Planning foot placements for a humanoid robot: a problem of inverse kinematics*

Oussama Kanoun † Jean-Paul Laumond† Eiichi Yoshida ‡

April 12, 2010

Abstract

We present a novel approach to plan foot placements for a humanoid robot according to kinematic tasks. In this approach, the foot placements are determined by the continuous deformation of a robot motion including a locomotion phase according to the desired tasks. We propose to represent the motion by a virtual kinematic tree composed of a kinematic model of the robot and articulated foot placements. This representation allows us to formulate the motion deformation problem as a classical inverse kinematics problem on a kinematic tree. We first detail the basic scheme where the number of footsteps is given in advance and illustrate it with scenarios on the robot HRP-2. Then we propose a general criterion and an algorithm to adapt the number of footsteps progressively to the kinematic goal. The limits and possible extensions of this approach are discussed last.

1 Introduction

Humanoid robots that feature bipedal locomotion have been under considerable attention, as of late. Remarkable controllers [Kajita 03] have been successfully implemented to make smooth and sophisticated dynamical walking motion possible on such systems [Sakagami 02, Kaneko 04, Akachi 06]. Besides bipedal locomotion, these systems feature a large freedom of motion due to their high number of articulations. While robotic arms were composed of 6 articulated bodies, some of the humanoid robots now feature more than 40 degrees of freedom. To control the motion of such structures, special numerical frameworks have been proposed [Liégeois 77, Nakamura 91, Siciliano 91, Baerlocher 04, Khatib 04, Mansard 07, Kanoun 09], allowing real-time control based on desired dynamic or kinematic goals.

The problem we are focusing on is an algorithmic problem that interrogates both the bipedal locomotion and whole-body motion capabilities of the robot:

*A short version of this paper appears in the 9th IEEE-RAS International Conference on Humanoid Robots, Paris, December 2009.

†Oussama Kanoun and Jean-Paul Laumond are with JRL, LAAS-CNRS, University of Toulouse, 7 avenue du Colonel Roche 31077 Toulouse, France. {okanoun, jpl} @laas.fr

‡Eiichi Yoshida is with JRL, Research Institute of Intelligent Systems, National Institute of Advanced Industrial Science and Technology (AIST), 1-1-1 Umezono, Tsukuba, Ibaraki 305-8568 Japan. e.yoshida@aist.go.jp

where should the robot place itself in order to accomplish an arbitrary kinematic goal?

All rigorous answers we found in available works are based on probabilistic search algorithms. In this approach, the parameters of the problem are found by random exploration of the entire control space. A first application for humanoid robots was shown by [Kuffner 02] to build dynamically stable joint motion with a single foot displacement. [Escande 09] used the same class of algorithms to plan successive contact ports between the robot and its environment leading to accomplish a task. With the increase of average processing power, applying search algorithms on high-dimensional systems such as humanoid robot has become affordable. Nonetheless, we believe that these powerful methods should be saved for complex situations where a local strategy does not suffice.

Some works attempted a different use of probabilistic algorithms [Yoshida 07, Diankov 08]. The humanoid robot is viewed as a wheeled robot that must reach a goal position and orientation related to the kinematic tasks. The search algorithm finds a collision-free path along which an independent method plans stable stepping motions. The advantage of this approach is the reduction of the dimension of the problem down to three (two translations and one rotation for each node of parameters in the searched space). It needs, however, a reliable inference of the goal position and orientation from the task and robot's geometry. In a trial to compensate for this drawback, a two-time strategy has been proposed by [Yoshida 07]: first infer a gross goal position and orientation for the given task, then plan a path to it and finally determine if there is a need to fine-tune the position and orientation by a single step based on a task-specific strategy. The advantage of this method is to tackle the problem in progressive difficulty. Nonetheless, the chain of sub-problems suffers from the performance of the weakest link which is the final heuristical strategy.

In the approach we propose hereby, we seek to determine the foot placements by the continuous deformation of the motion of the robot, including a locomotion phase, according to the desired tasks. We propose to represent the entire motion by a virtual kinematic tree composed of a kinematic model of the robot and articulated foot placements. This representation will allows us to formulate the motion deformation problem as a classical inverse kinematics problem on a kinematic tree.

The contributions of this work is first in an original modeling of the footsteps planning problem as an inverse kinematics problem and second in a general criterion and algorithm to adapt the number of footsteps according to the tasks.

We begin by briefly recalling a framework for the kinematic control of highly-articulated structures (section 2). Then we show how inverse kinematic problems for footsteps planning are constructed (section 3) and applied on the humanoid model HRP-2 (section 4). We follow on with an algorithm that adapts the number of footsteps automatically according to a generic criterion (section 5) and we conclude with a discussion on the practical usage, limits and extensions of the proposed approach (sections 6-7).

2 Numerical inverse kinematics framework

In this section, we recall a general framework for the control of a highly-articulated systems with kinematic tasks and constraints.

