Efficient Corners Detection in Robot Soccer

Master Thesis Proposal

am Fachgebiet Agententechnologien in betrieblichen Anwendungen und der Telekommunikation

Prof. Dr.-Ing. habil. Sahin Albayrak

Fakultät IV Elektrotechnik und Informatik

Technische Universität Berlin

vorgelegt von:

Yan Ma (322995)

Betreuer:

Dr. Yuan Xu

Abstract

Self-localization in the RoboCup often requires using different landmarks. Corners created by field lines are very important landmarks, and detecting corners efficiently will greatly improve robot's performance. This thesis seeks to improve the performance of corner detection algorithms within the setup of the RoboCup standard platform league. We attempt to do so by making use of an algorithm utilizing both the mean/middle point and the size of found white areas. We further hope to improve efficiency by making several other adjustments, such as increasing scanline density of the next frame around points of interest found on the last frame of the same camera.

1. Introduction

RoboCup is an international robotics competition founded in 1997. Robot Soccer is a complex challenge for Artificial Intelligence. It is an important area of research, because of its big emphasis on reliable and quick real time image processing and due to its strong link with other related fields such as artificial intelligence and locomotion [1]. A player has to be able to identify his teammates in their position, his current movement and the direction of motion, in order to predict his future activity. This is an issue during a soccer match played by humans as well as by humanoid robots [2]. In order to perceive the environment quickly and accurately, the robot is equipped with a variety of sensors. The ability to identify objects with the integrated cameras plays a central role.

The RoboCup's popularity has grown increasingly, what directly resulted in extension of its area of application. Beside of RoboCup Soccer, other fields of interest as i.e. RoboCup Rescue for developing of rescue robots, RoboCup Junior for increasing the popularity of the RoboCup idea among young people as well as RoboCup @Home for robot assignments in domestic household environment. In the RoboCup Soccer there are 5 leagues: Humanoid, Middle Size, Simulation, Small Size, and Standard Platform [3]. In this thesis, only Standard Platform League (SPL) will be covered.

2. Background

Computer Vision

Broadly speaking, Computer Vision is a large field, largely concerned with the physical structure of a three-dimensional world by the automatic analysis of images of that world [4]. This is a very broad definition, but the same would be true for human or more broadly biological vision.

Proper vision is of the utmost importance for the function of any vision based autonomous robot. Computer vision in robots is a broad field covering applications as diverse as vision/navigational aids for blind people, feature detection for satellites or autonomous robots for indoor use.

In this thesis we will focus on the computer vision for a humanoid robot in the RoboCup.

Computer Vision in Robotics and RoboCup

The RoboCup gives certain sets of unusual challenges to the robots. First, the robots are autonomous and computationally limited. Second, they interact not only with a moving object (the ball) but also with other autonomous players of their own and their opposing team. The presence of an opposing team greatly complicates the task. The opposing team will, either by design or coincidence, hinder the robots. For example by blocking vision, collisions, or by obtaining control of the ball. The quick time scale of events, the often changing conditions and the presence of opposition turns the RoboCup into the very challenging Computer vision task outside of direct military applications.

Computer Vision in RoboCup

The NAO is used in the academic research, the specifications of the installed hardware can be found in Table 1.

Table 1: Hardware specifications of the Nao [7]

CPU	ATOM Z530 1.6GHz CPU
RAM	1 GB
USB memory	2 GB flash memory / 4 to 8 GB flash memory dedicated to user purposes
Digital Camera	2 cameras with 1280x960 resolution and 30 fps
Communication	Wi-Fi, Ethernet
Operating system	Linux
Multimedia	Loudspeakers, Microphones
Weight	~4.8kg
Height	~58cm
Sensors	Hall effect sensors, gyrometer, accelerometer, channel sonar, bumper, I/R, Tactile sensor, FSR, head capacitive sensors

Two cameras are installed vertically to one another in the middle of the head portion of the NAOs. The top camera is used for localization on the playing field and gives a good overview of all events before the robot. The low contrast is useful to recognize balls directly in front of their feet as well as identify their own and others foot movements. One camera has a resolution of 640 × 480 pixels; the other one has 1280 x 960 resolutions and a frame rate of up to 30 fps. Both cameras can parallel run [2]. See Figure 2 for camera’s specification.

