Robot Motion Planning with Many Degrees of Freedom*

Chenyu MA, Wei LI, Yang YANG
National Laboratory of Intelligent Technology and Systems,
Department of Computer Science, Tsinghua University, Beijing, 100084, P. R. China

Liuchen CHANG
Department of Electrical Engineering, University of New Brunswick, Canada E3B 5A3

ABSTRACT

Because a robot usually operates in environments with uncertain information, path planning for the robot should be performed in real-time so as to match process speed based on sensors. This paper presents an efficient approach to behavior based motion planning of a manipulator with many degrees of freedom, which is based on an algorithm for fast mapping obstacles from a robot's workspace (W-space) into its configuration space (C-space). In the paper, several types of behaviors are defined to formulate the manipulator motion, while rules of choosing behaviors are also given. The proposed method is suitable for the requirement on real-time robot path planning in uncertain environments.

1. INTRODUCTION

In recent years, the use of robots is playing more and more important roles in flexible manufacturing systems, the aerospace area and some other areas. The key issue of application of robots is automatically generating a collision-free path to avoid collision with obstacles in unknown environments. In this case, sensors must be used to acquire information on environments. This greatly demands that robots must have an ability to do path-planning work in real-time.

One of the most important issues in robot path planning is building the model of robot C-space. For robot motion planning, a great deal of research has been devoted to computing a C-space over the past 20 years. One of the most widely used approaches is to decompose the C-space into regular and irregular cells [1][2]. However, these C-space algorithms are not efficient enough to be practical in real-time path planning because large amounts of the computational time are needed to deal with robot's kinematics and geometry as well as the obstacles' geometry before searching for a path, and their computational cost grows exponentially with the number of degrees of freedom. Besides, the efficiency of path searching using cell decomposition approach depends largely upon the size of cells generated. If the decomposed cells are large, the searching process is fast but path searching may fail due to the loss of accuracy; if the decomposed cells are small enough, collision-free paths can be found, but the search process requires more execution time.

Another approach to obstacle mapping is to compute boundaries between a robot and obstacles in terms of the robot's kinematics on the assumption that the robotic arms and obstacles are all polyhedral. For instance, Lumelsky in [3] has studied different kinematics configurations corresponding to robotic arms with revolute or sliding joints. Furthermore, Ge and McCarthy have presented the algebraic equations of C-space obstacles [4]. Obviously, solving these algebraic equations is a time consuming work during path searching.

Recently, an approach to reflections at C-space obstacles is proposed for motion planning with many degrees of freedom [5][6]. However, this approach is suitable for known static environments. In addition, the graph built by random configurations can not preserve connectivity, although the free space is a connected set.

This paper proposes an approach to mapping an obstacle in real-time, and a behavior based method for path planning of a manipulator with many degrees of freedom. The article is organized as follows: In section 2 , the method of fast mapping obstacles from W-space to C-space is discussed [7][8][9][10]; Section 3 and section 4 will give a new method to behavior-based motion planning control of a manipulator with many degrees of freedom; Section 5 reports some of simulation results on robot's path planning to show the effectness of the proposed method; Section 6 gives the brief conclusions of the work and the future prospect of related problem.

* This research is supported by China "863" high technology project.

0-7803-2559-1/95 $4.00 © 1995 IEEE

2. FAST MAPPING OBSTACLES INTO C-SPACE

Usually, mapping obstacles from W-space to C-space needs large amount of computing time and memories for dealing with robot's kinematics and geometry. But, for a given robot, its kinematics and geometry information are unchanged, this leads to the idea to define the fundamental area and fundamental obstacles [7]. Since most robot's workspaces are symmetric, they can be formed by moving an area around symmetric axes of the workspaces. Such an area is known as a fundamental area of a robot. For example, for a two-link manipulator shown in Fig. 1, its fundamental area can be represented as in Fig. 2 . On a defined fundamental area, some grid points in the W-space are further defined as fundamental obstacles in [7][8] since they have the most fundamental geometry among all kinds of obstacles (they are zero-dimension obstacles). In Fig. 2, three fundamental obstacles A, B and C are shown as examples. The images of A, B and C in C-space, which are represented as C_A, C_B and C_C, can be seen from Fig. 3.

An important feature of fundamental obstacles is that they are independent of obstacles in a dynamic environment. Therefore, such fundamental obstacles can be pre-processed, i.e., they can be off-line mapped in configuration space according to robot's kinematics and geometry. Since the robot's kinematics and geometry are pre-processed, computation time for the work of mapping obstacles is greatly reduced, this gives probability for robot motion planning in real-time.

