Learning Tone: Towards Robotic Xylophone Mastery

Jiawei Zhang[1], Taemoon Jeong[2], Sankalp Yamsani[1], Sungjoon Choi[2], and Joohyung Kim[1]

***Abstract*— Audio information plays an important role in various robotic manipulation tasks, such as pouring and music performance: the produced sound can be the informative indicator for evaluating actions. However, it is rarely explored in reinforcement learning methods, due to the unique nature of audio information that it is challenging to simulate in a simulator or use it as a direct feedback. Therefore, in this paper, we propose a reinforcement learning method based on audio feedback, attempting to train a dexterous hand to play the xylophone in the real world. By optimizing the dexterous hand's actions using the produced sound, we are able to make the characteristics of the sound—such as amplitude, waveform shape, and timing—similar to human performance.**

I. INTRODUCTION

Deep Reinforcement Learning (DRL) has been proven to play a crucial role in the field of robotic control, especially in bipedal, quadrupedal, and dexterous manipulation in recent years. For successful deploying reinforcement learning on robots, researchers have explored several modalities of data to help robots make decisions. However, compared to the variety of information that humans perceive when making decisions, the range of modalities available to robots for learning is still limited.

Dexterous manipulation based on visual and tactile information has been widely explored to perform various contact-rich tasks such as object relocation, in-hand manipulation and so on [1]–[3]. However, audio information is rarely utilized, yet in many tasks, audio information plays a crucial role. For example, when pouring water into an opaque cup, humans can judge the progress by the change in pitch. Especially in music performance, humans directly assess the quality of the performance through the sound produced by the instrument.

We conclude the difficulty of applying audio information to the following reasons: 1. Audio information is challenging to simulate in simulation environments, making it difficult to evaluate the robot actions without running tests in the real world. 2. Audio information conveys indirect information towards robot actions such as loudness and frequency, compared to visual information [4], [5] which can directly allow the robot to perceive object positioning, contact, and tactile information. Therefore, in the task of audio based dexterous manipulation, learning-based methods are rarely used.

Considering the difficulties, in this work, we explored the potential of doing reinforcement learning in the real world via audio feedback. The goal of this paper is to enable a dexterous hand to strike the xylophone in a way that produces sounds similar to those made by human. This requires the learned actions to achieve clarity, volume, and timing close to those produced by human strikes.

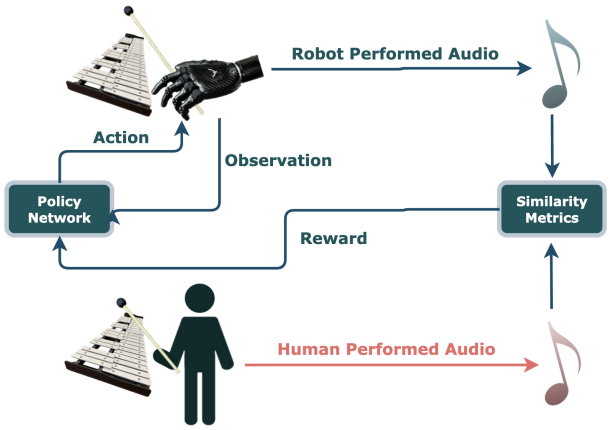


Fig. 1. Workflow of proposed method. The dexterous hand is controlled by a policy network to play the xylophone. The generated sound will be collected and then compared with desired human performed sound. The comparison of some audio features are then used as rewards to refine the policy network.

As shown in Figure 1, a human first demonstrates a desired audio. The dexterous hand plays the xylophone to generate a robot performed audio. The robot performed audio is then compared with the desired human performed audio. The similarity of audio data and the state of the dexterous hand, serving as rewards and observations respectively, will be used for model-free reinforcement learning that will update the action policy for the dexterous hand to generate good sound. The designed reward function is based on some key similarities of features that are more effective for optimizing robot actions: including amplitude, onset timings, and waveform shape.

The remainder of this paper is structured as follows: Section II discusses background on robotic dexterous manipulation, robot music performance, using sound to train a policy. Next Section III describes the environmental setup and Section IV details the algorithm design of our work, including how we design the reward function to produce good sound based on audio information and the key designs of the training algorithm. Finally Section V presents training results and compares the learned performance strategy with the demonstrated human performance.

