Humanoid Motion Planning using Multi-Level DOF Exploitation based on Randomized Method

Eiichi Yoshida

AIST/IS-CNRS/STIC Joint French-Japanese Robotics Lab (JRL)
Intelligent Systems Research Institute, Natl Inst of AIST, AIST Central 2, Umezono 1-1-1, Tsukuba, Ibaraki 305-8568 Japan
JRL/ Laboratoire de Robotique de Versailles, 10/12 Avenue de l'Europe 78140 Vélizy, France
e.yoshida@aist.go.jp

***Abstract*— This paper addresses a multi-level exploitation of degree of freedom (DOF), for humanoid motion planning based on a randomized method. The improvement of autonomy and mobility is required so that humanoid robot can perform tasks in various environments. Although a humanoid robot has many DOFs, all of them do not have to be controlled depending on the situation and the required tasks. Utilizing rapidly-exploring random trees (RRTs) as an efficient planning tool, a method of multi-level DOF exploitation is developed that adjusts the controlled DOFs according to the detected environmental situation. That allows the planner to deal with only the necessary sets of DOFs instead of exploring the search space of all the DOFs, which leads to efficient humanoid motion planning. The proposed method is verified through simulations using software platform OpenHRP and HRP-2 humanoid robot model.**

***Index Terms*— Humanoids, motion planning, randomized planning, rapidly-exploring random trees (RRTs)**

I. Introduction

Humanoid robots are currently being investigated intensively along the rapid progress of hardware. Since they have approximately the same form and sometimes the same size as humans, they are expected to perform various tasks to assist humans in their daily life. Other possible applications include the replacement of humans to operate machines or to inspect equipments in hazardous environments such as constructing sites or nuclear plants.

To achieve those tasks, humanoid robots should have sufficient autonomy and mobility. This paper deals with the motion generation of humanoid robots so that they can move around in environments with obstacles. Humanoid robots have many degrees of freedom (DOFs) to be controlled and they should be coordinated correctly when they walk around and perform tasks.

Obviously, it is not a good idea to try to design motion separately for all the DOFs in the configuration space. For this purpose, a model-based pattern generator [1] is useful. It provides walking pattern for the humanoid that can be controlled like an omni-directional vehicle, which makes it easier to plan the humanoid motion. We will make use of this type of pattern generator since it allows the motion planner to work within workspace. When the robot is moving in large open free space, the motion can be controlled by a few parameters. However, when the robot faces obstacles, some other motion than the walking pattern may be required (Fig.1). We propose a multi-level approach for usage of those DOFs to cope with the problem in order to improve the efficiency of planning.

Motion planning has also been an issue investigated continuously [2]. Recently, a method called "randomized planning" has been attracting robotic researchers' interest [3], [4]. The principal idea is to sample randomly in the configuration space to explore the path to the goal. We adopt a method called rapidly exploring random trees (RRTs) to benefit from a number of advantages of RRT planner, including simple implementation, uniform search, and applicability to a dynamic system with differential constraints.

In this paper, an approach for multi-level exploitation of DOFs is addressed for a humanoid robot based on model-based pattern generator and randomized motion method. The

