Multiple Camera Self-calibration and 3D Reconstruction Using Pedestrians

Michael Hödlmoser and Martin Kampel

Computer Vision Lab, Vienna University of Technology
Favoritenstraße 9/183, 1040 Vienna

Abstract. The analysis of human motion is an important task in various surveillance applications. Getting 3D information through a calibrated system might enhance the benefits of such analysis. This paper presents a novel technique to automatically recover both intrinsic and extrinsic parameters for each surveillance camera within a camera network by only using a walking human. The same feature points of a pedestrian are taken to calculate each camera's intrinsic parameters and to determine the relative orientations of multiple cameras within a network as well as the absolute positions within a common coordinate system. Experimental results, showing the accuracy and the practicability, are presented at the end of the paper.

1 Introduction

According to [1], the task of calibrating a camera can mainly be divided into two approaches, namely calibration using a known calibration object and self-calibration. When the dimensions of an object are known, the 3D informations of a scene can be extracted by establishing correspondences between different views showing the same calibration object. In case of self-calibration, the dimensions of the calibration object are not known and geometric features (e.g. vanishing points (VPs)) are taken for camera calibration.

Surveillance scenarios might deal with the analysis of humans. Therefore, pedestrians are predestined to be used as calibration object for self-calibrating a camera. Calibrating a camera using a pedestrian was first introduced by Lv et. al. in [2]. Top and bottom points are determined and three VPs are extracted. A closed-form geometric solution is used to obtain the intrinsic parameters afterwards. The extrinsic parameters are only determined for one camera.

In this paper we present a novel and practical approach to determine both intrinsic and extrinsic parameters for a network of surveillance cameras by analyzing a pedestrian. We develop a 3D scene reconstruction method based on pairwise determination of relative orientations. Different to [3], the calibration method is replaced by a self-calibration approach similar to the one presented in [2]. We use top and bottom points for calculating three VPs, but instead of using a geometric closed form sulution, the intrinsic parameters are then extracted from the image of the absolute conic (IAC), which should lead to faster computational time. In a next step, the extracted top and bottom points are also used as input for gathering the relative orientations between the cameras within a network. To extract the scaling factor for the absolute translation of all cameras, the user needs to predefine the height of the walking person.

The paper is organized as follows: Section 2 gives an overview on related work concerning self-calibration and scene reconstruction using VPs and pedestrians. Section 3 explains our approach of calculating intrinsic and extrinsic parameters for each camera using a walking human. Experiments using synthetic and real world data are carried out in Section 4.

2 Related Work

Camera self-calibration and 3D reconstruction have been studied extensively in the last few years. Beardsley first described the extraction of intrinsic camera parameters from three VPs in [4]. By the determination of three VPs within one image, the principal point and the focal length can be recovered sequentially. In [5], camera calibration using three VPs of an image is described. Their semi-automatic auto-calibration method uses building facades to determine three VPs. The user needs to select a set of parallel image lines in order to search for correct VP initialization. After initialization, the intrinsic parameters are recovered. The relative rotation between a camera pair is estimated using the calculated points on the plane at infinity, the translation is calculated by using further points of interest in a scene. Based on the work of [2], Junejo presents a quite similar calibration approach for pedestrians walking on uneven terrains in [6]. The VPs do not need to be orthogonal to each other and the intrinsic parameters are estimated by obtaining the infinite homography from all VPs. A direct method for self-calibrating a camera by observing a pedestrian is presented in [7]. Based on top and bottom points of a walking human, the pose matrix is estimated column by column by exploiting cross-ratio constraints. Having two VPs and a vanishing line (VL), the third VP can be calculated. The VPs are then directly used to calculate the pose of the camera.

A scene reconstruction method for two cameras only is presented in [8]. The recover of the intrinsic parameters is based on the algorithm presented in [6]. The relative orientation is afterwards calculated using all VPs and the infinite homography. A pose estimation supporting multiple cameras is presented in [3]. A framework for manual camera calibration and 3D scene reconstruction for teleimmersion purposes using corresponding points of interest is proposed. The determination of intrinsic parameters is performed using Bouguet's Camera Calibration Toolbox for Matlab[1]. For reconstructing the scene, two LED markers having a fixed distance and defining a virtual calibration object are used. After the marker is detected on the image plane, the fundamental matrix between two cameras is determined and the relative rotation is calculated.

[1] `http://www.vision.caltech.edu/bouguetj/calib_doc/`, last retrieved on 07.07.2010.