For mapping a 3D obstacle with complex geometry, the fundamental area and fundamental obstacles are used to decompose the 3D obstacle into a finite set of 2D obstacles and to model these 2D obstacles by fundamental obstacles. Since fundamental obstacles and their images describe the key relationship between the W-space and the C-space, mapping the 3D obstacle can be performed by a union operation. A key step in obstacle mapping by primitive obstacles is superimposing primitive images. Usually, their superimposition is performed by activating all irregular cells. However, many irregular cells needs to be activated more times since the majority of the primitive obstacles' images for mapping a same Cartesian obstacle completely or partially overlap with each other. In this paper, the idea of superimposing fundamental obstacles' images is to compute their contours by selecting critical fundamental obstacles of a Cartesian obstacle. Its computational cost is about constant in mapping a two-dimensional (2D) obstacle. An exact analytical model for computing the images of fundamental obstacles in C-space and a geometric algorithm for fast mapping a complex obstacles are presented in [11].

For quick path searching, slice C-space obstacles computed for building a free subspace [8]. In order not to deal with trivial parts of C-space that do not affect searching for a collision free path from start point to goal point. Since, using the proposed approach, the boundaries of the C-space obstacles are represented by line segments rather than irregular cells, this representation is effective for computing slice C-space obstacles to build a free subspace. Such a free subspace instinctively keeps a connectivity between starting and goal configurations and has more computation accuracy than a free subspace represented by slice projections [1].

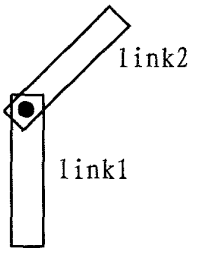


Fig. 1 a two-link manipulator

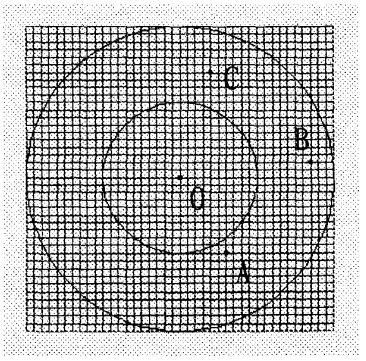


Fig. 2 fundamental area and fundamental obstacles

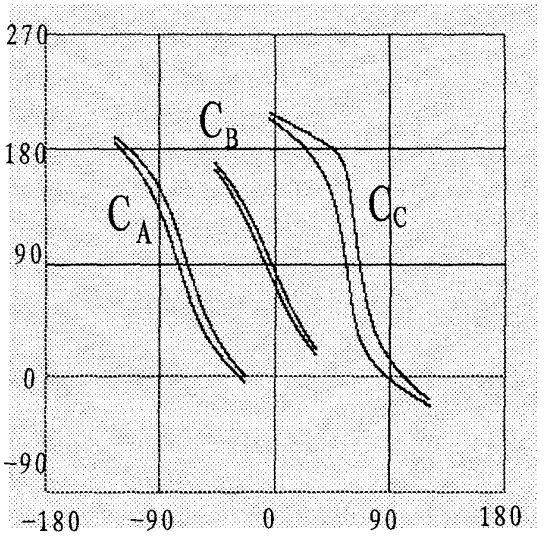


Fig. 3 images in the configuration space

3. DEFINING BEHAVIORS OF A MANIPULATOR

For a manipulator with many degrees of freedom, a key issue to path planning is letting its end-effector move from start

point to goal point without colliding with any obstacles, or searching for a collision free path. The information which the robot can obtain is from sensors like cameras, ultrasonic sensors, etc., which are set in the links of the manipulator or in environments. However, the information from sensors is usually so uncompleted that it is impossible for robot to build a precise and whole model in real-time. In this case, behavior based planning, which is on the basis of the stimulus-response behavior in bio-systems, have its advantage in solving this problem.

Fig. 4 shows a six-link manipulator with many degrees of freedom. The six links are respectively named link 1, link 2, .., link 6 from the base to the end of the manipulator. The problem of its motion planning lies in, to reach a specified target, which link and how many links should be moved? And what a kind of behavior should be fired?

