Team Description Paper Dutch Nao Team

Harold Ruiter, Gijs de Jong, and Macha Meijer

University of Amsterdam,
Amsterdam Science Park 904, 1098 XH Amsterdam, Netherlands
https://www.uva.nl/

Team Information

The Dutch Nao Team from the Netherlands comprises 24 active members and a staff member from the University of Amsterdam. The team consists of students pursuing degrees in both Bachelor's and Master's programs in Artificial Intelligence and Computer Science. Over the past thirteen years, the team has acquired a total of 27 NAO robots, although not all of them are currently operational. Currently, the team has nine NAO V6 robots. The team does not intend to purchase any new NAOs this year.

Team Name
Dutch Nao Team

Team Leader
Harold Ruiter

Team contact email address
info@dutchnaoteam.nl

Team website URL
https://www.dutchnaoteam.nl/

Country of origin
Netherlands

University/company affiliation(s)
University of Amsterdam
Intelligent Robotics Lab

1 Code Usage

Until 2022, the Dutch Nao Team has been using a framework written in C++. In 2023, we participated with an adapted version of the 2021 HULKs code release 3 while we were transitioning to our new framework yggdrasil. From 2024 onwards, we have been participating in the RoboCup events with yggdrasil, a framework fully built in the Rust programming language. All functionalities in this framework have been developed by the Dutch Nao Team, and the framework does not use any other team's code.

2 Own Contribution

In the past year, the Dutch Nao Team has made a lot of significant advancements which will be used at RoboCup 2025. These advancements include the development of a new framework and walking engine. Additionally, we have developed new techniques for ball, robot, and whistle detection. An elaborate overview of all the projects conducted in 2024 can be found in [1].

2.1 Framework

In 2023 we started creating our own framework from scratch, in order to deepen the understanding of the NAO within the team and improve on what we believe to be a shortcoming of the currently available frameworks. Concretely, we put a heavy emphasis on creating a fully parallelizable design that future proofs the team for future iterations of the NAO, as it is likely have more processing power and physical cores available. A current trend within the SPL is to have each system running at a different frequency assigned to a dedicated thread. With `yggdrasil`, systems are (unless specified) independent of threads and instead executed in parallel as tasks on the application's different thread pools. This gives us two major benefits:

1. Because systems are no longer tied to specific threads, we can simply schedule more tasks on threads that are not executing anything, allowing for more complete utilization of the full processing power of the NAO.
2. When a new model of the NAO with more available cores/threads releases, we can simply scale up the size of the thread pool to fully utilize the extra processing power.

2.2 Walking engine

This year we focused on building out the required infrastructure for motion reinforcement learning (RL). While high-level behavioral policies can be trained in simplified environments where the robot's dynamics are less important, low-level policies such as motion control require a significantly more complex environment. For such tasks, the accuracy of the robot's dynamics in simulation is important.

Even small differences in the robot's dynamics, often referred to as "the reality gap" ([2], [3] and [4]), can cause a significant drop in performance when the policy is used on the physical robot.

To address this, we focused on reducing this gap by using the MuJoCo [5] physics engine to model the NAO robot. MuJoCo is a general purpose physics engine that provides us with all the tools we need to accurately simulate the NAO robot, such as a gyroscope, accelerometer and pressure sensors.

Due to the limited computational resources available, we plan to enhance existing systems using these learned policies. The first component we want to enhance is the walking engine, specifically the gait generator. Previous work [6] has shown that enhancing an existing gait generator using a policy learned in simulation leads to a more generic and stable gait. [7] showed that this technique is generalizable to different robots, and we are currently working on bringing this to the NAO.

2.3 Ball detection

In order to quickly find a ball within an image, we split up the detection process in two steps: a proposal and classification step. In the proposal step, patches that potentially contain a ball are identified, and in the classification step for each patch it is determined if it actually contains a ball. Both steps have been subject to scrutiny and improvement over the course of the past year.