[1]Jiawei Zhang, Sankalp Yamsani, Joohyung Kim are with the University of Illinois Urbana-Champaign, Urbana, 61801 IL, USA. [2]Taemoon Jeong and Sungjoon Choi are with the with the Department of Artificial Intelligence, Korea University, Seoul, Korea. Author contact information is {jiaweiz9, joohyung}@illinois.edu.

II. Related Work

A. Model-based and Learning-based Methods for Dexterous Manipulation

Before Deep Reinforcement Learning (DRL) prevailed in dexterous manipulation, some model-based methods have achieved significant results in multi-fingered tasks, especially in that manipulating rigid objects such as spheres and polygonal objects in hand grasping [6], [7]. There has also been more simulation work that manipulates cloth [8] into desired motions. Advancements in machine learning have enabled the use of learning-based methods for handling complex tasks that are challenging to model explicitly. These include learning to grasp unknown objects, spinning a pen, and solving a Rubik's cube [9]–[11]. However, despite the impressive achievements these frameworks make use of visual and tactile information in learning-based frameworks. There have been few attempts to integrate audio feedback effectively. This is the challenge that this paper aims to explore.

B. Musical Performance Robots

Musical robots have been an interest in robotics. Previous works include designing a new mechanism to help adapt to play instruments such as a drum and piano [12], [13]. There have been further attempts at learning to play the musical notes of songs on a piano with dexterous hands as well [14]. While these robotic systems can physically manipulate and play various musical instruments, they cannot often utilize crucial auditory feedback to truly learn instrument performance. These systems typically employ more simplistic contact-based or binary representations to generate the desired musical output.

In our previous work [15], we utilized dexterous hands to mimic the tapping motion of a finger on a drum through the use of auditory feedback. The drum's waveform is characterized by a quick spike, rather than a sustained, "carrying" signal. In the current study, we aim to investigate training models on these "carrying" waveforms, as well as utilizing tools for generating high-quality sounds.

III. Environmental Setup

In this section we discuss our environments setup for experiments. We fist demonstrate the equipment required for experiments and the workspace, and then introduce the two ways in which the dexterous hand holds the mallet.

A. Workspace

Fig. 2 shows the aerial view of the workspace depicting the systems utilized in this paper. The environment includes a Plug and Play Robotic Arm System (PAPRAS) [16] a pluggable 6 Degree of Freedom (DoF) general purpose manipulator, a dexterous 6 DOF gripper, an audio recorder, a xylophone mallet, and a xylophone. To maintain consistency in the training environment, the following factors were fixed: grasp position of the mallet, mallet strike zone, and distance between the recorder and xylophone. Due to the limited workspace of the dexterous gripper, the mallet has been attached to the thumb, which restricts the mallet's freedom of movement and allows for better control.

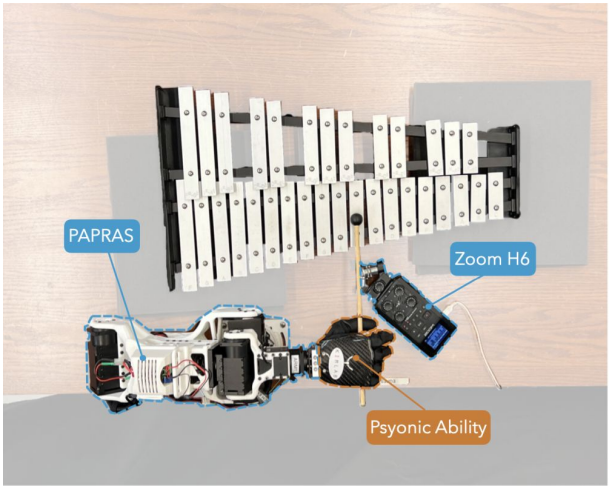


Fig. 2. Environmental Setup: we utilize a 6 DoF manipulator, 6 DoF dexterous hand, an audio recorder, and a xylophone attached to a table setting.

