Computer Vision Assignment 2

Maitreyee Mulawkar 13D170011 Anand Bhoraskar 130050025
Lokit Kumar Paras 130050047

February 11, 2017

Honor Code : We have not referred or shared our solutions with any other team.

1

$$R = \begin{bmatrix} 0.9962 & 0.0001 & 0.0874 \\ -0.0011 & 0.9999 & 0.0115 \\ -0.0874 & -0.0115 & 0.9961 \end{bmatrix}$$

$$t = \begin{bmatrix} -2.5097 \\ -0.3123 \\ -5.5828 \end{bmatrix}$$

In each iteration, R^* and t^* are computed using the solution to orthogonal Procrustes problem. When a_i and b_i correspond,
$\bar{a} = (\sum_{i=1}^{n} a_i)/n$ and $\bar{b_2} = (\sum_{i=1}^{n} b_i)/n$
$a1_i = a_i - \bar{a}$ and $b1_i = b_i - \bar{b}$
Taking SVD decomposition of $B1^T A1 = U\Sigma V^T$

$$R^* = V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & det(VU^T) \end{bmatrix} U^T \tag{1}$$

$$t^* = \bar{a} - R^*\bar{b}$$

Net rotation matrix R and net translation t will be initialized to I and zero vector respectively. They will be modified in each iteration as

$$R = R^* R$$

$$t = t^* + R^* t$$

Command in MATLAB used for nearest neighbor computation: **knnsearch** with 'distance' and 'euclidean'

Figure 1: Before first iteration

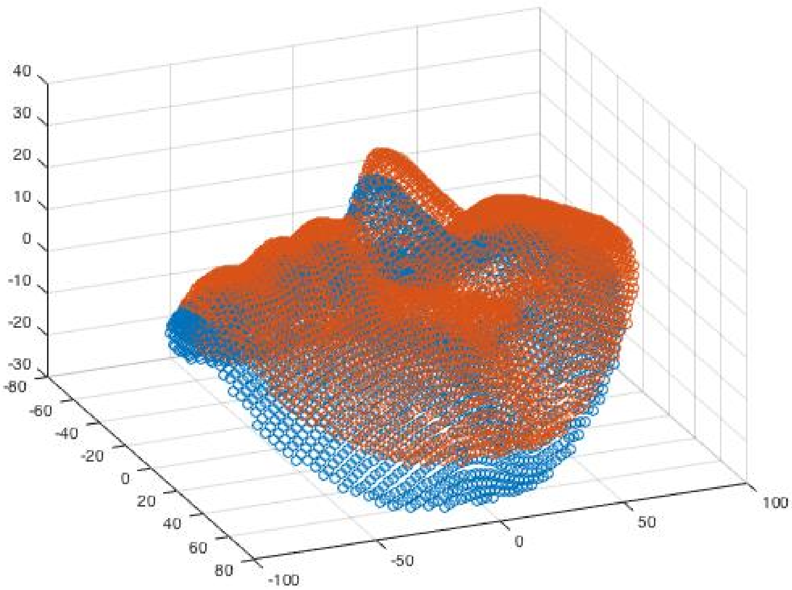
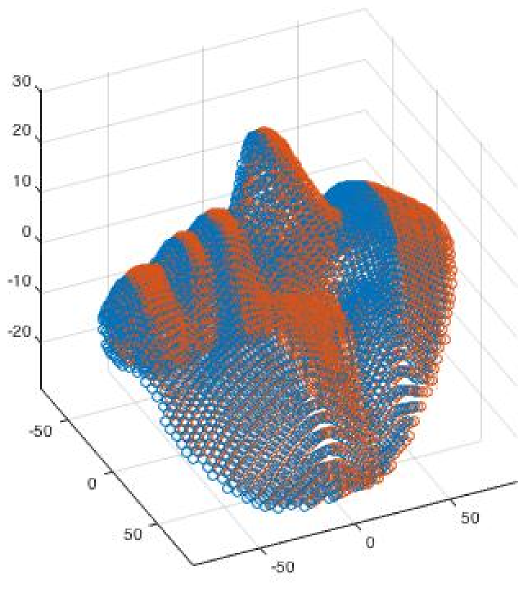


