A Flexible Grasping Policy Based on Simple Robot-Camera Calibration and Pose Repeatability of Arm

Guowei Cui, Guangda Chen, Zekun Zhang, and Xiaoping Chen(✉)

Multi-Agent Systems Lab, School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, Anhui, China
{cuigw,cgdsss,kzz}@mail.ustc.edu.cn, xpchen@ustc.edu.cn

Abstract. Manipulation has been a rising focus of robotics research, from industrial automation to personal service robots. Most existing control method are based on accurate kinematic modelling of robot manipulators and robot calibration that needs trained personnel, laboratory environment and data collection is time-consuming. However, robots, especially service robots, similar to other mechanical devices can be affected by slight changes or drifts caused by wear of parts, dimensional drifts, and tolerances, and all of them need a new calibration. Most methods lack robustness due to these slight changes. In this work, for mobile robots, we propose a convenient robot-camera calibration approach and a flexible control approach based on simple robot sensor calibration and pose repeatability of arm. In prototype grasping experiments on the robot Kejia, the approach is validated to be effective and robust, achieving a high success rate of grasping, and using the same control policy, we achieve 1st place in Manipulation and Object Recognition of RoboCup@Home League 2016, here Manipulation and Object Recognition is a sub challenge of Home League.

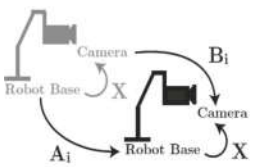
Keywords: Robot-camera calibration · AR tag · Pose repeatability
Grasping · Mobile robot

1 Introduction

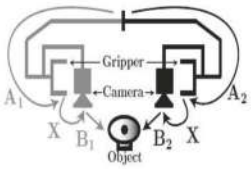
Vision-guided system have been widely employed in robotics such as service robots, automated guided vehicles, etc. To make it work, robot-sensor calibration is essential. The most common mathematical representations for the robot-sensor calibration problem is the form: $\mathbf{AX} = \mathbf{XB}$, proposed by Tsai and Lenz [1] and Shiu and Ahmad [2], as shown in Fig. 1 [14].

Prototype robot-sensor calibration methods are based on tracking a stationary object's pose change in both tracker's coordinate system and the camera's optical axis. Most of the methods need detect a stationary object from different poses and then solving for rotation and translation either separately [1], jointly

Z. Chen et al. (Eds.): ICIRA 2018, LNAI 10985, pp. 89–99, 2018.
https://doi.org/10.1007/978-3-319-97589-4_8



(a) **AX** = **XB** where the unknown X represents the transformation from robot base to camera.



(b) **AX** = **XB** where the unknown X represents the transformation from robot gripper to camera.

Fig. 1. Robot-sensor calibration.

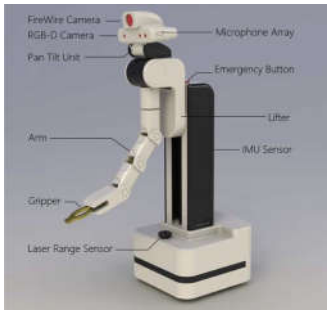


Fig. 2. Robot Kejia.

[4], or iteratively [5,6]. The calibration can be reduced to paired-point registration [7,8,12,19], Voruganti and Bartz [9] and Chen et al. [10] used a calibrated, tracked planar chessboard pattern for robot-sensor calibration, where the 3D position of the chessboard corners is determined in both the tracker's coordinate system (via tracking) and the camera's optical axis (by solving some forms of Perspective-n-Point problem). [11] trained a large convolutional neural network to predict the probability that task-space motion of the gripper will result in successful grasps, independent of camera calibration or the current robot pose, but it required large-scale data collection.

Except robot-sensor calibration, most grasping methods also rely on kinematic calibration [1–7], which is also called level-2 calibration [15]. Kinematic calibration consists of four sequential steps: modelling, measurement, identification and compensation. Measurement is the most difficult and time-consuming phase of robot calibration.

