Computer Vision 2 (2022) **(Due: May 13, 2022)**

Assignment 2

Name: Siddharth Chaubal, Anne Chel, Venkat Mohit Sornapudi

1 Introduction

This report discusses the methods, implementations and results of the experiments performed to reconstruct 3D structure from a given dataset. In Section 2, a brief overview of used methods is provided, where sub-section 2.1 describes methods such as Eight-point Algorithm, Normalized Eight-point Algorithm, and Normalized Eight-point Algorithm with RANSAC that are used to construct the fundamental matrix. Sub-section 2.2 explains 3D reconstruction by affine Structure-from-Motion (2.2.1, 2.2.2) and its application using a graphical interface called COLMAP (2.2.3). In section 3, the implementation details, observations and inferences of experiments is provided. In Section 4, a short summary of the obtained results is given together with shortcomings. The used dataset contains 49 artificial images of a house viewed from different angles. Another dataset contains 116 natural images of Sarphati monument viewed from different angles to employ COLMAP. A dense Point-View matrix (3.2) is also provided from which we can compare the obtained custom PVM (3.2).

2 Background

This section describes the necessary background information and theory for the algorithms and methods implemented in this research.

2.1 Fundamental Matrix

The fundamental matrix describes the transformation between points in two different views. These points are selected by feature extractors, such as rotation-invariant Harris Corner Detection or rotation- and scalar-invariant SIFT. SIFT returns matches for both views based on the similarity of their descriptor vectors. These descriptor vectors describe the local region of each feature. Good features should have distinctive regions; a blue patch of the sky can probably be matched to all other blue sky patches and is therefore not a good feature. The transformation matrix can be described with the following equation, where x_i represent a feature point in the original image and x_i' represents its corresponding feature point in the target image in homogenous coordinates.

$$x_i' F x_i^T = 0 \tag{2.1}$$

This fundamental matrix F dramatically reduces the space of where a feature in image 1 can be located in image 2, i.e. it will lie on the epipolar line given by:

$$l' = F^T p \tag{2.2}$$

Equivalently a point in image 2 will lie on the epipolar line in image 1 described by

$$l = F p' \tag{2.3}$$

The above equation (2.1) can be rewritten as $Fx = 0$

$$\begin{pmatrix} x_1x_1' & x_1y_1' & x_1 & y_1x_1' & y_1y_1' & y_1 & x_1' & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_nx_n' & x_ny_n' & x_n & y_nx_n' & y_ny_n' & y_n & x_n' & y_n & 1 \end{pmatrix} \begin{pmatrix} f_{11} \\ f_{21} \\ f_{31} \\ f_{12} \\ f_{22} \\ f_{31} \\ f_{13} \\ f_{23} \\ f_{33} \end{pmatrix} = 0 \tag{2.4}$$

To calculate the fundamental matrix F one needs a minimum of 8 feature correspondences and each correspondence forms an epipolar constraint. The above constraint specifies that the fundamental matrix F transforms point 1 exactly to point 2, i.e. the error or distance is exactly zero. Often the number of point-correspondences is much higher than 8, such that we need to find one F that best fits all correspondences and the epipolar constraints are not met for all points.

2.1.1 Eight-point Algorithm

As described in the previous section to solve for the fundamental matrix F one needs at least eight point-correspondences. The eight-point algorithm first finds n matched feature correspondences. These will form the nx9 matrix A in equation (2.4). The SVD (generalized Eigenvalue Decomposition when matrix is not square) is applied to A. In a perfect world equation (2.4) would be exactly zero, this is however often not the case, i.e. no singular value is zero, so the smallest singular value will be selected, such that F is the last column of the right singular vectors of A. The rank of F should be exactly two, otherwise there will be no solution other than zero. Another SVD is performed on F (reshaped 3x3) where the smallest singular value be enforced to be zero such that the rank is 2 (when rank is not 2 the epipolar lines do not intersect in the epipole), i.e. force the last element in the diagonal singular value matrix D to be zero; with this new D F can be recalculated.

2.1.2 Normalized Eight-point Algorithm

Point-correspondences contain noise and could therefore lead to incorrect results for F. Richard I. Hartley[] showed that normalizing the feature-correspondences significantly improve results, i.e. F better captures the transformation between point correspondences. In his work normalization results in two seperate zero-centered point sets with standard deviation of $\sqrt{2}$ through two different transformation matrices T, such that

$$p_i^* = Tp_i \qquad \text{and} \qquad p_i'^* = Tp_i' \tag{2.5}$$

These normalized point-correspondes, p^* and p'^* form the input the previously described eight-point algorithm to find the fundamental matrix F^*. This matrix F^* needs then to be de-normalized using the obtained transformation matrices:

$$F = T'^T F^* T \tag{2.6}$$

2.1.3 Normalized Eight-point Algorithm with RANSAC

To further reduce the negative effect of noisy matches in matrix A, RANSAC can be applied. RANSAC iteratively takes 8 point-correspondes and estimates the fundamental matrix F. With fundamental matrix the error for all other point-correspondences can be calculated. A common way to formulate this error is by the Sampson Distance, where the error (distance with the perfect solution 0) in epipolar contraint (2.1) is normalized by both points on the epipolar lines.

