Efficient, Precise, and Convenient Calibration of Multi-Camera Systems by Robot Automation

Tobias Werner, David Harrer, and Dominik Henrich

Lehrstuhl für Robotik und Eingebettete Systeme,
Universität Bayreuth, D-95440 Bayreuth, Germany,
tobias.werner@uni-bayreuth.de,
http://robotics.uni-bayreuth.de

Abstract. Future use cases for stationary robot manipulators envision shared human-robot workspaces. However, shared workspaces may contain a priori unknown obstacles (e.g. humans). Robots must take these obstacles into account when moving (e.g. through online path planning). To this end, current research suggests real-time workspace monitoring with a calibrated multi-camera system. State-of-art solutions to camera calibration exhibit flaws in the above scenario, including long calibration times, excessive reprojection errors, or extensive per-calibration efforts. In contrast, we contribute an approach to multi-camera calibration that is at once efficient, precise, and convenient: We perform fully-automated calibration of each camera with a robot-mounted calibration object. Subsequent multi-camera optimization equalizes reprojection error over all cameras. After initial setup, experiments attest our contribution minor reprojection errors in few minutes time at one button click. Overall, we thus enable frequent system (re-)calibration (e.g. when moving cameras).

Keywords:
multi-camera systems, camera calibration, shared workspace monitoring, human-robot collaboration, obstacle reconstruction, path planning

1 Introduction

Human-robot collaboration promises to combine the individual virtues of humans and robot manipulators. Respective research envisions various future use cases for robots, from flexible industrial automation to applications in small businesses and the service sector. However, human-robot collaboration mandates shared human-robot workspaces, which contain a priori unknown obstacles (e.g. humans or human-placed objects). The robot manipulator must thus be able to avoid obstacles in real-time. State-of-art solutions for real-time obstacle avoidance (e.g. [1], [2]) propose a two-tier approach: Real-time workspace monitoring with a multi-camera system finds 3D workspace volumes that are occupied by obstacles. Concurrently, real-time path planning generates collision-free or risk-minimized robot trajectories around occupied workspace volumes.

Crucial to workspace monitoring and subsequent path planning is a precise calibration of the multi-camera system. In particular, calibration must provide

precise approximations of both intrinsic and extrinsic camera parameters (i.e. in reference to the coordinate system of the robot manipulator). However, the context of shared human-robot workspaces poses additional challenges for camera calibration: Users for instance may arbitrarily attach additional cameras (e.g. when flexibly changing workspace layout), users may advertedly or inadvertedly change camera positions, or vibrations on ad-lib camera mounts may cause slight camera displacement. Camera calibration for human-robot workspaces hence need not only be precise, but must also be efficient (to enable fast or even online recalibration) and convenient (to enable effortless calibration by non-experts).

State-of-art approaches do not meet the above three criteria. In contrast, we contribute a novel approach to calibrating a multi-camera system that is precise, efficient, and convenient: We automate calibration by a robot-mounted calibration object to enable online camera calibration at a single button click.

The remainder of our work is structured as follows: Section 2 surveys alternative approaches to camera calibration. In Section 3, we present and discuss our approach to efficient, precise, and convenient multi-camera calibration by robot automation. Section 4 continues with an evaluation of our contribution in precision and efficiency. Section 5 concludes with an outlook on future work.

2 Related Work

We discuss related work on camera calibration in three distinct categories: single-camera calibration, single-camera calibration with subsequent multi-camera optimization, and full multi-camera calibration.

Single-camera calibration usually estimates initial camera parameters (e.g. with a homography [3], direct linear transforms [4], or explicit ray equations [5]), then refines initial parameters (e.g. with Levenberg-Marquardt [3], non-linear optimization [4], or gradient descent [5]) for a more precise result. Comparative evaluation (e.g. [6], [7]) indicates that homography with Levenberg-Marquardt optimization (e.g. [3]) yields most precise results with minor runtime overhead.

As a multi-camera system consists of individual cameras, it is possible to calibrate each individual camera with one of the above single-camera calibration approaches. Subsequent optimization (e.g. with a spanning tree over camera pairs with intersecting frusta [8]) can apply knowledge about the multi-camera system to further refine parameter precision (e.g. by reducing cyclic errors [8]).

