

# Soccer Behaviors for Humanoid Robots

Jörg Stückler and Sven Behnke

Humanoid Robots Group, Computer Science Institute  
University of Freiburg, Georges-Köhler-Allee 52, 79110 Freiburg, Germany  
Email: { stueckle | behnke }@informatik.uni-freiburg.de

**Abstract**—As the RoboCup Humanoid League moves from less demanding competitions to soccer games, the complexity of behavior control increases significantly. This calls for structured behavior engineering.

In this paper, we describe the design of the behavior control software for the Humanoid League team NimbRo. The control software is based on a framework that supports a hierarchy of reactive behaviors. It is structured both as an agent hierarchy (joint – body part – player – team) and as a time hierarchy. The speed of sensors, behaviors, and actuators decreases when moving up in the hierarchy. The lowest levels of this framework contain position control of individual joints and kinematic interfaces for body parts. At the next level, basic skills like omnidirectional walking, kicking, and getting-up behaviors are implemented. These are used by soccer behaviors like searching for the ball, approaching the ball, avoiding obstacles, and defending the goal. Finally, on the team level, the robots communicate via a wireless network to share information about the world state and to negotiate roles like attacker and defender.

Our robots performed well at the RoboCup 2006 competition, which took place in Bremen, Germany. The KidSize robots won the Penalty Kick competition and came in second in the overall Best Humanoid ranking, next only to the titleholder, Team Osaka.

## I. INTRODUCTION

In the RoboCup Humanoid League, robots with a human-like body plan compete with each other [1]. The robots must have two legs, two arms, a head, and a trunk. Size restrictions make sure that the center of mass of the robots is not too low, that the feet are not too large, and so on. The robots are grouped into two size classes: KidSize (up to 60cm) and TeenSize (80cm-160cm). The fully autonomous humanoid robots must be able to walk on two legs. The RoboCup Humanoid League was established in 2002 and has developed quickly since. Due to the complexity of building and controlling humanoid soccer robots, in the first years, only less demanding competitions like walking around a pole and penalty kick were held. At RoboCup 2005, 2 vs. 2 soccer games were introduced for the KidSize robots. The Humanoid League rules [2] have been derived from the FIFA laws. Some simplifications apply, however. For example, the offside rule is not observed and key objects are color-coded in order to simplify the perception of the game situation.

The complexity of playing soccer games is much higher than the complexity of penalty kick. The ball might be at any position on the field and the robots need to search for it if they lost track of its position. The robots must also perceive at least the two goals and the other players. Higher-level behaviors require self-localization on the field. The distances to walk

are much longer. Hence, the walking speed must be higher. As two robots play together, there is need for coordination. While some teams use one dedicated goalie and one field player, other teams use two field players. This makes dynamic role assignment necessary. Last, but not least, in soccer games robots of the two teams interact physically when going for the ball. This disturbs walking and leads to falls. The robots need to get up from the ground by themselves in order to continue play. As a result of these difficulties, in the recent RoboCup 2006 competition in Bremen, only a fraction of the teams able to play penalty kick was able to play decent soccer games.

The other RoboCupSoccer leagues have been facing the complexity of soccer games for some years now. There, tools for structured behavior engineering have been developed. For example, Jaeger and Christaller proposed the Dual Dynamics architecture [3], which has been used in the MiddleSize League. The architecture distinguishes elementary behaviors, which implement a target dynamics, and complex behaviors, which control the activation of elementary behaviors. Another tool used in the MiddleSize League is the BAP-framework of Utz et al. [4], which allows for specifying hierarchical, event-driven, behavior-based control systems. In the Four-Legged League, the German Team developed XABSL [5]. It allows for XML-based specification of hierarchies of behavior modules that contain state machines for decision making. State transitions are modeled as decision trees. Parts of the German Team system are used now in the Humanoid League by Darmstadt Dribblers, Humanoid Team Humboldt, and BreDoBrothers. Another example for a behavior architecture used in more than one league is the architecture proposed by Laue and Röfer [6], which combines action selection and potential field motion planning. It was used to control SmallSize and Aibo soccer robots.

To implement the behavior control software for the humanoid soccer robots of our team NimbRo, we used a framework that supports a hierarchy of reactive behaviors [7]. This framework has been originally developed for the FU-Fighters SmallSize robots. It was later adapted to the FU-Fighters MiddleSize robots and also used by CMU in the Four-Legged League [8]. We adapted it for the control of soccer playing humanoid robots by extending the agent-hierarchy to: joint – body part – player – team. The lowest levels of this hierarchy contain position control of individual joints and kinematic interfaces for body parts. At the next level, basic skills like omnidirectional walking, kicking, and getting-up behaviors are implemented. These are used at the player level by soccer



Fig. 1. NimbRo 2006 KidSize robot Paul.

behaviors like searching for the ball, approaching the ball, avoiding obstacles, and defending the goal. Finally, on the team level, the players communicate via a wireless network to share information about the world state and to negotiate roles like attacker and defender.

The remainder of this paper is organized as follows. In the next section, we describe our KidSize 2006 robots. We cover mechanical design, electronics, perception, and infrastructure components. Sec. III describes our behavior control framework. The implementation of basic skills is covered in Sec. IV. Section V explains the design of our soccer behaviors. Finally, we present the results of using the soccer behaviors at RoboCup 2006.

