Planning Whole-body Humanoid Locomotion, Reaching, and Manipulation

Eiichi Yoshida, Claudia Esteves, Oussama Kanoun, Mathieu Poirier, Anthony Mallet, Jean-Paul Laumond and Kazuhito Yokoi

Abstract In this article we address the planning problem of whole-body motions by humanoid robots. The presented approach benefits from two cutting edges of recent advancement in robotics: powerful probabilistic geometric and kinematic motion planning and advanced dynamic motion control for humanoids. First, we introduce a two-stage approach that combines these two techniques for collision-free simultaneous locomotion and upper-body task. Then a whole-body motion generation method is presented for reaching including steps based on generalized inverse kinematics. The third example is planning of whole-body manipulation of large object by "pivoting", by making use of the precedent results. Finally, an integrated experiment is shown in which the humanoid robot interacts with its environment through perception. The humanoid robot platform HRP-2 is used as the platform to validate the results.

1 Introduction

As the progress in the hardware of humanoid robot has recently been accelerated, a number of applications is now expected. Their anthropomorphic shape is advan-

Eiichi Yoshida and Kazuhito Yokoi
CNRS-AIST JRL (Joint Robotics Laboratory), UMI 3218/CRT, National Institute of Advanced Industrial Science and Technology (AIST), Umezono 1-1-1, Tsukuba, Ibaraki, 305-8568, Japan, e-mail: {e.yoshida,Kazuhito.Yokoi}@aist.go.jp

Claudia Esteves
Facultad de Matematicas, Universidad de Guanajuato, Guanajuato, 36000 Gto., Mexico. e-mail: cesteves@cimat.mx

Oussama Kanoun, Mathieu Poirier, Anthony Mallet and Jean-Paul Laumond
LAAS-CNRS, 7 avenue du Colonel Roche, F-31077 Toulouse, and Universite de Toulouse ; UPS, INSA, INP, ISAE ; LAAS ; F-31077 Toulouse, France.
e-mail: {okanoun, mpoirier, mallet, jpl}@laas.fr

tageous in moving in environments designed for humans, using the machines or tools designed for humans, and also performing interactions and assistive tasks for humans. For the humanoid to execute effective tasks, the whole-body motion coordination is indispensable. The humanoid motion is characterized by its redundancy and underactuation. The former is obvious, as humanoid robots have usually more than thirty degrees of freedom. The latter means the "base frame" of a humanoid robot, for example its waist, can only be controlled indirectly by articulated legs, unlike wheeled mobile robots. In this article, in order to tackle this planning problem of whole-body humanoid motions, we present the approaches that combine the probabilistic geometric/kinematic motion planning and the dynamic motion generation as follows. At the planning stage, the global humanoid motion is modeled by a kind of vehicle with the bounding volume that is fully actuated to plan the basic path. It is then transformed into dynamically executable humanoid motion by the dynamic motion generator. The whole-body humanoid motions handled in this article include collision-free locomotion, reaching and manipulation.

In the rest of this section, we briefly introduce the basic planning tools and the software and hardware platform which is the common part throughout this article. In Section 2, a two-stage approach for collision-free locomotion is presented. Then we address the dynamic whole-body motion generation by taking the example of reaching including stepping in Section 3. The whole-body manipulation is then dealt with in Section 4 by benefiting from both the collision-free path planning technique and whole-body motion generation. An experiment that integrates the planning and perception is described in Section 5 by using the humanoid robot HRP-2 before concluding the article.

1.1 Basic motion planning methods

For complex robotic systems like humanoid robots, it is reasonable to employ efficient algorithms based on probabilistic planning methods such as diffusing methods like Rapidly-exploring Random Trees (RRT) [11, 23] or sampling method like Probabilistic RoadMap (PRM) [17] and all their variants (see [4, 22] for recent overviews). In those methods, the path search is usually made in the configuration space such as the joint angles. The probabilistic methods compute a graph called a roadmap whose nodes are collision-free configurations chosen at random and whose edges model the existence of collision-free local paths between two nodes.

In probabilistic planning method, the graph is built incrementally by shooting configurations at random. Configurations and local paths between them are included in the graph as soon as they are collision-free. This construction of the graph is called the learning phase. Once the graph is built, then the query phase consists in first adding both starting and goal configurations of the given problem to the roadmap, and then search the graph for a path.

On the other hand, in diffusing methods the graph is built gradually by expanding the tree from the start and/or goal configurations. After a configuration is ran-

domly generated, the nearest configuration is identified. Then a new configuration is computed that advances by a given distance from the nearest node to towards the randomly generated one. If this new configuration and the local path to it are collision-free, then they are added to the graph as a new node and a new edge. This diffusion is repeated until a path is found that connects the start and goal configurations.

In this way, the probabilistic motion planner eventually finds collision-free paths as connected components of the roadmap, which is proven as probabilistic completeness. There are several important components in implementing this probabilistic motion planning: collision checkers, steering methods and path optimizers. Collision checkers validate configurations and paths. Any available free or commercial libraries can be used for this function. A "steering method" is a method that computes an admissible path from an initial configuration to final one in the absence of obstacle. In this article, dedicated steering methods are devised depending on the planning problems. Finally, since paths planned by probabilistic motion planners are often redundant, a "path optimizer" is necessary to remove the redundant parts to obtain a shorter path. For instance, we can employ the "adaptive shortcut" path optimization algorithm proposed by [11].

As described later in Section 2, we generally adopt a two-stage approach for whole-body motion planning. At the first stage, the basic geometric and kinematic motion planner described here is applied to collision-free path planning for a simplified model of the humanoid, like its bounding volume. This is because the computation cost would be too high if random sampling is directly applied to the complex dynamic system with many degrees of freedom.

1.2 Hardware and software platform

We implement motion planners presented in this article in a common software framework "Humanoid Path Planner" [36] on the basis of the motion planning software kit KineoWorks™ [21] as shown in Fig. 1. As illustrated lower part in Fig. 1, this software kit provides the basic methods of planning like PRM or RRT as "roadmap builders", as well as the aforementioned basic functions of collision checking mechanism, a template of steering method and path optimizers.

The object-oriented architecture allows the users to define the specific planner depending on the tackled problem as an inherited class of a general robot motion planner. The planner takes care of interaction with basic functions introduced in 1.1. The problem-specific components, especially steering methods for walking and manipulation are inherited from the template including basic functions. Since the basic interface of class definition of the general framework is common, it is relatively easy to implement those problem specific parts once the planning problem is defined.

The HPP framework also includes a walking pattern generator presented in Section 2 and a dynamic whole-body motion generator in Section 3.

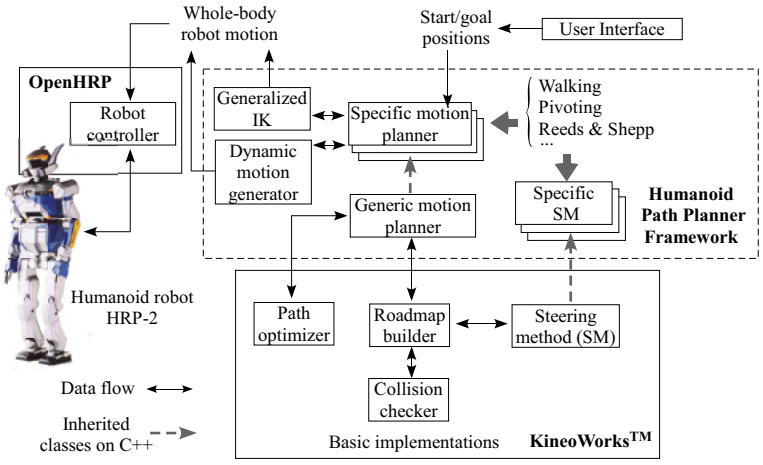


Fig. 1 Architecture of Humanoid Path Planner framework that facilitates implementation of robotic motion planner according to specific problems.

As the hardware platform, we utilize the humanoid robot HRP-2 [16] shown in Fig. 1 which measures 1.58m and weighs 58kg, with 30 DOFs. It has two rotational joints (pitch and yaw) at the chest that provides a wide workarea of upper-body. Six-axis force sensors at the ankles, a rate gyroscope and an accelerometer for attitude estimation are equipped for stability control. Once the whole-body humanoid motion is generated, it is passed to the robot simulator and controller OpenHRP [15].

2 Collision-free Locomotion: Iterative Two-stage Approach

We have proposed a general and practical planning framework that integrates a geometric path planner and a dynamic motion generator [34]. It is based on an iterative two-stage motion planning method, by exploiting the efficiency of probabilistic planning and advanced dynamic motion generation. Within the classification proposed in [18], our two-stage approach fits alongside state-space and sensor-based approaches. In the first stage, a "path" is generated by geometric and kinematic planning, which is transformed into a dynamically executable "trajectory" through appropriate dynamic motion generators in the second stage. Due to dynamic effects, the path supporting the new trajectory may differ slightly from the initial path. Then the output trajectory is again verified with respect to the collision avoidance by the first stage and reshaped if necessary. This process is iterated until a valid dynamic trajectory is obtained.

