Human Assisted Computer Vision on Industrial Mobile Robots

Rasmus S. Andersen * **Casper Schou** * **Jens S. Damgaard** *
Ole Madsen * **Thomas B. Moeslund** **

* *Dep. of Mechanical and Manufacturing Engineering, Aalborg University, Denmark*
** *Dep. of Architecture, Design and Media Technology, Aalborg University, Denmark*

ABSTRACT

Much research is directed at developing increasingly efficient and flexible production, and one important potential advancement is *Autonomous Industrial Mobile Manipulators* (AIMM's). The idea behind AIMM's is to have robots that have the ability to perform a wide variety of tasks, and which can easily and efficiently be reconfigured when the requirements changes. In this paper, the paradigm of *skill based* programming is investigated, and in particular how computer vision abilities can be integrated in this. Three applications of computer vision developed in a skill based framework are presented; namely vision pick, quality control, and fast calibration. All three are implemented on Aalborg University's AIMM, *Little Helper*, and tested in a real-life industrial environment at the Danish company Grundfos A/S.

1. INTRODUCTION

The globalization has for several 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 is to automate production by using robots. A major limitation for the application of robots is, however, the scale of production. Construction of an automated production line is a major investment, and configuration of robots to perform the required operations is a time consuming task, that must be performed by highly specialized engineers. Thus, installation of a new, fully automated production line can only be justified if the quantity of identical items to be produced is very large. Robots have therefore proven to be particularly useful in industries such as in the car manufacturing industry, where a large quantity of identical products have to be produced.

For many kinds of production, the amount of identical items is, however, not large enough to justify investment in automated robotic production lines. Much research have therefore been directed towards developing more flexible types of automation. The organization *European Robotics Technology Platform* (EUROP) published in 2009 a *Strategic Research Agenda for European Robotics* (SRA), which outlined areas that European robotics research should focus on as well as metrics for each area EUROP (2009a,b). The core requirements for future robotics include:

- Reconfigurability: It must be possible to reconfigure both robots and other production hardware fast and easy, to prevent expensive idle time for long periods between production of (possibly small) production series.
- Human Robot Interaction: The communication between robot and human operators must be intuitive and to an increasing degree use languages and interfaces that are natural to humans.
- Autonomy (ability to function in less structured environments): On a car manufacturing plant, robots work in highly structured environments, virtually without any human presence at all. This approach is not sufficient for smaller production series. Thus, the robots must have a larger degree of autonomy, enabling them to perform tasks in dynamic environments, that cannot be precisely modeled before production begins.

One type of industrial robot that is well suited for such a production scenario, is the *Autonomous Industrial Mobile Manipulator* (AIMM). Although AIMM's are not yet in industrial use, they are already able to move autonomously around in changing environment and perform a wide variety of tasks. Since AIMM's must be designed to function in less than fully structured environments, they depend very much on their ability to sense both objects and the world around them. The focus of this paper is to investigate methods to do such sensing by using computer vision in a fast and easily reconfigurable way.

1.1 Related Research

Ordinary RGB cameras are used to give vision functionality to robots in various fields, including navigation, object manipulation, and interaction with humans, cf. Hvilshøj et al. (2009); Guizzo and Ackerman (2012); Nava et al. (2011). The human visual system includes additionally information about depth, and several approaches have been taken to provide this information to robots also, including stereo vision (Murray and Little, 2000), time-of-flight (TOF) depth cameras (Klank et al., 2009), and

⋆ This research was partially funded by the European Union project TAPAS under the Seventh Framework Programme.

depth cameras based on structured light (Siegwart and Nourbakhsh, 2004). While stereo vision is the closest analogue to the human visual system, it is far simpler to use active technologies such as TOF or structured light. With the launch of the Microsoft Kinect in 2010, which combines an ordinary RGB camera with a depth camera based on structured light, the price and accessibility of quality depth video imaging was all of a sudden reduced dramatically (Shotton et al., 2011; El-laithy et al., 2012). This dramatically increased the scientific interest in taking advantage of depth information in combination with RGB images for all areas, where RGB images was also previously used Tölgyessy and Hubinskỳ (2010); Benavidez and Jamshidi (2011); León et al. (2011). Recently, the smaller but equally powerful competitor *Asus Xtion Pro Live* was launched.

