EXTRACTING ACTIONS FROM INSTRUCTION MANUAL AND TESTING THEIR EXECUTION IN A ROBOTIC SIMULATION

Pham Ngoc Hung[1] and Takashi Yoshimi[2]
[1] Graduate School of Engineering and Science, Shibaura Institute of Technology, Tokyo, Japan, e-mail: nb14506@shibaura-it.ac.jp
[2] College of Engineering, Shibaura Institute of Technology, Tokyo, Japan, e-mail: yoshimit@shibaura-it.ac.jp

Received Date: February 3, 2016

Abstract

Service robots are expected to execute household manipulation tasks on daily basis. These tasks are often related to the operation of home appliances and can be decomposed into specific manipulation actions such as pick up a cup, place a cup, press a button, turn a knob, open or close a cover, etc. This research proposes to use task instructions collected from manuals of household appliances to extract a series of actions that a robot needs to perform in order to accomplish the task. We describe the extraction method of action and object from these task instructions including syntax parsing of sentences and searching the pairs of action and object in each parse tree based on part-of-speech. The extracted actions then will be executed by a robot. We apply a simulation environment with ROS, Gazebo simulator and a virtual robot PR2 for testing their execution. In experiments, we implement the extraction method and evaluate the extracted results of actions and object. In addition, we test the execution of actions pick and place in the proposed simulation environment.

Keywords: Action execution, Action extraction, Simulation, Service robot, Task instructions

Introduction

Robotic assistants for daily household tasks are promising targets of the robotic technology. It is expected that robots can accomplish complete tasks at home in a natural and friendly way. However, common everyday tasks such as dispensing water from water thermos pot, making a cup of coffee by coffee maker, cooking something by microwave oven, setting or tidying up a table, and so on, are still complex for robots due to the challenges in understanding the tasks and perceiving related object or equipment as well as surrounding environment. Therefore, the execution of these tasks requires and produces the great deal of knowledge and experience for robots.

There have been some recent researches emphasizing on solving the problem of understanding and performance of everyday tasks by robots. Micheal Beetz and et al. in [1, 2, 3, 4] translated the task instructions from websites into the almost executable robot plans. Their researches aim at building up a robot knowledge framework which shares the large amount of robot knowledge about tasks, actions as well as related objects and world. Mario Bollini and et al. [5] described a method of interpreting and executing recipes with a cooking robot. The authors mapped from natural language instructions to robotic instructions by designing an appropriate state-action space. Dipendra K Misra and et al. [6] carried out grounding the natural language instructions with appropriate environment context and task constraints.

In our study, we consider two features of daily household tasks. On the one hand, these tasks are often related to operating a household appliance such as water thermos pot, coffee maker, microwave oven or refrigerator, which are usually accompanied with user manuals

on certain tasks including “how to use” or “how to operate.” Figure 1 shows an example of an instruction statement taken from the manual of a water thermos pot describing the task “dispensing water”. On the other hand, each task can be decomposed into isolated actions such as pick up a cup, place a cup, press a button or key, turn a knob, open or close a cover, and so on. These actions are popular and may be repeated many times in one or some tasks every day. To accomplish this kind of tasks, hence, there are two challenges facing robots: what actions the robot needs to perform to accomplish the whole task and how the robot perform action individually. To solve the first problem, in the work at [7], we proposed the extraction method of action and object from task instructions to decompose the task into isolated actions which a robot needs to perform to accomplish that task. This method parses the syntax structure of guide sentences to obtain the parse tree of each sentence, then extracts one action and its accompaniment object from each parse tree based on part-of-speech. The obtained result is the pair of action and object which is to execute by robots in next stage. In this paper, we improve the extraction algorithm from our previous work [7] with two improvement outputs. First, as a matter of fact one guide statement may comprise of more than one action, for this we improve the algorithm to extract all possible pairs of action and object in each statement. Second, in a case there are a number of actions guided in a statement, possibly one certain action does not need performing because it is indirect action in the consequence of one main action. For this, by improving our algorithm such indirect actions shall be removed from extracted results.

