The rest of the paper is organized as follows. Section 2 defines a reaching motion planning problem. Sections 3 and 4 present details of the planning and execution phases, respectively. Section 5 shows some simulation results. Finally, Section 6 concludes the paper.

2. Problem statement

The proposed method plans and executes a reaching motion from the following inputs.

Geometric models: We assume that the environment is measured by sensors such as a laser range finder while the robot is exploring; a voxel map is generated by accumulating these measurements. The voxel map is converted into a sphere tree[7] by assigning a sphere to each voxel. The shapes of the robot are approximated using spheres and capsules. These models are used for collision detection during the planning phase and distance computation during the execution phase.

Initial configuration: We assume that the robot has already been navigated by a human operator to an appropriate standing position from which the robot can reach the target.

Workspace goal region: The reaching target does not necessarily need to be a point in 6D space. It is given as a region called *Workspace Goal Region (WGR)*[8], which is appropriate for target objects.

The arm used for reaching is not explicitly given, but automatically selected by the planner.

The planned and executed motion has the following constraints.

Foot positions: Steps are not used and feet maintain their initial positions.

Static stability: To maintain the static stability, the horizontal position of the center of gravity (COG) is kept stationary above its support polygon.

3. Reaching motion planning

3.1. Fast configuration projector

In general, time-consuming processes involved in the motion planning are (1) collision detection between the robot and environment and (2) the projection of randomly sampled configurations onto manifolds where constraints are respected. Because these processes are executed multiple times (approximately one million times in our example in Table 1) to find an initial path and optimize it, they should be done efficiently. Unfortunately, the configuration projection tends to be computationally heavy because a humanoid robot is highly redundant and must respect constraints such as foot position/orientation and COG position. The usual approach is to project a configuration onto a constrained manifold by numerically solving whole body inverse kinematics. This projection process is a bottleneck toward achieving a solution to the motion planning problem.

We aim to remove this bottleneck by introducing a fast configuration projector based on an approximation of the mass distribution that enables an analytical solution of inverse kinematics.

First, we assume that the whole mass is concentrated on a point fixed to the trunk link. Hereafter, the COG position computed using distributed masses is called *the exact COG position* $\boldsymbol{p}_{\mathrm{c}}^{\mathrm{exact}}$ and that using a concentrated mass is called *the approximated COG position* $\boldsymbol{p}_{\mathrm{c}}$, which is determined in such a way that it coincides with $\boldsymbol{p}_{\mathrm{c}}^{\mathrm{exact}}$ when the robot is in the initial standing configuration. Based on this assumption, the static stability is easily maintained by choosing the trunk position such that the horizontal position of $\boldsymbol{p}_{\mathrm{c}}$ does not move. We therefore do not need to compute the COG Jacobian or solve inverse kinematics numerically.

In addition, we fix some joints and split the robot's kinematic chain into subchains, so as to benefit from the analytical solutions of inverse kinematics. Neck, waist, and finger joints are fixed and kinematic chains of arms and legs are connected to the trunk. In this paper, we assume that the robot's arms and legs have six degrees of freedom (DOFs), as is the case of our humanoid robot, HRP-2.[9] Should a robot have limbs with a higher DOF, the proposed method can be used by fixing some of the joints or determining these joints by random sampling.

Using these approximations, randomly sampled configurations are projected onto a constrained manifold in the joint space by (1) determining the trunk position/orientation and (2) determining joint angles from the inverse kinematics solution of 6-DOF chains. Usage of the analytical inverse kinematics solution helps speed up this computation.

