Assignment 2: CS 763, Computer Vision

Due: 17th February 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. Put the pdf file and the code for the programming parts all in one zip file. The pdf should contain the names and ID numbers of all students in the group within the header. The pdf file should also contain instructions for running your code. Name the zip file 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 17th February. Late assignments will not be accepted due to the upcoming midterms since the solutions need to be released. 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. We have defined the concept of the Shannon entropy in class. Given a discrete random variable X having probability mass function $P(X) = (p_1, p_2, ..., p_N)$, prove that $H(X) \geq 0$ where $H(X)$ is the Shannon entropy of X. Recall that $H(X) = -\sum_{i=1}^{N} p_i \log p_i$ and that $\sum_{i=1}^{N} p_i = 1$ and $\forall i, 0 \leq p_i \leq 1$. Also prove that a uniform distribution ($ie\ \forall i,\ p_i = \frac{1}{N}$) maximizes the Shannon entropy. To this end, find a stationary point of $J(X) = H(X) - \lambda(\sum_{i=1}^{N} p_i - 1)$ where λ is a Lagrange multiplier to impose the hard constraint that the probabilities all sum up to 1. [2+4 = 6 points]

   Solution: For the first part, observe that $H(X) = -\sum_{i=1}^{N} p_i \log p_i$. As each p_i satisfied $0 \leq p_i \leq 1$, we have $\log p_i \leq 0$. Hence for every i, we have $p_i \log p_i \leq 0$, and hence the negative summation of such terms in non-negative.

   For the second part, consider $J(p_i) = -\sum_{i=1}^{N} p_i \log p_i - \lambda(\sum_{i=1}^{N} p_i - 1)$. Setting the derivative w.r.t. p_k to 0 gives us, $-(1 + \log p_k) - \lambda = 0$. Rearranging this, gives us $p_k = e^{-\lambda - 1}$. As $\sum_{k=1}^{N} p_k = 1$, we have $1 = Ne^{-\lambda-1}$, which gives $\lambda = \log N - 1$ and hence $p_k = e^{-\log N} = \frac{1}{N}$. Note that the first derivative test in this case yields the maximum (and not the minimum) because $\frac{\partial^2 J}{\partial p_k^2} < 0$ for all k.

   What would you do if you had to find the probability mass function (pmf) with the minimum entropy? Intuitively, it is the pmf in which $p_i = 1$ for some i and $p_i = 0$ for all other i and the entropy is 0 (its least possible value). This can also be reasoned out by noting that the entropy is a function of variables $\{p_i\}_{i=1}^{N}$ defined over a bounded domain called as a 'standard N-simplex', which is defined as $\Delta^N = \{(p_1, p_2, ..., p_N) \in \mathbb{R}^N | \sum_{i=1}^{N} p_i = 1 \text{ and } p_i \geq 0 \text{ for all } i\}$. The minimum will occur at one or more of the corners of the simplex which are precisely points with exactly one coordinate being one and the rest being zero.

2. This is a straightforward exercise to make sure you understand the basic update equations in the Horn-Shunck algorithm for optical flow. As seen in class, we seek to minimize the quantity $J(\{(u_{ij}, v_{ij})\})$ w.r.t. the optical flow vectors $(u_{i,j}, v_{i,j})$ at all pixels (i, j), where $J(\{(u_{i,j}, v_{i,j})\}) = \sum_{i=1}^{N}\sum_{j=1}^{N}(I_{x;i,j}u_{i,j} + I_{y;i,j}v_{i,j} + I_{t;i,j})^2 + \lambda((u_{i,j+1} - u_{i,j})^2 + (u_{i+1,j} - u_{i,j})^2 + (v_{i,j+1} - v_{i,j})^2 + (v_{i+1,j} - v_{i,j})^2)$. Setting the partial derivatives w.r.t. $u_{k,l}$ and $v_{k,l}$ to 0, prove that

$$u_{k,l} = \bar{u}_{k,l} - \frac{I_{x;k,l}(I_{x;k,l}\bar{u}_{k,l} + I_{y;k,l}\bar{v}_{k,l} + I_{t;k,l})}{I_{x;k,l}^2 + I_{y;k,l}^2 + 4\lambda} \tag{1}$$