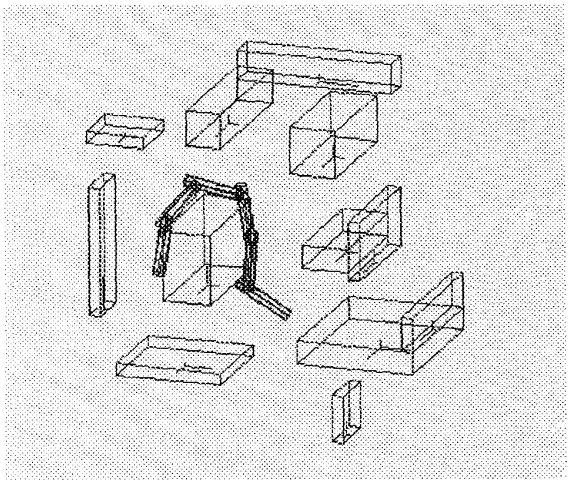


Fig. 4 a six-degree manipulator and obstacles

According to our knowledge, for manipulator motion planning with many degrees of freedom the following types of behavior should be defined:

Orienting to a target: This behavior is used to control the azimuth of a link to orientate a specified target.

Moving toward a target: This behavior is used to move the end-effector toward the target.

Avoiding obstacles: This behavior is used to avoid collision between links and obstacles.

Following edges: This behavior is used to control a link to follow edges of obstacles, if the end-effector can not orientate to the target due to the obstacles.

Folding up links: This behavior is used to control some links to fold up the end-effector from narrow channels.

Stretching links: This behavior is used to control some links to stretch the end-effector toward the target.

Reaching a target: This behavior is used to move the end-effector to reach a target when the target is located in a sub-workspace that is formed by the last two links i and i-1.

In order to formulate reactive behavior quantitatively, a manipulator with many degrees of freedom is decomposed into several two-link sub-manipulators, which can possess two or three degrees of freedom since such a two-link sub-manipulator constitutes a sub-workspace like a W-space of the manipulator in Fig. 1, and its collision-free paths can be efficiently planned by the approach discussed above. For example, the six-link manipulator described in Fig. 4 can be divided into 5 two-link sub-manipulators: sub-manipulator 1 consists of link 1 and link 2; sub-manipulator 2 consists of link 2 and link 3; ..., sub-manipulator 5 consists of link 5 and link 6.

4. FORMULATION OF REACTIVE BEHAVIORS

In order to formulate behavior, a base of rules must be built on the basis of defining different types of behaviors.

Orienting to a target

For a path planning work for a manipulator, the first idea is letting its links orientate to target, if its environments allow it to do so. This behavior is only available to the last sub-manipulator made by link 5 and link 6. Thus the following rule is acceptable:

Rule 1: If (sub-manipulator can orienting to the target) then

(orienting to the target)

Moving toward a target

The goal is to move the manipulator to the target, so behavior "moving to the target" can be done when there is no obstacle or obstacles are far from it in front of the moving-to-goal orientation:

Rule 2: If (there is no obstacle near the end-effector between current point to the goal point) then (moving toward the target)

Avoiding obstacles

When some other behaviors are fired, especially when the links near the base move, it is nearly impossible for robot's some other link not to collide with obstacles in complex environment. Thus behavior "avoid obstacles" is used to avoid collision between other links and obstacles. This "avoid collision" behavior may cause another "avoid collision" behavior:

Rule 3: If (behavior's result will cause link i's collision with obstacles) then (let link i avoiding obstacles)

In fact, this rule should a group of rules containing every link of the robot.

Following edges

In the case when the robot can not orientate to the goal because the obstacle do not allow it to do so, the behavior, letting the link follow the edges of the obstacle, is very important for robot to moving to target. This behavior usually happens only for the last two links.

Rule 4: (If robot can not orientate to target) then (let link j "following edges")

Folding up links

When there is a narrow channel affecting manipulator moving to target, behavior "folding up links" is a useful behavior. Often, this behavior starts from the link near the base of the manipulator other then any other links.

Rule 5: (If robot can not move toward target and link 1,..,j-1 are already folded up) then (folding up link j)

This behavior often causes another same behavior fired for other links, that means, there are always a lot of links are needed to folded up.

Stretching links

When the end manipulator can not arrive at target and there is no a lot of obstacle between end-effector and the goal point, behavior "stretching links" is fired. Different from "folding up links", this behavior begins from links near the end-effector.

Rule 6: (If manipulator j can not arrive at target and link j is folded up and all of link k are not folded up, k>i) then (stretching link j)

This behavior often cause another same behavior fired for other links like behavior "folding up links".