3 Calibration and 3D Reconstruction from Pedestrians

Calibrating a camera network consists of three steps, namely the determination of intrinsic parameters (focal length f, principal point pp) for each camera, the extraction of pairwise relative orientations ($\Delta R, \Delta t$) between two cameras and the determination of the absolute orientations (R, t) of all cameras in one common coordinate system.

This section describes the extraction of intrinsic and extrinsic parameters from a sequence of a walking human. The pedestrian is observed for at least 2 frames over time and top and bottom points are extracted. These points are taken to calculate three VPs. The intrinsic parameters are then determined from the IAC. The same points of the walking pedestrian are also used to recover the cameras' extrinsic parameters. To extract the scaling factor for the absolute translation of all cameras, the user needs to predefine the height of the walking person.

3.1 Estimation of VPs

By using the extracted top and bottom points, two VPs and a VL can be calculated. Since the third VP can be determined, the IAC can be formed using all three VPs. By applying the Cholesky decomposition [9], the intrinsic parameters can be extracted from the IAC. There are only two constraints related to this approach namely (a) the top and the bottom points need to have a fixed distance in 3D coordinates over time and (b) all top points and all bottom points are coplanar respectively. Both constraints are fulfilled by a pedestrian having a constant height walking on an even ground plane.

Determination of Top and Bottom Points: To determine top points ($T = \begin{pmatrix} t_x \\ t_y \end{pmatrix}$) and bottom points ($B = \begin{pmatrix} b_x \\ b_y \end{pmatrix}$) of a walking human, the human silhouette must be extracted in a first step. This step is crucial for all further calculations as an error of 1-2 pixels in top and bottom locations can result in wrong VP estimations and therefore wrong calibration results (see Section 4). A simple background subtraction is done to determine a rough silhouette of the pedestrian. The resulting foreground bounding box is used as input for the grab cut algorithm [10]. This enables the accurate extraction of the pedestrian from the background. The bounding box for the accurate determined pedestrian is calculated and divided in an upper and a lower part. The center of mass is calculated for the whole bounding box, for the lower and for the upper part respectively. By minimizing the least squared distance to all three center points, a line is fitted through the centers of mass, which gives a vertical VL of the walking human. The top and bottom points are then given by the intersection of the vertical VL and the bounding box. Figure 1(a) shows a random input image from a video sequence used for the experiments in Section 4, Figure 1(b) shows the extracted walking person within a rectangular bounding box and the top and bottom points. Figure 1(c) shows all pedestrian locations and the corresponding top and bottom points, represented by dots, within one frame.

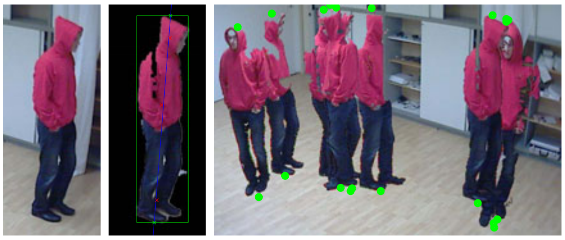

Fig. 1. (a) Example input image, (b) extraction of top and bottom points and (c) all extracted top and bottom locations

Approximation of the Vertical VP: According to [2], vertical VPs vz_{ij} can be calculated by using the top and bottom points of a walking human. Therefore, the human must be shown in a frame sequence consisting of at least two frames, i and j, where $i \neq j$. The vertical VPs can be expressed by $vz_{ij} = \overline{T_iB_i} \times \overline{T_jB_j}$.

In a next step, a general vertical VP, $vz = \begin{pmatrix} vz_x \\ vz_y \end{pmatrix}$, needs to be calculated. When using synthetic data, all vertical VPs must converge in one general VP. Since top and bottom location initialization may be noisy and therefore not all vertical VPs must converge to one point, an approximation needs to be performed. Using cross ratio in 2D, the relationship between the vertical VP, top and bottom points of one frame showing a pedestrian is given by [7]

$$vz_x(t_y - b_y) + vz_y(b_x - t_x) = (b_xt_y - t_xb_y). \quad (1)$$

When two or more instances of the pedestrian are given, an equation system $A \cdot vz = r$, where r is the result vector, can be formed to solve the equation. A least squares solution can be gathered by solving

$$vz = (A^TA)^{-1}A^Tr. \quad (2)$$

To make the solution more robust, all $\begin{pmatrix} n \\ 2 \end{pmatrix}$ possible combinations for n frames are used for further calculations.

