CS283 Final Project Report

Liam Mulshine

December 11, 2017

1 Introduction

What do robotics, forensics, and autonomous driving have in common? Each makes heavy use of a computer vision system, often one that requires extreme precision. As these fields, and others, advance, the demands that each makes on the precision of its vision system heightens. Correspondingly, the need for simple and accurate methods for camera calibration grows stronger. In his paper, "Self-Calibration from Multiple Views with a Rotating Camera," Richard Hartley proposes a novel way to calibrate a camera from as few as three images taken from the same point in space. The method that he proposes demonstrates that a near-optimal estimate of a camera's intrinsic parameters can be determined via a least-squares optimization, which makes it suitable for applications in which calibration-ease is a key requirement.

This paper will proceed as follows. First I will provide a brief background on the concept of camera calibration. Then I will then discuss Hartley's solution to the calibration problem, deriving it from the absolute conic (a key feature in projective geometry) and from the general projective camera, $P = K[R|t]$. Lastly, I will discuss my implementation of Hartley's algorithm and the results that I achieved on both synthetic and real image data.

2 Background

Camera calibration refers to the process of identifying a camera's intrinsic parameters. The intrinsic parameters define how a camera projects points from the 3-D world, to points in the image plane. Considering this, it makes sense that proper calibration is required for computer vision systems that make measurements from image data on objects in the 3-D world (such as those used in robotics, forensics, and autonomous vehicles, for example).

In total, a camera has 5 independent intrinsic parameters. These parameters are α_x, α_y (representing the focal length in pixel coordinates), p_x, p_y (representing the principal point within the image plane), and s (representing the pixel skew factor), which make up the upper-triangular intrinsic parameter matrix, K

$$K = \begin{bmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

The intrinsic parameter matrix is used to define a general projective camera, P, which takes homogeneous 3-D world points and projects them to homogeneous 2-D image points: $P = K[R|t]$. In the expression for P, R is a rotation matrix that defines the orientation of camera, P, with respect to the world coordinate system, and t, known as the camera center, is the camera's offset from the world origin.

Hartley proposes a method to determine the intrinsic parameter matrix, K, from only three views, each taken from the same point in space, but differing by a rotation about a unique axis [4]. In the next section, I will discuss the form of this solution, first using a geometric model and then using properties of the general projective camera that we've discussed.

3 Methods

Hartley's simple calibration method is based on the infinite homography relation. This relation can be derived in two ways: with the absolute conic, and with the projective camera model. Hartley derives this relation via the projective camera model in his paper [4]. However, in order to provide a better understanding of what this relationship entails geometrically, I will discuss an alternative derivation from the absolute conic, demonstrated by Hartley and Zisserman [5] and Agapito [1].

3.1 Camera Calibration using a Geometric Model

First I will provide some background on the conic.

3.1.1 Background on Conics

The conic is a geometric feature often used in Projective Geometry to determine scene structure, camera motion, calibration parameters, etc. A conic can be described in inhomogeneous coordinates by a second-degree equation in the plane

$$ax^2 + bxy + cy^2 + dx + ey + f = 0 \tag{1}$$

This equation can be transformed into homogeneous coordinates by introducing a third coordinate, x_3, and defining $x = \frac{x_1}{x_3}$, $y = \frac{x_2}{x_3}$. It follows that the second-degree polynomial representation of the conic in homogeneous coordinates is

$$ax_1^2 + bx_1x_2 + cx_2^2 + dx_1x_3 + ex_2x_3 + fx_3^2 = 0 \tag{2}$$

In homogeneous coordinates, the conic is defined only up to scale. Therefore, while it is defined by 6 parameters, it has only 5 degrees of freedom. This fact will be important later on when formulating the solution to the self-calibration problem.

The equation above can be expressed in matrix form as

$$\boldsymbol{x}^T C \boldsymbol{x} = 0 \tag{3}$$

$$C = \begin{bmatrix} a & \frac{b}{2} & \frac{d}{2} \\ \frac{b}{2} & c & \frac{e}{2} \\ \frac{d}{2} & \frac{e}{2} & f \end{bmatrix}$$

$\boldsymbol{x} = [x_1, x_2, x_3]^T$, is a homogeneous point on the conic, C [5].

