4F12: Computer vision and robotics crib 2014

Dr. Richard E. Turner

May 15, 2014

1 (a) The expression for filtering is:

$$S(x,y) = \sum_{u,v} G_\sigma(u,v) I(x-u, y-v)$$

where $G_\sigma(u,v)$ is a low pass filter. (The continuous version also acceptable)

Smoothing removes high-pass noise from the image. This is especially important for algorithms which compute derivatives of the image (such as Canny and Marr edge detection, Harris corner detection and blob detection). The derivative operator corresponds to a high-pass filter which is severely affected by noise. [10%]

(b) First, for separable filters, such as the widely used Gaussian filter,

$$G_\sigma(u,v) = \frac{1}{2\pi\sigma^2} \exp(-\frac{1}{2\sigma^2}(x^2+y^2)),$$

the 2D convolution can be written as two 1D filtering operations,

$$S(x,y) = \sum_{u,v} G_\sigma(u,v) I(x-u, y-v) = \sum_u g_\sigma(x-u) \sum_v g_\sigma(y-v) I(u,v).$$

Here $g_\sigma(x)$ is a 1D Gaussian $g_\sigma(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{1}{2\sigma^2}x^2)$. Second, the support of the filter can be truncated by removing the tail regions in which the function falls below a certain threshold, such as $\frac{1}{1000}$ of the peak.

A naïve implementation of the filtering operation has computational cost $\mathcal{O}(N^2K_1^2)$ where N^2 = number of pixels, K_1^2 = number of pixels in non-truncated 2D Gaussian filter. The optimised version is $\mathcal{O}(N^2K_2)$ where K_2=extent of 1D Gaussian filter and $K_2 < K_1$. [25%]

(c) The Canny edge detection algorithm carries out the following computations on the smoothed image in order to locate the position and orientation of edges:

1. gradients: find gradient of smoothed image pixels $\nabla S(x,y)$
2. non-maximal suppression: place edgels where $|\nabla S(x,y)|$ greater than local values of $|\nabla S(x,y)|$ in directions $\pm \nabla S(x,y)$
3. threshold: only retain $|\nabla S(x,y)| \geq$ thresh
4. return: output edge positions (x_i, y_i) and orientations $\frac{\nabla S(x_i,y_i)}{|\nabla S(x_i,y_i)|}$

[25%]

(d) The Harris corner detection algorithm carries out the following computations on the smoothed image in order to locate the position of corners:

1. compute gradients of smoothed image: $\nabla S(x, y) = (S_x, S_y)$
2. form outer product of gradients and smooth using another broader Gaussian low pass filter (**note that there are two smoothing operations here, a fact that was often neglected by candidates in the exam**),

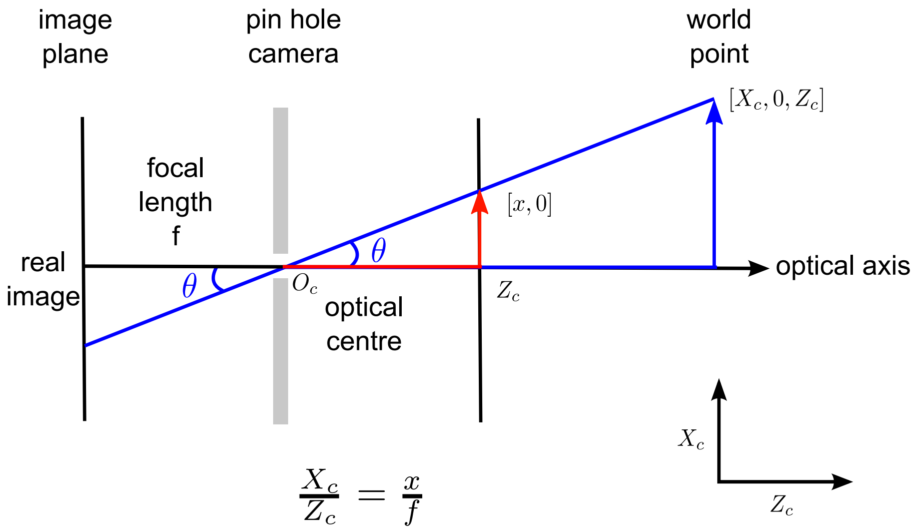
$$A = \begin{bmatrix} \langle S_x^2 \rangle & \langle S_x S_y \rangle \\ \langle S_x S_y \rangle & \langle S_y^2 \rangle \end{bmatrix}.$$

