MAJOR TECHNICAL PROJECT ON

SEEING OBJECTS THROUGH MOVING LEAVES

INTERIM PROGRESS REPORT

to be submitted by

S.PRIYADHARSHINEE
B14117

for the award of the degree
of

BACHELOR OF TECHNOLOGY IN
COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING AND ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MANDI, MANDI

November 2017

1 Introduction

We all appreciate nature and scenery in some form or the other. Often times, especially as a photographer, you might find that the background scenery is being disrupted by an obtrusive foreground object. Be it an electric pole/ signal tower which comes in the way of your sight, or a tree/bush which obstructs your vision.
While cutting the obstructions down is neither feasible, nor lawful, and since ignoring them isn't exactly an option either, we move to science to solve this issue.
It is obvious to us that even if a tree is in front of you, you can still see parts of the background behind it through the gaps. And that when the leaves sway in the wind, you can see different parts of the scenery at different times.
Another such observation is that when the occlusion is small and close enough to your eye (or your lens), it becomes so blurred that it's almost invisible. Say for example, a couple of hair strands in front of your eye doesn't disrupt your sight. Combining these two observations, we take this problem of making the leaves disappear and try solving it using methods of computer vision.

2 Background

As discussed earlier we're trying to solve this problem using methods of computer vision. Several other people have tried solving a similar problem using various techniques. We'd discuss the ones having a similar approach to ours.
The primary approach of our problem is that we'd try and simulate a big aperture using methods of Synthetic Aperture Imaging so that our occlusion gets invisible.
Using SAI techniques to see through occlusion has been used several times. Results of these vary on varying subjects, images and techniques used to simulate a synthetic aperture.
The core of Synthetic Aperture Photography is to project all of the different views of the desired image (with occlusions) onto a virtual focal plan,e and then average it on that plane. Thus the occlusion, which would be located at different locations in different images would get averaged out and become blurred in the resulting image.
In order to related different views of the same image the concept of homographies is used.
Vaibhav et al. [1] used a dense camera array to simulate a synthetic aperture, although his primary work was regarding the calibration of light field. While Zhang et al [2] used an IMU based single camera system to take multiple photos and simulate a synthetic aperture.
We'd be using an IMU based single camera to take a video of the occlusion, changing the orientation of the camera at all points. The frames at different intervals would be used as the different views for SAP.
A plenoptic camera can also be used, and if the occlusion is small, a single image of the occluded scenery would be sufficient, as the plenoptic camera captures the 4D light field information for every image. Extracting information from this, however, would not be similar to the procedures required for a normal IMU-system camera.

3 Problem Statement

Our aim in this project is to develop a system which can be used to see through moving occlusions. In our context we'd be working on leaves as our occlusion and scenery as our background or main subject. In order to achieve this we are currently focusing on using Synthetic Aperture to solve this although there might be several other approaches to it. The reason for using this approach is that it is more geometrically sound and synthetic aperture imaging is a widely researched area as well.
In our case we'd take input several images of the scenery from different angles and try and blur out the leaves from the image. We'd try and compare the results obtained from using images from various angles and results from a static camera with moving occlusion.
We're using a single camera and moving it around to take several photos of the scenery from several angle. This approach of simulating a large aperture has its own complications as compared to using a dense camera array. But the benefit of using this is that it is cheap and allow more flexibility than the static dense camera array. The main challenge in this would be to project these images onto a common virtual focal plane.
We'll briefly discuss the theory involved and the notations used in achieving the same.

3.1 Perspective Projection

Projective geometry models well the imaging process of a camera because it allows a much larger class of transformations than just translations and rotations, a class which includes perspective projections. Of course, the drawback is that fewer measures are preserved certainly not lengths, angles, or parallelism.

3.1.1 Co-ordinate Systems

In general, we'd focus majorly on two co-ordinate systems:

- World co-ordinate system
- Camera Co-ordinate system

The World coordinates, in homogeneous form, is given by $\begin{bmatrix} X & Y & Z & 1 \end{bmatrix}^T$.
In homogeneous coordinates, $\begin{bmatrix} kX & kY & kZ & k \end{bmatrix}^T$ is equivalent to $\begin{bmatrix} X & Y & Z & 1 \end{bmatrix}^T$
Similarly for the Camera coordinates, the homogeneous form is given by $\begin{bmatrix} x & y & 1 \end{bmatrix}^T$, which is equivalent to $\begin{bmatrix} zx & zy & z \end{bmatrix}^T$.
Homogeneous coordinates are also called as Projective coordinates.
To represent a line in the projective plane, we begin with a standard Euclidean formula for a line $ax + by + c = 0$ and use the fact that the equation is unaffected by scaling to arrive at the following:

$$aX + bY + cW = 0$$
$$u^T p = p^T u = 0$$

where $u = \begin{bmatrix} a & b & c \end{bmatrix}^T$ is the line and $p = \begin{bmatrix} X & Y & W \end{bmatrix}^T$ is a point on the line. Thus we see that points and lines have the same representation in the projective plane.

