THE CHALLENGE OF MOTION PLANNING FOR SOCCER PLAYING HUMANOID ROBOTS

Stefano Carpin, Marcelo Kallman, Enrico Pagello

Preprint

Abstract

Motion planning for humanoids faces several challenging issues: high dimensionality of the configuration space, necessity to address balance constraints in single and double support mode, higher levels of planning for coordination of different skills, etc. While the above challenges hold for any humanoid robot, the soccer scenario adds difficulties rarely addressed in humanoid motion planning research, as for example: dynamic environments with active opponents, the requirement to perform short- and long-term plans for performing soccer-relevant actions, and the necessity to plan movements purposely terminating with a collision with the ball. These aspects open a completely new scenario for researchers. This paper surveys state of the art research in motion planning for humanoid robots with focus on outlining connections, differences, and identifying the key aspects that ought to be addressed when developing effective humanoid soccer players.

1 Introduction

The essence of the RoboCup vision can be paraphrased as the development of humanoid robots with human-like physical and cognitive capabilities. The chosen deadline is the year 2050. Soccer playing requires multiple cognitive skills, like team strategy, online learning, and real time sensor processing. On top of that, by its very own nature, soccer involves strong physical abilities. If the ambitious vision is to ever be achieved, humanoid robots with skills far beyond any currently used robots need to be developed. While to the outside observer these goals may seem unreachable in the given time frame, those who witnessed the tremendous developments in the first ten years of RoboCup competitions have probably different expectations. RoboCup early days were characterized by teams composed by differential drive robots playing almost static games in a wall surrounded arena. Nowadays, omnidirectional vehicles participate in highly competitive games on playing fields that resemble more and more a real soccer pitch[1]. In parallel, competitions with humanoid robots have started since 2002. However, even the novice will notice that while certain technologies or algorithms can be seamlessly moved or adapted from wheeled platforms to humanoid robots, mobility requires to enter a completely different domain.

This paper focuses on this latter aspect. We provide a survey of some results available for humanoid motion planning, with a specific emphasis on problems, or advantages, distinguishing the *soccer* scenario. The following characteristics are dominant for soccer-playing humanoids:

- Soccer is a fast game. It follows that motion plans have to strive for the generation of speedy gaits. Methods exploiting the assumption of slowly evolving statically stable postures are doomed to be on the losing side. Furthermore, motion planning will typically be divided in two phases: (1) an off-line phase for computing parameterized motions (e.g. gaits), and (2) an on-line reactive phase for selecting and coordinating plans computed off-line.
- Soccer is played on a plain field where the only obstacles are other robots (opponents or team mates), the ball, and the goals. Simplified (primitive-like) geometries are enough to describe the shape of

[1] during RoboCup 2007 a game between humans and the world champion middle size team took place. The robot team displayed unexpectedly good defensive behaviors against human players.

the obstacles. However most of the time their location is not precisely known, or changes dynamically. Elaborate approaches for determining optimal strategies for overcoming obstacles are therefore practically pointless in this scenario.

- Soccer requires unique tasks to be solved. For instance defending from an adversarial shot may require a motion purposefully terminating with a fall on the ground. Robots acting as goalies need to quickly recover from these situations in order to continue their games. The majority of humanoids motion planning methods ignore the problem of diving to the floor and rapidly recovering the upright position.
- Robots need to kick the ball. Most of motion planning research strives for the generation of motions where no impulsive interaction between the robot and the environment occurs. Ball kicking violates this assumption.
- Robots will ideally also need to move while controlling the ball, dribble, and pass the ball to other team members. Planning a motion that constantly corrects the desired ball trajectory with small kicks is certainly challenging, as is the problem of planning the needed motion to stop an incoming ball passed from a team member. These problems have not yet been addressed by researchers.

This paper is organized as follows. Section 2 formally defines the motion planning problem. Stability, one of the major issues in humanoid locomotion, is introduced and discussed in section 3. Next, section 4 outlines existing approaches to humanoid motion planning problems. Section 5 surveys currently used methods in the RoboCup competition. Finally, conclusions are offered in section 6.

2 Motion Planning for Humanoids: Problem Formulation

Early results on robot motion planning outlined that the problem suffers from the so-called *curse of dimensionality*, i.e. under the widely accepted conjecture that $P \neq NP$, time complexity is exponential in the number of degrees of freedom (d.o.f.) [1, 2]. Even if robots participating in RoboCup competitions are often simpler than sophisticated commercial platforms like the famous Qrio or Asimo, their number of d.o.f. is nevertheless high. As frame of reference, during the 2006 RoboCup competition, the number of d.o.f. for the robots involved in the Humanoid kid size league varied between 17[3] and 28[4]. Figure 1 illustrates a typical setup for the joints controlling the legs of a humanoid robot, with 6 d.o.f. per leg.

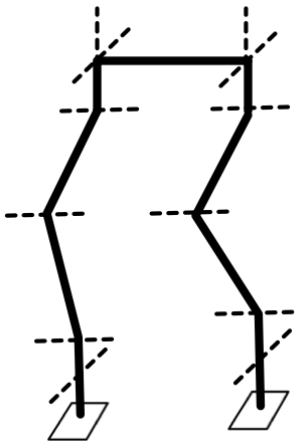

Figure 1: A typical mechanical setup for the legs of a humanoid robot used in the RoboCup competition. Six degrees of freedom are used to move each leg: three in the hip, one on the knee and two for the ankle (dashed lines in the figure show the rotational axis of the various joints).