Figure 2: After last iteration



2

1. EXPL_PARS_CALIB requires at least 8 points for calibration wheres PROJ_MAT_CALIB requires atleast 6 points
2. PROJ_MAT_CALIB finds only the projection matrix while EXPL_PARS_CALIB explicitly finds all extrinsic parameters of the camera along with α and f_x.
3. EXPL_PARS_CALIB requires image center which can be estimated using orthocenter property
4. PROJ_MAT_CALIB is faster but can be used only when the projection matrix is sufficient.

3

$\tilde{R} = argmin_Q trace((Q - R)^T(Q - R))$
$\tilde{R} = argmin_Q trace(Q^TQ + R^TR - Q^TR - R^TQ)$
$\tilde{R} = argmin_Q trace(I + R^TR - Q^TR - R^TQ)$
$\tilde{R} = argmin_Q trace(const - Q^TR - R^TQ)$
$\tilde{R} = argmin_Q trace(-Q^TR - R^TQ)$
$\tilde{R} = argmin_Q trace(-Q^TR - R^TQ)$
$\tilde{R} = argmax_Q trace(Q^TR + R^TQ)$
$\tilde{R} = argmax_Q trace(Q^TR + (R^TQ)^T)$
$\tilde{R} = argmax_Q trace(2Q^TR)$
$\tilde{R} = argmax_Q trace(2Q^TUSV^T)$
$\tilde{R} = argmax_Q trace(Q^TUSV^T)$
$\tilde{R} = argmax_Q trace(V^TQ^TUS)$

Now V^T, Q^T and U are orthonormal. Hence max value would be O :
$max_{O^TO=I} trace(OS)$
$= max(S_{11}O_{11} + S_{22}O_{22} + S_{33}O_{33}) \leq S_{11} + S_{22} + S_{33}$
Reason being O is orthonormal and no O_{ij} can be greater than 1 in magnitude and S_{ii} is positive for all i.
This means we get the max for $O = I$
$V^TQ^TU = I$
$Q^T = VU^T$
$Q = UV^T$
Which means that:
$\tilde{R} = UV^T$
However, this means that we find closest orthonormal matrix to the R which was obtained by minimizing the error term without putting an orthonormal constraint on R.
What this means is that had we minimized out error term with the constraint that R is orthonormal, the R we would have obtained would end up being different than $\tilde{R}$. This is the error that this method introduces.

4

Given:
$\{p_{1i} = (x_{1i}, y_{1i})\}_{i=1}^{n}$, $\{p_{2i} = (x_{2i}, y_{2i})\}_{i=1}^{n}$
$p_{2i} = \alpha R p_{1i} + t + \eta_i$, here η_i is the error

We will minimize

$$\boxed{E(R,t,\alpha) = \sum \|\alpha R p_{1i} + t - p_{2i}\|^2} \tag{1}$$

At the minima, $\frac{\partial E}{\partial t} = 0$

$$\frac{\partial E}{\partial t} = \sum_{i=1}^{n} 2(\alpha R p_{1i} + t - p_{2i}) = 0$$

$$2(\alpha R \sum_{i=1}^{n} p_{1i} + tn - \sum_{i=1}^{n} p_{2i}) = 0$$

Let $\bar{p_1} = (\sum_{i=1}^{n} p_{1i})/n$ and $\bar{p_2} = (\sum_{i=1}^{n} p_{2i})/n$. Hence,

$$\boxed{t = \bar{p_2} - \alpha R \bar{p_1}} \tag{2}$$

Using (1) and (2),

$$E(R,\alpha) = \sum_{i=1}^{n} \|\alpha R(p_{1i} - \bar{p_1}) - (p_{2i} - \bar{p_2})\|^2$$

