

# Robot Control For Dynamic Environment Using Vision And Autocalibration

Thomas Dall Larsen, Jacob Lildballe, Nils A. Andersen & Ole Ravn  
Department of Automation  
Technical University of Denmark  
Building 326, DK-2800 Lyngby, Denmark  
E-mail: or@iau.dtu.dk

## Abstract

*To enhance flexibility and extend the area of applications for robotic systems, it is important that the systems are capable of handling uncertainties and respond to (random) human behaviour. A vision system must very often be able to work in a dynamical “noisy” world where the placement of objects can vary within certain restrictions. Furthermore it would be useful if the system is able to recover automatically after serious changes have been applied, for instance if the camera has been moved.*

*In this paper an implementation of such a system is described. The system is a robot capable of playing checkers with a human opponent using a camera to determine where the board and pieces are located and when the human has made his move. The paper primarily focuses on the vision aspect of the implemented system.*

## 1 Introduction

The use of vision as a sensor in robotics has increased in the recent years as computer speed has improved and the cost involved decreased. The advantages of using vision in robot installations working in changing environments is immense as such systems can be difficult or impossible to implement using traditional sensors. Furthermore systems that interact with or observe humans has been improved with the advent of vision. The problems with vision based robot systems often concern the camera/robot coordination as well as the extraction of useful information from the camera data in a feasible and robust way.

For industrial robot systems it is important to ensure an easy way to calibrate and adjust to changing surroundings, i.e. new camera positions, new robot

kinematics, other assignments, etc. Especially when the robot is going to interact with humans this becomes more evident as very strict safety precautions must be applied to avoid harmful conflicts.

An example of a vision based human interactive system was set up by the Department of Automation to explore the possibilities in the above mentioned problems. The task was to create an automatic player for the classic checkers game that could play a complete game with a human player using as little “help” in the form of guide marks, special boards, etc. as possible. Another important specification was that the system should be easy to use and modify for other purposes. The reason a game system was chosen was that such a system involves solving a wide range of problems with vision/robot control as well as human and environment interactions. The development of this application required work in the following disciplines:

- Camera/robot calibration.
- Vision based feature extraction.
- Robot control.
- Man/machine interaction.
- Structured software development.

This paper focuses mainly on the vision aspects of the problem but also includes a short description of the robot and software developed to solve it. The system and the modular software developed for it is described in section 2. The vision methods applied are more thoroughly described in section 3. This section include a description of the camera calibration problem (also described in [5], [7, 8] and [9]) and the methods used for extracting features from a vision input source. The achieved results from real-life tests

of these methods and the overall system performance is provided in section 4.

## 2 The system setup



Figure 1: The vision/robot system setup.

The system used for the implementation of the checkers playing robot is based around a three degrees of freedom (3DOF) robot connected to a computer system. On figure 1 the typical setup is shown. The robot is placed next to a table in such way that the entire checkers board is inside its workspace. The dotted board to the right of the checkers board is a calibration tile used for camera calibration (see section 3.1 or [5]). The video sensor is a standard black and white surveillance camera placed on a tripod in such a way that it can overview the scene including the robot and checkers board.

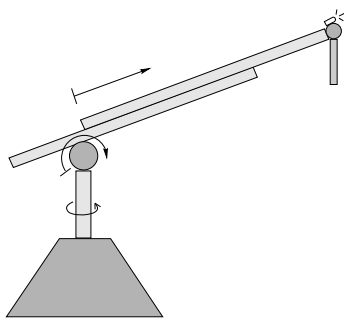


Figure 2: The 3DOF robot with electromagnet end-effector and a LED.

The robot (see figure 2) has a LED mounted at the end of the third joint. This LED is used for easy localization of the robot's position using the vision system. The end-effector of the robot is an electromagnet hanging from the third joint. To control the robot each of the three joints has a position and a tacho feedback. These signals are used in two control loops with the inner loop being an analog velocity feedback loop and the outer loop a digital control loop based on the position feedback and controlled by the computer system described below.

