HW 5 and 6 - Motion Tracking and Lucas–Kanade with Affine Motion

Amir Sabbaghziarani - 002852171

(a) Derivation of the Motion Tracking Equation (Optical Flow Constraint)

Image formation and motion

Consider a grayscale video sequence represented by an intensity function

$$I(x, y, t),$$

where

- (x, y) denotes the spatial image coordinates, and
- t denotes time (frame index or continuous time).

Let a physical point in the scene project to pixel location $(x(t), y(t))$ at time t. After a short time interval Δt, this point moves to a new image location

$$(x(t + \Delta t), y(t + \Delta t)) = (x(t) + u\,\Delta t,\ y(t) + v\,\Delta t),$$

where

$$u = \frac{dx}{dt}, \qquad v = \frac{dy}{dt}$$

are the horizontal and vertical components of the image velocity (optical flow) at that point.

Brightness constancy assumption

The fundamental assumption used to derive the motion tracking equation is the *brightness constancy* assumption:

> *The apparent brightness of a moving point stays constant over a short time interval.*

Formally, for a point moving with the image velocity (u, v),

$$I(x(t), y(t), t) = I(x(t + \Delta t), y(t + \Delta t), t + \Delta t).$$

Using the notation $x = x(t)$ and $y = y(t)$ for brevity, this can be written as

$$I(x, y, t) = I(x + u\,\Delta t,\ y + v\,\Delta t,\ t + \Delta t). \tag{1}$$

First-order Taylor expansion

Assuming that the motion and the time step Δt are small, expand the right-hand side of (1) using a first-order Taylor expansion around (x, y, t):

$$I(x + u\,\Delta t,\ y + v\,\Delta t,\ t + \Delta t) \approx I(x, y, t) + I_x(x, y, t)\,(u\,\Delta t) + I_y(x, y, t)\,(v\,\Delta t) + I_t(x, y, t)\,\Delta t,$$

where

$$I_x = \frac{\partial I}{\partial x}, \qquad I_y = \frac{\partial I}{\partial y}, \qquad I_t = \frac{\partial I}{\partial t}$$

are partial derivatives of the image intensity.

Substitute this approximation into (1):

$$\begin{aligned} I(x, y, t) &= I(x + u\,\Delta t,\ y + v\,\Delta t,\ t + \Delta t) \\ &\approx I(x, y, t) + I_x\,u\,\Delta t + I_y\,v\,\Delta t + I_t\,\Delta t. \end{aligned}$$

Subtract $I(x, y, t)$ from both sides:

$$0 \approx I_x\,u\,\Delta t + I_y\,v\,\Delta t + I_t\,\Delta t.$$

Factor out Δt (which is nonzero and small) and divide both sides by Δt:

$$0 \approx I_x\,u + I_y\,v + I_t.$$

Motion tracking equation (optical flow constraint)

In the limit as $\Delta t \to 0$, the approximation becomes an equality, yielding the *optical flow constraint equation* (also called the *motion tracking equation*):

$$I_x(x, y, t)\,u(x, y, t) + I_y(x, y, t)\,v(x, y, t) + I_t(x, y, t) = 0. \tag{2}$$

In vector form, let

$$\nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix}, \qquad \mathbf{v} = \begin{bmatrix} u \\ v \end{bmatrix}.$$

Then (2) can be written compactly as

$$\nabla I^\top \mathbf{v} + I_t = 0.$$

This equation expresses that the component of the motion vector (u, v) along the spatial intensity gradient ∇I is constrained by the temporal change in intensity I_t. Because there is a single scalar equation for two unknowns u and v, additional assumptions or constraints (such as spatial smoothness or parametric motion models) are required to uniquely determine the motion.

(b) Lucas–Kanade Algorithm for Affine Motion

Now suppose that the motion of a local image patch is not just arbitrary at each pixel, but follows an *affine* motion model. The Lucas–Kanade algorithm estimates the parameters of such motion by minimizing the discrepancy between two images over a small window.

Affine motion model

Assume that the motion field in a neighborhood can be expressed as an affine function of the pixel coordinates (x, y):

$$u(x, y) = a_1 x + b_1 y + c_1, \tag{3}$$
$$v(x, y) = a_2 x + b_2 y + c_2. \tag{4}$$

The six parameters

$$\mathbf{p} = \begin{bmatrix} a_1 & b_1 & c_1 & a_2 & b_2 & c_2 \end{bmatrix}^\top$$

describe the local affine transformation (including translation, rotation, shearing, and scaling).