B. Mallet Holding Methods

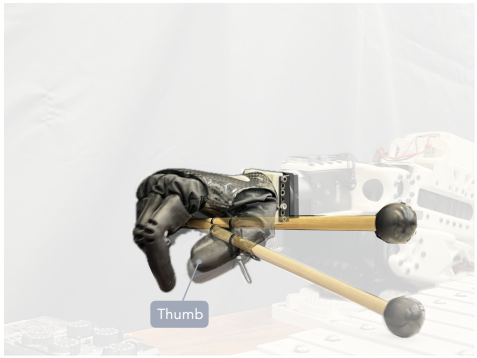
When humans play the xylophone, there are many different gripping techniques, each involving a complex combination of finger, wrist, and arm movements. Due to the underactuated nature of the dexterous hands and the large action space to explore, it is extremely challenging to simulate actions that are completely consistent with human striking motions. Therefore, we use simplified striking action models in this study: 1. **Thumb-Control** model which involves only thumb movements, and 2. **Thumb-Wrist-Control** model which involves the coordination between thumb and wrist.

As shown in Fig. 3a, our thumb-control model is to control the mallet using thumb only. As shown in the Fig. 3b, our thumb-wrist-control model is to control the mallet by using thumb and wrist at the same time. The thumb rotation serves as the primary driving mechanism to strike the key with high speed, while the wrist generates slow rotation to extend the striking force range. We used position control for both the thumb and the wrist.

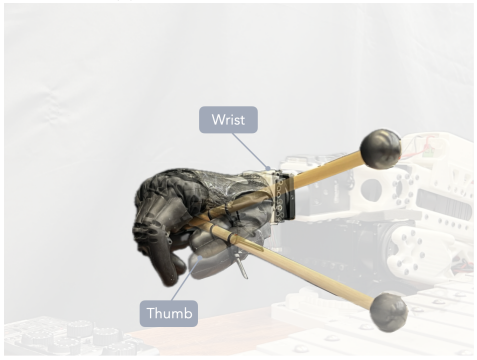
We will control the position of the fingers or the fingers and wrist at a frequency of 30Hz, meaning that a position control command is sent every 0.033 seconds.

IV. Algorithm

We employ Proximal Policy Optimization (PPO) for model-free reinforcement learning in the real world. The main components of the PPO algorithm include the observation space O, the action space A, and the reward function R. The design of the observation and action spaces defines the problem's scope, which impacts whether reinforcement learning can successfully converge to the optimum. Meanwhile, the design of the reward function is crucial for effective convergence to the optimum. Below, we define our



(a) Thumb-Control Model



(b) Thumb-Wrist-Control Model

Fig. 3. Two Mallet Holding Methods

observation space, action space, and reward function design, and provide the rationale for each design decision.

A. Problem Formulation

The problem can be summarized as follows: While executing a sequence of control actions $a(t)$, record the produced audio $d_{\text{rec}}(t)$ so that the resulting audio sequence d_{rec} closely matches the reference audio d_{ref}. Here, the d_{rec} d_{ref} are 2-sec audio data sampled at 44,100 Hz. Because the tail of sound produced by the xylophone exists, we only execute the control actions during the first 1.5 seconds, then remain stationary, waiting until the full 2-sec audio is collected.

B. Observation Space and Action Space

1) Thumb-Control Model: In the thumb control model, only the thumb can move. The observation space includes the current thumb joint angle q_t and the joint angle at the previous time step q_{t-1}. Additionally, we include the time step information because in practice, we found that without it, the actions tend to get stuck in a stationary point. To ensure that the length of the time step vector is consistent, we use a sine function to encode the time steps which are normalized by the length of episode L. The observation at time step t of the thumb-control model can be written as:

$$o_t = (q_t + \sin\left(2\pi\frac{t}{L}\right), q_{t-1} + \cos\left(2\pi\frac{t}{L}\right)) \tag{1}$$

Because the Psyonic Ability Hand is under-actuated for small angle movements at certain angles, we discretize the output of the policy network into 6 values that can always be executed reliably, i.e., $a = \text{Discrete}(6)$. This also improves the exploration efficiency of the policy network. Each output value corresponds to a specific joint movement angle (in degree) as follows:

a	0	1	2	3	4	5
Δq	-20	-10	0	5	10	20

TABLE I. Correspondence between thumb joint movement angles and outputs of policy network in thumb-control model