Although the necessity of reshaping for dynamic collision-free motion has been recognized [19], it has not been systematically addressed. Our method is inspired

by a technique for key frame editing in the context of computer animation [8]. This approach places more emphasis on the gradual transition from the colliding trajectory by maintaining the motion timing in constrained environments. In this work we also emphasize the practical aspect of our approach through realistic simulations and experiments.

2.1 Two-stage planning framework

The proposed general framework of dynamic collision-free motion planning based on the two-stage planning method is illustrated in Fig. 2. It outputs dynamic collision-free trajectories from the inputs of initial and goal configurations together with the geometry information of the environment. The resulting dynamic trajectories should be ready to be given to low-level robot controllers.

The path planner finds a geometric and kinematic collision-free path in 3D at the first stage (upper part of Fig. 2). Any available planning method can be used for this part. We utilize PRM in our case for the first stage. Then in the second stage, the dynamic motion generator to transform the given path into dynamically executable robot trajectory (lower part in Fig. 2). A dedicated dynamic controller can be put depending on the application.

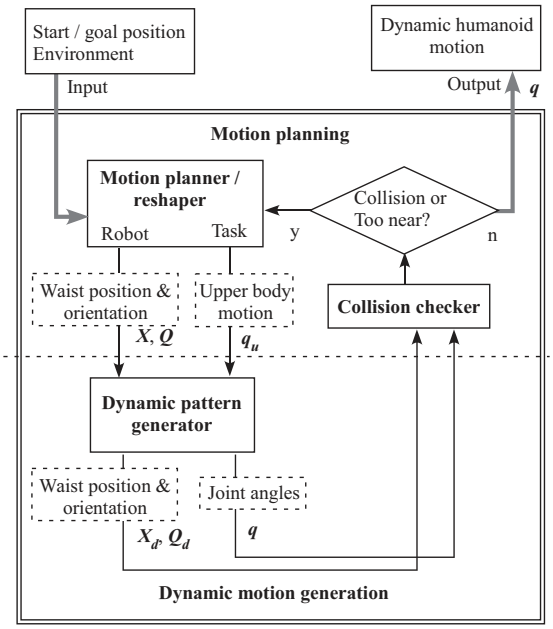


Fig. 2 Two-stage motion planning framework. In the first stage the geometric and kinematic planner plan the collision-free path that is transformed into dynamic motion in the second stage. If collisions are detected the path is sent back to the first stage. This process is repeated until a collision-free dynamic trajectory is obtained.

The generated trajectory may deviate from the planned path due to robot dynamics, which may cause unpredicted collision with obstacles. The reshaper is placed in the first stage as a mechanism that interacts with the dynamic motion generator iteratively to remove those local collisions. Practically, if the collisions are detected in the dynamic trajectory, the colliding portion is locally deformed by increasing the "tolerance" of the obstacle for the robot to move away from the obstacles.

If the dynamically colliding local paths become blocked by the "grown" obstacles, a replanning process is activated. In this process, the path planner searches for another path that avoids the blocked passage.

We utilize here a functional decomposition of the robot body that has already been applied to motion planning for virtual mannequins [6]. At the first stage, the robot is modeled as a geometric parallelepiped bounding box (Fig. 3). Only that box and the object to be manipulated are considered with respect to collision avoidance. The robot motion is expressed by the planar position and orientation of its waist $r(x, y, \theta)$ (3 DOF) and the object motion by its position and orientation $R_o(x_o, \Theta_o)$ (6 DOF) with respect to a global coordinate system Σ_0. The configuration space to be searched is then 9-dimensioned.

In our case, the robot path is planned as Dubins curves composed of line segments and arcs of a circle [5]. Given the configuration of the robot waist and object, the joint angles (q_u) of the upper-body motion are derived by using inverse kinematics described later.

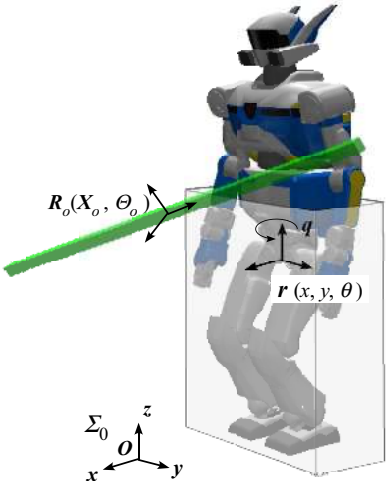


Fig. 3 Humanoid modeled by rectangle box with a bar. In the first stage the geometric and kinematic path planner generates collision-free path for the 9 DOF system including robot waist (r, 3DOF) and object (R_o, 6DOF).

2.2 Second Stage: Smooth Path Reshaping

Then at the second stage, the planned motions r and q_u are given to the dynamic pattern generator [13] of humanoid robots to transform the input planar path into a dynamically executable motion. The walking pattern generator is based on preview control of zero moment point (ZMP) proposed by Kajita et. al [13]. The reference ZMP trajectory is derived from the foot placements obtained from the planned planer robot path. Based on preview control of ZMP position for an invert pendulum model, this method is able to generate dynamically stable biped walking motion that always maintains the ZMP inside the support polygon formed by the foot (feet). Moreover, the pattern generator can combine upper-body motion q_u as auxiliary input to compute the mixed whole-body motion.

If collisions are found within the upper part of the body, the following reshaping procedure is applied. After identifying the endpoints each colliding portion, a collision-free configuration is found in a free space within a nearby reachable area after increasing the "tolerance" that grows the obstacles. All the configurations within the portion are then replaced by this collision-free configuration. Next, anticipation and regaining motions are computed to smoothly connect the reshaped portion with the collision-free part of the original trajectory. Finally, inverse kinematics (IK) is applied to satisfy the constraints of the hands at each sample of the reshaped trajectory that synchronizes the upper body task with the lower body motion. As a result, this reshaping eliminates the collision locally as shown in Fig. 4.

We have applied the proposed method to plan a motion to carry a bulky object in an environment with several obstacles as shown in Fig. 5. The proposed method is implemented as an off-line planner on the assumption that the environment is completely known. In this case, what matters is not the weight of the object but its geometric complexity. Figure 6 shows the experimental results of the planned motion.

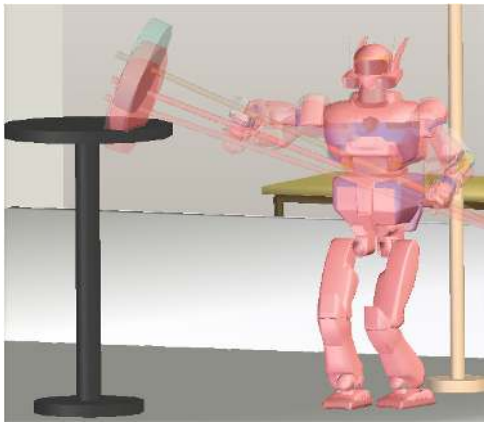


Fig. 4 Transition of robot configurations during the reshaping. The colliding part of the carried object goes away from the obstacle by increasing tolerance.

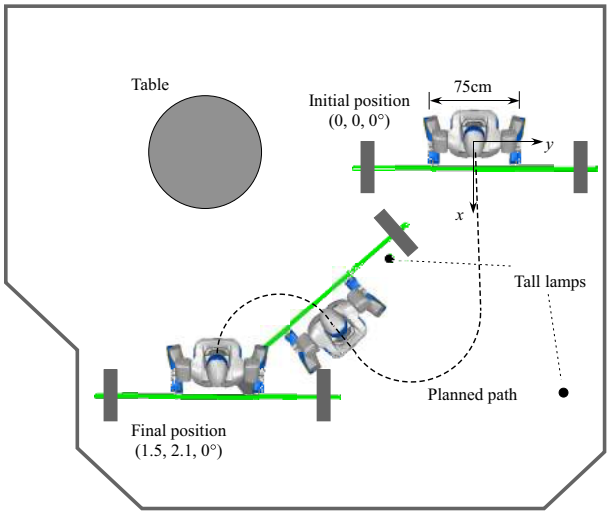


Fig. 5 Top view of the simulation and experiment environment with two poles and a table. The initial and final configurations of the robot are also shown.

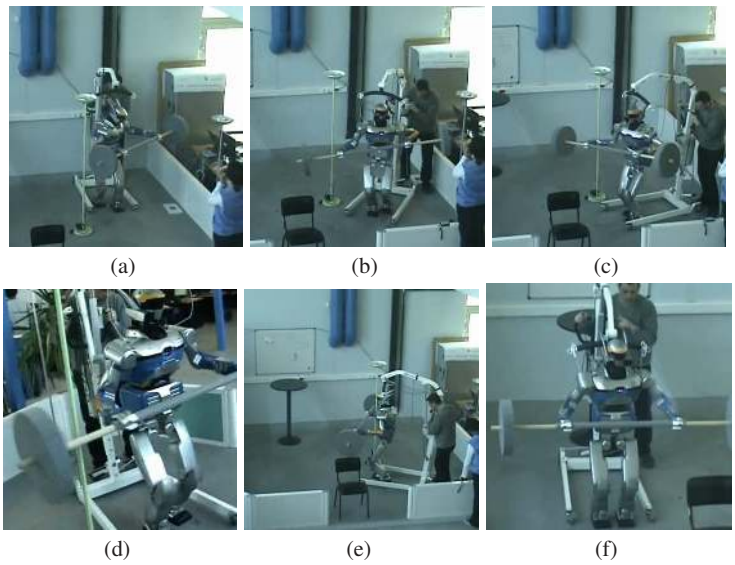


Fig. 6 Experiment of 3D collision-free motion for bar-carrying task at JRL-France

Since the distance between the two lamps is shorter than the bar length, the bar should pass through with an angle. At the beginning of the motion, the computed trajectory for the bar makes the robot move to the left, then walk forward with a

certain angle to path through the gap (Fig. 6a,b). Here the motion of the upper part of the robot is computed using a generalized inverse kinematics and the chest is moved consequently to complete both tasks.

