

Calibration Wizard: A Guidance System for Camera Calibration Based on Modelling Geometric and Corner Uncertainty

Songyou Peng, Peter Sturm

▶ **To cite this version:**

Songyou Peng, Peter Sturm. Calibration Wizard: A Guidance System for Camera Calibration Based on Modelling Geometric and Corner Uncertainty. ICCV 2019 - International Conference on Computer Vision, Oct 2019, Seoul, South Korea. pp.1497-1505, 10.1109/ICCV.2019.00158 . hal-02368999

HAL Id: hal-02368999
https://inria.hal.science/hal-02368999

Submitted on 18 Nov 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Calibration Wizard: A Guidance System for Camera Calibration Based on Modelling Geometric and Corner Uncertainty

Songyou Peng*
ETH Zurich
songyou.peng@inf.ethz.ch

Peter Sturm
INRIA Grenoble – Rhône-Alpes
peter.sturm@inria.fr

Abstract

It is well known that the accuracy of a calibration depends strongly on the choice of camera poses from which images of a calibration object are acquired. We present a system – Calibration Wizard – that interactively guides a user towards taking optimal calibration images. For each new image to be taken, the system computes, from all previously acquired images, the pose that leads to the globally maximum reduction of expected uncertainty on intrinsic parameters and then guides the user towards that pose. We also show how to incorporate uncertainty in corner point position in a novel principled manner, for both, calibration and computation of the next best pose. Synthetic and real-world experiments are performed to demonstrate the effectiveness of Calibration Wizard.

1. Introduction

Camera calibration is a prerequisite to many methods and applications in computer vision and photogrammetry, in particular for most problems where 3D reconstruction or motion estimation is required. In this paper we adopt the popular usage of planar calibration objects, as introduced[1] by [17, 23] and made available through OpenCV [2] and Matlab [1], even though our approach can be directly applied to 3D calibration objects too.

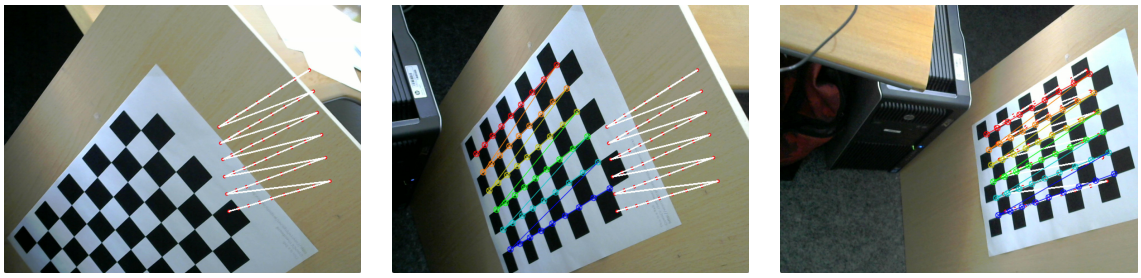
It is well known that when using planar calibration targets, the accuracy of the resulting calibration depends strongly on the poses used to acquire images. From the theoretical study of degenerate sets of camera poses in [17, 23], it follows for example intuitively that it is important to vary the orientation (rotation) of the camera as much as possible during the acquisition process. It is also widely known to practitioners that a satisfactory calibration requires images such that the target successively covers the entire image area, otherwise the estimation of radial distortion and other parameters usually remains suboptimal. We have observed that inexperienced users usually do not take calibration images that lead to a sufficiently accurate calibration.

Several efforts have been done in the past to guide users in placing the camera. In photogrammetry for instance, the so-called network design problem was addressed, through an off-line process: how to place a given number of cameras such as to obtain an as accurate as possible 3D reconstruction of an object of assumed proportions [10, 11]. Optimal camera poses for camera calibration have been computed in [14], however only for constrained camera motions and especially, only for the linear approach of [23], whereas we consider the non-linear optimization for calibration. Also, these poses are difficult to realize, even for expert users. The ROS [12] monocular camera calibration toolbox provides text instructions so that users can move the target accordingly. More recently, some cameras, such as the ZED stereo system from StereoLabs, come with software that interactively guides the user to good poses during the calibration process; these poses are however pre-computed for the particular stereo system and this software cannot be used to calibrate other systems, especially monocular cameras.

In this paper we propose a system that guides the user through a simple graphical user interface (GUI) in order to move the camera to poses that are optimal for calibrating a camera. Optimality is considered for the bundle adjustment type non-linear optimization formulation for calibration. For each new image to be acquired, the system computes the optimal pose, i.e. which adds most new information on intrinsic parameters, in addition to that provided by the already acquired images. The most closely related works we are aware of are [13, 15]. They also suggest next best poses to the user. However, unlike ours, they are both strategy-based methods, where suggestions are selected from a fixed dataset of pre-defined poses, which may not be enough for various camera models or calibration targets. In our approach, each new suggested pose results from a global optimization step. Furthermore, we propose a novel method for incorporating the uncertainty of corner point po-

*Most work was done while he was an intern at INRIA.

