








PrIcosa: High-Precision 3D Camera Calibration with Non-Overlapping Field of Views

Oguz Kedilioglu^{1,*}^a, Tasnim Tabassum Nova^{1,*}^b, Martin Landesberger²^c, Lijiu Wang³^d,
Michael Hofmann³^e, Jörg Franke¹^f and Sebastian Reitelshöfer¹^g

¹*Institute for Factory Automation and Production Systems (FAPS),*

Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), 91058 Erlangen, Germany

²*Technische Hochschule Ingolstadt, Ingolstadt, 85049, Germany*

³*Heinz Maier-Leibnitz Zentrum (MLZ), Technical University of Munich, Garching, 85748, Germany*

Keywords: 3D Camera Calibration, Non-Overlapping Field of Views, Icosahedron, Probabilistic.

Abstract: Multi-camera systems are being used more and more frequently, from autonomous mobile robots to intelligent visual servoing cells. Determining the pose of the cameras to each other very accurately is essential for many applications. However, choosing the most suitable calibration object geometry and utilizing it as effectively as possible still remains challenging. Disadvantageous geometries provide only subpar datasets, increasing the need for a larger dataset and decreasing the accuracy of the calibration results. Moreover, an unrefined calibration method can lead to worse accuracies even with a good dataset. Here, we introduce a probabilistic method to increase the accuracy of 3D camera calibration. Furthermore, we analyze the effects of the calibration object geometry on the data properties and the resulting calibration accuracy for the geometries cube and icosahedron. The source code for this project is available at GitHub (Nova, 2024).

1 INTRODUCTION


Reconciling the two divergent goals of flexibility and accuracy is becoming an increasingly important task. As product variety and complexity increase, so do the demands on automation. To satisfy these requirements, a growing number of sensors are applied. These sensors must be integrated accurately and robustly into a coherent system in order to realize their full potential. That is why the calibration process is crucial. All subsequent steps rely on its performance.


Optical sensors are one of the most widely used sensor types for guiding flexible automation hardware such as industrial 6-axis robot arms. For the calibration of these camera systems, often 2D calibration


objects are utilized (Li et al., 2013) (D’Emilia and Di Gasbarro, 2017), (Lv et al., 2015). However 2D calibration objects only work when all cameras see the same calibration pattern. If the angles between the camera axes become too large and the field of views of the cameras do not overlap, then 3D calibration objects must be used. They allow the calibration of cameras without overlapping fields of views.


Various 3D geometries are available for the calibration objects, such as cubes (Tabb and Medeiros, 2019), (Rameau et al., 2022), (An et al., 2018), pyramids (Abedi et al., 2018), or icosahedrons (Ha et al., 2017). Usually, these objects consist of flat faces, here called boards, which are arranged into a rigid 3D structure. Distinct fiducial markers are placed on each board to enable individual identification. The size and number of these boards, and their relative pose to each other determine the quality of the images that can be captured by a multi-camera system for a given set of calibration object poses. Too few boards at the calibration object result in fewer detected boards by the cameras. If the angle between the camera axis and the board surface normal vector becomes too large, then the detection accuracy of the boards also suffers. And


^a <https://orcid.org/0000-0002-3916-805X>


^b <https://orcid.org/0009-0002-2745-6908>

^c <https://orcid.org/0000-0003-1104-7114>

^d <https://orcid.org/0009-0005-1338-3228>

^e <https://orcid.org/0000-0003-4936-9960>

^f <https://orcid.org/0000-0003-0700-2028>

^g <https://orcid.org/0000-0002-4472-0208>

^{0*}These authors contributed equally to this work.

if too many boards are added to a calibration object, then the available surface area per board becomes too small to contain enough feature points, which hurts the detection accuracy as well. So finding the right balance between number of boards, board size and shape, and relative board arrangement is crucial. This is where our first contribution comes in. We provide a quantitative comparison of the 3D board geometries cube and icosahedron.

Another aspect that we address concerns the calibration process itself (Figure 1; Step 5). The set of boards representing the full calibration object and the set of cameras that have to be calibrated together provide a combinatorial set of possible equations that describe their relations to each other, as indicated in Figure 2. These equations contain homogeneous transformations between camera-to-board, the cameras themselves, and the boards themselves. The accuracy of the camera-to-board transformations depends on the detection accuracy and the intrinsic and extrinsic camera parameters. For determining camera-to-camera transformations, hand-eye calibration is used, with its accuracy depending on the number and variety of the calibration object poses. Consequently, some equations yield less accurate transformations, while others produce better ones. The quality of these equations ultimately affects the calibration results. This is where our second contribution comes in. We provide an approach for selecting a favorable subset of all possible transformation combinations that improve the accuracy of the subsequent calibration algorithm.

Several critical factors that can influence the accuracy of hand-eye calibration have already been identified by Tsai and Lenz (Tsai and Lenz, 1989). They referred to each robot pose as a "station." According to their observations, accuracy is affected by the interstation rotation angle, the angle between different interstation rotation axes, the distance between the camera and the calibration board, the number of stations, and the rotation and translation errors at each station. They recommended using a larger number of stations with significant variations in rotation angles to improve accuracy. Additionally, they proposed a five-station configuration to ensure that the stations are uniformly distributed around the image frame. They also demonstrated that the distance between the camera and the calibration board, as well as the rotation and translation errors at each station, have a linear effect on accuracy.