3.2. Planning a motion

Because a humanoid robot is highly redundant and the goal is determined by the WGR, we need to sample goals for use as seeds of search trees. The configuration space used to find goals is defined as $\boldsymbol{q}_{\mathrm{goal}} = (\boldsymbol{p}_{\mathrm{e}}^T \ \boldsymbol{r}_{\mathrm{e}}^T \ z_{\mathrm{t}} \ \boldsymbol{r}_{\mathrm{t}}^T)^T$. This is a concatenation of an end-effector position $\boldsymbol{p}_{\mathrm{e}}$, an end-effector orientation $\boldsymbol{r}_{\mathrm{e}}$, the height of the trunk z_{t}, and an orientation of the trunk $\boldsymbol{r}_{\mathrm{t}}$. Given the robot model $\mathcal{M}$ for collision detection, a randomly generated configuration $\boldsymbol{q}_{\mathrm{goal}}$ and several constants, Algorithm 1 computes a posture $\boldsymbol{q}_{\mathrm{all}}$ that respects given positions/orientations of links without moving the horizontal position of $\boldsymbol{p}_{\mathrm{c}}$. If it succeeds, it returns true. All positions and orientations in Algorithm 1 are expressed in the world coordinate system. First, $\mathcal{M}$ is rotated and translated without changing the horizontal position of $\boldsymbol{p}_{\mathrm{c}}$ (line 1 and 2). Then, the arms and legs are checked to determine if they can reach specified positions/orientations. Note that 'solve{RightArm,LeftArm,RightLeg,LeftLeg}IK()' are functions used to analytically solve inverse kinematics of a kinematic chain; $\boldsymbol{p}_{rf}$, $\boldsymbol{r}_{rf}$, $\boldsymbol{p}_{lf}$ and $\boldsymbol{r}_{lf}$ are the feet positions

and orientations. The right arm is used preferentially if both arms are reachable (from line 3 to 5). When Algorithm 1 returns True, the configuration is used as a node of the search tree.

Algorithm 1 ProjectConfig

Input: $\mathcal{M}$, $\boldsymbol{q}_{\text{goal}}$, $\boldsymbol{p}_{\text{c}}$, $\boldsymbol{p}_{rf}$, $\boldsymbol{r}_{rf}$, $\boldsymbol{p}_{lf}$, $\boldsymbol{r}_{lf}$
Output: True/False, $\mathcal{M}.\boldsymbol{q}_{\text{all}}$

1: $\mathcal{M}$.rotateAroundPoint($\boldsymbol{p}_{\text{c}}$, $\boldsymbol{r}_{\text{t}}$)
2: $\mathcal{M}$.setHeight(z_{t})
3: **if** $\mathcal{M}$.solveRightArmIK($\boldsymbol{p}_{\text{c}}$, $\boldsymbol{r}_{\text{t}}$, $\boldsymbol{p}_{\text{e}}$,$\boldsymbol{r}_{\text{e}}$) $\neq$ True **then**
4: **if** $\mathcal{M}$.solveLeftArmIK($\boldsymbol{p}_{\text{c}}$, $\boldsymbol{r}_{\text{t}}$,$\boldsymbol{p}_{\text{e}}$,$\boldsymbol{r}_{\text{e}}$) $\neq$ True **then**
5: **return** False
6: **if** $\mathcal{M}$.solveRightLegIK($\boldsymbol{p}_{\text{c}}$, $\boldsymbol{r}_{\text{t}}$, $\boldsymbol{p}_{rf}$,$\boldsymbol{r}_{rf}$) $\neq$ True **then**
7: **return** False
8: **if** $\mathcal{M}$.solveLefLegIK($\boldsymbol{p}_{c}$, $\boldsymbol{r}_{t}$, $\boldsymbol{p}_{lf}$,$\boldsymbol{r}_{lf}$) $\neq$ True **then**
9: **return** False
10: **return** True

A reaching motion is planned using IKBiRRT.[8] The configuration space for the reaching motion planning is defined as $\boldsymbol{q}_{\text{plan}} = (\boldsymbol{q}_{\text{arms}}^T\ z_{\text{t}}\ \boldsymbol{r}_{\text{t}}^T)^T$ where $\boldsymbol{q}_{\text{arms}}$ is a vector of the arms' joint angles. While IKBiRRT grows a tree, Algorithm 1 is used to project configurations on local paths that connect sampled configurations and nodes nearest to them. However, lines 3–5 of Algorithm 1 are skipped because $\boldsymbol{q}_{\text{arms}}$ are given by random sampling.

4. Execution with real-time compensation

As the path is a sequence of discrete postures, a posture is computed by interpolating them using a clamped cubic spline and executed at each control cycle.

