Easy-to-Use and Accurate Calibration of RGB-D Cameras from Spheres

Aaron Staranowicz[1], Garrett R. Brown[1], Fabio Morbidi[2], and Gian-Luca Mariottini[1]

[1] Univ. of Texas at Arlington, CSE Dept.
500 UTA Boulevard, Arlington, TX 76019, USA
[2] Inria Grenoble Rhône-Alpes
655 Avenue de l'Europe, Montbonno t, France

Abstract. RGB-Depth (or RGB-D) cameras are increasingly being adopted for real-world applications, especially in areas of healthcare and at-home monitoring. As for any other sensor, and since the manufacturer's parameters (e.g., focal length) might change between models, calibration is necessary to increase the camera's sensing accuracy. In this paper, we present a novel RGB-D camera-calibration algorithm that is easy-to-use even for non-expert users at their home; our method can be used for any arrangement of RGB and depth sensors, and only requires that a spherical object (e.g., a basketball) is moved in front of the camera for a few seconds. A robust image-processing pipeline automatically detects the moving sphere and rejects noise and outliers in the image data. A novel closed-form solution is presented to accurately compute an initial set of calibration parameters which are then utilized in a nonlinear minimization stage over all the camera parameters including lens distortion. Extensive simulation and experimental results show the accuracy and robustness to outliers of our algorithm with respect to existing checkerboard-based methods. Furthermore, an *RGB-D Calibration Toolbox* for MATLAB is made freely available for the entire research community.

Keywords: RGB-Depth Cameras, Camera Calibration, Kinect, Computer Vision.

1 Introduction

RGB-Depth (or RGB-D) cameras consist of an *RGB* and a *depth* sensor that capture color images along with per-pixel depth information (depth map) [1, 2]. These features have promoted the wide adoption of low-cost RGB-D cameras (e.g., the Microsoft *Kinect* [1]) in numerous at-home applications, such as body tracking [3, 4], gait monitoring for tele-rehabilitation [5, 6], tracking of facial expressions [7], object and gesture recognition [8, 9, 10].

All the aforementioned applications require the precise knowledge of the *RGB-D camera-calibration parameters*, i.e., the relative position and orientation of its on-board RGB and depth sensors, as well as of their parameters (focal lengths,

R. Klette, M. Rivera, and S. Satoh (Eds.): PSIVT 2013, LNCS 8333, pp. 265–278, 2014.

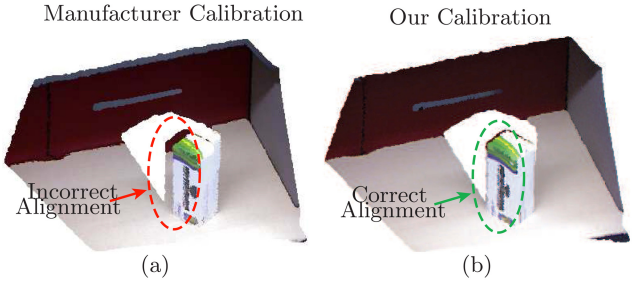


Fig. 1. *(a)* Scene perception with Kinect's manufacturer calibration. The texture is not correctly aligned with the 3-D point cloud; *(b)* Improved scene perception after our RGB-D camera calibration.

principal points, and lens-distortion [11]). As shown in Fig 1, calibrating RGB-D cameras is indeed essential to improve the sensing accuracy of these cameras since the calibration parameters can differ in each model from the manufacturer's settings.

Calibrating RGB-D cameras still represents an open challenge, especially for what concerns accuracy and ease of use for non-expert users at their home. Some approaches [12, 13] require the impractical use of an external source of infrared light (e.g., a halogen lamp). Jung et al. [14] use a custom-made large wood panel with tens of circular holes, which requires the user to perform an initial time-consuming manual correspondence association of the holes' centers between each RGB and depth image pair. Other methods [15, 16, 17, 18] have simply extended standard corner-based calibration methods to RGB-D cameras; however, these methods are not robust to outliers, and are not accurate as they make use of only a small number of corners detected in the depth image. Furthermore, they require the user to provide an initial estimate of the calibration parameters which might not be available for other RGB and depth-sensor arrangements other than the Kinect. The existing ROS Kinect-calibration toolbox [13, 19] can only be used with the Microsoft sensor, and it cannot be used to calibrate sensors (e.g., Time of Flight (ToF) cameras) that do not provide an infrared (IR) image. Furthermore, the toolbox is not user-friendly and the calibration must be performed offline since RGB and IR images cannot be simultaneously captured by the Kinect.

In this paper, we present a novel method for the calibration of RGB-D cameras that is accurate and easy-to-use even for non-expert users. Our method is *practical* since the user is only required to move a spherical object (e.g., a basketball) in front of the camera. No geometrical knowledge about the size of the basketball is required and an image-processing pipeline has been designed that automatically detects the spherical object in both the RGB image (as an ellipse) and the depth map (as a sphere), while discarding spurious data (outliers).

