University of Amsterdam
Faculty of Science
The Netherlands

Dutch Nao Team

Technical Report

Authors:
Hidde Lekanne gezegd Deprez
Douwe van der Wal
Pieter Kronemeijer
Michiel van der Meer
Linda Petrini
Thomas Groot
Jier Nzuanzu
Caitlin Lagrand

Supervisor:
Arnoud Visser

December 31, 2018

Abstract

In this Technical Report, the Dutch Nao Team lists its progress and activities in the past year with the previous report [1] as a starting point. Besides new developments this report also lists older developments when relevant. Progress has been made in developing vision and motion modules. In vision both a new ball detection and a field detector have been implemented. For motion, processes have been streamlined making integration of preferable walking engines possible. Moreover debugging tools were development to be used for the framework in the form of a live interface. Additionally the team participated in several events namely the Robotic Hamburg Open Workshop (RoHOW), RoboCup Iran Open in Tehran and the 2018 RoboCup in Montreal.

Contents

1 Introduction

The Dutch Nao Team consists of thirteen Artificial Intelligence students; six Master students and seven Bachelor students, who are supported by a senior staff member, Arnoud Visser. The team was founded in 2010 and competes in the RoboCup Standard Platform League (SPL); a robot football league in which all teams compete with identical robots to play football fully autonomously. The league was started to incentivize the development in robot science. Its current goal is to play against the human world champion of 2050, and win. Since all teams participating in the Standard Platform League are obliged to use identical robots, the focus of the league is solely software oriented rather than hardware oriented. The robots need to be able to play autonomous. This includes finding the ball, locating itself and making decisions on how to play next, as well as communicating with teammates and being able to walk.

Figure 1: Team photo in Montreal

2 Hardware

2.1 The Nao robot

The Nao robot is a programmable humanoid robot made by Softbank Robotics, formerly known as Aldebaran Robotics. Up until 2018, all versions of the Nao were equipped with the same computational hardware, only differing in their sensors or actuators. With the release of the sixth version (V6), a significant change in both hardware and proprietary software caused the team to design two versions of our framework, each specific to the robot version. Below, the older robot versions (V4 and V5) are detailed, with Section 2.1.1 specifying the Nao V6.

The CPU used by the robot is a 32-bit 1.6 GHz Intel Atom single physical core which is hyper-threaded, combined with 1 GB of RAM and 8 GB of storage. Since there are only two logical cores, CPU resources are scarce, which limits calculation-heavy tasks such as pixel-wise image operations and (deep) neural network approaches. The Nao has two high-definition (HD) cameras and an inertial board that are both used for the competition. The HD cameras are located in the head of the Nao, one is aimed forward to look for far away objects and the other is aimed downwards for looking at objects close to the feet of the Nao. The inertial board is used for determining whether the robot has fallen, which happens regularly during matches. The Nao also possesses four sonars, an infrared emitter and receiver, pressure sensors and tactile sensors. Except for the pressure sensors, these sensors are used less frequently than the cameras and inertial board, as they are more prone to breaking down, resulting in faulty measurements. The pressure sensors however become one of the most important sensors, since the new walking engine heavily relies on the pressure information coming from the feet.

From the V4 and up the Nao robot has 25 degrees of freedom in its joints. The joints in the legs allow for stable bipedal movement and various kicking motions, the arms are generally used for helping the robot stand up again after falling and for stability while walking. It is also permitted for the goalie to hold the ball in its hands, but it is highly uncommon for teams to make use of this. Even though every robot is supposed to be the same, individual differences are noticeable when the robots is playing football. The movement of older robots is less stable and fluent, since the joints of these robots have been worn out. In order to ensure a robust walk for every robot, the joints for each individual robot need to be calibrated. Additionally, each robot's camera can shift inside its enclosure, resulting in a slightly offset transformation with respect to the robot centre. To correct for this, the cameras can be corrected by performing camera calibration.

