Coordinating ROS-based Robot Control Demonstrations with the Video Playback to Enhance E-learning

Kouhei Takimoto[1,a)] Katsuya Matsubara[1,b)] Sho'ji Suzuki[1,c)]

Abstract:
ROS has become the de-fact robotic software platform because of plenty of useful libraries and tools for robotics developers. Accordingly, the growing demand for learning robot programming with ROS rapidly increases. The skill of robot programming that should be learned consists of not only knowledge of functionalities of tools and the library APIs but also the practical technique of robot control and the latter would be acquired through enough experiences, which will be accumulated with trial and error repeatedly. Unfortunately, most of the legacy e-Learning platforms for robot programming could be far from satisfying learners in seeing realistic reactions of a robot from the mixed results of issued control commands and interference from the real-world situation. One of the main reasons is that interference from the real-world situation cannot be realized perfectly in the virtual world, and the motivation for continuous trial and error, which must be necessary to learn the practical techniques of robot control, may be lost in simulation-based e-Learning. To improve the learning of robot programming, we propose an enhanced e-Learning platform that allows the coordination between a ROS robot and video content, which means an actual robot work synchronously with the video playback. This paper describes a method of implementing a mechanism for working robots by syncing the video playback that allows simple programs to be embedded in the video and the results of an evaluation of the practicality of the proposed system. And our finding from user evaluations has been that the mechanism that makes the actual ROS robot coordinate with the playback of an e-Learning video can give effectiveness for learning robot control programming.

Keywords: ROS, e-Learning, robot control programming

1. Introduction

Robot Operating System (ROS)[1], an OSS robot control middleware, has been increasingly adopted in many robot systems because of providing rich-functional tools, reusable modules and drivers, and a flexible and easy-to-constructible programming model[2]. Consequently, robot industries critically demand engineers who have been trained in ROS. Nowadays, we can find many materials and platforms to learn robot controls with ROS programming[3]. Compared to the standard programming for computations and data processing, robot programming requires not only an understanding of language syntaxes, library APIs, and also physical reactions of robot control, such as how the acceleration ratio, equilibrium, and tire slip while a robot moves. Unfortunately, text and video materials, often used for self-study, could not be enough to imagine and understand the physical behaviors of a robot while learning.

Besides, the suppression of face-to-face lectures and practices during the COVID-19 pandemic boosts demand for e-Learning. The e-Learning allows making the quality of lessons to be uniform without affecting instructors' skill and condition and to study at any time and place repeatedly. In contrast, e-Learning has some issues in that it may be hard to keep the motivation to continue learning, and it may be unsuited for practical training such as robot controls. So, most of the current e-Learning materials for robot control programming explain robot behavior using videos of robots in action or virtual robots reproduced as 3DCG. In terms of understanding robot behaviors in the real world and maintaining the motivation for learning and as mentioned above, we claim that the videos and 3DCG could be unsatisfied.

To improve the e-Learning of robot programming, we propose a coordination mechanism between the physical robot and the video stream, which means an actual robot work synchronously with the video playback. Fig. 1 illustrates the overview of our system. A learner can watch a video of robot programming through the Web application of the system. And specified timings during the video playback can automatically trigger appropriate robot control programs which link the video. Then a physical robot was placed beside would operate in synchronization with the video. Consequently, the user can see that the program and robot are moving synchronously in the real world, while taking advantage of the advantages of e-Learning, such as the ability to repeat learning at any time and place, and the uniform quality of the teaching materials.

The main contributions of this paper are as follows.

- We will discuss the features and challenges of existing e-Learning systems for learning robot control programming and explain the advantages of the robot in person operating in synchronization with the explanations in lecture videos.
- We propose a new system in which learning is not completed only in the virtual world, but can obtain the benefits of both

[1] Future University Hakodate, Hakodate, Hokkaido 041–8655, Japan
[a)] g2121032@fun.ac.jp
[b)] matsu@fun.ac.jp
[c)] ssuzuki@fun.ac.jp

