2023 32nd IEEE International Conference on Robot and
Human Interactive Communication (RO-MAN)
August 28-31, 2023. Paradise Hotel, Busan, Korea

2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN) | 979-8-3503-3670-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/RO-MAN57019.2023.10309587

Flexible Control and Task Manager System for Non-Contact Delivery Robots in COVID-19 Isolated Facilities

SungJoon Cho, Yisoo Lee, KangGeon Kim, Yong Seok Ihn, Jun-Sik Kim*, and Bum-Jae You*

***Abstract*— The COVID-19 pandemic has caused a global public health crisis, leading to increased costs for operating essential quarantine facilities and risks of infection for medical staff. To address these issues, we have developed a non-contact delivery robot, UTD-pro, that can deliver food and supplies to patients in an isolated residential treatment center. In case of emergency situations such as communication failure with the control system or malfunction of specific modules, the non-contact delivery robot should be able to perform commands robustly or return home. Otherwise, as before, medical staffs may have to enter the high-risk environment again. In this paper, we propose a flexible and robust remote control system and a robot task management system that enables the robot to execute commands correctly. Our robot task manager allows for autonomous task execution with a single command from the remote control system. Each robot task operates independently, allowing users to change task plans flexibly during robot operation. The sensor data communication and task execution communication processes are also independent, preventing any issue in one process from affecting the other. By applying our system to the robot, we conducted a long-term delivery experiment for a total of 7.723km for 79 days, and achieved 125 successes out of 149 trials. This experiment proves that our system can lead to stable delivery processes and contribute to the high reliability of the control system and the robot task manager.**

Fig. 1: UTD-pro is a non-contact delivery robot that can deliver lunch boxes and items to isolated residential treatment centers without requiring any physical contact. The robot can deliver twenty items in total on each round. It is equipped with wheels for mobility, a camera for table detection, and an arm for moving items.

I. INTRODUCTION

The COVID-19 pandemic has resulted in a significant number of patients worldwide and had a huge impact on our daily lives. Due to the contagious disease, people started to reduce direct contact among humans and this has become an opportunity to actively apply engineering technology in our daily lives [1]. To minimize direct contact and protect mental and physical health during the COVID-19 outbreak, various contactless delivery robots [2]–[4] and social robots [5] have been researched. In addition, medical robots for treating COVID-19 patients have also been researched [6]. In addition to medical robots, quarantine facilities have become essential to prevent further spread of viruses. However, operating isolation facilities has raised new issues such as the risk of infection for medical professionals and the cost of personnel and management. To solve these problems, research on delivery robots for food and medical supplies within isolation facilities and hospitals has been conducted [7], [8].

Researchers are studying various delivery robots to alleviate people's inconveniences. These robots include delivery robots that help deliver heavy packages to customers' homes to reduce the effort of postal delivery [9] and robots that deliver food to customers from restaurants to assist employees [10], [11]. Since these mobile robots perform planned tasks based on human commands, communication with the control system for receiving human commands and the robot task manager system play a crucial role. There have been many studies on the remote control of robots by humans. Some studies have used Android in the control system [12], and there are also cases of incorporating Google Glass [13]. There are also studies that use various communication methods such as Wi-Fi and Bluetooth to remotely control robots [12], [13], systems based on cloud platforms [14], and research based on ROS interface [15]. All of them have the common goal of remotely controlling and monitoring robots.

We have developed a non-contact delivery robot, UTD-pro, to deliver food and supplies to isolated patients. The delivery robot must be able to move to its destination on its own and perform the planned service at the correct location. In this paper, we propose a flexible task manager and a remote control system to enable efficient and robust work in isolation facilities.

*Jun-Sik Kim and Bum-Jae You are co-corresponding authors.
The authors are with the Center for Intelligent & Interactive Robotics, KIST (Korea Institute of Science and Technology), Seoul, 02792, South Korea. {chosj, yisoo.lee, danny, yongseok.ihn, junsik.kim, ybj}@kist.re.kr

This work was supported by Korea Advanced Research Program through the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) 2020M3H8A1114928.

