2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Error Recovery Using Task Stratification and Error Classification for Manipulation Robots in Various Fields

Akira Nakamura, Kazuyuki Nagata, Kensuke Harada, Natsuki Yamanobe,
Tokuo Tsuji, Torea Foissotte and Yoshihiro Kawai

Abstract— **Dexterous manipulation is an important function for working robots. Manipulator tasks such as grasping, assembly and disassembly can generally be divided into several motion primitives. We call such motion primitives "skills" and explain how most manipulator tasks can be composed of sequences of these skills. We will address the issues involved with various types of robots such as maintenance robots and service robots. We have considered hierarchizing the manipulation tasks of these robots since their tasks have become more complex than ever before. Additionally, as errors are seen likely to increase in complex tasks, it is important to implement effective error recovery technology. This paper presents our proposal for a new type of error recovery that uses the concepts of task stratification and error classification which can be expressed specifically using flow charts.**

I. INTRODUCTION

To be useful in several fields, manipulation robots need to be able to perform various tasks using special techniques. We analyzed human motions in such tasks as assembly and disassembly and found that typical movements consisted of several significant motion primitives. We call these motion primitives "skills" and have demonstrated that most tasks of a manipulator robot can be composed of sequences of such skills [1]–[5]. In the hierarchy of manipulator control, skill level control is positioned between task level control and servo level control. Programmers can create a task program easily as a sequence of skills without needing to take into account servo level control.

We have researched maintenance robots working in various types of plants, including power plants [3]-[6]. We have also considered manipulation robots used for the maintenance of household electrical appliances and consumer electronics [7]-[9]. The robot opens and closes the cases of system components and personal computers for replacing parts. Such maintenance tasks require the use of many manipulation skills, and the composition of those tasks is complex. Therefore, we considered employing task stratification with the manipulation skills to make the development more manageable [10].

Manipulation tasks with skills are performed in theory by sequences of, for example, visual sensing, geometric modeling, planning and execution. During actual manipulation, however, errors often occur for various reasons and processes are interrupted. Failures can be caused by errors in execution, planning, modeling and sensing. We have grouped such errors into several classes according to their potential causes. If an error occurs, the parameters of planning, modeling or sensing are corrected as required by specifying the class, and then the task process is performed again using the corrected data after it returns up to a certain step according to the error classification. We have already proposed a method of error recovery that uses the concept of error classification, as shown in Reference [10]. Thereafter, we further considered error recovery for stratified tasks by using a method of shifting to higher layer tasks if necessary, and we have been able to apply our approach to complex maintenance tasks.

To make this approach useful in more wide-ranging fields, it is necessary to consider error recovery techniques for robots active in other fields besides maintenance. Therefore, we take into account the tasks of a service robot operating with humans in a daily life environment [11]-[12]. There are two kinds of operations, local transition with manipulation and global transition with transfer, and these operations are repeated many times in a service robot. Both sensing before execution to complete the planning and sensing after execution to determine an error are generally needed, so the task sequence may be more complicated than that of a maintenance robot. Furthermore, a supplementary module that slightly corrects the execution result is added into the error recovery process for each motion primitive. The recovery passes through *a forward correction process*, so it is different from the above-mentioned recovery through *a backward correction process.* The range of the error recovery is extended by adopting both types of methods. In this paper, we propose a new type of error recovery that uses the concepts of task stratification and error classification.

After a brief literature review in the next section, Section III explains manipulation skills and the stratification of tasks using an example of a maintenance robot. The classification of errors and error recovery in the task hierarchy are presented in Section IV. The tasks of a service robot are explained in Section V, and our improved error recovery process is shown using a flow chart in Section VI.

A. Nakamura, K. Nagata, K. Harada, N. Yamanobe and Y. Kawai are with the Vision and Manipulation Research Group, Intelligent Systems Research Institute, National Institute of Advanced Industrial Science and Technology (AIST), Central 2, Umezono 1-1-1, Tsukuba, Ibaraki 305-8568, Japan (a-nakamura@aist.go.jp, k-nagata@aist.go.jp, kensuke.harada@aist.go.jp, n-yamanobe@aist.go.jp, tokuo.tsuji@aist.go.jp, y.kawai@aist.go.jp)

T. Tsuji is with Faculty of Information Science and Electrical Engineering, Kyushu University, Fukuoka 819-0395, Japan (tsuji@ait.kyushu-u.ac.jp).

