2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Learning of Tool Force Adjustment Skills by a Life-sized Humanoid using Deep Reinforcement Learning and Active Teaching Request

Yoichiro Kawamura, Masaki Murooka, Naoki Hiraoka, Hideaki Ito, Kei Okada and Masayuki Inaba

***Abstract*— The purpose of this study is to make life-sized humanoid robots acquire tool manipulation skills that require complicated force adjustment. The difficulty in acquisition of tool manipulation skills comes from the hardship in physical modeling. Recent research have revealed that deep reinforcement learning (DRL), a model-free approach, performs superior in such tasks. However, DRL in general has a drawback in sample efficiency, and this becomes critical in robot learning especially in life-sized humanoid robots. In this study, we propose an integrated system incorporating DRL method and active learning. Our method also leverages a variety of previous studies on life-sized humanoid robots to overcome the sample efficiency issue. We demonstrated the effectiveness of our proposed system through a hacksaw skill acquisition and a Japanese planer (Kanna) skill acquisition by a life-sized humanoid robot.**

I. Introduction

Just as we humans have used tools to expand the range of activities, robots can also use tools to expand the range of activities. Hence, the tool manipulation by robots has been a research subject for many years. The difficulty in the tool manipulation with a robot comes from the hardship in creating a physical model for a task. Mainly this is because the tool manipulation is an indirect operation, causing the increase in the number of uncertain parameters and unobservable states. Especially for the tool manipulation that requires complex force adjustment such as sawing (Fig. 1), it is extremely difficult to create a physical model for a task.

In order to overcome the difficulty of modeling, model-free reinforcement learning (RL) has been actively studied in many years and has been widely applied to the robotics research. As for the tool manipulation, tasks such as pancake flipping and Ball-in-a-Cup are realized by RL [1] [2]. In particular, deep reinforcement learning (DRL) has been attracting attention because it has made significant achievements in complex tasks such as end-to-end picking control, in-hand manipulation, and door opening from scratch [3] [4] [5]. A drawback of the DRL is a low sample efficiency, and this becomes a crucial issue when it comes to a real robot, especially with a life-sized humanoid robot which has the higher running cost such as the risk of falling down than normal manipulators.

To overcome the sample efficiency issue, we propose two approaches summarized in the following.

Y. Kawamura, M. Murooka, N. Hiraoka, H. Ito, K. Okada, and M. Inaba are with Department of Mechano-Informatics, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan y-kawamura@jsk.imi.i.u-tokyo.ac.jp

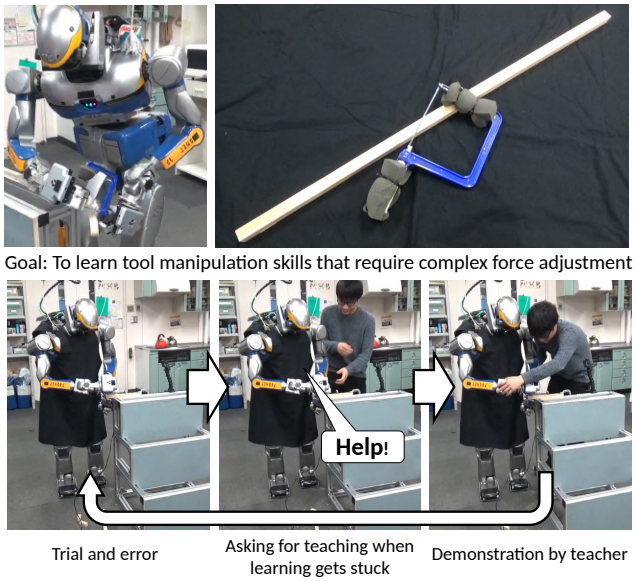


Fig. 1: Top: The goal of this research is that the robot learns the tool manipulation skills that require complicated force adjustment such as sawing. Bottom: In order to acquire those skills, we propose a method in which the robot actively requests teaching when the robot is not learning the skills well.

A. Our Approach and Related Work

1) Integrated System for Skill Acquisition by Life-sized Humanoid Robot: Thanks to the various previous research on a humanoid robot such as fall prevention and motion generation, we already have a lot of knowledge, and we can benefit from these studies, instead of learning everything end-to-end. Also, we focus on the characteristic that it is relatively easy to know how to move a tool, even though adjusting the force is difficult. For example, we know in advance that we have to push and pull the saw when we use it for wood cutting. According to these observations, in order to improve the sample efficiency, we make use of those previous studies and reduce the parts to be learned by DRL. Specifically, the force adjustment part is acquired by DRL, and we utilize motion generation and balance control from previous research.