Our calibration algorithm is *accurate* since it relies on a novel least-squares (LS) method that is used to precisely initialize a nonlinear minimization stage over all the camera parameters including lens distortion and to remove outliers

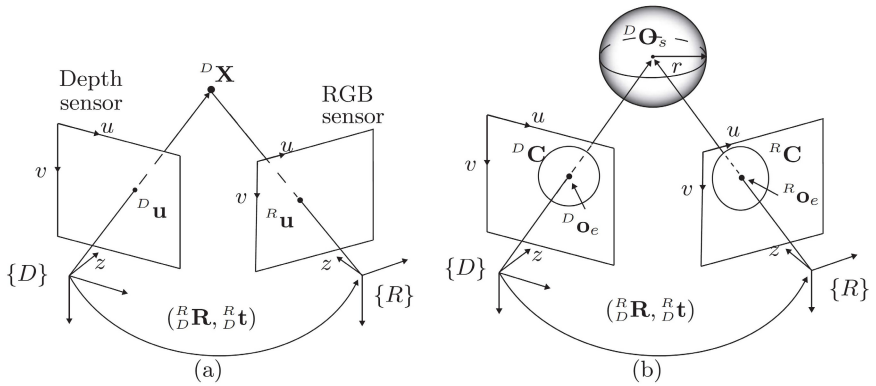


Fig. 2. *(a) Imaging model of an RGB-D camera*: (*left*), depth sensor $\{D\}$; (*right*), RGB sensor, $\{R\}$; $({}^{R}_{D}\mathbf{R}, {}^{R}_{D}\mathbf{t})$ is the 3-D rigid-body motion between the two sensors. *(b)* A *sphere* with radius r and center ${}^{D}\mathbf{O}_s$ (in $\{D\}$) is projected onto the RGB-camera image plane at an image conic, ${}^{R}\mathbf{C}$, (in general, an ellipse). The *centers of each ellipse*, i.e., ${}^{R}\mathbf{o}_e$ and ${}^{D}\mathbf{o}_e$ are the inputs of our calibration method.

by means of RANSAC [20]. This represents a significant improvement over existing algorithms (such as [15,18]) that require initial values for these calibration parameters to be provided by the user. It is also worth emphasizing here that the overall accuracy of our method is due to the adoption of spheres as calibration objects. In fact, the sensor noise in the RGB and depth images is minimized by the adoption of sphere- and ellipse-fitting algorithms in the image processing pipeline.

Our approach is *widely applicable*, and it can be potentially used to calibrate depth- and RGB-sensor pairs in *any* 3-D relative configuration (i.e., not necessarily pointing in the same direction as in the Microsoft Kinect). A new calibration toolbox for MATLAB, called *RGB-D Calibration Toolbox*, has been developed and made available on the Internet[1] for the entire research community.

The rest of this paper is organized as follows: Sect. 2 reviews some basic facts on pinhole camera modeling and image projection of spheres. Sect. 3 describes our calibration algorithm, and in Sect. 4 numerical and real-world experimental results are presented and discussed. Finally, in Sect. 5, conclusions are drawn and some promising subjects of future research are outlined.

2 Basics

2.1 RGB-D Camera Model

Figure 2(a) shows the general RGB-D imaging model considered in this paper. Under the pinhole camera model [11], a generic 3-D point in the *depth-sensor*

[1] `http://ranger.uta.edu/%7Egianluca/dcct/`

frame, $\{D\}$, ${}^{D}\mathbf{X} \triangleq [X^D,\, Y^D,\, Z^D]^T$, is projected at a pixel point ${}^{R}\mathbf{u} \triangleq [u^R,\, v^R]^T$ on the image plane of the *RGB-sensor frame* $\{R\}$, according to:

$$ {}^{R}\lambda\, {}^{R}\widetilde{\mathbf{u}} \;=\; {}^{R}\mathbf{K}\,[{}^{R}_{D}\mathbf{R} \mid {}^{R}_{D}\mathbf{t}]\, {}^{D}\widetilde{\mathbf{X}}\,, \tag{1} $$

where the tilde indicates the extension to homogeneous coordinates of a vector, and ${}^{R}\lambda$ is a scale factor, chosen so that the last coordinate of the term on the right-hand side of (1) is equal to 1. The intrinsic calibration matrices, ${}^{R}\mathbf{K}$ and ${}^{D}\mathbf{K}$, of the RGB and of the depth sensors are defined as:

$$ {}^{R}\mathbf{K} \triangleq \begin{bmatrix} f_u^R & s^R & u_0^R \\ 0 & f_v^R & v_0^R \\ 0 & 0 & 1 \end{bmatrix}, \quad {}^{D}\mathbf{K} \triangleq \begin{bmatrix} f_u^D & s^D & u_0^D \\ 0 & f_v^D & v_0^D \\ 0 & 0 & 1 \end{bmatrix}, \tag{2} $$

where f_u and f_v represent the focal lengths (in pixels) in both image-axes directions; $[u_0,\, v_0]^T$ is the principal point in pixels, and s is the skew factor [11]. Note that the superscripts D and R have been dropped since the previous expressions equally apply to both reference frames.

The depth sensor typically outputs a so-called *depth map*, i.e., a set of image points along with their corresponding depth value (in meters) on the z-axis. Each depth-map feature is thus defined as ${}^{D}\mathbf{u} \triangleq [u^D,\, v^D,\, Z^D]^T$. Differently from an RGB camera, ${}^{D}\mathbf{u}$ can be used to uniquely compute the corresponding point in *3-D coordinates*, ${}^{D}\mathbf{X}$, as:

$$ {}^{D}\widetilde{\mathbf{X}} = \begin{bmatrix} {}^{D}\mathbf{K}^{-1} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} {}^{D}\bar{\mathbf{u}}, \tag{3} $$

where $\mathbf{0}_{3\times 1}$ denotes a 3×1 matrix of zeros and the upper bar in ${}^{D}\bar{\mathbf{u}}$ indicates a vector obtained from ${}^{D}\mathbf{u}$ as follows:

$$ {}^{D}\bar{\mathbf{u}} \triangleq \left[u^D Z^D,\, v^D Z^D,\, Z^D,\, 1\right]^T. \tag{4} $$