### 2.1 The computer system

The choice of hardware platform to control the system was of course influenced by the currently available systems at the Department of Automation. As the task of doing vision control of a robot can be split into several individual sub-systems, a transputer system (see [6]) was chosen as platform, as the nature of the transputer systems enforces a hierarchical design of the software. A transputer is a special type of 16 or 32 bit processor developed especially for parallel processor systems. It has build-in hardware for high-speed communication (up to 20 Mbit/sec) with other transputers via serial links, as well as hardware support for fast pre-emptive multitasking. The applied system is build up of different types of transputers mounted on either PC or VME bus cards. Some of the seven transputer modules (TRAMs) has specialized hardware for different tasks, e.g. graphics, vision manipulation, I/O, etc. In the diagram shown in figure 3 the current transputer setup is presented including the link connections between them. The configuration of the links are decided by the estimated amount of communication between the transputers and by certain hardware restrictions. Transputers can communicate even if they are not directly linked; in this case a virtual link is established by the operating system via the transputers between them.

The three DATS (Data Acquisition Transputer Subsystem) transputer cards shown on figure 3 are specially developed I/O modules providing a range of interface possibilities. Each DATS card has the following features:

- An onboard T225 16 bit transputer with 64 Kb RAM.
- Eight A/D inputs and two D/A outputs.
- Eight digital inputs and outputs (on/off).
- A 16 bit encoder counter.

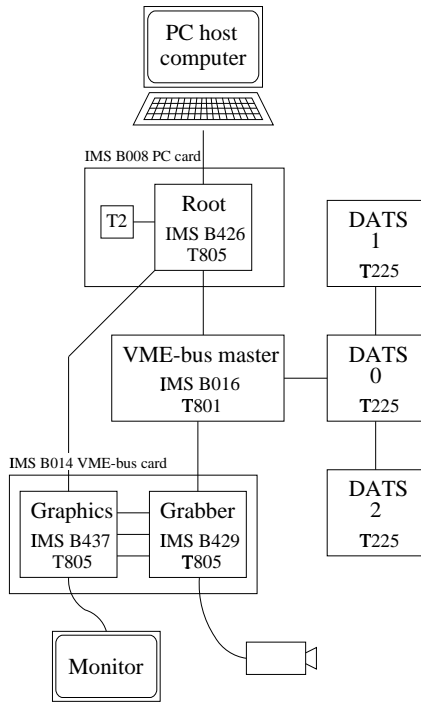


Figure 3: The transputer setup.

- External interrupt input.
- Watch-dog function disabling all outputs in case of an emergency.

For the robot control only two of the DATS cards are used as only four D/A outputs are needed for controlling the three joints and the electromagnet (through appropriate amplifiers). However if there is need for more than six D/A outputs, more DATS cards can be connected to the system.

For the vision handling the special TRAM IMS B429 is used. This TRAM has a 32 bit T805 transputer processor, 1 MB of program memory, a grey-scale video digitizer (with four video inputs), two IMS A110 circuits and three 512 KB video buffers. The IMS A110 circuits provides the possibility to do several useful vision functions in real-time. They can perform a 2D convolution of an  $768 \times 512$  picture in  $1/25$  of a second with kernels of dimensions up to  $6 \times 7$  or  $3 \times 14$  as well as thresholding, buffer subtractions, etc. This is very useful for the very computer-intensive but simple tasks often needed at the start of an image analysis.

## 2.2 The modular software

The control software was developed with a high level of modularity and portability in mind. Apart from the obvious advantages of such a design in the form of easier expandability and modifications, it is also necessary for programming the transputer system as the hardware resources used are distributed on several processors.

The software can be split into five major parts, i.e. the main program and four parts administrating a specific task or hardware. The hardware resource administrators are based on the client/server principle as they execute commands send through communication pipes from application programs. The four currently implemented client/servers are listed in table 1. The use of a server's functions from an application program is very simple as a library is provided with each server program containing easy to call functions that are used in the same way as normal function calls. The communication with the server are then handled by the library functions. This way the actual communication protocol used to transfer parameters and results between a server and an application program is hidden for the user, who can program in the same way as if the program was located on a single processor.

The server modules are designed in such a way that they can communicate with each other providing the possibility to form "families" of servers that enables higher level functionality as more advanced server functions can take advantage of facilities provided by other servers. As an example of the possibilities it is possible to instruct the vision server to control the robot movement directly by using vision information and then sending commands to the robot control server.

## 3 Vision system

To enable a robot to play a game of checkers against a human opponent using vision, it is necessary to solve the following vision related problems:

- To determine an accurate relationship between the camera images and the robot coordinate system (called a camera calibration).
- To find the checkers board in an image containing noise.
- To find and determine the color of a piece located on the board.