2.2 Structure from Motion

The goal of the Structure from Motion algorithm is to obtain a 3D model of 2D images depicting a certain object. This process starts with finding a measurement matrix, described in section 2.2.1. From this measurement matrix the 3D model can be constructed via factorization described in section 2.2.2.

2.2.1 Measurement Matrix (Point-View Matrix)

The measurement matrix contains tracked feature points throughout all views. Note that it can occur that features only appear in a selection of views and disappear in other views. The measurement matrix has shape $2M$xN, where M represent the number of views and N the number of unique features encountered in all views. Each row $i \bmod 2 = 0$ consists of the x-coordinates of tracked features and row $i \bmod 2 = 1$

2.2.2 3D shape via Point-View Matrix factorization

A distinction can be made between global structure from motion and incremental structure from motion, where the first builds the 3D model through one factorization scheme and the latter incrementally factorizes views. This incremental approach is necessary when the measurement matrix described in the previous section is very sparse, i.e. most features only occur in small subset of views.

In the case of a dense measurement matrix one could obtain the 3D model directly via the steps of the factorization scheme described in figure 1.

1. Subtract mean for all points in measurement matrix W
2. Perform SVD on these centered points $W = UDV^T$
3. Approximate rank: take first three columns of left and right singular matrices to obtain U^* and V^* and from the diagonal matrix containing the singular values to $D*$, if there is no noise the last entries would be zero, such that rank is 3, but here we need to enforce it
4. Rewrite equation $W = U^*D^*V^{*T}$ to $W = U^*D^{*\frac{1}{2}}D^{*\frac{1}{2}}V^{*T}$
5. Obtain structure S (3D model) by $S = U^*D^{*\frac{1}{2}}$
6. Obtain motion M by $S = D^{*\frac{1}{2}}V^{*T}$

Figure 1: Factorization scheme as described by [4]

In the case of a sparse measurement matrix one divides W into dense blocks per a certain frame rate[3]. Factorize each dense block following the steps in figure 1 such that the structure and motion matrices are returned for every block. All blocks are incrementally stitched to the main view. Note however that when stitching a block displacement can occur, see figures 2 and 3.

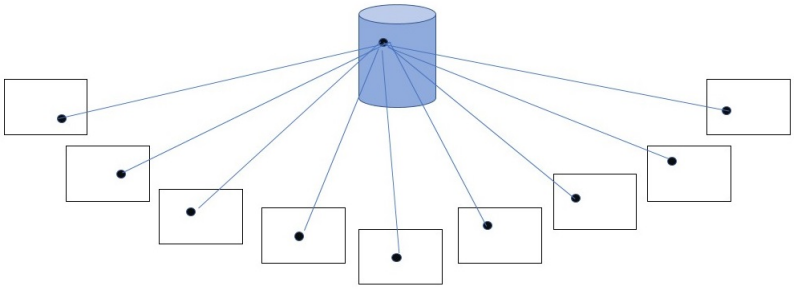

Figure 2: Schematic overview of obtaining a 3D coordinate using full dense measurement matrix with all frames

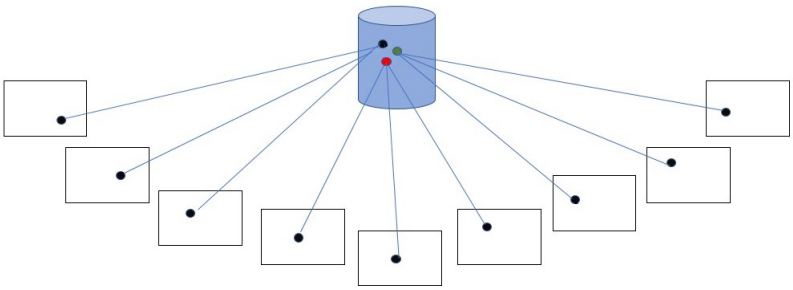

Figure 3: Schematic overview of obtaining a 3D coordinate using dense sub-blocks from measurement matrix using 3 frames

In 2 one factorization is performed such that a feature tracked in all views gets its real world coordinate. In 3 sub-blocks for three frames are used to obtain three structure matrices. Ideally these would all align to the same point, however due to the dividing in blocks, this could not be the case. The Procrustes analysis is therefore applied before stitching. Procrustes returns the transformation from two point sets such that the sets align.

