Research Article

Human-Robot Interaction Learning Using Demonstration-Based Learning and Q-Learning in a Pervasive Sensing Environment

Yunsick Sung, 1 Seoungjae Cho, 2 Kyhyun Um, 3 Young-Sik Jeong, 3 Simon Fong, 4 and Kyungeun Cho 3

1 The Department of Game Mobile Contents, Keimyung University, Daegu 704-701, Republic of Korea
2 Department of Multimedia Engineering, Graduate School of Dongguk University, Seoul 100-715, Republic of Korea
3 Department of Multimedia Engineering, Dongguk University, Seoul 100-715, Republic of Korea
4 Department of Computer and Information Science, University of Macau, Macau 3000, China

Correspondence should be addressed to Kyungeun Cho; cke@dongguk.edu

Received 28 August 2013; Accepted 16 October 2013

Copyright © 2013 Yunsick Sung et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Given that robots provide services in any locations after they move toward humans, the pervasive sensing environment can provide diverse kinds of services through the robots not depending on the locations of humans. For various services, robots need to learn accurate motor primitives such as walking and grabbing objects. However, learning motor primitives in a pervasive sensing environment are very time consuming. Several previous studies have considered robots learning motor primitives and interacting with humans in virtual environments. Given that a robot learns motor primitives based on observations, a disadvantage is that there is no way of defining motor primitives that cannot be observed by a robot. In this paper, we develop a novel interaction learning approach based on a virtual environment. The motor primitives are defined by manipulating a robot directly using demonstration-based learning. In addition, a robot can apply Q-learning to learn interactions with humans. In an experiment, using the proposed method, the motor primitives were generated intuitively and the amount of movement required by a virtual human in one of the experiments was reduced by about 25% after applying the generated motor primitives.

1. Introduction

In pervasive sensing environments, robots can provide various services in an active manner. Irrespective of the locations of humans, robots can provide services after they move toward humans based on the information of the humans’ daily life [1]. However, the following problems may occur after a robot learns interactions with a human. First, a robot cannot learn interactions with humans rapidly, which leads to learning time problems during interaction learning. Therefore, it is necessary to learn interactions with humans without the participation of humans. Second, an interaction between a robot and a human could injure the latter, because of the incomplete perception of robots. Therefore, protective equipment is required by humans.

Previous studies have considered interaction learning with a human in virtual environments, which can solve the problems described above [2–4]. A virtual robot can generate its motor primitives by observing a virtual human in virtual environments and utilizing demonstration-based learning. However, unobservable motor primitives cannot be generated. In addition, because of the differences in the appearance of a robot and a human, the motor primitives of robots may differ from the movements of humans and it might not be possible to perform the motor primitives generated for a robot. Thus, the methods used to generate different motor primitives need to be improved. Further research is required to determine how to teach motor primitives to a robot while learning interactions with humans in a virtual environment.
In this paper, we propose a virtual pervasive sensing environment-based interaction learning method that utilizes demonstration-based learning to learn motor primitives and Q-learning to execute motor primitives. The motor primitives are defined during manipulations based on demonstration learning, so the motor primitives can be generated intuitively by users who are not programmers. The application of Q-learning allows the newly generated motor primitives to be performed without modifying any of the algorithms after their production.

The remainder of this paper is organized as follows. Section 2 introduces demonstration-based learning approaches and virtual environment-based learning. Section 3 proposes an interaction learning method for a virtual pervasive sensing environment. Section 4 presents the results of interaction learning experiments in virtual pervasive sensing environments. Finally, we provide our conclusions in Section 5.

2. Related Work

Various types of learning algorithms are required to allow robots to interact with humans. In this section, we summarize related research into the learning of motor primitives and the learning of interactions with humans in virtual environment.