Proposals In the first step, ball proposals are generated. Because of the limitations that the NAO hardware poses, the major point of focus when developing the model was time efficiency. With that in mind, we focused on: (a) designing an efficient system with low runtime, (b) minimizing the number of proposals to reduce the runtime of the classifier step, all the while not losing out on accuracy. More details on this can be found in [1].

To reduce runtime, proposals are induced not from the camera image, but from the generated scan lines, which results in a significantly smaller search space. To find potential ball locations from scan lines, we make several assumptions. We search through the scan lines, and flag each area where the assumptions hold as a potential ball location.

Upon visually testing the model, we notice that the proposals are generally sensible and expected given our assumptions, while still employing a time-wise efficient model. Furthermore, we note that the number of proposals is kept low, also achieving the goal of not overloading the ball classifier.

Classification model To classify the ball proposals, a ball classification model was developed. During development, the goal was to make the model as light as possible, so that we are able to classify as much proposals as possible in the cycle time available, while maintaining almost perfect performance. The ball classifier that was developed uses 939 parameters and has an inference time of around $350\mu s$. The model has an accuracy, recall, precision and F1-score of above 99%.

The model architecture makes heavy use of the Depthwise Separable Convolution [8] for inference speed. Additionally it uses a Squeeze-Excite (SE) block [9] between the two spatial reduction layers in order to learn the dependencies between channels. This allows the model to focus more on the relevant channels, and achieves a computationally efficient channel attention mechanism. This block is important for the model's ability to generalize, achieving $0.3 \pm 0.2\%$ higher F1-score compared to models without the SE

2.4 Robot detection

Since robots have black and white components, similar to the ball, it can be hard to distinguish ball patches from robot patches. It was observed that in the ball proposal phase, there were a lot of proposals on other robots. Because of this, many proposals had to be processed, resulting in a longer cycle time. Therefore, a robot detection model was developed, with as the main goal to filter out all proposals that were on a detected robot. Additionally, the output of the model can be used for object avoidance and path planning.

The detection model developed is a single-shot detector (SSD) with a custom backbone. This means that first, feature extraction is done using the backbone. After this, bounding boxes are fitted on the features by the SSD, the boxes are refined, and subsequently classified. Since the images from the robot are in YUV color space, and converting them takes up valuable time, the model was specifically designed to be able to work with YUV images.

The inference time of the model is $19ms \pm 8$, which is fast enough to run on the robot in real-time. The model uses 49,412 parameters. The mAP of the model on the test set is 31.1%, which was good enough in practice.

2.5 Whistle detection

Another key area that has seen significant improvement during the past year was automated whistle detection. The development consisted of two steps. The first step, converting a raw waveform to a spectrogram, is typically done using the Fourier transform (FT) [10]. However, a spectrogram does not account for frequencies changing over time, even though our goal is to classify at which moments in time a whistle sound is present and at which ones not for an audio sequence. Hence, we need an adapted version of the FT: the short time Fourier transform (STFT) [11], which returns an array of spectrograms by repeatedly computing the FT for a sliding window over time. Because at the time of development no suitable Rust implementation of the STFT was available, we implemented it ourselves using the FT provided by the RustFFT[1] crate.

For the second step, developing the whistle detection method, we compared two different methods. We used an *algorithmic* approach described by a member of the HULKs team[2] as a baseline. We compared this with a *machine learning*

[1] https://github.com/ejmahler/RustFFT
[2] https://github.com/ykonda/Masterthesis

approach leveraging a feedforward neural network (FNN) that we developed. Analog to the previous method, this method only considers the high 2kHz to 4kHz bandwidth. However, instead of using a handcrafted detection mechanism, we train a FNN to classify whistles. The previous iteration of our whistle detection model trained on the entire frequency spectrum, but we found it was overfitting on lower frequencies to such a degree it became nearly unusable. For training, we used the dataset provided by [12]. Another key factor we changed with respect to our previous iteration is that we use our Rust implementation of the SFTF in both the framework and to preprocess the training data. Previously, we used the Pytorch implementation in the training code and modeled our Rust implementation after the Pytorch version. However, the implementations were not precisely equal, which adversely affected training.

