


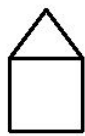
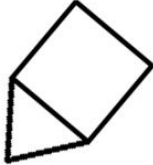
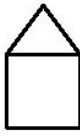

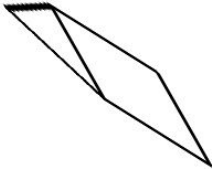
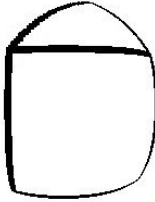
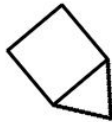
| | | |
|--|-------------------|--|
| Fundamentals of Computer Vision - Final Exam | Dr. B. Nasihatkon |  دانشگاه صنعتی خواجه نصیرالدین طوسی K. N. TOOSI UNIVERSITY OF TECHNOLOGY |
| Name: | ID: | Tir 1396 - June 2017 |

Image transformation (25 points)

A) Image A in the figure below undergoes different geometric transformations resulting in images B-G. Under each image B-G write down the type of the geometric transformation (translation, Euclidean, similarity, affine, perspective, or none of these). You have to write the most specific transformation (i.e. you will not get a point if you write Euclidean instead of translation) **(6 points)**

| | | | |
|---|--|---|--|
|  |  |  |  |
| A | B: Similarity | C: Translation | D: Perspective |
|  |  |  | |
| | E: Affine | F: None | G: Euclidean |

B) Consider a point $(\mathbf{x}, \mathbf{y}) = (20, 40)$ in an image. We want to transform this point by 3 different homography matrices below. Notice that here the first and second coordinates are \mathbf{x} and \mathbf{y} respectively. For each matrix your final result must be a 2D vector $(\mathbf{x}', \mathbf{y}')$ (i.e. not in homogeneous coordinates). **(9 points)**

$$\begin{bmatrix} 2 & 0.5 & -20 \\ 1 & 0 & -10 \\ 0.1 & 0.2 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0.5 & 40 \\ 1 & 2 & -80 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -2 & -0.5 & 0 \\ 3 & 0.1 & 0 \\ -0.4 & 0.1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 0.5 & -20 \\ 1 & 0 & -10 \\ 0.1 & 0.2 & 1 \end{bmatrix} \begin{pmatrix} 20 \\ 40 \\ 1 \end{pmatrix} = \begin{pmatrix} 40 \\ 10 \\ 1 \end{pmatrix} \Rightarrow (x', y') = \left(\frac{40}{11}, \frac{10}{11}\right) \approx (3.64, 0.91)$$

$$\begin{bmatrix} 1 & 0.5 & 40 \\ 1 & 2 & -80 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} 20 \\ 40 \\ 1 \end{pmatrix} = \begin{pmatrix} 80 \\ 20 \\ 1 \end{pmatrix} \Rightarrow (x', y') = (80, 20)$$

$$\begin{bmatrix} -2 & -0.5 & 0 \\ 3 & 0.1 & 0 \\ -0.4 & 0.1 & 1 \end{bmatrix} \begin{pmatrix} 20 \\ 40 \\ 1 \end{pmatrix} = \begin{pmatrix} -60 \\ 64 \\ -3 \end{pmatrix} \Rightarrow (x', y') = \left(\frac{-60}{-3}, \frac{64}{-3}\right) = \left(20, -\frac{64}{3}\right) \approx (20, -21.33)$$

C) This question is about geometric properties preserved by different types of image transformations. Fill out the table below. **(10 points)**, each wrong answer has **0.25 negative marks**, effective only for this question)

Note: self intersecting quadrilateral are still considered quadrilaterals.

| | Translation | Euclidean | Similarity | Affine | Perspective |
|--|-------------|-----------|------------|--------|-------------|
| distance between a pair of points remains constant | True | True | False | False | False |
| angle between two lines remain constant | True | True | True | False | False |
| lines remain lines | True | True | True | True | True |
| angle between a line and the x-axis remains constant | True | False | False | False | False |
| quadrilaterals remain quadrilaterals | True | True | True | True | True |
| parallel lines remain parallel | True | True | True | True | False |
| circles remain circles | True | True | True | False | False |
| ratio between the areas of two shapes | True | True | True | True | False |

| | | | | | |
|----------------|--|--|--|--|--|
| stays constant | | | | | |
|----------------|--|--|--|--|--|

RANSAC (15 points)

Assume we are to find an **affine** transformation which maps image **I** to image **J**. A point **x** in image **I** is mapped to its corresponding point **y** in image **J** with a map $\mathbf{y} = \mathbf{A} \mathbf{x} + \mathbf{b}$. Our task is to find **A** and **b**. We do this by finding point correspondences between the images.

