Auto-calibration for Image Mosaicing and Stereo Vision

Alexey Spizhevoy and Victor Eruhimov

Itseez Ltd. and Lobachevsky State University of Nizhni Novogord, Russia
{alexey.spizhevoy,victor.eruhimov}@itseez.com

Abstract. The paper investigates the auto-calibration problem for mobile device cameras. We extend existing algorithms to get a robust method that computes internal camera parameters given a series of distant objects images. The algorithm is tested on real images generated by several different cameras. We estimate the impact of errors in camera calibration parameters in image mosaicing and 3D reconstruction problems.

Keywords: auto-calibration, camera parameters, errors effect, real datasets, image mosaicing, stereo.

1 Introduction

The goal of calibration is to determine internal camera parameters within the given projection model. The problem arises in a number of emerging computer vision applications such as augmented reality, 3D reconstruction, and image mosaicing (or stitching). As academy and industry becomes gradually more interested in using mobile devices for computer vision, the importance of phone/tablet cameras calibration is clear.

Nowadays the problem of camera calibration is usually solved by using special calibration patterns (see [3], [4]). While pattern-based methods are quite accurate, it can be difficult to use them due to necessity of taking shots of a special calibration object like a chessboard. Also, manual calibration harms user experience that is considered crucial for mobile applications. As a result software developers and researchers are very interested in auto-calibration methods.

Auto-calibration is the process of estimating internal camera parameters directly from multiple uncalibrated images. This area of computer vision is in active research stage. From one hand there are papers describing successful attempts of using auto-calibration methods in practical tasks (e.g. augmented reality, 3D reconstruction, image mosaics, see [6], [7], [8], [10], [11]). As the topics of these papers aren't camera auto-calibration itself, they don't contain thorough investigations of the used methods with numerical evaluation, tested on challenging dataset. As a consequence, when one faces a computer vision problem that requires camera parameters, it's very difficult to select a robust auto-calibration method and reuse previous results. There is research that is directly devoted to

M.L. Gavrilova et al. (Eds.): Trans. on Comput. Sci. XIX, LNCS 7870, pp. 63–77, 2013.

the auto-calibration problem (see [9], [12]). Unfortunately, these papers either don't compare with state-of-the-art pattern-based calibration methods or provide evaluation for synthetic datasets only. Some of these papers describe results for real datasets, but obtained under almost ideal conditions like no noise, no hand shaking, see [12]. So to the best of our knowledge we are not aware of a research paper that describes an auto-calibration method and provides sufficient experimental evidence showing robustness for practical applications.

While classical calibration methods are well studied, they suffer from some drawbacks, which follow from the fact that these methods use some extra information. For instance, there are calibration methods (see [1]) which require location of vanishing points (i.e. points where infinite lines are terminated under projective transformation) as input, but finding of these points automatically is a difficult problem.

This paper shows that under moderate assumptions an auto-calibration algorithm for rotational cameras presented in [1] can be used for practical applications with a necessary pre-processing step. We evaluate an implementation of the method for both simulated datasets and real image sequences generated by mobile phone cameras.

2 Rotational Camera Auto-calibration

2.1 Problem Statement

We use the following camera model which describes how a 3D scene point $(X, Y, Z)^T$ is projected into an image pixel with coordinates $(u, v)^T$:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \simeq K(R|T) \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{1}$$

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2}$$

where K is camera intrinsic parameters matrix (f_x, f_y are focal lengths in pixels, c_x, c_y are principal point coordinates); R, T are camera rotation 3×3 matrix and translation 3D vector; the sign $\simeq$ here denotes similarity up to scale.

The class of auto-calibration methods that we will consider requires an existence of homography mapping between all input images. The easiest way of generating a sequence of images with homography relationship using a mobile camera is to take shots of distance objects. Hence, within the scope of this paper we will make an assumption that camera translation T is negligibly small compared to the distance to the objects. We will call a device with $T = 0$ a "rotational camera".

We formulate the auto-calibration problem in the following way: given keypoints in input images taken by a rotational camera, and the keypoint correspondences between images, find the camera matrix K.

2.2 Intrinsic Parameters Error Effect

The estimation of K is never the final goal of a computer vision application. So, in order to understand how precise an auto-calibration method has to be, we need to consider a specific application. This section contains a theoretical and experimental analysis for the image mosaicing problem and provides experimental evaluation on the stitching module of OpenCV library [17]. Throughout this section we make an assumption that f_x equals to f_y for the sake of simplicity and without loss of generality, as images always can be scaled to achieve of unit pixel aspect ratio.