Optical flow constraint with affine motion

Insert the affine motion model into the optical flow constraint equation (2). At each pixel (x, y),

$$I_x(x, y)\, u(x, y) + I_y(x, y)\, v(x, y) + I_t(x, y) = 0. \tag{5}$$

Substituting (3) and (4):

$$I_x(x, y)(a_1 x + b_1 y + c_1) + I_y(x, y)(a_2 x + b_2 y + c_2) + I_t(x, y) = 0.$$

Rearrange to make the dependence on the parameters explicit. At a pixel (x, y), define the row vector

$$\mathbf{A}(x, y) = \begin{bmatrix} I_x x & I_x y & I_x & I_y x & I_y y & I_y \end{bmatrix},$$

and the parameter vector

$$\mathbf{p} = \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2 \end{bmatrix}.$$

Then the constraint at that pixel can be written as

$$\mathbf{A}(x, y)\, \mathbf{p} + I_t(x, y) = 0. \tag{6}$$

For a single pixel, this is still one equation in six unknowns. However, the same parameters $\mathbf{p}$ must satisfy the constraint for all pixels in a local neighborhood (or window) W around the point of interest.

Least-squares formulation (Lucas–Kanade idea)

The Lucas–Kanade method assumes that *within a small window W, the motion parameters are constant.* This gives one linear equation (6) for each pixel $(x, y) \in W$, all sharing the same parameters $\mathbf{p}$. In practice the data are noisy and the brightness constancy assumption is only approximate, so one solves the system in the least-squares sense.

Define the residual at pixel (x, y):

$$r(x, y; \mathbf{p}) = \mathbf{A}(x, y)\, \mathbf{p} + I_t(x, y).$$

To estimate $\mathbf{p}$, minimize the sum of squared residuals over the window:

$$E(\mathbf{p}) = \sum_{(x,y)\in W} r(x,y;\mathbf{p})^2 = \sum_{(x,y)\in W} \big(\mathbf{A}(x,y)\,\mathbf{p} + I_t(x,y)\big)^2. \tag{7}$$

This is a standard linear least-squares problem in $\mathbf{p}$.

Normal equations for affine Lucas–Kanade

To minimize (7), set the gradient of $E(\mathbf{p})$ with respect to $\mathbf{p}$ to zero:

$$\frac{\partial E}{\partial \mathbf{p}} = 2 \sum_{(x,y)\in W} r(x,y;\mathbf{p})\,\mathbf{A}(x,y)^\top = \mathbf{0}.$$

Substitute $r(x,y;\mathbf{p}) = \mathbf{A}(x,y)\,\mathbf{p} + I_t(x,y)$:

$$\sum_{(x,y)\in W} \big(\mathbf{A}(x,y)\,\mathbf{p} + I_t(x,y)\big)\,\mathbf{A}(x,y)^\top = \mathbf{0}.$$

Rearrange terms:

$$\left(\sum_{(x,y)\in W} \mathbf{A}(x,y)^\top \mathbf{A}(x,y)\right)\mathbf{p} = -\sum_{(x,y)\in W} \mathbf{A}(x,y)^\top I_t(x,y).$$

Define

$$\mathbf{H} = \sum_{(x,y)\in W} \mathbf{A}(x,y)^\top \mathbf{A}(x,y) \quad \text{(a } 6\times 6 \text{ matrix, the Hessian or structure tensor for affine parameters)},$$

and

$$\mathbf{b} = -\sum_{(x,y)\in W} \mathbf{A}(x,y)^\top I_t(x,y) \quad \text{(a } 6\times 1 \text{ vector)}.$$

Then the normal equations become

$$\mathbf{H}\,\mathbf{p} = \mathbf{b}. \tag{8}$$

If $\mathbf{H}$ is invertible (that is, the window has sufficient texture and is not degenerate), the solution is

$$\mathbf{p} = \mathbf{H}^{-1}\,\mathbf{b}. \tag{9}$$

This gives the least-squares estimate of the affine motion parameters for the local window.