To estimate rotation and translation vectors, the OpenCV (Bradski, 2000) ArUco pose estimation algorithm is commonly used. (Ošćádal et al., 2020) introduced a benchmark to assess the accuracy of pose

estimations. During their experiments, they found that estimating the Z-axis of the board, which looks perpendicularly away from the board surface, was particularly error-prone, as small changes in the position of the board could cause significant shifts in the Z-axis (Yaw). Additionally, they noted that the accuracy of roll estimation depends on the view angle of the camera.

We take these findings about the connection between view setup and the resulting calibration accuracy into consideration in order to improve our data quality. But our approach goes further by improving the calibration accuracy for a given dataset. Even if not all parts of the dataset are desirable, our approach can still extract beneficial information from it. This is achieved, among other things, by the fact that our probabilistic method goes beyond simple pose rules. We call our method *PrIcosa*, because of its Probabilistic nature and the utilization of the *Icosahedron* shape.

2 METHODOLOGY

In this section, the complete pipeline (Figure 1) of our multi-camera calibration framework *PrIcosa* is described. It is a six-step process, starting with the intrinsic calibration and ending with bundle adjustment for fine tuning.

2.1 Board Detection & Intrinsic Calibration

The goal of intrinsic parameter calibration is to establish a correspondence between 3D points in the real world and their corresponding 2D image projections. To initiate this calibration, we collect pairs of correspondences between 3D points on the ChArUco board pattern and their corresponding 2D image coordinates from multiple images that contain these patterns. These correspondences serve as the foundation for initializing the intrinsic parameters of the camera, represented by " K ", as well as the distortion coefficient, represented by " $dist$ ". For perspective cameras, we adopt the widely recognized calibration technique outlined by Zhang (Zhang, 2000).

The procedure starts by selecting all available images to compute an initial estimation of the intrinsic parameters and distortion coefficients. However, for certain views that exhibit significant errors due to various factors, we introduced a filtering mechanism. Views that display notable errors are systematically excluded from further analysis, leaving behind a subset of views that are deemed more reliable. This re-

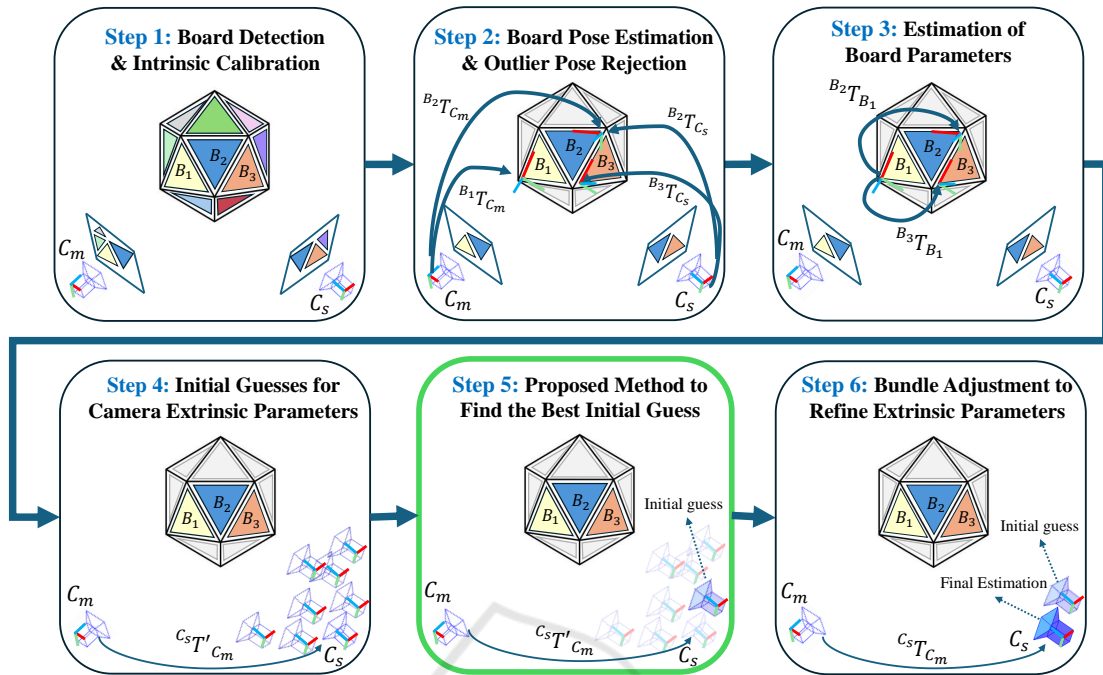


Figure 1: The complete pipeline of PrIcosa consists of 6 steps. Step 1: Each board is uniquely identified by its distinct ChArUco markers, and intrinsic parameters are calculated after detection. Step 2: Camera-to-board projection is calculated using intrinsic parameters and views with large re-projection error are rejected. Step 3: Transformation from master board to slave boards are calculated. Step 4: Initial guess of master camera to slave camera $C_s T'_{C_m}$ is calculated for all possible equations. Step 5: Probabilistic method is applied to select the best initial guess of $C_s T'_{C_m}$. Step 6: Refinement of initial guess and calculation of final estimation is performed using bundle adjustment.

finer set of views is then utilized to recalculate the intrinsic parameters and distortion coefficients. This process of refinement is performed iteratively, with the recalculated parameters being obtained from the filtered subset of images. The iterations continue until the level of error reaches a predefined threshold of improvement, essentially ensuring that the calibration process converges to a stable solution.