2.1.1 V6

The new V6 is a 64-bit system and has a 1.91Ghz atom E3845 CPU which is a Quad core with one thread per core. Ram has increased to 4 Gb and storage has increased to 32Gb SSD. Furthermore the WiFi connection has improved and the fingertips are more resistant to impact. With the added computing power, new approaches to issues like localisation and ball detection will be possible. Unfortunately the V6 has not been used yet in the standard platform league, mainly due to the fact that there is no DCM available. The DCM is used communicate between our code and the hardware of the robot making direct control possible. Without DCM the robot is only controllable through Naoqi which . In addition, no modules have been developed with the computational capabilities of the V6 in mind.

3 Framework

Our framework was created last year. Its structure is based on [2] and is further elaborated in [1]. Moreover it uses Kahns algorithm [3] to link different modules in a proper sequential order.

3.1 Module Communication

This year the need arose for the ability to execute some modules at a different update frequency than others. Specifically the modules responsible for motion would benefit from a high constant frequency while other modules (vision for instance), could significantly slow the update frequency with computational expensive tasks. Additionally the computation needed for a task could be variable, depending on the situation the robot finds itself in.

The proposed solution was separating the motion modules and other modules on two separate threads. Moreover multithreading support would benefit execution times on the V6 which has four cores and four threads. The main issue to solve was information exchange between modules on separate threads. As different threads can have different update times, memory access needs to be safe and preferably managed without additional specifications inside each module.

Thus a new message system was proposed. Its new specifications dictate that any module can send an output messages which contains a pointer to a representation object. Every module in need of that message gets a copy in their message queue. Entries in these queues can only be read and/or deleted, not modified. This queue has a maximum size after which the oldest message gets deleted. The module accessing it's queue can do so by popping messages.

3.2 Interface

This year a graphical interface was introduced, to allow easier and faster debugging by providing direct access to sensor data (including camera data), visualisations of values changes and other higher abstractions, e.g. the ball position in the image provided by the ball detector.

The interface is based on the library NanoGUI[1]. Since the interface runs locally, communication with the robot is performed by assigning to the interface its own module. This module is in charge of receiving the required representations from the robot and sending them to the interface, so that the displayed values and images can be updated frequently enough.

The interface is organised on a widget-based system. Two main widgets allow one to choose what robot to connect to and what type of representation to open between Actuators, Behaviours, Infrastructure, Modelling, Motion and Sensors. Most widgets simply display numerical or boolean values, with the added possibility to plot the value changing in time. The camera widget differentiates between Lower and Upper camera, allowing one to see the visual input received by the robot in real time. Moreover, the camera widget allows to visualise the output of other modules such as the Ball Detector, the Field Boundary Detector, the Line Model and the Line Detector. The Behaviours widget shows the sorted list of Decisions and relative Considerations, as they are are ranked in our current implementation.
It is straightforward to see the added value of the interface, since the ability to visualise the camera input in real time, as opposed to saving a conspicuous amount of images and examining them one

[1]https://github.com/wjakob/nanogui

Figure 2: View of the interface with widget examples

by one. Also showing the sorted decisions as they are "taken", instead of having to read through the logs, make the debugging process for all other modules simpler and less time-consuming.

4 Motion

4.1 Walking Engine

This year the built-in walk of NaoQi that was previously used was replaced by implementing the B-Human version of the Walk2014Generator [4] developed by rUNSWift. The Walk2014Generator is not only a stable and fast walking engine, but the B-Human implementation is modular and well documented, making it the best candidate. The walk generator and coinciding inverse kinematics were implemented in the DNT framework in order to create a basic walking engine. Components like the dynamic CoM balancing were removed to ensure that the walk generator would be modular. Weight shift miss detection and Sagittal and coronal balance adjustments of the ankles however were left in.

DNT also added its own features to the walking engine, such as a kicking mode and exact step mode. The kicking mode activates the in-walk kick procedure which can be adjusted in power. First a balancing step is executed, then an optional step to ensure that the correct foot is the kicking foot and after that the kick is executed. The exact step mode allows the user to request a number of steps with a given distance in cm. This feature can be used for accurate positioning during (penalty) kicks.

While the current implementation works to a satisfying degree there is room for improvement, especially for fast parameter tuning. Integrating the walking engine with the interface would be a step in the right direction and possibly allow the development of a balancing module in the future.