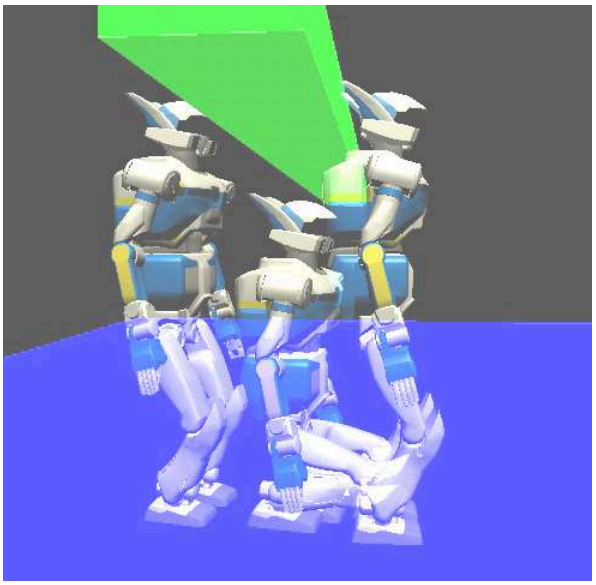


Fig. 1. Humanoid robot going under an obstacle

goal of the paper is to devise an efficient motion planner that allows the robot to exploit its DOFs adaptively and efficiently according to the environment. The randomized RRT planner outputs the pattern generator parameters within workspace, which is considered as reduced DOF control mode. If collision is previewed during its motion, the planner goes to augmented DOF control mode to avoid obstacles. This allows the planner to explore a reduced set of DOFs in large free space, and to plan using required additional DOFs when some constraints are detected. It can be used for example when the robot should go underneath obstacles as shown in Fig. 1.

This paper is organized as follows. Section II presents related work about humanoid robot motion planning. After introducing RRT planner in Section III, a motion planner that enables multi-level exploitation of DOF is described in IV based on this randomized method. Section V shows some simulation results before concluding the paper in Section VI.

II. Related work

Humanoid motion planning has become a major research topic in recent years. Kuffner and his colleagues have been making continuous contribution to humanoid motion planning [5]. Footstep planning on rough terrain [6] and navigation in environments with movable obstacles [7] have been addressed as well as dynamic balancing motion [8]. Kanehiro et. al [9] present locomotion planning for humanoid robots to pass through narrow spaces by discretely changing the locomotion modes such as walking, sidestepping or crawling. An integrated motion control scheme for running and walking has been proposed by Nagasaka et al.[10]. Okada et al. [11] have addressed motion planning for collision-free full body posture control by dividing the robot into movable, fixed and free limbs using RRT planner. He has also showed task-oriented motion planning [12]. Humanoid motion planning for task execution is another issue tackled recently, for such manipulation tasks as pushing [13], [14], lifting [15], or pivoting [16] objects. Despite not exactly humanoid motion planning, interesting research has been conducted on motion generation for digital actor animation [17].

Although there are a few studies that tune the number of controlled DOFs to increase workspace [18] or to execute dynamically prioritized task [19], motion planning that can adjust the controlled DOFs including walking has been less exploited. This paper aims to generate whole-body walking motion whose degrees of freedom can be adaptively controlled according to the environment. It can lead to future development of a planning system that can integrate global motion planning with local collision avoidance and task-specific motion generation.

III. Motion Planning using Rapidly-Exploring Random Trees (RRTs)

RRTs have been proposed by LaValle and Kuffner as a randomized motion planning method [3]. The key idea is to incrementally expand search trees that attempt to explore the state space in a rapid and uniform manner. It has been proven that this method is probabilistic complete. It means that a desired path will eventually be found as the number of vertices approaches infinity. It is advantageous to employ this planner for many-DOF system like humanoid since it has such good characteristics as simplicity of implementation for exploring many DOFs and possibility of including differential constraints.

The basic algorithm is shown in Fig. 2, which makes a tree $\mathcal{T}$ grow in the space of configuration q of a moving entity, which can be an object, a vehicle, or a robot. At each step, after generating a random configuration node q_{rand}, the function EXTEND($\mathcal{T}$, q_{rand}) is called to extend the tree by using NEW_CONFIG(q, q_{near}, q_{new}) function as illustrated in Fig. 3. NEW_CONFIG() selects a node q_{near} nearest to q based on a given metric, and then generates a new node q_{new} that advances to q to grow the tree $\mathcal{T}$. In this way, the tree rapidly explores the state space through biased search of large unexplored region.

Several RRT-based motion planners have been proposed according to the problem. For example, RRT-Connect [20] is a bidirectional planner that explores RRTs from initial and goal position in environments, possibly with obstacles. The search can be accelerated using bidirectional planning.