For solving traditional motion planning problems, the elevated number of d.o.f. calls for the use of randomized algorithms[5], since combinatorial techniques become too slow. In its basic form the path planning problem is traditionally formulated as follows. Let $\mathcal{C}$ be the configuration space induced by the robot's d.o.f., and let $\mathcal{C}_{free}$ and $\mathcal{C}_{obs}$ be a partition of $\mathcal{C}$ into the spaces of free and obstacle configurations. Given $q_s, q_g \in \mathcal{C}_{free}$, determine a continuous function $f : [0, T] \rightarrow \mathcal{C}_{free}$ such that $q_s = f(0)$ and $q_g = f(T)$.

However, the legged morphology of humanoid robots poses critical challenges to the already arduous path planning problem. When the robot stands on two feet it forms a closed chain with the ground (double-support mode), while when it balances on a single foot (single-support mode) it does not form a chain anymore (but rather a tree). From a theoretical point of view this means that different manifolds characterize these two situations. In practice, randomized planning algorithms have often to be modified in order to address both support modes and this issue leads to the concept of planning considering motion primitives.

Another particular problem is that configurations in $\mathcal{C}_{obs}$ arise not only due to self-collisions and collisions with obstacles. Humanoid robots have to move according to plans that not only avoid collisions, but also keep away from *unstable* configurations. Non-stable configurations will therefore also belong to $\mathcal{C}_{obs}$. While static stability can be enforced simply looking into the kinematic state of individual poses of the robot, dynamic stability involves more parameters and other issues such as friction.

Planning full-body motions maintaining stability in a changing support mode constitutes the main characteristics of the typical humanoid motion planning problem. The most common approaches for addressing these challenges are presented below.

3 Stability

One manifest difference between wheeled mobile robots and humanoid robots stems from the concept of stability. Static and dynamic stability are discussed in this section.

3.1 Static Stability

Static stability has to be considered when no torque[2] is provided by robot's actuators. Basic principles of mechanics establish that balance is obtained when the sum of acting forces and moments is zero. In case of static stability, the only acting force is gravity. It is well known that a robot posture is statically stable if the projection on the ground of the position of the center of mass falls inside the convex hull of the supporting points. Figure 2 shows a humanoid robot in a statically stable position. Note however that in some complex situations friction cones and torques have also to be considered for ensuring static equilibrium, as is the case for multi-limbed robots that can climb[6].

The necessity to avoid configurations that are not statically stable introduces further constraints on the space of configurations to be searched by the motion planner. More specifically, according to the notation used by Kuffner et al.[7], static stable paths are searched in the set $\mathcal{C}_{stable} \subset \mathcal{C}$, where $\mathcal{C}_{stable}$ is the set of statically stable configurations.

3.2 Dynamic stability

When robot actuators deliver torques, dynamic stability instead of static stability has to be considered. This concept builds upon the concept of *zero moment point* (ZMP), introduced by Vukobratović more than thirtyfive years ago (see[8] for a recent synopsis of this concept and subsequent developments). Dynamic balance is particularly relevant during the stage of *single support*, i.e. when the robot stands on a single foot. As outlined by Vukobratović[8], ZMP can be defined, or interpreted, in different ways. First, it can be defined as *that point on the ground at which the net moment of the inertial forces and the gravity forces has no components along the horizontal axes.* When ZMP is outside the support polygon (i.e. the convex hull of the points in contact with the ground), the robot tilts over by rotating around one edge of the supporting

[2] it is assumed that all actuators are associated with revolute joints, therefore they provide torques rather than forces. This assumption is consistent with state of the art humanoid platforms used in RoboCup.

