Vision and Image Processing: Optical Flow

François Lauze
Department of Computer Science
University of Copenhagen

Plan for today

- Discuss a general introduction to True Motion and Apparent Motion.
- Discuss the Aperture Problem.
- Discuss approaches for apparent motion recovery.
- Present two old but still vigorous techniques:
  - Approach by Block-Matching
  - Lucas and Kanade approach
  - Horn and Schunck technique.
- Discuss and reflect on some non-dense feature based techniques.

Outline

What is Optical Flow?

Definition

(From Wikipedia) Optical flow or optic flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer (an eye or a camera) and the scene.

What is Optical Flow?

Definition

(From Wikipedia) Optical flow or optic flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer (an eye or a camera) and the scene.

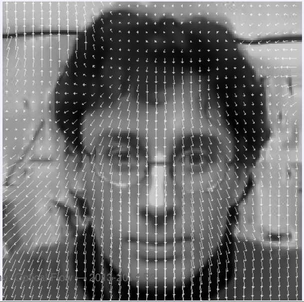
This raises questions about:

- image formation and projection of motion,
- link between projection of motion and observed patterns of apparent motions,
- and of course, how can it be recovered?

An example

An example

An example



Applications of Optical Flow techniques

Applications are numerous e.g.

- Robotics – robot navigation
- 3D scene understanding
- Surveillance
- Computer graphics and Augmented Reality
- Film restoration
- Motion compensation in Medical Images...

Outline

Standard Pinhole Camera Model

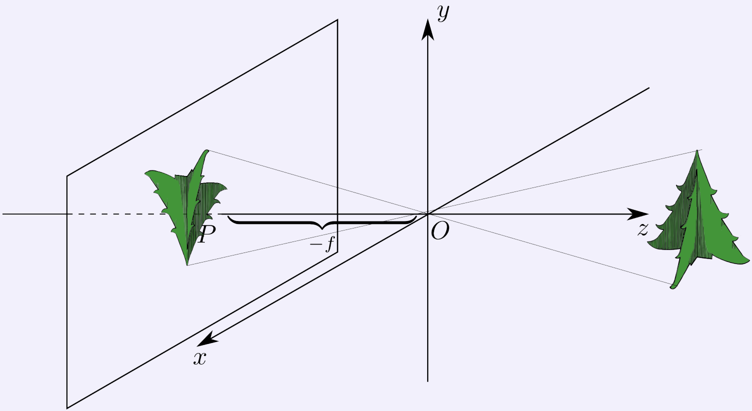
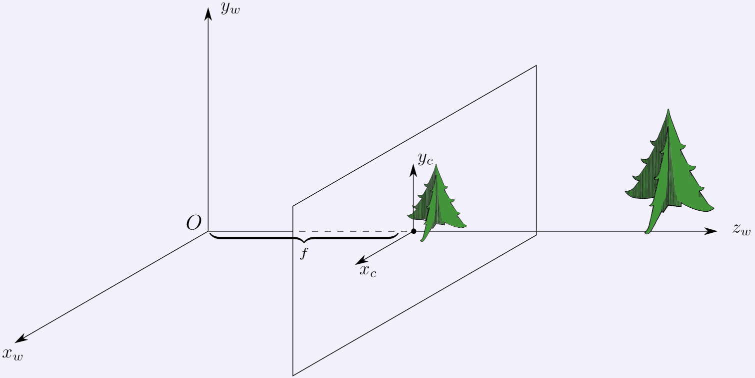


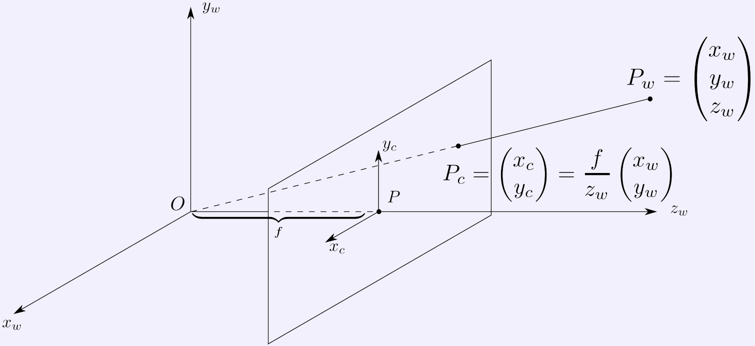
Image is flipped 180^o, rays pass through the aperture point to reach the image plane, at a distance f – the focal length from the aperture point.