3.2 Projection Matrix

The projection matrix is used in projecting a point $\big(x, y, z\big)$ in world coordinates into a camera, with respect to the camera parameters.

$$\begin{bmatrix} zx \\ zy \\ z \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

The projection matrix cumulates the effects of both the intrinsic camera parameters, as well as the extrinsic parameters relating to the camera position and orientation.

3.2.1 Intrinsic Parameters

Given the camera parameters :

- f = focal length
- m_x = number of pixel/unit distance in image coordinates along x
- m_y = number of pixels/unit distance in image coordinates along y
- $\big(p_x, p_y\big)^T$ are the coordinates of the principal point
- $\alpha_x = fm_x$, $\alpha_y = fm_y$, $x_0 = m_x p_x$, $y_0 = m_y p_y$
- s is the skew parameter

The intrinsic camera matrix, K is given by

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$$

3.2.2 Extrinsic Parameters

The extrinsic camera parameters account for the camera position and orientation, which is given by

$$\begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}$$

So combining the two camera parameters, the projection matrix P is given as follows:

$$P = K\big[R|t\big] \; and \; x = PX$$

3.3 Homography

In projective geometry, a homography is an isomorphism of projective spaces, induced by an isomorphism of the vector spaces from which the projective spaces derive. In the field of computer vision, any two images of the same planar surface in space are related by a homography (assuming a pinhole camera model).
A homography H between two images I_1 and I_2 is provided as $x_2 = Hx_1$ if x_1 and x_2 are correspondence points.
Given the two stereo camera matrices as $K\big[I|0\big]$ and $K^{'}\big[R|t\big]$, and the equation of the plane as $n^T\tilde{X} + d = 0$, we have

$$H = K^{'}\big(R - tn^T/d\big)K^{-1}$$

The infinite homography, H_∞, is the homography induced by the plane at infinity, π_∞.

$$H_\infty = \lim_{d\to\infty} H = K^{'}RK^{-1}$$

This means that H_∞ does not depend on the translation between views, only on the rotation and camera internal parameters.
Alternatively, corresponding image points are related as

$$x^{'} = K^{'}RK^{-1}x + K^{'}t/Z = H_\infty x + K^{'}t/Z$$

where Z is the depth measured from the first camera.

3.4 SFM

3.4.1 Epipolar Geometry

The epipolar geometry is the intrinsic projective geometry between two views. It is independent of scene structure, and only depends on the cameras internal parameters and relative pose. The epipolar geometry between two views is essentially the geometry of the intersection of the image planes with the pencil of planes having the baseline as axis (the baseline is the line joining the camera centres).
Suppose a point X in 3-space is imaged in two views, at x in the first, and $x^{'}$ in the second. As shown in figure 1 the image points x and $x^{'}$, space point X, and camera centres are coplanar at plane π. Clearly, the rays back-projected from x and $x^{'}$ intersect at X, and the rays are coplanar,

lying in π.

The epipole is the point of intersection of the line joining the camera centres (the baseline) with the image plane. Equivalently, the epipole is the image in one view, of the camera centre of the other view.

An epipolar plane is a plane containing the baseline.

An epipolar line is the intersection of an epipolar plane with the image plane. All epipolar lines intersect at the epipole.

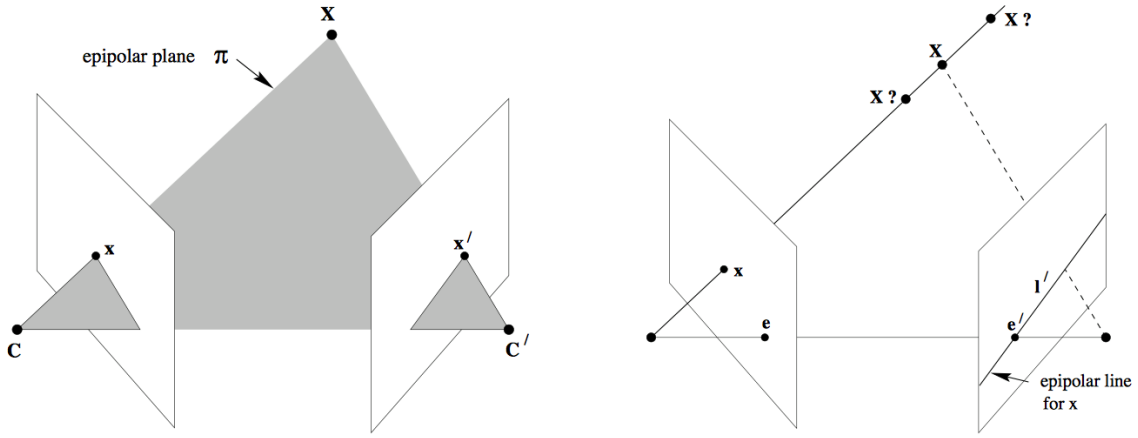


Figure 1: Epipolar Geometry - [3, p. 240]

3.4.2 The Fundamental Matrix

The fundamental matrix is the algebraic representation of epipolar geometry. Given a pair of images, it was seen in figure 1 that to each point x in one image, there exists a corresponding epipolar line $l^{'}$ in the other image. Any point $x^{'}$ in the second image matching the point x must lie on the epipolar line $l^{'}$.