[1]Planar targets were used before, but essentially in combination with motion stages, in order to effectively generate 3D targets [18, 7, 20, 9]



(a) (b) (c)

Figure 1. Illustration of the guidance process. (a) Calibration Wizard proposes the next best pose based on the previous calibration results. (b) The camera should be moved towards the proposed target pose. (c) When the camera is close enough to the suggested pose, the system acquires an image and then proposes a next pose. Demo code: https://github.com/pengsongyou/CalibrationWizard.

sitions rigorously throughout the entire pipeline, for calibration but also next best pose computation. Our approach is not specific to any camera model; in principle, any monocular camera model can be plugged into it, although tests with very wide field of view cameras need to be done.

The paper is organized as follows. Section 2 describes the theory and mathematical details of Calibration Wizard. Section 3 shows how to incorporate corner point uncertainty in the process. Experiments are reported in Section 4, followed by conclusions in Section 5.

2. Methodology

Our goal is to provide an interactive guidance for the acquisition of good images for camera calibration. An initial calibration is carried out from a few (typically 3) freely taken images of a calibration target. The system then guides the user to successive next best poses, through a simple GUI, cf. Fig 1. The underlying computations are explained in the following subsections.

2.1. Calibration formulation

Let the camera be modeled by two **local projection functions** $x = q_x(\Theta, S), y = q_y(\Theta, S)$ that map a 3D point S given in the local camera coordinate system, to the image coordinates x and y. These functions depend on intrinsic parameters Θ (assumed constant across all images). For example, the standard 3-parameter pinhole model consisting of a focal length f and principal point (u, v), i.e. with $\Theta = (f, u, v)^\top$, has the following local projection functions (a full model with radial distortion is handled in supplementary material, section 3):

$$q_x(\Theta, S) = u + f\frac{S_1}{S_3} \qquad q_y(\Theta, S) = v + f\frac{S_2}{S_3} \tag{1}$$

Let camera pose be given by a 6-vector Π of extrinsic parameters. We use the representation $\Pi = (t_1, t_2, t_3, \alpha, \beta, \gamma)^\top$, where $t = (t_1, t_2, t_3)^\top$ is a translation vector and the 3 angles define a rotation matrix as the product of rotations about the 3 coordinate axes: $R = R_z(\gamma)R_y(\beta)R_x(\alpha)$. R and t map 3D points Q from the world coordinate system to the local camera coordinate system according to:

$$S = RQ + t \tag{2}$$

Other parameterization may be used too, e.g. quaternions.

A camera with pose Π is thus described by two **global projection functions** p_x and p_y:

$$p_x(\Theta, \Pi, Q) = q_x(\Theta, RQ + t) = q_x(\Theta, S) \tag{3}$$
$$p_y(\Theta, \Pi, Q) = q_y(\Theta, RQ + t) = q_y(\Theta, S) \tag{4}$$

Since a planar calibration target is used, the 3D points Q are pre-defined and their corresponding Z coordinates Q_3 are set to 0. We now consider m images of a target consisting of n calibration points. Inputs to the calibration are thus the image points (x_{ij}, y_{ij}) for $i = 1 \cdots m$ and $j = 1 \cdots n$, which are detected by any corner detector, e.g. the OpenCV `findChessboardCorners` function [2]. For ease of explanation, we assume here that all points are visible in all images, although this is not required in the implementation.

Optimal calibration requires a non-linear simultaneous optimization of all intrinsic and extrinsic parameters (bundle adjustment). This comes down to the minimization of the geometric reprojection error [6]:

$$\min_{\Theta, \{\Pi_i\}} \sum_{i,j} (x_{ij} - p_x(\Theta, \Pi_i, Q_j))^2 + (y_{ij} - p_y(\Theta, \Pi_i, Q_j))^2 \tag{5}$$

Usually, local non-linear least square optimizers are used, such as Levenberg-Marquardt. Our system is independent of the optimizer used; all it requires is the computation, at the found solution, of the partial derivatives of (5), see next.

2.2. Computation of next best pose

We suppose that we have already acquired m images and estimated intrinsic parameters and poses from these, by solving (5). The goal now is to compute the next best pose; the objective is to reduce, as much as possible, the expected uncertainty on the estimated intrinsic parameters.

Let us consider the Jacobian matrix J of the cost function (5), evaluated at the estimated parameters. J contains the partial derivatives of the cost function's residuals, i.e. of terms $\hat{x}_{ij} = x_{ij} - p_x(\Theta, \Pi_i, Q_j)$ and $\hat{y}_{ij} = y_{ij} - p_y(\Theta, \Pi_i, Q_j)$. J contains one row per residual. Its columns are usually arranged in groups, such that the first group contains the partial derivatives with respect to the intrinsic parameters Θ, and subsequent groups of columns, the derivatives relative to extrinsic parameters of the successive images. The highly sparse form of J is thus as follows (we assume here that there are k intrinsic parameters):

$$J = \begin{pmatrix} A_1 & B_1 & 0 & \cdots & 0 \\ A_2 & 0 & B_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A_m & 0 & 0 & \cdots & B_m \end{pmatrix} \tag{6}$$

