2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Real-Time (Self)-Collision Avoidance Task on a HRP-2 Humanoid Robot

Olivier Stasse, Adrien Escande, Nicolas Mansard, Sylvain Miossec, Paul Evrard and Abderrahmane Kheddar

Abstract— **This paper proposes a real-time implementation of collision and self-collision avoidance for robots. On the basis of a new proximity distance computation method which ensures having continuous gradient, a new controller in the velocity domain is proposed. The gradient continuity encompasses no jump in the generated command. Included in a stack of tasks architecture, this controller has been implemented on the humanoid platform HRP-2 and experienced in a grasping task while walking and avoiding collisions with the environment and auto-collisions.**

Index Terms— **stack of tasks, proximity distance with continuous gradient, self-collision and obstacle collisions avoidance.**

I. Introduction

It is crucial for a robot to have, among others, the ability to (i) avoid undesirable collisions with both obstacles and own parts –for example undesirable auto-collisions for redundant robots such as humanoids– and (ii) detect desired collisions such as those necessary to perform contact-based tasks –for example haptic interaction with an environment or with own robot parts–. Collision detection and avoidance functions are needed either in simulation or in real implementation and can be used in off-line or on-line trajectory generation. Planning collision-free trajectories is an active area of research [1]. Open-loop planning is generally made off-line and can make use of simple binary collision detection algorithms whereas closed-loop trajectory generation is on-line, mostly embedded with the controllers and makes use of proximity distance collision to predict and prevent collisions [2]. In this paper we address the problem of on-line collision avoidance (including auto-collision avoidance) in the context of robotics and more particularly in humanoid robotics; although several closed-loop controllers have been proposed for various reactive task purpose full body motion, see for instance [3][4], surprisingly none of them guarantee, in an explicit way, non desirable collisions or auto-collisions. We propose an efficient algorithm to compute fast proximity distances that can be efficiently integrated in a low-level reactive control. We used strict-convexity bounding volumes close to polyhedral convex hulls in order to guarantee the continuity of the proximity distance gradient [5] and in the same time satisfying real-time control requirement. In [6] authors make use of cylinders and spheres to cover a redundant manipulator and compute in a efficient way proximity distance to be used as a secondary task in a kinematics task prioritization scheme and experimented on a 7dof redundant manipulator. Instability that may occur from discontinuity of the witness points has been addressed in configurations where the cylinders are nearly parallel. Our algorithm makes use of patches of spheres and toruses to cover in a more precise way the convex hull of robot's part and get rid of this problem. Our algorithm description and its technical implementation issues are presented in the first section.

The authors are with the AIST/CNRS Joint Japanese-French Robotics Laboratory, JRL, Tsukuba, Japan.
olivier.stasse, adrien.escande, nicolas.mansard, sylvain.miossec, paul.evrard@aist.go.jp, kheddar@ieee.org

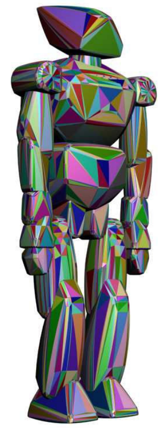

Fig. 1. STP-BV representation of HRP-2.

The very nature of reactive controllers is that they are local and not global such as in [7]. They are more suited to interactive task, and unstructured varying environments; they are also computationally lighter than planning. Our algorithm has been integrated as an additional task in the stack of the task sequencing architecture proposed in [8][9]. In humanoid robotics, a similar idea has been proposed in [10] where self-collision avoidance has been implemented on the simulator of ASIMO, their method makes use of an artificial force which changes the value of the desired posture in the gradient of a posture cost function projected in the null space of the main tasks; tasks are described in derivative of joints or Cartesian spaces (velocity). This virtual force penalizes a close proximity distance obtained from sphere swept lines [11] bounding volume of the humanoid. Their approach is very elegant and simple to implement, yet they did not consider stability problems that may occur from discontinuous witness points and their formalism would not allow marking auto-collision avoidance with higher priority. This is possible by our method. Other methods have been proposed based on convex hulls [12], or hybrid approachs [13]. Again, none of them consider the problem of discontinuity or include the constraint in a stack of controllers.

Our method has been ported and exemplified on an actual experiment involving the HRP-2 humanoid robot. The tasks consist in performing a ball-grasp guided by vision while at the same time walking stably and avoiding obstacles and auto-collisions. As a result of this experiment, open issues appear to be linked to task sequencing and prioritization and more generally the way to tackle tasks described through unilateral constraints... consisting in future work.

