Advanced Soccer Skills and Team Play of RoboCup 2017 TeenSize Winner NimbRo

Diego Rodriguez(✉), Hafez Farazi, Philipp Allgeuer, Dmytro Pavlichenko, Grzegorz Ficht, André Brandenburger, Johannes Kürsch, and Sven Behnke

Autonomous Intelligent Systems, Computer Science,
University of Bonn, Bonn, Germany
{rodriguez,farazi,pallgeuer,pavlichenko,ficht,behnke}@ais.uni-bonn.de,
http://ais.uni-bonn.de

Abstract. In order to pursue the vision of the RoboCup Humanoid League of beating the soccer world champion by 2050, new rules and competitions are added or modified each year fostering novel technological advances. In 2017, the number of players in the TeenSize class soccer games was increase to 3 vs. 3, which allowed for more team play strategies. Improvements in individual skills were also demanded through a set of technical challenges. This paper presents the latest individual skills and team play developments used in RoboCup 2017 that lead our team Nimbro winning the 2017 TeenSize soccer tournament, the technical challenges, and the drop-in games.

1 Introduction

Every year the RoboCup Humanoid League raises its bar in its competitions. This year, the league increased the allowed number of players in the TeenSize soccer games to 3 vs. 3, which encourages the development of more complex team play strategies. At the same time, a new competition called *Drop-in games* was introduced in which a team is composed of robots from different institutes or universities. This paper presents our recent developments to address these modifications and shows their performance in the competition. Our robots won all TeenSize 2017 competitions: the soccer tournament, the Drop-in games and the technical challenges. In RoboCup 2017 we used our fully open-source 3D printed platform, the igus® Humanoid Open Platform [1]. Moreover, we upgraded one of our classic robots, Dynaped, so that it is able to get up from the prone and supine lying positions in order to be rule-compliant. Both platforms are shown in Fig. 1, along with the human members of our team NimbRo. We released a video of the competition 2017 highlights[1].

[1] RoboCup 2017 NimbRo TeenSize highlights video: https://youtu.be/6ldHWWHfeBc.

H. Akiyama et al. (Eds.): RoboCup 2017, LNAI 11175, pp. 435–447, 2018.
https://doi.org/10.1007/978-3-030-00308-1_36

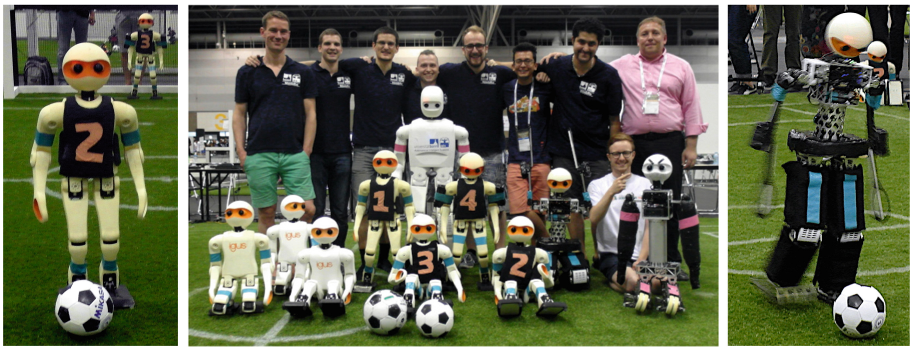


Fig. 1. Left: the igus® Humanoid Open Platform robot. Middle: the team NimbRo. Right: the upgraded Dynaped robot for RoboCup 2017.

2 Robot Platforms

Igus Humanoid Open Platform. Over the last four years, the igus® Humanoid Open Platform [1] (Fig. 1 left) has been developed as an open-source hardware and software project. Thanks to its 3D printed exoskeleton, the 92 cm tall robot weights only 6.6 kg. The platform incorporates an Intel Core i7-5500U CPU running a 64-bit Ubuntu OS, and a Robotis CM730 microcontroller board, which electrically interfaces with its Robotis Dynamixel MX actuators. The CM730 also incorporates 3-axis accelerometer and gyroscope sensors, for a total of 6 axes of inertial measurement. For visual perception, the robot is equipped with a Logitech C905 USB camera fitted with a wide-angle lens. The robot software is based on the ROS middleware, and is a continuous evolution of the ROS software that was written for the NimbRo-OP [2].

