Sensors **2014**, *14*, 13708-13729; doi:10.3390/s140813708

OPEN ACCESS
sensors
ISSN 1424-8220
www.mdpi.com/journal/sensors

Article

Calibration of an Outdoor Distributed Camera Network with a 3D Point Cloud

Agustín Ortega [1,*], Manuel Silva [2], Ernesto H. Teniente [1], Ricardo Ferreira [2], Alexandre Bernardino [2], José Gaspar [2] and Juan Andrade-Cetto [1,*]

[1] Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Llorens Artigas 4-6, Barcelona 08028, Spain; E-Mail: ehomar@iri.upc.edu

[2] Institute for Systems and Robotics, Instituto Superior Técnico, Av. Rovisco Pais 1, Lisbon 1049-001, Portugal; E-Mails: manuel.silva.thesis@gmail.com (M.S.); ricardo@isr.ist.utl.pt (R.F.); alex@isr.ist.utl.pt (A.B.); jag@isr.ist.utl.pt (J.G.)

* Authors to whom correspondence should be addressed; E-Mails: aortega@iri.upc.edu (A.O.); cetto@iri.upc.edu (J.A.-C.); Tel.: +34-93-4010-775 (J.A.-C.).

Received: 5 March 2014; in revised form: 17 July 2014 / Accepted: 17 July 2014 / Published: 29 July 2014

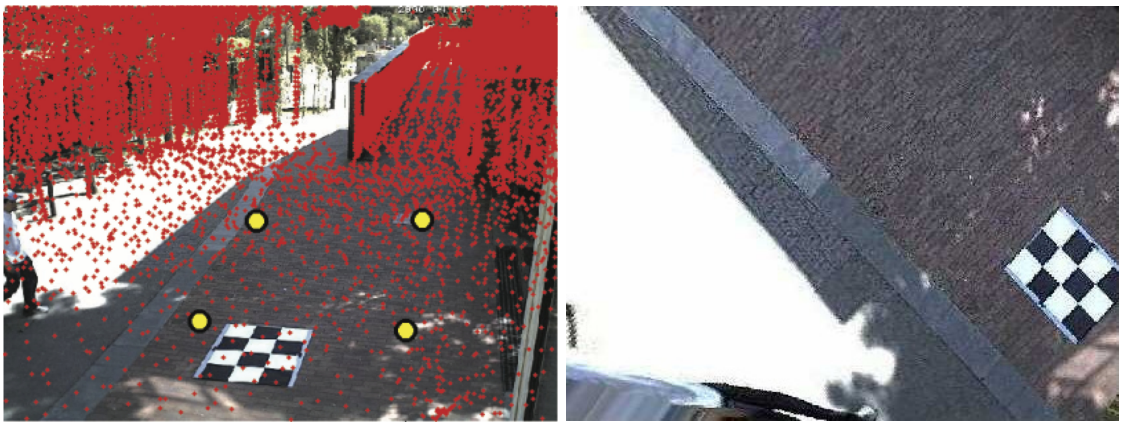
Abstract: Outdoor camera networks are becoming ubiquitous in critical urban areas of the largest cities around the world. Although current applications of camera networks are mostly tailored to video surveillance, recent research projects are exploiting their use to aid robotic systems in people-assisting tasks. Such systems require precise calibration of the internal and external parameters of the distributed camera network. Despite the fact that camera calibration has been an extensively studied topic, the development of practical methods for user-assisted calibration that minimize user intervention time and maximize precision still pose significant challenges. These camera systems have non-overlapping fields of view, are subject to environmental stress, and are likely to suffer frequent recalibration. In this paper, we propose the use of a 3D map covering the area to support the calibration process and develop an automated method that allows quick and precise calibration of a large camera network. We present two cases of study of the proposed calibration method: one is the calibration of the Barcelona Robot Lab camera network, which also includes direct mappings (homographies) between image coordinates and world points in the ground plane (walking areas) to support person and robot detection and localization algorithms. The second case consist of improving the GPS positioning of geo-tagged images taken with a mobile device in the Facultat de Matemàtiques i Estadística (FME) patio at the Universitat Politècnica de Catalunya (UPC).

Keywords: camera network calibration

1. Introduction

The development of powerful laser sensors combined with simultaneous location and mapping (SLAM) methodologies [1] allows the possibility to have available high precision 3D maps registered over large areas [2]. These maps come usually in the form of large point clouds and are typically used to support robot navigation systems [3]. These point clouds are usually acquired with laser range finders, cover the complete area of the network and, in particular, contain the areas corresponding to the fields of view of the cameras. This paper exploits these technologies, proposing a methodology to calibrate an outdoor, non-overlapped, distributed camera network using such range data. See Figure 1.

Figure 1. Results of the proposed calibration system. (**Left**) Plane selection in a graphical user interface and registration of the laser range data with a view from one of the cameras in the network; (**Right**) recovered orthographic view of the ground plane. The chess pattern shown is not used for calibration; it serves just to visually evaluate the quality of the ground-plane rectifying homography.



Traditional techniques for camera calibration require the use of non-planar [4] or planar [5] patterns, usually made of points, lines or checkerboards [6,7], conics [8] or, even, ARTagmarkers [9]. Unfortunately, for large outdoor camera networks, calibration patterns of reasonable sizes often project on images with very small resolution, mainly because cameras are usually located at a considerable height with respect to the floor, consequently making pattern segmentation difficult. In addition, a pattern-based independent calibration of each camera would require a secondary process to relate all camera coordinate systems to a global reference frame, but establishing this relation with small to null overlapping fields of view is nearly impossible. For planar scenarios, a direct linear transformation (DLT) [10] suffices to estimate image to plane homographies [11]. Unfortunately, the planar scenario assumption is too restrictive, especially in situations with nonparallel, locally planar surfaces, such as ramps and plazas, which often occur in real urban environments, as in our case.

An interesting technique to calibrate the camera network without the need of a pattern is with the aid of a bright moving spot [12]. The technique assumes overlapping fields of view to estimate the epipolar

geometry to extract homographies, estimate depth and, finally, compute the overall calibration of the camera network. In our case, the cameras' fields of view seldom overlap, and the visibility of the bright spot does not always hold in sunlight. Another alternative is to place an LED light on a moving robot and to track it with a secondary robot equipped with a laser sensor, relating their position estimates to the camera network [13]. Yet another system that relies on tracking a moving object to estimate the extrinsic parameters is [14], which assumes a constant velocity model for the target. Tracking a moving target each time the system needs recalibration might be prohibitive. The estimation of the camera location purely by analyzing cast shadows is also mathematically possible, but with very low position accuracy in practice [15], and if one is interested only in the topology of the network configuration and not in a metric calibration, multi-target tracking of people could also be an alternative [16]. In contrast to these approaches, we opt for a system that does not rely explicitly on a moving pattern or shadow to calibrate the network and that produces an accurate metric calibration.

For camera network systems that incorporate camera orientation control (pan and tilt) and motorized zoom, it is possible to use such motion capabilities to first estimate the intrinsic parameters rotating and zooming, fitting parametric models to the optical flow, and then to estimate extrinsic parameters aligning landmarks to image features [17]. Unfortunately, in our case, the cameras are not active. Another option is to use stereo pairs instead of monocular cameras at each node in the network. In this way, local 3D reconstruction can be obtained directly within each node and registered globally using graph optimization techniques [18]. Overlapping fields of view are still necessary in that case. A third option to calibrate the camera network, albeit relative translation, is to use a vertical vanishing point and the knowledge of a line in a plane orthogonal to the vertical direction on each camera image [19]. A different, but related, problem is the relative positioning of one or more cameras with respect to a range sensor. To that end, calibration can be achieved using a checkerboard pattern, as in [20]. A related methodology to calibrate extrinsically an omnidirectional camera using point correspondences between a laser scan an the camera image is proposed in [21]. In contrast to our approach, the method assumes known intrinsic camera parameters. For a method to calibrate the laser intrinsic parameters instead, the reader is referred to [22].

We benefit from the availability of a dense point cloud acquired during a 3D laser-based SLAM session with our mapping mobile robot [3]. The set contains over eight million points and maps the environment with accuracies that vary from 5 cm to 20 cm, approximately. These data replace the need for a checkerboard pattern, a tracked beam, a robot or active capabilities of the cameras and are used as external information to calibrate the network.