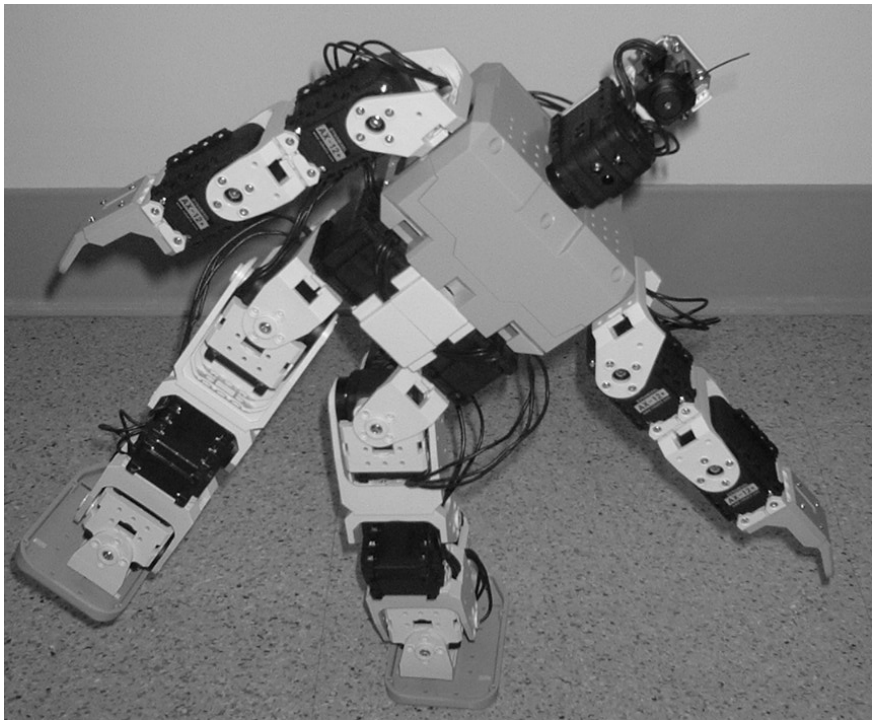

Figure 2: A statically stable position for a simulated humanoid robot. The projection of the center of mass on the ground falls inside the contact are of the left foot.

feet[3]. The ZMP condition is necessary and sufficient for dynamic stability. An alternative, but equivalent, interpretation was given by Arakawa and Fukuda[9]. They define ZMP as the point p where $T_x = 0$ and $T_y = 0$, where T_x and T_y represent the moments around the x and y axis generated by the reaction force R and reaction torque M, respectively. They then state that *when p exists within the domain of the support surface, the contact between the ground and the support is stable.* Figure 3 illustrates this interpretation of ZMP.

Figure 3 also hints that the closer the ZMP (point p) to the center of the support surface, the better. A ZMP close to the boundary in fact implies very little robustness to the unavoidable errors arising while moving and to possible unpredicted collisions.

3.3 Planning in single and double support modes

Planning the locomotion of legged humanoids involves planning synchronized single and double support phases. When the obstacles to be avoided are simple the locomotion can be performed with a parameterized gait, which is then simply steered to follow a desired planar trajectory without the need of handling individual leg trajectories. In cluttered environments however, locomotion can only occur with precise leg trajectory planning.

The problem of multi-limbed locomotion is similar to that of manipulation: in both cases the problem can be reduced to planning discrete changes in contact state and continuous motions between contact states. Locomotion can therefore be seen as the problem of manipulating the environment. The combination of the discrete and continuous problems leads to a much larger combined search space and specific strategies have to be developed in order to simplify the problem. For instance, the approach of footstep planning first searches for discrete feet placements to then link footsteps with leg motions. Leg motion can also be individually parameterized, avoiding expensive trajectory planning for each individual legs. This notion of controlling

[3]to be more precise, ZMP definition is sound only when it falls inside the support polygon. So, technically, ZMP *outside the support polygon* means ZMP undefined.

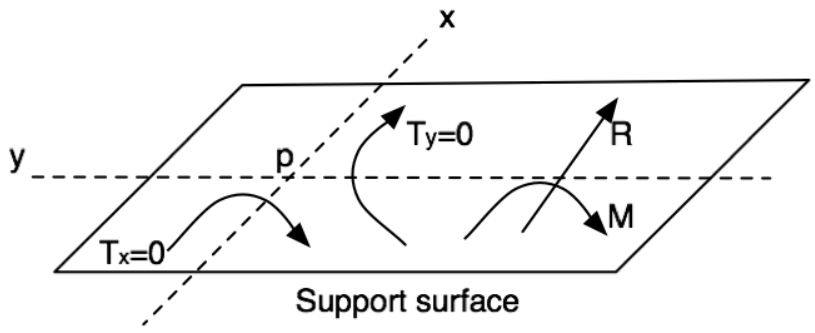

Figure 3: A graphical interpretation of the ZMP criterion. If the point p is inside the support surface, dynamic balance is obtained.

simpler parameterized and specialized motion is often called *planning with motion primitives*. In this case each primitive is a leg controller. Approaches for sequencing and synchronizing such primitives[6, 10, 11] are based on this framework.

4 Motion Planning for Humanoids: Solving Approaches

In this section we discuss the main solutions proposed in the literature addressing different aspects of the motion planning problem applied to humanoid robots.

They differ for the underlying hypothesis and also for the solution philosophy. In general, two major strategies are followed for the locomotion problem. The first consists in generating motion plans online, i.e. to repeatedly join leg configurations until the desired locomotion is achieved. Although more general and appealing, this method is usually very slow. The second builds upon the observation that *walking* is a periodic activity, and that complex gaits can be decomposed into simpler motion primitives that can be precomputed offline, either by an algorithm or by a human operator. Motion planning in this case means composing a sequence of primitives that achieve the given task. This approach is certainly more efficient, but is less general, because the resulting motion is limited by the chosen primitives set. A poorly chosen or limited primitives set could result in highly suboptimal motions or in the inability to solve the given motion query.

4.1 Planning leg motions for locomotion

An interesting solution belonging to the first category was proposed by Kuffner et al.[7]. Their approach builds upon the Rapidly-Exploring Random Trees (RRT) algorithm[12, 13], and reportedly has been the first successful attempt to provide a general motion planner for humanoid robots whose performance was confirmed on a physical robot. As every RRT based method, geometrical representations of the environment and of the robot are needed. The algorithm computes a trajectory between two given configurations that are both static stable. The authors introduce an additional subset of $\mathcal{C}_{free}$, namely $\mathcal{C}_{valid} = \mathcal{C}_{free} \cap \mathcal{C}_{stable}$. This is needed because a static stable configuration is not necessarily collision free, and the path has to satisfy both constraints. Algorithms 1 and 2 show the pseudocode for the RRT algorithm adapted to search paths in $\mathcal{C}_{valid}$. Validity has also to be assured by routine NEW_CONFIG (algorithm 2), which is responsible for computing a valid configuration toward the given random configuration q, when possible.