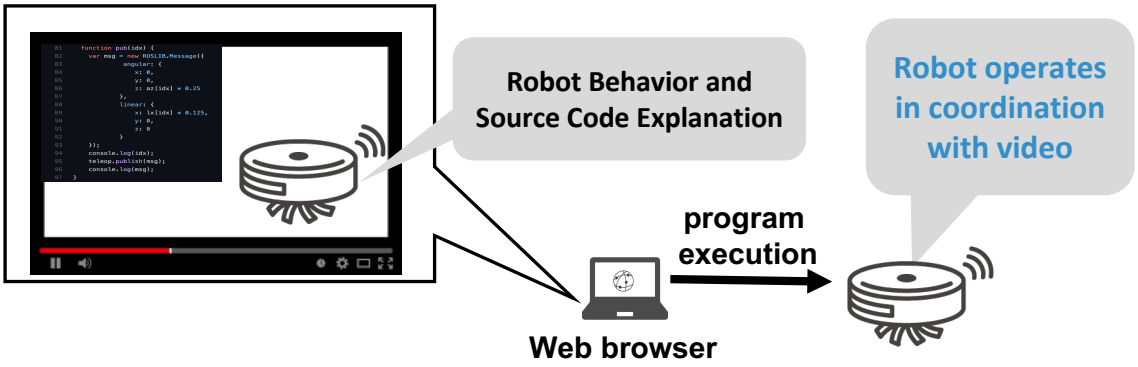

Fig. 1 Proposal System

by linking the contents in the virtual world with robots in the real world.

- We describe how to implement video-robot coordination using a framework for embedding a program into a video.
- We demonstrated the practicality of the proposed system based on the results of latency measurement until the video and the robot synchronize, and the consequences of experiments to see the effects of the proposed approach when used for learning.

2. Issues with Existing E-Learning Systems

This section introduces four existing e-Learning systems for learning robot control programming and discusses issues common to all of them.

2.1 Existing E-Learning Systems

Udemy[4] offers video content that explains how to program robot controls and demonstrates the robot's behavior. Udemy is an e-Learning System that allows users to learn using slides that explain specialized knowledge about robots and videos that record users writing source code while explaining it.

The Construct[5] is a platform that provides learning content that can see the robot's behavior through a simulator or a camera. The Construct allows users to view source code and texts explaining specialized knowledge while describing control commands and source code in a virtual development environment provided on a browser to run the robot on a simulator. The robot can also be operated through the camera by reserving the authority to operate the actual robot owned by The Construct.

A study that supports learning robot control programming with ROS online is this study by Gustavo et al. This study proposed a system that displays simulator operations and output results on a Web application by writing ROS source code on the Web application and executing the program. The proposed approach enables learners to learn robot control programming in a remote environment without having to build a development environment, compile a program, and so on.

As stated above, research and development of content to support online robot control programming learning have been conducted. However, there has been no work on a learning support system for robot control programming using ROS that combines the advantages of online learning with the knowledge gained from using actual robots.

2.2 Issues

Existing e-Learning systems for robot control programming use videos and simulators to see the behavior of the physical device of the robot, but seeing the behavior of the robot using videos and simulators is an issue in that it is impossible to verify in person that the program and the robot are moving in sync in the real world. There are three advantages to seeing the robot's behavior in person: understanding the difference between the virtual and physical worlds, the fun of seeing an actual object move, and trial and error by touching the robot and the surrounding environment.

Regarding the first advantage, when a robot is operated in a virtual world such as a simulator, its behavior is not affected by the surrounding environment, such as friction with the ground or air resistance, and it behaves exactly as instructed by the program. On the other hand, when the robot is operated in the real world, it is affected by the surrounding environment and does not always behave as instructed by the program. Therefore, it is necessary to create programs that consider the surrounding environment of the physical world. Still, it is not sufficient to move the robot into the physical world from the beginning of robot development. Using simulators in robot control programming has advantages, such as the ability to program executed simulations that require a large number of trials in parallel at a low cost, and to test programs that may harm the environment if the robot does not run correctly. Therefore, it is necessary to understand the difference between the virtual and physical worlds and to use them differently.

Regarding the second advantage, many of the current contents for learning robot control programming use simulators and the like to see the behavior of the program and the robot. However, because programming is a simple task and the robot on the simulator operates inorganically, and without sound, it does not provide a memorable experience. By running a robot assembled with a program created by the user, the user can gain a natural feeling of programming and enjoyment and can maintain motivation to continue learning.

Regarding the third advantage, in robot control programming, it is necessary to consider not only the program, but also the state of the robot, such as the effect on performance due to low battery power or component failure, and the impact of the robot's surrounding environment. On the other hand, this knowledge cannot

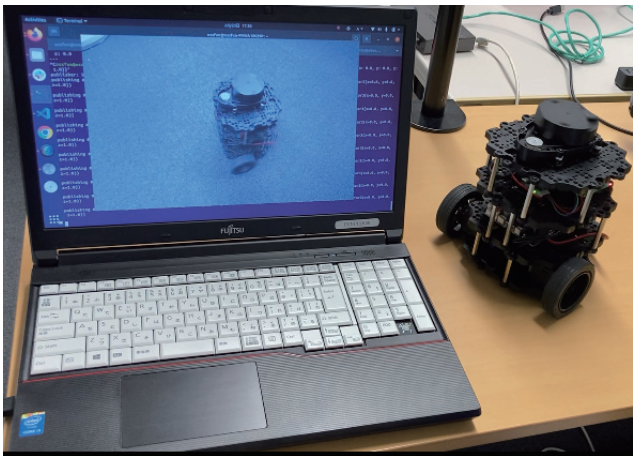

Fig. 2 Examples of System Usage

be acquired by learning only with simulators. In learning using an actual robot that is moving in person, students can the robot's movements. If the robot is not moving as programmed, it can gain knowledge through trial and error, such as re-programming the robot to take the surrounding environment into account or seeing for hardware causes.

