Exam in SSY097

June 5, 2018

Allowed materials: Pen/pencil, eraser.

The exam consists of six problems. Make sure that you have them all.

- Motivate all answers carefully.
- Use a new paper for each new numbered problem.
- Write on one side of the papers only.
- Write your anonymous number on each new page.
- Avoid using a red pen.
- If you want the result registered as SSY096, write this on the cover page.

Grades

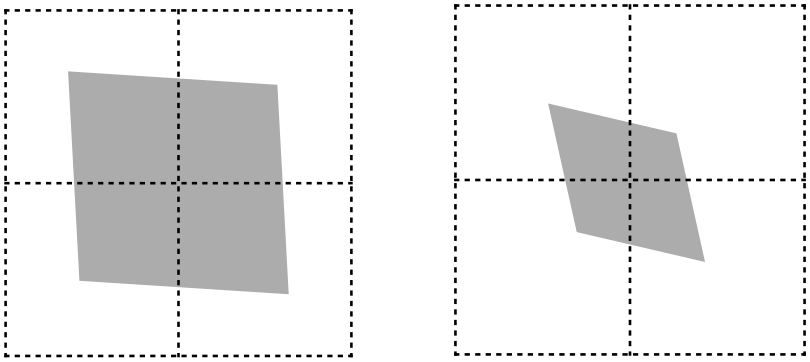
≥ 8 **points** Grade: 3

≥ 11 **points** Grade: 4

≥ 14 **points** Grade: 5

1 SIFT, 3 points

(a) A SIFT-like descriptor (as the one in Lab 1) was computed for the image patches, P, and Q, below. The regions used are indicated with grey dashed lines so these lines are not part of the actual image. Note that unlike for original SIFT only four regions are used. Illustrate (for example using vector bouquets) what the SIFT-like descriptors for P and Q will look like. It should be clear from your description how many elements there are in the descriptor vector.



(b) Consider performing Sift matching with the Lowe ratio criterion between two images. Assume that the descriptors from the first image are

$$d_1 = (2.8), \quad d_2 = (3.1), \quad d_3 = (11) \quad d_4 = (15) \tag{1}$$

and from the second

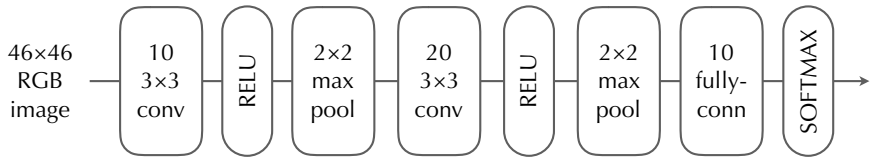
$$q_1 = (2.7), \quad q_2 = (3.0), \quad q_3 = (13) \tag{2}$$

Compute the matches and Lowe ratio for each d_i. (Note that the descriptors are one-dimensional rather than 128-dimensional. This means that you can compute the descriptor distance as $|q_i - d_k|$. Which match is most certain according to the Lowe ratio?

(c) Why do we use the Lowe ratio rather than the descriptor distance as a measure of certainty?

2 Statistical learning, 3 points

The network below could be used for classification with 10 output labels, (e.g., the digits 0-9). The output from the last softmax layer is 10 *probabilities* p_k. Naturally these values depend on the current input image I. We can emphasize this dependence by writing $p_k(I)$.



(a) At each step of stochastic gradient descent we randomly select a training example I_i. Write down the loss for a training example I_i with true label 5. As in the lab, use the negative log-likelihood loss.

(b) Write down the formula for stochastic gradient for the case in **(a)**. No credits for the general formula, you have to use your result from **(a)**.

(c) For each layer in the network above, write down the number of trainable weights (parameters). Assume that the conv-layers do not use padding.

(d) How could we modify the network above to make it fully-convolutional without changing the output for a 46×46 RGB image?

3 Image registration, 3 points

(a) An affine transformation used to register two 3D images can be parameterized as

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & j \end{pmatrix} \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \tag{3}$$

We want to use Ransac to estimate such a transformation. Explain how to construct the minimal solver used in Ransac by getting a system $M\theta = b$ and how to solve it in Matlab.

(b) Let K be the number of measurements used by the minimal solver in **(a)**. Assume that after a Ransac iteration we have K inliers. How much can we trust this solution? Motivate as well as you can. Do not just give an answer!