2.2 Image Projection of a Sphere

A sphere with 3-D center, ${}^{D}\mathbf{O}_s$, and radius r, (see Fig. 2(b)) can be algebraically represented by [11]:

$$ {}^{D}\mathbf{Q} \;=\; \begin{bmatrix} \mathbf{I}_3 & -{}^{D}\mathbf{O}_s \\ -{}^{D}\mathbf{O}_s^T & {}^{D}\mathbf{O}_s^T\, {}^{D}\mathbf{O}_s - r^2 \end{bmatrix}, \tag{5} $$

where $\mathbf{I}_3$ denotes the 3×3 identity matrix. In general, a sphere ${}^{D}\mathbf{Q}$ projects in the image plane at an ellipse, and its projection in the RGB image at the ellipse ${}^{R}\mathbf{C}$ is given by [21]:

$$ {}^{R}\mathbf{C}^* = {}^{R}_{D}\mathbf{P}\; {}^{D}\mathbf{Q}^*\, {}^{R}_{D}\mathbf{P}^T \text{ where } \quad {}^{R}_{D}\mathbf{P} \triangleq {}^{R}\mathbf{K}[{}^{R}_{D}\mathbf{R}|{}^{R}_{D}\mathbf{t}], \tag{6} $$

being ${}^{R}\mathbf{C}^* = \text{adj}({}^{R}\mathbf{C})$ (the adjugate of ${}^{R}\mathbf{C}$) the dual conic of ${}^{R}\mathbf{C}$, and ${}^{D}\mathbf{Q}^* = \text{adj}({}^{D}\mathbf{Q})$, (note that the adjugate is equal to the inverse if the matrix is invertible). The conics ${}^{R}\mathbf{C}$ and ${}^{D}\mathbf{C}$ will play a key role in the RGB-D camera-calibration algorithm described in the next section.

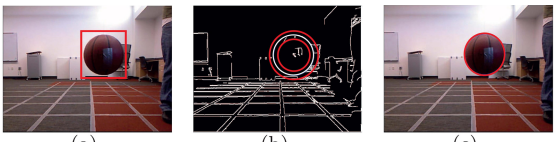
3 Joint RGB-Depth Camera Calibration

3.1 Overview

During calibration we assume that the user moves the spherical calibration object in front of the RGB-D camera. *Feature-Extraction* pipeline detects and tracks in real time the two ellipses ${}^{R}\mathbf{C}$ and ${}^{D}\mathbf{C}$ corresponding to the projection of the sphere in both the RGB and depth sensor, respectively.

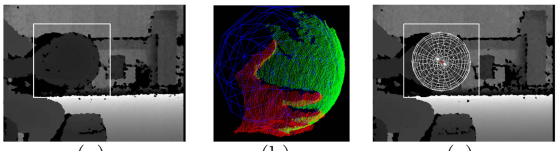
Given a set of ellipses acquired over a certain time frame, our goal is to compute an estimate of the intrinsic parameters ${}^{D}\mathbf{K}$ and ${}^{R}\mathbf{K}$ (plus the radial lens-distortion parameters, $\mathbf{k}_c^D$ and $\mathbf{k}_c^R$), as well as of the extrinsic parameters ${}^{R}_{D}\mathbf{R}$ and ${}^{R}_{D}\mathbf{t}$. We do this in two steps: first, an initial and accurate least-squares solution is obtained with a novel formulation that leverages all the detected (image) centers ${}^{R}\mathbf{o}_e$ and ${}^{D}\mathbf{o}_e$ of the two ellipses. Second, the parameters and inliers estimated in the previous phase are used to provide a refined (and final) estimate of all the camera-calibration parameters.

It is worth pointing out here the novelty and importance of the least-square approach to get an initial and accurate estimate of the camera parameters, while almost all of the state-of-the-art calibration methods require a separate (thus sub-optimal) estimate. Sometimes, the manufacturer's calibration parameters are used as initial values (e.g., the Kinect). However, these are not available for general RGB- and depth-sensor arrangements (e.g., Time-of-flight camera and webcam). In addition, the calibration parameters can vary significantly among different models (cf. Fig. 1(a)).



(a) (b) (c)

Fig. 3. *RGB Feature-Detection Pipeline: (a)* A background subtraction is used to limit the search image space. *(b)* Canny-edge detector is used to find image edges; upper and lower bounds are used in Hough voting to find the edges of the ellipse. *(c)* RANSAC-based ellipse fitting finally detects the ellipse.



(a) (b) (c)

Fig. 4. *Depth Feature Detection Pipeline: (a)* The depth image presents irregular sphere edges: the white box shows the search area obtained with background subtraction. *(b)* RANSAC on point cloud to detect inliers (green) and outliers (red) and to fit the sphere (blue). *(c)* The fitted-sphere center is reprojected on the image (red dot).

3.2 Feature Detection Pipeline

In this phase, the RGB and the depth images are processed to automatically detect and track the two ellipses, ${}^{R}\mathbf{C}$ and ${}^{D}\mathbf{C}$, which correspond to the projection of the sphere on the image planes of the two sensors. In our implementation, the user is simply required to move the sphere in front of the RGB-D camera. We have observed that tracking one ellipse in each image is more robust than tracking single corner features (e.g., in the case of a checkerboard calibration rig), since our method leverages ellipse and sphere-fitting algorithms that make use of *all* the observed points on an ellipse to minimize the image noise while removing potential outliers.

