Assignment 1: CS 663, Fall 2024

Due: 23rd August before 11:55 pm

Remember the honor code while submitting this (and every other) assignment. You may discuss broad ideas with other students or ask me for any difficulties, but the code you implement and the answers you write must be your own. We will adopt a zero-tolerance policy against any violation.

Submission instructions: Follow the instructions for the submission format and the naming convention of your files from the submission guidelines file in the homework folder. Please see assignment1.zip in the homework folder. For all the questions, write your answers and scan them, or type them out in word/Latex. In eithe case, create a separate PDF file. The last two questions will also have code in addition to the PDF file. Once you have finished the solutions to all questions, prepare a single zip file and upload the file on moodle before 11:55 pm on 23rd August. We will not penalize submission of the files till 10 am on 24th August. **No assignments will be accepted after this time.** Please preserve a copy of all your work until the end of the semester. **Your zip file should have the following naming convention:** RollNumber1_RollNumber2_RollNumber3.zip for three-member groups, RollNumber1_RollNumber2.zip for two-member groups and RollNumber1.zip for single-member groups.

1. Consider that you want to align two XRay images of a patient's forearm bone, one taken with pixel size 0.5×0.5 and the other with pixel size 0.25×0.25, using a control point based approach. What motion model will you adopt to solve this problem and why? Now consider that the second image had a pixel size of 0.25×0.5. What motion model will you adopt to solve this problem and why? In both cases, do not use unnecessarily complicated motion models than what is required for the task at hand. All dimensions are in millimetres. [5+5 = 10 points]
   Solution: The motion between the two images will be rigid (2D rotation + 2D translation) if the pixel resolution were the same in both images. This is because the forearm bone is a rigid object. But in the first case, the pixel resolution of the two images are differing by a constant scaling factor. Hence the motion model consists of uniform scaling, rotation and translation (all three in 2D) if pixel index-based coordinates (row and column indices) are used. If instead, you converted the pixel index-based coordinates to geometric coordinates using knowledge of the resolution, the motion model is just rotation and translation (both in 2D) because the scaling factor will be accounted for.

   In the second case, the first and second images have a pixel resolution of 0.5×0.5 and 0.25×0.5 respectively. Hence, the motion model is non-uniform scaling, rotation and translation (all three in 2D) if pixel index-based coordinates (row and column indices) are used. If instead, you converted the pixel index-based coordinates to geometric coordinates using knowledge of the resolution, the motion model is just rotation and translation (both in 2D) because the scaling factors will be accounted for.

   Grading scheme: 5 points for the first part. 3 points for identification of the motion model and 2 points for identification of the coordinate type (geometric or index-based). If you write '2D affine' for your answer, you will lose 1 point as this motion model accounts for unnecessary degrees of freedom. The same marking scheme is applicable to the second part.

2. A student is trying to align three images I_1, I_2, I_3. Let us assume that they are related to each other by purely translational motion. Let the motion from image I_i to I_j be denoted by $\boldsymbol{u_{ij}} \in \mathbb{R}^2$ where $i, j \in \{1, 2, 3\}, i \neq j$. What is the relationship between $\boldsymbol{u_{12}}, \boldsymbol{u_{23}}, \boldsymbol{u_{13}}$? When you perform motion estimation in practice, will that relationship hold in practice? Why (not)? There is no need to write any code for this. [10 points]
   Answer: The relationship between $\boldsymbol{u_{12}}, \boldsymbol{u_{23}}, \boldsymbol{u_{13}}$ is given by $\boldsymbol{u_{12}} + \boldsymbol{u_{23}} = \boldsymbol{u_{13}}$ using a simple law of vector