Alternative Pinhole Camera Model



In this model the image plane is in front of the aperture point. Gives a trick to preserve orientations (standard CCD camera implement it as a 180^o rotation of the camera plane)

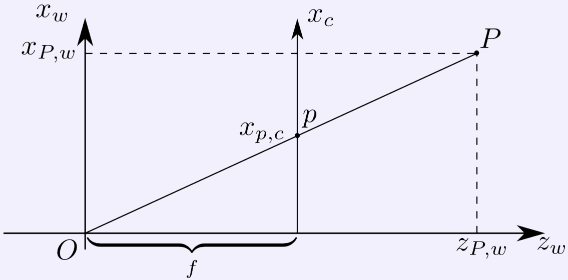
Pinhole Camera Model I



Assume the “world” coordinates are (x_w, y_w, z_w) centered at the aperture point O and the camera image plane is parallel to the xy plane and goes through P. Distance OP focal length f.

Pinhole Camera Model II

The 2D to 1D case:



Clearly, one has

$$\frac{x_{p,c}}{f} = \frac{x_{P,w}}{z_{P,w}} \iff x_{p,c} = f\frac{x_{P,w}}{z_{P,w}}$$

(This is known as the Intercept Theorem or Thales Theorem).

Projection Onto Camera Plane

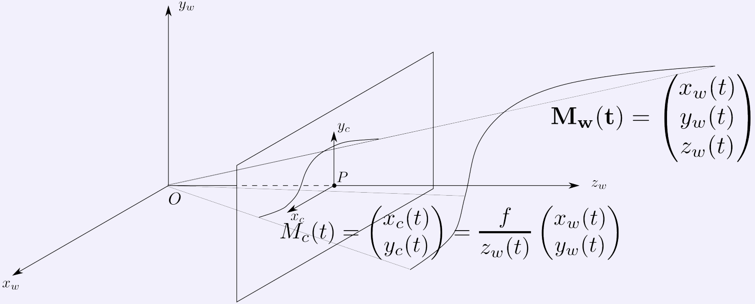
- The same is true for projection of a point in the 3D world to a point in the 2D camera plane.

$$P_w = \begin{pmatrix} x_w \\ y_w \\ z_w \end{pmatrix} \quad \Longrightarrow \quad P_c = \frac{f}{z_w}\begin{pmatrix} x_w \\ y_w \end{pmatrix}$$

- When P_w is moving with time, i.e., $P_w = P_w(t)$, $t =$ time parameter,

$$P_c(t) = \frac{f}{z_w(t)}\begin{pmatrix} x_w(t) \\ y_w(t) \end{pmatrix}$$

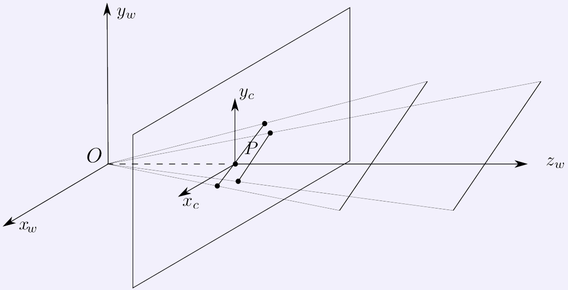
Pinhole Camera Model and Motion I



- Motion in the "world" induces motion on the camera plane. But the relation is not simple, $z_w(t)$ in the denominator.

Pinhole Camera Model and Motion II

- If z_w does not change, much easier, but it means that the motion trajectory is parallel to the camera plane.
- Two objects with same speeds, moving parallel to the xy-plane but with different z-values will show different apparent motion: the object farther to the camera plane appears to move slower: motion parallax.



- Other phenomena?

True vs Apparent Motion

- Apparent motion usually originates from true motion
- But there are ambiguities in apparent motion, e.g.,
- Motion parallax phenomenon or slower displacement of larger objects?
- Camera zooming in / out (or eq. object getting closer or farther from camera) vs. object changing size?
- Other image cues / perceptual a priori are used for disambiguation.
- But we can still be fooled – and computers too.