Explicit form of the matrix HH and vector bb

For clarity, expand $\mathbf{A}(x,y)$:

$$\mathbf{A}(x,y) = \begin{bmatrix} I_x x & I_x y & I_x & I_y x & I_y y & I_y \end{bmatrix}.$$

Then

$$\mathbf{A}(x,y)^\top \mathbf{A}(x,y) = \begin{bmatrix} (I_x x)^2 & I_x^2 xy & I_x^2 x & I_x I_y x^2 & I_x I_y xy & I_x I_y x \\ I_x^2 xy & (I_x y)^2 & I_x^2 y & I_x I_y xy & I_x I_y y^2 & I_x I_y y \\ I_x^2 x & I_x^2 y & I_x^2 & I_x I_y x & I_x I_y y & I_x I_y \\ I_x I_y x^2 & I_x I_y xy & I_x I_y x & (I_y x)^2 & I_y^2 xy & I_y^2 x \\ I_x I_y xy & I_x I_y y^2 & I_x I_y y & I_y^2 xy & (I_y y)^2 & I_y^2 y \\ I_x I_y x & I_x I_y y & I_x I_y & I_y^2 x & I_y^2 y & I_y^2 \end{bmatrix},$$

and $\mathbf{H}$ is the sum of these matrices over all pixels in W.

Similarly,

$$\mathbf{A}(x,y)^\top I_t(x,y) = \begin{bmatrix} I_x x I_t \\ I_x y I_t \\ I_x I_t \\ I_y x I_t \\ I_y y I_t \\ I_y I_t \end{bmatrix},$$

so

$$\mathbf{b} = -\sum_{(x,y)\in W} \begin{bmatrix} I_x x I_t \\ I_x y I_t \\ I_x I_t \\ I_y x I_t \\ I_y y I_t \\ I_y I_t \end{bmatrix}.$$

Iterative Lucas–Kanade procedure with the affine model

In practice, the motion between the two images may not be infinitesimally small. Lucas–Kanade is therefore implemented as an *iterative* algorithm that refines the estimate of $\mathbf{p}$ by repeatedly solving a linearized problem. A high level procedure is as follows.

Consider two images:

$$I_1(x,y) \quad \text{(reference image, at time } t\text{)},$$

$$I_2(x,y) \quad \text{(next image, at time } t+\Delta t\text{)}.$$

The goal is to find affine parameters $\mathbf{p}$ such that a window W in I_1 matches a transformed window in I_2.

Warp function

Define the affine warp function

$$W(x,y;\mathbf{p}) = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x + u(x,y) \\ y + v(x,y) \end{bmatrix} = \begin{bmatrix} x + a_1 x + b_1 y + c_1 \\ y + a_2 x + b_2 y + c_2 \end{bmatrix}.$$

Equivalently, in homogeneous coordinates,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1+a_1 & b_1 & c_1 \\ a_2 & 1+b_2 & c_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

Energy to minimize

Define the registration error between the template window in I_1 and the warped window in I_2:

$$E(\mathbf{p}) = \sum_{(x,y)\in W} \big(I_2\big(W(x,y;\mathbf{p})\big) - I_1(x,y)\big)^2.$$

The Lucas–Kanade method minimizes this energy by iterative linearization.

Linearization and incremental update

Assume that we have a current estimate $\mathbf{p}$ and want to compute a small increment $\Delta\mathbf{p}$. Consider the energy

$$E(\mathbf{p}+\Delta\mathbf{p}) = \sum_{(x,y)\in W} \big(I_2\big(W(x,y;\mathbf{p}+\Delta\mathbf{p})\big) - I_1(x,y)\big)^2 .$$

Linearize the warped image $I_2\big(W(x,y;\mathbf{p}+\Delta\mathbf{p})\big)$ around $\mathbf{p}$ using a first-order Taylor expansion in $\Delta\mathbf{p}$:

$$I_2\big(W(x,y;\mathbf{p}+\Delta\mathbf{p})\big) \approx I_2\big(W(x,y;\mathbf{p})\big) + \nabla I_2\big(W(x,y;\mathbf{p})\big)^\top \frac{\partial W(x,y;\mathbf{p})}{\partial \mathbf{p}} \Delta\mathbf{p}.$$

Here,

- $\nabla I_2 = [I_{2x}, I_{2y}]^\top$ is the spatial gradient of I_2, evaluated at the warped coordinates,
- $\dfrac{\partial W(x,y;\mathbf{p})}{\partial \mathbf{p}}$ is the Jacobian of the warp with respect to the parameters.

For the affine warp, the Jacobian is

$$\frac{\partial W(x,y;\mathbf{p})}{\partial \mathbf{p}} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix}.$$

The *steepest descent image* row at pixel (x,y) is

$$\mathbf{J}(x,y) = \nabla I_2\big(W(x,y;\mathbf{p})\big)^\top \frac{\partial W(x,y;\mathbf{p})}{\partial \mathbf{p}} = \begin{bmatrix} I_{2x}x & I_{2x}y & I_{2x} & I_{2y}x & I_{2y}y & I_{2y} \end{bmatrix},$$

which has exactly the same structure as $\mathbf{A}(x,y)$ defined earlier in terms of I_x and I_y.