Server name	Description
DATS server	Controls I/O channels on I/O cards
Robot server	Controls robot movements by using the DATS server modules for I/O
Vision server	For vision handling and manipulations
Graphics server	The graphics system

Table 1: The four client/server programs used in the software system.

- Being able to detect when the human opponent has made a move.

In this section the solution to these problems is described.

### 3.1 Camera calibration

The calibration of the camera is primarily used when the robot coordinate of an image point is needed, for instance when a piece is to be picked up, but it is also quite useful when objects are to be found in the image because it enables the physical dimensions of the objects to be used as a noise filter. The camera calibration determines two sets of parameters:

- The internal parameters for the camera and frame grabber (focal length, optical center, distortion and aspect ratio).
- The external orientation and translation of the camera relative to the robot. These parameters describe a coordinate transformation and can be assembled in a homogeneous  $4 \times 4$  matrix as described by Denavit and Hartenberg [1].

The approach taken throughout this paper is to use as much prior knowledge of the world as possible. For flexibility it is desirable that the camera may be positioned freely. This means that no knowledge of the external parameters can be assumed. The internal parameters on the other hand are constant once a camera has been chosen (there may be a slight temperature dependence but with a crystal driven clock this should be insignificant). It is therefore chosen to perform one accurate off-line calibration using a well defined calibration tile (see figure 4) to determine the internal parameters and to use these parameters as constants in future on-line calibrations.

In the on-line calibration a tile can only be used with some difficulty because it would have to be located in a fixed position relative to the robot. This leads to restrictions in the camera placement and requires an accurate (perhaps manual) determination of the position of the tile in the robot coordinate system. Therefore the on-line calibration is performed

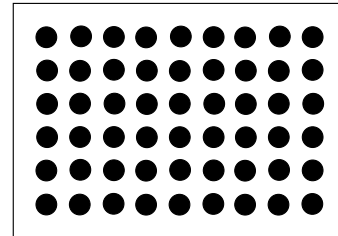


Figure 4: The calibration tile used for determination of internal camera parameters.

by moving the robot to a number of positions within the camera view and determine the pixel position of a fixed spot on the robot (a LED mounted on its third joint - see figure 2) from an image grabbed with the camera. This generates a set of related pixel and robot positions that along with the internal parameters of the camera are used to calculate the homogeneous transformation matrix. The calibration algorithm used for the off-line calibration is proposed by Tsai and Lenz in [7, 8] and Allan Theil Sørensen in [5]. The on-line calibration uses the same algorithm but avoids some of the calculations as the internal parameters are already known. The robot positions used for the calibration are selected to be positioned in a known plane in the cartesian space of the robot as this is required by the Tsai and Lenz method.

This approach means that the on-line calibrations becomes linear though an accurate non-linear model of the camera is used. The approach has two main advantages:

- It increases the speed of the on-line calibrations greatly because calibration methods dealing with camera distortion tends to involve time consuming non-linear approximations.
- It increases the accuracy of the calibrations because a well defined calibration tile is used in the sensitive non-linear part of the calibration. The unavoidable inaccuracies in the positioning of the robot will therefore have as little effect on the calibration as possible.

Because the on-line calibration does not require any human interaction the system is capable of performing an auto-calibration if the camera or robot base has moved.

### 3.2 Finding objects from vision data

As already mentioned there is no restrictions in the placement of the camera relative to the robot as long as it can overview the entire work scene. This means that the image of a given object before the camera is calibrated is somewhat unpredictable. The size and the shape of the objects depends strongly on the distance and angling of the camera; a 3D object projected on a 2D image plane can change dramatically as the camera is tilted or rotated. Besides the projection, the non-linear distortion in the camera also has some effect, especially on objects placed near the edges of the image.

This means that it may be a great advantage to transform the 2D image to 3D robot coordinates, where the object obviously is much more well defined. To perform this transformation it is necessary to have some prior knowledge about the placement or dimension of the object because when transforming from 2D to 3D there is a degree of freedom which has to be specified. In many practical applications, including the checkers player, the work scene is a horizontal plane, so the required information will be the equation of the work plane.

