Nao detection with a cascade of boosted weak classifier based on Haar-like features

Duncan S. ten Velthuis
0577642

Bachelor thesis
Credits: 18 EC

Bachelor Opleiding Kunstmatige Intelligentie

University of Amsterdam
Faculty of Science
Science Park 904
1098 XH Amsterdam

Supervisor
dr. A. Visser

Informatics Institute
Faculty of Science
University of Amsterdam
Science Park 904
1098 XH Amsterdam

July 18th, 2014

Abstract

In robot football (RoboCup Standard Platform League), the detection of Nao Robots is essential for building a world model. Reliable knowledge about the environment allows the robots in the team to avoid obstacles, have better goal scoring opportunities and pass the ball to other team members. Most detection algorithms in literature are too computational expensive in order to be benificial for the Nao Robot. In this thesis, we propose an approach that uses the Viola and Jones' method for detecting Nao Robots. We hypothesize that this method can be competitive against current state of the art Nao Robot recognition methods. This is accomplished by training and testing strong classifiers based on a cascade of boosted weak classifiers on simple Haar-like features. The proposed approach has competitive results, but still underperforms against current state of the art methods.

Contents

1 Introduction

Recognizing Robots (Nao Robots) during Standard Platform League (SPL) football matches has been a longstanding research topic. Most algorithms used for recognition are computationally expensive and lead to a higher overall loss than gaining a tactical advancement [Reinhardt, 2011, Fabisch et al., 2010]. Detecting other Nao Robots on the playing field helps with tasks such as path planning, obstacle avoidance, better goal scoring opportunities and passing the ball to other team members.

Cascade of Weak Classifiers is a technique to recognize objects, which previously has only been used inside the Simulation Rescue League. Not only did the researchers achieve a high performance with this technique but it was also relatively low in complexity [Flynn, 2009]. Therefore, it is able to run on computationally limited hardware such as the Nao Robot. Having all the algorithms in the Nao Robots low in complexity enables the Nao Robots to use multiple algorithms during the game. These algorithms are required to address different problems that arise when Robots play football games autonomously.

In this thesis, a video dataset is collected from the Nao Robots during football matches which are played according to the SPL football rules. All the data has been annotated by hand and will be used as ground truth when comparing the various algorithms on their performance. This dataset will be divided into a training set and a test set. A boosted cascade of decision trees [Flynn, 2009] will be trained using the training set, in order to find the best selection of weak classifiers to detect Naos. The trained classifier will be tested on the test set and compared to an existing SPL perception technique of the same test set. The hypothesis of this paper is that a higher performance will be achieved with the boosted cascade of decision trees than with the current state-of-the-art recognition.

Both algorithms will be evaluated based on their performance using the same video to compare both algorithms and annotating by hand a valid experiment is ensured to prove which of the algorithms has the highest performance.

This thesis is structured as follows, Chapter 2 gives an overview of the related research regarding object detection for Nao robots. In Chapter 3, an introduction is given to the robotics field and the RoboCup initiative is explained. Chapter 4, gives an overview of the computer vision theory that is related to this thesis. Thereafter, the Viola and Jones method is explained in detail in Chapter 5. In Chapter 6, the method and approach are illustrated and in Chapter 7 the result will be presented. Furthermore, Chapter 8 will analyze and discuss the results of the algorithm. Finally, in Chapter 9 the

conclusion is given and directions for future research will be proposed in Chapter 10.

2 Related Work

This thesis aims to find an alternative way to detect Nao Robots in images. This chapter gives an overview of the related research that has been done regarding Nao Robot detection. During a robot football game it is essential to identify other Nao Robots on the field, in order to adopt an appropriate strategy to approach the football game. For this reason, Nao Robot detection is an important topic in the RoboCup SPL and various approaches have been tried which have been published in the literature on this topic [Fabisch et al., 2010, Reinhardt, 2011, Engel, 2013]. In this chapter, first the various related Nao Robot detection algorithms are described and thereafter an argumentation is given for the approach proposed by this thesis.

Currently, there are only two teams in the Standard Platform League (SPL), who have successfully implemented a Nao Robot detection algorithm. B-Human [Fabisch et al., 2010] detects Nao Robots based on color segmentation of the blue and pink jerseys which the Nao Robots wear during SPL games. When a possible jersey is found they use additional scans around the area to ensure a Nao Robot has been detected. This approach can reliably detect Nao Robots up to 2.2 metres away. SPL team HTWK's [Reinhardt, 2011, Engel, 2013] method for detecting Nao Robots uses a simple neural network of 4 nodes. In this approach first the known objects, which are the goal, ball and lines are detected by pre-processing the image based on a scanline technique. Thereafter, the unidentified regions in the image that are inside the field are fed to the neural network as input in order to classify whether this region is a Nao Robot.

Outside of the RoboCup SPL, there is a similar computer vision problem, which is the detection of humans. Nao Robots have a shape similar to that of humans, hence similar techniques can be implemented.

Historgrams of Oriented Gradient (HOG) by Dalal and Triggs [2005] was the first paper to report a successful approach for human detection employing HOG. Their work shows that using HOG as low-level features can be used to describe the appearance and shape of the object through the distribution of local intensity gradients and edge directions. It has been shown that using HOG is better for detecting shapes, where Haar-like features are better at discriminating between face-like objects [Zhu et al., 2006, Negri et al., 2007].

However, it has been noted that the use of HOG is still too complex for Nao Robots to use during SPL games [Fabisch et al., 2010].

Implementing local scale invariant features (SIFT) Lowe [2004] was able to recognize humans using local image features which are invariant to scaling, rotation, translation and more suited for lighting changes. By filtering for stable features in scale space, image keys are created that identify candidate objects based on nearest neighbor indexing.

One of the earlier successes in finding the location of an object in an image was achieved by a technique using a boosted cascade of simple features [Viola and Jones, 2001], a Machine Learning technique that trains a classifier to detect an object based on simple features which is capable of processing images rapidly while achieving high detection rates. Lienhart and Maydt [2002] extended this work using rotated Haar-like features decreasing the false positive rate by 10%. Flynn [2009] successfully implemented the Viola and Jones [2001] technique in the Robocup Rescue Simulation League. They used the classifier to detect the faces and feet of victims, while detecting plants and chairs for localization yielding positive results for possible expansion into SPL.

3 Robotics

Robotics is the field of technology that focuses on robots in the broadest sense. Classical thinkers already thought of machines that can be operated by humans or machines that can operate autonomously. Actual research into the use of robots did not grow until the twentieth century Nocks [2007].

The field of robotics has developed rapidly since the first time the name was used in a Czech play in 1920. Much has happened since that time. For example: Shakey, self driving cars, Aibo, Nao Robot, DARPA challenge, drones and new practical uses are still being produced to this day. Currently most research is done by the military, trying to build robots so no human beings need to be put into danger or the battlefield.

Healthcare is where a lot of research is done with robots. For example, in certain operations robots or remote-controlled robotic arms are being used which facilitates performances of certain procedures [Barbash and Glied, 2010]. Robots are also used in therapeutic activities for the elderly in long-term care facilities to improve their communication and interaction skills. This is done through the usage of animal like robots and having the elderly interact with these robots[Wada et al., 2004]. These examples show that the field of robotics is broad and that new technologies can be useful to society.

Research is essential in this field of work to test new abilities and improve the skills of robots. The RoboCup is one of the most important driving events with regard to research in the field of Artificial Intelligence. The RoboCup will be discussed in the following section.