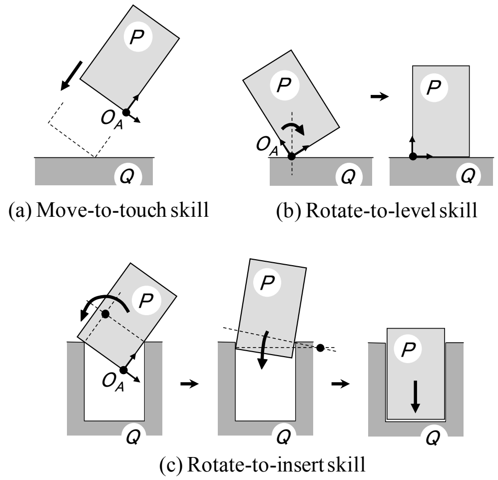


Fig. 1 Three fundamental skills

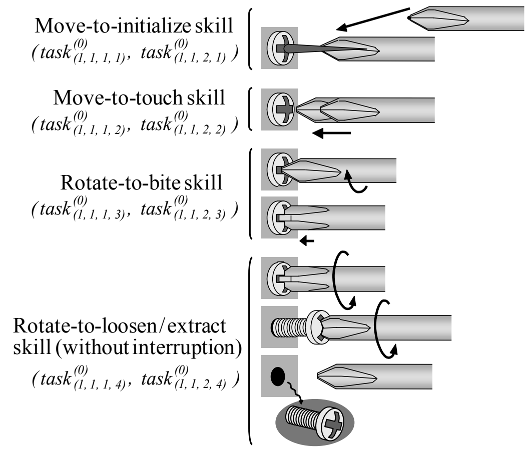


Fig. 2 Skill sequence of loosening a Phillips screw using a Phillips screwdriver

II. RELATED WORKS

Various approaches for error recovery have been reported [13]–[24]. Most of the research, however, targets trajectory and dynamics error compensation, and there are few studies considering task failure, which is a higher class of error to deal with in manipulation robots.

Donald performed a pioneer study on fine motion planning that considered error handling [13]. The possibility of failure in manipulation sequences is pursued in depth in this study. However, a fundamental scheme in which a recovery task is performed after having returned to a previous step was not investigated.

Baydar *et al.* have researched failure and recovery at task level [16]. Since they used Bayesian reasoning and Genetic algorithms for the recovery logic, their method is suitable for simple tasks such as repetitive work. However, the method is difficult to apply for complicated tasks consisting of various elements of manipulations. Furthermore, the method sorts errors at a higher level, for instance grasping failures and sensor failures, not at an essential and fundamental level such as causes of failure that have been taken into account in our method. Therefore, recovery by returning to the shortest previous step cannot be performed.

Recently, a method of manipulation that uses a skill library with failure detection has been proposed by Pastor *et al* [24]. However, replaceable recovery schemes after failures are not considered, since their research targets trajectory error compensation.

III. STRATIFICATION OF TASKS

Let us first explain our concept of skills and stratification of tasks. In this section, although the tasks of maintenance robots are mainly taken into account, the tasks of service robots operating in a human daily life environment may be considered similar. See References [2], [3], [7], [8] for more details.

A. Manipulation Skills

In assembly and disassembly tasks, the skills in which the contact states vary are particularly significant. In References [3], [5], we considered three skills, "move-to-touch," "rotate-to-level" and "rotate-to-insert," which all play an important part in such tasks.

(i) Move-to-touch Skill: This skill is defined as the transition of a grasped object P in a constant direction that continues until contact with another object Q occurs (Fig. 1(a)).

(ii) Rotate-to-level Skill: This skill is defined as the rotation around either a contact point or a contact edge to match the face of the grasped object P with the face of another object Q (Fig. 1(b)).

(iii) Rotate-to-insert Skill: This skill is the motion of rotating the object P obliquely into the hole in another object Q to insert it accurately (Fig. 1(c)).

A specific task is composed of sequences of skill primitives such as these move-to-touch, rotate-to-level and rotate-to-insert skills. The skill sequences can be decided by several methods. We have already presented a method that uses variations of the number of contact points in the skill primitives [3], [8].

Moreover, many skills can be defined based on slightly changed versions of these three fundamental skills. For example, the rotate-to-bite and rotate-to-loosen skills that are used in the task of loosening a screw while manipulating a screwdriver as shown in Fig. 2 are derived by performing small changes of these fundamental skills.

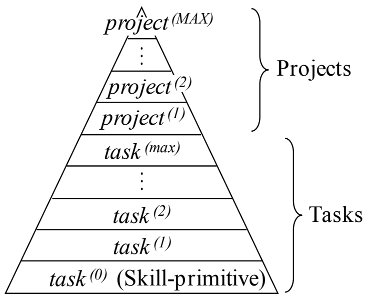


Fig. 3 Manipulation hierarchy

B. Hierarchy of Tasks

Manipulation tasks composed of several skills have been considered previously as described in Reference [9]. However, actual tasks composed of many skills are in fact more complex and a stratification of the tasks is preferable for efficient management and execution.

We have described the hierarchizing of manipulation tasks based on a bottom-up approach [10]. If we ignore the servo layer, the skill layer, which consists of elements such as the move-to-touch and rotate-to-bite skills, is located in the lowest layer called the $task^{(0)}$ layer. Each skill is performed using the processes of visual sensing, geometric modeling, planning and execution. One tier above the $task^{(0)}$ layer is called the $task^{(1)}$ layer. Similarly, the $task^{(i+1)}$ is composed of sequences of $task^{(i)}$ elements (Fig. 3). The top layer, where the error recovery loop is closed, is called $task^{(max)}$ and one tier above $task^{(max)}$ is called the project layer. The project layer is a high-ranking class that is not affected by the recovery loop. The project layer might also be hierarchized, but we will not discuss it here.