When the object is transformed to robot coordinates it is possible to use prior knowledge of the objects dimensions to increase the reliability and decrease the noise sensitivity of the object recognition. If the calibration is sufficiently accurate, objects can in fact be distinguished within the resolution of the camera. If the resolution of the camera for instance is  $512 \times 512$  pixels, and the distance to the work plane is one meter, this means that the size of the objects in a plane parallel to the image plane can be determined approximately within  $1 \text{ mm}$ . Prior knowledge of object dimensions combined with a good camera calibration is therefore an efficient way to separate objects from noise and can furthermore be used to compensate for a lack of information in the image, for instance due to an inappropriate camera position.

#### 3.2.1 Finding the checkers board

In figure 5 a noisy picture containing a checkers board is depicted. The checkers board consists of  $8 \times 8$  equally sized dark and bright squares. Because of the significant difference in pixel intensity be-

tween neighbour squares, the structure of the checkers board can be greatly enhanced using an edge detection algorithm. After applying this the grabbed image is transformed to the image shown in the middle in figure 5. Regardless of the camera position the edge filtered image will always contain the  $9 \times 9$  board lines shown in the figure (if the distortion in the camera is within reasonable limits), because of the linear nature of the projection.

These lines can therefore be used to find the position of the board in the image. To extract the line equations from the image a Hough transformation [2] of the image is performed. The Hough transformation is chosen because it suppresses noise on the line very effectively. The pieces on the checkers board often hides parts of the lines (as shown on figure 5), but this is ignored by the Hough transform. Algorithms based upon an evaluation of the form and surroundings of the line (like the canny edge algorithm [4]) will have great difficulties dealing with this noise.

To perform a Hough transformation a parameterization of the lines must first be chosen. When choosing the parameterization it is important to ensure that the parameters are limited and unambiguous. In figure 6 a line representation using the slope of the line  $\alpha$  and the orthogonal distance  $s$  from the line to origo. The sign of  $s$  equals the sign of  $y$  where the line intersects with the  $y$ -axis.

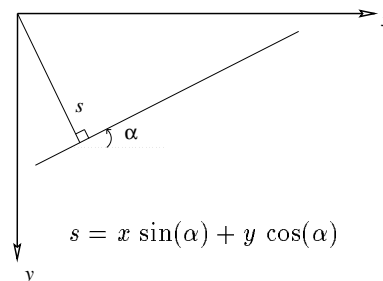


Figure 6: Representation of lines.

If the resolution of the screen is  $512 \times 512$  pixels, these parameters will be limited by:

$$-\frac{\pi}{2} \leq \alpha \leq \frac{\pi}{2} \quad , \quad -724 < s < 724$$

The Hough transformation calculates for every image pixel with a value above a certain threshold all possible lines traversing that pixel. This is done by calculating  $s$  for a number of different  $\alpha$ 's and for every  $(\alpha, s)$  incrementing a value in a Hough matrix. After searching through the edge filtered image

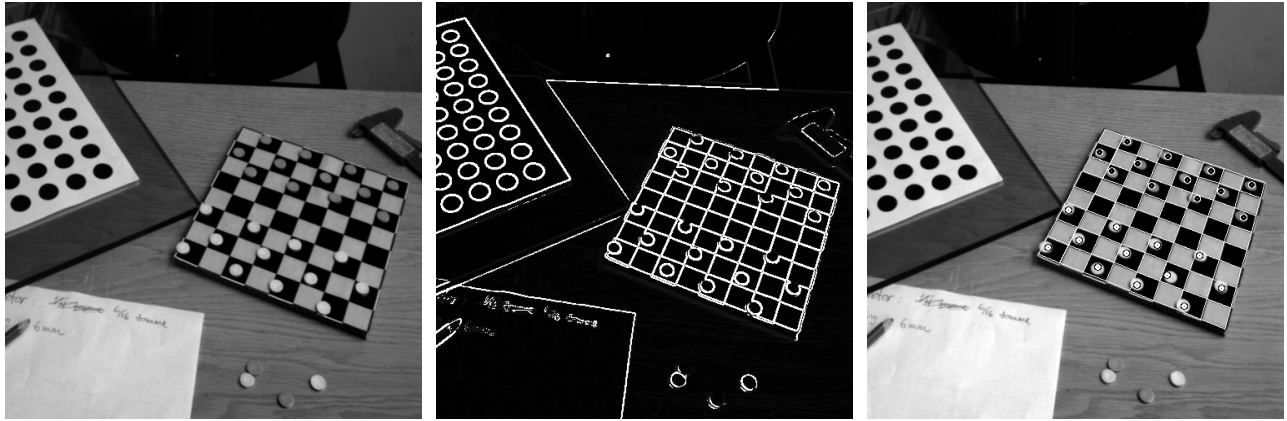


Figure 5: Grabbed image of checkers board. The image in the middle has been filtered with an edge detecting algorithm. On the image to the right the board and the pieces have been found and marked.