Dynaped. Dynaped has been playing since RoboCup 2009 for team NimbRo. Its original design had 14 degrees of freedom (DOF): 5 DoF per leg, 1 DoF per arm, and 2 DoF in the neck. Dynaped is distinguished by its effective use of parallel kinematics coupled with high torque, provided by pairs of EX-106 actuators in the roll joints of the hip and ankle, and pitch joints in the knee. All other DoFs are driven by single actuators. The torso is constructed entirely of aluminum and consists of a cylindrical tube and a rectangular cage that holds the electronics. Similar to the igus® Humanoid Open Platform, Dynaped is equipped with an Intel Core i7-5500U CPU, a CM740 controller board (newer version of CM730), and a USB camera with a wide-angle lens. Dynaped used the same software components as the igus® Humanoid Open Platform such as perception, bipedal walking, team communication, soccer behaviors, among others, thanks to their modularity and robustness. Dynaped's competition performance and hardware design, contributed to NimbRo winning the Louis Vuitton Best Humanoid Award in both 2010 and 2012 [3].

In order to be allowed to play in RoboCup 2017, Dynaped needed to be upgraded. Having only 1 DOF per arm, Dynaped was not able to get up either from prone or supine lying position. We included thus an additional DOF in each arm, namely a pitch elbow. Both the arm and the forearm are made from carbon fiber and reinforced against torsion with aluminum bars (Fig. 1 right).

3 Software Design

3.1 Visual Perception

As part of the preprocessing, we convert the RGB images to the HSV space because of its intuitive nature and handful separation of brightness and chromatic information. To compensate the high distortion coming from the wide-angle lenses, we use the pinhole camera model to compensate radial and tangential distortion. Using this distortion model, once the object of interest is identified, we project it into egocentric world coordinates. For calibrating the pose of the camera, we use the Nelder-Mead method [4] to minimize the reprojection error of detected field lines.

Ball Detection. We utilize a histogram of oriented gradients (HOG) descriptor in a cascade classifier trained using AdaBoost with positive and negative samples. Because of the computational cost of using sliding windows [5] we use the descriptor only on those candidates that are preselected based on shape, size, and colour.

Line Detection. On artificial grass, the painted field lines are not clearly white; thus an edge detector followed by probabilistic Hough line detection are implemented. Line segments are filtered to avoid false positives. Finally, the remaining similar line segments are merged to produce fewer larger lines. The shorter line segments are used for detecting the field circle, while the remaining lines are passed to the localization method. Figure 2 shows an example of the output of the object detection. For more details, please refer to [6].

Localization on the Soccer Field. We propose a multi-hypothesis model to estimate the three-dimensional robot pose (x, y, θ) on the field. We mainly use integrated gyroscope values as the source of orientation information and treat the heading initialization a classification problem making use of the specific possible locations where the robot can be placed according to the rules, namely, at the touch-line in the robot's own half facing the field, or near the center circle and goal area facing the opponent goal. We start four instances of our localization module with different initial hypothesis locations. For each instance, we handle the unknown data association of ambiguous landmarks, such as goal posts and T-junctions. Over time, we try to update the location of the hypothesis towards an estimated position. We update the location based on a probabilistic model

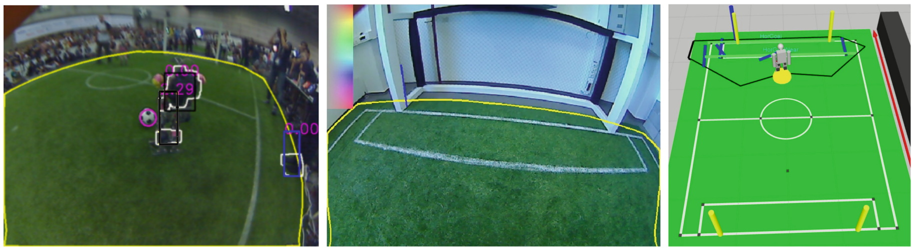


Fig. 2. Left: Image with ball (magenta circle), obstacles (black square) and field boundary (yellow lines) detections on RoboCup 2017. Middle: one input image for our localization method. Right: visualization of the localization of the robot in Rviz. (Color figure online)

of landmark observations involving mainly lines and goal posts [6]. The inputs to the probabilistic model come from the vision module and dead-reckoning odometry data. Finally, the hypothesis with the highest probability prevails and the others are deleted. A sample output of the localization is shown in Fig. 2.

