in Proc. of IV Jornadas de Engenharia Electrónica
e Telecomunicações e de Computadores, pp174-179
November 2008, Lisbon, Portugal

CALIBRATING A NETWORK OF CAMERAS
Based on Visual Odometry

Nuno Leite, Alessio Del Bue, José Gaspar
Instituto Superior Técnico,
Universidade Técnica de Lisboa, Portugal
nunompleite@gmail.com; adb,jag@isr.ist.utl.pt

Keywords: Camera Network, Visual Odometry, Calibration, and Estimation of Camera Pose.

Abstract: This paper presents a methodology to estimate the calibration of a network of cameras, possibly with non-overlapping fields of view. Calibration comprises both the intrinsic and extrinsic parameters of the cameras and is based on a mobile robot with the capability of estimating of its pose in a global frame. The robot is equipped with one calibrated camera which we assume that can be oriented in a manner to observe world points also seen by the network of cameras.

Our methodology is based on matched scale invariant features (SIFT) reconstructed to 3D points using e.g. SLAM, and focus on the problem of transporting the robot coordinate system to the fixed cameras. The reconstructed 3D points and their images on the fixed cameras are proposed as a solution for the calibration problem. In order to test the validity of our methodology we constructed a VRML scenario, thus having low noise images and ground truth information. Results show the successful calibration of three fixed cameras using a mobile camera that acquired about thirty images.

1 INTRODUCTION

The increasing need of surveillance of public spaces and the recent technological advances on embedded video compression and communications made camera networks ubiquitous. Typical environments include single rooms, complete buildings, streets, highways, tunnels, etc. While the technological advances already allowed such a wide installation of camera networks, the automatic understanding and processing of the video streams is still an active research area.

One of the crucial problems in camera networks is to obtain a correct calibration of each camera in terms of a unique reference frames. This condition is a fundamental feature required for further higher level processing (i.e. people/car tracking, event detection, metrology) and nowadays one of the most complex problems in Computer Vision. The problematics generally arise from the lack of overlapping field of views (FOV) of the camera which does not allow the estimation of a common reference frame for each camera. In such scenario, exactly geolocating each sensor is extremely complex without the aid of special equipment (moving calibration patterns or GPS) or a priori reference images such as a panorama of the given environment.

Previous approaches are mainly focused in detecting enough common features between images in order to link each cameras to the reference coordinate systems. This can be achieved using a set of panorama images which links the non-overlapping field of views or by a mobile platform which set the reference by navigating into each camera field of view. The first case is usually followed by a pre-generation of a panorama using a standard Pan-Tilt-Zoom (PTZ) cameras [14] which is then followed by pairwise and global alignment of the set of cameras using standard bundle adjustment [16]. In the case of a mobile platform, the reference is always given by the mobile platform which carries a calibration pattern or takes snapshots of the scene thus obtaining an overlapping FOV. In [15, 1] a set of images taken from a mobile platform are registered to a wide set of omnidirectional cameras which allows a consistent overlaps in the given scene. Differently in [13] a robotic platform carries a pattern represented by a set of markers. Calibration is given by registering the pattern to each view given the reference of the robot. This solution however requires each camera seeing the mobile robot.

In our methodology we use of the robot in a different manner of other works [12, 15, 1, 13]: instead of building large calibration patterns to transport with the robot, and imposing the constraint that the fixed cameras can effectively see the robot, the robot equipped with the camera just has to see scene points also observed by the fixed cameras. This is simpler, considering that the camera on-board can even be a versatile PTZ camera.

In order to calibrate the camera network we follow the approach of reconstructing some points of the scenario. These points are expressed in a global (world) coordinate frame provided by self-localization information assumed to exist in the mobile robot. Given the reconstructed 3D-points of the environment and their images we can calibrate the network of cameras using standard computer vision methodologies [7].

The reconstruction of 3D points comprises two main steps, namely matching image points and computing their locations. We do the matching based on SIFT features, state of the art features well known to provide a very robust matching procedure [10], and the computation of the 3D locations is based on vSLAM [5, 8].

In order to test the accuracy of the methodology we followed the approach of creating one simulated environment based on VRML. VRML allows creating different types of scenarios, rendered as images with low levels of noise[1], on which we can test feature detection and matching. The ground truth allows also to evaluate the quality of both the reconstruction of points and camera poses.