2.1 Task and constraint definition

Let q be the joints configuration defining the posture of the robot, F a differentiable vector function of q. The equation

$$F(q) = 0 \tag{1}$$

defines an *equality task* on the robot. For a highly-articulated system, equation (1) often appears under-determined and is solved numerically following the ordinary differential equation

$$\frac{\partial F(q)}{\partial q}\,\dot{q} = -\lambda F(q) \tag{2}$$

where λ is a real positive constant. *Inequality tasks* are defined and solved in a similar fashion. Consider the inequality system

$$G(q) \leq 0 \tag{3}$$

where G is a differentiable vector function of q. This task can be solved by following the ordinary differential inequality

$$\frac{\partial G(q)}{\partial q}\,\dot{q} \leq -\lambda G(q) \tag{4}$$

Calling equation (1) and inequality (3) a *constraint* or a *task* depends on whether they are permanently required or desired.

2.2 Task resolution

The linear systems (2) and (4) are solved in the joint velocities $\dot{q}$. If the partial derivatives keep a full rank, the successive integrations $q = q + \alpha\dot{q}$ lead to a configuration q^* satisfying the task, otherwise the process is trapped in a local minimum. In either case, the convergence can be detected for an equality task (1) by evaluating the convex function

$$q \mapsto \|F(q)\|_2 \tag{5}$$

and for an inequality task (3) by evaluating the convex function

$$q \mapsto \sqrt{\sum_j max(0, g^j(q))^2} \tag{6}$$

where g^j is the j-th line in the inequality $G(q) \leq 0$.

Because of the numerous degrees of freedom in a humanoid robot, simultaneous tasks may usually be controlled at the same time. To anticipate a conflict between the tasks it is possible to assign control priorities such that a critical task could prevail over a task of less importance. For instance, in the case of a humanoid robot standing in quasi-static motion, the position of the center of mass must project within the support polygon at all times. An algorithm that solves a hierarchy of equations (2) and inequalities (4) was presented by [Kanoun 09]. For a set of prioritized tasks $\{\mathcal{T}\}$, a call to this algorithm will be denoted by

$$\dot{q} = solve(\{\mathcal{T}\}, q) \tag{7}$$

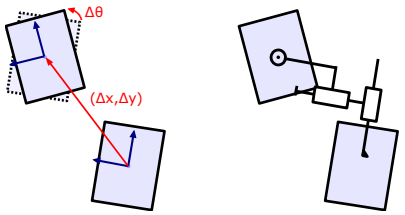


Figure 1: Two successive footprints are viewed as two virtual rigid bodies joined by two prismatic joints and one revolute joint around the vertical $\vec{z}$

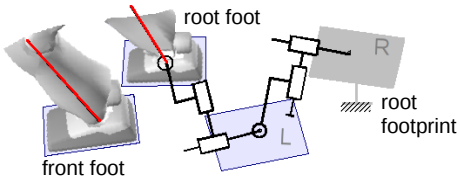


Figure 2: The virtual structure connecting a humanoid robot's articulated model to a sequence of footprints. The last foot to step is labeled *front foot*, the last but one step is made by the *root foot* and the initial support foot coincides with the *root footprint*

3 Construction of inverse kinematics problems for footsteps planning

The principle of this approach is the continuous optimization of foot placements with respect to the desired kinematic goals and this is achieved by solving inverse kinematics problem over a virtual articulated structure linking the kinematic frame of the robot to its successive foot placements.

3.1 Virtual articulated structure

Consider a robot that made $p \geq 1$ steps to complete a kinematic task. Consider two successive footprints as virtual rigid bodies connected by two prismatic joints and one revolute joint around the vertical $\vec{z}$ (figure 1). Define the configuration of the i-th relative footprint position by

$$Q_i = (\Delta x_i, \Delta y_i, \Delta\theta_i) \tag{8}$$

Define a virtual structure connecting a humanoid robot's articulated model to a sequence of footprints (see figure 2). The configuration of such a system can be written as

$$\tilde{q} = (Q_1, ..., Q_{p-1}, q) \tag{9}$$

Notice that last step Q_p does not appear in $\tilde{q}$. In fact, the last step implicitly results from the joint configuration q of the humanoid robot. What we have

constructed is a tree of articulated rigid bodies that captures both the state of the robot and that of the path it has followed to achieve a task.

Suppose that the task which motivated the stepping of the robot is a kinematic equality task $F(q) = 0$. If we consider the successive feet placements as extra unknowns of problem, the equation to solve becomes

$$F(\tilde{q}) = 0 \tag{10}$$

then equation (10) defines an inverse kinematics problem that can be solved numerically. The same can be formulated for inequality tasks which we would write $G(\tilde{q}) \leq 0$. The solution of these inverse kinematic problems is a configuration $\tilde{q}^*$ representing how the robot can walk in order to reach its kinematic goals. In the following we detail the kinematic constraints that must be taken into account.

3.2 Constraints on the humanoid model

For the humanoid model at the end of the virtual structure, kinematic constraints are needed to:

- enforce the joint limits
- enforce the static equilibrium
- avoid self collision
- keep the feet flat on the ground level

The last constraint is trivial and consists in setting the vertical position of the front foot (defined in figure 2) to that of the ground and authorizing its rotation around the vertical $\vec{z}$.