3.2 Bipedal Walking

The gait of our robots is formulated in three different spaces: joint space, inverse (Cartesian) space and abstract space. The last one is a convenient formulation for humanoid robots for balancing and walking presented in [7]. The central pattern generator based gait is an extension of our previous work [8]. Starting from a halt pose defined in the abstract space, the central pattern adds waveforms features like leg lifting, leg swinging, arm swinging, among others. The resulting abstract configuration is then transformed to the inverse space where more motion components are added. The result is finally converted into joint space to command the actuators. Several feedback mechanisms have been implemented on top of the open-loop gait that help to stabilize the robot [9]. Each of these mechanisms acts as a PID-feedback controlling the fused angle deviations of the robot by adding corrective actions to the central pattern generated waveforms. The fused angles are an intuitive representation of orientations that offer benefits over Euler angles for balance [10]. These mechanisms, illustrated in Fig. 3, include arm angle, hip angle, continuous foot angle, support foot angle and CoM shifting. The step timing is computed using our capture step framework [11], based on the lateral CoM state [9].

3.3 Soccer Behaviors

Based on the visual perception of the game state, including ball detections, obstacle detections, and the estimated pose of the robot on the field, our robots must decide on and execute a strategy for playing soccer. This primarily involves localizing the ball and scoring a goal while avoiding obstacles, but also extends to team communications, team play, and coordination of the game using the

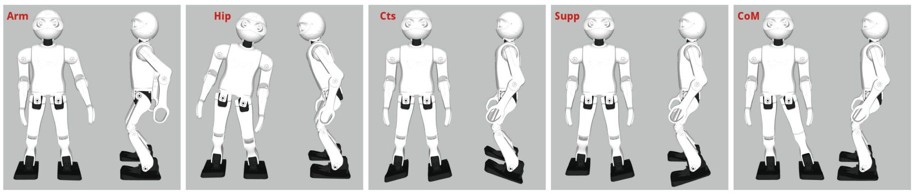


Fig. 3. Corrective actions in the sagittal and lateral planes, from left to right: arm angle, hip angle, continuous foot angle, support foot angle, and CoM shifting actions. The actions have been exaggerated for clearer illustration.

information from the RoboCup Humanoid league game controller. A custom two-layered hierarchical finite state machine (FSM) has been implemented for this purpose and runs in a separate behaviors node. The lower of these two layers is referred to as the Behavior FSM, and is responsible for implementing low-level skills such as searching for the ball, going to the ball, dribbling and kicking. The upper layer is referred to as the Game FSM, and builds on the skills implemented in the lower layer to implement game behaviors such as default ball handling, which attempts to kick or dribble a ball into goals, and positioning, which is used for the auto-positioning setup phase during a kickoff. In general, the game states combine groups or sequences of skills to execute certain soccer game state-specific behaviors.

3.4 Team Play

Teams participating in the TeenSize class in RoboCup 2017 can be composed by a maximum of three robots: one goalkeeper and two field players. We define dynamic *Player Tasks* which are frequently reassigned during the game. This task tells the robot what it is supposed to do according to its own state in the field and the state of its teammates. We define the following tasks: Attack, Defend, KeepGoal, ChangeTask and WaitClearOut. In addition, we define a task manager which is in charge of the safe assignment of these tasks. Each of this tasks is associated with a respective state of the game FSM (Fig. 4).

Attack Task. A robot with this task, known as *striker*, has active interaction with the ball. In possession of the ball, the robot will try to score either by kicking directly or by dribbling to get a better position for kicking the ball. The robot will also reach the ball and search for the ball in case the robot does not possess it. Searching for the ball, the robot goes first to the place where the ball was last seen. It turns on the spot and if the ball location is still unknown, it will go to the penalty marks, first to the closest and then to the furthest one. Reaching the ball means to place the robot behind the ball so it can kick or dribble. When approaching the ball, the robot does consider the ball as an obstacle in order to avoid undesired hits.

