CS-E4850 Computer Vision,
Answers to Exercise Round 8

Yangzhe Kong, Student number: 765756

November 7, 2019

Exercise 1. Face tracking example using KLT tracker.

c) We can observe that when the image is being rotated, the number of tracked features also begin to drop. Also an interesting point is that when the image is being rotated slowly at first, the loss in the number of tracked features is small but then when the rotation speeds up a little bit, the number of tracked features drops intensively, by half. And finally we only have few tracked features left.

d) We can try to restrict rotation or any other movement that can cause huge drop of number of tracked features. For instance, trying to reduce large movements over a short peroid of time is a good idea.

Exercise 2. Kanade-Lucas-Tomasi (KLT) feature tracking

We first show what the solution in the lecture slides says.
From the Brightness Constancy Equation:

$$\begin{aligned} I(x, y, t-1) &= I(x + u(x, y), y + v(x, y), t) \\ &\approx I(x, y, t) + I_x u(x, y) + I_y v(x, y) + I_t \end{aligned}$$

Since we assume the brightness is constant and the movement is small, we can get

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

$$\nabla I \cdot (u, v) + I_t = 0$$

We can concatenate the equations for all the pixels in the window as the form of matrix multiplication:

$$\begin{bmatrix} I_x(x_1) & I_y(x_1) \\ I_x(x_2) & I_y(x_2) \\ \vdots & \vdots \\ I_x(x_n) & I_y(x_n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(x_1) \\ I_t(x_2) \\ \vdots \\ I_t(x_n) \end{bmatrix}$$

$$A_{n\times 2}d_{x\times 1} = b_{n\times 1}$$

And then we can find the solution by Lucas-Kanade Flow

$$(A^\top A)d = A^\top b$$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

Now we can show this solution is the same as the one we will derive using the formulas from **BakerMathews.pdf**.
We let $W(x;p)$ denote the parameterized set of allowed warps, where $p = (p_1, p_2, \cdots, p_n)^\top$ is a vector of parameters.
The warp $W(x;p)$ takes the pixel x in the coordinate frame of the template T and maps it to the sub-pixel location $W(x;p)$ in the coordinate frame of the image I.
In the case of translation, we have:

$$W(x;p) = \begin{bmatrix} x+u \\ y+v \end{bmatrix}, \Delta p = \begin{bmatrix} u \\ v \end{bmatrix}$$

And the Jacobian of the warp is

$$\frac{\delta W}{\delta p} = \begin{bmatrix} \delta W_x/\delta u & \delta W_x/\delta v \\ \delta W_y/\delta u & \delta W_y/\delta v \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

And we substitute it in the equation 10:

$$\Delta p = H^{-1} \sum_x [\nabla I \frac{\delta W}{\delta p}]^\top |T(x) - I(W(x;p))|$$

Where

$$\begin{aligned} H &= \sum [\nabla I \frac{\delta W}{\delta p}]^\top [\nabla I \frac{\delta W}{\delta p}] \\ &= \sum \frac{\delta W}{\delta p}^\top \nabla I^\top \nabla I \frac{\delta W}{\delta p} \\ &= \sum \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\delta I}{\delta x} \\ \frac{\delta I}{\delta y} \end{bmatrix} \begin{bmatrix} \frac{\delta I}{\delta x} & \frac{\delta I}{\delta y} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\ &= \sum \begin{bmatrix} \frac{\delta I}{\delta x}\frac{\delta I}{\delta x} & \frac{\delta I}{\delta x}\frac{\delta I}{\delta y} \\ \frac{\delta I}{\delta y}\frac{\delta I}{\delta x} & \frac{\delta I}{\delta y}\frac{\delta I}{\delta y} \end{bmatrix} \\ &= \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{bmatrix} \end{aligned}$$

So equation 10 can be expressed as

$$\begin{aligned}
H\Delta p &= \sum [\nabla I \frac{\delta W}{\delta p}]^{\top} |T(x) - I(W(x;p))| \\
\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} &= \sum [\nabla I \frac{\delta W}{\delta p}]^{\top} |T(x) - I(W(x;p))| \\
\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} &= \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} | - I_t| \\
\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} &= -\sum \begin{bmatrix} I_x I_t \\ I_y I_t \end{bmatrix}
\end{aligned}$$