## II. NIMBRO 2006 KIDSIZE ROBOTS

### A. Mechanical Design

Fig. 1 shows Paul, one of our 2006 KidSize robots. As can be seen, the robot has human-like proportions. Its mechanical design focused on simplicity, robustness, and weight reduction. The KidSize robots have a height of 60cm and weigh only 2.9kg, including batteries. They are driven by 24 Dynamixel actuators: 8 per leg, 3 in each arm, and two in the trunk. For the leg and the trunk joints, we use the DX-117 actuators (66g, 37kg-cm). Three orthogonal axes constitute the 3DOF hip joint. For the hip pitch and roll axes, we use two of these actuators in a parallel master-slave configuration. This doubles the torque and lowers operating temperatures. Two orthogonal servos form the 2DOF ankle joint and one servo drives the knee joint. The trunk joints are in the pitch and yaw axes. The arms do not need to be as strong. They are powered by DX-113 actuators (58g, 10.2kg-cm). Two orthogonal servos constitute the shoulder joint and one servo drives the elbow joint. The skeleton of the robot is constructed from aluminum extrusions with rectangular tube cross section. We removed all material not necessary for stability. The feet, the forearms, and the head are made from sheets of elastic carbon composite material.

### B. Electronics

The Dynamixel actuators have a RS-485 differential half-duplex interface. Each robot is equipped with a HCS12 microcontroller board, which manages the detailed communication with all Dynamixels. The Dynamixel actuators have a flexible interface. Not only target positions are sent to the actuators, but also parameters of the control loop, such as the compliance. In the opposite direction, the current positions, speeds, loads, temperatures, and voltages are read back. In addition to these joint sensors, each robot is equipped with an attitude sensor, located in the trunk. It consists of a dual-axis accelerometer (ADXL203,  $\pm 1.5g$ ) and two gyroscopes (ADXRS 300,  $\pm 300^\circ/s$ ). The sensor signals are digitized with A/D converters of the HCS12. The microcontroller communicates with the Dynamixels via RS-485 at 1MBaud and with a main computer via a RS-232 serial line at 115KBaud. Every 12ms, target positions and compliances for the actuators are sent from the main computer to the HCS12 board, which distributes them to the actuators. The HCS12 sends the preprocessed sensor readings back.

We use a Pocket PC as main computer, which is located in the upper part of the robot. The FSC Pocket Loox 720 has a weight of only 170g, including the battery. It features a 520MHz XScale processor, a touch-sensitive display with VGA resolution, wireless LAN, a RS-232 serial interface, and an integrated 1.3 MPixel camera. This computer runs behavior control, computer vision, and wireless communication. We took the integrated camera out of the Pocket PC and connected it via an extension cable. Located above the Pocket PC, it looks in forward direction through a wide-angle converter.  $640 \times 480$  images can be captured at 15fps using DMA. In addition, a Lifeview FlyCam CF 1.3M that has been fitted to an ultra-wide-angle lens is looking in backward direction. Our soccer robots are powered by high-current Lithium-polymer rechargeable batteries, which are located in their lower back. Four Kokam 910mAh cells last for about 25 minutes of operation.

### C. Perception

Our robots need information about themselves and the situation on the soccer field to act successfully.

1) *Proprioception*: The readings of accelerometers and gyros are fused to estimate the robot's tilt in roll and pitch direction. The gyro bias is automatically calibrated and the low-frequency components of the tilt estimated from the accelerometers are combined with the integrated turning rates to yield an estimate of the robot's attitude that is insensitive to short linear accelerations. As described above, joint angles, speeds, and loads are also available. Temperatures and voltages are monitored to notify the user in case of overheating or low batteries.

2) *Visual Object Detection*: The only source of information about the environment for our robots is their camera. The wide field of view of the cameras allows the robots to see their own feet and objects above the horizon at the same time (see left part of Fig. 2). Our computer vision software

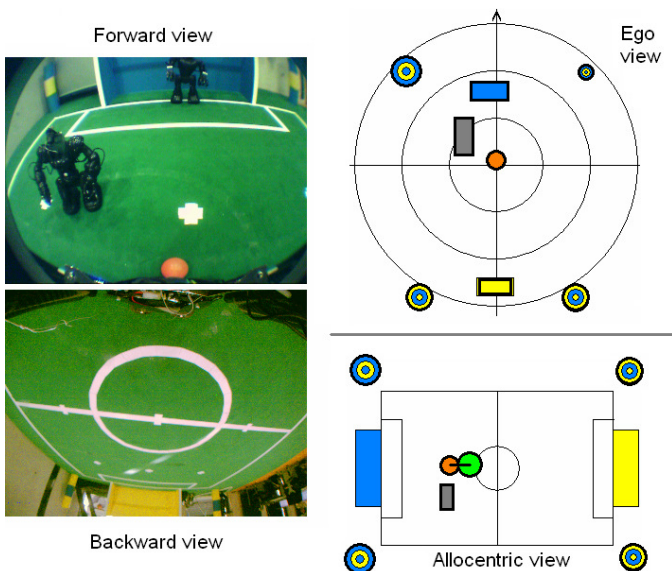


Fig. 2. Left: Images of the two cameras mounted on the robot. Upper right: Egocentric coordinates of key objects (ball, goals, corner poles, obstacle) detected in the image. Lower right: Localization of the robot, the ball, and the obstacle on the soccer field.

detects the ball, the goals, the corner poles, and other players based on their color in YUV space. Using a look-up table, the colors of individual pixels are classified into color-classes that are described by ellipsoids in the UV-plane. In a multistage process we discard insignificant colored pixels and detect colored objects. We estimate their coordinates in an egocentric frame (distance to the robot and angle to its orientation), based on the inverted projective function of the camera. We correct first for the lens distortion and invert next the affine projection from the ground plane to the camera plane. The estimated egocentric coordinates of the key objects are illustrated in the upper right part of Fig. 2. Here, the objects detected by both cameras are fused, based on their confidence. The objects are also merged with previous observations, which are adjusted by a motion model, if the robot is moving. This yields a robust egocentric world representation.