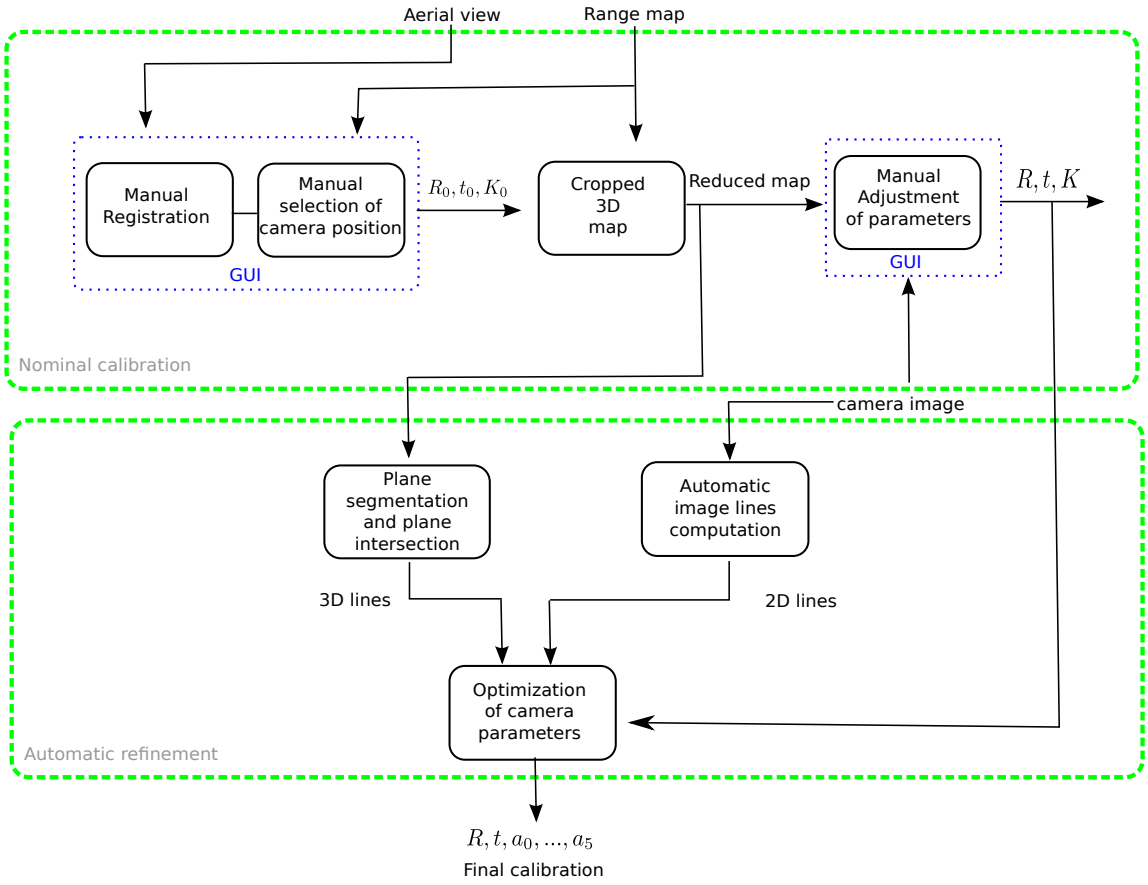
Our work is largely related in spirit to the method described in [23], in which a set of images are registered into a urban point cloud. One major difference of the approach is on the assumptions made with respect to the characteristics of the scene during 3D feature extraction. In particular, the above-mentioned technique exploits the fact that buildings have strong parallel edges and that these cluster with similar orientation. On the contrary, we exploit the fact that in urban scenes, large planar regions also meet at long straight edges. In contrast with [24], in which edge parallelism is used to calibrate only the attitude and focal length of cameras for traffic monitoring, we use edge information to calibrate also the camera location.

The rest of the paper is organized as follows. We explain first how nominal calibration is executed and then how this calibration is refined, first by extracting 3D features from the point clouds and optimizing over the reprojection error of their matching to those found in images. When needed, a final refinement step is computed by means of the DLT-lines algorithm. Experiments on a pair of scenarios are presented to show the feasibility of the proposed solution. The paper is concluded with some remarks and future work.

2. Nominal Calibration

The proposed calibration procedure is illustrated in Figure 2. It consists of two main steps. In the first step, the internal camera parameters are initialized to the manufacturer specs (image pixel width and depth and focal length), and a nominal calibration of the camera external parameters is obtained by manually registering the point cloud to an aerial image of the experimental site with the aid of a graphical user interface, prompting the user to coarsely specify the camera location, orientation, height and field of view. These initial parameters allow the cropping of the full point cloud into smaller regions of interest compatible with the field of view of each camera. The user can then adjust the registration by manually modifying each of the parameters (see the video associated with [25], available in the IEEE Xplore digital library).

Figure 2. Two-step calibration methodology. In the first step, a graphical user interface is used to assist in an initial manual registration of the point cloud. The second step refines this registration, matching 2D image lines with 3D edges in the point cloud.



In the second step, an automatic refinement of the camera calibration parameters is obtained by matching 2D image lines to their corresponding 3D edges in the point cloud. To this end, the point cloud is segmented into a set of best fitting planes with large support using local variation [26], and straight edges are computed from the intersection of perpendicular planes in the set. The extracted 3D lines are associated with 2D image lines, and this information is fed to a non-linear optimization procedure that improves both intrinsic and extrinsic parameters. Finally, the homographies of the walking areas are computed for the planar regions in the range data. The final output of the whole calibration procedure consists in: (1) the extrinsic camera parameters, *i.e.*, the position and orientation of each camera in the world frame; (2) the intrinsic camera parameters (focal distance, image center aspect ratio and distortion terms); and (3) the homographies of each walking area.

The first step of the calibration procedure needs to be performed only once, during the camera network installation or when the network topology changes, *i.e.*, cameras are added/moved, and takes only a couple of minutes. The second step, which does not require user intervention, can be executed as frequently as needed to keep the system calibrated, despite small modifications in camera orientation due to weather conditions and maintenance operations. In the following paragraphs, we detail the nominal calibration.

For the nominal calibration, the user interacts with a graphical user interface to coarsely locate each camera with respect to the reference frame of the point cloud and the viewing direction in the ground plane. For an easier interaction, the point cloud appears registered with an aerial view of the environment. The registration of the point cloud with the aerial view is computed manually using the DLT [10]. The camera parameters are initialized with default intrinsic parameters.

We then compute initial values for the camera pose: the center of projection $\mathbf{t}$ is simply the user selected point $\mathbf{p}_1$, located at a user define height (*i.e.*, 6 m in our application). See Figure 3. The two selected points $\mathbf{p}_1$ and $\mathbf{p}_2$ set the azimuth direction ψ. The elevation ϕ gives a user-defined inclination to the ground (17° in the shown example), and the roll ρ is set to π to account for the proper axes changes. These parameters suffice to compute the initial rotation matrix $\mathbf{R} = \mathbf{R}_\rho \mathbf{R}_\psi \mathbf{R}_\phi$ with:

$$\mathbf{R}_\psi = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \; \mathbf{R}_\phi = \begin{bmatrix} \cos(\phi) & 0 & -\sin(\phi) \\ 0 & 1 & 0 \\ \sin(\phi) & 0 & \cos(\phi) \end{bmatrix}, \text{ and } \mathbf{R}_\rho = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix}$$

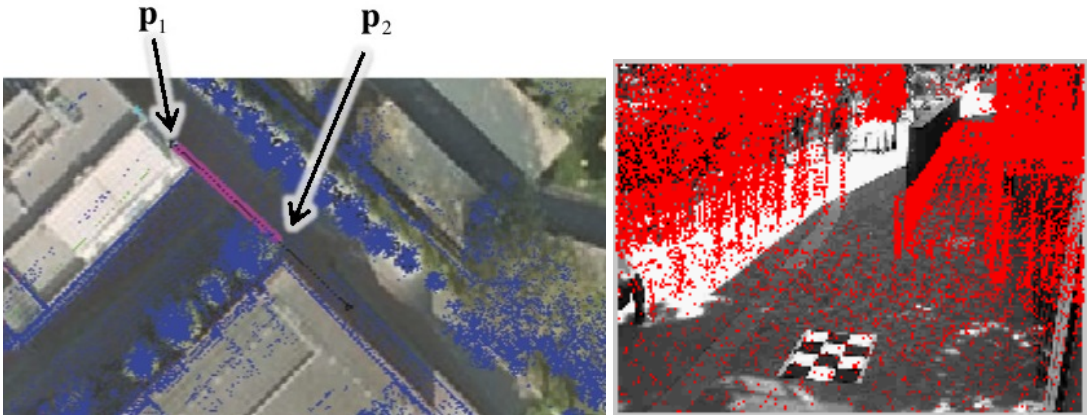
Assuming that the principal point (u_0, v_0) is located at the image center, we can compute an initial estimate for the camera intrinsic parameters using as input the focal length f and the CCD pixel size in millimeters k_u and k_v, *i.e.*, $\alpha_v = f k_v$, $\alpha_u = f k_u$, and:

$$\mathbf{K} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

The initial estimate of the perspective projection matrix for each camera is:

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}] \tag{2}$$

Figure 3. Graphical user interface. (**Left**) The point cloud is shown to the user overlaid on top of an aerial view of the environment. The user is prompted to select (1) a coarse camera location $\mathbf{p}_1$; and (2) the viewing direction $\mathbf{p}_2$ indicated by the magenta line; (**Right**) During the initialization process, the user can manually adjust intrinsic and extrinsic parameters on a projected view of the point cloud.



Once this initial estimate is obtained for a particular camera, the user can further adjust each parameter manually, whilst a projection of a cropped section of the point cloud that falls within the viewing frustum is visualized in the image. Note, however, that this initial estimate does not take into account radial distortion parameters. These are refined along with the rest of parameters in the second step of the method.

3. Calibration Refinement

To improve camera calibration from the nominal parameters, we propose an automatic method, where relevant 3D edges are extracted from the point cloud and matched to corresponding image lines. In practice, the method works well with about a half-dozen lines selected from each camera image.

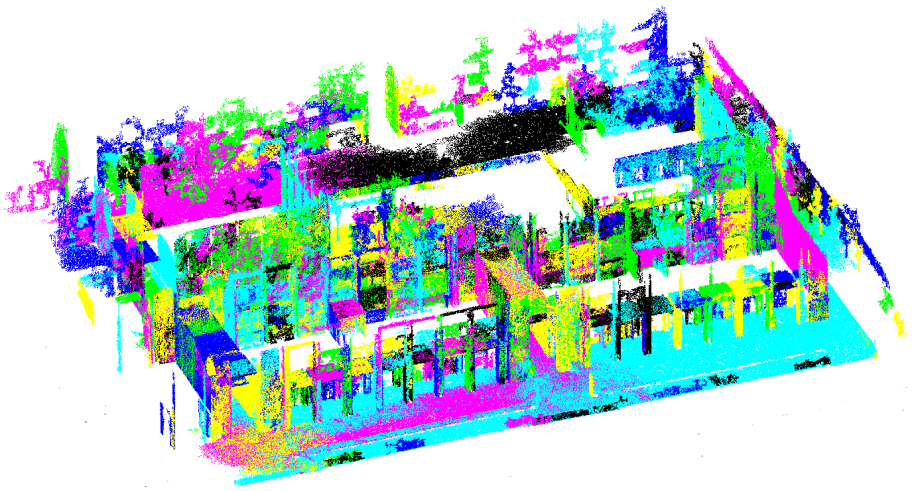
The procedure uses the nominal calibration as an initial rough approximation and can be executed anytime to recover from small perturbations in camera orientation or any other parameter, due to weather (wind, rain, *etc*.) or maintenance operations (repair, cleaning).