Since AIMM's must have the ability to move between workstation, calibration to new workstations is a particular useful aspect, which has also received some attention in the literature. In 2000, a general method for camera calibration was developed by Zhang (2000). This has later become extremely popular, due to implementations provided both for C/C++ in OpenCV and for Matlab in the Matlab camera calibration toolbox. This is designed specifically for estimating parameters, intrinsic as well as extrinsic, for cameras, and not directly applicable for calibration of robots. Another approach by Alici and Shirinzadeh (2005) calibrates industrial robots with very high precision, but this require a laser tracker to be located close to the calibration point. Thus, it is not suitable for AIMM's, which should be able to work in industrial environments without requiring extensive and/or expensive changes.

Two approaches from Hvilshøj et al. (2010) are specifically developed to AIMM's. A fast approach use in addition to a camera a laser for distance measurements, and a slower but very precise method makes only use of a camera on the tool. Both methods have, however, disadvantages: The fast approach requires that a laser is mounted on the tool of the robot. More equipment on the tool means less possible payload, and must therefore be avoided if possible. In the more precise approach, a large number of images are captured of a calibration board, and the execution time is about 60 seconds. If the robot is moving frequently between workstations, such non-productive time must be minimized.

A last approach, described in Pedersen (2011), applies haptic rather than vision based calibration. This approach is a 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 the workstation must be have large surfaces in all three directions. Also, it is only applicable on robots with force feedback control.

1.2 Skill Based Computer Vision

A traditional and widely used way of programming robots is the *Sense-Plan-Act* (SPA) paradigm (Nilsson, 1993). Using this, the robot moves between the three states: Sense, plan and act. In the sensing state, information from sensors are used to update and maintain a world model. In the planning state, high level logic plans on basis of this world model what the robot has to do, and in the acting state the plan is carried out, typically using control theory. Two limitations of the SPA paradigm is that it does not well support reusability of code, and that the complexity of maintaining a complete world model can be very high.

The paradigm of robot skills attempts to counter both limitations of the SPA paradigm by introducing a layered architecture, where each layer executes its own SPA loop. The idea of using layers to provide better possibilities for reusing code was presented as early as in 1986 by Brooks (1986), but research to provide even more reusable and more generic solutions continue, cf. Gat (1998); Bjorkelund et al. (2011). In the skill paradigm, programming is divided in three layers. Different naming conventions exist, and here the layers are named *device primitives*, *skills*, and *tasks*. The purpose of the layered programming structure is to wrap the difficult and low level robotic knowledge in the lower levels, allowing non-expert users to focus on teaching tasks on a much higher level.

In the ongoing research project 'Little Helper' at Aalborg University, AIMM's have been developed on the basis of the skill paradigm since 2008. In close collaboration with both academic and industrial partners, it is attempted to make the technology ready for industrial use.

In this paper, the integration of computer vision in the skill based framework is presented. The vision algorithms are implemented using a commercial computer vision system based on *Labview* as well as the open source library *zbar*, and the focus here is on how to integrate and use this in the skill based robotic framework. First, the skill paradigm and the vision system applied are described in detail. Subsequently, three developed applications of computer vision are described: A generic pick skill using vision, quality control integration, and a fast calibration based on recognizing QR codes. Finally it is discussed how the results can be generalized, and where future research should be directed. The results presented in the paper are from a midway demonstration in the EU project TAPAS, performed in a factory owned by the Danish company Grundfos A/S.

2. METHODS

2.1 The Concept of Robot Skills

The architecture in the skill paradigm used here consists of three layers:

(1) **Device primitives:** Basic functions of one device, such as the robot, tool or a camera. Example: *Open gripper*.
(2) **Skills:** A predefined sequence of device primitives, that form a coherent action. In Björkelund et al. (2011), a skill is defined as "productive sensor-based robot motions". Example: *Pick up object* O_1.
(3) **Tasks:** Responsible for achieving the overall goals of the robot, while at the same time completely decoupled from the internals of the robot. The robot itself can thus in principle be replaced without replacing the task layer, as long as the new robot provides the same skills. Example: *Pick up 10 units of object* O_1 *at location* L_1*, and place them in a bin at location* L_2.

It does only make sense to execute a place skill, if the robot is holding an object. This means that a precondition for a place skill is, that an object is held. In general a skill has both pre- and post-conditions, and the skill only functions if these are met. This property is in general called that skills are *situated*.

