Exam-like Solutions

1.1
$$w(i,j,k,l) = \exp\left(-\frac{(i-k)^2+(j-l)^2}{2\sigma_r^2} - \frac{\|f(i,j)-f(k,l)\|^2}{2\sigma_d^2}\right)$$

1.2
$$G(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2}{2\sigma^2}\right)\exp\left(-\frac{y^2}{2\sigma^2}\right) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{x^2}{2\sigma^2}\right)\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{y^2}{2\sigma^2}\right)$$
$$= G(x)G(y)$$

1.3
$$S(i,j) = S(i-1,j) + S(i,j-1) - S(i,j) + f(i,j)$$

1.4
$$(f*g)(x) = \int f(u)g(x-u)\,du \qquad \text{Let: } \tau = x-u \text{ and } d\tau = (-du)$$
$$= \int f(x-\tau)g(\tau)\,(-d\tau)$$
$$= -\int g(\tau)f(x-\tau)\,d\tau$$
$$= \int g(u)f(x-u)\,du$$
$$= (g*f)(x)$$

1.5
$$f*(g+h)(x) = \int f(u)(g+h)(x-u)\,du$$
$$= \int f(u)(g(x-u)+h(x-u))\,du$$
$$= \int f(u)g(x-u) + f(u)h(x-u)\,du$$
$$= \int f(u)g(x-u)\,du + \int f(u)h(x-u)\,du$$
$$= (f*g)(x) + (f*h)(x)$$

1.6 Compute the SVD of the kernel matrix:

$$K = U\Sigma V^T$$

Take the highest singular values and vectors

horiz. kernel: $k_x = \sqrt{\sigma_0}\,u_0 \quad k_y = \sqrt{\sigma_0}\,v_0^T$

1.7 Show that $p_u(z) = \frac{1}{255}$

$$p_h(z) = \frac{\partial}{\partial z} F_u(z) = \frac{\partial}{\partial z} F_f(h^{-1}(z)) = p_f(h^{-1}(z)) \frac{\partial}{\partial z} h^{-1}(z)$$

$$\overset{(1)}{=} \frac{1}{255} \frac{\partial}{\partial z} h(h^{-1}(z)) \frac{\partial}{\partial z} h^{-1}(z)$$

(1) $\frac{\partial}{\partial z} h(z) = 255\, p_f(z)$

$$= 1$$

1.8 Proof: Every linear, shift-invariant operator T can be written as a convolution

1. Represent the image through a convolution

$$(f * \delta)(i,j) = \iint f(u,v)\, \delta(i-u, j-v)\, du dv$$

2. Apply T on the image

$$T \circ (f * \delta)(i,j) = T \circ \iint f(u,v)\, \delta(i-u, j-v)\, du dv$$

$$= \iint f(u,v) (T \circ \delta)(i-u, j-v)\, du dv \quad | \text{ linearity}$$

$$= \iint f(i-u, j-v) (T \circ \delta)(u,v)\, du dv \quad | \text{ shift invariance}$$

$$= (f * (T \circ \delta))(i,j)$$

2.1 $f + (f - f * g) = 2f - f * g = f * (2e - g)$

2.2

$$h[m,n] = \frac{\sum_{k,l} (g[k,l] - \bar{g})(f[m+k, n+l] - \bar{f})}{\sum_{k,l} (g[k,l] - \bar{g})^2 \sum_{k,l} (f[m+k, n+l] - \bar{f})^2}$$

2.3 Sampling theory: Multiply the continuous image with a sha function (impulse train)

$$s(x) = \sum_{n=-\infty}^{\infty} \delta(x + n x_0) \qquad x_0\text{: samp. frequency}$$

$$f_s(x) = f(x)\, s(x)$$

This corresponds to a convolution in the frequency domain with a sha function of $1/x_0$ which produces infinite copies of the frequency spectrum of the image. In order to prevent aliasing, overlapping of frequencies producing new frequencies, we have to sample at at least double the frequency of the max frequency observed in the original image.

$$u_{max} \leq \frac{1}{2 x_0}$$

2.4

1. Produce Laplacian pyramids of both images
2. Produce gaussian pyramid of the to-blend region
3. Blend pyramids:

$$LS_n(i,j) = G_n(i,j)\,LA_n(i,j) + (1-G_n(i,j))\,LB_n(i,j)$$

4. Combine Laplacian pyramids:

$$G_n = LS_n + \text{expand}(G_{n+1})$$

3.1
For edge pixels:

For gradient directions:

$d = x\cos\Theta + y\sin\Theta$

$H[\Theta, d] += 1$

Find maximum in hough space to define lines

3.2 Use formula: $(y-y_0)^2 = 4p(x-x_0)$

3.3 Iteratively compute the center of density and shift the mean towards it until convergence

Mean shift vector: $$m(x) = \left(\frac{\sum_i x_i\, g(\|x_i - x\|^2)}{\sum_i g(\|x_i - x\|^2)} - x \right)$$

4.1

Iterate until convergence:

1. Assign each data point to the closest cluster center
2. Recompute cluster centers dependent on points

convergence: no change in centers $\Rightarrow$ no new assign.