   Here $\langle f(x, y) \rangle = \sum_{u,v} G_{\sigma'}(u, v) f(x - u, y - v)$ and $\sigma' > \sigma$
3. locate corners and threshold: find locations where $\det(A) - \kappa \, \mathrm{trace}(A)^2 \geq \mathrm{thresh}$
4. return corner positions (x_i, y_i)

[30%]

(e) Edges do not allow motion to be resolved in the direction of the edge (the so called aperture problem). For this reason, corners are superior to edges as interest points for tracking. [10%]

2 (a) In the pin-hole camera a narrow aperture allows an image to be formed on an image plane:



$$\frac{X_c}{Z_c} = \frac{x}{f}$$

switch image coordinates to avoid minus sign

An identical expression can be derived for the component orthogonal to X_c and x: $y/f = X_c/Z_c$.

These can be seen as the same expressions as given in the question by multiplying out the matrix expressions: $sx = \lambda f X_c$, $sy = \lambda f Y_c$ and $s = \lambda Z_c$. Eliminating s and λ gives the expression above.

The perspective projection equations are non-linear due to the division by Z_c. By recasting the equations into homogeneous coordinates, perspective projection becomes a linear operation. This is especially useful when combining perspective projection with the rigid body transformation and CCD imaging transformations, which are also linear operations in these coordinates. This simplifies mathematical and algorithm work considerably.

[10%]

(b) Consider two parallel planes in world coordinates

$$n_x X_c + n_y Y_c + n_z Z_c = d_1$$
$$n_x X_c + n_y Y_c + n_z Z_c = d_2$$

Transform to homogeneous world coordinates using $X_c = \frac{X_1}{X_4}$, $Y_c = \frac{X_2}{X_4}$ and $Z_c = \frac{Z_1}{X_4}$:

$$n_x X_1 + n_y X_2 + n_z X_3 = d_1 X_4$$
$$n_x X_1 + n_y X_2 + n_z X_3 = d_2 X_4$$

By considering the point at infinity encoded by $X_4 = 0$, this implies that in homogeneous coordinates the two hyperplanes intersect along a common plane $n_x X_1 + n_y X_2 + n_z X_3 =$

0. We now transform this plane into homogeneous image coordinates,

$$\tilde{\mathbf{x}} = \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ \tilde{x}_3 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}$$

Therefore, $n_x\tilde{x}_1/f + n_y\tilde{x}_2/f + n_z\tilde{x}_3 = 0$. We now transform this back into Cartesian image coordinates using $x = \tilde{x}_1/\tilde{x}_3$ and $y = \tilde{x}_2/\tilde{x}_3$,

$$n_x x + n_y y + n_z f = 0$$

This is the equation for the horizon line.

[40%]

(c) (i) We start with the equation of the ellipse,

$$\left(\frac{X_c - X_0}{a}\right)^2 + \left(\frac{Z_c - Z_0}{b}\right)^2 = 1 \text{ where } Y_c = Y_0,$$

and substitute in the perspective projection equations, $x = \frac{fX_c}{Z_c}$, $y = \frac{fY_c}{Z_c} = \frac{fY_0}{Z_c}$, giving:

$$1 = \frac{1}{a^2}\left(Y_0\frac{x}{y} - X_0\right)^2 + \frac{1}{b^2}\left(Y_0\frac{f}{y} - Z_0\right)^2$$
$$y^2 = \frac{1}{a^2}(Y_0 x - X_0)^2 + \frac{1}{b^2}(Y_0 f - Z_0)^2$$