This example also shows that the complete 3D geometry of the object has been considered in the collision-detection and path planning procedure and no bounding box has been used (see Fig. 6d) where the concave part of the disk and the bar is close to the lamp. The complete trajectory execution time is around 28 sec.

3 Reaching: Generalized Inverse Kinematic Approach

We here address the problem of how to re-position the humanoid body when performing reaching or grasping tasks for a target far away. The proposed method is based on reshaping the support polygon of the humanoid robot to increase its workarea by coupling generalized inverse kinematics and dynamic walking pattern generator [35]. While using inverse kinematics, the global motion is guaranteed to be dynamically stable. Such a property is a direct consequence of ZMP control provided by the pattern generator we use.

The generation of whole-body dynamic motion is closely related to motion planning. Whereas the motion planning takes charge of global plan from initial to goal configurations, a whole-body motion generation concerns how to make valid local motions by taking account of several constraints. So it is important in order to create feasible dynamic trajectories from motions that have been provided by the global motion planner.

Khatib and his colleagues have been working on dynamic motion generation for humanoid robots by using task specification in operational space approach [27]. In their work a hierarchical controller synthesizes whole-body motion based on prioritized behavioral primitives including postures and other tasks in a reactive manner. Kajita et al. proposed a "resolved momentum control" to achieve specified momentum by whole-body motion [14]. Mansard et al. [24] proposed a task sequencing scheme to achieve several tasks including walking and reaching at the same time.

Figure 7 illustrates the proposed motion generation framework with an example of a reaching task [35]. Priorities are given to the target task as well as to other tasks such as the position of center of mass (CoM). We employ generalized inverse kinematics to generate a whole-body motion for those tasks based on the given priorities [25]. During the motion, several constraints are monitored which are expressed by such measures as manipulability for whole-body, end-effector errors from target, or joint limits.

If the task cannot be achieved because those monitored constraints are not satisfied, a reshaping planner of support polygon is activated automatically to increase accessible space of the robot, keeping the inverse kinematics working to achieve the tasks. The reshaping is performed based on geometric planning to deform the support polygon in the direction required by the specified task. Thanks to the usage of free-floating base, the changes in support phase can be easily integrated in the

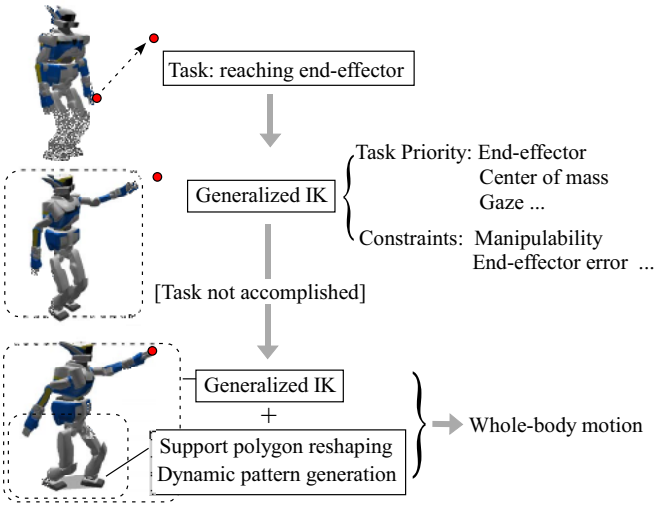


Fig. 7 A general framework for task-driven whole-body motion including support polygon reshaping [35]. If the desired tasks cannot be achieved, support polygon is reshaped to increase the workspace.

computation. As a result, the stepping motion is generated using a biped walking pattern generator [13] and the blended whole-body motion including the target task is recalculated.

Our contribution is to consider the possibility of reshaping the support polygon by stepping to increase the accessible space of the end-effectors in the 3D space. Our approach makes use of the whole body 3D space as opposed 2D in [39]. Moreover, in spite of our reasoning being based on inverse kinematics and simple geometric support polygon reshaping, our method guarantees that the motion is dynamically stable. This property is a consequence of the pattern generator [13] we use to generate the stepping behavior.

3.1 Method overview

The support polygon reshaping integrates two important components, the generalized inverse kinematics and dynamic walking pattern generator. The former provides a general way to deal with the whole-body motion generation to perform the prioritized tasks. The latter takes charge of the stepping motion to change the foot placements.

Figure 8 shows an overview of the method. The task is specified in the workspace as $\dot{x}_j$ with priority j from which the generalized IK solver computes the whole-body motion as joint angles $\dot{q}$ of the robot. Meanwhile, several criteria such as manipu-

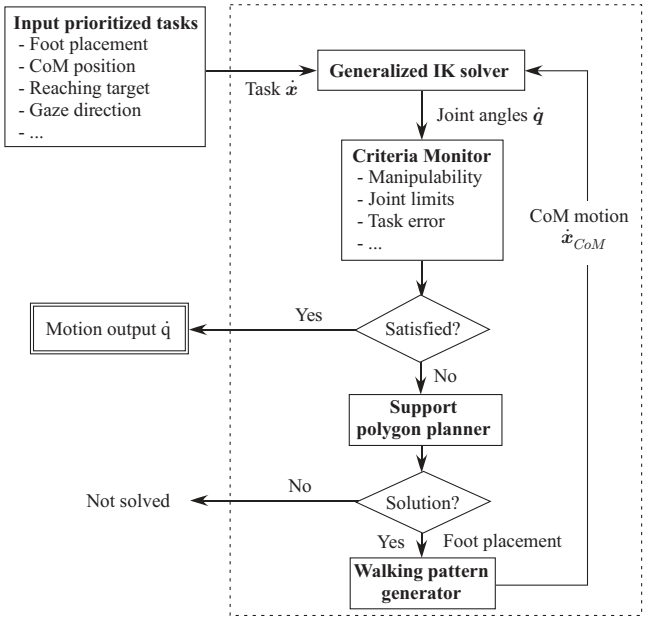


Fig. 8 Method overview of the whole-body motion generation. Once the support polygon is reshaped, the dynamic stepping motion is computed as the CoM ($\dot{x}_{CoM}$) from the foot placement and again given to the generalized IK solver.

lability or joint limits are monitored if they do not prevent the desired whole-body motion.

As long as the criteria are satisfied, the computation of whole-body motion continues until the target of the task is achieved. If the task cannot be achieved due to unsatisfied criteria, the support polygon planner is triggered in order to extend reachable space. A geometric module determines the direction and position of the deformation of support polygon so that the incomplete task is fulfilled. The position of a foot is then derived to generate the motion of CoM $\dot{x}_{CoM}$ by using a dynamic walking pattern generator [13].

Using this CoM motion, the original task is then redefined as the whole-body motion including stepping that is recalculated using the same generalized IK solver.

The generalized IK solver benefits from the redundancy of the mechanism to choose the solution that best solves the task according to some constraints. Among these works, inverse kinematics algorithms that project tasks with lower priority into the null space of the Jacobian of the higher priority tasks have been widely studied (e.g., [3, 25, 28, 32]).

3.2 Generalized inverse kinematics for whole-body motion

3.2.1 Inverse Kinematics for Prioritized Tasks

Let us consider a task $\dot{x}_j$ with priority j in the workspace and the relationship between the joint angle velocity $\dot{q}$ is described using Jacobian matrix, like $\dot{x}_j = J_j\dot{q}$. For the tasks with the first priority, using pseudoinverse $J_1^{\#}$, the joint angles that achieves the task is given:

$$\dot{q}_1 = J_1^{\#}\dot{x}_1 + (I_n - J_1^{\#}J_1)y_1 \tag{1}$$

where y_1, n and I_n are an arbitrary vector, the number of the joints and identity matrix of dimension n respectively.