Outline

Image Content Change and Apparent Motion

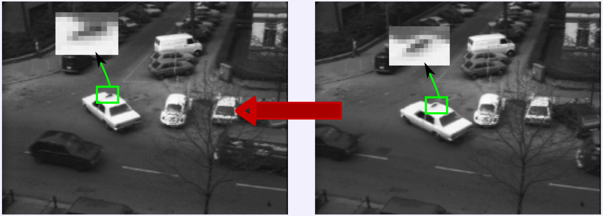
- Motion is observed via image content change:
- If a punctual object projected at pixel p moves with (projected) motion $\vec{v_p}$, it should be observed at pixel $p + \vec{v_p}$.
- Idea: A image dependent quantity $F(I)$ at position p in first image should be observed at position $p + \vec{v_p}$ in the second image:
$$F(I_1)(p) = F(I_2)(p + \vec{v_p})$$

Displaced Frame Difference

- Most usual choice: intensity of objects is preserved during motion:

$$I_1(p) = I_2(p + \vec{v}_p) \tag{3.1}$$

- Only valid under limited illumination conditions – the Lambertian Model.
- Equation (3.1) sometimes called Displaced Frame Difference Equation (DFDE)



Optical Flow Constraint Equation I

- Recall the Displaced Frame Difference Equation for $p = (x_0, y_0)$ and $\vec{v_p} = (v_{p1}, v_{p2})^T$

$$I_2(x_0 + v_{p1}, y_0 + v_{p2}) - I_1(x_0, y_0) = 0.$$

Apply the Taylor Formula:

$$I_2(x_0+v_{p1}, y_0+v_{p2})-I_1(x_0, y_0) \simeq \nabla_{(x_0,y_0)} I_2 \cdot \vec{v_p} + \underbrace{I_2(x_0, y_0) - I_1(x_0, y_0)}_{I_t}$$

- The quantity I_t is the "time-derivative" of the observed moving image. Equation above is called the Optical Flow Constraint Equation for the pair of images (I_1, I_2)

Optical Flow Constraint Equation II

- Consider that I_1 and I_2 are the observations of the Image Sequence $I(x, y, t)$ between time t_0 and $t_1 = t_0 + dt$ (dt small).
- Above formula can be rewritten as

$$I(p + \vec{v}, t + dt) - I(p, t) \approx \nabla_{(p,t)} I \cdot (v_1, v_2, dt)^T \approx 0$$

- This is the Optical Flow Constraint Equation (OFCE) for Image sequence $I(-, -, t)$ for a small displacement v_p.

Optical Flow Constraint Equation III

- Very similar (and equivalent). Assume that point p on the image plane moves with times: $p = p(t) = (x(t), y(t))$ and that intensities along trajectory is conserved

$$f(t) = I(p(t), t) = \text{constant}.$$

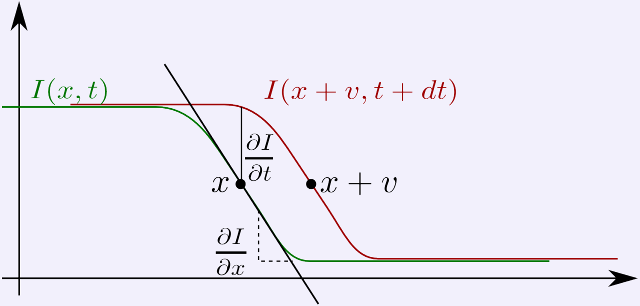
Then, by differentiation one gets

$$f'(t) = \nabla_{(p,t)} I \cdot \frac{dp}{dt} + \frac{\partial I}{\partial t} = 0$$

(The gradient is the spatial gradient here). This is another of the commonly used forms of the OFCE.

- Note that dp/dt here is the instantaneous velocity at time t (in pixels per second) while v_p from previous slide is a motion / displacement in pixels).

