# Assignment 2: CS 763, Computer Vision

Due: 12th Feb before 11:55 pm

**Remember the honor code while submitting this (and every other) assignment. All members of the group should work on and understand all parts of the assignment. We will adopt a zero-tolerance policy against any violation.**

**Submission instructions:** You should ideally type out all the answers in Word (with the equation editor) or using Latex. In either case, prepare a pdf file. For assignment submission, follow the instructions for arrangement of folders and subfolders as given in `http://www.cse.iitb.ac.in/~ajitvr/CS763_Spring2017/HW2/HW2_Alignment.rar`. Create a single zip or rar file obeying the aforementioned structure and name it as follows: A2-IdNumberOfFirstStudent-IdNumberOfSecondStudent-IdNumberOfThirdStudent.zip. (If you are doing the assignment alone, the name of the zip file is A2-IdNumber.zip). Upload the file on moodle BEFORE 11:55 pm on 12th February. Late assignments will be assessed a penalty of 50% per day late. Note that only one student per group should upload their work on moodle. Please preserve a copy of all your work until the end of the semester. If you have difficulties, please do not hesitate to seek help from me.

1. Imagine you have two point-sets $A$ and $B$, each containing a (possibly unequal) number of 3D points. You are told that the point-sets are related to each other by an unknown but small 3D rotation $\boldsymbol{R}$ and small 3D translation $\hat{\boldsymbol{t}}$, besides a small amount of measurement noise, i.e. $\boldsymbol{a_i} = \boldsymbol{R}\boldsymbol{b_i} + \hat{\boldsymbol{t}} + \boldsymbol{\eta_i}$ where $\boldsymbol{a_i}, \boldsymbol{b_i}$ are a pair of corresponding points. However, now suppose that the correspondence between the points is unknown to you. In other words the $i^{\text{th}}$ point in $A$ need not physically correspond to the $i^{\text{th}}$ point in $B$. So consider the following iterative algorithm: (1) For each point in $A$, we assign its nearest neighbor in $B$ (in terms of Euclidean distance) to be its corresponding point. (2) In the second step, we use a least squares method to determine the rotation and translation given all such pairs of corresponding points. (3) In the third step, we apply the estimated rotation and translation to every point in $B$. These three steps continue in an iterative manner until the estimated rotation in a given step is close to identity and the translation is close to 0.
   Your task is as follows: Implement the aforementioned algorithm using datasets provided in the homework folder. Your code should display an overlay of points from $A$ and transformed points from $B$ using the scatter3 command in MATLAB and include the plots from the first and last iteration in your report. Determine $\boldsymbol{R}$ and $\hat{\boldsymbol{t}}$ and explain how you computed these quantities in your report. Your report should also mention which command in MATLAB you used for nearest neighbor computation. Use 'uigetdir' to allow the user to pick the folder containing the data. [20 points]

   **Answer:** This is the well-known ICP (Iterated Closest Point) Algorithm. The method of computation of $\boldsymbol{R}$ and $\hat{\boldsymbol{t}}$ is detailed in the solution to problem 4. The knnsearch function from MATLAB is quite well suited to nearest neighbor computation. You can find MATLAB implementation of ICP at `https://in.mathworks.com/help/vision/ref/pcregrigid.html` or at `https://in.mathworks.com/matlabcentral/fileexchange/27804-iterative-closest-point`.

   **Marking Scheme:** Correct implementation of the formulae from problem 4 carries 8 points, 5 points for a good final alignment, 5 points for the final result for $\boldsymbol{R}$ and $\hat{\boldsymbol{t}}$ which can be computed using least squares on the final set of correspondences, i.e. in the last step. Deduct 2 points for not using 'uigetdir', deduct 2 point for not displaying the point overlay and deduct 1 point for not including the plots in the report.