Although we confirmed that the approximation error of the COG position is less than a few centimeters, it is preferable to keep the horizontal position of the exact COG position ($\boldsymbol{p}_{\text{c}}^{\text{exact}}$) constant in order to maximize the stability margin. The postures are thus executed while compensating for the approximation error in real time.

The compensation is done by modifying postures so that the horizontal position of $\boldsymbol{p}_{\text{c}}^{\text{exact}}$ is maintained constant at each control cycle. Although the required modification is small, in some case, all constraints may not be simultaneously satisfied because the postures are modified by local optimization through solving whole body inverse kinematics. To continue execution even in such cases, constraints are first prioritized and the postures then modified so that constraints are respected as far as possible by solving prioritized whole body inverse kinematics.[10] The following four priority levels are used.

(1) Joint limits and collision avoidance constraints have the highest priority because the robot will cause immediate damage to itself or the environment if they are not respected. First, spheres of the environment model within the given distance from the robot are picked up. Then, *velocity dampers*[11] are applied to each pair of environment sphere and robot geometry, both evaluated as inequality constraints in the inverse kinematics computation.
(2) Maintaining foot position/orientation and the horizontal position of $\boldsymbol{p}_{\text{c}}^{exact}$ have the second priority since these are important to keep static stability. These are equality constraints.
(3) Maintaining a hand position/orientation constraint has the third priority. Although moving a hand to the specified position is the main objective, it has a lower priority compared to the above two levels since ensuring the robot's safety is critical for enabling continued exploration.
(4) Residual redundancy is used to realize a posture given by the initial trajectory.

Each posture can be modified in real time because the modification is required only once for each control cycle and the processing time is shorter than the control cycle.

5. Simulation results

5.1. Performance evaluation

To confirm the efficiency of the proposed framework, reaching motions in an industrial plant shown in Figure 2 are planned and executed on a simulator. In this example, we defined the WGR as follows:

$$[x_{\min}\ x_{\max};\ y_{\min}\ y_{\max};\ z_{\min}\ z_{\max};\ \phi_{\min}\ \phi_{\max};\ \theta_{\min}\ \theta_{\max};\ \psi_{\min}\ \psi_{\max}] = \left[0.71\ 0.71;\ 0.03\ 0.03;\ 0.9\ 0.9;\ -\frac{\pi}{2}\ \frac{\pi}{2};\ -\frac{\pi}{2}\ \frac{\pi}{2};\ -\pi\ \pi\right] \quad (1)$$

This WGR means that a hand can grasp the target valve at the fixed position and from any direction in the half space shown in Figure 1.

The entire environment, having dimensions 4 m × 5 m, is given as a voxel map. The resolution of the voxel map

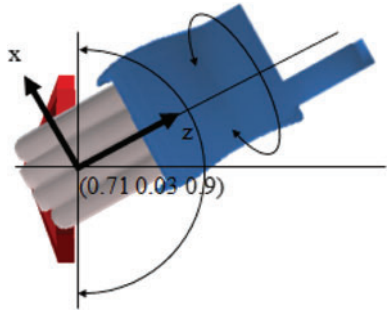

Figure 1. WGR is defined such that a hand can grasp the target valve at the fixed position and from any direction in the half space.

Table 1. Performance of reaching motion planning.

Average planning time	226 ms
Collision detection	41% of total time, 382875 calls (24 μs/call)
ProjectConfig for goal sampling	28% of total time, 991363 calls (6.6 μs/call) (Only 100 calls for valid goal postures)

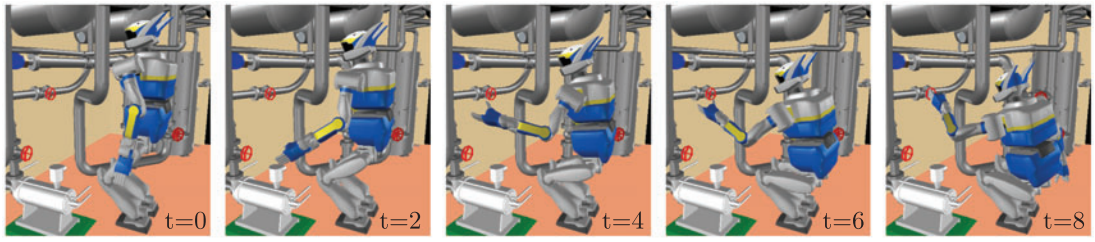

Figure 2. An example of planned and executed reaching motions by the left arm.