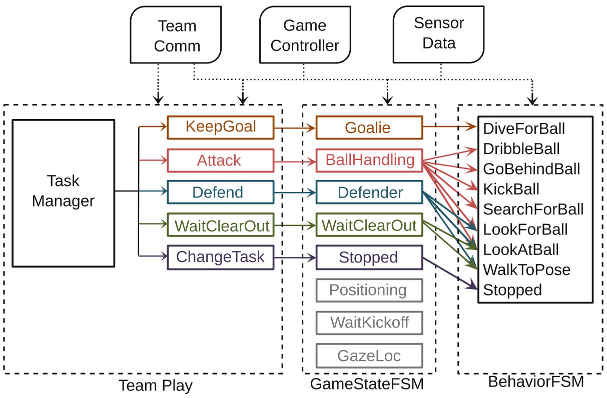


Fig. 4. Based on sensor data, the game controller and the team communication, the task manager assigns a task to each robot. Each task is associated to a state of the Game FSM, which triggers a sequence of behaviors in a lower-level Behavior FSM.

Defend Task. The *defender* robot is not supposed to have contact with the ball but to be ready to change its task and approach the ball if necessary. Its position is defined by a vector coming from the middle of its own goal towards the position of the ball. The magnitude of this vector is defined proportional to the distance of the ball to the own goal and saturated in order to avoid collisions with team members. In case the ball is not visible, this magnitude is calculated using the pose of the striker. In this manner, the robot is able: (i) to block opposite direct shots, (ii) to be ready for one-vs-one fights, and (iii) to get possession of the ball in case the previous striker is taken out of the match. With respect to the orientation, the robot tries to look in the direction of the ball. Figure 5 shows the pose of the defender for different ball locations. In order to avoid collisions with other teammates, the initial shortest path (straight line) is modified such that any teammate in between the path is surrounded (Fig. 6).

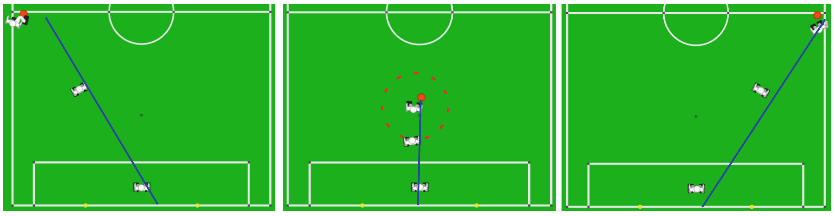


Fig. 5. Different poses that the defender might have according to the ball pose. On the middle the minimum distance between field players is shown in red. (Color figure online)

KeepGoal Task. The behavior of this task mainly depends on the proximity of the ball, of the opponents, and of the teammates to the own goal. Because

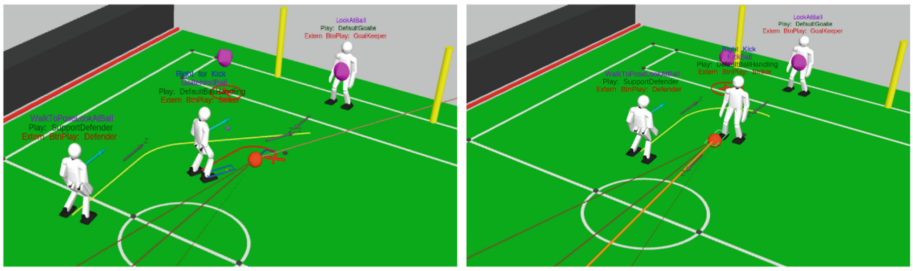


Fig. 6. Collision avoidance between field players. The defender considers the pose of the striker and surrounds it.

the other teammates have to coordinate their behaviors according to what the goalkeeper is doing, the decision if the goalkeeper has to clear out the ball is managed by the task manager. When the goalkeeper receives the signal to clear out the ball, it goes to the ball and hits it in direction of the opposite goal. It can also get a signal to dive including the diving direction. On the other hand, if the task manager decides that the goalkeeper does not need to clear out the ball, the robot will only move laterally on the same horizontal level of the ball position.