3) *Self-Localization*: The relative coordinates suffice for many relative behaviors like positioning behind the ball while facing the goal. To keep track of non-visible goals or to communicate about moving objects with other team members, we need the robot coordinates in an allocentric frame ( $(x, y)$ -position on the field and orientation  $\theta$ ). We solve self-localization by triangulation over pairs of landmark observations, i.e. detected goals and corner poles. When observing more than two landmarks, the triangulation results are fused based on their confidence. Again, the results of self-localization are integrated over time and a motion model is applied. The lower-right image of Fig. 2 illustrates the resulting allocentric representation.

#### D. Infrastructure

To support a team of soccer playing robots some infrastructure components are needed.

1) *Wireless Communication*: The Pocket PCs of our robots are equipped with wireless network adapters. We use the wireless communication to transmit debug information to an external computer via UDP, where it is logged and visualized. This allows the behavior engineers to keep track of the perceptions and the chosen actions. The wireless network is also used for transmitting the game state (kickoff, penalty, etc.) from the external PC to the robots. The robots communicate with each other to share perceptions and to negotiate roles.

2) *Simulation*: In order to be able to design behaviors without access to the real hardware, we implemented a physics-based simulation for the robots. This simulation uses the Open Dynamics Engine [9]. It was very helpful for the design of getting up motions, goalie motions, and team behaviors. The simulator is also used to develop learning algorithms, which are then applied to the real robot.

### III. BEHAVIOR ARCHITECTURE

We control the robots using a framework that supports a hierarchy of reactive behaviors [7]. This framework allows for structured behavior engineering. Multiple layers that run on different time scales contain behaviors of different complexity. When moving up the hierarchy, the speed of sensors, behaviors, and actuators decreases. At the same time, they become more abstract.

The framework forces the behavior engineers to define abstract sensors that are aggregated from faster, more basic sensors. One example for such an abstract sensor is the robot's attitude that is computed from the readings of accelerometers and gyros. Abstract actuators give higher-level behaviors the possibility to configure lower layers in order to eventually influence the state of the world. One such abstract actuator is the desired walking direction, which configures the gait engine, described below, implemented in the lower control levels. The behaviors within one layer of the behavior framework are activated according to the current state of its sensors. Activation is indicated by an activation factor in the interval  $[0, 1]$ . Each active behavior can manipulate the actuators in its layer. If multiple behaviors try to manipulate the same actuator, the actuator is set to the weighted sum of desired values, where the activation factors are used as weight. To prevent conflicting behaviors from being active at the same time, behaviors can inhibit other behaviors. If an inhibiting behavior is not completely active, the inhibited behaviors share the remaining activation, such that the activation factors sum to 1.

The control hierarchy of our soccer robots is arranged in an agent hierarchy:

- multiple joints (e.g. left knee) constitute a body part (e.g. left leg),
- multiple body parts constitute a player (e.g. field player), and
- multiple players constitute a team.

Two teams can be controlled simultaneously. The behavior framework manages all but the motor control loop within the Dynamixel actuators, which has been implemented by Robotis.

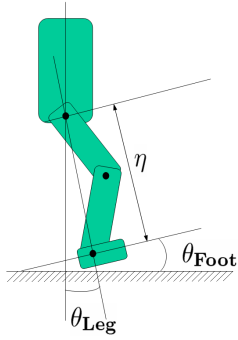


Fig. 3. Kinematic interface to a leg.

The behaviors on the lower level in the framework implement basic skills which generate target positions for individual joints at a rate of 83.3Hz. To abstract from the individual joints, we implemented here a kinematic interface for the body parts. It allows, for example, to independently change leg extension  $\eta$ , leg angle  $\theta_{\text{Leg}}$ , and foot angle  $\theta_{\text{Foot}}$ , as illustrated in Fig. 3. A detailed description of the kinematic leg interface is given in [10]. Several basic skills, described below, use this kinematic interface. The next higher level contains soccer behaviors which are executed at a rate of 41.7Hz. This structure restricts interactions between the system variables and thus reduces the complexity of behavior engineering.

#### IV. BASIC SKILLS

Fundamental for playing soccer are the abilities to walk and to kick. As body contact between the physical agents is unavoidable, the capability of getting up after a fall is also essential. For keeping a goal, the robot must be able to perform special motions. The basic skills are implemented on the body part layer. Fig. 4 illustrates the inhibitory structure of the basic skills and how the motor control layer is configured through actuators.

##### A. Omnidirectional Walking

We use the leg interface to implement omnidirectional walking. Shifting the weight from one leg to the other, shortening of the leg not needed for support, and leg motion in walking direction are the key ingredients of this gait. In contrast to the low-frequency gait of our 2005 robots [10], we were able to increase the step frequency significantly to 3.45Hz.