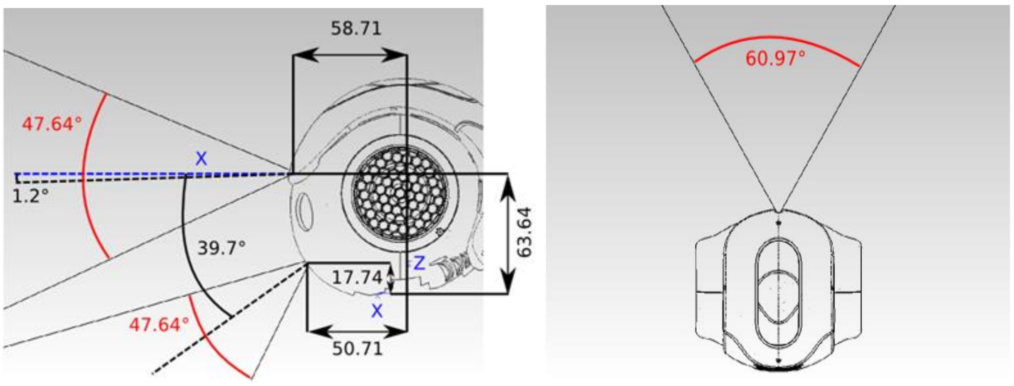

Figure 2: Cameras' specification of NAO [7].

Standard Platform League

In SPL, five NAO humanoid robots play on a 10.4 X 7.4 m² field according to RoboCup 2014 rule [6]. The knowledge of the field will be described in more detail in the following section.

Soccer Field

The measurement of the soccer field, which spans the total carpet area, is 10.4 meter length and 7.4 meter width. These dimensions are illustrated in the Figure 1.

Table 2: Specific information for SPL field [6].

ID	Description	Length(in mm)
A	Field length	9000
B	Field width	6000
C	Line width	50
D	Penalty mark size	100
E	Penalty area length	600
F	Penalty area width	2200
G	Penalty area distance	1300
H	Center circle diameter	1500
I	Border strip width	700

