Vision-based approach for task reconstruction of a robot

Seungyoub Ssin[1], Seoungjae Cho[2], Kyhyun Um*[1], Kyungeun Cho[1]

[1] Dept. of Multimedia Engineering, Dongguk University-Seoul, Seoul 100-715, Republic of Korea
[2] Dept. of Multimedia Engineering, Graduate School of Dongguk University, Seoul 100-715, Republic of Korea
*Corresponding Author: khum@dongguk.edu

Abstract: In this study, we developed a task reconstruction technique with a high success rate, which allows a robot to acquire information to create and reconstruct tasks by using a two-dimensional (2D) camera mounted on the ceiling in a closed space. The robot uses information related to each joint based on the motions learned via imitation learning when specific behaviors are performed. The robot then uses the 2D camera to reconstruct tasks based on the location data related to objects in images after they have been moved. The reconstructed tasks are executed in a closed space where a robot can move around. We introduce a technique that creates and reconstructs tasks, which can be implemented in sequential order using the extracted location data.

[Ssin S, Cho S, Um K, Cho K. **Vision-based approach for task reconstruction of a robot**. *Life Sci J* 2014;11(10):439-444] (ISSN:1097-8135). http://www.lifesciencesite.com. 60

Keywords: Behavior; imitation learning; location-based; task reconstruction; two-dimensional camera

1. Introduction

The robot industry has focused on the development of robots that could replace humans in manufacturing industries by performing tasks that require simple repetitive motions. At present, advanced robots are used widely to perform multiple tasks in a variety of fields such as healthcare, defense, and routine work.

However, robots are not capable of executing tasks in conditions that may prevent them from performing predefined movements in a previously defined zone. Programming may be limited if the development of robots depends only on the artificial intelligence embedded in the robot. Thus, a large range of tasks cannot be specified during development because of the memory limitations of robots. To address these issues, an approach must be developed that includes an external system based on vision technology using a two-dimensional (2D) camera, which does not rely only on the robot's memory and programming.

In this work, we describe the task generation and reconstruction methods for a robot based on images obtained by a 2D camera. The robot generates tasks based on data acquired during imitation learning. And the task data are then reconstructed to execute them successfully even though the location of the target object is changed.

2. Related Work

Imitation learning is a technique that allows anyone to teach a robot without specific programming knowledge. Muelling used this approach to teach the motions of a robot directly by holding the arms of a table tennis robot and performing movements (Muelling et al, 2012). However, the locations of the objects needed to be the same within a closed space to reconstruct the tasks using the motions acquired by imitation learning. Peña-Cabrera used this technique to identify and trace objects by attaching colored balls (iconographic symbols) to them (Peña-Cabrera et al. 2011). The location data of objects and a robot can then be captured with a 2D camera and used to implement robot movement tasks. Furthermore, a robot can move toward an object by communicating with a server and planning its path using the A* algorithm (Peña-Cabrera et al. 2011)(Calinon, 2009). Behaviors other than movement tasks can be generated based on their correlations with movements. Hierarchical task network (HTN) algorithm is conventionally used to plan tasks of robots (Yang et al, 2013)(Gaschler et al, 2013)(Magnenat et al, 2012). Yang and Lee segmented a task into subtasks using HTN as the task reconstruction approach. They also used this approach to identify a leaf node in a method and its preconditions (Yang et al, 2013).

In the present study, a robot recognized the locations of objects by using a 2D camera in a closed space. We also applied an approach to create and reconstruct the requisite behaviors. We introduce the technique used to make a robot implement tasks without failure in an environment that changes after learning.

3. Behavior Definition

Nao, the robot used in the present study, has 25 joints in total. When each joint is moved, the joint sends information to embedded program functions. The robot produces a database that contains the joint data generated during learning, which is saved as a file. Figure 1 shows the process used for learning imitation information, which is saved as a file that contains the extracted data.

The saved file is processed in the behavior unit, before being sent to the robot and implemented as part of a task.

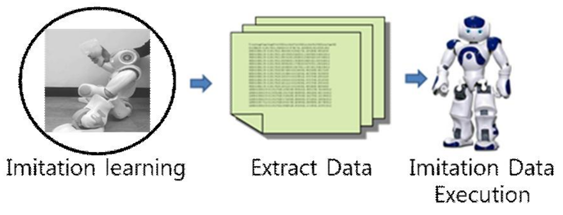


Figure 1. Process used to provide behavioral data to a robot based on imitation learning.