2) Learning with Active Teaching Request: Numbers of techniques have been studied to utilize demonstrations to increase sample efficiency on DRL [6]–[10]. A life-sized humanoid robot has the strong advantage that a person can teach the robot the skills just as that person would teach another person. However, there are three major problems

978-1-7281-6211-9/20/$31.00 ©2020 IEEE

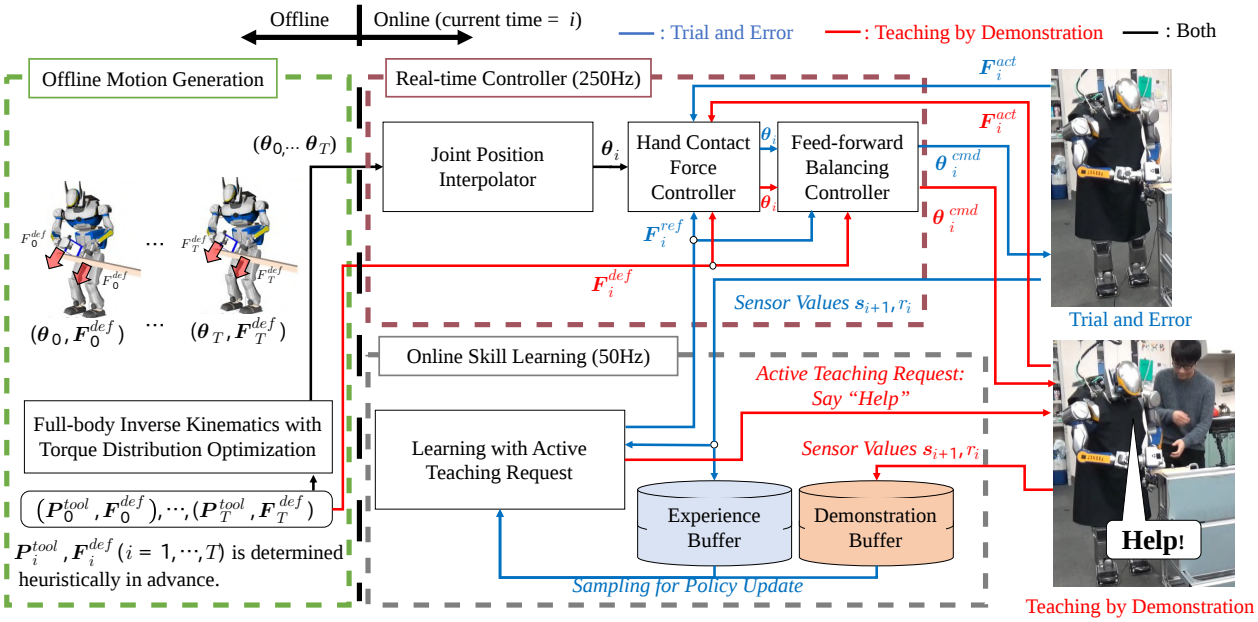


Fig. 2: The overview of the integrated system proposed in this study.

to be solved in order to utilize demonstrations with a life-sized humanoid robot. (i) Firstly, we cannot know in advance how many demonstrations should be given. (ii) Secondly, old demonstrations are inaccurate because the real-world environment changes slightly over time. (iii) Thirdly, the teacher's actions are not always accessible.

To overcome these problems, we propose a novel learning method called *Learning with Active Teaching Request* (LATR) as an extension of off-policy DRL such as Deep Q-network (DQN) [11] and Deep Deterministic Policy Gradient (DDPG) [12]. As the name suggests, LATR is inspired by active learning techniques. By the active teaching method, we can overcome the problem (i). Related previous work on RL and active learning include the reward shaping and interactive human guidance [13], active reward learning [14], and active deep Q-learning based on the confidence of the action [15]. Our approach is similar to these studies, but in LATR, the robot determines whether it needs teaching on the basis of transition of rewards. This approach is based on the following simple observations: When a person acquires tool manipulation skills, he/she first performs trial and error by him/herself, and when he/she finds difficulty in trial and error, he/she requests an expert to teach. In addition, in order to overcome the problems (ii), LATR raises the importance of the latest trial and error. Also, in order to overcome the problems (iii), LATR estimates the teacher's actions online. We will explain the detail of LATR in Section III.

B. Contribution of This Study

The contributions of this study are as follows.

- An integrated system that learns tool manipulation skills with DRL while making use of the knowledge that has been obtained with the model based research on life-sized humanoid robots.
- A learning method that actively requests teaching based on the transition of the reward and estimates the teacher's actions online.
- Acquisition of skills in sawing and planing wood using a life-sized humanoid robot.
- Presenting application to making a bookshelf with learned skills.

In the following, Section II describes the detail of the integrated system mentioned in I-A.1, and Section III describes LATR in detail. We demonstrate the effectiveness of our approach through the experiments in Section IV.

II. Integrated System for Skill Acquisition by Life-sized Humanoid Robot

In this study, we propose an integrated system for skill learning that utilizes the knowledge that has been studied in life-sized humanoid robots for many years. Fig. 2 shows the overview of the proposed system. The proposed system consists of three components: offline motion generation, online skill learning, and real-time control.

A. Offline Motion Generation

As mentioned in the Section I, it is relatively easy to know how to move the tools in advance. Therefore, this component creates a motion for manipulating the tool offline.

Firstly, in order to create the motion, we determine the sequence of the pose of the tool $\boldsymbol{P}_i^{tool}$ and the sequence of the target contact reaction force $\boldsymbol{F}_i^{def}$ heuristically. Note that the target contact reaction force $\boldsymbol{F}_i^{def}$ is an approximate value, which is not appropriate for actually using a tool.