2. Enlist any four differences between the camera calibration algorithms EXPL_PARS_CAL and PROJ_MAT_CALIB from chapter 6 of the Trucco and Verri book. [10 points]

**Answer:** The former algorithm requires $N \geq 7$ world-image point-pairs whereas the latter requires $N \geq 6$ world-image point-pairs. The former requires prior computation of the optical center using the orthocenter theorem, whereas the latter infers the optical center within the algorithm itself. The former sets up one equation per corresponding world-image point-pair which represents a ratio of the numerators for the expressions for the $x$ and $y$ coordinates, whereas the latter algorithm uses two equations per corresponding world-image point-pair. The former algorithm infers the individual parameters such as $f/s_x, f/sy, \boldsymbol{R}, \hat{\boldsymbol{t}}$ whereas the latter infers the calibration matrix directly first and then infers the individual parameters. (Aside: Both are examples of least squares algorithms, and both employ a post-hoc correction for the estimation of $\boldsymbol{R}$.)

**Marking scheme:** 2.5 points per point of difference. Other sensible differences also given full credit.

3. In the camera calibration algorithm we studied in class, it turns out that the estimate of the rotation matrix (let's call it $\hat{\boldsymbol{R}}$) is not orthonormal. The book by Trucco and Verri suggests the following procedure to 'correct' this issue by replacing $\hat{\boldsymbol{R}}$ by $\tilde{\boldsymbol{R}} = \boldsymbol{U}\boldsymbol{V}^T$ where $\hat{\boldsymbol{R}} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$ is the SVD of $\hat{\boldsymbol{R}}$. Prove that $\tilde{\boldsymbol{R}}$ as obtained by this procedure is given as $\tilde{\boldsymbol{R}} = \text{argmin}_{\boldsymbol{Q}} \|\boldsymbol{Q} - \hat{\boldsymbol{R}}\|_F^2$ subject to the constraint that $\boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{I}$ (In other words, prove that $\tilde{\boldsymbol{R}}$ is the orthonormal matrix closest to $\hat{\boldsymbol{R}}$ in the Frobenius sense). Also this correction step brings out a limitation of this camera calibration algorithm. State that limitation. [7+3 = 10 points]

**Solution:** We are basically searching for an orthonormal matrix $\boldsymbol{Q}$ which is closest (in the Frobenius sense) to matrix $\hat{\boldsymbol{R}}$. Thus we seek to minimize $E(\boldsymbol{Q}) = \|\boldsymbol{Q} - \hat{\boldsymbol{R}}\|_F^2$. The minimum of this is equal to the maximum of $F(\boldsymbol{Q}) = \text{trace}(\boldsymbol{Q}^T\hat{\boldsymbol{R}})$. Now using SVD, we express $\hat{\boldsymbol{R}} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$, which gives us $F(\boldsymbol{Q}) = \text{trace}(\boldsymbol{Q}^T\boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T) = \text{trace}(\boldsymbol{S}\boldsymbol{V}^T\boldsymbol{Q}^T\boldsymbol{U}) = \text{trace}(\boldsymbol{S}\boldsymbol{Z})$ where $\boldsymbol{Z} = \boldsymbol{V}^T\boldsymbol{Q}^T\boldsymbol{U}$ is orthonormal. As seen in class, this reduces to $\sum_i \boldsymbol{S}_{ii}\boldsymbol{Z}_{ii}$ which is maximized when $\boldsymbol{Z}$ is the identity matrix. This produces $\boldsymbol{I} = \boldsymbol{V}^T\boldsymbol{Q}^T\boldsymbol{U}$, yielding $\boldsymbol{Q} = \boldsymbol{U}\boldsymbol{V}^T$. The camera calibration algorithm solves a least squares problem which yields us a non-orthonormal estimate of the rotation matrix. We correct for this non-orthonormality using this SVD-based method. However, the resulting solution for the rotation matrix is no more guaranteed to be a least squares optimal solution! Therefore this correction factor is ad-hoc.