C. Stratification of Maintenance Tasks

Let us consider the typical tasks involved in the repair of a personal computer (Fig. 4(a)) as an example of stratification. This maintenance $project^{(1)}$ with component replacement for an electrical appliance is performed as shown in Fig. 4(b). The task sequence { $task^{(2)}_{(1,\ i_2)}$ }of the case opening $task^{(3)}_{(1)}$ is shown in Fig. 4(c). If there are two Phillips screws on Side(R), $task^{(2)}_{(1,\ 1)}$ is composed of two tasks of loosening each of the two Phillips screws using a Phillips screwdriver which can be described as $task^{(1)}_{(1,\ 1,\ 1)}$ and $task^{(1)}_{(1,\ 1,\ 2)}$, and those skill sequences { $task^{(0)}_{(1,1,1,\ i_0)}$ } and { $task^{(0)}_{(1,1,2,\ i_0)}$ } are shown in Fig. 2. These skill primitives are described in detail in Reference [9]. Moreover, these skill sequences will be called the *minimum traceable unit*, which means the smallest unit in which it is necessary to return to the first node of a skill primitive sequence if an error occurs. The loosening tasks at the Rear and Side(L) called $task^{(2)}_{(1,\ 3)}$ and $task^{(2)}_{(1,\ 5)}$, respectively, are similar to $task^{(2)}_{(1,\ 1)}$ at Side(R). After all the screws are extracted, the task of removing the case is performed as shown in Fig. 4(d). Then, the $task^{(1)}_{(1,\ 6,\ 1)}$ layer,

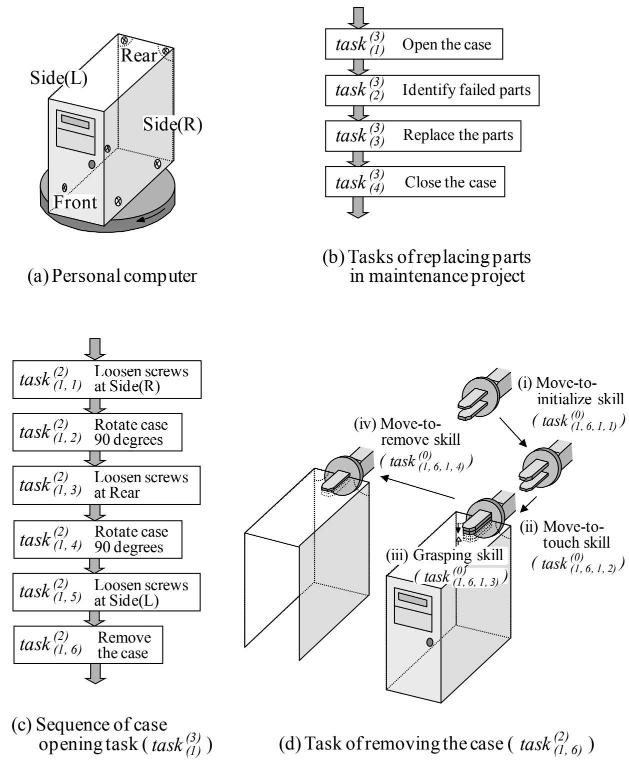


Fig. 4 Tasks of replacing parts of a personal computer

which has no meaning, adds one tier below $task^{(2)}_{(1,\ 6)}$ to make the number of layers the same. The tasks of rotating 90 degrees using a manipulator in $task^{(2)}_{(1,\ 2)}$ and $task^{(2)}_{(1,\ 4)}$ are performed by a procedure that refers to the task of removing the case.

IV. Error Recovery in Stratified Tasks

In an ideal environment, tasks are completed without any errors occurring. In actual manipulation tasks, however, errors often do occur from various causes. We will describe our concept of error classification and process flow with error recovery in the task hierarchy.

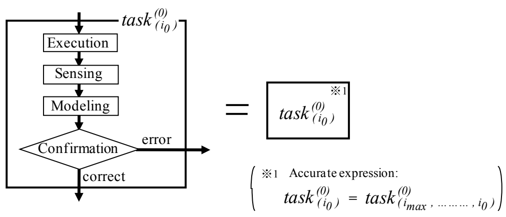
A. Classification of Errors

Failures can be caused by several kinds of errors such as control errors, modeling errors and visual sensing errors. We group the error states into several classes according to possible causes. The classes of errors are described in detail in References [9]-[10].

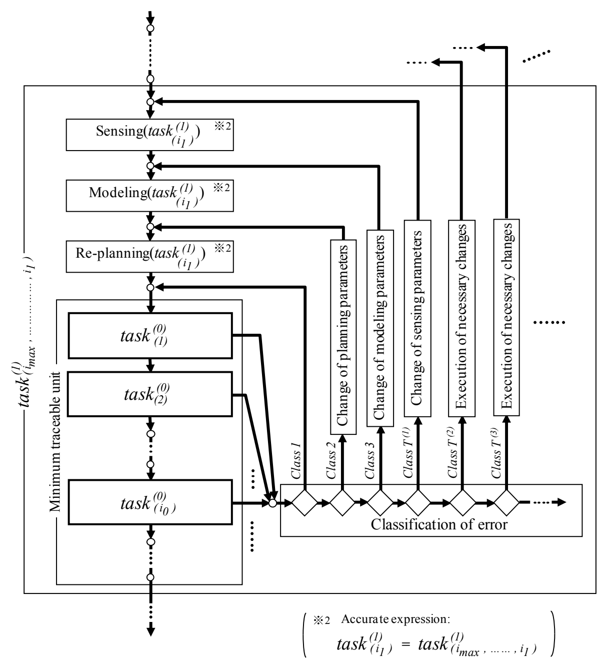
・Execution error: This is a mechanical error caused in the manipulator mechanism such as a gear backlash.