$$A_i = \begin{pmatrix} \frac{\partial \hat{x}_{i1}}{\Theta_1} & \cdots & \frac{\partial \hat{x}_{i1}}{\Theta_k} \\ \frac{\partial \hat{y}_{i1}}{\Theta_1} & \cdots & \frac{\partial \hat{y}_{i1}}{\Theta_k} \\ \vdots & \ddots & \vdots \\ \frac{\partial \hat{x}_{in}}{\Theta_1} & \cdots & \frac{\partial \hat{x}_{in}}{\Theta_k} \\ \frac{\partial \hat{y}_{in}}{\Theta_1} & \cdots & \frac{\partial \hat{y}_{in}}{\Theta_k} \end{pmatrix} \quad B_i = \begin{pmatrix} \frac{\partial \hat{x}_{i1}}{\Pi_{i,1}} & \cdots & \frac{\partial \hat{x}_{i1}}{\Pi_{i,6}} \\ \frac{\partial \hat{y}_{i1}}{\Pi_{i,1}} & \cdots & \frac{\partial \hat{y}_{i1}}{\Pi_{i,6}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \hat{x}_{in}}{\Pi_{i,1}} & \cdots & \frac{\partial \hat{x}_{in}}{\Pi_{i,6}} \\ \frac{\partial \hat{y}_{in}}{\Pi_{i,1}} & \cdots & \frac{\partial \hat{y}_{in}}{\Pi_{i,6}} \end{pmatrix}$$

where A_i are matrices of size $2n \times k$, containing the partial derivatives of residuals with respect to k intrinsic parameters, whereas B_i are matrices of size $2n \times 6$, containing the partial derivatives with respect to extrinsic parameters. Now, the so-called information matrix is computed as $J^\top J$:

$$J^\top J = \begin{pmatrix} \sum_i A_i^\top A_i & A_1^\top B_1 & A_2^\top B_2 & \cdots & A_m^\top B_m \\ B_1^\top A_1 & B_1^\top B_1 & 0 & \cdots & 0 \\ B_2^\top A_2 & 0 & B_2^\top B_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ B_m^\top A_m & 0 & 0 & \cdots & B_m^\top B_m \end{pmatrix} \tag{7}$$

Like J, $J^\top J$ is highly sparse. Importantly, its inverse $(J^\top J)^{-1}$ provides an estimation of the covariance matrix of the estimated intrinsic and extrinsic parameters.

For camera calibration, we are only interested in the covariance matrix of the intrinsic parameters, i.e. the upper-left $k \times k$ sub-matrix of $(J^\top J)^{-1}$. Due to the special structure of $(J^\top J)^{-1}$, it can be computed efficiently. Let

$$\begin{aligned} U &= \sum_i A_i^\top A_i \\ V &= \text{diag}\left(B_1^\top B_1, \cdots, B_m^\top B_m\right) \\ W &= \begin{pmatrix} A_1^\top B_1 & \cdots & A_m^\top B_m \end{pmatrix} \end{aligned}$$

Then,

$$J^\top J = \begin{pmatrix} U & W \\ W^\top & V \end{pmatrix} \tag{8}$$

and as described in [6], the upper-left sub-matrix of $(J^\top J)^{-1}$ is given by $\Sigma = (U - WV^{-1}W^\top)^{-1}$, i.e. the inverse of a $k \times k$ symmetric matrix.

Let us now return to our goal, determining the next best pose Π_{m+1}. The outline of how to achieve this is as follows. We extend the Jacobian matrix in Eq. (6) with a part corresponding to an additional image, whose pose is parameterized by Π_{m+1}. The coefficients in A_{m+1} and B_{m+1}, are thus functions of Π_{m+1}. Naturally, Π_{m+1} is also implicitly embedded in the intrinsic parameter's covariance matrix associated with this extended system. To reduce the uncertainty of the calibration, we wish to determine Π_{m+1} that makes Σ as "small" as possible. Inspired by [4], we choose to minimize the trace of this $k \times k$ matrix. Since we wish to compute the next best pose within the entire 3D working space, we use a global optimization method. Our experiments suggest that simulated annealing [19] or ISRES [16] work well for this small optimization problem [2]. Especially the latter works in interactive time.

Note that the computation of the partial derivatives used to build the A_i and B_i matrices can be done very efficiently using the chain rule. Further, computation of Σ for different trials of Π_{m+1} can also be done highly efficiently by appropriate pre-computations of the parts of matrices U and $WV^{-1}W^\top$ that do not depend on Π_{m+1}. See more details in the supplementary material, section 2.

3. Taking into Account the Uncertainty of Corner Points

So far, we have not used information on the uncertainty of corner point positions: in Eq. (5), all residuals have the same weight (equal to 1). Ideally, in any geometric computer vision formulation, one should incorporate estimates of uncertainty when available. In the following, we explain this for our problem, from two aspects: first for the actual calibration, i.e. the parameter estimation in Eq. (5). Second, more originally, for computing the next best pose.

3.1. Corner Uncertainty in Calibration

Consider a corner point extracted in an image; the uncertainty of its position can be estimated by computing the autocorrelation matrix C for a window of a given size around the point (see for instance [5]). Concretely, C is an estimate of the inverse of the covariance matrix of the corner position. Now, let C_{ij} be the autocorrelation matrix for the jth corner in the ith image. The C_{ij} can be incorporated in the calibration process by inserting the block-diagonal matrix composed by them, in the computation of the information

[2] We did not consider the stopping criterion in the current version, but one could simply stop our method when the relative residual of the trace of the covariance matrix mentioned above is smaller than a threshold.