**Marking Scheme:** 3 points for stating the limitation. 2 points for the step for maximizing $F(\boldsymbol{Q})$. 2 points for the step on maximizing $\sum_i \boldsymbol{S}_{ii}\boldsymbol{Z}_{ii}$ and 3 points for the final answer $\boldsymbol{Q} = \boldsymbol{U}\boldsymbol{V}^T$.

4. Consider two sets of corresponding points $\{\boldsymbol{p}_{1i} = (x_{1i}, y_{1i})\}_{i=1}^n$ and $\{\boldsymbol{p}_{2i} = (x_{2i}, y_{2i})\}_{i=1}^n$. Assume that each pair of corresponding points is related as follows: $\boldsymbol{p}_{2i} = \alpha\boldsymbol{R}\boldsymbol{p}_{1i} - \hat{\boldsymbol{t}} + \boldsymbol{\eta}_i$ where $\boldsymbol{R}$ is an unknown rotation matrix, $\hat{\boldsymbol{t}}$ is an unknown translation vector, $\alpha$ is an unknown scalar factor and $\boldsymbol{\eta}_i$ is a vector (unknown) representing noise. Explain how you will extend the method we studied in class for estimation of $\boldsymbol{R}$ to estimate $\alpha$ and $\hat{\boldsymbol{t}}$ as well. Derive all necessary equations (do not merely guess the answers even if they appear 'correct'). [20 points]

**Solution:** Define $\hat{\boldsymbol{t}} = -\hat{\boldsymbol{t}}$. Let $\boldsymbol{P}_1$ be the $2 \times N$ matrix whose $i$-th column contains $\boldsymbol{p}_{1i}$. Likewise, define $\boldsymbol{P}_2$.
We can find $\hat{\boldsymbol{t}}$ by minimizing $E(\hat{\boldsymbol{t}}) = \sum_{i=1}^N \|\boldsymbol{p}_{1i} - \alpha\boldsymbol{R}\boldsymbol{p}_{2i} - \hat{\boldsymbol{t}}\|^2$, which gives $\sum_{i=1}^N(\boldsymbol{p}_{1i} - \alpha\boldsymbol{R}\boldsymbol{p}_{2i} - \hat{\boldsymbol{t}}) = 0$, i.e. we get $\hat{\boldsymbol{t}} = \frac{1}{N}(\sum_{i=1}^N \boldsymbol{p}_{1i} - \alpha\boldsymbol{R}\sum_{i=1}^N \boldsymbol{p}_{2i}) = \bar{\boldsymbol{p}}_1 - \alpha\boldsymbol{R}\bar{\boldsymbol{p}}_2$ where $\bar{\boldsymbol{p}}_1$ and $\bar{\boldsymbol{p}}_2$ are the average of all the points in sets $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ respectively. But we need $\alpha$ and $\boldsymbol{R}$ to find $\hat{\boldsymbol{t}}$, so what do we do? Observe that $\sum_{i=1}^N \|\boldsymbol{p}_{1i} - \alpha\boldsymbol{R}\boldsymbol{p}_{2i} - \hat{\boldsymbol{t}}\|^2 = \sum_{i=1}^N \|\boldsymbol{p}_{1i} - \alpha\boldsymbol{R}\boldsymbol{p}_{2i} - \bar{\boldsymbol{p}}_1 + \alpha\boldsymbol{R}\bar{\boldsymbol{p}}_2\|^2 = \sum_{i=1}^N \|(\boldsymbol{p}_{1i} - \bar{\boldsymbol{p}}_1) - \alpha\boldsymbol{R}(\boldsymbol{p}_{2i} - \bar{\boldsymbol{p}}_2)\|^2$. Now solve for $\boldsymbol{R}$ from the SVD of $\bar{\boldsymbol{P}}_2\bar{\boldsymbol{P}}_1^T$ where $\bar{\boldsymbol{P}}_1$ is the $2 \times N$ matrix whose $i$-th column contains $\boldsymbol{p}_{1i} - \bar{\boldsymbol{p}}_1$ (likewise $\bar{\boldsymbol{P}}_2$). In other words, $\boldsymbol{R} = \boldsymbol{V}\boldsymbol{U}^T$ where $\boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T = \bar{\boldsymbol{P}}_2\bar{\boldsymbol{P}}_1^T$.

