Fast Calibration of Industrial Mobile Robots to Workstations using QR Codes

Rasmus S. Andersen[1], Jens S. Damgaard[1], Ole Madsen[1], Thomas B. Moeslund[2]
[1]Dep. of Mechanical and Manufacturing Engineering, Aalborg University, Denmark, {rsa,cs,om}@m-tech.aau.dk
[2]Dep. of Architecture, Design and Media Technology, Aalborg University, Denmark, tbm@create.aau.dk

***Abstract*--Much research is directed at developing increasingly efficient and flexible production, and one important potential advancement is the robots known as *Autonomous Industrial Mobile Manipulators* (AIMMs). The idea behind AIMMs is to have robots that can move around and have the ability to perform a wide variety of tasks, while at the same time being fast and easy to configure and program. When an AIMM moves from one workstation to another, it is essential that it is able to calibrate its position with respect to that new station. In this paper, a new and fast calibration method based on QR codes is proposed. With this QR calibration, it is possible to calibrate an AIMM to a workstation in 3D in less than 1 second, which is significantly faster than existing methods. The accuracy of the calibration is ±4 mm. The method is modular in the sense that it directly supports integration and calibration of a new camera. The calibration have been implemented on Aalborg University's AIMM, *Little Helper* [8,9], and tested both in a laboratory and in a real-life industrial environment at the Danish pump manufacturer Grundfos A/S.**

***Index Terms*—Calibration, QR code, industrial mobile manipulator**

I. Introduction

Globalization has for decades moved manufacturing jobs from western countries to low-wage developing countries. This has put pressure on both wages and the productivity of production in the industrialized countries. One efficient way of increasing productivity and save jobs is to automate production by using robots.

Traditional industrial robots are not an option for all production companies, however. A robotic, automated production line is a major investment, and configuration of robots to perform the required operations is a time consuming task, that require highly specialized engineers. Thus, installation of a new, fully automated production line can only be justified if the quantity of similar items to be produced is large. Robots have therefore proven to be particularly useful for industries such as car manufacturing, where a large quantity of identical products have to be produced. For industries and companies that produce smaller product series, it cannot be justified to acquire and setup full automation. This is the case both for small and mid-sized companies, but also for larger companies that operate in shifting markets, where it is necessary to constantly change and develop the product lines.

An attempt to develop robots that are better suited for such production scenarios is the *Autonomous Industrial Mobile Manipulator* (AIMM). An AIMM is a robot which is able to move around on its own and perform tasks such as transportation, part feeding and assembly operations. Thus, an AIMM should not be considered as a traditional industrial robot, but instead as tool; providing human workers the possibility to be more efficient. Like other tools, AIMMs should not require the environment to be changed dramatically.

An AIMM consists in general of four elements: A moving platform, a manipulator, a tool (e.g. a gripper), and sensors. Industrial manipulators are in general able to move with a very high accuracy, while platforms are dependent on wheels, surfaces, and sensors, and cannot be expected to have a similar degree of accuracy. When an AIMM is first taught the location of a workstation and the tasks it must perform there, it has a certain position and orientation, let's say $\boldsymbol{x}$. When the AIMM later arrives autonomously at that same workstation to perform its task, the limited precision of the platform will most likely cause it to stop not *exactly* at $\boldsymbol{x}$. Instead, a small error $\boldsymbol{\delta}$ will be introduced; causing the actual position and orientation of the AIMM to be $\boldsymbol{x} + \boldsymbol{\delta}$. When the AIMM subsequently uses machinery and items at the workstation, it is necessary to compensate for $\boldsymbol{\delta}$ to utilize the high precision of the manipulator.

In this paper, a fast calibration technique for AIMMs based on QR codes is presented. Related research on calibration techniques for AIMMs are presented in Section 1.A, and in Section 2, an algorithm for calibrating an AIMM to a QR code is presented. In Section 3 it is described how this algorithm can also be used to calibrate a camera with respect to the AIMM, it is mounted on. Results from laboratory tests and from a test in a factory at the Danish pump manufacturer Grundfos A/S are described in Section 4, and finally in Section 5, conclusions are drawn and limitations and possible future developments are discussed.