Finally, multi-camera calibration can determine parameters for all cameras at once without preceding single-camera calibration. This implies finding matching features over all cameras (e.g. by manually waving a bright light spot in all cameras [9]), followed by parameter estimation (e.g. by matrix factorization [9]). In preliminary experiments, we found this variant to be less precise than single-camera calibration alternatives, especially for cameras with distinct distortions.

Overall, the default practice for calibration is to use a hand-held calibration object. This practice is unfavorable in multi-camera systems: Occlusions (e.g. by the robot) enforce multiple calibration poses. Selecting and capturing those poses in turn is time-consuming, error-prone, and requires expert knowledge.

3 Our Approach

Our approach to multi-camera calibration shares one prerequisite with related work: We need multiple views onto a calibration object as input, and we need to know the pose of this calibration object in respect to the robot manipulator.

To solve both problems at once, we attach the calibration object (in our case, a checkerboard pattern on a wooden support frame) to the end effector mount of the robot manipulator. From CAD data of the robot and a one-time manual measuring, we can then determine a very precise estimate for the pose of the calibration object with respect to the robot even for an arbitrary choice of joint angles. In other words, we can now move the calibration object with the robot manipulator while precisely knowing the pose of the calibration object. Figure 1 illustrates our setup. Note real-world applications may replace the cumbersome checkerboard with a pattern that is conveniently imprinted onto the robot casing.

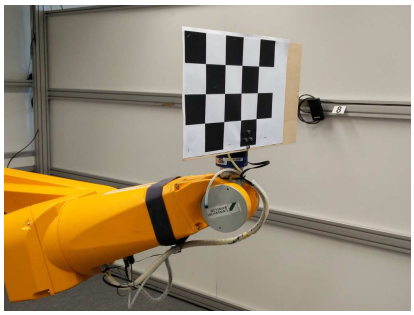

Fig. 1. Calibration object (a checkerboard pattern) mounted to the robot end effector.

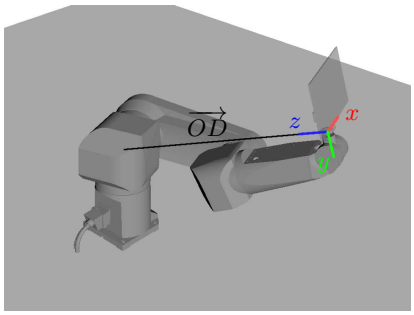

Fig. 2. Pose of the calibration object in the robot software with axes and generating vector $\overrightarrow{OD}$.

After mounting and measuring the calibration object once, we arbitrarily can perform our multi-camera calibration. Each subsequent calibration process takes three distinct steps: In the first step, the robot moves the calibration object through the scene while all cameras record images of the calibration object from different perspectives. In a second step, each camera is calibrated individually through a state-of-art single-camera calibration. In the third and final step, the information from overlapping camera frusta is used to refine individual results. Our calibration approach, in terms of related work, thus belongs to the category of single-camera calibration with subsequent multi-camera optimization. In the following, we discuss all three steps of our approach in greater detail.

3.1 Planning Robot Movement

Target poses for robot movement during the first step must satisfy two requirements: Target poses must avoid occlusions of the calibration object (e.g. by the

robot casing) in camera images, and target poses must be uniformly distributed over the workspace. Both requirements improve precision and efficiency of calibration by creating many correspondences over many cameras in a short time.

To satisfy above requirements, we choose origins for the calibration object on a sphere centered around the robot base (see Figure 4). A sphere radius r near workspace limits ensures that all but the first two robot joints remain fixed and thus we avoid self-collisions without explicit checks. A later online recalibration furthermore can use one of the available path planners (e.g. [1]) or collision mitigation (e.g. by soft or artifical skins) to cope with existing obstacles in the robot workspace on transfer movements, including the floor of the workspace.

To generate almost uniformly spaced points on the sphere, we pick random spherical coordinates $\theta \in [-180°, 180°)$ and $\varphi \in [-90°, 90°)$. An additional transform $\varphi = \arccos(2x_{\text{random}} - 1),\ x_{\text{random}} \in [0, 1)$ avoids dense sampling at the poles due to singularities. From spherical coordinates and previously defined radius r, we find a Cartesian position for the origin of the calibration object,

$$(r\sin(\varphi)\cos(\theta), \quad -r\cos(\varphi), \quad r\sin(\varphi)\sin(\theta)).$$

A respective orientation then is chosen for an upright calibration object that faces away from the sphere origin. The normalized vector from robot base to the origin of the calibration object becomes the z-axis in the local coordinate system of the calibration object (see Figure 2). The remaining axes are chosen as an orthogonal system with an x-axis parallel to the floor plane. Finally, the robot moves to each generated pose, and cameras record images of the scene.