The epipolar line is the projection in the second image of the ray from the point x through the camera centre C of the first camera. Thus, there is a map $x \rightarrow l^{'}$ from a point in one image to its corresponding epipolar line in the other image.

Let the homography between the two image planes (x to $x^{'}$) be H_{π}.

For the epipolar line $l^{'}$, we have $l^{'} = e^{'} \times x^{'} = \left[e^{'}\right]_{\times} x^{'}$ since it passes through both $x^{'}$ and $e^{'}$.

Since $x^{'} = H_{\pi} x$, we have

$$l^{'} = \left[e^{'}\right]_{\times} H_{\pi} x = Fx$$

where we define $F = \left[e^{'}\right]_{\times} H_{\pi}$, the fundamental matrix.

Since $\left[e^{'}\right]_{\times}$ has rank 2 and H_{pi} rank 3, F **is a matrix of rank 2**.

Similarly, for corresponding points x and $x^{'}$, we have

$$x^{'T} F x = 0 \tag{1}$$

3.4.3 The Essential Matrix

The essential matrix is the specialization of the fundamental matrix to the case of normalized image coordinates.
If the calibration matrix K is known, the point $\widehat{x} = K^{-1}x = \big[R|t\big]X$, is the image point expressed in normalized coordinates where R and t denote the rotation and translation respectively.
The fundamental matrix corresponding to the pair of normalized cameras is customarily called the essential matrix, so it is given by

$$E = \big[t\big]_{\times}R = R\big[R^{T}t\big]_{\times} \tag{2}$$

Similarly,

$$\widehat{x}^{'T}E\widehat{x} = 0 \tag{3}$$

Substituting for $\widehat{x}$ and $\widehat{x'}$, we have $x^{'T}K^{'-T}EK^{-1}x = 0$.
Thus we get

$$E = K^{'T}FK \tag{4}$$

3.4.4 Estimation of Fundamental Matrix

From equation (1), we see that each point correspondence x_i = $[x_i\ y_i\ 1]^T$ and $x_i^{'}$ = $[x_i^{'}\ y_i^{'}\ 1]^T$, generates one constraint on the elements of the fundamental matrix F.

$$\begin{bmatrix} x_i^{'} & y_i^{'} & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = 0$$

For n pairs of correspondences, the constraints can be rearranged as linear equations in the 9 unknown elements of the fundamental matrix:

$$\begin{bmatrix} x_1^{'}x_1 & x_1^{'}y_1 & x_1^{'} & y_1^{'}x_1 & y_1^{'}y_1 & y_1^{'} & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^{'}x_n & x_n^{'}y_n & x_n^{'} & y_n^{'}x_n & y_n^{'}y_n & y_n^{'} & x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0$$

or in matrix form:

$$Af = 0 \tag{5}$$

where A is an $n \times 9$ measurement matrix, and f represents the elements of the fundamental matrix f_{ij} as a 9-vector. . Given 8 or more correspondences a least squares solution can be found as the unit eigenvector (f is defined up to an arbitrary scale) corresponding to the minimum eigenvalue of $A^T A$ obtained through Singular Value Decomposition. A unique solution of F is thus obtained.

Consideration in F estimation:
Data normalization is a key point in fundamental matrix estimation. It has been proved that the computation should not be applied directly to the raw data in pixels due to potential uncertainties given by huge numbers.
We've used the method proposed by Hartley, and it is based on two transformations: First, the points are translated so that their centroid is placed at the origin; Secondly, the points are scaled so that the mean of the distances of the points to the origin is $\sqrt{2}$.
Another consideration is that we need to ensure that the rank of F is 2, also F has seven degrees of freedom (even though there are nine elements, the common scaling factor is not significant which removes one DOF. Also, the $detF = 0$ constraint removes another DOF. We enforce the rank 2 constraint by taking the SVD of F and then forcing it's 3rd singular value to be zero and then recalculating F using this new decomposition.

3.4.5 Estimation of Essential Matrix

Estimation of Essential Matrix from Fundamental Matrix is fairly easy if the camera intrinsic matrices are known. It is done using the equation (4).