What about $\alpha$? Realize that it is only a scaling factor which will get absorbed in the $\boldsymbol{S}$ matrix. In other words if $\boldsymbol{X}$ has SVD given by $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$, then theSVD of $\alpha\boldsymbol{X}$ is given by $\alpha\boldsymbol{X} = \boldsymbol{U}\alpha\boldsymbol{S}\boldsymbol{V}^T$. We can now solve for $\alpha$ by minimizing $\sum_{i=1}^N \|(\boldsymbol{p}_{1i} - \bar{\boldsymbol{p}}_1) - \alpha\boldsymbol{R}(\boldsymbol{p}_{2i} - \bar{\boldsymbol{p}}_2)\|^2$. This yields $\alpha = \dfrac{\sum_i \boldsymbol{p}_{1i}\boldsymbol{R}\boldsymbol{p}_{2i}}{\sum_i(\boldsymbol{R}\boldsymbol{p}_{2i})^2}$. Given $\alpha$ and

$\boldsymbol{R}$, we can now find $\hat{\boldsymbol{t}}$ and hence $\boldsymbol{t}$.

Marking scheme: 4 points for derivative w.r.t. $\hat{\boldsymbol{t}}$ and the expression for $\hat{\boldsymbol{t}}$ in terms of $\alpha$ and $\boldsymbol{R}$. 4 points for substituting this expression towards solving for $\boldsymbol{R}$. 8 points for the final expression for $\boldsymbol{R}$ out of which 4 points are for realizing that the $\alpha$ terms gets absorbed in the singular values. 4 points for the final answer for $\alpha$ (only 2 points for a description of the approach without final expression). Some students may substitute the parametric form for $\alpha\boldsymbol{R}$ in terms of $\alpha$, $q_1 = \cos\theta$ and $q_2 = \sqrt{1-q_1^2} = \sin\theta$. This will produce quadratic equations. This is a sensible approach (but inefficient and inaccurate), but we will not grant it full credit (deduct 5 points) as the question specifically refers to the method done in class - which is basically the SVD method. Deduct 2 points if the students have not paid attention to the relation between $\boldsymbol{t}$ and $\hat{\boldsymbol{t}}$.

5. In this task, we will register two pairs of images with each other: (1) The famous barbara image (regarded as a fixed image) to be registered with its negative (regarded the moving image), and (2) a flash image (regarded as a fixed image) and a no-flash image (regarded as the moving image) of a scene. We will use the joint entropy criterion we studied in class as the objective function to be minimized for alignment. Download all required images from the homework folder. Convert all images to gray-scale (if they are in color). Note that the flash image and the no-flash image have different image intensities at many places, and the no-flash image is distinctly noisier. In the beginning you may want to either downsample or work with smaller portions of the flash and no-flash images.

   For each of the two cases, rotate the moving image counter-clockwise by 23.5 degrees, translate it by -3 pixels in the X direction, and add uniform random noise in the range [0,8] (on a 0-255 scale). Note that the rotation must be applied about the center of the image. Set negative-valued pixels to 0 and pixels with value more than 255 to 255. Now perform a brute-force search to find the angle $\theta$ and translation $t_x$ to optimally align the modified moving image with the fixed image (in each case), so as to minimize the joint entropy. The range for $\theta$ should be between -60 and +60 in steps of 1 degree, and the range for $t_x$ should be between -12 and +12 in steps of 1. Compute the joint entropy using a bin-size of 10 for both intensities. Plot the joint entropy as a function of $\theta$ and $t_x$ using the surf and imshow commands of MATLAB. Comment on the difference (if any) between the quality of alignment for the first and second pair of images. Use 'uigetdir' to allow the user to input the images to be registered.