3.1. 3D Edge Computation

The computation of straight lines from the point cloud relies on identifying and intersecting planes. The method to segment planar regions is motivated by Felzenszwalb's algorithm to 2D image segmentation [27] and extended to deal with non-uniformly sampled 3D range data [26]. The algorithm sorts point to point distances for each point's k-nearest neighbors and then traverses the list of sorted distances in increasing order, growing planar patches by joining points that meet two matching criteria, *i.e.*, distance constraints and orientation constraints. Thanks to the use of union by rank and path compression [28], the algorithm runs nearly in linear time with respect to the number of points in the point cloud. The ANN library proved an adequate tool for the computation of approximate nearest neighbors [29]. If computation time becomes an issue, libnabo could be used as an alternative [30].

Figure 4 shows an example of the segmentation results. In the figure, all points corresponding to the same planar patch are represented with the same color. Note, however, that even when colors repeat for different planes, their labels are different.

Figure 4. Segmentation of a point cloud into planar regions. Colors represent individual planar regions detected by the algorithm.



In the segmentation algorithm, local surface normals $\mathbf{n}$ are computed for each point in the point cloud, fitting local planar patches. To account for global variation, planar patches are merged, growing a forest of trees based on curvature and mean distance. The curvature between two candidate regions is computed from the angle between their two normals, which must be below a user-selected threshold t_c,

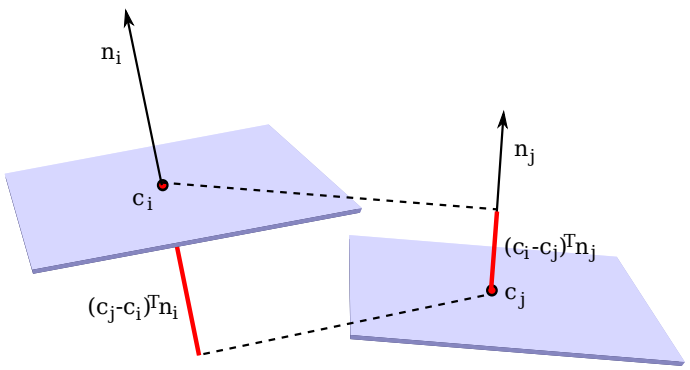
$$|\arccos(\mathbf{n}_i^T \mathbf{n}_j)| < t_c \tag{3}$$

Two segments passing the curvature criteria are merged if they also pass a distance constraint. That is, if the sum of distances between their centers along weighted orthogonal directions is below a user-selected threshold t_d,

$$\frac{k_i \mid (\mathbf{c}_i - \mathbf{c}_j)^T \mathbf{n}_j \mid + k_j \mid (\mathbf{c}_j - \mathbf{c}_i)^T \mathbf{n}_i \mid}{k_i + k_j} < t_d \tag{4}$$

with k_i and k_j the number of points each segment holds and $\mathbf{c}_i$ and $\mathbf{c}_j$ the segment centers. See Figure 5.

Figure 5. Projection of the region centers $\mathbf{c}_i$ and $\mathbf{c}_j$ onto neighboring planar patches.



Once a set of segments is obtained, their intersecting edges are computed, and the ones with sufficient support from their generating planes and with good orthogonality conditions are selected for projection onto the images. The method is summarized in Algorithm 1.

Algorithm 1 The algorithm to find edge lines at orthogonal plane intersections within the point cloud.

EDGEEXTRACTION(M)
INPUT:
M: Point cloud.
OUTPUT:
E: 3D lines.

$D \leftarrow \emptyset$
for each $\mathbf{p}_i \in M$ **do**
  $N_i \leftarrow$ FINDNEIGHBORS($\mathbf{p}_i, M$)
  $\mathbf{n}_i \leftarrow$ COMPUTENORMAL($\mathbf{p}_i, N_i$)
  LABEL($\mathbf{p}_i$) $\leftarrow i$
  $D \leftarrow D \cup$ FINDDISTANCES($\mathbf{p}_i, N_i$)
end for
$D \leftarrow$ SORTDISTANCES(D)
for each $d_k \in D$ **do**
  $\mathbf{c}_i \leftarrow$ START(d_k)
  $\mathbf{c}_j \leftarrow$ END(d_k)
  if LABEL($\mathbf{c}_i$) $\neq$ LABEL($\mathbf{c}_j$) **then**
    if $|\cos^{-1}(\mathbf{n}_i^T \mathbf{n}_j)| < t_c$ **then**
      if $\frac{k_i|(\mathbf{c}_i-\mathbf{c}_j)^T\mathbf{n}_j|+k_j|(\mathbf{c}_j-\mathbf{c}_i)^T\mathbf{n}_i|}{k_i+k_j} < t_d$ **then**
        MERGETREES($\mathbf{c}_i, \mathbf{c}_j$)
      end if
    end if
  end if
end for
$E \leftarrow \emptyset$
$L \leftarrow$ LABELS(M)
for each pair of segments $(\mathbf{s}_i, \mathbf{s}_j) \in L$ with respective $(\mathbf{n}_i, \mathbf{n}_j)$ **do**
  if ORTHOGONAL($\mathbf{n}_i, \mathbf{n}_j$) **then**
    $E \leftarrow E \cup$ PLANEINTERSECTION($\mathbf{n}_i, \mathbf{n}_j$)
  end if
end for

3.2. Optimization

Straight image lines are extracted from the camera images using the method proposed in [31]. The line set is pruned to those lines larger than a predefined threshold.

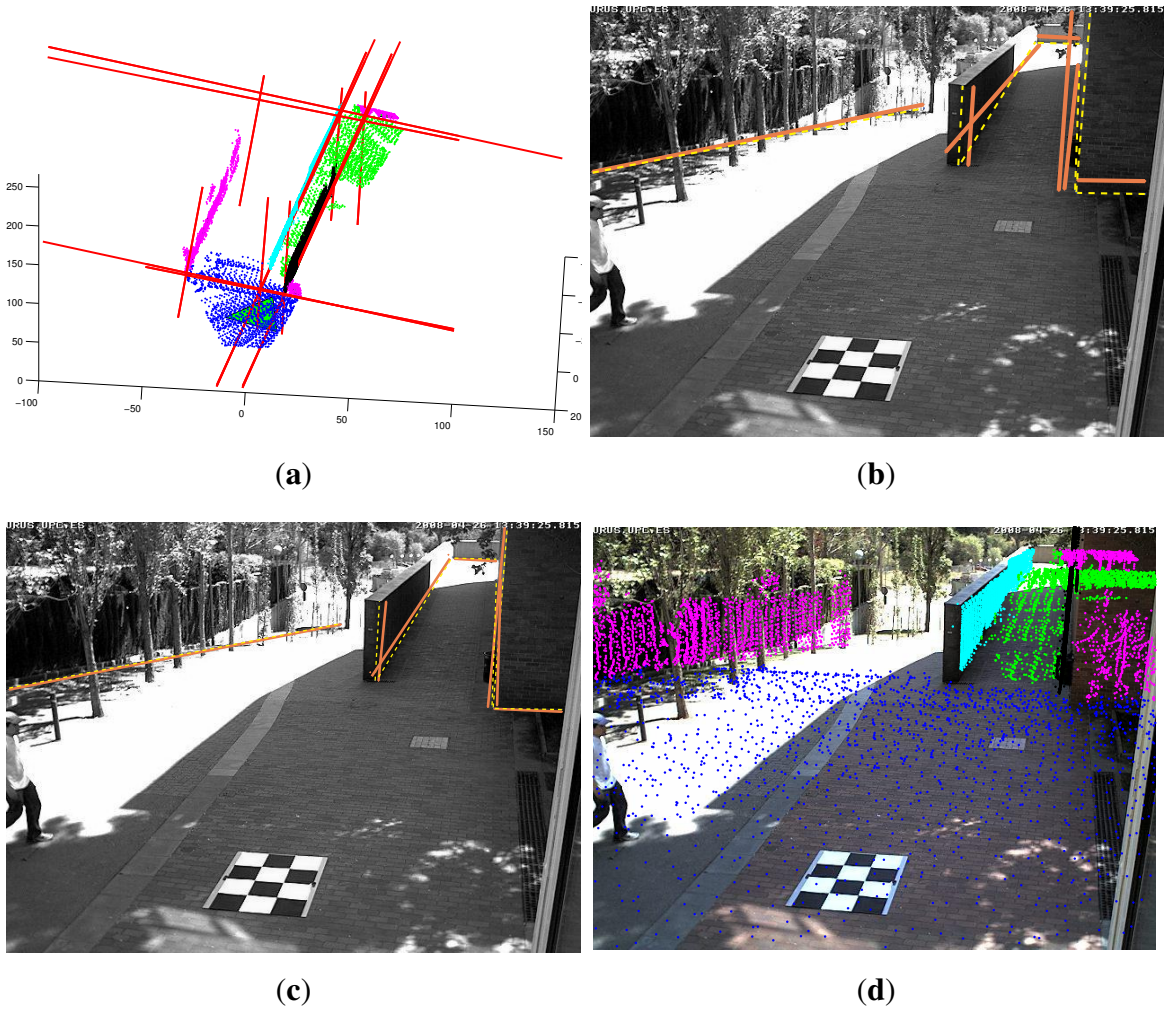
3D model lines are projected onto the image plane using the projection matrix computed during the nominal calibration step. 2D-3D line association is attained by matching such projections to the closest 2D image line.

Once the 3D-2D association is established, nonlinear optimization is performed to improve the nominal calibration by minimizing the squared sum of the line endpoint projection errors:

$$\underset{\vartheta}{\text{minimize}} \sum_{i} \|\mathbf{u}_i - \mathbf{u}_{id}(\vartheta)\|^2 \tag{5}$$

where $\mathbf{u}_{id}(\vartheta)$ is the distorted projection of the 3D endpoint $\mathbf{p}_i$, $\vartheta = (\mathbf{K}, \mathbf{R}, \mathbf{t}, a_1, a_2)$ are the set of parameters being optimized and $\mathbf{u}_i$ is the image point. The optimization is solved using Levenberg–Marquardt nonlinear optimization. See Figure 6.