In the skill paradigm, each skill has a *teach* and an *execute* phase. If the user wants the robot to pick up an object of type O_1 from location L_1, a pick is chosen and taught. After teaching is completed, the robot is able to execute the same skill, thus picking a new object of the same type from location L_1 on its own. Prerequisites include here include that the robot is already located at (or close to) location L_1, and that an object of type O_1 is present at the location.

2.2 Flexible Setup with External Computer Vision System

The computer vision system used here is based on the Vision Builder software in Labview. It is able to perform a large number of 2D vision tests, and it employs an intuitive interface, allowing non-experts to configure it with very little training. A screen image is shown in Figure 1.

This paper is, however, not concerned with the vision system itself, but with the integration and use of vision systems in general in the skill based robotic framework. The vision system is not installed on the robot itself. Instead, a protocol based on TCP/IP has been developed, which allows the robot to communicate with an external vision system. To be able to integrate the robot in different kinds of production lines, support for both local camera (mounted on the robot) as well as external cameras has been included in the protocol. For the experiments using the vision system that are described here, two cameras are used; one placed on the tool of the robot, and one fixed at an assembly cell. Both cameras are of the type DMK 31BF03-Z2, which have motorized zoom and a resolution of 1024×768. The first camera is used to detect and pose estimate rotor caps to be able to pick them up, while the fixed camera is used to perform quality control during assembly. Each of these cases are presented in the following subsections, along with the remaining experiment; providing fast calibration by pose estimation of QR codes.

2.3 Generic Vision Pick

The developed vision pick skill consists like all skills of a training and an execution phase. In the training phase, the following parameters are taught:

Fig. 1. Vision detection system in execution mode.

Start and end position of camera: These positions define both the route the camera will take when searching for an object to pick up, as well as an *acceptance region* for objects. During execution, the robot will move the camera from the start towards the end position. With short intervals, the robot will stop and grab an image in search of an object to pick up. Whenever an object has been found, it is calculated if the object is located in the acceptance region; between the two points. If this is the case, the robot cancels the movement towards the endpoint and picks up the object instead.

Detection height: Height of the feature, that the vision system is able to detect.

Grasping height: Appropriate height for grasping the object.

All the required parameters are taught by manually moving the robot arm around, and thus no programming skills are required. The parameters are illustrated in Figure 2. The principle for execution is to first capture an image, and then try to detect the location of a particular feature on the object to pick up in this image. This 2D position can be transformed into a 3D vector from the camera's focal point to the image plane, by applying the intrinsic parameters of the camera. By extending this vector, it will ultimately intersect the plane with the (taught) detection height, as illustrated in Figure 2. If this 3D intersection point is located in the acceptance region, the object can be picked. An additional height; the grasping height, is taught during training, and this enables the robot to grasp the object on a suitable position. The algorithm for execution of the vision pick skill is described in detail in Table 1.

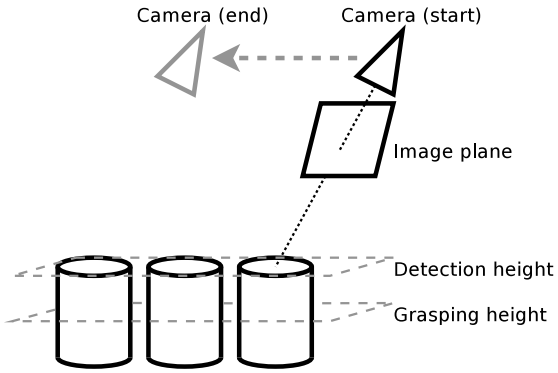


Fig. 2. Execution of the vision pick algorithm require the parameters $\boldsymbol{p}_{\text{cam,start}}$, $\boldsymbol{p}_{\text{cam,end}}$, h_{detect}, and h_{grasp} which are shown in the figure. All the parameters are specified during teaching.

The setup is shown in Figure 3 for two different locations. The Figures 3(a) and 3(c) show the robot searching for rotor caps, while Figure 3(b) and 3(d) show the robot actually picking up a robot cap. Note that the same skill is used at the different locations; only the parameters that are set during teaching differ.