With this background, I can begin to derive the form of the absolute conic and how it relates to camera calibration. In the next section, I will show that the absolute conic depends purely on the intrinsic parameters of a camera. Then I will discuss the method used to solve for the absolute conic.

3.1.2 Calibration from the Absolute Conic

The conic is useful in a variety of situations. The absolute conic in particular is widely used in camera calibration. In The absolute conic is the point conic lying on the plane at infinity, which satisfies the following two relationships in homogeneous coordinates

$$\begin{cases} x_1^2 + x_2^2 + x_3^2 = 0 \\ x_4 = 0 \end{cases} \tag{4}$$

Since, $x_4 = 0$, the relationship above can be written in matrix form as

$$[x_1, x_2, x_3] I [x_1, x_2, x_3]^T = XX^T = 0 \tag{5}$$

With this background, we can show that the image of the absolute conic depends only on the camera intrinsics. Therefore, if the absolute conic is identified, the camera is effectively calibrated, since the intrinsic parameters can be directly found. I will start by defining a homogeneous point on the absolute conic, X, in P^3. Since the absolute conic lies on the plane at infinity, the point, X, must have the following form: $X = [\hat{X}|0] = [(x_1, x_2, x_3)|0]$. Also, note that the image of points on the plane at infinity are not affected by camera translation. Therefore, given a camera matrix, $P = K[R|t]$, where K is the intrinsic parameter matrix, R is the orientation of the camera and t is the camera's offset from the world origin, the image of homogeneous world points, X, can be expressed as

$$\boldsymbol{x} = PX = K[R|t]\hat{X} = KR\hat{X} \tag{6}$$

Since K is upper triangular and R is a rotation matrix, K and R are invertible. Therefore, the homogeneous world point, $\hat{X}$, can be written as

$$\hat{X} = R^{-1}K^{-1}\boldsymbol{x} \tag{7}$$

This result suggests that the mapping from 2D projective space to 3D projective space is invertible for points on the absolute conic.

From (5) we know that the absolute conic can be defined as

$$\hat{X}^T\hat{X} = 0$$

Substituting expression (7) for $\hat{X}$, we derive the intended result

$$\begin{aligned}(R^{-1}K^{-1}\boldsymbol{x})^T R^{-1}K^{-1}\boldsymbol{x} &= 0\\ \boldsymbol{x}^T K^{-T} R R^{-1} K^{-1}\boldsymbol{x} &= 0\\ \boldsymbol{x}^T(K^{-T}K^{-1})\boldsymbol{x} &= 0 \qquad (8)\end{aligned}$$

From (6) we know that $\boldsymbol{x}$ corresponds to points on the image of the absolute conic. Relating result (8) to the constraint for conics in P^2, (3), the term $\omega = K^{-T}K^{-1} = (KK^T)^{-1}$ must represent the image of the absolute conic. Since ω is defined only in terms of the intrinsic parameter matrix, finding the absolute conic within a set of images equates to calibrating the camera that took those images.

3.1.3 Solving for the Absolute Conic

We saw that the calibration problem boils down to solving for the absolute conic. It turns out that the absolute conic can be determined from the absolute quadric [5, 2]. The absolute quadric, denoted as, Q^*_∞, is the dual of the absolute conic. Whereas the absolute conic is defined by a specific set of points on the plane at infinity, the absolute quadric is defined by the set of planes that lie tangent to the absolute quadric on the plane at infinity. Its homogeneous form is $diag(1, 1, 1, 0)$ [5]. One important characteristic of the absolute quadric is that it projects to the dual image of the absolute conic [5]. We can demonstrate this given the camera matrix, $P_i = K[R|t]$

$$\begin{aligned}\omega^*_i &= P_i Q^*_\infty P_i^T\\ &= KRQ^*_\infty R^T K^T\\ &= KK^T \qquad (9)\end{aligned}$$