For the task with second priority $\dot{x}_2$, the joint velocities $\dot{q}_2$ is calculated as follows [25]:

$$\begin{aligned} \dot{q}_2 &= \dot{q}_1 + \hat{J}_2^{\#}(\dot{x}_2 - J_2\dot{q}_1) + (I_n - J_1^{\#}J_1)(I_n - \hat{J}_2^{\#}\hat{J}_2)y_2 \\ &\text{where} \quad \hat{J}_2 \equiv J_2(I_n - J_1^{\#}J_1) \end{aligned} \tag{2}$$

where y_2 is an arbitrary vector of dimension n. It can be extended to the task of j^{th} ($j \geq 2$) priority in the following formula [3, 28].

$$\begin{aligned} &\dot{q}_j = \dot{q}_{j-1} + \hat{J}_j^{\#}(\dot{x}_j - J_j\dot{q}_{j-1}) + N_j y_j \\ &N_j \equiv N_{j-1}(I_n - \hat{J}_j^{\#}\hat{J}_j),\ \hat{J}_j \equiv J_j(I_n - \hat{J}_{j-1}^{\#}\hat{J}_{j-1}) \end{aligned} \tag{3}$$

3.2.2 Monitoring Task Execution Criteria

While the motion is being computed by the generalized IK, several properties are monitored.

One of the important measures is the manipulability [40] defined as:

$$w \equiv \sqrt{\det\{JJ^T\}} \tag{4}$$

This measure is continuously tracked during the motion generation as well as others such as joint angle limits or end-effector errors from the target. If it becomes below a certain value, it means that it is difficult to achieve the task.

Joint limit constraints can be taken into account by introducing a selection diagonal matrix $S = \text{diag}\{S_1, \ldots S_n\}$ ($S_i = 0$ or 1) to be multiplied to Jacobian to select the activated joints if the corresponding joint reaches a limit angle. The selection matrix is I_n if all the joints are used to achieve the task.

As shown in Figure 8, when one or more monitored measures go out of the admissible range to prevent the task from being achieved, the support polygon reshaping is launched to extend the accessible space as detailed next.

3.2.3 Support polygon Reshaping

Figure 9 shows the proposed support polygon reshaping scheme. This simple algorithm allows the humanoid robot to make a step motion, keeping a large margin of accessible area for the task by facing the upper body to the target direction.

Then the CoM motion $\dot{x}_{CoM}$ is computed from the new foot position by the walking pattern generator based on the preview control of ZMP [13]. The basic idea is to calculate the CoM motion by anticipating the desired future ZMP positions derived from the footsteps.

Finally the original task is redefined as another problem of whole-body task using this newly generated CoM motion with an additional task of CoM, which is represented by CoM Jacobian [29]. The same generalized IK solver framework is used to incorporate the motion required for the task and the stepping motion in the whole-body level.

The manipulability measure of the arm during the forward reaching task is provided in Fig. 10. Without reshaping, the arm approaches singular configuration where the manipulability becomes lower than the threshold at 2.3[s] and the computation keeping the same support polygon is discarded. The reshaping starts at this moment to recalculate the overall whole-body motion including stepping motion. We can see the manipulability regains higher value at the final position.

3.3 Results

We have conducted experiments of the generated motion using a humanoid platform HRP-2 for front and sideways reaching tasks that requires stepping as shown in Fig 11. As can be seen, the robot successfully performed the desired reaching task through whole-body motion that unifies reaching task and stepping motion by keeping dynamic balance. Note that the tasks of keeping gaze direction towards

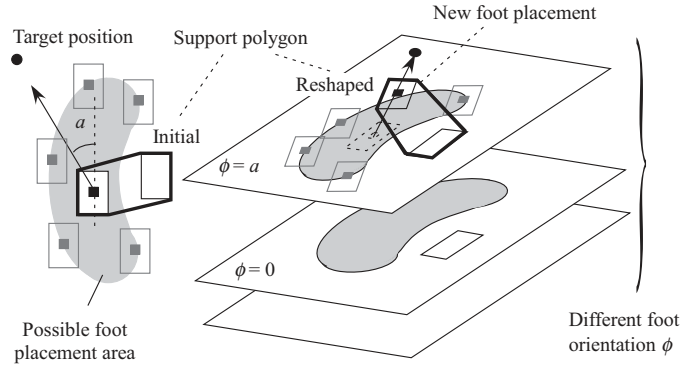


Fig. 9 Support polygon reshaping method

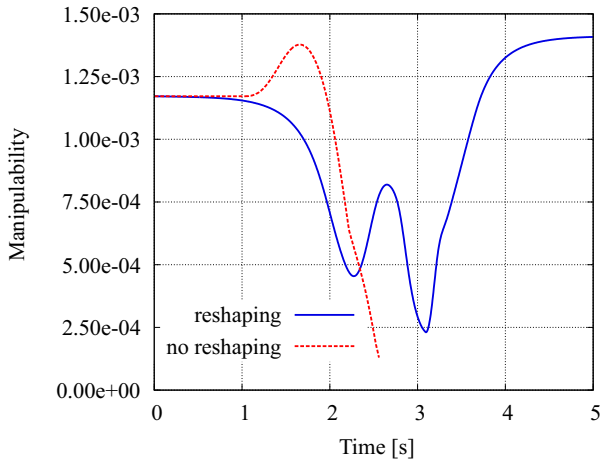


Fig. 10 Manipulability for front reaching. Without support polygon reshaping, the manipulability measure decreases below the threshold. Although it also decreases with reshaping, the manipulability increases in the course of stepping motion.

Fig. 11 Experimentation of front reaching task. The robot goes through a posture that is not statically stable (b) to finish stepping in (c). The final goal of the end effector is achieved at (d). The gaze direction is always maintained in the direction of the end-effector goal.

the end-effector target position are taken into account in this experiment. The final CoM apparently goes out of the initial support polygon: this means the reaching task could not have been performed without stepping.

4 Manipulation: Pivoting a Large Object

Manipulation requiring whole-body motion is one of the tasks that are appropriate for humanoid robots. In order to manipulate cumbersome object, humans often manipulate without lifting but by using the contact with the ground (Fig. 12). In this research, we apply such a manipulation method to humanoid robots.

There are several task-specific whole-body motions that have been intensively investigated: pushing [12, 9, 31], and lifting [10], and pivoting [33, 37]. Currently, many researchers are intensively working to integrate those recent developments with global motion planner.

Among them, the pivoting manipulation has several advantages such as precise positioning, stability and adaptability over other methods like pushing or lifting. For those reasons, pivoting based manipulation has potential of widening the capacity of manipulation of humanoid robots.

We introduce here a whole-body motion planner that allows a humanoid robot to autonomously plan a pivoting strategy that accounts for the various constraints: collision avoidance, legs-arms coordination and stability control.

The motion planning algorithm we propose consider a two-stage approach: a first collision-free path is computed, and then it is iteratively approximated by a sequence of pivoting motions.

4.1 Pivoting and small-time controllability

The robot starts inclining the box to realize a single contact point between the box and the floor. The contact point is a corner of the box. Then the robot performs a rotation of the box around the vertical axis on that corner. Then it sets the object horizontally along the box edge. Such an edge is said to be the supporting edge. We here model the problem of 3D box pivoting as the problem of pivoting a 2D segment around its endpoints.

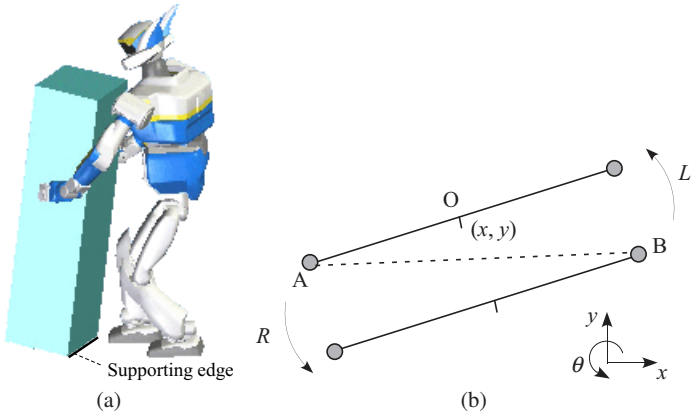


Fig. 12 Supporting edge and pivoting problem modeling. (a) The pivoting sequence is planned using rotation of the endpoints of this edge. (b) The 3D pivoting problem is reduced to how to displace a line segment on vertices A or B.

A system is said to be controllable if it may reach any arbitrary configuration q_{goal} from any other q_{init} [30]. It is said to be small-time controllable if the set of admissible configurations $Reach_q(T)$, which can be reached from configuration q before a given time $T(>0)$, contains a neighborhood of q. This property should hold at any configuration q for any T. It means that the system can move anywhere in the area η without leaving an imposed neighborhood V as shown in the left of Fig. 13.

Small-time controllability is a critical property in path planning. The main consequence is depicted on the right side of Fig. 13: any collision-free path can be approximated by a sequence of both collision-free and admissible motions as follows. Starting from the initial configuration q_{init}, take any collision-free neighborhood V_1. Then the system can advance to a configuration q_1 on the path within $Reach_{q_1}(T)$ without going out of V_1. The same procedure is repeated until the system reaches the goal configuration q_{goal} (Fig. 13). This type of analysis plays an important role in nonholonomic motion planning [20].

We have proven that the considered pivoting system is small-time controllable by using Lie Algebra Rank Condition (LARC) [30]. First, the vector field of the motion in the space (x, y, θ) is considered for the rotation motions R and L turning around the corner A and B respectively. Then the Lie Bracket $[L, R]$ is computed to demonstrate that the three vector fields L, R and $[L, R]$ span a three-dimensional space. For details, the readers are referred to the reference [37].

