OPTICAL FLOW

The optical flow is a motion estimation method, it deals with motion in images (image frame). It studies the image changes as the objects move or the camera moves. The optical flow is estimated from a sequence of images by assuming constant intensity (brightness constancy), This implies that pixels do not change intensity when their location changes with time. This assumption is expressed by the following equation

$$\frac{I(x(t), y(t), t)}{dt} = 0$$

Note that

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

is the total derivative of the intensity function. After taking the partial derivatives, the brightness constancy is expressed as follows:

$$\frac{I(x(t), y(t), t)}{dt} = \frac{\partial I}{\partial x} \frac{dx}{dt} + \frac{\partial I}{\partial y} \frac{dy}{dt} + \frac{\partial I}{\partial t} \frac{dt}{dt}$$

This equation gives the relationship between the partial derivatives. It can be written as

$$I_x v_x + I_y v_y + I_t = 0$$

where $I_x, I_y, I_t$ are the partial derivatives given by

$$I_x = \frac{\partial I}{\partial x}$$

$$I_y = \frac{\partial I}{\partial y}$$

$$I_t = \frac{\partial I}{\partial t}$$

Equation (5) is called the intensity change constraint equation. The constraint equation applies to every pixel in each image frame. We have

- $\nabla I = (I_x, I_y)$ intensity gradient
- $v = (v_x, v_y)$ pixel velocity components.
- $I_t$: time rate of change of the intensity at that pixel of the image.

**Example 1**

Let

$$f(x(t), y(t), t) = 2x^2(t) + 3y(t) + 3t$$

Find the constraint equation.

By taking the partial derivatives, we can write:

$$4x\dot{x} + 3\dot{y} + 3 = 0$$

**Example 2**

For the cube of figure 2, find $I_x, I_y, I_t$. Write and plot the constraint equation.

We have from the cube

$$I_x = 0.25(2 + 4 + 6 + 7) - 0.25(3 + 1 + 5 + 8) = 0.5$$

$$I_y = 0.25(3 + 4 + 7 + 8) - 0.25(1 + 2 + 5 + 6) = 2$$

$$I_t = 0.25(5 + 6 + 7 + 8) - 0.25(1 + 2 + 3 + 4) = 4$$

The constraint equation is

$$0.5v_x + 2v_y = -4$$

Figure 3 shows of the constraint equation.
Fig. 2. Example for computing the partial derivatives

Fig. 3. The constraint equation for example 2.

Analysis of the constraint

A single constraint does not provide enough information to solve for \( v_x, v_y \). The plot of the constraint for the previous example is shown in figure 3. All points on the line are valid solutions for \((v_x, v_y)\) i.e., there is an infinite number of solutions.

Using brightness intensity gradients at two points

To determine \( v_x \) and \( v_y \) we can use two adjacent points: \((x_1, y_1), (x_2, y_2)\). The following assumptions are made:
- Local gradients are different.
- The image velocity is the same.
Under these assumptions we can write

\[
v_x I_{x1} + v_y I_{y1} = -I_{t1} \quad (15)
v_x I_{x2} + v_y I_{y2} = -I_{t2} \quad (16)
\]

We can write the 2 equations compactly in matrix form:
\[
Mv = b \quad (17)
\]
where
\[
v = \begin{bmatrix} v_x \\ v_y \end{bmatrix};
M = \begin{bmatrix} I_{x1} & I_{y1} \\ I_{x2} & I_{y2} \end{bmatrix};
b = \begin{bmatrix} -I_{t1} \\ -I_{t2} \end{bmatrix} \quad (18)
\]

Let
\[
\Delta = I_{x1}I_{y2} - I_{x2}I_{y1} \quad (19)
\]

There is a solution only and only if \( \Delta \neq 0 \). In this case we have
\[
v = M^{-1}b \quad (20)
\]

A plot of the constraint lines is shown in figure 4. The solution is the intersection of the two constraint lines.