This is another ellipse of the following form,

$$0 = \frac{Y_0^2}{a^2}x^2 + \left(\frac{X_0^2}{a^2} + \frac{Z_0^2}{b^2} - 1\right)y^2 - \frac{2}{a^2}Y_0X_0xy - \frac{2}{b^2}fY_0Z_0y + \frac{f^2Y_0^2}{b^2}.$$

For the image ellipse to be a circle two conditions must be met. First, $\frac{2}{a^2}Y_0X_0xy = 0$ (**the cross term must vanish**) which implies $X_0 = 0$ (the ellipse is centred on the x-axis). This ensures the image is an axis aligned ellipse. Note that $Y_0 = 0$ implies a line rather than a circle. Second, $\frac{Y_0^2}{a^2} = \frac{Z_0^2}{b^2} - 1$ which ensures that the major and minor axes of the image ellipse are identical. [40%]

(ii) The weak perspective camera assumes that the variation in depth of the objects in the image is small compared to the distance of the camera from the scene. In this case $\frac{\Delta Z_c}{Z_c} \propto \frac{b}{Z_0} \approx 0$ and the ellipse projects to a line in the image. [10%]

3 (a) Perspective projection can be written in terms of homogeneous world coordinates, $\tilde{\mathbf{X}}$, and homogeneous pixel coordinates, $\tilde{\mathbf{w}} = (su, sv, s)$ as $\tilde{\mathbf{w}} = \mathbf{K}[\mathbf{R}|\mathbf{T}]\tilde{\mathbf{X}}$.

There are two scenarios to consider:

First we apply this expression to two views of the same scene. W.l.g. we set the translation to zero, $\mathbf{T} = \mathbf{0}$, and the rotation of the first camera to identity, $\mathbf{R} = \mathbf{I}$. Allowing for the rigid body rotation and intrinsic parameters of the second camera to change (but not the position) we have,

$$\tilde{\mathbf{w}} = \mathbf{K}[\mathbf{I}|\mathbf{0}]\tilde{\mathbf{X}} = \mathbf{K}\tilde{\mathbf{X}}$$
$$\tilde{\mathbf{w}}' = \mathbf{K}'[\mathbf{R}|\mathbf{0}]\tilde{\mathbf{X}} = \mathbf{K}'\mathbf{R}\tilde{\mathbf{X}}$$

Therefore, $\tilde{\mathbf{X}} = \mathbf{K}^{-1}\tilde{\mathbf{w}}$ and substituting this into the second equation above yields,

$$\tilde{\mathbf{w}}' = \mathbf{K}'\mathbf{R}\mathbf{K}^{-1}\tilde{\mathbf{w}}$$

Second, consider viewing a plane. Wittout loss of generality we can assume that the plane is aligned with the z-axis and drop a column out of the projection matrix $\mathbf{P} = \mathbf{K}[\mathbf{R}, \mathbf{T}]$:

$$\tilde{\mathbf{w}} = \begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{31} & p_{32} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \tilde{\mathbf{P}} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

Therefore, $\tilde{\mathbf{w}}' = \tilde{\mathbf{P}}\text{'}(\tilde{\mathbf{P}})^{-1}\tilde{\mathbf{w}}$ [30%]

(b) (i) Each point provides two constraints. The homography has 8 degrees of freedom so four points are required. [10%]

(ii) SIFT can return imperfect matches due to 1) interest points not being detected in corresponding locations due to occlusion, noise in the camera, the scene and lighting conditions not being static, the changes in viewpoint being too substantial for SIFT to cope with 2) descriptors in the neighbourhood of two non-corresponding interest point are similar then mismatches can occur. For example, if image contains many similar objects/parts of objects. [20%]

(iii) Commented pseudo code for the RANSAC algorithm:

```
H = eye(3,3)                       ▷ homography, $H = K'RK^{-1}$, initialised to identity
nBest = 0
for int i = 0; i < nIterations; i++ do
    P4 = SelectRandomSubset(P)             ▷ select subset of points (e.g. 4)
    Hi = ComputeHomography(P4)             ▷ compute homography from subset
    nInliers = ComputeInliers(Hi)          ▷ compute no. of consistent points
    if nInliers > nBest then
        H = Hi
        nBest = nInliers
    end if
end for
```