3.4.6 Estimation of R and t from E

The rotation and translation matrices can be obtained by the decomposition of E into a skew-symmetric matrix corresponding to translation and an orthonormal matrix corresponding to the rotation between the views, as per the equation given in (2). The latter is in fact only possible if the essential matrix has rank 2 and two equal singular values.
Since we had already forced the rank of F to be 2 we need not do it again. We ensure that the two singular values of E are equal by taking the SVD of E and recalculating E by taking $diag(1, 1, 0)$ as the new singular values as the common scale factor is insignificant.
Numerical considerations This decomposition can be achieved by computing the singular value decomposition of the essential matrix.

$$E = USV^T \tag{6}$$

where S contains the two equal singular values and the matrices U and V are orthogonal.The translation and axis and angle of rotation can then be obtained directly up to arbitrary signs and

unknown scale for the translation by :

$$[T]_x = U \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} U^T \tag{7}$$

$$R = U \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} V^T \tag{8}$$

The projection matrices follow directly from the recovered translation and rotation by aligning the reference coordinate system with the first camera to give:

$$P = K[I \mid 0]$$
$$P^{'} = K^{'}[R \mid T]$$

where T is typically scaled such that det(T) = 1. Four solutions are still possible due to the arbitrary choice of signs for the translation T and rotation R, however the correct one is easily disambiguated by ensuring that the reconstructed points lie in front of the cameras.

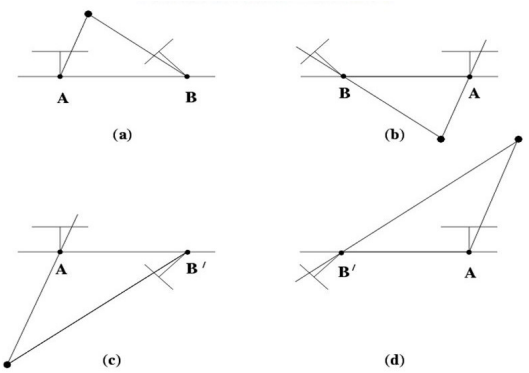


Figure 2: The visualization of the four possible solutions. In the left and right sides there is baseline reversal, while in top and bottom rows camera B rotates 180° about the baseline. Source: [3]

3.4.7 Triangulation

In order to find out the correct R and T from the four possible solutions, we compute the 3D points from their measured image positions in two or more views and given camera projection matrices. This is triangulation. Ideally, 3D points should lie at the point of intersection of the back-projected rays. However, because of measurement noise, back-projected rays will not generally intersect. Thus 3D points must be chosen in such a way as to minimize an appropriate error metric. Another popular strategy of estimating the 3D points is based on the relation:

$$x \approx PX \tag{9}$$

where x is the pixel position in image of 3D point X, and P is the projection matrix. Thus, the homogenous 3-vectors $[x]$ and PX are parallel, it is possible to write:

$$[x] \times PX = 0 \tag{10}$$

This equation has three rows but provides only two constraints on X since each row can be expressed as a linear combination of the other two. All such constraints can be arranged into a matrix equation of the form

$$AX = 0 \tag{11}$$

where A is a $3n \times 4$ matrix and n is the number of views in which the reconstructed point is visible. The required solution for the homogenous 3D point X minimizes det(AX) subject to det(X) = 1 and is given by the eigenvector of $A^T A$ corresponding to the smallest eigenvalue. It can be found by the singular value decomposition of the symmetric matrix $A^T A$.

4 Work Done During 7th Semester

As per the timeline decided previously, we have been able to achieve our goals more or less. We have been able to recreate the results of SAP on a custom setting i.e. using planar scenery, little in plane rotation and using only translation. After reading and studying about the core concepts of Image Processing and Computer vision we did small experiments and tasks that were related to our work for a better understanding of the theory.

4.1 Work Done Before First Evaluation

Prior to the first evaluation, we were trying to get a clear idea of our problem statement, and grasp some of the core concepts of the computer vision. So mostly our work before the first evaluation was literature review and reading books. Since the two related works Vaibhav et al. [1] and Zhang et al. [2] require a good understanding of the core concepts in order to fully understand and then recreate (or modify) their work, we started reading the book Multi View Geometry [3] and related those concepts with the ones discussed in the papers.