With two points only, the results are not always reliable. It is also possible to take more points and use their constraints to solve for the optical flow. We again assume that the partial derivatives can be estimated and that all points move at the same speeds. The problem can be formulated as an optimization problem. Consider the cost function:
\[
\int \int (v_x I_x + v_y I_y + I_t)^2 \, dx \, dy \quad (21)
\]
By taking the derivative with respect to \( v_x \) and \( v_y \), we get:
\[
\int \int (v_x I_x + v_y I_y + I_t)I_x \, dx \, dy = 0 \quad (22)
\]
\[
\int \int (v_x I_x + v_y I_y + I_t)I_y \, dx \, dy = 0 \quad (23)
\]
In the discrete case, for image block of size \( m \times n \), the cost function is
\[
\sum_{i=1}^{n} \sum_{j=1}^{m} [v_x I_x + v_y I_y + I_t]^2
\]
(24)

A least squares method can be used to solve for the optimal flow based on the cost function given by (24).

**Lucas and Kanade Method**

This method is less sensitive to noise than the point wise method. Better robustness is achieved by combining information from different pixels by taking a window or a local image. Pixel \( p^+ \) is the center of a \( n \times n \) window as shown in figure 5-(a). For simplicity, consider a 3 by 3 window (we have a total of nine elements). The constraints equations are
\[
\begin{align*}
I_x(p_1)v_x + I_y(p_1)v_y &= -I_t(p_1) \\
I_x(p_2)v_x + I_y(p_2)v_y &= -I_t(p_2) \\
&
\vdots \\
I_x(p_9)v_x + I_y(p_9)v_y &= -I_t(p_9)
\end{align*}
\]
(25)

In this case \( I_x, I_y, I_t \) are calculated at each pixel \( p_i = (i_j, j_i) \) in the window. For an image of \( n \times n = N \) pixels, we can write under matrix form:
\[
M = \begin{bmatrix}
I_x(i_1, j_1) & I_y(i_1, j_1) \\
I_x(i_2, j_2) & I_y(i_2, j_2) \\
\vdots & \vdots \\
I_x(i_N, j_N) & I_y(i_N, j_N)
\end{bmatrix}
\]
\[
b = \begin{bmatrix}
I_t(i_1, j_1) \\
I_t(i_2, j_2) \\
\vdots \\
I_t(i_N, j_N)
\end{bmatrix}
\]

\[
v = \begin{bmatrix} v_x \\ v_y \end{bmatrix}
\]
(28)

Equation (25) has the general form
\[
Mv = b
\]
(29)

The system has more equations than variables. It is over determined. Lucas and Kanade suggested to use a least squares method to solve. The goal is to minimize the cost function:
\[
\sum_{x,y \in \Omega} W(x,y) [\nabla I(x,y,t)v + I_t(x,y,t)]^2
\]
(30)

where
- \( \Omega \) is a small spatial neighborhood
- \( W \) is a window function that gives more importance to the center of the neighborhood than those at the periphery (typically 2D Gaussian window)
- \( \nabla \) is the symbol for the gradient.

The solution is
\[
M^T M v = M^T b
\]
(31)

\[
v = (M^T M)^{-1} M^T b
\]
(32)

We can also write
\[
\begin{bmatrix} v_x \\ v_y \end{bmatrix} = [C]^{-1} [D]
\]
(33)

where
\[
C = \begin{bmatrix}
\sum_i w_i I_x^2(p_i) & \sum_i w_i I_x(p_i) I_y(p_i) \\
\sum_i w_i I_x(p_i) I_y(p_i) & \sum_i w_i I_y^2(p_i)
\end{bmatrix}
\]
(34)

\[
D = \begin{bmatrix}
-\sum_i w_i I_x(p_i) I_t(p_i) \\
-\sum_i w_i I_y(p_i) I_t(p_i)
\end{bmatrix}
\]
(35)

**Horn and Schunk Algorithm**

This is an iterative method based on the assumption of the smoothness of the flow in the entire image. The method assumes that brightness varies smoothly (except when occlusion happens). One way to express the smoothness constraint is by using the magnitude of the gradient of the velocity
\[
\left( \frac{\partial v_x}{\partial x} \right)^2 + \left( \frac{\partial v_x}{\partial y} \right)^2 + \left( \frac{\partial v_y}{\partial x} \right)^2 + \left( \frac{\partial v_y}{\partial y} \right)^2
\]
(36)