One-dimensional signals



OFCE in 1D reads

$$\frac{\partial I}{\partial x} v + \frac{\partial I}{\partial t} = 0 \quad v = -\frac{I_t}{I_x}$$

(I_x alternate notation for $\frac{\partial I}{\partial x}$, idem for t)

Aperture Problem

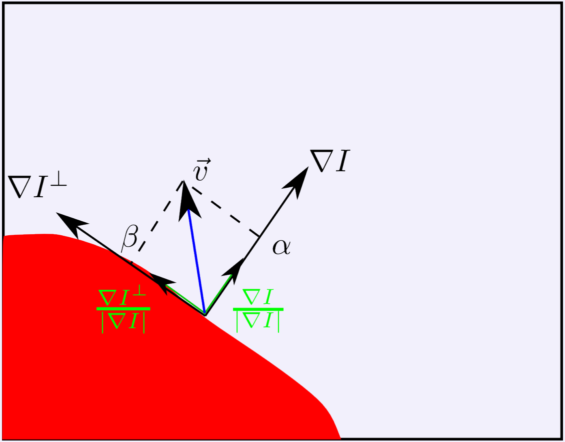
- In dimension 2, things are more complicated:

$$\frac{\partial I}{\partial x} v_1 + \frac{\partial I}{\partial y} v_2 + \frac{\partial I}{\partial t} = 0$$

- 1 equations for 2 unknowns! This is the punctual form of the aperture problem.
- Only the component of $\vec{v}$ parallel to ∇I can be computed

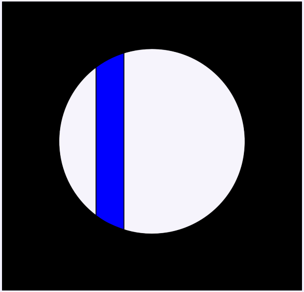
$$\vec{v} = (v_1, v_2)^T = \alpha \frac{\nabla I}{|\nabla I|} + \beta \frac{\nabla I^\perp}{|\nabla I|}$$

$$\alpha = -\frac{I_t}{|\nabla I|}$$


∇I
$\vec{v}$
$\nabla I^{\perp}$
β
α
$\frac{\nabla I^{\perp}}{|\nabla I|}$
$\frac{\nabla I}{|\nabla I|}$

Aperture Problem II

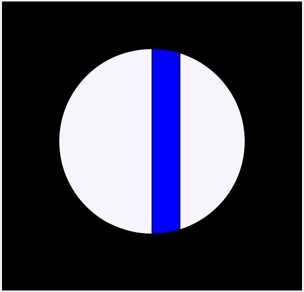
- Perception of apparent motion depends on structures and their size:



Moving bar structure Apparent motion

Aperture Problem II

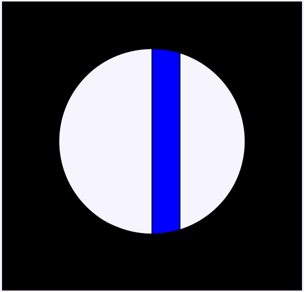
- Perception of apparent motion depends on structures and their size:



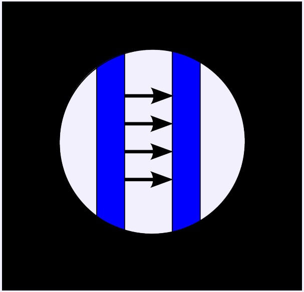
Moving bar structure Apparent motion

Aperture Problem II

- Perception of apparent motion depends on structures and their size:



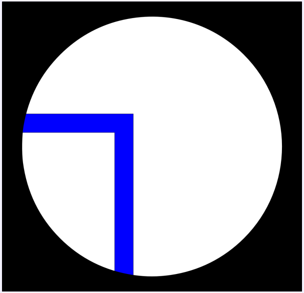
Moving bar structure



Apparent motion

Aperture Problem II

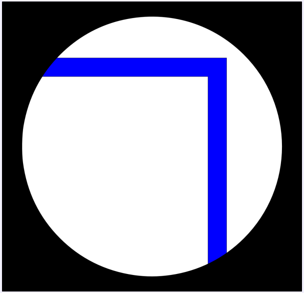
- By "increasing the aperture", more visible structure:



Moving corner structure Apparent motion

Aperture Problem II

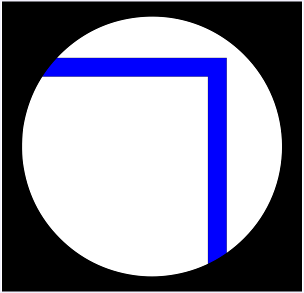
- By "increasing the aperture", more visible structure:



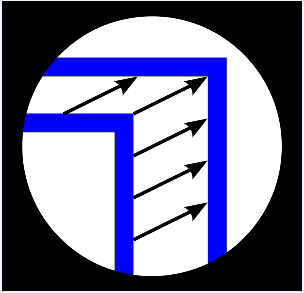
Moving corner structure Apparent motion

Aperture Problem II

- By “increasing the aperture”, more visible structure:



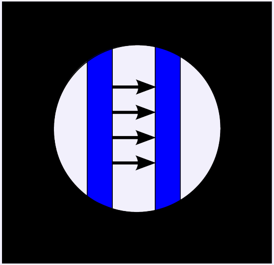
Moving corner structure

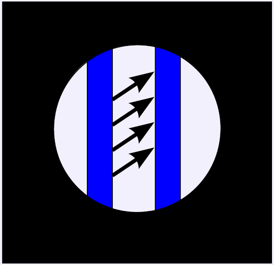


Apparent motion

Aperture Problem III

- In the absence of other cues, the simplest motion is perceived:

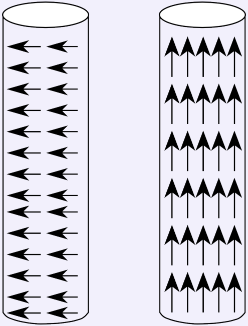




- In this case, perceived motion is the shortest, and orthogonal to structure.
- But these two motion vectors are equally valid. Their component orthogonal to the moving structure are equal and this is the component parallel to image gradient.
- Competition between cues and simplicity is part of optical flow algorithms.

Wrong Motion Perception: The Barber Pole Illusion

Pole is turning from right to left
Perceived motion is upward!

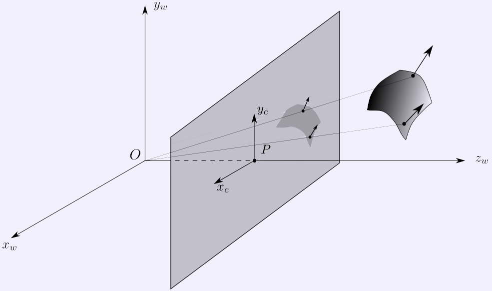


True and perceived motion

Outline

Important Principles Behind Recovery

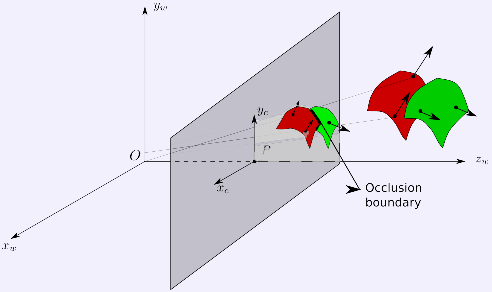
- Motion Coherence: pixels in an object of a scene move coherently:



- Motions vectors also do not change too much with time: temporal coherence (less used).
- Spatial coherence may solve the aperture problem.

Occlusions and Disocclusions

- Two or more objects moving in the scene at different depths and with different motions . One may partially hide another: Motion discontinuity.



- Using too large spatial coherence may use motions from different objects: need to find these boundaries. Usually are seen as image edges.
- But not all image edges are object boundaries. And some boundaries are not always clear.

Noise, Measurement Errors and Other Disturbances

Neither the DFDE $I_2(x + v_1, y + v_2) - I_1(x, y) = 0$ nor the OFCE $I_x v_1 + I_y v_2 + I_t = 0$ hold for most sequences / image pairs, even for “exact” motion vector $\vec{v} = (v_1, v_2)^T$. Several reasons For that:

- Noise alter pixel values:
- Subpixelic motion means partial pixel effects
- Change in lightning condition: no perfect Lambertian scenes
- Occlusion / disocclusion.
- Other reasons...?

How to Deal With Them

- Solution: use in least-squares settings: square-residual

$$(I_2(x+v_1, y+v_2) - I_1(x,y))^2$$

  should be as small as possible.
- Can also use absolute value residual

$$|I_x v_1 + I_y v_2 + I_t|$$

  as small as possible (better for occlusion / disocclusion).
- Robust statistic approach too.
- We only consider least squares approaches here.