2.2.3 COLMAP

COLMAP is a software that provides a graphical-based interface for 3D reconstruction. First, it performs incremental Structure-from-Motion (SfM) on overlapping images of a any particular object. SfM includes the following steps: 1. Feature detection and extraction, 2. Feature matching and geometric verification, 3. Structure and motion reconstruction . Then, Multi-View Stereo (MVS) is applied over the output of SfM to create depth and/or normal maps. Later, a dense point cloud of the scene is produced by fusion of these maps. Finally, methods such as Poisson surface reconstruction [2] are used to get the 3D surface geometry of the scene.

3 Experiments

3.1 Estimating Fundamental Matrix

The three different methods for the 8-point algorithm described in 2.1.1, 2.1.2 and 2.1.3 are implemented and the epipolar lines, described in section 2.1, are visualized using their estimated fundamental matrices. Features are detected and described using SIFT. Some examples are visible in figures 4 and 5 on the next page.

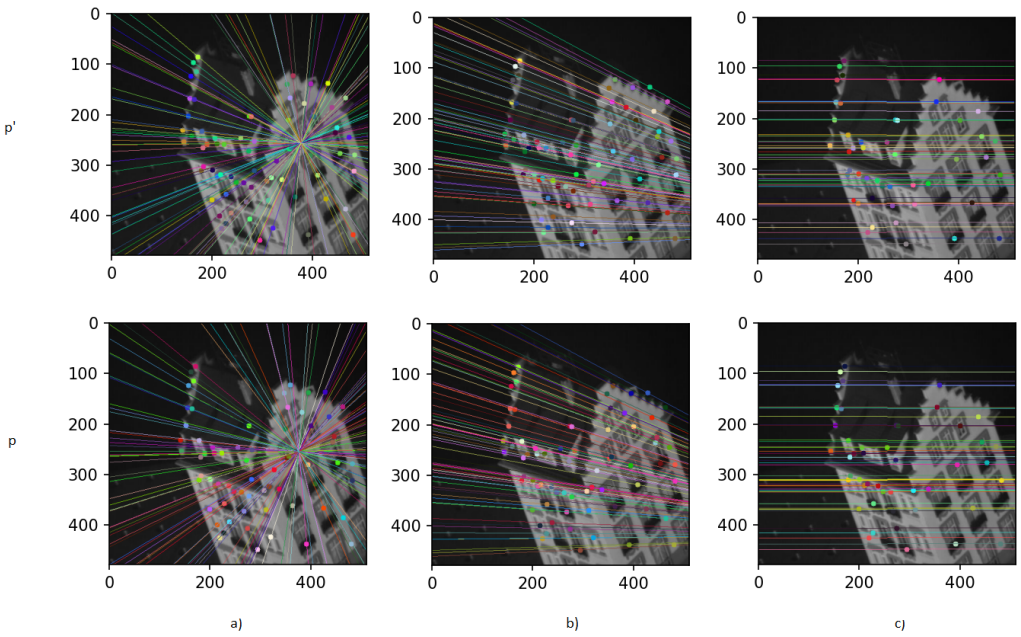

Figure 4: Epipolar lines obtained via fundamental matrix for frame 1 (p) to frame 2 (p') of a) the 8-point algorithm, b) the normalized 8-point algorithm and c) the normalized 8-point algorithm with RANSAC. For all methods the distance ratio between descriptors to obtain matches is set to 0.1. For RANSAC the Sampson distance threshold is set to 5e-5 and the number of iterations is 500.

The epipolar lines in p' and p should describe the same appearance in the image, for instance in figure 5 in all methods the epipolar line in p follows the shape of the edge of the roof on the left, in p' the different epipolar line also follows the shape of the edge of the roof on the left although the angle of the house is different. To further examine the correctness of the epipolar lines the epipolar constraint from equation (2.1) is examined. For figures 4 and 5 this results in table 1 below.

	Regular 8-P	Normalized 8-P	Normalized 8-P with RANSAC
$p'_{f2}Fp_{f1}$	0.0779	0.00364	-1.467e-16
$p'_{f4}Fp_{f1}$	0.692	-0.367	-0.254

Table 1: Examining epipolar constraint depicted by $p'Fp^T$ for the epipolar lines obtained in figure 4 (first row) and figure 5 (second row) for frame 1 and frame 1, and frame 1 and frame 49 respectively.