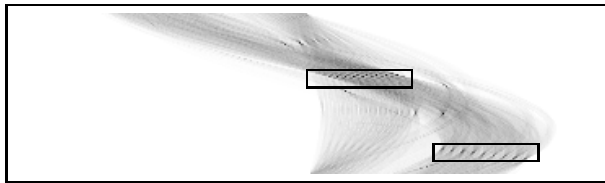


Figure 7: The Hough matrix generated after transforming the edge filtered image above. The  $9 + 9$  board lines (shown in boxes) as well as the noise lines appears as maxima (dark spots) in the matrix.

in figure 5 the Hough matrix will look as depicted in figure 7 with image lines appearing as maxima. As the image may contain a lot of additional (noise) lines, the Hough matrix will contain several maxima besides the ones coming from the checkers board. To separate the noise lines from the board lines some knowledge about the structure of the board must be utilized. The structure of the  $9+9$  lines in the image plane however depends strongly upon the placement of the camera. Therefore the lines are transformed to the robot coordinate system assuming all lines are in the plane of the work scene. This means that the board lines now are equally sized and placed in a quadratic grid with well known physical dimensions (i.e. the board dimensions). Therefore the separation of the board lines from the noise is now a rather trivial task.

As the Hough transformation is a computer intensive task to perform it is desirable only to do this operation when needed. Therefore a simple method to determine if the board has moved since the last image

grabbed is to observe whether the board lines in the edge filtered image have not moved. If the lines have moved significantly an entirely new board search using the Hough transform must be performed. If the lines have only moved slightly, the board position can be updated immediately.

### 3.2.2 Finding the checkers pieces

To be able to distinguish the dark and bright (red and white) checkers pieces from each other and from the background using a black and white camera, it is necessary to chose the board and piece colors intelligently. The board is therefore chosen so that the light background fields causes pixel intensities in the grabbed image between the intensities caused by the dark and bright pieces. This means that the color of a piece can be determined directly from the neighbouring bright fields as illustrated on figure 8. This ensures that if the lighting changes in time or within the board area due to highlights, shadows, etc. the threshold value will adapt as long as the above conditions for pixel intensities are maintained. If the camera adjusts the gain automatically (AGC), this method will work under a wide variety of lightings.

When the physical center of the top of the piece is to be found, the camera calibration and the physical dimensions of the piece can be exploited. If for instance the bottom corner of the piece in the image is found it can be transformed to robot cartesian coordinates and translated to the top of the piece. The piece can then be picked up by the robot by moving the robot magnet to the calculated cartesian

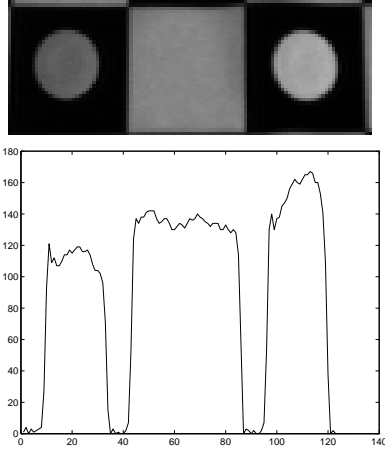


Figure 8: Intensities of the checkers pieces and the board fields.

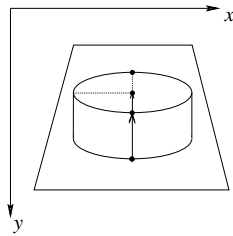


Figure 9: The pixel position of a checkers piece's bottom edge transforms into the cartesian position of its lower bottom edge whereby the cartesian center of the piece is easily calculated based on its physical dimensions.

piece position. Naturally this requires that the camera calibration and the modeling of the robot is quite precise.