4.2

k-means:

+ easy to implement, generic framework
+ always converges
+ fast and efficient computation
+ enforces to find k diff. clusters
- prone to outliers
- k has to be set
- sensitive to initialization

Mean shift:

+ robust against outliers
+ no direct assumption on cluster shapes
- density might be inefficient to compute
- harder to implement
- multiple runs for multiple clusters, might converg to same cluster

4.3.

0. Define every point as a separate cluster

1. Compute proximity matrix according to the inter-class similarity metric
(max, min. group average, center distance, Ward's method)

Iterate until convergence:

2. Find closest clusters and merge

3. Update proximity matrix

4.4. Solve the min. norm cut problem through:

$$\frac{y^T(D-W)y}{y^TDy}$$

This gives the generalized eigenvalue problem:

$$(D-W)y = \lambda D y$$

This can be transformed to a standard eigenvalue problem and solved upwards:

$$D^{-\frac{1}{2}}(D-W)D^{-\frac{1}{2}}z = \lambda z \qquad z = D^{\frac{1}{2}}y$$

Binary: We can now take the second smallest eigenvector, reshape it to the image size and threshold it to get the segmentation

Multi: Stack the k-smallest (except smallest) into the Laplacian Eigenmap and perform k-means in that

4.5 Define Rayleigh coefficient with known optimization problem

$$\frac{w^T S_B w}{w^T S_w w} \quad \text{with} \quad \min -\frac{1}{2} w^T S_B w \quad \text{s.t. } w^T S_w w = 1$$

Formulate through Lagrange equation

$$L(w,\lambda) = -\frac{1}{2} w^T S_B w + \frac{1}{2}\lambda(w^T S_w w - 1)$$
$$= -\frac{1}{2} w^T S_B w + \frac{1}{2}\lambda w^T S_w w - \frac{1}{2}\lambda$$
$$= -\frac{1}{2} w^T (S_B - \frac{1}{2}\lambda S_w) w - \frac{1}{2}\lambda$$

Compute derivative and set zero:

$$\frac{\partial}{\partial w} L(w,\lambda) = -\frac{1}{2}(S_B - \frac{1}{2}\lambda S_w) w \stackrel{!}{=} 0$$

$$\Leftrightarrow -\frac{1}{2} S_B w + \frac{1}{2}\lambda S_w w = 0$$

$$\Leftrightarrow S_B w = \lambda S_w w$$

4.6 $$\min \text{NormCut}(C_1, C_2) = \min \text{Cut}(C_1, C_2)\left(\frac{1}{\text{Vol}(C_1)} + \frac{1}{\text{Vol}(C_2)}\right)$$

5.1 $$E_{total} = E_{internal} + E_{external}$$

$$E_{internal} = \left|\frac{\partial v}{\partial s}\right|^2 + \left|\frac{\partial^2 v}{\partial s^2}\right|^2 = \sum_{i=0}^{n-1} \|v_{i+1} - v_i\|^2 + \sum_{i=0}^{n-1} \|v_{i+1} - 2v_i + v_{i+1}\|^2$$
$$E_{external} = -\sum_{i=0}^{n-1} |G_x(v_i)|^2 + |G_y(v_i)|^2$$

5.2 $$E(C) = -\int_0^1 |\nabla I(C(s))|^2 ds + \alpha \int_0^1 C_s(C(s)) ds + \beta \int_0^1 C_{ss}(C(s)) ds$$

5.3 $$\frac{\partial C(s,t)}{\partial t} = -\frac{\partial E}{\partial C} = \nabla|\nabla I(C)|^2 - \alpha C_{ss}(C) + \beta C_{ssss}(C)$$

$$C_{ss}(C) = \frac{u(s_{i+1}) - 2u(s_i) + u(s_{i-1})}{\partial s^2}$$

$$C_{ssss}(C) = \frac{u(s_{i+2}) - 4u(s_{i+1}) + 6u(s_i) - 4u(s_{i-1}) + u(s_{i-2})}{\delta s^4}$$