A. Related Research

Since AIMMs must have the ability to move between workstations, calibration to new workstations is a particular useful aspect, which has received some attention in the literature. In 2000, a general method for camera calibration was developed by Z. Zhang et. al. [13]. This has later become extremely popular, especially due to a free implementation provided for Matlab in the Matlab Camera Calibration Toolbox. This is designed specifically for calibrating cameras, and not directly applicable for calibration of robots, however.

Most research within calibration of industrial robots has been concerned with calibration of the robot (manipulator) itself; e.g. estimating its kinematic parameters [12,2,1,10]. On the other hand, most research within positioning of mobile robots has been directed at making navigation possible [3,6,5]. In [3], methods for positioning of mobile robots are categorized. The method proposed in this paper is in the category *artificial landmarks*, and one interesting existing method is from [5]. Here, a small mobile robot uses a RGB-camera to calibrate itself to a 3D marker with a modified POSIT algorithm, and a mean accuracy of ±1.6 cm is achieved. This is not particularly useful for AIMMs though; partly because 3D markers are impractical in an industrial environment, and partly because the accuracy is relatively low. This is generally the case for algorithms developed for navigation.

Two approaches from [7] are specifically developed for AIMMs. Both approaches make use of 2D markers, and the first approach works by capturing multiple images using a camera mounted on the end-effector of the robot. This provides a precise calibration of ±0.1 mm. In the second approach, a laser for distance measurements is mounted on the end-effector in addition to the camera. This makes it possible to reduce the execution time at the expense of the accuracy, which is lowered to ±1.0 mm. Both methods have disadvantages: In the precise approach, a large number of images of a calibration board are captured, and the execution time is about 60 seconds. If the robot is moving frequently between workstations, such non-productive time must be minimized. The second approach is faster because only one image needs to be captured, but it is also less precise, and it requires more equipment on the end-effector. More equipment on the end-effector means that the robot can carry less payload, and additional equipment must therefore be avoided if possible.

A last approach, described in [11], applies haptic rather than vision based calibration. This approach is able to calibrate very precisely in about 30 seconds by measuring locations on the workstation in three orthogonal directions. The disadvantage with this method is, in addition to the relatively long execution time, that all workstations are required to have large stable surfaces in all three directions. Also, it can only be used on robots that are able to measure external forces.

II. Fast QR Calibration to a Workstation

We propose to do calibration using traditional two-dimensional QR codes. A QR code is placed at each workstation in a way that is visible to the robot, as illustrated in Fig. 1. Before the robot starts production, it must have learned where the QR codes are positioned relative to the workstation. When the robot arrives at a new workstation, it takes an intensity (monochrome) and a depth picture of the QR code and uses these to determine the relative pose of the QR code. This makes it possible for the robot to calculate its own position relative to the workstation, and the approach requires minimal changes to the environment. The only requirement is that a relatively large QR code must be placed at all workstations.

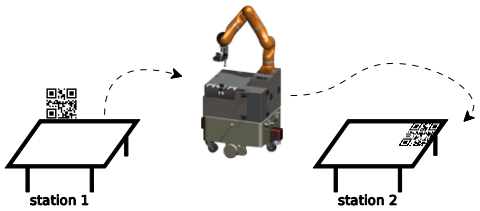

Fig. 1. A QR code is fixed at every workstation. Using this, the robot can calibrate itself to each station.

The choice of QR codes instead of traditional calibration boards (e.g. checkerboards), provides several advantages. Firstly, QR codes can hold information which can ensure that the correct QR code is used. Secondly, it is possible to distinguish the corners of a QR code, which can prevent errors. The corners of both a QR code and a checkerboard can be estimated very precisely, but an advantage with checkerboards is, that all corners inside the board are similarly well distinguishable. This is a necessary when working with uncalibrated cameras, but when the intrinsic parameters of the camera have already been calibrated properly, the advantage is small.