These advantages can be obtained because of the actual robot, and it isn't easy to acquire them only in the virtual world. In the proposed section, we describe a learning system that can see the robot's behavior in person, while obtaining the advantages of e-Learning.

3. Video-Robot Coordination

This paper focuses on e-Learning for robot control programming, to improve motivation for learning and to implement learning with an awareness of the robot's behavior in the real world. Therefore, we propose a system that synchronizes the lecture video with the robot's explanation and supports the user's understanding of the program. An example of system usage is illustrated in Fig. 2. In Fig. 2, a learner is viewing a lecture video on robot control programming on a PC. When the robot in the video rotates, the actual robot in the learner's hand rotates in coordination with it.

3.1 System Architecture

The configuration of the proposed system is illustrated in Fig. 3. The proposed approach has three components: a Web application for viewing robot control programming learning videos, a PC for intermediating communication between the Web and the robot, and an actual robot for seeing the robot's behavior in person. The Web application implements a video viewer function, a program execution coordinated with the playback position function, and a playback state change handler function. The PC implements a protocol bridge function. Implementing these functions enables the robot to start learning without having to describe again the programs in the actual robot specifically for the proposed system.

3.2 Learning Video with a Robot Coordination

The contents used in the proposed system include a video

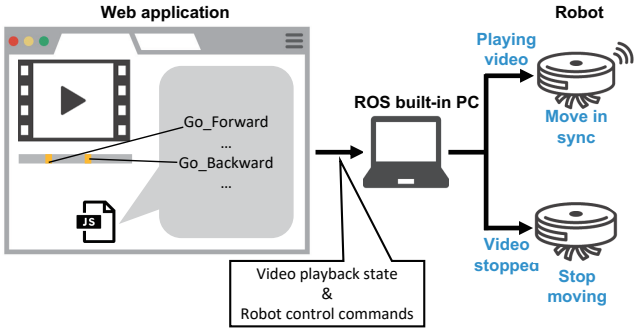

Fig. 3 System Architecture

watching the robot in action and an explanation of the program, and a script that specifies the playback position for the learner's robot to operate in synchronization with the video and describes the robot control commands. By embedding scripts in videos, it is possible to send robot control commands according to the playback position of the video. We will describe considering when creating content a script to stop the robot after the script to operate the robot, the correct period to execute control commands (to keep the actual robot operating), and does not describe control commands that operate multiple consecutive actions at a single playback position specified in a script as follows.

Regarding the first thing to consider, depending on the type of robot and the environment in which it is operated at the learner's hand, the robot may continue to refer to the script it operates on, which may cause it to behave unexpectedly. Therefore, it is necessary to accurately describe the robot from the start to the end of its operation.

Regarding the second thing to consider, the proposed system sends a stop control command to the robot when the playback state of a video changes, so if the duration of the control command execution is not correctly described, the video and the robot may not be coordinated when the video playback state changes from paused to playing, or when the playback position is skipped. This is due to the specification of the proposed system, which reads the control command to be sent at the playback position from the head when the video state changes from paused or skipped to playing. Therefore, by specifying the playback position at which the robot control command specified in the script is sent until just before the playback position at which the following control command is sent, the robot will perform the behavior it was performing before the video was paused again when the video playback state changes from paused to playing. If the playback position is skipped, the robot can operate by referring to the robot control command specified at the skipped playback position.

Regarding the third thing to consider, the proposed system has a control command that is read from the head and sent at the playback position when the video state transitions from paused or skipped to being played. According to the above specification, if control commands that execute multiple consecutive operations are described in a single playback position, depending on the learner's operation timing, the robot may reach the next playback position without running all the control commands described in one playback position. When the robot runs multiple

