1. Filtering

Gamma Corr.: $h(x) = f(x)^{1/\gamma}$

Histogram equal.: $h(f(x)) = F(f(x)) \cdot 255$

Convolution: $(f*g)(x) = \int f(u) g(x-u) du$

→ linear, associative, commutative, shift inv., distributes over add., scalar out, identity

Histogram: $p_f(z) = \frac{|\{x \in \Omega : f(x) = z\}|}{|\Omega|}$

Cumulative dis.: $F_f(x) = \int_0^x p_f(x) dx'$

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

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

Integral image: $S(i,j) = \sum_{k=0}^{i} \sum_{l=0}^{j} f(k,l) = S(i-1,j) + S(i,j-1) - S(i-1,j-1) + f(i,j)$

Kernel separability: $K = U \Sigma V^T$ horiz.: $\sqrt{\sigma_0} u_0$ vert.: $\sqrt{\sigma_0} v_0^T$

2. Fourier & Pyramids

Euler's Formel: $e^{\pm j\theta} = \cos\theta \pm j \sin\theta$

Fourier Transform: $F(u) = \int f(x) e^{-2\pi j u x} dx = \frac{1}{N} \sum_{n=0}^{N-1} f(n) e^{-2\pi j \frac{n}{N} u}$

Inv. Fourier Trans.: $f(x) = \int F(u) e^{2\pi j u x} dx = \sum_{u=0}^{N-1} F(u) e^{2\pi j \frac{n}{N} u}$

Amplitude: $A = \pm\sqrt{R(\omega)^2 + I(\omega)^2}$ **Phase:** $\phi = \tan^{-1}\left(\frac{I(\omega)}{R(\omega)}\right)$

Conv. Theorem: $F[g*h] = F[g]F[h]$ $F[gh] = F[g] * F[h]$

Gaussian Pyramid: Gaussian blur + down sample

Laplacian Pyramid: $L_i = G_i + \text{expand}(G_{i+1})$

Laplacian of Gaussian: $f + (f - f*g) = 2f - f*g = f*(2e - g)$

Sampling: $f_s(x) = f(x) s(x) = f(x) \sum_{n=-\infty}^{\infty} \delta(x - n x_0)$

Nyquist Theorem: $u_{max} \le \frac{1}{2x_0}$

Correlation: $h[m,n] = \sum_{k,l} g[k,l] f[k+m, l+n]$

Zero-Mean Corr: $h[m,n] = \sum_{k,l} (g[k,l] - \bar{g}) f[k+m, l+n]$

Sum-square Diff.: $h[m,n] = \sum_{k,l} (g[k,l] - f[k+m, l+n])^2$

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

Laplacian Blending: $L(i,j) = G(i,j) \cdot LA(i,j) + (1 - G(i,j)) LB(i,j)$

→ Collapse $LS_i = LS_i + \text{expand}(LS_{i+1})$

3. Edges & Contours

Sobel: $\begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$ $\begin{matrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{matrix}$

Precision: $tp/(tp+fp)$

Recall: $tp/(tp+fn)$

Canny Edge:
1. Compute gradients
2. Compute direction & magnitude
3. Non-maxima suppression
4. Hysteresis: $\|\nabla f\| \ge t_1$ → edge
   $t_0 \le \|\nabla f\| < t_1$ → maybe edge
   $\|\nabla f\| < t_0$ → no edge

Gradient: $\theta = \tan^{-1}\left(\frac{g_y}{g_x}\right)$ $M = \sqrt{g_x^2 + g_y^2}$

Conv. Derivative: $\frac{\partial}{\partial x}(f*g)(x) = f(x) * \frac{\partial}{\partial x} g(x)$

Chamfer Dist.: $d(T,I) = \frac{1}{|T|} \sum_{t \in T} d_I(t)$

Distance Transform: Init: Sett all ∞
Forward: Min dist to left & up
Backward: Min dist to right & down

Hough: For edge pixels:
(further for loops for hough variables):
Compute parameters of structure
H[params] += 1
Find maxima in Hough space

Minkowski Dist.: $d_p(x,y) = \left(\sum_i |x_i - y_i|^p\right)^{1/p}$ $d_\infty(x,y) = \max(|x_i - y_i|)$

Mean Shift: $P(x) = \frac{1}{n} \sum_{i=1}^{n} K(x - x_i)$ $K(x) = c \prod_{i=1}^{n} k(x_i)$ or $K(x) = c k(\|x\|)$

$\nabla P(x) = \frac{2c}{N} \sum_{i=1} g(\|x_i - x\|^2) \left(\frac{\sum_i x_i g(\|x_i - x\|^2)}{\sum_i g(\|x_i - x\|^2)} - x \right)$

Curve Formulations:
- Line: $x\cos\theta + y\cos\theta = d$
- Parabola: $(y-y_0)^2 = 4p(x-x_0)$
- Circle: $(x-x_0)^2 + (y-y_0)^2 = r^2$
- Ellipse: $(x-x_0)^2/a^2 + (y-y_0)^2/b^2 = 1$

4. Segmentation

K-Means: 0. Randomly initialize k clusters
Iterate until convergence:
1. Determine points in each cluster
2. Recompute centers based on points

Agglomerative Clustering: 0. Initialize each point as cluster
1. Compute proximity matrix
Iterate until convergence:
2. Merge two closest cluster
3. Update proximity matrix

Material Classification: Compute similarity to examples
$\chi^2(h_i, h_j) = \frac{1}{2} \sum_{k=1}^{K} \frac{[h_i(k) - h_j(k)]^2}{h_i(k) + h_j(k)}$

Inter-Cluster similarity: min, max, distance center, group avg.