For the detection of the ellipse ${}^{R}\mathbf{C}$ in the RGB image, a background subtraction algorithm is first used to determine a search area for the moving sphere (see Fig. 3(a)). Then a Canny-edge detector [22] is applied to that image portion, and a circular Hough transform [23] is used to automatically obtain the pixel coordinates of those points lying on the ellipse (as in Fig. 3(b)). Finally, a RANSAC-based ellipse-fitting algorithm [24, 25] is used to estimate the parameters of the ellipse ${}^{R}\mathbf{C}$, such as its center ${}^{R}\mathbf{o}_e$ (see Fig. 3(c)).

For the detection of the ellipse ${}^{D}\mathbf{C}$ in the depth image, adopting the same strategy for the RGB image would not work, as the edges of the sphere in the depth image are highly irregular (see Fig. 4(a)). Our approach consists of first using a random ${}^{D}\mathbf{K}$ to reproject in 3-D with (3) those depth-image points ${}^{D}\mathbf{u}$ within a search area around the sphere detected with background subtraction. Note that the random ${}^{D}\mathbf{K}$ does not distort, but only scales the resulting 3-D point cloud. We then use a Hough voting scheme to detect which 3-D points belong to the sphere, and RANSAC to fit a sphere. The corresponding image points in the depth map are finally deemed as inliers (cf., Fig. 4(b)). Finally, the center of the sphere is reprojected on the depth image thus obtaining the center of the ellipse (red dot in Fig. 4(c)).

As mentioned above, our feature-detection pipeline has two major advantages when compared to the detection and tracking of corner features in checkerboard-based methods [16,17,14,13]. First, because of the Hough voting and the RANSAC fitting, our feature-extraction phase reduces noise in the images and is robust to outliers (such as user's hand or fingers while holding the spherical object). Second, the feature-detection procedure is *completely automatic* and *robust* to severe occlusions that can often occur in a non-lab setting.

3.3 Initial Least-Squares Solution

Differently from the existing methods, our calibration strategy does not require the intervention of an expert user to initialize the RGB-D calibration parameters. We obtain an initial estimate of ${}^{R}\mathbf{K}$ by using the state-of-the-art method described in [26] which uses the same RGB images obtained from Sect. 3.2. Our novel least-squares method accurately estimates the camera parameters ${}^{D}\mathbf{K}$, ${}^{R}_{D}\mathbf{R}$, and ${}^{R}_{D}\mathbf{t}$, by using the extracted ellipse centers.

By substituting (3) in (1), and by using the ellipse center positions ${}^{D}\mathbf{o}_e$ and ${}^{R}\mathbf{o}_e$ (pixels), it is straightforward to obtain the following expression[2]:

$$\left[{}^{R}\mathbf{K}^{-1}\,{}^{R}\widetilde{\mathbf{o}}_e\right]_{\times}\underbrace{\left[{}^{R}_{D}\mathbf{R}\,{}^{D}\mathbf{K}^{-1}\mid{}^{R}_{D}\mathbf{t}\right]}_{{}^{R}_{D}\mathbf{M}}\,{}^{D}\bar{\mathbf{o}}_e=\mathbf{0}_{3\times1}, \tag{7}$$

where ${}^{D}\bar{\mathbf{o}}_e\in I\!R^4$ is obtained from ${}^{D}\mathbf{o}_e\in I\!R^3$ as the ${}^{D}\bar{\mathbf{u}}$ in (4). We have empirically observed that the sample mean of the z-coordinates of the visible points on a sphere, can be successfully used as Z^D in ${}^{D}\bar{\mathbf{o}}_e$.

The interesting fact about (7) is that this expression can be solved as a standard direct-linear-transformation (DLT) method [11]. A closed-form estimate of ${}^{R}_{D}\mathbf{M}$ is then obtained from (7) from at least six RGB-depth image pairs of a single sphere. Once the ${}^{R}_{D}\mathbf{M}$ matrix has been estimated and normalized (so that ${}^{R}_{D}\mathbf{M}_{3,3}=1$), its fourth column coincides with ${}^{R}_{D}\mathbf{t}$, while ${}^{D}\mathbf{K}$ and ${}^{R}_{D}\mathbf{R}$ are easily obtained from a QR factorization of its left-upper 3×3 submatrix.

Our DLT solution uses RANSAC to remove outliers (e.g., blurred images). Degenerate cases in solving the DLT problem [11] occur when all the spheres' centers lie on a common plane or line. While this case is quite rare in practice, we address this possible problem by checking the rank of the homogeneous system used by DLT to discard those ellipses sampled at each RANSAC iteration.

3.4 Nonlinear Minimization Step

In this phase, we use the initial estimates as inputs to a nonlinear-minimization procedure to obtain refined Maximum-Likelihood (ML) estimates of *all* the calibration parameters (intrinsic, extrinsic, and lens distortion).

Consider the pinhole camera model in (1); by assuming that each point in $\{R\}$ follows a Gaussian distribution with the ground-truth position as its mean and measurement-noise covariance $\mathbf{\Phi}$, the log-likelihood function can be written as:

$$\mathcal{L}_1=-\frac{1}{2N}\sum_{i=1}^{N}\boldsymbol{\epsilon}_i^T\mathbf{\Phi}^{-1}\boldsymbol{\epsilon}_i, \tag{8}$$

where

$$\boldsymbol{\epsilon}_i={}^{R}\widetilde{\mathbf{o}}_e^i-\frac{1}{{}^{R}\lambda_i}L(\mathbf{k}_c^R)\,{}^{R}\mathbf{K}\,[{}^{R}_{D}\mathbf{R}\,|\,{}^{R}_{D}\mathbf{t}]\,{}^{D}\widetilde{\mathbf{O}}_s^i\,, \tag{9}$$

where ${}^{R}\lambda_i$ is an unknown scale factor. Note that ${}^{D}\widetilde{\mathbf{O}}_s^i$ is the center of the $i-$th sphere obtained from both the depth-map points and the current estimate of ${}^{D}\mathbf{K}$, by using our sphere-fitting RANSAC algorithm described in Sect. 3.2, and that the radial-distortion function $L(\mathbf{k}_c^R)$ is as described in [11].