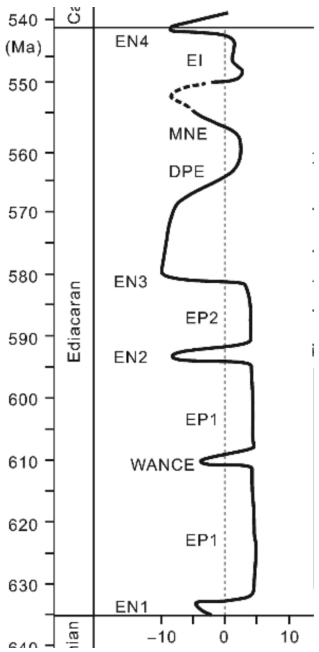

Figure 1: Graph used for question 2.

addition. [5 points for this relationship or an equivalent one, such as $\boldsymbol{u_{13} + u_{32} = u_{12}, u_{32} = -u_{23}}$] When you perform motion estimation in practice, this relationship may generally not hold because the estimated motion may contain errors due to various issues: image noise, local optima in the parameter landscape, interpolation artefacts, etc. [2 points for stating that this relationship will not hold and 3 points for giving reasons why – at least 2 reasons need to be given]

3. You are viewing the graph in Fig. 1 from a research paper. Unfortunately from the graph, the (x, y) values at only a few points can be observed. You need to obtain the (x, y) values at many other points. Hence you can do the following: you extract the image from the paper, and open it through MATLAB which provides a function called impixelinfo. This function gives you the (x, y) coordinates of any spatial location pointed by your mouse. However, the coordinate system of the graph and that of MATLAB will be different. Describe a procedure to convert the coordinates of any point from MATLAB's coordinate system to the coordinate system of the graph. This will help you obtain the (x, y) coordinates in the coordinate system of the graph. Support your answer with suitable equations. There is no need to write any code for this. [15 points]
Solution: Pick any 3-4 points from the graph with observable coordinates in the graph's coordinate system. Let us call these points $\{(g_{xi}, g_{yi})\}_{i=1}^{4}$. The easiest ones to pick are the two points where the dotted line intersects the two solid parallel horizontal lines, the point close to C1 (see figure) and the point of intersection of the lines EN1 and Ediacaran. However any 3+ points for which the coordinates in the graph are accurately observable, can be chosen. Open the figure in MATLAB and note down the corresponding coordinates in MATLAB's coordinate system using `impixelinfo`. Call these coordinates $\{(m_{xi}, m_{yi})\}_{i=1}^{4}$. Then we will have $\boldsymbol{G} = \boldsymbol{MA}$ where $\boldsymbol{G}$ is the 4×3 matrix whose first column contains x coordinates $\{g_{xi}\}$, second column contains y coordinates $\{g_{yi}\}$, and third column contains all ones. Likewise, $\boldsymbol{M}$ is the 4×3 matrix whose first column contains x coordinates $\{m_{xi}\}$, second column contains y coordinates $\{m_{yi}\}$, and third column contains all ones. Also, $\boldsymbol{A}$ is the 3×3 affine motion matrix which can be obtained as $\boldsymbol{A} = \boldsymbol{M}^T\boldsymbol{M}^{-1}\boldsymbol{M}^T\boldsymbol{G}$. Now, if you observe the coordinates of any point in MATLAB, you can get the coordinates of the same point in the graph coordinate system using the earlier relation.
marking scheme: The equations carry 7.5 points. The broad idea carries 7.5 points.

4. Suppose the motion model between two images is expressed as follows: For any pair of physically corresponding points (x_1, y_1) in image 1 and (x_2, y_2) in image 2, we have: $x_2 = ax_1^2 + by_1^2 + cx_1y_1 + dx_1 + ey_1 + f, y_2 = Ax_1^2 + By_1^2 + Cx_1y_1 + Dx_1 + Ey_1 + F$ where $a, b, c, d, e, f, A, B, C, D, E, F$ are all (unknown) constants. Explain how you will perform motion estimation using control points. Write down all key equations and the solution in the form of matrices and vectors. There is no need to write any code for this. [15 points]