4.2 Collision-free pivoting sequence planning

We here take into account the naturalness of the targeted solution: we want the robot to walk either forward or backward and to avoid sideways steps. When walking forward or backward the robot direction remains tangent to the path it follows as a wheeled mobile robot does. Such constraints are known to be nonholonomic. It has

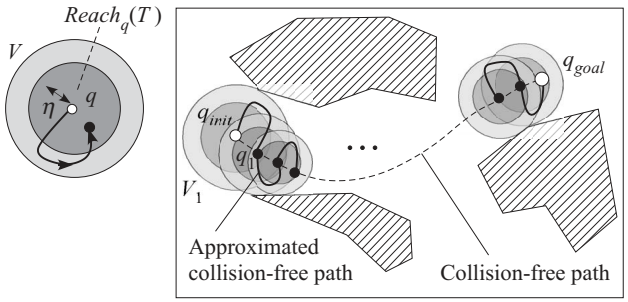


Fig. 13 A system is small-time controllable from q if $Reach_q(T)$ contains a neighborhood of q for all neighborhoods V for any time $T > 0$.

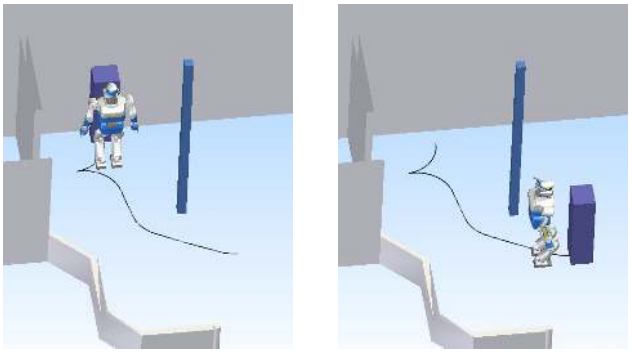
been recently demonstrated that they account for the natural human locomotion [2]. Our motion planning point of view then benefits from well experienced approaches in nonholonomic motion planning [4, 20, 22]. Among them we have chosen the probabilistic sampling approach with a steering method computing Reeds and Shepp curves [26], composed of arc of a circle and straight line segments. Reeds and Shepp curves possess a geometric property accounting for small-time controllability, a critical property for the planning method completeness.

By applying the general scheme composed of collision-free probabilistic motion planning, and path optimization at the first stage, we can obtain a path for the bounding volume shown in Figure 14.

The manipulated object is placed near the wall and supposed to be displaced on the other side of an obstacle. As can be seen, the backward motion of Reeds and Shepp curve is utilized appropriately to move the object away from the wall. Then the path switches to forward motion to reach the goal by avoiding the obstacle.

The collision-free path computed at the first stage should be converted into a sequence of collision-free pivoting sequences. The pivoting sequence generation is then based on two elementary operators: pivoting along a straight line segment and along an arc of a circle to follow Reeds and Shepp curves.

The computation of the pivoting sequence along a straight line segment is illustrated in Fig. 15. Let D be the length of the straight line segment of the path to follow. As defined earlier, the length of the supporting edge is $2l$. Considering the constraint of the reachable area of robot arms, we introduce an angle β such that the robot is able to perform an elementary pivoting motion of total angle 2β. After initializing the process by a pivoting of angle β, we then apply N times the elemen-



(a) Initial state (b) Goal State

Fig. 14 Optimized collision-free path for a manipulated box object and the humanoid robot using Reeds and Shepp curves. The path allows the humanoid to move the object away from the wall starting from the initial state (a) by taking advantage of backward motion. Then the path switches to forward motion to avoid obstacle and to move the object to the goal (b).

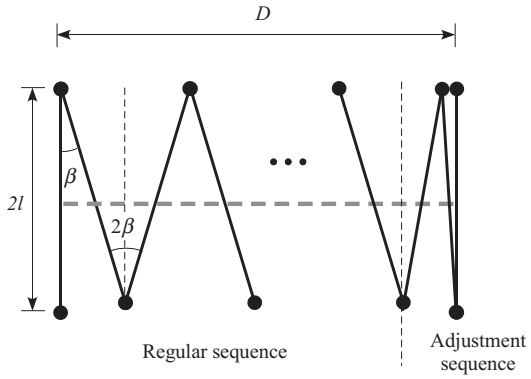


Fig. 15 Transforming a straight line segment path into a pivoting sequence. The pivoting sequence is planned using rotation of the end-points of the supporting edge. During the regular sequence, rotations of same angles are repeated before adjustment sequence that positions the line segment at the endpoint.

tary pivoting motion of angle 2β, N being defined as the greater integer verifying $D > 2Nl\sin\beta$. The same principle applies to the arcs of a circle.

We should notice that the rotation angle β may be tuned for obstacle avoidance purpose. Indeed, the first stage of the algorithm provides a collision-free path that guarantees collision-freeness for the sliding supporting edge. As this rotation angle decreases, the final swept volume of the generated pivoting sequence converges to initial one swept by the supporting edge when sliding along the Reeds and Shepp path. This property accounts for the small-time controllability of the pivoting system we have considered in Section 4.1. The 3D collision detection can be done by estimating the swept volume of the object attached to the supporting edge during the rotational motion.

As a consequence, the two-stage strategy we have developed inherits from the probabilistic completeness of the motion planner used at the first stage. The approximation scheme by on pivoting sequence generation does not introduce any incompleteness thanks to small-time controllability.

4.3 Whole-body motion generation and experiments

The Reeds and Shepp curve and the pivot sequence generation are implemented as problem-specific steering methods shown in Fig. 1. After the pivoting sequence is generated, it should be realized by the humanoid robot by using its two arms. The humanoid motion should be generated in such a way that constraints like dynamic balancing and arm manipulation motion are satisfied at the same time. Moreover, stepping motion should be added in order to continue the manipulation when necessary.

For this purpose we adopt the dynamic whole-body motion generation in Section 3. Since all the joints are involved to make those complicated combined mo-

tions, we can expect better performance in the sense of reachable space than a functional decomposition utilized in [33].

The method is illustrated in Fig. 16. There is a motion manager that a whole-body motion manager receives the desired hand trajectories and keeps track of the current robot configuration. Then it computes the trajectories or constraints that are supplied to the generalized IK solver. The solver is also given trajectories of feet or CoM as well as their position and orientation constraints as prioritized tasks. With those inputs, the generalized IK solver computes the whole-body motion as joint angle trajectories.

When the pivoting rotation requires large horizontal displacement, a stepping motion is planned at the same time of the hand motion. The stepping foot is determined depending on the rotation direction. Then the new foot position is computed in such a way that the foot keeps its orientation in parallel with the base Reeds and Shepp curves with an appropriate distance to the object. The same dynamic whole-body motion generator presented in Section 3 is applied to compute the coordinated arm and foot-stepping motions.

We have conducted the experiments with the humanoid robot platform HRP-2 in the environment shown in Fig. 14 for whole-body motion planning for pivoting of a box-shape object.

The execution time of the entire pivoting sequence is 281 seconds, which corresponds to 56200 command outputs at the rate of 5ms for each of 30 joints. The

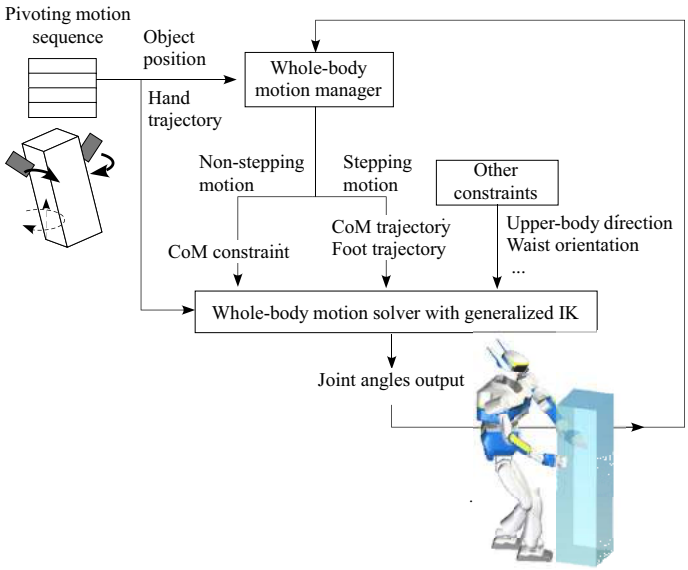


Fig. 16 Usage of generalized inverse kinematics utilized for whole-body motion for pivoting based manipulation.

computation time was 291.7 seconds with a PC of Intel Core2 Duo CPU at 2.13GHz, which is comparable to the actual task execution time.

The experimental results are shown in Fig. 17 to validate the proposed method. The motion has been planned offline with the prior knowledge of the object and environment. The humanoid robot executes the complex pivoting manipulation with a coordinated whole-body motion including simultaneous manipulation and foot-stepping. As can be seen, the robot could accomplish the long pivoting sequence.