### 3.3 Detecting a move

When the checkers player is playing against a human player it must be able to detect when the human has made a move. This can either be done by letting the human press a button when the move is completed or by automatically detecting when the human has manipulated the board in a legal manner. The detection can be performed by observing a line along each of the four edges of the board and detect when they are interfered. The interference must be a significant change in a range of monitored pixels as smaller changes can be caused by altering lighting. Whenever one or more of the lines have been violated by the human player the checkers player awaits until the lines are restored again and checks if a legal move has been made by monitoring the board and

piece positions. If no legal move is detected the user is notified and the line monitoring repeats itself. If the algorithm does not detect that the human player has made a move, a button can be pressed to inform the computer that a move was made. The line monitoring algorithm can fail if the human makes very slow movements as the changes in the line intensities are then mistaken for light changes.

## 4 Results

As stated in section 3 it is essential with the chosen approach to obtain a very accurate camera calibration. For instance when an object is to be picked up, the robot is positioned in a coordinate calculated using an image point and the camera calibration. If the camera calibration lacks accuracy the coordinate will be wrong and the robot will miss the object.

To ensure that the camera calibration becomes adequately good, some effort were put into verifying that the robot reproducibility and the modeling of the robot was precise enough. When these conditions were fulfilled and a sufficiently advanced calibration method was used (like the Tsai and Lenz method) experiments with the checkers playing robot showed that it is possible to obtain accuracies good enough to let the robot play a game of checkers, i.e. to extract information from vision data that can be transformed with an accuracy of approximately  $1 \text{ mm}^3$ .

However the experiments showed that the placements of the cartesian positions used to calculate the calibration parameters are quite crucial. It appears that the further away from the work scene the calibration positions are placed the more inaccurate the calibration tends to become. This is of course not that surprising but the magnitudes of the deviations are. Tests shows that when the calibration positions are placed in the same plane as the robot should work in, the results are good. However when the positions are chosen in a plane as little as  $10 \text{ cm}$  above the plane of the work scene the uncertainty of the calculated pixel to cartesian positions is more than  $5 \text{ mm}$ . That means it is necessary to perform the calibrations with cartesian positions placed very close to the plane that the robot should work in. Further investigations into this problem are on-going.

When the accuracy of the calibration and the robot is sufficient, the constant use of knowledge regarding physical dimensions and previous placement of board, pieces, camera, etc. ensures that the checkers player performs well in a dynamical work scene with (somewhat random) human behaviour. The

checkers player is capable of calibrating itself at start up, and later to detect whether the camera or robot base has moved and then recalibrate. Without human guidance it can locate the checkers board and pieces and determine if anything has moved. The checkers player works even though the light is shifting (in time or space) and is very tolerant regarding noise in the picture. The checkers player can therefore without further instruction interact with anyone who knows the rules of checkers. This has been tested at an exhibition in Copenhagen in february 1996 where the robot played against the public. During the five days of the exhibition the checkers player did not fail once in finding the board and pieces or performing the moves.

## 5 Conclusion

The methods and vision aspects involved in the development of a flexible robot/vision system was described in this paper. The camera calibration determining the relationship between the robot and camera coordinate system was performed using an approach proposed by Tsai and Lenz. For feature extraction the Hough transformation in connection with the calibration was employed. The use of the calibration enabled an effective noise filtering because the physical dimensions of the objects could be utilized in the object recognition.

## References

- [1] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *Journal of Applied Mechanics*, Vol. 77, pp. 215-221, 1955.
- [2] P. V. C. Hough, "A Method and Means for Recognizing Complex Patterns," *U. S. Patent 3,069,654*, 1962.
- [3] T. D. Larsen & J. Lildballe, "Vision based Robot Control (in Danish)," *Department of Automation, DTU*, 1995.
- [4] Milan Sonka, Vaclav Hlavac og Roger Boyle, "Image Processing, Analysis and Machine Vision," *Chapman & Hall computing*, 1993.
- [5] Allan Theil Sørensen, "Analysis of Real-Time Vision in Control and Automation Systems," *Department of Automation, DTU*, 1994.
- [6] "Ansi C Toolset User Manual," *SGS-THOMSON, INMOS doc. nr. 72 TDS 224 00*, 1990.
- [7] Roger Tsai, "An efficient and accurate camera calibration technique for 3D machine vision," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1986.
- [8] Roger Tsai, "A versatile calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE Journal of Robotics and Automation*, 1987.
- [9] O. Ravn, N. Andersen, and A. Sørensen, "Auto-calibration in automation systems using vision. In Proceeding of the 3rd International Symposium on Experimental Robotics," *ISER'93, Kyoto, Japan*, October 1993.