・Planning error: This is an error caused by inaccurate parameter values in planning.

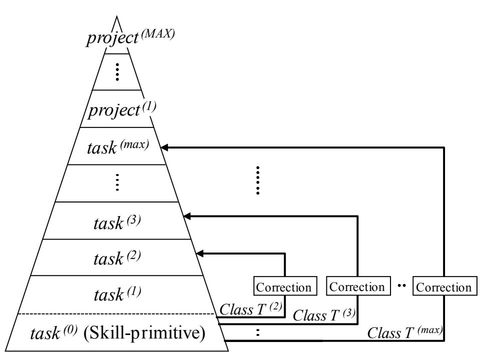
・Modeling error: This is an error caused by differences between the real object and the geometric model in the software.



(a) Sequence from execution to Confirmation step in each skill primitive



(b) Backward correction process to starting points of a minimum traceable unit



(c) Backward correction process to starting points of the upper task levels

Fig. 5 Process flow of error recovery

・Sensing error: This is an error occurring during visual sensing.

Merely remedying the causes of these errors does not always solve the problem. For instance, it may be necessary to return to a previous step when the working environment is greatly changed by the error.

B. Error Recovery based on Classification

We have considered the process of maintenance tasks in which a sequence of {execution, visual sensing, geometric modeling, confirmation} as shown in Fig. 5(a) is performed at each $task^{(0)}_{(i_0)}$ after visual sensing, geometric modeling and planning of a total task are performed [10]. Then, the visual sensing and geometric modeling that is performed for the confirmation at $task^{(0)}_{(i_0)}$ can be also used as the initial data for the next primitive $task^{(0)}_{(i_0+1)}$. Especially, References [7], [8] show simplified models which can be used in visual sensing and planning at each $task^{(0)}_{(i_0)}$ and are useful also for confirmation. The action of a task is aborted midway if an error is identified at the Confirmation step, without taking error recovery into account.

To be able to perform a given task to the end, we have proposed a process flow of stratified tasks that takes error recovery into account in Reference [10], and an outline is shown in Fig. 5(b) and (c). This process is performed based on recovery through *a backward correction process*. In $task^{(0)}_{(i_0)}$, sensing and modeling are performed to confirm the task scenario after the execution of the skill primitives is done (Fig. 5(a)). At the Confirmation step, the result is judged as correct or failed by an automatic process or by a human operator. Error recovery is performed using the following error classification.

Class 1: When it is judged to be an execution error, $task^{(1)}_{(i_1)}$ is executed again without correcting the parameters.

Class 2: When it is judged to be a planning error, $task^{(1)}_{(i_1)}$ is executed again with a change in the planning parameters.

Class 3: When it is judged to be a modeling error, $task^{(1)}_{(i_1)}$ is executed again with a change in the modeling parameters.

Class $T^{(1)}$: When it is judged to be a sensing error, $task^{(1)}_{(i_1)}$ is executed again with a change in the sensing parameters.

Class $T^{(2)}$: $task^{(2)}_{(i_2)}$ is executed again after the execution of the necessary changes and returns to the start at one tier above the layer $task^{(1)}_{(i_1)}$.

:

:

Class $T^{(max)}$: $task^{(max)}_{(i_{max})}$ is executed again after the execution of the necessary changes and returns to the start at (max - 1) tier above the layer $task^{(1)}_{(i_1)}$.

Class $T^{(max+1)}$: When it is judged that too many changes will be required, the process being executed is aborted.

For Class 1, Class 2, Class 3, or Class $T^{(1)}$ errors, the process flow must return to the indicated step before the *minimum traceable unit* as shown in Fig. 5(b). Similarly, for Class $T^{(2)}$, Class $T^{(3)}$, ⋯, or Class $T^{(max)}$ errors, the process flow must return to the starting point of the indicated upper task layer as shown in Fig. 5(c). See Reference [25] for the restoration technique in detail for each class of error.