In the following, we assume that the environment is known using certain sensors, and that planning is executed off-line.

The RRTs can include differential constraints where simple

```
BUILD_RRT(q_init)
1   T.init(q_init)
2   for k=1 to K do
3     q_rand ← RAND_CONFIG();
4     EXTEND(T, q_rand);
5   Return T;

EXTEND(T, q)
1   q_near ← NEAREST_NEIGHBOR(q, T);
2   if NEW_CONFIG(q, q_near, q_new) then
3     T.add_vertex(q_new)
4     T.add_edge(q_near, q_new)
5     if q_near = q then
6       Return Reached;
7     else
8       Return Advanced;
9   Return Trapped;
```

Fig. 2. Basic algorithm of RRT extension

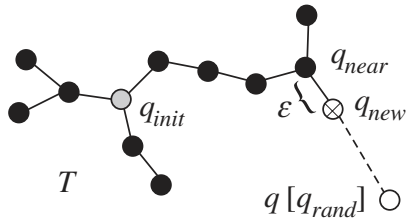


Fig. 3. Description of function EXTEND($\mathcal{T}$, q)

interpolation between configurations does not apply, which is in general the case including humanoid robots. In this case, the relationship between control input and resulting configuration are specified. This is an extension of RRTs for kinodynamic planning [21]. In the algorithm of Fig. 2, the configuration q is replaced by the state x that describes robot's current state. Also, NEW_CONFIG(q, q_{near}, q_{new}) is replaced by GEN_STATE(x, x_{near}, x_{new}, *Inputs*, Δt) that generates next state x_{new} advancing to the given state x from its nearest neighbor x_{near} in the tree, by selecting control input from the possible set *Inputs*, as shown later in Fig. 5. In the following discussion, we will adopt this extended model including differential constraints to deal with the general case of humanoid.

IV. Multi-level DOF exploitation for humanoid motion

Since a humanoid robot has many DOFs, it is inefficient to explore all the joint angle values in the configuration space to generate motion patterns. When we humans walk, the motion of our joints is coordinated even though we are not conscious of their individual motion.

Similar argument can be applied to the motion generation of humanoid robots. If it goes around a large free space, it can be regarded as a box that can be approximated by an omni-directional vehicle, which makes motion planning easier.

Other degrees may have to be activated when the robot encounters obstacles as shown in Fig. 4. In this paper, this idea is implemented in the framework of randomized motion planner RRTs. As shown in the figure, when an obstacle is detected in a certain region around the robot, the new state related to motion search is activated. In the example of Fig. 4, parameters (x, y, θ) are used to plan the motion in a large free space. When obstacle collision with the head part of the robot is previewed, a new DOF z is introduced in the searching space to avoid the obstacle in the downward direction. When the collision becomes less probable, this DOF is removed from the searching space to reduce computational cost. In the same way, if other parts of the robot detect some constraints from the environment, corresponding new states may be added in the searching space.

The algorithm is detailed in Fig. 5. In this algorithm, the overall state of the robot is described by global state x^G, including the position and orientation of the base frame of the robot with all the joint angles. The RRTs consist of nodes including this state x^G.

During the planning process, according to the detected situation in relation with the environment, an appropriate set of states x^c to be controlled is extracted by function CONVERT_STATE(). In the case of Fig. 4, this set x^c is (x, y, θ) when no obstacle is detected in its surroundings. Then if collision becomes probable, x^c is now set to (x, y, z, θ) to add another degree of freedom to control. From this state, possible control inputs *Input* are derived. If the dimension of x^c is low, the number of *Input* is small and computational cost can be reduced.

Function GEN_STATE() corresponds to NEW_CONFIG() in Fig. 2 and tries to find a new state that approaches the given state x^G using the derived inputs. In this function, the following must be taken into account (Fig. 6).

Differential Constraints.