(c) Assuming that we have 50% outliers, on average how much longer does it take to find a good affine transformation in 3D compared to 2D (as in the lab). You can assume that running the minimal solver once takes equally long for the two cases.

4 Triangulation, 3 points

Consider n cameras with known camera matrices $P_1, \ldots, P_n$, where $n > 10$. We have found matching Sift points $u_1, \ldots u_n$ in the n views and want to find a corresponding 3D point U.

(a) Write a function `ransac_triangulation` that solves this problem using Ransac (similarly to Lab 4). You can write Matlab code or pseudocode. You can use the functions

```
Uhat = minimal_solver(Ps, us)
```

```
reprojection_errors = reprojection_errors(Ps, us, U)
```

without explaining how they work.

(b) For a general camera matrix P, a 3D point, U and an image point $u = (x, y)$, write down formulas for the reprojection residual. The formulas should be on the form

$$r_x = \ldots \tag{4}$$

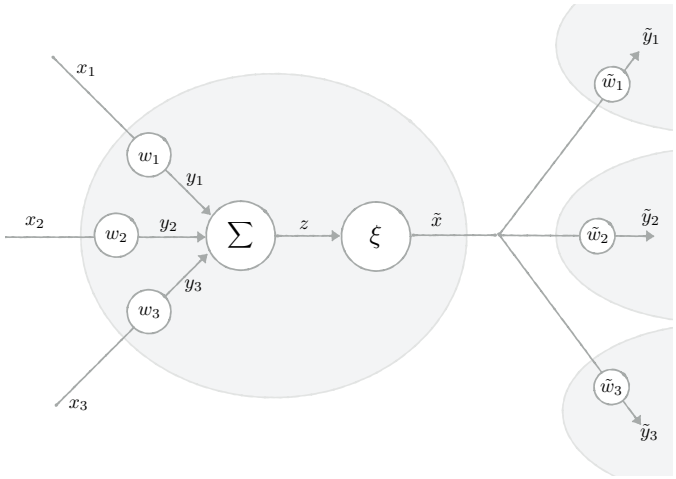
$$r_y = \ldots \tag{5}$$

Make sure to clearly define any new variables that you introduce. Explain when the formulas are valid.

5 Backpropagation, 3 points

The figure below shows a small part of a neural network. Here, ξ denotes the logistic sigmoid function

$$\xi(z) = \frac{e^z}{1 + e^z}. \tag{6}$$



Consider training the weights of this network using stochastic gradient descent on the loss function,

$$L(\theta) = \sum_{i=1}^{n} L_i(\theta), \tag{7}$$

where θ are all the weights. The current values are

$$w_1 = 1, \quad w_2 = -1, \quad w_3 = 2, \quad \tilde{w}_1 = 2, \quad \tilde{w}_2 = 4, \quad \tilde{w}_3 = 1. \tag{8}$$

When training a neural network backpropagation is used to find the gradient of the partial loss ∇L_i, but first a forward pass is performed to find all intermediate values in the network. This gave

$$x_1 = 2, \quad x_2 = 4, \quad x_3 = 1, \quad z = 0, \quad \tilde{x} = 0.5, \tag{9}$$

To find all the partial derivatives in ∇L_i we move backwards through the network. We have computed

$$\frac{\partial L_i}{\partial \tilde{y}_1} = 1, \quad \frac{\partial L_i}{\partial \tilde{y}_2} = 1, \quad \frac{\partial L_i}{\partial \tilde{y}_3} = 2. \tag{10}$$

(a) Derive a formula for

$$\frac{\partial L_i}{\partial w_1}. \tag{11}$$

Motivate your answer carefully.

(b) Compute a numerical value for

$$\frac{\partial L_i}{\partial w_1}. \tag{12}$$

(c) With a learning rate of 0.01, what would the next value for w_1 be?

6 Model estimation, 3 points*

In line fitting we are given a set of 2D points $u_i = (x_i, y_i)$ with $i = 1, \ldots, n$ and want to find the parameters of a line $ax_i + by_i + c = 0$. A good way to measure the residual of a specific measurement u_i is as its perpendicular distance to the line. Using this definition derive an explicit formula for the least squares solution to line fitting.