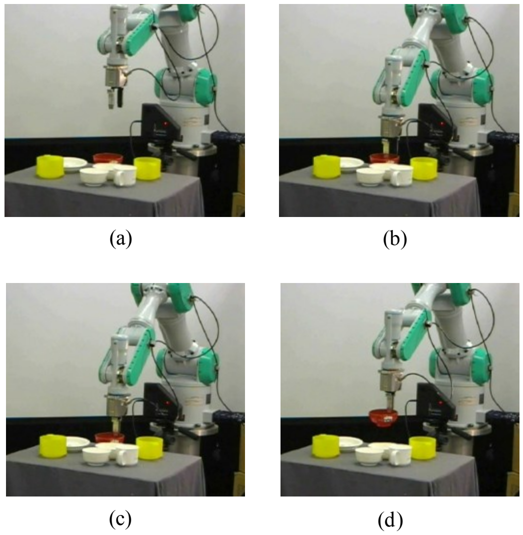


Fig. 6 Experimental scene using a manipulator arm to mock up a service robot

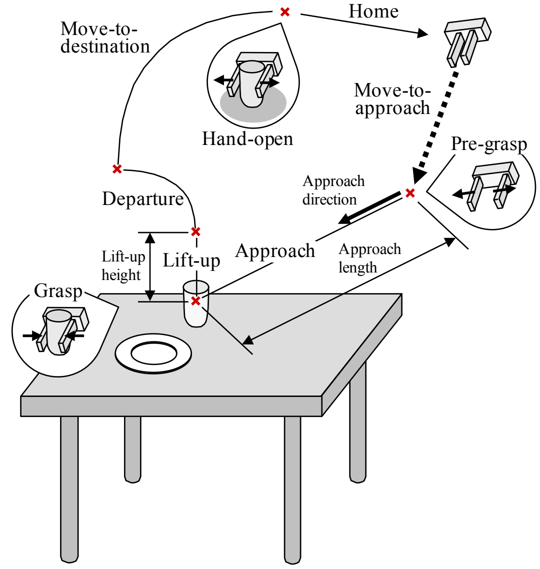


Fig. 7 Picking task using a gripper

V. Tasks of a Service Robot

To make our error recovery process more widely applicable besides maintenance, it is necessary to take recovery techniques for robots active in various other fields into account. Therefore, let us take up the task of a service robot operating in a human daily life environment. And we will choose a typical task of picking up an indicated object using the parallel jaw gripper on the robot [11], [12].

Figure 6 shows an experimental scene of picking up tableware such as cups, bowls and plates using our manipulator with a gripper. The processes of the task are illustrated in Fig. 7, and the flow of the skill sequence is shown in Fig. 8. These primitive motions are as follows,

($Skill_1$) Move-to-approach: This motion moves the robot hand to the starting point of the approach motion.

($Skill_2$) Pre-grasp: The robot hand opens to grasp the target object.

($Skill_3$) Approach: This motion is that of the robot hand moving to the grasping point at low speed.

($Skill_4$) Grasp: The robot hand grasps the target object.

($Skill_5$) Lift-up: This motion is that of the robot lifting up the grasped object.

($Skill_6$) Departure: This motion is that of the robot hand moving to the safety area.

($Skill_7$) Move-to-destination: This motion is that of the robot hand moving to the destination point.

($Skill_8$) Hand-Open: The gripper opens to put the object.

($Skill_9$) Home: The robot hand returns to the starting point for the next approach motion.

To correct the robot's motions at each step, a manual operation module for robot control has been inserted in the terminal processing of each primitive motion. For example, slight errors concerning the position and orientation of the object after transition and the condition of the grasped object can be corrected. This process is recovery through *a forward correction process* contrary to recovery through *the backward correction process* described in Section IV.

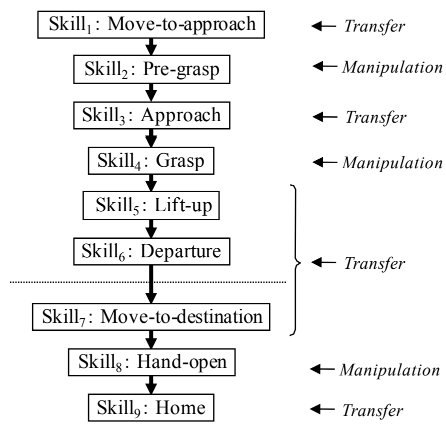
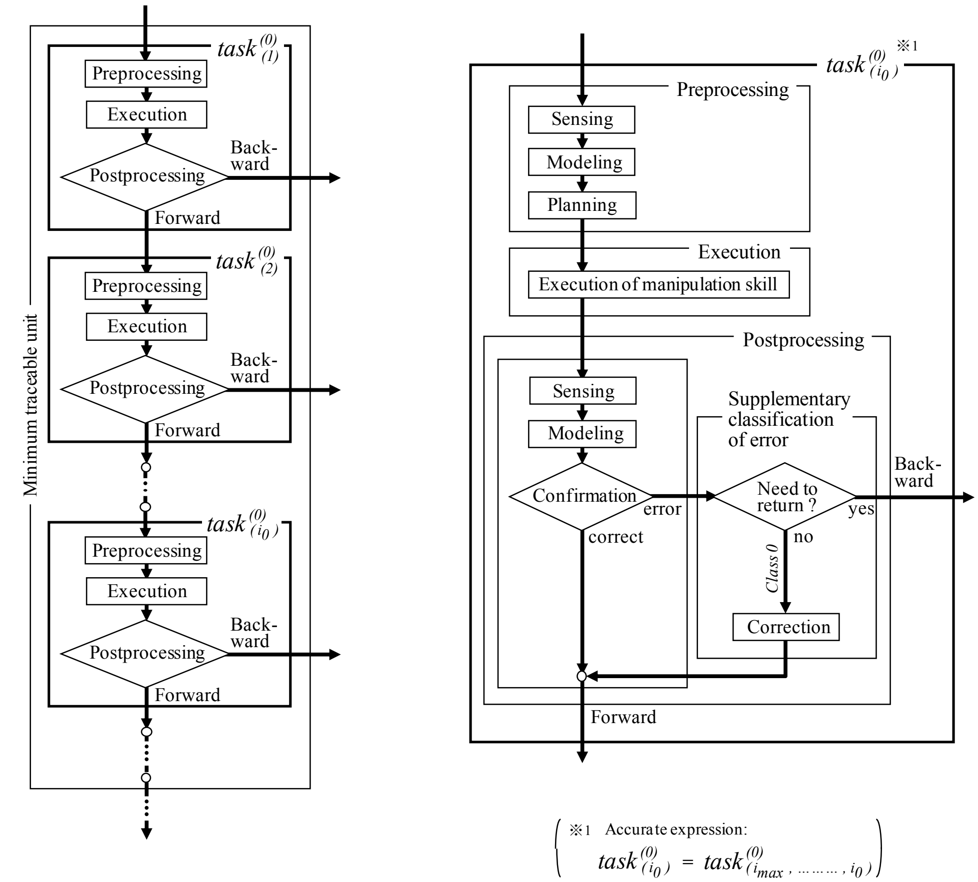