Determination of the Horizontal VL and the Two Missing VPs: The horizontal VPs can be calculated by $vx_{ij} = \overline{T_iT_j} \times \overline{B_iB_j}$. When having synthetic data, all VPs must lie on one line called the VL. Since top and bottom location initialization may be noisy and therefore not all horizontal VPs must lie on one line, the line is fitted by minimizing the least squared distance to all horizontal VPs. Figure 2(a) shows the general vertical VP and a selection of horizontal VPs of four images of a walking human. As three VPs orthogonal to each other are needed for the intrinsic calibration of a camera, we can use vz and the VL for the determination of two more VPs, vx and vy. If the VL has a gradient equal to zero, it is called the horizontal line. The second VP vx can easily be

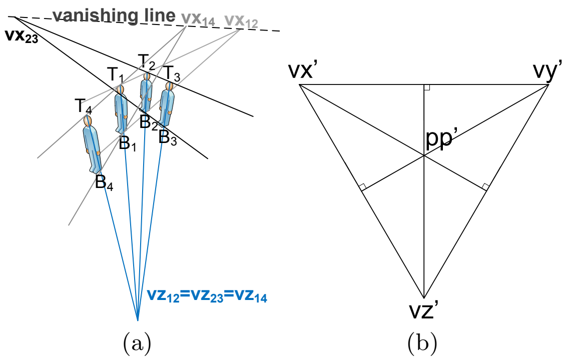

Fig. 2. Determination of (a) VL and a general vertical VP by using all horizontal and vertical VPs and (b) the calculation of two missing VPs by the triangle spanned by vx',vy' and vz'

determined by taking an arbitrary point on the VL. Before calculating vy, the VL needs to be aligned with the horizontal line. This is established by rotating the VL by the negative gradient of the VL. By aligning the VL, the whole scene, including all points needed for further calculations, must also be rotated ($vx = vx', vy = vy', vz = vz'$). The principal point pp can safely be assumed to be the image center and pp is rotated to obtain pp'. As pp' is the orthocenter of the triangle spanned by the three VPs, vy' can be calculated geometrically as shown in Figure 2(b) and described in [2]. To obtain vx,vy,vz, the scene is rotated back around the image origin by the gradient of the VL.

3.2 Retrieving the Camera Calibration Matrix

After the calculation of all three VPs, the IAC, ω, can be determined. Under the assumption of squared pixels and zero skew, ω has the form of

$$\omega = \begin{bmatrix} \omega_1 & 0 & \omega_2 \\ 0 & \omega_1 & \omega_3 \\ \omega_2 & \omega_3 & \omega_4 \end{bmatrix} \tag{3}$$

According to [1], ω can be determined from three orthogonal VPs. The intrinsic parameters are directly related to ω by $(KK^T)^{-1} = \omega$. By applying the Cholesky decomposition [9], the intrinsic parameters can be extracted from the IAC.

3.3 Calibration of Camera Pairs

Top and bottom points of the walking pedestrian are also used to recover the scene structure, which in other words mean the determination of the cameras' extrinsic parameters. Before calculating an absolute orientation within a common world coordinate system, a relative orientation between each camera pair needs to be determined. The orientation between two arbitrary cameras consists of a relative rotation ΔR and a relative translation Δt. Two cameras are related by the so called epipolar geometry, which can be described by a fundamental matrix [1]. By gathering eight point correspondences between the cameras, the fundamental matrix can be calculated by the normalized 8-point algorithm, presented in [1].

When the intrinsic parameters are already known, the normalized corresponding points can be determined. Performing the 8-point algorithm by using normalized corresponding points leads to a resulting matrix called the essential matrix. The essential matrix is defined by

$$E = [\Delta t]_\times \Delta R = \Delta R[\Delta R^T \Delta t]_\times \tag{4}$$

The relative rotation and translation can directly be extracted from the essential matrix. As there are four solutions for ΔR and Δt, the solution where a positive depth is gathered for any projected point must be picked, as described in [1].