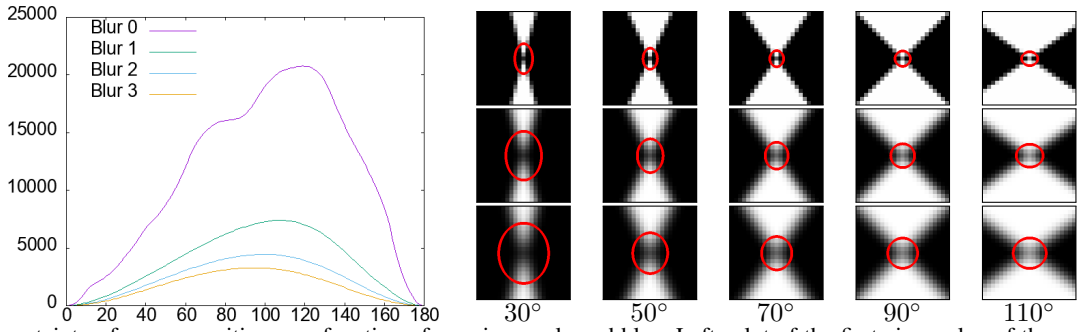


Figure 2. Uncertainty of corner position as a function of opening angle and blur. Left: plot of the first eigenvalue of the autocorrelation matrix, over opening angle and for different blur levels (Gaussian blur for $\sigma = 0, 1, 2, 3$). Right: corners for different opening angles and blur levels ($\sigma = 0, 1, 2$) and computed 95% confidence level uncertainty ellipses (enlarged $10\times$ for display).

matrix of Eq. (7):

$$H = J^\top \text{diag}(C_{11}, C_{12}, \cdots, C_{1n}, C_{21}, \cdots, C_{mn}) J \quad (9)$$

This uncertainty-corrected information matrix can then be used by Gauss-Newton or Levenberg-Marquardt type optimizers for estimating the calibration (for other optimizers, the autocorrelation matrix may have to be used differently) as well as for the quantification of the uncertainty of the computed intrinsic parameters.

3.2. Next Best Pose Computation

The second usage of corner uncertainty concerns the computation of the next best pose. In particular, for each hypothetical next pose that we examine, we can project the points of the calibration target to the image plane, using this hypothetical pose and the current estimates of intrinsic parameters. This gives the expected positions of corner points, if an actual image were to be taken from that pose. Importantly, what we would like to compute in addition, is the expected uncertainty of corner extraction, i.e. the uncertainty of the corner positions extracted in the expected image. Or, equivalently, the autocorrelation matrices computed from the expected pixel neighborhoods around these corner points. If we are able to do so, we can plug these into the estimation of the next best pose, by inserting the expected autocorrelation matrices in Eq. (7) the same way as done in Eq. (9).

Before explaining how to estimate expected autocorrelation matrices, we describe the benefits of this approach. Indeed, we have found that without doing so, the next best pose is sometimes rather extreme, with a strong grazing viewing angle relative to the calibration target. This makes sense in terms of pure geometric information contributed by such a pose, but is not appropriate in practice, since with extreme viewing angles image corners are highly elongated: they may be difficult to extract in the actual image and also, their uncertainty is very large in one direction. While this may be compensated by using images acquired from poses with approximately perpendicular viewing directions one from another, it is desirable and indeed more principled to fully integrate corner uncertainty right from the start in the computation of the next best pose.

Let us now explain how to compute expected autocorrelation matrices of corner points, for a hypothetical next pose. This is based on a simple reasoning. The overall shape of an image corner (in our case, a crossing in a checkerboard pattern), is entirely represented by the "opening angle" of the corner. What we do is to precompute autocorrelation matrices for synthetic corners for the entire range of opening angles, cf. Fig. 2: the top row on the right shows ideal corners generated by discretizing continuous black-and-white corners, for a few different opening angles. For each of them, the autocorrelation matrix (cf. [5]) was computed; as mentioned, its inverse is an estimate of the corner position's covariance matrix. The figure shows the plots of 95% uncertainty ellipses derived from these covariance matrices (enlarged 10 times, for better visibility). Such ideal corners are of course not realistic, so we repeat the same process for images blurred by Gaussian kernels of different σ (2nd and 3rd rows of the figure). Naturally, blurrier corner images lead to smaller autocorrelation matrices and larger uncertainty ellipses.

One may note several things. First, between $30°$ and $90°$ opening angles, the largest uncertainty differs by a factor of about 2, see the uncertainty ellipses in Fig. 2. Second, intuitively, the uncertainty ellipse of a corner with opening angle α is the same as with opening angle $180° - \alpha$, but turned by $90°$ (cf. the 3rd and 5th columns in Fig. 2, for $70°$ and $110°$). Hence, the eigenvalues of the autocorrelation matrix of a corner with opening angle α, are the same as that for $180° - \alpha$, but they are "swapped" (associated with the respective opposite eigenvector).

The left part of Fig. 2 shows plots of the first eigenvalue (associated with eigenvector $(0, 1)$) of the autocorrelation matrix C as a function of opening angle. Due to the above observation, the second eigenvalue (associated with eigenvector $(1, 0)$) associated with opening angle α is simply given by the first eigenvalue associated with $180° - \alpha$. The

graphs on the left of the figure confirm that increasing blur decreases the autocorrelation matrices eigenvalues. Let us note that we also simulated Gaussian pixel noise on the corner images; even for larger than realistic noise levels, the impact on the results shown in Fig. 2, was negligible.