Answer: Let (x_{1i}, y_{1i}) and (x_{2i}, y_{2i}) be pairs of corresponding control points in the two images. Then we have the following system of equations:

$$\begin{pmatrix} x_{21} & x_{22} & . & . & x_{2n} \\ y_{21} & y_{22} & . & . & y_{2n} \end{pmatrix} = \begin{pmatrix} a & b & c & d & e & f \\ A & B & C & D & E & F \end{pmatrix} \begin{pmatrix} x_{11}^2 & x_{12}^2 & . & . & x_{1n}^2 \\ y_{11}^2 & y_{12}^2 & . & . & y_{1n}^2 \\ x_{11}y_{11} & x_{12}y_{12} & . & . & x_{1n}y_{1n} \\ x_{11} & x_{12} & . & . & x_{1n} \\ y_{11} & y_{12} & . & . & y_{1n} \\ 1 & 1 & . & . & 1 \end{pmatrix} \tag{1}$$

which can be expressed as $\boldsymbol{P_2} = \boldsymbol{C}\boldsymbol{Q_1}$ where $\boldsymbol{P_2}$ has dimensions $2 \times n$, $\boldsymbol{C}$ has dimensions 2×6 and $\boldsymbol{Q_1}$ has dimensions $6 \times n$. Here n is the number of control points. Here both $\boldsymbol{P_2}$ and $\boldsymbol{Q_1}$ are known or can be derived from the control points, and $\boldsymbol{C}$ is the matrix of parameters which can be estimated in the form $\boldsymbol{C} = \boldsymbol{P_2}(\boldsymbol{Q_1}\boldsymbol{Q_1}^\top)^{-1}$. **Marking scheme:** 10 points for the correct system of equations with the meaning of all symbols explained (unless they were given in the question) and 5 points for the solution using the pseudo-inverse.

5. Read in the images T1.jpg and T2.jpg from the homework folder using the MATLAB function `imread` and cast them as a double array. Let us call these images as J1 and J2. These are magnetic resonance images of a portion of the human brain, acquired with different settings of the MRI machine. They both represent the same anatomical structures and are perfectly aligned (i.e. any pixel at location (x, y) in both images represents the exact same physical entity). We are going to perform a simulation experiment for image alignment in a setting where the image intensities of physically corresponding pixels are different. To this end, do as follows:

   (a) Write a piece of MATLAB code to rotate the second image by $\theta = 28.5$ degrees anti-clockwise. You can use the `imrotate` function in MATLAB to implement the rotation using any interpolation method. Note that the rotation is performed implicitly about the centroid of the image. While doing so, assign a value of 0 to unoccupied pixels. Let us denote the rotated version of J2 as J3.

   (b) Our job will now be to align J3 with J1 keeping J1 fixed. To this end, we will do a brute-force search over θ ranging from -45 to +45 degrees in steps of 1 degree. For each θ, apply the rotation to J3 to create an intermediate image J4, and compute the following measures of dependence between J1 and J4:

   - the normalized cross-correlation (NCC)
   - the joint entropy (JE)
   - a measure of dependence called quadratic mutual information (QMI) defined as $\sum_{i_1}\sum_{i_2}(p_{I_1I_2}(i_1, i_2) - p_{I_1}(i_1)p_{I_2}(i_2))^2$, where $p_{I_1I_2}(i_1, i_2)$ represents the normalized joint histogram (*i.e.*, joint pmf) of I_1 and I_2 ('normalized' means that the entries sum up to one). Here, the random variables I_1, I_2 denote the pixel intensities from the two images respectively. For computing the joint histogram, use a bin-width of 10 in both I_1 and I_2. For computing the marginal histograms p_{I_1} and p_{I_2}, you need to integrate the joint histogram along one of the two directions respectively. You should write your own joint histogram routine in MATLAB - do not use any inbuilt functions for it.

   (c) Plot separate graphs of the values of NCC, JE, QMI versus θ and include them in the report PDF.