Figure 6. Optimization. (**a**) Computation of plane intersections in the range data; (**b**) Projection of lines onto the image plane using the nominal calibration parameters; (**c**) line matching optimized in the image plane; (**d**) segmented point cloud reprojected onto the calibrated image.



(**a**) (**b**)

(**c**) (**d**)

In this step of the method, image distortion is modeled based on even powers of the radial distance in the image plane:

$$\mathbf{u}_{id} = \mathbf{u}_{in} + (1 + \sum_{j=1}^{2} a_j r^{2j})(\mathbf{u}_{in} - \mathbf{u}_0) \tag{6}$$

where a_j are the distortion parameters, $r^2 = \|\mathbf{u}_{in} - \mathbf{u}_0\|^2$ is the radial distortion factorization, $\mathbf{u}_0$ is the computed principal point, $\mathbf{u}_{in}$ is the normalized (pinhole) image projection of point $\mathbf{p}_i$:

$$\begin{bmatrix} \mathbf{u}_{in} \\ 1 \end{bmatrix} \sim \mathbf{K}[\mathbf{R}|\mathbf{t}] \begin{bmatrix} \mathbf{p}_i \\ 1 \end{bmatrix} \tag{7}$$

and $\sim$ denotes equality up to a scale factor.

3.2.1. Initialization of the Optimization

The nominal calibration introduced in Section 2 is, in general, sufficient to initialize the calibration optimization formalized in Equation (5). However, one finds that while it is usually simple and very effective to observe precisely some parameters as the camera horizontal position in the coordinates of a laser range finder map, other parameters, such as camera height, 3D rotation, focal length or principal point, are more challenging. The possibility of automating the initialization of the optimization procedure is a convenient feature for calibrating cameras in a network.

The initialization of the calibration optimization process may be setup to find from zero till all of the calibration parameters. Zero means using all of the nominal calibration results for initializing the optimization. All means (re-)finding all initialization parameters from image and laser range finder data. In between are several cases of interest, many of which have solutions published. For example, if a camera has its intrinsic parameters calibrated before being mounted in place, then one just has to estimate the extrinsic parameters by solving the well-known Perspective-n-Point (PnP) problem [32]. In the following, we detail two cases. In the first case, we show how to estimate all of the parameters from 3D lines imaged by the camera to calibrate. In the second case, we consider that the camera position is precisely known and detail how to estimate the intrinsic parameters and 3D rotation.

As proposed in [33], the use of image lines instead of isolated points in the camera calibration process brings an advantage. Image processing can be used for fine tuning the location of the lines in the image and therefore automatically improving the calibration data input. In this section, *DLT-Lines* is presented as a method to initialize the optimization step, allowing one to estimate simultaneously the camera projection matrix and radial distortion, from the 3D point cloud and 2D lines.

Considering the shorthand notation for image points $\mathbf{m}_i = [\mathbf{u}_i^T \ 1]^T$ and 3D points $\mathbf{M}_i = [\mathbf{p}_i^T \ 1]^T$ the perspective camera model, Equation (7), becomes $\mathbf{m}_i \sim \mathbf{P}\mathbf{M}_i$.

The projection of a 3D line $\mathbf{L}_i$ to the camera image plane can be represented by the cross product of two image points in projective coordinates:

$$\mathbf{l}_i = \mathbf{m}_{1i} \times \mathbf{m}_{2i} \tag{8}$$

Any point $\mathbf{m}_{ki}$ lying in the line $\mathbf{l}_i$ implies that $\mathbf{l}_i^T \ \mathbf{m}_{ki} = 0$. Hence, applying the multiplication of $\mathbf{l}_i^T$ to both sides of the perspective camera model, *i.e.*, $\mathbf{l}_i^T \ \mathbf{m}_{ki} = \mathbf{l}_i^T \ \mathbf{P} \ \mathbf{M}_{ki}$, leads to:

$$\mathbf{l}_i^T \ \mathbf{P} \ \mathbf{M}_{ki} = 0 \tag{9}$$

where $\mathbf{M}_{ki}$ is a 3D point in projective coordinates lying in $\mathbf{L}_i$. The properties of the Kronecker product [34] allow one to obtain a form factorizing the vectorized projection matrix:

$$(\mathbf{M}_{ki}^T \otimes \mathbf{l}_i^T) \ vec(\mathbf{P}) = 0 \tag{10}$$

Considering $N \geq 12$ pairs $(\mathbf{M}_{ki}, \mathbf{l}_i)$, one forms a matrix $\mathbf{B}$, $N \times 12$, by stacking the N matrices $\mathbf{M}_{ki}^T \otimes \mathbf{l}_i^T$. An example of $N = 12$ arises when one observes six 2D lines imaging six 3D lines, L_i $(i = 1, ..., 6)$, each one represented by two end points, $L_i \leftrightarrow (M_{i1}, M_{i2})$. Alternatively, given a 3D line $\mathbf{L}_i$ and its projection represented by the image line $\mathbf{l}_i$, any 3D point lying on the 3D line $\mathbf{L}_i$ can be paired with the 2D line $\mathbf{l}_i$. On the other hand, any image line $\mathbf{l}_i$ can be paired with any 3D point lying on $\mathbf{L}_i$, *i.e.*, more than one image line can be paired with a 3D point.

The least squares solution, more precisely the minimizer of $\|\mathbf{B}\ vec(\mathbf{P})\|^2$ subjected to $\|vec(\mathbf{P})\| = 1$, is the right singular vector corresponding to the least singular value of $\mathbf{B}$.

Note that the perspective camera model, as presented in Equation (7), does not contain yet the radial distortion. To include radial distortion, we use Fitzgibbon's division model [35]. An undistorted image point, $\mathbf{u} = [u\ \ v]^T$, is computed from a radially distorted image point $\mathbf{u}_d = [u_d\ \ v_d]^T$ as $\mathbf{u} = \mathbf{u}_d/(1 + \lambda \|\mathbf{u}_d\|^2)$, where λ represents the radial distortion parameter. The division model allows one to define a line $\mathbf{l}_{12}$ as the cross product of two points:

$$\mathbf{l}_{12} = \begin{bmatrix} u_{1d} \\ v_{1d} \\ 1 + \lambda s_1^2 \end{bmatrix} \times \begin{bmatrix} u_{2d} \\ v_{2d} \\ 1 + \lambda s_2^2 \end{bmatrix} = \hat{\mathbf{l}}_{12} + \lambda \mathbf{e}_{12} \tag{11}$$

where s_i is the norm of distorted image point i, $s_i^2 = u_{id}^2 + v_{id}^2$, the distorted image line is denoted as $\hat{\mathbf{l}}_{12} = [u_{1d}\ v_{1d}\ 1]^T \times [u_{2d}\ v_{2d}\ 1]^T$ and the distortion correction term $e_{12} = [v_{1d}s_2^2 - v_{2d}s_1^2,\ u_{2d}s_1^2 - u_{1d}s_2^2,\ 0]^T$.

Applying Equation (11) on Equation (10) leads to the following equation:

$$\left(\mathbf{M}_{k12}^T \otimes (\hat{\mathbf{l}}_{12} + \lambda \mathbf{e}_{12})^T\right) vec(\mathbf{P}) = 0 \tag{12}$$

which can be rewritten as:

$$(\mathbf{B}_{ki1} + \lambda \mathbf{B}_{ki2})\ vec(\mathbf{P}) = 0 \tag{13}$$

where $\mathbf{B}_{ki1} = \mathbf{M}_{k12}^T \otimes \hat{\mathbf{l}}_{12}^T$, $\mathbf{B}_{ki2} = \mathbf{M}_{k12}^T \otimes \mathbf{e}_{12}^T$ and $\mathbf{M}_{k12}$ denotes the $k - th$ 3D point projecting to the distorted line $\hat{\mathbf{l}}_{12}$.

To solve Equation (13) instead of Equation (10), we still need to consider $N \geq 12$ pairs $(\mathbf{M}_{ki}, \hat{\mathbf{l}}_i)$, where $N = k_{max} i_{max}$, and form now two $N \times 12$ matrices, $\mathbf{B}_1$ and $\mathbf{B}_2$, instead of just $\mathbf{N}$, by stacking matrices $\mathbf{B}_{ki1}$ and $\mathbf{B}_{ki2}$. Left-multiplying the stacked matrices by $\mathbf{B}_1^T$ results in a polynomial eigenvalue problem (PEP), which can be solved, for example, in MATLAB using the `polyeig` function. It gives, simultaneously, the projection matrix, in the form of $vec(\mathbf{P})$, and the radial distortion parameter λ.

Having estimated the projection matrix, $\mathbf{P}$, the camera intrinsic and extrinsic parameters can be obtained by QR-decomposition [10]. More precisely, given the sub-matrix $\mathbf{P}_{3\times3}$ containing the first three columns of $\mathbf{P}$ and $\mathbf{S}$ an anti-diagonal matrix:

$$\mathbf{S} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{14}$$

the QR-decomposition allows factorizing ${\mathbf{P}_{3\times3}}^T \mathbf{S} = \mathbf{Q}\mathbf{U}$, where $\mathbf{Q}$ is an orthogonal matrix and $\mathbf{U}$ is an upper triangular matrix. Then, the intrinsic parameters and the rotation matrices are computed

as $\mathbf{K} = -\mathbf{S}\mathbf{U}^T\mathbf{S}$ and $\mathbf{R} = \mathbf{Q}^T\mathbf{S}$. Finally, the camera position is obtained with $\mathbf{t} = \mathbf{K}\mathbf{P}_4$, where $\mathbf{P}_4$ is a 3×1 vector containing the fourth column of $\mathbf{P}$. If the diagonal of $\mathbf{K}$ contains negative values, then it is corrected by post-multiplying by a diagonal matrix. In MATLAB/Octave `D= diag(sign(diag(K))); K= K*D; R= D*R; t= D*t;`. In addition, since $\pm\mathbf{P}$ are both solutions of Equation (10), the factorization of $\mathbf{P}$ may imply $det(\mathbf{R}) = -1$. If $det(\mathbf{R}) = -1$, then the factorization of $\mathbf{P}$ is repeated using $-\mathbf{P}$.