5.4 $C(s) = \{x \in \Omega : \phi(x) = 0\}$

5.5 Signed distance function: $\phi(s) = \pm \mathrm{dist}(s, C)$

5.6 Geodesic active contours:

Combination of level sets and representing the contour through continuous energy function using the geodesic length along the contour

$$E(C) = \int_0^1 \omega(|\nabla I|^2)\, ds$$

5.7 Employ the Euler-Lagrange equations

$$\frac{\partial C}{\partial t} = -\nabla\phi^T \frac{\partial E}{\partial C} = -\omega \nabla\phi \,\mathrm{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right) + (\nabla\omega^T \nabla\phi)$$

$$= \frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{\phi_x^2 + \phi_y^2}$$ | penalize elasticity

$$+ \max(\omega_x, 0)(\phi(x+1,y) - \phi(x,y))$$
$$+ \min(\omega_x, 0)(\phi(x,y) - \phi(x-1,y))$$
$$+ \max(\omega_y, 0)(\phi(x,y+1) - \phi(x,y))$$
$$+ \min(\omega_y, 0)(\phi(x,y) - \phi(x,y-1))$$ | move along image gradients

6.1
$$P(\beta,\gamma) + P(\alpha,\tau) \geq P(\beta,\tau) + P(\alpha,\gamma)$$

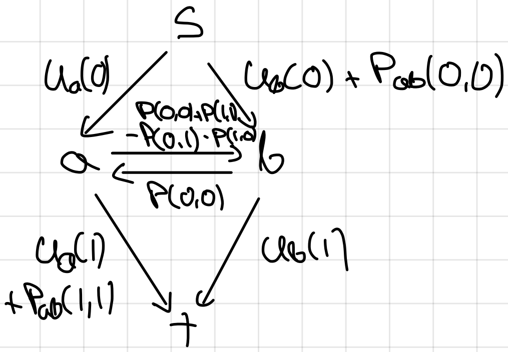
for $\beta > \alpha$ and $\tau > \gamma$

6.3
$$P(\omega_{1..n}) = \frac{1}{n}\prod_i^n \phi(\omega_i) = \frac{1}{n}\sum_i \exp(\varphi(\omega_i) \qquad \varphi(\omega_i) = \log \phi(\omega_i)$$

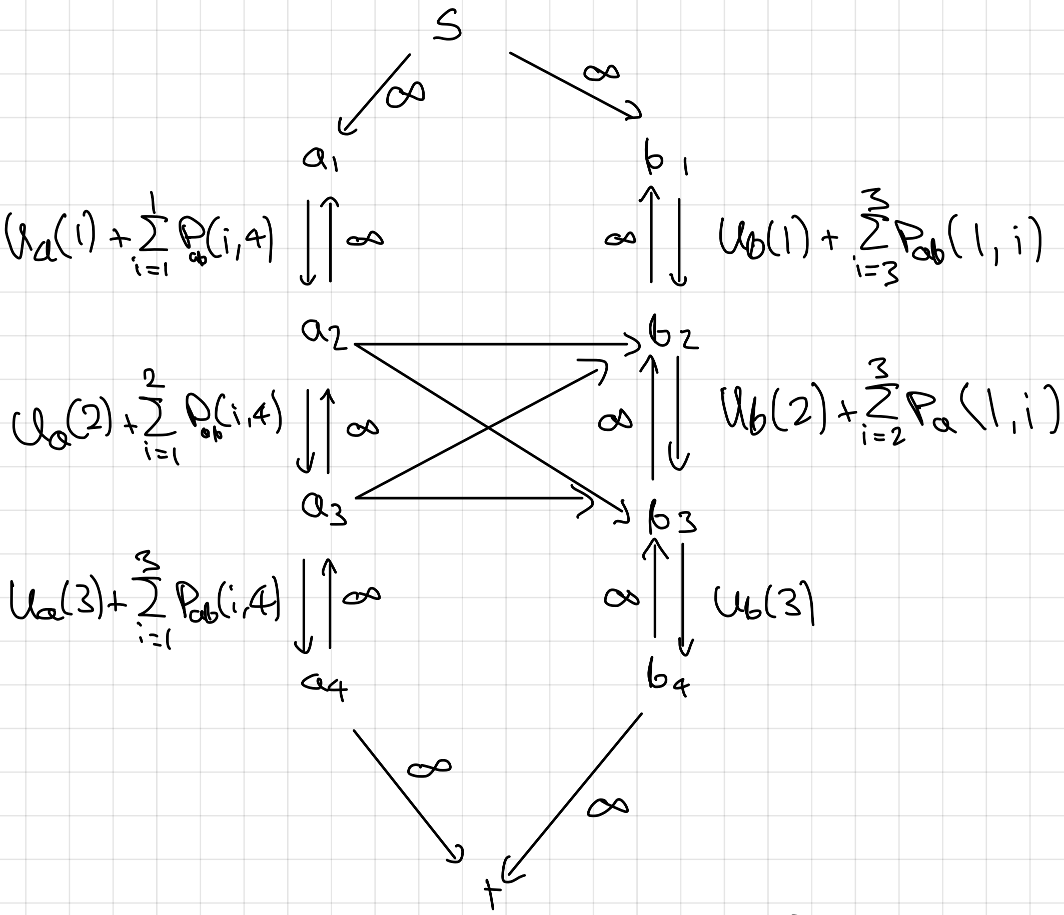
6.4 $$\hat{\Theta} = \underset{\Theta}{\mathrm{argmax}}\left[\prod_{i=1}^{I} P(x_i|\Theta) P(\Theta)\right]$$