Joint limits are simply written as $q \leq q_{max}$ and $q \geq q_{min}$. From these expressions, linear differential inequalities like (4) are derived and treated as constraints in the inverse kinematics solver (7).

The static equilibrium for a humanoid robot standing on a horizontal ground is verified when the projection of the center of mass of the robot is inside its support polygon. Call C the projection of the center of mass on the ground. The constraint that keeps C inside the support polygon is composed of a variable number of the linear differential inequalities. One linear inequality constraint is added when an edge of the support polygon is directly facing C (see figure 3). The maximal velocity of C towards the polygon edges is controlled by the inequality

$$-\langle \frac{d\overrightarrow{PC}}{dt} | \vec{n} \rangle \leq \lambda(\langle \overrightarrow{PC} | \vec{n} \rangle - d) \tag{11}$$

where $\vec{n}$ is the normal vector to the considered edge and d is the minimal authorized distance.

Avoiding self-collision between two objects is equivalent to preventing the shortest distance between them from becoming null. Between a point A and a point B collision is avoided if we observe the constraint

$$\langle \overrightarrow{AB} | \vec{n} \rangle - d \geq 0 \tag{12}$$

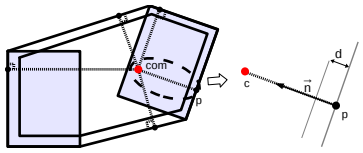


Figure 3: The projection C of the center of mass on the ground is controlled to remain strictly within the support polygon. This is achieved by linearly decreasing the velocity of C towards the edges of the support polygon

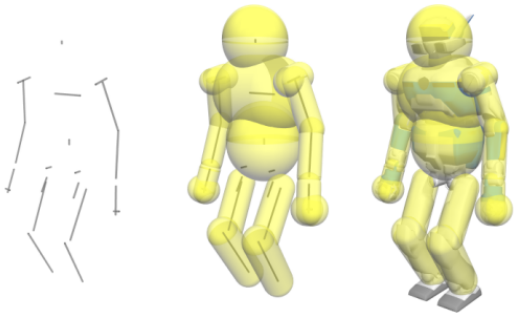


Figure 4: The rigid bodies in the robot are considered as line segments each with a forbidden radial zone covering the actual geometry. The advantage of such a model is the reduction of the dimension of collision avoidance constraints

where d is the minimal distance to be kept between points A and B, and $\vec{n} = \frac{\overrightarrow{AB}}{\|\overrightarrow{AB}\|}$. The differential system following from constraint (12) is thus

$$-\langle \frac{d\overrightarrow{AB}}{dt} | \vec{n} \rangle \leq \lambda(\langle \overrightarrow{AB} | \vec{n} \rangle - d) \tag{13}$$

A similar constraint was used by [Faverjon 87] on the the pairs of points realizing the shortest distance between a mobile robot and obstacles with strictly-convex shapes. A recent work by [Kanehiro 08] focused on the general case where the objects are non convex polyhedra. However, the generic method becomes costly when the geometrical model of the robot is made of a large number of polyhedra. Therefore, it was applied here to a simplified model (figure 4) where the bodies are defined by 3D line segments each with a forbidden radial zone around to cover the original geometry. The shortest distance between line segments is less expensive to compute than that between polyhedra. Besides, a maximum of three constraints like (13) are needed between each pair of segments in order to prevent the collision of the cylindrical zones.

3.3 Constraints on the foot placements

The footprints that compose the beginning of the virtual structure are considered two by two, in successive support polygons.

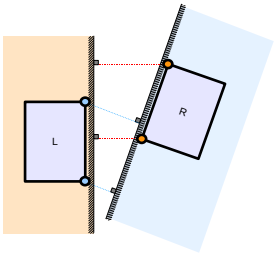


Figure 5: Each footprint lies in an associated half plane where the other foot cannot enter. This constraint is achieved by bounding the velocity of the interior corners of each foot towards the border of the forbidden half plane

In every support polygon, the relative position and the relative orientation of the feet are constrained in order to define an admissible stepping region. These constraints ensure that the involved footsteps are feasible in quasi-static motion. The shapes of the admissible stepping regions depend on the robots' capabilities. The simplest admissible region can be described by the constraints

$$\begin{aligned}\Delta x_{min} \leq \Delta x \leq \Delta x_{max}\\ \Delta y_{min} \leq \Delta y \leq \Delta y_{max}\\ \Delta \theta_{min} \leq \Delta \theta \leq \Delta \theta_{max}\end{aligned}$$

The opposite footprint in the support polygon is subject to the same (mirrored) bounds.

The above constraints are not adapted to prevent an overlapping between the foot placements in a support polygon. To address this problem, each footprint is further forbidden from entering a half-plane containing the other foot (figure 5). This constraint is equivalent to keeping both inner corners of a foot outside of the other foot's half plane. The corresponding ordinary differential inequalities are derived following example (12)-(13).