It is possible to stitch images without involving camera matrix. In that case a user wouldn't be able to select another surface for projection except for plane, that can be inappropriate for big panoramas because of big deformations. A plane projection surface generates deformations in the panorama image are visible when the vector of camera orientation differs a lot from the projection plane normal. The most convenient projection surface for the case of rotational cameras is a sphere.

Below we analyze warping errors when the projection surface is a sphere. To compute the error for each image we do the following:

1. For each pixel $q = (x, y, 1)^T$ of the source image we find a ray, passing through the corresponding scene point from camera center, as $r = K^{-1}q$, where K is the camera matrix.
2. We find the intersection point $(X, Y, Z)^T$ of the ray with the unit sphere centered at the origin. This point spherical coordinates u, v after scaling by constant s are point coordinates on the final panorama (s is usually selected being roughly close to the focal length in pixels):

$$u = s \cdot \tan^{-1}(\frac{X}{Z}) \tag{3}$$

$$v = s \cdot (\pi - \cos^{-1}(\frac{Y}{\sqrt{X^2 + Y^2 + Z^2}})) \tag{4}$$

3. To calculate per pixel error we project points using the ground truth camera matrix

$$K^{(gt)} = \begin{pmatrix} f^{(gt)} & 0 & c_x^{(gt)} \\ 0 & f^{(gt)} & c_y^{(gt)} \\ 0 & 0 & 1 \end{pmatrix} \tag{5}$$

and its estimation

$$K^{(est)} = \begin{pmatrix} f^{(gt)} f^{(rel)} & 0 & c_x^{(gt)} c_x^{(rel)} \\ 0 & f^{(gt)} f^{(rel)} & c_y^{(gt)} c_y^{(rel)} \\ 0 & 0 & 1 \end{pmatrix} \quad (6)$$

where $f^{(rel)}, c_x^{(rel)}, c_y^{(rel)}$ are estimated camera parameters relative to the ground truth. The distance between two points obtained using $K^{(gt)}$ and $K^{(est)}$ is the warping error in the pixel p.

According to the presented algorithm we first get two ray directions:

$$r^{(gt)} = \begin{pmatrix} X^{(gt)} \\ Y^{(gt)} \\ Z^{(gt)} \end{pmatrix} = \left(K^{(gt)}\right)^{-1} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad (7)$$

$$r^{(est)} = \begin{pmatrix} X^{(est)} \\ Y^{(est)} \\ Z^{(est)} \end{pmatrix} = \left(K^{(est)}\right)^{-1} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad (8)$$

Then we use (3) and (4) to get pixels coordinates $(u^{(gt)}, v^{(gt)})^T$ and $(u^{(est)}, v^{(est)})^T$. The final pixel warp error equals to $\sqrt{(u_{gt} - u_{est})^2 + (v_{gt} - v_{est})^2}$. We assess warping errors for the case of 2048×1536 images and using the following camera matrix as a reference:

$$K^{(gt)} = \begin{pmatrix} W + H & 0 & \frac{W}{2} \\ 0 & W + H & \frac{H}{2} \\ 0 & 0 & 1 \end{pmatrix} \quad (9)$$

where W and H are image width and height respectively.

The warping error function charts for 5% relative errors in camera intrinsic parameters are shown in figures 1 and 2. We can see from charts, that when relative error in camera parameters is 5% warp error reaches 60 pixels, that seems to be high enough for leading to visible artifacts.

In order to evaluate the artifacts, we stitched 1536×2048 images using camera matrix $K^{(pt)}$ as ground truth $K^{(gt)}$, where $K^{(pt)}$ was the camera matrix obtained via a pattern based calibration method. Also we did experiments using camera matrix $K^{(est)}$, where each parameter was modified (one at a time) to get 10% error (relative to $K^{(pt)}$). We got panoramas without visible artifacts, see figure 3. Small artifacts are highlighted with red color, but the quality of the panoramas is much higher that we could expect from theoretical analysis.