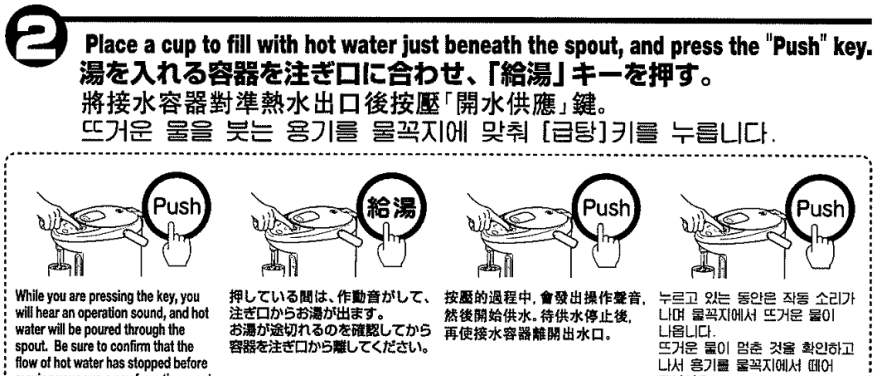

Figure 1. An example shows one guide statement of task “dispensing water” taken from the user manual of a water thermos pot

Once specific actions are extracted, it is necessary to provide the robot the execution detail of action which is the sequence of motion primitives for accomplishing that action. For example, to perform the action ‘pick up a cup’, a robot program can be written manually to control a robot arm with the sequence of motion primitives including: move the robot arm to approaching pose, open the gripper, move the arm to grasping pose, close the gripper and move arm away. In this work, we also propose to apply a robotic simulation for testing the execution of extracted action. By applying a robotic simulation which has many available functions for a virtual robot, the extracted actions can be tested whether their execution are good or not. We exploit a simulation with ROS (Robotics Operating System), Gazebo simulator and virtual robot PR2 which is equipped with adequate motion and sensing abilities for executing desired actions.

The next content of this paper is divided into two main sections. In the first one, we describe the improved method of action and object extraction from task instruction with concrete steps including: parsing grammatical structure, removing indirect action, searching action and object on parse tree and implementing the experiment to evaluate

extracted results. In the second section, we present applying the simulation for testing action execution with a demonstration of “pick” and “place” actions.

Extraction of Action and Object from Instruction

From the linguistic point of view, verbs denoting actions and objects very important in a statement because they can briefly convey the key message of the statement. It becomes especially important to the instruction in user manuals because they are often imperative statement that guide to perform certain activities. For example, with the instruction shown as in Figure 1: “Place a cup to fill with hot water just beneath the spout, and press the ‘Push’ key”, where there have two important pairs of verbs and correspondent objects ‘place-cup’ and ‘press-key’ describing the actions that need to be performed in this instruction. Basing on this key point, the extraction of action and object from instruction statements is an appropriate solution to provide robots with necessary knowledge about what to do as guided by the instruction to accomplish a task.

The extraction of information such as subjects and verbs from sentences is a quite popular technique in natural language processing. This technique is often applied in summarizing document or enriching knowledge for conceptual ontology. Delia Rusu and et al. [8] presented an approach to extracting subject-predicate-object triplets from English sentences by using four different well-known syntactical parsers including Stanford Parser, OpenNLP, Link Parser, and Minipar. However, their extraction algorithm determined only one pair of verb and object in each sentence. This leads to ignorance of other action verbs if the sentence combine more than one action verb. In natural language sentence, action and object are identified based on the part-of-speech (POS) of words in the grammatical structure. Accordingly, the extraction method of action and object is based on two main steps. Firstly, parsing the syntax structure of sentences to achieve parse trees, then searching on each parse tree to determine action and object based on POS tags that are labeled in the parse tree. The flow of this method is shown in Figure 2.

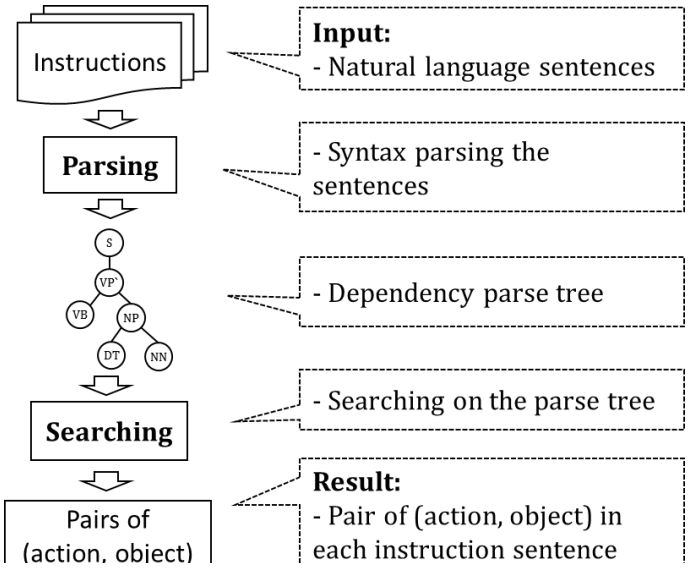

Figure 2. The extraction method of action and object from instructions

As aforementioned, we improve this extraction method after the parsing step to obtain all possible pairs of action and object and remove the indirect action which does not require to be performed.