A critical point for this algorithm is the necessity to determine the *distance* between two configurations q and q' in order to find out the *nearest neighbor* (algorithm 2, line 4). Although any metric in the form

$$\rho(q, q') = \sum_{i=1}^{n} c_i ||q_i - q'_i||$$

Algorithm 1 Construction of the RRT

BUILD_RRT(q_{init}, K)
INPUT a starting configuration q_{init} and the number of iterations K
OUTPUT a RRT τ
τ.init(q_{init})
for $k = 1$ to K **do**
  $q_{rand} \leftarrow$ RANDOM_STABLE_CONFIG
  EXTEND(τ, q_{rand})
end for
RETURN τ

Algorithm 2 EXTENSION of the RRT

EXTEND(τ, q)
INPUT a tree τ and a random configuration q
RETURN *Trapped* or *Reached* or *Advanced*
$q_{near} \leftarrow$ NEAREST_NEIGHBOR(q, τ)
if NEW_CONFIG(q, q_{near}, q_{new}) **then**
  τ.add_vertex(q_{new})
  τ.add_edge(q_{near}, q_{new})
  if $q_{new} = q$ **then**
    RETURN *Reached*
  else
    RETURN *Advanced*
  end if
end if
RETURN *Trapped*

with $c_i \geq 0$ is valid, the choice of the coefficients c_i has great impact. The distance between two configurations should indicate the *cost-to-go* or the effort to move the humanoid from q to q'. At the moment no general criteria are known, and iterated heuristic attempts are rather used. As acknowledged by the authors, this approach still needs improvements in order to become a viable alternative for robot soccer. Computation times ranging from 30 to 600 seconds (on state of the art machines in 2002) also indicate that special care has to be taken when applying it. A way to reduce the time is to decrease the dimension of the search space by omitting, for example, the torso and the arms. This strategy needs however to include the inertial effects of the excluded parts.

4.2 Biped locomotion planning

The approach of planning footsteps for achieving locomotion among obstacles using RRTs has also been proposed by Kuffner and colleagues[14], and several extensions have been recently developed and successfully applied to humanoid robots[15, 16].

The basic idea of the main algorithm can be summarized as follows. Given a set of discrete feet placements, forward dynamic programming (guided by a greedy A* heuristic) is used to compute an optimal sequence of footsteps. Starting from an initial biped configuration q_{init}, a search tree of possible foot placements is constructed. A priority queue Q is used to store the nodes of the search tree which are possible to be expanded. At each iteration, the planner removes the lowest-cost (higher priority) node q_{min} from Q. Valid successor nodes of q_{min} are generated that correspond to all potential valid placement locations for the next footstep, and are inserted in Q. The process continues iteratively. When a successor node is generated that falls within a predefined goal region, the search terminates and a solution path back to the root node of the tree can be computed and returned. Pre-computed leg motions to connect the sequence of footsteps are used for achieving performances in the range from seconds to minutes.

A variation of this approach has been proposed by Kallmann et al.[11] where footsteps do not need to be

explicitly generated. Instead, RRT trees expanded in each support mode are analyzed for possible transition points and then the optimal sequence of transition points is obtained by forward dynamic programming. The advantage of this approach is that leg motions are planned at the configuration-space level and therefore solutions composed of complex foot placements and leg motions can be found. The drawback is that more computation is employed, often in the order of minutes. Examples of obtained results for a simple biped structure are shown in Figure 4 and Algorithm 3 summarizes the approach. In practice the performance of the algorithm is dramatically improved with the inclusion of: (a) search heuristics for culling unnecessary search branches (of the dynamic programming component) and (b) Inverse Kinematics to perfectly adjust foot positions to the environment in order to promote the discovery of configurations possibly serving as transition point to a new support mode[11].

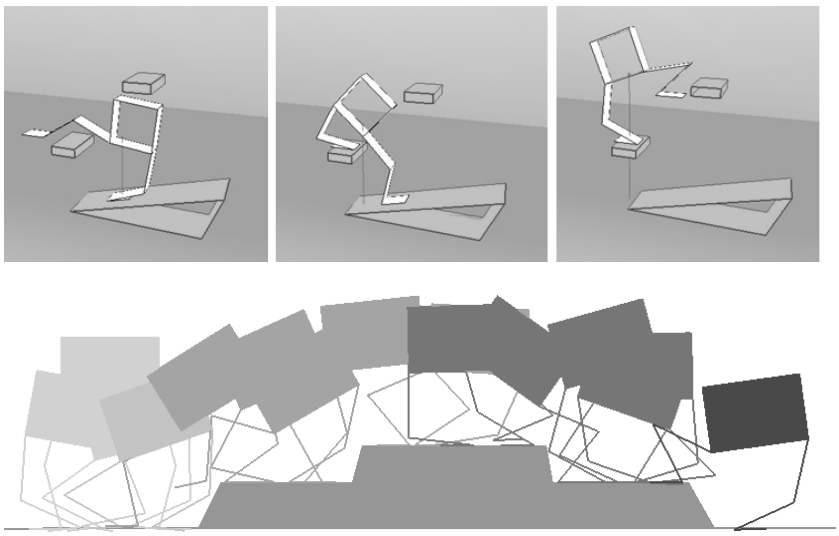

Figure 4: Complex biped locomotion can be achieved with a combination of RRT configuration space searches (for each support mode) and forward dynamic programming, achieving optimal sequencing of transitions.