4.2 Interpolation engine

Last year an attempt was made to make a general motion engine that uses Choregraphe [5] style keyframes, but it could not be used at the RoboCup in Nagoya due to stability issues. This year, development continued, resulting in a module capable of taking a keyframe (a sequence of time and angle pairs for every joint of the robot) and outputting joint angles in every new cycles, interpolating linearly between the given angles. This module will finish all motions it has started. Currently, this module has keyframes for front and back get-up, stand and kick motions.

For a long time, this module conflicted with the built-in Naoqi MotionProxy, which is the Naoqi module for handling all motion. This module was needed to make the robot walk, until a proper walking engine was put in the framework. In certain situations, both the DNT framework and the MotionProxy would send motion commands to the robot, resulting in the robot trembling a lot during motions. Despite our best efforts, this could not be solved completely. After removal of the Naoqi MotionProxy, all motions ran smoothly.

4.3 Forward Kinematics

This year the forward kinematics has been inspired by an old Nao robotics team Kourette, [6]. This approach was integrated in the existing framework as an whole representation of the robot joints with the torso as a reference point. The joints of the robot are represented in the Denhavit-hartenberg convention. Meaning that a transformation of a right arm, for example, from one moment in time in another moment is with reference to the torso as the base, see equation 1.

$$\mathbf{T}_{base}^{end} = \mathbf{T}_{base}^{0}\mathbf{T}_{0}^{j}\left(\theta_0 \cdots \theta_j\right)\mathbf{T}_{j}^{end} \tag{1}$$

In 2 is a formal way where a joint j is represented with respect to its previous joint $j-1$, where R is the rotation matrix and A is the translation matrix.

$$T_{j-1}^{j} = \mathbf{R}_x(\alpha_j)\mathbf{A}\left([a_j 00]^T\right)\mathbf{R}_z(\theta_j)\mathbf{A}\left([00d_j]^T\right) \tag{2}$$

The Denhavit-Hartenberg parameters in equation 2 are; a the link length of the joint to its adjacent, α the link angle of the z_i axes to the $z_i - 1$ axes, d the link offset along the $z_i - 1$ axis and θ the angle about the $z_i - 1$ from $x_i - 1$ to x. The rotation matrix used for the implementation of the forward kinematics is according to the *Yaw-Pitch-Roll* convention, where there is a rotation around the z, y then x axis if a joint contains all rotation,see equation 3.

$$\mathbf{R} = \mathbf{R}_z(a_z)\mathbf{R}_{\mathbf{y}}(a_y)\mathbf{R}_{\mathbf{x}}(a_x) \tag{3}$$

Every joint transformation, for example from right-arm(rArm) to left-foot(lFoot) is a chained transformation and can be written as such (4).

$$\mathbf{T}_{rArm}^{lFoot} = \left(\mathbf{T}_{torso}^{lFoot}\right)^{-1}\mathbf{T}_{torso}^{rArm} \tag{4}$$

The right-arm transformation should be inversed and chained with the left foot transformation. Because there is one reference point (the torso) and all joints are reachable from that point, a whole robot model is possible. This has the consequence of making transformation between any two joints possible. Hence accurate simulation of the robot joints is possible, the robot model.

Specific to our framework, any module might request a affine transformation matrix to a joint or from one joint to another.

5 Vision

5.1 Field Detection

As the Nao robot only has to concern itself with objects which are on the field, detection of that field may lead more accurate result as means of noise reduction. Furthermore computational resources could be spared by reducing the area to be analysed. Other teams [7] already perform field detection mainly using the colour of the field. One assumption often made is there is only one field and the surroundings of the field aren't green. This translates into a top down approach trying to identify the edge of field while assuming everything under that edge is part of the field.

In order to best define the colour green the colour space is transformed from rgb to hsv. The hue value can be used for defining the green colour while the saturation and value are used to cope with light conditions and camera deviations. To reduce the computational cost of the hsv transformation and later the field detection, we reduce the size of the image by a factor four (640x480 to 160x120) reducing the amount of pixels to process by a factor of 16. Other teams (like the Argentinian one in it' s latest research) use similar re-sizing.