Now, suppose we have two camera matrices, $P_0 = K[I|0]$ and $P_1 = K[R_1|0]$. Notice the specific form of these camera matrices. Both cameras have identical camera centers: the world origin. Camera, P_0, is defined to be oriented along the world coordinate axes and camera P_1 is a rotated version of camera P_0. The dual image of the absolute conic (DIAC), ω^*, can be written in terms of the intrinsic parameter matrix, K, as $\omega^* = KK^T$. With this setup, Hartely and Agapito show that (9) reduces to the infinite homography constraint [1, 4]

$$\omega^* = H_i \omega^* H_i^T \tag{10}$$

where, H_i, is the infinite homography projecting image points from camera P_0 to image points from camera P_1. Since the infinite homography, H_i, and the image of the absolute conic, w_i, are defined up to scale, this equality for any H_i is only defined up to scale. However, constraining H_i such that $\det(H_i) = 1$ ensures that this scale factor is equal to 1. This results from the following. Suppose that the scale factor making (10) a true equality is β, and that H_i is constrained such that $det(H_i) = 1$. Then

$$\begin{aligned} \beta\omega^* &= H_i \omega^* H_i^T \\ \det(\beta\omega^*) &= \det(H_i \omega H_i^T) \\ \beta^3 \det(\omega^*) &= \det(H_i)\det(w_i)\det(H_i) \\ \beta^3 \det(\omega^*) &= \det(\omega^*) \\ \beta &= 1 \end{aligned}$$

Thus, we have a closed form solution (albeit not a fully constrained one) for the dual image of the absolute conic in terms of the infinite homography. A fully constrained solution for ω^* is discussed in section 3.

3.2 Calibration from the Projective Camera

Hartley reaches an equivalent result in his paper, "Self-Calibration from Multiple Views with a Rotating Camera", by taking advantage of the properties of the camera model.

The derivation begins with two cameras P_0 and P_i, each with the same camera center: the world origin. The camera only differ by a rotation, so the cameras can be represented as, $P_0 = K[R_0|0]$ and $P_i = K[R_i|0]$. Since the last column of each camera matrix is the zero vector, each camera can be written as a 3x3 matrix, $P_i = KR_i$.

The world point, $\hat{X}$, is mapped to the image points, $x_0 = P_0\hat{X} = KR_0\hat{X}$ and $x_i = P_i\hat{X} = KR_i\hat{X}$ by camera's P_0 and P_i respectively. Since K, R_0, and R_i are invertible, points between images from P_0 and P_i are related by

$$\begin{gathered} R_0^{-1}K^{-1}x_0 = R_i^{-1}K^{-1}x_i \\ x_0 = KR_0R_i^{-1}K^{-1}x_i \end{gathered}$$

Therefore, a homography, mapping image points from camera P_0 to image points from camera, P_i exists

$$H_i = KR_0R_i^{-1}K^{-1} = KR_{0,i}K^{-1} \tag{11}$$

From (11) we can write $R_{0,i} = K^{-1}H_iK$. Also, since $R_{0,i}$ is a rotation matrix, $R_{0,i} = R_{0,i}^{-T}$, so

$$\begin{aligned} K^{-1}H_iK &= (K^{-1}H_iK)^{-T} \\ K^{-1}H_iK &= (K^{-1}H_i^{-1}K)^T \\ K^{-1}H_iK &= K^TH_i^{-T}K^{-T} \\ H_i(KK^T) &= (KK^T)H_i^{-T} \\ (KK^T) &= H_i(KK^T)H_i^T \end{aligned} \tag{12}$$

where, $\omega = KK^T$ is the dual image of the absolute conic, derived above. This expression is identical to the infinite homography relation, (10), which provides the means to determine the intrinsic camera parameter matrix, K.

4 Problem Solution

Camera calibration is complete once a solution, (KK^T), to equation (12) is found. One can rewrite (12) in terms of the image of the absolute conic, $\omega = (\omega^*)^{-1}$

$$\omega = H_\infty^{-T}\omega H_\infty^{-1} \tag{13}$$

where the image of the absolute conic can be expressed as a symmetric matrix in terms of its 6 independent parameters

$$\omega = \begin{bmatrix} a & b & c \\ b & c & d \\ c & e & f \end{bmatrix}$$

The homography, H_∞, is the infinite homography (i.e. the homography mapping image points generated by camera matrices with the same intrinsic parameters and camera center). Such a homography can be determined from image point correspondences alone. Provided the homography, H_i, equation (13) gives rise to a set of 9 equations relating the parameters in ω, 6 of which are linearly independent. Unfortunately, Hartley demonstrates that these 6 independent equations are not sufficient to solve for the 6 independent parameters of ω, by showing that there exists a family of homographies, $H \cdot \mathrm{diag}(\alpha, 1, 1)$ that satisfy (13). Thus, two views is not enough to solve for ω, provided that no additional constraints on the camera intrinsics are specified. To solve for ω, three or more views are required, where each view differs by a rotation about a unique axis.