Reaching a target

When robot have the path to move to target, Just do it. The case happens when the target is located in a sub-workspace that is formed by the last two links i and i-1 and this is a path.

Rule 7: (if there is a path in sub-workspace formed by the last 2 links) then (reaching the target)

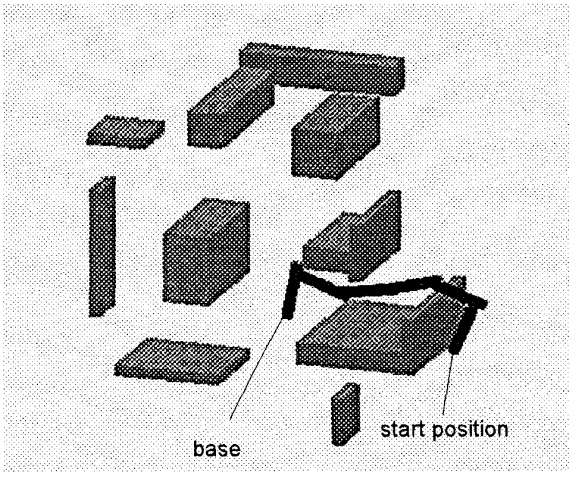
Through above rules, a behavior based method can be realized for robot motion planning with many degrees of freedom. The robot can choose its behavior according to its position, the circumstance of the robot and the position of a target.

5. EXPERIMENTAL RESULTS

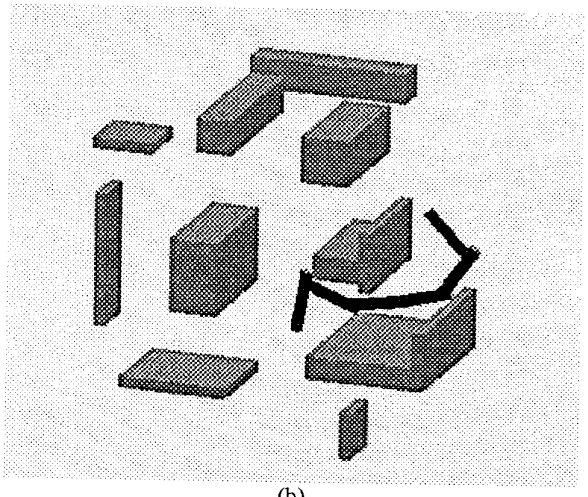
In order to demonstrate the effectiveness of the proposed approach, we report graphical simulation experiments.

Fig. 5 simulates the manipulator motion from a specified start position and a specified target position. This work is finished in SGI workstation. Fig. 5(a) through 5(h) describe some main configuration of the six-link manipulator when its end-effector moves from the start point to the goal point, among them Fig. 5(a) and 5(h) show the initial configuration and the last configuration.

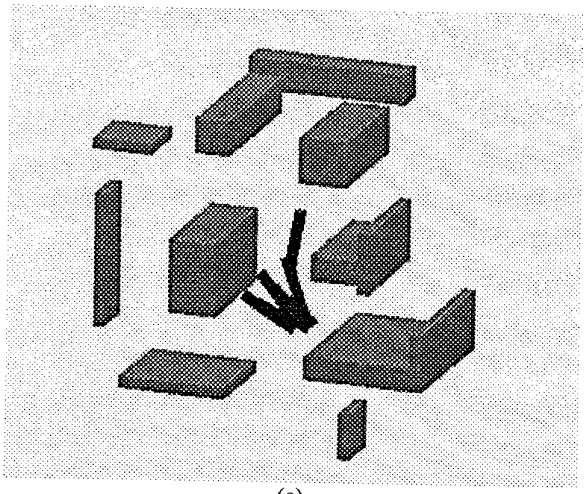
Since there is no obstacle affecting robot "orienting to target", the first behavior robot performed, showing in Fig. 5(b), is just pointing to the goal. Since robot can not "moving to target", so it goes along the edge of the obstacles, this can be seen from Fig. 5(c). In Fig. 5(c) robot can not go out from the narrow channel, so it choose behavior "folding up links", which is composed of "folding up link 2", "folding up link 3", and "folding up link 4" (Fig. 5(d)). Thus the robot has enough space to move its end-effector. However, a obstacle stands there avoiding robot's "moving to target", so robot performs "following edges" to move its link 5 and link 6 to find a path in C-space to move to another point (position in Fig. 5(e)). In Fig. 5(f), having finished its "following to edge" behavior, it start to "orienting to target" and "moving to target", but it still can not "reaching to target". Since there is no a lot of obstacles in front of the robot, it can stretch its links. Now, the robot repeated behavior "orienting to target" and "stretching links" in Fig. 5(g). At last, as shown in Fig. 5(h), robot performs "reaching target" ,finishing its task.