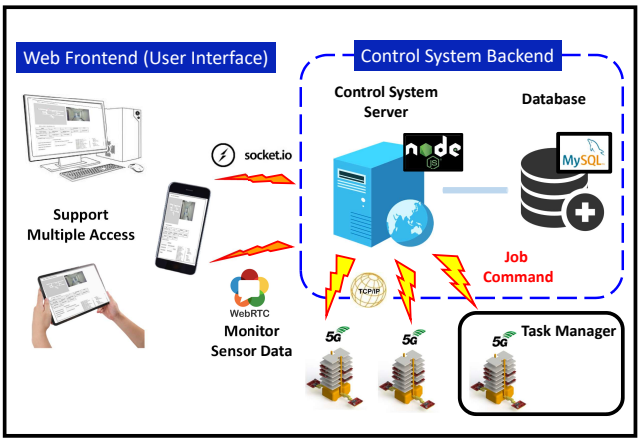


Fig. 2: System overview of our proposed approach. The control system consists of a web frontend and a backend. Users can send commands through the control system frontend, and the control system backend can send them to the robot task manager. A robot task manager can also sends its status information to the control system backend.

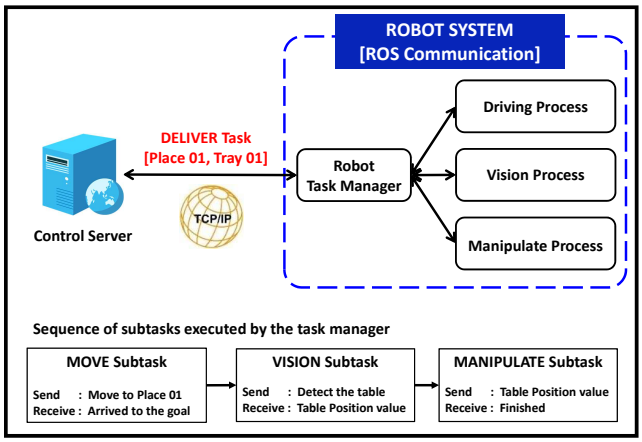


Fig. 3: Once the robot task manager receives a command from the control system backend, it proceeds to execute the command with other processes. The robot task manager must receive a response from the respective process in order to proceed to the next subtask.

As our main contribution, we present:

- UTD-pro can perform tasks and return home autonomously without continuous communication with the remote control system.
- The robot task manager has various subtasks that can run multiple processes within the robot, and these subtasks can be assembled freely to perform various tasks.
- Each subtask runs independently, allowing for fault tolerance, and enabling the user to modify the task plan during the process.
- The robot sends sensor data to the user online. We propose a method that operates real-time sensor data transmission process independently of communication with the control system to avoid interfering with the robot's robust task performance.

We conducted a long-term delivery experiment for 79 days using UTD-pro, and demonstrate the flexible and robust task performance and stability of the robot.

II. Method

A. Overall Structure

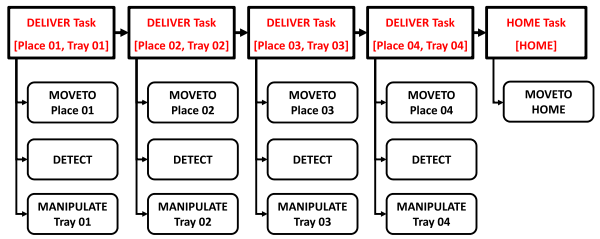
UTD-pro has been designed to deliver multiple lunch boxes and items to the user's designated location through contactless delivery. When a user sends a command, the robot must move to the correct location and deliver a large quantity of items to the appropriate table. To accomplish this, the robot system is equipped with a forward-facing camera, 2D LiDARs, wheel parts and a driving system, for navigation to the destination, as shown in Fig. 1. Additionally, the robot system includes a vision system consisting of a camera capable of recognizing a table for placing items, twenty trays for loading the items, and a robot arm system for retrieving items from the tray and placing them at the desired location.

The entire system for operating the robot is composed of a remote control system and a internal task manager, as shown in Fig. 2. The control system consists of a control system web frontend that enables users to send commands conveniently and a control system backend that serves as an intermediary between the user and the robot. Commands sent by the user are transmitted to the robot's task manager through the control system backend. The task manager breaks down the assigned task sequence into smaller, more specific subtask sequences and sends appropriate commands for each subtask stage, allowing the robot to perform its work accordingly. Examples of subtask processes include the robot's navigation, vision, and manipulation parts.

