2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Easy Robot Programming for Industrial Manipulators by Manual Volume Sweeping

Yusuke MAEDA, Tatsuya USHIODA and Satoshi MAKITA

Abstract— **In this paper, we propose a robot programming method for industrial manipulators. It enables novice human operators to teach "good" robot motions in a short period of time. First in this method, a human operator makes a robot manipulator sweep a volume by its bodies. The manipulator is equipped with a force sensor on its wrist and damping-controlled; the operator can move it freely by grasping its end-effector. The swept volume stands for a part of the manipulator's free space, because the manipulator has passed through the volume without collisions. Next, the obtained swept volume is used by a motion planner to generate a well-optimized path of the manipulator automatically. Finally, the planned motion is executed by the manipulator. Even non-skilled operators can generate robot motions with short cycle time by doing the above procedure. The effectiveness of our method is successfully demonstrated in teaching experiments for a comparison with other conventional methods.**

I. Introduction

Robot programming is indispensable for current industrial manipulators to execute tasks. Human operators have to teach motions in detail to robot manipulators by, for example, conventional teaching playback. However, robot programming is complicated and time-consuming for novice operators and the cost for training them is often unaffordable in small-sized companies. That is one of the biggest issues that prevent the dissemination of robot utilization. Accordingly, easy robot programming methods are highly demanded.

There have been some efforts to ease the difficulty of robot programming. Offline programming [1] [2] is becoming popular because it offers safe robot programming without occupying actual robots. Techniques for robot motion planning, which have been studied extensively for a few decades (e.g., [3]–[6]), are available on some offline programming systems; automatic generation of (sub-)optimal robot motions is possible.

However, offline programming has a difficulty in coping with the mismatch between the virtual and real worlds. All the robots have errors in their alignment and motions. Therefore robot motions calculated in offline programming systems must be modified to cancel such errors—this is not trivial stuff. Moreover, for automatic planning of collision-free motions, obstacles around robots must be modeled and inputted to the systems. This is also cumbersome.

Y. MAEDA is with Division of Systems Research, Faculty of Engineering, Yokohama National University, 79-5 Tokiwadai, Hodogaya-ku, Yokohama 240-8501 JAPAN, maeda@ynu.ac.jp

T. USHIODA and S. MAKITA are with Maeda Lab, Dept. of Mechanical Engineering, Div. of Systems Integration, Graduate School of Engineering, Yokohama National University.

As for online programming, lead-through teaching [7] [2] (or direct teaching [8]) is used as an easy robot programming method. In direct teaching, human operators can move manipulators freely by grasping their end-effectors with the help of force sensors attached on them. Then, desired paths are taught as sequences of configurations and played back accurately by the manipulators thanks to their high repeatability. This technique makes teaching-playback intuitive even for novice operators. In this method, however, human operators must teach "good" robot motions by themselves yet. The optimal robot motions are not straightforward because of the nonlinearity of the robot kinematics and the differences in specifications among the joints, and therefore it is difficult for novice operators to teach robot motions with short cycle time.

In this paper, we present another approach to easy robot programming. In our method, a human operator moves a manipulator around to sweep a volume by its bodies, and the swept volume is used for motion planning. This method enables unskilled operators to generate robot motions with short cycle time.

In the next section, we outline our proposed method for robot programming. In Section III to V, each of the steps that constitute our method is described. In Section VI and VII, our method was tested in teaching experiments in comparison with other conventional approaches. Finally, we conclude this paper in Section VIII.

II. Overview of Proposed Robot Programming

Hasegawa et al. presented an idea to obtain free space information for robot manipulators based on swept volumes by teleoperation [9]. A volume swept by manipulator's bodies stands for (a part of) the free space of the manipulator, because the manipulator has passed through the volume without collisions. Similar idea can be also found in [10], where a swept volume by the manipulated object is used to generate collision-free paths. In this paper, we apply Hasegawa's idea to ordinary robot programming, by using manual volume sweeping instead of teleoperation.

The teaching procedure in our method is as follows (Fig. 1):

1) A robot manipulator is set to be in damping control. A human operator moves the manipulator around so that it does not collide with obstacles and sweeps a volume by its bodies (Fig. 2). Configurations of the manipulator (i.e., all the joint variables) are recorded during the manual volume sweeping.