6.5. $$C_{ab}(i,j) = P_{ab}(i,j) + P_{ab}(i-1,j-1) - P_{ab}(i-1,j) - P_{ab}(i,j-1)$$

6.6



6.7



7.1 Lower bound: $B(\{q_i(h_i)\},\theta)=\sum_{i=1}^{I}\int q_i(h_i)\log\left[\frac{P(x_i,h_i|\theta)}{q_i(h_i)}\right]dh_i$

$$\leq \sum_{i=1}^{I}\log\left[\int P(x_i,h_i|\theta)\,dh_i\right]$$

E-step: Maximize lower bound w.r.t. $q_i(h_i)$

$$q_i(h_i)=P(x_i|h_i,\theta)=\frac{P(h_i|x_i,\theta)P(x_i|\theta)}{P(x_i)}$$

7.2

M-step: Maximize bound w.r.t. θ

$$\hat{\theta}=\underset{\theta}{\operatorname{argmax}}\left[\sum_{i=1}^{I}\int P(x_i,h_i|\theta)\,dh_i\right]$$

7.3

$$P(x|\theta) = \sum_{i=1}^{K} \lambda_i \mathbb{N}_x[\mu_i, \Sigma_i]$$

7.4

$$q_i(h_i) = r_{ik} = \frac{\lambda_k \mathbb{N}_x[\mu_k, \Sigma_k]}{\sum_{j=1}^{K} \lambda_j \mathbb{N}_x[\mu_j, \Sigma_j]}$$

7.5

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\left[\sum_{i=1}^{I} \sum_{k=1}^{K} r_{ik} \log\left[\lambda_k \mathbb{N}_x[\mu_k, \Sigma_k] \right] \right]$$

7.6

$$\lambda_k = \frac{\sum_i r_{ik}}{\sum_i \sum_k r_{ik}} \qquad \mu_k = \frac{\sum_i r_{ik} x_i}{\sum_k r_{ik}} \qquad \Sigma_k = \frac{\sum_i r_{ik}(x_i - \mu_k)(x_i - \mu_k)^T}{\sum_k r_{ik}}$$

7.7

7.9

weights: $w_i = \sum_{u=1}^{m} \delta(b(x_i)-u)\sqrt{\frac{\hat{q}_u}{\hat{p}(y_0)}}$

Mean shift: $\hat{y}_1 = \frac{\sum_i x_i w_i g(\|\frac{\hat{y}_0 - x_i}{h}\|^2)}{\sum_i w_i g(\|\frac{\hat{y}_0 - x_i}{h}\|^2)}$

Similarity: $\rho[\hat{p}(y_1), \hat{q}] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(y)\hat{q}_u}$

Algorithm: While $\rho[\hat{p}(y_1), \hat{q}] < \rho[\hat{p}(y_0), \hat{q}]$:

$\hat{y}_1 = \frac{1}{2}(\hat{y}_1 - y_0)$

Update until: $\|\hat{y}_1 - \hat{y}_0\| < \varepsilon$

8.1

Prediction: $P(x_t | z_{1 \dots t-1}) = \int P(x_t | x_{t-1}) P(x_{t-1} | z_{1 \dots t-1}) dx_{t-1}$

Filtering: $P(x_t | z_{1 \dots t}) = \frac{P(z_t | x_t) P(x_t | z_{1 \dots t-1})}{\int P(z_t | x_t) P(x_t | z_{1 \dots t-1}) dx_{t-1}}$

8.2

Temporal evolution: $x_t = \mu_p + \Psi x_{t-1} + \varepsilon_p = N_{x_t}[\mu_p + \Psi x_{t-1}, \Sigma_p]$

Measurement eq.: $z_t = \mu_m + \Phi x_t + \varepsilon_m = N_{z_t}[\mu_m + \Phi x_t, \Sigma_m]$

Prediction: $x_t^p = \mu_p + \Psi x_t \quad \Sigma_t = \Sigma_m + \Psi \Sigma_{t-1} \Psi^T$

Filtering: $x_t = x_t^p + K(z_t - \mu_m - \Phi x_t^p) \quad \Sigma_t = (1 - K\Phi)\Sigma_t^p$