3.2 Camera Calibration

For every recorded image of every camera we check whether the image completely contains the calibration object. If the calibration object is partially or completely missing (e.g. due to occlusions or frustum limits), we discard the image for the respective camera. Otherwise, we store object feature points in image coordinates alongside the 3D pose of the calibration object.

Once we have stored a preset number of images for an individual camera, we calibrate this camera through the popular openCV library: We first calculate intrinsic parameters and distortion coefficients without starting estimate from proxy points with $z = 0$ (i.e. as required by the implementation [3]). Thereafter, we find 3D feature points from saved poses, known object dimensions, and measured object transforms. We continue by refining camera intrinsics with correspondences between real image-space and 3D feature points. In a final step we use the openCV direct linear transform to find extrinsic camera parameters.

3.3 Parameter Optimization

Although preceding steps already perform automated camera calibration, we can still improve the precision of calibration: We can exploit the additional knowledge that we have a multi-camera system. To this end, we have evaluated two different methods: Stereo optimization based on camera pairs, and global error minimization.

Stereo Optimization The idea of stereo optimization for camera pairs (as inspired by [8]) is to equalize the error in extrinsic parameters over neighboring cameras. For a given camera i and another, close-by camera j with many shared correspondences, we first estimate a relative transform ${}^{i}\mathbf{K}_{j}$ from the pose of camera j to the pose of camera i by stereo calibration (e.g. with Levenberg-Marquardt or the openCV). Applying ${}^{i}\mathbf{K}_{j}$ to the pose of camera j yields a new estimate for the pose of camera i. Note this estimate differs from preceding extrinsics, as it exploits multi-camera connectivity through stereo calibration.

We now have two estimates for the pose of camera i: The single-camera guess and its counterpart from stereo calibration. Averaging the translational and the rotational components of these pose estimates gives us a new parameter set for camera i. While mean translations are trivial, a mean rotation is not clearly defined. Research (see [10]) proposes rotations that exhibit least deviation from originals. For quaternions $\mathbf{q}_i$ and $\mathbf{q}_j$ with real part w_i, w_j, this leads us to use

$$\mathbf{q}_{\text{mean}} = \sqrt{\frac{w_i(w_i - w_j + z)}{z(w_i + w_j + z)}}\mathbf{q}_i + \text{sign}(\mathbf{q}_i^T\mathbf{q}_j)\sqrt{\frac{w_j(w_j - w_i + z)}{z(w_i + w_j + z)}}\mathbf{q}_j,$$

$$\text{with} \qquad z = \sqrt{(w_i - w_j)^2 + 4w_i w_j(\mathbf{q}_i^T\mathbf{q}_j))^2}.$$

Camera i adopts the mean extrinsics. Finally, iterating over all close-by pairs (i, j) of cameras (possibly multiple times) increases precision of extrinsics.

Error Minimization In contrast to the camera pairs of stereo optimization, our global error minimization considers all cameras at once. At first, we discard all images on which the calibration object is only recorded by one camera, as we can not get any correspondences from those images. We are conversely left with all 3D feature points that have at least two 2D feature correspondences. Thereafter, we perform the actual optimization with Levenberg-Marquardt: In every iteration, we calculate the mean reprojection error and its gradient in the extrinsic parameter space of all cameras with correspondences on the current image set. After termination (e.g. due to epsilon error or limited iteration count), we continue on the next set of camera images.

4 Evaluation

Our test environment consists of a mock-up robot workcell ($4\,\text{m} \times 4,5\,\text{m} \times 2,8\,\text{m}$) with a Stäubli RX130 robot and eight inexpensive, consumer grade Logitech C930e Full HD webcams (see Figure 3). For testing our calibration approach, we recorded two image sequences, each with 75 images per camera. We then used one image sequence for calibration and the other one for evaluating our results. As calibration object, we used a checkerboard pattern with 4×3 relevant corners. Our error measure is the popular reprojection error (i.e. the mean squared error for an image-space distance metric between camera projections of 3D features and the corresponding 2D feature points).