Ward's method: $d(A,B) = \sum_{i \in A\cup B} \|x_i - m_{A\cup B}\|^2 - \sum_{i\in A} \|x_i - m_A\|^2 - \sum_{i \in B} \|x_i - m_B\|^2$

Spectral Clustering: $\frac{y^T(D-W)y}{y^T D y}$ w/ $y(i) \in \{1, b\}$ $yD1 = 0$ $b = \frac{\sum_{x_i>0} d_i}{\sum_{x_i<0} d_i}$

Generalized Eigenvalue: $(D-W)y = \lambda D y$

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

Binary: take 2^{nd} smallest λ

Multi: k-means in Laplacian Eigenmap (k smallest λ)

Graphs:
- Degree: $D(x) = \sum_{i \in X} w_{ij}$
- Volume: $Vol(C) = \sum_{i \in C} D(x_i)$
- Cut: $Cut(C_1, C_2) = \sum_{i \in C_1} \sum_{j \in C_2} w_{ij}$
- Min Norm. Cut: $\min Cut(C_1, C_2) \left(\frac{1}{Vol(C_1)} + \frac{1}{Vol(C_2)} \right)$

5. Snakes and Level Sets

Energy: $E_{total} = E_{external} + E_{internal}$

Internal: $E_{internal} = \alpha \left|\frac{\partial v}{\partial s}\right|^2 + \beta \left|\frac{\partial^2 v}{\partial s^2}\right|^2 = \sum_{i=0}^{n-1} \alpha \|v_{i+1} - v_i\|^2 + \beta \|v_{i+1} - 2v_i + v_{i-1}\|^2$

External: $E_{external} = -\sum_{i=0}^{n-1} |G_x(x_i, y_i)|^2 + |G_y(x_i, y_i)|^2$

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

Dynamic Programming:

Objective: $\arg\min \left[\sum_{i=0}^{n-1} U_n(w_n) + \sum_{i=0}^{n-1} P_n(w_n, w_{n+1}) \right]$

$U_n(w_n) = -\|G(w_n)\|$ $P_n(w_n, w_{n+1}) = \alpha \|w_{n+1} - w_n\|^2$

Graph: Columns: vertices Nodes: unary costs
Rows: neighborhood nodes Edges: Pairwis costs

Algorithm: Compute minimal costs to reach each node, then backtrace

$S_{n,k} = U_n(w_n = k) + \min(S_{n-1,L} + P_n(w_n = k, w_{n-1} = L))$

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

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

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

Level sets: $C = \{x \in \Omega . \phi(x) = 0\}$
→ implicit repr. through ϕ

Transform: $\phi(x,y) = \pm dist((x,y), C)$

Optimization: $\frac{\partial \phi}{\partial t} = -\nabla \phi^T \frac{\partial C}{\partial t}$

Geodesic Active Contours: level set + geodesic dist

$E(C) = \int_0^1 w(|\nabla I(C(s))|^2) |C_s(s)| ds = \int_C w(|\nabla I|^2) ds$

Opt.: $\frac{\partial C}{\partial t} = -\frac{\partial E(C)}{\partial C} = -wkn - (\nabla w^T n) n$ $n = \frac{\nabla \phi}{|\nabla \phi|}$

$= w |\nabla \phi| div\left(\frac{\nabla \phi}{|\nabla \phi|}\right) + (\nabla w^T \nabla \phi)$ $k = div(n)$

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

$+ \max(w_x, 0)(\phi(x+1,y) - \phi(x,y)) + \min(w_x, 0)(\phi(x,y) - \phi(x-1,y))$

$+ \max(w_y, 0)(\phi(x,y+1) - \phi(x,y)) + \min(w_y, 0)(\phi(x,y) - \phi(x,y-1))$

6. Markov Random Fields

Potential: $P(w)=\frac{1}{Z}\prod_{i=1}^{I}\phi_i(w_{c_i})=\frac{1}{Z}\exp[-\sum_{i=1}^{I}\psi[w_{c_i}]]$

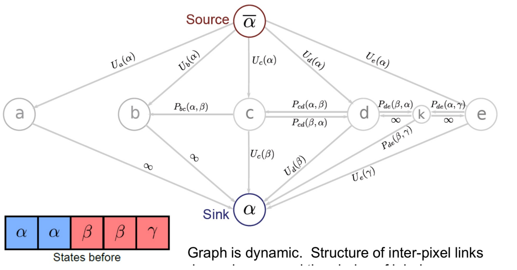
Denoising: $P(w_{1\ldots N}|x_{1\ldots N})=\frac{\prod_{i=1}^{I}P(x_i|w_i)P(w_i)}{P(x_{1\ldots N})}$

$P(x_i|w_i=0)=\text{Bern}_{x_i}[\rho]$

$P(x_i|w_i=1)=\text{Bern}_{x_i}[1-\rho]$

$P(w_{1..N})$: MRF

Alpha Expansion: Expand label by label



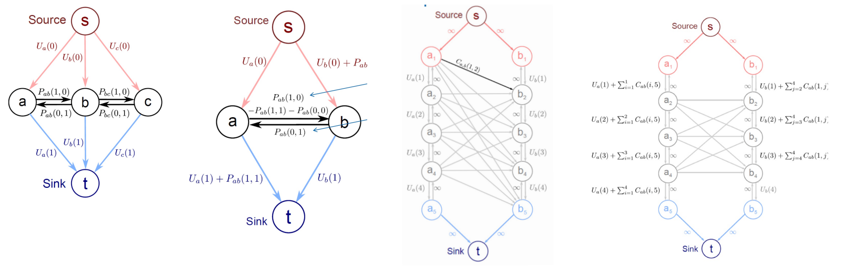
ML: $\hat{\Theta}=\underset{\Theta}{\text{argmax}}[\prod_{i=1}^{I}P(x_i|\Theta)]$

