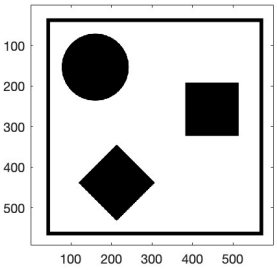
Lab 2 - Theoretical questions

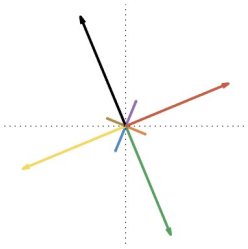
Fanze Meng - fanze@chalmers.se

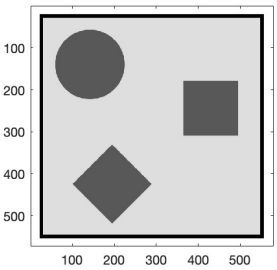
1 Question1

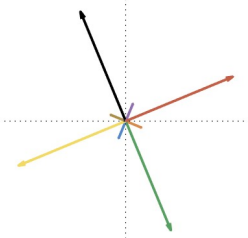
1.1

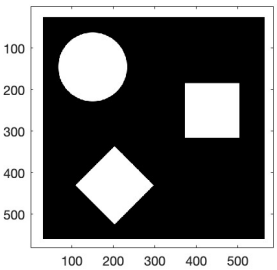
The SIFT keypoints described in this paper are particularly useful due to their distinctiveness, which enables the correct match for a keypoint to be selected from a large database of other keypoints. This distinctiveness is achieved by assembling a high-dimensional vector representing the image gradients within a local region of the image. The keypoints have been shown to be invariant to image rotation and scale and robust across a substantial range of affine distortion, addition of noise, and change in illumination.
It seems that patch A and patch C have the same SIFT descriptor. Since there are just differences in a totally converted color between those two patches. In this condition, the SIFT descriptor would create the same gradient. In a negative (black and white reversed) image, the highest and lowest points are completely reversed, meaning the keypoints should stay the same. However, in other images, variations in grayscale values will cause shifts in the positions of extreme points.

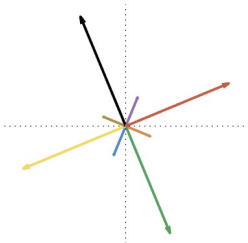

100
200
300
400
500
100 200 300 400 500

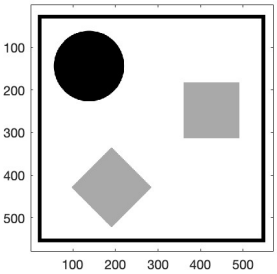


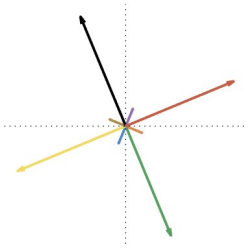

100
200
300
400
500
100 200 300 400 500




100
200
300
400
500
100 200 300 400 500




100
200
300
400
500
100 200 300 400 500



1.2

1. **Invariant to changes in scale and rotations**: By assigning a consistent orientation to each keypoint based on local image properties, the keypoint descriptor can be represented relative to this orientation and therefore achieve invariance to image rotation.

2. **Invariant to uniform additive and multiplicative changes in brightness**: Use the local image descriptor. A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location. These are weighted by a Gaussian window. Then these samples are then accumulated into orientation histograms. Finally, the vector is normalized to unit length. A change in image contrast in which each pixel value is multiplied by a constant will multiply gradients by the same constant, so this contrast change will be canceled by vector normalization. A brightness change in which a constant is added to each image pixel will not affect the gradient values, as they are computed from pixel differences. Therefore, the descriptor is invariant to affine changes in illumination.

1.3

A similarity transformation between two images is defined by a rotation by an angle θ, a change in scale by a factor s, and a 2D translation $\mathbf{t} \in \mathbb{R}^2$.
To compute a similarity transformation from a single match between a SIFT feature in one image and a SIFT feature in a second image, we can follow these steps:

1. **Extract SIFT Features**: Firstly, detect and extract SIFT keypoints and descriptors from both images.

2. **Match Features**: Use a matching algorithm to find the best match for each SIFT feature in the first image from the set of SIFT features in the second image.

3. **Estimate Transformation Parameters**: With a single match, estimate the transformation parameters of the similarity transformation:

   (a) **Rotation (θ)**: Estimate the rotation angle from the orientation information associated with the matched SIFT features.

   (b) **Scale (s)**: Estimate the scale change factor from the scale information associated with the matched SIFT features.

   (c) **Translation ($\mathbf{t}$)**: Estimate the translation vector from the spatial locations of the matched keypoints.

4. **Combine Transformation Parameters**: Combine the estimated rotation angle, scale change factor, and translation vector to form the similarity transformation matrix:

$$\begin{bmatrix} s \cdot \cos(\theta) & -s \cdot \sin(\theta) & t_x \\ s \cdot \sin(\theta) & s \cdot \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

5. **Apply Transformation**: Apply the computed transformation matrix to the keypoints in one image to transform them into the coordinate system of the other image.

This process allows for the computation of a similarity transformation between two images based on a single match between SIFT features.