II. Proximity distance having continuous gradient

A. A new strictly convex bounding volume

Proximity distance and more generally collision avoidance have been widely studied, which results in numerous algorithms and schemes (the interested reader may refer to the recent, exhaustive and excellent review books [14][15]). However, little attention has been paid to the continuity properties of the proximity distance. It has been pointed out however that in singular cases, the gradient of this distance is not continuous, generating oscillations or misbehaviours in the control scheme [6]. For a complete discussion on the continuity problem as well as a method to regularize the proximity distance gradient, see [5]. The main idea is to build strictly convex hulls of the robot bodies in a way that can be seen as a slight blowing up of the usual convex hull. It is realized through patches of spheres and toruses. Such a volume is called Sphere-Torus-Patches Bounding Volume (STP-BV). Its advantages are (i) to ensure continuity of the gradient, and (ii) to accurately approximate the convex hull of the object (in typical cases we experienced, the volume increase from convex hull to STP-BV is 1–2%), while maintaining the number of collision pairs low (one volume per body) as well as the computation time for each of them.

One of the most interesting property of STP-BV main result presented is it is sufficient to have only one strictly convex body to have a continuous minimum distance between two witness points of two convex bodies. This means that the STP-BV construction is not mandatory for obstacles to hold the continuity. It is sufficient to build the patches only for the robot's bodies. The HRP-2's associated representation is depicted in figure 1. The computation of the gradient then from [5] of this distance is now detailed, as it is a key point to compute a control law to avoid self-collision.

B. Computing Proximity Distance's Gradients

Let us note δ the distance between two convex objects O_1 and O_2. The relative position between two objects is parameterized by the actuated joints of the robot noted q. Assuming that the witness points of the STP-BV of objects O_1 and O_2 are respectively $SP^1_{\min}$ and $SP^2_{\min}$ then the gradient of the distance can be written:

$$\frac{\partial \delta}{\partial q} = n_d^\top \left(\frac{\partial SP^1_{\min}}{\partial q}(q) - \frac{\partial SP^2_{\min}}{\partial q}(q) \right) \tag{1}$$

with n_d the normal unit vector derived from the features. Intuitively the gradient is orthogonal to the vector defined by the two witness points.

For a point P of fixed coordinates (x, y, z) in the local frame of an object O at the configuration q, the gradient has the following expression:

$$\frac{\partial P}{\partial q}(q) = xJ_1(q) + yJ_2(q) + zJ_3(q) + J_4 \tag{2}$$

obtained by deriving

$$\begin{aligned} P(q) &= R(q)(x, y, z)^\top + T(q) \\ &= xC_1(q) + yC_2(q) + zC_3(q) + T(q) \end{aligned} \tag{3}$$

where R is a rotation matrix, C_i its columns and T is the translation vector. The J_i are the gradient matrices of the C_i and T. This matrices can be analytically computed beforehand and are called hereafter *pregradient matrices*.

III. Stack of tasks

The stack of tasks is a structure that orders the set of tasks that are currently active. Only the tasks in the stack are taken into account in the control law. The task at the bottom level has priority over all the others, and the priority decreases as the stack level increases. The control law is computed from the tasks in the stack, in accordance with three rules:

- any new task added in the stack does not disturb the tasks already in the stack.
- the control law is continuous, even when a task is added or removed from the stack. The robot is controlled through the joint velocity $\mathbf{\dot{q}}$. A break of continuity would mean an infinite acceleration during a short period of time, which would imply that the control is not correctly applied.
- if possible, the additional constraints should be added to the control law, but without disturbing the tasks in the stack.

The control law is computed from the stack, using the redundancy formalism introduced in [16]. The additional constraints are added at the very top of the stack, which means that they are taken into account only if some degrees of freedom (DOF) remain free after applying the active tasks. This priority order may seem illogical, considering that the constraints are obstacles that the robot should avoid above all. However, the positioning task has priority since it is the task we want to see completed, despite the presence of the obstacles. The high-level controller is then used to ensure that the constraints are respected when it is obvious that the robot will violate them.

