NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING

EXAMINATION FOR
Semester 1 AY2012/2013

CS4243
COMPUTER VISION & PATTERN RECOGNITION

November 2012 Time Allowed: 2 Hours

INSTRUCTIONS TO CANDIDATES

1. This examination paper contains **FIVE (5)** questions and comprises **TWELVE (12)** printed pages, including this page.
2. Answer **ALL** questions. The maximum mark is 100.
3. Write your answers in the space provided in this booklet. Use the reverse sides if necessary.
4. Write legibly. You may use pen or pencil.
5. This is an OPEN BOOK examination.
6. Please write your Matriculation Number below.

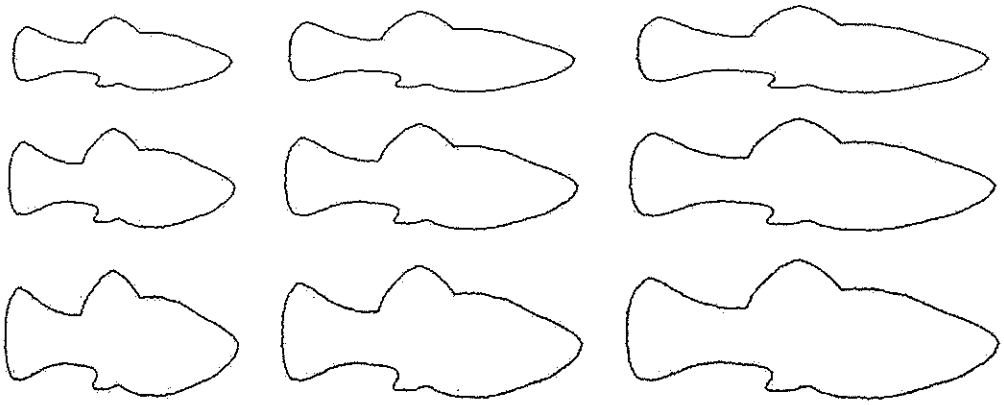
Matriculation No.: _____

This portion is for the examiner's use only.

Question	Marks	Remarks
Q1		
Q2		
Q3		
Q4		
Q5		
Total		

Q1: Active Shape (15 marks)

You are leading a team of computer vision experts to segment the contours of a particular kind of fish from images. Your team collects sample images of the fish, whose contours are shown below:



Your team marks 20 consistent landmark points on all the fish contours, deriving for each contour a shape vector as follows:

$$\mathbf{s}_i = (x_{i1}, y_{i1}, x_{i2}, y_{i2}, \ldots, x_{i20}, y_{i20})\,. \tag{1}$$

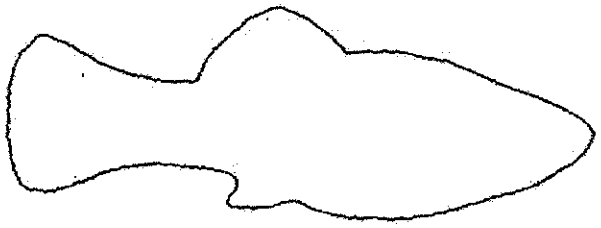
Then, you spatially align the fish contours and apply Principal Component Analysis to the aligned contours.

Q1(a) (5 marks)

How many dimensions (modes) are required to capture the shape variations of these fish contours?

Q1(b) (10 marks)

The following figure shows a zoom-in view of the mean shape. Draw in the figure a new shape generated by the active shape model for the fish contours whose first shape parameter (e_1) is negative and second shape parameter (e_2) is positive.



Q2: Background Removal (30 marks)

Your company Intelli-Sense develops computer vision applications for surveillance companies. One of the applications performs real-time background removal of surveillance video based on video frame averaging. The average M_k over k video frames I_i is given by the formula:

$$M_k = \frac{1}{k}\sum_{i=1}^{k} I_i \ . \tag{2}$$

But, you have learned that a more efficient and effective way to compute the average is to compute the **incremental average**:

$$M_k = \frac{k-1}{k} M_{k-1} + \frac{1}{k} I_k \ . \tag{3}$$

As the video runs the whole day, from morning to evening, with changing lighting condition, you figure out that you need to perform **running average** of video frames. The running average A_k of length l at frame k is the average over a window of l frames up to frame k (Figure 1):