Such results are obtained because current stitching applications (including the one used for testing) use seam estimation methods to minimize visible artifacts, see [13]. After estimating seams special blending methods are used to hide discrepancies between images, see [14]. So even if the image registration step introduces moderate errors, a combination of modern seam estimation and blending methods can remove a lot of possible artifacts. But if errors in camera

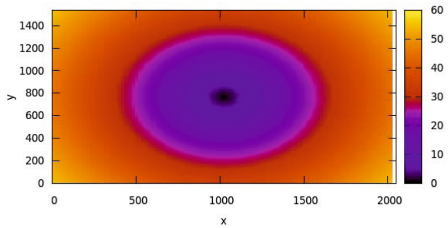

Fig. 1. Pixel warp error for $f^{(rel)} = 1.05$

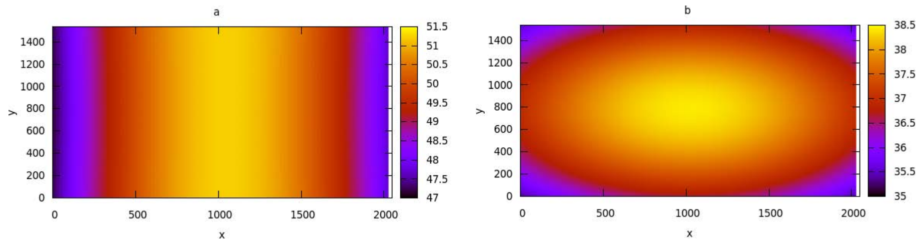

Fig. 2. Pixel warp error for a) $c_x^{(rel)} = 1.05$, b) $c_y^{(rel)} = 1.05$

parameters is too high then it's almost impossible to hide stretches and other artifacts, see figure 4 with results for $f^{(rel)} = 0.7$ (i.e. 30% relative error).

Also it should be mentioned that motions between images are estimated to minimize overall re-projection error (that is minimizing visible mis-registration error) according to the current camera matrix. This step is very important as minimizing re-projection errors leads to minimizing visible artifacts even if the camera matrix was estimated inaccurately.

From these results it follows that if one has a high quality stitching algorithm then the effect of errors in camera matrix isn't very high, and methods less accurate than pattern based calibration can be used for camera parameters estimation. This is a good application for auto-calibration that is not as precise as pattern-based calibration but still generates a reasonable estimation of camera intrinsic parameters.

2.3 Proposed Algorithm

A robust auto-calibration algorithm faces many challenges coming from data generated by a mobile device. Some input images can be noisy, can differ in illumination, and undesired objects such as user fingers can be present in the camera field of view. All these issues can affect the quality of extracted features, and can lead to mis-registration. Hence, let alone the core auto-calibration problem, we have to address these issues. This is why we start with a description of our registration algorithm.

The outputs of the registration algorithm is the images graph, where vertices are images from the input image sequence, and two images are connected with

Fig. 3. Panoramas for $f^{(rel)} = 1.1$, $c_x^{(rel)} = 1.1$, and $c_y^{(rel)} = 1.1$ respectively

Fig. 4. Panorama for $f^{(rel)} = 0.7$ with visible artifacts and stretches

the edge iff we were able to register them with a homography transformation. Here is the description of the registration pipeline:

1. Find keypoints and their descriptors of each image. We use SURF detector and descriptor implemented in OpenCV library, see [15].
2. For each image pair find matches between keypoints. We use FLANN matcher integrated into OpenCV library, see [16].
3. For each image pair estimate 2D homography and compute number of inlier matches, see 2.3.
4. For each image pair determine whether matches between these images are trustworthy, see section 2.3. The decision is made for image pair, not for each match. So if we're confident then we add an edge between two corresponding vertices into images graph.
5. Retain the biggest connected component from the images graph. Also retain only matches for confident image pairs and continue working with this connected component.

Computing Match Confidence. We follow the method proposed in [2], where it is applied to extract a subset of images from the original raw set for subsequent stitching.

Suppose we have n_f feature matches. The correctness of an image match is represented by the binary variable $m \in \{0,1\}$. The event that the i^{th} feature match $f^{(i)} \in \{0,1\}$ is an inlier/outlier is assumed to be independent Bernoulli event, so the total number of inliers n_i is Binomial. If $m = 1$ then n_i has the $B(n_i; n_f, p_1)$ distribution function, and $B(n_i; n_f, p_0)$ otherwise, where p_1 is the probability that a feature is an inlier given a correct image match, and p_0 is the probability a feature is an inlier given a false image match.