WaitClearOut. Different to the FIFA rules, the RoboCup Humanoid League has a special rule that prohibits more than one robot to be in the own area for more than 10 s. For this reason, when the ball is the own goal area, an additional coordination between team members is required. When a robot announces that it will clear out the ball from the own goal area, a *WaitClearOut* task reassignment occurs. In this task, the robots go closer to the ball without entering the own goal area to avoid the illegal defense. The path is planned such that the robot will not block the shot from an other robot clearing out the ball. In addition, the target pose in this task is assigned such that two robots will not collide with each other (Fig. 7).

The task manager of each robot determines the desired task and triggers special events that have to be coordinated. The desired task depends on each role. A goalkeeper only has assigned the task *KeepGoal*, while a field player can alternate between *Attack*, *Defend*, *ChangeTask* and *WaitClearOut*. In order to handle possible noise or very fast alternating data that could lead to a continuous task change, the request is only made once the decision is confirmed for a defined number of consecutive cycles. This voting system is persistent for all decisions taken by the task manager. If there is only one field player in the match, it will be assigned to *Attack*. The robot with the task of *Defend* can request a task change to the striker, but not vice versa. If the defender estimates that it is the closest to the ball, it will request a task change. When a striker leaves the field, because of a service or a penalization, it announces its egress such that a task assignment can be done immediately.

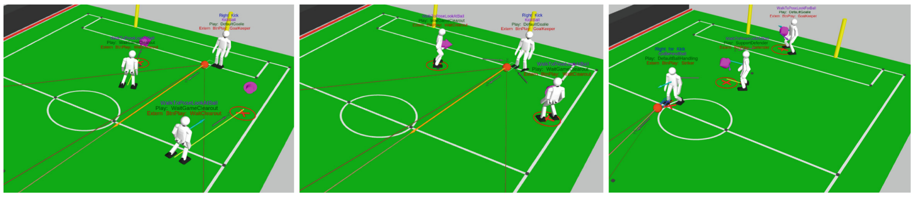


Fig. 7. Ball clear out by the goalkeeper. The yellow line represents the planned path, while the surrounded red cross refers to the instantaneous 2D target pose. Note that the robots always look at the ball. The figure at the rightmost shows the game after the clear out. (Color figure online)

Clearing out the ball by the goalie is a special event to be handled by the task manager because it implies task changes of the field players. In this case, the rest of the field players change to the *WaitClearOut* task, that place them in a strategic position once the ball has been cleared out. The decision of clearing out the ball by the goalie depends mostly on the ball location. The field is divided in three regions (Fig. 8). The first region is the goal area with an additional outer tolerance. In goal area, the goalie gets the highest priority, and when it is asked to clear out the ball, the field players set their task to *WaitClearOut*. The second region limits the possible areas where the goalkeeper can clear out the ball. If the ball is further away, the goalie just waits for its teammates. In this region, however, the goalie only chases the ball to clear it out if there is no teammate close to the own goal, i.e., if there is no teammate between our goal and the presence line (Fig. 8). In the third region, the goalkeeper takes a more passive role and gazes at the ball to be ready for diving.

The task assignment is based on an asynchronous request-and-response system that ensures that there is only one robot actively interacting with the ball. This prohibits, for example, that two robots try to kick the ball simultaneously which could lead to team self-collisions. The request for a task reassignment depends on the state of the robot and its teammates. This state comprises current task, ball distance, ball visibility, ball possession, ball location, current robot pose, if the robot is active and if the robot is not fallen. This information is estimated using the team communication data broadcast at a rate of 8 Hz. Only recent

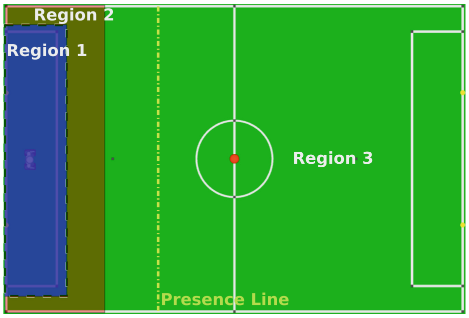


Fig. 8. Areas determining goalkeeper ball clearing behavior. In Region 1, goalkeeper needs to clear out the ball. In Region 3, the robot is remains in its goal. In Region 2, the goalkeeper clears out the ball if there is no field player between the own goal and the presence line (yellow dotted line). (Color figure online)

data (received in the last 5 s) is, however, considered because of possible hardware failures or communication errors.