3.1 RoboCup

One of the biggest events involving robotics is the RoboCup, a league created not only for Artificial Intelligence research, but also research from different fields to work towards a common goal. This goal is to have a game in 2050, between a team of autonomously playing robots and the winners of the FIFA World Cup of that year[1]. The RoboCup should be seen as an incentive to push the boundaries of Robotics research and Artificial Intelligence to a new and higher level. Having a real-life competition brings the possibility for evaluating algorithms and theories in a distinct environment, which would not be able to be re-created without an immense budget for each individual team Research areas that are covered are multi-agent systems, complex behavior, context recognition and Computer Vision.

The RoboCup not only includes football competitions but also RoboCup Rescue for the use of robotics during emergencies, RoboCup @Work for robotic systems for use in an factory setting, the RoboCup @Home for robotics used in everyday task and RoboCup Junior, which is used to create enthusiasm under children for the robotics field. The RoboCup in general is a pillar for robotics to build a better understanding under the population on the usefulness of robotics.

For researchers it is a platform to exchange ideas, share solutions and with the help of the annual symposium that is given at the end of each RoboCup a place to publish research[2].

3.1.1 Leagues

The following is a short description of all leagues:

@Home - The RoboCup@Home league is the largest international annual competition for autonomous service robots. It focuses on creating independent assistive robot technology for domestic use and is part of the RoboCup initiative. The robots are evaluated through a series of benchmark tests in a realistic non-standardized home environment. The focus of these tests lies on the following but is not limited to it: Navigation and Mapping in dynamic

[1] `http://www.robocup.org/`

[2] `http://www.robocup2013.org/symposium-2/`

environments, Computer Vision and Object Recognition under natural light conditions, Adaptive Behaviors, Human-Robot-Interaction and Cooperation, Ambient Intelligence, Object Manipulation, Behavior Integration, Standardization and System Integration.

Rescue league - RoboCup Rescue focuses on assistance of robotics with disaster management in hostile environments which can be hazardous for humans. RoboCup Rescue was started after Kobe City was hit by the Great Hanshin-Awaji earthquake on the 17th of January 1995. Killing thousands and affecting hundreds of thousands. By analyzing the Hanshin-Awaji earthquake it was concluded that information systems should be created with the following requirements:

- Rapid support for the planning of disaster alleviations, search and rescue.
- Dependability and robustness of the information system during emergency and routing operations.
- Collection, accumulation, relay, selection, summarization, and circulation of essential information.

Given the previous requirements, the goal of the RoboCup Rescue project is to encourage research and development in this socially significant field at multiple levels involving physical robotic agents for search and rescue, information infrastructures, multi-agent team work coordination, personal digital assistants, a standard simulator and decision support systems, evaluation benchmarks for rescue strategies and lastly robotic systems that are all integrated into an inclusive system in the future[3].

@Work - RoboCup @Work is a new competition in the RoboCup that targets the use of robots in work-related scenarios. It aims to foster research and development that enables use of innovative mobile robots equipped with advanced manipulators for current and future industrial applications, where robots cooperate with human workers for complex tasks ranging from manufacturing, automation, and parts handling up to general logistics. The University of Amsterdam is active in this competition [Negrijn et al.].

[3] `http://www.robocuprescue.org/`

3.1.2 Football

The main focus of the RoboCup is the Football league, where the research is focussed on cooperative multi-agent systems in dynamic adversarial environments. In this league every robot is fully autonomous. The football league is also a great opportunity to entertain and educate the general public on the field of robotics.

The football competition of the RoboCup is divided in the following competitions:
Simulation[4] - The Simulation League is one of the oldest leagues and focuses on Artificial Intelligence and Team strategy. No real robots are used, instead the league comprises of 2 leagues: 3D League and 2D League. In both leagues teams play with eleven independent simulated robots (agents) utilizing team strategies using multi-agents systems. The 3D league uses humanoid robots simulated in a 3D environment, striving to reproduce the software programming challenges that occur when working with real physical robots.

Small Size League[5] - The primary focus of the Small Size League is of multi-agents systems in a dynamic environment with a hybrid centralized system. Each team plays with six circular robots which use wheeles as means for transportation and they must be 18 cm in diameter and no higher than 15 cm. The Robots play football on a green field of 6 meters long and 5 meters wide. All robots are tracked using two overhead cameras attached 4 meters above each half of the playing surface. Using a standardized vision system that tracks colored markers on a top of each robot, each team can control their robots. Off-field computers are used for perform most of the computation needed for playing a football game, controlling each robot and communication of the referee decisions.

Middle Size League[6] - In the Middle Size League both teams play with five autonomous robots on a 18 by 12 meter field. Each robot is equipped with an onboard computer to analyse the game with its sensors (ie. a 360 degree camera) and control the robot while communicating with its teammates over a wireless network. Each team has custom build robots which only need to meet requirements such as a maximum weight of 40 kg and a maximum height of 80 cm. All teams develop their own mechatronics and software. Research of this competition is focused on autonomous behavior and embed-

[4] `http://www.robocup.org/robocup-soccer/simulation/`
[5] `http://robocupssl.cpe.ku.ac.th/`
[6] `http://www.robocup2013.org/middle-size-league/`

ded systems.

Humanoid League[7] - In this league human-like robots play football with sensors which mimic the same sort of senses as the human senses. No additional sensors are allowed that would otherwise give the robots more ways to get perception of the football game than a human would be capable of. Challenges for this competition lie in dynamic walking and kicking while maintaining balance and overall perception of the game.
The Humanoid League comprises of three size-classes:

- Kid-Size (40-90 cm Height): Play games off 3 vs 3.
- Teen-Size (80-140 cm Height): Play games off 2 vs 2.
- Adult-Size (130-180 cm Height): One Robot per team, which switch between taking a penalty and goal keeping.

All three competitions play on fields of 9 meters long and 6 meters wide, though Kid Size uses a smaller goal and each competition has his own size ball.

3.2 Standard Platform League

Standard Platform league (SPL)[8] is the last football competion of the RoboCup. The focus on this Thesis lies in the detection of Nao Robots during SPL football games. Here follows a short introduction of the SPL, the robots used and the challenges that exist in this league. In contrast to other leagues SPL teams use standardized robots in each team. The French company Aldebaran builds and repairs each Nao Robot, ensuring no alterations are made by teams and the focus lies on programming software.

The SPL started in 1996 using the early version of Sony's AIBO, a four legged dog-like robot which was at the time also used in the 3D simulation league. Beneficial is that algorithms could be built and tested in the Simulation League and be transferred and tested to real-life conditions. Revealing possible real-life issues that probably could not have been found with only testing in a simulator eg. inaccuracies with motors used in robots. Without the SPL the transfer between simulation to hardware would not be possible and not only does SPL benefit from the work done in the Simulation League, but also vice versa [Xu et al., 2010a]. Having a competition with standardized robots utilizes the possibility for teams to exchange not only ideas and

[7] `http://www.informatik.uni-bremen.de/humanoid/bin/view/Website/WebHome`
[8] `http://www.informatik.uni-bremen.de/spl/bin/view/Website/WebHome`

solve problems, but also share actual code. The team BHuman has been sharing parts of their framework and motion solutions for some time now **?**. Other leagues who play with different hardware cannot as easily exchange software. SPL team Berlin United and FU-manoids Xu et al. [2010b] are building a cross-platform solution. Another benefit of the SPL using one platform is the possibility of mass production and keeping costs lower than when building custom unique robots. Not only is there no need for having separate specialised members in your team for hardware modification,, but time can also be saved by not having to deal with trial and error problems that evolve with development of new hardware techniques.