The average error of the final position of the carried object was 0.46m (0.31m and 0.35m short in x and y directions respectively), and 5° in orientation θ. The error 0.46m represents 9% of the length 5.1m of the whole trajectory of the carried object. This confirms that the manipulation has been executed relatively accurately considering the lack of sensor feedback of the object location during the manipulation.

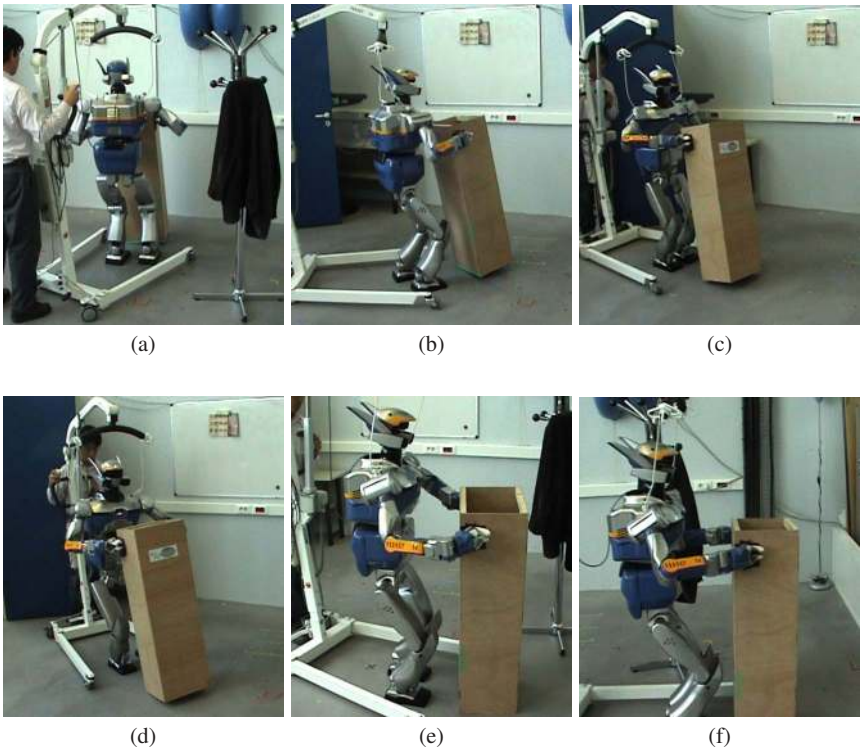


Fig. 17 Experimental results. Starting from the initial position (a) with obstacle at right-hand side, the humanoid robot manipulates the object backwards away from the wall (b). After switching motion direction to forward (c), the robot continues to manipulate the object to the goal position by avoiding the obstacle (d-f).

4.4 Regrasp planning

So far we developed the whole-body manipulation method on the assumption that the robot can carry the object without changing the grasping points. However, when there are narrow spaces, the robot should sometimes release the object and hold it with another position according to the situation.

We here provide a humanoid robot with more flexibility in whole-body pivoting manipulation by including regrasp planning. The robot releases the object when it cannot go further towards the goal position and grasp it again to continue manipulation.

The difficulty resides in finding narrow passages for the robot and object together and in combining the paths with different grasping positions to plan a manipulation motion to achieve the goal. We here address the regrasp planning problem for pivoting manipulation through a roadmap-multiplexing approach [38].

Fig. 18 illustrates the overview of the planning scheme. Several grasping positions are possible for a given object position. The roadmaps are built for the combined bounding volumes of both the robot and the object. We suppose that there are different grasping positions that allow the robot to hold the object. There are two types of roadmap: the first is the "manipulation roadmap" $\mathcal{G}^i_{manip}$ for i-th grasping position r^i_{grasp} expressed as the relative robot position with respect to object. In this roadmap, the robot and the object move together. The second type of roadmap is the "regrasping roadmap" $\mathcal{G}_{reg}$ where the robot moves alone between different grasping positions.

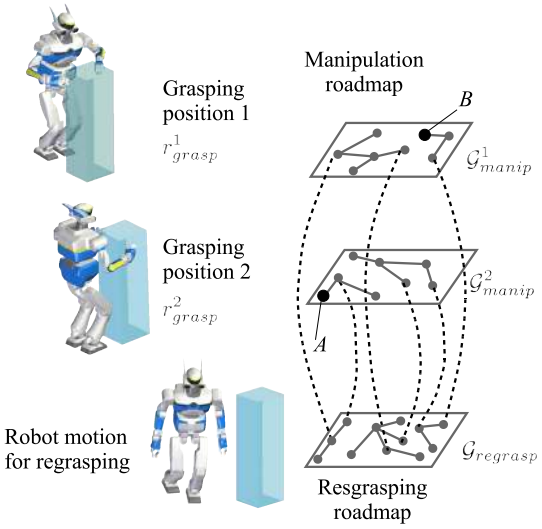


Fig. 18 Roadmap multiplexing. Different manipulation roadmaps $\mathcal{G}^1_{manip}$ and $\mathcal{G}^1_{manip}$ are connected by way of the regrasping roadmap $\mathcal{G}_{reg}$.

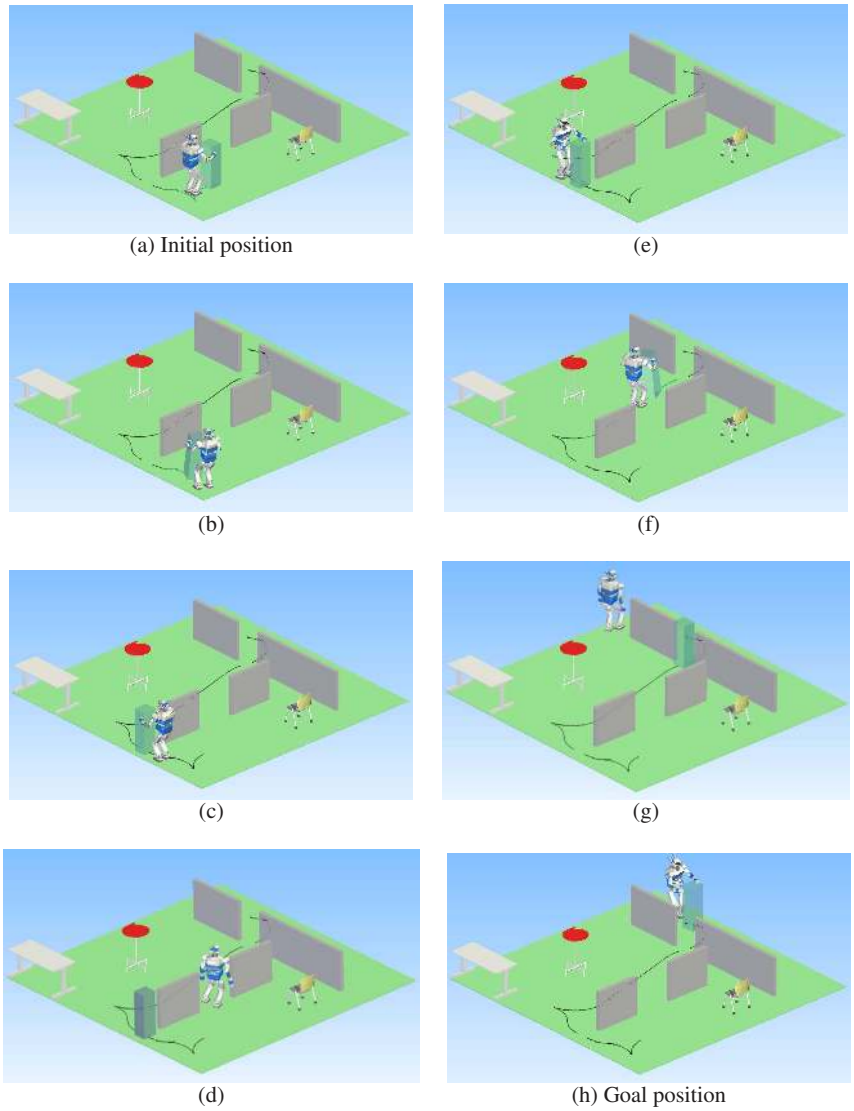


Fig. 19 Simulation result of regrasp planning. Starting from initial position (a), the humanoid robot makes pivoting sequences (b) first puts the object to the entry of passage (c). It leaves the object and walks freely by combining forward and sideways walking (d) to regrasp the object on the other side (e). Then the robot goes towards another narrow passage (f) and make another regrasp sequence (g) to arrive at the goal (h).

As can be seen in the figure, manipulation roadmaps $\mathscr{G}^1_{manip}$ and $\mathscr{G}^2_{manip}$ for different grasping positions are interconnected via the regrasping roadmap $\mathscr{G}_{reg}$. For

instance, the path from *A* to *B* is possible only by alternating the different grasping positions.

Figure 19 shows the result of regrasp planning. The humanoid robot HRP-2 should carry a box-share object from the initial position (Fig. 19a) to its goal (Fig. 19h). The humanoid robot displaces the object at the entry of a narrow passage (Fig. 19b, c). Then it releases the object and walk to the other side of the wall (Fig. 19d). By combining backward and forward manipulation, the humanoid goes to another narrow passage (Fig. 19e, f). After another regrasping, the object is carried to the goal position (Fig. 19g, h).