1) Ensuring the priority: Let $(\mathbf{e_1}, \mathbf{J_1})$ ... $(\mathbf{e_n}, \mathbf{J_n})$ be n tasks. The control law computed from these n tasks should ensure the priority, that is the task $\mathbf{e_i}$ should not disturb the task $\mathbf{e_j}$ if $i > j$. A recursive computation of the joint velocity is proposed in [16]:

$$\begin{cases} \mathbf{\dot{q}_0} = 0 \\ \mathbf{\dot{q}_i} = \mathbf{\dot{q}_{i-1}} + (\mathbf{J_i P^A_{i-1}})^+ (\mathbf{\dot{e}_i} - \mathbf{J_i \dot{q}_{i-1}}), \quad i = 1..n \end{cases} \tag{4}$$

where $\mathbf{P^A_i}$ is the projector onto the null-space of the augmented Jacobian $\mathbf{J^A_i} = (\mathbf{J_1}, \dots \mathbf{J_i})$ and $\widetilde{\mathbf{J_i}} = \mathbf{J_i P^A_{i-1}}$ is the limited Jacobian of the task i . The robot joint velocity realizing all the tasks in the stack is $\dot{\mathbf{q}} = \dot{\mathbf{q}}_\mathbf{n}$. The projector can be recursively computed by

$$\mathbf{P}^\mathbf{A}_\mathbf{i} = \mathbf{P}^\mathbf{A}_\mathbf{i-1} - (\mathbf{J_i}\mathbf{P}^\mathbf{A}_\mathbf{i-1})^+\mathbf{J_i}\mathbf{P}^\mathbf{A}_\mathbf{i-1} \tag{5}$$

2) Ensuring the continuity: From (4), the control law is obtained by imposing a reference velocity $\dot{\mathbf{e}}_\mathbf{i}$ for each task in the stack. Generally, a exponential decrease is required by imposing the first order differential equation $\dot{\mathbf{e}}_\mathbf{i} = -\lambda_i \mathbf{e}_\mathbf{i}$. However, this equation does not ensure the continuity of the robot velocity when the stack is changed. In [17], we proposed a solution to properly smooth the robot velocity at the transition, by imposing a specific second order equation:

$$\ddot{\mathbf{e}}_\mathbf{i} + (\lambda_i + \mu)\, \dot{\mathbf{e}}_\mathbf{i} + (\lambda_i \mu)\, \mathbf{e}_\mathbf{i} = 0 \tag{6}$$

where λ_i is the gain that tunes the convergence speed of task $\mathbf{e}_\mathbf{i}$, and μ sets the transition smoothness of the global control law. The control law is obtained by introducing (6) in (4):

$$\begin{cases} \dot{\mathbf{q}}_\mathbf{i} = \dot{\mathbf{q}}_\mathbf{i-1} + (\mathbf{J_i}\mathbf{P}^\mathbf{A}_\mathbf{i-1})^+(-\lambda_i \mathbf{e}_\mathbf{i} - \mathbf{J_i}\dot{\mathbf{q}}_\mathbf{i-1}) \\ \dot{\mathbf{q}} = \dot{\mathbf{q}}_\mathbf{n} + \mathbf{G}^\mathbf{A} e^{-\mu(t-\tau)} \left(\dot{\mathbf{e}}(\tau) + \Lambda \mathbf{e}(\tau)\right) \end{cases} \tag{7}$$

where τ is the time of the last modification of the stack, $\Lambda = \operatorname{diag}(\lambda_i)$ and $\mathbf{G}^\mathbf{A}$ is defined so that $\dot{\mathbf{q}}_\mathbf{n} = \mathbf{G}^\mathbf{A} \begin{bmatrix} \dot{\mathbf{e}}_\mathbf{1} \\ \vdots \\ \dot{\mathbf{e}}_\mathbf{n} \end{bmatrix}$ in (4).

3) Adding the secondary constraints: The constraints are added using the Gradient Projection Method [18], [2]. The constraints are described by a cost function V. The gradient $\mathbf{g}(\mathbf{q})$ of this cost function can be considered as an artificial force, pushing the robot away from the undesirable configurations. It is introduced as the last task of the stack. It has thus to be projected onto the null space of each task into the stack. Using (7), the complete control law is finally

$$\dot{\mathbf{q}} = \dot{\mathbf{q}}_\mathbf{n} + \mathbf{G}^\mathbf{A} e^{-\mu(t-\tau)} \left(\dot{\mathbf{e}}(\tau) + \Lambda \mathbf{e}(\tau)\right) - \kappa \mathbf{P}^\mathbf{A}_\mathbf{n}\mathbf{g} \tag{8}$$

The reader is invited to refer to [17], [19] for more details.