Parsing Grammatical Structure

In the first step, syntax parsing, the input data which are natural instruction sentences in text files are parsed to determine the syntax structure of each sentence. There are some syntax parsing tools satisfying this situation. This research applied a state-of-the-art tool in this field to parse the instruction sentences, a well-known parser Stanford Parser [9] which is a Probabilistic Context Free Grammar (PCFG) parser working out the grammatical structure of sentences [10]. This parser generates the dependence parse trees. The parse tree is labeled with part-of-speech (POS) tag for words and phrases in the sentence depending on its grammatical structure. Stanford dependence parse tree is represented in text form as an example follows:

```
(ROOT
  (S
    (VP
      (VP (VB Place)
        (NP (DT a) (NN cup)
          (S
            (VP (TO to)
              (VP (VB fill)
                (PP (IN with)
                  (NP (JJ hot) (NN water)))
                (PP
                  (ADVP (RB just))
                  (IN beneath)
                  (NP (DT the) (NN spout))))))))
      (CC and)
      (VP (VB press)
        (NP (DT the) (JJ Push) (NN key))))))
```

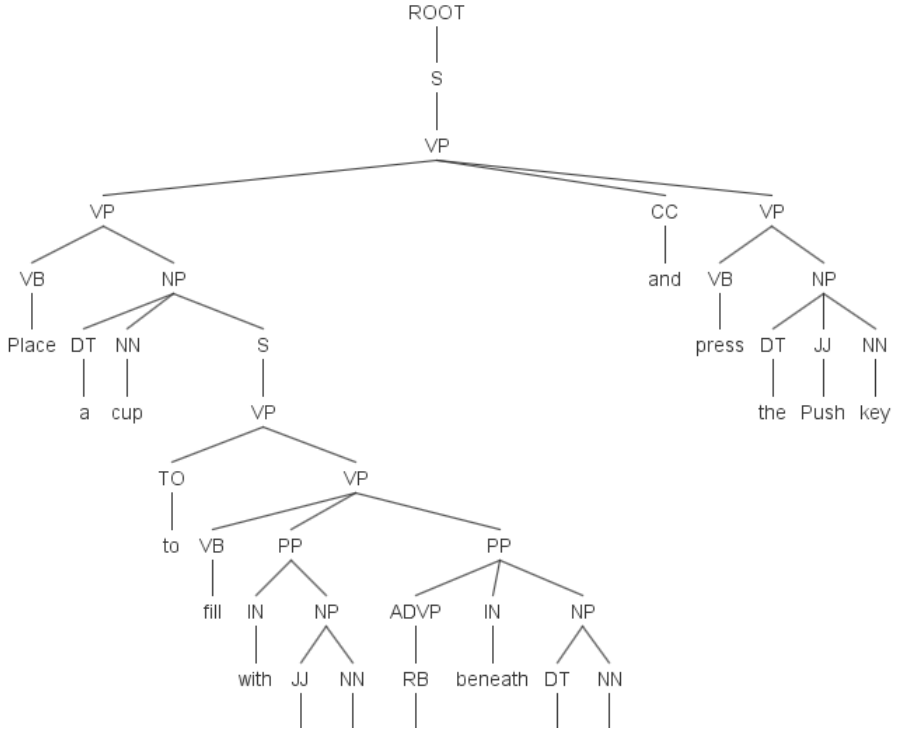

Figure 3. An example of parse tree outputted by Stanford Parser

The parse tree consists of nodes labeled with part-of-speech (POS) tags for words and phrases in the sentence. Figure 3 shows an example of Stanford parse tree in visualization. The leaf nodes are POS tags that associate with words in the sentences, non-leaf nodes are POS tags associate with phrases in grammatical structure of the sentences. Example of POS tags are NN for a noun, VB for a verb, DT for a determiner, NP for a noun phrase, PP for a preposition and so on. This statistical parser still makes some mistakes, but commonly it is evaluated working rather well and used widely in natural language processing.