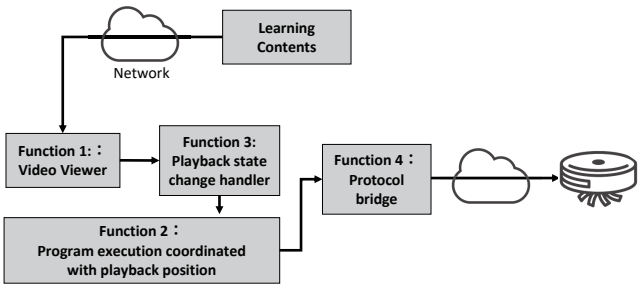

Fig. 4 Functional Block Diagram

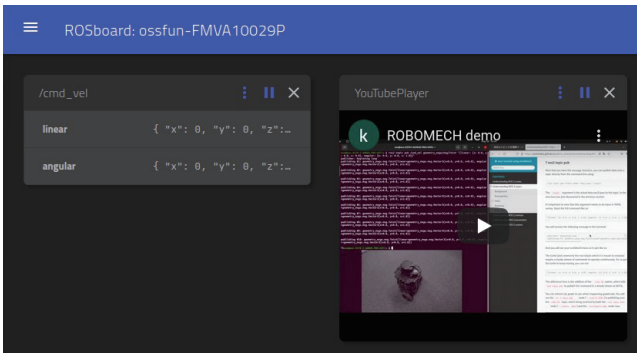

Fig. 5 E-Learning System UI

```
1   1
2   00:00:00,000 --> 00:00:01,000
3   alert("hello world");
```

Fig. 6 Srt Script Example Description

consecutive behaviors in a video, the playback position can be thinly divided and, control commands can be characterized as independent behaviors, allowing the robot to run straight behaviors even if the video state transitions during playback.

3.3 Learning in the Proposed System

Our system allows learners to see, in person, the robot's reactions which are explained in the video, as described in Section 2.2. The difference between the virtual and physical worlds can be understood by demonstrating the robot's behavior, which is affected by the surrounding environment, such as friction with the ground and air resistance, which cannot be known only through e-Learning. The fun of having an actual robot in action can be acquired by adding scenes of a real robot in action to e-Learning, which tends to be monotonous learning so that the contents of learned programs and control commands remain in the user's mind as experiences. Trial-and-error exposure to the robot and the surrounding environment can be acquired by viewing and understanding the robot's behavior in coordination with the video, then changing some of the programs and control commands to see the differences in the robot's behavior.

4. Implementation

The functions implemented in the proposed system and the communication flow are demonstrated in Fig. 4. This section describes the technical issues and solutions for implementing each of the functions demonstrated in Fig. 4, as well as the learning content and robot behavior used in the evaluation section that follows.

4.1 Video Viewer

A technical issue in implementing the video viewer function is that existing video viewers do not support embedding programs such as sending control commands to the robot. To solve the issue, we created a viewer with embedded video using YouTube's Iframe Player API[8] for the interface used in rosboard[7]. Rosboard is a ROS node that executes a web server to visualize the data sent and received by the ROS robot in a browser and was used because it is less dependent on the ROS version. The Iframe Player API embeds YouTube videos into web pages, allowing JavaScript programs to control the videos. Implementing a video viewer on a web server executed by rosboard allows the embedding of external programs. A screenshot of the implemented application is illustrated in Fig. 5.

It is designed to be able to view videos while retaining the functionality of the rosboard. This makes it possible to see and learn what kind of data the robot in the video is exchanging with the ROS network while the video and robot coordinate.

4.2 Program Execution Coordinated with Playback Position

A technical issue in implementing program execution coordinated with the playback position function is that there is no mechanism to synchronize the program executed in the created video viewer with the playback position of the video. To solve the issue, we use srt.js[9]. srt.js is a framework that can embed JavaScript programs in the form of srt files, officially supported as a subtitle format for video content on YouTube and other sites, and can execute programs depending on the playback position on the video. In srt.js, the Iframe Player API is used to get the playback position of the embedded video at regular intervals. When it matches the playback position described in the srt file, as illustrated in the second line of Fig. 6, the program is executed from the third line to the next playback position specified in the srt file. By incorporating srt.js into the system, it is possible to execute the programs in coordination with the playback position of the video.

4.3 Playback State Change Handler

A technical issue in implementing the playback state change handler function is that even if the playback state of the video changes during the coordination between the video and the actual robot, the control commands that were being sent before the change continue to be sent. A technical issue is similar to the specification in the proposed system described in section 3.2. To solve a technical issue, we examined the possible cases of learner operations on the video, and the results are shown in Table 1.

When an operation such as pause, skip, or end is performed on the video, the linked robot stops, and when the video is played back (resumed) from the pause, the operation that was completed before the pause is resumed. By detecting changes in the playback state of the video, robot control commands are sent to the

Table 1 Operation Patterns Performed by the Learner on the Video

Operation	Approach
pause skip finish	Stops the linked robots
play(resume)	Resume the action you were performing before the pause
fast-forward rewind	(not covered)