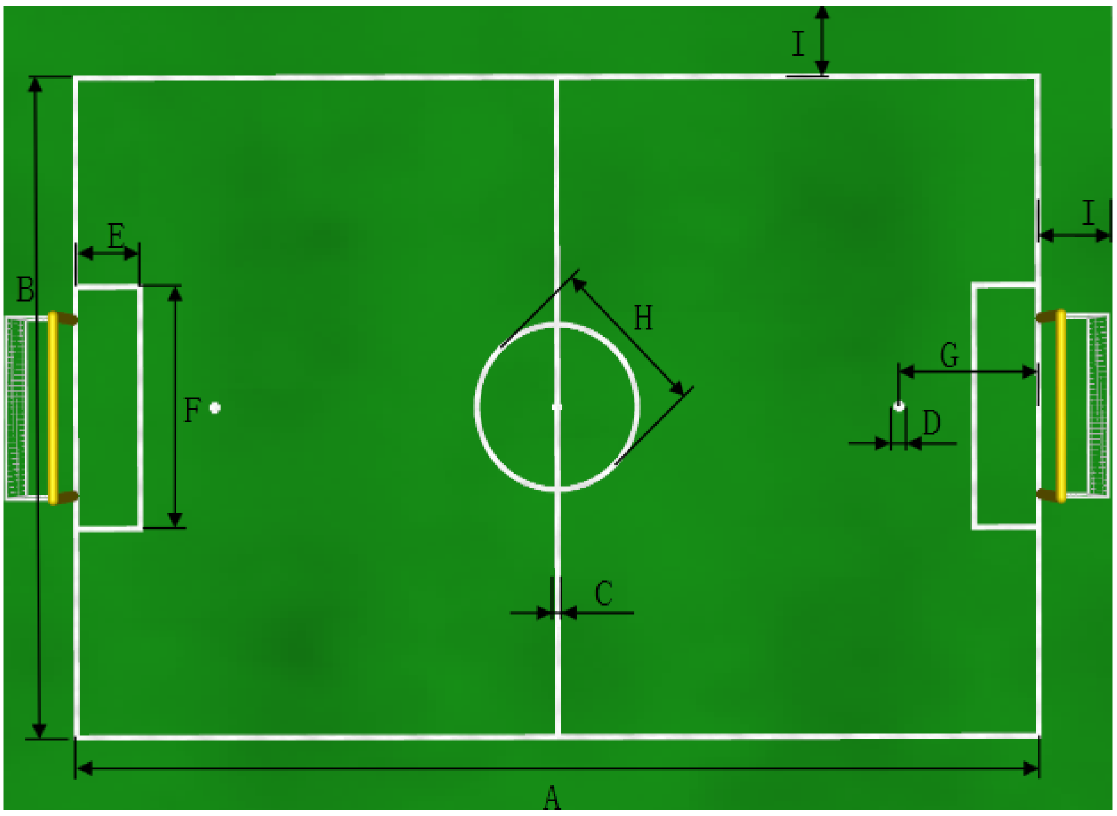

Figure 1: Schematic diagram of the SPL field, see table 2 specific information [6].

Features in the Field

On the field of a RoboCup SPL, line defines the scope of robot soccer field activities. Lines are an important feature of the robot to determine their position on the field. There are four different types of features, which could be extracted out of the lines. While the three line intersection types are named as T, L and X crossings, the fourth one has a form of a circle placed in the middle of the field and crosses the middle line exactly in two points [8]. These four types of features are the so-called corners. See Figure 3.

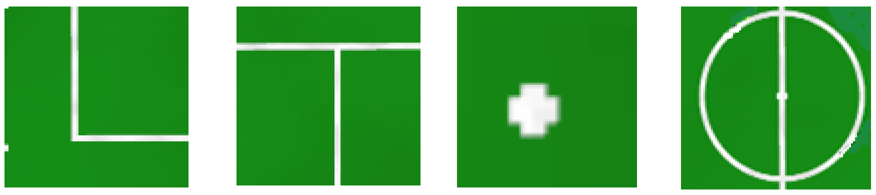

Figure 3: L type, T type, X type and Ө type.

There are a number of difficulties in quickly assessing lines and corners. One difficulty is that lines have their own broadness which is subject to changes based on distance from the robot. In addition, different objects such as humans or other robots can obstruct the view. A further difficulty is that the net-structure behind the goal is also made up of lines, and can hinder effective detection of lines and corners. Trade-offs also has to be made when it comes to processing speed and precision. In many cases, having both is not feasible, and naive approaches are often inefficient. Another issue is the prevalence of false positive inherent to nearly all methods. Particularly noteworthy are situations where the robot observes landmarks like lines, goals or corners on other fields and then arrives at wrong conclusions regarding his own position.

3. State of Art

The computational limitations of the NAO robot make it impossible to process the entire image in real time. Therefore, different methods of changing the effective resolution without losing an undue amount of information have been utilized. The most common approach is to process via scanlines. Only information along a scanline is processed further. The naive approach to distribute scanline starting points uniformly is of course non optimal, and attempts have been made to improve this state.

Härtl and Röfer et al. [10] have shown how the efficiency of using scanlines approaches can be improved. Their scanline density is not constant, but incorporates likely assumptions concerning the visual field and possible detectable objects. Effectively they divide the visual input area into sub regions of varying scanline density, and also use different scanline densities for horizontal and vertical scanlines in a given sub region.