Removing Indirect Action

Instruction statements may be a simple grammatical pattern consisting of one action verb and one object or may be a compound sentence that combines more than one action. In the case of the sentence with more than one action, a certain action may not require to be performed, so-called *indirect action* in the sentence. Back to the example of instruction sentence given in Figure 1, the action 'fill' (with hot water) is an indirect action which is not necessary to perform. Hence, we consider to remove this action from extracting results. This situation mainly occurs when the instruction sentence contains a dependence clause that can be omitted without changing the full meaning of the sentence. As shown in Figure 3, the action 'fill' is located in a sub tree with node "S" which is a dependence clause and can be removed. From this crucial point, we proposed a solution for removing the indirect action by pruning the sub tree of dependence clause from the parse tree of original instruction sentence before searching necessary pairs of action and object.

To this end, we exploited tree surgeon function which is known as TSurgeon tool [11] available in Stanford NLP package. It provides the way of tree editing based on the set of operations that are applied to tree locations matching a tregex pattern which is similar to regular expression for tree matching. In the example as shown in Figure 3, the condition to identify the sub tree of dependence clause is that its root node is labeled with 'S' and is dominated by a VP sub tree. The tregex pattern looks like "S=node >> VP". Then, the surgery operation 'prune' is executed to remove this sub tree from primary parse tree. Figure 4 depicts the parse tree after pruning the dependence sub tree of the given sentence example. The pairs of action and object are extracted in this edited tree. In case there is not dependent clause in a sentence matching with the mentioned-above condition, the parse tree does not change.

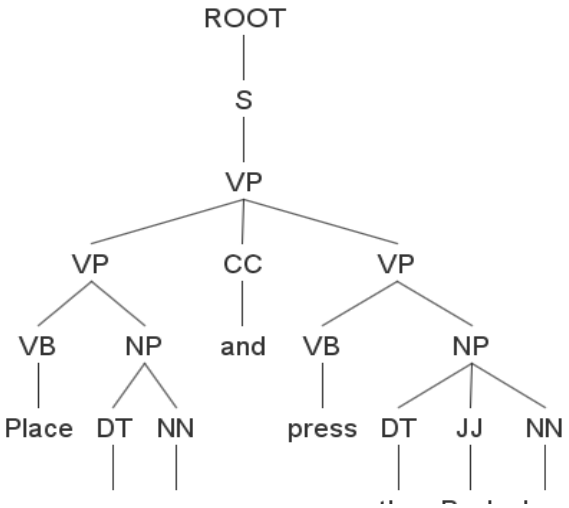

Figure 4. Parse tree of given example after pruning dependence sub tree

Searching Action and Object

The second step of the extraction method is searching actions and objects on each parse tree. After achieving the parse tree and removing the sub tree of dependence clause if possible, we searched actions and objects based on POS tags labeled in each parse tree. As mentioned above, to avoid skipping an action in the sentence containing more than one action, the searching procedure need identifying all possible pairs of action and object. To do so, it is necessary to sequentially search each action verb and its corresponding object noun in the parse tree. Firstly, one verb is searched in the first VP sub tree and assigned as the first action. With each action verb found in the VP sub tree, an object is assigned by noun following it by searching in siblings sub trees of that VP sub tree. Object is assigned as the last noun that is found firstly in sibling sub trees "NP" or if not it is assigned as the first noun found in sibling sub trees "PP."

In given example, the instruction sentence is parsed into a VP tree at top-level as shown in Figure 3. After pruning the sub tree of dependence clause, the edited parse tree is as shown in Figure 4. The searching procedure is on this tree. The first verb found in the first VP sub tree 'place' is assigned as first action. Then, its following noun 'cup' which is found in the NP sibling sub tree is assigned as the object that accompanies with the action 'place'. Next, the second pair of action 'press' and object 'key' is determined in same way. The searching procedure is repeated until the end of each parse tree to obtain all possible pairs of action and object.