Similar approaches to the footstep planning have also been proposed for multi-limbed robots that can climb[10, 6]. The basic idea is to divide the problem in two phases. First, a stance graph is computed where nodes represent valid stable configurations and edges represent that a transition is possible between the two end nodes of the edge. Finally a second graph is used (the transition graph) in order to search which transitions can be instantiated in terms of validity and in terms of getting closer to the goal. This two-stage search postpones validity checking to only when necessary and introduces significant speedups to the entire search procedure. This method has been successfully applied to the four-limbed climbing robot LEMUR[10], and a similar approach has been applied to a humanoid model[6]. In this later work, a formal notion of motion primitive is employed in order to optimize the generated motions connecting transition points.

4.3 Planning walking patterns

An alternative to online motion planning is the generation of so-called *offline patterns*, i.e. walking primitives that can be used as building blocks in order to elaborate complex motions. The method proposed by Huang et al.[17] has the desirable property of generating gaits such that the hip motion is optimized to keep the ZMP in the center of the support region. Therefore this approach aims to maximize *gait robustness*. The method analyzes the periodic nature of walking and, striving for motion speed, it relaxes some important assumptions, like the constraint that the foot stays always parallel to the ground, even while advancing. The mathematical model, not complicated but too long to be reported here, first computes the foot trajectory using a third-order spline interpolation. Next, the hip trajectory is also computed using a third-order spline.

Algorithm 3 Multi-Support RRT

```
1: LOCOMOTION_RRT(q_init, G)
2: INPUT starting configuration q_init and goal region G
3: OUTPUT solution locomotion reaching G
4: Q.insert(q_init)
5: while Q not empty do
6:   q_min ← Q.remove_mincost_element()
7:   τ ← NEW_EMPTY_RRT_IN_APPROPRIATE_SUPPORT_MODE(q_min)
8:   while Not (trapped during several iterations) do
9:     q_rand ← VALID_RANDOM_CONFIG_IN_CURRENT_SUPPORT_MODE
10:    EXTEND(τ, q_rand)
11:    if G reached then
12:      build and return solution path
13:    end if
14:  end while
15:  for each node q_τ ∈ τ do
16:    if q_τ allows a transition in support mode then
17:      Q.insert(q_τ)
18:    end if
19:  end for
20: end while
```

The use of third-order splines implies the desirable fact that second order derivatives are continuous, and then amenable of being appropriately tracked by the servos.

It is important to note that the Computer Animation domain offers several advanced solutions for planning parameterized and dynamically-stable walking gaits. There is a vast literature on the domain with impressive results for instance able to independently parameterize locomotion direction and torso direction[18], and to maintain balance during locomotion even when external forces are applied to a physically-based humanoid simulation[19]. The proposed approaches are very relevant to the humanoid soccer domain and extensions could be applied to real humanoids.

Another issue that has been recently addressed is the synchronization of two motion primitives being controlled concurrently, in particular when one of the primitives is considered to be a parameterized locomotion gait. Such problem arises when planning the whole-body motion of goalies for achieving locomotion concurrently synchronized with the needed arm motions in order to deflect an incoming ball. The work of Shapiro et al.[20] proposes a method that can lead to a straightforward solution to this problem: first, a suitable locomotion sequence is generated and parameterized by the time component, then a RRT search is performed in configuration-time space in order to plan a synchronized motion for the arms. Synchronization is achieved by ensuring that the sampling performed in the arm configuration-time space always results in a valid (in terms of both collisions and balance) whole-body posture when combining the arm motion with the locomotion sequence at the sampled point in time. Examples of obtained results applied to synchronizing leg and arm motions with locomotion are shown in Figure 5. In the long term, it is likely that synchronization between leg and arm motions will play an important role for humanoid robots playing soccer, in particular for goalies.

4.4 Gait optimization

Learning optimized gaits, a form of off-line walking patters, has been investigated by Hu et al.[21]. They distinguish two phases in the gait pattern: a *swing phase*, when the robot stands on a single foot, and *double support phase*, when both feet are on the ground. Constraints and performance criteria are then introduced in order to cast the gait search as a constrained optimization problem. Constraints are due to the geometry of the robot, limits on the forces and velocities, and dynamic stability. The performance (cost) function to be minimized is the sum of two terms, one measuring energy consumption, and the other the ZMP displacement. The evolutionary distributed algorithm (EDA) is used to search the minimum. EDA is similar to a genetic