$$v_{k,l} = \bar{v}_{k,l} - \frac{I_{y;k,l}(I_{x;k,l}\bar{u}_{k,l} + I_{y;k,l}\bar{v}_{k,l} + I_{t;k,l})}{I_{x;k,l}^2 + I_{y;k,l}^2 + 4\lambda} \tag{2}$$

where $\bar{u}_{k,l}$ and $\bar{v}_{k,l}$ are as defined in the lecture slides. Also verify the Jacobi update equations given by the following:

$$u_{k,l}^{(t+1)} = \bar{u}_{k,l}^{(t)} - \frac{I_{x;k,l}(I_{x;k,l}\bar{u}_{k,l}^{(t)} + I_{y;k,l}\bar{v}_{k,l}^{(t)} + I_{t;k,l})}{I_{x;k,l}^2 + I_{y;k,l}^2 + 4\lambda} \tag{3}$$

$$v_{k,l}^{(t+1)} = \bar{v}_{k,l}^{(t)} - \frac{I_{y;k,l}(I_{x;k,l}\bar{u}_{k,l}^{(t)} + I_{y;k,l}\bar{v}_{k,l}^{(t)} + I_{t;k,l})}{I_{x;k,l}^2 + I_{y;k,l}^2 + 4\lambda} \tag{4}$$

[6 points]
Solution: See lecture slides. Note one important point: λ here is a regularization parameter which encourages smoothness. It is not to be confused with the Lagrange multiplier λ which is used to impose a hard constraint. A hard constraint means the solution is not allowed to disobey the constraint. For example, all probability values must sum to one. A regularization parameter is by definition much softer - it merely acts as a penalty term, but does not impose the constraint.

3. You know that both the Horn-Shunck as well as Lucas-Kanade methods bank on the brightness constancy assumption. Given a pair of images, let us suppose that this assumption holds good for most physically corresponding pixels, but not for some $p\%$ of the pixels. Briefly explain how you will modify the Horn-Shunck method and Lucas-Kanade method to deal with this. [3+3 = 6 points]
Solution: For the L-K method, you can use RANSAC or LMedS instead of least-squares. For Horn and Shunck, you can minimize the following functional instead of the conventional one discussed in class: $J(u, v) = \int\int_{\Omega}(|I_x u + I_y v + I_t| + \lambda(u_x^2 + u_y^2 + v_x^2 + v_y^2))dxdy$. This penalizes the absolute value of the term $I_x u + I_y v + I_t$ instead of its square, yielding more robustness as the L_1 norm is more robust to outliers than the L_2 norm.

4. In the first camera calibration we studied in class, it turns out that the estimate of the rotation matrix (let's call it $\hat{\mathbf{R}}$) is not orthonormal. The book by Trucco and Verri suggests the following procedure to 'correct' this issue by replacing $\hat{\mathbf{R}}$ by $\tilde{\mathbf{R}} = \mathbf{U}\mathbf{V}^T$ where $\hat{\mathbf{R}} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ is the SVD of $\hat{\mathbf{R}}$. Prove that $\tilde{\mathbf{R}}$ as obtained by this procedure is given as $\tilde{\mathbf{R}} = \text{argmin}_{\mathbf{Q}}\|\mathbf{Q} - \hat{\mathbf{R}}\|_F^2$ subject to the constraint that $\mathbf{Q}\mathbf{Q}^T = \mathbf{I}$. Also this correction step brings out a limitation of this camera calibration algorithm. State that limitation. [5+1 = 6 points]
Solution: We are basically searching for an orthonormal matrix $\mathbf{Q}$ which is closest (in the Frobenius sense) to matrix $\hat{\mathbf{R}}$. Thus we seek to minimize $E(\mathbf{Q}) = \|\mathbf{Q} - \hat{\mathbf{R}}\|_F^2$. The minimum of this is equal to the maximum of $F(\mathbf{Q}) = \text{trace}(\mathbf{Q}^T\hat{\mathbf{R}})$. Now using SVD, we express $\hat{\mathbf{R}} = \mathbf{U}\mathbf{S}\mathbf{V}^T$, which gives us $F(\mathbf{Q}) = \text{trace}(\mathbf{Q}^T\mathbf{U}\mathbf{S}\mathbf{V}^T) = \text{trace}(\mathbf{S}\mathbf{V}^T\mathbf{Q}^T\mathbf{U}) = \text{trace}(\mathbf{S}\mathbf{Z})$ where $\mathbf{Z} = \mathbf{V}^T\mathbf{Q}^T\mathbf{U}$ is orthonormal. As seen in class, this reduces to $\sum_i \mathbf{S}_{ii}\mathbf{Z}_{ii}$ which is maximized when $\mathbf{Z}$ is the identity matrix. This produces $\mathbf{I} = \mathbf{V}^T\mathbf{Q}^T\mathbf{U}$, yielding $\mathbf{Q} = \mathbf{U}\mathbf{V}^T$.