A. Estimation of the Coordinate System of a QR Code

An AIMM needs to operate in three dimensions. It is therefore necessary to determine the position and orientation of the QR code in 3D, which is illustrated in Fig. 2. For this to be possible, a depth camera is an advantage. The camera must be calibrated with respect to its intrinsic parameters, and the intensity and depth images must be calibrated with respect to one another. Beside from this, the type of camera is not important for the algorithm, and for the laboratory tests documented here, a Primesense Carmine 1.09 was used. This works much like the well-known Microsoft Kinect, only it is able to detect depths much closer to the camera.

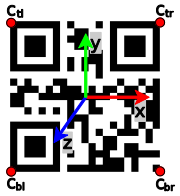

Fig. 2. Corners and coordinate system of a QR code.

When a calibration is initiated, the intensity images are initially searched for QR codes. There are several libraries available that provide this functionality, and here *zbar* is chosen, because this directly provides the location of the corners of QR codes in the images. When a QR code has been found, all 2D points inside the code in the image plane are extracted. This is done by dividing the code into two triangles, and for each of the triangles calculate whether each point is inside or outside. The process can be speeded up by omitting all points with an x- and y-value larger or smaller than the largest/smallest value of the QR-code corners.

Using the extracted points, a 3D plane can be estimated. A popular approach to rapid detection of planes is to use RANSAC, which provides a very fast

estimation while suppressing noise. However, this is not well suited for this application, because the Primesense camera provides discretized depth values. Thus, if the angle of the QR code compared to the camera is very small (the QR code is almost facing the camera), very few depth values might be present on the QR code, and RANSAC would risk using only one of these values. Instead, traditional Least Squares plane fitting is used. To eliminate noise, a number of iterations are executed, each omitting points far from the previous solution. When the process has converged, the plane normal is used as the $\boldsymbol{z}$-direction of the QR code's coordinate system. The remaining axes can be calculated from the corners of the QR code:

$$\boldsymbol{x}_{\text{initial}} = \text{norm}\big((c_{\text{tr}} - c_{\text{tl}}) + (c_{\text{br}} - c_{\text{bl}})\big) \quad (1)$$

$$\boldsymbol{y}_{\text{initial}} = \text{norm}\big((c_{\text{tr}} - c_{\text{br}}) + (c_{\text{tl}} - c_{\text{bl}})\big) \quad (2)$$

$$\boldsymbol{x}_{\text{avg}} = \text{norm}(\boldsymbol{x}_{\text{initial}} + \boldsymbol{y}_{\text{initial}} \times \boldsymbol{z}) \quad (3)$$

$$\boldsymbol{y} = \text{norm}\big(\boldsymbol{z} \times \boldsymbol{x}_{\text{avg}}\big) \quad (4)$$

$$\boldsymbol{x} = \text{norm}(\boldsymbol{y} \times \boldsymbol{z}) \quad (5)$$

where c_{tl}, c_{tr}, c_{bl}, and c_{br} are the location of the top-left, top-right, bottom-left, and bottom-right corner of the QR code, respectively, and 'norm' normalizes the vectors to a length of one. The repeated calculations of $\boldsymbol{x}$ and $\boldsymbol{y}$ are necessary because the initial axes might not be exactly orthogonal. This approach in effect forces $\boldsymbol{x}$ and $\boldsymbol{y}$ to be on the plane that has $\boldsymbol{z}$ as a normal. Equation (3) makes sure that $\boldsymbol{x}_{\text{initial}}$ and $\boldsymbol{y}_{\text{initial}}$ are weighted approximately equal when determining the final, orthogonal $\boldsymbol{x}$ and $\boldsymbol{y}$ axes.

B. Transformation to Robot Coordinate System

The transformation from the QR code to the robot base frame can be determined as:

$$^{\text{B}}_{\text{QR}}T = {}^{\text{B}}_{\text{C}}T \cdot {}^{\text{C}}_{\text{QR}}T \quad (6)$$

where $^{\text{B}}_{\text{C}}T$ is the (fixed) transformation from the camera to the robot's base, and $^{\text{C}}_{\text{QR}}T$ is the transformation from the QR code to the camera.