2.2 Board Pose Estimation and Outlier Rejection

Board pose estimation is the most crucial phase in the multi-camera calibration process where the spatial orientation and position of each camera in relation to a calibrated reference board are determined. In this step, we estimate the relative pose of all the cameras for each observed board using the intrinsic parameters that were calculated in the previous step. For estimating the pose, we use the OpenCV implementation of solvePnP iterative algorithm. In real-world scenarios, noise and various factors can introduce errors in the calibration process. To ensure the robustness and reliability of the calibration results, we filter out the poses that exhibit re-projection error beyond a certain

threshold of acceptance. These outlier poses are excluded from subsequent calibration steps.

2.3 Board Parameter Calculation

Our objective is to determine the relative poses between calibration boards to ultimately assemble them into 3D objects. These 3D objects are composed of multiple planar calibration boards. When a single image captures two or more of these boards, it provides an opportunity to estimate their relative poses, establishing pair-wise relationships. We collect measurements from all images where pairs of boards are visible together and compute the average inter-board rotation and translation. For this step, we followed the procedure discussed in (Rameau et al., 2022).

2.4 Extrinsic Camera Parameter Calculation

2.4.1 For Overlapping Cameras

When two cameras share an overlapping field of view such that they see the same calibration pattern, then the process of determining the transformation be-

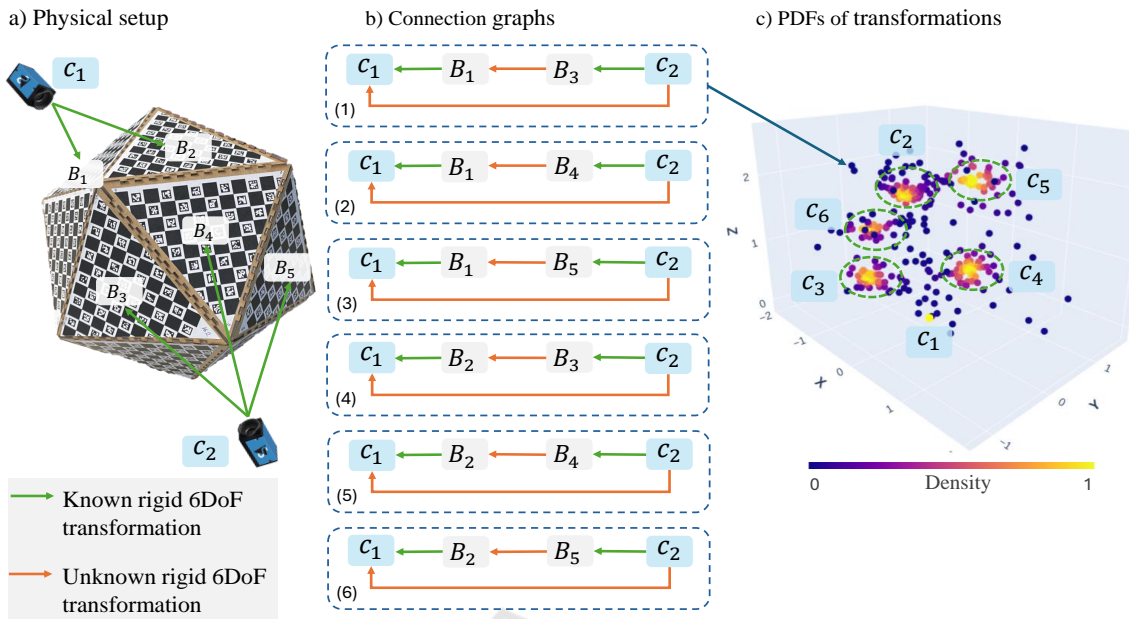


Figure 2: The physical setup of the calibration system, shown in a) with only two of the six cameras, can be represented by connection graphs, which can be seen in b) and correspond to 6DoF rigid transformations. Most of the time, there are multiple paths to create a closed chain of transformations between two cameras. The selection of this chain, or connection graph, affects the quality of the camera calibration significantly. In c) every dot represents a camera position after choosing one of the possible transformation chains. One probability density cluster for each of the six cameras is included in the diagram. For each camera the transformation chain corresponding to the point in the cluster center is chosen to get the best calibration results.

tween them becomes more straightforward. In this particular phase, we have gathered all pairs of cameras that possess a common field of view, and subsequently, we have employed the OpenCV stereoCalibrate method to compute their spatial relationship.