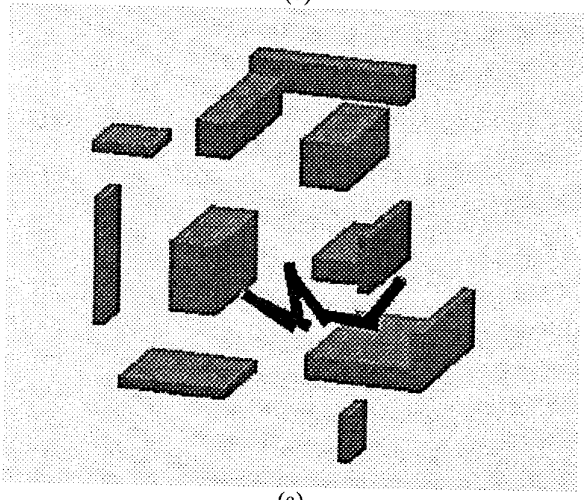
(a)



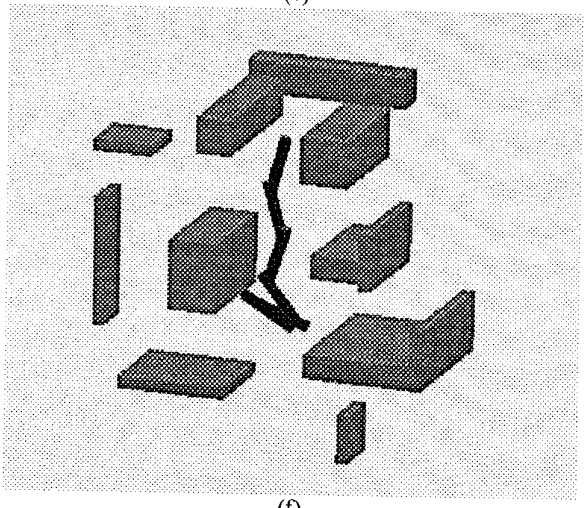
(b)



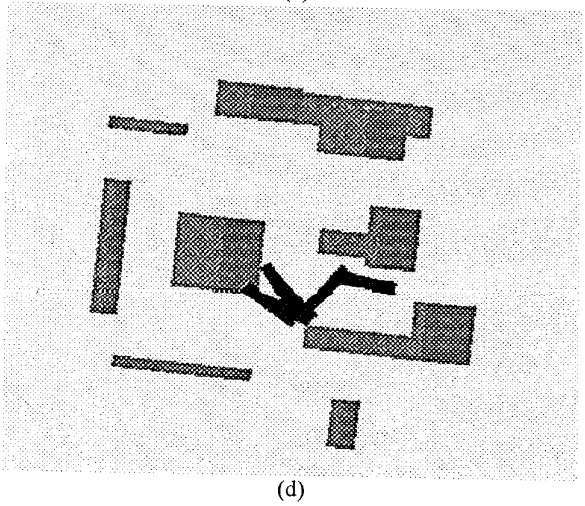
(e)



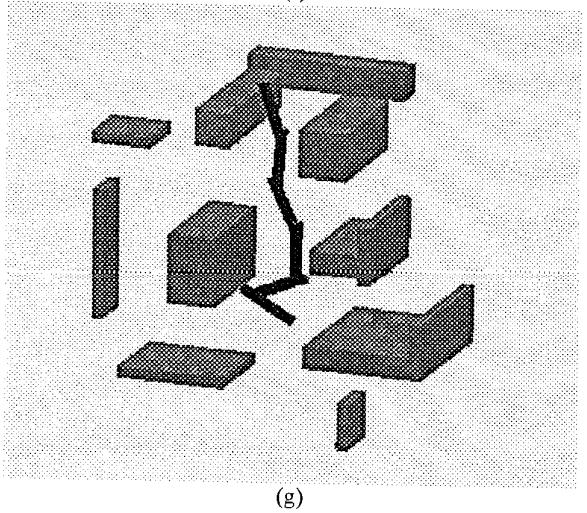
(c)