New states are calculated through integration using the adjusted state x^c and given input set *Inputs* by

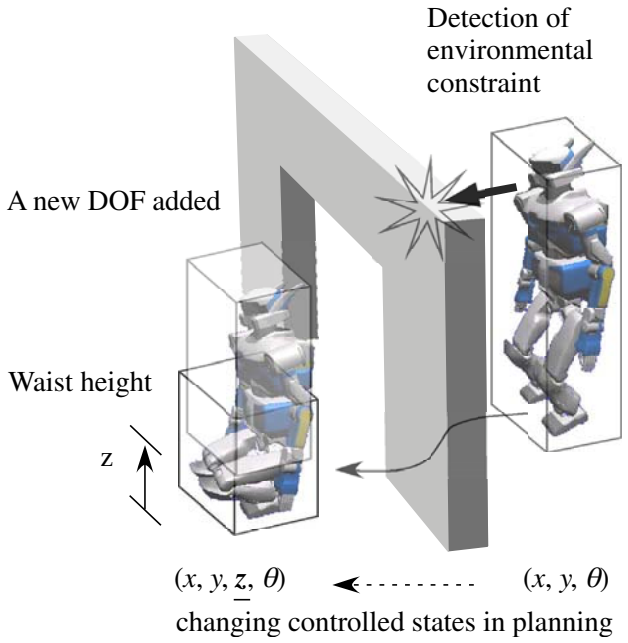


Fig. 4. Multi-level exploitation of DOF according to environment

BUILD_RRT_STATE(x^G_{init})
1 $\mathcal{T}$.*init*(x^G_{init})
2 **for** k=1 to K **do**
3 $x^G_{rand} \leftarrow$ RAND_STATE();
4 EXTEND_STATE_MULTIDOF($\mathcal{T}$, x^G_{rand});
5 Return $\mathcal{T}$;

EXTEND_STATE_MULTIDOF($\mathcal{T}$, x^G)
1 $x^G_{near} \leftarrow$ NEAREST_NEIGHBOR(x^G, $\mathcal{T}$);
2 *Inputs* $\leftarrow$ GET_INPUTS(x^G_{near});
3 $\{x^c, x^c_{near}\} \leftarrow$ CONVERT_STATE($\{x^G, x^G_{near}\}$);
4 **if** GEN_STATE(x^c, x^c_{near}, x^c_{new}, *Inputs*, Δt); **then**
5 $x^G_{new} \leftarrow$ GLOBAL_STATE(x^c_{new});
6 $\mathcal{T}$.*add_vertex*(x^G_{new});
7 $\mathcal{T}$.*add_edge*(x^G_{near}, x^G_{new});
8 **if** NEAR_ENOUGH(x^G_{near}, x^G) **then**
9 Return *Reached*;
10 **else**
11 Return *Advanced*;
12 Return *Trapped*;

Fig. 5. Algorithm of search using multi-level DOF exploitation in state space based on RRT

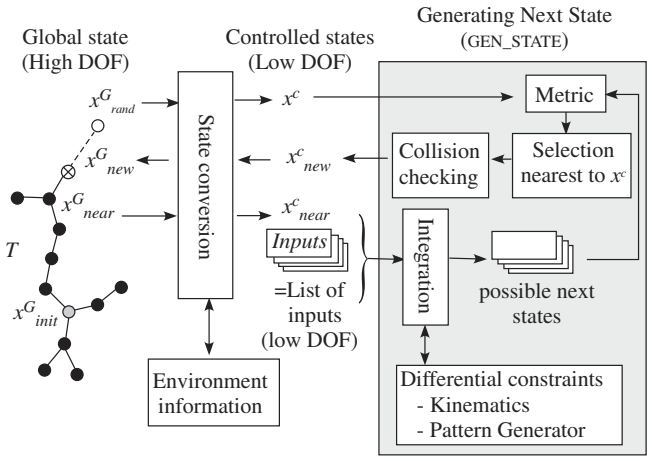


Fig. 6. Extending RRT by generating new state by adjusting DOF of controlled states

respecting the differential constraint of the robot. In the case of the humanoid, the new states are derived based on the pattern generator. Differential constraints should be taken into account in cases where the robot dynamics cannot be described only using linear interpolation between states and severe constraints are often imposed on the allowable velocities. For example, aircraft, nonholonomic wheeled vehicles, or legged robots with fewer than four legs. In general the state transition of robot dynamics can be given as $\dot{x} = f(x, u)$ [3] for state x and input u.