   Also, determine a scenario (for the first pair of images) where the images are obviously misaligned but the joint entropy is (falsely and undesirably) lower than the 'true' minimum. Again, display the joint entropy as mentioned before. Include all plots in your report. [20 points]

   **Solution:** The code is in the homework folder. The undesirable minimum is obtained for translation parameters yielding a very trivial overlap between the two images. For the image and its negative, as well as the flash and no-flash pair, the results of registration are fine.

   **Marking Scheme:** 7 points for correct implementation of joint entropy including handling of intensities falling outside the [0,255] range, either by clipping or rescaling. 6 points for the double for loop that includes a brute-force search over rotation and translation. 3 points for plotting the joint entropy function. 4 points for plotting the case of the undesirable minimum. If the results of registration are not inaccurate, please check the correctness of joint entropy calculation.

6. Refer to the paper 'Goal-directed video metrology' by Reid and Zisserman from `http://www.robots.ox.ac.uk/~vgg/publications/papers/reid96.pdf` which is an excellent and interesting application of visual metrology, and which is believed to have solved a long-standing controversy in a famous football match. The paper proposes an algorithm that takes as input two images $I$ and $I'$ (taken from two viewpoints) of a portion of the football field near a goal-post. Let $P$ be the point of intersection of a line passing through an arbitrary point in 3D (say corresponding to the location of the football in mid-air) and striking the ground plane perpendicular to it. The paper aims to predict the location $p$ and $p'$ of the image of $P$ inside $I$ and $I'$ respectively. Your task is to read and understand section 2 and figure 2 of the paper, and answer the following questions:

(a) How does the homography computation help in obtaining $p$ and $p'$?

(b) Why is it important to compute the vertical vanishing point $v$ and $v'$ in the two images?

(c) What is the significance of the lines $L_s$ and $L'_s$ in the task of obtaining $p$ and $p'$? [10 points]

**Solution:** These answers are quite straightforward once you understand the methodology. Refer to figure 2 of the paper. The aim is to determine the image of point $\boldsymbol{P}$ in the two pictures (or any one of them). Let $\boldsymbol{v}$ and $\boldsymbol{v'}$ be the vertical vanishing points in the two pictures (the goal posts are considered vertical). Let $\boldsymbol{B}$ be the physical location of the football. The images of the vertical line $\boldsymbol{BP}$ in the two pictures are $\boldsymbol{vb}$ and $\boldsymbol{v'b'}$ respectively. Now $\boldsymbol{vb}$ can be considered to be the image of line $\boldsymbol{L_s}$, the shadow of line $\boldsymbol{BP}$, in the first picture. Likewise $\boldsymbol{v'b'}$ can be considered to be the image of line $\boldsymbol{L'_s}$, the shadow of line $\boldsymbol{BP}$, in the second picture. In 3D space, the shadow lines intersect at point $\boldsymbol{P}$ whose image we are interested in. We need the intersection of two lines in order to find $\boldsymbol{p}$ in the first picture. If we had the image (denoted as $\boldsymbol{s}$) of $\boldsymbol{L'_s}$ in the first picture, then we could easily find $\boldsymbol{p}$ from the intersection of $\boldsymbol{s}$ and $\boldsymbol{vb}$. But how do we get $\boldsymbol{s}$? For that, we use the planar homography between the two pictures! Corresponding points on $\boldsymbol{s}$ in the first picture and $\boldsymbol{v'b'}$ in the second picture are related by this homography! Now, we are equipped to answer the questions:

(a) How does the homography computation help in obtaining $\boldsymbol{p}$ and $\boldsymbol{p'}$? Because $\boldsymbol{p}$ is the intersection of lines $\boldsymbol{s}$ and $\boldsymbol{vb}$ in the first picture. Line $\boldsymbol{s}$ is obtained by applying the homography to the line $\boldsymbol{v'b'}$ in the second picture. Note that $\boldsymbol{s}$ and $\boldsymbol{v'b'}$ are images of line $\boldsymbol{L'_s}$ in the first and second pictures respectively.