Behavior refers to almost insignificant small movements, which are the basic units of robot motion, including “raise an arm,” “lower an arm,” or “hold.”

The behaviors are reconstructed in a sequence that differs from the initial learning process based on a probabilistic approach during task reconstruction.

The behavioral data can be generated for an infinite range of tasks, which can include activities required for daily living. In the present study, we aimed to obtain behavioral cues using a 2D camera in a closed space (test house) during a task reconstruction experiment.

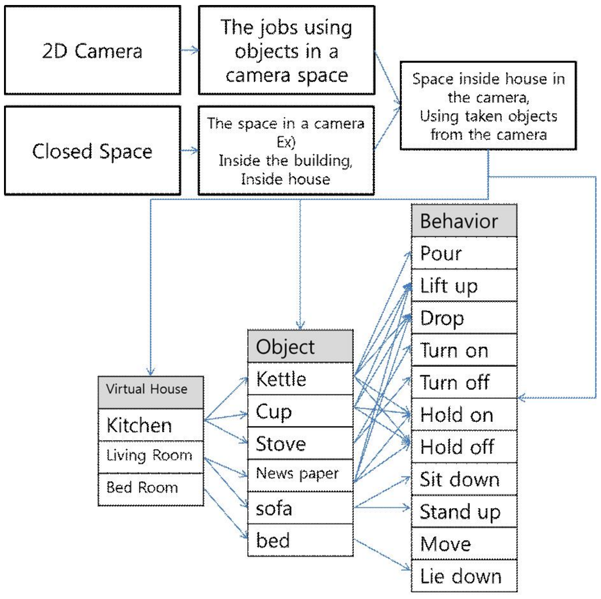


Figure 2. Process used to define the behavioral data.

The experiment only used objects that could be observed by a 2D camera. The closed space was assumed to be the internal space of a house or a building, which comprised the areas observed by a 2D camera mounted on the ceiling. We defined the test house as the closed space. The internal space of the house was divided into three sections, i.e., the kitchen, living room, and bedroom, to make the environment similar to an actual house. Figure 2 shows the objects located in each of the three sections. The behaviors were executed by using the smallest number of movements to control each object, which ranged from pouring to lying down. The final list excludes several behaviors because we found that they were not appropriate motions for a test robot to implement during the experiment, including sitting on a sofa or standing up.

Table 1. List of defined behaviors

ID	Name	Description
B0	Bookmark	Task segmentation bookmark
B1	Stand to attention	Basic attention motion of a robot
B2	Raise	Raise an object
B3	Put down	Put an object down
B4	Turn a stove on	Turn a stove on
B5	Turn a stove off	Turn a stove off
B6	Open	Open
B7	Close	Close
B8	Pour	Pour water
B9	Lie down	Lie down
B10	Move	Move

The 11 motions defined in Table 1 are motions required during daily living, which were implemented in the test house (closed space).

4. Object Extraction

For task reconstruction, the robot recognizes the required object at first. Then the robot moves toward the object location and acquires an object. To implement these functions, the robot needs to know the required object in advance and it then recognizes the locations of the object. In this experiment, the closed space was observed using a 2D camera. Thus, the robot recognizes the locations after the object had been extracted.

The 2D camera is mounted on the ceiling and it was used to observe the test house area below the ceiling. Images are acquired from above to identify the locations of objects by dividing the floor area of the test house into several cells.

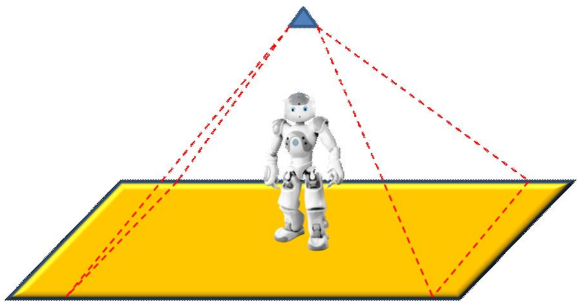


Figure 3. A two-dimensional camera capturing images from the area below the ceiling.

Figure 4 shows an image of the test house, which is obtained using the approach illustrated in Figure 3.