978-1-4244-1647-9/08/$25.00 ©2008 IEEE.

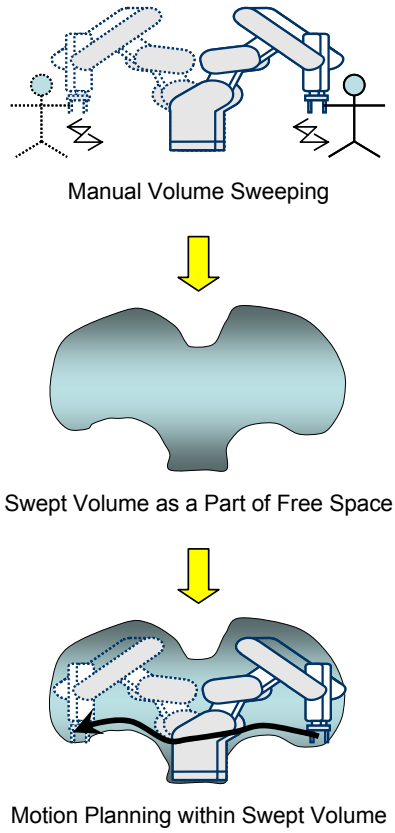


Fig. 1. Overview of Proposed Robot Programming Method

2) The swept volume by the bodies of the manipulator is calculated from the recorded configurations (Fig. 3). The swept volume is a part of the free space of the manipulator [11], because the manipulator has passed through the volume without collisions.
3) A motion planner generates a suboptimal path from a start configuration to a goal in the swept volume (Fig. 4).
4) Finally, the manipulator follows the planned path. (Fig. 5).

By manual volume sweeping, information on the free space of the manipulator can be obtained easily and intuitively. Then, a robot motion planning technique is employed to generate a well-optimized robot motion within the free space automatically. The above procedure enables even non-skilled operators to generate robot motions with short cycle time. We describe more details of the procedure in the following sections.

III. Manual Volume Sweeping

First in our robot programming method, a robot manipulator is force-controlled (or damping-controlled) so that it can be moved freely by external forces. Then a human operator moves the manipulator manually, like direct teaching (Fig. 2). However, unlike direct teaching, the desired path of the manipulator is not taught; the operator just moves the manipulator around so that its bodies sweep a volume

Fig. 2. Manual Volume Sweeping

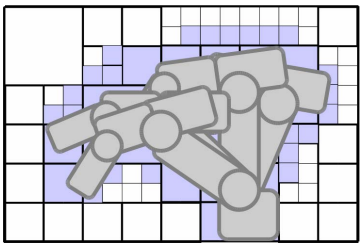


Fig. 3. Calculation of a Swept Volume

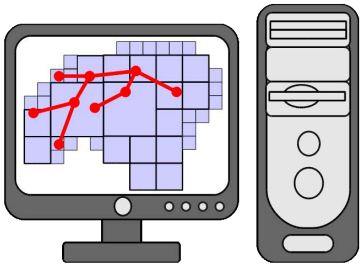


Fig. 4. Motion Planning

Fig. 5. Execution of Planned Motion

without collisions. The configurations of the manipulator during the volume sweeping are recorded.

The swept volume in Cartesian space is three-dimensional, while the motion of the manipulator during the volume sweeping forms only a one-dimensional path in the configuration space. The swept volume constitutes a part of the free space of the manipulator; that is, the manipulator can move freely within the volume without collision. Note that the volume in Cartesian space contains the manipulator's configurations through which it has *not* passed during volume sweeping; that allows a motion planner to generate a variety of paths.

In this step, the human operator can obtain information on the free space of the manipulator by very intuitive operation that does not need high skills. The information is used for motion planning.

The start and goal configurations of the manipulator are taught additionally during this manual volume sweeping in our method, because it is necessary to achieve high positional accuracy especially around the start and the goal configurations. The repeatability of industrial manipulators, on which their positional accuracy at the start and goal depends, is much better than the absolute accuracy of them, on which the accuracy of swept volumes depends.