Walking forward, to the side, and rotating on the spot are generated in a similar way. The three basic walking directions can be smoothly combined. The robots are able to walk in every direction and to change their heading direction at the same time. The gait target vector  $(v_x, v_y, v_\theta)$  can be changed continuously while the robot is walking. This makes it possible to correct for deviations in the actual walking direction and to account for changes in the environment by using visual feedback. When using this flexible gait, the maximal forward walking speed of the robots is approx. 25cm/s. Fig. 5 shows image sequences of the robot Franz walking forward, laterally, and turning. Behaviors of the upper level can control the gait

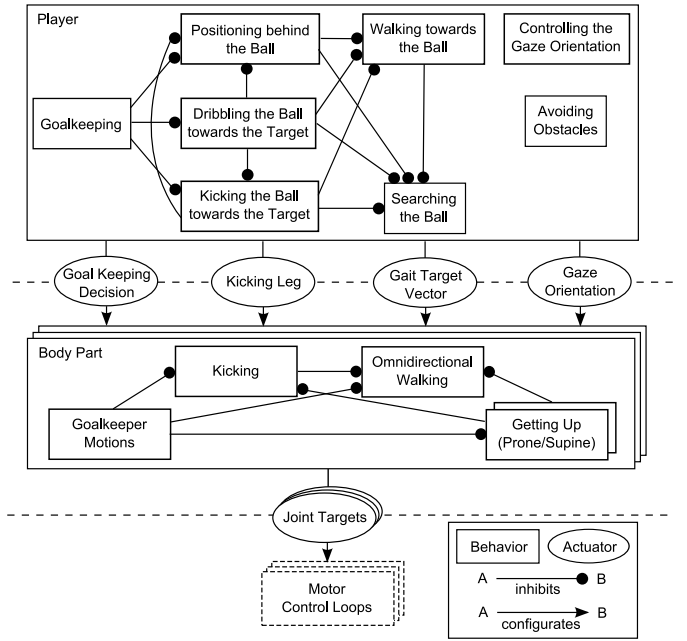


Fig. 4. Actuators, behaviors, and mutual inhibitions within the behavioral joint – body part – player hierarchy. Upper layer behaviors can configure lower layer behaviors by manipulating the upper layer actuators. The resulting values of the actuators depend on the activation factors and the inhibitory structure of the manipulating behaviors.

target vector with an actuator that enforces maximal speeds and accelerations.

##### B. Kicking

In addition to walking, we implemented kicking. An actuator allows behaviors in the upper level to trigger kicks with both the left and the right leg. Fig. 6 shows some of the trajectories generating the kicking motion. After inhibiting the walking behavior and coming to a stop, the robot moves its weight to the non-kicking leg (see hip roll angle). Then, it shortens the kicking leg, swings it back and accelerates forward. The kicking leg reaches its maximal speed when it comes to the front of the robot. At this point the hip pitch joint and the knee both move the foot forward and the ball is kicked. The kicking movement continues with deceleration of the foot and slow motion back to the bipedal stand. The resulting kick can be seen in Fig. 7.

##### C. Getting up after a Fall

Since in soccer games physical contact between the robots is unavoidable, the walking patterns are disturbed and the robots might fall. Hence, they must be able to detect the fall, to recognize their posture on the ground, and to get back into an upright posture. After falling, the robot's center of mass (COM) projection to the ground is outside the convex hull spanned by the foot-contact points. Additional support points like knees, elbows, and hands must be used in order to move the COM back inside the foot polygon.

Using their attitude sensors, the robots detect a fall, classify the prone or supine posture and trigger the corresponding getting-up sequence. We designed the getting-up sequences



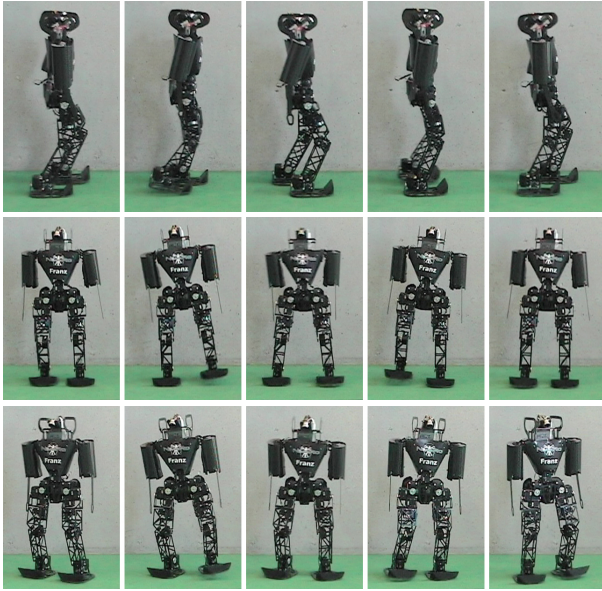


Fig. 5. Omnidirectional walking: forward, lateral, and turning on the spot.

in the simulator using sinusoidal trajectories [11]. Fig. 8 illustrates the four phases of getting up from the prone posture and Fig. 9 illustrates getting up from the supine posture. The getting-up sequences work very reliably. Under normal circumstances, i.e. appropriate battery voltage, the routines worked with 100 successes in 100 tests.

#### D. Goalkeeper Motions

The goalkeeper is capable of diving into both directions or to bend forward with spread arms. Fig. 10 shows Franz diving to the left. First, it moves its COM and turns its upper body towards the left while shortening the legs. As soon as it tips over its left foot, it starts straightening its body again. While doing so it is sliding on its hands and elbows. The fully extended robot covers the entire goal half. After the dive Franz gets up again, as described above.

### V. PLAYING SOCCER

In the following, we detail the soccer behaviors of the robots during a 2 vs. 2 soccer game, which build on the basic skills.

