Proc. 2007 IEEE Int. Conf. on
Robotics and Biomimetics, pp. 640-645,
Sanya, China, Dec. 2007.

Co-Development of Task Models through Robot-Human Interaction

Jun Miura* and Yoshinori Nishimura
Department of Mechanical Engineering, Osaka University

***Abstract*— Recently there is an increasing demand for service robots which help humans in home and office environments. Such a robot has to deal with a much wider range of tasks and environments than usual industrial ones, and it is very difficult for a manufacturer to give the robot all pieces of necessary knowledge in advance. This paper, therefore, proposes an interactive teaching method using *task models*. A task model describes what pieces of knowledge are necessary to achieve a task. The robot examines the task model, determines missing pieces of knowledge, and actively asks the instructor to teach them. This makes the teaching easier because the instructor does not have to think what and when to teach by himself. A task model, completed through interaction, is used for the robot to realize the corresponding task. We have developed a prototype of the task model-based teaching system and successfully applied it to teaching a service robot several tasks.**

I. Introduction

Personal service robots are one of the promising areas to which the robotics technologies can be applied [1], [3], [5], [13]. As we are facing the “aging society”, the need for robots which help human in various everyday situations is increasing. Possible tasks of such robots are: bringing a specified object to the user in the bed, cleaning a room, mobile aid, and so on.

Usually such service robots have to deal with a much wider range of tasks than existing industrial robots. Since it is almost impossible to give the robot complete knowledge of tasks in advance, on-site robot teaching will be very important.

Approaches to robot teaching can roughly be divided into two categories. One is the *direct method* in which the instructor teaches a robot each motion or each state to achieve step by step using, for example, teaching pendants, special pointing devices [7], [14], or a sequence of images [12]. This approach is simple to implement but requires much instructor’s efforts.

The other approach is the *indirect method* in which an instructor teaches “what the task is” to a robot. *Teaching by demonstration* methods are typical examples [4], [8], [9]. These works focus more on the recognition of human demonstration using vision. Although using a virtual world [2] may make it easier to recognize the demonstration, conversion of the recognition results to a feasible robot program still requires the robot to possess as much inference ability as a fully autonomous robot does.

To summarize the above discussion, direct methods are easily implemented on robots but require much instructor’s effort, while indirect methods are intuitive for the instructor but require a high-level ability of the robot. We are, therefore, developing a novel interactive teaching method which exists in between these approaches.

In interactive teaching, it is important for an instructor to determine when and what to teach to the robot. This is, however, difficult for the instructor due to two reasons. One is that the instructor does not usually have a concrete description (like a program) of everyday tasks, most of which he/she performs sometimes unconsciously. So it is tedious or hard for the instructor to make an explicit to-do list of the task. The other reason is that the instructor may not have a thorough understanding of the robot’s ability; in other words, he/she may not know exactly what the robot can (or cannot) do.

In this paper, we propose a practical approach to dealing with the above difficulties. A key idea is the use of *task model* which describes what pieces of knowledge are necessary in performing the task. It also describes the dependency relationships between the pieces of knowledge, represented as the network of them. Fig. 1 illustrates the interaction process guided by a task model. The robot examines the *slots* in the model to classify them into three categories: *known*, *unknown*, and *dependent on unknown*, and generates queries for filling unknown slots. This examine-query-fill-in process leads the interaction with the instructor, and continues until the robot acquires the necessary and sufficient knowledge of the task.

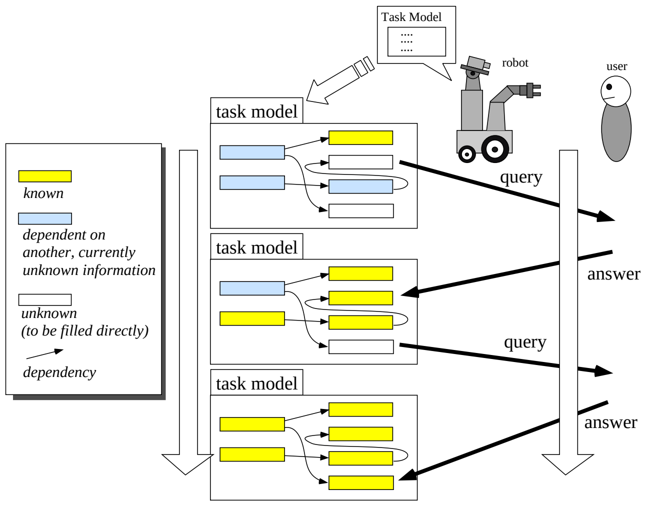


Fig. 1. Interaction process guided by task knowledge.