2 CAMERA NETWORK

In our work a camera network is defined as a set of static cameras placed arbitrarily in the scenario (see Fig.1). In general we will not assume any kind of overlapping of the fields of view[3]. The main objective is than to estimate the calibration of the cameras, more precisely their intrinsic and extrinsic (localization and orientation) parameters. The extrinsic parameters are to be estimated in a global reference frame.

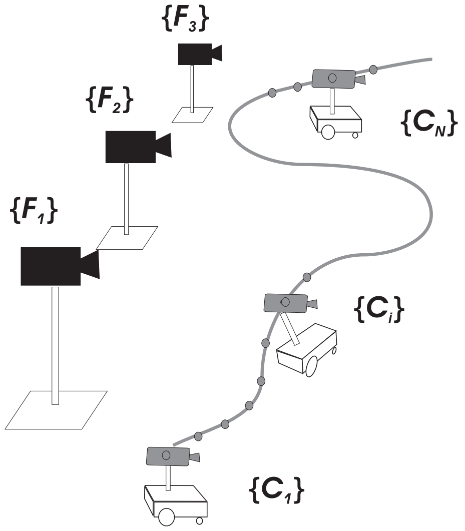


Figure 1: Camera network formed by three static cameras. The mobile camera transported by the robot is used to calibrate the network.

The N cameras of the camera network are assumed to be perspective (pinhole):

$$network = \{F_i : i = 1 \dots N\}$$

i.e. F_i are projection matrices, which we generically denoted as P in the following. In more detail, the projection of a 3D world point, $M = [X\ Y\ Z\ 1]^T$ to an image point, $m = [\lambda u\ \lambda v\ \lambda]^T$, in a camera, P, is represented as [4]:

$$m = PM, \quad P = K[R\ \ t] = [P_{1:3,1:3}\ P_{1:3,4}] \qquad (1)$$

where P can be decomposed in the intrinsic parameters matrix K, a rotation R and a translation t, and the subscripts $P_{a:b,c:d}$ denote selection of lines (a to b) or columns (c to d). The intrinsics parameters matrix, K is assumed to have an upper triangular form:

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (2)$$

where (α_x, α_y) represent scalings from meters to pixel coordinates, s is the skew coefficient, and $(x_0, y_0)^T$ represent the coordinates of the principal point. Finally, the rotation matrix R is unitary and thus combined with K implies that $\|P_{3,1:3}\| = 1$.

In this work, the calibration methodology of the camera network is based on reconstructed image points. The points to reconstruct are first selected from image points, using e.g. SIFT features [10], which are seen both by the static, F_i and the mobile, C_i cameras (see Fig.1). We assume that the mobile camera is calibrated and can be placed (oriented) to see sub-sets of scene points visible by the fixed cameras. The robot itself does not need to be seen by the fixed

[1] Mainly quantization noise associated to the rendering methods of VRML browsers.

cameras. However, we assume that the robot has its own global coordinate system and can estimate its localization while it moves. The localization method of the robot can be based on e.g. odometry, differential GPS, SLAM / vSLAM [5, 8], or any other method. In this work we focus on the aspect of transporting the localization information from the robot to the networked (fixed) cameras.

The calibration methodology comprises in essence two steps: (i) estimation of the projection matrices representing the cameras, F_i in a global reference frame;(ii) obtaining the intrinsic and extrinsic parameters by factorization. These steps are detailed in Sec.3.

3 CAMERA POSE

In this section we propose a methodology for estimating the pose of a fixed (networked) camera, knowing that the camera and the robot camera have overlapping FOVs. We follow an approach based on conventional camera calibration assuming that we have a good estimation of the scene structure provided by e.g. a vSLAM algorithm.

3.1 Camera Calibration

In order to estimate the calibration of a fixed camera we use 3D points reconstructed by the camera mounted on the mobile robot. Those 3D points are assumed to be visible, and matched, in both the mobile and the fixed cameras. The 2D images on the fixed cameras and the 3D knowledge of those points allows estimating the camera calibration, or in other words, its projection matrix.