Another source of useful information comes from the conic reprojection constraint in (6). As such, and similarly to the previous derivations, its log-likelihood can be written as:

$$\mathcal{L}_2=-\frac{1}{2N}\sum_{i=1}^{N}\frac{1}{\left(\sigma_q^i\right)^2}\|{}^{R}\mathbf{C}_i^*-{}^{R}_{D}\mathbf{P}\,{}^{D}\mathbf{Q}_i^*\,{}^{R}_{D}\mathbf{P}\|_F^2, \tag{10}$$

[2] $[\mathbf{x}]_{\times}$ denotes the skew-symmetric matrix associated to a vector $\mathbf{x}\in I\!R^3$.

where $\|\cdot\|_F$ denotes the Frobenius norm and the dual quadric ${}^{D}\mathbf{Q}_i^*$ is calculated with (5) by using the estimated 3-D center, ${}^{D}\widetilde{\mathbf{O}}_s^i$, and radius, r_i, of the i-th sphere determined as described in Sect. 3.2. As found in [18], the variance $\left(\sigma_q^i\right)^2$ has been chosen to be a quadratic function of the distance to each sphere Z^D, which in our experiments revealed effective. The radial distortion function $L(\mathbf{k}_c^D)$ is used here for the reprojection error of the ellipse center.

Combining (8) and (10) together, we maximize the overall log-likelihood as:

$$\underset{{}^{D}\mathbf{K},\mathbf{k}_c^D,{}^{R}\mathbf{K},\mathbf{k}_c^R,{}^{R}_{D}\mathbf{R},{}^{R}_{D}\mathbf{t}}{\operatorname{argmin}} \quad \rho_1\mathcal{L}_1 + \rho_2\mathcal{L}_2, \tag{11}$$

where ρ_1 and ρ_2 are positive weighting parameters.

4 Simulation and Experimental Results

4.1 Simulation Results

We realized an simulation scenario to assess the accuracy of our method against the ground-truth over a large set of realizations. In this scenario, a set of spheres are positioned in front of the camera, and are evenly spaced of 25 cm within a cubical volume. This arrangement was only selected to make our validation procedure repeatable and systematic: comparable results could be obtained with spheres randomly placed within the same cubical volume.

In the first test, we examined the performance of our method in the case of increasing image noise (over 100 iterations for each noise level), while observing 90 spheres. A zero-mean white Gaussian noise with increasing power was added to the RGB and to the depth images. In particular, the pixel standard deviations in both images, σ_p^R and σ_p^D, were chosen in the range from 0 to 1 pixels. A noise on the depth measurements was also simulated; in order to match our empirical observations of Kinect's accuracy, this standard deviation σ_m^D [m] has been chosen as a quadratic function of the distance of each sphere from the camera. In this first test, we also compared the performance of our calibration method against Zhang's method [16]. For testing Zhang's algorithm, we created twelve 9×9 calibration checkerboard patterns observed from 21 different viewpoints. Note that an initial comparison for different Kinect calibration methods was performed in an earlier work [27].

Figs. 5(a)-5(f) report the comparison between our method and Zhang's method for an increasing noise power. In particular, Figs. 5(a)-5(b) show the standard deviation for each of the translation error-vector components, $\mathbf{e} = [e_x,\ e_y,\ e_z]^T = {}^{R}_{D}\hat{\mathbf{t}} - {}^{R}_{D}\mathbf{t}$, for both our and Zhang's algorithms, respectively. We omitted the plots of the mean errors which were all around zero, thus indicating unbiased estimates. As the figures show, our method results in an error lower than 1 mm on each axis, whereas the error for Zhang's method is about 7 cm for the image noise equal to 1.5 pixels. Similarly, Figs. 5(c)-5(d) report the standard deviation of the estimated rotation between the RGB and depth cameras (roll-pitch-yaw angle errors in degrees) for our method and Zhang's, respectively. In this case our estimates are almost three orders of magnitude lower than Zhang's. These results clearly