Here is the final criterion used by the authors to accept an image match

$$\frac{B(n_i; n_f, p_1)P(m=1)}{B(n_i; n_f, p_0)P(m=0)} \geq \frac{p_{min}}{1-p_{min}} \tag{10}$$

Choosing the values for $p_1 = 0.6$, $p_0 = 0.1$, $P(m = 1) = 10^{-6}$ and $p_{min} = 0.999$ gives the condition

$$n_i > \alpha + \beta n_f \tag{11}$$

for a correct image match, where $\alpha = 8.0$ and $\beta = 0.3$. We decide whether a feature match is an inlier or an outlier by comparing reprojection error with a fixed threshold. We used the same value of 3 pixels for all datasets and that value worked good enough in practice, while for each particular dataset another threshold value can be better.

The value $\frac{n_i}{\alpha+\beta n_f}$ is used as the measure of confidence that it makes sense to use matches between an image pair. If it's greater than 1 then an image match is correct, false otherwise. In some practical cases it could be useful to increase this threshold as was found in experiments.

Figure 5 shows how reprojection error threshold affects on average camera parameters estimation relative error Q for one of real datasets.

$$Q = \frac{1}{4}(|f_x^{(rel)} - 1| + |f_y^{(rel)} - 1| + |c_x^{(rel)} - 1| + |c_y^{(rel)} - 1|) \tag{12}$$

Proposed Algorithm Details. For auto-calibration we use the algorithm for the rotation only cameras case proposed in [1]. Here is the brief description of that algorithm:

1. Normalize the homographies $H_{i,j}$ between views i and j such that $\det H_{i,j} = 1$.
2. Compute $\omega = (KK^T)^{-1}$ from the equations

   $$\omega = H_{j,i}^T \omega H_{j,i}$$

   for all image pairs i, j.
3. Compute K solving $\omega = (KK^T)^{-1}$ with the Cholesky decomposition.
4. Refine K by minimizing the re-projection error function

   $$err(K, R_1, ..., R_n) = \sum_{i,j,k} \|x_j^{(k)} - H_{i,j}x_i^{(k)}\|$$

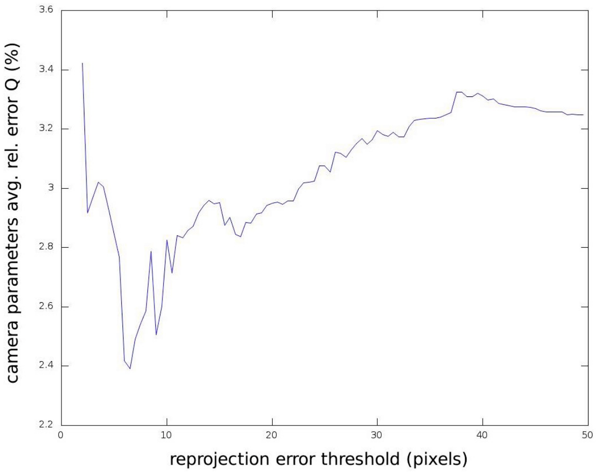

Fig. 5. Reprojection error threshold effect on camera parameters estimation errors. When the threshold is too low the algorithm is too sensitive to noise, while in the case of too high threshold even incorrect matches can be classified as inliers.

using parametrization of $H_{i,j} = KR_jR_i^TK^{-1}$ over camera rotations R_i, R_j and camera matrix K, where n is the number of images and $x_i^{(k)}, x_j^{(k)}$ are the position of k-th point measured in the i-th and j-th images respectively. We parametrize a rotation with a 3-dimensional vector directed parallel to the rotation axis and with the length equal to the rotation angle.

2.4 Experiments

We performed experiments on real datasets taken with Nokia 6303C mobile phone (1536×2048 resolution) and Logitech QuickCam Pro 900 (1600×1200 resolution).

Results for Nokia 6303C. Table 1 presents results we got using Nokia 6303C camera. We compare the auto-calibration results with pattern-based calibration: $f_x^{(err)} = f_x^{(rel)} - 1 = \frac{f_x^{(est)}}{f_x^{(pt)}} - 1$. The auto-calibration algorithm gives relative errors less than 10% on 3 out of 5 datasets. We have showed before that a relative error of less than 10% in camera parameters is enough for getting visually acceptable panoramas.

There are two factors affecting calibration quality. The first factor is the number of images in input dataset, because if the input dataset is too small then it doesn't provide enough information for camera auto-calibration. The second factor is non-zero translation presence, as the auto-calibration method we use was designed under the rotational camera assumption. This assumption is easily violated in practice as a user tends to rotate camera not around its optical center, but around device center (or itself), which is not the same.

