Proceedings of the 2006 IEEE/RSJ
International Conference on Intelligent Robots and Systems
October 9 - 15, 2006, Beijing, China

A Hybrid Approach to Practical Self Collision Detection System of Humanoid Robot

Kei Okada Masayuki Inaba
Graduate School of Information Science and Technology, University of Tokyo
#701, Engineering Building No. 8, 7-3-1, Hongo, Bunkyo-ku, Tokyo, Japan
k-okada@jsk.t.u-tokyo.ac.jp

Abstract— **Online self collision detection system for humanoid robots is an essentially important function for developing sensor based behaviors without worrying about breaking hardware.**

In this paper, we propose a practical and real-time self collision detection system for humanoid robots that satisfy both enough a range of movement and a safety margin. Previous researches usually add a safety margin around each link to cope with errors in both modeling and control. However this margin significantly decreases a range of movement of a joint, especially a compound joint, a joint between adjacent links and composed of 2 or 3 revolute joints whose axis intersect in a same point

In order to gain enough a range of movement and safety margin, we developed hybrid approach that uses both table based collision checking for compound joints and online geometrical model checking with a simplified link shape for other joints

We have experimentally evaluated our self collision detection system using a HRP2-JSK humanoid robot. Our demonstration shows that the robot automatically stops its motion when self collision occurs.

I. Introduction

Recently, as a highly stable walking control technology of a humanoid robot progresses, there has been increasing interest in sensor based whole body behavior control. On the contrary to carefully pre-programmed behaviors of conventional humanoid robots, a sensor based behavior of a humanoid robot changes it's motions on the fly. However these online behaviors causes a "self collision" that two or more links of a robot collide as shown in Fig.1.

A self collision yields not only serious damage to a robot itself, but also inflicts financial damage and psychological stress on researchers. Thus, a practical software for detecting self collisions in real-time is a fundamental and essential function for developing a sensor based behavior of a humanoid robot safely.

There has been very little work on real-time self collision detection for humanoid robots. Since humanoid robots has many degree of freedom and they usually have complex shape. For example, in the case of a humanoid robot with 33 links, collision pairs to be checked becomes 528 pairs and each link has about 1,000 polygons and the total number of polygons becomes more than 30,000. Therefore, previous researchers focused on reducing computational cost on collision detection.

Kanehiro et al. [1] reduced collision pairs from 350 to 36 for a 29 DOF humanoid robot using online and offline hybrid approach. Kuffner et al. [2] used simplified 3D model of a robot by transforming from polygon soup(314,588 polygons) to convex hull(2,702 polygons). They reduced collision pair to be checked from 435 to 76. Okada et al. developed precise collision detection system for humanoid robots that does not reduce the number of polygons and checks 109 out of 435 collision pairs by using AABB based collision detection technique [3].

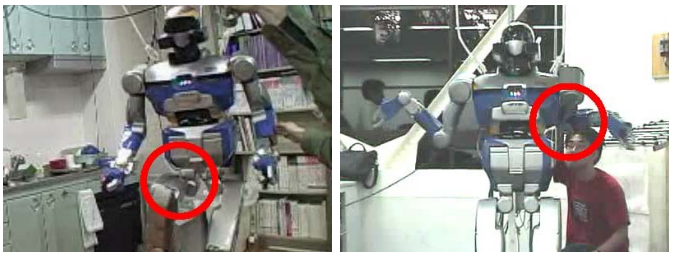


Fig. 1. Self collisions in a humanoid robot

Adding safety margin for modeling and control errors is significantly important for practical usage of self a collision detection system for humanoid robots. However these approaches reduce a range of movement, especially links in a compound joint(a joint between adjacent links and composed of 2 or 3 revolute joints whose axis intersect in a same point). Reducing a range of movement is disliked by researchers who willing to make maximum use of hardware capability of a robot. Then, they sometimes remove self collision detection functions. Consequently the possibility for damaging robot arises.