*Currently with Toyohashi University of Technology, Department of Information and Computer Sciences, Toyohashi, Aichi 441-8580, Japan. Email: jun@ics.tut.ac.jp

We have applied the idea of task model-based interactive teaching to a vision-based mobile robot [10]. We succeeded in teaching the robot a task of moving to another floor using an elevator. The task model was, however, constructed non-interactively in advance and was not modifiable. This paper thus develop a more general framework which supports both task model generation and modification through interaction. We also develop GUIs for support such interactions. We then test the framework in more complex service tasks for validating its applicability.

The instructor starts teaching by generating a task model using the GUI. The robot then collaborates with him/her in filling missing pieces of task knowledge to complete the task model. We call this process *co-development of task model*.

II. Task model

Task models describe various pieces of task knowledge such as attributes of objects, methods of finding object, and robot trajectories, and their dependencies. Task models consists of three submodels: *object model*, *robot model*, and *action model*. Once a complete task model is constructed, the robot can achieve the corresponding task by referring to the model.

A. Object model

An object model describes the attributes of the corresponding object. The attributes described include: object shape, object color, object position, positions of its parts, methods of finding the object, and methods of obtaining shape and color. Fig. 2 shows the object model for the leftmost object (this is actually a box with one single-swing door) shown in the Fig. 3. Other attributes can be added during teaching, if necessary. We also prepare object models as *template models* for typical objects like the ones shown in Fig. 3.

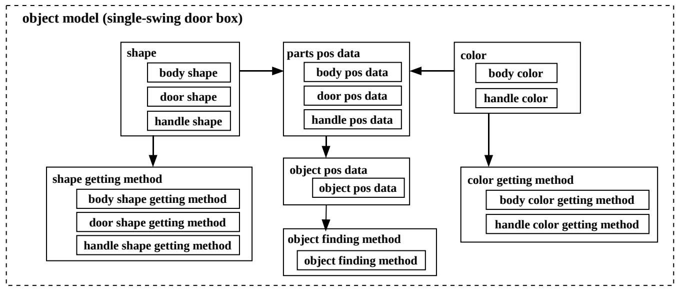


Fig. 2. The object model of *one single-swing door box*.

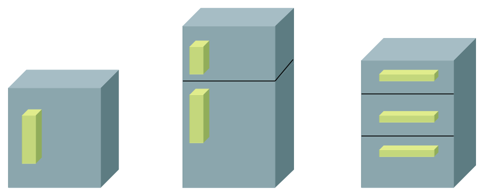


Fig. 3. Template models for objects with: *one single-swing door*, *two single-swing doors*, and *three drawers*, from left to right.

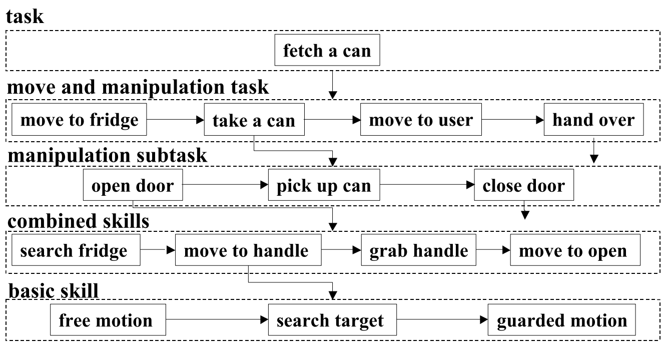


Fig. 4. A hierarchical structure of fetch-a-can task.

B. Robot model

A robot model describes the structure and the functions of the corresponding robot. The descriptions include: shape and positions of parts, kinematics, and sensing capability. The model also has basic motion planning and control skills.

C. Action model

1) Structure of action model: An action model describes the motion and sensing steps for achieving the corresponding task in a hierarchical way. Fig. 4 shows an example action model for the task of fetching a can from a distant refrigerator. This hierarchical decomposition is designed so that layers correspond to various notions of *actions*.

