

NimbRo AdultSize Team Description 2017

Grzegorz Ficht, Hafez Farazi, and Sven Behnke

Rheinische Friedrich-Wilhelms-Universität Bonn
Computer Science Institute VI: Autonomous Intelligent Systems
Friedrich-Ebert-Allee 144, 53113 Bonn, Germany

{ ficht | farazi | behnke } @ ais.uni-bonn.de
<http://www.nimbro.net>

Abstract. This paper describes the RoboCup Humanoid League team NimbRo AdultSize of Rheinische Friedrich-Wilhelms-Universität Bonn, Germany, as required by the RoboCup qualification procedure for the competition held from the 27th of July to the 30th of July 2017 in Nagoya, Japan. Our team uses self-constructed robots for playing soccer. This paper describes the mechanical and electrical design of the robots, covers the software used for computer vision, state estimation, and motion generation, and highlights some of our scientific achievements.

1 Introduction

Team NimbRo has been very active in the RoboCup competition in the past years by participating in the TeenSize league, which our robots won six times. Our self-constructed robots have also been successful in winning the Technical Challenges in 2012 and 2014. With the current rise in popularity of the AdultSize class, we want to transfer the knowledge and experience our team has gathered throughout all the years and create a new, bigger platform capable of performing more dynamic motions. Since the 2016 competition, we have made advancements to our open-source ROS framework, which is used as the base

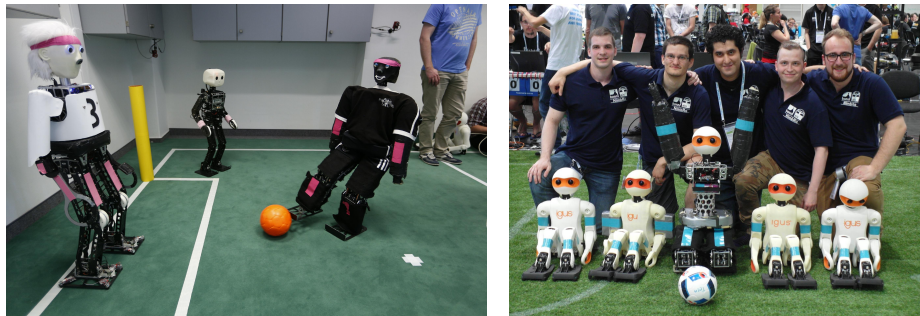


Fig. 1. Left: Previously built TeenSize robots. Right: Team NimbRo TeenSize during RoboCup 2016 in Leipzig, Germany.

for our AdultSize platform. The framework includes methods for attitude estimation, vision processing, localisation, soccer behaviors, servo communications, and sensor management. Our software shares its codebase with our TeenSize platform and does not contain software from any other teams.

2 Mechanical and Electrical Design

The most left part of Fig. 1 shows our tallest humanoid robot—Copedo. His mechanical design allows him to fall to the ground without being damaged. Copedo has the capability of performing get-up and throw-in motions due to having flexible arms. The effective use of parallel kinematics and force-sensing feet makes the robot’s walk very stable.

2.1 AdultSize Copedo

Copedo since his last RoboCup performance in 2015 in the TeenSize league has been extended to be 131 cm tall and 10.1 kg of weight. The update of Copedo is based on another one of our robots—Dynaped, as it was upgraded in 2016. Our team is in the process of fitting Copedo with new electronics to use our open source ROS-based framework just as Dynaped. After performing this upgrade, we hope Copedo to be a very effective player. Copedo is constructed from milled carbon fiber parts that are assembled to rectangular shaped legs and flat arms. The torso is constructed entirely from aluminum and consists of a cylindric tube that contains the hip-spine spring and a rectangular cage that holds the information processing devices. For protection, a layer of foam was included between the outer shell and the skeleton.



Fig. 2. Robot Copedo before the upgrade.

Most importantly, Copedo is equipped with 3-DoF arms that include elbow joints to enable the robot to stand up from the ground, to pick up the ball from the floor, and to perform the throw-in motion. Including a new igus® Humanoid Open Platform head with pan and tilt motions, Copedo will have 18 actuated DoF. The hip roll, hip pitch, and knee DoF are driven by master-slave pairs of Dynamixel EX-106+ actuators. All other DoFs are driven by single actuators including EX-106+ actuators for ankle roll, EX-106+ actuators for hip yaw and shoulder pitch, RX-64 actuators for shoulder roll and elbow, as well as the neck yaw and pitch. The robot has been fitted with cleats in the corners of its feet, to