2) Thumb-Wrist-Control Model: In the thumb-wrist-control model, the observation space includes the current angle of the thumb q_t, the previous angle of the thumb q_{t-1}, the current angle of the wrist w_t, and the previous angle of the wrist w_{t-1}. We also use the same time-step encoding method as in the thumb-control model. The observation at time step t for the thumb-wrist control model can be written as:

$$\begin{aligned} o_t = (q_t + \sin\left(2\pi\frac{t}{L}\right), q_{t-1} + \cos\left(2\pi\frac{t}{L}\right), \\ w_t + \sin\left(2\pi\frac{t}{L}\right), w_{t-1} + \cos\left(2\pi\frac{t}{L}\right)) \end{aligned} \tag{2}$$

The action space of thumb-wrist-control model is a multi-discrete space, where the thumb action has 6 options and the wrist action has 7 options. The correspondence between raw outputs of policy network and angles of movement of thumb joint (in degree) or wrist joint (in radiant) are summarized in followeing tables.

a_{thumb}	0	1	2	3	4	5
Δq	-20	-10	0	5	10	20

a_{wrist}	0	1	2	3	4	5	6
$\Delta w \times 10^{-2}$	-7.5	-3.75	-1.875	0	1.875	3.75	7.5

TABLE II. Correspondence between thumb and wrist joint movement angles and outputs of policy network in thumb-wrist-control model

C. Reward Function Designs

For effective reinforcement learning training, the design of the reward function is crucial. In the task of striking xylophone, one challenge is that we can only calculate the reward by comparing the collected audio with the reference audio after the striking action is fully completed. This leads to the well-known sparse reward problem, which hinders the convergence. A typical solution to this problem is reward shaping, where meaningful rewards are added at each step through manual design. This section will introduce our designed **Smoothed Step Reward** and **End of Episode Reward**.

1) Smoothed Step Reward: A good step-wise shaped reward should evaluate each step effectively without conflicting with the final evaluation on the whole episode. To achieve this, we first introduce the *Smoothed Step Reward*. At the end of each step, we first calculate the difference between the

