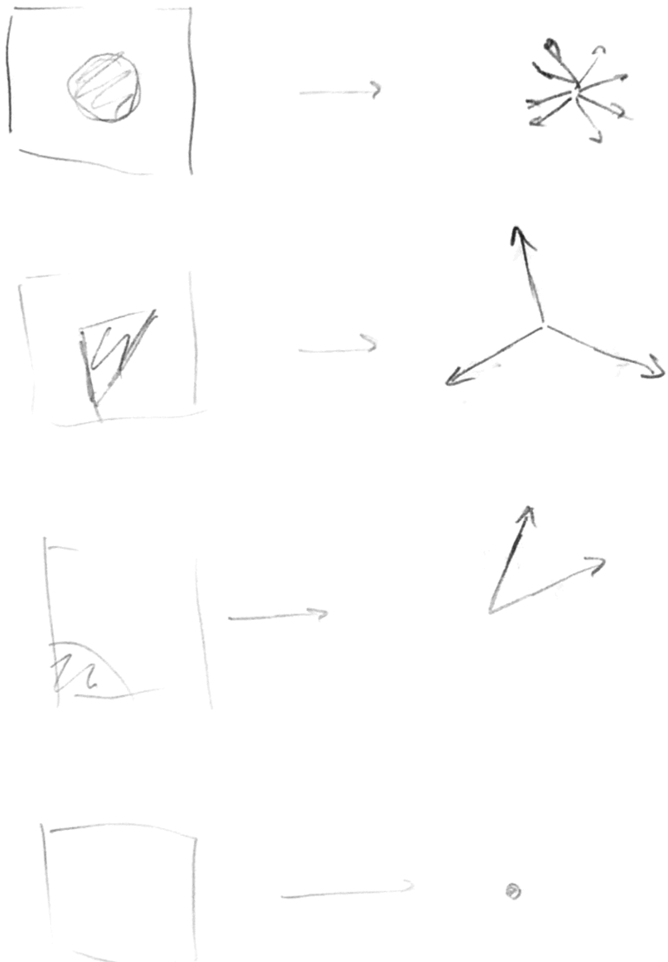
1a) The SIFT-like descriptor consists of four gradient histograms that look like this:



Each histogram is stored as 8 numbers →
$4 \times 8 = 32$-vector. Finally this vector is nomalized.

1(b) A: same orientations but upper right part has shorter gradients.

B: Same orientations but all gradients are slightly shorter. After normalization almost identical.

C: Same histograms but in different order. ⇒ very diffent descriptor.

Concl: B is probably most similar (after normalization)

NOTE: Some of you got a poor-quality copy of the exam. I will take this into account when marking!

(C) Too few regions: Poor descriptive power, many diffent patches get similar descriptors.

Too many regions: Poor robustness, Sensitive to noise.

2 (a) Update rule: Choose random data point (x_i, y_i) and update parameters as

$$\theta^{(k+1)} = \theta^{(k)} - \mu \nabla \mathcal{L}_i(\theta)$$

In this case:

$$\nabla \mathcal{L}_i = \frac{dL_i}{dp_i} \cdot \frac{dp_i}{ds_i} \cdot \begin{pmatrix} \frac{\partial s_i}{\partial w_1} \\ \frac{\partial s_i}{\partial w_2} \end{pmatrix}$$

---

where $\frac{dL_i}{dp_i} = \begin{cases} -\frac{1}{p_i} & \text{for positive} \\ \frac{1}{1-p_i} & \text{for negative.} \end{cases}$

$$\frac{dp_i}{ds_i} = \frac{e^{-s}}{(1+e^{-s})^2} = p_i(1-p_i)$$

$$\frac{\partial s_i}{\partial w_1} = x_i, \quad \frac{\partial s_i}{\partial w_2} = ye^{w_2 y_i}$$

$$\theta^{(k+1)} = \begin{cases} \theta^{(k)} + \mu(1-p_i) \begin{pmatrix} x_i \\ ye^{w_2 y_i} \end{pmatrix} & \text{if pos.} \\ \theta^{(k)} - \mu p_i \begin{pmatrix} x_i \\ y_i e^{w_2 y_i} \end{pmatrix} & \text{if neg.} \end{cases}$$

2b) # weights =

$$\underbrace{3 \cdot 3 \cdot 10 + 10}_{\text{conv}} + \underbrace{0}_{\text{relu}} + \underbrace{0}_{\text{max pooling}} +$$

$$+ \underbrace{3 \cdot 3 \cdot 10 \cdot 20 + 20}_{\text{conv}} + \underbrace{0}_{\text{relu}} = 100 + 1800 + 20$$

$$= 1920$$

2a) $100 \times 100 = 10000$ inputs

$50 \times 50 \times 20 = 50000$ outputs

so each output $y_k = \sum_{i=1}^{n} w_i x_i + w_0$

with 10001 weights

In all $50000 \cdot 10001$ weights =

$$\boxed{500\,050\,000}$$

3(a) We want a transformation from target to source.

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} \theta_1 & \theta_2 \\ \theta_4 & \theta_5 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \theta_3 \\ \theta_6 \end{pmatrix} \qquad (*)$$

The minimal case is three measurements $(\check{x}_i, \check{y}_i) \leftrightarrow (x_i, y_i)$. Assuming $i = 1, 2, 3$.