Next, we generate a sequence of joint positions of the robot $\boldsymbol{\theta}_i$ by using full-body inverse kinematics and torque distribution optimization [16]. Optimizing torque distribution prevents the motor temperature from rising, and makes it

possible for the robot to operate continuously for a long time. For efficient skill learning, it is especially important for robots to be able to operate continuously.

B. Online Skill Learning

The learning component runs at a lower frequency than the real-time controller. While learning, the component switches two modes, (1) trial and error and (2) teaching by demonstration. When the learning component selects trial and error mode, similar to the normal reinforcement learning, the component receives sensor values as a state, and the component outputs the target force applied to the tools as an action according to its policy. On the other hand, when the learning component selects teaching by demonstration mode, the robot first requests a human expert to teach by saying "Help!", and the human expert performs teaching by kinesthetic teaching. Then, during the demonstration, the robot saves the state transitions based on its sensor values and stores them in its demonstration buffer.

C. Real-time Control

The Real-time controller modifies the pre-generated motion to realize the target reaction force in both hands while preventing the robot from falling down, then the controller commands the joint positions to motors in the end. In the case of trial and error, the controller receives $\boldsymbol{F}_i^{ref}$ calculated by the learning component as the target contact reaction force. On the other hand, in the case of teaching, the controller receives $\boldsymbol{F}_i^{def}$ which is determined heuristically beforehand. The controller works as follows: 1) In order to realize $\boldsymbol{F}_i^{ref}$ or $\boldsymbol{F}_i^{def}$ in both hands, the hand contact force controller modifies positions of both hands by impedance control algorithm [17] according to errors between $\boldsymbol{F}_i^{act}$ and $\boldsymbol{F}_i^{ref}$ or between $\boldsymbol{F}_i^{act}$ and $\boldsymbol{F}_i^{def}$. 2) In order to compensate $\boldsymbol{F}_i^{ref}$ or $\boldsymbol{F}_i^{def}$ to keep balancing, the feed-forward balancing controller modifies the center of gravity. 3) The controller commands the joint positions to the motors.

III. Learning with Active Teaching Request

In this section, we detail the *Learning with Active Teaching Request* (LATR), a learning algorithm that acquires skills while actively requesting teaching. In Fig. 2, this section corresponds to the part surrounded by the dashed gray line.

A. Background: Off-policy Deep Reinforcement Learning with Replay Buffer

In recent off-policy deep reinforcement learning using experience replay represented by DQN and DDPG, an algorithm seeks a policy to maximize reward by repeating task trials. Although there are various derivations depending on whether the action is discrete or continuous, and what algorithm is used to update the policy, the basic framework is to repeat the following two steps:

- Action Execution:
  The agent's policy determines the action a_t based on the current state s_t. After executing action, the agent observes the next state s_{t+1} and reward r_t. Then, the transition (s_t, a_t, r_t, s_{t+1}) is stored in the replay buffer.
- Policy Update from Replay Buffer:
  Agent updates its policy using minibatch of transitions randomly sampled from replay buffer.

B. Overview of LATR: Learning with Active Teaching Request

In LATR, three steps are added to the conventional off-policy reinforcement learning: judgment of teaching necessity, learning from the demonstration buffer, and estimating teacher behavior. Algorithm 1 is a pseudo-code algorithm of LATR. The demonstration buffer described in the Algorithm 1 is a memory having the same data structure as the replay buffer and is used for storing transitions observed while teaching.

Algorithm 1 Learning with Active Teaching Request: LATR

$\mathcal{D}_{exp}$: replay buffer
$\mathcal{D}_{demo}$: demonstration buffer
e_{prev} : index of the last episode for which the teaching was performed
m : interval for checking teaching necessity

1: $\mathcal{D}_{exp} \leftarrow \{\}$ // Initialize $\mathcal{D}_{exp}$
2: $\mathcal{D}_{demo} \leftarrow \{\}$ // Initialize $\mathcal{D}_{demo}$
3: **for** $e = 1, .., M$ **do** // repeat episode for M times
4: **if** $checkTeaching(\mathcal{D}_{exp})$ and $e \ mod \ m == 0$ **then**
5: Request teaching by speaking
6: Observe demonstration transitions
7: Save demonstration transitions to $\mathcal{D}_{demo}$
8: $e_{prev} \leftarrow e$
9: **end if**
10: Observe initial state s_0
11: **for** $i = 0, .., T$ **do** // each episode has T frames
12: Execute action a_i and observe s_{i+1}, r_i
13: Store (s_i, a_i, r_i, s_{i+1}) to $\mathcal{D}_{exp}$
14: Update policy from replay buffer $\mathcal{D}_{exp}$
15: $updateFromDemo(e, e_{prev}, \mathcal{D}_{exp}, \mathcal{D}_{demo})$
16: **end for**
17: $updateDemonstrationBuffer(\mathcal{D}_{exp}, \mathcal{D}_{demo})$
18: **end for**

The processing from the 12th to 14th lines described in Algorithm 1 are the same as the processing performed in conventional off-policy reinforcement learning (DQN, DDPG, etc.). The details of $checkTeaching$ on line 4, $updateFromDemo$ on line 15, and $updateDemonstrationBuffer$ on line 17 are described in the subsections below.