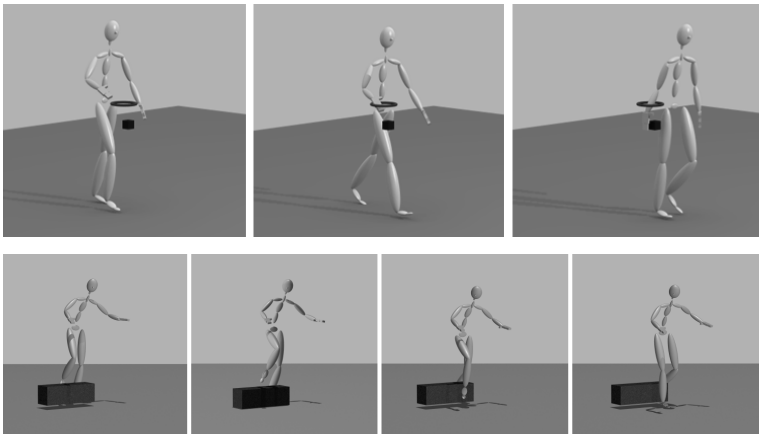

Figure 5: Examples of: arm motion planning for reaching a target in synchronization with locomotion (top), and leg motion planning for avoiding an obstacle during locomotion (bottom). Both examples were generated by the same configuration-time synchronization algorithm.

algorithm (GA) but instead of generating new solutions through mutation and recombination, a sampling process over the set of promising solutions is used. In practice, after having generated a set of initial solutions, new ones emerge by altering the best ones according to a multivariate Gaussian distribution. The authors show experimental results supporting the goodness of the EDA optimization step, that therefore appears to be an interesting possibility to generate offline optimized walking gaits. These gaits will then be selected to compose more complicated sequences.

4.5 Omnidirectional walking

The introduction of omnidirectional wheeled platforms has been a great breakthrough in the RoboCup middle and small size leagues. Starting from this standpoint Behnke has developed a procedure able to generate omnidirectional walking for bipedal robots[22]. Notably, this approach was introduced having the RoboCup competition in mind and was validated on the physical robots used during RoboCup competitions[23]. The algorithm accepts as input a vector consisting of three components, v_x,v_y and v_θ, that represent the desired lateral, forward and rotational speeds, respectively. The motion is broken down in the phases *shifting*, *shortening* and *moving*. In each stage simple trigonometric calculations yield the desired values for the 6 degrees of freedom controlling each leg. A valuable aspect of this research is indeed its simplicity, so that it can be implemented on PDAs. However, also according to the authors, gait generation is optimized for stability and generality, rather than for speed. Further investigations are also needed in order to take obstacles into account while planning these omnidirectional moves – an aspect neglected at the moment.

4.6 Getting up again

Falling down is typically seen as a negative event in humanoid robotics. In humanoid soccer, instead, certain tasks like defending from an adversarial shot hinge on the ability to promptly dive in the appropriate direction. As a consequence, the ability to quickly recover and get up again is also needed, because the game does not stop. Stückler et al. investigated this specific task, an aspect usually ignored in related literature[24]. Most teams are able to identify the event *tipped over* robot thanks to the on-board orientation sensors. The technique starts from the assumption that the robot can lie on the ground only in the supine or prone posture, due to its specific mechanical arrangement. The authors propose two distinct preprogrammed standing up procedures. The procedure to apply is easily determined upon inspection of the sensor indicating the sagittal tilt. In both cases a sequence of four motions manages to bring the robot back to the upright posture. The effectiveness of the two approaches was empirically verified during the RoboCup competition.

5 Current State of The Art in RoboCup

RoboCup 2007 in Atlanta saw the participation of 22 teams in the *KidSize* class and 7 teams in the *TeenSize* class. Class belonging is established by the value of the the H parameter defined as follows:

$$H = \min(H_{top}, 2.2H_{com})$$

where H_{top} is the height of the robot and H_{com} is the height of the center of mass[25]. Robots with H parameter between 30 and 60 centimeters belong to the KidSize class, while TeenSize robots have H between 80 and 160cm. Robots in the KidSize competition had a number of dofs between 16 and 26, while robots in the TeenSize class had a number of dofs between 12 and 30.
The *2 on 2* tournament for the KidSize class was won by the NimbRo team from the University of Freiburg[26]. The NimbRo team uses self developed robots with 24 dofs (8 in each leg, 3 in each arm, and two in the trunk). As formerly described, NimbRo robots are capable of omnidirectional walking. Special actions like getting back in the upright position and goalie motions to catch the ball are programmed off line as motion primitives and executed upon decisions taken by the behavior scheduler. In addition, NimbRo robots implement special features to deal with loss of control. A instability detector[27] is used to predict if the robot risks to fall. In such situation the robot tries to reach a squatting position, in order to lower the center of mass and gain stability. However, if the robot detects that falling is unavoidable, it relaxes its limbs to minimize the risk of damage. These motion capabilities appear to be not matched by other teams participating in the competition. Runner up in the 2 on 2 competition and winner of the best humanoid award was the Team Osaka. Team Osaka uses Robovie robots with 20 dofs[28]. Similarly to the NimbRo team, motion planning is performed in two ways. Special actions, like kicking, are programmed off-line by human operators. Real-time trajectories are instead computed on the fly based on desired intermediate goal positions that are reached via inverse kinematic.