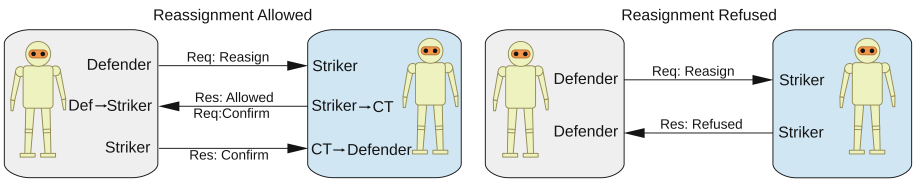


Fig. 9. Task assignment. A defender sends a request to change task. If the request is accepted, the striker is assigned to ChangeTask. It requests a confirmation, and the defender changes its task. If the request is refused, both robots keep their current tasks.

For task reassignment, the robot with the *Attack* task has a higher priority over *Defend*, i.e., a defending robot can only change its task if it is allowed by the current striker. When the defender finds itself in a better position to possess the ball, it requests to change tasks. If the striker also determines that its teammate is in a more convenient position, the striker changes its task to *ChangeTask* and sends back a response. The requester changes correspondingly its task and sends a confirmation such that the robot with task *ChangeTask* can change to the new task. On the other hand, if the original striker finds that it is in a better position to possess the ball, it sends back a negative response and no task reassignment takes place (Fig. 9).

The team play strategies presented here were used in the RoboCup2017 competition. Figure 10 shows official matches in which the striker and defender roles can be distinguished.

Fig. 10. Team play in RoboCup 2017.

3.5 Landing Motions

In a match during the competition, the robot might fall even with a robust and fine-tuned gait. The actions of our opponent can lead to difficult situations as happened in RoboCup 2016 [12]. This implies that the robot needs to know how to land. We designed landing motions that are activated when the robot would face an inevitable fall situation either in forward or backward direction. The aim of these motions is to protect the hardware of the robot from impacts in delicate parts of the robot, e.g. the knees. The designed motions are shown in Fig. 11.

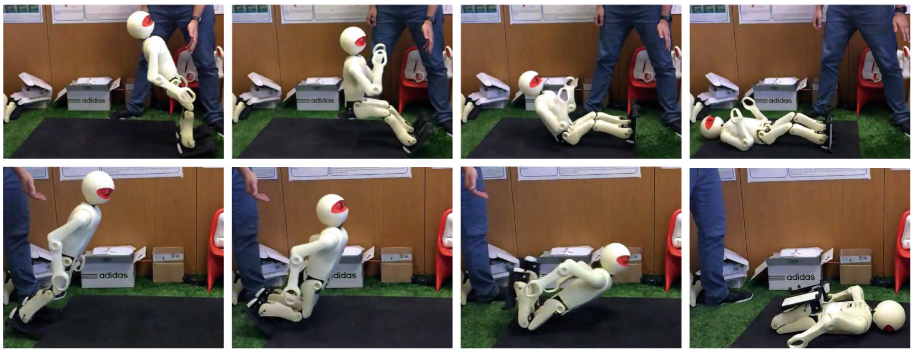


Fig. 11. Landing Motions in forward and backward direction.

3.6 Human-Robot Interfaces

To configure and calibrate the robots, a web application system is used with the robot PC as the web server. Even over a poor quality wireless network, the connection is robust by making use of the client-server architecture of web applications and the highly developed underlying web protocols. In addition, most of the processing is carried out by the client, resulting in a low computational cost for the robot. The web application allows the user: to start, to stop, to monitor ROS nodes, to display status information, to dynamically reconfigure the robot, and to visualize the processed vision, amongst others.