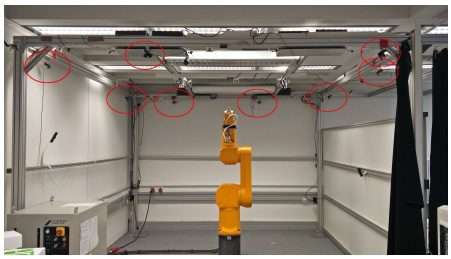

Fig. 3. Multi-camera system with eight consumer-grade Logitech C930e cameras (red circles) monitoring the workspace.

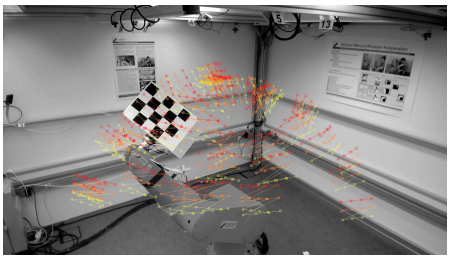

Fig. 4. Distribution of feature points (red and yellow) in a camera view, as collected over the entire calibration process.

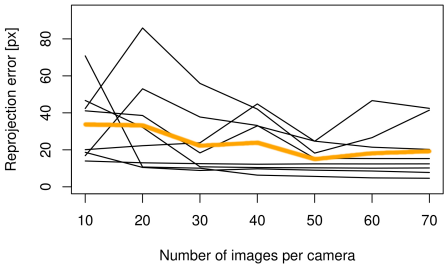

Fig. 5. Reprojection error of each camera with increasing number of input images, no intrinsic precalibration. Yellow: Mean of error over all eight cameras.

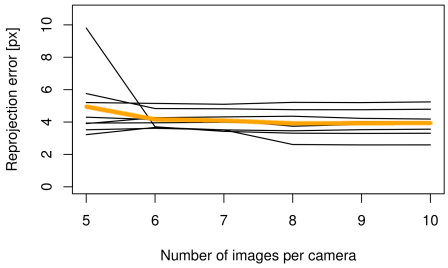

Fig. 6. Reprojection error of each camera with increasing number of input images, intrinsic precalibration. Yellow: Mean of error over all eight cameras.

4.1 Precision

Our first goal was to estimate an upper bound on the minimum reprojection error attainable in our single-camera calibration step (see Section 3.2). Additionally, we investigated the amount of images required for error convergence. We therefor calculated extrinsic and intrinsic parameters from the first $10, 20, ..., 70$ images and tested the resulting calibration against the 75 images of the testing dataset.

Figure 5 shows the change in reprojection error with increasing sequence length for each individual camera. As evident, more images in general lead to a lower squared mean of the reprojection error, with a viable 20 pixels average error after 50 images. Translating this error to the 3D workcell for Full HD cameras gives about 10 cm offset at the far end of the workspace and about 5 cm nearby the robot, well below tolerances for gross motion planning (see [1], [2]).

Still, the reprojection error for individual cameras varies moderately. It therefore is not possible to find an optimal number of input images. This precision problem stems from feature points that do not fully cover the field of view of some cameras: The manipulator range does not extend to all image corners, yet image corners exhibit greatest distortions and thus are particularly relevant for

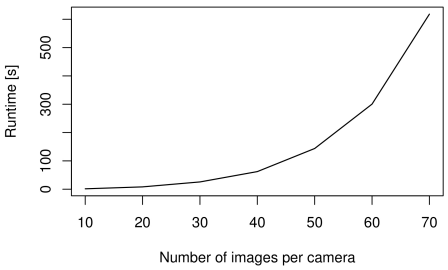

Fig. 7. Calibration runtime for increasing number of input images.

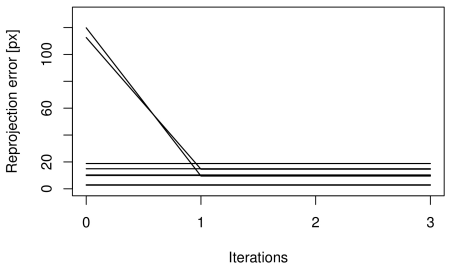

Fig. 8. Reprojection error over iterations of stereo optimization for artificial error.

camera intrinsics. As a solution, we derived an initial guess for intrinsics of each single camera by manually covering the entire field of view with a hand-held checkerboard. This significantly reduces later mean reprojection errors to about four pixels. Respective robot-supported camera calibration then exhibits stable and fast convergence after six images, as illustrated in Figure 6. While not quite as convenient as fully automated calibration, the optional pre-calibration step need only be performed once for each camera, and some manufacturers already provide individual intrinsics for each device.