Algorithms for Recovery

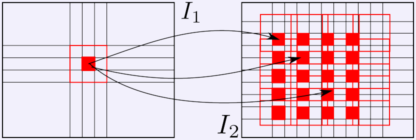
- Many families of algorithms for motion recovery.
- Since 1980, more than 3000 papers!
- We briefly look at three “grand old” classical ones:

1. Block Matching.
2. The Lucas and Kanade approach.
3. The Horn and Schunck approach.

Block Matching

- A conceptually very simple algorithm, use patch matching.
- Matching patches can solve for the aperture problem (though patch size can be an issue).
- Not always very precise, has difficulties to capture complex motion behaviors.
- Very fast and a lot of variations exist.
- Used in video compression (behind many mpeg-type encoders).

- Create a small block around 1 pixel in image I_1 Search for a similar block in image I_2 usually by minimizing sum of squares differences (SSD).



- "Kernel" block of size $s \times s$ (s usually odd, $s = 2k + 1$ with block centered around pixel $p = (x, y)$)

- Search window of size $(2\ell+1)\times(2\ell+1)$ (provides max displacement allowed) in image 2 centered at position (x, y).

- With odd size kernel size $2k+1$, score to minimize:

$$m_{v_1,v_2} = \sum_{i=-k}^{k}\sum_{j=-k}^{k}\left(I_1(x+i, y+j) - I_2(x+i+v_1, y+j+v_2)\right)^2$$

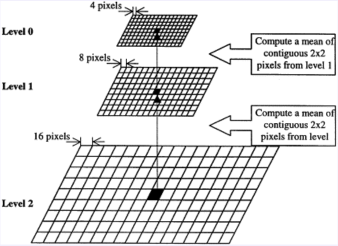
$$v_1 = -\ell\ldots\ell,\quad v_2 = -\ell\ldots\ell$$

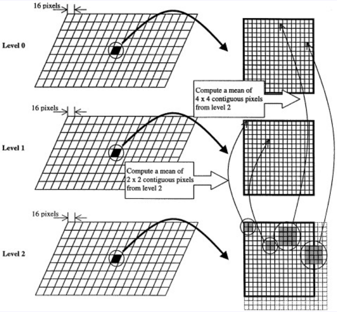
- Minimizing above score very similar to maximizing the correlation between patches.
- Given by

$$c(v_1, v_2) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} I_1(x+i, y+j) I_2(x+i+v_1, y+j+v_2)$$

- This a dot product between a fixed patch and a moving one!
- Normalize correlation could be used instead – provides the cosine of the angles between the fixed patch in image I_1 and the moving patch in image I_2.
- Correlation can be implemented fast using Fast Fourier Transform.

- Large displacements means a large search space ($\ell >> 0$)
- It can be reduced by a pyramid search approach (smoothing and downsampling)





Lucas and Kanade Algorithm

- The Lucas and Kanade approach is a least square approach. Assumes that displacement around at pixel p is small and approximately constant in a neighborhood of pixel p.
- Collect OFCE based equations for $p' \in W(p)$, $W(p)$ a window centered at p and solve for $\vec{v}$ such that

$$\sum_{p' \in W(p)} \left(I_x(p')v_1 + I_y(p')v_2 + I_t(p')\right)^2 = \min$$

- Classical least-square theory (or simple differential calculus) provides equation:

$$M\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \mathbf{r}$$

with

$$M = \sum_{p' \in W(p)} \begin{pmatrix} I_x^2(p') & I_x(p')I_y(p') \\ I_x(p')I_y(p') & I_y^2(p') \end{pmatrix}$$

$$\mathbf{r} = \sum_{p' \in W(p)} \begin{pmatrix} -I_x(p')I_t(p') \\ -I_y(p')I_t(p') \end{pmatrix}$$

- Instead of a standard window, W taken as a Gaussian window centered at p: the contribution of pixel p' is weighted by a Gaussian factor $w(p') \propto e^{-\frac{\|p'-p\|^2}{2\sigma^2}}$
- Equation to solve becomes

$$\underbrace{\sum_{p'} w(p') \begin{pmatrix} I_x^2(p') & I_x(p')I_y(p') \\ I_x(p')I_y(p') & I_y^2(p') \end{pmatrix}}_{J_\sigma(p)} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \underbrace{\sum_{p'} w(p) \begin{pmatrix} -I_x(p')I_t(p') \\ -I_y(p')I_t(p') \end{pmatrix}}_{L_\sigma(p)}$$

- The matrix $J_\sigma(p)$ is the structure tensor at p, the same used for Harris interest points!
- The vector $L_\sigma(p)$ contains the spatiotemporal related parts (it can be used to extend the structure tensor to a spatiotemporal tensor).