Let $q_{1i} = p_{1i} - \bar{p_1}$ and $q_{2i} = p_{2i} - \bar{p_2}$

$$E(R,\alpha) = \sum_{i=1}^{n} \|\alpha R q_{1i} - q_{2i}\|^2 = \sum_{i=1}^{n} (\alpha R q_{1i} - q_{2i})^T (\alpha R q_{1i} - q_{2i})$$

$$E(R,\alpha) = \sum_{i=1}^{n} (\alpha^2 q_{1i}^T R^T R q_{1i} - \alpha q_{1i}^T R^T q_{2i} - \alpha q_{2i}^T R q_{1i} + q_{2i}^T q_{2i})$$

Here, R is orthogonal matrix. So $R^T R = I$.
Also, $q_{2i}^T \alpha R q_{1i}$ is a scalar. Hence $q_{1i}^T R^T q_{2i} = (q_{2i}^T R q_{1i})^T = q_{2i}^T R q_{1i}$

$$E(R,\alpha) = \sum_{i=1}^{n} (\alpha^2 q_{1i}^T q_{1i} - 2\alpha q_{2i}^T R q_{1i} + q_{2i}^T q_{2i})$$

Since we are interested in minima, $\frac{\partial E}{\partial \alpha} = 0$

$$\frac{\partial E}{\partial \alpha} = \sum_{i=1}^{n} (2\alpha q_{1i}^T q_{1i} - 2 q_{2i}^T R q_{1i}) = 0$$

$$\boxed{\alpha = \frac{\sum_{i=1}^{n} q_{2i}^T R q_{1i}}{\sum_{i=1}^{n} q_{1i}^T q_{1i}}} \tag{3}$$

$$E(R) = \alpha^2 \sum_{i=1}^{n} q_{1i}^T q_{1i} - 2\alpha \sum_{i=1}^{n} q_{2i}^T R q_{1i} + \sum_{i=1}^{n} q_{2i}^T q_{2i}$$

Substituting α from (3),

$$E(R) = \frac{(\sum_{i=1}^{n} q_{2i}^T R q_{1i})^2}{\sum_{i=1}^{n} q_{1i}^T q_{1i}} - 2\frac{(\sum_{i=1}^{n} q_{2i}^T R q_{1i})^2}{\sum_{i=1}^{n} q_{1i}^T q_{1i}} + \sum_{i=1}^{n} q_{2i}^T q_{2i}$$

$$E(R) = -\frac{(\sum_{i=1}^{n} q_{2i}^T R q_{1i})^2}{\sum_{i=1}^{n} q_{1i}^T q_{1i}} + \sum_{i=1}^{n} q_{2i}^T q_{2i}$$

$$argminE(R) = argmin - (\sum_{i=1}^{n} q_{2i}^T R q_{1i})^2 = argmax \sum_{i=1}^{n} q_{2i}^T R q_{1i} = argmax\ tr(Q_2^T R Q_1)$$

We have solved this in class.

$$S = Q_1 Q_2^T = U \Sigma V^T$$

$$\boxed{R = V \begin{bmatrix} 1 & 0 \\ 0 & det(VU^T) \end{bmatrix} U^T} \tag{4}$$

Using (4), (3) and (2), we can estimate R, α and t.

5

Note: Pick the fixed image first and rotating image afterwards

Plots for Barbara:

Figure 3: Joint entropy function for Barbara

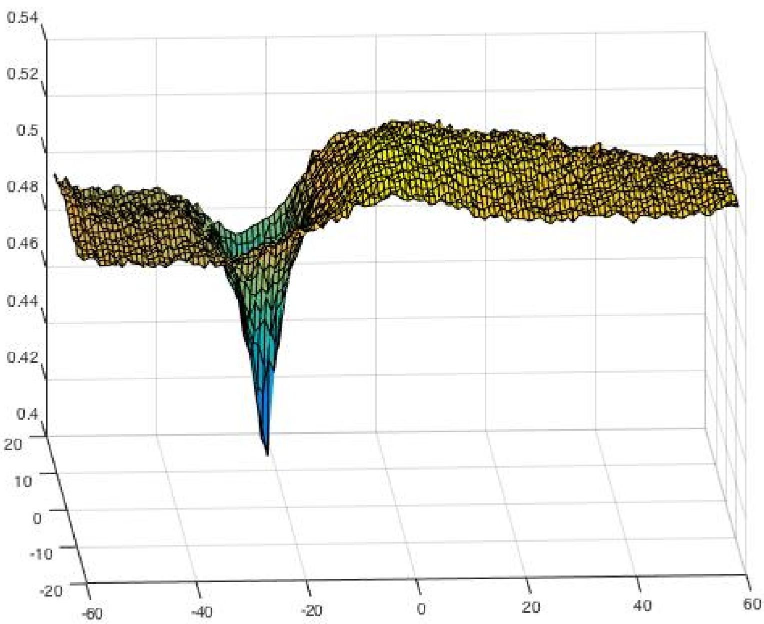


Figure 4: Joint entropy function for Barbara

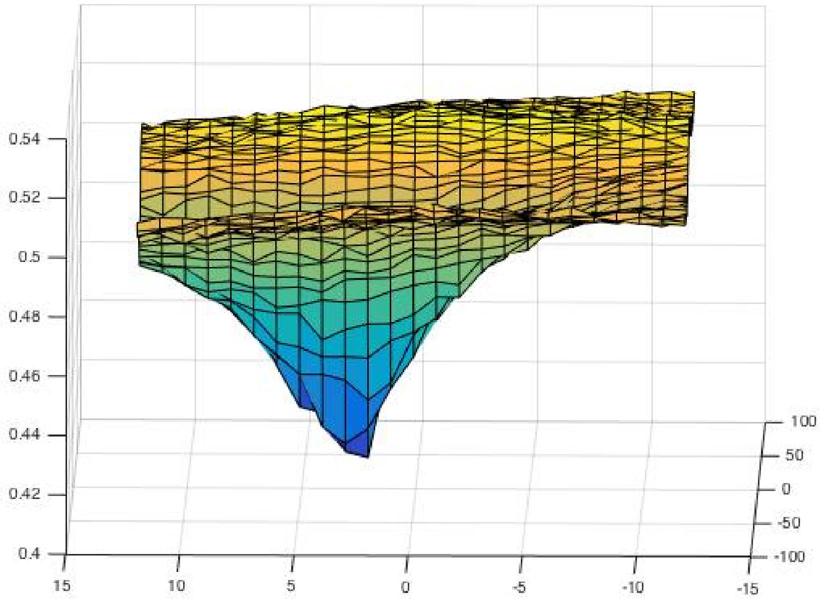
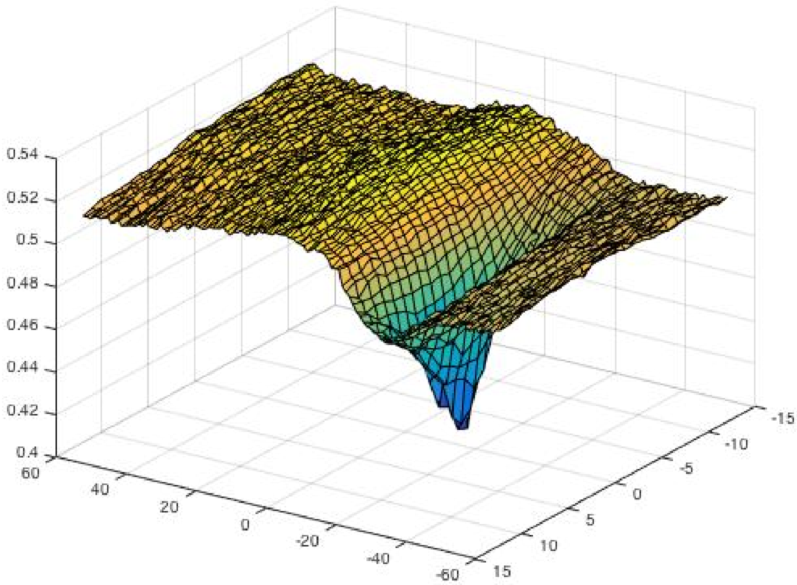