4.2 Efficiency

In order to record 75 images with every camera, the robot arm must approach between 250 and 300 poses. Moving to a new pose and waiting for network transfers of camera images takes about ten seconds in our setup. This adds to a total runtime of 40 to 50 minutes for recording all images. Calibration itself roughly takes ten minutes after all images have become available. Computational complexity is $O(k\ O_{\mathrm{NL}}(n))$, with k the number of cameras, n the number of images per camera, and $O_{\mathrm{NL}}(n)$ the undocumented complexity of the openCV non-linear optimization. Opposed to this effort, stereo optimization and error minimization have a negligible performance overhead in terms of few seconds. See Figure 7 for runtime trends. Our experiments with alternative approaches (e.g. [3] or [9]) indicate several hours of manual calibration for similar precision.

When using intrinsically pre-calibrated cameras, the number of images necessary for reasonable precision drops to about five per camera and the time for recording reduces to roughly ten minutes. Because computing an extrinsics-only calibration takes only seconds, the overall process in turn completes in a few minutes, which enables a rather fast reaction to changes in camera placement.

4.3 Optimization

Optimization strategies (see Section 3.3) show their main benefit when single-camera calibration had poor results (e.g. due to numerical issues or mediocre feature point localization) for a limited subset of all cameras. In this case, our

experiments indicate that either optimization strategy reduces a reprojection error of 100 pixels (e.g. induced by explicitly corrupting the estimated extrinsic parameters) to a nominal 15 pixels in Full HD input. See Figure 8 for error trends of stereo optimization, trends for global error minimization are similar. For more accurate results of single-camera calibration (e.g. enabled by intrinsic precalibration), both optimization variants have significantly less impact, with improvements of at most 3 pixels in the reprojection error.

5 Conclusion

Evaluation attests that our contribution enables efficient recalibration of a multi-camera system in few minutes, with suitable precision for gross motion planning and at the convenient click of a button. Experiments further suggest to use an intrinsic pre-calibration, with a fallback to either stereo or global optimization if intrinsics are not available. Optimization is particularly relevant for applications with non-expert users (who cannot reliably perform intrinsics calibration) and inexpensive cameras (which cannot economically be factory-calibrated). Future work may involve error-adaptive recalibration with a casing-mounted pattern.

Acknowledgements

This work has partly been supported by the Deutsche Forschungsgemeinschaft (DFG) under grant agreement He2696/11 SIMERO.

References

1. T. Werner, D. Henrich, D. Riedelbauch, "Design and Evaluation of a Multi-Agent Software Architecture for Risk-Minimized Path Planning in Human-Robot Workcells", Kongress Montage Handhabung Industrieroboter, 2017.
2. T. Werner, D. Henrich, "Efficient and Precise Multi-Camera Reconstruction", Int. Conf. on Distributed Smart Cameras, 2014.
3. Z. Zhang, "A flexible new technique for camera calibration", Transactions on Pattern Analysis and Machine Intelligence, 2000.
4. J. Heikkilä, O. Silvén, "A four-step camera calibration procedure with implicit image correction", C. S. Conf. on Computer Vision and Pattern Recognition, 1997.
5. R. Y. Tsai, "A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses", Journal on Robotics and Automation, 1987.
6. W. Li, T. Gee, H. Friedrich, P. Delmas, "A practical comparison between zhang's and tsai's calibration approaches", Int. Conf. on Image and Vision Computing, 2014.
7. H. Zollner, R. Sablatnig, "Comparison of methods for geometric camera calibration using planar calibration targets", Technical report, Pattern and Image Processing Group, Vienna University of Technology, 2004.
8. B. Li, L. Heng, K. Koser, M. Pollefeys, "A multiple-camera system calibration toolbox using a feature descriptor-based calibration pattern", Int. Conf. on Intelligent Robots and Systems, 2013.
9. T. Svoboda, D. Martinec, T. Pajdla, "A convenient multicamera self-calibration for virtual environments", PRESENCE: teleoperators and virtual environments, 2005.
10. F. L. Markley, Y. Cheng, J. L. Crassidis, Y. Oshman, "Averaging quaternions", Journal of Guidance, Control, and Dynamics, 2007.