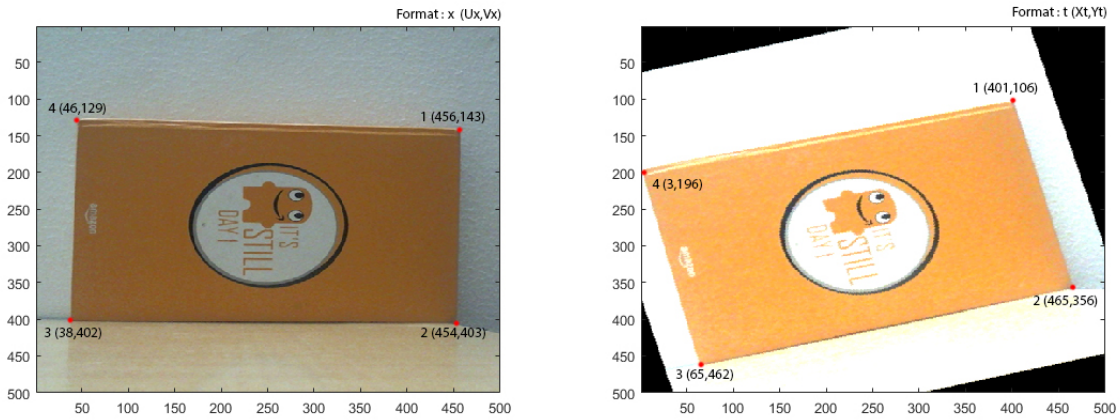
4.2 Work Done After First Evaluation

After reading the theory for quite a while we moved on to the application part and applied some of those concepts to perform some experiments that'd strengthen the concepts. We did the following work and their base and results are mentioned as well.

4.2.1 Generating Homographies

We took an image and then rotated it synthetically using the formula $H = KRK^{-1}$, where we set the R to a small angle known to us. After doing this, we took each pixel of a blank

canvas and mapped it to the corresponding pixel of the original image and copied the value of the original image onto the canvas. Thus generating the rotated image. The image used and the synthetic image we created is given in 3.



(a) Original Image (b) Rotated Image

Figure 3: Two images related by a known homography

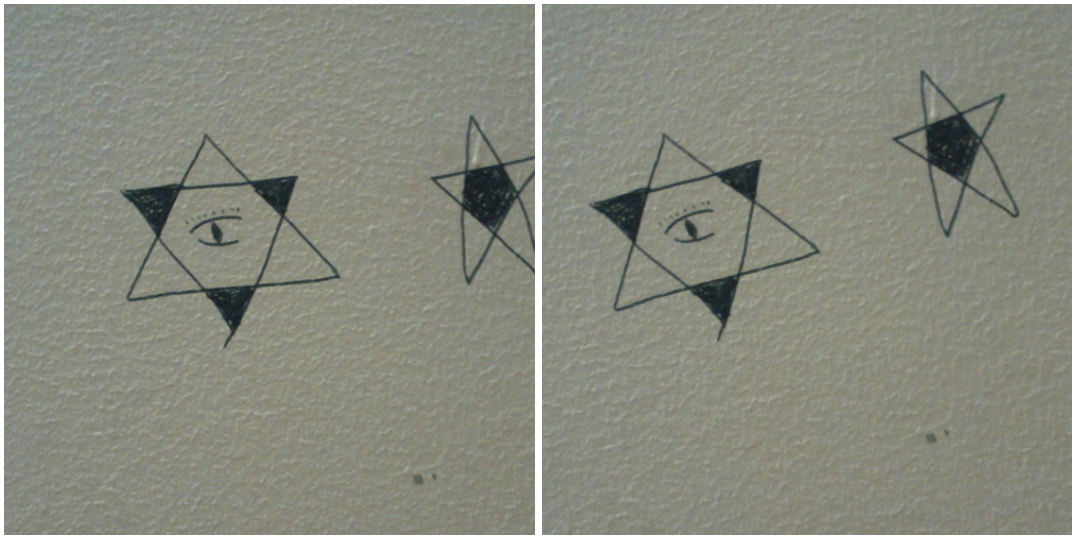
4.2.2 Estimating Homographies

Using the two images generated synthetically generated earlier (3) we manually selected 4 correspondence points (shown in the figure) and calculated the homography back. Afterwards, from the calculated homography we recalculated the R matrix using the same formula $H = KRK^{-1}$. The result of that experiment is written below. R is the original Rotation Matrix used and newR is the Rotation matrix obtained by homography estimation. As we can see that they are almost equal, we can conclude that this was a successful experiment.

$$R = \begin{bmatrix} 0.9659 & -0.2588 & 0 \\ 0.2588 & 0.9659 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix} \quad newR = \begin{bmatrix} 0.9676 & -0.2562 & 0.0005 \\ 0.2564 & 0.9651 & 0.0014 \\ 0.0000 & 0.0134 & 1.0051 \end{bmatrix}$$