Therefore, a practical self collision detection system which does not reduce a range of movement is highly demanded. In this paper, we present our hybrid self collision detection system. First, we classify self collisions into two groups as shown in Fig.2. One is termed *compound link pair* which is a collision between an adjacent links and composed of 2 or 3 revolute joints whose axis intersect in a same point. Another one is termed *simple link pair*. In Fig.2, a figure (1) shows a self collision of a simple link pair. Figures (2) and (3) show self collisions of a compound link pair. For a compound link pair, a very little safety margin is applied for assuring a large range of movement. On the other hand, conservative approach with enough safety margin is suitable is used for a simple link pair,

We also describe our implementation of the hybrid self collision detection system which uses memory based pre-

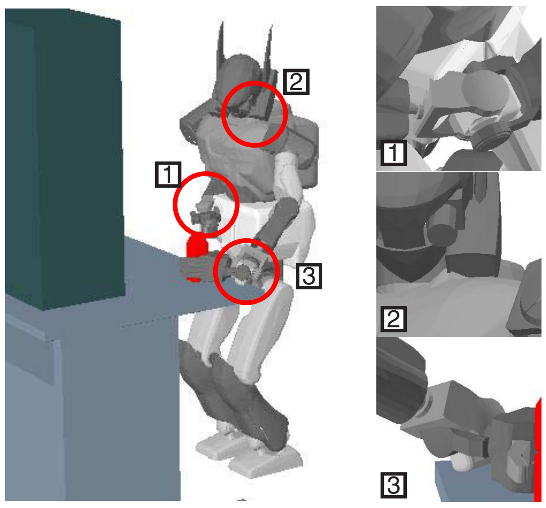
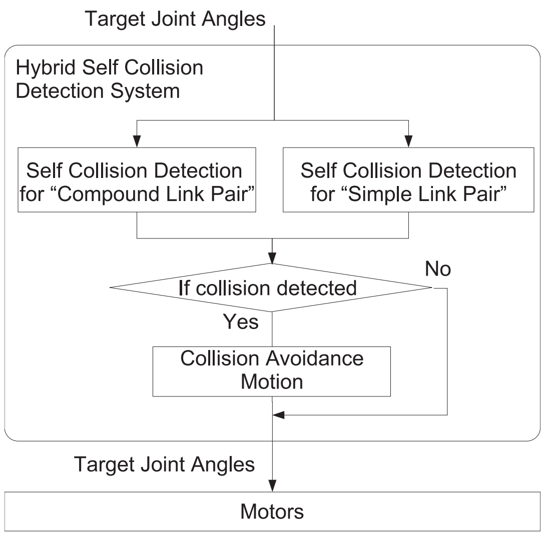


Fig. 2. An example of self collisions in a humanoid robot. 1) Self-collision of simple link pair, 2)3) Self-collisions of compound link pair



Target Joint Angles
Hybrid Self Collision Detection System
Self Collision Detection for "Compound Link Pair"
Self Collision Detection for "Simple Link Pair"
If collision detected
No
Yes
Collision Avoidance Motion
Target Joint Angles
Motors

Fig. 3. Hybrid self-collision detection system overview

computed collision checking method with a detailed link shape model of compound link pairs and online geometrical based checking method with a simplified link shaped model for simple link pairs.

Finally, we present experimental results of our self-collision detection system using HRP2-JSK humanoid robot[1]. Our demonstration shows that the robot automatically stops its motion when self collision occurs.

II. Design of Hybrid self-collision Detection System for Humanoid Robots

In order to design self collision detection system, following three issues are important: 1) selection of link pairs to be checked 2) collision detection calculation 3) motion generation for preventing collisions. Fig.3 shows an overview of our hybrid self collision detection system.

A. Selection of link pairs to be checked for a "Simple Link Pair Check"

"Simple Link Pair Check" detects self collisions between links in different limbs such as a left arm and a right arm. Conservative approaches such as adding safety margin around links or using an approximated model such as a convex hull are effective for reducing possibility of self collisions, because mechanical deflection or a joint angle error causes imprecise position of a link model which is used for a collision detection calculation.