3.4 3D Reconstruction

After pairwise calibration of the camera network, the absolute orientation for each camera must be determined. Therefore, we determine one camera to be the common world coordinate origin. Having two cameras a, b and their relative orientation, the absolute rotation R_b and absolute translation t_b of camera b is calculated by

$$R_b = \Delta R R_a,\ t_b = \Delta t + \Delta R t_a \tag{5}$$

Since the essential matrix is only determined up to a scale factor λ, this factor must be calculated to gain a scaled translation vector. Therefore, 3D points are calculated in the normalized world coordinate system using triangulation [1]. The scale factor can be calculated by

$$\lambda = \frac{1}{N}\sum_{i=1}^{N} \frac{H}{distance(T3D_i, B3D_i)}, \tag{6}$$

where $T3D$ and $B3D$ are the top and bottom points in the normalized 3D space, N is the number of all point pairs respectively, and H is the real height of the pedestrian, which must be provided by the user. The rotation and translation parameters are optimized by running the sparse bundle adjustment algorithm over all cameras within the network, as proposed in [11]. This algorithm simultaneously minimizes the reprojection error (RE) and the error of all absolute orientations, when intrinsic parameters are fixed.

4 Experiments

To show the accuracy of our proposed method, we do an evaluation using synthetic and real data. As we want to show that our algorithm is working precisely, we first evaluate it by using a synthetically generated scene. Additionally, the impact of the incorrect extraction of top and bottom points on the calibration results is demonstrated by introducing noise to the input points. As we also want to show the practical use of the method in real world scenarios, we secondly evaluate it by using real world data. REs and comparisons of the results of our method to calculated ground truth data (synthetic scene) or results of conventional calibration methods (real world scene) are introduced as evaluation metrics.

Synthetic Data: For evaluation purposes, a 3D scene is generated using OpenGL and ten top and bottom points are randomly positioned within the scene. The distance between top and bottom locations is 56 millimeter. Ten cameras are located randomly within the observed scene of 360 degrees where all top and bottom points can be seen from each camera. All cameras have the same focal length and principal point. The positions of the cameras and the calculated camera positions are shown in Table 1. Since OpenGL is using a different coordinate system than Matlab does and the coordinate origin is a different one than our implementation uses, we use the relative distance instead of a translation vector for comparisons. ΔR and Δt are the relative parameters for each camera to reference camera one. As can be seen, both the intrinsic and the extrinsic parameters can be extracted precisely. Reference values (RV) from the OpenGL scene are compared to calculated values (CV). All cameras offer a difference between calculated and reference focal length of $< 0.72\%$. The maximum difference of the relative translation is 0.03%, measured at camera seven. Since the top and bottom locations are calculated having no noise, it can be seen that the mean and the standard deviation of the RE (represented by MRE and SRE) is nearly zero. In a next step, we calculate the height of the synthetically generated distance between top and bottom locations. The mean height

Table 1. Comparison of calculated intrinsic / extrinsic parameters and their reference parameters of seven synthetic cameras. Additionally, REs are shown.

Cam #		f (pixel)	u_0 (pixel)	v_0 (pixel)	ΔR (degree)	Δt (mm)	MRE (10^{-3}pixel)	SRE (10^{-3}pixel)
01	CV	659.4054	320.0011	240.0001	(00.0000, 00.0000, 00.0000)	000.0000	0.0040	0.1928
	RV	660.0000	320.0000	240.0000	(00.0000, 00.0000, 00.0000)	000.0000	–	–
02	CV	659.4569	319.9791	239.9978	(60.0023, 00.0002, 00.0000)	371.3528	0.0040	0.2491
	RV	660.0000	320.0000	240.0000	(60.0000, 00.0000, 00.0000)	371.3080	–	–
03	CV	659.4182	319.9973	239.9998	(110.0048, 00.0010,-00.0005)	666.6406	0.0080	0.4019
	RV	660.0000	320.0000	240.0000	(110.0000, 00.0000, 00.0000)	666.5556	–	–
04	CV	659.7868	319.9553	239.9964	(135.0026,-00.0002, 00.0001)	782.5567	0.0060	0.2440
	RV	660.0000	320.0000	240.0000	(135.0000, 00.0000, 00.0000)	782.5418	–	–
05	CV	659.3608	319.9960	239.9997	(179.9988,-00.0002,-00.0001)	854.6013	0.0040	0.2986
	RV	660.0000	320.0000	240.0000	(180.0000, 00.0000, 00.0000)	854.5334	–	–
06	CV	659.4335	320.0258	240.0009	(224.9952, 00.0005, 00.0003)	786.9592	0.0050	0.2789
	RV	660.0000	320.0000	240.0000	(225.0000, 00.0000, 00.0000)	786.9350	–	–
07	CV	659.3384	319.9618	239.9895	(275.0126,-00.0026, 00.0002)	603.9850	0.0230	0.9620
	RV	660.0000	320.0000	240.0000	(275.0000, 00.0000, 00.0000)	603.7959	–	–
08	CV	659.3599	320.0348	240.0053	(290.0016,-00.0004,-00.0000)	492.7822	0.0040	0.2956
	RV	660.0000	320.0000	240.0000	(290.0000, 00.0000, 00.0000)	492.7633	–	–
09	CV	659.2860	319.9794	239.9975	(304.9972, 00.0003,-00.0004)	369.4673	0.0070	0.3678
	RV	660.0000	320.0000	240.0000	(305.0000, 00.0000, 00.0000)	369.4808	–	–
10	CV	659.3601	319.9663	239.9967	(339.9978, 00.0001,180.0000)	134.5118	0.0040	0.2187
	RV	660.0000	320.0000	240.0000	(340.0000, 00.0000,180.0000)	134.5223	–	–