Table 1. Relative errors in intrinsic camera parameters

Number of Images	Distance (m)	Relative Error (%)			
		$f_x^{(err)}$	$f_y^{(err)}$	$c_x^{(err)}$	$c_y^{(err)}$
6	2	8.5	11.1	4.7	4.6
7	0.5	-3.8	-2.6	-12.4	-6.2
9	2	-3.4	0.1	2.5	5.4
13	2	2.6	7.6	1.5	8.9
14	30	5.6	6.5	-1.9	4.2

Results for Logitech QuickCam Pro 900. Table 2 presents result we got using Logitech QuickCam Pro 900 camera. For this camera we achieved the relative error less than 9% in comparison with OpenCV pattern based calibration results.

Table 2. Relative errors in intrinsic camera parameters

Number of images	Distance (m)	Relative Error (%)			
		$f_x^{(err)}$	$f_y^{(err)}$	$c_x^{(err)}$	$c_y^{(err)}$
10	2	0.5	5.3	3.6	-0.3
30	2	1.2	4.4	0.7	2
57	2	0.3	3.1	1.5	3.2
10	2	-1.8	0.7	-2.5	1.3
30	2	1.9	6	-0.3	8.6
74	2	0.1	4.3	0.2	7.6

3 Stereo Rig Auto-calibration

3.1 Problem Statement

Stereo camera (or stereo rig) is a rigid couple of two mono cameras described by the model (1). The mapping between a scene 3D point $(X, Y, Z)^T$ and the corresponding pixels $(u_1, v_1)^T, (u_2, v_2)^T$ on two images obtained by the stereo rig looks as follows:

$$\begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} \simeq K(R|T) \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{13}$$

$$\begin{pmatrix} u_2 \\ v_2 \\ 1 \end{pmatrix} \simeq K(R_{rel}R|R_{rel}T + T_{rel}) \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{14}$$

where K is camera intrinsic parameters matrix defined in (2) and it's assumed to be same for the both cameras; R, T are stereo rig rotation 3×3 matrix and translation 3D vector; R_{rel}, T_{rel} are rotation 3×3 matrix and translation 3D vector between the cameras in the stereo rig.

Given pixel coordinates in a monocular image we can reconstruct only a 3D ray that contains the corresponding 3D point. But in the case of a stereo rig two cameras are available, so it's possible to reconstruct 3D scene points. To reconstruct a scene we must know rotation R_{rel} and translation T_{rel} between the cameras in the rig.

We formulate the problem of stereo rig auto-calibration in a way similar to rotation auto-calibration. Input data of the method are keypoints in image pairs taken by a stereo rig, which may undergo arbitrary Eucliden motion, and the correspondences between these keypoints. The final goal is to find camera intrinsic parameters matrix K, cameras relative rotation and translation, i.e. R_{rel} and T_{rel} respectively. Cameras relative rotation and translation are attributed as stereo rig parameters because the cameras are coupled rigidly, as consequence R_{rel} and T_{rel} remain constant over time.

3.2 Camera Intrinsic Parameters Error Effect

A typical task for a stereo rig is 3D reconstruction, i.e. inferring of 3D structure of a scene that is visible on input image pairs. In this section we estimate how errors in camera intrinsic parameters affect reconstruction precision. We make a thought experiment where we vary estimated camera intrinsic parameters while R_{rel} and T_{rel} remain constant and correct.

Suppose we have a 3D point $(X, Y, Z)^T$ and a stereo rig with two cameras located at points $c_1 = (0, 0, 0)^T$ and $c_2 = (0, 0, 1)^T$ respectively, so $T_{rel} = (0, 0, 1)^T$. We assume $R_{rel} = I$: that means both cameras in the stereo rig are oriented the same way. It should be mentioned that the measure unit isn't specified, so an estimation of T_{rel} is defined up to a scale. Regarding the cameras intrinsic parameters we make the same assumptions, as in section 2.2, equation (5). After all the assumptions we've made the stereo rig model looks like this:

$$\begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} \simeq K^{(gt)} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{15}$$

$$\begin{pmatrix} u_2 \\ v_2 \\ 1 \end{pmatrix} \simeq K^{(gt)} \left(\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} - c_2 \right) \tag{16}$$

where $K^{(gt)}$ is defined by (6). Given a 3D point images $(u_1, v_1)^T$, $(u_2, v_2)^T$, and camera intrinsic parameters estimation K_{est} we can reconstruct the point coordinates, they are as follows:

$$\begin{pmatrix} X^{(est)} \\ Y^{(est)} \\ Z^{(est)} \end{pmatrix} = \begin{pmatrix} X + \frac{c_x^{(gt)}(1-c_x^{(rel)})}{f^{(gt)}} Z \\ Y + \frac{c_y^{(gt)}(1-c_y^{(rel)})}{f^{(gt)}} Z \\ f^{(rel)} Z \end{pmatrix} \tag{17}$$