Let N to be number of all links of a humanoid robot, the number of link pairs is ${}_NC_2$. Assuming that collision between an adjacent link never occurs because of joint limits, there still ${}_NC_2 - (N-1)$ pairs to be checked. In the case of our HRP2-JSK humanoid robot which has a total of 33 links, collision pairs to be checked becomes 528 according to the equation ${}_NC_2 - (N-1)$.

[1]The modified version of HRP-2 robot which has 7 DOF in arms, stereo vision sensor and mic/speaker system [4]

Obviously, the number of link pairs to be checked is able to be reduced by considering kinematics and geometrical constraints of a robot. For example, links in a head and a toe never intersect each other.

We have also developed heuristics in order to reduce collision pairs for simple link pair check from 528 to 143 as listed below. Fig.4 shows correspondence table between a link name and a figure.

- Links in a leg never collide with links in the same leg.
- Links in a leg never collide with links in a torso, a chest and a head.
- Links in an arm never collide with links in the same arm.
- Links in a shoulder part never collide with any other links.
- A head yaw and a waist yaw never collide with any other links.
- A crotch yaw link, a crotch roll and ankle pitch link in a leg never collide with any other links, but a crotch yaw link only collide with a crotch pitch link in the same leg.
- A shoulder pitch and yaw links only collide with lower arm links in another arm.
- A neck pitch link never collide with any links in the legs and the torso link.

B. Selection of link pairs to be checked for a "Compound Link Pair Check"

"Compound Link Pair Check" detects self collisions in a compound joint, which is defined as a joint between adjacent links and composed of 2 or 3 revolute joints whose axis intersect in a same point.

Precision is fundamental to this compound pair collision detection for not reducing the range of movement, therefore for example, adding safety margin or using an approximated shape model is unsuitable for this collision detection.

For a compound link pair check, we select 6 pairs (12 joints), which are a chest and head, link pairs in a waist, link pairs in wrists and link pairs in hip joints.

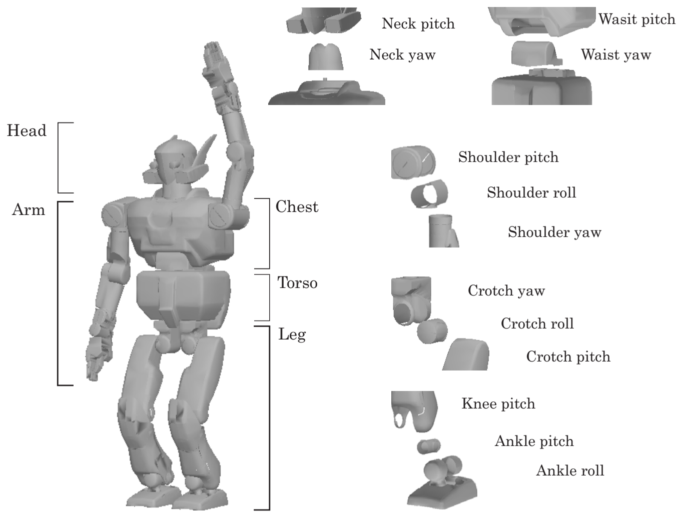


Fig. 4. Name of each links and it's correspond figure

Hip and shoulder joints are compound link joints composed of 3 revolute joints whose axis intersect in a same point, which is regarded as a spherical joint. We consider these joints as one compound joint composed of 2 revolute joints since the center link (shoulder roll and crotch roll links in this case) never collide with it's adjacent links (shoulder pitch and yaw links for a shoulder roll link) because of joint limits.

C. Self collision detection calculation

There are two main approaches for self collision detection calculation. One is the memory based off-line approach that pre-compute collisions for all possible posture in advance and stores the results in a look-up table. However, this approach has serious drawbacks when used for robots with many DOFs because of large memory space for look-up table is required.