We suppose that the robot moves between several operation positions where it manipulates objects, will small adjustment movements around the positions, if necessary. The first layer (*move and manipulation task*) describes such alternate execution of movement and manipulation. The second layer (*manipulation subtask*) describes the steps of manipulation tasks. These layers correspond to our ordinary, object-centered notion of actions. *Take a can* task in the first is, for example, decomposed into three manipulation subtasks: *open door*, *pick up can*, and *close door*.

The third layer describes the decomposition of manipulation subtasks using *combined skills*, which correspond to the robot-centered notion of actions. Each combined skill is then composed of *basic skills*, which are general, reusable actions similar to *skills* in *skill-based manipulation* research [6].

2) Basic skills: Each basic skill describes *actual* motion and/or sensing steps to execute. The basic skills have several attributes to be assigned through interactive teaching.

We currently prepare the following seven basic skills:

- *search target*: for detecting a target object. Requires a detection method, object name / type, and sensor to use.
- *search target to grasp*: for detecting a target object which the robot hand grasps. Requires a detection method, object name / type, and sensor to use.
- *control hand*: for moving the hand. Requires a control method and object name / type.
- *control mobile base*: for moving the mobile base. Requires a control method and sensors to use.

- *free motion*: for performing a free motion. Requires a motion planner and a motion controller.
- *guarded motion*: for performing a guarded motion near objects. Requires a control method and sensors to use for collision detection.
- *compliant motion*: for performing a compliant motion. Requires a control method and sensors to use for verifying contacts.

These basic skills are composed of more fine-grained *basic functions* such as the one for obtaining sensor data or moving the hand to some direction. New basic skills can be interactively constructed using the basic functions.

D. Implementation issues

We developed a C++ library for representing and traversing tree structures of task models, as shown in Figs. 2 and 4.

III. Task model-based interactive teaching

In the task model-based interactive teaching framework, as described above, the robot determines missing pieces of knowledge (i.e., *unknown* slots to be filled) in the task model, and generates queries for them; that is, each piece of information is requested when it is necessary. This examine-query-fill-in process leads the interaction with the instructor, and continues until the robot has the necessary and sufficient knowledge of the task (see Fig. 1).

The interactive teaching usually takes the following steps:

- Teach object information using object models.
- Generate action models.
- Refine action models with action execution by the robot.

The order of these steps is, however, more flexible. If an object to manipulate is at a distant location, for example, the robot has to go there to complete the object model (because the object is not observable from the current position). In that case, an object model with a part of its slots (e.g., its location) being filled is first constructed. Since the other details of the object is not necessary for the robot to go there, it can execute “move to the object” action step. When the robot manipulates the object, it knows that the details are necessary and asks the instructor to teach them.

The following subsections explains the three steps in detail.

A. Object model-based teaching

The descriptions in object models are divided into two categories: object attributes and methods of acquiring them. Acquisition methods are given by the instructor; object attributes are either given by the instructor or filled using the corresponding acquisition methods. We develop a GUI for interactively teaching object information as shown in Fig. 9.

We here explain the details of interactions when the instructor teaches, upon request from the robot, the width of a refrigerator to the robot. Fig. 5 shows the relationships between the related pieces of knowledge. Fig. 6 shows the outline of interactions between the robot and the instructor.

The actual steps of interactive teaching are as follows. The robot examines the dependency (see Fig. 5) and finds that

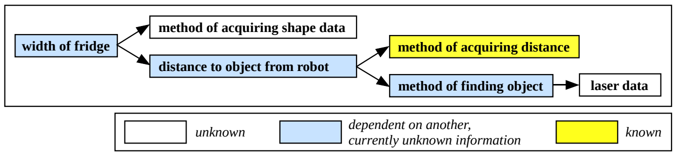

width of fridge
method of acquiring shape data
distance to object from robot
method of acquiring distance
method of finding object
laser data
unknown
dependent on another, currently unknown information
known

Fig. 5. Dependency for “width of fridge”.