Let us finally explain how to use these results. First, we determine the average blur level in the already acquired images, from the strength of image gradients across edges, and then select the graph in Fig. 2 associated with the closest simulated blur (for more precision, one could also compute the graph for the actual detected blur level). Let the function represented by the graph be $f(\alpha)$ – one can represent it as a lookup table or fit a polynomial to the data of the graph (we did the latter). This allows to compute the diagonal coefficients of the autocorrelation matrix from the opening angle, as $f(\alpha)$ and $f(180° - \alpha)$. Second, so far we have only considered axis-aligned corners. If we now consider a corner with opening angle α, but that is rotated by an angle β, then its autocorrelation matrix is nothing else than:

$$C = \begin{pmatrix} \cos\beta & -\sin\beta \\ \sin\beta & \cos\beta \end{pmatrix} \begin{pmatrix} f(\alpha) & 0 \\ 0 & f(180° - \alpha) \end{pmatrix} \begin{pmatrix} \cos\beta & \sin\beta \\ -\sin\beta & \cos\beta \end{pmatrix} \tag{10}$$

We now have all that is needed to incorporate corner uncertainty in next best pose computation. For each hypothetical pose we project, as shown above, all calibration points. For each point (corner), using its neighbors, we can compute the opening angle α and rotation angle β and thus, the expected autocorrelation matrix C. It can then be inserted in the computation of the information matrix, like in Eq. (9).

The effect of this approach on proposing the next best pose is to strike a balance between maximizing pure geometric "strength" of a pose (often achieved by strong tilting of the calibration pattern) and maximizing corner extraction accuracy (in fronto-parallel poses, corners are overall closest to exhibiting right angles, i.e. where their autocorrelation matrices have maximal trace).

3.3. Possible Extensions

So far we have described the basic idea for incorporating corner uncertainty. The following extensions may be applied; we plan this for future work. The values plotted in Fig. 2 are obtained for corners exhibiting the full range of 256 greylevels (black to white). In real images, the range is of course smaller. If the difference between largest and smallest greylevels is x, then the plotted values (left of Fig. 2) are divided by $255^2/x^2$ (easy to prove but not shown due to lack of space). In turn, the uncertainty ellipses are scaled up by a factor of $255/x$. This should be taken into account when predicting auto-correlation matrices for the next pose. The range of greylevels depends on various factors, such as distance to the camera and lighting conditions. One can learn the relationship between pose and greylevel range for a given calibration setup as part of the calibration process and use it to predict the next best pose.

Similarly, one might predict expected blur per corner, based on a learned relationship between blur and distance to camera, e.g. by inferring the camera's depth of field during calibration. We observed that within the depth of field, image blur is linearly related to the distance to the camera.

Using these observations should allow to further improve the next best pose proposal, by achieving an even better compromise between geometric information of the pose and accuracy of image processing (here, corner extraction), both of which influence calibration accuracy.

4. Results and Evaluation

Synthetic and real-world experiments are performed here to evaluate the effectiveness of our Calibration Wizard. Note that in the optimization process, we ensure that all corner points should be within the image plane, otherwise the optimization loss is set to an extremely large value.

4.1. Synthetic evaluations

To assess the proposed system, we simulate the process of camera calibration with pre-defined intrinsic parameters, with Matlab. Here we first briefly introduce the procedure of producing random checkerboard poses.

Data preparation. First, 9×6 target points are defined, with Z components set to 0. Next, the 3D position of the virtual camera is created, with X and Y coordinates proportional to Z within a plausible range. Then the camera is first oriented such that its optical axis goes through the center of the target and finally, rotations about the three local coordinate axes by random angles between $-15°$ and $15°$ are applied. Now, from the position, rotation matrix and the given intrinsic parameters, we can project the 3D target points to the image, and finally add zero-mean Gaussian noise with the same noise level to them. Moreover, we ensure that all 54 points are located within the field of view of a 640×480 image plane.

Evaluation of accuracy and precision. We primarily compare the calibration accuracy obtained from random images, with that from images acquired as proposed by our system, with and without taking into account the autocorrelation matrix explained in the previous section. To this end, the experimental process is as follows: create 3 initial random images, based on which we have 3 paths to acquire the calibration results:

- Produce many other random images
- Obtain 17 images proposed by the wizard, so $3 + 17 = 20$ images in total
- Obtain 17 wizard images taking the autocorrelation matrix into account

100 trials are performed for each experiment, hence we acquire 100 samples of intrinsic parameters for each. In the

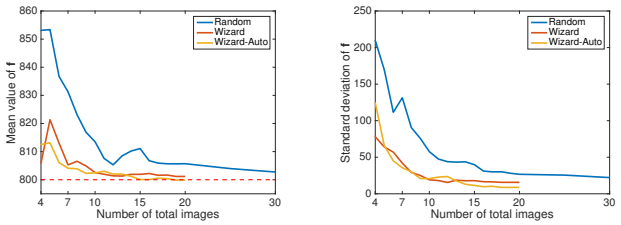


Figure 3. Comparisons of the focal length estimated from three schemes on synthetic data: randomly taken images, calibration wizard and wizard using autocorrelation matrix. $f = 800, (u, v) = (320, 240), k_1 = 0.01, k_2 = 0.1$. Initial calibration was done with 3 random images. Left: Mean values of the estimated focal length, where the red dashed line represents the ground truth $f = 800$. Right: Standard deviations of the estimated focal length. Wizard images provide significantly more accurate and precise calibration results than random ones.