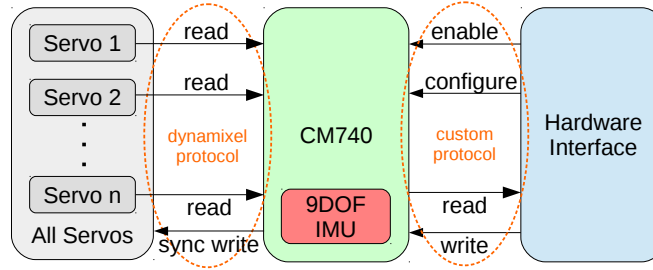


Fig. 3. Custom communication scheme for older Dynamixel actuators.

assist walking in artificial grass. The cleats have been mounted on beams with strain gauges to enable force sensing which is helpful for dynamic walking.

As of now, Copedo is being fit with a dual-core Intel Core i7-5500U CPU, which has four logical cores and a base frequency of 2.4 GHz with Turbo Boost up to 3.0 GHz. The PC is fitted with 4 GB of RAM and a 128 GB ADATA SX300 solid state disk. Available communication interfaces include USB 3.0, HDMI, Mini DisplayPort and Gigabit Ethernet. The PC is connected to the Robotis board, which communicates with all actuators on a RS485 star topology bus. The CM740 incorporates a 3-axis accelerometer and gyroscope, a 3-axis magnetometer can be added for a total of 9 axes of inertial sensory data.

3 Control

3.1 Estimation

State estimation is a necessity when it comes to controlling as complex systems as Humanoid Robots since many types of feedback rely on it to achieve stability during various motions like walking and kicking. Our state estimation includes an automatic calibration procedure for the gyroscope measurements with online recalibration of the gyroscope bias, which minimizes the drift down to $0.02^\circ/\text{s}$. With this technique, we achieve reliable sensor measurements which are then fused into a quaternion estimate of the orientation of the robot using the attitude estimator described in [2] of which a C++ implementation is available freely online [1]. The estimate is then filtered with a nonlinear passive complementary filter and represented using fused angles [3].

Based on the estimated attitude of the robot combined with joint angle feedback and a kinematic model, we achieve full body pose estimation. Based on that information, we compute the robot's center of mass position and velocity. With hysteresis, the support foot is taken to be the one with the lower vertical coordinate in respect to the rotated kinematic model.

3.2 Communications

Copedo was built in 2011 when the currently popular MX-64 and MX-106 actuators were not released yet, so our robot still uses the older RX-64 and EX-106+

models. In physical terms, the actuators are very similar with just minor differences in the range of movements possible. The biggest difference comes from the lack of a bulk read instruction in the older models, which is essential for allowing fast communications with multiple actuators with a single instruction. This requires each actuator to be read separately which in turn greatly slows down the control loop by increasing the latencies with each additional device on the RS485 bus. This is a critical point for dynamic and precise control, where fast reaction to changes in the estimated state is necessary. We have written a custom firmware for the CM740 to mitigate this limitation. In this scheme, the CM740 no longer acts as a passthrough from the PC to the servos, but communicates with both sides in parallel (see Fig. 3). On the actuator side, the CM740 loops through all registered devices on the Dynamixel bus querying them for status data. Communications with the PC are done with an extension of the original dynamixel protocol, which allows the use of the same, well-developed error handling as our original firmware. This approach allows us to still use the board as a passthrough device when needed. The CM740-PC protocol has been extended by four new instructions:

- Configure extended packet communication,
- Enable extended packet communication,
- Send extended packet, and
- Receive extended packet.

Starting this custom communications requires the hardware interface to first send a configuration packet with a list of servo ID numbers paired with their model types. This informs the CM740 which servo registers it needs to keep reading from and writing to. Typically, these registers correspond to position, torque, and controller gain data. The extended read packets contain the most recent data from all of the designated devices, with return values that increase whenever a device has been successfully read. This indicates if communications with the device are still happening. The write packets consist of the current position setpoints and compliance values for the servos. This approach allows all devices on the Dynamixel bus to be read at least once before a new read packet is sent with a frequency of 100 Hz. This transfer rate with the used hardware would not be achievable with the traditional request-response transmission paradigm.

3.3 Balanced Walking with Capture Steps

Robust walking in the setting of RoboCup is a challenge due to the unpredictable nature of the artificial grass surface, and due to the frequent and often large disturbances that the robot experiences from collisions with the ball, goal posts, and other robots.