Algorithm is very simple:

- Compute the structure tensor image J_σ (derivatives + Gaussian Smoothing / convolution)
- Compute the spatiotemporal part L_σ (also derivatives + Gaussian Smoothing / Convolution)
- For each p in image, compute $\vec{v}_p$ as solution of 2×2 system of equations

$$J_\sigma(p)\vec{v}_p = L_\sigma(p)$$

With simple finite difference implementation and/or Gaussian filtering (and Gaussian derivatives), a few lines in Python (or Matlab!)

- σ is a scale parameter for structures. A small σ means solving very locally around p, with risk of aperture problem persisting. A large σ removes aperture problem but might integrate incompatible data – different objects with different motion, (dis)occlusion...
- The assumption of small motion often violated. A hierarchical / pyramid approach as in Lauze-Kornprobst-Mémin 2004, possible.
- Some Extension to handle complex measurement noise include a so-called Total-Least-Squares approach and more complex smoothing than Gaussian smoothing for computing structure tensor.
- Some of these modification have also a hierarchical / pyramid implementation.

The Horn and Schunck Approach

- The Horn and Schunck approach is least-squares based and attempts to compute a smooth motion vector field
- It uses the OFCE
- is assumes that two motion vectors are similar (small variations between them)
- It solves a regularized least-squares problem

$$\min_{\vec{v}} \sum_p \left(I_x(p)v_1 + I_y(p)v_2 + I_t(p)\right)^2 + \alpha \sum_p \sum_{p' \sim p} \|\vec{v}_{p'} - \vec{v}_p\|^2$$

($p' \sim p$ means p' neighbor of p)

- The vector part $\|\vec{v}_{p'} - \vec{v}_p\|^2 = (v_{1p'} - v_{1p})^2 + (v_{2p'} - v_{2p})^2$.

- This is a typical trade-off problem: the first part means to stick as much as possible to the observed data, the second part means that the solution should be simple (smooth).
- The parameter α controls the smoothing of the solution.
- A high α means a very smooth solution, a small α means sticking better to the observed data.
- This second part adds the equation missing from the aperture problem!
- However, discontinuities mean that the difference $\|\vec{v}_{p'} - \vec{v}_p\|^2$ may become very large: not favored by a solution.
- H & S well known to smooth discontinuities, but can still provide pretty good results.

Solving for the Horn and Schunck Flow

- Least-square theory provides the associated normal equations for the vector field minimizing the Horn and Schunck criterion. There are two families of coupled equations, one for $p \mapsto v_{1p}$, the other for $p \mapsto v_{2p}$.

$$\begin{cases} I_x(p)\left(I_x(p)v_{1p} + I_y(p)v_{2p} + I_t(p)\right) + \alpha \sum_{p' \sim p}\left(v_{1p} - v_{1p'}\right) & = 0 \\ I_y(p)\left(I_x(p)v_{1p} + I_y(p)v_{2p} + I_t(p)\right) + \alpha \sum_{p' \sim p}\left(v_{2p} - v_{2p'}\right) & = 0 \end{cases}$$

- Usually take a 4-points neighborhood around p denoted n, e, s and w (for north-east-south-west), so that

$$\begin{aligned} \alpha \sum_{p' \sim p}\left(v_{ip} - v_{ip'}\right) &= 4\alpha v_{ip} - 4\alpha \frac{v_{in} + v_{ie} + v_{is} + v_{iw}}{4} \\ &= 4\alpha\left(v_{ip} - \bar{v}_{ip}\right) \end{aligned}$$

with $\bar{v}_{ip}$ being the average neighbor values of v_{ip}.