2.4.2 For Non-Overlapping Cameras

When cameras do not have overlapping fields of view, meaning they are not capturing the same scene simultaneously, then more sophisticated approaches are necessary. In such cases, traditional calibration methods that rely on shared scene points between cameras are not directly applicable. To address this challenge, we employ a hand-eye calibration approach. We designate one camera as the master (reference) camera and the others as slave cameras. The calibration boards viewed by the master camera are termed master boards, while those captured by slave cameras are called slave boards. In Figure 2, we illustrate a situation where the camera C_1 captures boards B_1 and B_2 , and another camera C_2 captures the boards B_3 , B_4 , and B_5 . In this setup, C_1 serves as the master camera with B_1 and B_2 as master boards, while C_2 is a slave camera with B_3 , B_4 , and B_5 as its slave boards. Each calibration group follows the sequence: *slave_camera* \rightarrow *slave_board* \rightarrow *master_board* \rightarrow

master_camera. This configuration can be mathematically modeled using the equation $AX = ZB$, which can be effectively solved using the hand-eye calibration method proposed by Tsai et al. (Tsai and Lenz, 1989).

However, it's important to note that not all hand-eye calibration groups have a diverse range of images needed for accurate calibration. To address this limitation, we have introduced an additional filtering technique to exclude irrelevant poses from these groups. Initially, we gather all possible one-to-one pose combinations for hand-eye calibration. Then, we analyze the rotation vectors associated with these poses. If there is any pose angle that significantly deviates from the others, the hand-eye calibration algorithm can encounter issues related to rotation normalization. To address this issue, we have introduced a precautionary measure to filter out problematic poses before initiating the hand-eye calibration algorithm. In this regard, we have employed the mean-shift clustering algorithm (Carreira-Perpinán, 2015) to group similar poses together. If any resulting cluster contains only one pose element, we automatically discard that cluster. Subsequently, we proceed to compute the hand-eye calibration for the cameras with the remaining poses.

2.4.3 Probabilistic Method for Outlier Rejection

When dealing with multiple planar calibration objects, a single camera can capture different boards in various frames, which introduces complexity to the calibration system.

For the particular situation observed in Figure 2, we observe six calibration groups and derive the following six equations (1)-(6) to compute the transformation from the camera C_2 to the camera C_1 (${}^{C_1}T_{C_2}$). These equations result in six different ${}^{C_1}T_{C_2}'$ transformations. In an ideal setting, all six of these transformations would align perfectly. However, in practical scenarios, this is not the case.

$${}^{B_1}T_{C_1} {}^{C_1}T_{C_2}' = {}^{B_1}T_{B_3} {}^{B_3}T_{C_2} \quad (1)$$

$${}^{B_1}T_{C_1} {}^{C_1}T_{C_2}' = {}^{B_1}T_{B_4} {}^{B_4}T_{C_2} \quad (2)$$

$${}^{B_1}T_{C_1} {}^{C_1}T_{C_2}' = {}^{B_1}T_{B_5} {}^{B_5}T_{C_2} \quad (3)$$

$${}^{B_2}T_{C_1} {}^{C_1}T_{C_2}' = {}^{B_2}T_{B_3} {}^{B_3}T_{C_2} \quad (4)$$

$${}^{B_2}T_{C_1} {}^{C_1}T_{C_2}' = {}^{B_2}T_{B_4} {}^{B_4}T_{C_2} \quad (5)$$

$${}^{B_2}T_{C_1} {}^{C_1}T_{C_2}' = {}^{B_2}T_{B_5} {}^{B_5}T_{C_2} \quad (6)$$

In the next step, we calculate the probability density function (PDF) of these six ${}^{C_1}T_{C_2}'$ transformations. Among these possibilities, we select the transformation that exhibits the highest probability as our initial estimate. For calculating PDF, we use the Gaussian Kernel Density Estimation method (Węglarczyk, 2018). The process is explained in Algorithm 1. It is important to note that this step applies to both non-overlapping and overlapping scenarios.

2.5 Bundle Adjustment

In this phase, we refine all camera-to-camera and board-to-board transformations to minimize the overall re-projection error using bundle adjustment. For clarity, we denote the master camera as C_m and slave cameras as C_s . The sets of master and slave boards are represented as B_m and B_s , respectively.

We calculate the re-projection error for each of the observed images. For instance, for pose ' p ' camera C_s observes board B_s . For the same pose ' p ' master camera C_m observes board B_m . Then we can calculate the camera C_s to board B_s transformation as follows:

$${}^{B_s}T_{C_s} = {}^{B_s}T_{B_m} {}^{B_m}T_{C_m} {}^{C_m}T_{C_s} \quad (7)$$

If camera C_s (with intrinsic parameters K_s) detects N image points (x) on board B_s at pose ' p ' and their corresponding 3D points are X , the re-projection error for that pose is:

Algorithm 1: ${}^{C_1}T_{C_2}$ Calibration.