The epipolar constraints are not exactly met depending of course on the different parameters for the descriptor distance threshold, the Sampson distance and the number of iterations in RANSAC. In consecutive frames the

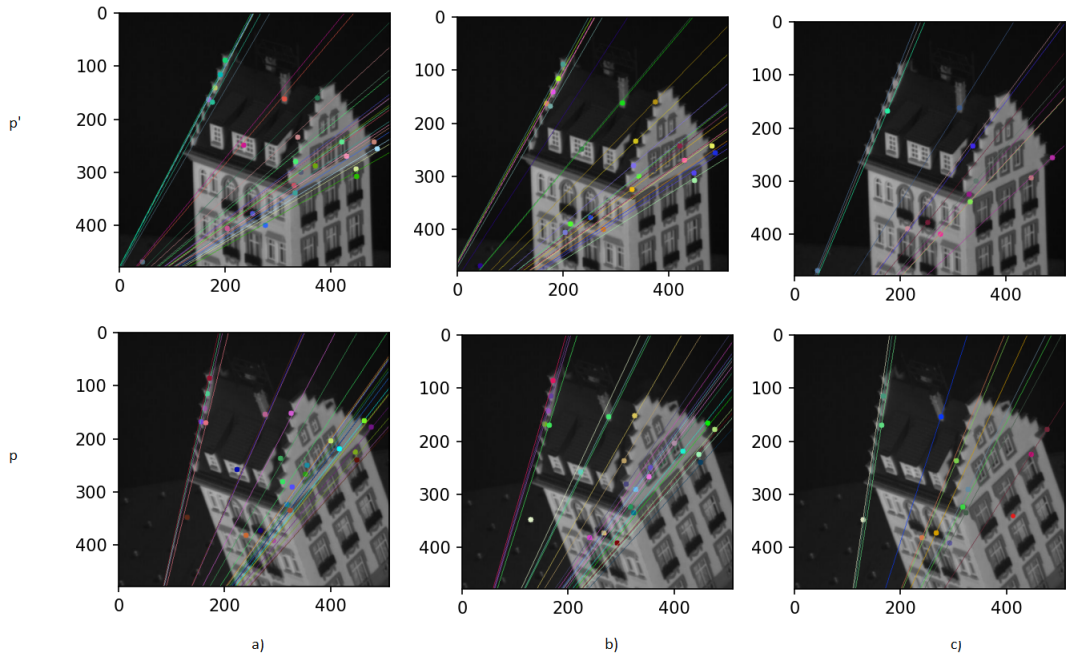

Figure 5: Epipolar lines obtained via fundamental matrix for frame 1 (p) to frame 49 (p') of a) the 8-point algorithm, b) the normalized 8-point algorithm and c) the normalized 8-point algorithm with RANSAC. For all methods the distance ratio between descriptors to obtain matches is set to 0.5. For RANSAC the Sampson distance threshold is set to 5e-2 and the number of iterations is 500.

error is much lower; the frames do not change massively and the transformation is therefore more easy to find. The normalized 8-point algorithm with RANSAC approaches zero. A quick check tells us that the normalization scheme described in section 2.1.3 indeed transform the point to have ca. 0 mean (e-18) and standard deviation of ca. $\sqrt{2}$. A trend can be observed that the normalized 8-P algorithm performs better than the un-normalized version, whereas RANSAC outperforms the normalized version. RANSAC also reduces the number of matches as is visible in the figures; filtering out bad matches to estimate the fundamental matrix.

3.2 Obtaining the Point-View matrix

Point-view matrix is an efficient way of storing the information from matched features of the images, which can be later used in Structure-from-Motion. It is constructed by storing information from each feature point in each column of the matrix. Each feature points coordinates (x,y) of an image are placed in consecutive rows. These rows start from first image to the last image in a consecutive manner.

The given PVM is dense and contains no empty cells. It is a perfect matrix to be directly used for Structure-from-Motion from all the images.

We constructed the PVM in an iterative manner by collecting features from each image that are matched with the previous and successive images. The threshold for the distance between SIFT descriptors is set to 0.5 Since consecutive views do not change much, i.e. a certain feature in p only slightly changes its x- and y-coordinates in the next view, another distance threshold is emposed, that is the sum of absolute differences between locations of keypoints in p and keypoints in p should be smaller than 4.

6.a shows the PVM constructed using all the given images. We can clearly observe that it is very sparse. This sparse matrix can be divided into dense sub-blocks, i.e. selecting only points in n frames which occur in all n frames.6 illustrates the relative growth in the density of the matrices by using smaller PVMs.

Without using this sub-blocking technique a possible solution to make the sparse matrix completely dense, is to handpick features and their tracks in all images, now every entry is filled in the PVM. This solution will work on the House dataset since the views do no change much; when encountering views with 360 degrees this technique will not work. Another approach would be to let a machine learning algorithm determine what should be on the empty positions.