Rearranging $(*)$ for $i = 1, 2, 3 \Rightarrow$

$$\underbrace{\begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \end{pmatrix}}_{M} \Theta = \underbrace{\begin{pmatrix} \check{x}_1 \\ \check{y}_1 \\ \check{x}_2 \\ \hat{y}_2 \\ \hat{x}_3 \\ \hat{y}_3 \end{pmatrix}}_{b}$$

Solved in Matlab as $\Theta = M \backslash b$

(b) The chance of getting a good solution in a Ransac iteration is 0.5^3 for affine and 0.5^2 for similarity $\Rightarrow$

On average we need twice as many iterations for a good affine transformation.

Each iteration takes approx as much time

so: $\boxed{\text{Twice as long}}$

4 a) We have 3 unknowns in $U = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$

Camera equation

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = P \begin{pmatrix} U \\ 1 \end{pmatrix} = \begin{pmatrix} \leftarrow a \rightarrow \\ \leftarrow b \rightarrow \\ \leftarrow c \rightarrow \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (*)$$

Each camera/view $\Rightarrow$ 3 equations and 1 extra unknown. We need two views, we use ~ (tilde) for the second view

Rearranging (*) as

$$\underbrace{\begin{pmatrix} a_1 & a_2 & a_3 & -x & 0 \\ b_1 & b_2 & b_3 & -y & 0 \\ c_1 & c_2 & c_3 & -1 & 0 \\ \cancel{\tilde{a}_1} & \cancel{\tilde{a}_2} & \cancel{\tilde{a}_3} & \cancel{0} & \cancel{-\tilde{x}} \\ \tilde{b}_1 & \tilde{b}_2 & \tilde{b}_3 & 0 & -\tilde{y} \\ \tilde{c}_1 & \tilde{c}_2 & \tilde{c}_3 & 0 & -1 \end{pmatrix}}_{M} \underbrace{\begin{pmatrix} X \\ Y \\ Z \\ \lambda \\ \tilde{\lambda} \end{pmatrix}}_{\theta} = \underbrace{\begin{pmatrix} -a_4 \\ -b_4 \\ -c_4 \\ \cancel{-\tilde{a}_4} \\ -\tilde{b}_4 \\ -\tilde{c}_4 \end{pmatrix}}_{r}$$

We remove a random row and solve using

$\theta = M | r$

4b) Camera eq: $\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = PU$

Third row $\Rightarrow$ $\lambda = -4 + 2 = -2$

so the point is behind the camera.

This is <u>not</u> consistent with

$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 42 \\ 156 \end{pmatrix}$

5a) A CNN can be trained to give the size as output. We could train a CNN to estimate the size given a detection from the existing detector. The structure can be similar to the detector but with a squared loss instead of neg. log-likelihood.

5b) A max at $\hat{u} = (x, y) = (4, 3)$

To get sub-pixel accuracy we est the derivatives at this point

$$f'_x \approx \frac{8-6}{2} = 1, \quad f'_y \approx \frac{10-6}{2} = 2$$

$$f''_{xx} \approx 8+6-2\cdot 12 = -10, \quad f''_{yy} \approx 10+6-24 = -8$$

$$f''_{xy} \approx \frac{1}{4}(8+6-4-4) = 1.5$$

Taylor exp about $(4,3)$

$$f(x, y) \approx 12 + \begin{pmatrix} 1 & 2 \end{pmatrix}(u - \hat{u}) +$$

$$+ \frac{1}{2}(u-\hat{u})^T \begin{pmatrix} -10 & 1.5 \\ 1.5 & -8 \end{pmatrix} (u-\hat{u})$$

5b (cont) At local max the partial der. should be zero so:

$$\boxed{\underbrace{\begin{pmatrix} -10 & 1.5 \\ 1.5 & -8 \end{pmatrix}}_{M} \underbrace{(u-\hat{u})}_{\theta} = \underbrace{\begin{pmatrix} -1 \\ -2 \end{pmatrix}}_{b}}$$

[Sub-pixel det: $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 3.14 \\ 4.28 \end{pmatrix}$]

6] Let $\hat{u}$ be the estimated 3D point

and $\mathcal{L}(u) = \sum_{\text{inliers}} r_i^2(u)$.

If there are "many" u s.t $\mathcal{L}(u) \approx \mathcal{L}(\hat{u})$ then we are uncertain.

Idea :

low certainty — high certainty.

Let $\Delta = u - \hat{u}$, $\bar{r} = \bar{r}(\hat{u})$ and $J = J(\hat{u})$.

$$\mathcal{L}(u) \approx (\bar{r} + J\Delta)^T(\bar{r} + J\Delta) =$$

$$= \bar{r}^T\bar{r} + 2\bar{r}^T J\Delta + \Delta^T J^T J\Delta$$

If Gauss-Newton has converged, then $\bar{r}^T J = 0$. so

$$\mathcal{L}(u) = \bar{r}^T\bar{r} + \Delta^T Q \Delta = \mathcal{L}(\hat{u}) + \Delta^T Q \Delta$$

Q here is symmetric and positive semidefinite

6 (cont) If the smallest eigenvalue of Q is zero or close to zero then there is a direction in which $\mathcal{L}(\alpha)$ increases slowly = large uncertainty.

If the smallest eigenvalue is large then all three* eigenvalues are large and $\mathcal{L}(\alpha)$ increases rapidly in all directions = = small uncertainty.

* $Q = J^T J$ is symmetric $\left[Q^T = J^T (J^T)^T = J^T J \right]$ and positive semi-definite $\left[x^T Q x = x^T J^T J x = (Jx)^T (Jx) = \| Jx \|^2 \geq 0 \right]$ so there is an ON-basis of eigenvectors with non-negative eigenvalues.