4 Game Performance

In RoboCup 2017, our robots scored 37 goals in 12 matches and did not receive any goal. 27 goals were scored during seven games of the TeenSize tournament and the remaining 10 goals were scored in five Drop-in games. This proved the robustness of the methods presented in this paper. Our robots won all the matches in the TeenSize tournament including the final 2:0 vs. HuroEvolutionTN (Taiwan). In the Drop-in games, the individual skills were tested and our robots have obtained 21 points in total, having a margin of 19 points to the team in second place.

5 Technical Challenges

In RoboCup 2017 there were four technical challenges. They were designed to evaluate specific capabilities of robots in isolation, separately from the regular games. In this section we briefly describe our strategies for these challenges.

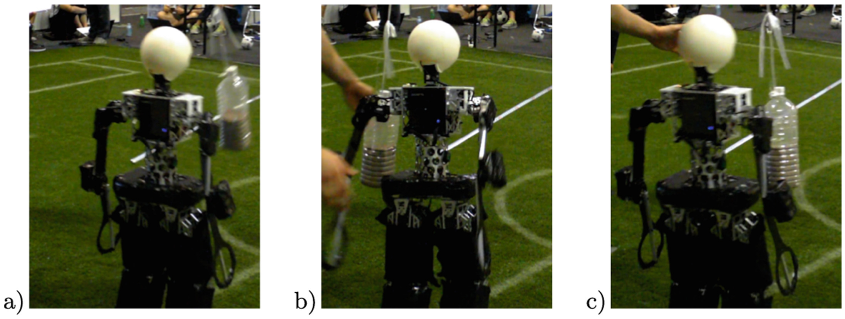
a) b) c)


Fig. 12. Dynaped withstanding a push from the front. (a) Before impact. (b) Immediately after impact. One can see that the robot is unstable. (c) Stable posture is recovered.

Push Recovery. In this challenge, the robot is pushed from the front and from the back. The goal is to withstand the impact and recover a stable posture. In RoboCup 2017, Dynaped successfully completed this challenge, withstanding the push from a 1.5 kg pendulum which was retracted by 55 cm (Fig. 12).

High Jump. In this challenge, the task is to jump as high as possible and remain in the air as long as possible. In order to perform this task, a jumping motion was designed using our keyframe editor. In RoboCup 2017, one of ours igus® Humanoid Open Platform robot successfully performed a jump of height 4.5 cm, remaining 0.192 s in the air and stand stable afterwards.

High Kick. The goal of this challenge is to kick the ball over the obstacle into the goal. In order to complete this challenge and overcome as high an obstacle as possible, a modified foot was used by one of our igus® Humanoid Open Platform robots for kicking. The foot had a smooth concave shape which allowed it to scoop the ball effectively and, hence, kick it upwards, overcoming the obstacle. In RoboCup 2017, our robot was able to complete this challenge with a height of 8 cm. The execution of a high kick over a 21 cm obstacle, recorded during testing in our lab, is shown in Fig. 13.

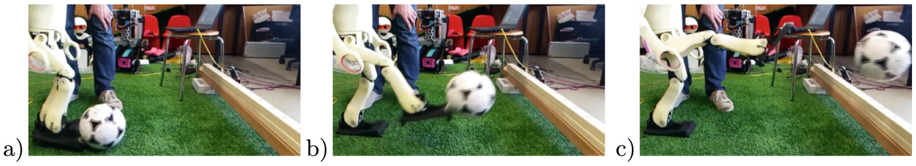
a) b) c)


Fig. 13. Robot performing a high kick over a 21 cm obstacle. (a) The ball is being scooped by the foot. (b) The ball is kicked upwards. (c) The ball overcame the obstacle.