IV. The control law related to self-collision

A. Gradient Projection Method

We want now to find the cost function to be minimized in order to respect the constraint of self-collision avoidance. This problem is written

$$\min\ V(q),\ q \in \mathbb{R}^n \tag{9}$$

with n the number of DOFs of the robot. The classical iterative solution is then to move the robot along the gradient of the function:

$$\dot{\mathbf{q}} = -\kappa \mathbf{g}(\mathbf{q}) = \nabla_q^\top \tag{10}$$

The task to avoid self-collision is straightforward and inspired from the joint limit avoidance scheme proposed by Khatib [2].

Let us consider Φ a set of parameters to span the constraint space. The optimal force to satisfy this problem is then [20]:

$$\mathbf{g}_\Phi(\mathbf{q}) = \left(\frac{\partial \Phi}{\partial \mathbf{q}}\right) \nabla_\Phi^\top V_\Phi \tag{11}$$

For instance as proposed by [2] we can set:

$$V(\mathbf{q}) = \frac{1}{2}\sum_{i=1}^{n} \frac{\alpha_i^2}{\Delta \bar{q}_i} \tag{12}$$

with $\Delta \bar{q}_i = \bar{q}_i^{\max} - \bar{q}_i^{\min}$ the size of the joint space for joint i, and

$$\alpha_i = \begin{cases} \mathbf{q}_i - \bar{q}_{li}^{\min}, & \text{if } \mathbf{q}_i < \bar{q}_{li}^{\min} \\ \mathbf{q}_i - \bar{q}_{li}^{\max}, & \text{if } \mathbf{q}_i > \bar{q}_{li}^{\max} \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

we would like to find a similar function to avoid the self collision.

B. A cost function to avoid self-collision

As we want to keep distance between two bodies of the robot to be above a certain threshold a, we propose the following cost function:

$$\begin{aligned} V(\mathbf{q}) &= \sum_{i \in \mathcal{C}_p} g_i(\mathbf{q}) \\ g_i(\mathbf{q}) &= \begin{cases} (\delta_i(\mathbf{q}) - a_i)^2 & \text{if } \delta_i(\mathbf{q}) < a_i \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{14}$$

with $\mathcal{C}_p = \{\mathcal{C}_j, \forall g_j(\mathbf{q}) < \alpha a_j\}$, $1 < \alpha \in \mathbb{N}$. and $\mathcal{C} = \{\mathcal{C}_0, ..., \mathcal{C}_m\}$ is the set of m pairs of bodies which are checked for collision. a_i represents the *activation* distance between the two bodies of pair $\mathcal{C}_i$.

C. Gradient of the cost function

The gradient of the cost function is obtained by simple derivation:

$$\frac{\partial V_i(\mathbf{q})}{\partial \mathbf{q}} = \quad 2(\delta_i(\mathbf{q}) - a_i)\frac{\partial \delta_i(\mathbf{q})}{\partial \mathbf{q}} \tag{15}$$

if $\delta_i(\mathbf{q}) < a_i$, $\frac{\partial V_i(\mathbf{q})}{\partial \mathbf{q}} = 0$ otherwise. The final Jacobian for the cost function is:

$$\frac{\partial V(\mathbf{q})}{\partial \mathbf{q}} = \sum_{i \in \mathcal{C}_p} J_i(\mathbf{q}) \tag{16}$$

which size is $1 \times n$. It is also possible to have a weighted function on all the pair of bodies formulate as:

$$g_i(\mathbf{q}) = w_i(\delta_i(\mathbf{q}) - a_i) \tag{17}$$

But this will imply to compute the gradient on each pair.

D. Exponential decay

As self-collision avoidance (SCA) is of prime interest for safety reason, we would like to put it as one of the first task to be realized. In order to insert the SCA task inside the stack of tasks, we need to create a projector to constraint the lower priority tasks inside the null space of SCA's task. Integrating the Jacobian of the SCA task directly and its related null-space as proposed in 4, is not a good idea. Indeed there is no constraint regarding the evolution of the function with respect to time. Therefore we finally introduce the following task: $\mathbf{e} = V(\mathbf{q})$, $\dot{\mathbf{e}}^* = -\lambda \mathbf{e}$. The second equation means that the desired task velocity is an exponential decay.

