\nabla v \cdot \nabla \delta v) \right] dx\,dy$$

$$= \iint \left[2(\nabla I \cdot X + I_t)\nabla I \cdot \delta X + 2\alpha^2 (\nabla u \cdot \nabla \delta u + \nabla v \cdot \nabla \delta v) \right] dx\,dy$$

by parts

$$\iint \nabla u \cdot \nabla \delta u \, dx\,dy$$

$$\nabla u \cdot \delta u \Big|_{\partial D} - \iint \Delta u \cdot \delta u$$

$$\frac{\delta L}{\delta X} \cdot \delta X = \iint \left[2(\nabla I \cdot X + I_t) \nabla I \cdot \delta X - 2\alpha^2 (\Delta u, \Delta v) \cdot \delta X \right] dx\,dy$$

$$= \iint \left[2(\nabla I \cdot X + I_t) \nabla I - 2\alpha^2 (\Delta u, \Delta v) \right] \cdot \delta X \, dx\,dy$$

two components in integrand must vanish:

$$2(\nabla I \cdot X + I_t) I_x - 2\alpha^2 \Delta u = 0$$

$$2(\nabla I \cdot X + I_t) I_y - 2\alpha^2 \Delta v = 0$$

$\Rightarrow$

$$I_x^2 u + I_x I_y v = \alpha^2 \Delta u - I_x I_t$$

$$I_x I_y u + I_y^2 v = \alpha^2 \Delta v - I_y I_t$$

$\Rightarrow$ def. of Laplacian in paper

$$(\alpha^2 + I_x^2) u + I_x I_y v = \alpha^2 \bar{u} - I_x I_t$$

$$I_x I_y u + (\alpha^2 + I_y^2) v = \alpha^2 \bar{v} - I_y I_t$$

$\Rightarrow$

$$\begin{bmatrix} \alpha^2 + I_x^2 & I_x I_y \\ I_x I_y & \alpha^2 + I_y^2 \end{bmatrix} \begin{Bmatrix} u \\ v \end{Bmatrix} = \begin{Bmatrix} \alpha^2 \bar{u} - I_x I_t \\ \alpha^2 \bar{v} - I_y I_t \end{Bmatrix}$$

$\alpha^4 + \alpha^2 I_x^2 - \alpha^2 I_y^2$

$\alpha^2(\alpha^2 + I_x^2 + I_y^2)$

$$\alpha^2(\alpha^2 + I_x^2 + I_y^2) \begin{Bmatrix} u \\ v \end{Bmatrix} = \begin{bmatrix} \alpha^2 + I_y^2 & -I_x I_y \\ -I_x I_y & \alpha^2 + I_x^2 \end{bmatrix} \begin{Bmatrix} \alpha^2 \bar{u} - I_x I_t \\ \alpha^2 \bar{v} - I_y I_t \end{Bmatrix}$$

define $\Delta u, \Delta v$

$\Rightarrow$

$$\cancel{\alpha^2 + I_x^2} \quad u = (\alpha^2 + I_x^2 + I_y^2)^{-1} \left[(\alpha^2 + I_y^2)\bar{u} - I_x I_y \bar{v} - I_x I_t \right]$$

$$v = (\alpha^2 + I_x^2 + I_y^2)^{-1} \left[(\alpha^2 + I_x^2)\bar{v} - I_x I_y \bar{u} - I_y I_t \right]$$

⇒ ALTERNATIVELY

$$\cancel{(\alpha^2 + I_x^2 + I_y^2)(u - \bar{u}) = -}$$

$$u = \bar{u} - (\alpha^2 + I_x^2 + I_y^2)^{-1} \left[I_x (I_x \bar{u} + I_y \bar{v} + I_t) \right]$$
$$v = \bar{v} - (\alpha^2 + I_x^2 + I_y^2)^{-1} \left[I_y (I_x \bar{u} + I_y \bar{v} + I_t) \right]$$

this is the solution.

but since we need to find these u, we use gradient descent.

called Gauss-Seidel & is used for linear systems.

note that solution to u is linear in $\bar{u}$ and $\bar{v}$

⇒

if

$$x = Ay$$

solution given by

x

$$r_k = (x_k - Ay_k)$$

this is defect.

$$Ax = 0$$
$$Ax_k = r_k$$

$$x_{k+1} = x_k - r_k = x_k - (x_k - Ay_k) = Ay_k$$

but then y changes (implicitly)

⇒ $r_{k+1} = x_{k+1} - Ay_{k+1}$

⇒

$$x_{k+1} = Ay_k$$

$$u^{n+1} = \bar{u}^n - I_x (\alpha^2 + I_x^2 + I_y^2)^{-1} (I_x \bar{u}^n + I_y \bar{v}^n + I_t)$$
$$v^{n+1} = \bar{v}^n - I_y (\alpha^2 + I_x^2 + I_y^2)^{-1} (I_x \bar{u}^n + I_y \bar{v}^n + I_t)$$

in

Iterative Methods.

$$Ax = b$$

$$\downarrow$$

invertible.

$$(S + N)x = b$$

$$\Rightarrow Sx = b + Nx$$

$$\Rightarrow x = S^{-1}(b + Nx)$$

$$x_{n+1} = S^{-1}(b + Nx_n)$$

$$= S^{-1}b + S^{-1}Nx_n$$

$$Ax^* = b$$

$$\Rightarrow Sx^* = b + Nx^*$$

$$x^* = S^{-1}b + S^{-1}Nx^*$$

$$\boxed{x^* = x - e}$$

$$r = Ax - b$$

$$\boxed{r = Ax - Ax^* = Ae}$$

true solution

$$\Rightarrow$$

$$x - \underbrace{A^{-1}}_{\text{don't have}} r = x^*$$

$$r_k = Ax_k - Ax^*$$

$$r_{k+1} = Ax_{k+1} - Ax^*$$

$$= AS^{-1}(b - N\cancel{x}_n) - Ax^*$$

$$= AS^{-1}(b) - b + AS^{-1}Nx_n$$

$$=$$

in our case, structure of optical flow lets us solve this pixel by pixel, since S is the diagonal part of A and N is the off diagonal part.