C. Judging Demonstration Necessity

The determination as to whether to request the demonstration from human expert is made simply by comparing the recent rewards with the past rewards. Algorithm 2 shows a pseudo-code representation of this process. Let $R_{k;0}$ be the average reward from the last k episodes to the present, let $R_{2k;k}$ be the average reward from the last $2k$ episodes to the last k episodes, and let ir (improve threshold) be the threshold that determines whether the reward is improving.

Note that $R_{k;0}$ represents recent rewards and $R_{2k;k}$ represents past rewards. In LATR, $R_{k;0}$ and $R_{2k;k}$ are compared, and if $R_{k;0}/R_{2k;k} > ir$, it is judged that the reward is sufficiently improved and teaching is not necessary. Conversely, if $R_{k;0}/R_{2k;k} \leq ir$, it is determined that teaching is necessary.

Qualitatively, ir is a parameter indicating how much the robot depends on the human expert. When ir is ∞, the robot requests teaching every time, and when ir is zero, the robot performs a trial and error by itself every time.

Algorithm 2 Judgment of Demonstration Necessity

k : window size for comparing reward
ir : task improvement threshold
Require:
$\mathcal{D}_{exp}$: replay buffer
Ensure:
TRUE or FALSE : TRUE when teaching is needed and FALSE for otherwise

function $checkTeaching(\mathcal{D}_{exp})$
  Get $r_{t-2k}, ..., r_t$ from experience memory $mathcal D_{exp}$
  $R_{k;0} \leftarrow (r_{t-(k-1)} + ... + r_{t-1} + r_t)/k$
  $R_{2k;k} \leftarrow (r_{t-2k} + ... + r_{t-(k+1)} + r_{t-k})/k + 1$
  if $R_{k;0}/R_{2k;k} \leq ir$ **then**
    Return TRUE
  else
    Return FALSE
  end if
end function

D. Policy Update from Demonstration

In the off-policy deep reinforcement learning algorithms such as DQN, DDPG, and their derivatives, the policy is updated by using the transition sampled from the replay buffer. In LATR, the policy is not only updated from the replay buffer, but also updated from the demonstration buffer. The same algorithm is used for updating policies from the demonstration buffer as for updating policies from the replay buffer. The benefit of this is that LATR can be applied generally to off-policy deep reinforcement learning using replay buffers.

Especially in LATR, when updating the policy from the demonstration buffer, the number of updating policy N_{demo}, is adjusted. Algorithm 3 shows the procedure of the policy update from demonstration. The adjustment is performed as follows:

1) Set N_{demo} high immediately after teaching
2) Decrease N_{demo} linearly with the number of episodes passed since last teaching

The reason for (1) is to perform learning so as to emphasize the imitation of the teacher and to find a better trajectory in a wide search space at an early stage. The reason for (2) is to take an adaptive action in a changing environment. In the real world, the environment changes slightly depending on the time. Therefore, we aim to acquire adaptability to the environment by placing importance on information based on the trial and error of the robot acquired in the recent environment, rather than the information at the time of teaching acquired in the old environment.

Algorithm 3 Update Policy from Demonstration

N_{demo} : number of iteration to update policy from demonstration buffer
N_{demo_ini} : initial N_{demo} to set after teaching.
r_d : ratio of decay
Require:
$\mathcal{D}_{exp}$: replay buffer
$\mathcal{D}_{demo}$: demonstration buffer
e_{cur} : index of current episode
e_{prev} :index of the last episode for which the teaching was performed

function $updateFromDemo(e_{cur}, e_{prev}, \mathcal{D}_{exp}, \mathcal{D}_{demo})$
  $N_{demo} \leftarrow \lfloor N_{demo\_ini} - r_d * (e_{cur} - e_{prev}) \rfloor$
  if $N_{demo} \geq 1$ **then**
    for i = 1 ... N_{demo} **do**
      Sample random transitions from $\mathcal{D}_{demo}$
      Update Policy using sampled transitions
    end for
  end if
end function

E. Estimating Teacher's Action

Generally, even if the state transition at the time of teaching is accessible, the action taken by the teacher is not always accessible. For example, when only an image of a dashboard camera is given as teacher data for automatic driving, the operation of the steering wheel and the accelerator cannot be fully accessible from the image alone. Similarly, teaching a robot how much force to apply to a tool through kinesthetic teaching is the case where the action of the teacher is not fully accessible. Therefore, in LATR, we train state transition model $f : S \times S \rightarrow A$ with the state transitions (s_t, a_t, s_{t+1}) in replay buffer. Then, by using this model, we estimate the teacher's action. Algorithm 4 shows a pseudo-code of this procedure. Specifically, the state transition model was implemented with a neural network in LATR. The estimation of the teacher's action by the model f is performed for all the transitions stored in the demonstration buffer at the end of each episode. This is to ensure that the transitions in the demonstration buffer is always based on the latest estimation when updating the policy.

Related previous studies include model-based reinforcement learning [18] and system identification [19]. In these methods, model of the state transitions is learned from the record of the movement of the robot and are used for learning policy or designing controllers. LATR uses these techniques to estimate teacher's actions.