The motor primitives learned by robots are very important for achieving their goals. The repulsion of robots can be reduced by different motor primitives. Different types of research are ongoing to produce motor primitives for robots that appear more natural, like those of humans. For example, a related study defined natural motor primitives for following the shortest path [5, 6]. A genetic algorithm was used to generate these movements. Following mutation, the motor primitives that failed to follow the shortest path were eliminated and new motor primitives were generated. Another approach is to use demonstration-based learning [7–9]. Demonstration-based learning algorithms learn each motor primitive separately based on repetition, before analyzing the same learned motor primitives [7, 10]. Another approach involves learning motor primitives by dividing a series of movements [8], where each motor primitive is defined as a part of the series of movements. Furthermore, an approach was proposed that generates motor primitives as a hierarchical tree [9, 11]. Within the same hierarchical tree, a robot executes the same motor primitive initially but executes different motor primitives in different states. The motor primitives are usually generated by planning algorithms [12]. However, some problems may occur if planning algorithms are applied. For example, planning algorithms are defined based on the generated motor primitives. If the motor primitives change, the planning algorithms must also be changed to execute the motor primitives. An advantage of demonstration-based learning is that humans can define motor primitives without any requirement for programming. However, this advantage does not apply to planning algorithms. Therefore, algorithms are required that are not affected by changes to the motor primitives.

There is a method that learns the interaction with humans by utilizing motor primitives after generating the motor primitives using demonstration-based learning [13]. A previous study defined a virtual human and a virtual robot, where the former is a virtual agent that behaves in virtual environments in the same way as a human in a virtual environment, while the latter behaves like a real robot. Therefore, a virtual robot interacts with a virtual human to learn an interaction with a real human. If a virtual human executes a motor primitive, the virtual robot also executes the motor primitive at the same time. However, virtual-based interaction learning has problems. For example, the motor primitives used by a virtual robot cannot be generated if a virtual human does not execute the motor primitives, because they are generated by observing the virtual human. Therefore, another approach is required for generating motor primitives.

Thus, we propose a new approach for defining the motor primitives for a virtual robot. We also apply Q-learning to solve the problem of executing motor primitives, which does not require any changes after the modification of motor primitives.

3. Virtual Learning Framework for Human-Robot Interaction

3.1. Concept. In a pervasive sensing environment, it takes a long time to learn interactions with humans and the number of interactions with robots is limited. Therefore, the number of interactions should be reduced to increase the amount of the learning to facilitate the high quality execution of motor primitives. In our approach, the interactions are learned via a virtual pervasive sensing environment, so no interactions are required in real pervasive sensing environments, as shown in Figure 1.

We define two types of virtual agents for learning in a virtual pervasive sensing environment: a virtual human and a virtual robot. The virtual human acts like a human while the virtual robot executes motor primitives to collaborate with the virtual human. The virtual robot learns interactions with real humans by interacting with virtual humans. The learning result is then embedded in the real robot. The real robot executes motor primitives based on the results of virtual learning to interact with a real human.
Table 1: Approaches used in different stages of interaction learning by robots.

| Stage                  | Type of agent | Learning approach                                |
|------------------------|---------------|--------------------------------------------------|
| Motor primitive learning | Virtual robot | Direct manipulation of a robot                    |
| Collaboration learning  | Virtual robot | Interaction with a virtual human by Q-learning [13] |

There is no requirement for interactions with real humans. The learning time problem is always invoked if a human is involved during learning processes, which makes it very hard to reduce the learning time. However, the learning time can be reduced more by increasing the speed of interactions between a virtual human and a virtual robot. This is because a virtual human and a virtual robot do not need to execute motor primitives at the same speed as a real human and a real robot.

In our approach, interaction learning includes human modeling, motor primitive learning, collaboration learning, deployment, and collaboration stages. In this paper, we only propose the processes used during the motor primitive learning stage and the collaboration learning stage as shown in Table 1. During the human modeling stage, humans control a virtual human to make them act like humans by executing predefined motor primitives. The virtual humans learn how to execute motor primitives by analyzing the human control process. During the motor primitive learning stage, humans control the virtual robots directly to teach them how to move, and the virtual robots then generate their own motor primitives. Next, the virtual robot interacts with a virtual human by executing the learnt motor primitives. During this interaction, the virtual robot learns how to provide services to humans. The results obtained from motor primitive generation and from interactions are then applied in a real robot, which can interact with real humans.

