4F12 Computer Vision (2016)

Q1(a) (i.) Low pass filter used $G_\sigma(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$ 1m

(1) marks

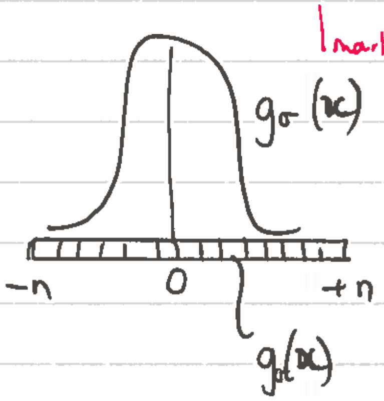
$$= g_\sigma(x)\, g_\sigma(y)$$

(2) $$S(x,y) = \sum_{-n}^{n}\sum_{-n}^{n} I(x-u, y-v)\, g_\sigma(v)\, g_\sigma(u)$$ 2 marks

where kernel size is $(2n+1)$

$2n+1$ samples from a 1D gaussian 1 mark

(1)



(ii). $\nabla S = \nabla\left(G_\sigma(x,y) * I(x,y)\right) = (S_x, S_y)$ where $S_x = \frac{\partial S}{\partial x}$

2D vector

$S_y = \frac{\partial S}{\partial y}$

(1) $$\frac{\partial S(x,y)}{\partial x} \simeq \frac{S(x+1,y) - S(x-1,y)}{2}$$ 1D kernel $\boxed{\tfrac{1}{2} \mid 0 \mid -\tfrac{1}{2}}$

(1) $$\frac{\partial S(x,y)}{\partial y} \simeq \frac{S(x,y+1) - S(x,y-1)}{2}$$

$\begin{array}{|c|}\hline \tfrac{1}{2} \\ \hline 0 \\ \hline -\tfrac{1}{2} \\ \hline\end{array}$

0 → x, ↓ y

1.(a)(iii.)

Generate discrete set of smoothed images $S(x, y, \sigma_i)$ logarithmically spaced with s images per octave

$$\sigma_i = 2^{i/s}\sigma_0 \quad \text{and} \quad \sigma_{i+1} = \sigma_i \, 2^{\frac{1}{s}}$$ (1)

- implement all gaussian blurs by 2 x 1D convolutions with 'small' filter kernels
- apply incremental blur

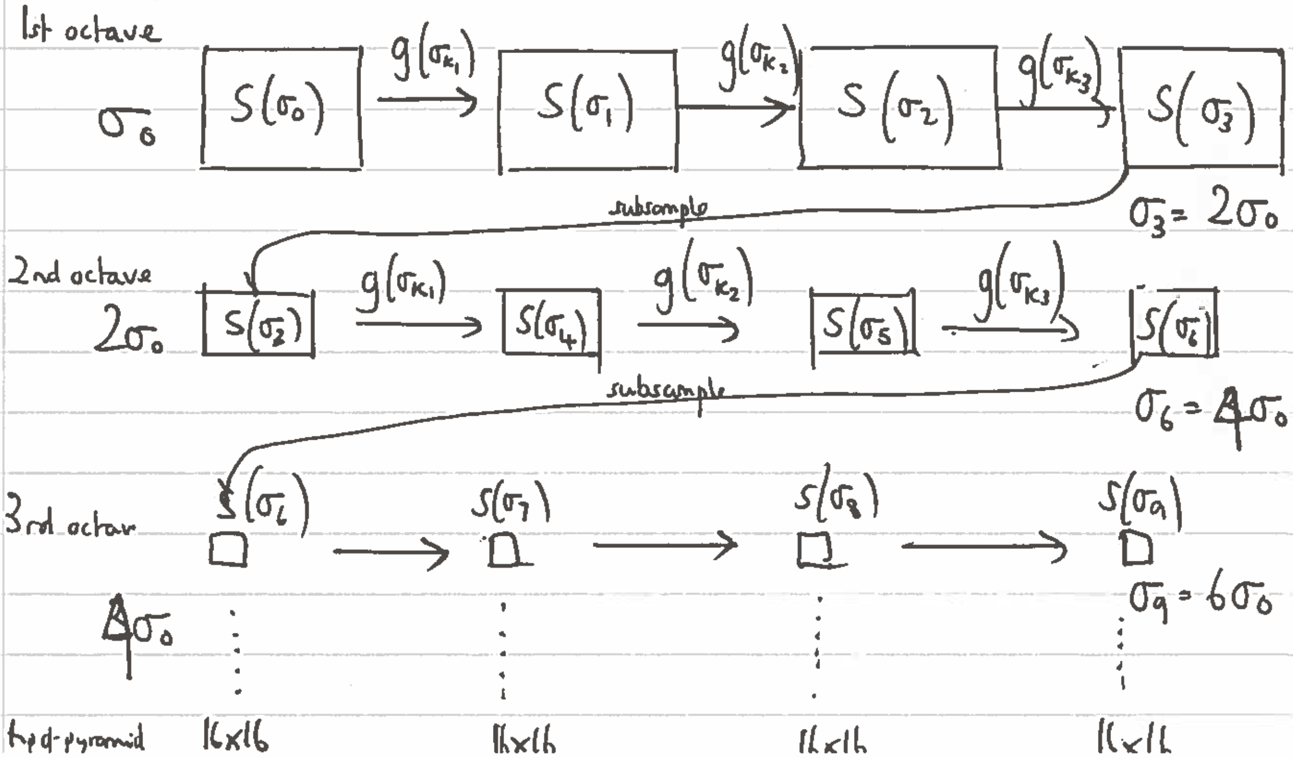
$$g(\sigma_{i+1}) = g(\sigma_i) * g(\sigma_K)$$