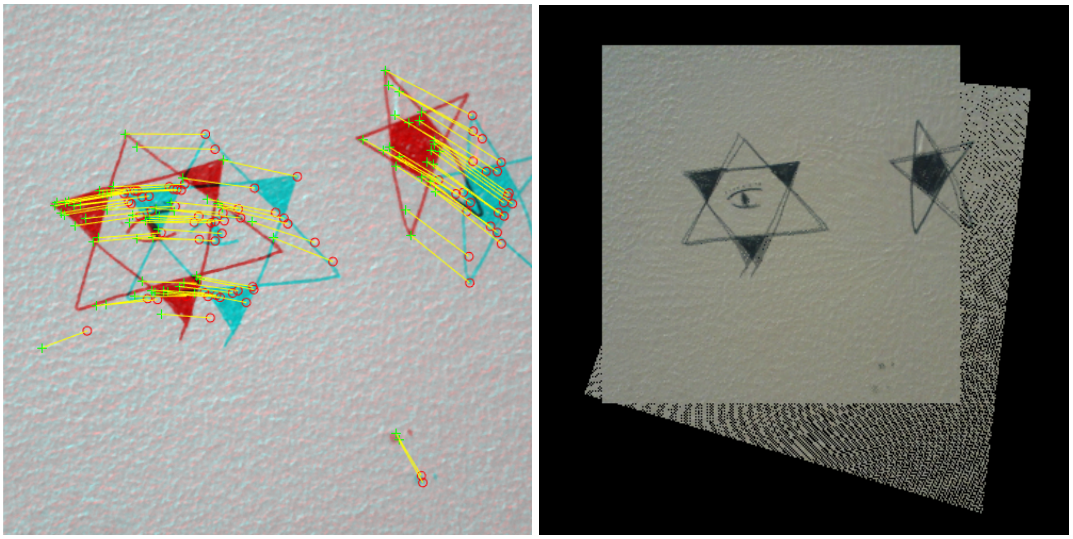
4.2.3 Mosaicing

After understanding the homography now we moved on to applying it, because in order to apply SAP we need to relate pixels of different images and average them. So we did mosaicing for the same. We took two different images and found the correspondence points using feature detection technique of MATLAB and then matched the corresponding features. Using these corresponding features we calculated the homography and then create a new image which was the mosaic of these two images. The images used (4) and the final result (5) are given below.



(a) Original Image (b) Rotated Image

Figure 4: The two input images



(a) Matched Features (b) Final Result

Figure 5: (a) The image depicting the matched features of the two images, (b) The result of mosaicing

4.2.4 Calibration

Previously we were using a webcam which was already calibrated. Now, since the quality of this webcam is less and thus lesser details about the scenery would be captured, we used a DSLR since it had more megapixels. Now, in order to get the intrinsic camera matrix we need to calibrate the camera. For this a well devised method is provided by MATLAB, in which you click several photos from various angle of a rectangular checkerboard pattern of known dimension and then calculate the intrisic camera matrix.

4.2.5 Structure From Motion (Synthetic Image)

Since homographies are good only when there is only an in-plane rotation and almost no translation, we used structure from motion to get the extrinsic projection parameters from several view points. Before moving onto a real world example we used synthetic co-ordinates on an imaginary cube and simulated two cameras and calculated the essential matrix and got the rotation and translation vector back. The visualization of the points used in given in 6.

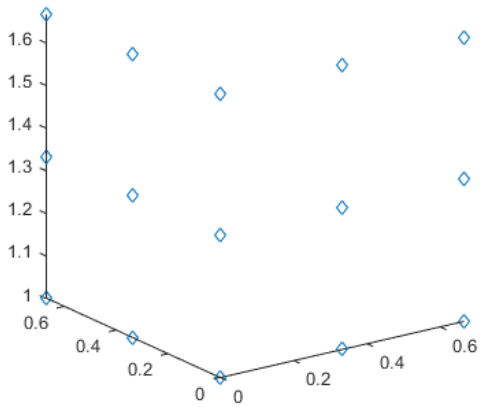


Figure 6: The visualization of the image points used

Using these points on the face of a cube, two views were created using a synthetic camera intrinsic matrix and two different R and T matrices. After doing the SFM we got the following result. E is the original essential matrix calculated using the formula $E = [t]_x * R$ and $newE$ is the computed essential matrix.

$$E = \begin{bmatrix} 0 & -1.0000 & 0 \\ -0.9524 & 0 & 0.3048 \\ 0 & 0 & 0 \end{bmatrix} newE = \begin{bmatrix} 0.0060 & -1.0000 & 0.0018 \\ -1.0134 & -0.0017 & 0.3281 \\ 0.0030 & -0.0027 & -0.0010 \end{bmatrix}$$

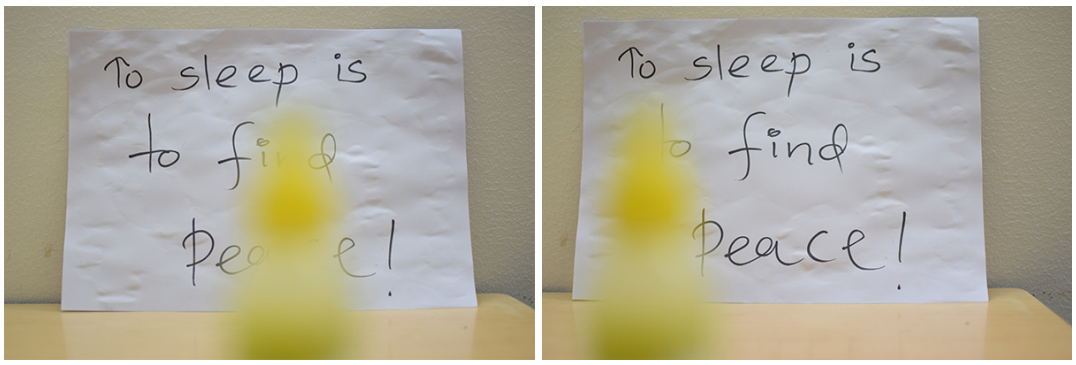
From this computed essential matrix we calculated the R and T matrix, which came out to be similar to what was used. In the following matrices, R is the original rotation matrix and $newR$ is computed rotation matrix.