Another approach is to use geometric model based on-line approach that calculate if link pairs are intersecting by using collision detection algorithms. This approach is able to classify into two types. One is exact collision detector which finds collisions between non-convex polygon soups and another is that only detects collisions between convex hulls. The drawback of this approach is its computation time, however current CPU power enables realizing on-line self-collision detection for humanoid robots with 20-30 DOFs.

Therefore, geometric model based online approach is suitable for a self collision calculation for a simple link pair check since it requires to check more than 100 self-collisions link pairs. For a compound link pair check, both approaches can be applied. In this paper, we adopt lookup table approach.

D. Motion generation for preventing collisions

When a system detects self-collisions, collision avoidance motions must be executed. To stop motion when it detect collisions is the simplest solution, however this solution can lose while body balance of a robot, especially when a robot is walking. This idea also has a disadvantage that a robot is not able to accomplish a original task goal. For example, if a system stops a motion when a robot tries to reach some object, a robot is not able to realize the reaching task.

Hence, motion generation after self-collisions are detected can be viewed from following three viewpoints.

1) Which joints should be controlled: The simplest solution stops whole body motion when self-collision detected. Other idea is to stop only collided links or limbs.
2) Keeping balance of a robot: Current online balance control algorithm enables a robot to keep balance with arbitrary upper body motions. However generating a collision preventing motion when detecting self-collisions a between upper body and a leg or between legs is difficult problem.
3) Reaching original goal: This issue includes the difficult problem which is "what is a goal?", however in this paper, we define a original goal as target posture for simplicity. In order to recover from collided posture to original motion goal, motion planning technique is able to applied

III. LOOKUP TABLE BASED OFFLINE COLLISION CHECKING FOR COMPOUND LINK PAIR

We adopted memory based off-line approach for compound link pair checking by using a lookup table. We assume each link have one actuator and one joint. Therefore, one compound link pair consists of two links and two joints. One joint in compound link pair has the target joint which is another joint in the compound link pair. First, the minimum and maximum joint angle information is updated by using joint angle of the target joint and lookup table which stores the minimum and maximum joint angles. Then it updates the joint angle to falls in the range of movement if the joint angle exceeds the minimum or maximum joint angle.

A. Lookup table based collision checking software implementation

The structure of look-up table is presented in Fig.5. A pair of `joint_id` and `target_id` presents a link pair. `angle_limit` presents the angle limit of the joint which is indicated by `joint_id` with respect to the angle of `target_id` joint.

From the line 13 to 25 in Fig.5 shows the lookup table for yaw joint of a neck. `HEAD_JOINT0` is the id number of the yaw joint and `HEAD_JOINT1` is the id number of the pitch joint of the head. `-30` and `75` shows the minimum and the maximum angle of the joint angle and the range of the movement of the pitch joint of the head(`HEAD_JOINT1`). `{-45, 45}` of the line 18 indicates the minimum and the maximum angle of the yaw joint(`HEAD_JOINT0`) when the joint angle of target joint which is the pitch joint angle of the head(`HEAD_JOINT1`) is -30 degree. In the same way, next line shows the range of movement of pitch joint when the pitch joint angle is -20 degree.

In order to find the range of movement of the joint, we use geometric model based self-collision detection. First, we set

```
typedef struct {
  float min, max;
} MinMaxAngle;

typedef struct {
  int joint_id, target_id;
  int start_angle, end_angle;
  MinMaxAngle angle_limit[];
} MinMaxTableInfo;

MinMaxTableInfo MinMaxTable[TABLE_SIZE] = {

/* HEAD_JOINT0 (:neck-y) */
 {HEAD_JOINT0,  /* joint-id */
  HEAD_JOINT1,  /* target-id */
  -30,  /* min-angle */
   75,  /* (- max-angle min-angle) */
  {{-45, 45},     /* joint-angle = -30 */
   {-45, 45},     /* joint-angle = -29 */
   {-45, 45},     /* joint-angle = -28 */
  .....
   {-5, 5},       /* joint-angle = 43  */
   {-4, 4},       /* joint-angle = 44  */
   {-2, 2}        /* joint-angle = 45  */
  }},

/* HEAD_JOINT1 (:neck-p) */
 {HEAD_JOINT1,  /* joint-id */
  HEAD_JOINT0,  /* target-id */
  -45,  /* min-angle */
   90,  /* (- max-angle min-angle) */
  {{-30, 18},     /* joint-angle = 45  */
  .....
   {-30, 45},     /* joint-angle = 1   */
  .....
   {-30, 19}      /* joint-angle = -45 */
  }}
}
```