5 Motion in Real World: Integrating with Perception

In this section the presented motion planning methods are integrated with perception, principally vision, to make actions in the real world. This integration allows the robot to execute such commands as "go to the yellow table" and "take the orange ball."

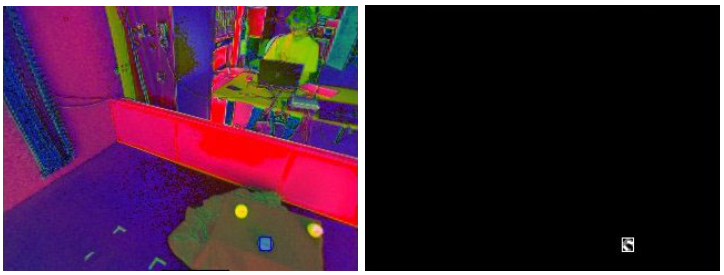
5.1 Object recognition and localization

The HRP-2 robot is equipped with two pairs of firewire digital color cameras, configured as two independent stereo-vision camera pairs. We here utilize standard state of the art components to implement a simple function of object recognition and localization.

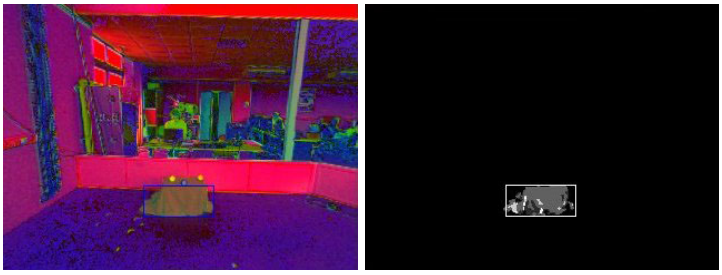
For the detection, the model of the objects to be detected are previously learned using two dimensional histogram in the $\{Hue, Saturation\}$ color space by taking a sample image with a color space.

The object detection is performed by *back projecting* the object histogram onto a video image. The back projection image is obtained by replacing each $\{H, S, V\}$ pixel value by the corresponding value in the histogram, leading to a probability image where each pixel value is the probability of that pixel to belong to the object model. The *Continuously Adaptive Mean SHIFT CamShift* algorithm then locates the object center and orientation in the back projection image.

A stereo-vision algorithm by pixel correlation is applied on the stereo image pairs, and produces a dense three dimensional image of the current scene. Even though pixel correlation is known to give poor results in indoor environments, the objects to localize are sufficiently textured so that precise enough 3D points can be obtained in the vicinity of the objects.



Detection of the object



Detection of the table

Fig. 20 Table and Object detection. Left images show the HSV image and right images are the back projection of the table color model in the source image. Rectangle is the result of the execution of the CAMSHIFT algorithm on the back projection image.

5.2 Coupling the motion planner with perception

The motion planners presented in previous sections are integrated with the vision system so that the robot can execute a task composed of navigation and object grasping.

For navigation, we apply the same type of two-stage motion planner for navigation planning presented in Section 2. At the first stage, a collision-free smooth locomotion path is calculated for the approximated bounding box. It is desirable for the robot to walk forward in order to look at the object and to take it. This preference can be modeled as a nonholonomic constraint in the same way as in Section 2 and 4 and we can benefit from well-developed planning method of a smooth path for car-like robot [20]. Then the path is transformed into dynamic humanoid locomotion at the second stage by applying the dynamic walking patter generator in the same way as in Section 2. This navigation planner allows the humanoid robot to go in front of the visually located colored table several meters away by avoiding known obstacles as shown in Figure 21.

Finally, the whole-body motion generator presented in Section 3 is used for the grasping task. Given the object location from the vision system, the whole-body

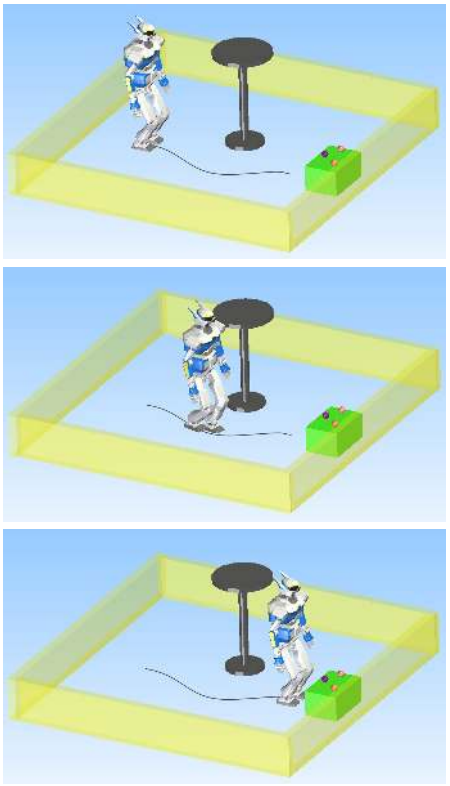


Fig. 21 A planned smooth walking trajectory to a target position

motion generator computes automatically a reaching motion, including stepping depending on the detected object location if necessary.

All the software of perception, motion planning, dynamic motion generation, and controller is installed on the computers on board. In order to build the necessary software components, we used the standard LAAS control architectures tools. In particular, we used the GenoM [7] tool that is able to generate robotics components. GenoM components can encapsulate C or C++ source code into an executable component that provides requests that can be invoked through simple scripts or through more complex supervision software. The components can also be dynamically interconnected together at run-time, providing a modular and programmable control architecture.

Figure 22 shows a subset of important components that have been defined for the experiment. All the components but the real-time control (OpenHRP [15]) runs on a Linux 1.8 GHz Pentium-M processor. The real-time part is operated by Art-Linux [1] on a similar hardware.

The vision processing chain is made up of three components: image acquisition (`camera`), stereo-vision by pixel correlation (`stereo`) and object detection and localization (`hueblob`). The two motion-related software components for navigation

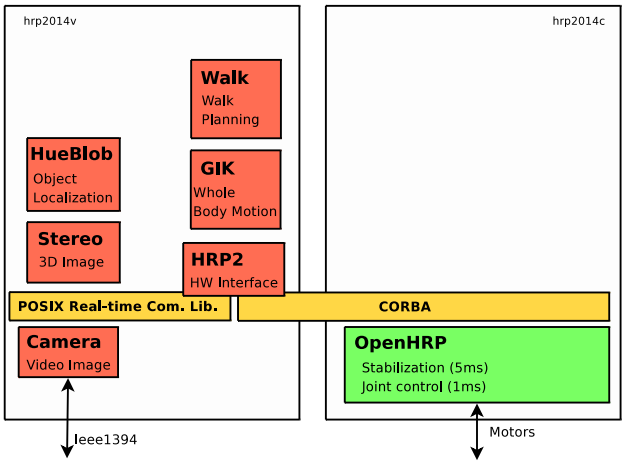


Fig. 22 Software components running onboard the robot.

and whole-body motion generation are implemented as components `walk` and `gik` respectively. Finally, an interface component (`hrp2`) makes the connection with the OpenHRP software and bridges the CORBA communication bus of OpenHRP to the GenoM communication bus (*Posix Real-Time communication library* on Figure 22).

All these components define requests that can be invoked by a human operator, by supervision software or by the natural language processing system.

5.3 Experiments

We have conducted experiments to validate the integrated system. The humanoid robot is given a task to take a colored ball and put it at another place. The task is decomposed into several generic action commands, such as detection and localization of a learned object, locomotion to a location, and hand reaching to a position in 3D, with other simple tasks like turning on the spot and gripper opening and closing.

A simple supervision system that can invoke the actions with scripts is utilized to manage the robot behavior easily. Each action can report failures (*e.g.* failure in grasping an object). It is thus possible to implement error recovery strategies by analyzing the reports of the actions. In the following experiment, each action is associated with a vocal command to allow the user to give a sequence of commands to the robot in an interactive manner.

Figure 23 shows snapshots of experiments. Since the ball is too far away to be detected with camera at the initial position, the humanoid robot first localizes the green box on which the balls are placed (Fig 23a). The robot walks with a smooth trajectory in front of the box (Fig 23b) and localizes precisely the colored ball to grasp (Fig 23c). Then the whole-body reaching motion is executed to grasp the ball

(a) Localization of the box (b) Walking to the box

(c) Detection and localization of the ball (d) Whole-body reaching for grasping

(e) Locomotion to another location (f) Putting the ball on the detected table

Fig. 23 Ball-fetching task using visual perception and motion planner

(Fig 23d). After turning, the robot is told to detect a colored table and walks towards it always with a smooth trajectory (Fig 23e). Finally it puts the ball on the table again with whole-body motion (Fig 23f).

This experiment was conducted more than ten times in an exposition in front of the public using vocal interaction by a human operator. Since the location of the robots and objects are different at every demonstration, it happened that the robot failed to grasp with unexpected disturbances or localization errors. However, the task could be executed again successfully thanks to the generality of the action commands, by just repeating the same action command. As a result, all the demos were successful including those retries. This validates the reliability of the proposed motion planner, the integrated perception system and also the robustness of task execution framework.