Metric.

To perform a randomized search, a notion of distance, called metric, is used to expand the tree in the "neighborhood" of existing nodes incrementally in state space. It is therefore necessary to define the metric between the states. It is often a hard problem to have a good metric between the states depending on the planning problem. In the proposed algorithm for humanoid robot motion planning, the metric distance between the given state and the newly generated state is computed based on the global state description to maintain consistency. Namely, a weighted distance for selected sets of global states is utilized depending on the planning problem. For the obstacle avoidace problem in the next section only the position and orientation are taken into account. But of course, we can add joint angles or other global states for different types of the problem where posture is more important.

Collision Checking.

The newly generated state should also go through collision checking. If there exist collision-free states in the possible sets of calculated next states, the function NEW_CONFIG() selects the nearest one to x^c, and returns *true*.

Based on this multi-level selection of states to be controlled, RRTs are expected to explore the large state space in an efficient manner. This method can also be applied to other planning method derived from basic RRT planner, for example bidirectional planner such as RRT-Connect [20].

V. Simulation using HRP-2 humanoid platform

We have conducted a simulation of the proposed pivoting motion using the humanoid robot simulator OpenHRP [23], [24] where the humanoid robot HRP-2 (Fig. 7) [22] is modeled. HRP-2 has 30 degrees of freedom with 1.54 [m] in height and 58 [kg] in weight. Wrists and ankles are equipped with force sensors.

Control software based on OpenHRP to facilitate the development of applications that use the robot hardware HRP-2 is provided by GeneralRobotix Inc. [25]. This control software has several features such as binary compatibility between simulation and hardware, an architecture that allows parallel development of user control software as modules called "plugins," and built-in control modules that facilitate development.

To perform planned motion we utilize such built-in modules as an online pattern generator based on 3D linear inverted pendulum mode [1] and a kinematic computation. The online pattern generator can generate the walking pattern as the output of a given command. It accepts the command in the form of (x, y, θ) in the holonomic omni-directional vehicle mode with (x, y) being the relative position to go to and θ the relative angle. It has also an arc path mode.

The output walking pattern is computed in a dynamically stable manner so that ZMP stays in the support polygon of the robot feet and is provided as the sequence of joint angles of the robot at each control cycle of 5[ms]. It can also generate patterns for different height of waist position z. The pattern generator accepts a sequence of (x, y, θ) and can execute it by connecting them to generate walking motion. We employ the

Fig. 7. Humanoid robot HRP-2

omni-directional mode in our simulation. Any combination of movements is possible as long as each fraction of movements is expressed in the above form since the pattern generator can divide the movements into appropriate footsteps. Even though smoother path is preferable, ragged one can also be accepted without problem in this case.

To implement the RRT planner, we adopt the MSL library [26] for RRT planner. It provides several different randomized motion planning algorithms with graphic user interface and collision detection. The user can add its own class of robot and planner as a derived class from the provided basic classes in C++ language. The interface with the OpenHRP simulator has been implemented through CORBA (Common Object Request Broker Architecture) [27] objects.

We have conducted a simulation using the above simulation setup. Table I shows several parameters used in the simulation. There exist two obstacles, one of them is short and lies on the ground and the other one is wide and placed at a high position (see Fig. 9). The inputs of the robot are given as a simple set of movements and will need to be reconsidered to generate smoother movements. When the robot controls the state z, the robot stops walking and changes the height of its waist.