Given a set of scanline-generated input, the Hough transform can be adapted towards arbitrary shapes, and thus, in theory [5], used to detect corners. We don’t think that relying solely on such an approach is an optimal solution. The weaknesses of Hough transforms, including computational expense, would likely have an effect. Several teams have meanwhile moved away from Hough based approaches.

One such team, the NAO Devils Dortmund [9] used positional and gradient data of intersections between their scanlines and actual field lines. This was followed by a clustering heuristic based on each intersections respective position and gradient to generate likely line candidates. This was further enhanced by an intermediate circle detection step. This approach had been proved successes, but struggled when it came to distinguishing white goal-net corners/lines from field corners/lines and identification of the center circle required a considerable part of it to be within the robots current field of vision. Looking to improve things, the team discovered that the NAOs cameras built in “edge-enhancers” considerably degraded the information associated with gradients by creating gradient artifacts. Instead of the previous gradients/positional information, the clustering algorithm uses their originating scanlines, their relative distance and the presence of points classified as white that are between the connections. Computational efficiency is increased by not calculating the deviation of each line candidate after an addition, and instead simply measuring if the new line chain is within the bounds of tolerance of the old one.

A somewhat but not exactly similar approach is used by the Dutch NAO Team [8], contrary to most other approaches; they scan for Black-White-Black transitions after a binarization step transforming a green field into black has been performed. As opposed to using the actual edges, they calculate the middle point between two black to white changes along one scanline. This reduces the number of "interesting points" for further computations; it also prevents issues with wrong mapping between 2 edges of the same line.

After such a set of points is obtained, they are stored as pixel coordinates in a vector. From here, the algorithm removes the first point, and then looks for the 5 closest points to it. For these 5 points, the whiteness ratio of the space between these points is checked. For uninterrupted points on one line, this value should be 1. The closest point with a "high enough" white ratio is connected; this process is repeated until the new line breaks the boundary established by an error function.

Once a line is finished, the algorithm checks if it is already existing line can be used as an extension of the current line. To do this effectively, a line end and starting point is defined, if there is overlap between either of these points and points already a part of another line, the lines are "test fused" and their error size is computed. If the error is within the threshold, the lines are fused permanently. If not, the information is stored as a corner candidate. The current Dutch NAO Teams corner recognition mechanisms distinguishes between 4 cases of corners, and uses additions to not calculate corners on the margins of the picture. The 4 corner types (T, L, X, Ɵ) can be distinguished using their geometric properties, and represent useful information for robot self-localization.

The major difference between the approach of the NAO Devils Dortmund and the Dutch Nao team is that the Dutch make the step from 2 edges to 1 line at a very early part of their line detection, while the NAO Devils do not do this. The NAO Devils uses a more complex line candidate clustering algorithm to account for that difference later.

Another team, B-Human [11], moved into the opposite direction to the Dutch NAO team as far as line shapes are concerned and is, broadly speaking, fitting "White regions" (which may be lines, but could also be something else) into trapezoid regions. If for example a corner cannot be reconciled with the shape of a trapezoid, the process breaks the noncompliant white space up into 2 or more trapezoids and continues. Line candidate Trapezoids are connected to each other to form lines if their color difference is low. The fact, that the absolute color is not important, makes the process more robust to illumination changes. After the line recognition step is completed, corners are detected by simply looking at intersections.

4. Proposed Approach

The target of thesis is to improve line detection efficiency and also enable our robots to detect and distinguish different types of corners (T, L, X, Ɵ). We propose to do this in the following steps:

1. Creating scanlines. The start points of scanlines are efficiently distributed, information about their start points is stored. See Figure 4.