We can build an error function which, obviously, doesn't depend on X and Y:

$$err(K^{(gt)}, K^{(est)}, Z) = \left| \begin{pmatrix} X^{(est)} \\ Y^{(est)} \\ Z^{(est)} \end{pmatrix} - \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \right| = |Z| \left| \begin{pmatrix} \frac{c_x^{(gt)}(1-c_x^{(rel)})}{f^{(gt)}} \\ \frac{c_y^{(gt)}(1-c_y^{(rel)})}{f^{(gt)}} \\ f^{(rel)} - 1 \end{pmatrix} \right| \tag{18}$$

Making the same assumption about view of the matrix $K^{(gt)}$ as in section 2.2, we present the reconstruction error function plots on figure 6.

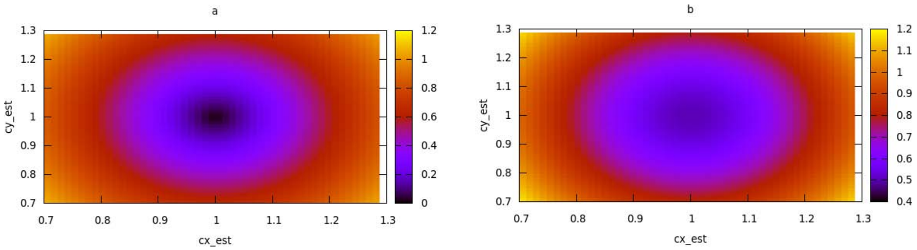

Fig. 6. Reconstruction error plots for $Z = 10$, a) $f^{(rel)} = 1$, b) $f^{(rel)} = 1.05$

Since the estimation of T_{rel} we get is defined up to a scale, we may assume without loss of generality, that the measure unit is 10 cm, so stereo rig baseline is 10 cm and point Z coordinate is 1 m. Then in figure 6, b we see that the reconstruction error achieves about a few centimeters when $f^{(rel)} = c_x^{(rel)} = c_y^{(rel)} = 1.05$, i.e. when there is 5% relative error in all intrinsic camera parameters.

3.3 Proposed Algorithm

In this section a stereo rig auto-calibration algorithm we built is described. The input of the algorithm are a set of image pairs, with keypoints and correspondences between them, and an initial guess for camera intrinsic parameters. The output of the method is refined camera intrinsic parameters K, rotation matrix R_{rel} and translation vector T_{rel} between cameras in the stereo rig. Here is a brief description of the algorithm:

1. For all input stereo pairs compute a fundamental matrix $F_{L,R}$ for points of left and right images of the pairs.
2. Select high quality subset of image pairs for future processing, see section 2.3.

   (a) Build a graph $G = (V, E)$, where vertices V are stereo pairs and E are edges. $(i, j) \in E$ iff matches between the left images from the i-th and j-th stereo pairs satisfy to the estimated fundamental matrix $F^{i,j}_{L,L}$ with error less than a threshold.
   (b) Leave only the biggest connected component in the graph G.
3. For each edge $(i, j) \in E$:
   (a) Compute projective reconstructions for the pairs i and j.
   (b) Find homography $H^{i,j}$ mapping i-th point cloud to j-th point cloud.
   (c) Upgrade the reconstructions from projective to Eucliden using camera intrinsic parameters initial guess and the homography $H^{i,j}$.
4. Refine camera intrinsic parameters matrix K, relative rotation matrix R_{rel} and relative translation vector T_{rel}.

For details on how to find a fundamental matrix, obtain projective reconstruction and other projective geometry related steps we refer to [1].

3.4 Experiments

We performed experiments on real datasets taken by a LG-P920 mobile phone stereo camera (1600×1200 resolution), and a stereo camera Videre STH-DCSG-9cm with resolution 640×480.