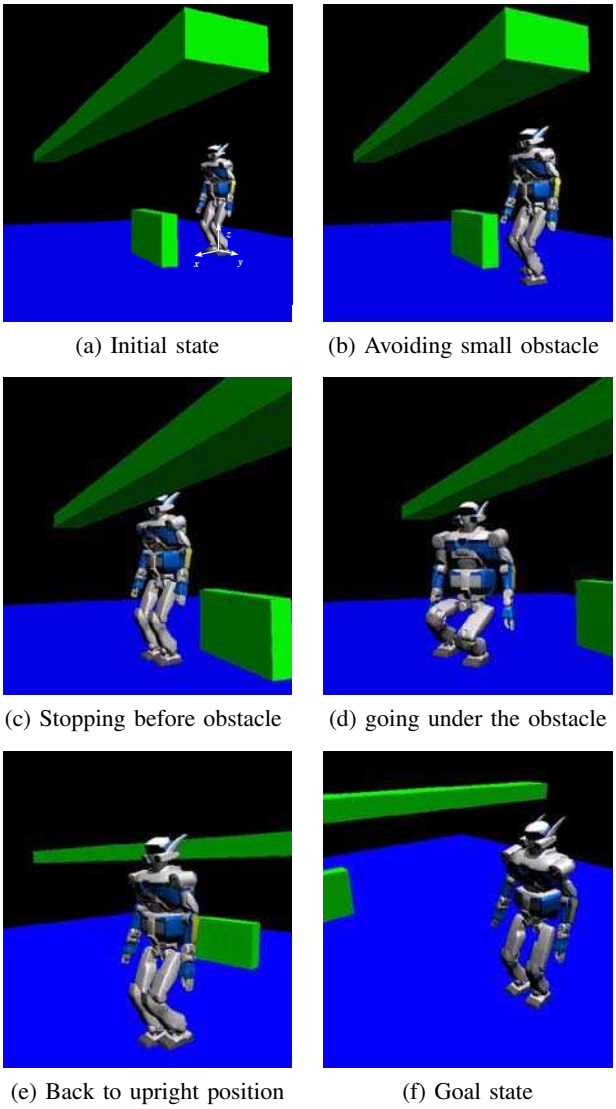
The metric is calculated in such a way that the contribution of each component is balanced. It is a weighted distance between the position (x, y, z) and the orientation represented by quaternion (q_1, q_2, q_3, q_4) of the robot. As the scale of those states is different in the given environment, those weights are defined to equilibrate the effect of those states. The weight w_z of waist height z is set large as its range is smaller than that of variables x and y. Likewise, since q_i takes values between 0 and 1, we also put a larger value so that its contribution to metric is equivalent to the position states. Automatic definition of metric depending on the problem is an issue for future work.

Figure 8 shows snapshots of the planned motion. As a planner, the bidirectional search RRT-Connect [20] is employed. As can be seen, the robot avoids the low obstacle and lowers its body when it is necessary to go through the other obstacle. The planner controls three states (x, y, θ) when no obstacle is detected within distance D of the head part. Planning is conducted for states (x, y, z, θ) where the head part is close to the upper obstacle. An example of explored RRTs is shown in Fig. 9. In this way, the simulation demonstrates the basic function of the proposed planner that can generate collision-free path reaching the goal by changing the controlled states.