first test, we set $f = 800, (u, v) = (320, 240)$ and radial distortion coefficients $k_1 = 0.01, k_2 = 0.1$. Fig. 3 illustrates the statistical results of the estimated focal length from 100 trials. It can be easily noticed that the focal length acquired using our wizard is not only much closer to the ground truth, but also more concentrated (precise) than the estimation from pure random images. For example, the estimated focal length acquired from only 3 random + 4 wizard images has outperformed the one from 20 random images. Moreover, not shown in the graph: 3 random + 17 wizard images still give higher accuracy than 60 random images, which directly demonstrates the usefulness of our approach.

However, in this experiment, we notice that our system does not show much advantage over the randomly-taken images of the estimated distortion coefficients k_1 and k_2. Thus, a second experiment is performed with larger radial distortion coefficients $k_1 = 0.5$ and $k_2 = 1$, while the focal length and principal point stay the same. Fig. 4 shows the effectiveness of the proposed system, especially with the consideration of autocorrelation matrix for target points. When the radial distortion is large, we notice that not only both distortion coefficients, but also the focal length and principle points (not shown here) estimated from purely random images deviate much from the ground truth, as was also reported in [21]. In contrast, our system still manifests the ability of centering around the ground truth with incomparably low standard deviation. Furthermore, compared to the simple case of the proposed system, both first-order statistics features appear to be most desirable when considering the autocorrelation matrices for the feature points.

Robustness to noise. We are also interested in the performance of our approach with respect to the level of noise added to 2D corner points. In this experiment, we compare 4 different configurations: 20 random images, 40 random images, 3 random + 17 wizard images and 3 random + 17 wizard-Auto images. Zero-mean Gaussian noise with standard deviation of $0.1, 0.2, 0.5, 1$ with respect to 2 pixels has been added to the image points respectively, and the comparisons are shown in Fig. 5. Specifically, it can be distinctly seen from the figure that, even when unrealistically strong noise is added ($\sigma = 2$), both versions of our approach (3 random + 17 wizard images) still provide better accuracy than even 40 random images. More synthetic experiments can be found in the supplementary material.

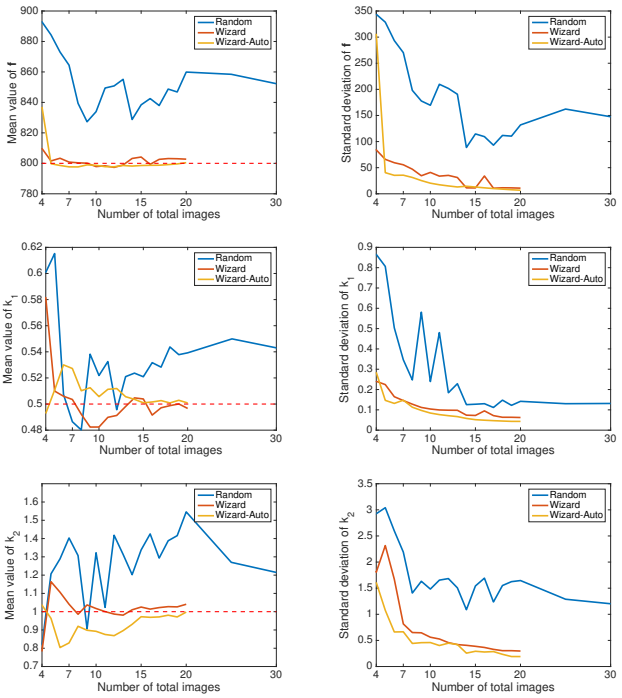


Figure 4. Comparisons of the intrinsic parameters estimated from three schemes on synthetic data: randomly taken images, calibration wizard without and with autocorrelation matrix. $f = 800, (u, v) = (320, 240), k_1 = \mathbf{0.5}, k_2 = \mathbf{1}$. Wizard images achieve superior performance over random images on all intrinsic parameters. Considering the autocorrelation matrices can further provide the most accurate and precise estimation outcomes.

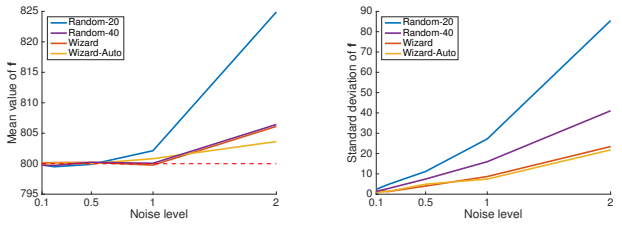


Figure 5. Comparisons among various calibration schemes of the robustness to noise. Zero-mean Gaussian noise with standard deviation of $0.1, 0.2, 0.5, 1$ respectively 2 pixels is added to 2D target points. The focal length estimated from both of our methods with only 20 images, is more accurate (left) and precise (right) than that from 40 random images, especially when the noise level is high.