3.2. Human-Robot Interaction Framework and Processes. The roles of real humans are divided into two groups during whole learning processes: one for residents and the other for operators. Operators teach real robots while residents live in pervasive sensing environments. All of the virtual humans in the virtual pervasive sensing environment are virtual residents. We also define a robot server as a server that generates motor primitives and policies, which transfers data between a real robot and a virtual robot. Our proposed framework is shown in Figure 2.

First, an operator controls a virtual human via a user interface. During the motor primitive learning stage, there are two modules in a real robot: a motor measurer and a motor primitive generator. The motor measurer is deployed in a real robot. When the operator manipulates a real robot directly, the motor measurer determines the degrees of the joints in the real robot. The motor primitive generator is embedded in the robot server rather than the real robot, which separates the dependency of the motor primitive generator from the robot platform. The generated motor primitives are deployed in the virtual robot and the real robot.

During the collaboration learning stage, a policy generator and a motor primitive executor are utilized to learn the interactions between a resident and a real robot based on the interactions that occur between a virtual human and a virtual robot. The motor primitive executor executes the generated motor primitives and the policy generator then generates the results of the interaction. The interaction results are then deployed in the real robot. Finally, the real robot can provide various services by executing the motor primitives based on the interaction learning results.

In our approach, a robot executes multiple motor primitives. $M_i$ is the $i$th motor primitive. A motor primitive is defined as a part of a series of movements, which is described by multiple joints of the robot. Therefore, $M_i$ comprises multiple joints. The $k$th joint of the $i$th motor primitive is defined
by $M_{i,k}$. $M_{i,k,h}$ is the $h$th measured $M_{i,k}$. If $\xi$ is the number of joints, $M_i$ is $\langle M_{i,1}, \ldots, M_{i,\xi} \rangle$. Each joint moves irregularly. $t_{i,h}$ denotes the time when $M_{i,k,h}$ is executed. Finally, the set $M$ is a motor primitive set. Figure 3 shows the example of the configuration of the motor primitive set.

To eliminate any differences between motor primitives of a virtual robot and a real robot, the motor primitive generator generates the same motor primitives for both. To reduce the number of movements measured, any movements are eliminated that do not change as much as the difference calculated using (1). After similar movements are eliminated, the motor primitives are generated using the remaining measured movements. Consider

$$(M_{i,k,1} - M_{i,k-1,1})^2 + (M_{i,k,2} - M_{i,k-1,2})^2 + \cdots < \delta^2. \quad (1)$$

Given that pervasive sensing environment is usually complex, the policy generator used by our approach utilizes $Q$-learning [14] to execute the generated motor primitives, because $Q$-learning has the advantage that a model of the environment does not need to be defined. In addition, $Q$-learning algorithm does not need to be modified after the motor primitives are generated. The policy generator selects motor primitives depending on the current state $s$ and sends the selected motor primitive to the motor primitive executor for execution. After executing each motor primitive, the corresponding reward of the executed motor primitives is calculated and transferred back to the policy generator. The policy generator updates the $Q$-values with the reward using

$$Q(s, M) \leftarrow Q(s, M) + \alpha \times \left[ r + \gamma \max Q(s', M') - Q(s, M) \right], \quad (2)$$

where $M$ is an executed motor primitive, $s$ is a state, $r$ is a reward after executing $M$, $s'$ and $M'$ are the next state and the next motor primitive, respectively, $\alpha$ denotes the learning rate, and $\gamma$ is a discount factor.

The motor primitive executor receives motor primitives from the motor primitive generator and executes the motor primitives according to the decisions made by the policy generator. After executing the motor primitives, the corresponding reward of the executed motor primitives is transferred to the policy generator.