IV. Calculation of Swept Volumes

Next we calculate an explicit representation of the swept volume from the configurations recorded in the previous step and the shapes of the manipulator's bodies. Because it is not easy to obtain an exact expression of the swept volume, we use an octree to represent it approximately.

First, we sort the recorded configurations and remove duplicate ones. Then the octree is constructed incrementally by checking interference between the manipulator bodies at each instant and voxels that composes the workspace of the manipulator. When the interference checking is finished for all the recorded configurations, we obtain an octree that represents (a part of) the free space of the manipulator (Fig. 6).

In our current implementation, an octree for the occupied volume by the innermost immobile body (link 0) is calculated first. Second, the swept volume by link 1 is added to the octree. We can skip many of the recorded configurations for calculating swept volume by link 1 because of the duplication of innermost joint angles. Then, we shuffle the order of the joint data sets for calculation of the swept volumes by link 2 through link n (terminal link). The shuffle leads to effective computation in most cases; the effect of skipping the duplication of joint angles is not significant for link 2 and after, and the octree grows slowly when it is constructed incrementally for the sorted joint data because neighbors of the sorted data usually have only slight differences.

V. Motion Planning and Execution

The calculated swept volume in the previous step corresponds to the free space of the manipulator. In other words, the complement of the swept volume can be regarded as virtual obstacles. Thus, we run a motion planner to generate a path between the start and goal configurations without collision with the virtual obstacles. The path should be well-optimized so that the manipulator can move along it in a short time. Our method does not impose special requirements on the motion planner, thus we can choose any one among many available options.

Finally, a robot motion, which follows the planned path at the highest speed, is executed by the manipulator.

VI. Experimental Setup

We prepared an experimental setup for teaching experiments as shown in Fig. 7.

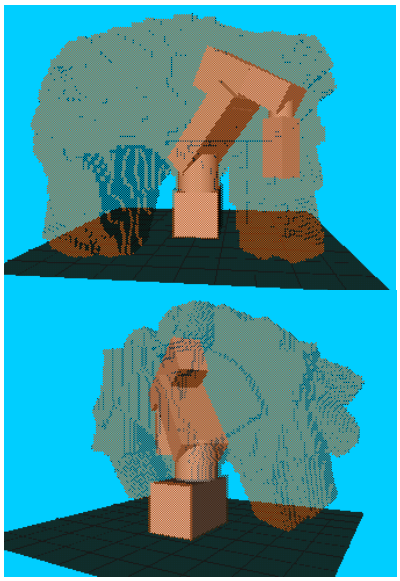


Fig. 6. Swept Volume

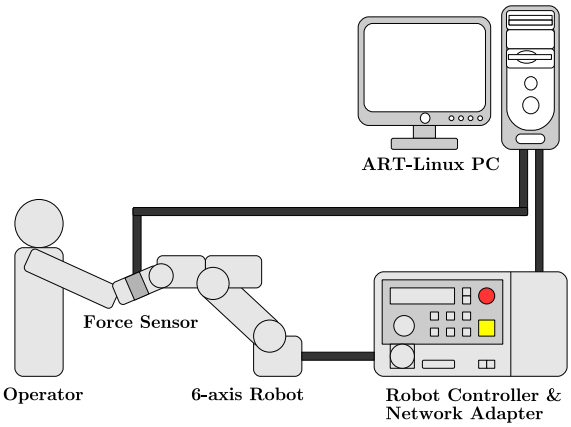


Fig. 7. Experimental Setup

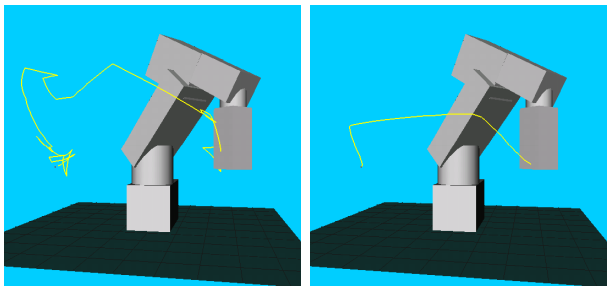
A. Hardware Setup

We used a 6-axis industrial manipulator, RV-1A by Mitsubishi Electric. We can send a motion command from a PC to the manipulator at 7.1 [ms] intervals via Ethernet. The actual joint data can be received from the manipulator by the PC at the same intervals. A 6-axis force/torque sensor, 67M25A-140 by Nitta, is attached at the wrist of the manipulator for force control.