4 Illustration

We have applied our footstep planner to a variety of scenarios. The relative position and orientation of the feet were bounded as follows (left foot w.r.t right foot, the opposite case is taken symmetrical):

$$\begin{aligned}-0.22m < \Delta x < 0.22m\\ 0.07m < \Delta y < 0.25m\\ -0.1rad < \Delta \theta < \frac{\pi}{4}rad\end{aligned}$$

These bounds are small enough to guarantee quasi-statical stepping for the robot HRP-2. We did not judge useful for a first implementation to accurately estimate the maximal stepping region of our robot and we set these rather conservative constraints.

4.1 Reaching an object

In this scenario, the robot stands 2m away from the target ball. A simple obstacle is modeled with a disc region and the feet are constrained to avoid the corresponding area placed on the ground. To do this, the relative velocities between the center of the disc and its projection on each footprint were bounded following example (12)-(13). Figure 6 (Extension 1) shows the state of the kinematic structure at various intermediary steps from the iterated inverse kinematics problem. Initially, the virtual chain is folded down and all support polygons coincide with the start polygon. The chain unfolds continuously until the robot has satisfied its reaching task. For this test scenario, nine steps were needed to accomplish the task. This resulted in (28 degrees of freedom of HRP-2) + (3x9 degrees of freedom from the virtual chain of support polygons) = a total dimension of 55 for the state parameters. The computation time was 3.2 seconds on a 2.13GHz Intel$^{(R)}$ Core$^{(TM)}$2 CPU.

4.2 Picking up from the floor

This scenario shows best why a separation of the locomotion and the manipulation functions is limiting. The objective here is to pick up a small object lying on the ground between the feet of the robot. A classical ad-hoc method would detect that the object is within reach and that no locomotion is required, yet the robot would fail in grasping the target. By authorizing a few steps and using the new approach, the required stepping is found in a seamless way. To avoid stepping on the object before reaching for it, the footprints are constrained to avoid a virtual obstacle covering the object. Notice the generality of the approach since the only difference of input between this scenario and the previous one is in the coordinates of the target and obstacle. Figure 7 (Extension 2) shows a progression to the solution for this scenario which took 0.9s to solve. The actual motion where the robot steps over the planned footprints was calculated using numerical inverse kinematics with a dynamic stepping pattern generator described by [Kajita 03]. The coupling between those two frameworks was previously described by [Yoshida 06]. This motion was validated on the humanoid robot HRP-2 as shown in figure 8 (Extension 2).

4.3 Recovering a comfortable posture

In this third scenario, the robot has already performed a motion to look at the ball without making steps. The achieved posture is awkward and continuing to stare at the target in that shape is not very well looking. The robot may recover a comfortable posture by making a few steps. A similar scenario was presented by [Sreenivasa 09] with a heuristical method to derive the position of the required footsteps. We tackled the same problem in a generic way using our planner: we defined a comfortable posture q_{rest} for the robot and specified a desired posture task as

$$q - q_{rest} = 0 \tag{14}$$

q being the configuration of the humanoid robot. This task was applied under the constraint that the robot continued to look at the ball. This was expressed as

$$\overrightarrow{OG} \times \overrightarrow{OB} = \vec{0}$$

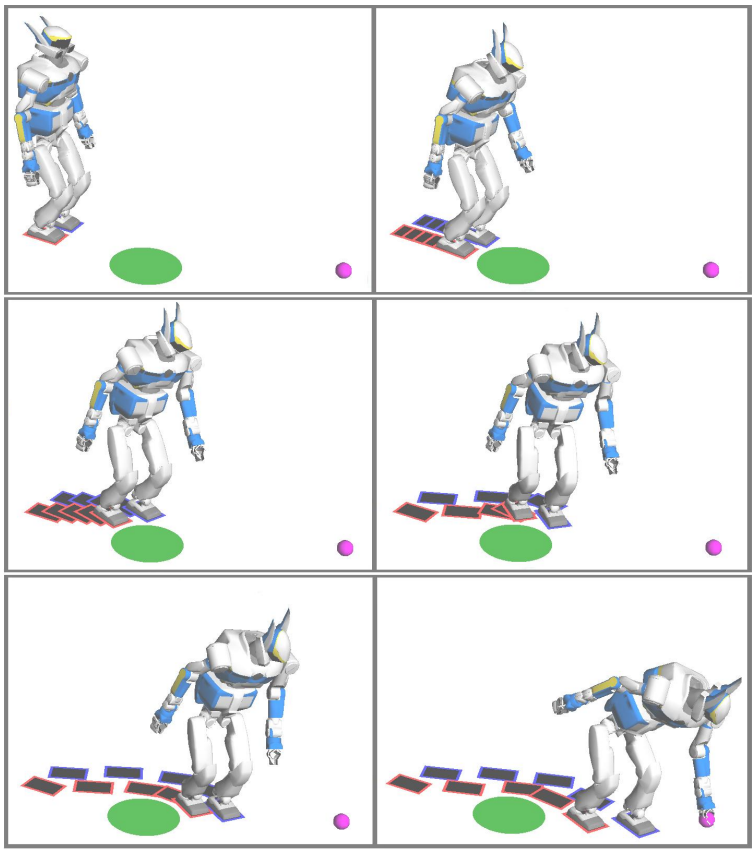


Figure 6: States of the full kinematic structure at different iterations of the inverse kinematics problem (see also Extension 1). The task was to grasp the ball without stepping on the oval region defining an obstacle. The last view shows the solution footprints retained for the actual robot locomotion.