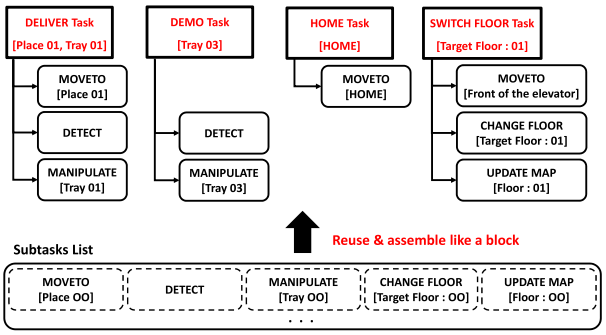
While the robot is performing the subtask sequence, the task manager sends real-time information about the robot's current status to the monitoring system. The monitoring system can display this information to the user in real-time and store it in a database for later use in improving and providing feedback on the robot system. The control system web frontend can also show the robot's sensor data online. After completing the delivery, the robot returns to home on its own to prepare for the next delivery or charge its battery. The remote control system backend can manage multiple robots with the same task manager process, and the user can send different commands to each UTD-pro as needed. In addition, multiple users can make requests to the robots by accessing the backend system with various devices.

B. Robot Task Manager

The task manager of the robot receives a task sequence from the control system and breaks it down into appropriate subtasks. For example, if it receives a DELIVER task, the task is specified into several detailed subtasks such as



(a) Multiple task sequence



(b) Generating various tasks

Fig. 4: Example of a task sequence and subtasks (a) It is possible to compose a sequence of tasks continuously, using the same subtasks. (b) The created subtasks can be freely reused and assembled to create various tasks.

MOVETO, DETECT, and MANIPULATE. It is possible to create tasks by combining various subtasks in different ways, and new subtasks can be easily created if necessary. The task manager sends commands to other processes to execute the planned subtasks in order. Communication with processes takes place within the ROS(Robot Operating System) interface.

An example process of executing a single DELIVER task in the task manager is shown in Fig. 3. The task manager sends a command to the driving process to move to Place 01 to execute the first subtask, MOVETO. The driving process travels to the stored location of Place 01 and sends real-time driving information to the task manager. When the destination is reached, an arrival message is sent, and the task manager sends a message to the vision process to detect the table for the next DETECT subtask. The vision process detects the table within the camera field of view and calculates the coordinates. The calculated coordinates are sent to the task manager, which then sends the table coordinates to the manipulate process to perform the next subtask. After completing the task of moving the object, the manipulate process sends an arm status to indicate that the command has been completed, and the task manager moves on to the next subtask or completes the task sequence.

Fig. 4(a) shows a task sequence for delivering items in trays 01∼04 to Place 01∼04, respectively. When continuous DELIVER task commands are received in the control system, the same subtask sequence is repeated, but the tray number to retrieve the delivery location and item should be assigned differently. This is because different option values can be applied even when reusing the same subtask. Lastly, there is a task to return HOME. The HOME task consists of only one MOVETO subtask and can be used when only moving to the home location is required. The reason why customization is possible, such as with the HOME task, is that tasks can be created by combining subtasks as desired, as shown in Fig. 4(b). By flexibly changing tasks and subtasks like building blocks, a new task sequence can be created every time depending on the delivery environment and robot situation. For example, if the robot needs to take an elevator by itself, a new subtask can be created to call the elevator and move to another floor, and then combined with the MOVETO subtask.

The robot task manager sends a command to another process only once when starting subtask execution, without continuously sending command data. This was designed to prepare for cases where even communication between processes within the robot cannot be guaranteed to be absolutely stable. The reason why the entire task can be carried out with only a small amount of essential data transmission is that the vision and manipulation processes are designed to be robust against situational changes. Even if the table's position changes, the vision process can robustly detect the table, and the manipulation process can move the item flexibly based on the coordinate results. As a result, the task manager can carry out the entire task robustly by transmitting only necessary information to other processes, thus reducing the burden on communication.

C. Control System Backend

The control system backend serves as an intermediary between the robot task manager and the control system web frontend, communicating via TCP/IP and socket.io, respectively. It receives user requests from the web frontend, sends a list of tasks in JSON format for achieving the user's requests to the robot task manager, and stores the status information periodically sent by the robot in the database, which is then forwarded to the control system web frontend.

Continuous communication between the control system backend and the task manager is not required, as the robot can perform its tasks autonomously without any additional control. While consistent communication is crucial when sending detailed command data in a control system, minimizing the need for continuous communication enables the robot to perform tasks robustly, even in an unstable communication environment. This is possible because the robot task manager can break down the received task sequence commands into subtasks, assemble them, and perform the necessary tasks by itself without relying on constant communication with the control system backend.