The measuring techniques of robot-sensor calibration or kinematic calibration, vary considerably in accuracy, ease of use and cost, but all share the following drawbacks [15]:

- Trained personnel are required to operate the measuring devices.
- Data collection is time-consuming.
- Except few of them, these techniques are based laboratory environments.
- The set-up and measurement procedures required a lot of human intervention.

We design a calibration method that can calibrate the pose of a RGB-D camera relative to the center of the robot, and a grasping policy that doesn't need hand-eye calibration, only needs some compensation. The calibration framework is of great convenience since it only needs a desktop and AR Tag. Based on simple robot-sensor calibration, we validate the grasping policy on a prototype grasping task on the robot Kejia, which is a robot as Fig. 2 shown, and the designed grasping approach shows robustness and adaptability in grasping process.

2 Robot-Camera Calibration

In this work, we introduce a method of calibrating the initial pose of the camera without a tracking device. Initial pose of robot's camera can be denoted as a six-tuple $(x, y, z, \gamma, \beta, \alpha)$ in robot base frame, where (x, y, z) are translation offsets and (γ, β, α) are RPY (roll-pitch-yaw) rotation angles. The translation offsets and rotation angles describe translation transformation $T_{XYZ}(x, y, z)$ and rotation transformation $R_{XYZ}(\gamma, \beta, \alpha)$, and the complete transform $\mathbb{C}$ from camera frame to robot base frame is defined as $T_{XYZ}(x, y, z)R_{XYZ}(\gamma, \beta, \alpha)$.

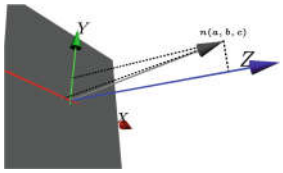


Fig. 3. Plane detected and its normal vector.

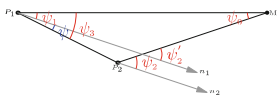


Fig. 4. Horizontal profile of moving.

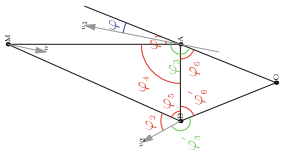


Fig. 5. Horizontal profile of rotation.

2.1 Roll and Pitch Rotation

In this case, we are using a robust RANSAC estimator of the Point Cloud Library (PCL) [16] to calculate the best possible model of the planar surface in $aX + bY + cZ + d = 0$ form. As shown in Fig. 3, the *roll* rotation causes the normal vector to change along the Y axis and clockwise rotation causes component b to increase, and the *pitch* rotation causes the normal vector to change along the X axis but clockwise rotation causes component a to decrease.

So we can adjust the *roll* and *pitch* rotation values, γ and β, based on the components a and b of the normal vector of the plane detected by the RGB-D camera. To improve the accuracy of plane estimation, ROI (range of interest) and multiple measurements averaging are helpful, and points that are more than 2 m away from the camera are also filtered. The ROI is a center area of the image, and 1/4 size of the origin image. Data used in plane estimation must be transferred from camera frame to robot base frame, since the plane observed is perpendicular to axis Z of the robot.

2.2 Yaw Rotation

We calibrate yaw rotation by means of geometric skill with an AR Tag [17], which allows for video tracking capabilities that calculate a camera's position and orientation relative to physical markers in real time. In this work, we use multi-tag bundles, this can allow for the estimation of the pose of a many-sided object, even when some of the tags cannot been seen. It can also lead to more

stable pose estimates and provide robustness to occlusion, the pose of the tags is provided by ROS (Robot Operating System) package *ar_track_alvar*.

To measuring the *yaw* rotation value, the marker was attached to the wall or ground, no special pose requested, camera moved from point P_1 to P_2, as shown in Fig. 4, where M is the center of the multi-tag bundles, $\mathbf{n_1}$ and $\mathbf{n_1}$ are camera's direction vector in A and B. Vector $\boldsymbol{P_1P_2}$ is parallel to robot's direction vector and ψ, the angle between camera's direction vector and robot's direction vector, is *yaw* rotation value. Here is an assumption that the forward trajectory is a straight line. There is some bias when moving straight, but the bias conforms to be very small, the ability of moving straight is analyzed in Sect. 5.1.