(37)

Horn and Schunk suggested to minimize the following cost function
\[
\xi^2 = \int \int (\lambda^2 J_1^2 + J_2^2) dx dy
\]
(38)

where
\[
J_1 = I_x v_x + I_y v_y + I_t
\]
(39)

and
\[
J_2^2 = \left( \frac{\partial v_x}{\partial x} \right)^2 + \left( \frac{\partial v_x}{\partial y} \right)^2 + \left( \frac{\partial v_y}{\partial x} \right)^2 + \left( \frac{\partial v_y}{\partial y} \right)^2
\]
(40)

where \( J_2 \) is a measure of the departure from smoothness.
After using optimization techniques to solve, the method can be summarized as follows:

1) Initial velocity vectors \( v(i, j) = 0 \) for all \((i, j)\)

2) Integer \( k \) denotes the number of iterations. Compute \( v_x \) and \( v_y \) for all pixels at each iteration:

\[
v_x^k(i, j) = \bar{v}_x^{k-1} - I_x(i, j) \frac{P(i, j)}{D(i, j)} \tag{41}
\]

\[
v_y^k(i, j) = \bar{v}_y^{k-1} - I_y(i, j) \frac{P(i, j)}{D(i, j)} \tag{42}
\]

with

\[
P = I_x \bar{v}_x + I_y \bar{v}_y + I_t \tag{43}
\]

\[
D = \lambda^2 + I_x^2 + I_y^2 \tag{44}
\]

The partial derivatives \( I_x, I_y, I_t \) are estimated from pairs of consecutive images. \( \lambda \) is the Lagrange multiplier.

3) Stop when the error between two consecutive iterations is small.

**EXAMPLE**

For the two frames of figure 5-(c), find the optical flow

1) graphically,

2) using Horn–Schunk method

3) using Lucas–Kanade method

- At the first pixel

\[
I_x = 0.25(19 - 17) = 0.5 \tag{45}
\]

\[
I_y = 0.25(22 - 14) = 2 \tag{46}
\]

\[
I_t = 0.25(26 - 10) = 4 \tag{47}
\]

and the constraint equation is

\[
0.5v_x + 2v_y = -4 \tag{48}
\]

- At the second pixel

\[
I_x = 0.25(21 - 19) = 0.5 \tag{49}
\]

\[
I_y = 0.25(18 - 22) = -1 \tag{50}
\]

\[
I_t = 0.25(23 - 17) = 1.5 \tag{51}
\]

and the constraint equation is

\[
0.5v_x - v_y = -1.5 \tag{52}
\]

The constraints lines are shown in figure 6. The intersection point is \((-4.6667, -0.8333)\). The result is confirmed using the Lucas–Kanade method and the Horn-Schunk method.

![Fig. 6. The constraint lines](image)

| \( i \) | \( [125; 346] \) |
| \( i \) | \( [365; 871] \) |
| \( i \) | \( 0.5 \) |
| \( i \) | \( 1 = 2 \) |
| \( i \) | \( 4 \) |
| \( i \) | \( 2 = 0.5 \) |
| \( i \) | \( 1 = 4 \) |
| \( i \) | \( 0 \) |
| \( i \) | \( 2 = 1 \) |
| \( i \) | \( 0 < a < B \) |
| \( i \) | \( 1 = 1.99 \) |
| \( i \) | \( 1 = 1 \) |
| \( i \) | \( 1 = 1 \) |
| \( i \) | \( 1 = 2 \) |
| \( i \) | \( 1 = 1 \) |
| \( i \) | \( 1 = 2 \) |
| \( i \) | \( 1 = 1 \) |
| \( i \) | \( 1 = 2 \) |
| \( i \) | \( 1 = 1 \) |
| \( i \) | \( 1 = 2 \) |
| \( i \) | \( 1 = 1 \) |
| \( i \) | \( 1 = 2 \) |

**Solution:**

\[
v = -4.6667 \]

\[
v = -0.8333 \]

![Fig. 7. Code for Lucas–Kanade and Horn-Schunk methods](image)