Action-Object Extraction Algorithm

From the steps as mentioned above, the extraction algorithm of action and object from instruction sentences is built as follows:

```
function  getParseTree(sentence)
    word_list = tokenize the sentence
    parse_tree = parse the word_list
    return parse_tree

function pruneDependenceTree(parse_tree)
    new_tree = parse_tree after pruning dependence sub tree
    return new_tree

function  extractActionObject(new_tree)
    ao_list = []
    VP_subtree = <the top-level VP sub tree in new_tree>
    for each verb_leaf_node found in VP_subtree:
        ao_pair = []
        action = <word associated to verb_leaf_node>
        ao_pair.append(action)
        siblings = <siblings sub trees of verb_leaf_node>
        for each sib_tree in siblings:
            object = <last noun in "NP", "PP" sib_tree>
            if object:
                break
            object = <first noun in other sib_tree>
        ao_pair.append(object)
    ao_list.append(ao_pair)

    return ao_list
```

In this algorithm, each extracted object is assigned by a single noun. However, in some cases, better result can be achieved if an object is possibly identified by a noun phrase which provides more detailed information, for example, 'coffee cup', 'Push key', 'Lock/Unlock key'. In order to assign an object by a two-noun phrase, the searching procedure can be improved by finding and storing all nouns in a noun phrase and assigning two last ones, if any, for object instead of only one.

Experiments and Results

We collected test data including natural instruction sentences of several different tasks taken from home appliance manuals such as making coffee by a coffee maker, dispensing water from thermos pot, cooking something by microwave oven, etc. These task instructions are stored in text files as input data. We implemented the program of proposed algorithm to extract all possible pairs of action and object in instructions from each input text file. The extracted results are evaluated by comparing manually the pairs of action and object with respective to each described in the primary sentence to decide the accuracy of outputs. The below tables show two examples of extracted results corresponding to two input texts. In the first example, the input texts consist of only two sentences guiding a simple task 'dispensing water', a principle task described in almost all of water thermos pot user manuals. In this case, all extracted results are correct as shown in Table 1. Objects are also assigned by two nouns, if any, including a main noun and an auxiliary which provides more specific information on that object.

Text 1. Instructions of "dispensing water" task taken from a water thermos pot user manual

1. Press the "Lock/Unlock" key once.
2. Place a cup to fill with hot water just beneath the spout and press the "Push" key.

Table 1. Extracted Result of Action and Object from Task Instruction "Dispensing Water"

Sentence No.	Action	Object
1	'Press'	'Lock/Unlock', 'key'
2	'Place'	'cup'
	'press'	'Push', 'key'

In the second example, the input texts are the instructions of a more complex task, which combine many operations taken from a coffee maker user manual. The extracted results as represented in table 2 contain 6/10 instruction sentences obtaining the correct pairs of action and object (Sentences No. 1, 4, 7, 8, 9, 10). Four sentences (2, 3, 5, 6) with complex grammar structure return incorrect or empty actions due to the mistakes in syntax parsing by the Stanford parser.

Text 2. Instructions of "how to operate" task taken from a coffee maker user manual

1. Place the Brewer on a flat surface, remove protective sheet and plug into outlet.
2. Do not remove or puncture the foil lid of the K-Cup portion pack.
3. Lift front facing of the brewer to insert K-Cup.
4. Place chosen K-Cup into the K-Cup Assembly Housing.
5. Lower the front facing completely and firmly to close the Lid and puncture the K-Cup portion pack.
6. Depress the water marking button and the hot water tank will open automatically.
7. Fill the Hot Water tank with filtered or bottled water up to the FILL LEVEL indicator.
8. Close the Hot Water Tank cover.
9. Place a coffee cup in the dispense area on the drip tray.
10. Press the BREWNOW button.

Table 2. Extracted Results of Action and Object from Task Instruction "How To Operate"

Sentence No.	Action	Object
1	'Place'	'Brewer'
	'remove'	'sheet'
	'plug'	'outlet'
2	'remove'	''
	'puncture'	'pack'
3	''	''
4	'Place'	'K-Cup'
5	''	''
6	'open'	''
7	'Fill'	'water', 'tank'
8	'Close'	'tank', 'cover'
9	'Place'	'coffee', 'cup'
10	'Press'	'button'

The accuracy of extracted results depends on those of both syntax parsing tool and the searching procedure of proposed method. Stanford parser applied in this study works with very high accuracy but still has errors when the instruction sentences are formed by complex grammatical structures.

We did many experiments with other input texts, and showed here two examples. From these examples, we can confirm the availability of the processing steps of proposed method, and pointed out different cases of instruction sentences. We will consider a statistic evaluation with a larger number of data based on the category of grammatical patterns such as simple grammatical pattern consisting of only one pair of action and object or consisting of more than one pair of action and object; grammatical pattern containing a dependence clause; complex grammatical patterns including negative pattern, passive pattern, pattern with several actions but pairs of action and object are not corresponding.