From position P_1 and P_2, the position observed of M are (x_1, y_1, z_1) and (x_2, y_2, z_2), which are values transformed from camera frame to robot base frame based on *roll* and *pitch* values calibrated in Sect. 2.1. Ignoring the height, only considering the horizontal component, easy to obtain $\psi_1, \psi_2, |MP_1|, |MP_2|$.

Since $\boldsymbol{P_1P_2}$ is robot's direction vector and the relative positional relationship between the motion center and the camera during the moving has not changed, camera's direction vectors in point A and B, $\mathbf{n_1}$ and $\mathbf{n_2}$, are parallel to each other, so $\psi_0 = \psi_2' - \psi_1 = \psi_2 - \psi_1$. Combining the cosine theorem, $|AB|$ and ψ_3 can be derived, then $\psi = \psi_3 - \psi_1$, and ψ is the yaw rotation value α in the six-tuple $(x, y, z, \gamma, \beta, \alpha)$.

2.3 Offsets

To get the translation offsets, (x, y, z), we make the robot to rotate a certain angle, like Fig. 5 describes, the point O is the motion center of the robot that is not moving during the rotation, point A is the camera's position before rotation and point B is the position after rotation and M is multi-tag bundles. Vectors $\mathbf{v_1}$ and $\mathbf{v_2}$ that are parallel to robot's direction vector after RPY rotation calibrated are camera's direction vector in A and B. Figure 5 is based on a priori knowledge and the camera is in front of the motion center of the robot.

Before and after the rotation, camera's relative position to robot's motion center O has not changed, φ_3' and φ_3 are equal to each other, and φ_6' is equal to φ_6 since $|OA|$ equal to $|OB|$. The multi-tag bundles can define a coordinate system. Assume that the coordinate system define by the M-centered tags is $\boldsymbol{\Psi}$. Robot can get the position of A and B in $\boldsymbol{\Psi}$ based on the observations of M in point A and B. We can get $|AB| = |A_\Psi$. According to the cosine theorem, φ_4 and φ_5 can be derived since we got the length of AB, MA and MB. Observing Fig. 5, easy to found:

$$\begin{aligned} \varphi_3 &= \varphi_1 + \varphi_4 + \varphi_6 \\ \varphi_3' &= 2\pi - \varphi_2 - \varphi_5 - \varphi_6' \\ \varphi_6 &= (2\pi - \varphi_1 - \varphi_4 - \varphi_2 - \varphi_5)/2 \end{aligned} \tag{1}$$

Then $|AB|$ and φ can be derived. Since $|OA|$ and φ have been calculated, the translation offsets can be derived:

$$\begin{aligned} x &= |OA| \cos \varphi \\ y &= |OA| \sin \varphi \\ z &= (-z_1 - z_2)/2 \end{aligned} \tag{2}$$

3 Hand-Eye Calibration

To simplify it, an AR Tag is attached to the wrist, the part of the arm closest to the gripper. The arm is to be set a initial state without an exact calibration, we manually put the arm in a straight line, pointing down to the ground, as the initial state, and each joint's configure is set to be *zero*. This is not very accuracy without measurement or calibration, which can be compensated by an AR tag mounted on the wrist, as Fig. 6 shown. We manipulate the arm to some specific poses, observing the AR tag mounted on the wrist, to get the pose of the end effector by applying transformation from the AR tag to the end effector, then modelling the pose of the end effector as hand-eye transform with the fixed arm configuration, that configuration consists of the angle of each joint.

4 The Grasping System

An overview of our system is illustrated in Fig. 8. At each stage, the robot makes an *observation* of the world, which is input to a *processing* module to generate object information. This object information is combined with the camera's current pose in the *predict* module to generate a success belief of this grasping. The robot make an *grasping* if the belief is acceptable, or a *planning* module uses the current belief and object information to determine the next location to make an observation. The target location is passed to a *navigation* module which drivers the robot to the desired goal.