Given $M \geq 2$ such views, an over-constrained system of the form, $Ac = 0$ can be constructed, where A is a 6M x 6 matrix and c is a vector holding the 6 independent parameters that define the image of the absolute conic. The solution to this over-constrained system can be solved in a least-squares sense through the singular value decomposition of A. After the SVD of A is performed, $A = UDV^T$, the column vector of V corresponding to the smallest singular value is used as the optimal solution for c. The dual image of the absolute conic, $\omega = (KK^T)^{-1} = K^{-T}K^{-1}$ can be constructed from the parameters in c. The Cholesky factorization of ω, $L = \text{chol}(\omega) = K^{-1}$ then defines the intrinsic parameter matrix, $K = L^{-1}$.

The result can be minimized further through an iterative minimization scheme; however, according to Hartley, iterative minimization does not improve the estimate of K considerably [4]. Consequently, I did not pursue this path.

5 Numerical Implementation

My numerical implementation was carried out in MATLAB. In order to solve for the intrinsic parameters using the infinite homography relation, (24), I needed data to determine the infinite homography between various camera views. To test basic functionality, I first implemented a script that simulated the setup described in Hartley's paper. To do so, I followed the implementation procedure below:

1. Define an arbitrary intrinsic parameter matrix, K
2. Construct 4 camera views from K with camera centers at the world origin. Each camera should differ by a rotation about a unique axis.
3. Generate a list of $N \geq 5$ 3-D homogeneous world points
4. Project the world points into the image plane using the 4 defined camera views. Optionally, introduce a source of noise to the projection coordinates to test robustness.
5. Find a total of 3 homographies that relate points in the image taken by the first camera to points in the images taken by each of the other cameras. These homographies correspond to the infinite homographies needed to solve for the image of the absolute conic.
6. For each infinite homography relation, construct the 6x6 matrix, A_i, that satisfies $A_i c = 0$, where c is composed of the 6 independent elements of the image of the absolute conic.
7. Stack each matrix A_i from the 3 views into an 18x6 matrix, A, and solve $Ac = 0$ in a least squares sense via singular value decomposition

This process can be generalized to $M \geq 2$ views. Note that I solve for the infinite homographies using my implementation of the normalized DLT algorithm from the second assignment [5].

6 Results

Using the method described above, I tested my implementation using both synthetic data and real image data. First I will discuss the results that I achieved on synthetic data, and then move on to the results achieved using real image data. Finally, I will discuss various sources of error that I've identified in the methods that I used.

6.1 Testing on Synthetic Data

I ran 1 iteration of the calibration procedure above, for 5 different noise levels (in pixels). This experiment is close to identical to the test implemented by Hartley; however, Hartley does not provide much detail about the nature of the noise that he adds to his synthetic data. Considering this, I tested with two different noise sources: a standard normal source with zero mean and standard deviation equal to the noise level, and a uniform random noise source whose range is equal to the noise level. I've tabulated my results using the standard Gaussian noise source below:

Noise	α_x	α_y	p_x	p_y	s
0	1000.0	1100.0	200.0	300.0	0.0
0.125	1000.1	1099.8	199.9	300.0	-0.5
0.5	1000.5	1000.2	200.6	300.0	-1.1
1.0	997.7	1096.4	202.2	298.1	7.1
2.0	991.4	1096.1	204.6	299.0	18.0
4.0	1014.6	1109.5	192.7	300.1	-18.7
8.0	973.0	1090.7	218.0	290.1	36.9
16.0	1025.1	1096.1	137.9	316.8	-10.7

I ran 100 iterations of the procedure described above, each with the same initialization (i.e. the same camera views), but with different realizations of Gaussian random noise. For each iteration, I projected the 3-D homogeneous world points into the image plane using the camera with the estimated intrinsic parameter matrix, and calculated the reprojection error with respect to the image projected by the camera with the true intrinsic parameter matrix. I averaged the reprojection error over all iterations for various levels of noise. Below are my results:

Noise	Average Error	σ
0	0.00	0.00
0.125	0.29	0.17
0.5	1.08	0.68
1.0	2.62	1.37
2.0	4.83	2.42
4.0	10.22	5.56
8.0	19.40	11.92
16.0	39.90	23.03

Hartley does not measure the reprojection error associated with each calibration estimate. However, the reprojection error provides a nice quantitative measure of the camera calibration quality. We see that noise levels under 0.5 pixels result in an average reprojection error of approximately 1 pixel or less, with relatively low standard deviation. As the noise level affecting the set of point matches grows, however, the averaged reprojection error increases linearly.