Fig. 5. Structure of lookup-table for checking compound link pairs

the angle of the target joint (for example, pitch joint of the head). Then we set the angle of the current joint (yaw joint of the head) from the original minimum joint angle to the maximum joint angle with every 1 degree and calculate if a self-collision occurs.

IV. Geometric model based online collision checking for simple link pair

- 3D geometric model information for objects
  Usually it requires polygon triangles of the object surfaces.
- Transformation matrix of each object
  For updating current coordinates (the position and rotation) of the object.

Then we are able to perform collision query to see if there exists collided objects using the 3D geometric model information and the transformation matrix information.

First, the forward kinematics solver finds current coordinates which is the position and rotation of each joint by using join angle information and kinematics information of a robot. Then, the coordinates of 3D geometric model is updated in order to perform collision query.

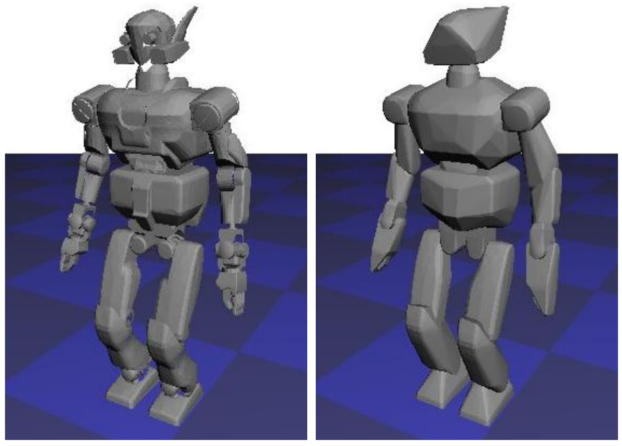


Fig. 6. Detailed model of a HRP2-JSK robot and convex hull model

A. Selection of collision detection library

Research on collision detection algorithm has a large and extensive history in the computer geometry literature [5], [6].

Existing software packages for collision detection are classified into two types. One is exact collision detector that is able to finds collisions between non-convex polygon soups and another are that only detects collisions between convex hulls.

In the case of our HRP2-JSK humanoid robot, a total number of polygons of original 3D model which is used for exact collision check is 31,690 and simplified 3D model of that is 13,690 by transforming to convex hull, which is used for convex hull's collision check. Fig.6 shows detailed shape model for exact collision check and convex hull model of the same robot.

When using exact collision detection libraries with original detailed 3D models are able to perform precise collision checking, however the drawback of this approach is computation cost since a large number of polygons must be checked. On the other hand, collision detection libraries for convex hull has advantages in computational cost however approximating complex humanoid robot shape to much simpler convex hulls may reduces the range of movement of each joints.

In our hybrid approach for humanoid collision detection system, link pairs that a range of movement becomes critical, are checked by lookup table based collision checking method, therefore a collision checking for simple link pair is able to use libraries for convex hull is able to adopt.

V. Experiment

A. Enough range of movement in compound link pairs

Table.I shows a range of movement in the wrist joint of HRP2-JSK robot. Fig.7 shows a wrist link pair when a wrist pitch joint angle is 0 [degree] and -53[degree] which is a limit angle.