Figure 5: Joint entropy function for Barbara



Plots for flash-noflash

Figure 6: Joint entropy function for flash-noflash

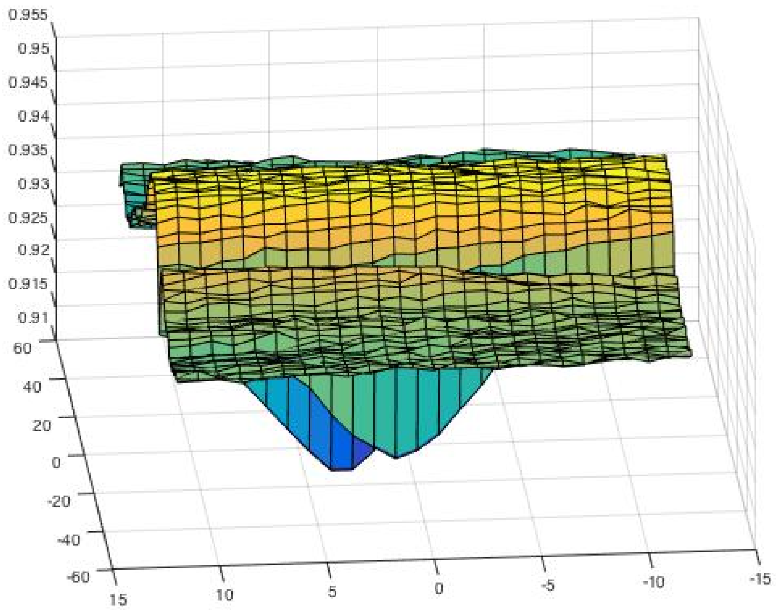
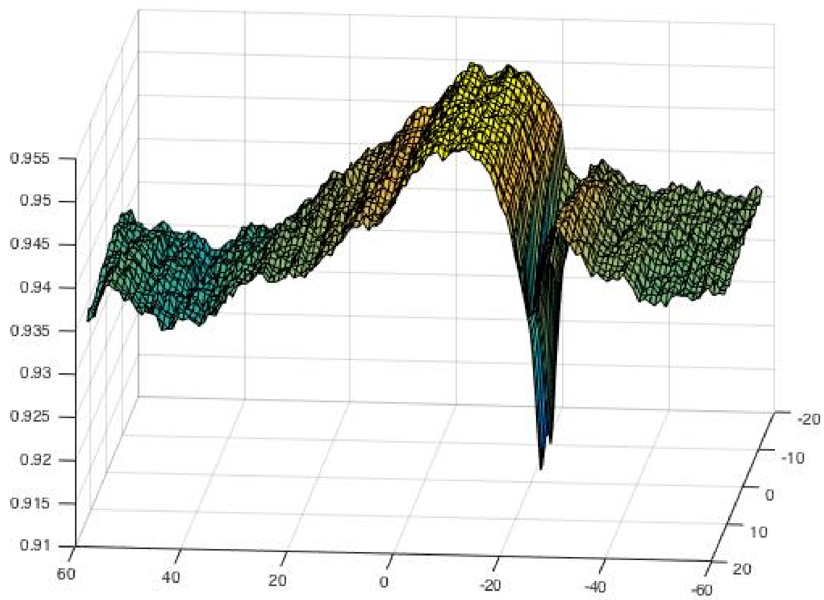