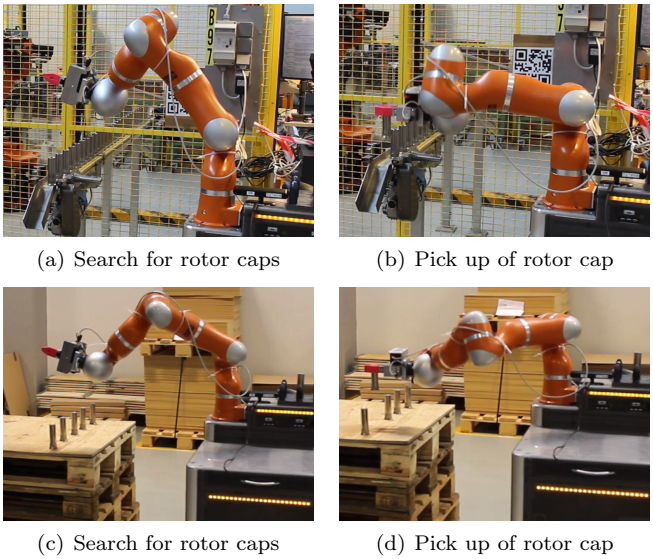
2.4 Quality Control

As with the vision pick skill, also the quality control is developed by utilizing the vision system described in Section 2.2. Thus, both cameras on the robot as well as external cameras can be used. Configuring quality control

(1) Move camera to (taught) start position, $\boldsymbol{p}_{\text{cam,start}}$.
(2) Capture image and send it to the vision system.
(3) Vision system detects object in the (2D) image, and returns this position to the AIMM.
(4) **IF** *the AIMM does not receive a valid position*
THEN
- Move camera one step along the line from $\boldsymbol{p}_{\text{cam,start}}$ to $\boldsymbol{p}_{\text{cam,end}}$.
- **IF** *the camera already was at* $\boldsymbol{p}_{cam,end}$
THEN the skill has failed
ELSE continue at 2.

(5) Calculate a 3D line from the camera through the (undistorted) image location, received from the vision system.
(6) Calculate the intersection point between the line and the horizontal plane with the (taught) height h_{detect}. This gives the 3D position of the object in camera space.
(7) Transform the object position from the camera space to the robot's base space.
(8) Replace the height of the position with the (taught) value, h_{grasp}, to make the robot grasp the object at a suitable location.
(9) Project the object position on the line between $\boldsymbol{p}_{\text{cam,start}}$ and $\boldsymbol{p}_{\text{cam,end}}$.
(10) **IF** *the projected object position is between* $\boldsymbol{p}_{cam,start}$ *and* $\boldsymbol{p}_{cam,end}$
THEN pick up the object at the calculated 3D position
ELSE the object position is not in the acceptance region, continue at 2.

Table 1. Algorithm for execution of vision based pick skill.



(a) Search for rotor caps (b) Pick up of rotor cap

(c) Search for rotor caps (d) Pick up of rotor cap

Fig. 3. Execution of vision pick skill. The robot captures images while moving in (a). The images are send to the quality control system shown in Figure 1, and whenever a rotor cap is detected, the robot picks it up, as shown in (b). Figures (c) and (d) show the same skill executed at a different location.

is mainly done at the vision system, and the robot itself only needs to know the name of the particular test to perform. In the skill framework, quality control can in general be viewed as a post-condition check, and if this fails, appropriate handling must be implemented. For the tests described here, this can either be to report an error, or to wait a short while and try again. The algorithm for execution of the developed quality control skill is shown in Table 2.

(1) The AIMM signals to vision system to perform (taught) quality control.
(2) Vision system performs control, and replies success/failure.
(3) **IF** *success*
THEN the AIMM continues
ELSE perform appropriate error handling (wait and try again, or call operator).

Table 2. Algorithm for execution of quality control skill.

2.5 Fast calibration

As mentioned in the introduction, the purpose of this skill is to provide calibration in three dimensions, faster than the existing calibration approaches developed by Hvilshøj et al. (2010), which have durations of 10 seconds and above. This is attempted by using the Kinect-like camera Asus Xtion Pro Live, that provides calibrated and undistorted images in both RGB and depth. In our approach, the calibration is implemented as a unique skill, thus having both a teaching and an execution phase. The phases are, however, almost identical. The purpose of both teaching and execution is to find the coordinate system of the (fixed) QR code; the QR frame. Subsequently all locations must be given relative to this frame.