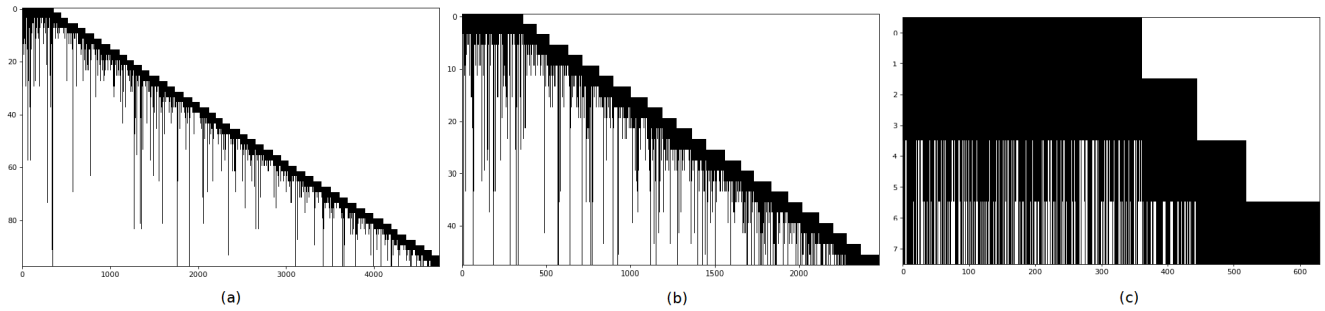

Figure 6: Binary PVM using SIFT threshold of 0.5 and distance between keypoints of 4 on a) all images b) first 24 images c) first 4 images

3.3 Structure from Motion

The PVM described in the previous section, as well as the provided dense PVM, are used as input to the SFM (Structure from Motion) algorithm via the factorization steps described in figure 1. The sparse PVM is divided into dense blocks for 3 frames and for 4 frames and each structure matrix is stitched to the main view as described in section 2.2.2. The result of the 3D model with the provided dense PVM is visible in figure 7.

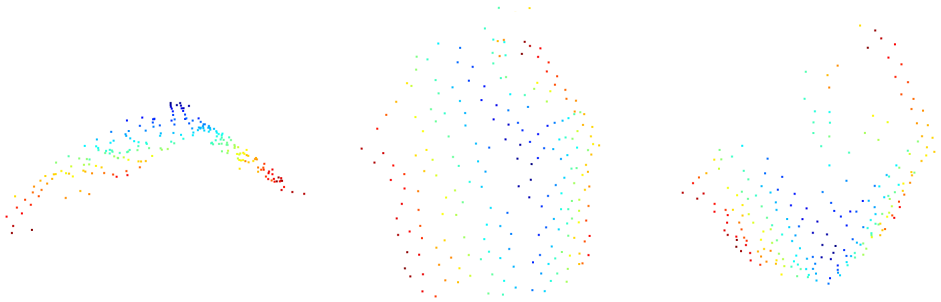

Figure 7: Obtained 3D model for provided PVM for the House data-set containing 101 views, since the matrix is dense we can directly find S through global SFM

The result of using one dense block of the custom PVM matrix is visible in figure 8

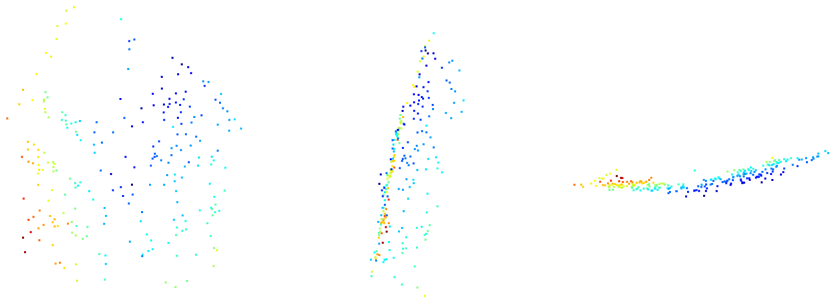

Figure 8: Obtained 3D model using one dense block for frames 1-3.

The stitched result of the custom PVM matrix is visible in figure 9
Using different frame rates, i.e. comparing dense blocks consisting of 3 or 4 frames did not yield visible differences in the obtained 3D models. A significant improvement was made by further filtering the obtained matches, not only by the descriptor distance, but also the physical distance between the keypoints in p and p'. As hypothesized the location of a keypoint can not change significantly in the next consecutive frame, since the views change only very slightly. The result of the 3D model without using this distance threshold is visible in figure 14 in Appendix A1. A not so surprising difference between the 3D model obtained by the provided dense PVM in 7 is that the house is that the angle between walls is larger and walls are more prominent, this