IV. Experiment

In order to confirm the effectiveness of the proposed method, LATR, we conducted an experiment with a simula-

Algorithm 4 Learning physical model and estimating action to reproduce demonstration

f : neural net model that predict a from (s, s_{next})

Require:

$\mathcal{D}_{exp}$: replay buffer

$\mathcal{D}_{demo}$: demonstration buffer

1: **function** $updateDemonstrationBuffer(\mathcal{D}_{exp}, \mathcal{D}_{demo})$

2: Train f with transitions that are randomly sampled from $\mathcal{D}_{exp}$.

3: Estimate teacher's action for all transitions stored in $\mathcal{D}_{demo}$.

4: Update $\mathcal{D}_{demo}$ with newly estimated teacher's action

5: **end function**

tor. In addition, as a real-world experiment of skill acquisition by a life-sized humanoid, we conducted a hacksaw skill acquisition and a Japanese planer (Kanna) skill acquisition by a life-sized humanoid robot HRP2 [20].

A. Simulated Experiment: Inverted Pendulum

1) Experimental setup: An experiment to learn the control of inverted pendulum was performed on a simulator [21]. Learning was performed using DQN and LATR at different irs. For comparison, the policy update algorithm of LATR was set in the same way as DQN. In another word, the procedures from the 12th to 14th lines described in Algorithm 1 are the same procedures of DQN. Also, LATR and DQN shares the same network structures for Q-network.

For Q-network, we used two-hidden-layer network with size 150-150 and used hyperbolic tangent function (Tanh) as activation. The network size was determined empirically so that the learning process terminates within 20ms (50Hz) per frame. Discount factor of $\gamma = 0.95$ was chosen, and for exploration strategies, epsilon-greedy with $\varepsilon = 0.2$ was chosen for both LATR and DQN. For f in Algorithm 4, we used one-hidden-layer network with size of 100 and used ReLU for activations. Adam optimizer [22] is chosen with base learning late of 0.001 for the learning of both Q-network and f. As for m in Algorithm 1 and k in Algorithm 2, we chose $m = k = 5$.

The default reward has been overwritten, and the agent receive the reward of 1 at the end of the episode if the agent could stand up the inverted pendulum more than 195 frames, otherwise -1. The agent only receives rewards at the end of an episode and receives zero rewards during the episode. The maximum simulation time of each episode was set to 200 frames, and we conducted 10 sets of 750 episodes of learning for each algorithm. As for the teacher's demonstration, we used a record of the trajectory of a previously trained agent executing a task.

2) Experimental result: The results of the learning curve for DQN and LATR at different ir are shown in Fig. 3. Table I represents the average of the number of teachings. For the inverted pendulum control task, Fig. 3 shows that the more the request for teaching is, the faster the learning is raised.

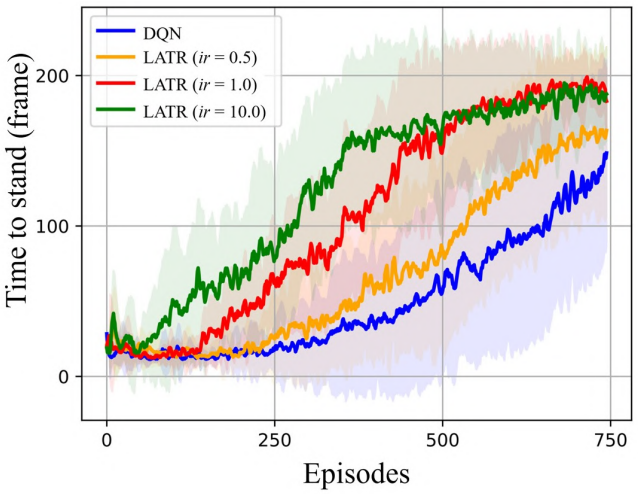


Fig. 3: The figure shows the learning curves comparing DQN and LATR (ours) with different parameters of ir. The bold line shows the average, and the filled range represents the range of the variance 1σ.

TABLE I: Average number of requests for teaching

DQN	LATR (ir=0.5)	LATR (ir=1.0)	LATR (ir=10.0)
0	1.6	59.3	148.0

B. Real-world Experiment: Hacksaw manipulation skills

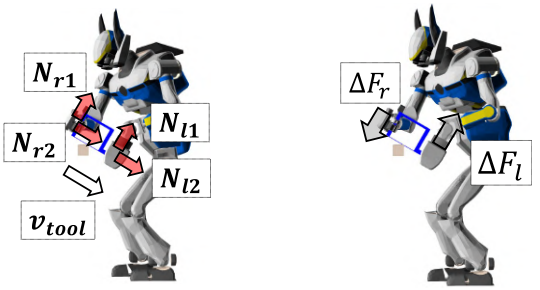
1) Experimental setup: In this experiment, DQN was also used as the off-policy reinforcement learning algorithm for LATR. The sawing motion was created in advance as described in Section II.

State features are five-dimensional and include the reaction forces of both hands and the velocity of the tool. The reaction forces are two-dimensional for each hand: the direction in which the hacksaw is pushed and pulled, and the direction in which the hacksaw is pressed. State is a 20-dimensional vector that connects state features for the last four frames. The reaction forces are measured using a 6-axis force sensor attached to the robot's wrists.