After evaluating the models on the test dataset, we find that the FNN approach achieves a near perfect accuracy of 99% and substantially outperforms the algorithmic baseline of 93%. Lastly, we performed a practical test by simulating an exceedingly noisy match environment and sporadically blowing a whistle during the course of roughly 15 minutes. We found that the model was able to correctly identify all whistles.

2.6 Field boundary detection

Some vision modules should only consider parts of the image containing the soccer field, since detecting lines or balls outside of the field would waste computational resources.

To address this, we implemented a neural network-based field segmentation model, based on the work of [13]. Our approach follows the original method by predicting the y-position of the field boundary at 40 evenly spaced points across the image. By fitting a line through these predicted points, we create a boundary where all pixels below the line are classified as part of the field.

The architecture described in [13] follows a two-stage design based on [14]. The backbone consists of 3-4 inception V3 blocks, which handle feature extraction. These features then pass through a single-layer convolutional bottleneck that outputs the predicted boundary point positions.

Our architecture is very similar, but we switch out the convolutional layers with depth wise separable convolutions and use the Sigmoid-weighted Linear Unit (SiLU) [15] as non-linearity instead of the traditional ReLU. The original EfficientNet architecture shows that this activation results in better performance [16] as it does not immediately discard negative inputs which makes training more stable.

3 Unpublished Results

From 2020 onwards, the Dutch Nao Team participated in all main RoboCup competitions. The results of these competitions can be found on https://spl.

robocup.org/history/. Furthermore, the team participated in the GORE 2022 and 2023, and the RoboCup German Open 2024, of which the results can be found on https://spl.robocup.org/events/. The team also participated in several editions of the RoHOW. Prior to the RoboCup 2025, the Dutch Nao Team intends to participate in the German Open 2025.

4 Impact

The active participation of the Dutch Nao Team in RoboCup over the past years has had significant impact on the team itself, the University of Amsterdam, and the SPL. Over the past year, the team has gained members, now having 25 active members from different studies related to computer science and AI. Because of this, the team has gained professionality in its organization. Furthermore, over the years, the Dutch Nao Team has supported several theses and projects that led to publications [17]–[23] and projects that led to publications on a large variety of topics [24]–[26]. Additionally, the Dutch Nao Team is part of the Lab42 community[3], which is the centre for Digital Innovation and AI of the University of Amsterdam.

In 2024, the team has worked on several projects (see Section 2), and has the intent to publish these results on RoboCup events in 2025. Furthermore, the Dutch Nao Team is the only team in the Challenge Shield of the SPL that developed its own framework, something we believe provides a significant input to the SPL and encourages new development and research.

5 Other

Aside from the events at the University of Amsterdam that the Dutch Nao Team is a part of, DNT organizes and is part of a lot of events for the entire community. The team has a collaboration with Startup Village Amsterdam[4]. In practice, this means that delegations from all over the world come and visit the Intelligent Robotics Lab and the team gives presentations and demonstrations on the work we do as a team. Furthermore, we give programming workshops for high-school students, both in our robotics lab and at events like the Career Day[5]. Lastly, we participate in events that are free for the entire community, such as 24UurOost[6], an event organized by the city of Amsterdam, where we give presentations about the work that we do for the people of Amsterdam.

References

[1] H. Ruiter, G. de Jong, M. Meijer, *et al.*, "Dutch nao team - technical report," Tech. Rep., Dec. 30, 2024.

[3] https://lab42.uva.nl/
[4] https://www.startupvillage.nl/
[5] https://www.techportal.nl/career-day
[6] https://www.iamsterdam.com/uit/24uur-amsterdam/oost

[2] N. Jakobi, P. Husbands, and I. Harvey, "Noise and the reality gap: The use of simulation in evolutionary robotics," in *Advances in Artificial Life: Third European Conference on Artificial Life Granada, Spain, June 4–6, 1995 Proceedings 3*, Springer, 1995, pp. 704–720.

[3] K. Bousmalis, A. Irpan, P. Wohlhart, *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2018, pp. 4243–4250.