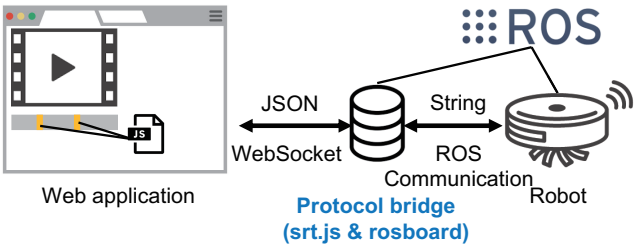

Fig. 7 Broker Function

robot in person to cause it to perform operations following its actions. Fast-forwarding and rewinding operations are not supported by YouTube, which is used to acquire video content for the proposed system, and are, therefore, not included in the scope of this paper. The Iframe Player API is used to detect changes in the video playback state. When the video playback state changes during pausing or buffering, the system sends a stop control command to the robot to implement state management. The reason for focusing on modifications to buffering is that the Iframe Player API does not have a function for detecting when a video is skipping playback positions, and buffering occurs when skipping on YouTube.

4.4 Protocol Bridge

A technical issue in implementing the protocol bridge function is the inability to communicate directly between the Web application and the ROS system due to differences in the programming languages and communication protocols used in development. To solve the technical issue, we implemented a protocol bridge that intermediates communication using a web server executing on rosboard. The constructed protocol bridge is illustrated in Fig. 7.

By implementing the protocol bridge, robot control commands can be sent as JSON data via WebSocket between the web application and the protocol bridge. Control commands formatted in the protocol bridge can be sent as string data via the ROS communication protocol. Therefore, a JavaScript program written in an srt file (srt script) can send control commands to a ROS node written in C++ or Python.

4.5 Learning Content

As learning contents used in the evaluation, we created videos and srt scripts based on the text of a part of the contents that can be learned in the official ROS 2 tutorial that many ROS 2 users use for learning immediately after installation, modified so that they can learn using an actual robot. The text is designed to help learners learn about basic ROS concepts and command line tools. The video was created with an awareness that the video only can be used as learning content by recording the voice explaining the contents of the text and the actual robot operating according to the commands. In the srt script, the video and the robot were coordinated in three parts of the learning contents: the part where the robot knows commands to monitor data flowing in the ROS system (coordination part 1), the part where the robot knows commands to send data to the ROS system via one-way communication (coordination part 2), and the part where the robot learns commands to send and receive data via bi-directional communication to and from the ROS system (coordination part 3). In addition, since some robot functions are difficult to operate in coordination with video, only those robot functions that are appropriate for the proposed system are included. The function of the robot suitable for the proposed approach does not depend on the previous state of the robot, and the modules that were synchronized can be stopped with the pause of the video. By contrast, functions that depend on the previous state of the robot (e.g., drone flight control) are not suitable for the proposed system. Considering the functions ideal for the proposed approach, we determined that the robot moves slowly forward at the first coordination point, quickly rotates in place at the second coordination point, and makes a sound while the robot remains stationary at the third coordination point. The reason the robot's behavior is different at each coordination point is to investigate the effect of the robot's behavior on the learning effect, and we expect to obtain different results for each of them.

5. Evaluation

In this section, we describe two experiments performed to evaluate the practicality of the proposed system.

5.1 Latency in the Video-Robot Coordination

The proposed system aims to improve the learner's learning efficiency and motivation by linking the video and the actual robot in person. However, when content requiring more precise synchronization with the video is used for learning, the latency between the playback of the video and the operation of the actual robot becomes a problem. Therefore, the latency from the execution of the srt script until the robot receives the control command and the latency from changing the playback state of the video until the system detects the change in playback state are measured to evaluate the coordination performance of the system.

The latency from the execution of the srt script until the robot receives the control command is demonstrated in Fig. 8. Here, the difference between the Unix time at the timing when the srt script containing the control command for the robot to move forward after the video was played and the Unix time at the timing when the robot received the control command via ROS communication after processing the data in the PC, 50 times each in the Wi-Fi and wired environments the outputs were got 50 times in the Wi-Fi environment and 50 times in the wired environment.

The latency from when the video playback state is changed to when the system detects the change in playback state is demonstrated in Fig. 9. Latency was measured by outputting the Unix time at the timing when the video was played and the srt script that changed the video playback state to paused was executed, and the Unix time at the timing when the system detected the video

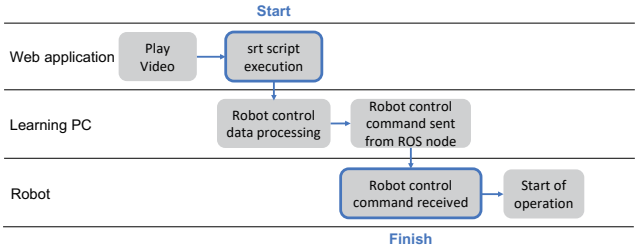

Fig. 8 Latency: From the Time the Srt Script is Executed until the Robot Receives the Control Command