Faster walking speed and higher stability is often the deciding factor when it comes to playing soccer. Playing better with and without the ball is largely the effect of a well-working gait. Copedo utilizes the Capture Step Framework, which also enables him to walk omnidirectionally. The robot not only manages

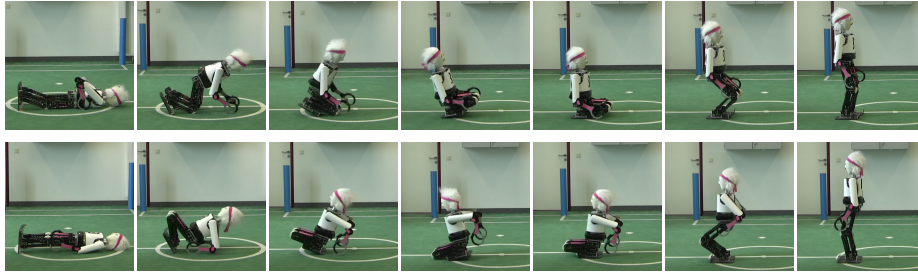


Fig. 4. Top row: Get-up motion from the prone posture. Bottom row: Get-up motion from the supine posture. In both motion sequences, the robot passively rocks back and forth on the foot edges from frames 3 to 5.

to locomote on an even surface, but can also cope with various disturbances, such as pushes, collisions, and stepping on the feet of an opponent as presented by Missura and Behnke in [8]. The actuation during walking is compliant and the robot walks with stretched knees.

3.4 Dynamic Robot Motions

Humanoid robots are designed with mainly walking in mind, but can also benefit from having predefined, proven to work motions. Playing these motions is the role of the motion player. In case of falling to the ground, the robot can quickly get up by performing a set of pose keyframes that are then automatically interpolated to become a smooth complete motion [4]. These motions can be made quite dynamic depending on the user's preference and even extended by the use of feedback mechanisms. To design the motions, a trajectory editor has been created. A few features of it that are noteworthy include: feedback mechanism configuration, left/right motion mirroring and operating in different spaces (e.g. joint, inverse, abstract). An example of the get-up motions are pictured in Fig. 4.

4 Computer Vision

The main source of perceptual information for humanoid robots on the soccer field is the camera, which is fitted with a wide-angle lens with an infrared cut-off filter to maximise the number of usable pixels and minimise the level of distortion, without significantly sacrificing the effective field of view. Our vision system is able to detect the field boundary, line segments, goal posts and other robots using texture, shape, brightness and colour information. After identifying each object of interest, by using appropriate intrinsic and extrinsic camera parameters, we project each object into world coordinates. The intrinsic camera parameters are pre-calibrated, but the extrinsic parameters are calculated online by consideration of the known kinematics and estimated orientation of the robot. Although we have the kinematic model of robot, some variations still occur on the real hardware, resulting in projection errors, especially for distant

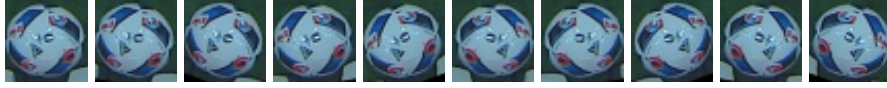


Fig. 5. Extending one positive sample (leftmost) to ten, by applying rotations and mirroring operations.

objects. To address this, we utilised the Nelder-Mead [9] method to calibrate the position and orientation of the camera frame in the head. More details can be found in [6].

4.1 Ball detection

Our ball detection approach is divided into two stages. First, ball candidates are generated based on colour segmentation, colour histograms, shape, and size. White connected components in the image are found, and the Ramer-Douglas-Peucker [10] algorithm is applied to reduce the number of polygon vertices in the resulting regions. This is advantageous for quicker subsequent detection of circle circular. The detected white regions are searched for at least one third full circle shapes within the expected radius ranges. Colour histograms of the detected circles are calculated for each of the three HSV channels, and compared to expected ball colour histograms using the Bhattacharyya distance. Circles with a suitably similar colour distribution to the expected one are considered to be ball candidates.