Issues regarding playing autonomous football with Nao Robots in the SPL create many problems wherefore much research has to be done in order to fix these problems. Wiggers and Visser [2013], Reinhardt [2011], Mellmann and Scheunemann [2011], Fabisch et al. [2010] Examples range between: individual localisation, communication protocol, motions, team strategy, behavior, world models etc. Not only is the SPL league pushed by adding new rules each year that are more similar to official FIFA rules, but the league also pushes its members by adding challenges so that the teams must go beyond the current SPL football rules.

3.3 Nao

In 2006 a call was made by SPL to find a new standardized bi-pedal humanoid Robot to replace the AIBO in the league, Alderbaran won this call when they introduced their robot Nao. Since its introduction the Nao Robot has undergone almost yearly upgrades bringing the current version to 5.

The current features of the Nao Robot are listed below.

- Body with 25 degrees of freedom (DOF) whose key elements are electric motors and actuators
- Sensor network: two cameras, four directional microphones, sonar rangefinder, two IR emitters and receivers, one inertial board, nine tactile sensors and eight pressure sensors
- Various communication devices, including voice synthesizer, LED lights, and 2 high-fidelity speakers
- Intel ATOM 1,6ghz CPU (located in the head) that runs a Linux kernel and supports Aldebaran's proprietary middleware (NAOqi)
- 48.6-watt-hour battery

- Python and C++ with OpenCV support
- Connectivity trough Ethernet and Wifi

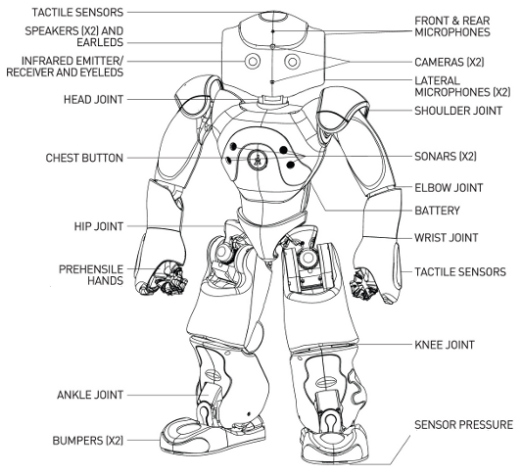


Figure 1: Nao robot (Courtesy Aldebaran Robotics)

The Nao Robot needs to autonomously solve the problem of playing SPL football games. Which means more objects needs to be detected than only other Nao Robots. For example: what is the field, where are the lines, where is the ball, where are the goals. In this thesis the focus is on the problem of detecting other robots.

One way of detecting other Nao robots is with the built-in sonar and the use of sophisticated modelling. Fabisch et al. [2010] proves that while some detection can be done, the sonar is quite inaccurate and can not detect Nao robots that lie on the ground.
Early vision based solutions involved detecting everything else (ie field, lines, ball and goals) and to consider significant gaps as obstacles.

4 Computer Vision

The focus of this research is to find possible solutions for detecting Nao Robots during SPL games while using the limited resources the Nao Robot gives us. Using Computer vision for detecting objects in images is a long-standing research topic Viola and Jones [2001], Fabisch et al. [2010], Dalal and Triggs [2005], Reinhardt [2011]where many challenges still need to be overcome, even without taking into account the limited resources on a NAO robot.

There are two ways of classifying objects with computer vision, *object recognition* and *object detection*, the first can tell if an object is in an image and the latter where an object is in an image. One of the first proposed techniques for object recognition was with the use of color histograms as an early view-based aproach, by Swain and Ballard [1991]. This technique is based on simple whole-image similarity metrics such as colors. Ground-truth pictures of objects are represented by the combination of its colors in a color histogram which is represented in a vector. Different vectors can characterize different objects and extracting a vector of a similar object in a different image would result in a similar vector, thus gaining the capibilty to recognize them as similar. A major issue with this approach is that it is very sensitive to the lighting conditions. Using local scale invariant features Lowe [1999] was able to recognize objects using their local image features which are invariant to scaling, rotation, translation and more suited for lighting changes. By filtering for stable features in scale space, image keys are created that identify candidate objects based on nearest neighbor indexing.However computing these features for every image is computational expensive. One widely successful algorithm for detecting faces is the Viola Jones algorithm Viola and Jones [2001]. This algorithm solves both tasks object *recognition* and *detection* in a very computationally efficient way. In the following section we will explain the Viola Jones algorithm in depth and how we can apply it to Nao detection. We compare it to a state of the art approach of the HTWK.

5 Viola and Jones

One of the earlier successes to identify *where* in an image an object is located was achieved with the Viola and Jones technique using a boosted cascade of simple features [Viola and Jones, 2001]. In this chapter we will discuss the Viola and Jones algorithm with the extended features of Lienhart and Maydt [2002].

Viola and Jones's Machine Learning algorithm trains the classifier using the Haar-like features found in the training samples.

The simple features used in Viola and Jones algorithm saw its orgin in Haar basis functions from Papageorgiou et al. [1998]. For real-time object recognition the feature extraction needs to be computed fast and Haar-like features are one of the simplest and easiest to compute Flynn [2009].

The algorithm trains on a set of training images, a positive set consisting solely of the to be trained object and a set of negative images without the object which is used to discriminate between useful and not useful features.

The positive images are all scaled to a same small sized window making it possible for scale invariance of the object by searching for the object with an increasingly scaled subwindow that slides over the images. Training with a smaller sized subwindow for the training images will decrease the number of features, but at the same time speeds up the training-phase of the algorithm.

In order to have object detection in real time classifiers need to be fast. Computational it would be expensive when in large images all features were used for classifying, whereas in large images pixels are in such a large quantity that some can become redundant, therefore approximating their values can be done without losing much off the original values. Hence images are resized to a smaller image, so faster detection can be done without significantly losing peformance.

As an intermediate representation of the image, Viola and Jones transform the image to an integral image. The pixel values of the integral image are the sum of pixels left and above of the orginal image. for example, the new values are denoted for coordinates (x,y), where as i(x,y) are the values of the orginal image and ii(x,y) the old image, with

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y')$$

(See fig 2)
Using the recurrence formula, where s(x,y) denotes the cumlative row sum, $s(x, -1) = 0$ and $ii(-1, y) = 0$

$$s(x, y) = s(x, y - 1) + i(x, y)$$
$$ii(x, y) = ii(x - 1, y) + s(x, y)$$

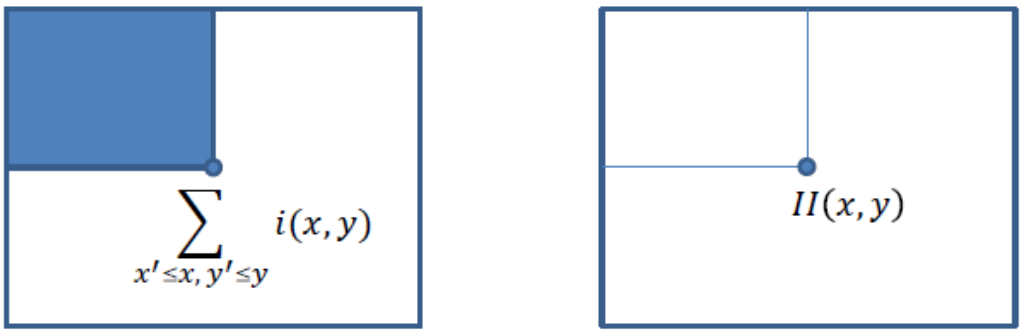


Figure 2: Left: original image. Right: intregal image.

The integral image can be computed in one run over the orginal image. The values of the integral image ii(x,y) contain the sum of pixels above and

left of the orginal images i(x,y), (see fig. 2 and 3) [9]. And pseudo code for building an integral image can be found in the appendix 12.