$$R = \begin{bmatrix} -0.9524 & 0 & 0.3048 \\ 0 & 1.0000 & 0 \\ -0.3048 & 0 & -0.9524 \end{bmatrix} newR = \begin{bmatrix} -0.9505 & -0.0033 & 0.3106 \\ -0.0052 & 1.0000 & -0.0052 \\ -0.3106 & -0.0065 & -0.9505 \end{bmatrix}$$

4.2.6 Structure from Motion (Actual Images)

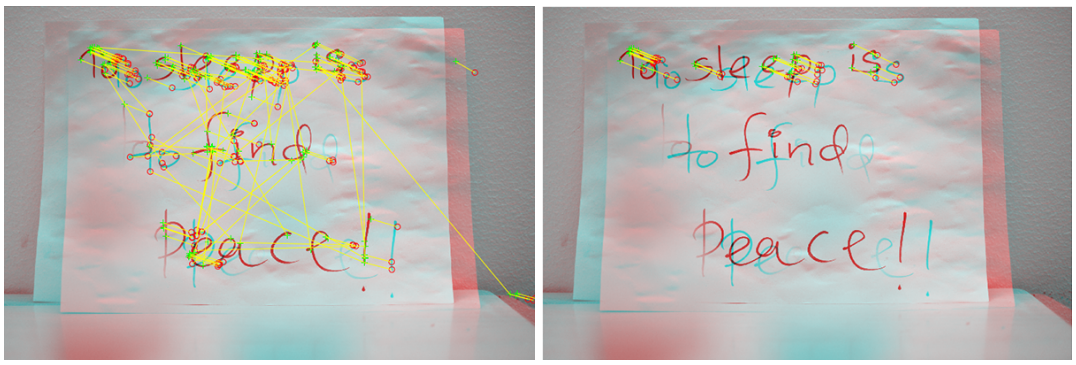
After verifying and understanding the working of SFM we tried SFM on actual images captured by our DSLR, although we used a restricted version of the problem as of now but it can be used

for complex sceneries as well. The difference between Synthetic SFM and this one was the point of correspondences. In the synthetic SFM we had actual world points and we plotted them onto two views i.e. we already had the correspondence points. While in this case we used the feature detection and matching systems provided by MATLAB to find the correspondence points. Further M-SAC, a modified version of RANSAC was used to remove the outliers in the images. After getting the inliers we used them as correspondence points and calculated the essential matrix and thus rotation matrix and translation vectors as well. These extrinsic parameters were further used for averaging the image as described in 4.2.7.



(a) Image1 (b) Image2

Figure 7: Two images used to calculate the essential matrix

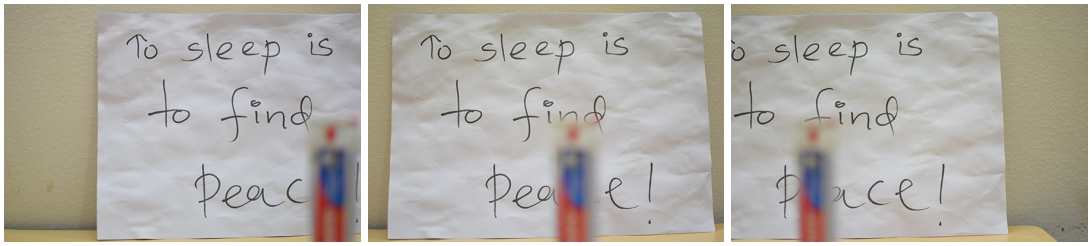


(a) Matched Points (b) Inliers

Figure 8: (a) Matched Features containing outliers (b) Inliers of the matched features

4.2.7 Synthetic Aperture Photography

After calculating the R and t from SFM we used it in the formula $x = (KRK^{-1})x_{ref} + Kt/Z$, and related points of various image and then using one of the image as a reference plane we plotted other images onto the reference plane and averaged the image out to get the result. As a result, the pixels where occlusion was there in one image was averaged out and subject behind the occlusion which was there in other images become available in the resultant image. Thus, we successfully simulated a large aperture using various images. The results of the experiment is given below.