The camera calibration algorithm solves a least squares problem since we are minimizing $\|\mathbf{A}\mathbf{v}\|_2^2$ w.r.t. $\mathbf{v}$ subject to the constraint $\mathbf{v}^t\mathbf{v} = 1$. This may yield us a non-orthonormal estimate of the rotation matrix. We correct for this non-orthonormality using this SVD-based method in an after-the-fact manner. However, the resulting solution for the rotation matrix (after the correction) is no more guaranteed to be a least squares

optimal solution! Therefore this correction process is ad-hoc. Ideally one would have desired a procedure which imposes the orthonormality of the rotation matrix within the estimate of $\mathbf{v}$, but I have not seen this method anywhere.

5. The input to the Tomasi-Kanade factorization algorithm for structure from motion is the set of x and y coordinates of $n \geq 4$ points $\{x_{ij}\}, \{y_{ij}\}$, corresponding to unknown non-coplanar 3D points $\{\mathbf{P}_j\}_{j=1}^n$ on a rigidly moving object, and tracked in $N \geq 3$ different images (or frames), $1 \leq j \leq n, 1 \leq i \leq N$. The images are acquired under orthographic projection. The algorithm proceeds by performing an SVD of the $2N \times n$ matrix $\tilde{\mathbf{W}} = \begin{pmatrix} \tilde{\mathbf{X}} \\ \tilde{\mathbf{Y}} \end{pmatrix}$ where $\tilde{\mathbf{X}}_{ij} = x_{ij} - \frac{1}{n}\sum_{j=1}^n x_{ij}, \tilde{\mathbf{Y}}_{ij} = y_{ij} - \frac{1}{n}\sum_{j=1}^n y_{ij}$. The output of the algorithm consists of (1) the 3D coordinates $\{\mathbf{P}_j\}_{j=1}^n$, and (2) the 3D rotational motion of the object from one frame to another. Now, if the object motion were 3D affine instead of 3D rigid, can the point coordinates $\{\mathbf{P}_j\}_{j=1}^n$ and the affine object motion still be unambiguously estimated by the algorithm? Why (not)? Write down all necessary equations. [6 points]

Ans: Without noise, $\mathbf{W}$ has exact rank 3, otherwise its rank is approximately 3. This is evident from the factorization:

$$x_{ij} = \mathbf{i}_i^T(\mathbf{P}_j - \mathbf{T}_i) \tag{5}$$
$$y_{ij} = \mathbf{j}_i^T(\mathbf{P}_j - \mathbf{T}_i) \tag{6}$$

where $\mathbf{T}_i$ is the vector from the origin of the world coordinate system to the origin of the coordinate system of frame i, and $\mathbf{i}_i, \mathbf{j}_i$ represent the first and second rows of the rotation matrix of frame i (i.e., vectors representing the coordinate axes of frame i as expressed in the world coordinate system). Combining all such equations together and deducting the mean point of every frame, we get the factorization: $\mathbf{W} = \mathbf{RS}$ where $\mathbf{R} \in \mathbb{R}^{2N \times 3}$ is the matrix representing the first two rows of the rotation matrix for all N frames, and $\mathbf{S} \in \mathbb{R}^{3 \times n}$ is the matrix of the 3D coordinates of all n points. Both $\mathbf{R}$ and $\mathbf{S}$ are obtained from the SVD of $\mathbf{W}$.