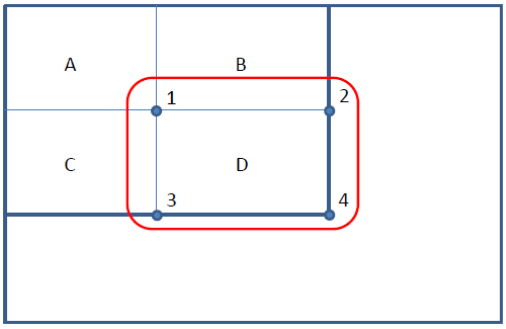
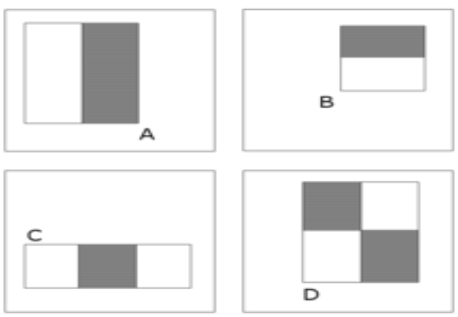
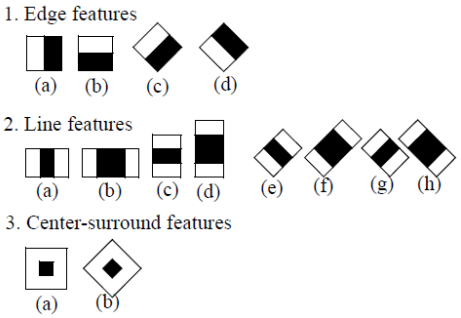


Figure 3: The sum of pixels in space D can be extracted with the values at position 1 to 4. With (4+1)-(2+3)=D

Haar-like features are rectangle features based on the subtraction of the sum of the pixel values found in the white part of the retangle with the sum of values found in the black part of the retectanle (see fig.4b, 4a). These features are found using rectangular sub-windows that are placed over the integral image. Which is one of the reasons that the Viola Jones algorithm is so computationally efficient, since computation of Haar-like features only needs four operations for each feature, once you have computed this integral image.



(a) Viola and Jones original rectangle features.



(b) Lienhart et al. extended features used in our current classifier.

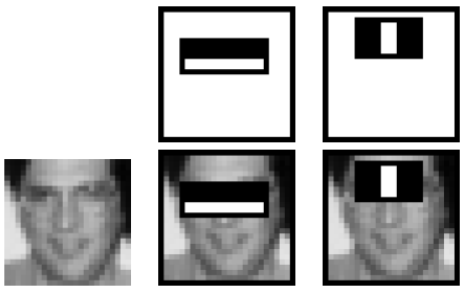
These features are placed and scaled independently in the x- and y-direction of the integral image, generating an overcomplete set of features.

[9]Images from: http://www.cvip.uofl.edu/wwwcvip/education/ECE523/Spring%202011/Lec7.pdf

Lienhart and Maydt [2002] calculated that from a 24x24 image 117.941 features are gained using all of their extended features, see figure 5a.

Feature Type	w/h	X/Y	#
1a ; 1b	2/1 ; 1/2	12/24 ; 24/12	43,200
1c ; 1d	2/1 ; 1/2	8/8	8,464
2a ; 2c	3/1 ; 1/3	8/24 ; 24/8	27,600
2b ; 2d	4/1 ; 1/4	6/24 ; 24/6	20,736
2e ; 2g	3/1 ; 1/3	6/6	4,356
2f ; 2h	4/1 ; 1/4	4/4	3,600
3a	3/3	8/8	8,464
3b	3/3	3/3	1,521
Sum			117,941

(a) Amount of extended features found from an 24x24 integral image



(b) Example of how features are calculated over images

[10]

With this method of extracting Haar-like features from an integral image make an overcomplete set of features, far exceeding the number off pixels in the original image.

Most algorithms deal with scale invariance by scaling the original image [Papageorgiou et al., 1998], building an 'image pyramid' where each image in the pyramid is 1.25 times smaller than its previous. Then with this set of images feature extraction is computed. Compared to Viola and Jones integral image and Haar-like features enables features to be computed at any position in the image and at any scale in only a few operations [Viola and Jones, 2001].

Because Viola and Jones method compute an over-complete set of features some form of reduction is required. As with building an integral image not all variables are needed to deduce the important variables that can tell the most about its region. Weak classifiers can effectively pick out a small amount of useful features which have significant variety.

5.1 Boosting

To find the important features boosting is used. Boosting comes from the machine learning model 'probably approximately correct' (PAC), with the idea that a set of weak classifiers, each of which could perform slightly better than random guessing, could be boosted into a strong classifier. Boosting was first described and build by Freund et al. [1999] for combining weak classifiers whose performance is significantly better that of the individual classifier.

[10]Images courtesy form [Viola and Jones, 2001, Lienhart and Maydt, 2002]

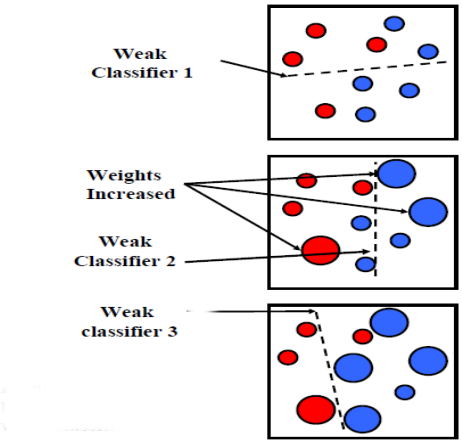


Figure 6: Showing construction of 3 weak classifiers each classifying on a set of weighted features based on the previous classification

Viola and Jones use the Adaptive Boosting (AdaBoost) method, which is adaptive in that subsequently classifiers are adapted in favour of features misclassified by the previous classifiers. Weights are assigned to the features misclassified by one of the earlier classifiers which are then used when training the next classifier. Once a set threshold is made (based on having a small enough false alarm rate) all classifiers are combined through a weighted majority vote. Resulting in one strong classifier that can detect with high accuracy if a new image contains the trained object. In the Viola and Jones method a weak classifier is restricted to select one Haar-like feature which best discriminates between the positive and negative examples. For each feature the weak classifier determines the best threshold classification function that minimizes the number of misclassified examples (see 6 and for pseudo code off AdaBoost see appendix 13).

More formal, a weak classifier $w(x)$ has features f, parity p which indicates the direction of the inequality sign and some threshold σ:

$$w(x) = \begin{cases} 1 & \text{if } pf(x)p\sigma \\ 0 & \text{otherwise} \end{cases}$$

where x is a sub-window of an image. Which means that the value of some feature exceeds some threshold, which has to be learned, the image is classified as positive [Flynn, 2009]. Viola and Jones' complete algorithm can be found in the appendix.

5.2 Cascade

Viola and Jones introduce a method for constructing a cascade of strong classifiers each build from a set of weak classifiers. To achieve a high detection rate and with a reduced computation time. The main idea is that the first layer is trained from a smaller and more efficient boosted classifiers, which can quickly reject most images for failing having the right positive features, trained in the training stage, while detecting nearly all positive images. Each stage is constructed training weak classifier using AdaBoost.

The final cascade can reject most of the images/subwindows with a few operations:

1. Evaluate the subwindows features
2. Compute the weak classifier for each feature
3. Sum the weak classifiers into one strong classifier

The final boosted cascade detection process has sort of a degenerate decision tree structure. What follows is the process of detection within the cascade.