[20%]

(iv) Assume that RANSAC returns a good solution when all of the members of the subset of matches that are randomly selected are inliers. If D points are selected on each iteration and there are many points (so that the fact that points are not selected with replacement can be ignored) the probability of this occurring on a single iteration of the algorithm is $(1-\rho)^D$. The probability of not selecting D inliers in T iterations is therefore $(1-(1-\rho)^D)^T$. Therefore, to achieve a desired probability of success $P_0 = 1-(1-(1-\rho)^D)^T$. Rearranging for T yields, $T = \frac{\log(1-P_0)}{\log(1-(1-\rho)^D)}$ **This question polarised candidates in the exam, even though the statistical reasoning required to solve the problem is quite simple.**

[20%]

4 (a) $\mathbf{X}_c' = R\mathbf{X}_c + T$ take the cross product with T giving $T \times \mathbf{X}_c' = T \times R\mathbf{X}_c$ and dot product with $\mathbf{X}_c'$ to yield $0 = \mathbf{X}_c'.(T \times R\mathbf{X}_c)$.

In order to derive the form of the essential matrix we write the cross product $T \times R\mathbf{X}_c$ in matrix form. First consider the cross product $T \times \mathbf{X}_c$,

$$T \times \mathbf{X}_c = \begin{bmatrix} T_z \\ T_y \\ T_z \end{bmatrix} \times \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} T_y z_c - T_z y_c \\ T_z x_c - T_x z_c \\ T_x y_c - T_y x_c \end{bmatrix}$$

Now rewrite the cross product as a matrix operation,

$$T \times \mathbf{X}_c = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = T_x \mathbf{X}_c$$

So, using this result, we have $\mathbf{X}_c'^T E \mathbf{X}_c = 0$ where the essential matrix is $E = T_x R$. [40%]

(b) Define $\mathbf{p}_e = (x_e, y_e, f)$. Since the epipole must lie on a line between the two optic centres $\lambda T = R\mathbf{p}_e + T$. Taking the cross product with T yields $E\mathbf{p}_e = \mathbf{0}$ [10%]

(c) (i) The form of the essential matrix in this case is:

$$E = \begin{bmatrix} 0 & 0 & d \\ 0 & 0 & d \\ -d & -d & 0 \end{bmatrix}$$

so the epipolar lines are given by $x' + y' = y + x$ [10%]

(ii) As the image planes are parallel (and only differ in displacement) the epipoles are located at infinity in the direction of the epipolar lines.

Candidates often confused epipoles and epipolar lines in the exam.

[10%]

(d) First relate expression for epipole in camera-centred coordinates, $\mathbf{p}_e = (x_e, y_e, f)$, to homogeneous pixel coordinates,

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} k_u & 0 & u_0 \\ 0 & k_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} k_u & 0 & u_0/f \\ 0 & k_v & v_0/f \\ 0 & 0 & 1/f \end{bmatrix} \begin{bmatrix} x \\ y \\ f \end{bmatrix}$$

$$\begin{bmatrix} fu \\ fv \\ f \end{bmatrix} = \begin{bmatrix} fk_u & 0 & u_0 \\ 0 & fk_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ f \end{bmatrix}$$

$$\tilde{\mathbf{w}} = K\mathbf{p}_e$$

We now have the following expressions:

$$\mathbf{p}_e'^T E \mathbf{p}_e = 0 \qquad \tilde{\mathbf{w}} = K\mathbf{p}_e \qquad \tilde{\mathbf{w}}' = K'\mathbf{p}_e'$$

eliminate $\mathbf{p}'_e$ and $\mathbf{p}_e$:

$$\tilde{\mathbf{w}}'^T K'^{-T} E K^{-1} \tilde{\mathbf{w}} = 0 = \tilde{\mathbf{w}}'^T F \tilde{\mathbf{w}}$$

where $F = K'^{-T} E K^{-1}$ is the 3 by 3 fundamental matrix
If F is known, the search for matches can be restricted to narrow bands around the epipolar lines. This turns a 2D search into a line search which significantly reduces the computational complexity of matching.

[30%]