As a safety margin increases, a range of movement decreases. This property makes it difficult to determine the value of the safety margin for practical use of collision detection

TABLE I

A RANGE OF MOVEMENT IN WRIST JOINT AGIANST SAFETY MARGIN

safety margin [mm]	min angle [degree]	max angle [degree]
0	-53	59
5	-42	50
10	-37	41
15	-29	33
20	-26	29
25	-21	26
30	-20	24

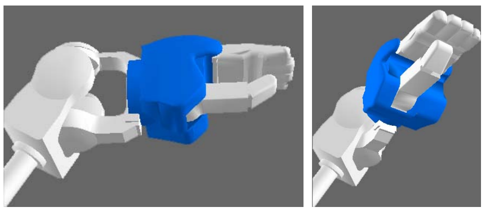


Fig. 7. Angle of wrist pitch is 0 (left)and , -53 [degree](right)

checking software. Since safety margin less than 10[mm] is not enough for collision checking between, for example, arm and leg, whereas 10[mm] safety margin significantly reduces a range of movement in the wrist joint.

Hybrid approach that use different strategy for compound link pair and simple link pair makes it possible to satisfy both enough a range of movement for compound link pair such as wrist joint and enough safety margin for other link pairs.

B. Implementation of self collision checking software on HRP2 robot system

We have experimentally evaluated the performance of existing collision detection libraries. We have build real-time self collision detection software plugin on HRP2 Humanoid Robot System [7]–[9] which has 1266Mhz PentiumIII with 512KB Cache and 512MB Memory running ART Linux [10].

In the case of HRP2 which is based on the OpenHRP control system, several functions are implemented as a CORBA server that includes the ModelLoader server for reading the model data of a humanoid robot and the Dynamics server that provides forward kinematics calculation using the loaded model data. Runtime software for the HRP2 robot must be written as a plugin that is loaded into the Controller server and runs at every 5[msec] cycle. Each plugin receives current sensory data and joint angle data and output reference angle data for each joint. Since the cycle of the Controller is 5 [msec] and we assume that acceptable computation time for collision checking calculation is 1[msec] as described in the previous work [1].

C. Computational Cost

We tested both collision detection library(CD library) for non-convex polygon objects and collision detection library for convex polygon objects that approximating complex humanoid robot shape to much simpler convex hulls. We selected AABB based implementation(OPCODE [11]) as a collision detection

TABLE II

COMPUTATIONL TIME OF SELF COLLISION DETECTION SOFTWARE

Model	CD library	Computational time
Detailed	Opcode	0.83645
Convex hull	Vclip	0.71709

library which is reported as the fastest among other libraries [3]. For the collision detection library for a convex hull checking, we selected standard V-Clip library [12]. Table.II shows the result of comparison of computational time between two collision detection libraries. Both experiments take about 0.7-0.8[msec], which is sufficient for our application.

D. Collision detection motion experiment

Fig.8 shows the input motions for checking developed self collision detection software. The robot try to close it's arms as shown in figures. This motion results self collision that one hand get into another hand. Our self collision software successfully detects self collided links that have red, blue and yellow color in the figures. The red colored link indicates that the link is collided with other link in the simple link pair. The blue colored link shows that the link is collided with the compound link pair. Yellow color means the link is collided with both the simple link pair and the compound link pair.

It also shows that our system is able to keep walking while two arms are collided each other. We have simple heuristic based table to which limb is to be stopped and in the case of collision between two arms, two legs is not need to stop it's motion.

VI. CONCLUSION

In this paper, we present a hybrid approach to archive practical and real-time self collision system for humanoid robots that combines lookup table which guarantees a full range of movement for links in a compound joint and conventional online geometric model check for other link pairs with enough safety margin. For example, in the case of a wrist joint of our robot, the previous range of movement becomes 112[degree] from 79[degree] which uses geometric based collision detection system with 10[mm] margin.

We have experimentally evaluated our self collision detection system using a HRP2-JSK humanoid robot. Our demonstration shows that the robot automatically stops its motion when self collision occurs.