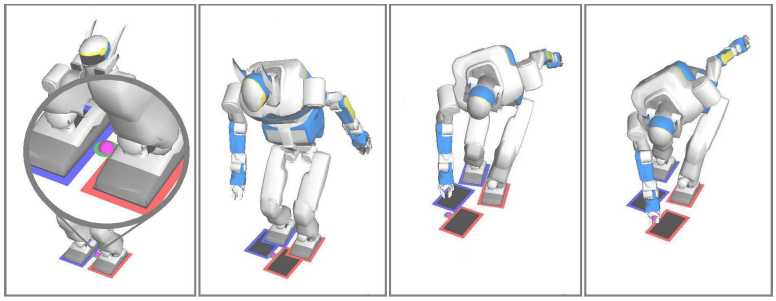


Figure 7: Planning footsteps to pick up an object on the floor. A virtual disc obstacle is added around the object to avoid stepping on it (see also Extension 2).

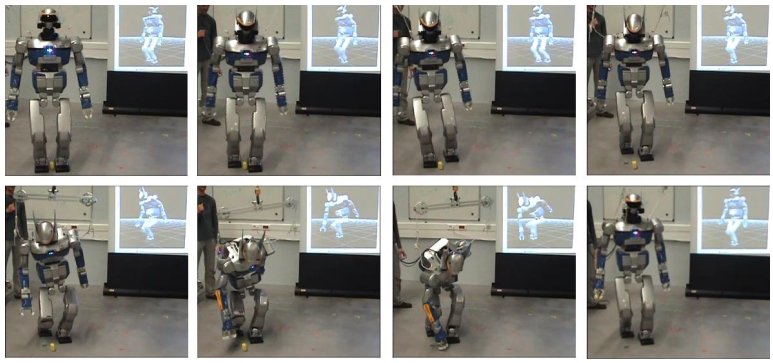


Figure 8: HRP-2 picking up an object lying between its feet. First a dynamic walk is planned over the support polygons produced by the local foot placement planner, then the whole body is driven by a reaching task while observing self-collision avoidance constraints (see also Extension 2).

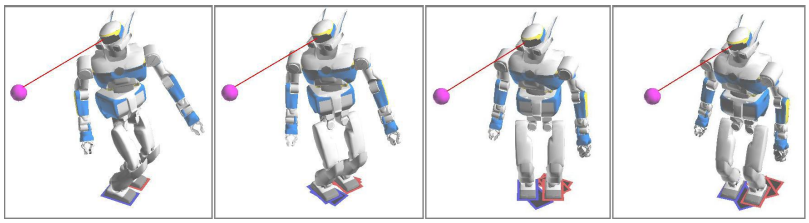


Figure 9: Planning footsteps to recover a comfortable posture in the middle of another task (here to look at the ball)

where O is a point on the optical axis, $\overrightarrow{OG}$ is a vector lying on the optical axis and B the position vector of the ball. Four steps were allowed to achieve a posture close enough to the initial configuration (see figure 9). The problem was solved in 0.3s.

5 Adapting the virtual model to the tasks

5.1 Adapting the number of footsteps

We described how to plan a sequence of footsteps based on an inverse kinematics approach where the number of footsteps was pre-determined. Now, we see how to adapt this number dynamically according to the desired kinematic goals. The proposed method is simple: at a given iteration of the inverse kinematics problem, a extra step is appended to the structure if the values of the tasks did not decrease *enough* during the last iteration. In other words, a step is added when the problem of inverse kinematics is about to become singular at one of its priority stages. To detect this event, the task value function (5) or (6) given in section 2.2 is watched for each task. For a given iteration p of the inverse kinematics problem, let $V_i^{(p)}$ denote the value of task T_i. If the condition

$$V_i^{(p-1)} - V_i^{(p)} > \epsilon_T \qquad \text{for } \epsilon_i > 0 \tag{15}$$