Testing Action Execution in a Robotic Simulation

The sequence of actions in a task will be executed by robots to accomplish that task. However, the information on action verb and corresponding object extracted from a task

instruction provided to robots is in fact insufficient to execute the action. The next challenge as mentioned is how the action is executed by a robot. In particular, the robot needs to be provided the execution detail of action which is a sequence of motion primitives to perform that action. Additionally, the execution of actions also take place in a specific context and the robot needs to have capabilities of motion as well as sensing related objects and surrounding environment. We utilized a robotic simulation with a specific robot to test the execution of the extracted action. The simulation helps to easily implement the execution of an action. In the scope of this work, we just generated manually the sequence of motion primitives for a specific robot to execute the desired action.

Action Execution by Robot

The execution of action consists of the sequence of motion primitives in relation to sensing the object required by that action as well as surrounding world. For instance, Figure 5 shows the execution details of the actions 'pick' up an object (a) and the action 'place' an object (b), each consists of a sequence of motion primitives to complete those actions as a whole.

The action 'pick up' a certain object aims to grasp the object in the environment. Assume that the object is within the reach of the robot so that this action does not include moving the robot close to the object location. For instance, the object is a cup in the tabletop and under the 'observation' of the robot. In this case, the robot first uses a sensing function to detect the object and determines a suitable grasping pose. Then, the motion function moves the robot arm to approaching pose and open its gripper. After that, the robot approaches its gripper to the selected grasping pose and close the gripper for holding the object. Finally, the robot moves arm away from the object location with the object in the gripper.

In contrast, the action 'place' an object is to put the object into a desired location. The object in the robot gripper is put down a specific position in the known surface like tabletop. In order to execute this action, the robot moves its arm toward a desired location, opens the gripper to release the object and then moves its arm away from the object location. The final goal is the object in the desired location. These actions are considered in the context without collision avoiding of motion primitives. However, this may be added into the ability of each desired motion primitive, which assumes that a robot has already, but not effect to the generalization of these motion primitives.

'Pick up an object'
Action details:
- Move arm to approaching pose
- Open gripper
- Reach to grasping pose
- Close gripper
- Move arm away

Goal:
Object in gripper

(a)

'Place an object'
Action details:
- Move arm to placing pose
- Open gripper
- Move arm away

Goal:
Object at desired location

(b)

Figure 5. The execution detail of action: 'pick' (a) and 'place' (b)

Robotic Simulation for Testing Action Execution

In this study, we explored the simulation using ROS, Gazebo simulator, and PR2 virtual robot. ROS (Robotic Operating System) [12] is a famous platform in robotic community, which helps to create robot applications. ROS provides the services expected from an operating system, including hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management. It also provides package libraries and tools for obtaining, building, writing, and running code across multiple computers [13]. ROS supports for programming with many kinds of robots including PR2.

Gazebo is a 3D simulator designed to accurately reproduce the dynamic environments a robot may encounter. Gazebo contains libraries for physics simulation, rendering, user interface, communication, and sensor generation. It can generate both realistic sensor feedback and reliable physical interactions between objects including an accurate simulation of rigid-body physics [14]. Gazebo can be used as an independent system that stands alone outside of ROS in the latest version of ROS and Gazebo. However, it can also work as a node in ROS that simulates robot motion and exchanges data with other nodes.

In order to provide a specific context for the execution of action, a world model can be created in Gazebo simulator. The world model is built by using physical object models. These object models are available in the model database or can be created based on XML format. An example of the simulation environment was built as shown in Figure 6.

PR2 robot is sufficiently equipped capabilities of movement and perception with two arms and a mobile base, stereo camera, Kinect sensor, laser range finder. There are many ROS packages which support well for motion and sensing functions of PR2. With these available ROS packages, it is easier for implement robot program for testing the execution of action.