Fig. 8 Skill sequence of picking up a cup

Also, Figure 8 shows that there are two kinds of skill motions in the picking task that are considerably different, *Transfer* and *Manipulation*, and these can be considered to be the *minimum traceable unit*. Since transfer is a global motion and manipulation is a local motion of the hand in general, it is fairly necessary to perform sensing and modeling before and after the execution, to derive the initial data and to confirm succession, respectively. This point is different from the tasks of maintenance robots which behave in a local environment.



(a) Skill primitive sequence in a minimum traceable unit

(b) Improved sequence in each skill primitive

Fig. 9 Processes before and after execution in each skill primitive

VI. IMPROVED ERROR RECOVERY FOR ROBOTS IN VARIOUS FIELDS

In this section, we will revise our process flow by fusing the recovery technique of maintenance robots with that of service robots operating in a human daily life environment.

As necessary for both the preprocessing and the postprocessing in each $task^{(0)}_{(i_0)}$, the process flow in the *minimum traceable unit* is improved to the skill primitive sequence shown in Fig. 9(a). However, if kinds of the successive skill primitives are the same, the sensing and modeling in the preprocessing of the next primitive $task^{(0)}_{(i_0+1)}$ might be able to be used together for the sensing and modeling in the postprocessing of $task^{(0)}_{(i_0)}$.

A detailed flow chart of each $task^{(0)}_{(i_0)}$ is shown in Fig. 9(b). In the Preprocessing step, visual sensing, using a three-dimensional measuring instrument such as a laser range finder and stereo vision, and geometric modeling in computer software are performed. Planning for execution of the skill primitive is performed sequentially. This means the auxiliary processes in which the indeterminate parameters necessary for operation of a manipulator are determined.

After the Execution step in which the skill primitive is performed, the Postprocessing step that is important for error recovery is performed. First, sensing, modeling and confirmation are performed similarly to that described in Section IV. At the Confirmation step, the result is judged to be correct or failed by an automatic process or by a human operator. If the judgment is failure, the process proceeds to the Supplementary classification step in which the necessity of a manual operation described in Section V is determined. If the processing can advance to the next primitive $task^{(0)}_{(i_0+1)}$ by the manual operation, the correction is performed. Let us call this type of error Class 0 as shown in Fig. 9(b), according to the classification described in Section IV. On the other hand, if a minor correction is not judged to be effective, it is necessary to proceed to recovery through *a backward correction process*. The return route after leaving the Supplementary classification step from the backward direction is decided by passing the classification of error step in Fig. 10. The procedure is the same as the method shown in Section IV. Therefore, the improved process flow with error recovery can be expressed by Fig. 10 linking to Fig. 9.