B. Software Setup

Robot control programs for manual volume sweeping and motion execution run on a real-time OS, ART-Linux [12]. In manual volume sweeping, the manipulator is damping-controlled with a small deadband for external forces.

We developed a program for computation of swept volumes based on FreeSOLID [13], an open source library for interference detection. The manipulator is modeled as a kinematic chain of five rigid cuboids and two rigid cylinders in the program. An octree to represent a swept volume is



(a) Original Path (b) Smoothed Path

Fig. 8. Planned Path by MPK

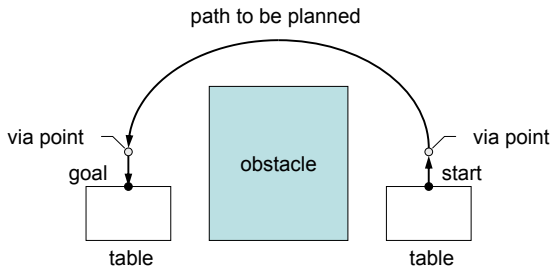


Fig. 9. Target Task

constructed by dividing a cube measuring 1200 [mm] per side, which fully contains the Cartesian workspace of the manipulator, up to depth level 7; that is, the minimum voxel in the octree is about 9.4 [mm] $\left(= 1200/2^7 \text{ [mm]}\right)$ per side.

We used Motion Planning Kit (MPK) [14] to generate collision-free paths. SBL (a single-query bi-directional probabilistic roadmap planner with lazy collision checking) [15] is available on MPK and paths planned by SBL can be improved by the smoothing functionality of MPK (Fig. 8). Each joint variable of the manipulator was normalized by its maximum speed so that the "shortest" path considering the difference of joint velocity specifications could be generated.

Calculation of swept volumes and motion planning were performed on a Linux PC with a Core 2 Quad Q6600 CPU (2.4GHz).

C. Target Task

We chose pick-and-place as the target task in the teaching experiments. For simplicity, however, we skipped open/close of the gripper in the experiments. The manipulator moves from a start configuration to a goal configuration avoiding an obstacle (Fig. 9).

Because it is difficult to sweep a volume in the neighborhood of the start and goal positions without digging into the ground, we set two via points above the start and goal points. The via points are taught in the step of manual volume sweeping, and the manipulator moves straight before the first via point and after the second via point. Accordingly, only the path between these via points is planned by MPK.

VII. Teaching Experiments

A. Conditions in Experiments

Our proposed method was tested in the teaching experiments in comparison with other conventional approaches. The details of the tested methods are as follows:

Robot programming by manual volume sweeping. This is our proposed method. In order to suppress the effect of the randomness in motion planning by SBL, we generate three different paths for each trial; the three paths are smoothed twice for each, and the best path of the three is selected. The manipulator follows the path at its highest speed.

Teaching playback using a teach pendant [7] [2]. A human operator teaches points by operating the manipulator using a teach pendant. Then the operator inputs a program in MoveMaster language with the pendant so that the manipulator moves along the taught points with linear interpolation at its highest speed.

Teaching playback by direct teach [8]. A human operator teaches points by direct teaching; the operator grasps the end-effector of the force-controlled manipulator and moves it to the points. Then the operator inputs a program with the pendant so that the manipulator moves along the taught points with linear interpolation at its highest speed.

Five male experimenters in their twenties who had no experiences in robot programming of industrial manipulators tried the above three approaches.