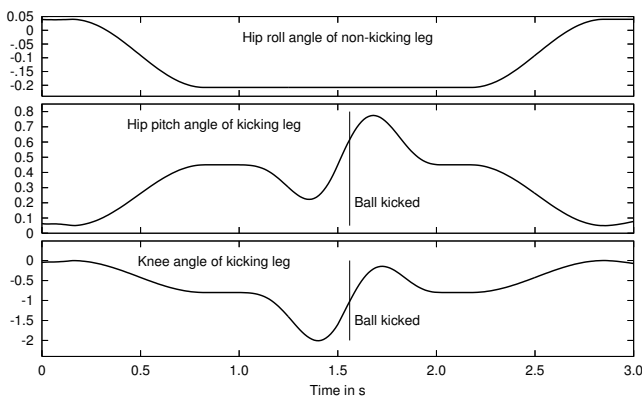


Fig. 6. Trajectories generated for kicking.

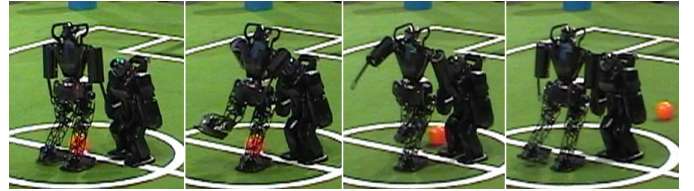


Fig. 7. Kicking the ball.

We also describe the concepts of world and internal state representation, which underlie the behaviors.

#### A. World and Internal State Representation

The soccer behaviors require knowledge of the current game situation. The visual perception supplies relative distance, angle, and perceptual confidence for the ball  $p_{Ball} = (d_{Ball}, \theta_{Ball}, c_{Ball})$ , the own goal  $p_{OwnGoal}$ , the opponent goal  $p_{OppGoal}$ , and the nearest obstacle  $p_{Obstacle}$ . The confidence is a value in the interval  $[0, 1]$ .

In the attacking role, the relative position and confidence of the opponent goal is used as the target to kick at. We refer to this target as the ball-target with relative position  $p_{BallTarget}$ . The decision for the kicking leg is made in every time step, depending on the relative position of the ball and the line from ball to ball-target, which we denote as ball-to-target-line. If the robot has to approach the ball-to-target-line from the right, it kicks with the left leg, and vice versa. To avoid oscillations it is important that the decision may only be changed if the distance of the robot to the ball-to-target-line is large.

To kick the ball with the chosen leg, the robot has to position itself behind the ball with lateral and sagittal offsets,  $\delta_l$  and  $\delta_s$  that depend on the distance between the legs and the length of the feet. The target position behind the ball,  $p_{BehindBall}$ , is calculated by adding the lateral and sagittal offsets to the ball position orthogonal to and along the ball-to-target-line, respectively. To generate smoothly approaching trajectories to this position, the sagittal offset is increased by an amount  $\delta_a$  that is proportional to the angle between the robot's heading direction and the ball-target. Fig. 11 illustrates two examples for positioning behind the ball with the left leg as kicking leg.

When playing as defensive field player, the own goal is used as ball-target, such that the position behind the ball is set to a position between ball and own goal. The distance kept to the ball depends on the distance to the own goal. A threshold for the minimal distance to the goal lets the robot stay out of its

- I. Lift the trunk and bring the forearms under the shoulders.
- II. Move the COM projection as close as possible to the leading edges of the feet by bending in the spine, the hip pitch and the knee joints.
- III. Straighten the arms to let the robot tip over the leading edges of the feet.
- IV. Bring the body into an upright posture.

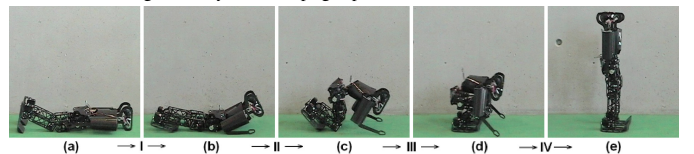


Fig. 8. (a)-(e) Starting and end positions of phases I-IV when standing up from the prone posture.

- I. Move the upper body into a sit-up posture and move the arms into a supporting position behind the back.
- II. Move into a bridge-like position using the arms as support.
- III. Move the COM over the feet by swinging the upper body to the front.
- IV. Move the body into an upright posture.

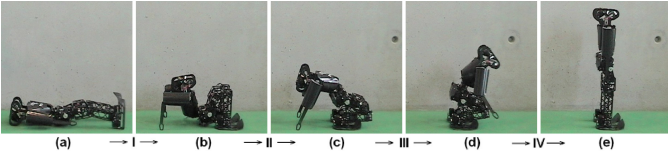


Fig. 9. (a)-(e) Starting and end positions of phases I-IV when standing up from the supine posture.

goal, as long as the ball is still far away. If the ball and the robot are near the goal, the robot keeps behind the ball at a minimum distance.

The robot maintains additional hypotheses about the relative ball location that are used for searching the ball. If a kick is triggered, one hypothesis is set in front of the robot at a distance depending on kick strength. The confidence of the hypothesis is discounted by the time since the kick started. Its relative position is altered according to the motion model. Additionally, hypotheses are maintained for the perceptions of the ball by other players on the field. The confidence of these hypotheses depends on the self-localization and ball perception confidences of the other players and the self-localization confidence of the robot itself.