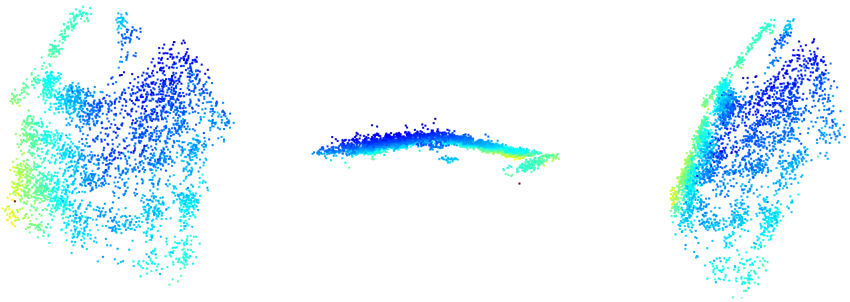

Figure 9: Obtained 3D model for custom PVM for the House data-set containing 49 views from frame rate of 3

could be due to the fact that this PVM contained twice as much views. In 7 the points in the 3D model seem to be clean and uniformly spread. From the approach in this result it is visible that the cloud is cluttered and contains some outlier points. This issue might be resolved by using RANSAC on the matches.

3.3.1 Affine ambiguity removal

To further enhance results affine ambiguity removal is implemented. Here, orthogonal axis are employed as new constraints. A quick test tells us that the the motion matrix M is now indeed perpendicular, i.e. for every row in M it is true that:

- $M[i]^T \cdot M[i] \approx 1$
- $M[i+1]^T \cdot M[i+1] \approx 1$
- $M[i]^T \cdot M[i+1] \approx 0$

Unfortunately employing this technique did not yield visible improved results in the 3D model. Suggesting that our imposed constraints on both descriptor distance and keypoint distance yielded sufficient results.

3.3.2 COLMAP

Using COLMAP in medium quality settings, we have successively reconstructed the scene from all and half (adjacent frames are reduced) of the images of Sarphati monument. We have also reconstructed the scene by removing most the background (unwanted) by cropping the images (from the half dataset) in 1:2 aspect ratio (keeping the height of the image unchanged).

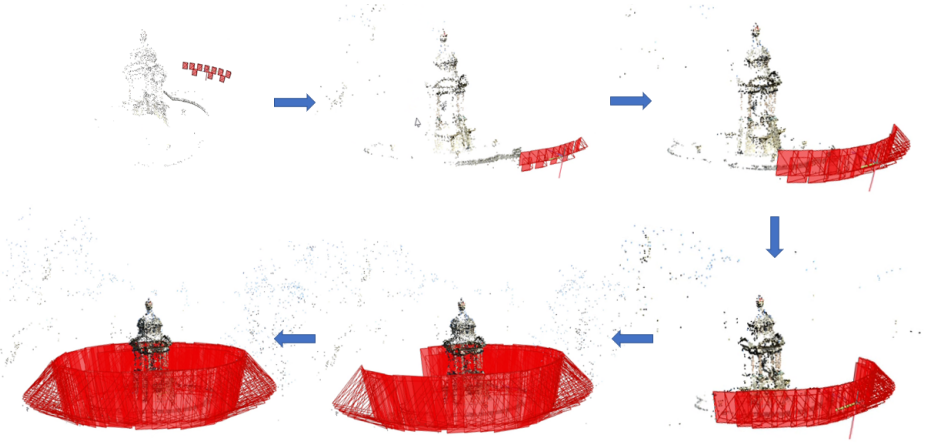

Figure 10: Steps of 3D reconstruction

From 10, we see that COLMAP iteratively adds new points by running SfM over each image. Figures 11, 12, and 13 show that fusion reconstruction looks noisy, pixelated and Poisson surface reconstruction gives a smoother surfaces. Poisson surface reconstruction effectively eliminates most of the unwanted background. By