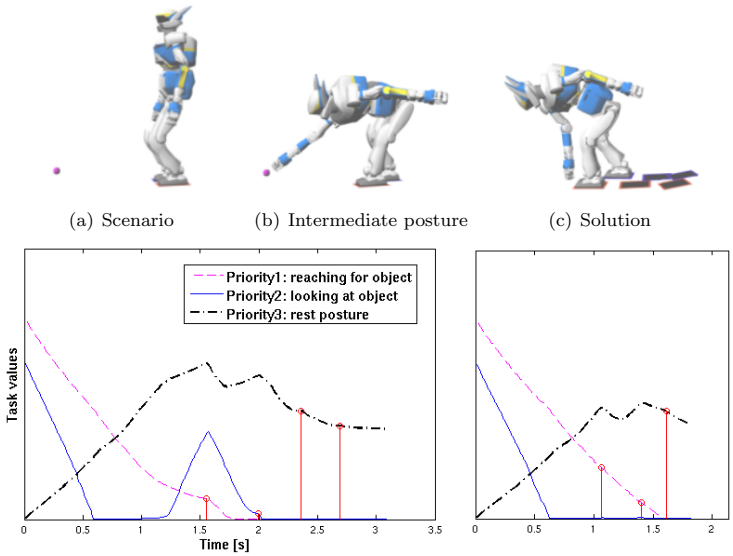
is not met, an extra step is added to the chain before iteration p+1. The constant ϵ_i represents the minimum value decrease that is expected per iteration for task T_i. In Algorithm 1 a footstep is added to the chain if all tasks T_i fail in achieving their respective minimum value decrease ϵ_i. Figure 11 illustrates the process of adding a footstep in the virtual structure.

According to the differential systems (2) and (4), the threshold ϵ_i for a task T_i is to be chosen less than $\lambda V_i^{(p)}$. Choosing too low a value will delay the addition of steps until the humanoid model is absolutely unable to move according to the tasks T_i (see figure 10(b)). On the opposite, high thresholds will make the algorithm too sensitive to performance inconsistencies between successive iterations and will trigger more step additions than needed. In practice, we choose average values for ϵ_i so that the steps are not added too late nor too soon (see figure 10(e)).

Suppose that the algorithm is started with a certain guess on the number of required footsteps. A way to avoid making more steps than necessary consists in forcing the footprints to remain at their initial position in a decreasing order of priority, such that the first footprints in the structure are the hardest to displace. At the end of the algorithm, the footprints that remained at their initial position and orientation are useless and can be discarded.

5.2 Adapting the start foot

The number of footsteps required to solve the problem may change depending on the choice of the first stepping foot. Which foot to start stepping with is a question that can be answered by calculating the solution for both choices, left and right. If the tasks are solved for both cases, the alternative with fewer steps may be preferred. To avoid an arbitrary decision in case the number of steps is identical for both choices, the start foot can be selected based on the



(a) Scenario (b) Intermediate posture (c) Solution

(d) Evolution with low thresholds (e) average thresholds

Figure 10: Evolution of task values for a simple scenario (a) using Algorithm 1. Each vertical line connects the time of a step addition to the curve of the task motivating the step. Because of the priority order, the steps are first added to reach, then to look at the object and finally to recover a rest posture. The choice of low threshold values delays the addition of steps (b) which affects the values of lower priority tasks (e.g bouncing for task 2 in (d)). Higher values make the algorithm react faster which tends to eliminate the bouncing effect and shorten the total computation time (e).

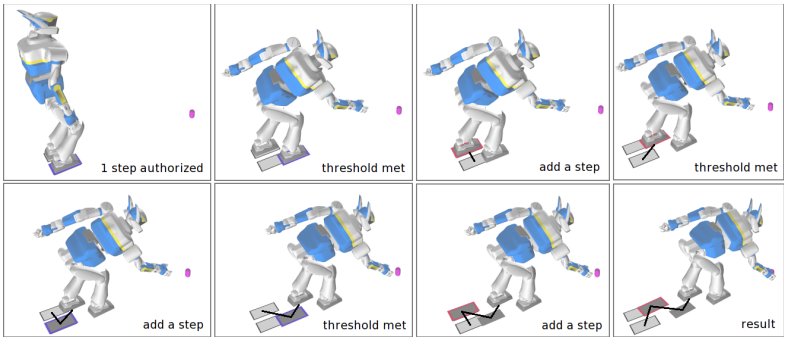


Figure 11: States of the virtual structure at different iterations of Algorithm 1. The structure starts with one step, the algorithm inserts an extra step at the root of the chain when the inverse kinematics problem tends to become singular.

Algorithm 1 Adaptive resolution of tasks $T_1 ... T_k$

1: **repeat**
2: **for** i from 1 to k **do**
3: Calculate value of task i: $V_i^{(0)}$
4: **end for**
5: Solve the prioritized differential systems (7) in $\dot{\tilde{q}}$
6: Integrate: $\tilde{q} \leftarrow \tilde{q} + \alpha\dot{\tilde{q}}$
7: **if** all tasks solved **then**
8: quit
9: **end if**
10: **for** i from 1 to k **do**
11: Calculate value of task i: $V_i^{(1)}$
12: Calculate value decrease $\Delta V_i = V_i^{(0)} - V_i^{(1)}$
13: **end for**
14: **if** all $\Delta V_i < \epsilon_i$ **then**
15: Add a footstep: $\tilde{q} \leftarrow (\Delta x, \Delta y, \Delta\theta, \tilde{q})$
16: Redo lines 2 to 13
17: **if** all $\Delta V_i < \epsilon_i$ **then**
18: quit
19: **end if**
20: **end if**
21: **until** all tasks solved

result of an extra posture task like in (14). Then, the choice of the start foot that produces the smallest residual task value $\|q_{robot} - q_{rest}\|$ is the one that is retained. More sophisticated criteria may naturally be used instead of this criterion intended as a default one.