A) What is the minimum number of (**correct**) point correspondences (i.e. how many pairs of points) needed to estimate the affine transform. **(3 points)**

3 matches are needed. (An affine map has 6 degrees of freedom (6 unknowns), and each pair of points provides 2 equations, thus 3 pair of points is needed to solve for the unknowns)

B) Assume that we have found a number of **putative** point correspondences in the form of (X_i, Y_i) $i = 1, 2, \dots, M$, where Y_i is the point in the second image matched to X_i in the first image, and M is relatively large. Among these only about **40 percent are true matches**. You are to implement a RANSAC algorithm to find the correct matches and estimate the transform. Write a pseudocode describing your algorithm by completing the code in the box below. Your algorithm should be tailored for the special case of affine transformation. **(6 points)**

Input:

- point correspondences $X_i, Y_i, i = 1, 2, \dots, M$
- a threshold δ

Output: the set **I** of inlier indices and the true affine map **A, b**

$I_{best} = \{\}$ **# empty set**

for iteration = 1 to N:

$j, k, l \leftarrow$ choose 3 random numbers from $1, 2, \dots, M$ without replacement

$A, t \leftarrow$ estimate_affine_map($\{(X_j, Y_j), (X_k, Y_k), (X_l, Y_l)\}$)

$I_{in} \leftarrow \{i \mid \|AX_i + t - Y_i\| < \delta, i \in \{1, 2, \dots, M\}\}$ **# indices of inliers**

if size of $I_{in} >$ size of I_{best} :

$I_{best} \leftarrow I_{in}$

$A, t \leftarrow$ estimate_affine_map($\{(X_i, Y_i) \mid i \in I_{best}\}$)

return A, t, I_{best}

C) What is the minimum number of samples **N** we can choose in the above algorithm when we want the algorithm succeed with a probability of 0.9999? **(6 points)**

Remember the relation between the probability of getting at least one sample with all inliers (p), the number of point matches to compute the transformation (s), and the

proportion of **outliers** (e): $(1 - p) > (1 - (1 - e)^s)^N$

$\log(1 - p) > N \log(1 - (1 - e)^s)$

$\Rightarrow N > \log(1 - p) / \log(1 - (1 - e)^s) = \log(0.0001) / \log(1 - (0.40)^3) \approx 139.26 \Rightarrow N_{min} = 140$

Features / SIFT (25 points)

- A) Describe each of the three major stages of the SIFT algorithm. You just have to state their purpose, not the steps taken in each of them. **(6 points)**
- a) SIFT detection

 - b) SIFT description

 - c) SIFT matching
- B) Which of the above steps can be done independently for each image. Which ones require both images? **(3 points)**
- C) RANSAC can be used to improve which of the above stages? How? **(3 points)**
- D) **Orientation assignment** is done in which of the above stages? What is its purpose? **(3 points)**
- E) A Kd-tree can be used in which of the stages above? What issue with the algorithm makes us consider using a Kd-tree or similar data structures? **(3 points)**

F) The matrices below show the output of the **absolute value** of the **Difference of Gaussians** operator applied on images with different bandwidths in one octave of in the SIFT algorithm. Identify the keypoint locations in the **scale-space** chosen by the SIFT algorithm, **before** removing the edges and low contrast points, and sub-pixel localization. For each keypoint report **x, y and scale**.

Note: Pixel indices start from **(x=0,y=0)**. As a reference, in the highlighted cell at scale=2, the scale-space location is (x=3, y=1, scale=2). **(7 points)**

Note: You will get **negative marks** for reporting wrong keypoint locations (only effective for this question).

Hint: The keypoints do not exist in the boundary pixels of the images, nor they can be in the first and last scales.

scale = 0

| | | | |
|---|---|---|---|
| 1 | 3 | 2 | 4 |
| 1 | 6 | 3 | 2 |
| 5 | 7 | 2 | 3 |
| 6 | 1 | 2 | 1 |

scale = 1

| | | | |
|---|---|---|---|
| 4 | 5 | 1 | 2 |
| 5 | 3 | 7 | 3 |
| 4 | 5 | 8 | 2 |
| 3 | 2 | 1 | 2 |

scale = 2

| | | | |
|---|---|---|---|
| 1 | 1 | 3 | 1 |
| 3 | 4 | 5 | 4 |
| 2 | 7 | 6 | 2 |
| 1 | 1 | 4 | 2 |

scale = 3

| | | | |
|---|----|---|---|
| 1 | 2 | 3 | 1 |
| 3 | 3 | 4 | 2 |
| 2 | 10 | 7 | 3 |
| 1 | 3 | 2 | 1 |

scale = 4

| | | | |
|---|---|---|---|
| 3 | 1 | 1 | 1 |
| 3 | 5 | 7 | 1 |
| 3 | 9 | 8 | 1 |
| 2 | 2 | 2 | 1 |

Generative Classification (11 points)

A) What is the difference between generative and discriminative models for classification? (4 points)