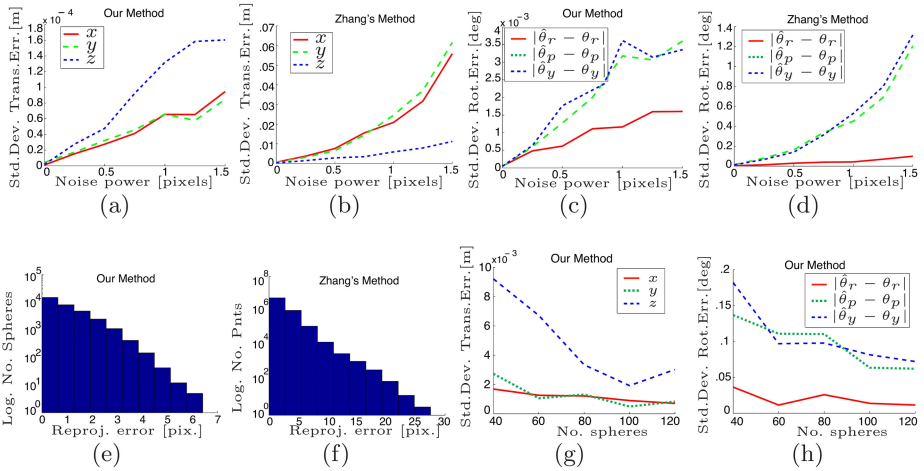


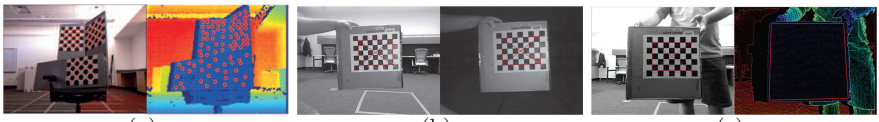
Fig. 5. *Simulation results*: *(a)-(d)* Translation and rotation estimation errors (standard deviation) with increasing noise power for our method (*(a)* and *(c)*) and Zhang's (*(b)* and *(d)*). *(e)-(f)* Distribution of the reprojection errors obtained using our method and Zhang's method (note the different scale on the vertical axis). *(g)-(h)* Translation and rotation errors for an increasing number of spheres.

indicate the power of our method: as expected, the conic fitting and the robust estimators of our method have the capacity to decrease the overall sensitivity to noise in the data, and allow a final accurate estimate. Figs. 5(e)-5(f) report the distribution of the pixel reprojection error when $\sigma_p^R = \sigma_p^D = 1$ pixel for 100 realizations. Our calibration method has a reprojection error always lower than 6 pixels while Zhang's method achieves a higher reprojection error (27 pixels).

In the second test, we studied the influence of an increasing number of spheres (from 40 to 120) on the estimated RGB-D calibration parameters. The standard deviation of the Gaussian image noises was fixed to $\sigma_p^R = \sigma_p^D = 1$ pixel. For each given number of spheres, we considered 100 realizations. Figs. 5(g)-5(h) illustrate the results as standard deviation of the translation and angular error, and show that 40 image-pairs of a sphere are generally sufficient to get a very accurate estimate (e.g., std. deviation lower than 1 mm for translation) of the calibration parameters.

4.2 Experimental Results

We used the Microsoft Kinect to compare the accuracy of our calibration method against Zhang's method [16], Herrera's method [17], and the Robot Operating System (ROS) Kinect calibration toolbox [19]. As detailed below, a ground truth set of calibration parameters was obtained by means of the Stereo Camera MATLAB Calibration Toolbox (CalTechTbx) [28]. We will first give a brief overview of the aforementioned calibration methods and describe their pros and cons. Then, we will present the calibration results.



(a) (b) (c)

Fig. 6. Feature extraction for: *(a)* Zhang's Method (RGB + Depth); *(b)* ROS Calib Tbx. (RGB + IR); *(c)* Herrera's Method (RGB + Disparity).

Our Calibration: For our method, we subsampled 130 RGB-D image pairs of a basketball (from 1 to 4 meters) from a video of approximately 30 seconds by running the feature-extraction procedure described in Sect. 3.2. We randomly subsampled every 7th video frame. Any spurious frames would be discarded in Sect. 3.3.

Overview of the Other Calibration Methods: The aforementioned methods differ from our calibration method since they rely on checkerboard calibration panels that must be simultaneously observed by both the RGB and depth sensor.

Each method uses a set of RGB images of the checkerboard to estimate ${}^{R}\mathbf{K}$. The estimation process is very similar among all these methods and rely on an accurate and automatic corner-extraction algorithm. However, note that Zhang's method cannot take advantage of this automatic algorithm and it required manual user intervention since 3 checkerboards must be simultaneously observed as a calibration scene. Also, these methods require the user to accurately specify the geometric properties of the checkerboard (e.g., size of the squares, the number of squares, etc.).

The major difference among these methods is in the acquisition and treatment of data from the depth sensor. Zhang's method requires the user to select a certain number of points on each checkerboard plane in the depth-map. These points are then converted to 3-D and used to extract the normal to each plane. Herrera's method is similar to Zhang's method, however, Herrera's method relies on the disparity image which is used to estimate a depth-map. Both the ROS toolbox and CalTechTbx require the infrared (IR) image from the Kinect, and then detect corner features in them as done for the RGB. Note that, in order to ensure the checkerboard is clearly visible in the IR image, an external (halogen) light must be used to enhance the contrast of the image and the Kinect's IR projector must be covered. Also note that these methods are mostly designed around the Kinect, since the IR image might not be available from other depth sensors (e.g., ToF cameras). Furthermore, Kinect does not allow simultaneous acquisition of IR and RGB; as a result, their calibration phase is not real time, and it requires slow asynchronous data collection. All these drawbacks make these methods clearly not usable by non-expert users.

Discussion: For Zhang's method in [16], we considered a set of 14 RGB-D image pairs of 3 rigidly-attached checkerboard panels (see Fig. 6(a)). The method requires an initial guess for ${}^{D}\mathbf{K}$; for that, we used the preset values in [16]. For Herrera's method in [17], we captured a set of 60 RGB-Disparity image pairs

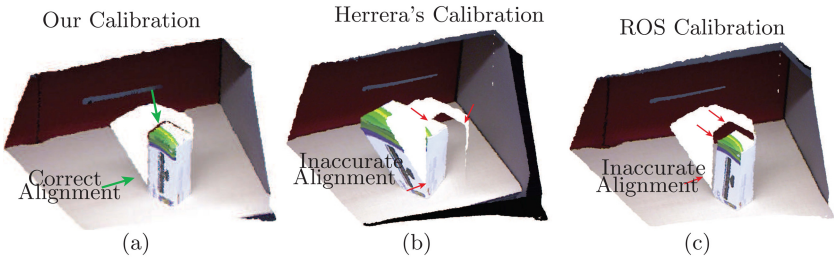


Fig. 7. *Experiment Results*: Each image is a colorized 3-D point cloud based on each algorithm's estimated calibration parameters. *(a)* Our Calibration. *(b)* Herrera's Calibration. *(c)* ROS Calibration.