### B. Soccer Behaviors

According to the current game situation, behaviors like searching the ball, positioning behind the ball, or avoiding obstacles are activated. These behaviors are implemented on the player level and use the actuator interface that basic skills of the lower layer provide. For example, they set the gait target vector or trigger a kick. Fig. 4 illustrates the inhibitory structure of the soccer behaviors and the actuator interface to the basic skills which reside on the lower layer.

1) *Searching the Ball*: Exploring the environment for the ball is always active, but inhibited by behaviors that activate when the ball has been perceived with a certain confidence. If a ball hypothesis with confidence over a certain threshold exists, the robot walks towards the most confident hypothesis. Otherwise, it turns towards the most confident hypothesis for a short time. If the ball still is not visible, it starts to walk around the center circle in a constant distance in order to inspect all parts of the field.

2) *Walking towards the Ball*: The robot walks straight towards the ball, if it perceives the ball. The own goal must be either not visible ( $c_{OwnGoal} < 0.1$ ) or far away ( $d_{OwnGoal} > 0.5m$ ) to avoid an own goal. This behavior controls the gait target velocity to keep the robot near the ball, e.g. if visual

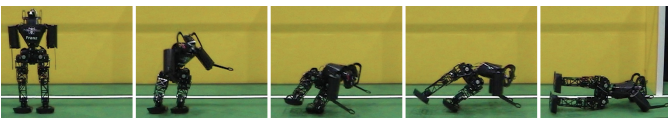


Fig. 10. Diving motion of the goalkeeper.

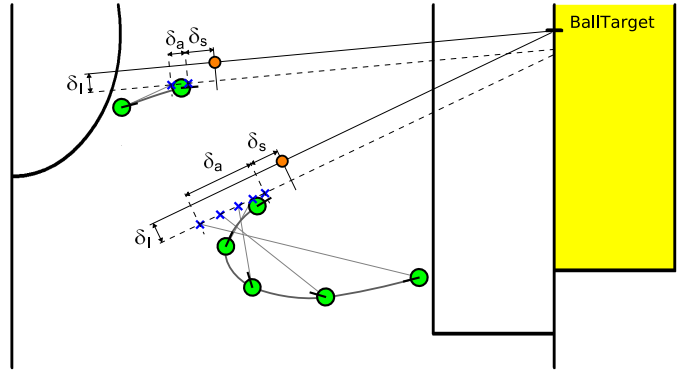


Fig. 11. Two examples showing sequences of robot poses, target positions behind the ball (blue crosses), and ball positions while approaching the ball with the left leg as kicking leg.

perception fails to detect the opponent goal. The behavior inhibits searching the ball.

3) *Positioning behind the Ball*: If the ball and the ball-target are perceived, the robot positions itself behind the ball, facing towards the ball-target. The robot is positioning on  $p_{BehindBall}$  by controlling the gait target velocity. If the distance to the target position is large ( $d_{BehindBall} > 0.5m$ ), the robot rotates towards the target position, such that it can approach it by mainly combining forward walking with turning. If it is near the target position ( $d_{BehindBall} \leq 0.25m$ ), the robot aligns itself towards the ball-target. For intermediate distances, the gait rotation is interpolated linearly between both alignment targets. The behavior also handles the case, when the ball is positioned between robot and target position. Here, the robot walks around the ball by walking towards the target position but avoiding the ball-to-target-line. When playing as defensive field player, the robot rotates towards the ball at any distance. It does not avoid the ball-to-target-line, because the ball-target is the own goal. This behavior inhibits walking towards the ball, such that the inhibited behavior may only be active, if the ball-target has not been perceived. It also inhibits searching the ball.

4) *Kicking the Ball towards the Target*: This behavior is activated as soon as the target position  $p_{BehindBall}$  has been reached with a certain precision in angle to the ball-target and in distance to the target position. If the precision conditions hold, a kick is triggered. Naturally, ball and ball-target must be perceived. To prevent own goals, the own goal must be either not visible or be seen in an angle that is larger than  $\pm \frac{\pi}{4}$ . If the ball comes into a kicking position by chance, the behavior initiates a kick with the corresponding leg. As the robot has to come to a complete stop before the kicking motion can be executed, the robot can cancel the kick, if the ball moves away in the meantime. This behavior inhibits searching the ball, walking towards the ball, and positioning behind the ball.

5) *Dribbling the Ball towards the Target*: If positioning behind the ball was not successful for a longer time, or the game started with a kick-off for the player, the robot activates dribbling the ball towards the ball-target for some time. Additional preconditions for activation are that the ball

and ball-target are perceived and the angle towards the ball-target is smaller than  $\pm\frac{\pi}{4}$ . Dribbling is performed by steering towards the ball. The forward walking speed is inversely related to the angle to the ball. In combination with positioning behind the ball, the robot is kept behind the ball, facing the ball-target when dribbling. Dribbling inhibits searching the ball, walking towards the ball, and positioning behind the ball. As we want the decision for dribbling to be strict, it also inhibits kicking the ball towards the target.