When initiating the calibration skill, the RGB images from the Xtion camera are searched for QR codes. There are several libraries available that provide this functionality, and here *zbar* is chosen, because this directly provide the location of the corners in the images. When a QR code has been found, the depth at each corner of the QR code is averaged over a number of images, and the QR code's coordinate system can then be calculated as:

$$\boldsymbol{x} = \frac{\boldsymbol{c}_0 - \boldsymbol{c}_3}{|\boldsymbol{c}_0 - \boldsymbol{c}_3|} \tag{1}$$

$$\boldsymbol{y} = \frac{\boldsymbol{c}_2 - \boldsymbol{c}_3}{|\boldsymbol{c}_2 - \boldsymbol{c}_3|} \tag{2}$$

$$\boldsymbol{z} = \frac{\boldsymbol{x} \times \boldsymbol{y}}{|\boldsymbol{x} \times \boldsymbol{y}|} \tag{3}$$

where $\boldsymbol{c}_{\text{n}}$ is the location of the n'th corner of the QR code, and the corners are numbered clockwise.

To be able to work in this coordinate system it must be converted into a complete transformation matrix. This is done by calculating a *translation* and a *rotation*. The translation $\boldsymbol{t}_{\text{QR}}$ is defined by the center of the QR code, and is thus calculated as the mean of the corners:

$$\boldsymbol{t}_{\text{QR}} = \sum_{n=0}^{3} \frac{\boldsymbol{c}_{\text{n}}}{4} \tag{4}$$

The rotation $\boldsymbol{r}_{\mathrm{QR}}$ is best defined as Euler angles, which can be calculated directly from the axes. The transformation from the QR code to the camera, ${}^{\mathrm{C}}_{\mathrm{QR}}T$, can be determined by combining the translation and rotation. The desired transformation is between the robot's base and the QR code, ${}^{\mathrm{B}}_{\mathrm{QR}}T$, and this is computed as:

$$ {}^{\mathrm{B}}_{\mathrm{QR}}T = {}^{\mathrm{B}}_{\mathrm{C}}T \cdot {}^{\mathrm{C}}_{\mathrm{QR}}T \tag{5} $$

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

Finally, the coordinate system given by this transformation is applied to the robot. The exact sequence of execution of the calibration skill is given in Table 3.

(1) **WHILE** *correct QR code not found*
- Search for QR code in RGB images
- Read QR code
- **IF** *the text of the QR code matches taught string* **THEN** exit while loop

(2) Capture a number of depth images.
(3) **FOR** *each corner of the QR code*
- At the location of the corner, calculate the mean of the depth values (ignore 0-values).

(4) **IF** *one or more corners have no depth values* **THEN** the skill has failed. Exit.
(5) Define a coordinate system at the QR code as in Equations (1)-(3).
(6) Calculate the **translation** of the QR code $\boldsymbol{t}_{\mathrm{QR}}$ as the mean of the corners.
(7) Calculate the **rotation** of the QR code's coordinate system $\boldsymbol{r}_{\mathrm{QR}}$ in Euler angles.
(8) Combine translation and rotation into a transformation matrix, ${}^{\mathrm{C}}_{\mathrm{QR}}T$.
(9) Calculate the transformation from the QR code's coordinate system to the robots base coordinate system, ${}^{\mathrm{B}}_{\mathrm{QR}}T$, as in Equation (5).
(10) Set the robots frame base to ${}^{\mathrm{B}}_{\mathrm{QR}}T$.

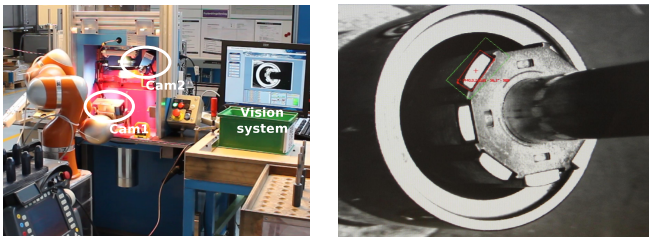
Table 3. Algorithm for execution of calibration skill based on QR codes. The Asus Xtion Pro Live was used for capturing RGB and depth images.

3. RESULTS

The three applications of computer vision have all been implemented on Aalborg University's AIMM *Little Helper*, and tested in a real-life industrial environment at a Grundfos factory. The vision pick skill was able to successfully pick an arbitrary number of rotor caps from two different locations, as shown in Figure 3. The precision was within ±5 mm, which was sufficient to correctly place the rotor caps at the desired locations afterwards.