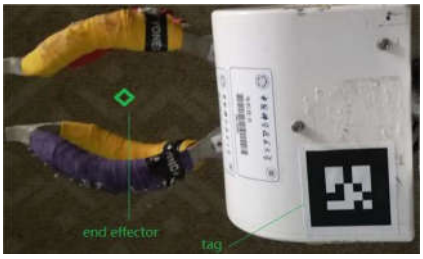


Fig. 6. The gripper and the AR tag.

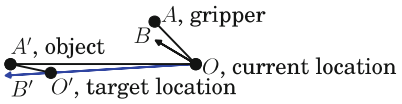


Fig. 7. Robot navigating from O to O'.

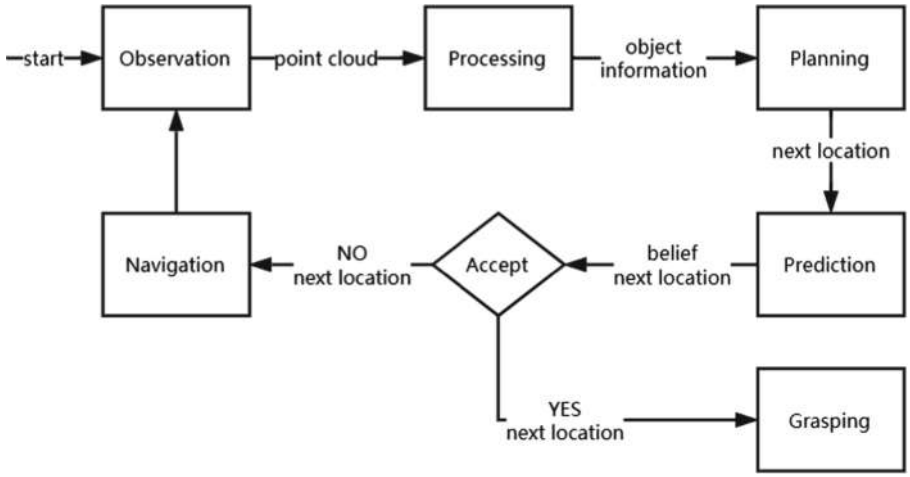


Fig. 8. Overview of the system components.

4.1 Planning

This subsection generate a next navigation location, as shown in Fig. 7, O and O' are robot's current location and target location that is where robot stands to make the gripper tries to reach object location A' from current location A, $\boldsymbol{OB}$, $\boldsymbol{O'B'}$ are robot's direction vector. We can compute OO' and $\angle BOO'$ based on the following conditions:

$$OA = O'A',\ \angle AOB = \angle AOB,\ OO' \parallel O'B'$$

4.2 Grasping

Observing the AR tag on the wrist under a certain arm configure, the position of the tag $(x_{tag}, y_{tag}, z_{tag})$ is passed to *predict* and *planning*, and angles $\{\theta_1, \theta_2, ..., \theta_N\}$ for robot's N joints is passed to the *grasping* module which applies these values to the joints to finish an grasping.

The *grasping* module is composed of a sequence actions: *turn left/right*, *moving forward/backward*, *elevator up/down* and *arm control*. Robot drives itself moving to make a success grasp due to the special arm pose.

4.3 Prediction

The distance between the location (x_g, y_g, z_g) the end effector and the observed object's location (x_o, y_o, z_o) has a marked impact on this belief. The object's length, width and height are l_o, w_o, and h_o, the gripper's length l_g, width w_g, height h_g. The end effector's position is generated by applying a transformation, that is constant after the AR tag is fixed on the wrist, to AR

tag's position. We define the belief as a combination of beliefs on three axes: $belief = belief_x belief_y belief_z$.