Next challenges includes: (1) Introducing motion planning technique to generate collision avoidance motion while guaranteeing an original task goal after detecting a self collision (2) Currently we have selected the collision detection margins experimentally which is 10[mm] for a simple link pair and 1[mm]] for a compound link pair. Some methodology including online margin selection with feed forward approach should be developed (3) Torque based self collision detection [13], [14] should be integrated since this method is able to detect collisions between a robot and an environment that

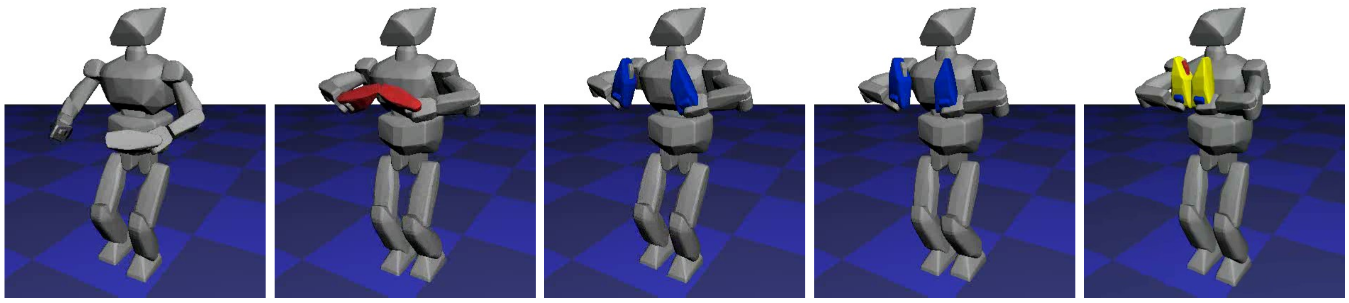


Fig. 8. Motion of HRP2-JSK Robot with Self Collided Motion(1)

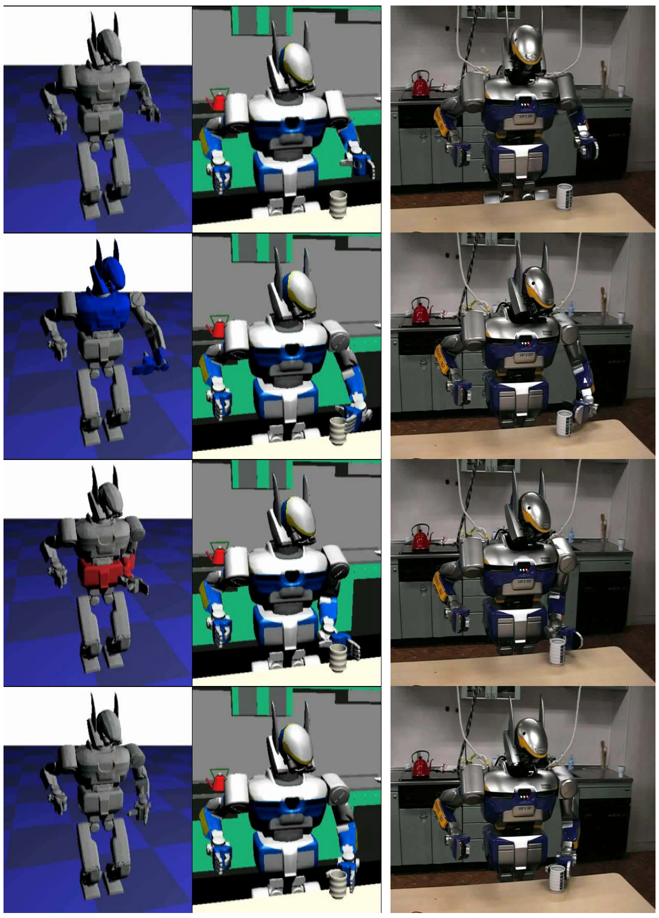


Fig. 9. Motion of HRP2-JSK Robot with Self Collided Motion(2)

the geometrical model based method is not able to detect in spite of it has drawbacks that it is potentially dangerous since collisions could not be detected until self collisions physically occurred.