Discrete actions are expressed by strengthening or weakening the target reaction force in each hand. We set five actions in each hand: strengthening, strengthening slightly, not chancing, weakening slightly, and weakening. The pair of the five actions in each hand was used as the discrete actions. Hence, the actions consist of 25 different action. In this study, we used ± 1 N change in target reaction force for strengthening or weakening and used ± 0.5 N for slighter changes. Fig. 4 explains the state and action used in this experiment.

One episode in learning was a movement in which the hacksaw was reciprocated once, and an episode was divided into 80 frames. The actions were determined in each frame. We conducted three sets of 110 episodes training each on LATR and DQN.

Rewards are determined based on the sound. Roughly, the reward is set to be high when the sound is close to the sound of the expert's operation, and the reward is set to be low when the sound is different from the expert's operation.



(a) State features include the reaction forces of both hands and the velocity of tool.

(b) Action is to increase or to decrease the target contact force in the next frame.

Fig. 4: State and action defined for hacksaw task.

Specifically, for 80 frames that constitute one episode, the sound in each frame is identified, and the total number of frames determined to be close to the expert operation is calculated as a reward. For example, if there are sounds that are determined to be close to a human sound in 30 frames out of 80 frames, the reward is calculated as 30. Sound identification is performed by classifying spectrogram images using a neural network that had been trained in advance. We chose NIN [23] as the network structure. Fig. 6 shows a spectrogram image when the expert operates a hacksaw. To collect a data set for pre-training, a microphone was placed beside the wood and the expert actually cut the wood with a hacksaw, as shown in Fig. 5. At this time, the expert not only operated the hacksaw well to collect positive examples but also intentionally operated the hacksaw poorly to collect negative examples. A total of 240 sound samples were prepared, including positive and negative examples. Among the prepared samples, 80% was set as the train data and 20% was set as the validation data. As a result of 20 epochs of training with the augmented data, the identification success rate with the validation data was 90.7 %.

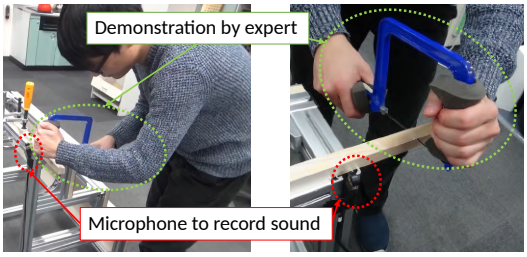


Fig. 5: The expert operates the hacksaw and samples the sound to train the NN to determine whether the operation sound of the hacksaw is close to the sound of expert's operation.

2) Execution of teaching: Fig. 7 shows how the human expert teaches to the robot. The human expert taught the robot by applying force to the link closer to the body than the link with the robot's 6-axis sensor. By doing so, the resultant force of the force applied to the tool by the human and the robot can be measured by the six-axis force sensor attached in robot's wrist. While teaching, the robot was controlled with impedance control so that the tool and wooden parts can

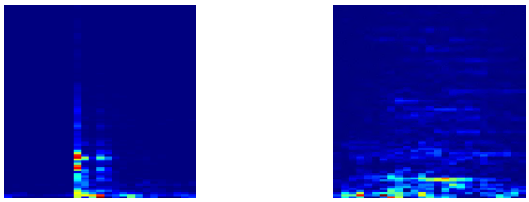


Fig. 6: Left: An example of a spectrogram image when the hacksaw is operated successfully. Right: An example of a spectrogram image when a hacksaw is operated poorly.

naturally interact with each other. In addition, we created an interface to input whether the teaching was successful, and if it failed, we did not save the data in the demonstration buffer, preventing the incorporation of incorrect teaching data.

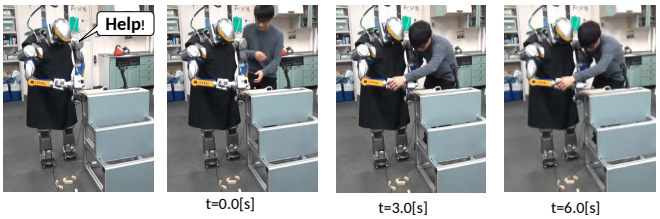


Fig. 7: The robot requests teaching based on recent reward transitions, and a human expert teaches the robot the appropriate target reaction force through kinesthetic teaching.

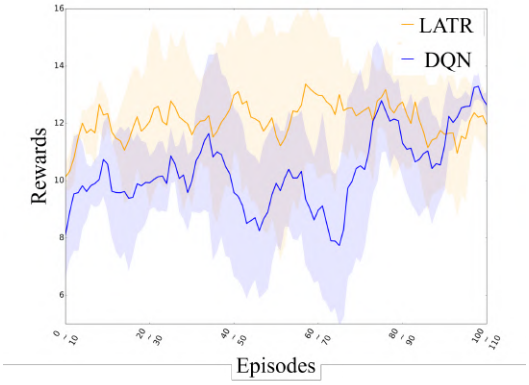


Fig. 8: This graph shows the reward gained by the HRP2 while learning wood cutting skill in average over 10 episodes.