Detection of an object on an unknown image is performed by sliding a sub-window over the image in x- and y-direction. Just like the training stage where the features are scaled this window is scaled after each round over the image so scale invariance of the trained object is solved (when the object appears bigger in the image). With the constraint that the window can not be smaller than the positive training images. The first layer with highest detection rate will evaluate the subwindow with his strong classifier which was trained by combining the weak classifier. This first layer will have the smallest amount of weak classifiers possible while still reaching a certain threshold based on rejecting a maximum false alarm rate. A positive result will trigger the second layer of the cascade to evaluate the image again with his strong classifier, which was build with more weaker classifiers that were found by boosted features from the previous layer. These weaker classifiers have to a achieve a smaller maximum false alarm rate, whereas their chance for a false image is smaller than the previous layer which already has a percentage of rejection false images. And each layer the strong classifier are build from more weak classifiers and need to hold a smaller maximum alarm rate.

Summary: Each layer feeds the next layer an image it classified as positive or rejects the image on which than no further classification is don, making it possible for quickly rejecting false images (see fig.7).

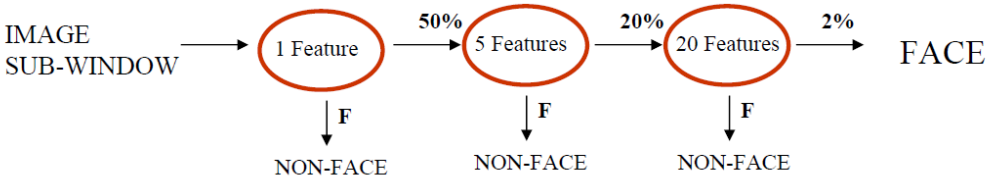


Figure 7: First weak classifier achieves 100% detection rate and about 50% false positive rate. Second weak classifiers achieves 100% detection rate and 40% false positive rate. Third feature classifier achieve 100% detection rate with 10% false positive rate (2% cumulative of all three)

[11] The overall max false alarm rate of the cascade is the root power of number of stages in the cascade and the given threshold for maximum false alarm rate, e.g. a 20 layer cascade with an maximum false alarm rate of 0.5 per layer will have an overall maximum false alarm rate off $0.5^{20} \approx 10^{-6}$.

5.3 Complexity of the Viola and Jones method and why it was chosen

To detect if a Nao is in an image, subwindows with the integral image of the original are fed to the cascade. For each subwindow $O(FS_f)$, where F is the number of Haar-like features used by the cascade and S_f is the size of the pixels contained by each feature. Using the cascade most sub-windows can quickly be discarded and only True Positive and False Positives will run the whole cascade. The total number of operations is $O(NFS_F)$ where N is the total amount of subwindows and is a function of the image size and number of scales included in the search [Flynn, 2009, p.36].

Viola and Jones show its promise for detection based off its previous results, it is computationally efficient achieved by using integral images and weak classifiers build with AdaBoost algorithm to reduce the number of features, using scaled invariant subwindows reducing the need for building a scaled image pyramid and the boosted cascade for quickly rejecting false images. All these methods achieve a classifier algorithm that can be used for real-time detection.

[11]Images 6 and 7, courtesy from `http://www.cvip.uofl.edu/wwwcvip/education/ECE523/Spring%202011/Lec7.pdf`

6 Method and Approach

For our Research it is important if we can identify Nao Robots during SPL games, making it important for developing good classifiers that can achieve a high performance during such games.

To train our classifier we used 3.397 images collected from Nao Robot cameras thanks to Thomas Reinhardt from HTWK. These images were collected during multiple games played by HTWK in different places (Eindhoven, Dortmund, Leipzig etc). Every image was screened for Nao Robots and annotated by hand by selecting the chest button and the middle between where the two feet touch the ground (see fig. 8). From 3.397 images 720 had between 1 to 4 Nao Robots in them, adding up to 1194 Samples of Nao Robots, see figure, of 20 samples and their positions. All images were 640 x480 in size.

To train our classifier we created 4 sets of images: a positive set consisting solely of Nao Robots on which our classifier will be trained, another positive set of images consisting of Nao Robots on which our classifier will be tested. And similar two negative sets, images with no Nao Robots in them for both training and testing purpose.

A Python script was build that would automatically extract randomly a given amount of samples from the images by copying a part of the Nao Robot from the image.

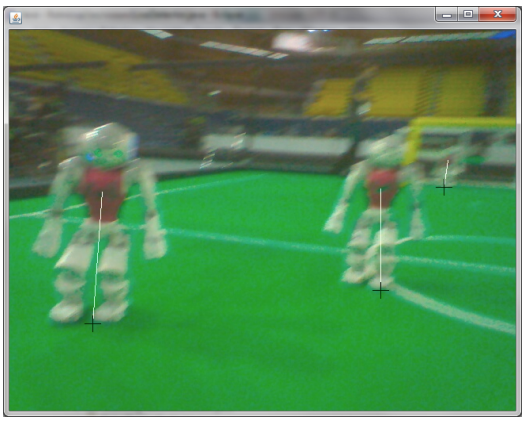


Figure 8: Example of the hand annotated data

Images with Nao Robots were not used for extracting samples during training were used for testing.

(a) Selection of sample from upper bodies used for training our classifiers.

(b) Selection of sample from lower bodies used for training our classifiers.

Parts of a Nao Robot were used, because, as discussed earlier, Haar-like features work best on detecting objects when the same features are in the same area. The algorithm is most used for detecting human frontal faces, for the reason that eyes and noses are in a fixed spot. The arms and legs of a Nao Robot are freely moving and can be in many different places when in motion, which would make it hard to detect. Which is why we trained 2 types of classifiers one on the lower body (chest button to the bottom of the feet see fig. upperB) and the upper body (chest button to top of the head (see fig. 9b).

To build a robust classifier its important to train the classifiers with a wide variation of different circumstances under which the object is captured. Normally, a classifier with a higher performance would be achieved by training on similar parts. For example, pictures of frontal faces of a Nao Robot should not be mixed with side views of the face. This way location of features will not differ which will make it more difficult to train the classifier. Negative images were used from a data-set which is freely available online [12], 3000 gray-scale images from different sizes where randomly used for either training the classifier or testing together with negative images gained from Thomas Reinhardt dataset. The only restriction for negative images is that it does not contain a Nao Robot.

[12]http://tutorial-haartraining.googlecode.com/svn/trunk/data/negatives/

For samples we first trained on the bottom half, everything down from the chest button down to the feet. For the second experiment we used the top half only using the head and shoulders of the Nao Robot. For the third experiment we used the whole body and for the fourth we used the combination of the head and shoulders classifier together with the legs classifier.

For both legs and head we used a sample of the same aspect ratio 1:1.5 [13]. Its important that for every object trained all samples keeping the same scale, so the classifier learns which features are staying in the same place.

We scaled the sample for our first experiments to 16 x 24 pixels, as a result of Flynn [Flynn, 2009, p. 40] found using a bigger window too not result in higher classification. Liendhardt et al Lienhart et al. [2003] found similar result showing that windows larger than 20 x 20 do not yield higher hitrates. Though in later experiments we increased the window to 20 x 30 with much higher performance.

OpenCV program CreateSample.exe was used for processing our samples which the program size normalizes and stores them in a vector file. By cutting given amount of samples automatically from the indicated images.

6.1 Training

Training and testing was done on:

64 Bit Ubuntu 13.10
Intel Core i7-2600 CPU @ 3.40GHz Octacore

In the beginning both OpenCV 2.4.8 Haartraining and TrainCascade were used to train different classifiers. The difference lies in Haartraining is an older version and can still be used to call OpenCV's Performance program. With the Performance program statistics can be automatically build when testing your classifiers and images are created where you can see the classifier recognizing each correct or incorrect found Nao Robots in the test set, by drawing rectangles. Unfortunally Haartraining is much slower than the program TrainCascade, which has more features and much faster training time, thanks to utilizing multi-threading. Traincasacde will find the same results when same parameters are utilized as during Haartraining, but also

[13] Width:Height

has extra features than Haartraining. The only downlfall was, that we had to build our own performance programming for extracting the performance of our trained classifiers.