Once communication between the control system backend and the robot is established, status information is received from the robot at regular intervals, and all data is stored in a MySQL database. The status information includes the robot's ID, user command, floor ID, battery status, and

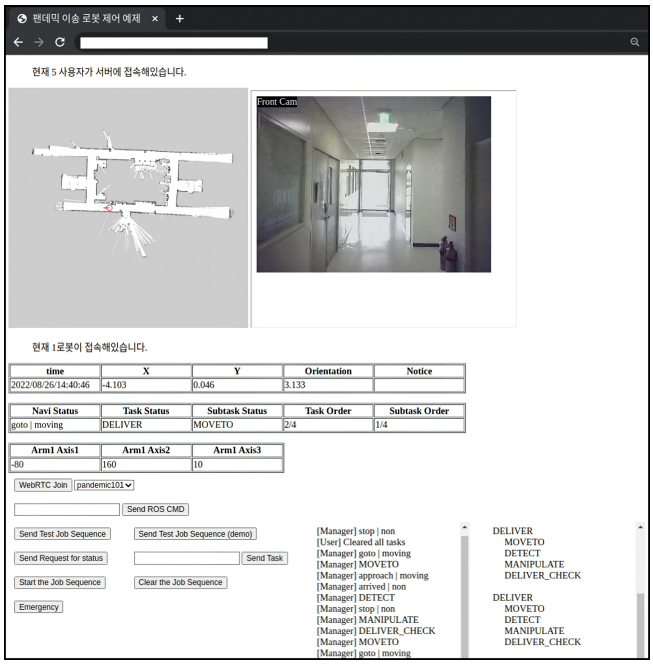


Fig. 5: Web frontend of the control system enables users to send commands to the robot and monitor the status of its execution. Real-time sensor data feed from the robot can also be accessed through WebRTC.

current pose, as well as the current task, current subtask, amount of total task, whole task list, and detailed progress information (arm axis, tray status, arm status, delivery check result, and navi status) for each process. The stored data is utilized for analyzing problems and processing times that occurred during the service.

D. Control System Web Frontend

The control system web frontend consists of buttons and blank spaces, as shown in Fig. 5, to enable users to easily send commands and display the current status of the robot. The web frontend sends a task sequence to the control system backend. Each task in the task sequence includes detailed option information on the place to deliver and the tray number in which the item to be delivered is. The robot's task manager can process this information properly by knowing only the delivery area number, allowing the driving process to reach the correct destination and complete the delivery. In addition, the command is flexible because the number of delivery destinations can be set freely within a limited number of trays, and the delivery destinations can also be set arbitrarily.

Users can intervene during the robot's command execution. In case of emergencies or changes of mind, they can send a stop command or delete the task list and modify the plan from the control system web frontend. Each subtask managed by the task manager is an independent process, so any problems with a specific subtask will not affect the operation of other subtasks. Additionally, the task manager has the feature of freely reorganizing subtasks according to the situation. Therefore, if a problem occurs during the robot's task execution, the user can intervene and make new changes to the current subtask plan. In this way, users can flexibly execute commands and respond to changes using the task manager in any situation.

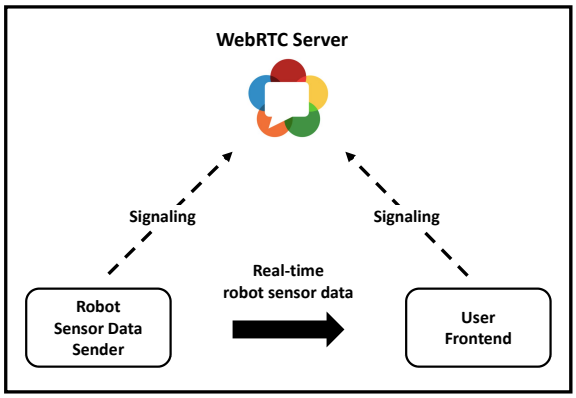


Fig. 6: P2P communication can be achieved through WebRTC. The robot and the user frontend establish an initial connection through signaling, and subsequently exchange data directly with each other.