6) *Avoiding Obstacles*: After a fall, the robot needs valuable time to get back on its feet. The main reason for our robots to fall is physical contact with other robots. Hence, obstacle avoidance is an important feature. The visual perception supplies the behavior with the nearest obstacle. If it is detected ( $c_{Obstacle} > 0.1$ ), in a certain distance ( $d_{Obstacle} < d_{Obstacle\_max}$ ), and in front of the robot ( $|\theta_{Obstacle}| < 1.2$ ), obstacle avoidance is activated by a factor that interpolates linearly between a minimum and a maximum distance for the obstacle,  $d_{Obstacle\_min}$  and  $d_{Obstacle\_max}$ . The avoidance sets the gait target actuator to a constant and a variable part of the direction from obstacle to robot. The strength of the variable part depends on the distance to the obstacle, similar to the activation factor. If the ball is between obstacle and robot, the variable avoidance is weakened, such that the robot moves more aggressively behind the ball. A stuck situation is indicated by a resulting gait target vector that is small in length for a longer time. In this case, the robot may sidestep the obstacle, if the ball is not between the obstacle in the front and the robot and is perceived on one side of the obstacle ( $|\theta_{Obstacle} - \theta_{Ball}| > \frac{\pi}{4}$ ). The action is cancelled, if either the preconditions for sidestepping do not hold anymore or some time has passed since sidestepping has been activated. The deactivation of sidestepping after some time is important, because the decision for the sidestep direction is made only once on activation.

7) *Controlling the Gaze Orientation*: Although the robot has wide-angled views to the front and the rear, it cannot perceive objects on the sides. Thus, a gaze control behavior is always active and primarily keeps the ball within an angular range of  $\pm\frac{\pi}{4}$  by twisting the upper trunk with the trunk yaw joint. If the ball is not visible or within range and the robot is localized, it aligns the upper body onto the line between the goals to keep the localization landmarks visible. This is achieved by keeping the angle to the line within the angular range of  $\pm\frac{\pi}{4}$ . The twisting of the trunk is limited to  $\pm\frac{\pi}{4}$  in each direction.

8) *Goalkeeping*: The goalkeeper's objective apparently is to keep the ball out of the own goal. While the ball is visible and is not in kicking distance to the robot, the goalkeeping behavior is active. Otherwise, the robot behaves like a field player and tries to shoot the ball towards the opponent goal. Hence, goalkeeping inhibits positioning behind the ball, kicking the ball, and dribbling the ball towards the target. Walking towards the ball and searching for the ball is not activated when playing as a goalkeeper. The goalkeeper stands still until it reacts on the ball. Balls close to the robot let it react immediately. It

uses the angle to the ball to determine the type of motion (diving left/right or bending forward). To achieve fast reaction on an approaching ball, the visual perception supplies the difference of the ball position between the last two images. The magnitude of this vector is interpreted as approaching speed. The goalkeeper does not react on small speeds. The type of the goalkeeper motion is determined by the intersection point of the moving ball direction and the goal line. At kick-off, the goalkeeper is placed in the goal. After a diving motion, it gets up and repositions itself in the goal while facing the opponent goal.

### C. Team Behaviors

The importance of team behaviors is still low in the Humanoid League, as only two players per team have competed so far. In Bremen 2006, most teams assigned one player to keep the goal clear and used the other player as field player. In our team, the players share perceptions via wireless communication. The ball perceptions communicated by other players are used for search. For the soccer play with two field players, we implemented simple but effective role negotiation between the players. As soon as one of our players has control of the ball, the other player goes to a defensive position between the ball and the own goal. A player takes control of the ball, if it is close to the ball and perceived it with high confidence. It loses control, if the ball gets far away or has low confidence. The thresholds for taking and losing control implement hysteresis to prevent oscillations of the control state.

## VI. RESULTS

Our robots performed well at RoboCup 2006, where 21 teams from eleven countries competed in the Humanoid League. In the 2 vs. 2 soccer round robin, they played 2 games and scored 12:0 goals. In the quarter final, they won 6:1 against team RO-PE from Singapore. They met the German-Japanese team Darmstadt Dribblers and Hajime in the semi-final. Our robots won 6:2. The final game was between our robots and Team Osaka, as in 2005. Our robots played well in the first half and scored a lead of 4:0. Fig. 12(a) shows one of the shots. After a goal directly from kick-off, the score at halftime was 4:1. Due to hardware problems of our robots, Team Osaka was able to reach a draw of 4:4 after regular playing time. As we already had taken the available two substitutions, we needed to continue playing with impaired robots in the extra time. The final score was 9:5 for Team Osaka.

Our KidSize robots also kicked penalties very reliably. In the Penalty Kick competition they scored in 31 of 34 attempts. In the KidSize Penalty Kick final (Fig. 12(b)) our robots won 8:7 against Team Osaka. In the technical challenge, our KidSize robot Gerd was one of the two robots able to walk across the rough terrain. Our KidSize robots also scored in the passing challenge. Our TeenSize robot Robotinho used a simplified version of the KidSize behaviors. It also reached the final of its Penalty Kick competition.



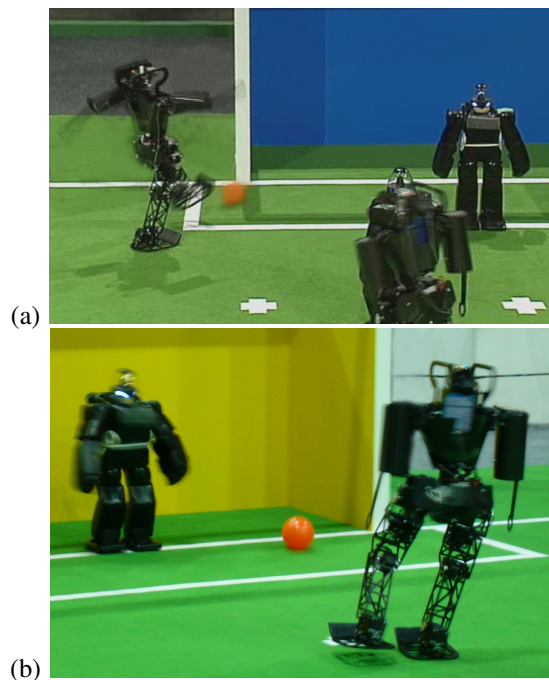


Fig. 12. RoboCup 2006: (a) 2 vs. 2 Soccer final NimbRo vs. Team Osaka. (b) KidSize Penalty Kick final NimbRo vs. Team Osaka.