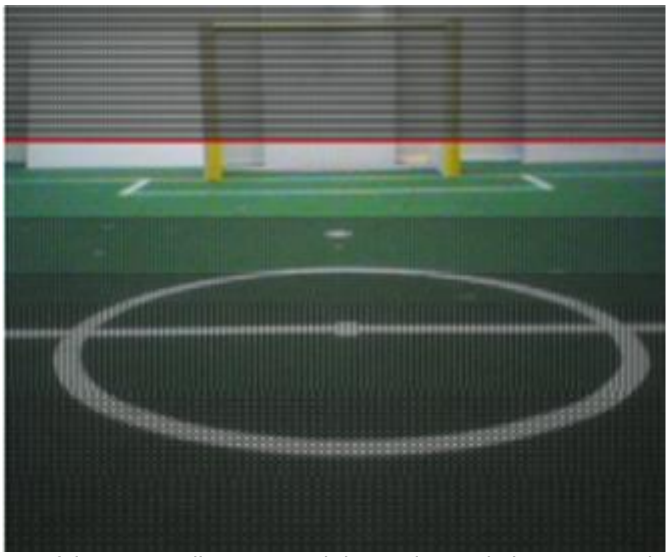
Figure 4: A state of the art scanline approach is used to optimize computational efficiency [10]

2. Excluding uninteresting objects. These extents to Use forward kinematics to exclude own robot parts from further calculations; Using interspersed orange only scanlines to scan for the ball; Ignore previously banned zones created by the "banzone" construct introduced by B-Human [12].

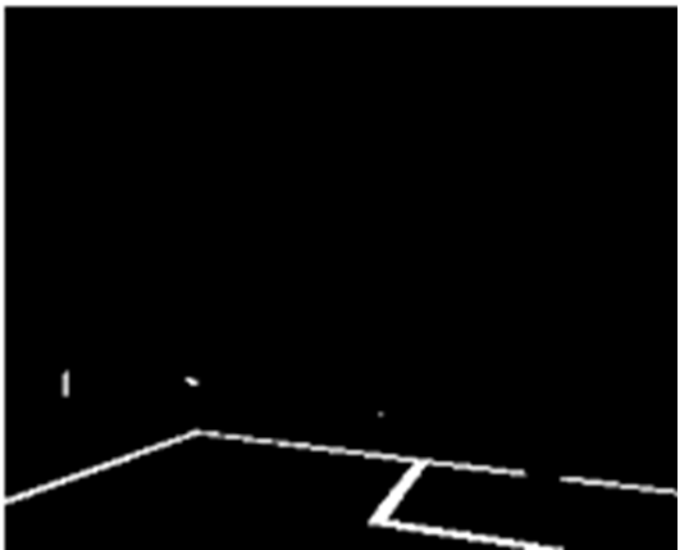
Figure 5: A possible situation after applying a black-white filter [8]

3. Appling a black white process for scanlines to distinguish white pixels from all others, see Figure 5.
4. Checking for Black-White-Black transitions along each scanline. Calculate the mean of the white area and use this point for further calculations. Calculate and store the length of the white area associated to a potential line spot, see Figure 6. This approach is, in general, very similar to the approach used by the Dutch NAO team, however, we store not only the mean coordinates but also store the length of the white area. We will use this information later.

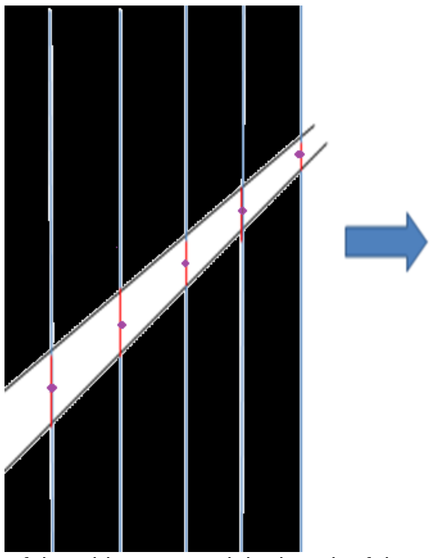
Figure 6: The mean of the white area and the length of the white area traversed by the scanline are stored.