4.2. Real-world evaluations

Although the performance of the Calibration Wizard has been demonstrated for the synthetic data, we ultimately want to evaluate the effectiveness of its proposed next best pose on real-world examples. We designed two experiments for this purpose where we also compare with calibrations obtained with freely taken images. Evaluating calibration results is difficult since ground truth is not readily available; we thus devised two experiments where calibration quality is assessed through evaluating results of applications – pose estimation and SfM. We used the commonly-used Logitech C270H HD Webcam in our experiments. It has an image size of 640×480 and around 60° field of view. Fig. 7 provides some sample calibration images. One may notice that wizard-suggested images indeed correspond to poses often chosen by experts for stable calibration: large inclination of the target along the viewing direction, targets covering well the field of view and/or reaching the image border.

In the following, we denote "x-free" the calibration results from x images acquired freely by an experienced user using OpenCV, compared to "x-wizard" where guidance was used.

Pose estimation. Similar to the experiment performed in [3], in order to quantitatively evaluate the quality of camera calibration, we design the first real-world experiment where, apart from the images used for calibration, we then also acquire a number of extra checkerboard images which are only used for evaluation, cf. Fig. 6.

First, 4 corner points are utilized to calculate the pose with EPnP [8], given the intrinsic parameters provided by the calibration. Then, since we have assumed the Z components of the target points to be 0 in the world coordinate system, it is straightforward to back-project the remaining 50 points to 3D, onto the target plane, using the calibrated intrinsic parameters and the computed pose (cf. Fig. 6 right). The smaller Euclidean distance between the back-projected and theoretical 3D points, the better the calibration.

There are 80 images in total for testing so we have $50 \times 80 = 4,000$ points for assessment. The mean and standard deviation of the 4,000 distance errors are applied as metric. Table 1 demonstrates that our system, when using only 15 images for calibration, still exceeds the performance of using 50 freely acquired images and exceeds that of using 20 such images by about 5%. This seemingly small improvement may be considered as significant since it may be expected that differences are not large in this experiment. Even with a moderately incorrect calibration, pose estimation from 4 outermost target points will somewhat balance the reconstruction errors for the inner corner points.

We also tested our approach on the FaceTime HD camera of a MacBook Pro. This camera has higher resolution and different field of view compared to other commonly use webcams, so it is a suitable alternative to show the robustness of our method. As shown in Table 2, adding only one or two wizard images can largely reduce the Euclidean distance and outperforms the results from freely taking many more images.

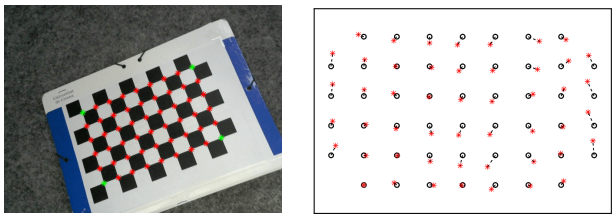


Figure 6. Pose estimation test. Left: checkerboard image where 4 green corner points are used for pose estimation, and the remaining 50 red points for reconstruction. Right: 50 ground-truth points in black and residuals between them and the reconstructed corner points in red (enlarged 50 times for visualization).

Table 1. Pose estimation test with Logitech C270H HD Webcam.

	mean	std		mean	std
3-free	0.856	1.130	3-free + 4-wizard	0.862	1.155
10-free	0.815	1.115	3-free + 6-wizard	0.783	1.092
20-free	0.802	1.115	3-free + 9-wizard	0.788	1.104
50-free	0.789	1.108	**3-free + 12-wizard**	**0.763**	**1.082**

Table 2. Pose estimation test with FaceTime HD camera.

	mean	std		mean	std
3-free	2.503	2.557	3-free + 1-wizard	1.455	1.630
10-free	1.664	1.839	**3-free + 2-wizard**	**1.165**	**1.491**
20-free	1.255	1.606			

Structure from Motion test. In this last experiment, we assess our Calibration Wizard by investigating the quality of 3D reconstruction in a structure-from-motion (SfM) setting. The object to be reconstructed is the backrest of a carved wooden bed, as shown in Fig. 8. We devised a simple but meaningful experiment to evaluate the quality of the calibration, as follows. We captured images from the far left of the object and gradually move to the right side, and then proceed backwards and return to the left, approximately to the starting point. The acquired images are then provided as input to VisualSfM [22]; we added an identical copy of the first image of the sequence, to the end of the sequence, but without "telling" this to the SfM tool and without using a loop detection method during SfM. The purpose of doing so is: if calibration is accurate, the incremental SfM should return poses for the first image and the added identical last image, which are close to one another. Measuring the difference in pose is not sufficient since the global scale of the reconstruction can be arbitrarily chosen by the SfM tool for each trial. So instead, we project all 3D points that were reconstructed on the basis of interest points extracted in the first image, using the pose computed by SfM for the identical last image, and measure the distance between the two sets of 2D points such constructed. This distance is inde-



Figure 7. Sample images used for the calibration in real-world tests. Top row: freely-taken images. Bottom row: wizard guided images.

Table 3. 2D errors of SfM tests under various calibration schemes.

Calibration scheme	mean	std	median
3-free	43.6	11.5	44.3
7-free	30.5	11.7	31.7
20-free	15.7	10.5	16.1
3-free + 2-wizard	17.4	10.8	13.2
3-free + 4-wizard	**14.4**	**9.1**	**10.6**

pendent of the scene scale and is thus a good indicator of the quality of the SfM result which in turn is a good indicator of the quality of the calibration used for SfM.