Green has been defined as a convex space in the hsv domain with hue, saturation and value bounds. A pixel with all values within these bounds is considered green. A sequence of green pixel value (from top to bottom) is considered part of the field. As this approach can be noisy a means smoothing is done. Every pixel differing in y position to much from the mean will be reassigned to the closest conforming edge. Furthermore a ransac [8] is done to further filter outliers.

Figure 3: Field filter annotating the edge of the field with red dots

Unfortunately the results in are not entirely desirable as the ball can be considered "outside of the field" if it's too close to the edge, the same holds for robots. A solution has been proposed to assume always assume one or two lines can represent the field edge, forcing a smooth edge. This will make sure the base of a robot or ball is always on the field, provided it's detected correctly. This is however not yet implemented.

5.2 Ball Detection

Last year a new ball detector has been deployed which detected the ball by generating candidates and uses increasingly computationally expensive checks to decide whether a candidate contains a ball. The final check consisted of a haar-classifier [9], which did not only detect the ball very well but also had a significant amount of false positives. Due to the new walking engine, our walking speed has increased significantly making the impact of a false positive greater due to walking further in the wrong direction. To reduce the amount of false positives, two changes have been made to the ball detector. Firstly, the haar-classifier has been replaced by a convolutional neural network for candidates generated from the top camera. Secondly the field border detection is used to prevent checking candidates outside the field.

The convolutional neural network used can be seen in table 1. The data set consisted of region of interests collected on different fields in Canada. There were 2971 labelled positive samples and 10.000 negative samples, which were shuffled randomly divided over a test set of 1000 samples and a train set of 11971 samples. Balancing the amount of positive and negative samples did not have a positive effect on the performance of the classifier due to a lack of data. This caused more false positives, but augmentation of the positive samples could provide a solution to train an robuster classifier.

Performance on test set, CNN vs haar

Table 1: Different classifiers

	True Negative	False Negative	True Positive	False Positive
Cascade	**5574**	**18**	507	314
CNN	5529	63	**635**	**186**

In addition to a new classifier to prevent false positives, the amount of negative region of interests generated have been reduced significantly by the use of the field detector. Knowing where the field ends, rules out a lot of potential locations where the ball could be and prevents regions of interest placed on confusing objects such as cables or shoes around the field.

5.3 Goal detection

Before any localisation was implemented; detection of the goal and ball would be a alleviate our immediate problem of scoring in a match. The method proposed was to assume that the poles, field markings and robots are white while the background is not. This would make it possible for the robot to detect goal poles among the robots and field markings reliably as there are distinct features setting them apart.

Features include shape: two poles have the same height with a set distance between them and connectivity: goal poles can always be connected by a white top bar. Together with candidate selection from the edge of the field with the help of the field detector other features could be filtered out such that only the goal poles remained.

Unfortunately at the Robocup the background was white because of a white fence surround the field. Thus we needed a different approach detecting the slight edges between the different kinds of white. There was not enough time to implement this new approach making the module unusable during the Robocup.

6 Results

6.1 Iran Open (Tehran)

The Iran Open was the first competition the team competed in this year. It took place from the 3rd to the 7th of April 2018 in Tehran, Iran. The Dutch Nao Team used its own framework, which was mainly the code that was used in Japan, but then documented and cleaned. In Iran, we still used the NaoQi walking engine, which resulted in a lot of fallen and broken robots. Since the robots were not able to play football without walking properly, we decided that the main priority from now on should be changing to a better walking engine as explained in Sect. 4.1.

Besides attending the Iran Open competition, the team visited the Qazvin Islamic Azad University, the university of the Iranian team (MRL). The team was invited to have a look at how they manage their team and to have a workshop to share knowledge about their contributions to the RoboCup. Our team presented two topics: our new ball detector and our framework.