2 Question2

2.1

It seems to be a method that we can replace a fully connected layer in a convolutional neural network by convolutional layers and can even get the exact same behavior or outputs. There are two ways to do this: one is choosing a convolutional kernel that has the same size as the input feature map or the other is using 1x1 convolutions with multiple channels. We can obtain the same outputs if we use convolutional layers where the kernel size is the same size as the input feature array. As for the original network, the input for the full-conn. layers (20) is 8*8*40, we have the new convolutional layer 20 8*8-filters. The new convolutional layer has 20*8*8*40 = 51,200 trainable parameters. Secondly, a 72×72 input RGB image has an input 9*9*40 for the new convolutional layer. Then the size of the output for a 72×72 input RGB image would be 2*2*20.

2.2

Convolutional layer and pooling layer will affect the receptive field, while the activation function layer usually has no effect on the receptive field, the step size of the current layer does not affect the receptive field of the current layer, the receptive field is not related to the padding, the formula for calculating the receptive field of the current layer is as follows:

$$RF_{i+1} = RF_i + (k-1) \times S_i \tag{1}$$

where RF_{i+1} is the reception field of current layer, RF_i is the reception field of last layer, k is the size of kernal, S_i denotes the product of the step lengths of all previous layers (excluding this layer). So the receptive field of the last convolutional layer (with 40 3 $\times$ 3 filters) in the network shown above can be calculated:

$$\begin{aligned}
\text{Raw} &= 1 \\
\text{Conv1} &= 1 + (3-1) \times 1 = 3 \\
\text{Pool1} &= 3 + (2-1) \times 1 = 4 \\
\text{Conv2} &= 4 + (5-1) \times 2 = 12 \\
\text{Pool2} &= 12 + (2-1) \times 2 = 14 \\
\text{Conv3} &= 14 + (3-1) \times 4 = 22
\end{aligned}$$

Thus the receptive field of the last convolutional layer (with 40 3 $\times$ 3 filters) in the network shown above should be 22.

2.3

Forward Pass: We can calculate some parameters as follows:

$$
\begin{aligned}
x_a &= a_i \times w_1 + b_1 = -8 \\
x_c &= c_i \times w_2 + b_2 = 7 \\
x_d &= d_i \times w_3 + b_3 = -17 \\
y_c &= \max(0, x_c) = 7 \\
y_d &= \max(0, x_d) = 0 \\
z &= x_a + y_c + y_d = -1 \\
p &= \text{sigmoid}(z) = \frac{1}{1+e^{-z}} = \frac{1}{1+e}
\end{aligned}
$$

Backward Pass: We can calculate the following derivatives:

2.3.1 Derivative $L_(i)$ with respect to $w_(1)$ and $b_(1)$:

$$
\begin{aligned}
\frac{\partial L_i}{\partial w_1} &= \frac{\partial L_i}{\partial p} \times \frac{\partial p}{\partial z} \times \frac{\partial z}{\partial x_a} \times \frac{\partial x_a}{\partial w_1} \\
&= -\frac{1}{p} \times p(1-p) \times 1 \times a_i \\
&= \frac{-4e}{1+e}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial L_i}{\partial b_1} &= \frac{\partial L_i}{\partial p} \times \frac{\partial p}{\partial z} \times \frac{\partial z}{\partial x_a} \times \frac{\partial x_a}{\partial b_1} \\
&= -\frac{1}{p} \times p(1-p) \times 1 \times 1 \\
&= \frac{-e}{1+e}
\end{aligned}
$$

2.3.2 Derivative $L_(i)$ with respect to $w_(2)$ and $b_(2)$:

$$
\begin{aligned}
\frac{\partial L_i}{\partial w_2} &= \frac{\partial L_i}{\partial p} \times \frac{\partial p}{\partial z} \times \frac{\partial z}{\partial y_c} \times \frac{\partial y_c}{\partial x_c} \times \frac{\partial x_c}{\partial w_2} \\
&= -\frac{1}{p} \times p(1-p) \times 1 \times 1 \times c_i \\
&= 0
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial L_i}{\partial b_2} &= \frac{\partial L_i}{\partial p} \times \frac{\partial p}{\partial z} \times \frac{\partial z}{\partial y_c} \times \frac{\partial y_c}{\partial x_c} \times \frac{\partial x_c}{\partial b_2} \\
&= -\frac{1}{p} \times p(1-p) \times 1 \times 1 \times 1 \\
&= \frac{-e}{1+e}
\end{aligned}
$$

2.3.3 Derivative $L_(i)$ with respect to $w_(3)$ and $b_(3)$:

$$\begin{aligned}\frac{\partial L_i}{\partial w_3} &= \frac{\partial L_i}{\partial p} \times \frac{\partial p}{\partial z} \times \frac{\partial z}{\partial y_d} \times \frac{\partial y_d}{\partial x_d} \times \frac{\partial x_d}{\partial w_3} \\ &= -\frac{1}{p} \times p(1-p) \times 1 \times 0 \times d_i \\ &= 0\end{aligned}$$

$$\begin{aligned}\frac{\partial L_i}{\partial b_3} &= \frac{\partial L_i}{\partial p} \times \frac{\partial p}{\partial z} \times \frac{\partial z}{\partial y_d} \times \frac{\partial y_d}{\partial x_d} \times \frac{\partial x_d}{\partial b_3} \\ &= -\frac{1}{p} \times p(1-p) \times 1 \times 0 \times 1 \\ &= 0\end{aligned}$$