MAP: $\hat{\Theta}=\text{argmax}[\prod_{i=1}^{I}P(x_i|\Theta)P(\Theta)]$

Max-Flow: Take saturated edges as cuts for segmentation

Graph Construction:

Binary: **General binary:** **Multi:** **Multi reparametrized:**



Multi cost: $C_{ab}(i,j)=P_{ab}(i-1,j-1)+P_{ab}(i,j)-P_{ab}(i-1,j)-P_{ab}(i,j-1)$

Submodularity: $P_{ab}(\beta,\gamma)+P_{ab}(\alpha,\delta)-P_{ab}(\beta,\delta)-P_{ab}(\alpha,\gamma)\geq 0$ for $\beta>\alpha$ & $\delta>\gamma$

7. Background & Tracking

Expectation Maximization:

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

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

$q_i(h_i)=\underset{q_i(h_i)}{\text{argmax}}[B[\{q_i(h_i)\},\Theta^{t-1}]]=P(x_i|h_i,\Theta)=\frac{P(x_i|h_i,\Theta)P(h_i|\Theta)}{P(x_i)}$

M-step: Max lower bound w.r.t. Θ

$\hat{\Theta}=\underset{\Theta}{\text{argmax}}[B[\{q_i(h_i)\},\Theta]]=\underset{\Theta}{\text{argmax}}[\sum_{i=1}^{I}\log[\int P(x_i,h_i|\Theta)dh_i]]$

Mixture of Gaussians: $P(x|\Theta)=\sum_{k=1}^{K}\lambda_k \text{N}_x[\mu_k,\Sigma_k]$

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

M-step: $\hat{\Theta}=\text{argmax}[\sum_{i=1}^{I}\sum_{k=1}^{K}r_{ik}\log[\lambda_k \text{N}_{x_i}[\mu_k,\Sigma_k]]]$

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

Dual: $f(\lambda_1,\ldots,\lambda_K,\lambda)=\sum_{i=1}^{I}\sum_{k=1}^{K}r_{ik}\log[\lambda_k \text{N}_{x_i}[\mu_k,\Sigma_k]]+\lambda(\sum_k\lambda_k-1)$

Mean Shift Tracking

Weights: $w_i=\sum_{u=1}^{m}\delta[b(x_i)-u]\sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{y}_0)}}$

Mean shift: $\hat{y}_1=\frac{\sum_{i=1}^{n_h}x_i w_i g(\|\frac{\hat{y}_0-x_i}{h}\|^2)}{\sum_{i=1}^{n_h}w_i g(\|\frac{\hat{y}_0-x_i}{h}\|^2)}$

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

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. Temporal Filtering

Bayesian Filtering:

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

Filtering: $P(x_t|z_{1\ldots t})=\frac{P(z_t|x_t)P(x_t|z_{1..t-1})}{\int P(z_t|x_t)P(x_t|z_{1\ldots t-1})dx_t}$

Kalman Filter:

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

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

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

Filtering: $x_t=x_t^p+K(z_t-\mu_m-\Phi x_t^p)$
$\Sigma_t=(1-K\Phi)\Sigma_t^p$
$K=\Sigma_t^p\Phi^T(\Sigma_m+\Phi\Sigma_t^p\Phi^T)^{-1}$

Extended Kalman Filter:

Linearization: $\Psi=\frac{\partial f[x_t,\varepsilon_p]}{\partial x_t}$ $\Upsilon_p=\frac{\partial f[x_t,\varepsilon_p]}{\partial\varepsilon_p}$ $\Phi=\frac{\partial g[x_t,\varepsilon_m]}{\partial x_t}$ $\Upsilon_m=\frac{\partial g[x_t,\varepsilon_m]}{\partial\varepsilon_m}$

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

Filtering: $x_t=x_t^p+K(z_t-g[x_t^p,0])$
$\Sigma_t=(1-K\Phi)\Sigma_t^p$
$K=\Sigma_t^p\Phi^T(\Upsilon_m\Sigma_m\Upsilon_m^T+\Phi\Sigma_t^p\Phi^T)^{-1}$

Iterated: iterate linearization & filtering for better measurement incorporation

Unscented Kalman Filter

Sigma Sampling: $\hat{x}_0=\mu_t$ $a_j=\frac{1-a_0}{2D_w}$
$\hat{x}_j=\mu_t\pm\sqrt{\frac{2D_w}{1-a_0}}\Sigma^{1/2}e_j$

Prediction: $x_t^p=\sum_{i=0}^{2D_w}a_i\hat{x}_i^p$
$\hat{x}_j^p=f[\hat{x}_j,0]$ $\Sigma_t^p=\sum_{i=0}^{2D_w}a_i(\hat{x}_i^p-x_t^p)(\hat{x}_i^p-x_t^p)^T$

Filtering: $\mu_z=\sum_{i=0}^{2D_w}a_i\hat{z}_j$
$\hat{z}_j=g[\hat{x}_i^p,0]$ $\Sigma_z=\sum_{i=0}^{2D_w}a_i(\hat{z}_j-\mu_z)(\hat{z}_j-\mu_z)^T$
$x_t=x_t^p+K(z_t-\mu_z)$
$\Sigma_t=\Sigma_z-K\Sigma_t^pK^T$
$K=(\sum_{i=0}^{2D_w}a_i(\hat{z}_i-\mu_z)(\hat{x}_i^p-x_t^p)^T)\Sigma_z^{-1}$