6 Limits and extensions

6.1 Limits

The footstep planner that we present is based on a gradient descent method. Therefore, when the gradient is canceled and cannot be recovered by addition of extra footsteps, this method fails to solve the problem. When this happens, the virtual structure is trapped in a state that does not satisfy all desired tasks. Technically, this local minimum can be traced back to a priority stage $i > 1$ of the inverse kinematics solver, where the tasks desired at that stage have become incompatible with the set of controls authorized by the priority stage $i - 1$ even with an extra footstep. In addition to the choice of tasks and their order of priority, the initial configuration of the humanoid model has a big influence on the success of this local planner, as do the factors λ which scale the convergence rate in the differential systems (2) and (4). As seen above, the computation time for common scenarios is short enough to allow detection of failures and switching to another strategy in a timely way. These other strategies could for instance rely on probabilistic algorithms.

6.2 Extensions

The principle of this method is to continuously optimize the parameters defining the placement of the footprints according to given tasks. It supposes that the locomotion mode is known, for instance walking on a flat terrain as we chose here. To build analogous footprint planners for other locomotion modes such as going up/down stairs, stepping over an obstacle, running or jumping, it is necessary to redefine the admissible regions where the footprint placements are allowed to vary. For instance, to step over an obstacle, only a region beyond the obstacle should be allowed for the stepping foot. To climb stairs, the admissible regions would be bands on the surfaces of the stairs. The footprints from different locomotion modes can be connected to each other in order to optimize all footprints simultaneously.

7 Conclusion

We have presented an inverse kinematics formulation of the footsteps planning problem. The principle of the approach is the continuous optimization of footprint placements with respect to the kinematic goals. The virtual structure linking the humanoid robot model to the deformable chain of footprints was the key to avoid designing ad-hoc stepping strategies for each type of kinematic goal, a point that was best demonstrated through the task of picking an object between the feet of the robot. With this approach, the locomotion function and the manipulation function are both accounted for in a single planning stage. In a sense, the non-actuated degrees of freedom of the robot, i.e. the translation and rotation of the robot in the workspace, are now virtually actuated through the redundant kinematic chain of footprints.

Our planner may not be suitable for tasks requiring a long locomotion. For such tasks, one would prefer an algorithm that plans a walk path to a remote goal position and orientation (see [Kuffner 03, Yoshida 08]) or an interactively guided walk such as proposed by [Chestnutt 09]. In our method, we see a fine-tuner that takes over the end of the locomotion and reshapes the last few steps precisely according to the tasks. The performance that we obtained for scenarios requiring a small number of steps indicates that this method is affordable for online planning.

In the future, we may take inspiration from works such as [Barraquand 91, Barraquand 92, Kuffner 01] to couple this local planner with a probabilistic search algorithm and build a global footsteps planner free of local minima. Further directions of work include footsteps planning for time-dependent tasks.

8 Acknowledgments

This work has been partly supported by Grant-in-Aid for Scientific Research (B) 21300078 and JST-CNRS Strategic Japanese-French Cooperative Program "Robot motion planning and execution through online information structuring in real-world environment". This work has also been supported by the ANR Project Locanthrope and by the FUI Project Romeo.

References

[Akachi 06] K.Akachi, G.Miyamori, T.Kawasaki, M.Ishizaki, T.Kimura, T.Isozumi, K.Kaneko, F.Kanehiro, H.Hirukawa and H.Hasunuma "The development of a humanoid robot: HRP-3" Nippon Robotto Gakkai Gakujutsu Koenkai Yokoshu, Vol 24, (2006)

[Baerlocher 04] P.Baerlocher and R.Boulic "An Inverse Kinematic Architecture Enforcing an Arbitrary Number of Strict Priority Levels" The Visual Computer, Springer Verlag, 20(6), pp 402-417, (2004)

[Barraquand 91] J.Barraquand and J.C.Latombe "Robot Motion Planning: A Distributed Representation Approach" International Journal of Robotics Research, 10(6), pp 628-649, (1991)

[Barraquand 92] J.Barraquand, B.Langlois and J.C.Latombe "Numerical Potential Field Techniques for Robot Path Planning" IEEE Transactions on Systems, Man, and Cybernetics, 22(2), pp:224-241, (1992)

[Chestnutt 09] J.Chestnutt, K.Nishiwaki, J.Kuffner and S.Kagami"Interactive Control of Humanoid Navigation" IEEE International Conference on Intelligent Robots and Systems, pp 3519-3524, (2009)

[Diankov 08] R.Diankov, N.Ratliff, D.Ferguson, S.Srinivasa, and J.Kuffner "Bispace planning: Concurrent multi-space exploration" Robotics: Science and Systems Conference, (2008)

[Escande 09] A.Escande, A.Kheddar, S.Miossec and S.Garsault "Planning Support Contact-Points for Acyclic Motions and Experiments on HRP-2" Experimental Robotics, Springer Tracts in Advanced Robotics, Vol 54, pp 293-302, (2009)