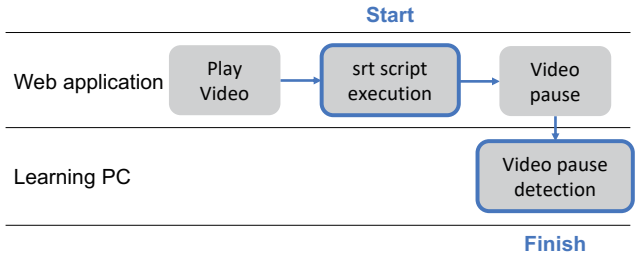

Fig. 9 Latency: From the Time the Video Playback State is Changed until the System Detects the Change in the Playback State

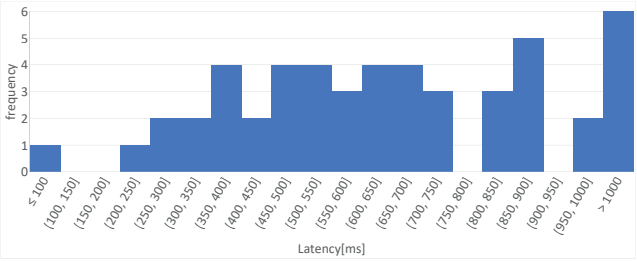

Fig. 10 Result: From the Time the Srt Script is Executed until the Robot Receives the Control Command(Wireless)

pause after the video was stopped, 200 times each, and finding the difference between them.

The experimental environment is shown in Table 2, the histogram of the latency measurement results from the execution of the srt script until the robot receives the control command is shown in Fig. 10 and Fig. 11, and the histogram of the latency measurement results from changing the playback state of the video until the system detects the change in playback state is shown in Fig. 12.

In Fig. 10, the maximum latency was 1419 ms, the minimum was 75 ms, and the average was 652.88 ms. In Fig. 11, the maximum latency was 1038 ms, the minimum was 64 ms, and the average was 543.94 ms. These results demonstrate that latency varies regardless of the network environment. Although the robot is sometimes able to recognize the discrepancy between the content and the robot, we do not consider this to be enough to affect the effectiveness of learning. In Fig. 12, the maximum latency was 77 ms, the minimum was 6 ms, and the average was 15.4 ms.

Table 2 Experimental Environment

Browser	Firefox Browser 96.0(64-bit)
Server OS	Ubuntu 20.04.3 LTS
Robot	Turtlebot3 Burger
Robot OS	Ubuntu 20.04.2 LTS
ROS 2	Foxy Fitzroy

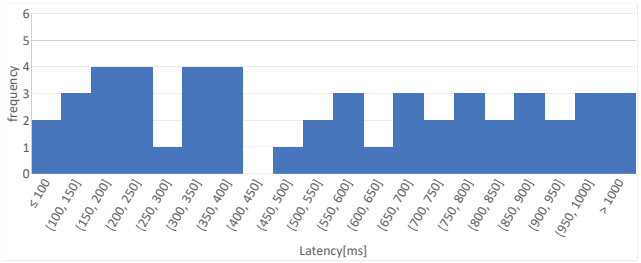

Fig. 11 Result: From the Time the Srt Script is Executed until the Robot Receives the Control Command(Wired)

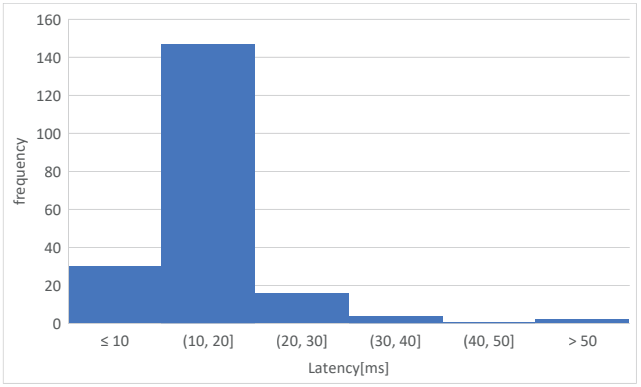

Fig. 12 Result: From the Time the Video Playback State is Changed until the System Detects the Change in the Playback State

The results demonstrate that the latency is stable and small, and there is little risk in the coordination when the video is paused.

5.2 Impact on Learning

Participants in the experiment answered a questionnaire regarding their knowledge of ROS, proceeded to learn about ROS using the proposed system and learning content, and answered a questionnaire regarding their learning details.

Before the experiment, participants who had known ROS before were asked to respond to a questionnaire about their learning history of ROS, and those who had learned ROS before were given an additional test on basic ROS concepts and command line tools to measure their level of understanding at the time before participating in the experiment. In the after-experiment questionnaire, all participants, regardless of their ROS learning history, were given a comprehension test similar to the one given before the investigation began, and were asked to evaluate on a five-point scale and to describe freely whether each part in which the video and the actual robot were coordinated was effective in helping them learn the robot.