where $\sigma_K = \sigma_i \sqrt{2^{\frac{2}{s}} - 1}$ (1)

- sub-sample (reduce image size by $\frac{1}{4}$) after scale doubles (i.e. $\sigma_i = 2\sigma_0$) to generate next octave (1)

- apply <u>same</u> $g(\sigma_K)$ kernels to next octave image (1)

See example with $s=3$ ($s=3$ used by LOWE in SIFT implementation).



(b)(i). Band-pass filtering with the Laplacian of a Gaussian

$$\nabla^2\big(G(x,y,\sigma_i) * I(x,y)\big) = \nabla^2 G(\sigma_i) * I = \nabla^2 S(\sigma_i)$$

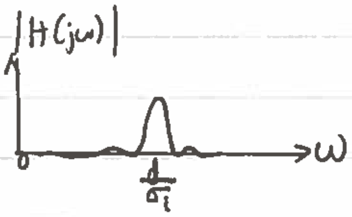
(2) $$\simeq S(\sigma_{i+1}) - S(\sigma_i)$$

i.e Approximated by difference neighbouring images in same octave for appropriate s

(1) DOG – difference of gaussian image if $\sigma_{i+1} \simeq 1.2 \to 1.5\,\sigma_i$

This is a band-pass filter since difference of gaussian in space and frequency

(1)

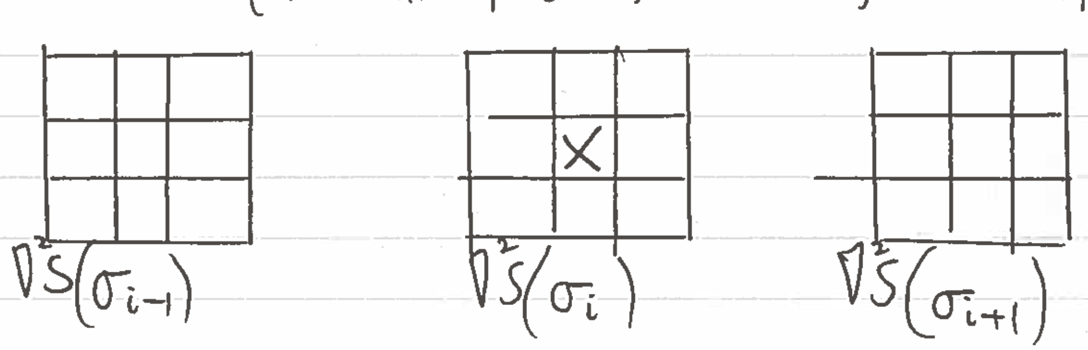


Find blob-like centre by looking for max/min of $\nabla^2 G(\sigma_i) * I$ resp

Search over scale-space representation – use DOG images for efficiency

(1) Need to check (at maximum) 26 neighbours in images beside interest pt.

use DOG images



(1) Position (x,y) and scale (σ_i) correspond to local max/min in pyramid of difference of gaussians images

1(b)(ii) Dominant orientation of feature in vicinity of blob centre:

(1) - Sample 16x16 pixels from $S(x, y, \sigma_i)$ at interest point centre (x_i, y_i)

(1) - Compute $\nabla S(x, y, \sigma_i)$ — ie. gradient magnitude and orientation

- ~~Produce a histogram of gradient magnitude against orientation - bins size of 10°~~

(1) Bin gradient magnitudes $\times$ $g(1.5\sigma_i)$ "weighted" in 10° bins from 0° to 360°

(1) - Smooth histogram and find dominant "peak".
This is orientation of feature point and used to give orientation invariance

- Can produce more accurate orientation by fitting a parabola to nearby bins and find θ

Q2.(a)(i). Assumptions: – pin-hole camera, planar perspective projection
– no non-linear distortion (eg radial lens)

– Image $u = \frac{x_1}{x_3}$, $v = \frac{x_2}{x_3}$

– Cartesian world 3D $X = \frac{X_1}{X_4}$, $Y = \frac{X_2}{X_4}$, $Z = \frac{X_3}{X_4}$

homogeneous co-ordinates

(2)

(ii). $$u_i = \frac{p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}}$$

$$v_i = \frac{p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}}$$

(2)

We can re-arrange to be linear in unknown p_{ij} and assemble into matrix below:

$$\underbrace{\begin{bmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -u_iX_i & -u_iY_i & -u_iZ_i & -u_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -v_iX_i & -v_iY_i & -v_iZ_i & -v_i \end{bmatrix}}_{2\times 12} \underbrace{\begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{31} \\ p_{32} \\ p_{33} \\ \vdots \end{bmatrix}}_{12\times 1} = 0$$

(2)

Q2(a)(iii)

- Need $N > 6$ 3D points of known (X_i, Y_i, Z_i) which span region/volume of interest at different depths (ie. NOT coplanar). Large FO

(2) Measure image (u_i, v_i) correspondences.

- $N > 6$ stack measurement equations in (a)(ii) to give a $2N \times 12$ matrix, A

(2) Solve $A\underline{p} = 0$ where $\underline{p} = \overset{12\times1}{\begin{bmatrix} p_{11} \\ \vdots \\ p_{34} \end{bmatrix}}$

- Solve by least-squares or find singular vector corresponding to smallest singular value (SVD).

ie. $$\lambda_1 < \frac{p^T A^T A p}{p^T p} < \lambda_{12}$$

This is a linear soln and only approximate.

- Use linear solution to initialize an non-linear optimisation (gradient descent)

(2) $$\min_{p_{ij}} \sum (u_i - \hat{u}_i)^2 + (v_i - \hat{v}_i)^2$$

where $\hat{u}_i$ and $\hat{v}_i$ are projected co-ordinates using p_{ij} and (u_i, v_i) is the measured image co-ordinates

Q2a(iv)

$$\overset{3\times4}{P} = \overset{3\times3}{K}\,\overset{3\times4}{[R\,|\,T]}$$

$$= [\underset{3\times3}{KR} \quad \underset{3\times1}{KT}]$$

(1)

We can decompose 9 elements of KR by QR decomposition

Obtain K and R from

$$\begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix}^T = (KR)^T = R^T T^T$$

$$|||$$

$$QR \text{ decomposition}$$

orthonormal / upper diagonal

Obtain position T

$$T = \overset{3\times3}{K^{-1}} \overset{3\times1}{\begin{bmatrix} p_{14} \\ p_{24} \\ p_{34} \end{bmatrix}}$$

(1)

Q2b(i) <u>Weak perspective</u>

$$Z_c = Z_A \quad \text{(orthogonal projection to plane at } Z_A\text{)}$$

(1)

$$\therefore \quad x = \frac{fX_c}{Z_A} \quad \text{and} \quad y = \frac{fY_c}{Z_A}$$

$$\therefore \begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} k \end{bmatrix} \underbrace{\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 0 & Z_A \end{bmatrix}}_{\text{weak perspective}} \begin{bmatrix} R & T \\ 000 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \underset{3\times4}{\overset{3\times4}{\begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ 0 & 0 & 0 & p_{34} \end{bmatrix}}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad ; \quad \underset{\text{image}}{\begin{bmatrix} u \\ v \end{bmatrix}} = \overset{2\times4}{\begin{bmatrix} 2\times4 \end{bmatrix}} \underset{\text{world}}{\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}$$

weak perspective projection matrix.

