Computer Vision (600.461/600.661)
Homework 4: Feature Matching and Optical Flow

Instructor: René Vidal

Due 10/02/2014, 11.59PM Eastern

1. **(15 Points) Corner localization via quadratic fit.** The second step of SIFT is to fit a quadratic function to the response of the Difference of Gaussian (DoG) filter applied to the image around each local maximum. Specifically, if $r(\boldsymbol{x})$ is the response at pixel $\boldsymbol{x} = (x, y)$, we seek a quadratic function $\frac{1}{2}\boldsymbol{x}^\top Q\boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x} + c$ that approximates $r(\boldsymbol{x})$ in a neighborhood of $\boldsymbol{x}$. We can do this by minimizing the sum of the squares of the fitting errors

$$\min_{Q,\boldsymbol{b},c} \sum_{\boldsymbol{u}} w(\boldsymbol{x}+\boldsymbol{u})(\frac{1}{2}(\boldsymbol{x}+\boldsymbol{u})^\top Q(\boldsymbol{x}+\boldsymbol{u}) + \boldsymbol{b}^\top(\boldsymbol{x}+\boldsymbol{u}) + c - r(\boldsymbol{x}+\boldsymbol{u}))^2, \tag{1}$$

where $\boldsymbol{u} = (u, v)$ is the displacement vector in a window around $\boldsymbol{x}$ and w is a weighting function inside the window (e.g., a Gaussian). Propose a least-squares like algorithm based on the SVD for computing the parameters Q, $\boldsymbol{b}$ and c. Recall that Q is a 2×2 symmetric negative definite matrix (to get a maximum).

ANSWER: Let $Q = \begin{bmatrix} q_1 & q_2 \\ q_2 & q_3 \end{bmatrix}$ and $\boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$. To simplify notation, let $\boldsymbol{x} + \boldsymbol{u}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$ for each point in the neighborhood of $\boldsymbol{x}$, $w(\boldsymbol{x}+\boldsymbol{u}_i) = w_i$ and $r(\boldsymbol{x}+\boldsymbol{u}_i) = r_i$, $i = 1, \ldots, N$. The problem in (1) can be written as:

$$\min_{Q,\boldsymbol{b},c} \sum_i w_i \left(\frac{1}{2} \begin{bmatrix} x_i & y_i \end{bmatrix} \begin{bmatrix} q_1 & q_2 \\ q_2 & q_3 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} b_1 & b_2 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} - c + r_i \right)^2$$

$$\min_{Q,\boldsymbol{b},c} \sum_i w_i \left(\frac{1}{2}(q_1 x_i^2 + 2q_2 x_i y_i + q_3 y_i^2) + b_1 x_i + b_2 y_i + c - r_i \right)^2$$

METHOD #1: The above optimization problem can be compactly written as $\min_{\boldsymbol{g}} \|A\boldsymbol{g} - \boldsymbol{h}\|_2^2$, where:

$$A = \begin{bmatrix} \sqrt{w_1}\cdot \begin{bmatrix} \frac{1}{2}x_1^2 & x_1 y_1 & \frac{1}{2}y_1^2 & x_1 & y_1 & 1 \end{bmatrix} \\ \vdots \\ \sqrt{w_N}\cdot \begin{bmatrix} \frac{1}{2}x_N^2 & x_N y_N & \frac{1}{2}y_N^2 & x_N & y_N & 1 \end{bmatrix} \end{bmatrix}, \quad \boldsymbol{g} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ b_1 \\ b_2 \\ c \end{bmatrix}, \quad \boldsymbol{h} = \begin{bmatrix} \sqrt{w_1} r_1 \\ \vdots \\ \sqrt{w_N} r_N \end{bmatrix}$$

Setting the derivative to zero we get $A^\top(A\boldsymbol{g} - \boldsymbol{h}) = \boldsymbol{0} \implies \boldsymbol{g} = (A^\top A)^{-1} A^\top \boldsymbol{h}$, where we assume that A is full rank 6. Alternatively, we may use the pseudo inverse of A to obtain the solution as $\boldsymbol{g} = A^\dagger \boldsymbol{h}$. Then, letting $A_{(N\times 6)} = U_{(N\times 6)} \Sigma_{(6\times 6)} V_{(6\times 6)}^\top$, the solution to the minimization problem is in the form of $\boldsymbol{g} = V\Sigma^{-1}U^\top \boldsymbol{h}$.