Temporal Models:

Brownian Motion: $x_{t+1}=x_t+\varepsilon$

Constant Velocity: $x_{t+1}=v+x_t+\varepsilon$

Constant Rotation: $x_{t+1}=Rx_t+\varepsilon$

Matrix inv: $(A^{-1}+B^TC^{-1}B)^{-1}=A-AB^T(BAB^T+C)^{-1}BA$

Fixed Lag Smoother:

Temp. ev.: $\begin{bmatrix}x_t\\ \vdots\\ x_t^{\tau}\end{bmatrix}=\begin{bmatrix}\Psi & & \\ 1 & 0 & \\ & \ddots & \\ & 1 & 0\end{bmatrix}\begin{bmatrix}x_{t-1}\\ \vdots\\ x_t^{\tau}\end{bmatrix}+\begin{bmatrix}\varepsilon_p\\ \vdots\\ 0\end{bmatrix}$

Meas. eq.: $z_t=[\Phi\ 0\ldots 0]\begin{bmatrix}x_t\\ \vdots\\ x_t^{\tau}\end{bmatrix}+\varepsilon_m$

Fixed Interval Smoother:
$x_{t|T}=x_t+C_t(x_{t+1|T}-x_t^p)$
$\Sigma_{t|T}=\Sigma_t+C_t(x_{t+1|T}-x_t^p)C_t^T$
$C_t=\Sigma_t\Psi^T\Sigma_t^{p-1}$

Particle Filter (Importance sampling)
1. Prediction: evolve particles
2. Update: weight particles acc. to the measurement model
3. Resampling: Resample particles using rejection sampling

Learning Model Parameters $\Theta=\{\mu_0,\Sigma_0,\Psi,\Sigma_p,\Phi,\Sigma_m\}$

E-step: Filtering via fixed interval smoother

M-step: Max. log likelihood w.r.t Θ

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

$\Psi=\frac{\sum_{t=2}^{T}x_tx_{t-1}^T}{\sum_{t=2}^{T}x_{t-1}x_{t-1}^T}$ $\Phi=\frac{\sum_{t=1}^{T}z_tx_t^T}{\sum_{t=1}^{T}x_tx_t^T}$

$\Sigma_p=\frac{1}{T-1}\sum_{t=2}^{T}(x_tx_t^T-\Psi x_{t-1}x_t^T)$

$\Sigma_m=\frac{1}{T}\sum_{t=1}^{T}(z_tz_t^T-\Phi x_tz_t^T)$

9. Shapes

Probabilistic: $\hat{w} = \underset{w}{\operatorname{argmax}}[P(w|x)]$
$= \underset{w}{\operatorname{argmax}}[\log P(x|w) + \log P(w)]$

Shape Template Model: Map landmark points to image via transform

Likelihood: $P(x|w,\psi) \propto \prod_{n=1}^{N} \exp[-(\text{dist}[x, \text{trans}[w_n, \psi]])^2]$

ML: $\hat{\psi} = \underset{\psi}{\operatorname{argmax}}[\sum_{n=1}^{N} -(\text{dist}[x_i, \text{trans}[w_n, \psi]])^2]$

Derivative: $\frac{\partial L}{\partial \psi} = -\sum_{n=1}^{N}\sum_{j=1}^{2} \frac{\partial \text{dist}[x, w_n]^2}{\partial w'_{jn}} \frac{\partial w'_{jn}}{\partial \psi}$

$\frac{\partial}{\partial w'_{jn}} \frac{1}{2}\text{dist}[x, w'_n]^2 \approx D(w'_n)\frac{1}{2}(D(w'_n + e_j) - D(w'_n - e_j))$

Statistical Shape Model: shape template with prior to allow elasticity

Prior: $P(w) = N_w[\mu, \Sigma]$

Learning: 0. Align examples with initial mean shape
Iterate until convergence:
1. Solve affine transform to mean shape
2. Apply separate transform to each shape
3. Recompute mean and var. for each point

MAP Inference: $\hat{w} = \underset{w}{\operatorname{argmax}}[\underset{\psi}{\max}[\log P(x|w,\psi) + \log P(w)]]$

Fitting affine transformation:

$$\begin{bmatrix} x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} x'_i \\ y'_i \end{bmatrix}$$

Pseudo inv: $x = (A^T A)^{-1} A^T b$

SVD: $b' = U^T b \quad y_i = b'_i / D_{ii} \quad x = Vy$

Stastical Models for Shape and Appearance

$P(h_i) = N_{h_i}[0, I] \quad P(w_i|h_i) = N_{w_i}[\mu_w + \Phi_w h_i, \sigma_w^2 I]$

$P(x_i|w_i, h_i) = N_{x_i}[\text{warp}[\mu_x + \Phi_x h_i, w_i, \psi], \sigma_x^2 I]$

Subspace Shape Models: Represent the shape through a linear comb. of latent variables

Model: $w_i = \mu + \Phi h_i + \varepsilon_i$

Prob.: $P(w_i|h_i, \mu, \Phi, \sigma^2) = N_{w_i}[\mu + \Phi h_i, \sigma^2 I]$

Prior: $P(h_i) = N_{h_i}[0, I]$ (only PPCA)