1 (*robot*) proposes methods of acquiring shape data.
2 (*instructor*) selects one of the proposed ones (see Fig. 7).
3 (*robot*) proposes methods of finding object.
4 (*instructor*) selects one of the proposed ones. Here the instructor selects the method of pointing the LRF (laser range finder) data.
5 (*robot*) acquires the LRF data and asks the instructor to point the corner of the fridge in the data (Fig. 9).
6 (*instructor*) points the corner.
7 (*robot*) calculate the position and the width of the fridge and asks the instructor to verify them.
8 (*instructor*) verifies it.

Fig. 6. Outline of interaction for teaching the fridge width.

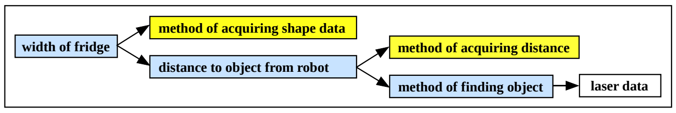

width of fridge
method of acquiring shape data
distance to object from robot
method of acquiring distance
method of finding object
laser data

Fig. 7. State after the instructor teaches a method of acquiring shape data.

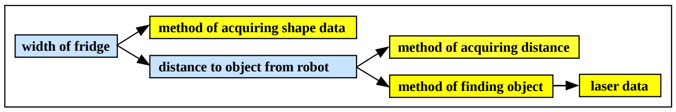

width of fridge
method of acquiring shape data
distance to object from robot
method of acquiring distance
method of finding object
laser data

Fig. 8. State after the instructor teaches a method of finding the fridge.

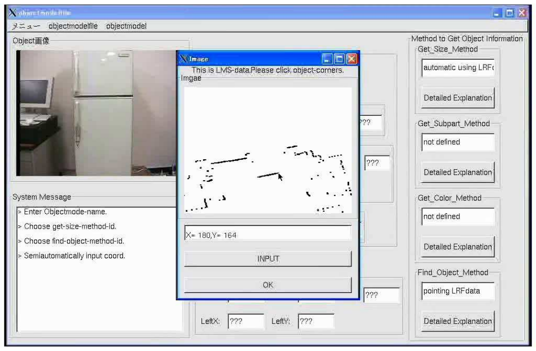
the “method of acquiring shape data” is missing; so it asks the instructor to teach it. Here the robot proposes two options: automatic using LRF data and manual input, and the instructor selects the first one. This interaction results in the state that the “method of acquiring shape data” is known (see Fig. 7).

The robot next identifies that the LRF data is necessary and it autonomously acquires the data and this data is turned to be *known*. The robot then comes to know that the “method of finding object” is necessary and asks the instructor. It proposes three options: automatic from LRF data, instructor’s pointing on the LRF data, and manual, and the instructor selects the second one. This results in the state shown in Fig. 8.

Examining the state shown in Fig. 8, the robot knows that the “distance to object from robot” is necessary and it can be calculated by finding the fridge. So the robot takes the LRF data and asks the instructor to point the fridge location in the data (see Fig. 9). This results in the state that all necessary information is known.

B. Action model-based teaching

The process of teaching actions (or plans) has two phases. The first phase is the action model generation. This corre-



This is LMS-data.Please click object-corners.

X= 180,Y= 164

System Message
> Enter Objectmode-name.
> Choose get-size-method-id.
> Choose find-object-method-id.
> Semiautomatically input coord.

automatic using LRF
not defined
not defined
pointing LRFdata

Fig. 9. GUI for pointing fridge location. The pop-up window in the center shows the LRF data including the fridge.

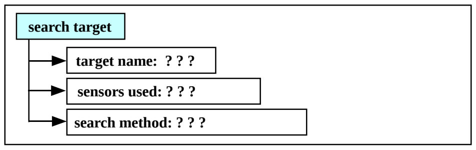
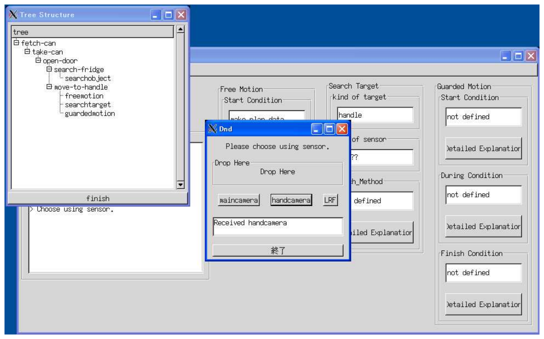


Fig. 10. Information dependency for “search target”.