[Faverjon 87] B.Faverjon and P.Tournassoud "A local based approach for path planning of manipulators with a high number of degrees of freedom" IEEE International Conference on Robotics and Automation, Vol 4, pp 1152- 1159, (1987)

[Fiacco 87] A.V.Fiacco and G.P.McCormick "Nonlinear programming: sequential unconstrained minimization techniques" Classics in Applied Mathematics, Society for Industrial Mathematics, (1987)

[Kajita 03] S.Kajita, F.Kanehiro, K.Kaneko, K.Fujiwara, K.Harada, K.Yokoi, and H.Hirukawa "Biped walking pattern generation by using preview control of zero-moment point" IEEE International Conference on Robotics and Automation, Vol 2, pp 1620-1626, (2003)

[Kanehiro 08] F.Kanehiro, F.Lamiraux, O.Kanoun, E.Yoshida and J-P.Laumond "A Local Collision Avoidance Method for Non-strictly Convex Polyhedra" Robotics: Science and Systems Conference, (2008)

[Kaneko 04] K.Kaneko, F.Kanehiro, S.Kajita, H.Hirukawa, T.Kawasaki, M.Hirata, K.Akachi and T.Isozumi "Humanoid Robot HRP-2" IEEE International Conference on Robotics and Automation, pp 1083-1090, (2004)

[Kanoun 09] O.Kanoun, F.Lamiraux, P-B.Wieber, F.Kanehiro, E.Yoshida and J-P.Laumond "Prioritizing linear equality and inequality systems: application to local motion planning for redundant robots" IEEE International Conference on Robotics and Automation, pp 2939-2944, (2009)

[Khatib 04] O.Khatib, L.Sentis, J.Park, J.Warren "Whole body dynamic behavior and control of human-like robots" International Journal of Humanoid Robotics, Vol 1, pp 29-43 (2004)

[Kuffner 01] J.Kuffner, K.Nishiwaki, S.Kagami, M.Inaba and H.Inoue "Footstep planning among obstacles for biped robots" IEEE International Conference on Intelligent Robots and Systems, (2001)

[Kuffner 02] J.Kuffner, S.Kagami, K.Nishiwaki, M.Inaba and H.Inoue "Dynamically-stable motion planning for humanoid robots" Autonomous Robots 12, No. 1, pp 105-118, (2002)

[Kuffner 03] J.Kuffner, S.Kagami, K.Nishiwaki, M.Inaba and H.Inoue "Online footstep planning for humanoid robots" IEEE International Conference on Robotics and Automation, Vol 1, pp 932–937, (2003)

[Liégeois 77] A.Liégeois "Automatic supervisory control of the configuration and behavior of multibody mechanisms," IEEE Transactions on Systems, Man and Cybernetics, 7, No. 12, pp 868-871, (1977)

[Mansard 07] N.Mansard and F.Chaumette "Task sequencing for sensor-based control" IEEE Transations on Robotics, 23(1), pp 60-72, (2007)

[Nakamura 91] Y.Nakamura "Advanced Robotics: Redundancy and Optimization" Addison-Wesley Longman Publishing, Boston (1991)

[Sakagami 02] Y.Sakagami, R.Watanabe, C.Aoyama, S.Matsunaga, N.Higaki, K.Fujimura, H.R.D.C. Ltd and J.Saitama "The intelligent ASIMO: System overview and integration" IEEE-RSJ International Conference on Intelligent Robots and Systems, Vol 3, (2002)

[Siciliano 91] B.Siciliano and J.-J.E.Slotine: "A general framework for managing multiple tasks in highly redundant robotic systems" IEEE International Conference on Robotics and Automation, pp 12111216, (1991)

[Sreenivasa 09] M.Sreenivasa, P.Soueres, J-P.Laumond and A.Berthoz "Steering a humanoid robot by its head." IEEE International Conference on Intelligent Robots and Systems, pp 4451-4456, (2009)

[Yoshida 06] E.Yoshida, O.Kanoun, C.Esteves and J-P.Laumond "Task-driven support polygon reshaping for humanoids" IEEE-RAS International Conference on Humanoid Robots, pp 208-213, (2006)

[Yoshida 07] E.Yoshida, A.Mallet, F.Lamiraux, O.Kanoun, O.Stasse, M.Poirier, P.F.Dominey, J-P.Laumond and K.Yokoi ""Give me the purple ball" he said to hrp-2 n. 14"IEEE-RAS International Conference on Humanoid Robots, pp 89-95, (2007)

[Yoshida 08] E.Yoshida, C.Esteves, I.R.Belousov, J-P.Laumond, T.Sakaguchi and K.Yokoi "Planning 3-D Collision-Free Dynamic Robotic Motion Through Iterative Reshaping" IEEE Transactions on Robotics, 24(5), pp 1186-1198, (2008)

Appendix A: Index to Multimedia Extensions

The multimedia extensions to this article are at: http://www.ijrr.org

Extension	Type	Description
1	Video	Scenario 1: reaching an object
2	Video	Scenario 2: picking up an object on the floor