Input: Camera Intrinsic(2.1), Board Poses(2.2)
Output: ${}^{C_1}T_{C_2}$

$Total_Cameras \leftarrow C_1, C_2;$
 $Total_Boards \leftarrow B_1, B_2, B_3, B_4, B_5;$
 $Master_Camera \leftarrow C_1;$
 $Master_Boards \leftarrow B_1, B_2;$
 $Slave_Boards \leftarrow B_3, B_4, B_5;$
for B_m **in** $Master_Boards$ **do**
 for B_s **in** $Slave_Boards$ **do**
 ${}^{C_1}T_{C_2}', {}^{B_m}T_{B_s} \leftarrow$
 $cv2.calibrateRobotWorldHandEye({}^{B_m}T_{C_1},$
 ${}^{B_s}T_{C_2});$
 $rtmatrix \leftarrow$ Collect ${}^{C_1}T_{C_2}';$
 $tvecs \leftarrow$ Collect translation vectors from
 ${}^{C_1}T_{C_2}';$
 end
end
 $density \leftarrow scipy.stats.gaussian_kde(tvecs);$
 $max_idx \leftarrow argmax(density);$
 $final\ {}^{C_1}T_{C_2} \leftarrow rtmatrix[max_idx];$

$$Re_p = \sqrt{\frac{1}{N} \sum_{i=1}^N \|x_i - K_s {}^{C_s}T_{B_s} X_i\|^2} \quad (8)$$

Similarly, we calculate the re-projection errors for all the images of all the cameras and take the average to calculate the overall re-projection error.

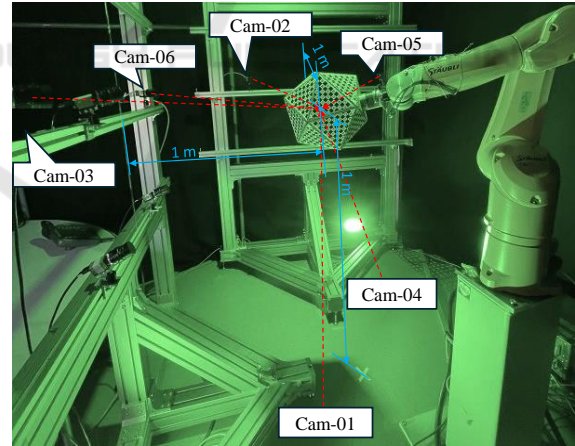


Figure 3: In our use case we have 6 cameras. The distance between each consecutive camera is around 1 meter and the distance between each camera pair baseline to the calibration object is also around 1 meter.

3 EVALUATION

We evaluate our results using two calibration objects: an icosahedron and a cube. For both of the objects, we use the same camera setup and robot gripper move-

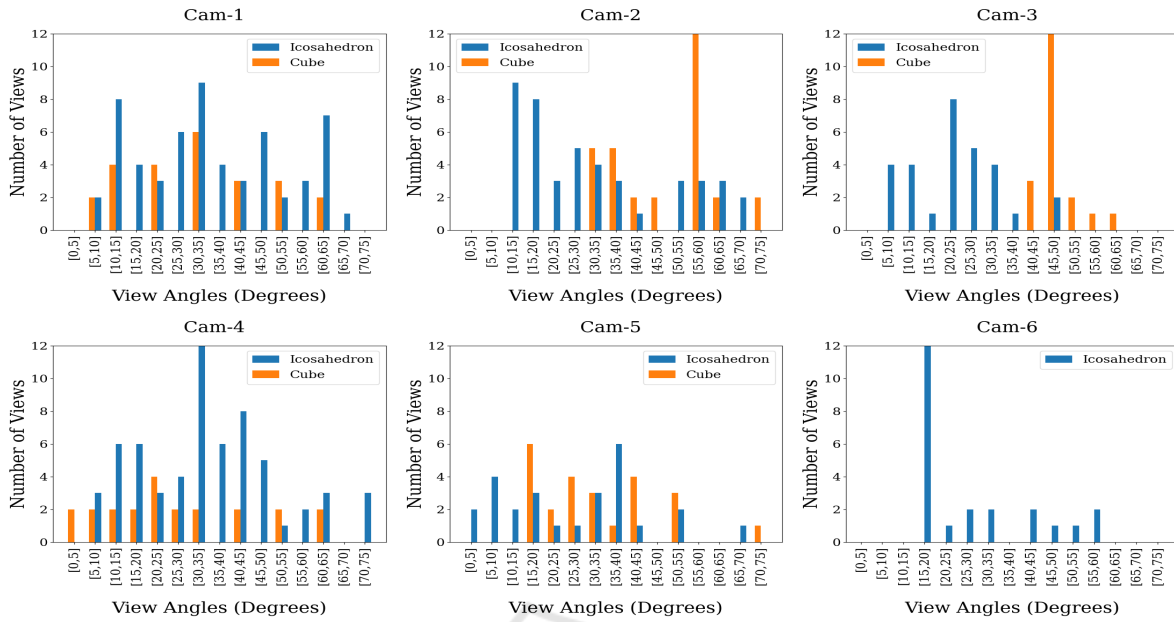


Figure 4: View angle variation for each camera. View angle is calculated from the camera to board projection matrix which is derived from solvePnP algorithm. Then Euler angles (Roll, Pitch and Yaw) are calculated from that matrix. $View\ angle = \sqrt{Roll^2 + Pitch^2}$.

ments for image acquisition. In the following subsections, we compare the outcomes obtained from these two calibration objects. We compare their differences in the variation of the views, the coverage in the image planes, and the resulting re-projection errors.