TABLE I

SIMULATION PARAMETERS (UNITS: [M],[DEG], COORDINATES: FIG. 9 A)

Environment size	$\pm 500 \times \pm 300$
Waist height	0.33 – 0.67
Initial Position (x_i, y_i, θ_i)	(0, 0, 0)
Goal Position (x_g, y_g, θ_g)	(3.5, 1, -90)
Obstacle 1	size 0.2 × 1.0 × 0.6 at (1.0, 0, 0.3)
Obstacle 2	size 0.2 × 6.0 × 0.2 at (2.0, 0, 1.5)
Inputs (x, y, θ)	(0.05,0,0), (0,±0.05,0), (0,0,±10)
Inputs (x, y, z, θ)	(0.05,0,0,0), (0,±0.05,0,0), (0,0.5,±0.05,0) (0,0,0,±10)
Metric w_z=400.0, w_q=2.0	$\sqrt{(x^2+y^2)+w_z z^2+w_q \sum_i^4 q_i^2}$
Collision previewing distance D	0.2[m] from the head part



(a) Initial state (b) Avoiding small obstacle

(c) Stopping before obstacle (d) going under the obstacle

(e) Back to upright position (f) Goal state

Fig. 8. Simulation of avoidance of different obstacles

We have compared the results using the proposed multi-level DOF exploitation planner with the planner that controls all the four states all the time. The result did not show significant improvement, both required about 400 nodes and 4000 collision checks during 10 seconds in this simple environment. The difference would be more significant in the case of more DOFs involved in this multi-level DOF exploitation scheme and more complex environment, which will be addressed in future work.

In this simulation, the RRT output is simply replayed as feed-forward input to the online pattern generator. For the

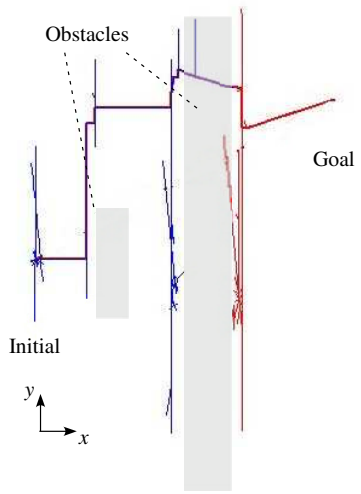


Fig. 9. Result of RRT explored for obstacle avoidance.

purpose of more precise collision checking and dynamic control during motion, closer integration of walking pattern generator to the planner is also one of the future research issues.

VI. Conclusion

In this paper we have presented a multi-level DOF exploitation for humanoid motion planning using a randomized method. Since a humanoid robot has many DOFs, it is preferable to adjust the controlled DOFs according to the detected situation. We have addressed a problem of collision avoidance that requires the robot to exploit extra DOFs to avoid obstacle, whereas usage of the pattern generator simply suffices when moving in a large open space. The proposed method is implemented using RRT as a randomized planner that can include differential constraints. The planner adjusts the controlled state when it previews some events like collision. This allows the planner to deal with only necessary states to control depending on its surrounding situation. A simple simulation is conducted to verify that the proposed method works correctly.

Although the simulation result shows a simple example, the proposed method can potentially be extended to more complex cases. For example, not only the position of the head but also that of the hands can be controlled if necessary. We can also consider the motion planning of a humanoid robot performing a pick and play task. If the task cannot be executed only by using the arm because of obstacles, the robot exploit movements of other DOFs, by moving its body or by walking if necessary.

Even if the planning is performed off-line the proposed method has the possibility of real-time extension using sensors like vision, since it integrates on-line pattern generator and simple planning algorithm.

Acknowledgment

The author expresses great thanks to Dr. Vincent Hugel of JRL who gave a lot of productive advice throughout the research work.