6 Conclusion

In this article, we have presented research results on whole-body motion planning from several aspects: collision-free locomotion with upper-body motion, whole-body reaching and manipulation. To cope with the redundancy and underactuation of humanoid robots, we have integrated probabilistic motion planning and dynamic motion generation of humanoid robots. The results have been validated by the humanoid robot platform HRP-2.

In our future developments, we keep pursuing our aim of improving the autonomy of the humanoid robots by increasing the variety of their motions and thus by enriching the possible behaviors. Besides this motion autonomy, we will also have to address the reactivity in the real environments. In the last part of this article we demonstrated an example of closed perception-behavior loop. However, fast and precise environment recognition as well as reactive motion planning scheme still needs to be investigated for the humanoid robot to adapt to more complex situations. We will address those problems in our future work.

References

1. ART linux. `http://sourceforge.net/projects/art-linux`
2. Arechavaleta, G., Laumond, J.P., Hicheur, H., Berthoz, A.: An optimality principle governing human walking. IEEE Trans. on Robotics **24**(1), 5 – 14 (2008)
3. Baerlocher, P., Boulic, R.: An inverse kinematics architecture enforcing and arbitrary number of strict priority levels. The Visual Computer **20**, 402–417 (2004)
4. Choset, H., Lynch, K., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L., Thrun, S.: Principles of Robot Motion: Theory, Algorithms, and Implementation. MIT Press (2006)
5. Dubins, L.E.: On curves of minimal length with a constraint on average curvature and prescribed initial and terminal positions and tangents. American Journal of Mathematics **79**, 497–516 (1957)

6. Esteves, C., Arechavaleta, G., Pettré, J., Laumond, J.P.: Animation planning for virtual characters cooperation. ACM Trans. on Graphics **25**(2), 319–339 (2006)
7. Fleury, S., Herrb, M., Chatila, R.: Genom: a tool for the specification and the implementation of operating modules in a distributed robot architecture. In: Proc. 1997 IEEE Int. Conf. on Intelligent Robots and Systems, pp. 842 – 849 (1997)
8. Gleicher, M.: Comparing constraint-based motion editing method. Graphical Models **63**, 107–134 (2001)
9. Harada, H., Kajita, S., Kanehiro, F., Fujiwara, K., Kaneko, K., Yokoi, K., Hirukawa, H.: Real-time planning of humanoid robot's gait for force controlled manipulation. In: Proc. 2004 IEEE Int. Conf. on Robotics and Automation, pp. 616–622 (2004)
10. Harada, H., Kajita, S., Saito, H., Morisawa, M., Kanehiro, F., Fujiwara, K., Kaneko, K., Hirukawa, H.: A humanoid robot carrying a heavy object. In: Proc. 2005 IEEE Int. Conf. on Robotics and Automation, pp. 1712–1717 (2005)
11. Hsu, D., Latombe, J.C., Sorkin, S.: Placing a robot manipulator amid obstacles for optimized execution. In: Proc. 1999 Int. Symp. on Assembly and Task Planning, pp. 280–285 (1999)
12. Hwang, Y., Konno, A., Uchiyama, M.: Whole body cooperative tasks and static stability evaluations for a humanoid robot. In: Proc. 2003 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 1901–1906 (2003)
13. Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., Hirukawa, H.: Biped walking pattern generation by using preview control of zero-moment point. In: Proc. 2003 IEEE Int. Conf. on Robotics and Automation, pp. 1620–1626 (2003)
14. Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., Hirukawa, H.: Resolved momentum control: Humanoid motion planning based on the linear and angular momentum. In: Proc. 2993 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 1644–1650 (2003)
15. Kanehiro, F., Hirukawa, H., Kajita, S.: OpenHRP: Open architecture humanoid robotics platform. Int. J. of Robotics Research **23**(2), 155–165 (2004)
16. Kaneko, K., Kanehiro, F., Kajita, S., Hirukawa, H., Kawasaki, T., M. Hirata, K.A., Isozumi, T.: The humanoid robot HRP-2. In: Proc. 2004 IEEE Int. Conf. on Robotics and Automation, pp. 1083–1090 (2004)
17. Kavraki, L., Svestka, P., Latombe, J.C., Overmars, M.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. on Robotics and Automation **12**(4), 566–580 (1996)
18. Kuffner, J.: Motion planning with dynamics. Physiqual (1998)
19. Kuffner, J., Kagami, S., Nishiwaki, K., Inaba, M., , Inoue: Dynamically-stable motion planning for humanoid robots. Autonomous Robots **12**(1), 105–118 (2002)
20. Laumond, J.P. (ed.): Robot Motion Planning and Control, *Lectures Notes in Control and Information Sciences*, vol. 229. Springer (1998)
21. Laumond, J.P.: Kineo cam: a success story of motion planning algorithms. IEEE Robotics & Automation Magazine **13**(2), 90–93 (2006)
22. LaValle, S.: Planning Algorithm. Cambridge University Press (2006)
23. LaValle, S., Kuffner, J.: Rapidly-exploring random trees: Progress and prospects. In: K.M. Lynch, D. Rus (eds.) Algorithmic and Computational Robotics: New Directions, pp. 293–308. A K Peters (2001)
24. Mansard, N., Stasse, O., Chaumette, F., Yokoi, K.: Visually-guided grasping while walking on a humanoid robot. In: Proc. 2007 IEEE Int. Conf. on Robotics and Automation, pp. 3042–3047 (2007)
25. Nakamura, Y.: Advanced Robotics: Redundancy and Optimization. Addison-Wesley Longman Publishing, Boston (1991)
26. Reeds, J.A., Shepp, R.A.: Optimal paths for a car that goes both forwards and backwards. Pacific Journal of Mathematics **145**(2), 367–393 (1990)
27. Sentis, L., Khatib, O.: Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. Int. J. of Humanoid Robotics **2**(4), 505–518 (2005)

28. Siciliano, B., Slotine, J.J.E.: A general framework for managing multiple tasks in highly redundant robotic systems. In: Proc. IEEE Int. Conf. on Advanced Robotics, pp. 1211–1216 (1991)
29. Sugihara, T., Nakamura, Y., Inoue, H.: Realtime humanoid motion generation through zmp manipulation based on inverted pendulum control. In: Proc. 2002 IEEE Int. Conf. on Robotics and Automation, pp. 1404–1409 (2002)
30. Sussmann, H.: Lie brackets, real analyticity and geometric control. In: R. Brockett, R. Millman, H. Sussmann (eds.) Differential Geometric Control Theory, *Progress in Mathematics*, vol. 27, pp. 1–116. Michigan Technological University, Birkhauser (1982)
31. Takubo, T., Inoue, K., Sakata, K., Mae, Y., Arai, T.: Mobile manipulation of humanoid robots – control method for com position with external force –. In: Proc. 2004 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 1180–1185 (2004)
32. Yamane, K., Nakamura, Y.: Natural motion animation through constraining and deconstraining at will. IEEE Trans. on Visualization and Computer Graphics **3**(9), 352–360 (2003)
33. Yoshida, E., Blazevic, P., Hugel, V., Yokoi, K., Harada, K.: Pivoting a large object: whole-body manipulation by a humanoid robot. J. of Applied Bionics and Biomechanics **3**(3), 227–235 (2006)
34. Yoshida, E., Esteves, C., Belousov, I., Laumond, J.P., Sakaguchi, T., Yokoi, K.: Planning 3D collision-free dynamic robotic motion through iterative reshaping. IEEE Trans. on Robotics **24**(5), 1186–1198 (2008)
35. Yoshida, E., Kanoun, O., Esteves, C., Laumond, J.P., Yokoi, K.: Task-driven support polygon reshaping for humanoids. In: Proc. 2006 IEEE-RAS Int. Conf. on Humanoid Robots, pp. 827–832 (2006)
36. Yoshida, E., Mallet, A., Lamiraux, F., Kanoun, O., Stasse, O., Poirier, M., Dominey, P.F., Laumond, J.P., Yokoi, K.: "give me the purple ball" – he said to HRP-2 N.14. In: Proc. 2006 IEEE-RAS Int. Conf. on Humanoid Robots (2007)
37. Yoshida, E., Poirier, M., Laumond, J.P., Alami, R., Yokoi, K.: Pivoting based manipulation by humanoids: a controllability analysis. In: Proc. 2007 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 1130–1135 (2007)
38. Yoshida, E., Poirier, M., Laumond, J.P., Kanoun, O., Lamiraux, F., Alami, R., Yokoi, K.: Regrasp planning for pivoting manipulation by a humanoid robot. In: Proc. 2009 IEEE Int. Conf. on Robotics and Automation (2009). To appear
39. Yoshida, H., Inoue, K., Arai, T., , Mae, Y.: Mobile manipulation of humanoid robots - a method of adjusting leg motion for improvement of arm's manipulability. In: Proc. 2001 IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics, pp. 266–271 (2001)
40. Yoshikawa, T.: Manipulability of robotic mechanisms. Int. J. Robotics Research **4**(2), 3–9 (1985)