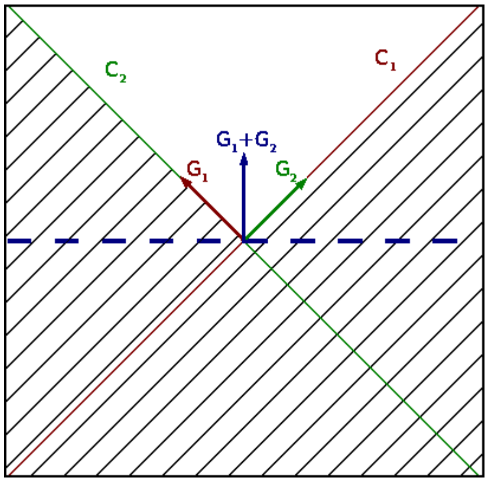

Fig. 2. Using a one-dimensional task for collision avoidance allows the violation of constraints.

E. A multi-dimensional task to avoid self-collision

In case of simultaneous collisions, it appears that task (14) can cause some problems. The null space of the task (14) allows collisions to occur, because the avoidance of potential collision is considered as a one-dimension task; the gradients of all the collision distances are summed. The null space of this task will thus be of higher dimension than it would have been if we had considered each collision separately. This is illustrated on figure 2. Let us suppose the operational space is a plane with two collisions detected. The figure represents the velocity plane. G_1 and G_2 represent the gradients of the collision distances. The hatched area represent the locus of velocities for which a collision will occur. C_1 and C_2 represent the collision constraints. In the framework of the stack of tasks, no motion is possible in such a case because the velocity along both G_1 and G_2 will be constrained to be zero and since G_1 and G_2 are independant, no more degree of freedom is available. However, if we use task (14), lower priority tasks will be projected onto the null space of *the sum* of G_1 and G_2. This null space is one-dimensional and thus allows motion along the dashed line which cause the violation of the collision constraints. In order to avoid this behavior, we replaced task (14) with the following:

$$V(\mathbf{q}) = \begin{pmatrix} \vdots \\ g_i(\mathbf{q}) \\ \vdots \end{pmatrix} i \in \mathcal{C}_p$$
$$g_i(\mathbf{q}) = \begin{cases} (\delta_i(\mathbf{q}) - a_i)^2 & \text{if } \delta_i(\mathbf{q}) < a_i \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

The dimension of (18) can change depending on the number of potential collisions. To avoid implementing dimension-varying tasks, we fixed the dimension of (18) to a constant N_c, so that it is possible to handle up to N_c collisions. We tested the case where $N_c = 5$.

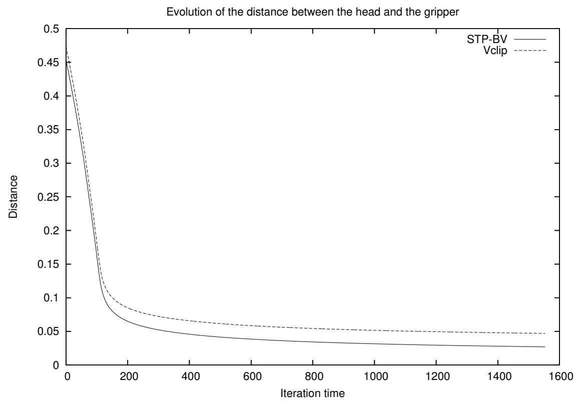

Fig. 3. Distance of the head and the gripper for $V(q) = (\delta_0(\mathbf{q}) - a_0)^2$

F. Problem of remaining stuck in collision

In some other experiments of the collision avoidance control, we observed that the robot was stuck at distance a_0 to contact, after it occurred. This behavior was observed even if another task with a lower priority was pushing the robot away from the collision. Such a behavior is due to the collision-avoidance task that exponentially decreases the distance between two bodies to the minimum distance. The minimum distance is thus asymptotically reached after a long time. The collision task is deactivated only when this minimum distance is reached. While the collision task is converging, no other task with a lower priority can act in its space to quit the collision. For the experiments presented in this paper, we believe this problem was not observed because collision constraints are strictly convex and that it is possible to quit a collision in the null-space of a collision task thanks to the high-redundancy of HRP-2. Solving such a problem requires to develop a new theoretical framework to deal with collision constraints, that are inequality constraints rather than equality constraints.