(a) Image1 (b) Image2 (c) Image3

Figure 9: Three images used to create the synthetic aperture image. This contains only translation

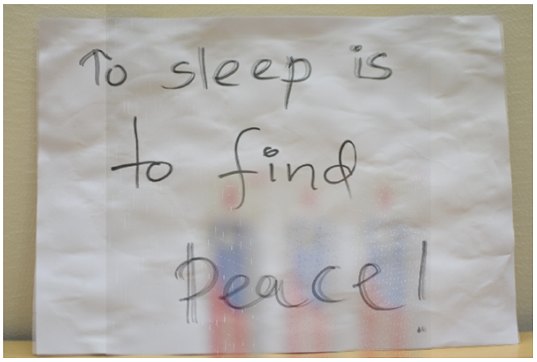
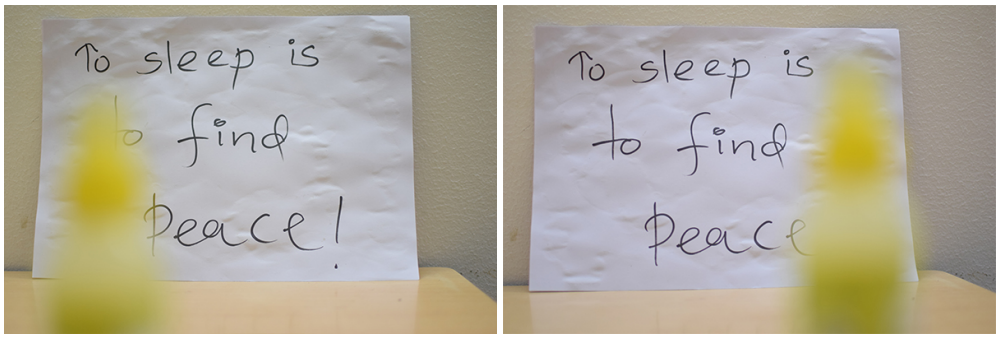


Figure 10: The resultant image of the three images shown above in 9



(a) Image1 (b) Image2

Figure 11: Two images having small rotation and translation

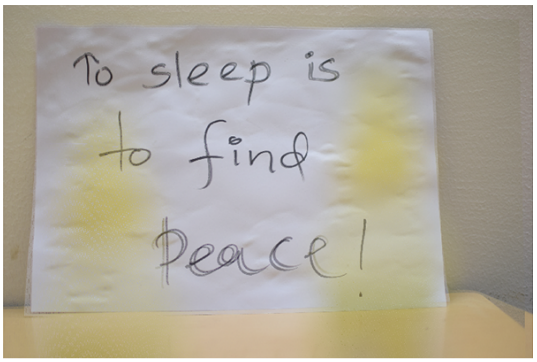


Figure 12: The resultant image of the two images shown above in 11

5 Tentative Plan

So far we have achieved our goals for the current semester, by implementing SAP on custom settings - planar background objects and restricted camera movement. Our next subtask is to implement enable the camera to move a bit more freely, and later remove the planar restrictions on the background object.
The tentative timeline is given below.

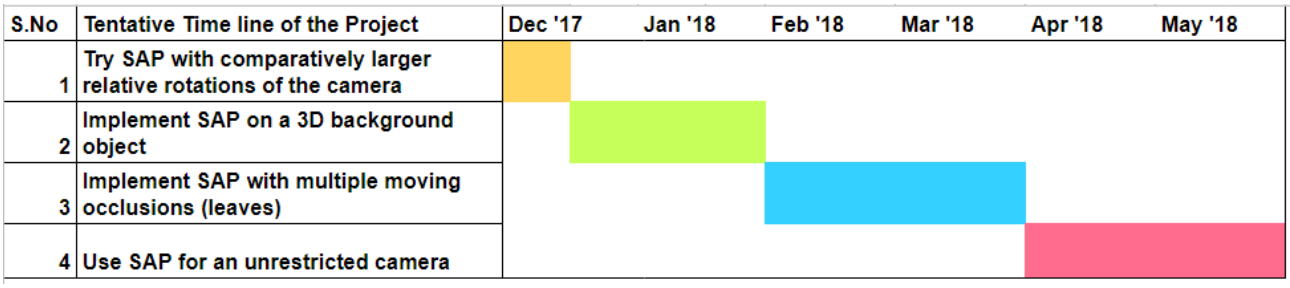


Figure 13: Timeline of MTP

References

[1] V. Vaish, B. Wilburn, N. Joshi, and M. Levoy, "Using plane+ parallax for calibrating dense camera arrays," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1, pp. I–I, IEEE, 2004.

[2] X. Zhang, Y. Zhang, T. Yang, and Y.-H. Yang, "Synthetic aperture photography using a moving camera-imu system," *Pattern Recognition*, vol. 62, pp. 175–188, 2017.

[3] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.