B. Experimental Results

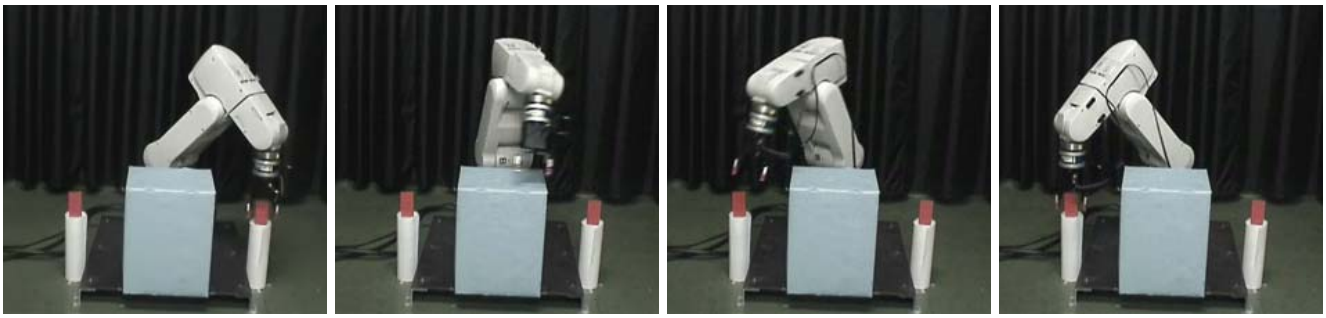
Teaching experiments were conducted by the five human operators. Some scenes of motions programmed by Operator B with our proposed method are shown in Fig. 10. Table I is the results of the calculation of swept volumes by our method.

The total time required for robot programming is depicted in Fig. 11. Apparently, "Direct Teach" is the best with regard to the total time, and our proposed method is worse as well as "Teach Pendant." However, as for the time required for manual operations (i.e., non-automated operations for robot programming), our proposed method is comparable with other conventional methods (Fig. 12). This is because considerable time for robot programming in our proposed method is dedicated to automatic computation: calculation of a swept volume and motion planning (Fig. 13).

The cycle times of the programmed motions are shown in Fig. 14. Our method achieved the shortest cycle time (22–44% shorter than those of the other conventional robot programming methods) for four operators (A through D) out of five. Thus, our proposed method can generate robot motions with short cycle time by small manual operations in most cases.

C. Discussion

The above results indicate that our method is an easy robot programming method for novice operators while it can generate "good" robot motions. Of course, our method might



(a) start (b) 0.8 [s] (c) 1.6 [s] (d) goal (1.9 [s])

Fig. 10. Motion Execution (Our Proposed Method)

TABLE I
CALCULATED SWEPT VOLUMES

Operator	A	B	C	D	E
# of recorded configurations	52 714	60 654	73 806	48 489	85 097
# of unique configurations	49 804	55 893	68 400	42 959	82 739
# of nodes in octree	24 477	25 652	26 748	23 644	22 304
swept volume [m^3]	0.174	0.184	0.191	0.171	0.152