V. Experiments

A. Control the distance between two objects.

In order to have a better understanding of the collision distance behavior and its jacobian inside the stack of tasks, let us consider a simpler version of the collision avoidance control law proposed previously. The cost function is set to:

$$V(q) = (\delta_0(\mathbf{q}) - a_0)^2 \tag{19}$$

where $\delta_0(\mathbf{q})$ is the distance between the head and the right gripper, and a_0 is set to 0.01 m. Here the test $\delta_0(\mathbf{q}) - a_0 > 0$ is removed. This task is interesting because as there are eight articulations from the gripper to the head it is redundant, so an infinite number of solutions is possible. Thus face crossings triggering discontinuities are very likely to occur. The evolution of the distances between the two bodies return by V-Clip and STP-BV are given in figure 3. Note that the graph's is very close to an exponential decrease. The evolution of the Jacobian's relevant components are depicted in figure 4. As expected, the legs were equal to zero, as well as the chest, the head's tilt, the free-flyer and the left arm.

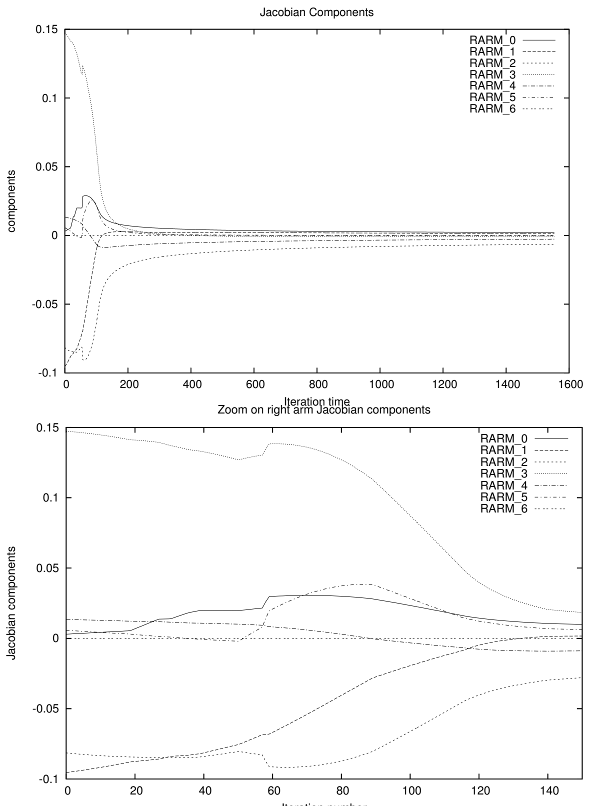

Fig. 4. Right arm's Jacobian components for the distance between the head and the gripper with $V(q) = (\delta_0(\mathbf{q}) - a_0)^2$

The only moving parts of the robot are the right arm and the head's pan. Between iterations 50 and 60 a discontinuity seems to have occur (figure 4 up), but a careful examination shows that this is only a quick continuous change (figure 4 bottom). This especially appears when two faces are parallel during which the witness points thus move very quickly along the STB-BV geometric shapes.

B. Avoiding known obstacle

In this section, the experiment fully exploits the redundancy of the HRP-2 humanoid robot by combining several tasks. The robot has to grasp a ball while walking. An image centering task ensures that the robot is always seeing the ball. An arm-orientation task makes sure that the gripper is ready to grasp. When the gripper is close enough the ball is catched. If the gripper misses, it is detected by monitoring the torque. A detailed description of the control laws can be found in [9]. The priorities of the tasks are the same, with the collision avoidance task having the second priority just after walking. Two original modifications have been realized to implement obstacle avoidance in the stack of tasks. Firstly a set of 116 pairs are chosen to track auto-collision. Bodies linked to each other are not included because they are treated by the joint limits constraints. The second originality is done through the inclusion of a 25 cm diameter ball in the collision pairs. The ball position is put in such way that the robot's hand comes into collision, see figure 5. The value starting the inclusion of the distance inside the control law is set to $a_i = 0.1$ m.