We have seen in class that $\mathbf{R}$ and $\mathbf{S}$ are known only up to an unknown invertible 3×3 transformation $\mathbf{Q}$ since $\mathbf{RS} = \mathbf{RQQ}^{-1}\mathbf{S}$ (see pages 205-207 of Trucco and Verri for more details). In fact $\mathbf{RQ}$ will not be a pure rotation matrix. We obtain $\mathbf{Q}$ by imposing the following set of constraints: the rows of $\mathbf{RQ}$ must have unit magnitude, and the first n rows of $\mathbf{RQ}$ must be orthogonal to the last n rows of $\mathbf{RQ}$. These constraints account for key properties of the rotation matrix. For general affine transformations, no such constraints are available. Hence we cannot find $\mathbf{Q}$ unambiguously. Hence the shape and motion can be estimated only upto an unknown affine transformation in 3D, unlike the case of rigid motion.

However, there is one more even stronger reason why 3D affine motion cannot be estimated. Let us even suppose we were able to estimate $\mathbf{R}$ magically in the case of affine transformation. But it contains only the first two rows of the transformation matrix for every frame. For rotation, the third row is the cross-product of the first two. That is not true for the affine transformation and hence the affine transformation cannot be ascertained.
Marking scheme: 5 points for stating it is rank 3 with a decent attempt at the proof. 5 points for stating that while the rotation matrix satisfies the property $\mathbf{R_3} = \mathbf{R_1} \times \mathbf{R_2}$, the affine matrix does not satisfy this property. Only 3 points given if you say that the metric constraints cannot be imposed to dsiambiguate the decomposition.

6. In this task, you will build up on the previous assignment to estimate the homography between two images. This time, you should use the RanSaC algorithm to estimate the homography in order to make the estimate resistant to the presence of incorrect point correspondences. The code for RanSaC for various problems including estimation of homographies is available at `http://www.csse.uwa.edu.au/~pk/research/matlabfns/`. You should work with the images in the folder `http://www.cse.iitb.ac.in/~ajitvr/CS763_Spring2016/HW2/Homography` and also on any one pair of pictures of an approximately planar scene taken with a real camera. (You should acquire these images yourself and make sure they have small non-overlapping areas to make this more interesting.) Also, in each case, you should warp one of the images so that it aligns with the other one. However, this time you should not crop the image so that no parts of either image are

deleted, and a true mosaic may be obtained. While you should use the RanSaC code as is, I recommend stepping through it to get a feel for what is going on inside. Display all the resulting mosaics in your report. State the number of iterations and all thresholds you used. The code performs a normalization step before computing the homography. Can you guess the reason for this normalization step? [5+3+2 = 10 points]

Solution: The reason for the normalization step is documented at `http://www.cs.cmu.edu/afs/andrew/scs/cs/15-463/f07/proj_final/www/amichals/fundamental.pdf` - it's explained in the context of the fundamental matrix, a concept we will study in stereo vision later on in the course, but the basic principles apply to homography as well. The normalization step basically transforms the x and y coordinates of the points in the first image such that the centroid of these points is 0 and they range from -1 to +1. Let this transformation matrix be called $\boldsymbol{T_1}$. The same procedure is repeated for the points of the second image yielding transformation matrix $\boldsymbol{T_2}$. The homography is computed using these normalized points in the form $\boldsymbol{T_1 p_1} = \mathbf{H}\boldsymbol{T_2 p_2}$, and the effective homography between the non-normalized points is then estimated as $\mathbf{H}_{final} = \boldsymbol{T_1}^{-1}\mathbf{H}\boldsymbol{T_2}\mathbf{p_2}$.

7. 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 `http://www.cse.iitb.ac.in/~ajitvr/CS763_Spring2016/HW2/ImageReg`. 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.

   For each of the two cases, rotate the moving image counter-clockwise by 28.5 degrees, translate it by -2 pixels in the X direction, and add Gaussian noise of standard deviation 10 (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 θ 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 θ 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 θ 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.

   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. [6+2+2 = 10 points]

   Solution: The code for this is in the homework folder. The case of the flash image (F) and no-flash image (G) pair is strange. Although the intensities in F and G at corresponding locations are not really functions of each other (in either direction), the joint entropy method still works fairly well - even under noise. It's just that the optimal alignment configuration yields a lower joint entropy than other configurations.

   A typical scenario where you will get a very low joint entropy even when the images are obviously not well registered is as follows: consider a large translation between the two images which allows for just a one-pixel overlap between them. This problem is called the field of view problem and haunts any algorithm for image registration that uses image similarity measures - be it mean squared error, joint entropy or anything else.