PCA: 0. Align examples with initial mean shape
1. Perform PCA on examples:
$\text{var}(v) = \sum_i \|(x_i - \bar{x})^T v\| = v(x-\bar{x})(x-\bar{x})^T v = vAv$
$AA^T = U\Sigma V^T V \Sigma U^T = UL^2U^T$
2. Take k largest eigenvectors as basis

Inference:
0. Initialize shape parameters b to 0
Iterate until convergence:
1. Compute model points: $x = \mu + Pb$
2. Solve for affine transform parameters (X_t, Y_t, s, θ) producing best alignment to Y
3. Project observed points Y to model: $y = T^{-1}(Y)$
4. (opt.) Project y into tangent space: $y' = y/(y^T\mu)$
5. Update model parameters: $b = P^T(y' - \mu)$

PPCA: 1. Learn μ, Φ and σ^2 via PCA method
2. Subtract mean and do eigen-decomposition
$W = [w_1 - \mu, \ldots, w_I - \mu] \quad WW^T = UL^2U^T$
3. Set parameters: $\hat{\sigma}^2 = \frac{1}{D-K}\sum_{j=K+1}^{D} L_{jj}^2$
$\Phi = U_K(L_K^2 - \sigma^2 I)^{1/2}$

Inference: iterate between:
1. Update weightings:
$\hat{h} = \underset{h}{\operatorname{argmax}}\left[\sum_{n=1}^{N} -(y_n - \text{trans}[\mu_n + \Phi_n h, \psi])^2/\sigma^2 - \log[h^T h]\right]$
2. Update the transformation parameters

10. Image Features

Alignment: 1. Extract features
2. Match features
3. Filter matches
4. Compute transf.

Harris Corner Detector:
1. Compute second moment matrix
2. Find points with corner response > t
3. Non-maxima suppression

Second Moment Matrix: $M = \sum w(x,y)\begin{bmatrix} I_x I_x & I_x I_y \\ I_y I_x & I_y I_y \end{bmatrix} = X\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_1 \end{bmatrix}X^T$

Interest Point: $\lambda_1 \gg 0 \quad \lambda_2 \gg 0$

Corner Response: $f = |M| - k(\text{Tr}(M))^2 = \lambda_1\lambda_1 - k(\lambda_1 + \lambda_2)^2 \quad k = 0.04$

Blob: Superposition of two Laplacians

Scale Invariant Feature Points:
1. Convolve with Laplacian = difference of gaussian
2. Reject points with low contrast: $\frac{\text{Tr}(H)^2}{\det(H)} < \frac{(r+1)^2}{r}$

SIFT Descriptor: 1. Extract gradients in neighborhood depending on scale
2. Sort gradients in 36 bins and determine dominant one
3. Extract 16 4x4 sub-blocks in a 16x16 window and sort in norm. 8 bin histogram
4. Subtract dominant orientation, cut-off over 0.2 and normalize

11. Image Alignment

RANSAC
Iterate until condition:
1. Randomly select a seed group
2. Compute the transform on the group
3. Find inliers to this transform
4. Find group with most inliers
5. Recompute transform on the group

Efficiency: N: #points, I: #inliers, m: #sample

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

#samples until hitting a good one w/ p=.95
$k \geq \log(1-p)/\log(1-P(\text{good}))$

Solving Homography

$$\begin{bmatrix} x_i & y_i & 1 & 0 & 0 & 0 & x_i x'_i & x'_i y_i & x'_i \\ 0 & 0 & 0 & x_i & y_i & 1 & y'_i x_i & y'_i y_i & y'_i \\ & & & & \vdots & & & & \end{bmatrix} \begin{bmatrix} h_{00} \\ \vdots \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

12. Optical Flow

Optical flow: $V_x = \frac{dx}{dt} \quad V_y = \frac{dy}{dt}$

Brightness Constancy: $I(x, y, t-1) = I(x + u(x,y), y + v(x,y), t)$
$\approx I(x,y,t) + I_x u(x,y) + I_y v(x,y)$

Impl.: $I_x u + I_y v + I_t = 0$

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

Impl.: $I_{xx}u + I_{yx}v + I_{tx} = 0 \ \wedge \ I_{xy}u + I_{yy}v + I_{ty} = 0$

Spatial Coherence: $\begin{bmatrix} P_x(1) & P_y(1) \\ \vdots & \vdots \\ P_x(25) & P_y(25) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} P_t(1) \\ \vdots \\ P_t(25) \end{bmatrix}$

Aperture problem: Brightness constancy underconstrained → predict only normal flow perpendicular to br. const. line

AAE $= \frac{1}{n}\sum_{i=0}^{n} \cos^{-1}\left(\frac{u_i u_i^c + v_i v_i^c + 1}{\sqrt{(u_i^2 + v_i^2 + 1)(u_i^{c2} + v_i^{c2} + 1)}}\right)$

AEE $= \frac{1}{n}\sum_{i=0}^{n} (u_i - u_i^c)^2 + (v_i - v_i^c)^2$

SIFT Flow: Match SIFT features and compute optical flow between matches

Motion Segm.: 1. Compute optical flow
2. Compute segmentation based on this

Motion Comp.: Intertwined estimation of opt. flow & segm.
$E(u, v, C) = \sum_{i=1}^{n} \int_{\Omega}(I_x u_i + I_y v_i + I_t)^2 dx + v|C|$

Lucas Kanade: Solve spatial coherence
1. Compute gradient image
For each pixel:
2. Compute second moment matrix in neighborhood:

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

3. Check, if eigenvalues of $A^T A$ are sufficiently large ⇒ corner ⇒ rank(2) ⇒ solvable
4. Solve: $\begin{bmatrix} u \\ v \end{bmatrix} = (A^T A)^{-1} A^T b$

Shi-Tomasi: Solve with Lucas Kanade and track features of corners to refine and check consistency

Horn Schunck: Minimize energy function on constraints

Assumptions: I: $I_x u + I_y v + I_t = 0$ II: $|\nabla I_x|^2 + |\nabla I_y|^2 \to \min$

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

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

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

Gradient Desc.: $u^{t+1} = u^t + \tau(\Delta u - \frac{1}{\alpha}(I_x u + I_y v + I_t)I_x)$

Energy: $E(u,v) = \int \Psi((I_x u + I_y v + I_t)^2)dx$ | robust func. against occlusions
$+ \gamma \int \Psi((I_{xx}u + I_{xy}v + I_{xt})^2 + (I_{xy}u + I_{yy}v + I_{yt})^2)dx$ | grad. const. against illum. changes
$+ \alpha \int \Psi(|\nabla u_x|^2 + |\nabla v_x|^2)dx$ | robust func. against discont.
$+ \beta \int \rho(x)\Psi((u(x) - u'(x))^2 + (v(x) - v'(x))^2)dx$ | correspondences

$\Psi(s^2) = \sqrt{s^2 + \varepsilon^2}$

$\Psi'(s^2) = \frac{1}{\sqrt{s^2 + \varepsilon^2}}$

13. Calibration

Projection: $(x,y,z) \to (f\frac{x}{z}, f\frac{y}{z}, f)$

Proj. Pipeline: World to camera
↓
projection
↓
camera to pixel
↓
distortion

Thin lens formula: $\frac{1}{D}+\frac{1}{D'}=\frac{1}{f}$

Depth of Field: depth range in focus
→ change through aperture size

FoV: $\varphi = \tan^{-1}\left(\frac{d}{2f}\right)$

Pinhole Model: $K = \begin{bmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
p: principle point
m: meters per pixel
f: focal length

Extrinsic: $X_{cam} = \begin{bmatrix} R & -RC \\ 0 & 1 \end{bmatrix}\begin{bmatrix} X \\ 1 \end{bmatrix}$

Non-linear calib.: Init linear → non-linear
$$\sum_{i=1}^{n}\sum_{j=1}^{m} \| m_{ij} - \pi(A, k_1, k_2, R_i, t_i, M_j) \|^2$$
→ Levenberg-Marquardt

Polynomial Distortion: $x_d = L(r)x \quad L(r) = 1 + k_1 r + k_2 r^2 + \dots$

Rad-tan Distortion: $x_d = (1 + kc(1)r^2 + kc(2)r^4 + kc(5)r^6)x + dx \quad dx = \begin{bmatrix} 2kc(3)xy + kc(4)(r^2+2x^2) \\ 2kc(4)xy + kc(3)(r^2+2y^2) \end{bmatrix}$

Linear Camera Calibration: Solve $x = PX \iff x \times PX = 0$:
→ 2 indep. eq. per point
$$\begin{bmatrix} 0 & X_1 & -y_1X_1 \\ X_1 & 0 & -x_1X_1 \\ & \vdots & \\ 0 & X_n & -y_nX_n \\ X_n & 0 & -x_nX_n \end{bmatrix}\begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

Plane Calibration: Calib. from homography between plane and image plane

1. Establish relation bet. homography and rotation
$[h_1 h_2 h_3] = \lambda A [r_1\ r_2\ t] \quad h_1 A^{-T}A^{-1}h_2 = 0 \quad h_1 A^{-T}A^{-1}h_1 = h_2 A^{-T}A^{-1}h_2$
2. Rewrite matrix and build linear equation system
$B = A^{-T}A^{-1} \quad \begin{bmatrix} v_{12}^T \\ (v_{11}-v_{22})^T \end{bmatrix} b = 0 \quad v_{ij} = [h_{i1}h_{j1}, h_{i1}h_{j2}+h_{i2}h_{j1}, h_{i2}h_{j2}, h_{i1}h_{j3}+h_{i3}h_{j1}, h_{i2}h_{j3}+h_{i3}h_{j2}, h_{i3}h_{j3}]^T$
3. Solve for the parameters
$v_0 = (B_{12}B_{13} - B_{11}B_{33})/(B_{11}B_{22} - B_{12}^2)$
$\lambda = B_{33} - (B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23}))/B_{11}$
$\alpha = \sqrt{\lambda / B_{11}}$
$\beta = \sqrt{\lambda B_{11}/(B_{11}B_{22} - B_{12}^2)}$
$c = -B_{12}\alpha^2\beta/\lambda$
$u_0 = cv_0/\alpha - B_{13}\alpha^2/\lambda$
$r_1 = \lambda A^{-1}h_1$
$r_2 = \lambda A^{-1}h_2$
$r_3 = r_1 \times r_2$
$t = \lambda A^{-1}h_3$
$\lambda = 1/\|A^{-1}h_1\|$

Distortion Calibration

Model: $\hat{u} = u + (u-u_0)(k_1(x^2+y^2) + k_2(x^2+y^2)^2)$ (same for $\hat{v}$)

Linear system: $\begin{bmatrix} (u-u_0)(x^2+y^2) & (u-u_0)(x^2+y^2)^2 \\ (v-v_0)(x^2+y^2) & (v-v_0)(x^2+y^2)^2 \end{bmatrix}\begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix}$
→ pseudo inv.