The transformation $^{\text{B}}_{\text{C}}T$ is determined when the camera is calibrated to the robot (described in Section 3) and $^{\text{C}}_{\text{QR}}T$ can be determined from the QR code's coordinate system from Equations (1) through (5). The transformation consists of a translation $\boldsymbol{t}_{\text{QR}}$ and a rotation $\boldsymbol{r}_{\text{QR}}$. The translation is in the $\boldsymbol{x}$- and $\boldsymbol{y}$-direction defined as the center of the QR code's corners; and in the $\boldsymbol{z}$-direction it is found by insertion in the QR plane equation found by Least Squares. This should minimize the variance compared to averaging the $\boldsymbol{z}$-values of the corners.

The rotation $\boldsymbol{r}_{\text{QR}}$ can be described using Euler angles, and these can be calculated directly from the $\boldsymbol{x}$-, $\boldsymbol{y}$, and $\boldsymbol{z}$-vectors, that define the coordinate system. This gives the full transformation $^{\text{C}}_{\text{QR}}T$, which makes it possible to calculate $^{\text{B}}_{\text{QR}}T$. Using $^{\text{B}}_{\text{QR}}T$, all positions that are defined with regards to the QR code can be transformed to the robots base:

$$^{\text{B}}P = {}^{\text{B}}_{\text{QR}}T \cdot {}^{\text{QR}}P \quad (7)$$

where $^{\text{QR}}P$ and $^{\text{B}}P$ is a point given in the the camera and base coordinate systems, respectively. Definition of the points in the QR coordinate system is done by the inverse of Equation (7). The exact sequence of the calibration is given in Table 1.

(1) **WHILE** *correct QR code not found*
- Capture a synchronized set of intensity and depth images.
- Search for QR code in the intensity image
- Read QR code
- **IF** *all corners have depth values* **THEN** correct QR code has been found!

(2) Extract all 3D points inside the QR code.
(3) Fit a plane to the points.
(4) Define a coordinate system at the QR code as in Equations (1)-(5).
(5) Calculate the **translation** of the QR code $\boldsymbol{t}_{\text{QR}}$ as the mean of the corners; adjusted in the $\boldsymbol{z}$-direction to be on the fitted plane.
(6) Calculate the **rotation** of the QR code's coordinate system $\boldsymbol{r}_{\text{QR}}$ in Euler angles.
(7) Combine $\boldsymbol{t}_{\text{QR}}$ and $\boldsymbol{r}_{\text{QR}}$ into a transformation matrix, $^{\text{C}}_{\text{QR}}\boldsymbol{T}$.
(8) Calculate the transformation from the QR code's coordinate system to the robots base coordinate system, $^{\text{B}}_{\text{QR}}\boldsymbol{T}$, as in Equation (6), and extract the translation and rotation.
(9) Repeat from (1) until the algorithm has run 10 times.
(10) **FOR** *each of the 6 degrees of freedom* discard the largest and smallest value, and take the mean of the rest.
(11) Combine the means in the final transformation, $^{\text{B}}_{\text{QR}}\boldsymbol{T}$.

Table 1. Algorithm for execution of calibration based on QR codes.

III. CALIBRATION OF CAMERA TO ROBOT

To make it possible to use a camera to calibrate to the external environment, the camera must first be calibrated to the robot. The necessary transformation between the robot base and the camera is $^{\text{B}}_{\text{C}}T$ (see Equation (6)). The procedure for calculation of this is illustrated in Fig. 3, where the desired transformation is indicated by the solid gray arrow. The tool is holding a calibration plate with a QR code attached. The dimensions of the plate are known, including the way it can be held by the tool. This means that all of the transformations indicated by dashed arrows can be determined:

- The transform between QR code and tool ($^{\text{T}}_{\text{QR}}T$) are known through the geometry of the calibration plate.
- The transform between base and tool ($^{\text{B}}_{\text{T}}T$) is known from the robot's kinematic model.
- The transform between camera and QR code ($^{\text{C}}_{\text{QR}}T$) can be determined using the calibration algorithm