$$A_k = \frac{1}{l}\sum_{i=k-l+1}^{k} I_i \ . \tag{4}$$

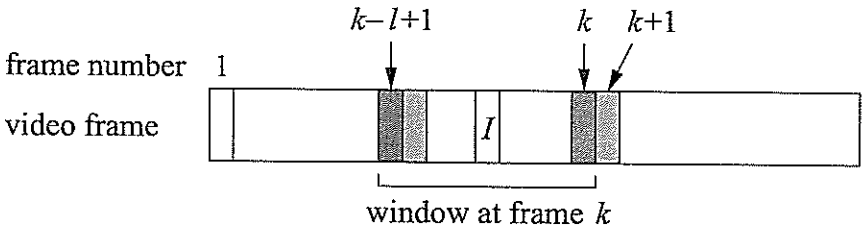


Figure 1: Running average over video frames.

2(a) (10 marks)

Based on Equation 3, derive the formula for recovering M_{k-1} from M_k and a video frame denoted as I in the window.

2(b) (10 marks)

Write a Python program fragment to compute the average of the first l frames in a video using the formula for **incremental average** (Equation 3). Use the Python and OpenCV functions listed in **Supplementary Notes for Q2**.

2(c) (10 marks)

Continue from the program in 2(b), write a Python program fragment to compute the running average of length l at each video frame from frame $l+1$ onward using **incremental running average**. To compute incremental running average, your program needs to move the window to the next frame and update the running average A_k using only the shaded frames shown in Figure 1. Use the Python and OpenCV functions listed in **Supplementary Notes for Q2**.

Supplementary Notes for Q2

Python and OpenCV functions for **2(b)** and **2(c)**:

- Keep the video frames in a Python list using the following operations:
  ```
  list = [] # create an empty list
  del list[i] # delete element at index i
  list.append(x) # append element to the end of the list
  ```
  Be reminded that Python list index starts at 0 instead of 1.

- Extract a video frame using the function:
  ```
  frame = cv.QueryFrame(invid)
  ```
  You may assume that `invid` is already given, and the length of the video is given in the variable `length`.

- You need to clone the video frame before storing them in the Python list (because `cv.QueryFrame` reuses the same memory space for all the frames):
  ```
  newimage = cv.CloneImage(oldimage)
  ```

- Create an empty image using the functions:
  ```
  image = cv.CreateImage((width, height), 8, 3)
  cv.SetZero(image)
  ```

- Perform weighted addition using the function:
  ```
  cv.AddWeighted(src1, weight1, src2, weight2, 0.0, target)
  ```

Q3: Image Registration (15 marks)

You are developing a sophisticated image registration algorithm, which requires a good initialisation. After reading some materials, you find that a reduced quadratic transformation can give an appropriate initialisation. The reduced quadratic transformation is given by the following equations:

$$\begin{aligned} x' &= a_1 + a_2\, x + a_3\, y + a_4\, x^2 + a_4\, y^2, \\ y' &= a_5 + a_2\, y - a_3\, x + a_6\, x^2 + a_6\, y^2. \end{aligned} \tag{5}$$

To solve for the transformation parameters, you collect n data points $\mathbf{p}_i = (x_i, y_i)^\top$ and their corresponding points $\mathbf{p}'_i = (x'_i, y'_i)^\top$, and form the system of linear equations

$$\mathbf{D}\,\mathbf{a} = \mathbf{b} \tag{6}$$

where

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix}. \tag{7}$$

3(a) (5 marks)

What are the elements in $\mathbf{b}$?

3(b) (10 marks)

What are the elements in D?

Q4: Image Morphing (25 marks)

A customer approaches your company for a special image morphing task. She likes the image to not only morph from one to the other, but also has the impression of rotating in depth, as illustrated in Figure 2. In this morphing sequence, the landmark points should still move smoothly along the morphing path.