The projection of a 3D world point, M to an image point, m, described by Eq.1, can be re-written in order to show explicitly a linear relationship with the entries of the projection matrix, P. Despite having a system of three equations in Eq.1, the equality up-to a scale factor, λ implies that one equation is dependent on the others. Vectorizing P as $p = [P_{1,1:4}\ P_{2,1:4}\ P_{3,1:4}]^T$, i.e. a twelve entries vector, and removing the scale factor λ from Eq.1, one obtains a system with just two equations:

$$\begin{bmatrix} M^T & 0^T & -uM^T \\ 0^T & M^T & -vM^T \end{bmatrix} p = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{3}$$

Considering a set of n 3D points and the corresponding 2D projections on the image plane, one obtains a $2n \times 12$ matrix A by stacking up the Eq.3 for each correspondence. In order to avoid a trivial solution for p in the system $A \cdot p = 0$, one can impose that $\|p\| = 1$. The solution for p is then the singular vector corresponding to the least singular value of A. Since p has twelve entries, and knowing that each 3D-2D pair of points gives two equations, than one needs at least six 3D-2D pairs of points.

Finally, in order to impose that $K_{3,3} = 1$ and that the rotation component in P is unitary, we set a unit norm to $P_{3,1:3}$ with:

$$P \leftarrow P / \|P_{3,1:3}\| . \tag{4}$$

3.2 Pose Estimation

Given the projection matrix, P representing a general perspective camera, Eq.1, and estimated using the method detailed in the previous section, we want to find explicitly the camera's position, t and orientation, R. This implies removing the intrinsic parameters information, K from P.

In other words, we want to decompose the projection matrix in its intrinsic and extrinsic parameters. In [7], there is proposed a direct factorization method of P using Givens matrices, however we follow an approach closer to the one in [6] which is based on (i) QR factorization of P, (ii) transformation from the QR to the RQ factorization, and (iii) sign correction of K.

QR based on Gram-Schmidt orthonormalization As noted in [6], many numerical packages provide the RQ factorization. Nevertheless, in case there is a need to write code from scratch, there are several methods to do the QR factorization. Most common methods are based on Householder matrices, Givens matrices or simply on the Gram-Schmidt orthonormalization. In this work we use the Gram-Schmidt process which proposes as the unity matrix just the orthonormal vectors found from the matrix being factorized:

$$P_{1:3,1:3} = Q \cdot R = \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ 0 & r_{22} & r_{23} \\ 0 & 0 & r_{33} \end{bmatrix} \tag{5}$$

where q_i denote the orthonormalized vectors of $P_{1:3,1:3}$, for instance $q_1 = P_{1:3,1} / \|P_{1:3,1}\|$, and $r_{i,j}$ are weighting factors found in the orthonormalization process[2].

Converting QR to RQ The QR decomposition factorizes $P_{1:3,1:3}$ as a unique product of an orthonormal (unity) matrix and an upper-triangular matrix, provided that the upper-triangular has the main diagonal entries all positive. Note that we want the reverse, i.e.

[2] See for example `http://www.tomzap.com/notes/MatricesM340L/Gram-Schmidt.pdf`.

an upper-triangular times an orthonormal, as in the projection equation Eq.1. We will see next that the QR factorization can be easily transformed to RQ.

Defining a matrix S as:

$$S = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

S has the property that left (right) multiplying a 3×3 matrix swaps its lines (columns). In addition $S^T = S$ and $S \cdot S = I$ (i.e. $S^{-1} = S$). Then applying QR factorization to $P^T_{1:3,1:3} \cdot S$,

$$P^T_{1:3,1:3} \cdot S = Q \cdot U$$

and doing some algebraic operations starting with a transpose and then using the properties of S, one has $P_{1:3,1:3} = S.U^T.Q^T = S.U^T.S.S.Q^T = (SU^TS) \cdot (SQ) = K \cdot R$. Note that this is already a RQ factorization as $K = S.U^T.S$ comprises the necessary line and column swapping to change the lower-left-triangular U^T to upper-triangular, and the $R = S.Q^T$ is still unity as the transpose and the operation of S do not affect the unity property.