6.2 Robotic Hamburg Open Workshop (RoHOW)

RoHOW[2] is an annual open workshop for SPL teams, organised by the German team `HULKs`[3]. It took place from the 1st to the 3rd of December 2017. During the event, test games are played, algorithms and ideas about challenges are shared, and lots of coding is done. Furthermore, the more experienced members of most teams gave presentations or workshops about a broad range of subjects, ranging from raising funds to be able to attend international competitions to tips and tricks with compiling your code. Overall, the atmosphere is relaxed due to the lack of pressure of competitive games, which makes it a great opportunity for new team members to get to know the flow of working with robots and being in the SPL.

6.3 RoboCup Montreal

The Dutch Nao Team qualified for the RoboCup 2018 (18 to 22 June 2018) in Montreal with a video[4] and a qualification paper [10]. This was the first event with the new walking engine and the field detector.

During the first round, the Dutch Nao Team played $0-0$ against both `Naova` and `Aztlan`. During the first match, the field detector was not used yet, because of some calibration problems. This resulted in a lot of walking either out of the field, or walking into other robots, causing a ‘pushing‘ penalty. However, during the second match, the field detector was used and it improved the match a lot. Unfortunately, no goal was scored, since the robot still did not know where the goal was.

[2]https://www.rohow.de/2017/de/
[3]https://www.hulks.de/
[4]`https://www.youtube.com/watch?v=-6J7s3rnm1E`

Figure 4: Team photo at the Iran Open 2018.

During the second round, the Dutch Nao team lost $0-2$ versus `Naova` and `NTU RoboPAL` and played $0-0$ versus `Unbeatables`. Again, the robots were able to play football, but without any localization, no goal could be made from our side.

The Open Challenge consisted of a Penalty Kick Competition this year, which took place at the very beginning of the competition. During the challenge, teams have to score a penalty starting from one of the five predefined spots. The spot is randomly picked by the referee. In the Play-In Round, the Dutch Nao Team had to play against the `HULKs` and won with 1-0. The next opponent was `Nao-Team HTWK`, which shot the ball closer to the goal than our robot could during the sudden death penalties, after no goals during the first three penalties.

6.4 Foundation

In 2016, the Dutch Nao Team started a foundation, in order to be able to receive money from companies in a transparent way. In its first year, the foundation proved somewhat successful at its task of funding the trip to the RoboCup, lifting some of the financial burden of getting to Japan. At the beginning of 2017 year, the team chose to put a lot more effort into sponsoring than the previous year, hoping to completely fund the trip to the RoboCup in Montreal.

In 2017-2018, the board of the foundation consisted of the following members:

1. Chair: Caitlin Lagrand
2. Vice-chair: Sébastien Negrijn
3. Secretary: Pieter Kronemeijer
4. Treasurer: Douwe van der Wal

6.5 Public events

The past year, the team put a significant amount of effort into providing demonstrations, lectures and workshops. As the year before, all the funds raised by these events were used to fund the trip to the RoboCup.

In addition to the events scheduled by the team, the University of Amsterdam also invites us to their public events. In doing so, the team normally uses this opportunity to interact with young kids, upcoming students or alumni as a way of educating the public about robots and AI.

Events of this year were:

1. Open day Robotics lab, where students at Bachelor and/or Master level visited the robotics lab to get acquainted with robotics.
2. Open day UvA, a day where the DNT-team showed how the NAO robots play footbal for a large public, from parents to children,students to professors.
3. World Summit AI, where our team had a stand.
4. PA Consulting were we gave an presentation about our robots and demo at our location to a group of people.
5. ICAB, where we gave an presentation about organising and working in a student team to employees of ICAB.
6. Calland Lyceum, a high school event, where students were encouraged to learn how to use the NAO robots.
7. Wie-kent-School, high school students with learning difficulties were introduced into the field of Robotics and AI in general
8. Fintech day at triodos Bank where the team had a stand to give robot demonstrations and gave a lecture.
9. TEQnation where the team had a stand to give robot demonstrations.
10. Ouder-kinder-dag where we had a small stand and gave some demo's of the robots walking and kicking.

This year the team raised about 6068.74 euros, 2500 was received from the Amsterdams Universiteitsfonds and the rest was earned with demonstrations and presentations for companies. In total 10 paid events were attended.