The results of the questionnaire before the start of the experiment showed that three participants had learned ROS before, and three had not. Of those who had learned ROS, one could not build a system from scratch but could build a partial function, one could not build a system function but knew how to use a simple command line tool, and one had a basic knowledge of ROS. The results of the comprehension test are shown in Fig. 13.

Participants with a history of learning (A, B, C) increased the number of correct answers from before to after the experiment. However, there was a difference in the growth of the correct answers. Participants with no learning history (D, E, and F) had

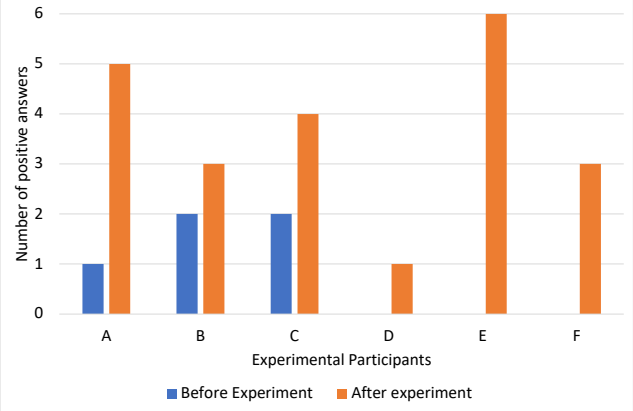

Fig. 13 Results of the Comprehension Test

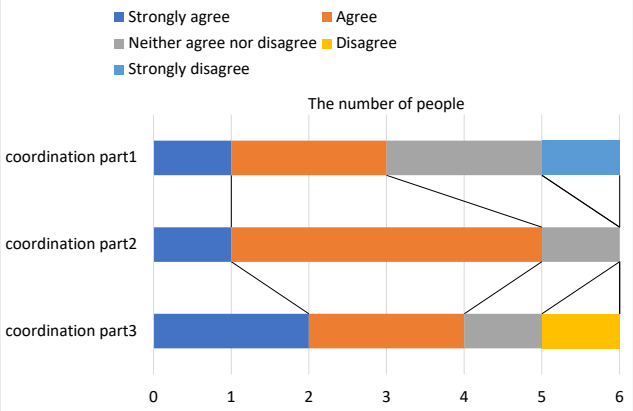

Fig. 14 Location-Specific Answers Coordinated with the Video

an increase in the number of correct answers from before to after the experiment, considering that they could answer zero questions correctly before the investigation. The results show that the proposed system and the learning contents are effective for learning.

The results of the question of whether the part that was coordinated with the videos was effective for learning are shown in the following Fig. 14.

As for a reason for the answer to the coordination part 1, some of the participants who answered ”strongly agree” or ”agree” said that it was easy to understand the actions of the commands executed on the video because the robot actually operated. In contrast, some of the participants who answered ”strongly disagree” said that they did not notice the coordination. As for a reason for the answer to the coordination part 2, some of the participants who answered ”strongly agree” or ”agree” said that it was interesting that they could see how the robot operated even with the robot in person and was impressed by the fact that the robot suddenly started to rotate. As for a reason for the answer to the coordination part 3, some of the participants who answered ”strongly agree” or ”agree” said that it was effective for learning that the sound changes when the command argument is changed while seeing the robot’s actual operation. In contrast, some of the participants who answered ”disagree” said that the same sound was played from the video, so they did not feel that the sounds were coordinated.

From the results of the questionnaire, it can be seen that the robot’s more significant movements, such as in the coordinated part 2, were more impressive than the smaller movements of the robot in the coordinated part 1, and thus received a highly rated score. The robot was stationary in the third part, so the robot did not seem to be synchronized with the video. However, the robot’s sound could be changed by simply changing some of the commands, suggesting that the learning from experience was significant. From the above, it can be said that using the proposed system for learning is effective, depending on the contents and the robot’s behavior.

6. Related Work

Kawatani et al.’s[10] study supports programming learning by coordinating sequential contents such as videos and slides with actual devices. This study proposes a system that can display source code and control IoT devices at specific timing by embedding programs into sequential contents, targeting IoT programming learning. The IoT devices in the sequential content and the IoT device in person work in coordination to enhance the learner’s understanding of IoT devices and motivation to learn.

A case study of developing a web interface for ROS is this study by Crick et al[11]. This study proposes a system that supports communication between the Web and ROS by making data sent and received in the ROS network available as JSON messages through the Web communication protocol. This makes it possible to develop robot applications without knowledge of robot control or sensing.

jupyter-ros[12] is an example of a web-based interactive program executed environment that enables ROS operation. Jupyter-ros allows users to develop and learn robots without being separated from their program because they can check the run results sequentially while writing node programs on the Web. The system is similar to the proposed approach in that it uses a web-based ROS system to provide interaction. It differs in that it is highly dependent on the ROS version and does not provide support for understanding the robot’s behavior.