of all ten top/bottom pairs is 56.00 millimeters, having a standard deviation of 2.0248e-004.

To proof the robustness of our algorithm, we run the method again by using a varying standard deviation on the top and bottom points. Table 2 shows the results obtained when using a raising standard deviation. As an example, we pick out the results of camera ten for calculations including noise. As can be seen, intrinsic parameters and the relative rotations remain stable (maximum $\Delta f = 8.71\%$, maximum $\Delta u_0 = 3.34\%$, maximum $\Delta v_0 = 0.60\%$, maximum $\Delta R = (0.07\%, 0.10\%, 3.34\%)$) whereas the distance between camera one and camera ten is getting smaller when noise is introduced (maximum $\Delta t = 26.17\%$).

Table 2. Intrinsics and relative orientation between reference camera one and camera ten when introducing noise

σ	$f(pixel)$	$u_0(pixel)$	$v_0(pixel)$	$\Delta R(degree)$	$\Delta t(mm)$
0.5	638.3583	323.4995	240.2694	(340.2192 -00.0780 179.7621)	123.3432
1.0	621.3275	322.8866	239.8885	(339.7636 00.3883 173.9788)	102.6989
1.5	602.4946	309.2980	238.5548	(340.5865 -00.5360 175.3318)	99.3099

The computation time of the whole calibration procedure using 2,3,4,5 and 10 cameras is 1.5, 2.7, 4.3, 5.6 and 13.8 seconds which means that by each camera used in the network, the computation time increases by approximately 1.4 seconds, which can be negligible in a calibration procedure.

Real Data: In this section we are using real data for evaluating our calibration method. This section should show the practical use of the approach. Basically, we divide our experiments in two different setups.

First three cameras are capturing a moving model figure (Figure 3(a)), having a height of 56 mm. Three Unibrain Fire-i webcams are used for this example as we want to show the remaining accuracy of the method using low cost cameras. The usage of three cameras also enables a observed scene of 360 degrees. This setup enables a more robust estimation of top and bottom locations since the figure is not really walking but only moving on a plane and therefore the two feet positions are always vertical.

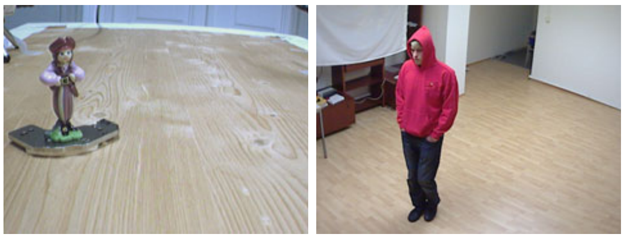
(a) (b)

Fig. 3. Input data for the evaluation setup using real data: (a) model figure and (b) walking human

Table 3. Intrinsics and extrinsics of three real cameras compared to the ground truth, given by the camera's data sheets. Additionally, MRE and SRE are shown.