1 (*robot*) asks the name of the target to search for.
2 (*instructor*) inputs the name. Here the instructor enters “handle.”
3 (*robot*) proposes several sensors to use (Fig. 12).
4 (*instructor*) selects one of the proposed ones. Here the instructor selects the hand camera (camera on the hand).
5 (*robot*) proposes several methods for search.
6 (*instructor*) selects one of the proposed ones. Here the instructor selects a template matching-based one.
7 (*robot*) displays the search result and asks the instructor to verify them.
8 (*instructor*) verifies it.

Fig. 11. Outline of interaction for teaching “search target.”



Please choose using sensor.

Choose using sensor.

Received handcamera

handle
not defined

Fig. 12. GUI for teaching “sensors used”.

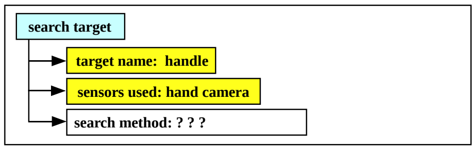


Fig. 13. State after interaction shown in Fig. 12.

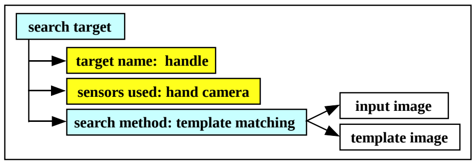


Fig. 14. State after information for “template matching” is given.

sponds to the teaching of what we can usually teach without knowing the actual situation; many *slots* in the task model are left unknown and some of assigned steps may be inappropriate. The second phase is the refinement of the action model with actual data and parameters.

1) Action model generation before execution: Action model generation is the process of generating the hierarchy of actions from top-level to the basic skill level (see Fig. 4). Note that this phase is also lead by the examine-query-fill-in loop (see Fig. 1). We develop a GUI using a tree-structure editor for action model generation as shown in Fig. 12.

We here describe a part of action model generation for *fetch-a-can* task shown in Fig. 4, namely, the teaching steps for basic skill “search target” used in combined skill “move to handle.” This basic skill needs the three attributes shown in Fig. 10. Fig. 11 shows the outline of the interactions between the robot and the instructor.

The actual steps of interactive teaching are as follows. The robot examines the dependencies (see Fig. 10) and finds that the name of the target is missing; so it asks the instructor to teach the name. The robot then asks for the sensor to use by proposing three options (camera on the body (or *main camera*), camera on the hand (or *hand camera*), and laser range finder) and the instructor selects the second one (see Fig. 12). The state of the model after this selection is shown in Fig. 13.

Next the robot asks for the “search method.” Search methods depend on the sensors used. Since the instructor selected the hand camera, the robot proposes four basic functions to be used as the search method: “get gray image,” “get edge image,” “line detection by Hough transform,” and “template matching.” The instructor selects the “template matching.”

Since “template matching” requires further information about “input image” and “template image”, two new dependency relationships are attached as shown in Fig. 14; these cannot be given in advance, they will be filled in the action model refinement phase.

2) Action model refinement through execution: Some knowledge should be taught *on-site* because it cannot be obtained in advance. Here we explain the process of on-site teaching (i.e., action model refinement), using as an example the teaching for skills “free motion” and “search target” included in combined skill “move to handle” (see Fig. 15).

From Fig. 15, the robot knows that the “target position” is necessary. Let us suppose that the method of acquiring this information is known, in which the instructor points the

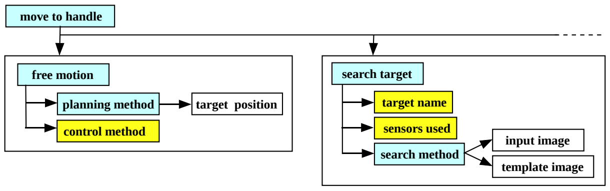


Fig. 15. Information dependency for “free motion” and “search target”.