The following parameters and features where used to train our classifiers:

Data - Is the place where the program stores its progress so it can be interrupted and in a later time can continue to train where it left. The final XML file used for classification is also stored here.

Vec - Contains the created Vector file by Createsamples.

Bg - Is a text file containing all the names and the location of negatives files the classifier can use for training.

NumStages - The amount of stages that need to be trained, either all stages are needed or the desired false alarm rate is met before and the training will be automatically stopped at that stage. Training can be intterupted at any time and continued at a later time even with a higher number of stages if so desired. We found that with training the upper part of the Nao Robot we needed less stages (around 17) than with the lower part (around 20). Because of the upper part we had less training samples than the bottom part.

minHitRate - The minimal desired hit rate for each stage classifier. The overall hit rate can be estimated as (MinHitRate n̂umberOfStages). Setting the minHitRate at 0.99 we ensured traininging a classifier with highets True Possitive possible. The lower miHitRate is set the less Nao Robots will be found.

maxFalsAlarmRate - The Maximal desired false alarm rate for each stage classifier. The overal false alarm rate can be determined with (maxFalseAlarmRate n̂umberOfStages). Max False Alarm Rate determinnes the rate that needs to be achieved on the negative images. During the training it needs to get lower than maxFalseAlarmRate ĈurrentStage in order to stop the training. This means that lowering the number for maxFalseAlarmRate fewer stages are needed for training, but will take longer to train each stage. Where as increasing this number more stages will need to be trained, but every stage will be trained faster.

numPos - Amount of positive samples that are used for training, from the vector file. Because, some positive samples can be excluded during training,

this number is needed to be smaller than the actual total amound of samples in the vector file. For example with training the bottom part we collected 995 samples and were able to train using 725 samples, without the training would terminate automatically because of inssufiecient amount of samples.

numNeg - Amount of negative images used during training.

w & *h* - Width and height of the samples in pixels used during training which needs to be the same as the width and height of the created samples.

MODE - Selects the type of feature sets used during traning. BASIC only uses upright feautuers. Nao Robots while moving will not always make fully horizontal photos therefore we trained using the ALL parameter, which uses the full set of upright and 45 degree rotated features set 4b.

bt - The following boosting techniques can be chosen: Gentle AdaBoost (GAB), Real AdaBoost (RAB) or Discrete AdaBoost (DAB). Based on Lienhart et al. [2003], Flynn [2009] we chose to use the default boosting GAB. Not only does GAB take less time to train, it is also able to achieve the highest accuracy. "For instance, at an absolute false alarm rate of 10 on the CMU test set, RAB deteted only 75.4% and DAB only 79.5% of all frontal faces, while GAB achieved 82.7% at a rescale factor of 1.1." [Lienhart et al., 2003, p.5]. The difference between each type lies in the ways they re-assign weights at each each stage of the algorithm/training.

featureType - Viola and Jones algorithm requires to work with Haar-like features however OpenCV can train an Cascade Classifier with HOG features. HOG features describe the shape and appearance of a local object by distribution of local intensity gradients or edge directions. HOG features are taken from images by deviding small regions within the pictures. And for all regions the sum the 1D histogram of gradient directions over the pixels of the cell. Then the combined histograms are stored in an feature vector. Regions are sized to set of pixels, for normalizing a better performance by accumulating a measure of local histogram energy over larger regions called "blocks". Finally, the results will be used to normalize all the cells within the block and get the final feature vector. Each block size is set to pixels and adjacent blocks are overlapped with each other.

Vertical Symmetry - When a trained object is vertical symmetrical the training classifier can be trained on half the object, making the time to train the

classifier much faster. A Nao Robot in motion has no vertical symmetry and such we did not use this parameter.

MaxDepth -With OpenCV strong classifiers are built from weak classifiers set in a decison tree like structure. Setting a higher depth for this tree ensures each strong classifier to become more robust in training, but also ensures a higher risk of over-fitting the features. For consistency all our classififers were trained with a depth of one and for comparisson we trained one classifier with the depth of 2 (see results).

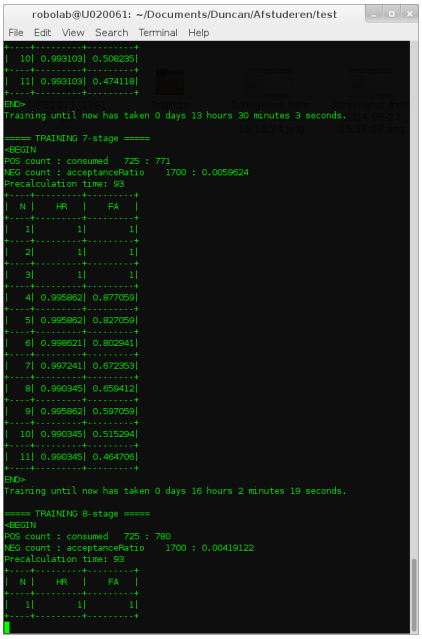


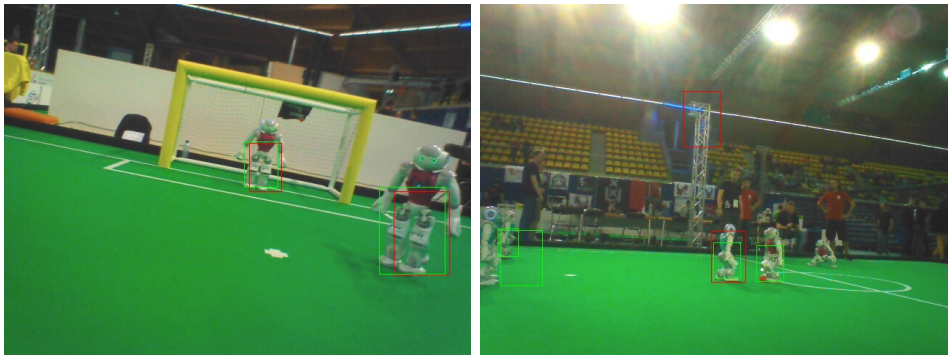
Figure 10: Screenshot of during our training, number of weak classifier that are need to achieve maxFalseAlarmRate

A program was written to test the performance of the different classifiers. 200 new Samples and 200 new images without any Nao Robots where used for testing.

7 Results

We trained around 20 different classifiers each with different parameters, or positve training samples (upper body and lower body) the following are the

most important results. With lower body (legs) 725 positive samples were trained with the Viola and Jones algorithm with 1700 negative image for referencing during training. After each training the classifiers were tested on images with 199 positive samples not used during training and 200 negative images for reference. A correctly detected Nao Robot was counted when the classifier could accurately detect a Nao Robot in the image, the same Nao Robot detected twice were counted as one correct detection.



(a) Correctly classified as Nao. Red bounding box is the output of our Classifier and Green is used for ground truth.

(b) Classification as NAO, false positives. Red bounding box is the output of our Classifier and Green is used for ground truth.

The correct detection of Nao Robots were counted as True Positive (TP). Misclassification was counted as False Positives (FP) when the background, in either the positive or negative images were detected as a Nao Robot. Robots that were not detected by the classifier are added as False Negatives (FN). Precision is the percentage of all the Correct detected Nao Robots (FP) compared with all detections (TP+FP). F1 is the measurement correct accuracy of the test with (2*TP)/(2*TP+FP+FN) = F1.