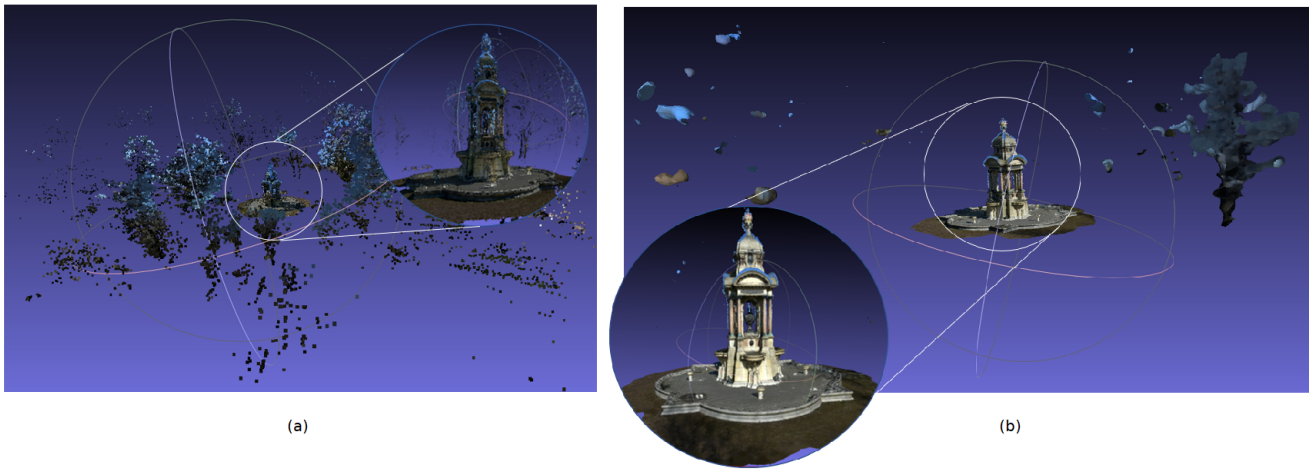

Figure 11: MeshLab visualizations of the 3D reconstructions made using all the images by a) fusion, b) Poisson surface reconstruction

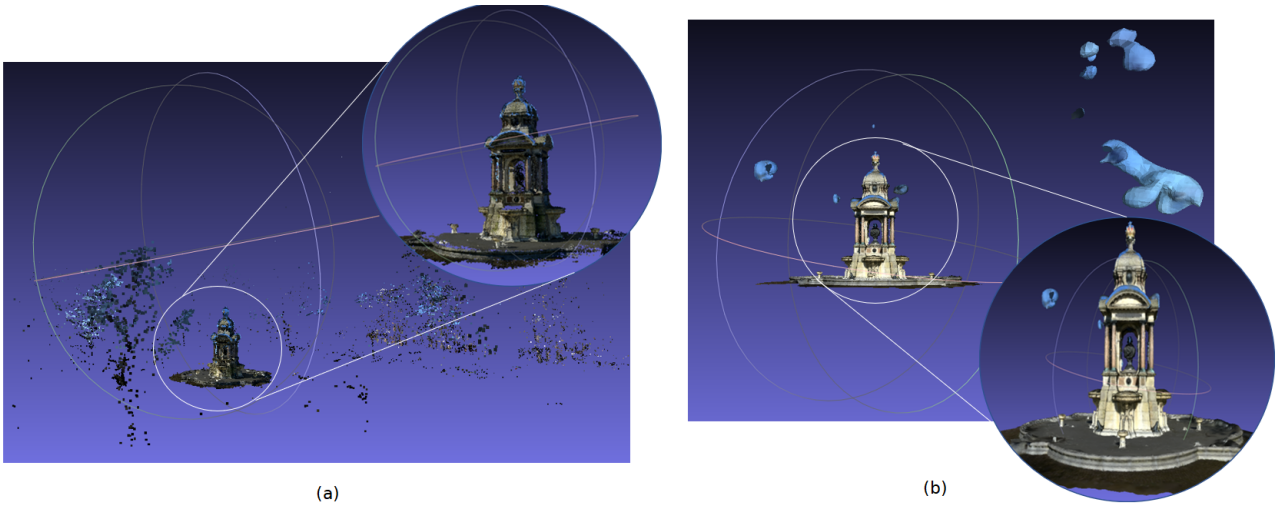

Figure 12: MeshLab visualizations of the 3D reconstructions using half of the images by a) fusion, b) Poisson surface reconstruction