Compared to last year we spend less time developing new demonstration modules as previous modules proved sufficient, though sometimes a few modifications were needed.

7 Contributions

List of people working on the additions mentioned in this report. Also significant modules which weren't in a finished state before the end of the year. Unfinished modules will be mentioned as such.

- **Caitlin**: worked on: interface, ball detection, forward kinematics, whistle detection, collision detection, behaviours
- **Douwe**: worked on: ball detection
- **Gautier**: helped with: line detection worked on: goal detection.
- **Hidde**: helped with: field detector; worked on: goal detection, line detection (unfinished)
- **Jier**: worked on: inverse kinematics
- **Linda**: worked on: interface
- **Lukas**: worked on: field detector, framework
- **Michiel**: worked on: inverse kinematics, framework maintenance, additional debug tools
- **Pieter**: worked on: interpolation
- **Sébastien**: worked on: framework, interface, team copy, localisation (unfinished)
- **Santosh**: worked on: whistle detection, collision detection
- **Thomas**: worked on: walking engine

8 Conclusion

The team is glad with the decision taken previous year to make a custom framework. The custom framework provides more flexibility in developing modules and debug tools compared to an imported one. Especially new member benefited from learning the framework with the help of people in our team who build it. The amount of modules implemented has increased since last year with this framework in place though results are still lacking. This is due the sheer amount of work needed to implement fundamental components such as localisation. Our team looks forward to next year when those fundamental components are scheduled to be implemented.

References

[1] Douwe van der Wal, Pieter Kronemeijer, and Caitlin Lagrand. *Dutch Nao Team - Technical Report*. Tech. rep. University of Amsterdam, Dec. 31, 2017. URL: `http://www.dutchnaoteam.nl/wp-content/uploads/2018/01/dnt_techreport_2017.pdf`. published (cit. on pp. 2, 4).

[2] Elizabeth Mamantov et al. "Robograms: A lightweight message passing architecture for robocup soccer". In: *Robot Soccer World Cup*. Springer. 2014, pp. 306–317 (cit. on p. 4).

[3] Arthur B Kahn. "Topological sorting of large networks". In: *Communications of the ACM* 5.11 (1962), pp. 558–562 (cit. on p. 4).

[4] Colin Graf and Thomas Röfer. "A center of mass observing 3D-LIPM gait for the RoboCup Standard Platform League humanoid". In: *Robot Soccer World Cup*. Springer. 2011, pp. 102–113 (cit. on p. 5).

[5] Emmanuel Pot et al. "Choregraphe: a graphical tool for humanoid robot programming". In: *Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on*. IEEE. 2009, pp. 46–51 (cit. on p. 6).

[6] Nikolaos Kofinas, Emmanouil Orfanoudakis, and Michail G Lagoudakis. "Complete Analytical Forward and Inverse Kinematics for the NAO Humanoid Robot". In: *Journal of Intelligent & Robotic Systems* 77.2 (2015), pp. 251–264. ISSN: 1573-0409. DOI: 10.1007/s10846-013-0015-4. URL: https://doi.org/10.1007/s10846-013-0015-4 (cit. on p. 6).

[7] Arne Hasselbring Yuria Konda Rene Kost Pascal Loth Lasse Peters Nicolas Riebesel Thomas Schattschneider Felix Warmuth Felix Wege Darshana Adikari Georg Felbinger. *HULKs Team Research Report 2017*. 2017. URL: https://hulks.de/_files/TRR_2017.pdf. published (cit. on p. 7).

[8] Martin A Fischler and Robert C Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6 (1981), pp. 381–395 (cit. on p. 7).

[9] 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*. Vol. 1. IEEE. 2001, pp. I–I (cit. on p. 8).

[10] Caitlin Lagrand et al. *Team Qualification Document for RoboCup 2018, Montreal, Canada*. Tech. rep. Science Park 904, Amsterdam, The Netherlands: University of Amsterdam, Jan. 31, 2018. URL: http://www.dutchnaoteam.nl/wp-content/uploads/2018/01/TeamQualificationDocument2C pdf. published (cit. on p. 9).