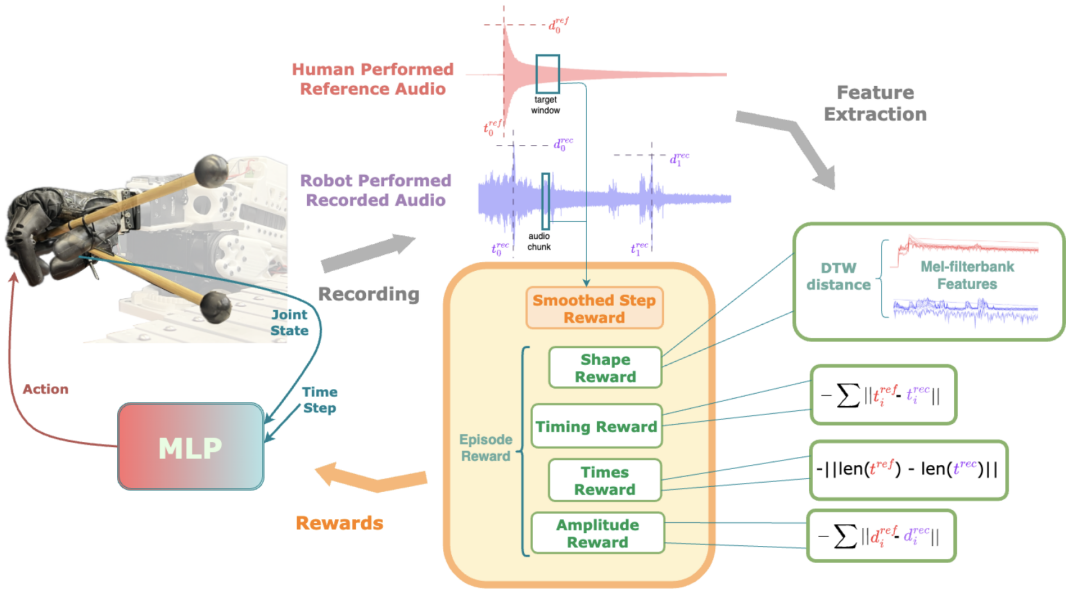


Fig. 4. An illustration of the algorithm. The actions of dexterous hand are controlled by a MLP, whose inputs (observations) includes the joint states and the current time step, and the outputs (actions) are the position commands for controlled joints. The generated audio will be recorded as waveform format. We extract several key features from both recorded audio and reference audio including **striking happened time (timing)** t_s^{ref}, t_s^{rec}; **striking amplitude (amplitude)** d_s^{ref}, d_s^{rec}; and **Mel-filterbank features (shape)** to compute different reward functions.

sound produced in this step and the sound from the previous step. If the amplitude difference is greater than 0.1, it is approximately considered a striking event. We then obtain a *sliding window* W of the reference audio corresponding to the current time step. If the striking occurs within this window, we give a positive reward; if it occurs outside the window, we give a negative penalty. In other cases, we simply calculate the amplitude difference between the recorded audio and the reference audio at the current time step as a penalty. The environment will first It can be formulated as follows:

$$R_{step}(t) = \begin{cases} 5 \times \beta(t^{\text{ref}}, t^{\text{rec}}), \text{if striking in } W; \\ -1, \text{if striking out of } W; \\ -|mean(d_t^{rec}) - mean(d_t^{ref})|, \text{else.} \end{cases} \quad (3)$$

Here, we use d_t^{rec} to represent the recorded audio data at time-step t, while d_t^{ref} to represent the reference audio data at time-step t. The function β is a linear interpolation that maps the distance of t^{rec} and t^{ref} into $(0.5, 1)$.

Another step-wise reward we introduce is the "step-wise action penalty," which applies a heuristic, minor penalty to each movement. It can be formulated as follows:

$$R_{step\_move}(t) = -|a(t) - a(t-1)| \quad (4)$$

where $a(t)$ represents the action taken at time-step t.

There are two benefits to this approach: As training progresses, the number of strikes can effectively decrease until convergence. It can effectively reduce unnecessary up-and-down jitter, making the striking actions more natural.

2) End of Episode Reward: The End of Episode Reward is used to evaluate the entire episode after it ends, aiming to provide higher rewards for episodes that produce better sounds. Since our goal is to generate good sound, the evaluation metrics should reflect the characteristics of good sound. The features we select include:

1. *Maximum Amplitude*. Each strike event should cause a similar maximum amplitude to the corresponding amplitude in the reference audio. The definition is formulated as:

$$R_{amp} = 1 - \sum \frac{|d_{s_i}^{rec} - d_{s_i}^{ref}|}{d_{s_i}^{ref}} \quad (5)$$

Here, $d_{s_i}^{rec}$ and $d_{s_i}^{ref}$ represents recorded audio and reference audio respectively.

2. *Timing*. The strikes should occur at time close to the expected time within the entire recording period, allowing the sound's tail to continue for a duration in the recorded audio.

To calculate the strike time through the amplitude waveform, we adapted the method from [17]. We first obtain the spectral flux onset strength envelope of the audio based on the log-power Mel spectrogram computed from the waveform [18]. By adjusting parameters, we can filter out those time that are considered to represent a strike event. The timing reward is then computed by

$$R_{timing} = 1 - \sum \frac{|t_{s_i}^{rec} - t_{s_i}^{ref}|}{2} \quad (6)$$

3. *Waveform shape*. A good waveform should have a gradually decreasing long-tail shape. The waveform shape of the produced sound should be similar to the waveform shape of expected reference audio, which can be regarded as having similar frequency components. Here, we first convert the two

pieces of audio into mel-filterbank spectrums, M^{rec} and M^{ref}. The Mel-filterbank can analyze the frequency components of audio and has a representation similar to human auditory perception. Then we adapt Dynamic Time Warping (DTW) algorithm [19] to compute the similarity between the two mel-filterbank spectrum series:

$$R_{shape} = DTW(M^{rec}, M^{ref}) \tag{7}$$

4. *Hitting times*. Although we have the shaped step movement penalty to indirectly reduce the hitting times, an explicit reward for hitting times is still necessary, because it will give the policy a stable signal to hit only once. The reward is defined as:

$$R_{times} = -\min\{|len(t_s^{rec}) - len(t_s^{ref})|, 20\} \tag{8}$$

D. Proximal Policy Optimization

We use PPO-Clip as the model-free reinforcement learning algorithm for training. PPO (Proximal Policy Optimization) [20] is an on-policy Actor-Critic algorithm, meaning it includes a policy network for sampling actions and a value network for evaluating the policy. Compared to vanilla Actor-Critic, the main feature of PPO is that it constraints the update of policy network not too far from the old one, ensuring that each update remains within a confidence interval. The loss function for the policy network of PPO-Clip can be written as:

$$\begin{aligned} L^{\text{CLIP}}(\theta) = \mathbb{E}_{s,a\sim\tau_{\theta_{\text{old}}}}[\min\{&r_\theta A_{\theta_{\text{old}}}(s,a), \\ &\text{clip}(r_\theta, 1-\varepsilon, 1+\varepsilon)A_{\theta_{\text{old}}}(s,a)\}] \end{aligned} \tag{9}$$

where

$$r_\theta = \frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}$$

is the probability ratio between the new and old policies, $A_{\theta_{\text{old}}}(s,a)$ is the generalized advantage estimation [21] for the old policy.

The ε controls the clipping range, ensuring the r_θ stays between $1-\varepsilon$ and $1+\varepsilon$. This ensures that the change in action probability resulting from each update does not exceed a certain trustworthy range, which has been proven to help maintain a degree of monotonic performance improvement.

In practice, we also introduce the entropy regularization to ensure a certain level of exploration during the later stages of training.

V. Experiments

We conducted various experiments using the proposed method in both the thumb-control and thumb-wrist-control models. These include imitating high amplitude (approximately 0.6) and low amplitude (approximately 0.45) in the both thumb-control and thumb-wrist-control model, and imitating two consecutive high amplitude strikes in the thumb-wrist-control model.

Parameters	Values
number of MLP layers	2
number of units per layer	64
number of epochs	20
number of steps per update	1000
γ	0.99
GAE λ	0.95
learning rate	0.03
entropy coefficient	0.01
weight of smoothed step	0.1
weight of amplitude	10/30
weight of shape	0.5
weight of timing	10
weight of times	10
weight of movement	0.005

TABLE III. Experiment Settings. Except for the double-hit experiment where the weight of amplitude is 30, it is 10 for all other experiments.

A. Experiment Settings

We used the PPO implementation from stable-baseline3 for all our experiments. Here, we list the hyper-parameters for both PPO and reward functions.

B. Thumb-Control Model

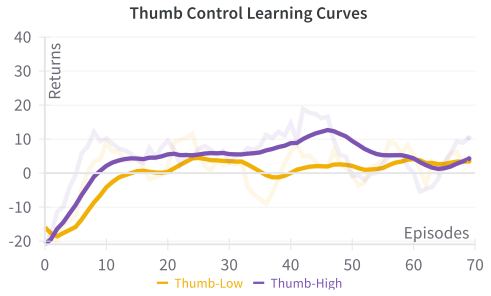
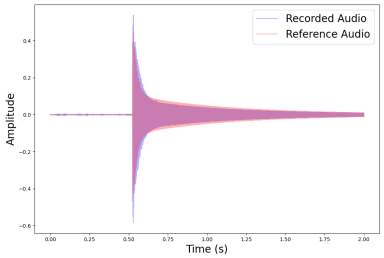
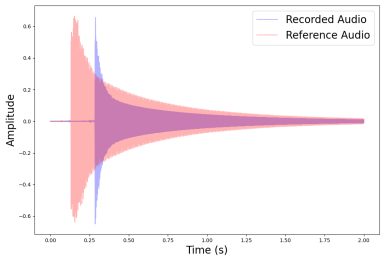