$K = \Sigma_t^p \Phi^T (\Sigma_m + \Phi \Sigma_t^p \Phi^T)$

8.3
- linear models
- independent measurements
- gaussian distributed errors

8.4 Sigma point sampling: $\hat{x}_0 = \mu$

$$\hat{x}_i = \mu \pm \sqrt{\frac{2D_w}{1-a_0}} \Sigma^{1/2} e_i$$

$$a_i = \frac{2D_w}{1-a_0}$$

Prediction: $\hat{x}_i^p = f[\hat{x}_i, 0]$

$$x_t^p = \sum_{i=0}^{2D_w} a_i \hat{x}_i^p \qquad \Sigma_t^p = \sum_{i=0}^{2D_w} a_i (\hat{x}_i^p - x_t^p)(\hat{x}_i^p - x_t^p)^T$$

Filtering: $\hat{z}_i = g[\hat{x}_i^p, 0]$

$$\mu_z = \sum_{i=0}^{2D_w} a_i \hat{z}_i \qquad \Sigma_z = \sum_{i=0}^{2D_w} a_i (\hat{z}_i - \mu_z)(\hat{z}_i - \mu_z)^T$$

$$x_t = x_t^p + K(z_t - \mu_z) \qquad \Sigma_t = \Sigma_z - K \Sigma_t^p K^T$$

$$K = \left(\sum_{i=0}^{2D_w} a_i (\hat{z}_i - \mu_z)(\hat{x}_i^p - x_t^p)^T \right) \Sigma_t^{-1}$$

8.4 Linearization: $\varphi = \frac{\partial f[x_t, \varepsilon_p]}{\partial x_t}$ $\Upsilon_p = \frac{\partial f[x_t, \varepsilon_p]}{\partial \varepsilon_p}$ $\phi = \frac{\partial g[x_t, \varepsilon_m]}{\partial x_t}$ $\Upsilon_m = \frac{\partial g[x_t, \varepsilon_m]}{\partial \varepsilon_m}$

Prediction: $x_t^p = f[x_{t-1}, 0]$ $\Sigma_t^p = \Upsilon_p \Sigma_p \Upsilon_p^T + \varphi \Sigma_{t-1} \varphi^T$

Filtering: $x_t = x_t^p + K(z_t - g[x_t^p, 0])$ $\Sigma_t = (1 - K\phi)\Sigma_t^p$

$$K = \Sigma_t^p K (\Upsilon_m \Sigma_m \Upsilon_m^T + \phi \Sigma_t^p \phi^T)$$

8.5

1. Prediction: Evolve the particles according to the evolution model

2. Filtering: Weight each particle according to the measurement model

3. Resampling: Resample particles according to their weights

8.6 Rejection sampling

1. Sample N random numbers in $[0,1] \rightarrow U[N]$

2. For i in $[1, \dots, N]$:

   While $P[j] < U[i]$; j++

   $P'[i] = P[j]$

8.7 Expectation Maximization

E-step: Filtering with fixed interval smoother

M-step: Maximize parameters w.r.t. the filtered states

$$Q = \log P(x,z) = -\sum_{i=1}^{T} \frac{1}{2}(z_t - \phi x_t)^T \Sigma_m^{-1} (z_t - \phi x_t) - \frac{T}{2}\log|\Sigma_m|$$
$$- \sum_{i=1}^{T-1} \frac{1}{2}(x_t - \phi x_{t-1})^T \Sigma_p^{-1} (x_t - \phi x_{t-1}) - \frac{T-1}{2}\log|\Sigma_p|$$
$$- \frac{1}{2}(x_1 - x_0)^T \Sigma_0^{-1} (x_1 - x_0) - \frac{1}{2}\log|\Sigma_0|$$

8.8

$$\varphi = \frac{\sum_i^{T-1} x_t x_{t-1}^T}{\sum_i^{T-1} x_{t-1} x_{t-1}^T} \qquad \phi = \frac{\Sigma_t^T z_t x_t^T}{\Sigma_t^T x_t x_t^T}$$

$$\Sigma_p = \frac{1}{T-1}\sum_{t=1}^{T-1}(x_t x_t^T - \phi x_t z_t^T) \qquad \Sigma_m = \frac{1}{T}\sum_{t=1}^{T}(x_{t-1} x_{t-1}^T - \varphi x_{t-1} x_T^T)$$

8.9

$$x_{t|T} = x_t + C_t(x_{t+1|T} - x_t^p)$$
$$\Sigma_{t|T} = \Sigma_t + C_t(\Sigma_{t+1|T} - \Sigma_t^p)\, C_t$$
$$C_t = \Sigma_t \varphi^T \Sigma_t^{p\cdot 1}$$