Figure 7: Joint entropy function for flash-noflash



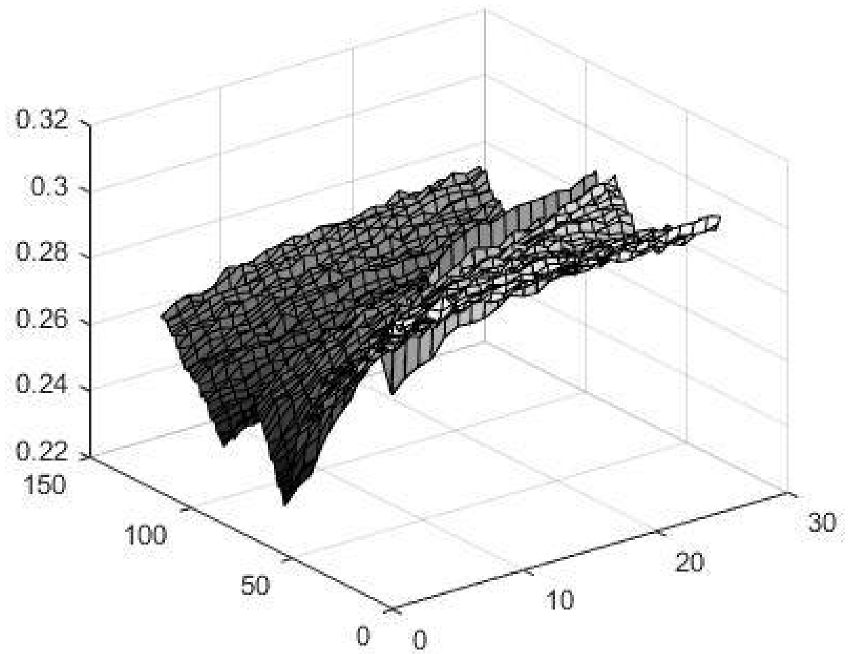
The rotational and translational alignment was perfect in the Barbara image.
For the no-flash image, the rotational alignment was perfect. However, the translation was overcorrected.

A scenario where it is clearly misaligned yet has lower entropy value is as follows:

Rotation = 180°

Translation = [-210,-60]

Figure 8: Wrong alignment yielding low entropy



6

(a) The homography computation provides a means of transferring lines between the two images and allows us to find the projection of line l' from second image to first image, i.e. s= $T^T l'$. Since l is the image of L_s, l' is the image of L'_s, and L_s and L'_s intersect at P, their images l and s= $T^T l'$ respectively will intersect at p(image of P). Also p' can be found using inverse homography.

(b) The vanishing points v and v' allow us to find the equations of the lines l and l' respectively in the two images. Also, they help us in finding L_s and L'_s lying on the ground plane, thus allowing us to apply transfer l' from second image to first image using homography.

(c) Apart from L, l and l' are also images of L_s and L'_s. And only because of the fact that L_s and L'_s lie on the ground plane, we are able to apply Homography on l' to get s = T^Tl', which in turn gives us p on intersecting with l. Without L_s and L'_s we can't apply homography, as L is not a line on the ground plane.

7

Without loss of generality, we can assume our WCS to coincide exactly with the CCS for the first camera position (center of CCS is same for both camera positions because there is no translation between them) Let P be the coordinate of the point in WCS:

$$P = [XYZ1]^T$$

Which means:

$$w_1 \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ w_1 \end{bmatrix} = K_1[I|0]P = K_1P$$

Note: Translation term is 0 because WCS coincides with the CCS. Rotation term is 0 for the same reason

Also:

$$w_2 \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} u_2 \\ v_2 \\ w_2 \end{bmatrix} = K_2[R|0]P$$

Note: Translation term is 0 because WCS center coincides with the CCS center.

Substituting P on from from the first equation to the second, we get:

$$w_2 \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = K_2RK_1^{-1}w_1 \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Multiplying both sides by $\frac{1}{w_1}$ we get:

$$\begin{bmatrix} u_2' \\ v_2' \\ w_2' \end{bmatrix} = \frac{w_2}{w_1} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = K_2RK_1^{-1} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Notice that multiplying $p_2 = [x_2y_21]$ by a constant still keeps it as a homogeneous representation of p_2. So:

$$p_2(homo) = K_2RK_1p_1(homo)$$