METHOD #2: Taking derivatives with respect to each one of the six variables and setting them to zero we get:

$$\frac{\partial J}{\partial q_1} = \sum_i w_i (\frac{1}{2}[x_i^2, 2x_i y_i, y_i^2] \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^\top + [x_i, y_i] \begin{bmatrix} b_1 & b_2 \end{bmatrix} + c - r_i) x_i^2 = 0 \tag{2}$$

$$\frac{\partial J}{\partial q_2} = \sum_i w_i (\frac{1}{2}[x_i^2, 2x_i y_i, y_i^2] \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^\top + [x_i, y_i] \begin{bmatrix} b_1 & b_2 \end{bmatrix} + c - r_i) 2x_i y_i = 0 \tag{3}$$

$$\frac{\partial J}{\partial q_3} = \sum_i w_i (\frac{1}{2}[x_i^2, 2x_i y_i, y_i^2] \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^\top + [x_i, y_i] \begin{bmatrix} b_1 & b_2 \end{bmatrix} + c - r_i) y_i^2 = 0 \tag{4}$$

$$\frac{\partial J}{\partial b_1} = \sum_i w_i (\frac{1}{2}[x_i^2, 2x_i y_i, y_i^2] \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^\top + [x_i, y_i] \begin{bmatrix} b_1 & b_2 \end{bmatrix} + c - r_i) 2x_i = 0 \tag{5}$$

$$\frac{\partial J}{\partial b_2} = \sum_i w_i (\frac{1}{2}[x_i^2, 2x_i y_i, y_i^2] \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^\top + [x_i, y_i] \begin{bmatrix} b_1 & b_2 \end{bmatrix} + c - r_i) 2y_i = 0 \tag{6}$$

$$\frac{\partial J}{\partial c} = \sum_i w_i(\frac{1}{2}[x_i^2, 2x_iy_i, y_i^2]\begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^\top + [x_i, y_i]\begin{bmatrix} b_1 & b_2 \end{bmatrix} + c - r_i)2 = 0 \tag{7}$$

This leads to 6 linear equations. One can write them in matrix form $\tilde{A}\boldsymbol{g} = \tilde{\boldsymbol{h}}$,

$$\tilde{A} = \sum_i w_i \begin{bmatrix} \frac{1}{2}x_i^4 & x_i^3y_i & \frac{1}{2}x_i^2y_i^2 & x_i^3 & x_i^2y_i & x_i^2 \\ x_i^3y_i & 2x_i^2y_i^2 & x_iy_i^3 & 2x_i^2y_i & 2x_iy_i^2 & 2x_iy_i \\ \frac{1}{2}x_i^2y_i^2 & x_iy_i^3 & \frac{1}{2}y_i^4 & x_iy_i^2 & y_i^3 & y_i^2 \\ x_i^3 & 2x_i^2y_i & x_iy_i^2 & 2x_i^2 & 2x_iy_i & 2x_i \\ x_i^2y_i & 2x_iy_i^2 & y_i^3 & 2x_iy_i & 2y_i^2 & 2y_i \\ x_i^2 & 2x_iy_i & y_i^2 & 2x_i & 2y_i & 2 \end{bmatrix}, \tilde{\boldsymbol{h}} = \sum_i w_ir_i \begin{bmatrix} x_i^2 \\ 2x_iy_i \\ y_i^2 \\ 2x_i \\ 2y_i \\ 2 \end{bmatrix}, \boldsymbol{g} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ b_1 \\ b_2 \\ c \end{bmatrix} \tag{8}$$

Under the assumption that $A \in \mathbb{R}^{6\times 6}$ is full rank, one can compute $\boldsymbol{g}$ as $\boldsymbol{g} = \tilde{A}^{-1}\tilde{\boldsymbol{h}}$.

ENFORCING NEGATIVE DEFINITENESS: Notice that our derivation did not constrain Q to be negative definite. To enforce that, we need to add the following constraints to the optimization:

$$q_1 \leq 0, \quad \text{and} \quad q_1q_3 - q_2^2 \geq 0.$$

Notice that the last constraint is non-linear. Solving the above least squares problem subject to nonlinear constraints is out of the scope of the class.