of a single checkerboard (see Fig. 6(b)). For both the CalTechTbx in [28] and ROS's method [19], we captured a set of 60 RGB-IR image pairs (see Fig. 6(c)). Note that while the number of images used by each method are different (e.g., our method uses 130 image pairs while Zhang's method uses 14 image pairs), the actual *number of measurements* is lower in our case. In fact, our method considers each image of a sphere as 1 measurement for a total of 130 measurements. Meanwhile Zhang's method considers 3 checkerboard panels per view with a total number of measurements to be ~ 2000 over the whole set of images.

In the first experiment, we qualitatively compared the calibration parameters obtained by our method with those estimated by other approaches. We do this by visualizing a textured point cloud corresponding to the observation of a box on top of a desk. This visualization nicely encompasses the accuracy over *all* the parameters since, the depth image is first mapped to 3-D using ${}^{D}\hat{\mathbf{K}}$, and then each 3-D point is expressed in the RGB frame by means of ${}^{R}_{D}\hat{\mathbf{R}}$ and ${}^{R}_{D}\hat{\mathbf{t}}$. Finally, a color is assigned by projecting this map onto the RGB image with ${}^{R}\hat{\mathbf{K}}$. Fig. 7 illustrates the results of this process: the accuracy of our calibration method (cf. Fig. 7(a)) is indeed much higher than Herrera's and ROS (cf. Figs. 7(b)-7(c). Zhang's results have not been reported because its large errors in the estimates (e.g., over 1 m. along the y direction) result in a meaningless reprojected map. Note that, this qualitative experiment focuses on the effect of the calibration algorithms; this is why we exclude the manufacturer's calibration (cf. Fig. 1(a)).

A quantitative assessment of the performance of our method is reported in Table 1, which shows the ${}^{R}_{D}\hat{\mathbf{R}}$, ${}^{R}_{D}\hat{\mathbf{t}}$, and ${}^{D}\hat{\mathbf{K}}$ produced by each method. The first three columns represent the ${}^{R}_{D}\hat{\mathbf{R}}$ expressed as roll ($\hat{\theta}_r$), pitch ($\hat{\theta}_p$), and yaw ($\hat{\theta}_y$) angles in degrees. The angles calculated from our method and Zhang's method are comparable with those of the CalTechTbx. Herrera's method and the ROS method seem to have a 2 degree bias along the yaw angle with respect to CalTechTbx. The fourth through the sixth columns report the three coordinates of ${}^{R}_{D}\hat{\mathbf{t}}$ expressed in meters. Our method and Herrera's method are similar to the CalTechTbx. Zhang's method, as aforementioned, produces a wrong estimate of ${}^{R}_{D}\hat{\mathbf{t}}$ with 1 meter along the y axis. The ROS method is biased along the y direction with 2.8 cm since only 1 RGB-IR image pair can be used by this toolbox to estimate the extrinsic parameters. The last four columns of Table 1 report ${}^{D}\hat{\mathbf{K}}$ expressed as $\hat{f}_u$ and $\hat{f}_v$ for the focal length and $\hat{u}_0$ and $\hat{v}_0$ for the principal point.

Table 1. *Experimental Results:* ${}^{R}_{D}\hat{\mathbf{R}}$, ${}^{R}_{D}\hat{\mathbf{t}}$, and ${}^{D}\hat{\mathbf{K}}$ for each method

	$\hat{\theta}_r$ [deg.]	$\hat{\theta}_p$ [deg.]	$\hat{\theta}_y$ [deg.]	$\hat{t}_x$ [m.]	$\hat{t}_y$ [m.]	$\hat{t}_z$ [m.]	$\hat{f}_u^D$ [pix.]	$\hat{f}_u^D$ [pix.]	$\hat{u}_0^D$ [pix.]	$\hat{v}_0^D$ [pix.]
CalTechTbx	-0.68	-0.69	0.25	-0.023	-0.000	0.001	585.9	587.0	314.0	253.0
Our	-0.04	-0.12	0.11	-0.025	-0.000	-0.000	555.4	559.0	312.0	250.0
Zhang	-0.11	0.11	0.01	-0.549	1.013	0.010	575.0	575.0	320.0	240.0
Herrera	0.56	0.82	2.13	-0.019	-0.006	0.004	577.4	523.0	317.0	222.8
ROS	0.09	0.00	2.72	-0.029	0.028	0.001	574.6	574.7	329.5	240.2

Zhang's method used a fixed preset ${}^{D}\hat{\mathbf{K}}$, which is not optimized during the iterations. Also observe that $\hat{f}_u^D$ and $\hat{f}_u^D$ are within 10 pixels for Herrera's method and 30 pixels for our method with respect to the CalTechTbx. The estimate for the distance between the RGB and depth sensors obtained by our method is the closest one to the ground truth, as well as to the ~ 2.5 cm that can be measured on the Kinect.

5 Conclusions

In this work, we have presented a novel and easy-to-use calibration method for RGB-D cameras, which utilizes the image projection of a sphere (observed from multiple viewpoints) as the only calibration object. Our method relies on a novel closed-form solution for the calibration of the depth sensor, which is used to accurately initialize a nonlinear minimization strategy providing a refined estimate of *all* the calibration parameters of the RGB-D camera (including lens distortion). The proposed algorithm is *practical*, since it needs no user intervention in extracting the image features when compared to existing calibration methods, and is *robust* to outliers. Extensive simulation and real-world experiments validate our algorithm. A MATLAB toolbox implementing our method has been developed, and it has been made freely available for the research community.