To convert the obtained distortion parameter λ to the distortion parameters in Equation (6), we sample the region of interest and find a least squares solution for the best parameter fit in this region. Starting from a set of camera points evenly sampled at pixel granularity covering the image dimensions $\{\mathbf{u}_{id}\}$, we apply the Fitzgibbon distortion model to obtain an uneven set of undistorted pixel coordinates $\{\mathbf{u}_i\}$. We next solve the optimization problem:

$$\underset{a_j}{\text{minimize}} \sum_i \left\| \mathbf{u}_0 + \left(1 + \sum_{j=1}^{2} a_j r^{2j}\right)(\mathbf{u}_i - \mathbf{u}_0) - \mathbf{u}_{id} \right\|^2 \tag{15}$$

which is a linear least squares problem in the variables a_j, where a closed form solution is available. For small distortions, we empirically find that $a_1 = \lambda$ and $a_2 = 0$ provide a good fit to initialize the main optimization algorithm.

3.2.2. Known Camera Location

In the case one knows accurately the camera location, e.g., the camera has been imaged by the 3D data acquisition system, then the number of degrees of freedom of the calibration problem is decreased. The DLT methodology presented in the previous section can be further simplified.

Subtracting the camera center to all points of the point cloud results in a coordinate system where the camera is at the origin, and thus, the projection matrix, $\mathbf{P} = \mathbf{K}[\mathbf{R}|\ \mathbf{t}]$ is equivalent to a simple homography, $\hat{\mathbf{P}} = \mathbf{K}\mathbf{R}$. Considering image lines $\mathbf{l}_i$ and 3D points, $\mathbf{p}_{ki} = [x_{ki}\ y_{ki}\ z_{ki}]^T$, imaged as points of the lines, recalling Equation (9), one has:

$$\mathbf{l}_i^T\mathbf{K}\mathbf{R}(\mathbf{p}_{ki} - {}^W\mathbf{t}_C) = 0 \tag{16}$$

where ${}^W\mathbf{t}_C$ denotes the camera projection center in world coordinates. As such, one obtains linear constraints similar to the ones already derived for *DLT-Lines*:

$$\left((\mathbf{p}_{ki} - {}^W\mathbf{t}_C)^T \otimes \mathbf{l}_i^T\right)\ vec(\mathbf{K}\mathbf{R}) = 0 \tag{17}$$

The length of $vec(\mathbf{K}\mathbf{R})$ is just nine, *i.e.*, the knowledge of the camera location saves three variables to estimate, and thus, the estimation process is intrinsically simplified. Finally, the projection matrix, $\mathbf{P}$, can be obtained decomposing $\hat{\mathbf{P}} = \mathbf{K}\mathbf{R}$ and adding the camera location as $\mathbf{P} = [\hat{\mathbf{P}} \mid \hat{\mathbf{P}}\ {}^W\mathbf{t}_C]$.

The calibration problem has been reduced to the estimation of a homography, represented by $vec(\mathbf{K}\mathbf{R})$ and, therefore, not including radial distortion. A similar form based on Equation (17) can be obtained for the radial distortion case represented in Equation (12):

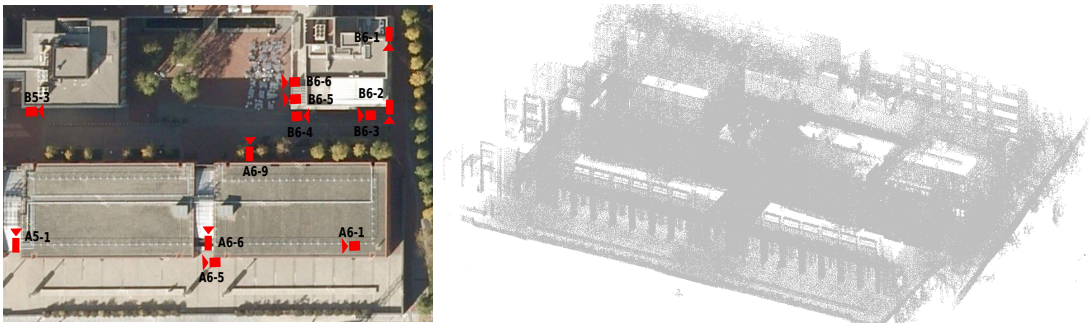
$$\left(\tilde{\mathbf{p}}_{k12}^T \otimes (\hat{\mathbf{l}}_{12} + \lambda\mathbf{e}_{12})^T\right)\ vec(\mathbf{K}\mathbf{R}) = 0 \tag{18}$$

where $\tilde{\mathbf{p}}_{k12} = (\mathbf{p}_{k12} - {}^{W}\mathbf{t}_{C})$. Equation (18) can be re-written in the form of Equation (13), which can be used to estimate the camera projection matrix $\mathbf{P}$ and radial distortion parameter λ, as shown before.

4. Experiments

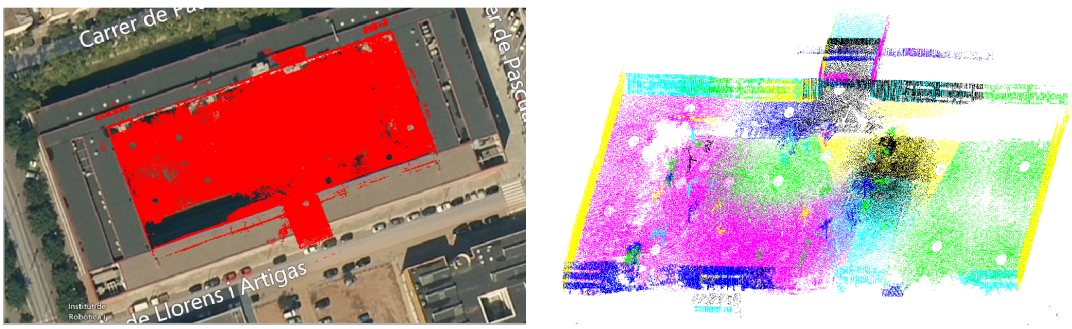
To show the cogency of the proposed calibration method, we show calibration results of two different outdoor scenarios. In both cases, the range data was gathered using a custom-built 3D laser with a Hokuyo UTM-30LX scanner mounted in a slip-ring. Each scan has 194,580 points with a resolution of 0.5° azimuth and 0.25° elevation. The first dataset was acquired in the Barcelona Robot Lab (BRL), located at the Campus Nord of the Universitat Politècnica de Catalunya (UPC). The BRL is a 10,000 m^2 facility for research in outdoor service mobile robotics. It is equipped with a camera network, from which we only use 12 of the 21 cameras. They are shown in Figure 7. For this dataset, our 3D laser scanner was mounted on Helena, a Pioneer 3AT mobile robot, acquiring a total of 400 scans; however, only 30 of them were necessary to cover the area of the selected cameras. The complete dataset is available online [36].

Figure 7. The Barcelona Robot Lab. (**a**) Aerial view of the camera distribution; and (**b**) the point cloud.



(**a**) (**b**)

Figure 8. The Facultat de Matemàtiques i Estadística (FME) scenario. (**a**) The point cloud registered onto an aerial view of the scene; and (**b**) the segmented point cloud.



(**a**) (**b**)

The second dataset was gathered in the inner courtyard of the Facultat de Matemàtiques i Estadística (FME), located at the Campus Sud of the UPC. For this dataset, the range sensor was mounted atop

our robot Teo, a rough outdoor terrain Segway RPM400 mobile robot. In this case, only 39 scans were collected. Figure 8 shows the point cloud registered onto an aerial view of the scene and the segmented planes. A mobile phone camera was used to acquire geo-tagged images from different position in this scenario.

In both cases, the point clouds generated from the aggregation of multiple scans are preprocessed to remove outliers, to smooth out the planar regions and to provide a uniform point distribution through subsampling. The details of the filtering scheme used follow [3]. The parameters used to segment the range data in both scenarios were $n = 25$ neighbors to fit planar patches, a distance threshold of $t_d = 0.5$ and a curvature threshold $t_c = 0.8$. Furthermore, we only consider lines intersecting orthogonal planes with a deviation of $\pm 3^\circ$ from orthogonality. For those cases when there were less than six independent lines detected for the calibration of a camera, each detected line was broken into three segments, and all of these were used for the optimization step. This allowed one to have a sufficiently large number of lines to find a least squares solution to Equations (17) and (18). In most of those cases, however, the DLT-lines algorithm did not contribute to the improvement of the solution, and the first optimization sufficed to find acceptable results.

For the BRL dataset, the initial elevation angle was set to 17°. We initialized on this value, because most of the cameras are located about 6 m above the ground, and objects in the images for which lines can be detected reliably are closer than 20 m. The horizontal field of view is initialized to 40°, which corresponds to an 8-mm lens in a 0.25-in CCD.

Table 1. Estimated internal camera parameters of the BRL scenario.