3) Experimental result: Fig. 8 shows the moving average of the reward while learning the hacksaw task. In the LATR, the robot asked humans for teaching 17 times on average. From Fig. 8, the rewards in LATR and DQN converge to the same after around 90th episode, but on the other hand, from 0th to 80th episode, the reward increases faster with LATR, suggesting that LATR enables efficient learning.

C. Real-world Experiment: Acquisition of Japanese Planer (Kanna) manipulation skills

1) Experimental setup: We also conducted experiments to learn the skills of Japanese Planer (Kanna). Japanese planers, like hacksaws, are also tools that require complex adjustment

of force. As for the state and the action in learning, the target reaction force of the hand and the speed of the tool were used as in the experiment with the hacksaw. Other parameters in the LATR were the same as in the hacksaw experiment. The reward was judged by the human expert. The reward is 1 if the robot can handle the planer correctly and -1 if it fails. Fig. 9 shows the situation when the planer was handled correctly and when it failed. As shown in Fig. 10, the teaching of the planer operation was performed by the human holding the upper arm of the robot and correcting the force to apply while the robot was operating the planer. The experiment with the planer took longer than the experiment in the hacksaw, so we conducted the experiment only with LATR.

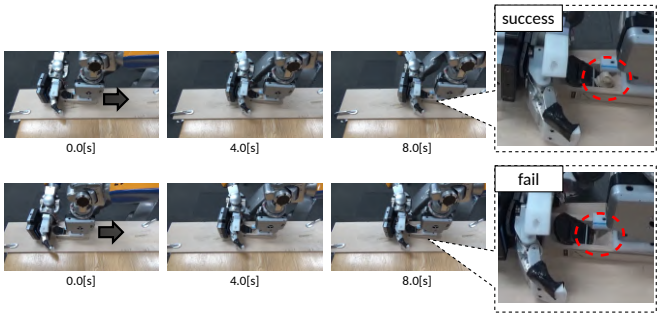


Fig. 9: Figures in upper row shows successful handling of the Japanese planes. On the other hand, the figure below shows when it failed.

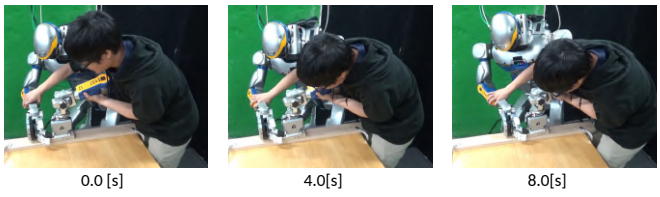


Fig. 10: Teaching force adjustment of kanna shaving motion by kinesthetic teaching.

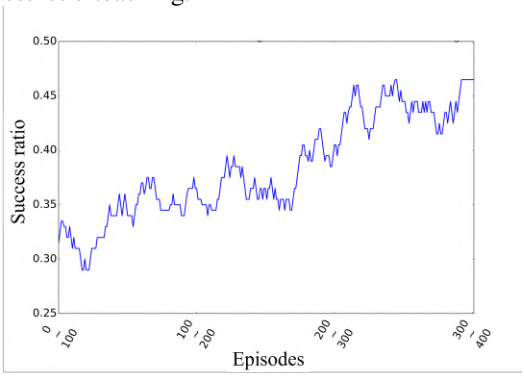


Fig. 11: Graph above shows transition of success ratio of kanna shaving motion in average over 100 episodes.

2) Experimental result: Fig. 11 is a graph showing a moving average of the success ratio. Since the reward is +1 for success and -1 for failure, the success ratio can be easily calculated from the reward. During training for 400 episodes, the robot requested teaching 32 times in total. From Fig. 11, the success ratio is gradually increasing, indicating that the robot mastered the skills to operate a Japanese planer.

D. Real-world Task: Building a Simple Bookshelf

We conducted experiment to make sure that the acquired skills are practically useful by making a small bookshelf using the tool operation skills acquired in the previous section. The completed view of bookshelf is shown in Fig. 12. As shown in Fig. 13, the bookshelf is made of (A) two square bars and (B) two flat plates. The steps for making the bookshelf are as follows:

- HRP2 chamfers parts (B) with a Japanese planer (Kanna).
- HPR2 cuts parts (A) off with a hacksaw to 45 degrees.
- A person assembles parts (A) and (B) with adhesive.

Fig. 14 shows a series of flows to build a small bookshelf using the acquired skills. This experiment showed that the acquired skills can be used practically.

Fig. 12: Completed view of bookshelf.

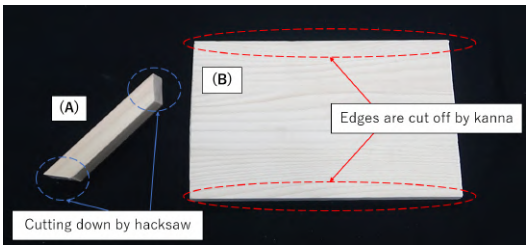


Fig. 13: (A) A rectangular lumber cut down by hacksaw. 2 of them are needed to complete bookshelf.
(B) A wooden plate whose edges are cut off by kanna. 2 of them are also needed.