(3)

s = constant

Q2 b(ii) Projection equations become linear.

$$u_i = p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14}$$

$$V_i = p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24}$$

marks (2) Solve optimally by ~~linear~~ Least squares Need only 4 pts to calibrate.

9

Q3(a)(i) - Sample 16x16 pixels from image $S(x, y, \sigma_i)$ and dominant orientation of interest point.

(1) - Compute gradients ∇S in 16x16 pixels

(1) — weight by g_σ where scale is $\frac{1}{2}$ window of pixels (8), to weight distance from blob centre.

— produce 4x4 cells (16 cells)

(1) — in each cell produce a HOG at 45° bins (8 dirⁿˢ)
ie. add gradient magnitudes to each bin (8 sums)

8 directions

— concatenate to give 128D vector (8x16).

— normalise to unit length

(1) — remove effect of outliers due to big lights by truncating all values >0.2 (ie max entry = 0.2)

— renormalise to unit vector of 128D (SIFT)

(ii) Invariant to lighting (gradients and normalisation) and saturation (truncating)
Encodes 2D shape in vicinity of blob centre
(2) (ie. edges and their orientations and spatial distribution)
Invariant to small translations (histogramming, pooling).

Q3(a)(iii). Each interest point produces a 128D descriptor vector

which is invariant to scale, orientation and lighting

— Finding a correspondence is equivalent to nearest neighbour NN matching of these vectors either by correlation of euclidean distance.

(1)

— Compute euclidean distance between $\underline{x}$ and nearest neighbour $\underline{x}_1$

or Accept as a match if $\dfrac{\underline{x} . \underline{x}_2}{\underline{x} . \underline{x}_1} < 0.7$ (no need for a global threshold

where $\underline{x}_2$ is second nearest neighbour.

(1) Efficient NN by using a k-D tree (binary search) Organize keypoint descriptors into a tree and search $\log_2 N$.

Q3(b)

(i) - Assume we have n correspondences or correct matches between the 2 views (u, v) and (u', v').

- Each correspondence must satisfy epipolar constraint

(2) $$\begin{bmatrix} u' & v' & 1 \end{bmatrix} \begin{bmatrix} & & \\ & F & \\ & & \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = 0$$

where using $n \geq 8$ we can solve for F and approximate to nearest 3x3 matrix with $\det F = 0$ by SVD.

(1) ie. Solve $A \underline{f} \approx 0$

$$\underbrace{\underline{f}}_{\begin{bmatrix} 9\times1 \\ \\ \end{bmatrix}}$$

(1) - $F = K^{-1} T_x R K^{-1}$

(1) - Solve for $E = K^T F K$ using known K (internal parameters)

- $E = T_x R$.

(1) - Solve for T_x and R matrices by SVD of $E = U \Lambda V^T$

Q3(b)(ii) Left projection matrix

$$P_1 = K[I|0] \quad \text{or} \quad \begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} & P_1 & \\ & 3\times4 & \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

right projection matrix

(2) $$P_2 = K[R|T] \quad \text{or} \quad \begin{bmatrix} su' \\ sv' \\ s \end{bmatrix} = \begin{bmatrix} P_2 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Q3b(iii) Each image pt gives equation of a ray (2 planes)
Solve by triangulation of rays.
Equivalent to solving for (X,Y,Z) from 4 linear equations by
(2) least squares.

Optimize both P_{ij} and $\underline{X}_k$ over many views by tracking features and bundle-adjustment.

(2) Complete reconstruction requires matching more densely. Use feature matches to guide dense pixel matching in stereo.

Q 4 a) Provide a probabilistic interpretation for the network's output and use this to justify the form of the objective function. [20%]

Answer

Let the probability that a person's age is t^n given the image of their face $Z^{(n)}$ and the network weights V, W be given by a Gaussian $p(t^{(n)}|Z^{(n)}, V, W) = \mathcal{G}(t^{(n)}; x^{(n)}, \sigma^2)$. In this way, the output of the network $x^{(n)}(Z^{(n)}, V, W)$ is the mean of the Gaussian and σ^2 is its variance. The probability of the dataset labels given the parameters is therefore:

$$p(D|V, W) = \prod_{n=1}^{N} p(t^{(n)}|Z^{(n)}, V, W)$$
$$= \frac{1}{\mathcal{Z}(\sigma^2)} \exp\left(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(t^{(n)} - x^{(n)})^2\right) \tag{1}$$