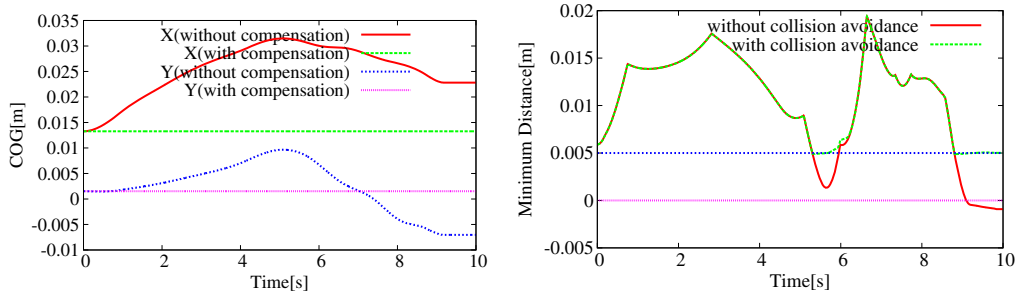

Figure 3. Left: comparison of COG trajectories. When the COG constraint is enabled, the horizontal component of the exact COG position remains stationary (green and pink lines). Right: Minimum distance between the robot and environment. The path is modified so that the minimum distance is never lower than the threshold (blue line).

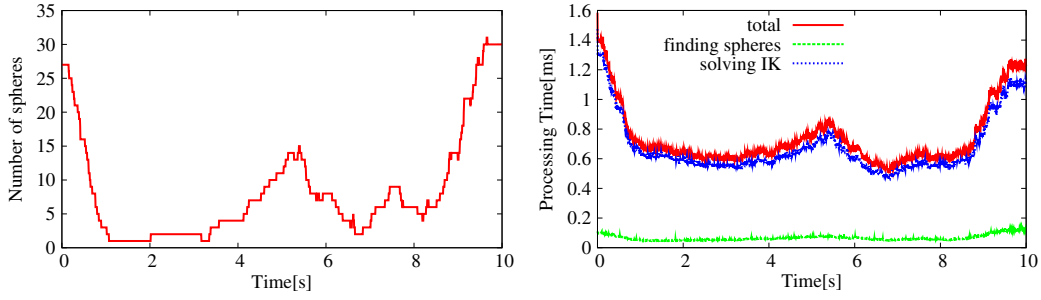

Figure 4. Left: the number of spheres within the distance threshold, Right: processing time used for finding those spheres, solving the prioritized inverse kinematics and their sum. The total time is sufficiently short to allow execution in real time.

is 2 cm and the map consists of 22,753 occupied voxels. It takes 20 ms to construct the sphere tree from the voxel map on a modern computer (CPU:Intel Core i7 3.2 GHz).

Table 1 shows performance indices obtained when 100 reaching motions are planned. The initial configuration is given as shown in the left most panel of Figure 2, and the robot is instructed to reach the red valve in front. The probability used to sample goals and the time limit of IKBiRRT are set to 10% and 3 s, respectively. The planner finds paths within the time limit in all trials and takes an average of 226 ms to find a motion. It is confirmed that the time required for an operator to wait before the robot starts to move is very short. Although around one million configurations are sampled to obtain goal postures, most of them are rejected on account of failures encountered when solving inverse kinematics, as shown Table 1. Usage

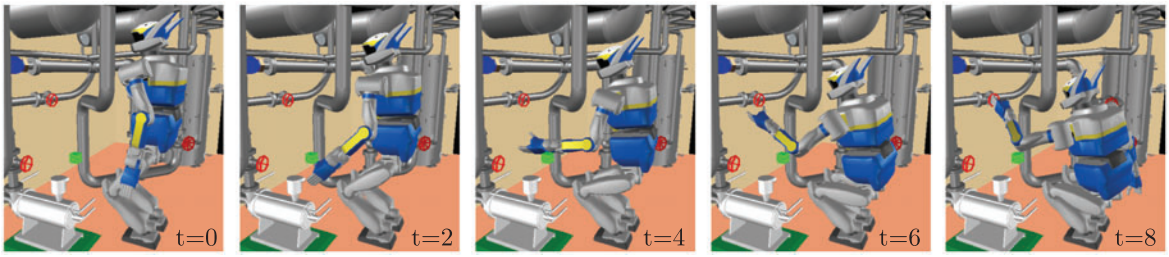

Figure 5. Deformation of the initial path. The initial path is deformed during the execution phase to avoid collisions with a newly appeared obstacle.