7. Conclusion

This paper describes the implementation and evaluation of a system that supports the understanding of the state of the program executing in the robot by coordinating the actual robot in person with the lecture video explanation. We implemented a function that sends control commands to the robot following the playback position of the video, and a function that manages the state of the robot following changes in the playback state of the video while the video and the robot are coordinated. The evaluation section described two experiments performed to evaluate the practicality of the proposed system. In an experiment to measure the latency when the video and the robot work together, the results show the learning content and robot behavior the proposed system can cover. Experiments investigating the impact on Learning, the results show the effectiveness and usefulness of the proposed approach when used for learning.

Future work will be conducted in the following three plans. First, implement the functions to edit and describe programs on the Web. Robot control programming learning provides more opportunities for learning through repeated trial and error than gen-

eral programming learning because of the presence of an actual operating robot. Therefore, the proposed system needs to have a function to help the learner learn how the robot's behavior will change if any part of the program changes after the learner sees the robot in operation in coordination with the video. Second, implement a function to visualize the node configuration of the ROS system synchronously on a browser. When programming robot control using ROS, it is necessary to understand which part of the ROS system the ROS node currently being developed corresponds to, but it isn't easy to understand the relationship between nodes from the source code in ROS systems, which often handle vast amounts of source code. Therefore, the proposed system must have the function of learning by looking at the relationship between nodes. Third, we modify the learning content and conduct an evaluation experiment using it. Based on the results and opinions of the questionnaires obtained in the experiment, we will create learning content that modified the robot's memorable behavior and the timing of the coordination of the robot. In addition, since experiments investigating the impact on learning were conducted on a small number of people, it was difficult to determine whether there were significant differences in the results of the questionnaire answers. In the next experiment, the number of participants will be increased to evaluate the practicality of the system.

Acknowledgments This work was supported by JSPS KAKENHI Grant Number JP21K11832.

References

[1] Quigley, M., et al.: ROS : an open-source Robot Operating System, *ICRA Workshop on Open Source Software*(2009).

[2] Okada, K.: ROS, 5 Years After, *Journal of the Robotics Society of Japan*, Vol. 35, No. 4, pp. 270–273, DOI: 10.7210/jrsj.35.270 (2017). (in Japanese).

[3] Yamauchi, Y., Fujiyoshi, H., and Obinata, G.: Robotics Engineering Education Based on ROS, *Journal of the Robotics Society of Japan*, Vol. 35, No. 4, pp. 299–302, DOI: 10.7210/jrsj.35.299 (2017). (in Japanese).

[4] Benesse.: Udemy, Udemy (online), available from ⟨https://www.udemy.com/⟩ (accessed 2022-08-19).

[5] The Construct.: The Construct: A Platform to Learn ROS-based Advanced Robotics Online, (online), available from ⟨https://www.theconstructsim.com/⟩ (accessed 2022-08-19).

[6] Casan, A.G., Cervera, E., Moughlbay, A.A., Alemany, A. and Martinet, P.: ROS-based online robot programming for remote education and training, *International Conference on Robotics and Automation*, pp. 6101–6106, (2015).

[7] dheera: rosboard, (online), available from ⟨https://github.com/dheera/rosboard/⟩ (accessed 2022-08-19).

[8] Google Developers: YouTube Player API Reference for iframe Embeds, (online), available from ⟨https://developers.google.com/youtube/iframe_api_reference/⟩ (accessed 2022-08-22).

[9] Kurihara, K., Hashimoto, M.: [srt.js: Framework for embedding IoT-directed extension programs into video content] srt.js:Eizo kontentu heno IoT sikou kakutyou puroguramu umekomi hure-muwa-ku, *Workshop on Interactive Systems and Software*, pp. 24–29, (2016). (in Japanese).

[10] Kawatani, T., Tsukada, K., Kurihara, K.: [IoTeach: IoT programming learning support system linking real world and sequential content] IoTeach: jitusekai to jyunjyogata kontentu wo renkei sita IoT gakusyu sien sisutemu, *Workshop on Interactive Systems and Software*, (2021). (in Japanese).

[11] Crick, C., Jay, G., Osentoski, S., Pitzer, B., and Jenkins, O.G.: Rosbridge: ROS for Non-ROS Users, *Robotics Research: The 15th International Symposium ISRR*, pp. 493–504, (2017).

[12] RoboStack: jupyter-ros, (online), available from ⟨https://github.com/RoboStack/jupyter-ros⟩ (accessed 2022-10-19).