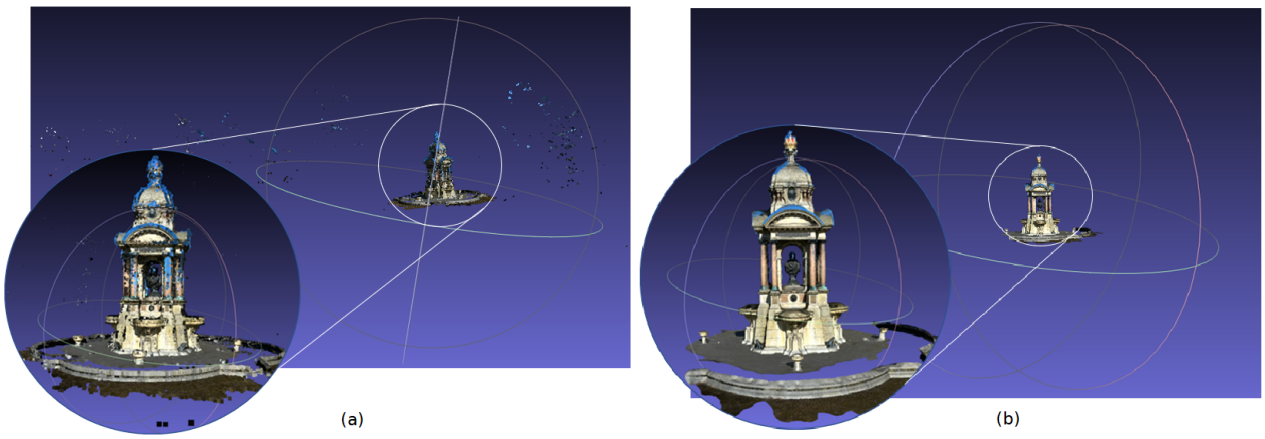

Figure 13: MeshLab visualizations of the 3D reconstructions using half of the images with reduced background by a) fusion, b) Poisson surface reconstruction

comparing 11 and 12, we observed that the outputs are almost the same quality, but reconstruction from all the

images brings more noise from the background in comparison to reconstruction from half of the images. The output of reconstruction on the cropped images (reduced background) i.e. 13 is surprisingly has almost no noise from background (especially in Poisson surface reconstruction). We can infer that even if can crop the images to the better portions we can eliminate most of the noise. But, on the other hand if we observe 13 closely, we can see that bottom platform of the monument is not reconstructed well. This might be a consequence of lesser information of lighting (reflections) compared to the full images. Hence, the best approach to get good reconstructions would be to have images which have the object homogeneously illuminated and with less background. This can be done by taking photos of the object in a well-constructed indoor environment. If the object if fixed and can't be moved to indoors, like the Sarphati monument, it is better to take photos at night using homogeneous artificial light and covering the background. If we do not have control over how the photos are shot, it is better to pre-process the images by increasing structure, sharpness, ambient lighting and reducing highlights if the light source comes in the photos.

4 Conclusion

In this report the estimated fundamental matrices and their corresponding epipolar lines for different methods of the 8-point algorithm are analyzed. From the obtained results it can be concluded that the normalized 8-point algorithm with RANSAC better follow the epipolar constraints.

The number of iterations in RANSAC impacts the overall result. Using to little iterations results in a very small set of inliers; whereas using a large number of iterations (for instance $n = 2000$) results in almost the same number of inliers as the 8-point algorithm without RANSAC. This report also showed that using a lower SIFT descriptor distance thresholds, the number of point-correspondences are reduced; with a threshold to low this could results in a set of matches that is to small. Employing another threshold on the distance between keypoints significantly improved the obtained 3D model in SFM. Adding ambiguity removal did not yield significant different results in the obtained 3D model. The same conclusion is drawn for using different frame rates.

To further enhance results on the obtained pointcloud through SFM one could use RANSAC to eliminate outliers before constructing the 3D model. Another interesting approach is to not use point-correspondences at all, a method described in the paper: "Structure from Motion without Correspondence"[1].

5 Self-Evaluation

Anne Chel and Venkat Mohit Sornapudi worked equally on code/software and report. Siddharth Chaubal could not contribute to the assignment.

6 Appendix

A1 - 3D model without distance threshold between keypoints in p and p'

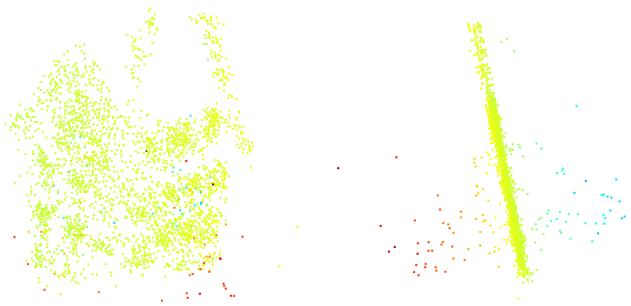

Figure 14: Two different views on the 3D model obtained without emposing a threshold on the distance between keypoints in p and p'. The pointcloud is flat.

References

[1] Frank Dellaert et al. "Structure from Motion without Correspondence". In: *Proceedings / CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 2 (Apr. 2000). DOI: 10.1109/CVPR.2000.854916.

[2] Michael Kazhdan and Hugues Hoppe. "Screened poisson surface reconstruction". In: *ACM Transactions on Graphics (TOG)* (2013). DOI: 10.1145/2487228.2487237.

[3] Fred Rothganger et al. "3D Object Modeling and Recognition Using Local Affine-Invariant Image Descriptors and Multi-View Spatial Constraints". In: *International Journal of Computer Vision* 66 (Mar. 2006), pp. 231–259. DOI: 10.1007/s11263-005-3674-1.

[4] Carlo Tomasi and Takeo Kanade. "Shape and motion from image streams under orthography: A factorization method". In: *International Journal of Computer Vision* 9 (Nov. 1992), pp. 137–54. DOI: 10.1007/BF00129684.