w08b — Optical flow: solution

Q1. Brightness constancy and the aperture problem.

A small patch keeps its intensity as it moves: $I(x, y, t) = I(x + u,\ y + v,\ t + 1)$. First-order Taylor expansion gives the optical flow constraint

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

One equation, two unknowns per pixel ⇒ the **aperture problem**. Only the flow component **parallel to the gradient** (normal to the edge) is recoverable; the component along the edge is unconstrained. The classic illustration is a moving grating seen through a small hole.

Q2. Where they have met M.

M is the **structure tensor**, used in **Week 8's corner detectors** (Harris and Shi-Tomasi, exercise `w08a`), where its eigenvalues decide whether a pixel is a corner.

The same matrix appears here because the least-squares solution to the flow system is well conditioned exactly when M is well conditioned. That is the link worth naming explicitly; it is the reason the two topics share a week.

This also explains the phrase **"good features to track"**. Shi and Tomasi's criterion is $\min(\lambda_1, \lambda_2)$, which is precisely the condition for M to be invertible and well conditioned, i.e. for the flow equations to have a stable unique solution. So the points that are good corners are, by exactly the same test, the points where motion can be measured reliably. Corner detection and feature tracking are two readings of one criterion.

Q3. The eigenvalue table.

Window contents	λ_1	λ_2	Flow recoverable?
A corner	large	large	**Yes** — M invertible and well conditioned; both components
A long straight edge	large	≈ 0	**Partly** — normal component only; M singular (aperture problem)
A textureless patch	≈ 0	≈ 0	**No** — $M \approx 0$, no information at all

Q4. 70-pixel motion with a 21×21 window.

The Taylor expansion behind the flow constraint is only valid for displacements small relative to the window and to the scale of local texture. A 70-pixel motion moves the patch entirely outside the window: the algorithm never "sees" the true correspondence, the linearisation is meaningless, and the estimate collapses to a small, wrong value. (Measured in Lab 7: single-scale LK reports about 26 px where the true displacement is about 70 px.)

The pyramid fix. Build a Gaussian pyramid. At level k the displacement is $70/2^k$ pixels — at level 3 that is roughly 9 px, comfortably within the window. Solve at the coarsest level, upscale the estimate by 2, warp the window by it, and refine at the next level down. Each refinement only has to explain a small residual, so the linearisation stays valid throughout.