V. Summary and Conclusions

In this study, we proposed a novel method for a life-sized humanoid robot to acquire tool manipulation skills that require complicated force adjustment. Various studies have been conducted on life-sized humanoid robots such as fall prevention and motion generation, and it is essential to benefit from these studies. We have proposed an integrated system that incorporates the latest deep reinforcement learning while utilizing these studies. The proposed integrated system consists of three components: off-line motion generation, high-frequency online control, and low-frequency learning components. In this study, by being benefited from the latest research in each field, we realized the acquisition of tool manipulation skills by a life-sized humanoid robot.

In addition, a learning algorithm with high sample efficiency is indispensable for reinforcement learning using a

Fig. 14: (a)(b) The HRP2 shaves and cuts off edge of wooden plate. (c)(d) The HRP2 cuts down rectangular lumber with the partner. (e) The partner assembles the parts. (f) The bookshelf is indeed useful.

real robot. In particular for a humanoid robot, the cost of running the robot is higher than normal manipulators. Hence, the sample efficiency is a critical issue. Despite this issue, a life-sized humanoid robot has the strong advantage that a person can teach the robot the skills just as that person would teach another person. In this study, we proposed a learning method called *Learning with Active Teaching Request* (LATR), in which a robot determines whether it needs teaching based on the transition of rewards and learns tool manipulation skills while teaching interactively. Furthermore, using the integrated system incorporating LATR, we realized the acquisition of hacksaw manipulation skills and Japanese planer manipulation skills by the life-sized humanoid robot. In addition, we demonstrated through a practical experiment, the bookshelf making task, that the acquired skills were indeed practical.

In future work, instead of creating the motions of the robot heuristically, we would like to create them by observing how a person operates a tool. Also, considering that it is actually costly for humans to teach, as an extension of LATR, we would like to investigate a learning method that considers the cost of human teaching.

REFERENCES

[1] Petar Kormushev, Sylvain Calinon, and Darwin G Caldwell. Robot motor skill coordination with em-based reinforcement learning. In *2010 IEEE/RSJ international conference on intelligent robots and systems*, pp. 3232–3237. IEEE, 2010.

[2] Jens Kober and Jan Peters. Learning motor primitives for robotics. In *2009 IEEE International Conference on Robotics and Automation*, pp. 2112–2118. IEEE, 2009.

[3] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, and Sergey Levine. Scalable deep reinforcement learning for vision-based robotic manipulation. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, Vol. 87 of *Proceedings of Machine Learning Research*, pp. 651–673. PMLR, 29–31 Oct 2018.

[4] OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Józefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, Vol. 39, No. 1, pp. 3–20, 2020.

[5] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 3389–3396. IEEE, 2017.

[6] Tim Brys, Anna Harutyunyan, Halit Bener Suay, Sonia Chernova, Matthew E Taylor, and Ann Nowé. Reinforcement learning from demonstration through shaping. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[7] Jessica Chemali and Alessandro Lazaric. Direct policy iteration with demonstrations. In Qiang Yang and Michael J. Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pp. 3380–3386. AAAI Press, 2015.

[8] Ionel-Alexandru Hosu and Traian Rebedea. Playing atari games with deep reinforcement learning and human checkpoint replay. *ECAI Workshop onEvaluating General Purpose AI*, 2016.

[9] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. In *Robotics: Science and Systems XIV*, 2018.

[10] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, et al. Deep q-learning from demonstrations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A Rusu, J. Veness, M. G Bellemare, A. Graves, M. Riedmiller, A. K Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, Vol. 518, No. 7540, p. 529, 2015.

[12] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[13] A. L. Thomaz, C. Breazeal, et al. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *Aaai*, Vol. 6, pp. 1000–1005. Boston, MA, 2006.

[14] Christian Daniel, Malte Viering, Jan Metz, Oliver Kroemer, and Jan Peters. Active reward learning. In *Robotics: Science and systems*, 2014.

[15] Si-An Chen, Voot Tangkaratt, Hsuan-Tien Lin, and Masashi Sugiyama. Active deep q-learning with demonstration. *Machine Learning*, pp. 1–27, 2019.

[16] Riku Shigematsu, Masaki Murooka, Yohei Kakiuchi, Kei Okada, and Masayuki Inaba. Generating a key pose sequence based on kinematics and statics optimization for manipulating a heavy object by a humanoid robot. In *IROS2019*, pp. 3852–3859, nov 2019.

[17] S. Nozawa, Y. Kakiuchi, K. Okada, and M. Inaba. Controlling the planar motion of a heavy object by pushing with a humanoid robot using dual-arm force control. In *2012 IEEE International Conference on Robotics and Automation*, pp. 1428–1435, May 2012.

[18] Kenji Doya, Kazuyuki Samejima, Ken-ichi Katagiri, and Mitsuo Kawato. Multiple model-based reinforcement learning. *Neural computation*, Vol. 14, No. 6, pp. 1347–1369, 2002.

[19] Lennart Ljung. System identification. *Wiley Encyclopedia of Electrical and Electronics Engineering*, pp. 1–19, 1999.

[20] K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, and T. Isozumi. Humanoid robot hrp-2. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, Vol. 2, pp. 1083–1090 Vol.2, April 2004.

[21] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[23] M. Lin, Q. Chen, and S. Yan. Network in network. *ICRL*, 2014.