3.1 Experimental Setup

We utilize six 20MP monochrome industrial cameras (DMK 33GX183 33G-Series from 'The Imaging Source') with 50 mm focal length. Each camera is placed approximately 1 m away from the motion center of the calibration object. The cameras are arranged as stereo pairs with a baseline of 1 m. The camera pairs each look at the calibration object from three perpendicular directions, as shown in Figure 3.

The icosahedron is composed of equilateral triangles with sides of 17 cm, and the cube has square faces with sides of 220 cm. Both have the same ArUco 6x6_1000 pattern, where each square size is 13 mm.

To automatically move the calibration object to different poses, we employ a Staubli TX2-60L robot. The calibration object was mounted on the robot gripper, and we used ROS1 Noetic (Open Source Robotics Foundation, 2007) to control the robot and MoveIt (Coleman et al., 2014) to plan the motions for reaching the desired target poses with the calibration object.

To move the object with the robot hand, we followed a very simple motion plan. We moved the ob-

ject around the Z axis of the gripper by an Euler angle of -280° to 280° . The dataset is available at (Nova and Kedilioglu, 2024).

We adopted the "multical" (Batchelor, 2024) GitHub repository as a baseline for our multi-camera calibration system and implemented necessary modifications to address the specific challenges of non-overlapping camera setups.

3.2 View Variation

To achieve robust intrinsic parameter calibration, it is essential to ensure a wide variety of views for all the cameras. Optimal performance appears to be obtained when the angle between the image plane and the pattern plane is around 45° (Zhang, 1999). Regarding camera extrinsic parameter calibration, particularly with the hand-eye calibration method, it is equally important to gather a large number of views with significant variation (Tsai and Lenz, 1989).

To calculate the angle between the image plane and the pattern plane, we utilized the solvePnP algorithm, which provides the transformation between the camera and the board coordinates. This transformation results in a homogeneous matrix that we convert into Euler form. This conversion yields three rotation vectors: Roll, Pitch, and Yaw. Since the rotation around the Z-axis (Yaw) shows limited variation in pose, we exclude the yaw angle and instead focus on the absolute values of the Roll and Pitch angles, which

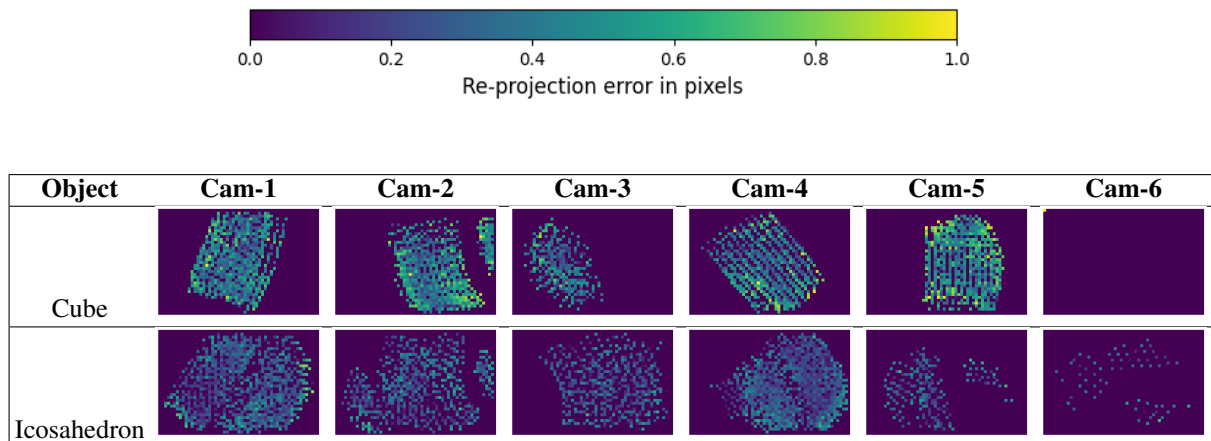


Figure 5: Image plane re-projection error after Bundle adjustment. It shows the average point error on a specific area of the image plane.

we define as the 'view angle'. For other computer vision tasks the rotation around the optical axis (= Yaw) is also not considered relevant (Mavrinac, 2012).

$$\text{View angle} = \|\text{Roll}^2 + \text{Pitch}^2\| \quad (9)$$

The comparison between the two calibration objects concerning view angle variation is illustrated in (Figure 4). Each bar in the plot represents the number of views within a specific view angle range. The plot clearly shows that images captured from the icosahedron object exhibit greater pose variety than those from the cube object. Additionally, it is noted that camera-6 did not detect any points for the cube calibration object, despite identical conditions for image acquisition being maintained for both objects.

3.3 Coverage in Image Plane

Distortion mostly affects near the edges of the image plane, so placing the calibration pattern near these edges is crucial. Ensuring a wider coverage area allows a more robust estimation of distortion parameters.

Figure 5 shows a heat map of the re-projection error for all of the points present on the image plane of each camera. This representation offers us a comprehensive insight into the extent of coverage of the image plane.