References

1. Microsoft® Kinect Camera (2011) (Web): `http://www.xbox.com/en-US/KINECT`.
2. Konolige, K.: Projected texture stereo. In: Proc. IEEE Int. Conf. Robot. Automat. In: Proc. IEEE Int. Conf. Robot. Automat, Anchorage, Alaska, U.S, pp. 148–155 (May 2010)
3. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from single depth images. In: Int. Conf. Vis. Pattern Rec. (2011)
4. Newcombe, R., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A., Kohli, P., Shotton, J., Hodges, S., Fitzgibbon, A.: KinectFusion: Real-time dense surface mapping and tracking. In: 10th IEEE Intl. Sym. on Mixed and Aug. Real., pp. 127–136 (2011)

5. Morbidi, F., Ray, C., Mariottini, G.L.: Cooperative active target tracking for heterogeneous robots with application to gait monitoring. In: Proc. IEEE Int. Conf. Intell. Rob. Sys., pp. 3608–3613 (September 2011)
6. Gabel, M., Gilad-Bachrach, R., Renshaw, E., Schuster, A.: Full body gait analysis with kinect. In: Intl. Conf. of the IEEE Eng. in Med. and Bio. Soc. (August 2012)
7. Cai, Q., Gallup, D., Zhang, C., Zhang, Z.: 3D Deformable Face Tracking with a Commodity Depth Camera. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part III. LNCS, vol. 6313, pp. 229–242. Springer, Heidelberg (2010)
8. Frank, B., Schmedding, R., Stachniss, C., Teschner, M., Burgard, W.: Learning Deformable Object Models for Mobile Robot Navigation using Depth Cameras and a Manipulation Robot. In: Proc. Robotics: Science and Systems VI's Workshop on RGB-D: Advanced Reasoning with Depth Cameras (June 2010)
9. Lai, K., Bo, L., Ren, X., Fox, D.: A Large-Scale Hierarchical Multi-View RGB-D Object Dataset. In: Proc. IEEE Int. Conf. Robot. Automat, pp. 1817–1824 (May 2011)
10. Ramey, A., Gonzalez-Pacheco, V., Salichs, M.A.: Integration of a low-cost RGB-D sensor in a social robot for gesture recognition. In: Proc. 6th Int. Conf. Human-robot Inter., Lausanne, Switzerland, pp. 229–230 (March 2011)
11. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge Univ. Press (2003)
12. Burrus, N.: Kinect calibration (Web): `http://nicolas.burrus.name/index.php/Research/KinectCalibration`
13. Mihelich, P., Konolige, K.: Technical description of kinect calibration (Web): `http://www.ros.org/wiki/kinect_calibration/technical`
14. Jung, J., Jeong, Y., Park, J., Ha, H., Kim, D.J., Kweon, I.: A Novel 2.5D Pattern for Extrinsic Calibration of ToF and Camera Fusion System. In: Proc. IEEE/RSJ Intl. Conf. on Intel. Rob. Syst., pp. 3290–3296 (September 2011)
15. Smisek, J., Jancosek, M., Pajdla, T.: 3D with kinect. In: IEEE Intl. Conf. on Computer Vision Workshops, pp. 1154–1160 (November 2011)
16. Zhang, C., Zhang, Z.: Calibration between Depth and Color Sensors for Commodity Depth Cameras. In: Intl. Workshop on Hot Topics in 3D, in Conjunction with ICME (July 2011)
17. Herrera, C., Kannala, J., Heikkilä, J.: Joint depth and color camera calibration with distortion correction. IEEE Trans. Pattern Anal. 34(10), 2058–2064 (2012)
18. Khoshelham, K., Elberink, S.O.: Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications. Sensors 12(2), 1437–1454 (2012)
19. Mihelich, P.: ROS openni-launch package for Intrin. and Extrin. Kinect Calib (2013), (Web): `http://www.ros.org/wiki/openni_launch/Tutorials/`
20. Fischler, M.A., Bolles, R.C.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Commun. ACM 24, 381–395 (1981)
21. Cipolla, R., Giblin, P.: Visual motion of curves and surfaces. Cambridge University Press, New York (2000)
22. Canny, J.: A Computational Approach to Edge Detection. IEEE Trans. Pattern Anal. 8(6), 679–698 (1986)
23. Ballard, D.H.: Generalizing the Hough Transform to Detect Arbitrary Shapes. Pattern Recog. 13(2), 111–122 (1981)
24. Fitzgibbon, A., Pilu, M., Fisher, R.B.: Direct least square fitting of ellipses. IEEE Trans. Pattern Anal. 21(5), 476–480 (1999)

25. Halıř, R., Flusser, J.: Numerically stable direct least squares fitting of ellipses. In: Proc. 6th Int. Conf. Cen. Eur. on Com. Graph. Vis., vol. 21(5), pp. 125–132 (Febraury 1998)
26. Wong, K., Zhang, G., Chen, Z.: A Stratified Approach for Camera Calibration Using Spheres. IEEE Trans. on Img. Proc. 20(2), 305–316 (2011)
27. Staranowicz, A., Mariottini, G.L.: A comparative study of calibration methods for kinect-style cameras. In: Proc. 5th Intl. Conf on PETRAE, pp. 49:1–49:4 (2012)
28. Camera Calibration Toolbox for Matlab (2010) (Web): http://www.vision.caltech.edu/bouguetj/calib_doc/