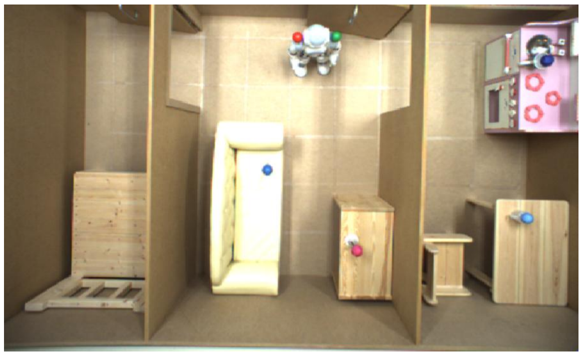


Figure 4. Test house and the objects observed using a two-dimensional camera.

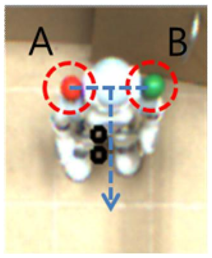


Figure 5. Approach used to recognize the direction and location of a robot based on two balls with different colors.

The approach used by the robot to recognize the locations of objects is slightly different from the general approach used to recognize objects. Two balls with different colors are attached to the shoulder of the robot, i.e., A and B in Figure 5. The center of both balls is the location of the robot. The normal vector generated by linking A and B with a line is the direction of the robot.

5. Robot Movement using a Two-dimensional Camera

The robot needs to move to perform its tasks. A system is also required to recognize the locations of objects and to facilitate the movement of the robot in a closed space. In the present study, we use a 2D camera and an iconographic technique to recognize these locations by attaching colored balls to the robot and various objects. As shown in Figure 6, we use OpenCV to extract specific colors from the images of the colored balls attached to objects, which were acquired with a 2D camera.

Figure 6 illustrates the process used to identify the locations of objects with colored balls.

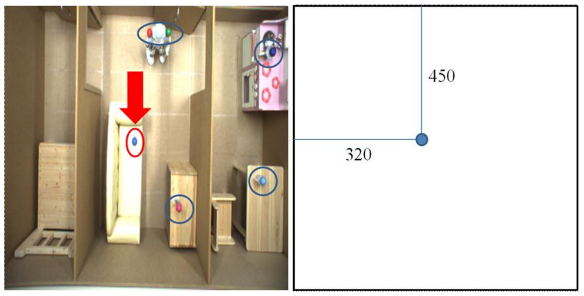


Figure 6. Extracting the locations of objects using colored balls.

The center of an extracted color is recognized as the location of a specific object. The direction and location of the robot are recognized using the two colored balls on the shoulders of the robot. The center coordinates of the two balls and the normal vector of the line that connected both balls are recognized as the location and the direction, respectively.

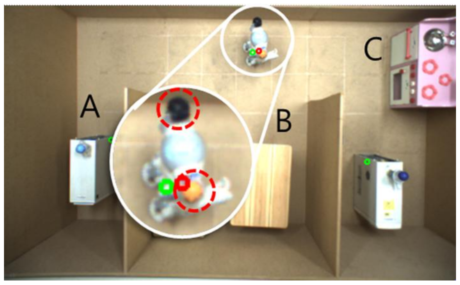


Figure 7. Recognizing the locations of the robot and an object using the iconographic symbol technique.

A* is used as the movement algorithm. Cell-based calculation is the simplest method for recognizing locations in the closed space. A* is used because it is suitable for path planning between start and end points.

```
FUNCTION MoveRobot( image, lsc, rsc, target )
BEGIN
    p, d ← CalcPositionDirection( image, lsc, rsc )
    Q ← CalcAstarPath( p, target )
    WHILE Q is not empty do
        next ← Q.Pop()
        length ← GetLength( next - p )
        angle ← GetAngle( d, next - p )
        socket.Send( length, angle )
        p, d ← CalcPositionDirection( image, lsc, rsc )
    END
END
```

Figure 8. A* movement algorithm

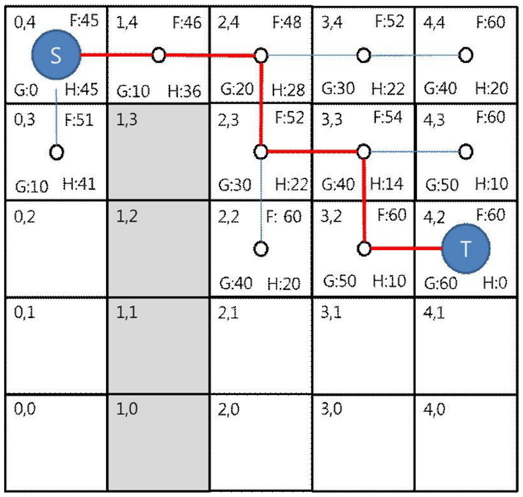


Figure 9. Example of the application of the A* algorithm to the actual movement of the robot.