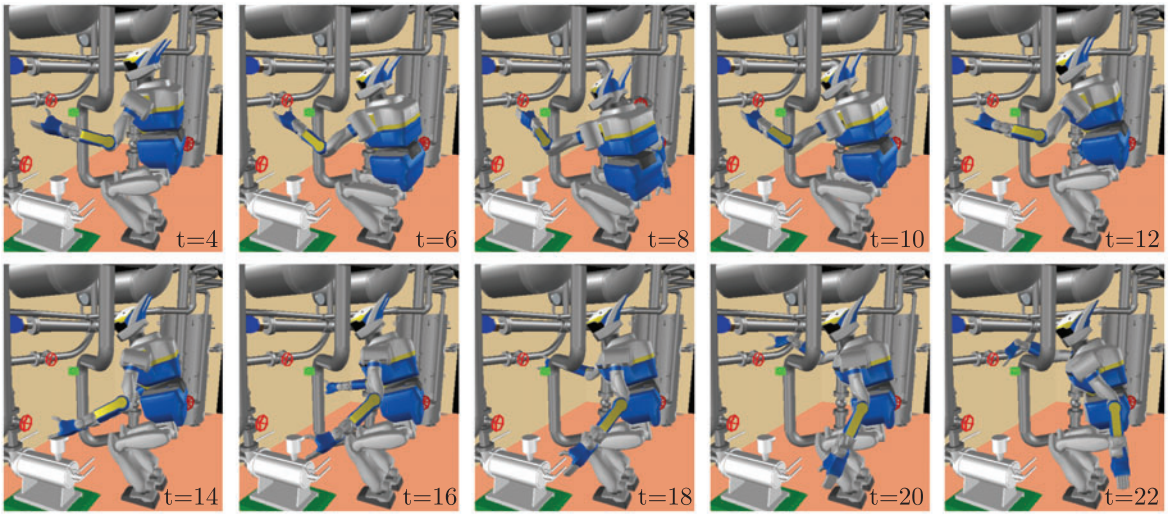

Figure 6. An example of replanning. When an obstacle appears near the target, replanning is activated and the other arm is used.

of analytical solutions of inverse kinematics is therefore advantageous for reducing the computation time. We confirmed that the average computation time of ProjectConfig with the numerical solution of inverse kinematics [10] was 792 μs per call, which was 120 times slower than 6.6 μs per call with the analytical solution. Since ProjectConfig occupies 28% of the total planning time, as shown Table 1, the total planning time will become 34 times longer when ProjectConfig is used with numerical solution; this would therefore require considerable patience from the operator.

Figure 2 shows an example of planned and executed reaching motions. In this example, the left arm is selected and the robot lowers its body to avoid collisions between its left shoulder and a pipe. The planned path consists of 6 configurations after smoothing.

Figure 3 (left) shows trajectories of the exact COG position ($\boldsymbol{p}_c^{\text{exact}}$) for the reaching motion by the right arm. If the initial trajectory is executed without compensation, $\boldsymbol{p}_c^{\text{exact}}$ moves a few centimeters. We can confirm that the horizontal position of $\boldsymbol{p}_c^{\text{exact}}$ does not move with the compensation. In Figure 3 (right), minimum distances between the robot and environment are plotted. Negative distances indicate collision of the robot with the environment. Hence, for safety purposes, the whole body inverse kinematics solver uses collision-avoidance constraints in order to maintain a distances of 5 mm or greater. The minimum distance should never be lower than the threshold.