In the second stage of processing, a dense histogram of oriented gradients (HOG) descriptor [5] is applied in the form of a cascade classifier, with use of the AdaBoost technique. By utilising this classifier, we reject those candidates that do not have the required set of HOG features. In contrast to what is suggested in [5] however, which was targeted at pedestrian detection, we do not use a multi-scale sliding window technique. Instead, to save computational time, we only apply the HOG descriptor to the regions suggested by the ball candidates. The aim of using the HOG descriptor is to find a description of the ball that is largely invariant to changes in illumination and lighting conditions. The HOG descriptor is not rotation invariant however, so to detect the ball from all angles, and to minimise the user’s effort in collecting training examples, each positive image is rotated by $\pm 10^\circ$ and $\pm 20^\circ$ and selectively mirrored, with the resulting images being presented as new positive samples, shown for example in Fig. 5, where the original sample is the leftmost. Greater rotations are not considered to allow the cascade classifier to learn the shadow under the ball. The described approach can detect balls with very few false positives, even in environments cluttered with white, and under varying lighting conditions. In our experiments, we found that this approach can detect a FIFA size 5 ball up to 4.5 m away with a success rate above 80% on a walking robot, and up to 7 m away on a stationary robot.

4.2 Localisation

Localisation of the robot on the soccer field—the task of estimating the 2D pose (x, y, θ) of the robot—is performed using the field line, centre circle and goal post detections. Each component of the 2D pose is estimated independently. To estimate the θ component, we keep track of initial orientation and maintain an internal correction term based on the angular deviation between the expected and detected orientations of the white lines. This approach does not rely on having an accurate gyroscope output, and in experiments was able to correct deviations up to 10° coming from the gyroscope. Using the estimated θ , which is normally quite exact, we can rotate every vision detection to align with the global field coordinate system. The detected line segments can thereby be classified as being either horizontal or vertical field lines. In each cycle of the localisation node, we use the perception information and dead-reckoning walking data to update the previously estimated 2D location. For updating 2D location, we distinguish x and y component using estimated θ . The y component of the localisation is updated based on the y -components of the detected centre circle, goal posts, and vertical field lines. With the assumption that the robot is always inside the field lines, the vertical sidelines can easily be differentiated and used for updates. The x -component of the localisation is analogously updated based on the x -components of the detected centre circle, goal posts and horizontal field lines. The horizontal lines belonging to the goal area are discriminated from the centre line by checking for the presence of a consistent goal post detection, centre circle detection, and/or further horizontal line that is close and parallel. This approach can easily deal with common localisation difficulties, such as sensor aliasing and robot kidnapping. In contrast to some other proposed localisation methods for soccer fields, this method is relatively easy to implement and very robust. Our experiments indicate that the mean error of our localisation is better than what was reported in both [7] and [11].

Acknowledgements

This research is supported by the Deutsche Forschungsgemeinschaft (German Research Foundation, DFG) under grants BE 2556/13.

Team Members

Team NimbRo commits to participating in RoboCup 2017 in Nagoya and to provide a referee knowledgeable of the rules of the Humanoid League. Currently, the NimbRo AdultSize soccer team consists of the following members:

Team leader: Sven Behnke

Team members: Grzegorz Ficht, Hafez Farazi and André Brandenburger

References

1. Philipp Allgeuer. Attitude Estimator, Jul 2014.

2. Philipp Allgeuer and Sven Behnke. Robust sensor fusion for robot attitude estimation. In *Proceedings of 14th IEEE-RAS Int. Conference on Humanoid Robots (Humanoids)*, Madrid, Spain, 2014.
3. Philipp Allgeuer and Sven Behnke. Fused Angles: A representation of body orientation for balance. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015.
4. S. Behnke and J. Stueckler. Hierarchical reactive control for humanoid soccer robots. 5(3):375–396, 2008.
5. Navtean Dalal and Bill Triggs. Object detection using histograms of oriented gradients. In *Pascal VOC Workshop, ECCV*, 2006.
6. Hafez Farazi, Philipp Allgeuer, and Sven Behnke. A monocular vision system for playing soccer in low color information environments. In *Proceedings of 10th Workshop on Humanoid Soccer Robots, IEEE-RAS Int. Conference on Humanoid Robots*, Seoul, Korea, 2015.
7. Tim Laue, Thijs Jeffry De Haas, Armin Burchardt, Colin Graf, Thomas Röfer, Alexander Härtl, and Andrik Rieskamp. Efficient and reliable sensor models for humanoid soccer robot self-localization. In *Fourth Workshop on Humanoid Soccer Robots*, pages 22–29, 2009.
8. Marcell Missura and Sven Behnke. Balanced walking with capture steps. In *Robot Soccer World Cup*, pages 3–15. Springer, 2014.
9. John A Nelder and Roger Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.
10. Urs Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer graphics and image processing*, 1(3):244–256, 1972.
11. Hannes Schulz and Sven Behnke. Utilizing the structure of field lines for efficient soccer robot localization. *Advanced Robotics*, 26(14):1603–1621, 2012.