2. **(20 Points) Feature point matching under a 2D rigid body motion.** Let I_1 and I_2 be two images related by an unknown 2D rotation $R \in SO(2)$ and an unknown 2D translation $\boldsymbol{t} \in \mathbb{R}^2$, i.e., $I_2(\boldsymbol{x}) = I_1(R\boldsymbol{x} + \boldsymbol{t})$. Let $\{\boldsymbol{x}_j\}_{j=1}^N$ be a set of image points (e.g., corners) extracted from I_1. Suppose you have run a feature matching algorithm and extracted a set of corresponding image points $\{\boldsymbol{y}_j\}_{j=1}^N$ in I_2, i.e., $\boldsymbol{y}_j \approx R\boldsymbol{x}_j + \boldsymbol{t}$. Propose an algorithm for computing the unknown transformation $(R, \boldsymbol{t}) \in SE(2)$ that minimizes the sum of squared errors:

$$\min_{R,\boldsymbol{t}} \sum_{j=1}^N \|\boldsymbol{y}_j - R\boldsymbol{x}_j - \boldsymbol{t}\|_2^2. \tag{9}$$

Specifically, show that the optimal translation is given by $\boldsymbol{t}^* = \bar{\boldsymbol{y}} - R^*\bar{\boldsymbol{x}}$, where $\bar{\boldsymbol{x}} = \sum \boldsymbol{x}_i/N$ and $\bar{\boldsymbol{y}} = \sum \boldsymbol{y}_i/N$, and that the optimal rotation is given by $R^* = \text{argmin}_{R\in SO(2)} \|Y - RX\|_F^2$, where $X = \begin{bmatrix}\boldsymbol{x}_1 - \bar{\boldsymbol{x}} \cdots \boldsymbol{x}_N - \bar{\boldsymbol{x}}\end{bmatrix}$ and $Y = \begin{bmatrix}\boldsymbol{y}_1 - \bar{\boldsymbol{y}} \cdots \boldsymbol{y}_N - \bar{\boldsymbol{y}}\end{bmatrix}$. Show that $R^* = \text{argmax}_R\langle Y, RX\rangle = \text{argmax}_R \operatorname{trace}(Y^\top RX)$. Parametrize R in terms of the rotation angle θ and show that

$$\theta^* = \underset{\theta}{\text{argmax}} \operatorname{trace}(X^\top Y)\cos(\theta) + \operatorname{trace}(X^\top \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} Y)\sin(\theta), \tag{10}$$

Find the optimal θ and show that the optimal R is given by

$$R^* = \frac{\begin{bmatrix} \operatorname{trace}(X^\top Y) & -\operatorname{trace}(X^\top \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} Y) \\ \operatorname{trace}(X^\top \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} Y) & \operatorname{trace}(X^\top Y) \end{bmatrix}}{\sqrt{\operatorname{trace}(X^\top Y)^2 + \operatorname{trace}(X^\top \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} Y)^2}}. \tag{11}$$

ANSWER: To minimize the cost E we set the first derivative to zero as:

$$\frac{\partial}{\partial \boldsymbol{t}}(\sum_{j=1}^N \|\boldsymbol{y}_j - R\boldsymbol{x}_j - \boldsymbol{t}\|_2^2) = -2\sum_{j=1}^N(\boldsymbol{y}_j - R\boldsymbol{x}_j - \boldsymbol{t}) = 0 \implies \boldsymbol{t}^* = \frac{1}{N}\sum_{j=1}^N(\boldsymbol{y}_j - R\boldsymbol{x}_j) = \bar{\boldsymbol{y}} - R^*\bar{\boldsymbol{x}}$$

Substituting this into the cost we have:

$$\min_R \sum_{j=1}^N \|\boldsymbol{y}_j - \bar{\boldsymbol{y}} - R(\boldsymbol{x}_i - \bar{\boldsymbol{x}})\|_2^2 = \|Y - RX\|_F^2 = \|Y\|_F^2 - 2\langle Y, RX\rangle + \|RX\|^2$$

Since $\|RX\|_F = \|X\|_F$ the first and the last terms are independent on R, and therefore the minimization problem is equivalent to maximizing the negative of the second term. The dot product $\langle Y, RX\rangle$ is equivalent to the $\mathrm{trace}(Y^\top RX)$, thus: $R^* = \mathrm{argmax}_R \langle Y, RX\rangle = \mathrm{argmax}_R \mathrm{trace}(Y^\top RX)$.