4. Experiment

4.1. Configurations of the Real and Virtual Pervasive Sensing Environments. In our experiment, we used a Nao as a real robot. We also built a model house, which was a suitable size for the Nao, as shown in Figure 4. The model house contained a kitchen, living room, and bedroom. The Nao learned during interactions with a real human.

The objective of the Nao was to transfer the objects required by a real human. After recognizing the object, the Nao moved toward the object initially. Next, it grabbed the object, moved toward the real human, and gave the object to the real human. In the experiments, we used the objects shown in Table 2. There were two types of objects: static objects that could not be moved and movable objects, which a Nao and a human could grab, carry, and put down.

The state space must be defined in advance to use $Q$-learning. In this experiment, we denoted the positions of the human and the robot based on their grid coordinates, after taking a picture using an omnicamera placed on the ceiling and dividing the picture into the grid shown in Figure 5. The size of each cell was set to the width of the Nao. Thus, 50 cells were defined. We defined each state based on the coordinates of the human, the robot, and the object located nearest to the human.

To learn interactions between a real human and a real robot, the virtual pervasive sensing environment used in this experiment was modeled in exactly the same way as the real pervasive sensing environment, as shown in Figure 6. Therefore, the structure and size of the virtual pervasive sensing environment were the same as the real pervasive
sensing environment. Objects were also deployed in the same way as the real pervasive sensing environment. We utilized two virtual agents as a virtual human and a virtual robot.

4.2. Configuration of the Motor Primitives. A real operator controlled a virtual robot, while a virtual human and a robot server were also used, depending on the stage. The robot followed a different process during each stage and the real operator also controlled the state of the real robot by touching a touch sensor on the head of the real robot.

The motor primitives of the robot were defined as follows. The real operator manipulated the robot directly to make the robot learn the motor primitives. There were two types of motor primitives. First, a type of motor primitive was predefined by programming, as shown in Table 3. For example, given that an initial motor primitive was required and that it was very hard to define a walking motor primitive by

| Location       | Object         | Object type |
|----------------|----------------|-------------|
| Kitchen        | Cup            | Movable object |
|                | Kettle         | Movable object |
|                | Chair          | Movable object |
|                | Kitchen table  | Static object |
|                | Stove          | Static object |
| Living room    | TV table       | Static object |
|                | TV (assumed)   | Static object |
|                | Couch          | Static object |
|                | Remote controller | Movable object |
|                | Newspaper      | Movable object |
| Room           | Bed            | Static object |

Table 2: Objects used in the experiments.

| Location  | Object     | Object type |
|-----------|------------|-------------|
| Kitchen   | Cup        | Movable object |
|           | Kettle     | Movable object |
|           | Chair      | Movable object |
|           | Kitchen table | Static object |
|           | Stove      | Static object |
| Living room | TV table   | Static object |
|            | TV (assumed) | Static object |
|            | Couch      | Static object |
|            | Remote controller | Movable object |
|            | Newspaper  | Movable object |
| Room      | Bed        | Static object |

Table 3: Predefined motor primitives for a virtual robot and a real robot.

| Notation | Name                   | Description                                                                 |
|----------|------------------------|-----------------------------------------------------------------------------|
| $M_0$    | Standing before        | If a real robot has not grabbed an object, it stands and waits to execute the next motor primitive |
|          | grabbing               |                                                                             |
| $M_1$    | Standing after         | If a real robot has grabbed an object, it stands and waits to execute the next motor primitive |
|          | grabbing               |                                                                             |
| $M_{10}$ | Walking                | A real robot follows a ball while remaining at a fixed distance from the ball |
Table 4: Virtual human animations.