Correcting the diagonal of K In general the QR and RQ factorizations can leave a sign ambiguity, which is removed by requiring that K has positive diagonal entries [7]. Defining a diagonal matrix, D having the signs of the main diagonal of K,

$$D = diag\{sign(K_{1,1}), sign(K_{2,2}), sign(K_{3,3})\}$$

one can correct the signs of K, and consequently update all the terms of the factorization with:

$$K \leftarrow K.D, \;\; R \leftarrow D.R, \;\; t = K^{-1}.P_{1:3,4}.$$

The algorithm of decomposing a projection matrix, symplified by using Matlab's QR factorization, is the following:

```
function [K, R, t]= proj_decomp(P)

% RQ from QR factorization
%
S= [0 0 1; 0 1 0; 1 0 0];
[Q,U]= qr(P(1:3,1:3)'*S); K=S*U'*S; R=S*Q';

% Correcting signs and computing t
%
D= diag(sign(diag(K)));
K= K*D;  R= D*R;  t= inv(K)*P(:,4);
```

4 RESULTS

In order to test our methodology, we built a simulated setup, comprising one mobile camera and three fixed cameras. The mobile camera moves in a way that its FOV has a large overlapping with the FOV of the fixed (networked) camera.

4.1 Simulated Setup

In this section we describe the VRML world built for our experiments. The VRML world is composed by one room with an exposition of Picasso paintings. The room is 10 meters long, 5 meters wide and 3 meters high. The paintings are located on the walls and spread in the middle of the room. See Fig. 2.

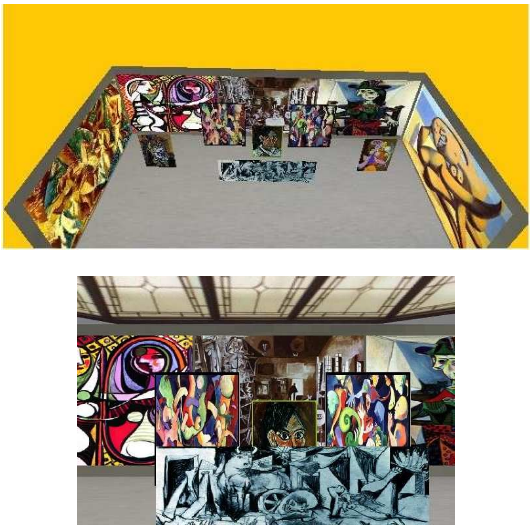


Figure 2: Simulated setup: VRML room with an exposition of Picasso paintings (top; removed ceiling and one wall), and an image acquired inside the room (bottom).

In our calibration methodology we use reconstructed 3D points, obtained in these experiments from a SLAM process using a demonstration toolbox[11]. In order to run the SLAM process, one needs the intrinsic parameters of the mobile camera.

The rendering of VRML depends significantly on the browser. One aspect that usually varies significantly is the size of the browsing window, which implies that the intrinsic parameters of the virtual camera also vary significantly.

In order to overcome this variability, we perform a calibration procedure as usual with a normal camera. More precisely, we place a calibration pattern in the scene and move it in the field of view of the camera. Then we use the Jean-Yves Bouguet's calibration toolbox [2].

The results of the calibration toolbox allow estimating the intrinsics matrix K with the structure shown in Eq.2. The intrinsics matrix allows then to realize euclidean reconstructions of image points matched or tracked in two or more images, and to calibrate the camera network. The following section shows results based on this idea.

4.2 Pose estimation

In this section we describe the estimation of the pose of three fixed cameras, given the images acquired by a (calibrated) mobile camera. The pose estimation is based on the calibration of fixed cameras as described in Sec.3.

The three static cameras are placed in a line 0.1 meters in front to the back wall, 2 meters above the floor. The cameras are separated by 2 meters. The middle camera is exactly in the center of the two most distant walls (i.e. 5 meters to the left and right walls). Fig. 3(a,b,c) shows the images acquired by the static cameras.

The camera on the robot moves on a line parallel to the baseline of the static cameras, 0.5 meters down and 0.4 meters ahead. Its first position is 3.5 meters from the left wall and it moves 3 meters more to the right. It captures one image every 0.1 meters. Fig. 3(d,e,f) shows the first, the second and the last images captured by the mobile camera.

After acquiring the static and the mobile images, we proceeded to detect and match SIFT features among all pairs of images, using the demonstration software [9]. The points identified by the SIFT are represented in the figures by yellow circles in Fig. 3(a,b,c,d,e,f).