8.10

Temporal evolution: $$\begin{bmatrix} x_t \\ \vdots \\ x_t^{\tau} \end{bmatrix} = \begin{bmatrix} \varphi & & & \\ 1 & 0 & & \\ & 1 & 0 & \\ & & \ddots & \\ & & 1 & 0 \end{bmatrix} \begin{bmatrix} x_{t-1} \\ \vdots \\ x_{t-1}^{\tau} \end{bmatrix} + \begin{bmatrix} \varepsilon_p \\ \vdots \\ 0 \end{bmatrix}$$

Measurement eq.: $$z_t = [\phi\ 0 \ldots 0] \begin{bmatrix} x_t \\ \vdots \\ x_t^{\tau} \end{bmatrix} + \Sigma_m$$

9.1 Shape defined through a set of landmark points

Likelihood: $P(x|\omega,\psi) \propto \prod_{i=1}^{N} \exp(-\text{dist}(x_i, \text{trans}(w_i, \psi))$

9.2 We can represent the shape with a prior distribution on the landmarks instead of fixed landmarks which allows some kind of elasticity

9.3 0. Align the points to a mean shape (reference example)

Iterate until convergence:

1. Solve affine transformation to mean shape

2. Apply transformation to each shape

3. Compute mean and variance for each landmark

9.4 A Subspace Shape Model represents the shape through a .et of lower-dimensional latent variables:

$w = \mu + \phi h + \varepsilon$

9.5 0. Initialize h to zero (mean shape): $y = \mu + \phi 0$

Iterate until convergence:

1. Compute affine transformation $T(x_t, y_t, s, \theta)$ best aligning y to the observed points

2. Apply inverse transform to the observed points ψ:

$y = T^{-1}(\psi)$

3. (optional) Map to tangent space $y' = y/(y^T\mu)$

4. Compute latent coefficients: $h = h^T(y' - \mu)$

9.6 1. Compute ϕ, μ, σ^2 according to PCA method

2. Subtract mean from points and perform eigen-decomposition:

$$W = [w_1 - \mu, \dots] \quad W^T W = U L^2 U^T$$

3. Set parameters: $\sigma^2 = \frac{1}{D-K} \sum_{i=K+1}^{D} L_{ii}^2$

$$\phi = U_K (L_K^2 - \sigma^2 I)^{\frac{1}{2}}$$

9.7 Iterate between

1. Update weightings:

$$\hat{h} = \underset{h}{\operatorname{argmax}} \left[\sum_{n=1}^{N} -(y_n - \operatorname{trans}(\mu + \phi h, \Psi))^2 / \sigma^2 - \log[h^T h] \right]$$

2. Update the transformation parameters

10.1 For each pixel:

1. Compute the second moment matrix:

$$\mu = \begin{bmatrix} \Sigma I_x I_x & \Sigma I_x I_y \\ \Sigma I_y I_x & \Sigma I_y I_y \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

2. Compute corner response and decide for corner if above threshold

10.2 $f(x) = |\mu| - k(\operatorname{Tr}(\mu))^2 = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \quad k \approx 0.04$

10.3 1. Compute gradients in neighborhood depending scale and sort into 128 bin histogram

2. Dominant orientation is bin with highest magnitude

3. Extract from a 16×16 neighborhood 16 4×4 subblocks and create 8 bin normalized histograms

$\rightarrow 4 \times 4 \times 8 = 128D$ descriptor

4 Cut-off over 0.2 and normalize

10.4 Filter the image at different scales with the Laplacian operator -> Laplacian pyramid

This can be approximated through the difference of gaussian

11.1

Iterate until convergence:

1. Randomly sample a seed group
2. Compute transform based on this
3. Compute inliers based on this and save number

4. Take set of most inliers and recompute transform

Convergence: # of inliers, steps etc.

11.2

$$P(good) = \frac{\binom{I}{m}}{\binom{N}{m}} = \prod_{j=0}^{m-1} \frac{I-j}{N-j}$$

N: # points, I: # inliers, m: # samples

Model how many bad examples we hit until we get a good sample with prob. p

$$P(\text{bad } k \text{ times}) = (1-P(good))^k \leq 1-p$$

$$\Leftrightarrow k \log[1-P(good)] \leq 1-p$$

$$\Leftrightarrow k \leq (1-p) / \log[1-P(good)]$$

11.3

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & x_1'x & x_1'y & x_1' \\ 0 & 0 & 0 & x_1 & y_1 & 1 & y_1'x & y_1'y & y_1' \\ & & & & \vdots & & & & \end{bmatrix} \begin{bmatrix} H_{00} \\ \vdots \\ H_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