The components of $belief$ are based on error $\tilde{e}(\tilde{e}_x, \tilde{e}_y, \tilde{e}_z)$ caused by actions described in the *grasping* module and the observations. If a success grasping happens, the object must can be put into the opened gripper, that means object's width is smaller than gripper's, so we define $belief_y$ as function

$$belief_y = 1 \ if \ (w_g - w_o - |\tilde{e}_y|) > |y_g - y_o| \ then \ 0 \tag{3}$$

It is hardly to define belief in X and Z axes like (3), for example, robot can lift the object by only gripping the front of the object. To simplify it, we define $belief_x$ and $belief_z$ as following:

$$\begin{aligned} belief_x &= 1 \ if \ (|x_o - x_g| + |\tilde{e}_x| < max(0.5l_o, 0.25l_g)) \ then \ 0 \\ belief_z &= 1 \ if \ (|z_o - z_g| + |\tilde{e}_z| < 0.5h_g) \ then \ 0 \end{aligned} \tag{4}$$

4.4 Error Estimation

Each grasping attempt in our experiments composed of five movements, and each of them results in an error:

- Observation e^{obs}.
- Forward or backward movement e^{mv}.
- Turning. This error is related to the target ψ of the robot's angular position relative to the origin and the actual turning angle ψ'.
- Elevator (the lifter as shown in Fig. 2) up and down e^{elev}.
- Arm moving. Since we use an AR Tag, there is no need to take the arm modelling into account, the error caused by grasping is mainly determined by the pose repeatability of the arm and can be ignored due to its high accuracy.

Errors, such like e^{mv}, e^{elev}, caused by robot's movement, can be sampled. The observation error is sensitive to robot-sensor calibration, if the observation (x_o, y_o, z_o) in Kinect frame, ground truth (x'_o, y'_o, z'_o), Kinect pose calibration result $(x_c, y_c, z_c, \gamma_c, \beta_c, \alpha_c)$ and current real camera pose $(x'_c, y'_c, z'_c, \gamma'_c, \beta'_c, \alpha'_c)$ are known, the observation error in base frame can be computed by (3). It is sensitive to the distance between the object and the gripper.

$$\begin{aligned} &\boldsymbol{T} = R_{XYZ}(\gamma_c, \beta_c, \alpha_c), \ \boldsymbol{T'} = R_{XYZ}(\gamma'_c, \beta'_c, \alpha'_c) \\ &\delta x = (x_o - x_a) - (x'_o - x'_a), \delta y = (y_o - y_a) - (y'_o - y'_a), \delta z = (z_o - z_a) - (z'_o - z'_a) \\ &\delta\gamma = \gamma_c - \gamma'_c, \ \delta\beta = \beta_c - \beta'_c, \ \delta\alpha = \alpha_c - \alpha'_c, \\ &e^{obs} = \boldsymbol{T} \begin{bmatrix} x_o - x_a \\ y_o - x_a \\ z_o - x_a \\ 1 \end{bmatrix} - \boldsymbol{T'} \begin{bmatrix} x'_o - x'_a \\ y'_o - y'_a \\ z'_o - z'_a \\ 1 \end{bmatrix} = \underbrace{(\boldsymbol{T} - \boldsymbol{T'}) \begin{bmatrix} x_o - x_a \\ y_o - x_a \\ z_o - x_a \\ 1 \end{bmatrix}}_{e^{obs1}} + \underbrace{\boldsymbol{T} \begin{bmatrix} \delta x \\ \delta y \\ \delta z \\ 1 \end{bmatrix}}_{e^{obs2}} \end{aligned} \tag{5}$$

In (3) (x'_a, y'_a, z'_a) is the end effector's ground truth position value, and (x_a, y_a, z_a) is the observation in Kinect frame.

As the observations of object and AR Tag get closer, δx decreases [18] and so on. In the experiments, we ignore e^{obs2}. The final error in three axes can be defined as following:

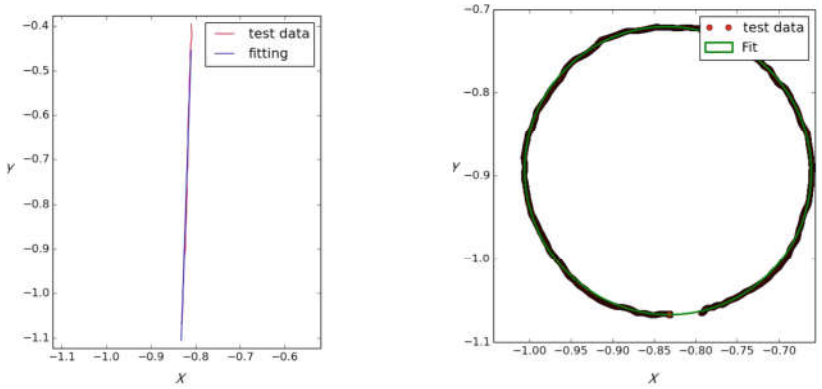
$$\begin{aligned} \tilde{e}_x &= e_x^{obs} + e^{mv} + d(\cos(\psi' - \psi) - 1) \\ \tilde{e}_y &= e_y^{obs} + d\sin(\psi' - \psi) \\ \tilde{e}_z &= e_z^{obs} + e^{elev} \end{aligned} \tag{6}$$

5 Experiments

In this section, we evaluate the accuracy and repeatability of the data of the Kinect and the movements of the robot. Finally, the grasping system is validated in prototype grasping experiments on the robot Kejia, achieving a high success rate of grasping. In order to finish these measurements, MCS (Motion Capture System) [20] is used to track the pose of the markers mounted on the robot.

5.1 Robot-Camera Calibration

As Fig. 9 shown, the ability of robot's moving straight and turning without center changed are strong. Based on turning (to find robot's center) and moving straight (to find the front vector of robot), we get the transformation from MCS frame to robot base frame $\boldsymbol{T_{mr}}$. By solving $\boldsymbol{AX} = \boldsymbol{XB}$ [1], the transformation from MCS frame to Kinect frame $\boldsymbol{X}$, can be obtained. Then to get the transformation from Kinect frame to robot base frame, $\boldsymbol{T_{kr}} = \boldsymbol{X^{-1}T_{mr}}$, and the pose of Kinect in robot base frame, $(x'_c, y'_c, z'_c, \phi'_c, \theta'_c, \psi'_c)$, which is used as ground truth. The calibration results based on our method and MCS are reported in Table 1.



(a) Line fitting. (b) Circle fitting.

Fig. 9. Accuracy of moving straight and turning, the data is metric. St. dev of the distance of measure points to the fitting line is 1.18 mm, st. dev of distance from points to the fitting circle's center based [13] is 1.34 mm.

Table 1. Calibration results specified in radians.

Method	Our		MCS
	Mean	St. Dev	Value
Roll	0.0166	0.0011	0.0100
Pitch	0.0580	0.0014	0.0819
Yaw	0.0129	0.0024	−0.0041

5.2 Robot Moving Error

The series of tests have been conducted to evaluate the accuracy of the movements of the robot. Four different of movements have been measured:

- Distance between target and stop position when moving in a straight line. We measured the moving ability by moving $d \in \{10\,\text{cm}, 15\,\text{cm}, 20\,\text{cm}, 25\,\text{cm}, 30\,\text{cm}\}$, and the largest bias is less than 5 mm and largest standard deviation is less than 0.6 mm.
- Rotation. The mean values are almost has $-2.5°$ bias, so we add a fixed value, $2.5°$, to each rotation value.
- Pose repeatability of arm. In our experiments, the arm pose repeatability is very strong, the st. dev in six DoF are very small, so in the grasping system, this factor can be ignored.
- Elevator up and down. This error is very subtle and can be ignored in the system.

5.3 Grasping

Using $bias + 3st.dev$ as the max error, and the grasping system is validated with robot Kejia, each test start position is different, 93 attempts have been successfully finished in the 100 time grasping attempts. The policy also attempted manipulation task successfully during the RoboCup@Home 2016, Kejia had finished 3 grabbing and placing in three minutes in *Manipulation and Object Recognition, 1st place* in this challenge.

Many teams did not pick any object successfully, we attributed the reason to their grasping policy is based on high-precision robot-sensor calibration methods that have the following drawbacks: (1) extra measuring devices are needed to be delivered to the venue; (2) data collection is time-consuming; (3) except few of them, these techniques are based laboratory environments. In light of our observation, no team tried to do the high-precision robot-sensor calibration due to transport, time, and environmental issues. While after long-distance transportation or reassembly, slight changes or drifts were caused by wear of parts, dimensional drifts, and tolerances, and all of them need a new calibration.