	$f(pixel)$	$u_0(pixel)$	$v_0(pixel)$	$\Delta R(degree)$	$\Delta t(mm)$	MRE(pixel)	SRE(pixel)
Model Figure							
GT	746.6666	320.0000	240.0000	–	–	–	–
cam01	766.0100	331.9616	239.3585	(00.0000 00.0000 00.0000)	00.0000	3.9084	2.4484
cam02	685.4828	336.4431	239.8932	(246.3855 17.7824 21.4211)	362.7431	2.9294	1.7585
cam03	709.1168	286.1695	243.2339	(135.3388 -12.9850 33.9369)	412.5273	3.8092	3.0101
Pedestrian							
GT	782.2222	320.0000	240.0000	–	–	–	–
cam01	781.8839	374.9783	243.3623	(00.0000 00.0000 00.0000)	0000.0000	3.8317	3.0453
cam02	783.5094	303.3124	241.6647	(266.5525 -36.3978 -18.9564)	4221.4870	3.8105	3.2615

Second two cameras (AXIS M1031-W), mounted on the wall of a living room, are capturing a walking human (Figure 3(b)) having the height of 195cm.

As we want to show that the proposed method is also working with a low number of input images, we use ten different moving positions of the model figure and ten walking positions of the pedestrian, respectively. Table 3 shows the calculated intrinsic parameters using the proposed method. The ground truth (denoted as GT in the table) for the intrinsics is given by the data sheet of the used cameras. Since the determination of top and bottom locations is noisy due to the nature of the person detector, it can be seen that the MRE is relatively high in both cases (e.g. 3.9084 and 3.8322 pixels in camera 1 for experiment 1 and 2, respectively). Due to manual measurement errors, it is impossible to recover the absolute translation and rotation precisely. To proof the accuracy of the proposed 3D reconstruction method on real data, we calculate the height of the model figure and the pedestrian. We are using all ten recovered top and bottom positions and calculate a mean height of all ten points, which is 56.00 millimeters (standard deviation = 01.2137 millimeters) for the model figure and 195.0005 centimeters (standard deviation = 4.1674 centimeters) for the pedestrian.

5 Conclusion

In surveillance scenarios, pedestrians are predestined to be used as input for self-calibrating a camera. We proposed a novel self-calibration and 3D reconstruction method to recover both intrinsic and extrinsic parameters of the devices within a camera network by only observing a pedestrian. The user needs to predefine the height of the walking person to determine the scaling of all cameras' absolute translations. As shown in our experiments, the accuracy of the method does only depend on the noise of top and bottom locations. As we want to use this for surveillance applications where pedestrians need to be classified and heights must be measured, the method is working accurate enough, where measured 195 centimeters offer calibration results having a standard deviation of 4.1674 centimeters. The computation time increases by each camera added to the network by approximately 1.4 seconds.

Acknowledgement

This work was partly supported by the Austrian Research Promotion Agency (FFG) under KIRAS project miniSPOT.net.

References

1. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2003)
2. Lv, F., Zhao, T., Nevatia, R.: Camera Calibration from Video of a Walking Human. IEEE Trans. on PAMI 28, 1513–1518 (2006)
3. Kurillo, G., Li, Z., Bajcsy, R.: Framework for hierarchical calibration of multi-camera systems for teleimmersion. In: Proc. of the IMMERSCON 2009, pp. 1–6 (2009)
4. Beardsley, P., Murray, D.: Camera calibration using vanishing points. In: Proc. of the BMVC, Leeds, UK, pp. 417–425 (1992)
5. Cipolla, R., Drummond, T., Robertson, D.P.: Camera Calibration from Vanishing Points in Image of Architectural Scenes. In: Proc. of the BMVC, Nottingham, UK, vol. 2, pp. 382–391 (1999)
6. Junejo, I.: Using Pedestrians Walking on Uneven Terrains for Camera Calibration. In: MVA (2009)
7. Kusakunniran, W., Li, H., Zhang, J.: A Direct Method to Self-Calibrate a Surveillance Camera by Observing a Walking Pedestrian. In: Proc. of DICTA, Melbourne, pp. 250–255 (2009)
8. Chen, T., Del Bimbo, A., Pernici, F., Serra, G.: Accurate self-calibration of two cameras by observations of a moving person on a ground plane. In: Proc. of the AVSS, London, UK, pp. 129–134. IEEE Computer Society, Los Alamitos (2007)
9. Press, W., Flannery, B., Teukolsky, S., Vetterling, W.: Numerical Recipes in C: The Art of Scientific Computing. Cambridge University Press, Cambridge (1992)
10. Rother, C., Kolmogorov, V., Blake, A.: "grabcut": interactive foreground extraction using iterated graph cuts. ACM Transactions on Graphics 23, 309–314 (2004)
11. Lourakis, M.I.: levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++ (2004), `http://www.ics.forth.gr/~lourakis/levmar/`