   (d) Determine the optimal rotation between J3 and J1 using each of these three measures. What do you observe from the plots with regard to estimating the rotation? Explain in the report PDF.

   (e) For the optimal rotation using JE, plot the joint histogram between J1 and J4 using the `imagesc` function in MATLAB along with `colorbar`. Include it in the report PDF.

   (f) We have studied NCC and JE in class. What is the intuition regarding QMI? Explain in the report PDF. (Hint: When would random variables I_1 and I_2 be considered statistically independent?) [2+10+2+3+3+5=25 points]

Answer: See code in homework folder. The maximum of QMI and the minimum of JE are quite close to 29 degrees. However the NCC measure completely fails as the assumption of linear relationship between the intensities of the two images fails. The QMI measure determines the squared difference between the joint PMF and the product of the marginals. If this difference is 0 (its minimum value), then the two random variables I_1, I_2 are statistically independent. When two images are well-aligned, the intensities cannot be statistically independent. Hence, we seek to maximize the difference between p_{12} and $p_1 p_2$ for all intensities. This is in line with the empirical result where the QMI achieves a maximum value close to 29 degrees.
Marking scheme: The marking scheme is already given. Deduct 1 point for every plot not included in the report (even if the code produces it).

6. Read in the images 'goi1.jpg' and 'goi2.jpg' from the homework folder using the MATLAB `imread` function and cast them as double. These are images of the Gateway of India acquired from two different viewpoints. As such, no motion model we have studied in class is really adequate for representing the motion between these images, but it turns out that an affine model is a reasonably good approximation, and you will see this. We will estimate the affine transformation between these two images in the following manner:

   (a) Display both images using `imshow(im1)` and `imshow(im2)` in MATLAB. Use the `ginput` function of MATLAB to manually select (via an easy graphical user interface) and store $n = 12$ pairs of physically corresponding salient feature points from both the images. For this, you can do the following:
   ```
   for i=1:12, figure(1); imshow(im1/255); [x1(i), y1(i)] = ginput(1);
   figure(2); imshow(im2/255); [x2(i), y2(i)] = ginput(1);
   ```
   Tips: Avoid selecting points which are visible in only one image. Try to select them as accurately as possible, but our procedure is robust to small sub-pixel errors. Make sure `x1(i),y1(i)` and `x2(i),y2(i)` are actually physically corresponding points. Salient feature points are typically points that represent corners of various structures.

   (b) Write MATLAB code to determine the affine transformation which converts the first image ('goi1') into the second one ('goi2').

   (c) Using nearest neighbor interpolation that you should implement yourself, warp the first image with the affine transformation matrix determined in the previous step, so that it is now better aligned with the second image. You are not allowed to use any implementation for this already available in MATLAB. Display all three images side by side in the report PDF.

   (d) Repeat the previous step with bilinear interpolation that you should implement yourself. You are not allowed to use any implementation for this already available in MATLAB. Display all three images side by side in the report PDF.

   (e) In the first step, suppose that the n points you chose in the first image happened to be collinear. Explain (in the report PDF) the effect on the estimation of the affine transformation matrix. [5+5+5+5+5=25 points]
   Answer and marking scheme: Marking scheme is already provided. See code in homework folder. If the n points were collinear: Let the matrix of points from image 1 and image 2 be denoted as $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ respectively. Both have size $3 \times n$. We have the relationship $\boldsymbol{A}\boldsymbol{P_1} = \boldsymbol{P_2}$, and the least squares solution for $\boldsymbol{A}$ is given as $\boldsymbol{A} = \boldsymbol{P_2}\boldsymbol{P_1}^t \text{inverse}(\boldsymbol{P_1}\boldsymbol{P_1}^t)$. However if the n points are collinear, then $\boldsymbol{P_1}\boldsymbol{P_1}^t$ will have rank equal to 1 and hence the solution for $\boldsymbol{A}$ will not be unique.