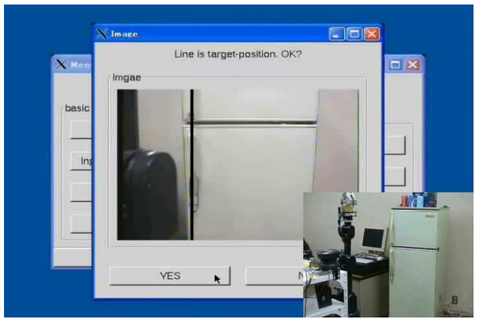


Fig. 16. GUI for teaching “target position”.

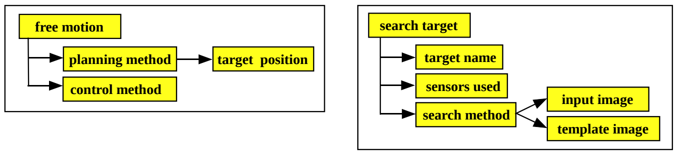


Fig. 17. Final state of “free motion” and “search target”.

location of the target position in the image of the main camera. So the robot acquires an image and asks the instructor to point the location (see Fig. 16).

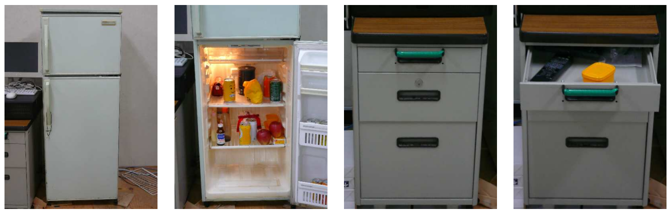
Then the robot asks for the “template image” is necessary. Let us suppose again that the method of acquiring this information is known, in which the instructor points the location of the template image in the image of the hand camera. So the robot acquires the image and asks the instructor to point the location. The robot finally acquires the refined action model shown in Fig. 17.

3) Teaching Base Motion: The instructor sometimes moves the robot to teach the motion between operation sites. We develop a GUI, by which the instructor can see images from the robot’s camera and issue move commands. Once the robot knows the destination of the movement, it automatically moves with iteratively planning the trajectory [11].

4) Verification and Modification of Action Models: The constructed and refined action model in a hierarchical structure can be viewed using the GUI. The GUI uses a tree-structure editor and the instructor can see the contents of arbitrary parts of the tree structure and can modify them if necessary; when one part is selected, the corresponding GUI is launched.

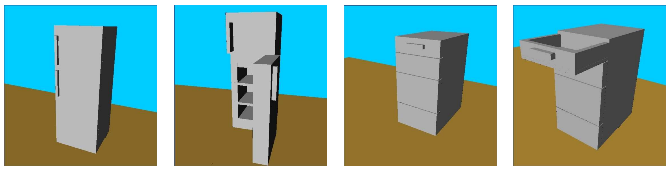
IV. Experimental results

We applied our interactive teaching system to two kinds of tasks. One task is to take a can from a refrigerator and the other is to take a box from a drawer. Fig. 18 shows the refrigerator and the drawer used.



(a) refrigerator. (b) drawer.

Fig. 18. Target objects for task.



(a) refrigerator. (b) drawer.

Fig. 19. Results of object model teaching.

A. Teaching Results

We taught the robot the tasks model for each task. Fig. 19 shows the object model of the fridge as the result of object model teaching; in Fig. 19, (a) and (b) show the model of the fridge and the drawer, respectively. Figs. 20 and 21 show the action model taught for the fridge and the drawer task, respectively. Shaded boxes in the figures indicate “basic skills” which are given to the system in advance; the other parts of the model are co-developed by the instructor and the robot. Since the overall structure of the models are similar, one can easily modify one model to generate the other. The time for teaching one task from scratch was about 10 minutes by a skilled instructor.

B. Execution Results

The robot successfully performed the tasks using the model taught by the interactive teaching. Figs. 22 and 23 show the snapshots of performing a part of the fridge task (*take a can*) and that of the drawer task (*take a box*), respectively.

V. Conclusions

This paper proposes a novel teaching framework, *task model-based interactive teaching* for co-development of task models. The task model describes what knowledge is necessary for each task and how pieces of knowledge are dependent on each other. By examining the state of the task model, the robot can determine missing pieces of knowledge and actively ask the instructor to teach them. This frees the instructor from thinking what and when to teach by himself/herself, thus decreasing the burden of the instructor.