Camera	Focal Length (pixels)	Principal Point (pixels)	Mean Reprojection Error (pixels)	SD (pixels)
A5-1	926.0, 926.0	499.2, 348.6	1.4	4.5
A6-1	857.8, 857.8	255.4, 338.6	1.6	5.3
A6-5	920.5, 920.5	314.0, 240.9	1.3	4.7
A6-6	842.3, 842.3	296.6, 145.4	0.7	4.2
A6-9	747.3, 747.3	366.6, 226.6	8.8	12.4
B5-3	801.2, 801.2	364.1, 202.1	5.3	6.1
B6-1	597.9, 597.9	388.8, 219.8	0.9	1.6
B6-2	817.9, 817.9	295.1, 262.1	2.6	4.4
B6-3	804.2, 804.2	374.1, 246.1	4.7	7.5
B6-4	824.1, 824.1	336.6, 237.2	3.6	6.5
B6-5	840.0, 840.0	317.9, 229.6	2.2	4.3
B6-6	862.5, 862.5	320.1, 247.1	4.6	8.5

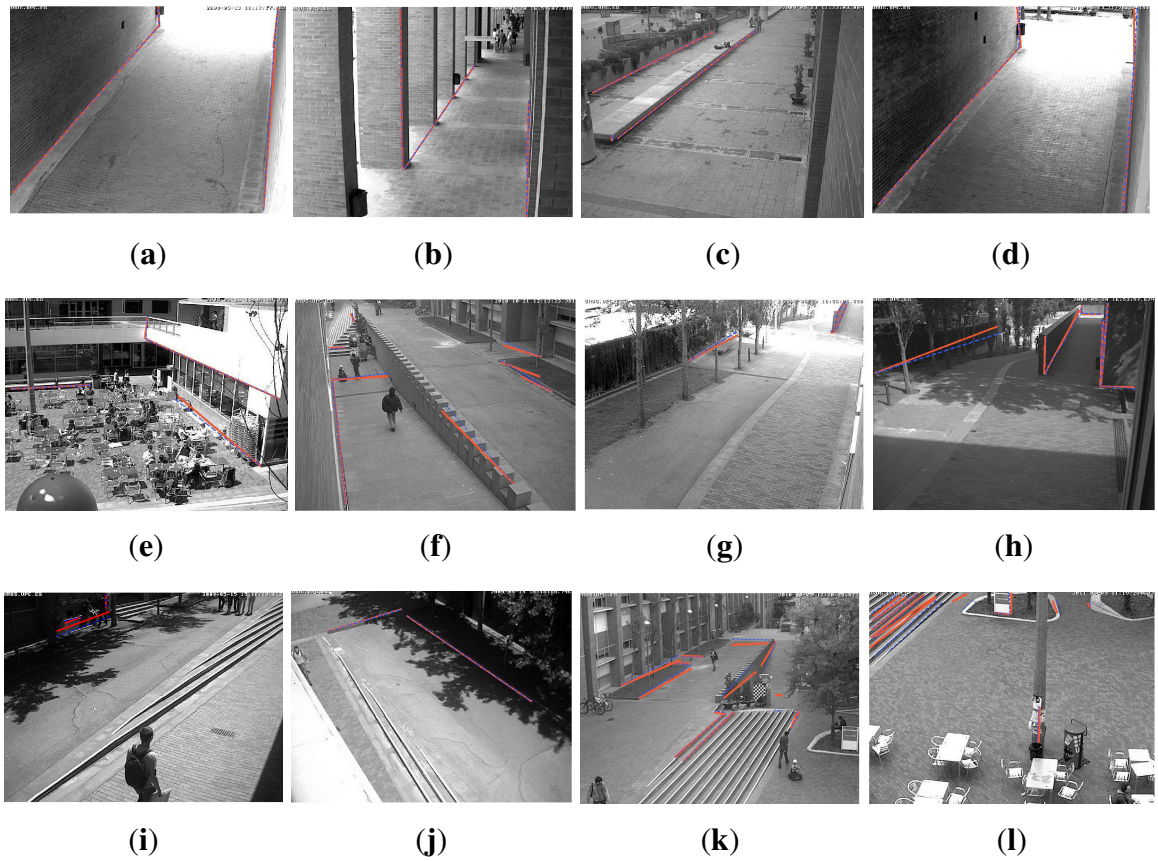
The final calibration of the internal camera parameters for the BRL set are given in Table 1, along with the mean reprojection error and standard deviation. Note that a comparison of these estimates to those obtained with a checkerboard is unrealistic due to the actual positioning of the cameras. An unfeasibly large checkerboard would be needed and moved along the whole camera workspace to achieve significant results. Table 2 gives the obtained camera locations for this experiment, together with the camera ground truth locations manually extracted from the 3D point cloud. These values should only be used as a reference, since small variations of focal length might have incidence in the final positioning of the camera along the principal axis, without detriment to the camera reprojection for a limited range of

depth values. The elapsed computation time for the calibration of each camera is also reported in the table. Figure 9 shows the reprojection of each matched line onto the corresponding camera images after the optimization is computed.

Table 2. Estimated external camera parameters of the BRL scenario.

Camera	Position (m)	Orientation (radians)	Ground Truth Position (m)	Elapsed Time (s)
A5-1	−37.93, −25.37, 3.88	−0.63, 1.85, 2.03	−37.91, −28.2, 3.75	95.2
A6-1	42.84, −25.47, 4.06	1.57, −0.73, −1.37	42.66, −24.95, 3.99	86.5
A6-5	12.76, −28.10, 2.58	1.61, −0.90, −1.18	12.39, −28.95, 3.10	45.2
A6-6	9.83, −22.91, 3.07	0.70, −2.21, −2.10	9.44, −23.5, 2.95	79.6
A6-9	19.32, −4.02, 3.49	0.10, 1.85, 1.94	19.03, −6.65, 3.11	56.8
B5-3	−39.93, −1.16, 2.23	1.20, 1.25, 0.88	−38.94, −2.63, 1.95	69.5
B6-1	52.22, 19.14, 3.53	1.43, 0.60, −0.09	51.98, 18.55, 3.11	66
B6-2	51.66, 7.50, 3.66	1.52, 0.45, −0.09	51.55, 7.20, 3.45	107.3
B6-3	50.48, 2.53, 3.10	1.50, −0.60, −1.33	50.18, 4.36, 2.90	98.6
B6-4	34.45, 1.21, 3.13	1.15, 1.17, 0.42	32.85, 1.63, 2.90	76.8
B6-5	31.80, 6.34, 2.93	1.50, −0.94, −1.31	31.88, 7.48, 5.10	123.9
B6-6	31.81, 14.74, 5.82	1.50, −0.82, −1.32	31.64, 15.59, 5.10	103.2

Figure 9. Results of the final calibration of the camera network for the BRL scenario. Optimization approximates the projected laser lines (**blue**) to the image lines (**red**). (**a**) A5-1; (**b**) A6-1; (**c**) A6-5; (**d**) A6-6; (**e**) A6-9; (**f**) B5-3; (**g**) B6-1; (**h**) B6-2; (**i**) B6-3; (**j**) B6-4; (**k**) B6-5; (**l**) B6-6.



(**a**) (**b**) (**c**) (**d**)

(**e**) (**f**) (**g**) (**h**)

(**i**) (**j**) (**k**) (**l**)

Figure 10. Calibrated camera locations for the Barcelona Robot Lab (BRL) dataset.

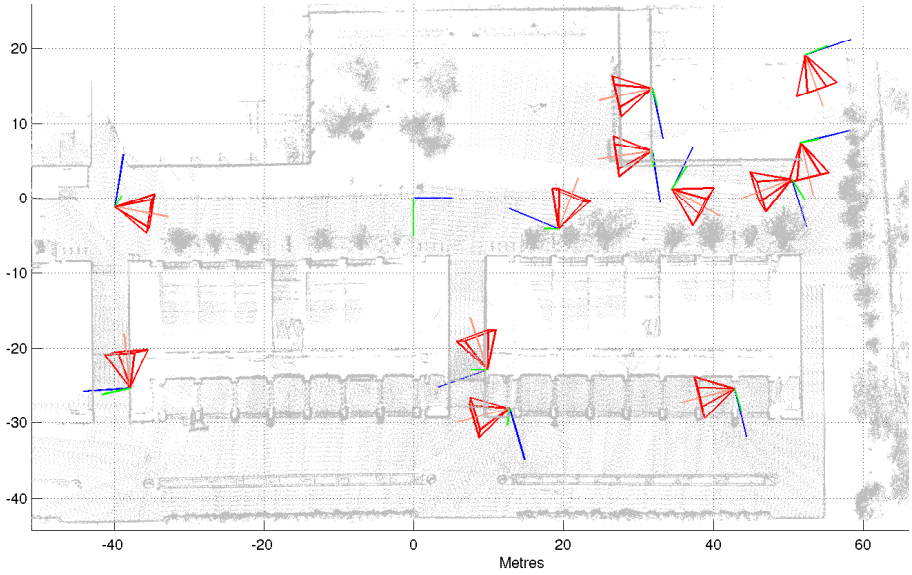
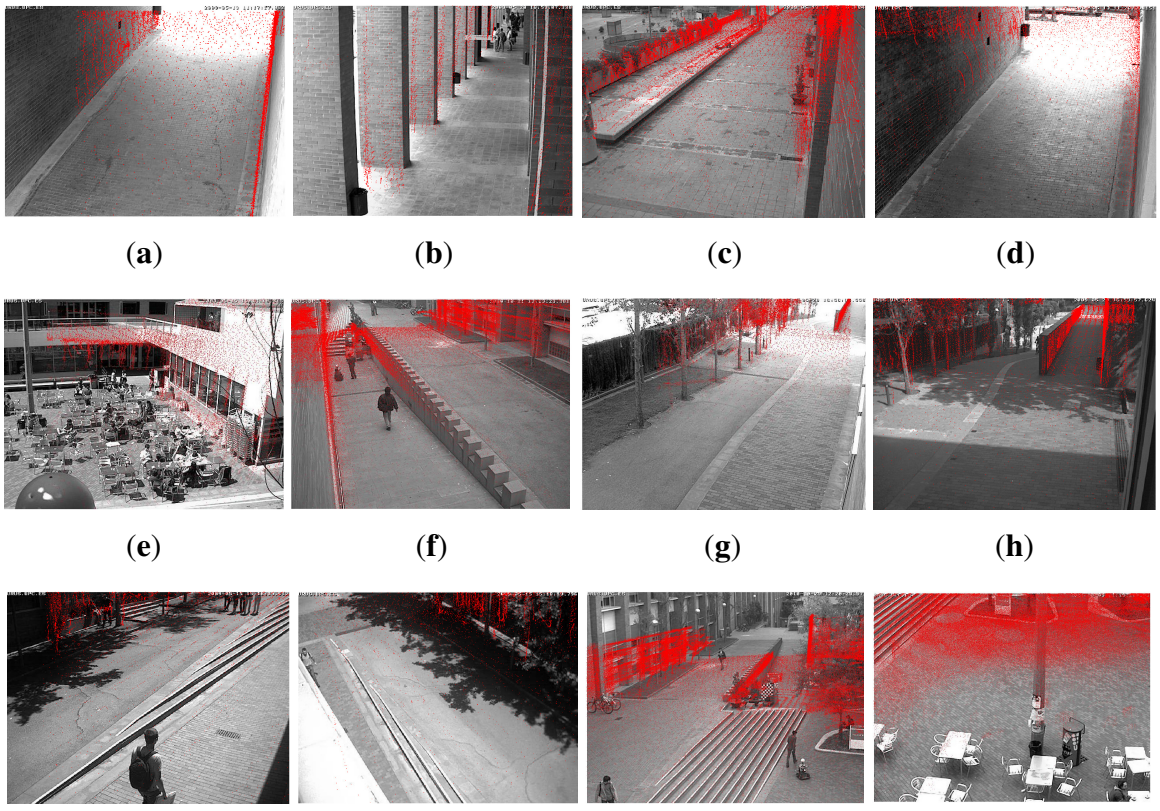


Figure 11. Visualization of range data on each of the camera views of the BRL scenario. (**a**) A5-1; (**b**) A6-1; (**c**) A6-5; (**d**) A6-6; (**e**) A6-9; (**f**) B5-3; (**g**) B6-1; (**h**) B6-2; (**i**) B6-3; (**j**) B6-4; (**k**) B6-5; (**l**) B6-6.