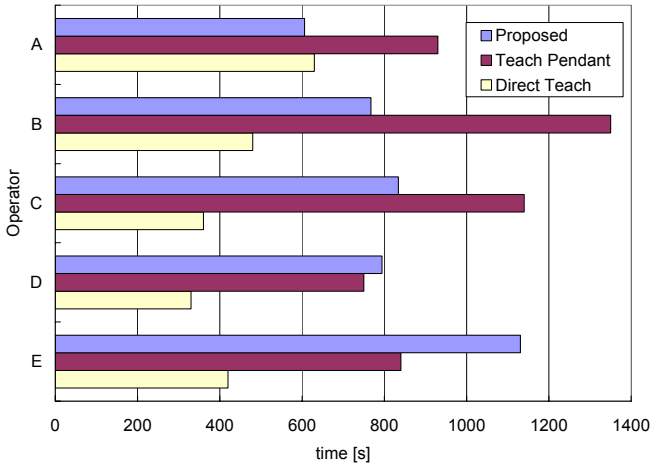


Fig. 11. Total Time for Robot Programming

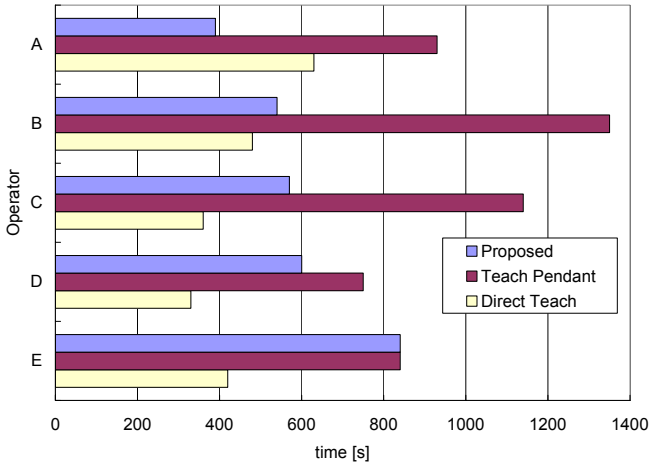


Fig. 12. Time for Manual Operations

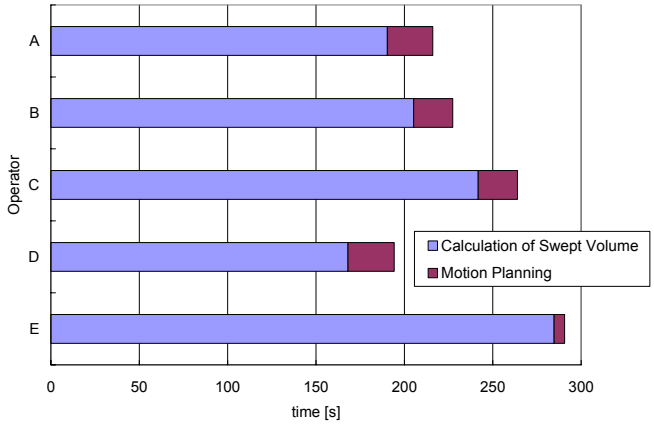


Fig. 13. Time for Computation in Our Proposed Method

be unsatisfactory for experts of robot teaching; it is suitable to unskilled operators.

A difficulty in our proposed method is the possibility of the failures of motion planning. In the case of Operator E, whose swept volume was the smallest, MPK failed to find a path from the start configuration to the goal, possibly because of the narrow passage problem; note that a swept volume may have narrow passages even if the actual free space does not. A simple guideline for human operators in manual volume sweeping, for example, moving the manipulator back and forth several times between the start and the goal configurations, may be helpful to avoid such problems. When motion planning fails, however, we can use a path that is taught in the step of manual volume sweeping. This path might be roundabout, but can be improved by smoothing to some extent. As a second alternative, we can perform additional volume sweeping to obtain more information on the free space of the manipulator.

Another difficulty is the computation time. Even though

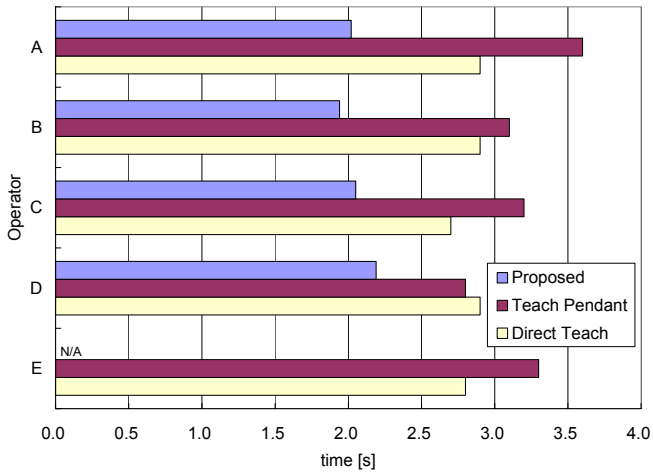


Fig. 14. Cycle Time of Programmed Motions

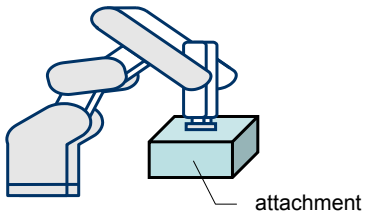


Fig. 15. Attachment for Efficient Volume Sweeping

human operators can do other things during the computation, shorter total time for robot programming is desirable. Around 25% of the total time were spent in the calculation of swept volumes as shown in Fig. 13. Its worst-case computational complexity is $\mathcal{O}(8^d n)$, where n is the number of recorded configurations and d is the maximum depth level of the octree to represent swept volumes; that is, time for the calculation of swept volumes depends on the resolution of them, thus further efforts for reducing the computation are required to obtain more accurate information on swept volumes. A way to reduce the total time is computing the swept volume concurrently with manual volume sweeping. Another easy solution is parallel computing; each subtree of the octree can be calculated in completely parallel and therefore additional computers can reduce the computation time significantly. We ran four processes in parallel on a PC with a quad-core CPU to compute the swept volume in this paper.