So far, we have successfully applied the proposed teaching framework to two kinds of tasks: taking a can from a refrigerator and taking a box from a drawer. A future work is to test the framework for more various tasks which commonly appear in

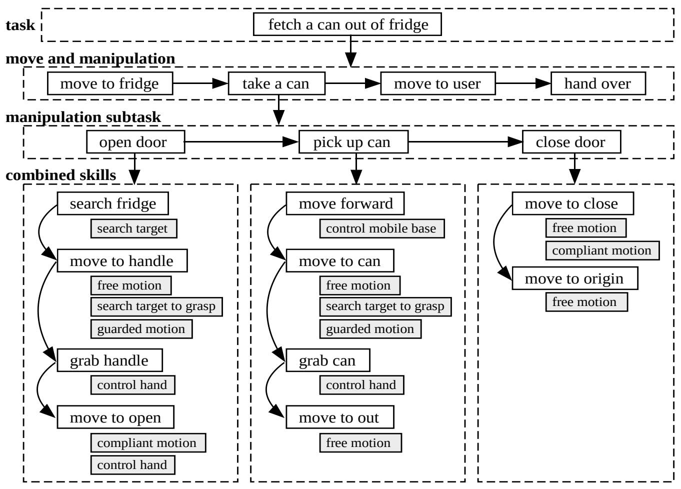


Fig. 20. The co-developed task model for the fridge task.

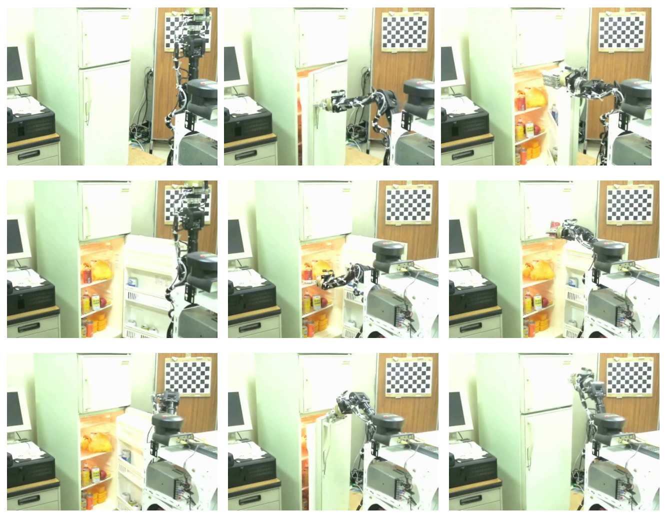


Fig. 22. Execution of "take a can" in Fig. 20.

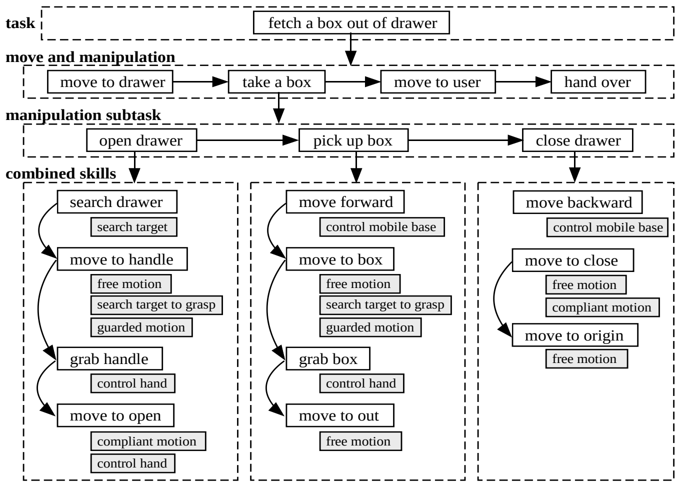


Fig. 21. The co-developed task model for the drawer task.