| Name                  | Description                                                                 |
|-----------------------|------------------------------------------------------------------------------|
| Standing              | Standing with arms down                                                      |
| One-hand grabbing     | Stretching arms, grabbing objects, and carrying objects while standing       |
| One-hand placing      | Stretching arms and placing one of the grabbed objects while standing        |
| Touching              | Turning the switch of a light or stove on or off                             |
| Receiving             | Receiving an object with the right hand                                       |
| Giving                | Giving an object with the right hand                                         |
| Walking               | Walking toward a specific object                                             |
| Sitting               | Sitting on a chair or couch                                                  |
| Laying                | Laying down on a bed                                                         |

Manipulation, the real robot executed two preprogrammed standing motor primitives and one walking motor primitive. The other type of motor primitive was defined by the manipulations performed by the operator.

For the walking motor primitive, the algorithm determined a path from the current coordinates to specific coordinates. We used the A* search algorithm because the grids of the virtual and real pervasive sensing environments were not complex and they only comprised 50 cells. For example, if a real operator was in the specific position where a virtual human needed to move, the virtual human moved to the position while avoiding objects and walls.

While the real robot was learning the motor primitives, the real robot measured its joints every 500 ms and transferred the values of the joints to the robot server. If the interval is set under 500 ms, the joints are not measured accurately, which delays the performance of the real robot.

We predefined the animation of the virtual human, as shown in Table 4. The objective of the Nao was to transfer objects for a virtual human, so the animation of the virtual human also focused on transferring objects.

4.3. Motor Primitive Generation Experiment. The first experiment aimed to generate motor primitives for the Nao. An operator defined the motor primitives from $M_2$ to $M_7$ by manipulating the arms and touching the touch sensors on the arms, as shown in Table 5. In this experiment, the operator only controlled the arms because the legs only moved when the robot walked.

The real robot executed a series of motor primitives. The end of a motor primitive was connected to the end of the next motor primitive in a natural manner. Thus, the standing motor primitives were executed after each motor primitive and the next motor primitive started after the end of the standing motor primitive. Therefore, we defined the sequence of motor primitives as shown in Figure 7.

Some of motor primitives could not be connected with the standing motor primitive because of the grabbed objects. Therefore, standing after grabbing was added. Standing after grabbing was performed after executing, receiving, or one-hand grabbing, followed by one-hand placing or giving.

Each motor primitive was generated based on separate manipulation performed by a real human. Figure 8 shows four of the generated motor primitives. Only five joints were measured, which were all related to the right hand. The generated motor primitive was then performed by the virtual robot.

4.4. Interaction Learning Experiment. We specified a scenario for learning the interactions. First, we applied our approach to the scenario where a human stood up, sat on a couch, and then read a newspaper after picking it up, as shown in the following list (a).

Interaction Learning Results

(a) Scenario where a virtual human lives alone is as follows:

(i) a virtual human sleeps,
(ii) the human wakes up on a bed,
(iii) the human walks to a couch,
(iv) the human sits on the couch for a while,
(v) the human stands up on the couch,
(vi) the human walks to a newspaper,
(vii) the human picks up the newspaper, and
(viii) the human reads the newspaper.

(b) Scenario where a virtual robot provides services is as follows:

(i) a virtual human sleeps,
(ii) the human wakes up on a bed,
(iii) the human walks to a couch,
(iv) while the human sits on the couch:
   (1) a virtual robot walks to a couch, and
   (2) picks up the newspaper.
(v) when the human stands up on the couch:
   (1) the robot walks to a virtual human, and
   (2) gives the newspaper.
(vi) the human receives the newspaper, and
(vii) the human reads the newspaper.

Figure 9 shows the accumulated rewards according to the increase in the amount of interaction learning. After 14,000, the robot started to learn the interaction. The previous list (b) shows the changed scenario by the virtual robot based on the result of the interaction after the interaction learning. If a virtual human lived alone, the virtual human walked to the newspaper and picked it up for itself. However, if a virtual robot was present, the virtual robot walked to the newspaper and picked it up, then walked to the virtual human, and gave it the newspaper.
5. Conclusion

In this paper, we developed an approach to virtual pervasive sensing environment-based interaction learning where the operators taught motor primitives to a real robot by manipulating its arms directly. The learned motor primitives were utilized by a virtual robot and executed to learn interactions with a human. The operators defined the motor primitives using manipulations, so various different types of motor primitives could be defined intuitively, which overcame the problems of previous approaches.