What follows are our collected measurements:

+725 20x30 Legs	TP	FP	FN	HitRate	Precision	F1
Legs	83	5	116	0.415	0.976	0.578
MaxDepth 2	42	4	157	0.21	0.913	0.342
HOG	8	3	191	0.04	0.727	0.076
16x24	63	3	136	0.315	0.955	0.475

The results of classifiers tested on 199 positve samples and 200 negative images. Trained on 725 positve and 1700 negative images.

+425 20X30	TP	FP	FN	HitRate	Precision	F1
Legs	23	11	176	0.115	0.676	0.197
Minimal HitRate 0.95	24	2	175	0.12	0.923	0.213
Upper Body	20	7	100	0.167	0.74	0.272
HOG Upper Body	3	1	117	0.025	0.75	0.048

Similar results of classifiers tested on 199 positve samples and 200 negative images. Trained with 425 positve and 1700 negative images. Both classifiers of Upper Body are tested on 120 positive samples.

+725 20x30 Legs	Legs	MaxDepth 2	HOG	16x24
Training Time (Hours:Minutes)	8:00	29:00	5:05 Min	2:39
Test Time (Seconds)	22.75	20.57	12.99	13.81

+425 20x30	Legs	MinHitRate 0.95	Upper Body [14]	HOG [15]
Training Time (Hours:Minutes)	3:31	5:20	2:47	3:47
Test Time (Seconds)	17.02	12.75	13.18	6.99

Both of the above results show how long training time was needed for each classifier. And how long each classifier would take to classify the same test set. NOTE: Both classifiers upper body are tested on 40% less images, hence the faster testing runs.

	TP	FP	FN	HitRate	Precision	F1
HTWK - Neural Network	686	26	508	0.575	0.963	0.72
Viola& Jones	467	10	727	0.391	0.979	0.559

For benchmarking we compared our results with HTWKs neural network classifier. Both classifiers were tested on the same training and test set. The Viola and Jones algorithm was our best performing lower body, classifier.

8 Discussion

8.1 Comparison with state of the art

As a benchmark the results of the system proposed are compared to HTWK's Nao Robot detection algorithm. HTWK preprocess the data raw image data according to Engel [2013]. After segmenting the input image in, they employ a stacked autoencoder approach for Nao Robot classification. They use a 4 hidden neuron autoencoder network for classification. The network is pretrained as autoencoder using dropout and L-BFGS for finding the most suitable parameters of the network. Additionally they use a sparsity penalty

on activation for the hidden units to regularize the network. In a latter stage supervised backpropagation is performed to train the autoencoders for detection of Nao Robots. More details can be found here Reinhardt [2011].

HTWK result stem from their score by testing their classifier on their test and training set which for comparison we did the same. It is clear that Viola and Jones under performces having a smaller Hitrate of 17,5% less than HTWKs algorithm. This can be explained by HTWKs way of pre-processing their images before classifying, which makes both method not fully comparable. In future work this could be solved by also first pre-process our images and only classify on unknow regions on the field (see fig.11b), which would make it less possible to miss classify a Nao Robot. Our implentation of Viola and Jones algorithm did achieve a higher precision than HTWKs classifier, by finding more False Positives in the images.

8.2 Important findings

Increasing the maximum depth of the tree of weak classifiers used in each layer of the cascade to build one strong classifier, did not as expected result in an higher hitrate. In fact it halved when compared to the same classifier with a depth of one. Which can be explained that algorithm of weak classifier overfitted on the training samples, decreasing its performance.

The HOG classifier clearly under performs from the Haar-like features from Viola and Jones. Because, HOG is build to detect shapes more than features who are fixed in possition (which Haar is better in). We expected for HOG to give more robust classifiers. But this can be explained by the fact that Nao Robots are moving during SPL games hence no fixed shape could be found.

The Viola and Jones classifier for upper body of the Nao Robot has an higher hitrate than the lower body classifier with the same amount of training samples. This was to be expected, where the lower body has more moving parts and less features in a fixed spot as say the neck and shoulders of the Nao Robot. Haar-like features prefer fixed positions for feauters, which explains these results. Its to be noted that the lower body had 40% less positive testing samples.

8.3 General remarks

Decreasing the minimal hitrate with with 0.04 procent lowerd the False Positives with almost 90

Its clear that more training samples will make for a better performing classifier. The increase of 300 training samples increased our classifiers hitrate

with almost 30

Scaling the sample images of the Nao Robots in a smaller window of 16 x 24 pixels compared to the 20 x 30 window has, as expected, a much lower Hit Rate. When training with a more pixel dense image the classifier is able to distinguish between more features. Making it possible to find more features in the integral images and possiblity to find better weak classifiers than within a smaller window. Scaling an image to a lower window with a lower pixel density, values are lost of the orginal image thanks to mergings to pixel averages.

Training on our OctaCore processor the training time of each classifier of 20 layers were far less than anticipated. In our research we found training times of 3 days for a similar problem. The longest training time was with a maximum depth of 2, all other classifiers were trained with a depth of one. thanks to the extra row of classifiers build in each decision like tree.

In the beginning of our research we ran a couple of experiments withour Lienhart et al. extended features. Compared to training with, we found and small increase of TP and a much larger decrease of FP which was anticipated by Lienhart and Maydt [2002], Flynn [2009]. We quickly decided that all other experiments would be run with Extended features, so consitincy would exist between each different classifier.

The XML files constructed after the training process consist of a large amount of features. The first layer of the cascade, thanks to the AdA Boost algorithm, have the least amount of features that can reject the most of images and the further an image goes in the layers the less chance there is that it will be rejected. Making the cascade a fast classifier to reject false images but positive images will have to go trough more layers when training has been done for more stages. Giving a smaller profit per layer while costing more time to classify a positive, depending on the classifiers task, these considerations should be made.

8.4 Implication for application

Dealing with uncertainties is an implication when working with detection algorithms. Finding a balance between achieving a high enough hit rate and still have high precisions should be decided for each individual task. For example when detection is used during life threatening situations highest possible hit rates are needed ie. a robot is searching for human victims in disaster areas. Which could be achieved by adding more layers to the cascade, but at the same time making the algorithm more computationally expensive. Which can be unwelcomed when building a world model during SPL games while detecting Nao Robots, where higher framerates are more prefered than

the highest hit rate. Approximating detection with uncertainty can be build into the world model and with a smart framework could work to actively reduce the uncertainty. Ie. a Nao Robot could look again if it really did see another Nao Robot to decrease its uncertainty or teammates knowledge could be incorporated into one global world model.

8.5 Implications for other algorithms

Viola and Jones' method has shown its potential for detecting Nao Robots, while this method is most widely used for its human face detection. It shows that algorithms used for human detection are possibly also good for Robot detection. Our results show that a face detection method can also be used for classifying limbs, when classifying for the lower body of the Nao Robot. An effect from this, could be potential for using Viola and Jones algorithm when detecting Human Limbs. The sort of detection needed when searching for victims in disaster areas.

9 Conclusion

The objective of this thesis was to test if the Viola and Jones algorithm, which creates a cascade of boosted weak classifiers on simple Haar-like features that acts as a strong classifier, could achieve competitive performance compared to the current state of the art in Nao Robot detection. Weak lassifiers were trained using 3000 images extracted from Nao Robots during a variety of SPL games. From these images 725 hand annotated positive samples were used for training and 200 positive samples for testing. Multiple cascades were trained according to Viola and Jones' method which were tested and evaluated. The strong classifier that performed best was compared to the current state of the art. Our classifier did not succeed in achieving a higher hit rate than the state of the art, although our method has potential, and with future work the gap between the two hit rates could be closed. Our implementation did succeed in achieving a higher precision which, based on the task at hand, is arguably more valuable than overall hit rate.