The image clearly shows that the icosahedron object achieves a larger coverage area with lower re-projection error compared to the cube object.

3.4 Per View Re-Projection Error

Figure 6 displays the final re-projection error results. In this experiment, 20 images from each camera were captured using the same robot motion and

camera setup. Each bar in the graph represents the re-projection error for an individual image. The results clearly indicate that for all cameras, the overall re-projection error when using the icosahedron object is consistently below 1 pixel. In contrast, the cube calibration object results in significantly higher re-projection errors. This demonstrates the superior accuracy and effectiveness of the icosahedron in calibrating the cameras.

3.5 Discussion

Throughout the experiments, we have highlighted the importance of maintaining large variability in camera poses for good-quality calibration. Figure 4 and 5 illustrate the superior coverage and view variation achieved using the icosahedron compared to the cube, and Figure 6 presents the resulting calibration results.

Despite the complexity of our camera setup, the distinctive shape of the icosahedron enabled us to attain the necessary view variety with minimal robot motion. While it might be possible to replicate similar results with a cube, it would require more intricate robot motions, which would need to be adjusted for different camera setups. In industrial settings with an increasing number of cameras, such adjustments become increasingly challenging and time-consuming. The icosahedron, therefore, offers an elegant and efficient solution, streamlining the calibration process in complex, multi-camera environments.

Even after meeting all calibration criteria, it is possible that sometimes the final re-projection error may still be high. This could occur if the bundle adjustment process converges at a local minimum. A good initial guess can help prevent this issue, and our probabilistic algorithm assists in selecting optimal initial guesses. During the extrinsic parameter

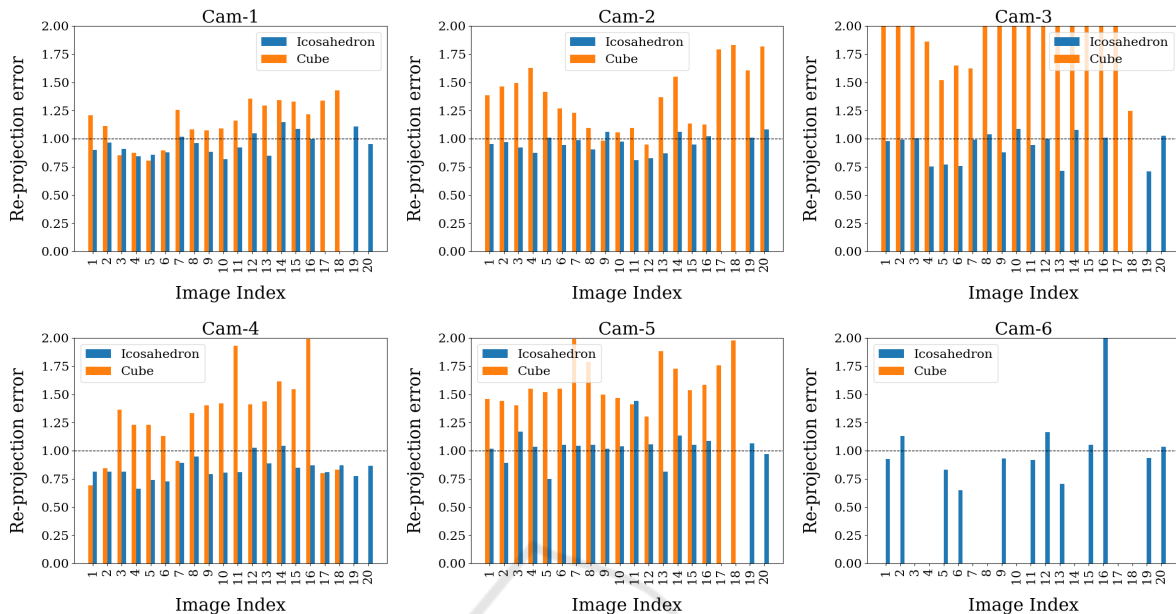


Figure 6: Per image re-projection error after bundle adjustment. One image consists of multiple views. Image re-projection error is the average error of all views.

calibration, we observed that some calibration groups were highly scattered. This scattering tends to occur when a group has fewer images, limited variety, or incorrect pose estimations. When using a calibration object with a large number of boards, it becomes possible to create a large number of calibration groups, which facilitates a more robust initial guess. For instance, with the icosahedron object, we have around 100 groups for each camera pair. While some groups are outliers, the inlier groups tend to cluster densely around the center (Figure 2(c)), aiding in the selection of initial parameters. In contrast, using the cube object results in an average of only 4 groups per camera, making it challenging to estimate a robust solution.

The structural advantage of the icosahedron, combined with the probabilistic approach of our PrIcosa framework, enables it to identify the optimal calibration group even from subpar datasets. For instance, when analyzing the icosahedron datasets from cameras 5 and 6, we observe that these cameras have limited view variation and image plane coverage (Figure 4 and 5). Despite these shortcomings, our framework still manages to find the best solution within the poor dataset (Figure 6).