The virtual human and the virtual robot used in our proposed method and Q-learning are suitable for single agent-based learning algorithms, so it is necessary to improve our proposed method by applying multi-agent-based Q-learning. A method is also required to allow a virtual robot to provide services to multiple virtual humans. Finally, an approach will be developed to facilitate the application of the learned interaction results to a real robot.

Acknowledgments

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 (2011-0011266). And this work was also 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).

References

[1] T. Teraoka, “Organization and exploration of heterogeneous personal data collected in daily life,” Human-Centric Computing and Information Sciences, vol. 2, no. 1, pp. 1–15, 2012.

[2] M. Lim and Y. Lee, “A simulation model of object movement for evaluating the communication load in networked virtual environments,” Journal of Information Processing Systems, vol. 9, no. 3, pp. 489–498, 2013.

[3] H. T. Panduranga, S. K. Naveen Kumar, and H. S. Sharath Kumar, “Hardware software co-simulation of the multiple image encryption technique using the xilinx system generator,” Journal of Information Processing Systems, vol. 9, no. 3, p. 499, 2013.

[4] Y. Sung and K. Cho, “Collaborative programming by demonstration in a virtual environment,” IEEE Intelligent Systems, vol. 27, no. 2, pp. 14–17, 2012.

[5] S. Ra, G. Park, C. H. Kim, and B.-J. You, “PCA-based genetic operator for evolving movements of humanoid robot,” in Proceedings of the IEEE Congress on Evolutionary Computation (CEC ’08), pp. 1219–1225, Hong Kong, China, June 2008.

[6] G. Park, S. Ra, C. Kim, and J.-B. Song, “Imitation learning of robot movement using evolutionary algorithm,” in Proceedings of the 17th World Congress, International Federation of Automatic Control (IFAC ’08), pp. 730–735, Seoul, Republic of Korea, July 2008.

[7] S. Calinon, F. Guenter, and A. Billard, “On learning, representing, and generalizing a task in a humanoid robot,” IEEE Transactions on Systems, Man, and Cybernetics B, vol. 37, no. 2, pp. 286–298, 2007.

[8] N. Koenig and M. J. Matarić, “Behavior-based segmentation of demonstrated task,” in Proceedings of International Conference on Development and Learning (ICDL ’06), 2006.

[9] M. N. Nicolescu and M. J. Matarić, “Natural methods for robot task learning: instructive demonstrations, generalization and practice,” in Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS ’03), pp. 241–248, Melbourne, Australia, July 2003.

[10] S. Calinon and A. Billard, “A probabilistic programming by demonstration framework handling constraints in joint space and task space,” in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS ’08), pp. 367–372, Nice, France, September 2008.

[11] M. N. Nicolescu and M. J. Matarić, “Extending behavior-based systems capabilities using an abstract behavior representation,” in Proceedings of the AAAI Fall Symposium on Parallel Cognition, pp. 27–34, 2000.

[12] M. J. Matarić, “Sensory-motor primitives as a basis for imitation: linking perception to action and biology to robotics,” in Imitation in Animals and Artifacts, pp. 391–422, MIT Press, 2000.

[13] Y. Sung and K. Cho, “A method for learning macro-actions for virtual characters using programming by demonstration and reinforcement learning,” Journal of Information Processing Systems, vol. 8, no. 3, pp. 409–420, 2012.

[14] C. J. C. H. Watkins and P. Dayan, “Q-learning,” Machine Learning, vol. 8, no. 3-4, pp. 279–292, 1992.
Submit your manuscripts at
http://www.hindawi.com