- The system can be rewritten (dropping the p) as

$$\begin{cases} \left(I_x^2 + 4\alpha\right) v_1 + I_x I_t v_2 & = 4\alpha \bar{v}_1 - I_x I_t \\ I_y I_t v_1 + \left(I_y^2 + 4\alpha\right) v_2 & = 4\alpha \bar{v}_2 - I_y I_t \end{cases}$$

or in matrix notation

$$\begin{pmatrix} I_x^2 + 4\alpha & I_x I_t \\ I_y I_t & I_y^2 + 4\alpha \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} 4\alpha \bar{v}_1 - I_x I_t \\ 4\alpha \bar{v}_2 - I_y I_t \end{pmatrix}$$

- Easy to solve, but solution at p depends on neighbor values which are found by solving a 2x2 system depending on their neighbor values which ...

- Iterative solution:
- Repeat until no change
  1. For p in image do
  2. Assume values at neighbor of p fixed (just for that visit)
  3. solve the system:
  4. Immediately replace old values at p by new ones
- This is a example of a Relaxation solver, more precisely, a Gauss-Seidel system solver.
- For the Gauss-Seidel problem, it is guaranteed to converge.

Outline

Conclusion

- Optical flow is the pattern apparent motion caused by relative motion of scene objects and camera.
- It can only be observed indirectly via changes in brightness / color and derived quantities in recorded images.
- In 2D and more, aperture problem causes indetermination of the flow at small scale.
- Flow recovery integrate data at larger scale to solve for it.
- A proper balance between solving for aperture problem and not going through occlusion boundaries is needed.
- Many more information at the Middelbury Optical Flow Database `http://vision.middlebury.edu/flow`

Short Bibliography (Uploaded in Absalon)

HS. B. K. Horn and B. G. Schunck. **Determining Optical Flow.**, Artificial Intelligence, 17: 185–203 (1981).

LK. B. D. Lucas and T. Kanade. **An Iterative Image Registration Technique with an Application to Stereo Vision**. Proceedings of Image Understanding Workshop: 121–130 (1981=.

VP. A. Verri and T. Poggio. **Motion Field and Optical Flow: Qualitative Properties.** IEEE Trans. Pattern Anal. Mach. Intell, 11(5): 490–498 (1989).

LKM. F. Lauze, P. Kornprobst and E. Mémin. **A Coarse-to-Fine Multiscale Approach for Linear Least Squares Optical Flow Estimation.** Proceedings of The British Machine Vision Conference, 2: 777-787 (2004).

Outline

Aparté: Recalls From Differential Calculus I

- Derivative (1D)

$$\lim_{h\to 0}\frac{f(x+h)-f(x)}{h} =: f'(x)$$

- This means that for small h, one has the Taylor Formula

$$f(x+h)-f(x)\approx hf'(x)$$

- Differential / Gradient and 2D Taylor Formula

$$\begin{aligned} f(x_0+h_1,y_0+h_2)-f(x_0,y_0) &\approx h_1\frac{\partial f}{\partial x}(x_0,y_0)+h_2\frac{\partial f}{\partial y}(x_0,y_0)\\ &= \nabla_{(x_0,y_0)}f\cdot(h_1,h_2)^T \end{aligned}$$

- Taylor Formula in 3D:

$$\begin{aligned} & f(x_0+h_1,y_0+h_2,z_0+h_3)-f(x_0,y_0,z_0)\approx\\ & h_1\frac{\partial f}{\partial x}(x_0,y_0)+h_2\frac{\partial f}{\partial y}(x_0,y_0)+h_3\frac{\partial f}{\partial y}(x_0,y_0,z_0)\\ & = \nabla_{(x_0,y_0,z_0)}f\cdot(h_1,h_2,h_3)^T \end{aligned}$$

Aparté: Recalls From Differential Calculus II

- The term $\nabla_{(x_0,y_0)} f$ is the 2D gradient of f at (x_0, y_0):

$$\nabla_{(x_0,y_0)} f = \begin{pmatrix} \frac{\partial f}{\partial x}(x_0, y_0) \\ \frac{\partial f}{\partial y}(x_0, y_0) \end{pmatrix}$$

- The term $\nabla_{(x_0,y_0,z_0)} f$ is the 3D (or spatiotemporal) gradient of f at (x_0, y_0, z_0)

$$\nabla_{(x_0,y_0)} f = \begin{pmatrix} \frac{\partial f}{\partial x}(x_0, y_0, z_0) \\ \frac{\partial f}{\partial y}(x_0, y_0, z_0) \\ \frac{\partial f}{\partial z}(x_0, y_0, z_0) \end{pmatrix}$$