The estimated camera poses are plotted in Figure 10. Camera viewpoints are represented with triangular pyramids that suggest the viewing direction. To empirically judge the quality of the calibration

results, we can also project all points from the map that fall in each camera viewing frustum onto the image plane. This is shown in Figure 11 for the BRL.

In the case of the FME scenario, all images were taken with a mobile phone. GPS readings on the phone were used as initial position estimates. The local world model was considered planar, so that a homography can be used to translate from GPS coordinates to the metric representation used in the point cloud.

Table 3 shows the results of the calibration of internal camera parameters. Since all images were computed using the same camera, the mean values obtained for the internal parameters can be used as a reference. These mean values are shown in the first line in the table.

Table 3. Estimated internal camera parameters for the FME scenario.

Camera	Focal Length (pixels)	Principal Point (pixels)	Mean Reprojection Error (pixels)	SD (pixels)
Cam-mean	2989.6, 2998.9	499.2, 348.6	9.8	15.3
Cam-1	2985.2, 2986.7	493.2, 343.2	15.3	26.8
Cam-2	2985.5, 2980.2	490.2, 346.1	1.2	3.6
Cam-3	2985.7, 2983.2	560.2, 350.8	2.4	6.0
Cam-4	2984.3, 2987.4	510.2, 340.3	3.6	9.0
Cam-5	2989.4, 2987.2	493.2, 341.6	6.5	8.6

Table 4 reports the different camera positions estimated with our algorithm and contrasted to the GPS readings on the phone. GPS coordinates are transformed to metric coordinates with the WGS84 standard and affine transformed with the DLT algorithm to align them with the FME building. Height values are not reported, as their readings from the phone GPS unit are unreliable.

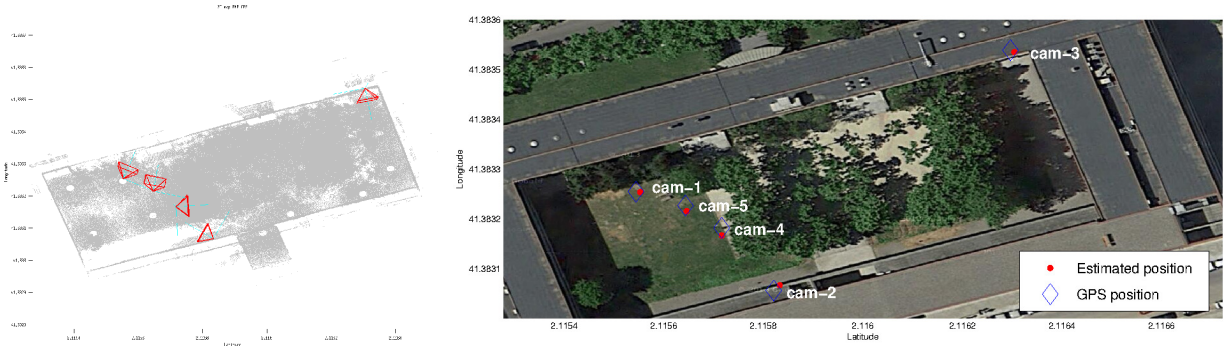
Table 4. Estimated external camera parameters for the FME scenario.

Camera	Position (m)	Orientation (radians)	Ground Truth Position (m)	Elapsed Time (s)
Cam-1	−21.29, 3.39	−1.21, −0.90, −1.76	−21.73, 4.04	80.9
Cam-2	0.04, −13.42	0.45, −2.23, 0.37	−0.02, −12.75	106.1
Cam-3	43.83, 15.27	2.27, 0.18, 2.13	42.70, 15.80	182.1
Cam-4	−8.61, −4.81	1.40, −1.27, 1.25	−9.38, −5.73	95.6
Cam-5	−13.97, −0.44	−0.81, −0.80, −1.77	−14.14, −0.56	112.7

This comparison is shown schematically in Figure 12. Figure 12a shows the estimated camera viewpoints, whereas Figure 12b shows a comparison between GPS readings (blue squares) and the camera poses computed with our method (red dots).

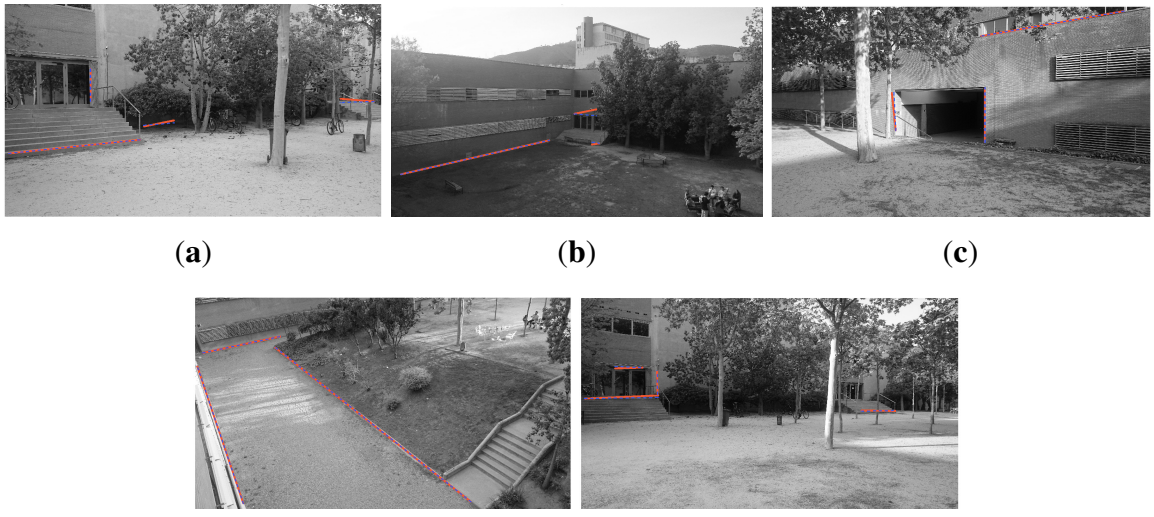
Figure 13 shows results of the optimization results for the FME dataset. In blue projected 3D lines and in red selected image lines. The heavy presence of unstructured data made it difficult to find a large number of support lines for calibration in this scenario, suggesting the need for calibration also with point/appearance features, together with lines. We leave this hybrid scheme as an open alternative for further development of the method.

Figure 12. Camera localization for the FME scenario. (**a**) Camera viewpoint estimates; and (**b**) a comparison between GPS measures (blue squares) and our method (red points).



(**a**) (**b**)

Figure 13. The results of the final calibration camera network. Optimization approximates the projected laser lines (blue) to the image lines (red) using the FME dataset. (**a**) Cam-1; (**b**) Cam-2; (**c**) Cam-3; (**d**) Cam-4; (**e**) Cam-5.



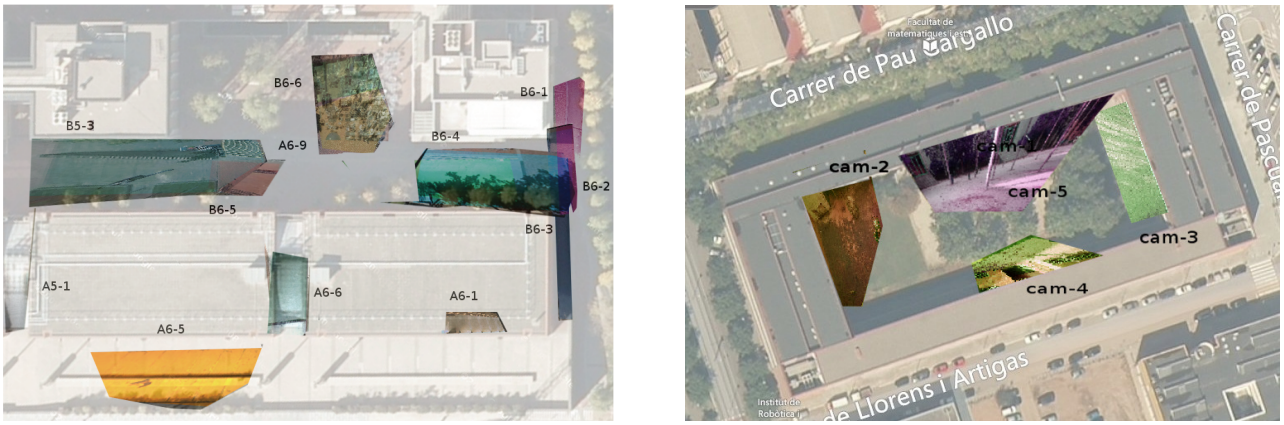
(**a**) (**b**) (**c**)

(**d**) (**e**)

Computing Homographies

To measure events occurring on the scene, such as path lengths or areas of crowdedness, it would be necessary to obtain direct mappings from images to planar regions in the floor. The idea is to have a practical way to transfer 2D images to the world coordinates of the targets detected. To this end, we compute the homographies of user-selected planes with the end of a graphical user interface. The user selects polygonal regions in the images, and the 3D points that project inside these polygons are used to approximate the 3D planes. The algorithm to compute the homographies is the standard direct linear transform [10]. Figure 14 shows the result of this computation for the two scenarios. Camera images, each one of a different color mask, are projected onto their corresponding planar regions in the map.