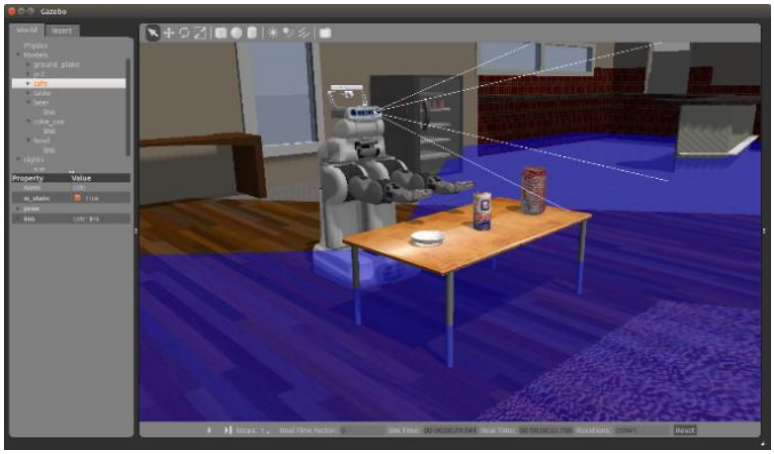

Figure 6. The simulation environment with world model and virtual PR2 robot

Experiment of 'Pick and Place' Actions

We select two actions 'pick and place' which are basic and popular in everyday manipulation tasks for experiment of the execution by PR2 virtual robot in the simulation as set above. The purpose is to test the execution details of action which are motion primitives defined beforehand.

In this experiment, we used the ROS package 'PR2 interactive manipulation' [15] to explore the execution of action 'pick up an object' and 'place the object' to fixed position

in the simulation. The object on the tabletop is available in object database. This object can be recognized by using point cloud data from Kinect sensor mounted on the robot head. This function is available in the package. The functions that control the motion of PR2 arm are also available in this package. Therefore, from these available primitive skills, it can be easier to implement the execution details of the actions 'pick' and 'place' as mentioned in Figure 5. Figure 7 shows main steps in the execution details which is the sequence of motion primitives in the execution of two actions 'pick' and 'place'. For 'pick' action, starting from initial state as Figure 7 (a), after recognizing successfully the object (a can), the robot moves his arm to approaching pose as Figure 7 (b). Then, the robot opens the gripper and moves to grasping pose as Figure 7 (c). Next, the robot closes the gripper to hold the object and moves the arm away with the object in the gripper. For 'place' action, the object keeping in the gripper now is moved to a desired location as Figure 7 (e). The robot opens gripper to release the object and moves the arm away from the object location.

By using the available functions of virtual robot PR2 in this simulation, we can build test programs for the execution of other actions. This solution also helps to create test program in high level based on primitive skills which are equipped for robots.

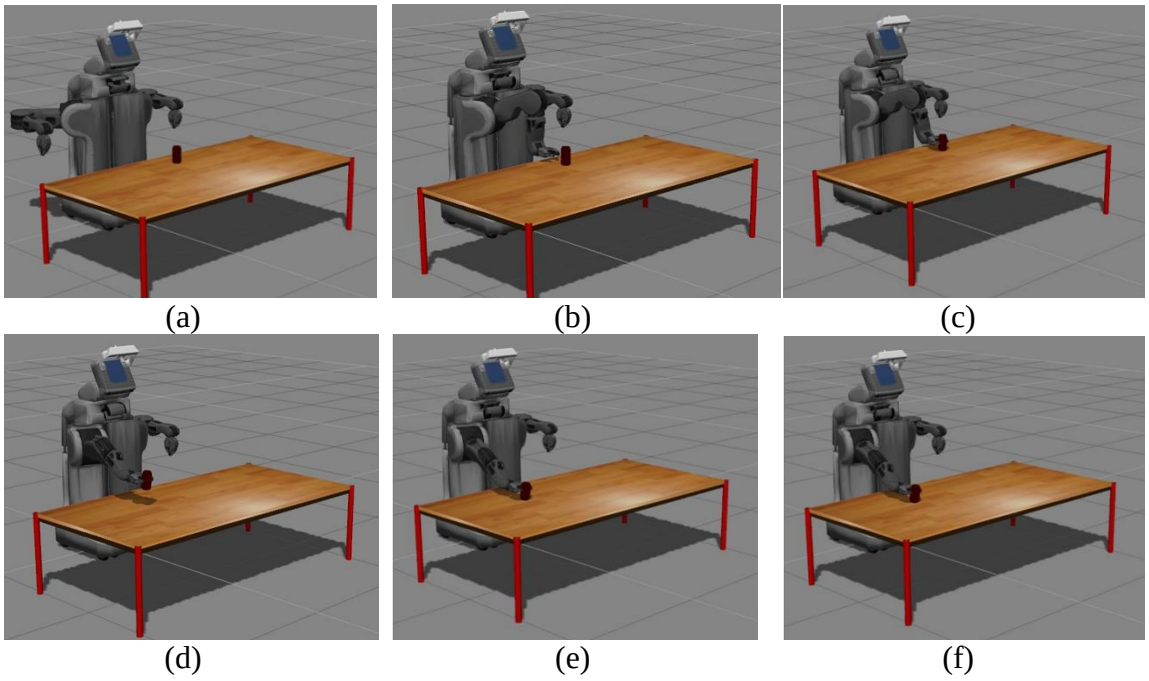

Figure 7. Testing the execution details of actions 'pick and place' by PR2 virtual robot in simulation