Results for LG-P920. To get ground truth stereo rig parameters we calibrated it using OpenCV. We conducted a few dozens of experiments, where intrinsic camera parameters initial relative error was selected from the $[-30\%, 30\%]$ range uniformly.

From table 3 we can see that on the first dataset we achieve less than 10% relative error in intrinsic camera parameters. Standard deviation of the relative error computed over the experiments is less than 8%. On the second dataset relative error in intrinsic camera parameters is less than 11%, while its standard deviation is less than 4%.

We also computed errors for relative rotation matrix R_{rel} and relative translation vector T_{rel} estimations. The ground truth values were computed using OpenCV calibration functionality. Instead of comparing rotation matrices directly, we first convert matrices to rotation vectors and then compare them. Also it should be mentioned that as reconstruction is built up to scale, so the relative translation vector is define up to scale too. That's why we work with translation vectors scaled in such way that its X component equals to 1. The table 4 contains relative rotation and translation vectors obtained using OpenCV calibration.

The results obtained for R_{rel} and T_{rel} using the proposed auto-calibration method are shown in tables 5 and 6.

Table 3. Camera intrinsic parameters relative errors

Dataset ID	Number of Images	Distance (cm)	Mean Relative Error (%)				Relative Error Std. Dev. (%)			
			$f_x^{(err)}$	$f_y^{(err)}$	$c_x^{(err)}$	$c_y^{(err)}$	$f_x^{(err)}$	$f_y^{(err)}$	$c_x^{(err)}$	$c_y^{(err)}$
1	6	30	1.8	2.5	-9.6	4.9	5.2	4.8	5.3	7.6
2	26	30	2.3	2.9	-10.9	1.1	1.1	1.1	1.7	3.9

Table 4. Ground truth rotation and translation vectors

Rotation Vector			Translation Vector		
x	y	z	x	y	z
0	0.04	0	1	-0.04	-0.12

Table 5. Estimated relative rotation and translation vectors means

Dataset ID	Mean Rotation Vector			Mean Translation Vector		
	x	y	z	x	y	z
1	0	0.003	-0.003	1	-0.005	-0.17
2	0.001	0.002	-0.003	1	-0.015	-0.13

Table 6. Estimated relative rotation and translation vector standard deviations

Dataset ID	Rotation Vector Std. Dev.			Translation Vector Std. Dev.		
	x	y	z	x	y	z
1	10^{-4}	$6 \cdot 10^{-5}$	$8 \cdot 10^{-5}$	0	0.002	0.017
2	$1.9 \cdot 10^{-5}$	$2.3 \cdot 10^{-4}$	$1.8 \cdot 10^{-5}$	0	0.002	0.002

Results for Videre STH-DCSG-9cm. To get ground truth stereo rig parameters we used the camera API. We conducted a few dozens of experiments, where intrinsic camera parameters initial relive error was selected from the $[-50\%, 50\%]$ range uniformly.

From table 7 we can see that on the first dataset we achieve less than 4% relative error in intrinsic camera parameters. Standard deviation of the relative error computed over the experiments is less than 2%. On the second dataset relative error in intrinsic camera parameters is less than 6%, while its standard deviation is less than 2%.

Table 7. Camera intrinsic parameters relative errors

Dataset ID	Number of Images	Distance (cm)	Mean Relative Error (%)				Relative Error Std. Dev. (%)			
			$f_x^{(err)}$	$f_y^{(err)}$	$c_x^{(err)}$	$c_y^{(err)}$	$f_x^{(err)}$	$f_y^{(err)}$	$c_x^{(err)}$	$c_y^{(err)}$
1	5	30	2.6	3.6	-1	0.2	1	0.3	1.6	0.9
2	5	30	0.5	-2.4	-5.4	-7.2	0.8	0.9	1.1	2

Table 8. Ground truth rotation and translation vectors

Rotation Vector			Translation Vector		
x	y	z	x	y	z
-0.005	$-2 \cdot 10^{-4}$	0.001	1	-0.004	-0.012

Table 9. Estimated relative rotation and translation vectors means

Dataset ID	Mean Rotation Vector			Mean Translation Vector		
	x	y	z	x	y	z
1	0.018	0.01	0.008	1	0.027	-0.024
2	0.026	0.03	0.001	1	0.007	-0.046

Table 10. Estimated relative rotation and translation vector standard deviations

Dataset ID	Rotation Vector Std. Dev.			Translation Vector Std. Dev.		
	x	y	z	x	y	z
1	1.8^{-4}	$5.8 \cdot 10^{-4}$	$1.7 \cdot 10^{-4}$	0	0.001	0.001
2	$3.7 \cdot 10^{-5}$	0.001	$4 \cdot 10^{-4}$	0	0.001	0.002

We also computed errors for relative rotation matrix R_{rel} and relative translation vector T_{rel} estimations. The table 8 contains relative rotation and translation vectors obtained using the camera API.