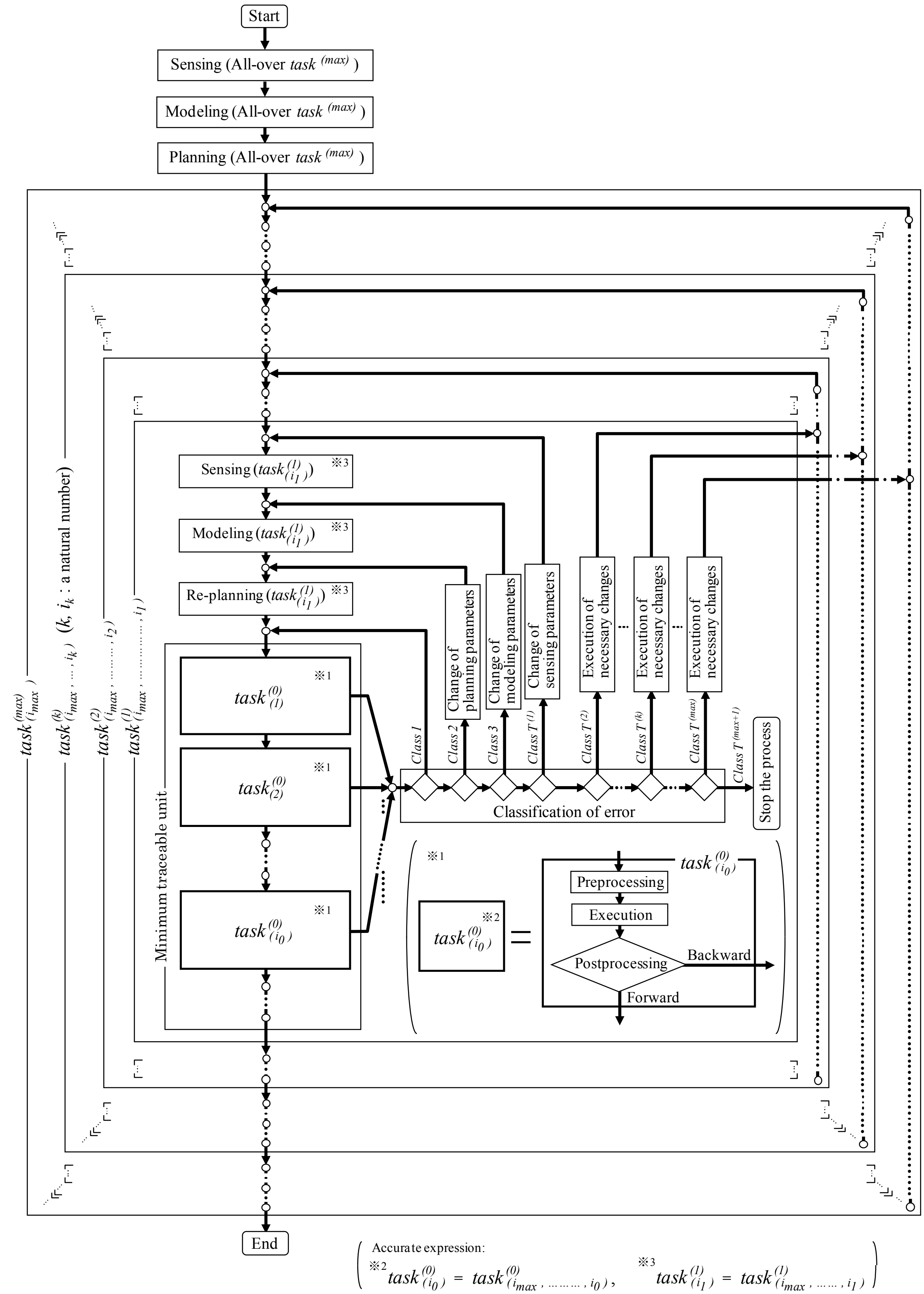


Fig. 10 Process flow with error recovery

We will describe the relative advantages of both forward and backward correction in the recovery processes. For recovery through *the forward correction process*, there is little wasted time and the operator can easily understand the overall image of the correction since the task sequence does not change. However, the correction of a major error is difficult. Furthermore, the possibility of the same error happening repeatedly in subsequent tasks is large, since essential corrections of the parameters of the robot system are not done. For recovery through *the backward correction process*, the possibility that controls can be done more accurately than the last time is large when the same skill primitive is executed again, since the appropriate correction to the proper values of the system parameters is done. The whole system model of the robot may become correct if the error recovery is performed repeatedly. However, it takes time for execution, and the system becomes large in both software and hardware.

VII. Conclusion

It is necessary to increase the reliability of robots active in various fields, and error recovery during complex tasks plays an important part. For this purpose, we have presented a processing flow for recovering from errors of various causes. The new method of error recovery that we have proposed uses the concepts of both task stratification and error classification. We have expressed it specifically using a flow chart. The range of the error recovery was extended by adopting both forward and backward correction processes of recovery.

An advantage of our method is that it takes into account classification of errors based on possible causes to facilitate the derivation of the recovery task. Recovery through the backward correction process can be derived by making necessary changes such as modifications of planning, modeling and sensing parameters. Even if the cause of an error cannot be decided, recovery may be possible by performing the correction sequentially from a lower class. There are few research studies that take into account a suitable restoration method using replaced tasks after a failure occurs. Furthermore, it becomes easy to consider the step to which the task should return by taking into account stratification of tasks.

In the future, we will further study optimum adjustment methods for the error recovery parameters in the backward correction process and a fully automatic method for confirming the achievement of various tasks. We will attempt to apply our method to real robot systems.

References

[1] T. Hasegawa, T. Suehiro and K. Takase, "A robot system for unstructured environments based on an environment model and manipulation skills," in Proc. IEEE Int. Conf. Robot. Autom., Sacramento, USA, 1991, pp. 916-923.

[2] T. Hasegawa, T. Suehiro and K. Takase, "A model-based manipulation system with skill-based execution," IEEE Trans. Robot. Autom., vol. 8, no. 5, pp. 535-544, 1992.

[3] A. Nakamura, T. Ogasawara, T. Suehiro and H. Tsukune, "Skill-based backprojection for fine motion planning," in Proc. IEEE/RSJ Int. Conf. on Intell. Robots Syst., Osaka, Japan, 1996, vol. 2, pp. 526-533.

[4] A. Nakamura, T. Ogasawara, T. Suehiro and H. Tsukune, "Fine motion strategy for skill-based manipulation," Artificial Life and Robotics, Springer, vol. 1, no. 3, pp. 147-150, 1997.

[5] A. Nakamura, T. Ogasawara, T. Suehiro and H. Tsukune, "Fine motion strategy in three-dimensional space using skill-based backprojection," Artificial Life and Robotics, Springer, vol. 2, no. 3, pp. 134-137, 1998.