6.2 Testing on Real Image Data

After testing my implementation on synthetic data, I turned to real image data. First I found the “true” intrinsic camera parameters with MATLAB’s camera calibration tool toolbox [3]. Then I calibrated it using a similar approach as before; however, since I was no longer using synthetic data, I needed to identify point matches between the reference image and each rotated view so that an infinite homography could be determined for each view. To do so, I made use of various MATLAB tools for image processing. Specifically, after importing each grayscale image into MATLAB in .tiff format, and correcting each image for radial distortion introduced by the lens, I followed the procedure below to determine the infinite homographies:

1. Detected interesting features in each image using MATLAB’s detectHarrisFeatures() tool
2. Matched interesting features between a reference image and each rotated view using MATLAB’s matchFeatures() tool
3. Passed the identified point matches through my RANSAC implementation from the second assignment to discard extreme outliers.
4. Calculated the infinite homographies mapping points in the reference image to points in each rotated view using the normalized DLT.

Using MATLAB’s calibration tool, I estimated the following intrinsic parameters for a camera with 22.7 mm sensor width, an output image size of 1280 x 854, and a lens with focal length of 50mm:

α_x	α_y	p$_x$	p$_y$	s
2600.0	2606.4	491.8	508.2	-7.5

These results are reasonable. A focal length in pixel coordinates of 2600 corresponds to a metric focal length,

$$f = \alpha_x \text{ x (sensor width)/(image width)} = 2600 \text{ x } 22.7/1280 = 46.1\text{mm}$$

which is quite close to the expected value of 50 mm. Additionally, the principle point is within the image frame (close to the center), and there is low skew. While this parameter matrix is truly just an estimate, I will use it as a baseline for comparison with my self-calibration routine.

The process of data collection for calibration with my implementation of Hartley's algorithm was a fairly extensive. I first identified scenes that might generate a large number of point matches, and then took images from multiple views at different points in space. Below is a scene on which my calibration data presented below is based on.

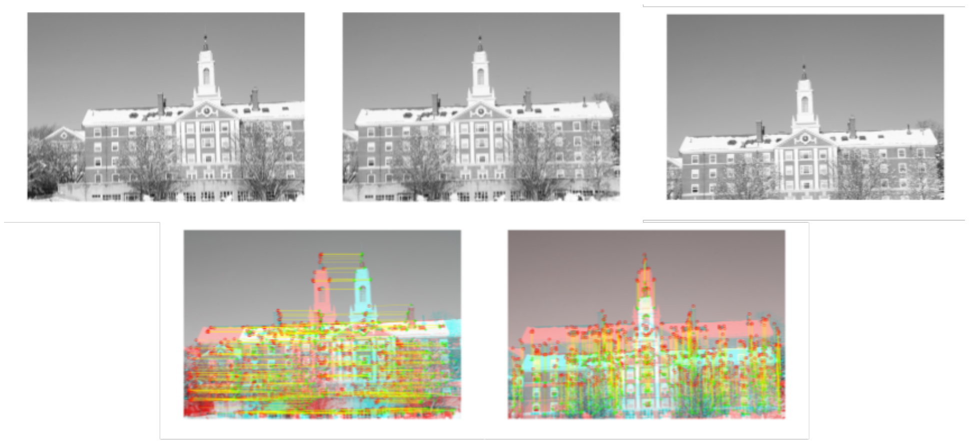

Figure 1: Three grayscale images of the Radcliff Quadrangle taken from the same camera center at different orientations. Beneath the grayscale images are figures demonstrating the relative point matches between the center image and each of its neighboring images.