The transformation between the robot base and the camera, $^{\text{B}}_{\text{C}}T$, can be estimated by combining all three known transformations:

$$^{\text{B}}_{\text{C}}T = {}^{\text{B}}_{\text{T}}T \cdot {}^{\text{T}}_{\text{QR}}T \cdot \left({}^{\text{C}}_{\text{QR}}T\right)^{-1} \quad (8)$$

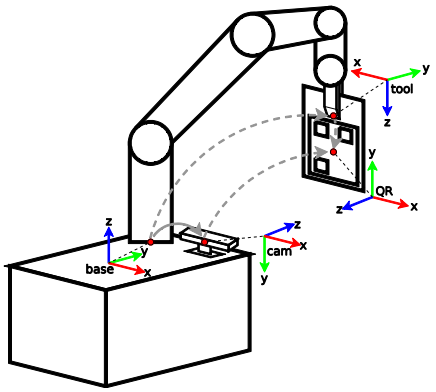

Fig. 3. All coordinate systems involved in calibration of the camera to the robot base frame.

The sequence for calibrating the camera to the robot is summed up in Table 2.

(1)	The tool moves automatically to a predefined horizontal position, where the calibration board can easily be put into place.
(2)	The board is manually put into place.
(3)	The tool closes on the calibration plate.
(4)	The tool (now holding the plate) is manually moved to a position where the plate is clearly visible to the camera.
(5)	The pose of the QR code is estimated, and the camera is calibrated to the robot using Equation (8).

Table 2. Sequence for calibration of the camera to the robot.

IV. Results

The QR calibration has been implemented on Aalborg University's AIMM, *Little Helper* [8,9]. The performance of the algorithm has been measured through laboratory tests described in Section IV.A and IV.B, and in Section IV.C, the performance of the algorithm is compared to existing methods. The usefulness of the algorithm has additionally been verified in a real-life industrial environment at a factory owned by Grundfos A/S, and this is described in Section IV.C.

A. Calibration of Camera

The camera was repeatedly calibrated as described in Table 2 and the variation in the results is shown in Fig. 4.

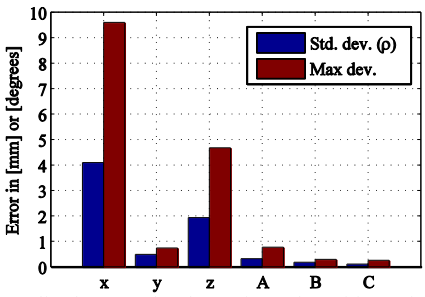

Fig. 4. Normalized errors in the estimated position of the camera relative to the robot. The first three columns are the translations given in mm, and the last three are the rotations in ZYX Euler angles given in degrees.

As it can be seen from the Figure, the translations were within ±10 mm for all directions, and the rotation was within ±1° for all angles. The variation was largest in the ***x***-direction, which is also the direction where the distance is largest; cf. the base coordinate system in Fig. 3. A total of 11 calibrations were carried out, and the average execution time was 33.9 seconds with a standard deviation of 12.2 seconds. This includes all steps described in Table 2.

B. Calibration to Workstation

The accuracy of the calibration to a workstation was measured repeatedly using the test setup shown in Fig. 5. First, the camera was calibrated to the AIMM using one of the calibrations from Section IV.A. Then, the robot was taught the position of each of the three points relative to the QR code. In each test, the robot then had to (re-)estimate the pose of the QR code and indicate the position of the points. The reason that three points were used is, that this makes it possible to evaluate the accuracy in both the translational and rotational dimensions. The positions indicated by the robot were finally measured manually in both the x-, y-, and z-directions with a precision of 0.5 mm. The distance in the z-direction was measured as the distance between the tool and the board.