B) Assume that we want to classify images in 3 categories of **Apple**, **Orange** and **Banana** for which the prior probabilities are $P(\mathbf{A}) = 0.3$, $P(\mathbf{O}) = 0.5$, $P(\mathbf{B}) = 0.2$. We see a new image \mathbf{I} from which we extract a feature vector \mathbf{x} . The likelihoods are as:

$$P(\mathbf{x} | \mathbf{A}) = 0.24$$

$$P(\mathbf{x} | \mathbf{O}) = 0.15$$

$$P(\mathbf{x} | \mathbf{B}) = 0.38$$

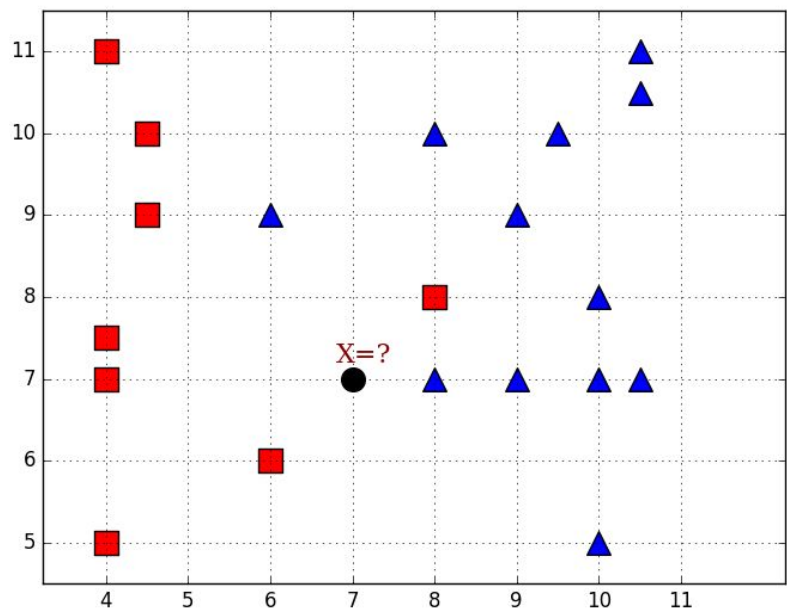
What should image \mathbf{I} be classified as? An apple, an orange, or a banana? Why?

Write down the complete derivations. (7 points)

k-Nearest Neighbours Classifier (11 points)

A) Briefly describe how Nearest Neighbour (NN) and k-Nearest Neighbour (kNN) classifiers work. **(3 points)**

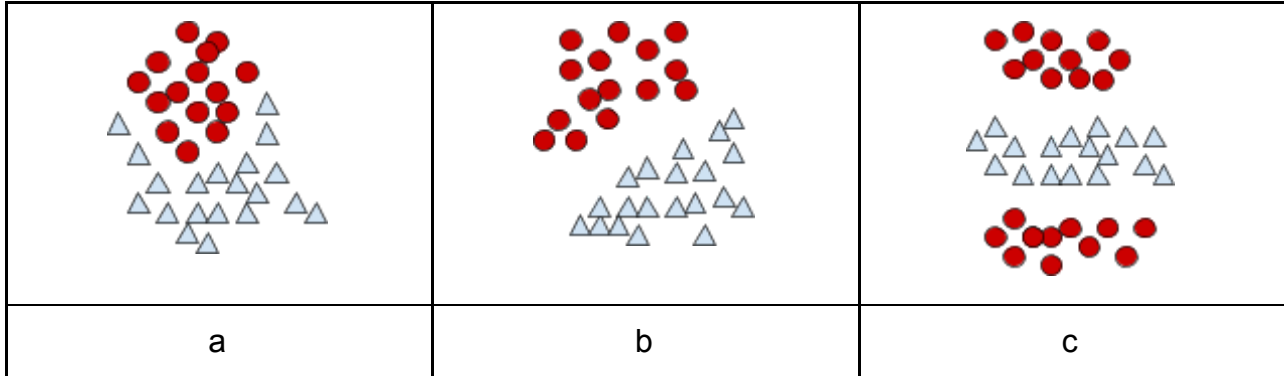
B) The figure below shows our training data of 2D features for two classes, name them Square and Triangle. We want to classify a new data point **X** shown by a circle in the figure. Mark the first, second, 3rd, 4th and 5th nearest neighbours (write numbers 1,2,3,4,5 beside each data point). **(3 points)**



C) What is the result of classification with NN, 3-NN and 5-NN classifier? Why? **(5 points)**

Support Vector Machines (SVM) (13 points)

Assume we want to classify our images into two classes **+1** and **-1**. We have extracted 3 different types of 2-dimensional features. The images below show our training data plotted in the three corresponding feature spaces.



A) Which feature types among the above (a, b or c) are suitable to use with a linear SVM. For each case explain **why or why not**. (3 points)

a)

b)

c)

B) If we want to do SVM classification with the remaining feature type(s), what method we can use? (3 points)

C) Your task in this question is to find **w** and **b** given a data set of 2D features. To do so, you need to solve a quadratic program. But, to make your job simpler, your good teacher tells you that the support vectors are **x=(-1, 2)** and **x=(-2, 1)** for class **-1** and **x=(1, 3)** for class **+1**. Calculate the vector **w** and scalar **b**. Write down the derivations. (Assume that the data points are linearly separable.) (7 points)