Next, we write the rotation in terms of the angle of rotation as:

$$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

Since

$$\begin{aligned}
\mathrm{trace}(Y^\top RX) &= \mathrm{trace}(X^\top R^\top Y) = \mathrm{trace}(X^\top \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} Y) \\
&= \mathrm{trace}(X^\top \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 \\ 0 & \cos(\theta) \end{bmatrix} Y + X^\top \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \sin(\theta) & 0 \\ 0 & \sin(\theta) \end{bmatrix} Y) \\
&= \mathrm{trace}(X^\top Y)\cos(\theta) + \mathrm{trace}(X^\top \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} Y)\sin(\theta) \\
&= A\cos(\theta) + B\sin(\theta)
\end{aligned}$$

where $A = \mathrm{trace}(X^\top Y)$ and $B = \mathrm{trace}(X^\top \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} Y)$, the first order condition for optimality is

$$\frac{\partial}{\partial \theta}(A\cos(\theta) + B\sin(\theta)) = -A\sin(\theta) + B\cos(\theta) = 0 \implies \tan(\theta^*) = \frac{B}{A}$$

Hence, we can compute $\cos(\theta^*)$ and $\sin(\theta^*)$ as:

$$\cos(\theta^*) = \frac{A}{\sqrt{A^2 + B^2}} \qquad \text{and} \qquad \sin(\theta^*) = \frac{B}{\sqrt{A^2 + B^2}}$$

Thus, the optimal R is as:

$$R^* = \frac{\begin{bmatrix} \mathrm{trace}(X^\top Y) & -\mathrm{trace}(X^\top \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} Y) \\ \mathrm{trace}(X^\top \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} Y) & \mathrm{trace}(X^\top Y) \end{bmatrix}}{\sqrt{\mathrm{trace}(X^\top Y)^2 + \mathrm{trace}(X^\top \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} Y)^2}}$$

3. **(15 Points) Optical flow with changes in illumination.** Let $I(x, y, t)$ be a video sequence taken by a moving camera observing a rigid, static and Lambertian scene. Assume that between two consecutive views there is an affine change in the image intensities, i.e., the brightness constancy constraint reads

$$I(x + u, y + v, t + 1) = aI(x, y, t) + b, \tag{12}$$

where $u(x, y)$ and $v(x, y)$ are the optical flow and $a(x, y)$ and $b(x, y)$ represent photometric parameters. Propose a linear algorithm for estimating (u, v, a, b) from the image brightness I and its spatio-temporal derivatives I_x, I_y, I_t. What is the minimum size of a window around each pixel that allows one to solve the problem?

ANSWER: After subtracting $I(x, y, t)$ on both sides, and applying the BCC, we obtain

$$I_x u + I_y v + I_t = (a - 1)I + b,$$

which reduces to the standard BCC when $a = 1$ and $b = 0$. This new BCC can be re-written as

$$I_x u + I_y v + (1 - a)I - b = -I_t \implies \begin{bmatrix} I_x & I_y & I & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 - a \\ -b \end{bmatrix} = -I_t.$$

From this equation, we can solve for the parameters $\theta = (u, v, a, b)$ in a least squares sense by assuming that θ is constant on a neighborhood Ω around each pixel. This leads to the following linear system of equations

$$\sum_{\Omega} \begin{bmatrix} I_x^2 & I_x I_y & I_x I & I_x \\ I_x I_y & I_y^2 & I_y I & I_y \\ I_x I & I_y I & I^2 & I \\ I_x & I_y & I & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 - a \\ -b \end{bmatrix} = -\sum_{\Omega} \begin{bmatrix} I_t I_x \\ I_t I_y \\ I_t I \\ I_t \end{bmatrix}.$$

Since there are four unknowns, we need at least 4 pixels, e.g. a 2×2 window. Since odd sized windows are preferred to compute optical flow at the center of the window, we can use a 3×3 window.