The practical and real-time self collision software is one of the key functions for developing sensor based behaviors. In our laboratory, this software saves us from breaking robot hardware, unnecessary expenses and suspension of software development a number of times.

References

[1] F. Kanehiro, N. Miyata, S. Kajita, K. Fujiwara, H. Hirukawa, Y. Nakamura, K K. Yamane, T. Kohara, Y. Kawamura, and Y. Sankai. Virtual Humaonid Robot Platform to Develop Controllers of Real Humanoid Robots without Porting. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'01)*, pages 1093–1099, 2001.

[2] J.J. Kuffner, K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba, and H. Inoue. Self-Collision Detection and Prevention for Humanoid Robots. In *In Proceedings of 2002 IEEE International Conference on Robotics and Automation (ICRA '02)*, pages 2265–2270, 2001.

[3] Kei Okada, Masayuki Inaba, and Hirochika Inoue. Real-time and precise self collision detection system for humanoid robots. In *Proc. of International Conference on Robotics and Automation (ICRA'05)*, pages 1072–1077, 2005.

[4] K. Okada, T. Ogura, A. Haneda, J. Fujimoto, F. Gravot, and M. Inaba. Humanoid Motion Generation System on HRP2-JSK for Daily Life Environment. In *Proc. of International Conference on Mechatronics and Automation (ICMA'05)*, pages 1772–1777, 2005.

[5] M. Lin and S. Gottschalk. Collision Detection between Geometric Models: A Survey. In *In the Proceedings of IMA Conference on Mathematics of Surfaces*, 1998.

[6] M. Lin, D. Manocha, J. Cohen, and S. Gottschalk. Collision Detection: Algorithms and Applications. In *In Proceedings of Algorithms for Robotics Motion and Manipulation*, pages 129–142, 1999.

[7] H. Hirukawa, F. Kanehiro, K. Kaneko, S. Kajita, K. Fujiwara, Y. Kawai, F. Tomita, S. Hirai, K. Tanie, T. Isozumi, K. Akechi, T. Kawasaki, S. Ota, K. Yokoyama, H. Honda, Y. Fukase, J. Maeda, Y. Nakamura, S. Tachi, and H. Inoue. Humanoid Robotics Platforms developed in HRP. In *Proceedings of the 2003 IEEE-RAS International Conference on Humanoid Robots (Humanoids 2003)*, 2003.

[8] F. Kanehiro, K. Fujiwara, S. Kajita, K. Yokoi, K. Kaneko, and H. Hirukawa. Open Architecture Humanoid Robot Platform. In *Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA '02)*, pages 24–30, 2002.

[9] Kei Okada, Takashi Ogura, Atushi Haneda, Daisuke Kousaka, Hiroyuki Nakai, Masayuki Inaba, and Hirochika Inoue. Integrated System Software for HRP2 Humanoid. In *Proc. of International Conference on Robotics and Automation (ICRA'04)*, pages 3207–3212, 2004.

[10] Y. Ishiwata, T . Matsui, and Y. Kuniyoshi. Development of Linux which has advanced real-time processing function. In *Proceedings of the 16th Annual Conference of Robotics Society of Japan*, pages 355–356, 1998 (in Japanese).

[11] OPCODE – Optimized Collision Detection –. *http://www.codercorner.com/Opcode.htm*.

[12] B. Mirtich. V-Clip: fast and robust polyhedral collision detection. Technical Report TR-97-05, Mitsubishi Electrical Research Laboratory, 1997.

[13] T. Takahashi, S. Tamura, N. Onoe, T. Moridaira, K. Nagasaka, and M. Nagano. *Robot equipment and Motion Control Method*. Japan Patent, 2004-283954, 2004.

[14] F. Seto, K. Kosuge, R. Suda, and Y. Hirata. Real-time Control of Self-collision Avoidance for Robot using RoBE. In *Proceedings of the 2003 IEEE-RAS International Conference on Humanoid Robots (Humanoids 2003)*, 2003.