Fig. 5. Learning Curve of Thumb Control

As shown in the Fig 5, the training reward curve under thumb control indicates that the policy network converges towards higher returns in the initial stage but fluctuates largely around 10 in the later stage. We found that this is because the dexterous hand's thumb pitch joint is difficult to control accurately, and performing single strike is quite likely to not performing any strike, thus even when sampling with nearly identical policies, it can result in two completely different outcomes: either striking once or not striking at all. This leads to the fluctuation in the training curve in the later stages and makes it difficult for the rewards to increase further. We show the learned results for the low amplitude and high amplitude reference audio under thumb control in Fig 6, w. Our two reference audio have different amplitudes (0.45 and 0.63) and different striking times (0.5 and 0.25), which can effectively demonstrate the effectiveness of our amplitude reward and timing reward. As can be seen from the figure, our proposed method can effectively learn the features of the reference audio. A noteworthy thing is that, in the high amplitude experiment, despite achieving similar



(a) Result of Low Amplitude



(b) Result of High Amplitude

Fig. 6. Waveform Comparison on Thumb Control

amplitude and timing, the shape of produced waveform still be obviously different. This might be due to the target striking in this experiment exceeding the capability of the Ability hand when only the thumb is actuated.

C. Thumb-Wrist-Control Model

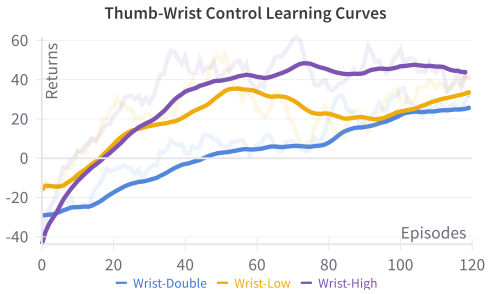
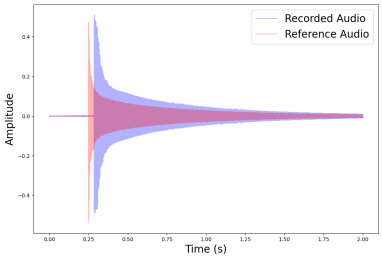
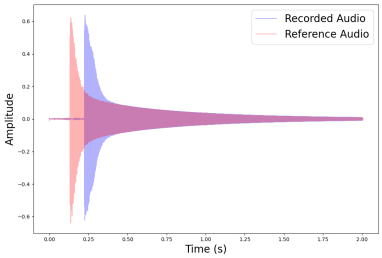


Fig. 7. Learning Curve of Wrist-Thumb Control

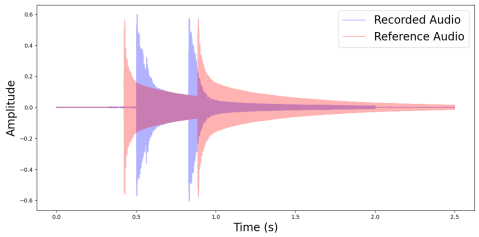
From the learning curve of the thumb-wrist control, we can see that the proposed method effectively enables the policy network to learn actions that produce higher rewards. The smoothed reward curve can converge above 30. Compared to the thumb control model, although the dimensions of observation and action spaces increase, the final converged reward is higher than that of the thumb control model. During training, we found that one reason for this is the higher control precision of the wrist motor, which results in better consistency in policy execution. Therefore, it is less likely to cause the fluctuations between striking once and no striking in the thumb control model.



(a) Result of Low Amplitude



(b) Result of High Amplitudes



(c) Result of Double High Amplitudes

Fig. 8. Waveform Comparison on Thumb Control

Under the thumb-wrist control model, we conducted experiments on imitating reference audio with high amplitude (approximately 0.6), low amplitude (approximately 0.45), and the more challenging double-hit (approximately 0.58). We presented the differences between the waveforms of the audio produced by the learned policies and the reference audio, as shown in the figure. It can be seen that our proposed method can learn the key features of the reference audio under thumb-wrist control. One notable thing is that in the low amplitude experiment, the waveform produced by the learned policy is wider than that of the reference audio. We believe this might because the dexterous hand cannot leave the keys quickly after striking, unlike a human hand.