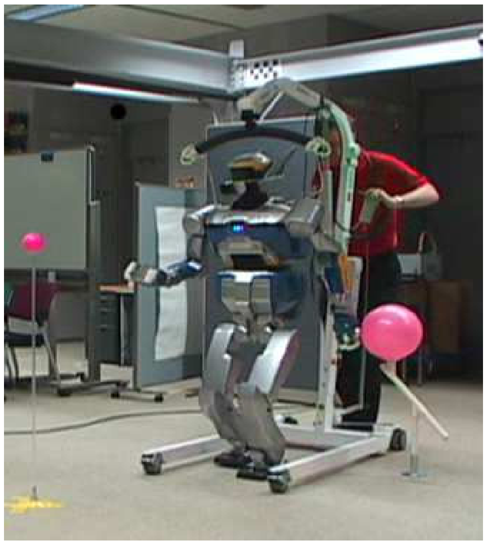

Fig. 5. Initial motion with collision

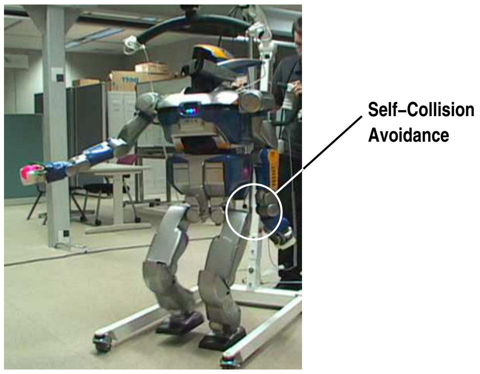

Fig. 6. Avoiding Self-Collision ($a_i = 0.01$ m).

The robot avoids the ball in two occasions. First by holding the hand when moving forward. Then the avoidance stops while the robot is moving away. In second when the robot swing back to the left for the next step the hand and the arm moves to the right to avoid collision.

As in [9]the high-level controller removes the orientation and grasp tasks right after grasping, and a task extending the right arm is put inside the stack. The CoM task then uses the left arm to maintain the CoM given by the pattern generator. However this motion makes the robot reach its joint limits. The left arm then moves toward the left leg, and avoid it thanks to self-collision pairs in the controller, see figure 6.

The described experiment can be watch in the companion video, and is depicted in figure 7.

VI. CONCLUSION

We have presented a new controller which can be used to avoid self-collision and collision with an external object. This new controller is based on a proximity distance which has a continuous gradient. A new geometric representation of the robot bodies bouding volume is at the heart of this

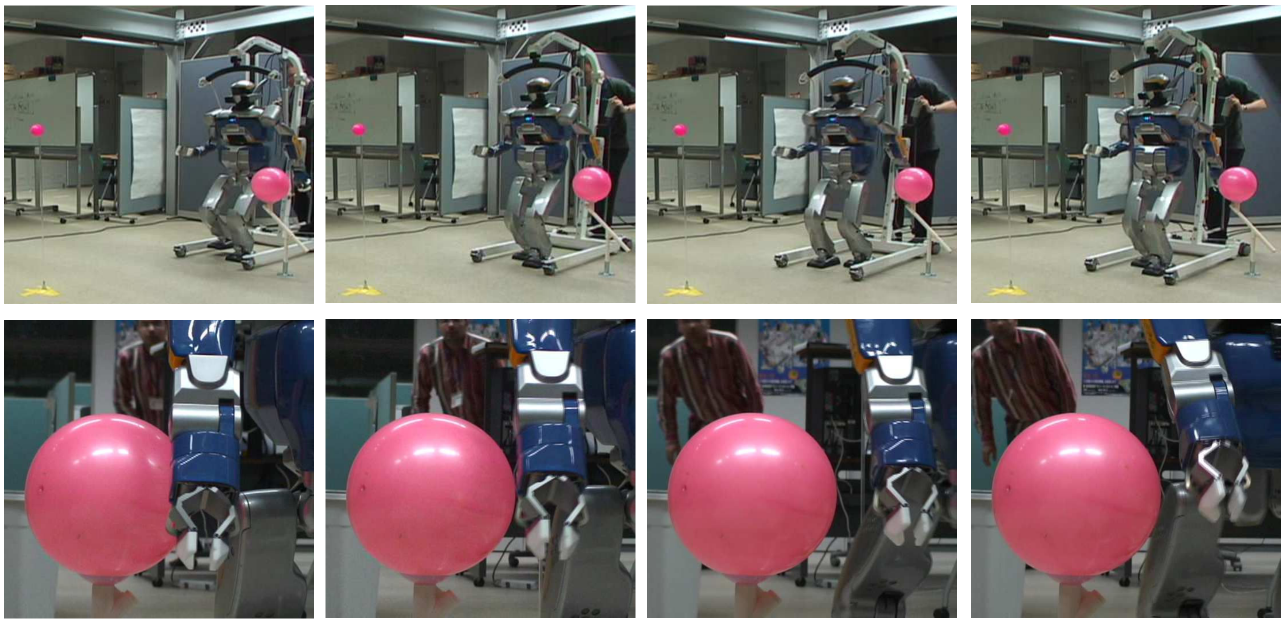

Fig. 7. HRP-2 reactively avoids the biggest ball.

new result. This controller has been implemented and tested on a real full-size humanoid robot.