The formula used by the actual A* algorithm is F = G + H, where G is the cost consumed from the starting point to the present point, H is the estimated cost from the present point to the destination, and F is the final cost obtained by adding G and H. In general, the A* algorithm is based on cell units, where diagonal movements are made after searching the eight surrounding cells. In this study, we only used four cells, i.e., up, down, left, and right, because the robot frequently stumbled when it hit its shoulders against a wall if a diagonal movement was allowed in the actual experiment.

Figure 9 shows a simulation of a test path obtained using the approach applied in the present study. The search is activated at (1,4) and (0,3) from the starting point S. The F value was lowest at (1,4), so it is selected as the next step in the movement path. In Figure 9, the shortest distance is the final line that connects the parent (previous) node to T, which was determined by searching from (2,4) to (4,2), as explained above. However, finding the movement path is not the final outcome because we also needed to move the robot. Thus, we added a function for rotating the robot 90° around in sections such as (2,4) during the middle of the process.

6. Task Reconstruction

The motion data generated during imitation learning are behavioral data that corresponded to each specific movement, including "raise a cup" and "put down a cup." To generate a task with behavioral data, the motions need to be connected and implemented in a specific sequence. We records specific tasks using the motion data acquired by demonstration learning. Next, the implementation of the recorded tasks is considered as the basic task implementation. The addition of a new movement task in a changed object location, the changed environment, and the reconstruction of the task in a different sequence are defined as task reconstruction.

Figure 10 shows the process used to reconstruct tasks. The behavior step records the data generated during imitation learning to produce a file. Each behavior is recorded as a task that combined different sequences. The recorded tasks are grouped based on their relevancy. B10, i.e., the movement behavior, is generated using images acquired by the 2D camera when reconstructing a task.

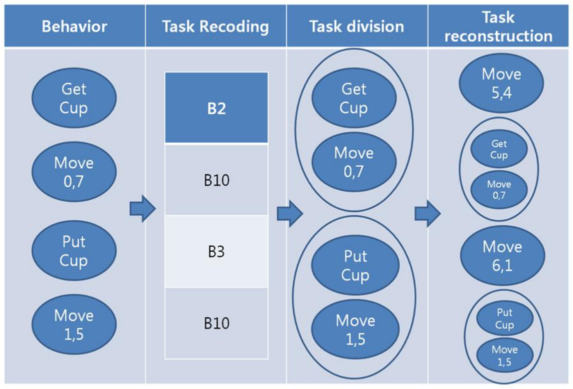


Figure 10. Task reconstruction process

To demonstrate the generation of movement behaviors and to reconstruct existing tasks, a scenario was created that required the generation of basic tasks. This scenario was: "get a glass of water from the kitchen (C)". The data learned for this scenario were saved in a separate task table. In Figure 7, the robot started from B and moved to a point in the closed space. The initial location of the glass of water was C, but it was changed to A after generating the learning data in the task reconstruction experiment. The robot found the glass of water by moving from B to C.

Thus, the robot learned how to collect water in a glass and return from C to B. During the implementation of the same task, the glass of water was moved to A. Furthermore, the robot acquired information that the glass of water had moved to A based on the observations made by the 2D camera. Next, the robot collected a glass of water by moving from B to A. The robot then moved to C, poured water in a glass, and returned to B. Figure 11 shows the task reconstruction algorithm used by the robot.

```
FUNCTION GenerateTask()
BEGIN
    OL ← GetCurrentObjects()
    IF  CanGenerateTask( OL ) THEN
        TSlist ← GetSubtaskListFromTaskMap( OL )

        WHILE TSlist is not empty do
            ExecuteTask( TSlist );
        ENDWHILE
    ENDIF
END
```

Figure 11. Task reconstruction algorithm

7. Experiment

The experiment compared the initial task implementation results with the results obtained after reconstruction using the same scenario to assess the task implementation results.

First, the robot learned the motions required for the defined scenario by imitation learning. The data were extracted by a separate server. In this scenario, the robot collected a glass of water from the kitchen. The required behaviors included "raise an object," "put an object down," and "pour." Raising the kettle and a glass were both treated as "raise an object." The robot learned the task in sequential order after learning the behaviors, as described above.