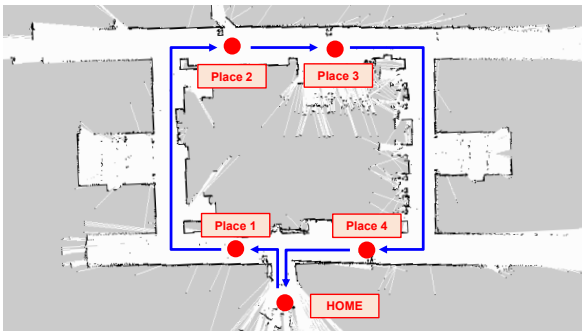


Fig. 7: The delivery experiment map includes four delivery points and HOME. The robot is tasked with delivering the correct tray of items to all delivery points and returning to HOME.

E. Real-time Communication of Sensor Data

In the control system web frontend, real-time sensor data of robots can be received as shown in Fig. 6. The web frontend is directly connected to the robot via WebRTC (Web Real-Time Communications) [16]. WebRTC is a technology that enables P2P connection between browsers without the need for intermediaries such as servers, except for the initial connection. P2P connection ensures fast speed since the devices are directly connected, and it provides security against intermediary attacks because HTTPS is enforced. Transmitting and receiving sensor data via P2P means that it is independent of the communication between the control system backend and the robot task manager. This parallel communication method reduces the burden on the control system

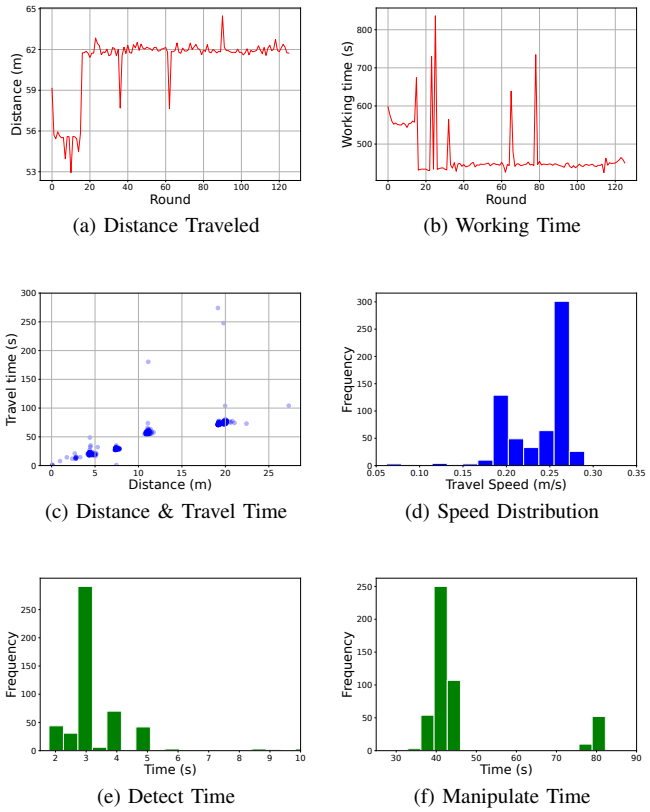
	Start time	End time	Total distance	Total time
...		...		
99	2022/11/16/11:08:48:367	2022/11/16/11:16:06:10	61.782	437.733
100	2022/11/16/11:16:17:17	2022/11/16/11:23:39:117	61.798	441.947
101	2022/11/16/11:23:51:117	2022/11/16/11:31:18:592	61.6	447.475
102	2022/11/16/11:31:27:957	2022/11/16/11:38:50:884	61.767	442.927
103	2022/11/16/11:39:06:969	2022/11/16/11:46:27:999	62.055	441.03
104	2022/11/16/11:46:37:489	2022/11/16/11:54:03:200	61.605	445.711
105	2022/11/16/13:56:48:761	2022/11/16/14:04:15:227	61.936	446.466
106	2022/11/16/14:04:30:530	2022/11/16/14:11:54:438	61.834	443.908
107	2022/11/16/14:12:19:650	2022/11/16/14:19:45:649	61.862	445.999
108	2022/11/16/14:43:27:631	2022/11/16/14:50:54:678	61.94	447.047
109	2022/11/16/14:51:56:890	2022/11/16/14:59:19:188	62.275	442.298
110	2022/11/16/15:00:28:677	2022/11/16/15:07:49:618	61.965	440.941
111	2022/11/16/15:08:04:598	2022/11/16/15:15:28:946	61.695	444.348
112	2022/11/16/15:52:06:918	2022/11/16/15:59:30:856	62.149	443.938
113	2022/11/16/20:03:21:963	2022/11/16/20:10:49:44	61.563	447.477
114	2022/11/18/14:59:09:109	2022/11/18/15:06:14:212	62.108	425.103
115	2022/11/18/15:07:06:251	2022/11/18/15:14:49:522	62.312	463.271
116	2022/11/18/15:15:00:102	2022/11/18/15:22:24:651	61.914	444.549
117	2022/11/18/15:26:04:737	2022/11/18/15:33:35:874	62.01	451.137
118	2022/11/18/15:34:06:822	2022/11/18/15:41:35:882	62.749	449.06
119	2022/11/18/15:42:03:361	2022/11/18/15:49:32:971	61.934	449.61
120	2022/11/18/17:21:58:378	2022/11/18/17:29:28:977	61.844	450.599
121	2022/11/18/17:32:35:503	2022/11/18/17:40:09:182	62.101	453.679
122	2022/11/18/17:40:19:278	2022/11/18/17:47:57:411	62.382	458.133
123	2022/11/18/17:48:08:980	2022/11/18/17:55:53:617	62.182	464.637
124	2022/11/23/16:59:22:744	2022/11/23/17:07:02:369	61.755	459.625
125	2022/11/24/16:26:28:991	2022/11/24/16:33:59:77	61.733	450.779