Figure 4 (left) shows the number of spheres within the distance threshold 2 cm from the robot and Figure 4 (right) shows the processing time to find these spheres, solving prioritized inverse kinematics and their sum. Many spheres are detected at the initial configuration because both shoulders are close to pipes. The processing time is proportional to the number of spheres and is popularly used to solve prioritized inverse kinematics. The processing time is short enough to allow execution in real time since the control cycle of HRP-2 is 5 ms.

5.2. *Integration with reactive planning framework*

In our previous study,[12] we proposed a reactive planning framework that combines the deformation and replanning methods. Once a collision-free path is planned and begins execution, the robot continues executing the path as long as

the path remains feasible with necessary local path deformation caused by the motion of obstacles. If the executed path becomes infeasible even after deformation, replanning is performed to find an alternative path. We validated the effectiveness of this method by applying it to redundant manipulators. However, since extensive computations are required, further improvement is necessary in order to allow the method to be applicable to humanoid robots.

Because the effectiveness of the proposed method is useful for planning the initial path and quickly replanning paths, the method has been integrated with the reactive planning framework. The execution phase works not only to compensate for the approximation error but, in this case, also to deform the initial trajectory in order to enable adaptability to small changes in the environment.

Figure 5 shows the case where an obstacle (green cube) appears after the initial path is planned. The initial path is the same as the path shown in Figure 2. When the environment model is updated and the obstacle is detected on the initial path, the initial path is deformed in order to avoid collisions with the obstacle. As a consequence, the left arm passes through the right side of the obstacle.

Figure 6 shows a case where an obstacle appears near the target (images for 0 and 2 s are omitted since they are the same as Figure 2. The obstacle prevents the robot from reaching the target. When the execution phase is complete, the position and orientations of the hand are checked in order to verify the robot reached the target. If the robot finds that its hand failed to reach the target, it restarts the planning phase, plans a new path with the updated environment model, and executes the new plan. In this case, the planner finds a path to reach the target using the right arm because the path from the left side is blocked by the obstacle and its execution starts immediately.

Even if the environment is not modified, the execution phase rarely fails because the hand position/orientation constraint has lower priority. This replanning function is also useful for such cases to attempt to complete the task by using a new path.

6. Conclusions

We presented an efficient reaching motion planning method for humanoid robots on exploration missions. Assuming that the robot is teleoperated by a human, we focused on reducing planning time, hence reducing operator time. The proposed method consists of a planning phase and an execution phase. The former consists of a reaching motion that is quickly planned by utilizing analytical solutions of inverse kinematics and the latter consists of the execution of the planned path while compensating for an approximation error in real time once other constraints are maintained. The effectiveness of the method was confirmed by generating reaching motions in a simulated industrial plant.

In this study, we did not deal with the manipulation of the target object. The robot in the presented example may not be able to open/close the valve if the reached position and orientation not be appropriate. Future work will therefore consider unifying both reaching and object manipulation in a single framework. Another issue is the selection of a standing position and stance, both which were predetermined in this study. Operators often find it difficult to choose an appropriate standing position and stance. We therefore plan to investigate the ability to automatically decide the standing posture that maximizes the workable range for the robot in future studies.

Notes on contributors

Fumio Kanehiro received the BE, ME, and PhD in engineering from The University of Tokyo, Japan, in 1994, 1996, and 1999, respectively. He was a Research Fellow of the Japan Society for the Promotion of Science in 1998-1999. In 2000, he joined the Electrotechnical Laboratory, Agency of Industrial Science and Technology, Ministry of Industrial Science and Technology (AIST-MITI). From April 2007, he was a visiting researcher at the LAAS-CNRS for one year and three months. He is currently a senior researcher of the Intelligent Systems Research Institute, National Institute of Advanced Industrial Science and Technology (AIST). His research interests include the software platform development and whole body motion control of the humanoid robot.

Eiichi Yoshida received ME and PhD degrees on Precision Machinery Engineering from Graduate School of Engineering, the University of Tokyo in 1993 and 1996, respectively. From 1990 to 1991, he was visiting research associate at the Swiss Federal Institute of Technology at Lausanne (EPFL). In 1996 he joined former Mechanical Engineering Laboratory, later reorganized as National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Japan. He served as Co-Director of AIST/IS-CNRS/ST2I Joint French-Japanese Robotics Laboratory (JRL) at LAAS-CNRS, Toulouse, France, from 2004 to 2008. Since 2009, he is Co-Director of CNRS-AIST JRL (Joint Robotics Laboratory), UMI3218/CRT, Tsukuba, Japan. His research interests include robot task and motion planning, modular robotic systems, and humanoid robots.