Figure 14. Computed homographies for the two scenarios. (**a**) BRL scenario; (**b**) FME scenario.



(**a**) (**b**)

5. Conclusions

In this paper, we have proposed a methodology to calibrate outdoor distributed camera networks having small or inexistent overlapping fields of view between the cameras. The methodology is based on the matching of line image features with 3D lines computed from dense 3D point clouds of the scene.

In the first stage, the user obtains the nominal calibration by using default intrinsic parameters for the cameras and indicating their positions and orientations on an aerial view aligned with the range map. Next, the calibration of each camera is improved by an automatic optimization procedure detecting lines in the 3D map and matching them with image lines. The lines are detected in the point cloud by automatically segmenting out planar regions and finding such plane intersections. The optimization procedure then minimizes the distance between points in the lines found in the map and their corresponding points in the image lines. The method has been used to calibrate the Barcelona Robot Lab, a 10,000 m^2 area for mobile robot experimentation, and for camera localization at the FME patio, both located at the UPC campus in Barcelona.

Future work will include an analysis of uncertainty for the external calibration using the DLT-lines algorithm and the combination of feature points together with lines for scenes with a limited structure.

Acknowledgments

Agustín Ortega and Ernesto H. Teniente acknowledge support from the Mexican Council of Science and Technology. This work received support from the Agència de Gestiò d'Ajuts Universitaris i de Recerca of The Generalitat of Catalonia (2012 CTP00013), the Spanish Ministry of Economy and Competitiveness, project PAU+: Perception and Action in Robotics Problems with Large State Spaces (DPI-2011-27510), the EU project, Cargo-ANTs: Cargo Handling by Automated Next Generation Transportation Systems for Ports and Terminals (FP7-SST-2013-RTD-1 605598), the Portuguese Science Foundation FCT project DCCAL: Discrete Cameras Calibration using Properties of Natural Scenes (PTDC/EEACRO/105413/2008), and the Portuguese Quadro de Referéncia Estratégico Nacional project HDA: High Definition Analytics (QREN I&D em Co-Promoção 13750).

Author Contributions

Agustín Ortega developed the nominal and refinement calibrations routines and drafted an initial version of the paper. Ernesto Teniente contributed with the 3D point cloud of the BRL site, and also helped with data acquisition for the FME experiment. Manuel Silva and Ricardo Ferreira developed the DLT-lines algorithm. Alexandre Bernardino, José Gaspar and Juan Andrade-Cetto supervised the work and finalized the manuscript. All of the authors read and approved the final manuscript.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Thrun, S.; Burgard, W.; Fox, D. *Probabilistic Robotics*; MIT Press: Cambridge, UK, 2005.
2. Merchán, P.; Adán, A.; Salamanca, S.; Domínguez, V.; Chacón, R. Geometric and colour data fusion for outdoor 3D models. *Sensors* **2012**, *12*, 6893–6919.
3. Valencia, R.; Teniente, E.; Trulls, E.; Andrade-Cetto, J. 3D Mapping for urban service robots. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009), Saint Louis, MO, USA, 10–15 October 2009; pp. 3076–3081.
4. Tsai, R. A versatile camera calibration technique for high accuracy 3D machine vision metrology using off-the-shelf TV cameras. *IEEE J. Robot. Autom.* **1987**, *3*, 323–344.
5. Zhang, Z. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 1330–1334.
6. Samper, D.; Santolaria, J.; Brosed, F.J.; Majarena, A.C.; Aguilar, J.J. Analysis of Tsai calibration method using two- and three-dimensional calibration objects. *Mach. Vis. Appl.* **2013**, *24*, 127–131.
7. De la Escalera, A.; Armingol, J.M. Automatic chessboard detection for intrinsic and extrinsic camera parameter calibration. *Sensors* **2010**, *10*, 2027–2044.
8. Cai, S.; Zhao, Z.; Huang, L.; Liu, Y. Camera calibration with enclosing ellipses by an extended application of generalized eigenvalue decomposition. *Mach. Vis. Appl.* **2013**, *24*, 513–520.
9. Fiala, M.; Shu, C. Self-identifying patterns for plane-based camera calibration. *Mach. Vis. Appl.* **2008**, *19*, 209–216.
10. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*, 2nd ed.; Cambridge University Press: Cambridge, UK, 2004.
11. Okuma, K.; Little, J.; Lowe, D.G. Automatic rectification of long image sequences. In Proceedings of the Asian Conference on Computer Vision (ACCV '04), Jeju Island, Korea, 27–30 January 2004.
12. Svoboda, T.; Martinec, D.; Pajdla, T. A convenient multicamera self-calibration for virtual environments. *Presence Teleoper. Virtual Environ.* **2005**, *14*, 407–422.
13. Yokoya, T.; Hasegawa, T.; Kurazume, R. Calibration of distributed vision network in unified coordinate system by mobile robots. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2008), Pasadena, CA, USA, 19–23 May 2008; pp. 1412–1417.

14. Rahimi, A.; Dunagan, B.; Darrell, T. Simultaneous calibration and tracking with a network of non-overlapping sensors. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004), Washington, DC, USA, 27 June–2 July 2004; pp. 187–194.
15. Junejo, I.; Foroosh, H. GPS coordinates estimation and camera calibration from solar shadows. *Comput. Vis. Image Underst.* **2010**, *114*, 991–1003.
16. Ukita, N. Probabilistic-topological calibration of widely distributed camera networks. *Mach. Vis. Appl.* **2007**, *18*, 249–260.
17. Bimbo, A.D.; Dini, F.; Lisanti, G.; Pernici, F. Exploiting distinctive visual landmark maps in pan-tilt-zoom camera networks. *Comput. Vis. Image Underst.* **2010**, *114*, 611–623.
18. Mavrinac, A.; Chen, X.; Tepe, K. An automatic calibration method for stereo-based 3D distributed smart camera networks. *Comput. Vis. Image Underst.* **2010**, *114*, 952–962.
19. Junejo, I.; Xiaochun, C.; Foroosh, H. Autoconfiguration of a dynamic nonoverlapping camera network. *IEEE Trans. Syst. Man Cybern. B Cybern.* **2007**, *37*, 803–816.
20. Geiger, A.; Moosmann, F.; Car, O.; Schuster, B. Automatic camera and range sensor calibration using a single shot. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2012), Saint Paul, MN, USA, 14–18 May 2012; pp. 3936–3943.
21. Scaramuzza, D.; Harati, A.; Siegwart, R. Extrinsic self calibration of a camera and a 3D laser range finder from natural scenes. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007), San Diego, CA, USA, 29 October–2 November 2007; pp. 4164–4169.
22. Mirzaei, F.; Kottas, D.; Roumeliotis, S. 3D LIDAR-camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization. *Int. J. Robot. Res.* **2012**, *31*, 452–467.
23. Liu, L.; Stamos, I. A systematic approach for 2D-image to 3D-range registration in urban environments. *Comput. Vis. Image Underst.* **2012**, *116*, 25–37.
24. Song, K.T.; Tai, J.C. Dynamic calibration of pan-tilt-zoom cameras for traffic monitoring. *IEEE Trans. Syst. Man Cybern. B Cybern.* **2006**, *36*, 1091–1103.
25. Ortega, A.; Dias, B.; Teniente, E.; Bernardino, A.; Gaspar, J.; Andrade-Cetto, J. Calibrating an outdoor distributed camera network using laser range finder data. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009), Saint Louis, MO, USA, 10–15 October 2009; pp. 303–308.
26. Ortega, A.; Haddad, I.; Andrade-Cetto, J. Graph-based segmentation of range data with applications to 3D urban mapping. In Proceedings of the 4th European Conference on Mobile Robots, Barcelona, Spain, 23–25 September 2009; pp. 193–198.
27. Felzenszwalb, P.F.; Huttenlocher, D.P. Efficient graph-based image segmentation. *Int. J. Comput. Vis.* **2004**, *59*, 167–181.
28. Cormen, T.H.; Leiserson, C.E.; Rivest, R.L. *Introduction to Algorithms*; MIT Press: Cambridge, UK, 1992.
29. Arya, S; Mount, D.M.; Netanyahu, N.S.; Silverman, R.; Wu, A.Y. An optimal algorithm for approximate nearest neighbor searching. *J. ACM* **1998**, *45*, 891–923.

30. Elseberg, J.; Magnenat, S.; Siegwart, R.; Nuechter, A. Comparison of nearest-neighbor-search strategies and implementations for efficient shape registration. *J. Softw. Eng. Robot.* **2012**, *3*, 2–12.
31. Von Gioi, R.; Jakubowicz, J.; Morel, J.M.; Randall, G. LSD: A fast line segment detector with a false detection control. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 722–732.
32. Fischler, M.; Bolles, R. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* **1981**, *24*, 381–385.
33. Silva, M.; Ferreira, R.; Gaspar, J. Camera calibration using a color-depth camera: Points and lines based DLT including radial distortion. In Proceedings of Workshop Color-Depth Camera Fusion in Robotics, Vilamoura, Portugal, 7 October 2012.
34. Lütkepohl, H. *Handbook of Matrices*; Wiley and Sons: Hoboken, NJ, USA, 1996.
35. Fitzgibbon, A. Simultaneous linear estimation of multiple view geometry and lens distortion. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001), Kauai, HI, USA, 8–14 December 2001; pp. 125–132.
36. Teniente, E.H.; Morta, M.; Ortega, A.; Trulls, E.; Andrade-Cetto, J. Barcelona Robot Lab Data Set. Avaiable online: http://www.iri.upc.edu/research/webprojects/pau/datasets/BRL/php/dataset.php (accessed on 28 July 2014).