Epipolar Geometry:
Baseline: line bet. camera centers
Epipole: intersection baseline & image plane
Epipolar line: proj. of camera ray from image A to B
Epipolar plane: plane of baseline & epipolar line
Essential Matrix. $E = [t]_\times R \quad l' = Ex$
→ transform bet. images, singular, rank=2, 5 DoF
Fundamental Matrix: $F = K^{-T}EK^{-1} \quad l' = Fx$
→ transform bet. uncalib. images, singular, rank=2, 7 DoF
Epipolar Constraint: $xEx' = 0$

Eight-Point Alg.: Solve the epipolar constraint for 8 points
$$\begin{bmatrix} u_iu_i' & u_iv_i' & u_i & v_iv_i' & v_iu_i' & v_i & u_i' & v_i' \\ & & & \vdots & & & & \end{bmatrix}\begin{bmatrix} F_{11} \\ \vdots \\ F_{33} \end{bmatrix} = -\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

Normalized Eight-Point: Numerical instabilities due to diff. magnitudes
1. Center at origin & scale s.t. mean squared diff. is 2 pixels at max
2. Apply eight-point algorithm
3. Enforce rank 2 through SVD
4. Scale back to original units

14. Stereo

Basic Stereo Matching:
For each pixel in image A:
1. Find epipolar line in image B
2. Match descriptor to point on that line
3. Compute disparity $x-x'$ and depth $= \frac{B \cdot f}{x-x'}$

Depth from disp: $\frac{z}{B} = \frac{z-f}{B+x'-x} \iff x - x' = \frac{B \cdot f}{z}$

Energy min.: $E(D) = \sum_i (W_1(i) - W_2(i+D(i)))^2 + \lambda \sum_{ij} \rho(D(i)-D(j))$

Photo consist.: Scenes that produce the same image
Photo Hull: Union of all photo consistent scenes
Visual Hull: comb. of silhouettes from all views

Plane Sweep Stereo
0. Choose reference frame
1. Warp images at discretized depths to the reference frame
2. For each pixel choose depth with least variance

Feature Matching: Start with initial matches and iteratively expand to nearby points using visibility const.

Stereo Image Rectification: Solve for homography which can be split in 3 separate transforms
1. Projective transform: make epipolar lines parallel: $H_p = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ w_a & w_b & 1 \end{bmatrix}$
→ enforce same focal length ⇔ homo. coord.
$\frac{w^TPP^Tw}{w^TP_cP_c w} \iff \frac{z^T\overbrace{e_\times^T PP^T e_\times}^{A} z}{z^Te_\times^T P P^T e_\times z} + \frac{z^TF^Tp'p'^TFz}{z^TF^TP_c'P_c'^TFz}$ w/ $w = [e_\times]z$, $w' = Fz$ → $A = D^TD$ → y highest λ of $D^{-T}BD^{-1}$ → $z = D^{-1}y$
2. Similarity transform: align epipolar lines $H_r = \begin{bmatrix} F_{32}-w_bF_{33} & w_aF_{33}-F_{31} & 0 \\ F_{31}-w_aF_{33} & F_{32}-w_bF_{33} & F_{33}+v_c' \\ 0 & 0 & 1 \end{bmatrix}$
3. Shear transform: map images to common image size $H_s = \begin{bmatrix} a & b & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
→ enforce perpendicular image axes of unit length
$a = \frac{h^2x_v^2 + w^2y_v^2}{hw(x_vy_u - x_uy_v)} \quad b = \frac{h^2x_ux_v + w^2y_uy_v}{hw(x_uy_v - x_uy_v)}$

Volumetric Stereo: Choose volumetric repr. and assign RGB values to photo-consistent voxels
Space Carving: Iteratively choose outside voxels and carve if proj. not photo-cons.
Reconst. from Silhouettes: A voxel is photo-consist. if it is in the object's silhouette in all views
Carved Visual Hulls.
1. Compute visual hull
2. DP to find rims and constrain them to be fixed
3. Carve visual hull to optimize photo-consistency

15. Structure from Motion

Ambiguity: we can only estimate the scene up to a transform Q: $x = PX = (PQ)(Q^{-1}X)$
projective: 15 DoF **Similarity:** 7 DoF
affine: 12 DoF **Euclidean:** 6 DoF

Bundle Adjustment: Optimize 3D points and camera pos. by min the reprojection error
$$E(P,X) = \sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij}\|x_{ij} - \pi(P_i, X_j)\|^2$$

LM: sparse hessian
Jacobian: $\begin{bmatrix} A^TA & A^TB \\ B^TA & B^TB \end{bmatrix}\begin{bmatrix} \delta_A \\ \delta_B \end{bmatrix} = \begin{bmatrix} A^T\varepsilon \\ B^T\varepsilon \end{bmatrix}$
A: camera
B: points

Levenberg-Marquardt: Dampened Gauss-Newton
$x^{t+1} = x^t + (J^TJ + \mu I)^{-1}J^T\varepsilon$

Non-rigid Sfm: ambiguity bet. motion & shape → separate shape in each view
$\mathrm{Rank}(W) \le \min(2F, P)$

Sfm Approach: Decompose measurement matrix into a motion and shape matrix: $W^{2m\times n} = M^{2m\times 3}S^{3\times n}$