After imaging these scenes I transferred the image data to my computer and then exported the images in .tiff format, with dimensions, 1280 x 854. I removed the effects of radial distortion using the radial distortion estimate from MATLAB's calibration tool for my camera and performed feature matching as described above to calculate the homographies mapping image points in the reference image to image points in the two alternate views. I ran 20 iterations of my calibration routine, each time using a homography determined via a RANSAC fitting technique, and selected the intrinsic parameter matrix that minimized the reprojection error. Below are my results:

Scene	α_x	α_y	p_x	p_y	s	Error
1	2585.0	2404.3	535.3	531.8	8.2	67.8

The reprojection error was calculated with respect to the image taken by the "ground truth" camera, using the intrinsic parameter matrix estimated by MATLAB's calibration tool. In the same way as in the synthetic data tests, reprojection error was determined by taking the mean squared error between the projection of 14 world points onto the image plane using the "ground truth" and estimated intrinsic parameter matrices respectively. Since the principal point of the estimated intrinsic parameter matrix is quite offset from the principal point from the "ground truth" intrinsic parameters, I would expect the reprojection error to be quite large. During the 20 iterations of my calibration scheme, some estimates were clearly off track. Theses estimates likely correspond to instances in which the homography generated by RANSAC poorly estimates the true homography. Below I discuss some of the sources of error in my methods.

6.3 Sources of Error

Comparing the reprojection error achieved here to that achieved with synthetic data at different noise levels indicates that point matches in my real image data were determined with low accuracy. There are 3 key explanations for this. First, and perhaps most importantly, inaccuracy in the radial distortion estimate affects each image and could increase point match errors considerably. At first, I did not correct for radial distortion, and noticed extreme errors that manifested themselves most intensely in the focal length. While not ideal, I corrected for radial distortion using MATLAB's estimated distortion terms. With a more accurate distortion correction scheme, higher precision homographies could be obtained, resulting in much more accurate intrinsic parameter estimates. Second, slight changes in the focal length resulting from adjusting the camera's aperture when focusing is almost unavoidable. Initially I attempted to maintain constant focus between the two calibration methods used: (1) MATLAB's calibrator and (2) my implementation of Hartley's method. However, adjusting the focus does not affect the intrinsic camera parameters considerably, so I loosened this constraint so that I could test on scenes with more distant focal planes, which are much less susceptible to error from slight camera center translations between views. Finally, while not a major source of error, it is important to remember that MATLAB's calibrator is truly an estimate, and its results cannot be accepted as the absolute ground truth.

7 Future Work

Identifying and analyzing the sources of error in Hartley's method highlight a few of its weaknesses. With more time and sufficient resources, I would work to reduce the impact of these error sources. In particular, I would look into methods to estimate and correct for radial distortion without having to rely on an outside camera calibration tool, since I found that radial distortion has a strong impact on the calibration result accuracy. I would then do more experimentation and research into the exact effect of adjusting the focus of my 50 mm lens on the camera's intrinsic parameters. Lastly, I would implement an iterative minimization scheme to improve upon the least-squares estimate obtained above.

8 Conclusion

Hartley's proposed method for self-calibration has a few desirable properties: (1) it is light-weight and can be easily implemented once the supporting theory is understood, (2) it can solved via a simple linear optimization and, thus, runs quickly and efficiently, and (3) it requires very little information about the camera in use (just confirmation that each view was taken from the same camera center). A key weakness, however, is that it is very sensitive to inaccurate point matches between views, with the mean-squared-reprojection-error increasing linearly with pixel noise. However, provided an accurate feature detection/point matching algorithm, Hartley's method is very robust and could be used for self-calibration in a wide range of real computer vision systems.

References

[1] Agapito, Lourdes de, Hayman, Eric, Reid, Ian D. "Self-Calibration of Rotating and Zooming Cameras." Information Engineering, Nov. 2001, www.robots.ox.ac.uk

[2] Triggs, Bill. Autocalibration and the Absolute Quadric. International Conference on Computer Vision Pattern Recognition (CVPR '97), Jun 1997, San Juan, Puerto Rico. IEEE Computer Society, pp.609–614, 1997.

[3] "Camera Calibrator." Single Camera Calibration App - MATLAB Simulink, www.mathworks.com/help/vision/ug/single-camera-calibrator-app.html.

[4] Hartley, Richard I. "Self-Calibration from Multiple Views with a Rotating Camera." Springer-Verlag, 1994.

[5] Hartley, Richard, and Andrew Zisserman. Multiple View Geometry in Computer Vision. Cambridge Univ. Press, 2015.