12.1 $I(x,y,t-1) = I(x+u(x,y), y+v(x,y), t)$

Linearization: $I(x,y,t-1) = I(x,y,t) + I_x u(x,y) + I_y v(x,y)$

Implies: $I_x u + I_y v + I_t = 0$

12.2 $\nabla I(x,y,t-1) = \nabla I(x+u(x,y), y+v(x,y), t)$

12.3

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ \vdots & \\ I_x(p_{25}) & I_y(p_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(p_1) \\ \vdots \\ I_t(p_{25}) \end{bmatrix}$$

12.4 The brightness constancy constraint is underdetermined and only defines a line, we can compute the normal flow to.

12.5 Solve the spatial coherence constraint for corners

1. Compute the second moment matrix

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

2. Check eigenvalues to determine if the pixel is a corner
3. Solve the system: $\begin{bmatrix} u \\ v \end{bmatrix} = (A^T A)^{-1} A^T b$

12.6 Assumption I: $I_x u + I_y v + I_t = 0$

II: $|\nabla u|^2 + |\nabla v|^2 \to \min$

$E(u,v) = \int (I_x u + I_y v + I_t)^2 \, dx dy + \alpha \int |\nabla u|^2 + |\nabla v|^2 \, dx dy$

Euler-Lagrange: $(I_x u + I_y v + I_t) I_x - \alpha \Delta u = 0$

$(I_x u + I_y v + I_t) I_y - \alpha \Delta v = 0$

$$\Delta u = \frac{u(i+1,j) + u(i-1,j) + u(i,j+1) + u(i,j-1) - 4u(i,j)}{h^2}$$

12.7/9/10 $E(u,v) = \int \Psi((I_x u + I_y v + I_t)^2) \, dx$

$+ \beta \int \Psi((I_{xx} u + I_{xy} + I_t)^2 + (I_{xy} u + I_{yy} v + I_t)^2) \, dx$

$+ \gamma \int \Psi(|\nabla u|^2 + |\nabla v|^2) \, dx$

12.8 Do not linearize

$$E(u,v) = \int \left(I(x,y,t-1) - I(x,y,t) \right)^2 dx + \alpha \int |\nabla u|^2 + |\nabla v|^2 dx$$

13.1 $x = PX$

$$(x,y,z) \to \left(f\frac{x}{z}, f\frac{y}{z}, f\right)$$

13.2 $\frac{1}{D} + \frac{1}{D'} = \frac{1}{f}$ → all points fullfilling this are in focus

13.3 Change the aperture size

13.4

$$K = \begin{bmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

13.5 Baseline: Line between camera centers

Epipolar line: Projection of camera ray from image A to image B

Epipole: Intersection baseline and image plane

Essential matrix: $E = [t_x]R$ transformation between camera centers

Fundamental matrix: Transformation between uncalibrated images

$$F = K^T E K^{-1}$$

13.6 $x_d = L(r)x$ $\quad L(r) = 1 + k_1 r + k_2 r^2 + \ldots$

13.7 $x_d = (1 + kc(1)r^2 + kc(2)r^4 + kc(5)r^6)x + dx$

$$dx = \begin{bmatrix} 2kc(3)xy + kc(4)(r^2 + 2x^2) \\ 2kc(4)xy + kc(3)(r^2 + 2y^2) \end{bmatrix}$$

13.8

Relation: $[h_1\ h_2\ h_3] = \lambda A^{-1} [r_1\ r_2\ t]$

$r_1 = A^{-1}h_1 / \|A^{-1}h_1\|$ $\quad r_2 = A^{-1}h_2 / \|A^{-1}h_2\|$ $\quad r_3 = r_1 \times r_2$

$t = A^{-1}h_3 / \|A^{-1}h_3\|$

13.9

Distortion model. $\hat{u} = u + (u - u_0)(k_1(x^2+y^2) + k_2(x^2+y^2)^2)$

$\hat{v} = v + (v - v_0)(k_1(x^2+y^2) + k_2(x^2+y^2)^2)$

Linear system:

$$\begin{bmatrix} (u-u_0)(x^2+y^2) & (u-u_0)(x^2+y^2)^2 \\ (v-v_0)(x^2+y^2) & (v-v_0)(x^2+y^2)^2 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix}$$

14.1 $\frac{Z}{B} = \frac{Z-f}{B+x'-x} \iff (B+x'-x)\frac{Z}{B} = Z-f$

$\iff Z - (x'-x)\cdot\frac{Z}{B} = Z - f$

$\iff x'-x = -\frac{f\cdot B}{Z}$

$\iff x - x' = \frac{f\cdot B}{Z}$

$\iff Z = \frac{f\cdot B}{x-x'}$

14.2 Photo hull: Set of scenes that produce photo-consistent images

Visual hull: Combination of silhouettes from all views

14.3 Map through homography that can be split into 3 separate transform

1. Projective transform: Make epipolar lines parallel

Solve $\frac{z^T e_x^T P P^T e_x z}{z^T e_x^T P_c P_c^T e_x z} + \frac{z^T F^T P' P'^T F z}{z^T F^T P_c' P_c'^T F z}$ $\quad H_p = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ w_a & w_b & 1 \end{bmatrix}$