Given the positions of the features in the images acquired by the mobile camera, we use the SLAM demonstration software[11] to estimate the 3D-position of the points in the world. The estimated 3D-positions of the points are represented in Fig. 3(g) by the green points. The purple points represent ground-truth data (reconstructed and ground-truth points are connected by blue lines). The ground-truth positions have been estimated by back-projecting the image points toward the known facets composing the VRML world. The alignment of the SLAM reconstruction and the VRML coordinate systems is done using a simple Procrustes procedure.

Finally, the reconstructed 3D points and their images in the fixed cameras are used to calibrate the fixed cameras. Fig. 3(g) shows the estimated positions and the orientations of the mobile camera (red), and of the static cameras (blue). Fig. 3(h) zooms the poses of the cameras. Ground-truth poses of the mobile and the static cameras, were again computed from the VRML, and are represented by cyan-lines (z axis) and black-dots (camera centers). The average error in the estimation position of cameras in each axis is less than 0.05 meters.

5 CONCLUSIONS

In this work we proposed a methodology for calibrating a network of cameras with non overlapping fields of view. The camera network is linked together by a mobile robot equipped with a camera. The accuracy of the estimated structure information of the scene is the key to our calibration procedure. Our experiments show that robot motions parallel to the baselines of the fixed cameras, with lengths on the order of magnitude of those of the baselines, provide good accuracy.

6 Acknowledgements

Research partly funded by the Portuguese FCT Programa Operacional Sociedade de Informação (POSI) in the frame of QCA III, the Portuguese FCT/ISR/IST plurianual funding through the POS Conhecimento Program that includes FEDER funds and the European Project FP6-EU-IST-045062 - URUS.

REFERENCES

[1] M. Antone and S. Teller. Scalable extrinsic calibration of omni-directional image networks. In *MIT Computer Graphics Group*.

[2] J. Bouguet. Demo software: Camera calibration toolbox for matlab. `http://www.vision.caltech.edu/bouguetj/calib_doc/`, 2008.

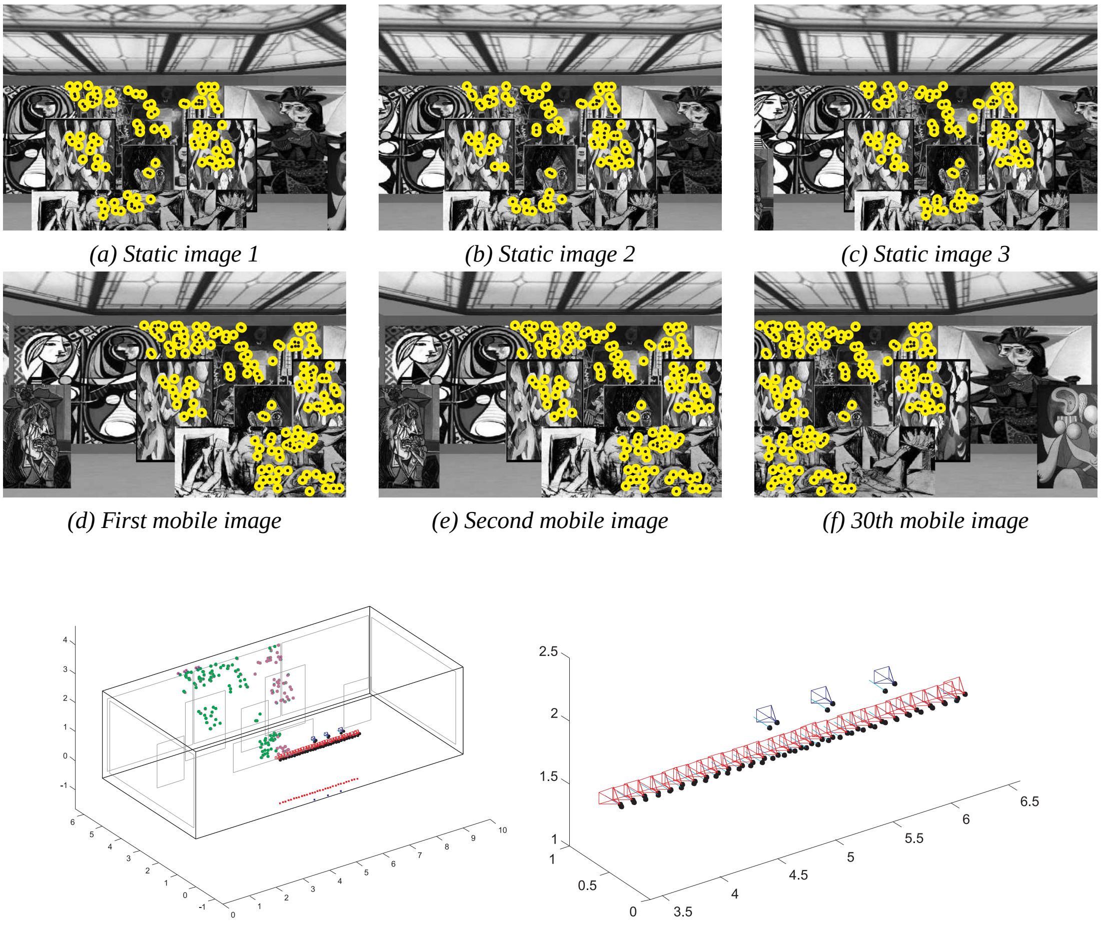
[3] S. Esquivel, F. Woelk, and R. Koch. Calibration of a multi-camera rig from non-overlapping views. In *Christian-Albrechts-University*.