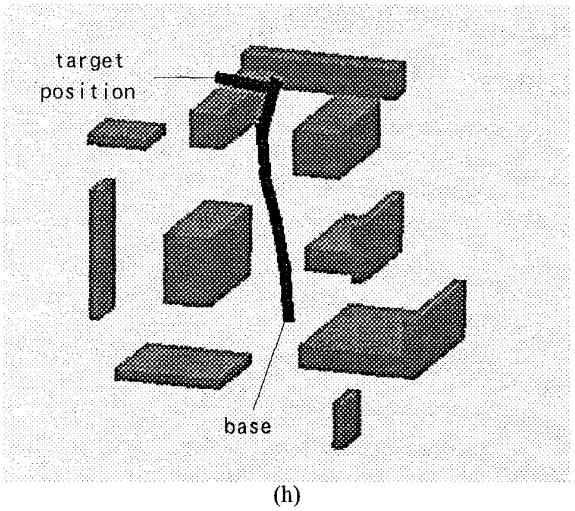
(f)



(d)



(g)



(h)

Fig. 5 (a)-(h) motion planning of a six-degree manipulator: from start point to target point

6. CONCLUSIONS

This paper presents a new method to motion planning for manipulators with many degrees of freedom, which is based on an algorithm for fast mapping obstacles from a robot's workspace into its configuration space.

Motion planning with many degrees of freedom suffers mainly from kinematically redundancy, i.e., it is hard to determine which links and how many links should be moved so that the end-effector can reach a target point from a start point in complex environments. Using reactive behavior with simple features, complex motion planning can be qualitatively formulated with ease. In order to realize these types of behavior quantitatively, an efficient algorithm is adopted for real-time motion planning with few degrees of freedom.

This approach has the following advantages:

First, motion planning is very fast, e.g., computational time for the examples in this paper is approximately from 300ms to 500ms in SGI Indigo workstation. This provides enough time for robot to process its searching for path.

Secondly, this method can be used in motion planning in uncertain environments. The robot do not need to know all information about its environment. All it needs to know is the position of the obstacles near it, which can easily acquired by sensors.

Furthermore, the singularity problem can be avoided since it is not necessary to compute inverse kinematics with redundancy.

Finally, this method can be easily implemented.

Further works can be done about how to choose rules in order to improve the effect of the method, how to set sensors in right position in the manipulator or/and in environments.

7. REFERENCES

[1] T. Lozano-Perez, "A Simple Motion - Planning Algorithm for General Robot Manipulators", *IEEE J. Robotics and Automation*, RA-3, 1987, pp.224-238.

[2] B. Faverjon, "Obstacle Avoidance Using an Octree in the Configuration Space of a Manipulator",*Proc. of IEEE Int. Conf. Robotics*, Atlanta, Mar. 1984, pp.504-512.

[3] V. Lumelsky, "Effect of Kinematics on Motion Planning for Planar Robot Arms Moving Amidst Unknown Obstacles", *IEEE J. Robotics and Automation*, RA-3, 1987, pp.207-223.

[4] Q. Ge and J.M. McCarthy, "An Algebraic Formulation of Configuration-Space Obstacles for Spatial Robots",*Proc. of IEEE Int. Conf. Robotics and Automation*, Cincinnati, Ohio, May 1990, pp.1542-1547.

[5] L. Kavraki, J. Latombe, "Randomized preprocessing of configuration space for path planning: Articulated Robots", *Proc. of 1994 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1994, pp.1764-1771.

[6] T. Horsch; F. Schwarz and H. Tolle, "Motion planning with many degrees of freedom-random reflections at C-space obstacles", *Proc. of 1994 IEEE Int. Conf. Robotics and Automation*, 1994, pp.3318-3323.

[7] W. Li, "Automatische Bestimmung Kollisionsfreier Bewegungsbahuen f ü r Industrieroboter", *Robotersysteme*, Vol. 6, 1990, pp.218-244.

[8] W. Li et al., "Solving the robotic 'Pick-and-Place' Pathfind Problem", *Manufacturing Review*, Vol. 6, No. 2, 1993, pp.114-129.

[9] W. Li, "Fast mapping obstacles in the configuration space", *Robotersysteme*, Vol. 7, 1991, pp.148-154.

[10] W. Li, "On real-time motion planning for robot application in flexible manufacturing systems",*Proc. of International Conference on Intelligent Manufacturing '95*, 1995, pp.628-633.

[11] C. Ma, W. Li et al., "Fast Computation of Configuration Space for Space Robot Path Planning",*Journal of Tsinghua University*, (in press), No. 5, 1995.