The main limitation of the current system is that the Jacobian components related to the free flyer's humanoid are not considered. Indeed this might involve to change in real-time the CoM trajectory and in some cases the foot position. A new challenge is to include obstacle avoidance in real-time biped walking engine.

In our future work, we will investigate new methods allowing to get rid of some numerical problems encountered by the adaptation on top of V-Clip.

Acknowledgment

This work is partially supported by grants from the ROBOT@CWE EU CEC project, Contract No. 34002 under the 6th Research program http://www.robot-at-cwe.eu.

References

[1] S. M. LaValle, *Planning Algorithms*. Cambrige University Press, 2006.

[2] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, Spring 1986.

[3] E. S. Neo, K. Yokoi, S. Kajita, F. Kanehiro, and K. Tanie, "A switching command-based whole-body operation method for humanoid robots," *IEEE/ASME Transactions on Mechatronics*, vol. 10, no. 5, pp. 546–559, 2005.

[4] L. Sentis and O. Khattib, "A whole-body control framework for humanoids operating in human environments," in *ICRA*, 2006, pp. 2641–2648.

[5] A. Escande, S. Miossec, and A. Kheddar, "Strictly convex hulls for computing continuous gradient proximity distances," *IEEE Transactions on Robotics*, under review.

[6] R. Patel, F. Shadpey, F. Ranjbaran, and J. Angeles, "A collision-avoidance scheme for redundant manipulators: theory and experiments," *Journal of Robotic Systems*, vol. 22, no. 12, pp. 737–757, 2005.

[7] S. R. Lindemann and S. M. LaValle, "Simple and efficient algorithms for computing smooth, collision-free feedback laws," *IEEE Transactions on Robotics*, To appear.

[8] N. Mansard and F. Chaumette, "Task sequencing for sensor-based control," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 60–72, February 2007.

[9] N. Mansard, O. Stasse, F. Chaumette, and K. Yokoi, "Visually-guided grasping while walking on a humanoid robot," in *ICRA*, Roma, Italy, 2007, pp. 3041–3047.

[10] H. Sugiura, M. Gienger, H. Janssen, and C. Goerick, "Real-time self collision avoidance for humanoids by means of nullspace criteria and task intervals," in *IEEE/RSJ International Conference on Humanoid Robots*, 2006, pp. 575–580.

[11] E. Larsen, S. Gottschalk, M. C. Lin, and D. Manocha, "Fast proximity queries with swept sphere volumes," Department of Computer Science, University of North Carolina, Tech. Rep., 1990.

[12] J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue., "Dynamically-stable motion planning for humanoid robots," *Autonomous Robots (Special issue on Humanoid Robotics)*, pp. 105–118, 2002.

[13] K. Okada and M. Inaba, "A hybrid approach to practical self collision detection system of humanoid robot," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2006, pp. 3952–3957.

[14] G. van den Bergen, *Collision detection in interactive 3D environments*, ser. The Morgan Kaufmann Series in Interactive 3D Technology, D. H. Eberly, Ed. Morgan Kaufmann Publishers, 2004.

[15] C. Ericson, *Real-time collision detection*, ser. The Morgan Kaufmann Series in Interactive 3D Technology, D. H. Eberly, Ed. Morgan Kaufmann Publishers, 2005.

[16] B. Siciliano and J.-J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *IEEE Int. Conf. on Advanced Robotics (ICAR'91)*, Pisa, Italy, Juin 1991, pp. 1211–1216.

[17] N. Mansard and F. Chaumette, "Tasks sequencing for visual servoing," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'04)*, Sendai, Japan, Novembre 2004, pp. 992–997.

[18] A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 7, no. 12, pp. 868–871, Dec. 1977.

[19] N. Mansard and F. Chaumette, "Visual servoing sequencing able to avoid obstacles," in *IEEE Int. Conf. on Robotics and Automation (ICRA'05)*, Barcelona, Spain, April 2005, pp. 3154–3159.

[20] Y. Mezouar and F. Chaumette, "Path planning for robust image-based control," *IEEE Trans. on Robotics and Automation*, vol. 18, no. 4, pp. 534–549, Août 2002.