4 CONCLUSIONS

We have shown that an icosahedron-shaped calibration object leads to considerably better calibration results than a cube-shaped calibration object for the same set of views and calibration object poses. The icosahedron represents a better balance between the quantity and size of the boards. It provides significantly more variety in the dataset and better coverage of the image planes with useful patterns. This enables smaller re-projection error values and improves the calibration accuracy.

To get the most out of the acquired dataset, we developed and evaluated a probabilistic method that uses an optimized subset of possible input equations for the optimization algorithms that are used to calibrate the cameras. By taking the cluster center of all possible camera positions generated by the probability density function, we can minimize the re-projection error in the calibration process.

Further research could consist of the exploration of different calibration patterns. The localization of the features of these patterns should be accurate and robust. The patterns should also work in the case of partial overlap. They should have a unique ID such that each board can be identified in the images. Their geometry should be able to handle sharp angles and still be accurate enough. This would allow to increase

the pose variety in the dataset.

Another aspect that deserves further investigation is the selection of the reference camera. Our current implementation is time-consuming, mainly in the bundle adjustment phase. This process involves selecting each camera as the reference camera in turn and performing the bundle adjustment repeatedly, resulting in increased time requirements as the number of cameras grows. A potential modification to address this issue involves dynamically selecting the best reference camera by analyzing the observed views of each camera. Our evaluation in the previous section demonstrated that cameras with a substantial degree of pose variability yield better results. Therefore, automatically determining the reference camera based on observed view characteristics could optimize the calibration process, especially in scenarios with a large number of cameras.

ACKNOWLEDGEMENTS

The project received funding from the German Federal Ministry of Education and Research under grant agreement 05K22WEA / 05K22WO1 (AutoTron).

REFERENCES

- Abedi, F., Yang, Y., and Liu, Q. (2018). Group geometric calibration and rectification for circular multi-camera imaging system. *Opt. Express*, 26(23):30596–30613.
- An, G. H., Lee, S., Seo, M.-W., Yun, K., Cheong, W.-S., and Kang, S.-J. (2018). Charuco board-based omnidirectional camera calibration method. *Electronics*, 7(12).
- Batchelor, O. (2024). multical. <https://github.com/oliver-batchelor/multical>. GitHub repository.
- Bradski, G. (2000). The opencv library. *Dr. Dobbs's Journal: Software Tools for the Professional Programmer*, 25(11):120–123.
- Carreira-Perpinán, M. A. (2015). A review of mean-shift algorithms for clustering. *arXiv preprint arXiv:1503.00687*.
- Coleman, D., Sucas, I., Chitta, S., and Correll, N. (2014). Reducing the barrier to entry of complex robotic software: a moveit! case study. *arXiv preprint arXiv:1404.3785*.
- D'Emilia, G. and Di Gasbarro, D. (2017). Review of techniques for 2d camera calibration suitable for industrial vision systems. In *Journal of Physics: Conference Series*, volume 841, page 012030. IOP Publishing.
- Ha, H., Perdoch, M., Alismail, H., Kweon, I. S., and Sheikh, Y. (2017). Deltile grids for geometric camera calibration. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5354–5362.
- Li, B., Heng, L., Koser, K., and Pollefeys, M. (2013). A multiple-camera system calibration toolbox using a feature descriptor-based calibration pattern. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1301–1307.
- Lv, Y., Feng, J., Li, Z., Liu, W., and Cao, J. (2015). A new robust 2d camera calibration method using ransac. *Optik*, 126(24):4910–4915.
- Mavrinac, A. (2012). Modeling and optimizing the coverage of multi-camera systems.
- Nova, T. T. (2024). PrIcosa: High-precision 3d camera calibration with non-overlapping field of views. https://github.com/TabassumNova/Multi_Camera_Calibration. GitHub repository.
- Nova, T. T. and Kedilioglu, O. (2024). Non-Overlapping Multi-Camera Calibration Dataset using Icosahedron and Cube Calibration Object. <https://doi.org/10.5281/zenodo.13294455>.
- Open Source Robotics Foundation (2007). Robot operating system (ros). <https://www.ros.org>. <https://www.ros.org>.
- Oščádal, P., Heczko, D., Vysocký, A., Mlotek, J., Novák, P., Virgala, I., Sukop, M., and Bobovský, Z. (2020). Improved pose estimation of aruco tags using a novel 3d placement strategy. *Sensors*, 20(17).
- Rameau, F., Park, J., Bailo, O., and Kweon, I. S. (2022). Mc-calib: A generic and robust calibration toolbox for multi-camera systems. *Computer Vision and Image Understanding*, 217:103353.
- Tabb, A. and Medeiros, H. (2019). Calibration of asynchronous camera networks for object reconstruction tasks. *CoRR*, abs/1903.06811.
- Tsai, R. and Lenz, R. (1989). A new technique for fully autonomous and efficient 3d robotics hand/eye calibration. *IEEE Transactions on Robotics and Automation*, 5(3):345–358.
- Węglarczyk, S. (2018). Kernel density estimation and its application. In *ITM web of conferences*, volume 23, page 00037. EDP Sciences.
- Zhang, Z. (1999). Flexible camera calibration by viewing a plane from unknown orientations. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 666–673 vol.1.
- Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334.