(a) Delivery log for successful cases

	Time	...	naviStatus	naviType	naviGoal	currentTaskID	currentSubtaskID	currentTask	currentSubtask	numTotalTask
0	2022/09/07/15:25:05:562		non	stop		1	0	DELIVER	MOVETO	5
1	2022/09/15/15:34:30:708		non	stop		2	2	DELIVER	MANIPULATE	5
2	2022/09/27/11:55:42:419		non	stop		0	2	DELIVER	MANIPULATE	5
3	2022/09/30/16:15:18:569		non	stop		0	3	DELIVER	DELIVER_CHECK	5
4	2022/10/04/14:35:20:139		moving	goto		2	0	DELIVER	MOVETO	5
5	2022/10/26/14:15:37:700		non	stop	100	0	2	DELIVER	MANIPULATE	5
6	2022/10/26/19:34:40:739		non	stop	100	0	2	DELIVER	MANIPULATE	5
7	2022/10/26/20:37:24:565		moving	goto	103	3	0	DELIVER	MOVETO	5
8	2022/11/02/11:08:08:757		moving	goto	101	1	0	DELIVER	MOVETO	5
9	2022/11/02/11:18:28:478		non	stop	101	0	2	DELIVER	MANIPULATE	5
10	2022/11/03/11:32:27:545		non	stop	100	0	2	DELIVER	MANIPULATE	5
11	2022/11/04/17:33:15:912		moving	goto	104	4	0	HOME	MOVETO	5
12	2022/11/09/13:57:02:129	...	non	stop	100	0	3	DELIVER	DELIVER_CHECK	5
13	2022/11/09/14:10:26:231		non	stop	100	0	2	DELIVER	MANIPULATE	5
14	2022/11/10/14:29:53:695		non	stop	100	0	2	DELIVER	MANIPULATE	5
15	2022/11/11/14:28:43:761		non	stop	100	0	2	DELIVER	MANIPULATE	5
16	2022/11/11/15:51:09:298		non	stop	100	0	3	DELIVER	DELIVER_CHECK	5
17	2022/11/15/13:47:31:820		non	stop	100	0	2	DELIVER	MANIPULATE	5
18	2022/11/15/14:32:49:445		moving	goto	101	1	0	DELIVER	MOVETO	5
19	2022/11/15/22:04:19:522		non	stop	101	1	2	DELIVER	MANIPULATE	5
20	2022/11/16/14:24:30:856		non	stop	102	2	2	DELIVER	MANIPULATE	5
21	2022/11/16/16:04:18:41		non	stop	100	0	2	DELIVER	MANIPULATE	5
22	2022/11/16/16:25:07:426		non	stop	100	0	2	DELIVER	MANIPULATE	5
23	2022/11/16/17:01:10:26		non	stop	100	0	2	DELIVER	MANIPULATE	5
24	2022/11/16/20:21:27:490		non	stop	103	3	2	DELIVER	MANIPULATE	5

(b) Delivery log for failed cases

Fig. 8: Example of delivery logs, (a) Based on the logs recorded in the database, the analysis of successful delivery cases can be performed. (b) By analyzing the results of delivery failures recorded in the database, the cause of errors can be inferred.