The quality control was used for a variety of different tests. The application of this integration is only limited by the capabilities of the vision system itself, which is not described here. An example is shown in Figure 4, where it is detected that a magnet has been correctly placed beside the rotor core.

The setup for using the calibration skill is shown in Figure 5. The switch in the Figure is used to enable and disable the conveyor belt. The purpose of the calibration is here to make it possible for the robot to operate the switch,



(a) Assembly (b) Quality control

Fig. 4. Quality control setup. The robot is performing assembly tasks to the left in (a), while the quality control system is running externally, shown on the screen to the right. Figure (b) shows a close up of the result. The green box is the region of interest (ROI), and the red marking is the detected magnet.

and the position of the switch can thus be considered as a position of interest. The position of the QR code relative to the position of interest of course affects the calibration precision, and especially three factors affect the overall precision:

(1) The position estimate of the corners of the QR code the the cameras RGB image. These positions can be determined with sub-pixel accuracy, and at a distance of about 1 m as used in this setup, the precision of the corners is within ±1 mm.
(2) The relative error in the depth values at the corners for repeated measurements. The depth sensor in the Asus Xtion is the same as in the 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 (El-laithy et al., 2012). No data are available on the relative repeatability error, but it has proven to be significantly smaller.
(3) The relative error in the depth values between the corners. No data are available on this precision, but this has also proven to be significantly less that the absolute error.

Especially the third factor; the relative error between the corners, is of interest, because this will cause the coordinate system at the QR code to have a slightly wrong orientation. A wrong orientation makes the error increase the longer the distance between the QR code and

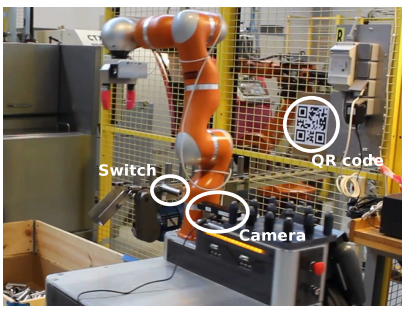


Fig. 5. Fast calibration using the Asus Xtion camera featuring calibrated RGB and depth images. The camera detects the pose of the fixed QR code, and subsequent movements with the robot are corrected accordingly.

the position of interest, and this is also what was found to be the case in the test scenario. Although no formal measurement of the precision has been carried out, visual inspection has shown that the precision is at least ±10 mm at any position. This precision proved to be sufficient to make it possible to operate the switch. In Table 4, the proposed calibration is compared to existing methods.

4. DISCUSSION

Integration of computer vision abilities into a skill based framework proved to be possible, and in this paper, three applications were successfully implemented. In particular the implemented quality control is very generic, and using the developed TCP/IP based protocol, the vision system could be changed without making any changes to the robot itself. This is also the case for the pick skill; however this has in the current implementation some limitations. It is currently assumed that the items to pick up are approximately placed in a line, as is for instance the case on a conveyor belt. Thus during teaching, the start and end location of the camera are taught. A further development should make it possible to define an arbitrary search region during teaching of the skill. For this, an optimal search pattern should automatically be calculated by the robot, taking into account that objects closest to the robot must be picked first. Positions and orientations of the camera during search should also be automatically determined.

The implemented calibration skill makes it possible to perform a very fast calibration compared to existing methods. This is especially important for industrial robots that are moving frequently between workstations. The precision was sufficient to perform the experiments described here, but for high-precision tasks it will be insufficient. There are two obvious ways of doing this:

- The Asus Xtion camera used, does in principle support RGB images with a 1280×960. A bug in the available open source drivers limited, however, the available resolution in our implementation to 640×480. Use of the full resolution images will definitely increase the precision of the QR code detection.
- From the depth image, only the four corner points were used. A better performance could be achieved by using the entire surface of the QR code, for instance by applying the RANSAC algorithm to filter out outliers.

It is impossible to say how much the precision can be improved. However, an experiment should be carried out to determine the precision exactly.

Method	**Duration**	**Precision**
Haptic[1]	30-45 sec	±1.0 mm
High speed[2]	10 sec	±1.0 mm
High precision[2]	60 sec	±0.1 mm
Proposed method	<1 sec	< ±10 mm

Table 4. Comparison of calibration methods. 1 are from Pedersen (2011); 2 are from Hvilshøj et al. (2010).