6 Discussion and Conclusions

The gap between the existing techniques and the needs for planning humanoid motions for the situations encountered in soccer competitions is clearly large. While most of the existing approaches have to be optimized for coping with the encountered real-time and dynamic environment constraints, several motion planning problems have not even been addressed so far.

6.1 Improving the performance of existing methods

One direction that has to be further explored is the inclusion of higher level mechanisms for classifying on-line encountered situations and make use of parameterized pre-planned solutions that could rapidly be adapted to solve problems on-line. One approach is to use a database of previously-computed plans and quickly re-use these plans to the new encountered situations. Although such direction is challenging, some first results have been proposed in the context of planning humanoid reaching motions[29] and have showed the ability to double the performance of traditional motion planning. This approach transforms all existing (off-line) motion planning techniques as viable techniques to build databases of example solutions that could be re-used on-line. Learning to plan motions remains a challenging problem and at the same time the most promising direction for achieving human-like performance in humanoids.

6.2 Open problems

Several of the problems listed in the introduction were not yet addressed by the research community. We provide here an overview of them.

- Kicking the ball is unique feature of Robot soccer (see figure 6). In fact in main stream humanoid research, contacts between the robot and the environment, besides walking, happen typically only for grasping. There appears to be few results available about *planning a good kick*. Many teams define a *kicking behavior* that is often programmed offline and then triggered during the game when needed.

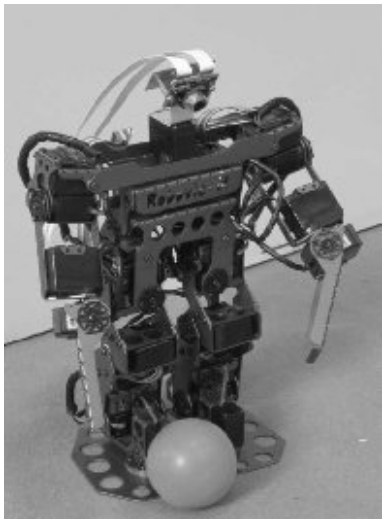

Figure 6: A robot approaching the ball in order to kick it to the goal.

An aspect that at the moment appears completely ignored is the impulsive collision with the ball. This is justified by the fact that the ball currently weights only a few grams and therefore the impact effects can be safely disregarded. However, on the way to develop robots that play soccer in conditions more and more resembling human soccer, this aspect needs to be addressed. A real soccer ball weights between 400 and 450 grams, and can reach remarkable speeds.

- Dribbling is typically a skill learned off-line that is specific to the position of the nearby adversary player(s) and their directions. The obvious approach here will build example situations and solutions, and reuse plans on-line using some classifier to drive the choice of the most appropriate one. The motion planning problem of generating (off-line) example solutions for dribbling problems remains challenging. If the motion of the adversary player(s) are known, the problem can be modeled as a manipulation or multi-legged locomotion problem. Even in this case, generalizations of example solutions to encountered new ones is a difficult issue without a guaranteed solution.

- Deflecting an incoming ball is a motion planning problem that goalies need to solve all the time during the games. In particular in real-scale goals, the problem remains a complex one and involves full-body planning and coordination. Although multi-primitive planning methods could be used for approaching the problem of blocking the incoming ball, two additional issues have to be addressed: (1) how to deal with the real-time requirements, and (2) how to continuously plan the placement of the goalie according to the position of the ball and, more importantly, the position of the adversary players. The most promising direction for these problems is again related to building databases of example situations in order to provide enough knowledge for real-time decisions and action plans. The problem can be scaled to higher levels in terms of planning strategies and also discovering the strategies of the adversary team, allowing the prediction of attacks and kicks.

6.3 Conclusions

In this paper we have outlined challenges, results and current approaches to motion planning for humanoid robots in the context of robot soccer. Extrapolating what happened in the small and middle size leagues, it is envisioned that the development of effective algorithms to move humanoid robots is one of the key aspect to develop winning robots.

An analysis of the team description papers produced for the 2006 and 2007 RoboCup competitions outlines that many teams still heavily rely on off-line programmed motions. This is understandable if one looks at the humanoid platforms used. Most of them rely on computational devices with modest computational

power. On top of that, most of this power is typically devoted to vision sensing and processing. With the unavoidable advent of faster and cheaper processing units to equip custom-built or general purpose humanoid robots, the execution of on-line algorithms could become feasible. It is reasonable to expect that the next generation of algorithms being used for robot soccer will be hybrid: a lot of the computation will still be done off-line in order to compute *canned* (example) motions but at the same time real-time motion planning will start to appear in terms of reusing selected examples and adapting them to the problem at hand, which has to be solved in real-time. This will lead to increasingly more precise motions and to the ability to control more complex (e.g. human-like) humanoids. However, it is also evident that currently proposed approaches need to be nevertheless improved in order to match the unusual characteristics distinguishing humanoid robot soccer. An aspect currently largely neglected, for example, is the generation of energy efficient motions and gaits.

In our opinion, humanoid soccer represents one of the most challenging fields for researchers in motion planning, and it still offers plenty of exciting unsolved issues.

References

[1] J. Reif, "Complexity of the mover's problem and generalization," in *Proceedings of the 20th IEEE Symposium on Foundations of Computer Science*, 1979, pp. 421–427.

[2] J. Canny, *The Complexity of Robot Motion Planning*. MIT Press, 1988.