5. Clustering all such line candidate points together, according to their positions and the “white area lengths”. The reason here is that the white area length for points that are on one line should be broadly similar, which would not be the case if we are looking at an also white robot.
6. Attempting to link it with the some points closest to it which still has a white ratio of around 1 for the line between the 2 points.
7. Finding suitable boundaries if a proposed line addition would be within the boundary established by the white area lengths of previous points belonging to a line, see Figure 7.

This should improve the effectivity in finding distant corners compared to an error function that would be the same over the entire line. We will investigate how to weigh previous line spots in an optimal way. Once a set of spots that make up a line can no longer be elongated further, it is stored as a line candidate, the starting and endpoint are also stored, and another spot is chosen as a starting point of the algorithm.

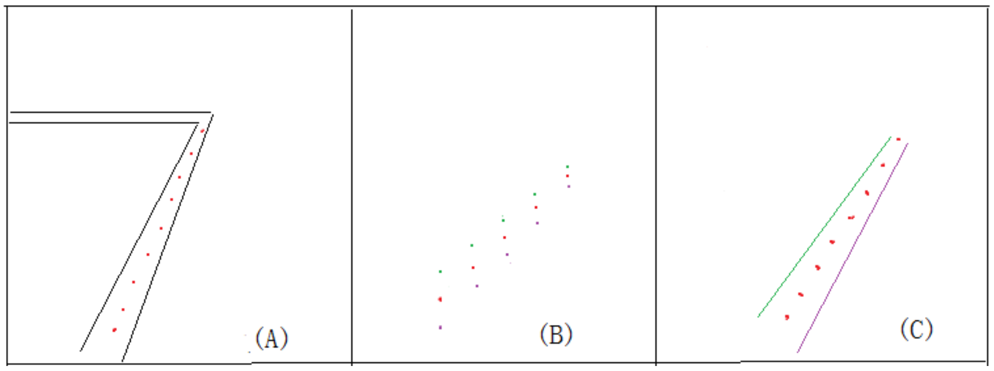

Figure 7 A: Sample input, vertical scanlines detect the transition and the mean of the lines is calculated. B: Using the stored data we recreate the original line boundary for purposes of generating picture C: where additional line candidates are evaluated based on whether or not they fit into the re-established boundary

8. If another line candidate is found, attempt to use this line as a direct elongation of any already preexisting line. For this, we first check if any spot belonging to this line candidate already belongs to another line candidate. If we find that this line overlaps with other lines, but cannot be used as an elongation, we have a corner candidate.
9. Distinguishing 4 different types of corners. T-Corners are t-shaped corners where 2 lines intersect, and where the starting point of one of those lines is also the point of the Intersection. L-Shaped corners exist when the end point/starting point of one line is the starting point/ end point of the other. X-Shaped corners exist when the intersection is neither a start nor an end point for either line. We hope to be able of inferring further information from the respective “white area lengths” for such intersections. In particular, we hope that white area length can be used to prevent us from mistaking the goal net for a set of corners.

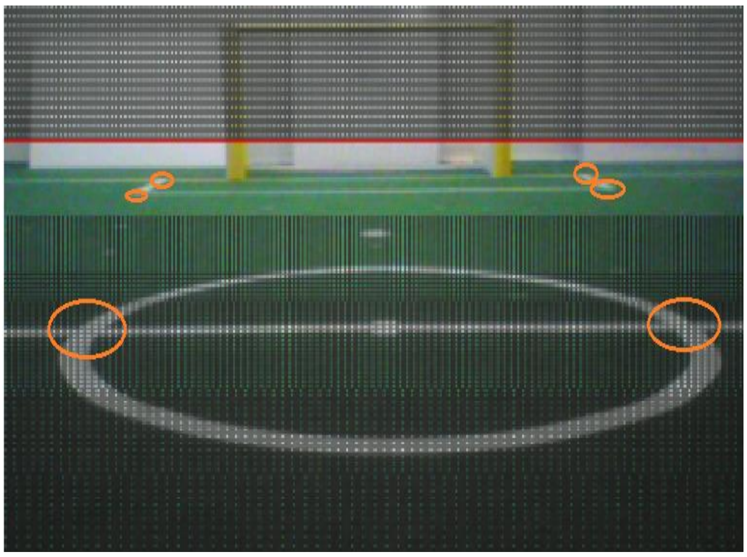
Figure 8: Scanline density is increased in the orange circles [8].