ACKNOWLEDGEMENTS

This research was partially funded by the European Union project TAPAS under the Seventh Framework Programme. The applications developed here was tested in an industrial environment at a factory owned by the Danish company Grundfos, and this was only possible due to a very close collaboration. Especially, thanks to J. Bigum and R. Larsen for assisting in developing the applied computer vision system.

REFERENCES

Alici, Gürsel and Shirinzadeh, Bijan. 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.

Benavidez, Patrick and Jamshidi, M. Mobile robot navigation and target tracking system. In *System of Systems Engineering (SoSE), 2011 6th International Conference on*, pages 299–304. IEEE, 2011.

Bjorkelund, Anders; Edstrom, Lisett; Haage, Mathias; Malec, Jacek; Nilsson, Klas; Nugues, Pierre; Robertz, Sven Gestegard; Storkle, Denis; Blomdell, Anders; Johansson, Rolf; Linderoth, Magnus; Nilsson, Anders; Robertsson, Anders; Stolt, Andreas, and Bruyninckx, Herman. On the integration of skilled robot motions for productivity in manufacturing. In *Assembly and Manufacturing, IEEE International Symposium on*, pages 1–9, 2011.

Björkelund, Anders; Malec, Jacek; Nilsson, Klas, and Nugues, Pierre. Knowledge and skill representations for robotized production. In *Proc. 18th IFAC World Congress*, 2011.

Brooks, Rodney. A robust layered control system for a mobile robot. *IEEE Journal on Robotics and Automation*, 2(1):14–23, March 1986.

El-laithy, RA; Huang, Jidong, and Yeh, Michael. 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.

EUROP. Robotics visions to 2020 and beyond - the strategic research agenda for robotics in europe (SRA). Technical report, EUROP, 2009a.

EUROP. Application requirements. Appendix to: Robotics visions to 2020 and beyond - the strategic research agenda for robotics in europe (SRA). Technical report, EUROP, 2009b.

Gat, Erann. On three-layer architectures. In Kortenkamp, David; Bonnasso, Peter R., and Murphy, Robin, editors, *Artificial Intelligence and Mobile Robots*, pages 195–210, 1998.

Guizzo, Erico and Ackerman, Evan. The rise of the robot worker. *Spectrum, IEEE*, 49(10):34 –41, october 2012. ISSN 0018-9235. doi: 10.1109/MSPEC.2012.6309254.

Hvilshøj, Mads; Bøgh, Simon; Madsen, Ole, and Kristiansen, Morten. 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.

Hvilshøj, Mads; Bøgh, Simon; Madsen, Ole, and Kristiansen, Morten. The mobile robot "Little Helper":

Concepts, ideas and working principles. In *ETFA*, pages 1–4, 2009.

Klank, Ulrich; Pangercic, Dejan; Rusu, Radu Bogdan, and Beetz, Michael. Real-time CAD model matching for mobile manipulation and grasping. In *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, pages 290–296. IEEE, 2009.

León, Adrián; Morales, Eduardo; Altamirano, Leopoldo, and Ruiz, Jaime. Teaching a robot to perform task through imitation and on-line feedback. *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 549–556, 2011.

Murray, Don and Little, James J. Using real-time stereo vision for mobile robot navigation. *Autonomous Robots*, 8(2):161–171, 2000.

Nava, Federico; Sciuto, Donatella; Santambrogio, Marco Domenico; Herbrechtsmeier, Stefan; Porrmann, Mario; Witkowski, Ulf, and Rueckert, Ulrich. Applying dynamic reconfiguration in the mobile robotics domain: A case study on computer vision algorithms. *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 4(3):29, 2011.

Nilsson, Nils J. *Principles of Artificial Intelligence*. Morgan Kaufmann Publishers, January 1993. ISBN 0934613109.

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

Shotton, Jamie; Fitzgibbon, Andrew; Cook, Mat; Sharp, Toby; Finocchio, Mark; Moore, Richard; Kipman, Alex, and Blake, Andrew. Real-time human pose recognition in parts from single depth images. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1297–1304. IEEE, 2011.

Siegwart, Roland and Nourbakhsh, Illah R. *Introduction to autonomous mobile robots*. MIT press, 2004.

Tölgyessy, Michal and Hubinskỳ, Peter. The kinect sensor in robotics education. In *Proceedings of 2nd International Conference on Robotics in Education*, 2010.

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