(a) Distance Traveled (b) Working Time (c) Distance & Travel Time (d) Speed Distribution (e) Detect Time (f) Manipulate Time

Fig. 9: Data analysis for the long-term evaluation, (a) Distance traveled for a task sequence (b) Working time for a task sequence (c) Correlation between distance traveled and travel time of a MOVE subtask (d) Speed histogram of a MOVE subtask (e) Time histogram of a DETECT subtask (f) Time histogram of a MANIPULATE subtask

backend server, and even if the communication process is interrupted, it does not affect the communication between the control system backend and the robot task manager, allowing the robot to perform tasks robustly. Conversely, even if the communication between the control system and the robot is interrupted, P2P data transmission and communication remain active, enabling the current situation of the robot to be grasped.

III. Results

We conducted a long-term experiment to demonstrate the reliability of our delivery robot system. The scenario involved users sending a job sequence to the robot, which would then depart from home and deliver lunch boxes to tables at Place 01-04, before returning to the home location, as shown in Fig. 7. The experiment lasted for a total of 79 days, during which the robot delivered approximately 7.723 km. We attempted the experiment 149 times and achieved 125 successful deliveries. Fig. 8(a) shows an example of the control system log for the 125 successful deliveries. For each delivery experiment, the distance traveled by the robot and the time taken to complete the entire set of commands are calculated. Fig. 8(b) shows the robot status information for the remaining 25 failed cases. The failure cases log records what the robot was doing and under what conditions it stopped working or malfunctioned. By analyzing the robot status information for the failed cases, we were able to infer what the robot was doing when it stopped or malfunctioned, and why it stopped. The reasons for failure were largely attributed to issues with USB connectivity. We were able to resolve all the causes of the problem by analyzing the robot status information for the failed cases. In addition, there were no issues caused by the proposed control system and robot task manager process. Through this experiment, we demonstrated the robustness of the control system and the robot task manager.

Fig. 9(a)-(f) show the results of the analysis conducted on successful delivery cases during the long-term experiment.

Fig. 9(a) represents the distance traveled data, indicating the distance covered by the robot during the execution of a task sequence. The robot traveled approximately 62m per task sequence and carried out deliveries. Fig. 9(b) displays the time taken to execute a task sequence command. If the robot was not obstructed while delivering items to four locations, it usually took around 450 seconds. It can be known that it takes about 100 seconds to move to the delivery point once and to move the lunch boxes. The reason for the decrease in working time from the 16th experiment is that we increased the driving speed and manipulation speed. Fig. 9(c) is a scatter plot created using distance and travel time data for each MOVE subtask. Fig. 9(d) shows that the robot's average speed is mostly around 0.2-0.3 m/s, calculated by dividing the travel distance by the travel time. Fig. 9(e) indicates that it took around 2.8-3.3 seconds on average to detect the table. Fig. 9(f) shows that it takes approximately 35-44 seconds to move the items once, depending on the tray number, as the robot arm goes up and down at different heights, causing a slight difference in the time required. Through the long-term experiment, we were able to analyze the approximate processing time and performance of each subtask process for numerous cases, understand how to improve our system to address occasional error situations, and demonstrate the robustness of our control system and the robot task manager.

IV. CONCLUSIONS

The increase in the number of COVID-19 infected patients has highlighted the importance of isolated residential treatment center, but this has led to problems such as the risk of infection for medical staff, as well as the cost of management of these facilities. Therefore, we have developed a non-contact delivery robot, UTD-pro, that can deliver food and supplies to isolated patients. In this paper, we propose a flexible control system and task manager that can help the robot complete deliveries and return home robustly in environments that are vulnerable to infection and where humans cannot enter. The robot task manager can flexibly assemble independent subtasks to create a task. It enables the robot to complete deliveries without the need for continuous communication with the control system, while also allowing for flexible modification of a task plan by the user. Furthermore, the sensor data communication process operates independently of the task execution process, allowing for more robust task performance without mutual interference. We have demonstrated the stable delivery capability of the robots task manager and the control system through long-term experiments.