[4] F. K. Gunnewiek, *Quantifying the reality gap in abstracted pedestrian detection in simulated environments*, Bachelor's thesis, Jun. 2023. [Online]. Available: https://staff.fnwi.uva.nl/a.visser/education/bachelorINF/Bachelor_Thesis_Fyor_Klein_Gunnewiek.pdf.

[5] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2012, pp. 5026–5033. DOI: 10.1109/IROS.2012.6386109.

[6] M. Rahme, I. Abraham, M. L. Elwin, and T. D. Murphey, "Dynamics and domain randomized gait modulation with bezier curves for sim-to-real legged locomotion," *arXiv preprint arXiv:2010.12070*, 2020.

[7] G. de Jong, L. Eshuijs, and A. Visser, "Learning to walk with a soft actor-critic approach," in *Proceedings of the 35th Benelux Conference on Artificial Intelligence (BNAIC 2023)*, 2023.

[8] L. Sifre and S. Mallat, "Rigid-motion scattering for texture classification," *arXiv preprint arXiv:1403.1687*, 2014.

[9] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[10] R. N. Bracewell, "The fourier transform," *Scientific American*, vol. 260, no. 6, pp. 86–95, 1989.

[11] M. Ashouri, F. F. Silva, and C. L. Bak, "Application of short-time fourier transform for harmonic-based protection of meshed vsc-mtdc grids," *The Journal of Engineering*, vol. 2019, no. 16, pp. 1439–1443, 2019.

[12] N. W. Backer, B. O. K. Intelligentie, and A. Visser, "Horn and whistle recognition techniques for nao robots," *Bachelor thesis, Universiteit van Amsterdam*, 2014.

[13] A. Hasselbring and A. Baude, "Soccer field boundary detection using convolutional neural networks," in *Robot World Cup*, Springer, 2021, pp. 202–213.

[14] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[15] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural networks*, vol. 107, pp. 3–11, 2018.

[16] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, PMLR, 2019, pp. 6105–6114.

[17] C. G. Lagrand, *Learning a robot to score a penalty – minimal reward reinforcement learning*, Bachelor's thesis, Universiteit van Amsterdam, Jun. 2017.

[18] S. Negrijn, "Exploiting symmetries to relocalise in robocup soccer," M.S. thesis, Universiteit van Amsterdam, Dec. 2017.

[19] R. van Heusden, *Making a robot stop a penalty – using q learning and transfer learning*, Bachelor's thesis, Universiteit van Amsterdam, Jun. 2018.

[20] T. Garritsen, *Using the extended information filter for localization of humanoid robots on a soccer field*, Bachelor's thesis, Universiteit van Amsterdam, Jun. 2018.

[21] H. L. gezegd Deprez, *Enhancing simulation images with gans*, Bachelor thesis, Universiteit van Amsterdam, Jul. 3, 2020. [Online]. Available: https://staff.fnwi.uva.nl/a.visser/education/bachelorAI/thesis_hidde_lekanne_deprez.pdf, published.

[22] X. Monté, *Neural factorization of shape and reflectance of a football under an unknown illumination*, Bachelor thesis, Universiteit van Amsterdam, Feb. 1, 2023. [Online]. Available: https://staff.fnwi.uva.nl/a.visser/education/bachelorAI/Monte_NERF_of_a_football_under_unknown_illumination.pdf, published.

[23] A. M. Bernardy, "Introducing object detection to ekf slam on a nao-robot football field,"

[24] C. Lagrand and M. van der Meer, "The roasted tomato challenge," *Amsterdam Science*, vol. 05, p. 4, Apr. 2017.

[25] R. van Heusden, "Making a robot stop a penalty," in *Proceedings of the 30th Belgian-Netherlands Conference on Artificial Intelligence (BNAIC)*, 's Hertogenbosch, The Netherlands, Nov. 2018, pp. 89–90.

[26] G. de Jong, L. Eshuijs, and A. Visser, "Learning to walk with a soft actor-critic approach," Nov. 2023.