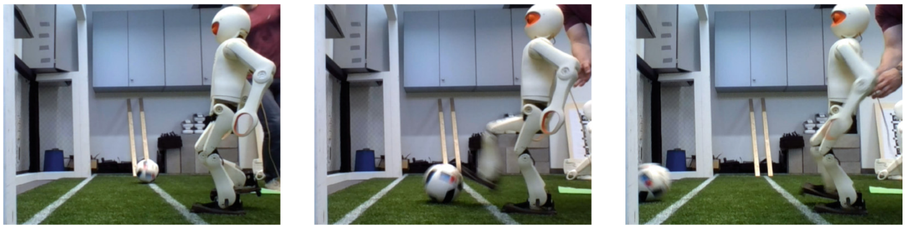


Fig. 14. Robot scoring from a moving ball. Left: the robot is waiting for the ball. Middle: the kick is executed. Right: a goal is scored.

Goal Kick from Moving Ball. The task of this challenge is to score a goal by kicking the moving ball into the goal. The ball is rolling along the goal area line. We solved this task as follows: first, we shift all weight onto one of the legs while having the other leg lifted and ready to kick; then, we estimate the velocity of the rolling ball and hence, time of arrival thereof to the kicking region; and, finally, we perform the kick according to the previous estimate. Following this strategy our robot was able to successfully complete this challenge. Our robot, performing this task during tests in our lab, is shown in Fig. 14.

6 Conclusions

In this paper, we presented the methods and approaches that lead us to win all possible competitions in the TeenSize class for the RoboCup 2017 Humanoids League in Nagoya: the soccer tournament, the Drop-in games, and the technical challenges. We presented individual skills regarding the perception and the bipedal gait, and their application in the technical challenges. Additionally, team skills were also extensively explained.

Acknowledgements. This work was partially funded by grant BE 2556/13 of German Research Foundation.

References

1. Allgeuer, P., Farazi, H., Schreiber, M., Behnke, S.: Child-sized 3D Printed igus humanoid open platform. In: Proceedings of 15th IEEE-RAS International Conference on Humanoid Robots (Humanoids) (2015)
2. Allgeuer, P., Schwarz, M., Pastrana, J., Schueller, S., Missura, M., Behnke, S.: A ROS-based software framework for the NimbRo-OP humanoid open platform. In: 8th Workshop on Humanoid Soccer Robots, International Conference on Humanoid Robots (2013)
3. Missura, M., Münstermann, C., Mauelshagen, M., Schreiber, M., Behnke, S.: RoboCup 2012 best humanoid award winner NimbRo TeenSize. In: Chen, X., Stone, P., Sucar, L.E., van der Zant, T. (eds.) RoboCup 2012. LNCS (LNAI), vol. 7500, pp. 89–93. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39250-4_9

4. Nelder, J.A., Mead, R.: A simplex method for function minimization. Comput. J. **7**(4), 308–313 (1965)
5. Dalal, N., Triggs, B.: Object detection using histograms of oriented gradients. In: Pascal VOC Workshop, ECCV (2006)
6. Farazi, H., Allgeuer, P., Behnke, S.: A monocular vision system for playing soccer in low color information environments. In: 10th Workshop on Humanoid Soccer Robots, IEEE-RAS International Conference on Humanoid Robots (2015)
7. Behnke, S.: Online trajectory generation for omnidirectional biped walking. In: Proceedings of 2006 IEEE International Conference on Robotics and Automation (ICRA) (2006)
8. Missura, M., Behnke, S.: Self-stable omnidirectional walking with compliant joints. In: 8th Workshop on Humanoid Soccer Robots, Humanoids Conference (2013)
9. Allgeuer, P., Behnke, S.: Omnidirectional bipedal walking with direct fused angle feedback mechanisms. In: Proceedings of 16th IEEE-RAS International Conference on Humanoid Robots (Humanoids) (2016)
10. Allgeuer, P., Behnke, S.: Fused angles: a representation of body orientation for balance. In: International Conference on Intelligent Robots and Systems (IROS) (2015)
11. Missura, M., Behnke, S.: Walking with capture steps. In: IEEE-RAS International Conference on Humanoid Robots, pp. 526–526 (2014)
12. Farazi, H., et al.: RoboCup 2016 humanoid teensize winner nimbro: robust visual perception and soccer behaviors. In: Behnke, S., Sheh, R., Sarıel, S., Lee, D.D. (eds.) RoboCup 2016. LNCS (LNAI), vol. 9776, pp. 478–490. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68792-6_40