Some of the operators reported that it was hard in our proposed method to judge whether the sufficient volume was swept or not during manual volume sweeping. Real-time display of swept volumes would help operators understand the progress of manual volume sweeping.

Manual volume sweeping can be more effective by using a suitably-shaped attachment on the manipulator as shown in Fig. 15. Such an attachment may be needed when the grasped object is large and therefore the volume which can be swept by the bodies of the manipulator is too small to plan collision-free paths. The use of such an attachment can be also useful for safety in manual volume sweeping when its material is chosen appropriately.

VIII. Conclusion

In this paper, we proposed an easy robot programming method for industrial manipulators. The method is based on the idea of Hasegawa et al. [9] that a volume swept by manipulator's bodies stands for (a part of) the free space of the manipulator.

In our proposed method, after the manual volume sweeping, which can be done by a human operator without special skills and long-time operations, a well-optimized robot motion is automatically generated and executed. This approach showed good performance in comparison with other conventional robot programming methods in most cases in the teaching experiments.

In future work, we will test the effectiveness of our method in various other robot tasks. We will also improve it further from the viewpoints pointed out in Section VII-C.

References

[1] Y. F. Yong, J. A. Gleave, J. L. Green, and M. C. Bonney, "Off-line programming of robots," in *Handbook of Industrial Robotics*, S. Y. Nof, Ed. Wiley, 1985, pp. 366–380.

[2] Y. Shamash, Y. Yang, and Z. Roth, "Teaching a robot," in *International Encyclopedia of Robotics: Applications and Automation*, R. C. Dorf, Ed. Wiley, 1988, pp. 1689–1701.

[3] Y. K. Hwang and N. Ahuja, "Gross motion planning—a survey," *ACM Computing Surveys*, vol. 24, no. 3, pp. 219–291, 1992.

[4] Z. Shiller, "Optimal robot motion planning and work-cell layout design," *Robotica*, vol. 15, no. 1, pp. 31–40, 1997.

[5] J.-C. Latombe, "Motion planning: A journey of robots, molecules, digital actors, and other artifacts," *Int. J. of Robotics Research*, vol. 18, no. 11, pp. 1119–1128, 1999.

[6] S. Ando, "A fast collision-free path planning method for a general robot manipulator," in *Proc. of 2003 IEEE Int. Conf. on Robotics and Automation*, 2003, pp. 2871–2877.

[7] M. P. Deisenroth, "Robot teaching," in *Handbook of Industrial Robotics*, S. Y. Nof, Ed. Wiley, 1985, pp. 352–365.

[8] H. Hirabayashi, K. Sugimoto, S. Arai, and S. Sakaue, "Virtual compliance control of multiple degree of freedom robot," *Trans. of Soc. of Instrument and Control Engineers*, vol. 22, no. 3, pp. 343–350, 1986, in Japanese.

[9] T. Hasegawa, K. Nakagawa, and K. Murakami, "Toward on-line transition to autonomous teleoperation from master-slave manipulation," in *Proc. of IEEE Int. Conf. on Robotics and Automation*, 2004, pp. 3715–3720.

[10] N. Delson and H. West, "Robot programming by human demonstration: The use of human inconsistency in improving 3d robot trajectories," in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1994, pp. 1248–1255.

[11] K. Abdel-Malek, J. Yang, D. Blackmore, and K. Joy, "Swept volumes: Foundation, perspectives, and applications," *Int. J. of Shape Modeling*, vol. 12, no. 1, pp. 87–127, 2006.

[12] Y. Ishiwata, "ART-Linux," http://art-linux.sourceforge.net/.

[13] G. van den Bergen, "FreeSOLID," http://sourceforge.net/projects/freesolid/.

[14] F. Schwarzer, M. Saha, and G. Sanchez, "Motion Planning Kit," http://ai.stanford.edu/~mitul/mpk/.

[15] G. Sánchez and J.-C. Latombe, "A single-query bi-directional probabilistic roadmap planner with lazy collision checking," *Springer Tracts in Advanced Robotics*, vol. 6, pp. 403–418, 2003.