10. Save the coordinates of “interesting features” and seed scanlines around those coordinates for the next picture. See Figure 8. Save the number of corners detected, in the next frame; compare that number with the current number of detected corners. This could be used as a quality benchmark concerning image trustworthiness for self-localization.

5. Summary

We hope to combine and improve upon existing aspects of the current state of the art. First, we aim to use smart scanlines, and also make them flexible by introducing a basic feedback loop.

Second, we wish to expand on the Dutch NAO team’s corner detection method by making use of all data their algorithm would calculate anyway, and by making our equivalent of their error function use this data. We will attempt to exploit that data even further, but cannot specify how much positional information we can extract from the knowledge of “white area length” as a function of position/distance along a line. Under good circumstances, this could be a powerful tool for localization.

Third, by integrating our scanline feedback approach with our approach for detecting corners, and then measuring the quantitative output (number of detected corners etc.) of each loop, we hope to arrive at a measurement for image processing quality. This could perhaps be used to improve localization by yielding information that can be used for improved weighting of different information.

6. Timeline

This master thesis is planned for the time between the April and September 2014 and scheduled as follows:

- **April:** Software /development setup and test set (images) collection
- **May / June:** Implement (coding)
- **June / July:** Experiments / testing, methods review
- **July / August:** Coding, experiments
- **August / September:** Writing

Bibliography

[1] Thomas Reihardt: *Kalibrierungsfreie Bildverarbeitungsalgorithmen zur echtzeitfähigen Objekterkennung im Roboterfußball*. Masterarbeit. HTWK Leipzig (2011)

[2] Martin Engel: *Anwendung von maschinellem Lernen zur echtzeitfähigen und kalibrierungsfreien Erkennung von humanoiden Fußballrobotern*. Masterarbeit. HTWK Leipzig. Medieninformatik(2012)

[3] Manuel Bellersen: *Austauschbare Module zur Integration von KI in den Nao-Simulator "Spielwiese"*.Bachelorarbeit. HTWK Leipzig(2012)

[4] David Vernon: Machine Vision*. Automated Visual Inspection and Robot Vision*. Book. Department of Computer Science Trinity College Dublin Ireland (1991)

[5] D.H. Ballard: *Generalizing the Hough Transform to detect arbitrary shapes*. Report. Computer Science Department, University of Rochester, Rochester; NY 1462, U.S.A (1979)

[6] RoboCup Technical Committee (2014 rules*): RoboCup Standard Platform League (NAO) Rule Book*. (2014)

[7] NAO Software 1.14.5 documentation. *https://community.aldebaran-robotics.com/doc/1-14/family/nao_h25/index_h25.html.* (Last visit: 10.03.2014)

[8] Amogh Gudi: *Feature Detection and Localization for the RoboCup Soccer SPL.* Project Report. University of Amsterdam.(2013)

[9] Stefan Tasse, Sören Kerner, Oliver Urbann, Matthias Hofmann, Ingmar Schwarz : *Nao Devils Dortmund Team Report* (2011)

[10] Alexander Härtl, Ubbo Visser, Thomas Röfer: *Robust and Ecient Object Recognition for a Humanoid Soccer Robot*. In RoboCup 2013: Robot Soccer World Cup XVII, Lecture Notes in Artificial Intelligence (2013)

[11] Thomas Röfer, Tim Laue, Judith Müller, Armin Burchardt, Erik Damrose: *B -Human Team Report and Code Release (2011)*

[12] Thomas Röfer; Tim Laue; Judith Müller; Michel Bartsch; Malte Jonas Batram: *B -Human Team Report and Code Release (2013)*