Note that we only match two consecutive frames instead of full-pairwise matching within the given sequence. In this case, 2D errors are accumulated so the reconstruction results highlight the calibration accuracy more strongly.

The experiment is described as follows. We first obtain the 5-parameter calibration result (including two radial distortion coefficients), from 3 freely acquired images ("3-free"). Then, on the one hand, another 17 images are taken, from which the intrinsic parameters of "7-free" and "20-free" are obtained. On the other hand, we take another 4 sequential images proposed by the Calibration Wizard, where we get the intrinsic parameters of "3-free + 2-wizard" and "3-free + 4-wizard". And now, we load VisualSfM with intrinsic parameters of these five configurations respectively, along with the backrest sequence taken by the same camera. It is worth mentioning that we conduct five trials of VisualSfM for each configuration in order to lessen the influence of the stochastic nature of the SfM algorithm.

Results are listed in Table 3, where we evaluate the 2D errors across all 5 trials. Some observations can be made: with only 5 images in total, "3-free + 2-wizard" has already provided an accuracy competitive to 20 freely-taken images. Both "7-free" and "3-free + 4-wizard" use 7 images for calibration, but it can be clearly noticed that the latter one has far lower errors in all aspects. It is reasonable to conclude that our method notably improves the quality of calibration and 3D reconstruction with a considerably small number of calibration images.



Figure 8. Structure from motion test. Top: Panorama stitched by Hugin (http://hugin.sourceforge.net), showing the test scene. Bottom: Result of applying VisualSfM [22] to build a 3D model. We started capturing images from the left and moved clockwise, finally came back approximately to the starting point.

Finally, it is worth mentioning that all real-world experiments were performed with a 2.7 GHz Intel i5 CPU (no GPU used). To compute the next best pose with a 9×6 target, our un-optimized C++ code took about $0.4s$ for 3 target images and $1.5s$ for 15 images (increasing roughly linearly per image), but we found that 10 images are usually sufficient for a good calibration.

5. Conclusions

Calibration Wizard is a novel approach which can guide any user through the calibration process. We have shown that accurate intrinsic parameters can be obtained from only a small number of images suggested by this out-of-the-box system. Some ideas for future work were already mentioned in section 3.3. We also plan to apply the approach to very wide field of view cameras such as fisheyes.

References

[1] J.-Y. Bouguet. Matlab camera calibration toolbox. *Caltech Technical Report*, 2000. 1

[2] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 1, 2

[3] H. Ha, M. Perdoch, H. Alismail, I. So Kweon, and Y. Sheikh. Deltille grids for geometric camera calibration. In *ICCV*, pages 5344–5352, 2017. 7

[4] S. Haner. *View Planning and Refractive Modeling for Structure and Motion*. PhD thesis, Lund University, 2015. 3

[5] C. Harris and M. Stephens. A combined corner and edge detector. In *Fourth Alvey Vision Conference*, pages 147–151, 1988. 3, 4

[6] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2003. 2, 3

[7] R. K. Lenz and R. Y. Tsai. Techniques for calibration of the scale factor and image center for high accuracy 3-d machine vision metrology. *PAMI*, 10(5):713–720, 1988. 1

[8] V. Lepetit, F. Moreno-Noguer, and P. Fua. EPnP: An accurate O(n) solution to the PnP problem. *IJCV*, 81(2):155, 2009. 7

[9] M. Li. Camera calibration of a head-eye system for active vision. In *ECCV*, 1994. 1

[10] S. Mason and A. Gruen. Automatic sensor placement for accurate dimensional inspection. *CVIU*, 3(61):454–467, 1995. 1

[11] G. Olague and R. Mohr. Optimal camera placement for accurate reconstruction. *Pattern Recognition*, 35(4):927–944, 2002. 1

[12] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. ROS: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009. 1

[13] A. Richardson, J. Strom, and E. Olson. AprilCal: Assisted and repeatable camera calibration. In *IROS*, 2013. 1

[14] C. Ricolfe-Viala and A.-J. Sanchez-Salmeron. Camera calibration under optimal conditions. *Optics Express*, 19(11):10769–10775, 2011. 1

[15] P. Rojtberg and A. Kuijper. Efficient pose selection for interactive camera calibration. In *ISMAR*, 2018. 1

[16] T. P. Runarsson and X. Yao. Search biases in constrained evolutionary optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 35(2):233–243, 2005. 3

[17] P. Sturm and S. Maybank. On plane-based camera calibration: A general algorithm, singularities, applications. In *CVPR*, 1999. 1

[18] R. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal on Robotics and Automation*, 3(4):323–344, 1987. 1

[19] P. J. Van Laarhoven and E. H. Aarts. Simulated annealing. In *Simulated annealing: Theory and applications*, pages 7–15. Springer, 1987. 3

[20] G.-Q. Wei and S. Ma. A complete two-plane camera calibration method and experimental comparisons. In *ICCV*, 1993. 1

[21] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *PAMI*, 14(10):965–980, 1992. 6

[22] C. Wu. Towards linear-time incremental structure from motion. In *3DV*, 2013. 7, 8

[23] Z. Zhang. A flexible new technique for camera calibration. *PAMI*, 22(11):1330–1334, 2000. 1