The results obtained for R_{rel} and T_{rel} using the proposed auto-calibration method are shown in tables 9 and 10.

4 Conclusion

We investigated the problem of auto-calibration for the case of rotational camera and stereo rig. We built a robust auto-calibration pipeline for both cases, that showed good results on real datasets.

We analyzed impact of errors in camera parameters on final results in such computer vision problem as image mosaicing. While errors in camera parameters can lead to big warping errors, we showed that using modern stitching algorithms relaxes requirements on camera parameters accuracy.

In one specific case we studied how errors in camera intrinsic parameters may affect reconstruction precision in the case of stereo rig auto-calibration.

We showed that it is possible to calibrate cameras without patterns, but the quality of input data is important for achieving accurate auto-calibration.

References

1. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press (2004) ISBN: 0521540518
2. Brown, M., Lowe, D.G.: Automatic Panoramic Image Stitching using Invariant Features. International Journal of Computer Vision 74(1), 59–73 (2007)
3. Zhang, Z.: A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 1330–1334 (2000)
4. Bradski, G., Kaehler, A.: Learning OpenCV: Computer Vision with the OpenCV Library. O'Reilly Media (2008)
5. Szeliski, R.: Image Alignment and Stitching: A Tutorial. Microsoft Research, TechReport, MSR-TR-2004-92 (2004)
6. Szeliski, R., Shum, H.-Y.: Creating full view panoramic image mosaics and texture-mapped models. In: Computer Graphics, SIGGRAPH Proceedings. Association for Computing Machinery, Inc. (1997)
7. Eden, A., Uyttendaele, M., Szeliski, R.: Seamless Image Stitching of Scenes with Large Motions and Exposure Differences. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2006)
8. Pollefeys, M.: Visual 3D Modeling from Images. Tutorial Notes (2000), `http://www.cs.unc.edu/~marc/tutorial/`
9. Nister, D.: Untwisting a projective reconstruction. International Journal of Computer Vision (2004)
10. Gibson, S., Cook, J., Howard, T., Hubbold, R., Oram, D.: Accurate Camera Calibration for Off-line, Video-Based Augmented Reality. In: ISMAR Proceedings of the 1st International Symposium on Mixed and Augmented Reality (2002)
11. Chen, J., Yuan, B.: Metric 3D reconstruction from uncalibrated unordered images with hierarchical merging. In: IEEE 10th International Conference on Signal Processing (2010)
12. Chandraker, M.K., Agarwal, S., Kriegman, D.J., Belongie, S.: Globally Optimal Algorithms for Stratified Autocalibration. IJCV 90(2), 236–254 (2010)
13. Kwatra, V., Schodl, A., Essa, I., Turk, G., Bobick, A.: Graphcut Textures: Image and Video Synthesis Using Graph Cuts. To appear in Proc. ACM Transactions on Graphics, SIGGRAPH (2003)
14. Adelson, E.H., Burt, P.J.: Multi-resolution Splining Using a Pyramid Image Representation. In: SPIE Applications of Digital Image Processing IV, pp. 204–210 (1983)
15. Bay, H., Tuytelaars, T., Van Gool, L.: SURF: Speeded Up Robust Features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006, Part I. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006)
16. FLANN - Fast Library for Approximate Nearest Neighbors (2011), `http://people.cs.ubc.ca/~mariusm/index.php/FLANN/FLANN`
17. OpenCV stitching module documentation (2012), `http://opencv.itseez.com/modules/stitching/doc/stitching.html`
18. Ji, Q., Dai, S.: Self-Calibration of a Rotating Camera with a Translational Offset. IEEE Trans. on Robotics and Automation 20(1) (February 2004)
19. Junejo, I., Foroosh, H.: Practical PTZ Camera Calibration using Givens Rotations. In: IEEE ICIP (2008)
20. Spizhevoy, A., Eruhimov, V.: Problem of auto-calibration in image mosaicing. In: International Conference on Computer Graphics and Vision, GraphiCon, Conference Proceedings, pp. 27–32 (2012)