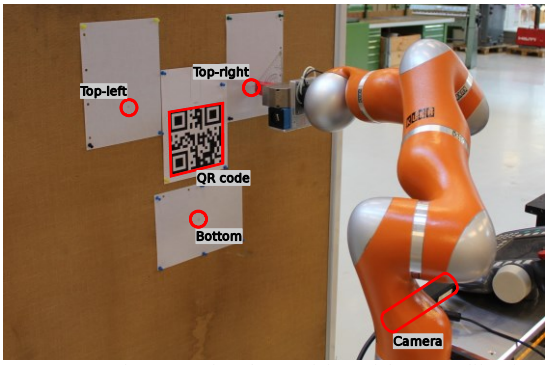

Fig. 5. Test setup for measuring the precision of the QR calibration. The points are all located exactly 23 cm from the center of the QR code, and the angles between the center and each of the points are 120°. The distance between the AIMM and the board is approximately 75 cm.

Two sets of measurements were carried out. In the first, the position of the robot relative to the board was static, and these tests measure the repeatability of the calibration. A total of 10 static tests were carried out. In the second, the relative position was variable; it was changed by moving the board between each calibration. The position of the board was changed up to ±15 cm sideways (four tests), 5 cm forwards (one test), and ±9.2° turned (two tests).

The normalized positions in the x- and y-direction are shown in Fig. 6 for the static tests and in Fig. 7 for the variable tests. It can be seen, that all errors are within ±1.1 mm for the static tests and within ±4 mm for the variable tests. In some cases, several tests yielded the same results (within the measurement precision), and this causes some dots in the Figures to be on top of each other. The results in the z-direction are not shown, but these were also within ±1.1 and ±4.0 mm for the static and variable tests, respectively.

Based on the measurements that are (partly) illustrated in Fig. 6 and 7, the normalized errors in the estimated co-

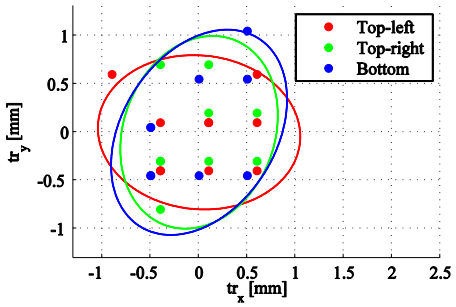

Fig. 6. Normalized errors in the x- and y-direction for the three points shown in Fig. 5 with a *static* position of the robot relative to the board. The ellipses illustrate the shape of the covariance matrix for these two directions, when a multivariate Gaussian distribution has been fitted for each of the three points. All errors are within ±1.1 mm.

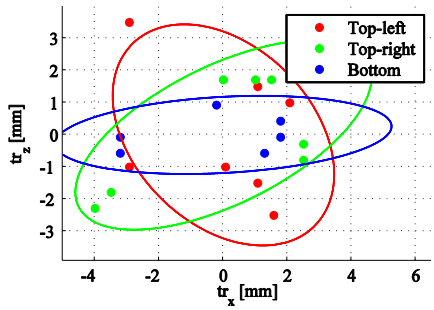

Fig. 7. Normalized errors in the x- and y-direction for the three points shown in Fig. 5 with a *variable* position of the robot relative to the board. The ellipses illustrate the shape of the covariance matrix for these two directions, when a multivariate Gaussian distribution has been fitted to each of the three points. All errors are within ±4 mm.

ordinate system can be calculated. These are shown in Table 3. The translation is defined with regards to the center of the QR code, given by the mean of the three points, and the rotation around each axis is defined by the plane spanned by the three points.

Type	Direction	Std. dev. (ρ)	Max error
Static	x	0.30 mm	0.57 mm
	y	0.30 mm	0.55 mm
	z	0.31 mm	0.77 mm
	θ_x	0.07º	0.11º
	θ_y	0.07º	0.10º
	θ_z	0.90º	1.60º
Variable	x	2.35 mm	3.38 mm
	y	0.34 mm	0.55 mm
	z	0.58 mm	1.12 mm
	θ_x	0.14º	0.23º
	θ_y	0.50º	0.76º
	θ_z	0.23º	0.39º

Table 3. Normalized translational and rotational errors for the static case (see Fig. 6) and for the variable case (see Fig. 7). The translational error is defined with regards to the center of the QR code.

The error is in the static tests significantly larger than in the variable tests, as could be expected. For the variable tests, the translational errors are within ±4 mm, and the angular errors are within ±1º. The duration of the calibration averaged around 0.9 second, and it was in all cases below 1 second.