In the overall Best Humanoid ranking, our KidSize robots came in second, next only to the titleholder, Team Osaka. Videos showing the performance of our robots at RoboCup 2006 can be found at <http://www.NimbRo.net>.

## VII. CONCLUSIONS

This paper described the design of the behavior control software for our 2006 KidSize robots, which successfully took part as team NimbRo at the RoboCup 2006 competitions. We implemented the control software in a framework that supports a hierarchy of reactive behaviors. A kinematic interface for body parts made it possible to abstract from individual joints when implementing basic skills like omnidirectional walking. These basic skills made it possible to abstract from body parts when implementing more complex soccer behaviors. At this player level, our humanoid robots are very similar to wheeled or four-legged soccer robots. Finally, at the team level, the players are coordinated through role negotiation.

Playing soccer with humanoid robots is a complex task, and the development has only started. So far, there has been significant progress in the Humanoid League, which moved in its few years from remotely controlled robots to soccer games with fully autonomous humanoids. Indeed, the Humanoid League is currently the most dynamic RoboCupSoccer league. We expect to see the rapid progress continue as more teams join the league. Many research issues, however, must be resolved before the humanoid robots reach the level of play shown in other RoboCupSoccer leagues. For example, the humanoid robots must maintain their balance, even when disturbed. Currently, we are working on postural reflexes, which should minimize the number of falls [12].

In the next years, the speed of walking must be increased significantly. We work on automatic gait optimization to increase both speed and stability. At higher speeds, running will become necessary. We recently started to explore this direction. The visual perception of the soccer world must become more robust against changes in lighting and other interferences. We continuously improve our computer vision software to make it more reliable.

The 2006 competition has shown that most teams were able to kick penalties, but that soccer games are much richer and more interesting. In the team leader meeting after the competition, the majority voted for abandoning penalty kick as a separate competition. Instead, the KidSize teams will focus on soccer games. Unfortunately, most teams do not feel ready to increase the number of players to more than two per team. This limits the possibilities for team play. As the basic skills of the humanoid soccer robots improve every year, teams will be able to focus on the more complex soccer behaviors and on team play. This will make structured behavior engineering a key factor for success.

## ACKNOWLEDGEMENTS

Funding for the project is provided by Deutsche Forschungsgemeinschaft (DFG) under grant BE 2556/2-1 and by Albert-Ludwigs-Universität Freiburg.

## REFERENCES

- [1] Sven Behnke. Playing soccer with humanoid robots. *KI – Zeitschrift Künstliche Intelligenz*, 3/06:51–56, 2006.
- [2] Sven Behnke. RoboCupSoccer Humanoid League rules and setup for the 2006 competition in Bremen, Germany: <http://www.humanoidsoccer.org/rules.html>.
- [3] Herbert Jaeger and Thomas Christaller. Dual dynamics: Designing behavior systems for autonomous robots. *Artificial Life and Robotics*, 2(3):108–112, 1998.
- [4] Hans Utz, Gerhard Kraetzschmar, Gerd Mayer, and Günther Palm. Hierarchical behavior organization. In *Proceedings of the 2005 International Conference on Intelligent Robots and Systems (IROS 2005)*, pages 2598–2605, Edmonton, Canada, 2005. IEEE/RSJ.
- [5] Martin Löttsch, Joscha Bach, Hans-Dieter Burkhard, and Matthias Jünger. Designing agent behavior with the extensible agent behavior specification language XABSL. In *RoboCup 2003: Robot Soccer World Cup VII*, volume 3020 of *LNCIS*, pages 114–124. Springer, 2004.
- [6] Tim Laue and Thomas Röfer. A behavior architecture for autonomous mobile robots based on potential fields. In *RoboCup 2004: Robot World Cup VIII*, volume 3276 of *LNCIS*, pages 122–133. Springer, 2005.
- [7] Sven Behnke and Raul Rojas. A hierarchy of reactive behaviors handles complexity. In *Balancing Reactivity and Social Deliberation in Multi-Agent Systems*, volume 2103 of *LNAI*, pages 125–136. Springer, 2001.
- [8] Scott Lenser, James Bruce, and Manuela Veloso. A modular hierarchical behavior-based architecture. In *RoboCup 2001: Robot Soccer World Cup V*, volume 2377 of *LNCIS*, pages 423–428. Springer, 2002.
- [9] Russel Smith. Open Dynamics Engine. <http://opende.sourceforge.net>.
- [10] Sven Behnke. Online trajectory generation for omnidirectional biped walking. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA'06)*, Orlando, Florida, pages 1597–1603, 2006.
- [11] Jörg Stückler, Johannes Schwenk, and Sven Behnke. Getting back on two feet: Reliable standing-up routines for a humanoid robot. In *Proc. of The 9th International Conference on Intelligent Autonomous Systems (IAS-9)*, Tokyo, Japan, pages 676–685, 2006.
- [12] Reimund Renner and Sven Behnke. Instability detection and fall avoidance for a humanoid using attitude sensors and reflexes. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2006)*, Beijing, China, pages 2967–2973, 2006.