Conclusions

In this paper, we presented the improvement method for extracting action and object from task instructions taken from the instruction manual of home equipment. We implemented the experiment program of this method and evaluated the extracted results by manual comparison. Although the accuracy of extracted results is subject to the complexity in grammatical structure of sentences in the task instruction, the extraction method is feasible and effective to decompose the task into a sequence of isolated actions. It provides the robot with knowledge of what actions the robot needs to perform in the task. On the other hand, this paper also describes our proposed solution to apply a simulation with ROS, Gazebo simulator and virtual robot PR2 for testing the execution of extracted actions. We tested the execution of actions 'pick' and 'place' to confirm that these actions are executed

well by a virtual robot with the sequence of motion primitives are generated. In future, the execution of actions can be built based motion primitives which are generated from human demonstration.

References

[1] M. Tenorth, D. Nyga, and M. Beetz, "Understanding and executing instructions for everyday manipulation tasks from the world wide web," In: *IEEE, International Conference on Robotics and Automation (ICRA)*, pp. 1486 - 1491, 2010.

[2] M. Tenorth, U. Klank, D. Pangercic, and M. Beetz, "Web-enabled robots," *Robotics Automation Magazine, IEEE*, Vol. 18, No. 2, pp. 58–68, 2011.

[3] D. Nyga, and M. Beetz, "Everything robots always wanted to know about housework (but were afraid to ask)," In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[4] M. Tenorth, A.C. Perzylo, R. Lafrenz, and M. Beetz, "Representation and exchange of knowledge about actions, objects, and environments in the roboearth framework," *IEEE Transactions on Automation Science and Engineering*, pp. 643-651, 2013.

[5] M. Bollini, S. Tellex, T. Thompson, N. Roy, and D. Rus, "Interpreting and executing recipes with a cooking robot," *Experimental Robotics*, Volume 88 of the series Springer Tracts in Advanced Robotics, pp. 481-495, 2013.

[6] D.K. Misra, J. Sung, K. Lee, and A. Saxena, "Tell me dave: Context-sensitive grounding of natural language to manipulation instructions," In; *Proceedings of Robotics: Science and Systems*, 2014.

[7] P.N. Hung, T. Yoshimi, "Extraction of actions and objects from instructions manual for executable robot planning," In: *Proceedings of 15th International Conference on Control, Automation and Systems (ICCAS)*, Korea, pp. 881-885, 2015.

[8] D. Rusu, L. Dali, B. Fortuna, M. Grobelnik, and D. Mladenić, "Triplet extraction from sentences," Paper presented at *Conference on Data Mining and Data Warehouses (SiKDD)*, Slovenia, 2007.

[9] The Standford Natural Language Processing Group, *The stanford parser: A statistical parser*," Retrieved from http://nlp.stanford.edu/software/lex-parser.shtml

[10] D. Klein, and C.D. Manning, "Accurate unlexicalized parsing,", *In: ACL'03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, Morristown, NJ, USA, pp. 423–430, 2003.

[11] R. Levy, and G. Andrew, "Tregex and tsurgeon: Tools for querying and manipulating tree data structures,"In: *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC)*, Genoa, Italy, 2006.

[12] "*ROS Introduction*," (n.d.) [Online]. Available: http://wiki.ros.org/ROS/Introduction [Accessed: Feb, 2016]

[13] "*ROS 101: Intro to the Robot Operating System*," (n.d.) [Online]. Available: http:// www.clearpathrobotics.com/blog/how-to-guide-ros-101 [Accessed: Jan 21, 2014]

[14] W. Qian, Z. Xia, J. Xiong, Y. Gan, Y. Guo, S. Weng, H. Deng, Y. Hu, and J. Zhang, "Manipulation task simulation using ros and gazebo,"In: *Proceedings of the 2014 IEEE International Conference on Robotics and Biomimetics*, Bali, Indonesia, 2014.

[15] "*PR2 Interactive Manipulation*," (n.d.) [Online]. Available: http://wiki.ros.org/ pr2_interactive_ manipulation [Accessed: July, 2013]