[3] J. Baltes, S. McCann, and J. Anderson, "Humanoid robots: Abarendou and daodan," in *Team description paper for Robocup 2006*.

[4] N. Mayer, M. Ogino, R. da Silva Guerra, J. Broedecker, S. Fuke, H. Toyama, A. Watanabe, K. Masui, and M. Asada, "Jeap team description," in *Team description paper for Robocup 2006*.

[5] S. Carpin, "Randomized motion planning – a tutorial," *International Journal of Robotics and Automation*, vol. 21, no. 3, pp. 184–196, 2006.

[6] T. Bretl, "Motion planning of multi-limbed robots subject to equilibrium constraints: The free-climbing robot problem," *International Journal of Robotics Research*, vol. 25, no. 4, pp. 317–342, 2006.

[7] J. Kuffner, S. Kagaami, K. Nishiwaki, M. Inaba, and H. Inoue, "Dynamically-stable motion planning for humanoid robots," *Autonomous Robots*, vol. 12, no. 1, pp. 105–118, 2002.

[8] M. Vukobratović and B. Borovac, "Zero-moment point – thirty five years of its life," *International Journal of Humanoid Robotics*, vol. 1, no. 1, pp. 157–173, 2004.

[9] T. Arakawa and T. Fukuda, "Natural motion of biped locomotion using hierarchical trajectory generation method consisting of ga, ep, layers," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1997, pp. 211–216.

[10] K. H. T. Bretl and J.-C. Latombe, "Non-gaited humanoid locomotion planning," in *IEEE Intl. Conference on Humanoid Robots 2005*, 2005.

[11] M. Kallmann, R. Bargmann, and M. J. Matarić, "Planning the sequencing of movement primitives," in *Proceedings of the International Conference on Simulation of Adaptive Behavior (SAAB)*, Santa Monica, CA, July 2004, pp. 193–200.

[12] S. LaValle and J. Kufner, "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[13] S. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1999, pp. 473–479.

[14] J. Kuffner, K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba, and H. Inoue, "Online footstep planning for humanoid robots," in *IEEE Int'l Conf. on Robotics and Automation (ICRA'2003)*. IEEE, September 2003.

[15] J. Chestnutt, P. Michel, K. Nishiwaki, J. Kuffner, and S. Kagami, "An intelligent joystick for biped control," in *Proceedings of the IEEE International Conference on Robotics and Automation*, May 2006, pp. 860 – 865.

[16] P. Michel, J. Chestnutt, J. Kuffner, and T. Kanade, "Vision-guided humanoid footstep planning for dynamic environments," in *Proceedings of the IEEE-RAS Conference on Humanoid Robots (Humanoids'05)*, December 2005, pp. 13 – 18.

[17] Q. Huang, K. Yokoi, S. Kajita, K. Kaneko, H. Arai, N. Koyachi, and K. Tanie, "Planning walking patterns for a biped robot," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 280–289, 2001.

[18] A. Treuille, Y. Lee, and Z. Popović, "Near-optimal character animation with continuous control," in *Proceedings of ACM SIGGRAPH*. ACM Press, 2007.

[19] K. Yin, K. Loken, and M. van de Panne, "Simbicon: Simple biped locomotion control," in *Proceedings of ACM SIGGRAPH*, San Diego, CA, 2007.

[20] A. Shapiro, M. Kallmann, and P. Faloutsos, "Interactive motion correction and object manipulation," in *ACM SIGGRAPH Symposium on Interactive 3D graphics and Games (I3D)*, Seattle, April 30 - May 2 2007.

[21] L. Hu, C. Zhou, and Z. Sun, "Biped gait optimization using estimation of distribution algorithm," in *Proceedings of the IEEE-RAS Interntaional Conference on Humanoid Roots*, 2005, pp. 283–288.

[22] S. Behnke, "Online trajectory generation for omnidirectional biped walking," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006, pp. 1597–1603.

[23] S. Behnke, M. Schreiber, J. Stückler, H. Strasdat, and M. Bennewitz, "Nimbro kidsize 2006 team description," in *Team description paper for Robocup 2006*.

[24] J. Stückler, J. Schwenk, and S. Behnke, "Getting back on two feet: reliable standing-up routines for a humanoid robot," in *Proceedings of the 9th International Conference on Intelligent Autonomous Systems*, 2006.

[25] RoboCup technical Committee, "Robocup soccer humanoid league rules and setup for the 2007 competition," 2007.

[26] S. Behnke, M. Schreiber, J. Stückler, H. Strasdat, and K. Meier, "NimbRo KidSize 2007 team description," in *Team description papers for Robocup 2007*, 2007.

[27] R. Renner and S. Behnke, "Instability detection and fall avoidance for a humanoid using attitude sensors and reflexes," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 2967–2973.

[28] R. Matsumura, N. Shibatani, T. Imagawa, T. Maeda, T. Miyashita, T. Takahashi, Y. Akazawa, N. Yamato, and H. Ishiguro, "TeamOSAKA(kid size) team description paper," in *Team description papers for Robocup 2007*, 2007.

[29] X. Jiang and M. Kallmann, "Learning humanoid reaching tasks in dynamic environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, pp. 1148-1143