Affine Sfm: $2mn \ge 8m + 3n - 12$
1. Center image points: $\hat{x}_{ij} = x_{ij} - \frac{1}{n}\sum_{k=1}^{n} x_{ik}$
2. Build measurement matrix: $D = \begin{bmatrix} \hat{x}_{11} & \cdots & \hat{x}_{1n} \\ \vdots & & \vdots \\ \hat{x}_{m1} & \cdots & \hat{x}_{mn} \end{bmatrix} = \begin{bmatrix} A_1 \\ \vdots \\ A_m \end{bmatrix}[X_1 \dots X_n]$
3. Perform SVD and take 3 largest singular values: $M = U_3W_3^{1/2} \quad S = W_3^{1/2}V_3^T$
4. Eliminate affine ambiguity C by enforcing euclidean constraints
→ perpendicular axes of unit length
$m_{i1}CC^Tm_{i1} = 1$
$m_{i2}CC^Tm_{i2} = 1$ → Solve for C using Newton or linearly with SVD/Cholesky → $M' = MC$, $S' = C^{-1}S$
$m_{i1}CC^Tm_{i2} = 0$

Projective Sfm: $2mn \ge 11m + 3n - 15$

Sequential Sfm:
0. Initialize structure from two views through triangulation using the fund. mat.
For each additional view:
1. Determine projection matrix using the visible 3D points
2. Compute new 3D points through triangulation using new camera

Linear Shape Model: linear comb. of basis shapes: $W = R\Omega B$
1. Outer SVD: $W = U\Sigma V^T = (U\Sigma^{1/2})(\Sigma^{1/2}V^T) = \hat{H}\hat{B}$
H: comb. of motion & shape weights $[H]_{ij} = w_{ij}R_i$
B: shape basis
2. Inner SVD: Decompose $\hat{H}$ in motion & shape weights.
$\hat{H}' = U\Sigma V^T = \hat{W}\hat{R}$
→ stack comp. belonging to a single view in a row
3. Metric rectify using orthonormality constraints

Linear Trajectory Model: linear comb. of traj. over time: $W = R\Theta A$

Non-rigid Duality: Shape weights are traj. basis and vice versa

16. Pose

Transform Matrix: $M = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$

Euler Angles: $(\theta_x, \theta_y, \theta_z)$ rotation in pre-defined axes order

Gimbal Lock: Axes of rotation align resulting in a temporary loss of a DoF

Axis Angle: $\theta, (\omega_x, \omega_y, \omega_z)$ rotation axis + magnitude

Twists: $\theta \hat{\xi} = \theta \begin{pmatrix} \hat{\omega} & v \\ 0 & 0 \end{pmatrix}$ axis angle + translation along axis

$$\exp(\theta \hat{\xi}) = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$$

Quaternions: $(q_1, q_2, q_3, q_4) = [q_w \; v]$

$$q_w = \cos\frac{\theta}{2} \quad v = \sin\frac{\theta}{2}\,\omega$$

Concat: $q_1 \circ q_2 = (q_{w1} q_{w2} - v_1 v_2,\; q_{w1} v_2 + q_{w2} v_1 + v_1 \times v_2)$

Rotate: $q \circ a \circ \bar{q}$ **Compl.:** $\bar{q} = (q_w - v)$

Plücker line: $L = (n, m)$ Point on line. $m = x \times n$

n: direction

m: momentum

Shortest dist.: $\|\Pi(X) \times n - m\|$

Euler → Matrix: $R = R_x R_y R_z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_x & s_x \\ 0 & s_x & c_x \end{bmatrix} \begin{bmatrix} c_y & 0 & -s_y \\ 0 & 1 & 0 \\ -s_y & 0 & c_y \end{bmatrix} \begin{bmatrix} c_z & s_z & 0 \\ -s_z & c_z & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Axis → Matrix: $R = \exp(\hat{\omega}\theta) \approx I + \hat{\omega} \sin\theta + \hat{\omega}^2 (1 - \cos\theta)$ $\hat{\omega} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}$

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

Mean rotation: $\frac{1}{2} \sum_i \pi_i \|\bar{r}^{-1} * r_i\|_2^2 \rightarrow \min$ $\hat{r}_{t+1} = \hat{r}_t * \left(\frac{\sum_i \pi_i (\hat{r}_t^{-1} * r_i)}{\sum_i \pi_i}\right)$

$$r_2 * r_1 = \log(\exp(r_2) \exp(r_1))$$

$$r_1^{-1} = \log(\exp(r_1)^T)$$

Mesh Fitting: $\underset{\theta, \hat{\xi}}{\operatorname{argmin}} \frac{1}{2} \sum_i \|\pi((I + \theta \hat{\xi}) X_i) \times n_i - m_i\|^2$

Linear system: $\begin{pmatrix} A_1 \\ \vdots \\ A_k \end{pmatrix} x^T = \begin{pmatrix} b_1 \\ \vdots \\ b_k \end{pmatrix}$ $A_i = \begin{pmatrix} 0 & n_{i3} & -n_{i2} & -X_{i3} n_{i3} - X_{i2} n_{i2} & X_{i1} n_{i3} & X_{i1} n_{i2} \\ -n_{i3} & 0 & n_{i1} & X_{i3} n_{i1} & -X_{i1} n_{i1} - X_{i3} n_{i3} & X_{i2} n_{i3} \\ n_{i2} & -n_{i1} & 0 & X_{i2} n_{i1} & X_{i3} n_{i2} & -X_{i2} n_{i2} - X_{i1} n_{i1} \end{pmatrix}$

$$b_i = \begin{pmatrix} X_{i3} n_{i2} - X_{i2} n_{i3} - m_{i1} \\ X_{i1} n_{i3} - X_{i3} n_{i1} - m_{i2} \\ X_{i2} n_{i1} - X_{i1} n_{i2} - m_{i3} \end{pmatrix}$$