$w = [e_x] z$
$w' = F z$

2 Similarity transform. Align epipolar lines

$$H_r = \begin{bmatrix} F_{32} - w_b F_{33} & w_a F_{33} - F_{31} & 0 \\ F_{31} - w_a F_{33} & F_{32} - w_b F_{23} & F_{33} + v_c' \\ 0 & 0 & 1 \end{bmatrix}$$

3. Shear transform: Map images to common image size

$$H_s = \begin{bmatrix} a & b & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

14.4 Represent space through voxels and assign RGB values to photo-consistent voxels

14.5

1. Compute visual hull
2. DP to find rims and constrain them to be fixed
3. Carve visual hull to optimize photo-consistency

15.1 Bundle adjustment aims at minimizing the reprojection error to estime the camera extrinsics and 3D points:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} \| x_{ij} - \pi(R_j, t_j, X_i) \|^2$$

15.2 Bundle Adjustment can be solved through the Levenberg-Marquardt algorithm.

Through the nature of BA the Jacobians have structure:

$$\left(\begin{array}{c|c} A^TA & A^TB \\ \hline B^TA & B^TB \end{array}\right)\left(\begin{array}{c} \delta A \\ \hline \delta B \end{array}\right) = \left[\begin{array}{c} A^T\varepsilon \\ \hline B^T\varepsilon \end{array}\right]$$

The resulting Hessian is inherently sparse through low correlation of camera parameters and points

15.3 In non-rigid SfM we have different shapes for each view and introduce additional ambiguity between motion and shape

$$\operatorname{rank}(W) \leq \min(2F, P)$$

15.4 1. Center the points and define measurement matrix W

2. Perform SVD to decompose W into motion and shape. We only take the three largest singular values and vectors

$$W = U\Sigma V^T \rightarrow M = U_3 \Sigma_3^{1/2} \quad S = \Sigma_3^{1/2} V_3^T$$

3. Eliminate affine ambiguity by enforcing euclidean constraints:

$$m_{i1} CC^T m_{i1} = 1$$
$$m_{i2} CC^T m_{i2} = 1 \quad \rightarrow \text{solve through Newton} \rightarrow$$
$$m_{i1} CC^T m_{i2} = 0 \qquad \text{or linearly}$$

$$M' = MC$$
$$S' = C^{-1} S$$

15.5

0. Solve for 3D points in the first two views through triangulation using the fundamental matrix

For each additional view:

1. Calibrate the new view using the visible 3D points
2. Recompute the visible 3D points through triangulation using the new view

15.6

Outter SVD: Decompose W into matrices $\hat{H}$ and $\hat{B}$

$$W = U \Sigma V^T = (U\Sigma^{\frac{1}{2}})(\Sigma^{\frac{1}{2}} V^T) = \hat{H}\hat{B}$$

$\hat{H}$: comb. of camera motion and shape weights
$\hat{B}$: shape basis

Inner SVD: Decompose $\hat{H}$ into the motion and weghts
→ Stack components of single view in single row $\hat{H}'$

$$\hat{H} = U \Sigma V^T = (U\Sigma^{\frac{1}{2}})(\Sigma^{\frac{1}{2}} V^T) = \hat{W}\hat{R}$$

Final perform metric rectify using orthonormality constraints to remove projective distortion

15.7

Linear shape model: $W = M \cdot W_s \cdot B$

Linear Traj. model: $W = M \cdot T \cdot W_t$

Shape basis can be used as traj. weights and vice versa

16.1 $(\Theta_x, \Theta_y, \Theta_z)$: rotation around x, y, z axes in predefined order

16.2 $(w_x, w_y, w_z), \Theta$: Define axis of rotation and magnitude

16.3 (q_1, q_2, q_3, q_4): 4d representation represent as imaginary numbers

16.4 When using Euler angles, rotation axes can align which leads to a momentary loss of a DOF

16.5

$$R = \exp(\theta\hat{\omega}) \approx 1 - \hat{\omega}\sin\theta + \hat{\omega}(1-\cos\theta) \qquad \hat{\omega} = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix}$$

16.6

$$\theta = \cos^{-1}\left(\frac{Tr(R)-1}{2}\right) \qquad \omega = \frac{1}{2\sin\theta}\begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix}$$