If communication between the control system and the elevator becomes possible, we plan to expand the system to allow the robot may be able to call the elevator and move to other floors to provide service, rather than being limited to the floor they are currently on. This would enable faster delivery services without the need for the robot to visit home or for the user to manually move the robot to a different floor. From now on, we plan to develop a system in which robots will autonomously move to different floors to perform deliveries and return to the home location. This process will involve generating and combining subtasks related to calling elevators, changing floors, and updating maps specific to each floor for navigation.

REFERENCES

[1] Xi Vincent Wang and Lihui Wang. A literature survey of the robotic technologies during the COVID-19 pandemic. *Journal of Manufacturing Systems*, 60(December 2020):823–836, 2021.

[2] Manuel Cardona, Fernando Cortez, Andres Palacios, and Kevin Cerros. Mobile robots application against covid-19 pandemic. *2020 Ieee Andescon, Andescon 2020*, 2020.

[3] Cheng Chen, Emrah Demir, Yuan Huang, and Rongzu Qiu. The adoption of self-driving delivery robots in last mile logistics. *Transportation Research Part E: Logistics and Transportation Review*, 146(January 2020):102214, 2021.

[4] Murad Mehrab Abrar, Raian Islam, and Md Abid Hasan Shanto. An Autonomous Delivery Robot to Prevent the Spread of Coronavirus in Product Delivery System. *2020 11th IEEE Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON 2020*, pages 0461–0466, 2020.

[5] Laura Aymerich-Franch and Iliana Ferrer. Liaison, safeguard, and well-being: Analyzing the role of social robots during the COVID-19 pandemic. *Technology in Society*, 70(September 2021):101993, 2022.

[6] Antonio Di Lallo, Robin R Murphy, Axel Krieger, Junxi Zhu, Russell H Taylor, and Hao Su. Medical Robots for Infectious Diseases. *IEEE Robotics Automation Magazine*, 28(1):18–27, 2021.

[7] Fernando Carreira, Tomé Canas, Arlindo Silva, and Carlos Cardeira. i-Merc: A mobile robot to deliver meals inside health services. *2006 IEEE Conference on Robotics, Automation and Mechatronics*, 2006.

[8] Supachai Vongbunyong, Salil Parth Tripathi, Kitti Thamrongaphichartkul, Nitisak Worrasittichai, Aphisit Takutruea, and Teeraya Prayongrak. Simulation of Autonomous Mobile Robot System for Food Delivery in In-patient Ward with Unity. *Proceedings - 2020 15th International Joint Symposium on Artificial Intelligence and Natural Language Processing, iSAI-NLP 2020*, 2020.

[9] Daegyu Lee, Gyuree Kang, Boseong Kim, and D. Hyunchul Shim. Assistive Delivery Robot Application for Real-World Postal Services. *IEEE Access*, 9:141981–141998, 2021.

[10] D. Saravanan, E. Racheal Anni Perianayaki, R. Pavithra, and R. Parthiban. Barcode system for hotel food order with delivery robot. *Journal of Physics: Conference Series*, 1717(1):1–5, 2021.

[11] Yunlong Sun, Lianwu Guan, Zhanyuan Chang, Chuanjiang Li, and Yanbin Gao. Design of a low-cost indoor navigation system for food delivery robot based on multi-sensor information fusion. *Sensors (Switzerland)*, 19(22), 2019.

[12] Jan Nádvorník and Pavel Smutný. Remote control robot using Android mobile device. *Proceedings of the 2014 15th International Carpathian Control Conference, ICCC 2014*, pages 373–378, 2014.

[13] Song Yu, Xi Wen, Wei Li, and Genshe Chen. Google glass-based remote control of a mobile robot. *Sensors and Systems for Space Applications IX*, 9838(May 2016):983812, 2016.

[14] Baoke Zhou, Wusheng Chou, and Shaobo Wu. Remote control system of mobile robot based on cloud platform. *2017 2nd International Conference on Robotics and Automation Engineering, ICRAE 2017*, 2017-December:94–98, 2018.

[15] Zhi Li and Xuesong Mei. Navigation and Control System of Mobile Robot Based on ROS. *Proceedings of 2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference, IAEAC 2018*, (Iaeac):368–372, 2018.

[16] Branislav Sredojev, Dragan Samardzija, and Dragan Posarac. WebRTC technology overview and signaling solution design and implementation. *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2015 - Proceedings*, (May):1006–1009, 2015.