C. Comparison to Existing Calibration Methods

In Table 4, the proposed calibration is compared to existing methods implemented at AAU. Published methods for traditional industrial robots and methods developed for navigation of mobile robots have very different requirements, and therefore only methods developed specifically for AIMMs are included. Since none of these methods include calibration of camera to AIMM, only calibration of AIMM to workstation is included here. The previous vision based methods were evaluated by repeated parking of the AIMM, followed by calibration and indication of one 2D reference point. The most comparable measurements performed of the proposed method is that of the position of the points in the variable case, illustrated in Fig. 7. Therefore, this is included for comparison.

Method	Duration	Precision
Haptic	30-45 sec	±1.0 mm
High speed	10 sec	±1.0 mm
High precision	60 sec	±0.1 mm
Proposed method	<1 sec	±4.0 mm

Table 4. Comparison of calibration methods. For the proposed method, the precision is for the estimation of the point positions for the variable case, illustrated in Fig. 7, since this is most comparable to the evaluation method used in [7].

It is clear from comparison, that the proposed method is less precise than previous methods, but instead at least 10 times faster; mostly due to the fact, that movement of the camera is not required between capturing images.

D. Application in an Industrial Environment

In addition to laboratory tests, the practical performance of the QR calibration has been verified in a real-life industrial environment at a factory owned by Grundfos A/S. The setup is shown in Fig. 8. The task of the robot was at this workstation to identify and pick up metal cylinders, located on the conveyor belt behind the robot. This is, however, a very difficult task as long as the conveyor belt is moving. Therefore, the robot should first turn off the conveyor belt using the switch that is indicated on the Figure. After the robot is done picking up cylinders, it should turn on the conveyor belt before leaving. The purpose of the QR calibration is to make it possible for the robot to operate the switch, even when the position of the robot varies. The precision proved to be sufficient to make this possible at every attempt.

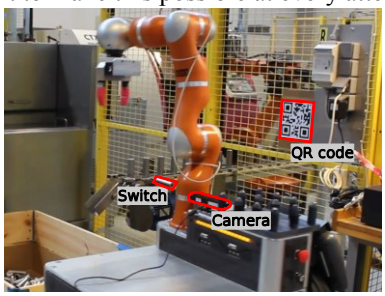

Fig. 8. Fast QR calibration using the Asus Xtion camera. The camera detects the position and orientation of the fixed QR code, and subsequent movements with the robot are corrected accordingly.

V. Discussion

The implemented calibration makes it possible to perform a very fast calibration in 3D compared to existing methods. This is especially important for AIMMs that are moving frequently between workstations. The precision is sufficient to perform the experiments

described here, but for tasks requiring a very high precision, it will be insufficient. Compared to existing methods, the execution time is reduced by (at least) a factor of 10, while the precision is reduced from ±1.0 mm to ±4.0 mm.

The same QR calibration algorithm is used to calibrate the camera to the robot, and this calibration can be completed in less than 1 min. The only requirement is a simple calibration board with known dimensions.

Four factors limit the overall calibration precision:

(1) The discretized position estimate of the corners of the QR code. The positions are in the current implementation determined with pixel accuracy in VGA resolution. The camera used for the laboratory tests described in this paper has a FoV of 57.5×45 degrees, and at a distance of 75 cm, as used in the laboratory tests, this gives discretized values with 1.2 mm apart in both the x- and y-direction. This corresponds to a precision for each QR code corner of ±0.6 mm. The fact that four corners and 10 averaging runs are used, might improve the precision to some degree.

(2) The precision in the depth values. Both of the depth sensors that were used have the same depth sensor as the Microsoft Kinect, and the absolute precision of the depth values provided by the Kinect has been shown to be within ±10 mm for distances between 0.8 m and 3.5 m when used indoor [4]. This is partially due to the fact, that the depth values are stored in 11 bits, and the resolution decreases logarithmic with the distance. No data are available on the relative repeatability error, but it has proven to be significantly smaller.