Placing independent zero-mean Gaussian priors over the weights $V_{i,j}$ and $W_{i,j}$ with variance $1/\alpha$ and $1/\beta$ respectively, yields:

$$p(V, W|\alpha) = \prod_{i,j} p(w_{i,j})p(v_{i,j}) = \prod_{i,j} \frac{1}{\mathcal{Z}(\alpha, \beta)} \exp\left(-\frac{1}{2}\left(\alpha W_{i,j}^2 + \beta V_{i,j}^2\right)\right) \tag{2}$$

In this way we can interpret the objective function as relating to the probability of the weight vector given the training data, $p(W, V|D, \alpha) = \frac{1}{\mathcal{Z}(\alpha,\beta)\mathcal{Z}(\sigma^2)} \exp(-G(V, W))$.

b) Describe how to train the network's convolutional weights W using gradient descent. Compute the derivative required to implement gradient descent. Simplify your expression and interpret the terms. [40%]

Answer

The gradient descent algorithm operates as follows:

i. initialise the weights (e.g. using Gaussian noise with a small variance)

ii. compute the derivative of the objective function with respect to the convolutional weights $\frac{dG(V,W)}{dW_{i,j}}$

iii. step down the gradient $W_{i,j} \leftarrow W_{i,j} - \eta\frac{dG(V,W)}{dW_{i,j}}$ (η is a user defined learning rate)

iv. loop to step (2) until $\Delta G(V, W) < \text{tol}$

To compute the derivative we use backpropagation (aka the chain rule)

$$\frac{d}{dW_{a,b}}G(V, W) = \sum_{n=1}^{N}\sum_{i,j} \frac{dG(V, W)}{dx^{(n)}}\frac{dx^{(n)}}{dy_{i,j}^{(n)}}\frac{dy_{i,j}^{(n)}}{da_{i,j}^{(n)}}\frac{da_{i,j}^{(n)}}{dW_{a,b}} + \beta W_{a,b}. \tag{3}$$

where each of the terms are,

$$\frac{dG(V, W)}{dx^{(n)}} = \frac{1}{\sigma^2}(x^{(n)} - t^{(n)}), \quad \frac{dx^{(n)}}{dy_{i,j}^{(n)}} = V_{i,j} \tag{4}$$

$$\frac{dy_{i,j}^{(n)}}{da_{i,j}^{(n)}} = f'(a_{i,j}^{(n)}) \quad \frac{da_{i,j}^{(n)}}{dW_{a,b}} = Z_{i-a,j-b}^{(n)}. \tag{5}$$

14/14

Q4 (cont)

Combining the terms together yields

$$\frac{\mathrm{d}}{\mathrm{d}W_{a,b}}G(V,W) = -\frac{1}{\sigma^2}\sum_{n=1}^{N}\left(\mathrm{t}^{(n)} - x^{(n)}\right)\sum_{i,j} V_{i,j} f'(a_{i,j}^{(n)}) Z_{i-a,j-b}^{(n)} + \beta W_{ik}. \tag{6}$$

So, the derivative is simply the sum over all datapoints of the error between the predicted and true ages $\left(x^{(n)} - \mathrm{t}^{(n)}\right)$ multiplied by the sensitivity of the network's output on the convolutional weights plus a linear weight decay term. The sensitivity is a convolution between the product of the output weights and non-linearity derivative $V_{i,j} f'(a_{i,j}^{(n)})$ and the image flipped in the x and y directions $Z_{-i,-j}^{(n)}$.

c) Describe enhancements to the architecture of the network that might improve its ability to estimate the age of a person from an image of their face. [40%]

Answer

There are lots of possible ways of improving the architecture of the network.

i. A first enhancement would use **additional sets of convolutional weights**. Currently the method only uses one set and this means that it is only able to extract a single feature (e.g. a specific oriented edge) to perform regression.

ii. A second enhancement, would use a **pooling/subsampling** stage after the non-linear stage. This would pool over a local neighbourhood and pick e.g. the max or average value. This will introduce shift invariance and reduce the number of parameters that are required in the layers above.

iii. A third enhancement would be to use a neural network with **many layers** each of which is structured as above. Together these enhancements lead to deep convolutional neural networks.

The answer should describe these enhancements in detail which is bookwork.