(b) Why is it important to compute the vertical vanishing point $\boldsymbol{v}$ and $\boldsymbol{v'}$ in the two images? We need two points to produce the vertical line in the image planes. One of these points is $\boldsymbol{b}$ or $\boldsymbol{b'}$ and the other could be $\boldsymbol{v}$ and $\boldsymbol{v'}$, or $\boldsymbol{p}$ and $\boldsymbol{p'}$. But we don't know $\boldsymbol{p}$ and $\boldsymbol{p'}$ because the ball is in mid-air! On the other hand, we can compute $\boldsymbol{v}$ and $\boldsymbol{v'}$ from the images of the vertical bars of the goal post.

(c) What is the significance of the lines $\boldsymbol{L_s}$ and $\boldsymbol{L'_s}$ in the task of obtaining $\boldsymbol{p}$ and $\boldsymbol{p'}$? Lines $\boldsymbol{L_s}$ and $\boldsymbol{L'_s}$ intersect at $\boldsymbol{P}$ in 3D space. Hence the images of these lines will intersect at $\boldsymbol{p}$ in the first picture and $\boldsymbol{p'}$ in the second picture.

**Marking scheme:** 3 points each for a sensible explanation and 1 is a bonus point.

7. Consider a picture of a static (possibly non-planar) scene acquired by a camera fixed on a tripod. Now the camera is rotated but it remains fixed on the tripod without any translation, and another picture of the same scene is acquired. Let $\boldsymbol{p_1}$ and $\boldsymbol{p_2}$ be the pixel coordinates of the images of some physical point in the scene in the two pictures respectively. Note that $\boldsymbol{p_1}$ and $\boldsymbol{p_2}$ are in different coordinate systems. Derive a relation between $\boldsymbol{p_1}$ and $\boldsymbol{p_2}$ in terms of the matrix $\boldsymbol{R}$ which represents the rotational motion of the camera axes from the first position to the second, and the intrinsic parameter matrix $\boldsymbol{K_1}$ and $\boldsymbol{K_2}$ of the cameras in the two viewpoints. Note that the instrinsic parameters could change if you changed the focal length, or (hypothetically) the resolution. [10 points]

**Solution:** Consider $\boldsymbol{p_{1h}} = \boldsymbol{K_1}(\boldsymbol{R_1}|\boldsymbol{t_1})\boldsymbol{P}$ in homogeneous coordinates corresponding to the point $\boldsymbol{p_1}$ in pixel coordinates. Now we know $\boldsymbol{p_{2h}} = \boldsymbol{K_2}\boldsymbol{R}(\boldsymbol{R_1}|\boldsymbol{t_1})\boldsymbol{P} = \boldsymbol{K_2}\boldsymbol{R}\boldsymbol{K_1}^{-1}\boldsymbol{p_{1h}} = \boldsymbol{H}\boldsymbol{p_{1h}}$ where $\boldsymbol{H}$ is a $3 \times 3$ homography matrix (not a planar homography matrix but a homography matrix of a different kind - a rotational homography matrix). Given a single pair of corresponding points $\boldsymbol{p_1}$ and $\boldsymbol{p_2}$ it is not possible to determine $\boldsymbol{H}$, but it is possible to determine this matrix using $N \geq 8$ pairs of such points using our usual homography estimation algorithm.

**Marking scheme:** 4 points for the expression for $\boldsymbol{p_{1h}}$, 2 points for the expression for $\boldsymbol{p_{1h}}$, and 4 points for the final expression.