Kazuhito Yokoi is Deputy Director of Intelligent Systems Research Institute, National Institute of Advanced Industrial Science and Technology (AIST) in Japan. He has received the ME and PhD degrees in Mechanical Engineering Science from the Tokyo Institute of Technology in 1986 and 1994, respectively. He is also the leader of Humanoid Research Group of IS/AIST, a member of CNRS-AIST JRL, UMI3218/CRT, an adjunctive professor of Cooperative Graduate School at University of Tsukuba, and the general manager of

the technology planning office of International Research Institute for Nuclear Decommissioning. From November 1994 to October 1995, he was a Visiting Scholar at Robotics Laboratory, Computer Science Department, Stanford University. His research interests include humanoids and human-centered robotics. He has received the Best Video Award of ICRA03 and ICRA04, the best paper award of SICE (Society of Instrument and Control Engineers), RSJ (The Robotics Society of Japan), and JSME (The Japan Society of Mechanical Engineer). He is a fellow of RSJ and JSME and an AdCom member of IEEE Robotics and Automation Society.

References

[1] LaValle SM. Rapidly-exploring random trees: a new tool for path planning. Computer Science Department, Iowa State University; 1998. (Tech Rep 98–11).

[2] Kavraki LE, Svestka P, Latombe J-C, Overmars MH. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Rob. Autom. 1996;12:566–580.

[3] Kuffner JJ, Kagami S, Nishiwaki K, Inoue H, Inaba M. Dynamically-stable motion planning for humanoid robots. Auton. Robot. 2002;12:105–118.

[4] Yoshida E, Esteves C, Belousov I, Laumond JP, Sakaguchi T, Yokoi K. Planning 3D collision-free dynamic robotic motion through Iterative reshaping. IEEE Trans. Rob. 2008;24:1186–1198.

[5] Harada K, Hattori S , Hirukawa H, Morisawa M, Kajita S, Yoshida E . Motion planning for walking pattern generation of humanoid robots. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Iobotics and Systems (IROS'07); 2007; San Diego, CA, USA. p. 4227–4233.

[6] Dalibard S, Nakhaei A, Lamiraux F, Laumond JP. Whole-body task planning for a humanoid robot: a way to integrate collision avoidance. In: Proceedings of the IEEE-RAS International Conference on Humanoid Robots; 2009; Paris, France. p. 355–360.

[7] Quinlan S. Efficient distance computation between non-convex objects. In: Proceedings of the 1994 IEEE International Conference on Robotics & Automation; 1994; San Diego, CA, USA. p. 3324–3329.

[8] Berenson D, Srinivasa SS, Ferguson D, Collet A, Kuffner JJ. Manipulation planning with workspace goal regions.In: Proceedings of the 2009 IEEE International Conference on Robotics & Automation; 2009; Kobe, Japan. p. 1397–1403.

[9] Kaneko K, Kanehiro F, Kajita S, Hirata M, Akachi K, Isozumi T. Humanoid robot HRP-2. In: Proceedings of the 2004 IEEE International Conference on Robotics & Automation; 2004; Barcelona, Spain. p. 1083–1090.

[10] Kanoun O, Lamiraux F, Wieber PB. Kinematic control of redundant manipulators: generalizing the task priority framework. IEEE Trans. Rob. 2011;27:785–792.

[11] Faverjon B, Tournassoud P. A local based approach for path planning of manipulators with a high number of degrees of Freedom. In: Proceedings of IEEE International Conference on Robotics and Automation; 1987; Raleigh, NC, USA. p. 1152–1159.

[12] Yoshida E, Kanehiro F. Reactive robot motion using path replanning and deformation. In: Proceedings of the 2011 IEEE International Conference on Robotics & Automation; 2011; Shanghai, China. p. 5456–5462.