D. Performing Music with Learned Skills

To showcase the learned striking skills, the system performs the simple yet well-known music ”Twinkle Twinkle Little Star”, utilizing the PAPRAS arm to move to the corresponding keys through predefined motions to the corresponding notes: C, D, E, F, G, A, and B. The arm movements and strikings are executed sequentially at a constant 2-second interval. Please refer to the submitted video to enjoy the music played by the robot.

VI. Conclusion

In this work, we explored the use of audio feedback and reinforcement learning to guide a dexterous hand in performing the task of playing the xylophone. By selecting certain features from the audio information, we can enable the dexterous hand to gradually learn to produce sounds similar to those played by humans. This demonstrates the potential for further applying audio information in robot learning.

To have better control, we used simplified movement patterns and attached mallet to thumb. A future direction could involve using audio information to guide robots in performing more complex actions, e.g. performing music without any constraints, which may require researchers to investigate whether sound generation can be better simulated or how to improve training efficiency in real-world.

References

[1] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *arXiv preprint arXiv:1709.10087*, 2017.

[2] H. Zhu, A. Gupta, A. Rajeswaran, S. Levine, and V. Kumar, "Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3651–3657.

[3] T. Chen, J. Xu, and P. Agrawal, "A system for general in-hand object re-orientation," in *Conference on Robot Learning*. PMLR, 2022, pp. 297–307.

[4] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray *et al.*, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.

[5] H. Van Hoof, T. Hermans, G. Neumann, and J. Peters, "Learning robot in-hand manipulation with tactile features," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2015, pp. 121–127.

[6] Y. Bai and C. K. Liu, "Dexterous manipulation using both palm and fingers," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 1560–1565.

[7] A. Bicchi and R. Sorrentino, "Dexterous manipulation through rolling," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 1, 1995, pp. 452–457 vol.1.

[8] Y. Bai, W. Yu, and C. K. Liu, "Dexterous manipulation of cloth," in *Computer Graphics Forum*, vol. 35, no. 2. Wiley Online Library, 2016, pp. 523–532.

[9] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations," in *Proceedings of Robotics: Science and Systems (RSS)*, 2018.

[10] Y. J. Ma, W. Liang, G. Wang, D.-A. Huang, O. Bastani, D. Jayaraman, Y. Zhu, L. Fan, and A. Anandkumar, "Eureka: Human-level reward design via coding large language models," 2024. [Online]. Available: https://arxiv.org/abs/2310.12931

[11] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang, "Solving rubik's cube with a robot hand," 2019. [Online]. Available: https://arxiv.org/abs/1910.07113

[12] J. Cho and S. Yim, "Wrist snap for humanoid robot drumming," in *2023 20th International Conference on Ubiquitous Robots (UR)*, 2023, pp. 753–758.

[13] Y.-F. Li and L.-L. Chuang, "Controller design for music playing robot — applied to the anthropomorphic piano robot," in *2013 IEEE 10th International Conference on Power Electronics and Drive Systems (PEDS)*, 2013, pp. 968–973.

[14] K. Zakka, P. Wu, L. Smith, N. Gileadi, T. Howell, X. B. Peng, S. Singh, Y. Tassa, P. Florence, A. Zeng, and P. Abbeel, "Robopianist: Dexterous piano playing with deep reinforcement learning," in *Conference on Robot Learning (CoRL)*, 2023.

[15] T. Jeong, S. Yamsani, J. Hong, K. Park, J. Kim, and S. Choi, "Generating realistic sound with prosthetic hand: A reinforcement learning approach," 2024, (accepted).

[16] J. Kim, D. C. Mathur, K. Shin, and S. Taylor, "Papras: Plug-and-play robotic arm system," 2023. [Online]. Available: https://arxiv.org/abs/2302.09655

[17] S. Böck and G. Widmer, "Maximum filter vibrato suppression for onset detection," in *Proc. of the 16th Int. Conf. on Digital Audio Effects (DAFx). Maynooth, Ireland (Sept 2013)*, vol. 7, 2013, p. 4.

[18] P. Pedersen, "The mel scale," *Journal of Music Theory*, vol. 9, no. 2, pp. 295–308, 1965. [Online]. Available: http://www.jstor.org/stable/843164

[19] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intell. Data Anal.*, vol. 11, no. 5, p. 561–580, oct 2007.

[20] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[21] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.