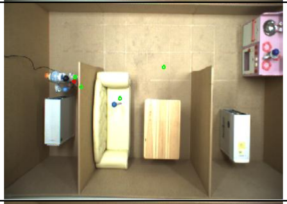
Next, the locations of the objects were changed and the procedure required to find the objects was generated. After data processing, a check was performed to determine whether the robot was configured correctly to implement the task in the sequence required by the user.

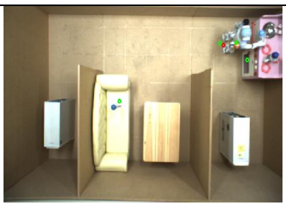
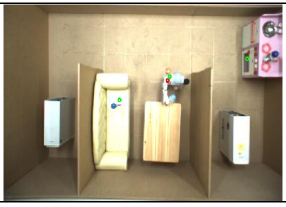
Table 2 shows the differences between the data during the initial task learning process and the behaviors after task reconstruction using the algorithm described above.

Table 2. Reconstructed task data

No	First Task	Task reconstruction
1	Start	Start
2	Get(Kettle)	Goto(7,2)
3	Pour(Kettle, Cup)	Get(Kettle)
4	Put(Kettle)	Goto(7,4)
5	Get(Cup)	Put(Kettle)
6	Goto(4,2)	Goto(1,2)
7	Put(Cup)	Get(Cup)
8	End	Goto(7,4)
9		Put(Cup)
10		Get(Kettle)
11		Pour(Kettle, Cup)
12		Put(Kettle)
13		Get(Cup)
14		Goto(4,2)
15		Put(Cup)
16		End

Table 3. Experimental results and movement processes executed by the robot

Step	Reconstructed Subtask	Implementation of the Process Based on the Camera Observations
1	Start	
2	Goto(7,2) Get(Kettle) Goto(7,4) Put(Kettle)	
3	Goto(1,2) Get(Cup) Goto(7,4) Put(Cup)	

4	Get(Kettle) Pour(Kettle, Cup) Put(Kettle) Get(Cup) Goto(4,2) Put(Cup)	
5	End	

In Table 3, step 1 shows the robot at the starting point (living room). In step 2, the robot moves to the bedroom and finds the kettle. In step 3, the robot goes to a small table in the kitchen to collect a glass. In step 4, the robot holds a kettle and pours the water into the glass. In step 5, the robot brings the glass to the starting point and puts it down.

The locations of the objects during imitation learning were changed when the robot implemented the tasks by itself in the experiment. However, the robot implemented the task successfully by reconstructing them during the experiment.

8. Conclusion

This study has described how we created and reconstructed tasks for a robot based on images acquired using a 2D camera. The robot implemented the task based on data acquired during imitation learning and the task data were then reconstructed to allow task execution after the locations of the objects were changed. The robot completed the task successfully although the locations of the objects changed.

When a 2D camera was used in the actual experiment, each area had different color values due to the effects of external light. These effects frequently caused the robot to fail to recognize an object. Further research needs to be conducted to improve the object recognition rate with the 2D camera by overcoming the effects of external light. Furthermore, we aim to investigate how to make the robot implement tasks in an expanded space, rather than task reconstruction in a specific closed space.

27/5/2014

Acknowledgements:

This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012R1A1A2009148).

Corresponding Author:
Prof. Kyhyun Um
Department of Multimedia Engineering
Dongguk University
Seoul 100-715, Republic of Korea
E-mail: khum@dongguk.edu

References

1. Muelling K, Kober J, Kroemer O, Peters J. Learning to Select and Generalize Striking Movements in Robot Table Tennis. Association for the Advancement of Artificial Intelligence 2012.
2. Peña-Cabrera M, Lopez-Juarez I, Ríos-Cabrera R, Castelán M, Ordaz-Hernandez K. Object Location in Closed Environments for Robots Using an Iconographic Base. Robot Arms 2011;11:201-214.
3. Calinon S. Robot Programming By Demonstration: a Probabilistic Approach. EPFL Press 2009;1:4-29.
4. Yang T, Lee W. A Service-Oriented Framework for the Development of Home Robots. International Journal of Advanced Robotic Systems 2013;10.
5. Gaschler A, Petrick R. P. A, Kroger T, Knoll A, Khatib O. Robot Task Planning with Contingencies for Run-time Sensing. IEEE International Conference on Robotics and Automation (ICRA) Workshop on Combining Task and Motion Planning 2013.
6. Magnenat S, Chappelier J, Mondada F. Integration of Online Learning into HTN Planning for Robotic Tasks. Association for the Advancement of Artificial Intelligence 2012.