6 Conclusions

We have presented a calibration error tolerance grasping system and a simple calibration approach to calibrate the camera to the robot. Our results with more than 100 attempts, validated with a mobile robot, show that policy and the calibration approach can reduce the work of calibration and work well for grasping.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (Grant No. 61573333), the Joint Funds of the National Natural Science Foundation of China (Grand No. U1613216) and the Fundamental Research Funds for the Central Universities (Grant No. 3022016WK6030000078).

References

1. Tsai, R.Y., Lenz, R.K.: A new technique for fully autonomous and efficient 3D robotics hand/eye calibration. IEEE Trans. Robot. Autom. **5**(3), 345–358 (1989)
2. Shiu, Y.C., Ahmad, S.: Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form AX = XB. IEEE Trans. Robot. Autom. **5**(1), 16–29 (1989)
3. Wang, H., Lu, X., Hu, Z., Li, Y.: A vision-based fully-automatic calibration method for hand-eye serial robot. Ind. Robot Int. J. **42**(1), 64–73 (2015)
4. Daniilidis, K.: Handeye calibration using dual quaternions. Int. J. Robot. Res. **18**(3), 286–298 (1999)
5. Dornaika, F., Horaud, R.: Simultaneous robot-world and hand-eye calibration. IEEE Trans. Robot. Autom. **14**(4), 617–622 (1998)
6. Heller, J., Havlena, M., Pajdla, T.: A branch-and-bound algorithm for globally optimal hand-eye calibration. In: IEEE Conference on Computer Vision and Pattern Recognition, Providence, pp. 1608–1615 (2012)
7. Horn, B.: Closed-form solution of absolute orientation using unit quatemions. JOSA A **4**(4), 629–642 (1987)
8. Schonemann, P.: A generalized solution of the orthogonal Procrustes problem. Psychometrika **31**, 1–10 (1966)
9. Voruganti, A.K.R., Bartz, D.: Alternative online extrinsic calibration techniques for minimally invasive surgery. In: Proceedings of the 2008 ACM Symposium on Virtual Reality Software and Technology, VRST 2008, pp. 291–292. ACM (2008)
10. Chen, E.C.: An augmented reality platform for planning of minimally invasive cardiac surgeries. Proc. SPIE Med. Imaging Int. Soc. Opt. Photon. **8316**, 831617 (2012)
11. Levine, S., Pastor, P., Krizhevsky, A., Quillen D.: Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. In: International Symposium on Experimental Robotics (ISER) (2016)
12. De la Escalera, A., Armingol, J.M.: Automatic chessboard detection for intrinsic and extrinsic camera parameter calibration. Sensors **10**(3), 2027–2044 (2010)
13. Halir, R., Flusser, J.: Numerically stable direct least squares fitting of ellipses. In: Proceedings of International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, pp. 125–132 (1998)

14. Shah, M., Eastman, R.D., Hong, T.: An overview of robot-sensor calibration methods for evaluation of perception systems. In: Workshop on Performance Metrics for Intelligent Systems, pp. 15–20 (2012)
15. Roth, Z., Mooring, B., Ravani, B.: An overview of robot calibration. IEEE J. Robot. Autom. **3**(5), 377–385 (1987)
16. Rusu, R.B., Cousins, S.: 3D is here: Point Cloud Library (PCL). In: IEEE International Conference on Robotics and Automation, Shanghai, pp. 1–4 (2011)
17. Fiala, M.: ARTag, a fiducial marker system using digital techniques. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), pp. 590–596 (2005)
18. Wasenmüller, O., Stricker, D.: Comparison of kinect V1 and V2 depth images in terms of accuracy and precision. In: Chen, C.-S., Lu, J., Ma, K.-K. (eds.) ACCV 2016. LNCS, vol. 10117, pp. 34–45. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-54427-4_3
19. Zhang, Z.: A flexible new technique for camera calibration. IEEE Trans. Pattern Anal. Mach. Intell. **22**(11), 1330–1334 (2000)
20. http://optitrack.com