Define the current residual

$$e(x,y;\mathbf{p}) = I_2\big(W(x,y;\mathbf{p})\big) - I_1(x,y).$$

Then the linearized energy in terms of $\Delta\mathbf{p}$ is

$$E(\mathbf{p}+\Delta\mathbf{p}) \approx \sum_{(x,y)\in W} \left(e(x,y;\mathbf{p}) + \mathbf{J}(x,y)\,\Delta\mathbf{p}\right)^2 .$$

Minimizing this quadratic function in $\Delta\mathbf{p}$ leads to the normal equations

$$\left(\sum_{(x,y)\in W} \mathbf{J}(x,y)^\top \mathbf{J}(x,y)\right)\Delta\mathbf{p} = -\sum_{(x,y)\in W} \mathbf{J}(x,y)^\top e(x,y;\mathbf{p}).$$

Define

$$\mathbf{H} = \sum_{(x,y)\in W} \mathbf{J}(x,y)^\top \mathbf{J}(x,y), \qquad \mathbf{g} = \sum_{(x,y)\in W} \mathbf{J}(x,y)^\top e(x,y;\mathbf{p}),$$

then

$$\mathbf{H}\,\Delta\mathbf{p} = -\mathbf{g},$$

and the update is

$$\Delta\mathbf{p} = -\mathbf{H}^{-1}\mathbf{g}.$$

Finally, update the parameters:

$$\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p},$$

and iterate until $\Delta\mathbf{p}$ is sufficiently small or a maximum number of iterations is reached.

Step-by-step summary of Lucas–Kanade with affine motion

For each feature (or window) to be tracked between I_1 and I_2:

1. **Initialization:**
   - Choose a window W around the point of interest in I_1.
   - Initialize the affine parameters $\mathbf{p}^{(0)} = \mathbf{0}$ (or use a prediction from previous frames).
2. **Precomputation (optional, for efficiency):**
   - Compute image gradients I_{2x} and I_{2y} for I_2.
3. **Iterative refinement (for $k = 0, 1, 2, \dots$):**

   (a) Warp the coordinates of W using the current parameters $\mathbf{p}^{(k)}$:
   $$(x', y') = W(x, y; \mathbf{p}^{(k)}).$$

   (b) Sample I_2 at the warped coordinates to obtain $I_2(x', y')$.

   (c) Compute the error image:
   $$e(x, y; \mathbf{p}^{(k)}) = I_2\big(W(x, y; \mathbf{p}^{(k)})\big) - I_1(x, y).$$

   (d) Evaluate image gradients of I_2 at the warped coordinates, $I_{2x}(x', y')$ and $I_{2y}(x', y')$.

   (e) For each pixel $(x, y) \in W$, form the steepest descent row
   $$\mathbf{J}(x, y) = \begin{bmatrix} I_{2x}x & I_{2x}y & I_{2x} & I_{2y}x & I_{2y}y & I_{2y} \end{bmatrix}.$$

   (f) Accumulate the Hessian and gradient:
   $$\mathbf{H} = \sum_{(x,y)\in W} \mathbf{J}(x, y)^\top \mathbf{J}(x, y), \qquad \mathbf{g} = \sum_{(x,y)\in W} \mathbf{J}(x, y)^\top e(x, y; \mathbf{p}^{(k)}).$$

   (g) Solve for the parameter increment:
   $$\Delta\mathbf{p} = -\mathbf{H}^{-1}\mathbf{g}.$$

   (h) Update the parameters:
   $$\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} + \Delta\mathbf{p}.$$

   (i) **Stopping criterion:** If $\|\Delta\mathbf{p}\|$ is below a small threshold or the maximum number of iterations is reached, stop.
4. **Result:** The final parameters $\mathbf{p}$ define the affine motion:
   $$u(x, y) = a_1 x + b_1 y + c_1, \qquad v(x, y) = a_2 x + b_2 y + c_2,$$
   and the corresponding affine warp $W(x, y; \mathbf{p})$ describes how the patch in I_1 moves into I_2.

This completes the derivation of the motion tracking equation from fundamental principles and the derivation of the Lucas–Kanade algorithm when the motion is modeled as affine.