$$e_{k+1} = x_{k+1} - x^* = S^{-1}b + S^{-1}Nx_k - x^*$$

$$= +S^{-1}Nx_k - S^{-1}Nx^*$$

$$= S^{-1}Ne_k$$

all good, so long as $\|S^{-1}N\| < 1$

$\Rightarrow$ eig. of $S^{-1}N$ are less than unity

error will reduce w/each iteration.

$$x_{k+1} = b + (S+N)x_k \; ?$$

Richardson iteration

$S = I, \quad N = (I - A)$

OFC + L_2 smoothness $\rightarrow$ iterative method.

In optical flow case, we have

$$Ax = b$$

$$\Rightarrow$$

$$(I + N)x = b \qquad\qquad N = I - A$$

$$\Rightarrow$$

$$I x_{n+1} = \underbrace{b}_{\text{image only terms}} + \underbrace{N x_n}_{\text{OFC + smoothness}}$$

But, we can do this differently.

$$u^{n+1} = \bar{u}^n - \frac{I_x \bar{u}^n + I_y \bar{v}^n + I_t}{1 + \alpha^2 (I_x^2 + I_y^2)} I_x$$

$$v^{n+1} = \bar{v}^n - \frac{I_x \bar{u}^n + I_y \bar{v}^n + I_t}{1 + \alpha^2 (I_x^2 + I_y^2)} I_y$$

in class

in this other case, we have

$$\mathcal{E} = \alpha^2 \mathcal{E}_{OFC} + \mathcal{E}_s$$

✱ discuss boundary conditions!

$$(u_x, u_y)^T \cdot \hat{n} = 0 \qquad (v_x, v_y)^T \cdot \hat{n} = 0$$

- the two algorithms are equivalent, but can differ.

need to verify this with my code.

did I program both?

In Matlab,

$\bar{u}, \bar{v}$ are obtained by convolution.

$(\alpha^2 + I_x + I_y)$ is a matrix
I_x, I_y, I_t are matrices
} constant once images are given

$\cancel{I_x\bar{u} + I_y\bar{v} + I_t}$

$I_x\bar{u}$, $I_y\bar{v}$ computed matrices.

quiver — tell them how it works!

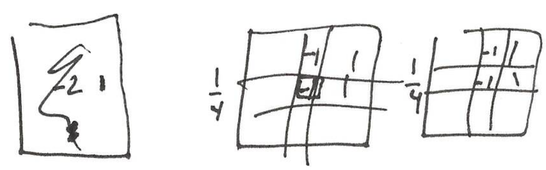
need two images, α^2 parameter, and # iterations.

$\begin{bmatrix}-1 & -1\\ 1 & 1\end{bmatrix}$

& only one for loop in MATLAB optical flow function.

$\begin{bmatrix}0 & -1 & 1\\ 0 & -1 & 1\\ 0 & 0 & 0\end{bmatrix}$

{ can use basic I_x, I_y, I_t
or can get fancier



H&S:

$$I_x = \frac{1}{4}\Big(\underbrace{I(i,j+1,k) - I(i,j,k)} + \underbrace{I(i+1,j+1,k) - I(i+1,j,k)} + \underbrace{I(i,j+1,k+1) - I(i,j,k)} + \underbrace{I(i+1,j+1,k+1) - I(i+1,j+1,k)}\Big)$$

$\frac{1}{4}\begin{bmatrix}-1 & 1\\ -1 & 1\end{bmatrix}$

I_y is similar

$$I_t = \frac{1}{4}\Big(\underbrace{I(i,j,k+1) - I(i,j,k)} + \underbrace{I(i+1,j,k+1) - I(i+1,j,k)} + \underbrace{I(i,j+1,k+1) - I(i,j+1,k)} + \underbrace{I(i+1,j+1,k+1) - I(i+1,j+1,k)}\Big)$$

$\begin{bmatrix}-1 & -1\\ -1 & -1\end{bmatrix}$ $\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix}$

average of forward Euler in space and time.

GRADIENT DESCENT OPTICAL FLOW:

instead of solving for the minimal solution, one can use gradient descent to arrive at it,

$$\frac{\delta u}{\delta \tau} = -\left\{(\nabla I \cdot X + I_t) I_x - \alpha^2 \Delta u\right\}$$

$$\frac{\delta v}{\delta \tau} = -\left[(\nabla I \cdot X + I_t) I_x - \alpha^2 \Delta u\right]$$

$\Rightarrow$

$$\frac{u^{n+1} - u^n}{d\tau} = (\nabla I \cdot X + I_t) I_x - \alpha^2 \Delta u$$

$\Rightarrow$

$\Delta u = u^{++} + u^{+-} + u^{-+} + u^{--} - 4u$

$$u^{n+1} = u^n - d\tau\left[(\nabla I \cdot X + I_t) I_x - \alpha^2 \Delta u\right]$$

$\delta u - u^{n+1}$

$$v^{n+1} = v^n - d\tau\left[(\nabla I \cdot X + I_t) I_y - \alpha^2 \Delta v\right]$$