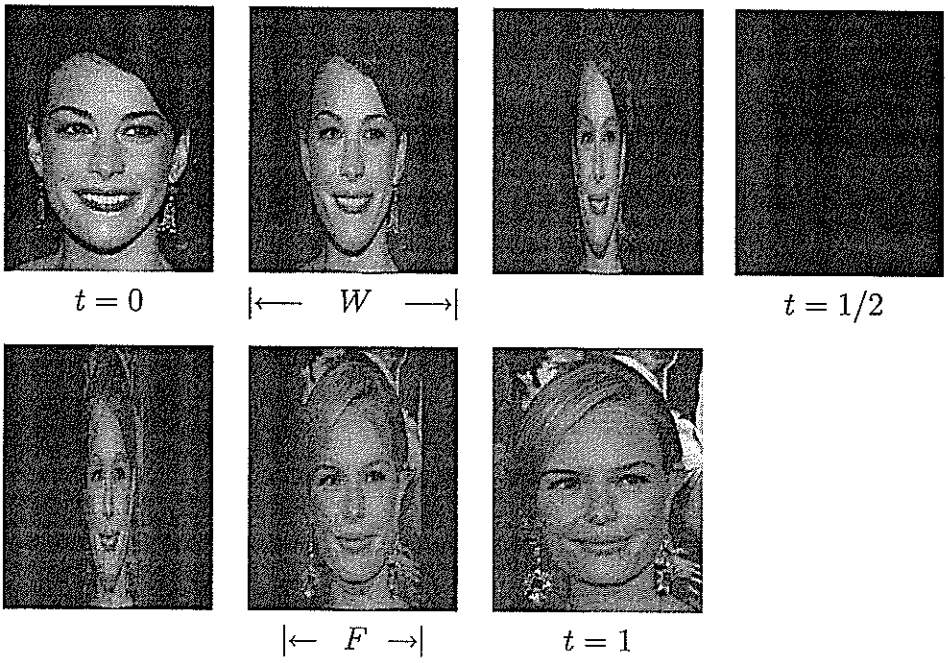


Figure 2: Fancy image morphing. W is the width of the image, F marks the face part of the image, and t denotes time.

In normal image morphing, the intermediate position $\mathbf{r}(t) = (x_r(t), y_r(t))$ of a landmark point is given by the linear interpolation of the corresponding points $\mathbf{p} = (x_p, y_p)$ and $\mathbf{q} = (x_q, y_q)$ in the first and last image:

$$\mathbf{r}(t) = (1 - t)\mathbf{p} + t\,\mathbf{q}\,. \tag{8}$$

In this special morphing, we want to keep $\mathbf{r}$ within the face part of the image as shown in Figure 2. So, we need to modify the formula for $\mathbf{r}$ so that its x-coordinate moves linearly towards the mid-line of the image for $0 \le t \le 1/2$, and away from the mid-line for $1/2 \le t \le 1$, such that

$$\mathbf{r}(t) = (x_r(t), y_r(t)) = \begin{cases} (x_p, y_p) & \text{when } t = 0, \\ \left(\frac{1}{2}W, \frac{1}{2}(y_p + y_q)\right) & \text{when } t = 1/2, \\ (x_q, y_q) & \text{when } t = 1, \end{cases} \tag{9}$$

where W is the width of the image as shown in Figure 2.

(Turn to the next page.)

4(a) (5 marks)

What is the formula for $y_r(t)$?

4(b) (10 marks)

What is the formula for $x_r(t)$ for $0 \leq t \leq 1/2$?

4(c) (10 marks)

What is the formula for $x_r(t)$ for $1/2 \leq t \leq 1$?

Q5: Robust Method for Homography (15 marks)

Homography is often used for stitching multiple views of a scene or registering the views to a common reference view. Given the corresponding points $\mathbf{x}_i = (x_i, y_i)$ and $\mathbf{x}'_i = (x'_i, y'_i)$ of two views of a scene, homography relates the points by the equation

$$\begin{bmatrix} \rho'_i x'_i \\ \rho'_i y'_i \\ \rho'_i \end{bmatrix} = \tilde{\mathbf{x}}'_i = \mathbf{H}\tilde{\mathbf{x}}_i = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}. \tag{10}$$

In actual implementation, the feature matching algorithm often makes mistakes in matching corresponding points. So, robust method is used to determine the best set of corresponding points that produce the homography with the smallest error.

Now, suppose that the corresponding points, with possible errors, are already computed. Write the algorithm that applies RANSAC to determine the best homography. Give the equations for measuring the errors of the homography on the data points. You may assume that the algorithm for computing homography for a given set of corresponding points is known.

(You may use this page for your answer.)

— END OF PAPER —