[4] O. Faugeras. *Three-Dimensional Computer Vision - A Geometric Viewpoint*. MIT Press, 1993.

[5] L. Goncalves, E. Di Bernardo, D. Benson, M. Svedman, J. Ostrowski, N. Karlsson, and P. Pirjanian. A visual front-end for simultaneous localization and mapping. In *International Conference on Robotics and Automation*, 2005.

[6] N. Gracias and J. Santos-Victor. Underwater video mosaics as visual navigation maps. *Computer Vision and Image Understanding*, 79(1):66–91, 2000.

[7] R. I. Hartley and I. Zisserman. Multiple view geometry in computer vision. pages 150–152, 2000.



(a) Static image 1

(b) Static image 2

(c) Static image 3

(d) First mobile image

(e) Second mobile image

(f) 30th mobile image

(g) Ground truth and calibration results

(h) Detail of the positions of the cameras

Figure 3: Calibration experiment. Images acquired by the three static cameras of the camera network (a,b,c), and three sample images acquired by the mobile camera (d,e,f). The yellow circles with black dots represent SIFT features used for calibration. Sub-figure (g) shows **ground-truth** 3D points (purple) and mobile / fixed camera locations (represented by a cyan line and a black point), and **reconstructed** 3D points (green) and mobile / fixed camera locations (red / blue pyramid, black centers), these position of the cameras are projected in the floor (red/blue squares), superimposed on a wireframe representation of the scenario. Zoom of the locations of the ground-truth and reconstructed cameras (h).

[8] N. Karlsson, E. Di Bernardo, J. Ostrowski, L. Goncalves, P. Pirjanian, and M. Munich. The vslam algorithm for robust localization and mapping. In *International Conference on Robotics and Automation*, 2005.

[9] David G. Lowe. Demo software: Sift keypoint detector. `http://www.cs.ubc.ca/~lowe/keypoints/`.

[10] David G. Lowe. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision*, pages 91–110, 2004.

[11] J.M.M. Montiel, Javier Civera, and Andrew J. Davison. Demo software: Slam using monocular vision. `http://www.robots.ox.ac.uk/~SSS06/Website/index.html`, 2006.

[12] A. Sanfeliu and J. Andrade-Cetto. Ubiquitous networking robotics in urban settings. In *Dept. System Engineering and Automation, Universitat Politècnica de Catalunya (UPC)*.

[13] T. Sasaki and H. Hashimoto. Camera calibration using mobile robot in intelligent space. In *SICE-ICASE, 2006. International Joint Conference*, 2006.

[14] S. N. Sinha and M. Pollefeys. Towards calibrating a pan-tilt-zoom camera network. In *Department of Computer Science, University of North Carolina at Chapel Hill*.

[15] S. Teller, M. Antone, Z. Bodnar, M. Bosse, S. Coorg, M. Jethwa, and N. Master. Calibrated, registered images of an extended urban area. In *MIT Computer Graphics Group*.

[16] B. Triggs, P. McLauchlan, R. I. Hartley, and A. Fitzgibbon. Bundle adjustment – A modern synthesis. In *Vision Algorithms: Theory and Practice*.