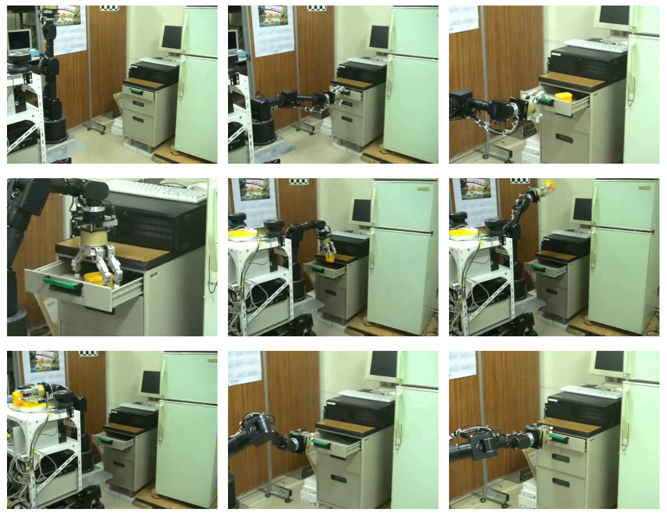


Fig. 23. Execution of "take a box" in Fig. 21.

our daily life. Such tasks may require more elaborated sensing and manipulation skills for execution; development of such a skill repertoire with an easy-to-use skill browser is another future work.

REFERENCES

[1] MORPHA project, *http://www.morpha.de/*.

[2] J. Aleotti, S. Caselli, and M. Reggiani. Leveraging on a Virtual Environment for Robot Programming by Demonstration. *Robotics and Autonomous Systems*, Vol. 47, No. 2/3, pp. 153–161, 2004.

[3] R. Bischoff. HERMES – A Humanoid Mobile Manipulator for Service Tasks. In *Proceedings of 1997 Int. Conf. on Field and Service Robots*, pp. 508–515, 1997.

[4] M. Ehrenmann, O. Rogalla, R. Zöllner, and R. Dillmann. Teaching Service Robots Complex Tasks: Programming by Demonstration for Workshop and Household Environments. In *Proceedings of 2001 Int. Conf on Field and Service Robots*, pp. 397–402, 2001.

[5] B. Graf, M. Hans, and R.D. Schraft. Mobile Robot Assistants. *IEEE Robotics and Automation Magazine*, Vol. 11, No. 2, pp. 67–77, 2004.

[6] T. Hasegawa, T. Suehiro, and K. Takase. A Model-Based Manipulation System with Skill-Based Execution. *IEEE Trans. on Robotics and Automation*, Vol. 8, No. 5, pp. 535–544, 1992.

[7] G. Hirzinger. Intuitive Motion Control – The SPACE MOUSE Story –. *J. of Robotics Society of Japan*, Vol. 17, No. 2, pp. 175–179, 1999.

[8] K. Ikeuchi and T. Suehiro. Toward an Assembly Plan from Observation Part I: Task Recognition With Polyhedral Objects. *IEEE Trans. on Robotics and Automat.*, Vol. 10, No. 3, pp. 368–385, 1994.

[9] Y. Kuniyoshi, M. Inaba, and H. Inoue. Learning by Watching: Extracting Resuable Task Knowledge from Visual Observation of Human Performance. *IEEE Trans. on Robotics and Automat.*, Vol. 10, No. 6, pp. 799–822, 1994.

[10] J. Miura, K. Iwase, and Y. Shirai. Interactive Teaching of a Mobile Robot. In *Proceedings of 2005 IEEE Int. Conf. on Robotics and Automation*, pp. 3389–3394, 2005.

[11] J. Miura, Y. Negishi, and Y. Shirai. Adaptive Robot Speed Control by Considering Map and Motion Uncertainty. *Robotics and Autonomous Systems*, Vol. 54, No. 2, pp. 110–117, 2006.

[12] Y. Nakamura and K. Arakawa. Teaching via Multimodal Communication. *J. of Robotics Society of Japan*, Vol. 17, No. 2, pp. 166–169, 1999. (in Japanese).

[13] N. Roy, G. Baltus, D. Fox, F. Gemperle, J. Goetz, T. Hirsch, D. Magaritis, M. Montemerlo, J. Pineau, J. Schulte, and S. Thrun. Towards Personal Service Robots for the Elderly. In *Workshop on Interactive Robots and Entertainment (WIRE 2000)*, 2000.

[14] F. Saito and T. Suehiro. Toward Telemanipulation via 2-D Interface – Concept and First Result of Titi. In *Proceedings of IECON 02*, pp. 2243–2248, 2002.