Team Description Paper Team HULKs*

Ole Felber, Pascal Gleske, Jonathan Hellwig, Jonas Holz, Konrad Valentin Nölle, Anna Relógio Stauber, Maximilian Schmidt, Julian Schuler, Vivienne Schwabe, Hendrik Sieck, and Patrick Göttsch[0000−0002−6642−8195]

Hamburg University of Technology, Hamburg, Germany
hulks@tuhh.de, http://www.hulks.de

1 Team Information

Team Name	HULKs
Team Leader	Patrick Göttsch
Team Contact Email	hulks@tuhh.de
Team Website	https://www.hulks.de and https://github.com/hulks/
Country of Origin	Germany
University Affiliation	Hamburg University of Technology

2 Code Usage

No code of other teams is used in the HULKs codebase. For inference of our neural networks, we depend on the CompiledNN library of B-Human. Our walking engine is inspired by rUNSWift concepts. Some of our standup motions are based on files from the NAO Devils.

3 Own Contributions

3.1 DiTEF - Distributed Task Execution Framework with Genetic Algorithm for Neural Network Architecture Search

The increasing application of artificial neural networks (ANN) in various domains of robotics with limited processing capabilities demands highly optimized ANN architectures. Efficient architecture search requires horizontal and vertical scaling of ANN evaluation. The HULKs present their approach and application of a scalable genetic algorithm based on distributed task execution which can be found at https://github.com/HULKs/ditef. The framework is able to scale dynamically and operate on heterogeneous hardware. Usage of the distributed genetic algorithm to evolve ANNs for the context of RoboCup soccer competitions is shown as a case study.

* Supported by TU Hamburg

3.2 Rust framework

We develop a general purpose robotics framework based on the Rust programming language. A modularized approach enables a parallelized execution in multiple threads. By utilizing code generation, we achieve a highly non-redundant and maintainable code base. An encapsulated framework makes it easy to focus on robotics code. Robotics domain and framework code are strictly separated. Rust's safety guarantees enable blazingly fast development. Runtime introspection of the type hierarchy is possible via a custom communication protocol. A hardware abstraction layer enables easy adaptation to the NAO and Webots simulator.

3.3 Ball Detection Network

We develop a multi-stage ball detection pipeline. It consists of a pre-classifier, classifier and positioner neural networks. The pre-classifier is a network prioritizing recall and execution speed to quickly eliminate unsuitable candidates. The classifier is optimized for precision to avoid false positives. The positioner accurately determines the position and radius of the ball in the sampled candidate image. All networks are optimized using DiTEF and are executed at runtime using CompiledNN.

3.4 Yocto Based Operating System for NAO

We provide a fully functional Yocto package targeting the NAO V6 platform. Yocto provides the necessary tools to build a highly customizable Linux distribution for embedded devices. All required recipes are open source and published at https://github.com/HULKs/meta-nao and https://github.com/HULKs/meta-hulks. We use Yocto to build a ready-to-flash OPN image and a toolchain for cross-compilation.

3.5 OpenAI Gym Environment

We investigate the use of a custom OpenAI gym environment for exploring the use of reinforcement learning in motion. The environment interacts with our framework making it possible to target the NAO robot or the Webots simulator.

3.6 Improved Walking Engine

Based on the ideas of the rUNSWift walking paper we implement a walking engine in Rust. We extend our control implementations for more stabilized step planning.

3.7 In-walk Kicks

Our walking engine supports the execution of dynamically planned in-walk kicks. It provides a forward-kick, turn-kick, and side-kick. During the execution of kicks, our walking engine ensures the stability of the robot.

3.8 Debug and Robot Management Tooling

We develop a new debug tool to dynamically inspect the runtime state of our control software. It also supports live camera feeds as well as plotting capabilities and changing parameters at runtime.

3.9 Behavior Simulator

To test and develop our robot behavior we provide a simulator to rapidly execute the behavior part of our robotics software.

3.10 Localization using UKF with line optimization

Our localization uses a multi hypothesis unscented Kalman filter which fuses odometry and line percept data. The line percepts are fitted to the field lines using an iterative closest line algorithm.

3.11 Automatic Calibration

The cameras are calibrated in an autonomous procedure. The extrinsic camera parameters are determined via a Levenberg-Marquardt optimization. To accurately find lines in the camera images the calibration procedure makes use of a sophisticated detection algorithm.

3.12 CompiledNN Bindings

To integrate the JIT-compiler for neural network inference CompiledNN https://github.com/bhuman/CompiledNN into our framework and tooling we developed open source bindings in both Rust and Python.

3.13 Webots Integration

Our open source release includes the models of the NAO V6 robot for use in the Webots simulator. To integrate the simulator with our framework we develop a Rust library containing bindings to the simulator.

4 Past History

Past Data

Competition	Tournament State	Opponent	Score*
RoboCup 2022	Quarter Final	Nao Devils	2:3
RoboCup 2022	Round 5	Nao Devils	2:1
RoboCup 2022	Round 4	B-Human	0:6
RoboCup 2022	Round 2	NomadZ	9:0
RoboCup 2022	Round 1	SABANA Herons	4:0
GORE 2022	Quarter Final	B-Human	0:10
GORE 2022	Round 6	Bembelbots	0:6
GORE 2022	Round 5	RoboEireann	1:9
GORE 2022	Round 4	SPQR	0:3
GORE 2022	Round 3	R-ZWEI KICKERS	0:4
GORE 2022	Round 2	Nao Devils	0:10
GORE 2022	Round 1	rUNSWift	0:6
GORE 2021	Round 3	Bembelbots	1:0
GORE 2021	Round 2	B-Human	0:10
GORE 2021	Round 1	Berlin United	4:1
RoboCup 2019	Quarter Final	Nao Devils	1:2
RoboCup 2019	CC Play In	NomadZ	9:0
RoboCup 2019	Second Round Robin	Bembelbots	3:1
RoboCup 2019	Second Round Robin	rUNSWift	0:5
RoboCup 2019	First Round Robin	B-Human	0:5
RoboCup 2019	First Round Robin	UPennalizers	10:0
GermanOpen 2019	3rd Place	rUNSWift	2:1
GermanOpen 2019	Semi Final	Nao-Team HTWK	1:3
GermanOpen 2019	Play-In Round	Nao Devils	5:1
GermanOpen 2019	Round Robin B	Nao-Team HTWK	0:4
GermanOpen 2019	Round Robin B	rUNSWift	1:2
GermanOpen 2019	Round Robin B	NomadZ	8:0

* from HULKs perspective.

We also participate in RoboCup 2021 in two out of four challenges. In the Obstacle Avoidance challenge the HULKs became second and on the 1vs1 challenge the HULKs finished on 9th to 12th place. Overall we reached rank nine out of fourteen.

Future participations

We are going to participate in GORE2023, April 2023, and RoboCup 2023, July 2023.

5 Impact

The HULKs have always been active in organizing events for our league as well as the humanoid league. These include seven RoHOW events as well as two German-Open-Replacement events (GORE). We are looking forward to our next GORE in April 2023 to which you are cordially invited.

As the first team developing our software fully open source we pioneer the free sharing of code and data to promote collaborative research in the SPL.

In participating in organizational and technical committees we shape the future of the SPL and research in humanoid robotics.

Our tooling to create self-packaged OPN files enabled teams to remotely deploy their robots and overcome the restrictions caused by the corona virus outbreak.