References

[1] S. Kajita, et al: "Biped Walking Pattern Generation by using Preview Control of Zero-Moment Point," Proc. 2003 IEEE Int. Conf. on Robotics and Automation, 1620-1626, 2003.
[2] J-C. Latombe, Robot Motion Plannining, Kluwer Academic Publishers, Boston, MA, 1991.
[3] S. LaValle and J. Kuffner: "Rapidly-Exploring Random Trees: Progress and Prospects," In Algorithmic and Computational Robotics: New Directions, 293–308, A K Peters, Wellesley, 2001.
[4] L. Kavraki, et al. "Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces," IEEE Trans. on Robotics and Automation, **12**-4, 566-580, 1996.
[5] J. Kuffner "Motion planning for humanoid robots." Proc. 20th Int'l Symp. Robotics Research (ISRR'03), 2003.
[6] J. Chestnutt and J. Kuffner. "A Tiered Planning Strategy for Biped Navigation," Proc. IEEE Int. Conf. on Humanoid Robotics (Humanoids'04), 2004.
[7] M. Stilman and J. Kuffner. "Navigation among movable obstacles: Real-time reasoning in complex environments," In Proc. IEEE Int. Conf. on Humanoid Robotics (Humanoids'04), 2004.
[8] J .Kuffner, et al. "Dynamically-stable motion planning for humanoid robots," Autonomous Robots, **1**-12, 105-118, 2002.
[9] F. Kanehiro, et al. "Locomotion Planning of Humanoid Robots to Pass through Narrow Spaces," Proc. 2004 IEEE Int. Conf. on Robotics and Automation (ICRA'04), 2004
[10] K. Nagasaka et al. "Integrated Motion Control for Walking, Jumping and Running on a Small Bipedal Entertainment Robot," Proc. 2004 IEEE Int. Conf. on Robotics and Automation (ICRA'04), 3189-3194, 2004.
[11] K. Okada, et al. "Integrated System Software for HRP Humanoid," Proc. 2004 IEEE Int. Conf. on Robotics and Automation (ICRA'04), 3207-3212, 2004.
[12] K. Okada, et al. "Environment Manipulation Planner for Humanoid Robots Using Task Graph That Generates Action Sequence," Proc. 2004 IEEE Int. Conf. on Intelligent Robots and Systems (IROS'04), 1174-1179, 2004.
[13] H. Harada, et al. "Real-Time Planning of Humanoid Robot's Gait for Force Controlled Manipulation," Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'04), 616-622, 2004.
[14] T. Takubo, et al. "Mobile Manipulation of Humanoid Robots - Control Method for CoM Position with External Force -," Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'04), 1180-1185, 2004.
[15] H. Harada, et al. "A Humanoid robot carrying a heavy object," Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'05), 2005.
[16] E. Yoshida, et al. "Pivoting Manipulation of a Large Object: A Study of Application using Humanoid Platform," Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'05), 1052-1057, 2005.
[17] Gustavo Arechavaleta, et al. "Planning Fine Motions for a Digital Factotum," Proc. 2004 IEEE Int. Conf. on Intelligent Robots and Systems (IROS'04), 822-827, 2004.
[18] Neo Ee Sian, et al. "A Framework for Remote Execution of Whole Body Motions for Humanoid Robots," Proc. 2004 IEEE-RAS/RSJ International Conference on Humanoid Robots (Humanoids 2004).
[19] L Sentis and O Khatib. "Prioritized Multi-Objective Dynamics And Control Of Robots In Human Environments," Proc. 2004 IEEE-RAS/RSJ International Conference on Humanoid Robots (Humanoids 2004).
[20] J. Kuffner and S. LaValle. "RRT-Connect: an dfficient approach to single-query path planning," Proc. 2000 IEEE Int. Conf. on Roboics and Automation, 995–1001, 2000.
[21] S. LaValle and J. Kuffner: Randomized kinodynamic planning, *Proc. 1999 IEEE Int. Conf. on Roboics and Automation* 473–479, 1999.
[22] K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi and T. Isozumi: "The Humanoid Robot HRP-2", Proc. of IEEE/RSJ Int. Conf. on Robotics and Automation, 1083-1090, 2004.
[23] F. Kanehiro, et al. : "Open Architecture Humanoid Robotics Platform," Proc. IEEE Int. Conf. on Robotics and Automation, 24-30, 2002.
[24] http://www.is.aist.go.jp/humanoid/openhrp/
[25] http://www.generalrobotix.com
[26] http://msl.cs.uiuc.edu/msl
[27] http://www.omg.org. Object Managing Group.