[6] A. Nakamura, T. Ogasawara, H. Tsukune and M. Oshima, "Surface-based geometric modeling using task-oriented teaching trees," in Proc. IEEE/RSJ Int. Conf. on Intell. Robots Syst., Osaka, Japan, 1996, vol. 3, pp. 1015-1022.

[7] A. Nakamura, T. Ogasawara, K. Kitagaki and T. Suehiro, "Using robust and simplified geometric models in skill-based manipulation," in Proc. IEEE/RSJ Int. Conf. on Intell. Robots Syst., Hawaii, USA, 2001, vol. 1, pp. 138-145.

[8] A. Nakamura, K. Kitagaki and T. Suehiro, "Using simplified geometric models in skill-based manipulation," Advanced Robotics, vol. 18, no. 8, pp. 835-858, 2004.

[9] A. Nakamura and K. Kitagaki, "Skill-based manipulation and error recovery in maintenance tasks," in Proc. Int. Symp. on Artificial Life and Robotics, Oita, Japan, Jan., 2007, pp. 179-182.

[10] A. Nakamura and T. Kotoku, "Systematization of error recovery in skill-based manipulation," Artificial Life and Robotics, Springer, vol. 14, no. 2, pp. 203-208, 2009.

[11] K. Nagata, Y. Wakita and E. Ono, "Task instruction by putting task information in work space," in Proc. IEEE Int. Conf. Robot. Autom., Roma, Italy, Apr. 2007, pp. 305-310.

[12] K. Nagata, T. Miyasaka, D. N. Nenchev, N. Yamanobe, K. Maruyama, S. Kawabata and Y. Kawai, "Picking up an indicated object in a complex environment," in Proc. IEEE/RSJ Int. Conf. on Intell. Robots Syst., Taipei, Taiwan, Oct. 2010, pp. 2109-2116.

[13] B. R. Donald, "Planning multi-step error detection and recovery strategies," Int. J. Robot. Res., vol. 9, no. 1, pp. 3-60, 1990.

[14] E. Z. Evans and C. S. George Lee, "Automatic generation of error recovery knowledge through learned reactivity," in Proc. IEEE Int. Conf. Robot. Autom., 1994, pp. 2915-2920.

[15] L. Seabra Lopes, L. M. Camarinha-Matos, "A machine learning approach to error detection and recovery in assembly," in Proc. IEEE/RSJ Int. Conf. on Intell. Robots Syst., Pennsylvania, USA, Aug. 1995, vol. 3, pp. 197-203.

[16] C. M. Baydar and K. Saitou, "Off-Line error prediction, diagnosis and error recovery using virtual assembly systems," in Proc. IEEE Int. Conf. Robot. Autom., Seoul, Korea, May 2001, pp. 818-823.

[17] T. Fukuda, M. Nakaoka, T. Ueyama and Y. Hasegawa, "Direct teaching and error recovery method for assembly task based on a transition process of a constraint condition," in Proc. IEEE Int. Conf. Robot. Autom., Seoul, Korea, May 2001, pp. 1518-1523.

[18] C. W. Moon and B. H. Lee, "A model-based error recovery scheme for a multi-robot system," Robotica, Cambridge Univ. Press, vol. 19, Issue 4, pp. 371-380, July 2001.

[19] Y. Yamada, T. Morizono, Y. Umetani and T. Yamamoto, "Human error recovery for a human/robot parts conveyance system," in Proc. IEEE Int. Conf. Robot. Autom., Washington, DC, USA, May 2002, pp. 2004-2009.

[20] K. Yamazaki, M. Tomono, T. Tsubouchi and S. Yuta, "Motion planning for a mobile manipulator based on joint motions for error recovery," in Proc. IEEE/RSJ Int. Conf. on Intell. Robots Syst., Beijing, China, Oct. 2006, pp. 7-12.

[21] M. Scheutz and J. Kramer, "Reflection and reasoning mechanisms for failure detection and recovery in a distributed robotic architecture for complex robots," in Proc. IEEE Int. Conf. Robot. Autom., Roma, Italy, Apr. 2007, pp. 3699-3704.

[22] J. C. Himmelstein, E. Ferre and J.-P. Laumond, "'Teleportation'-Based Motion Planner for Design Error Analysis," in Proc. IEEE Int. Conf. Robot. Autom., Kobe, Japan, May 2009, pp. 914-920.

[23] H. Dobashi, A. Noda, Y. Yokokohji, H. Nagano, T. Nagatani and H. Okuda, "Derivation of Optimal Robust Grasping Strategy under Initial Object Pose Errors," in Proc. IEEE/RSJ Int. Conf. on Intell. Robots Syst., Taipei, Taiwan, Oct. 2010, pp. 2096-2102.

[24] P. Pastor, M. Kalakrishnan, S. Chitta, E. Theodorou and S. Schaal, "Skill Learning and Task Outcome Prediction for Manipulation," in Proc. IEEE Int. Conf. Robot. Autom., Shanghai, China, May 2011, pp. 3828-3834.

[25] A. Nakamura and Y. Kawai, "Recovery technique from classified errors in skill-based manipulation," in Proc. Int. Symp. on Artificial Life and Robotics, Oita, Japan, Feb. 2010, pp. 1010-1013.