(3) The intrinsic calibration of the camera; both for intensity and depth images.

(4) The accuracy of the manipulator.

The first three of these factors are directly related to the physical RGB-D camera, and the precision could thus be improved by using a camera with better resolution and precision. The development of such cameras for reasonable prices is currently fast, and the Kinect 2 is one obvious possibility.

It should be noted that the existing vision based calibration methods mentioned only calibrates in 2D, while the proposed method calibrates in 3D. The extra dimension makes calibration possible, even if the floor cannot be assumed to be flat. If the floor *can* be assumed to be flat, this is an unnecessary complication, and it will be possible to increase the precision by forcing the y-axis of the QR code to be parallel with the vertical axis of the robot's base.

The fact that QR codes are used instead of traditional camera calibration shapes (e.g. checkerboards), gives rise to a number of potential uses. QR codes can hold information, and an AIMM can therefore distinguish between different markers. Thus, a number of QR codes can be placed close to one another, without any risk of detecting and calibrating to a wrong marker. Therefore, it is possible to calibrate to different areas or machines located close by. Also, if QR codes are placed on boxes or other items, the pose estimation algorithm can be used to locate and pick up desired objects.

ACKNOWLEDGMENT

This research was partially funded by the European Union under the Seventh Framework Programme project 260026 TAPAS - Robotics-enabled logistics and assistive services for the transformable factory of the future. The developed calibration method was tested in an industrial environment at a factory owned by the Danish company Grundfos, and this was only possible due to a close collaboration.

REFERENCES

[1] Gürsel Alici and Bijan Shirinzadeh. A systematic technique to estimate positioning errors for robot accuracy improvement using laser interferometry based sensing. Mechanism and Machine Theory, 40(8):879-906, 2005.

[2] David J Bennett, Davi Geiger, and John M Hollerbach. Autonomous robot calibration for hand-eye coordination. The International journal of robotics research, 10(5):550-559, 1991.

[3] Johann Borenstein, HR Everett, Liqiang Feng, and David Wehe. Mobile robot positioning-sensors and techniques. Technical report, DTIC Document, 1997.

[4] RA El-laithy, Jidong Huang, and Michael Yeh. Study on the use of Microsoft Kinect for robotics applications. In Position Location and Navigation Symposium (PLANS), 2012 IEEE/ION, pages 1280-1288. IEEE, 2012.

[5] Tommaso Gramegna, Lea Venturino, Grazia Cicirelli, Giovanni Attolico, and Arcangelo Distante. Optimization of the posit algorithm for indoor autonomous navigation. Robotics and Autonomous Systems, 48(2):145-162, 2004.

[6] Sergio Hernandez, Carlos A Morales, Jesus M Torres, and Leopoldo Acosta. A new localization system for autonomous robots. In Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on, volume 2, pages 1588{1593. IEEE, 2003.

[7] Mads Hvilshøj, Simon Bøgh, Ole Madsen, and Morten Kristiansen. Calibration techniques for industrial mobile manipulators: Theoretical configurations and best practices. In Robotics (ISR), 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK), pages 1-7. VDE, 2010.

[8] Mads Hvilshøj and Simon Bøgh. "Little Helper" – an autonomous industrial mobile manipulator concept. International Journal of Advanced Robotic Systems, 8(2), 2011.

[9] Mads Hvilshøj, Simon Bøgh, Ole Madsen, and Morten Kristiansen. The mobile robot "Little Helper": Concepts, ideas and working principles. In ETFA, pages 1-4, 2009.

[10] Yan Meng and Hanqi Zhuang. Autonomous robot calibration using vision technology. Robotics and Computer-Integrated Manufacturing, 23(4):436-446, 2007.

[11] M. Rath Pedersen. Integration of the kuka light weight robot in a mobile manipulator. Master's thesis, Aalborg University, 2011.

[12] ZVIS Roth, B Mooring, and Bahram Ravani. An overview of robot calibration. Robotics and Automation, IEEE Journal of, 3(5):377-385, 1987.

[13] Zhengyou Zhang. A flexible new technique for camera calibration. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 22(11):1330-1334, 2000.