10 Future Work

With a better annotated and larger data-set we are certain our classifier will improve with detecting Nao Robots. When building this data-set distinguish should be made between side and front view of the Nao Robot. Making the classifier more precise and giving the user more info on the direction of the

detected Nao Robot.

Training with larger sample windows of annotated Nao Robots would make the classifier more accurate. Experiments should be made to find the optimal size, with the realisation that Nao Robots that appear small in the image because of a larger distance could be missed. Because during detection the subwindow to slide over the image is never smaller than the trained resolution of the positive training samples.

Our classifiers show potential for being used during SPL games. Therefor the classifier should be build into a working framework of the Nao Robot and tested for further improvements. While searching for the best ratio between fast detection and a high enough hit rate.

Acknowledgments

This Thesis is dedicated to all my friends and family: 'Sorry for taking so long and thanks for all the fish'.
Thanks to Arnoud Visser, for without your guidance and patients this thesis would not have been possible.
An additional thanks to Thomas Reinhardt for his work on the training Data.
And most of all I would like to dedicated this Thesis to my Schatje. For without her trust, patients and support I could not have done this.

References

Gabriel I Barbash and Sherry A Glied. New technology and health care costs—the case of robot-assisted surgery. *New England Journal of Medicine*, 363(8):701–704, 2010.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1467360`.

Martin Engel. Anwendung von maschinellem lernen zur echtzeitfähigen und kalibrierungsfreien erkennung von humanoiden fußballrobotern, 2013. URL `http://naoteam.imn.htwk-leipzig.de/documents/BA_Martin_Engel.pdf?lang=en`.

Alexander Fabisch, Tim Laue, and Thomas Röfer. Robot recognition and modeling in the robocup standard platform league. In *Proc. 5th Workshop on Humanoid Soccer Robots at Humanoids*, 2010.

Helen Flynn. *Machine learning applied to object recognition in robot search and rescue systems*. PhD thesis, University of Oxford, 2009.

Yoav Freund, Robert Schapire, and N Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 1, pages I–900. IEEE, 2002.

Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *Pattern Recognition*, pages 297–304. Springer, 2003.

David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

Heinrich Mellmann and Marcus Scheunemann. Local goal model for a humanoid soccer robot. In Andrzej Skowron Magdalena Kacprzak Marcin Szczuka, Ludwik Czaja, editor, *Proceedings of the Workshop on Concurrency, Specification, and Programming CS&P 2011*, pages 353–360, Pułtusk, Poland, September 2011. Białystok University of Technology.

Pablo Negri, Xavier Clady, and Lionel Prevost. Benchmarking haar and histograms of oriented gradients features applied to vehicle detection. In *ICINCO-RA (1)*, pages 359–364, 2007.

Sébastien Negrijn, Janosch Haber, Ysbrand Galama, and Arnoud Visser. Uva@ work, team description paper rockin2014-toulouse, france.

Lisa Nocks. *The robot: the life story of a technology*. Greenwood Publishing Group, 2007.

Constantine P Papageorgiou, Michael Oren, and Tomaso Poggio. A general framework for object detection. In *Computer vision, 1998. sixth international conference on*, pages 555–562. IEEE, 1998.

Thomas Reinhardt. Kalibrierungsfreie bildverarbeitungsalgorithmen zur echtzeitfähigen objekterkennung im roboterfußball. Master's thesis, Hochschule für Technik, Wirtschaft und Kultur Leipzig, 2011.

Michael J Swain and Dana H Ballard. Color indexing. *International journal of computer vision*, 7(1):11–32, 1991.

Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

Kazuyoshi Wada, Takanori Shibata, Tomoko Saito, and Kazuo Tanie. Effects of robot-assisted activity for elderly people and nurses at a day service center. *Proceedings of the IEEE*, 92(11):1780–1788, 2004.

Auke Wiggers and Arnoud Visser. Discovering reoccurring motifs to predict opponent behavior. In *Advanced Robotics (ICAR), 2013 16th International Conference on*, pages 1–7. IEEE, 2013.

Yuan Xu, Heinrich Mellmann, and Hans-Dieter Burkhard. An approach to close the gap between simulation and real robots. In *Simulation, Modeling, and Programming for Autonomous Robots*, pages 533–544. Springer, 2010a.

Yuan Xu, Heinrich Mellmann, and Hans-Dieter Burkhard. An approach to close the gap between simulation and real robots. In *Simulation, Modeling, and Programming for Autonomous Robots*, pages 533–544. Springer, 2010b.

Qiang Zhu, M-C Yeh, Kwang-Ting Cheng, and Shai Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1491–1498. IEEE, 2006.

11 Appendix

Algorithm Integral Image

Input: an image I of size $N \times M$.
Output: its integral image II of the same size.
Set $\mathrm{II}(1,1) = I(1,1)$.
for $i = 1$ to N **do**
  for $j = 1$ to M **do**
    $\mathrm{II}(i,j) = I(i,j) + \mathrm{II}(i,j-1) + \mathrm{II}(i-1,j) - \mathrm{II}(i-1,j-1)$ and II is defined to be zero whenever its argument (i,j) ventures out of I's domain.
  end for
end for

Figure 12: Pseudo code of creating an integral image

Algorithm Adaboost

Input: n training examples $(x_i, y_i) \in \mathbb{R}^d \times \{-1, 1\}$, $1 \le i \le n$, number of training rounds T.
Parameter: the initial probabilistic weights $w_i(1)$ for $1 \le i \le n$.
Output: a strong learner/committee.
for $t = 1$ to T **do**
  Run Algorithm 5 to train a decision stump h_t using the weights $w_\cdot(t)$ and get its weighted error ϵ_t

$$\epsilon_t = \sum_{i=1}^{n} w_i(t) 1_{h_t(x_i) \neq y_i},$$

  if $\epsilon_t = 0$ and $t = 1$ **then**
    training ends and return $h_1(\cdot)$.
  else
    set $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$.
    update the weights

$$\forall i, \quad w_i(t+1) = \frac{w_i(t)}{2}\Big(\frac{1}{\epsilon_t} 1_{h_t(x_i) \neq y_i} + \frac{1}{1-\epsilon_t} 1_{h_t(x_i) = y_i}\Big).$$

  end if
end for
Return the rule

$$f^T(\cdot) = \mathrm{sign}\,\Big[\sum_{t=1}^{T} \alpha_t h_t(\cdot)\Big].$$

Figure 13: Pseudo code of AdaBoost method

Algorithm Detecting faces with an Adaboost trained cascade classifier

Input: an $M \times N$ grayscale image I and an L-layer cascade of shifted classifiers trained using Algorithm 10
Parameter: a window scale multiplier $\mathfrak{c}$
Output: $\mathcal{P}$, the set of windows declared positive by the cascade
Set $\mathcal{P} = \{[i, i+e-1] \times [j, j+e-1] \subset I : e = [\![24\mathfrak{c}^{\kappa}]\!], \kappa \in \mathbb{N}\}$
for $l = 1$ to L **do**
  for every window in $\mathcal{P}$ **do**
    Remove the windowed image's mean and compute its standard deviation.
    if the standard deviation is bigger than 1 **then**
      divide the image by this standard deviation and compute its features required by the shifted classifier at layer l with Algorithm 3
      if the cascade's l-th layer predicts negative **then**
        discard this window from $\mathcal{P}$
      end if
    else
      discard this window from $\mathcal{P}$
    end if
  end for
end for
Return $\mathcal{P}$

Figure 14: Pseudo code an Viola and Jones dection algorithm