electronics

MDPI

Article

Design and Implementation of Intelligent Automated Production-Line Control System

Jeng-Dao Lee *, Hung-Yu Hsu, Cheng-Yi Li and Jun-Yi Yang

Department of Automation Engineering, National Formosa University, Yunlin County 632, Taiwan; qet10336@gmail.com (H.-Y.H.); 40327210@gm.nfu.edu.tw (C.-Y.L.); 40527236@gm.nfu.edu.tw (J.-Y.Y.)
* Correspondence: jdlee@nfu.edu.tw

Abstract: Owing to the different consumption habits of people, the market has rapidly changed, and customized demands have increased in recent years. This issue has led to the need for manufacturing industry to improve their production-process automation system control to become faster and more flexible. An intelligent automated production-line control system (IAPLCS) is proposed in this study, in which a graphic control system (GCS) is designed to integrate equipment communications in automated production. This system runs complex packaged programs for function blocks, which enable users to easily build function blocks in GCS. Users can plan the control flow of the system to control the machine operation. Not only is a web interface set in the IAPLCS, which makes it easier for users to operate and monitor, but also Internet-of-Things (IoT) sensors can be added for data collection and analysis based on the characteristics of the production line and equipment. Finally, the proposed control system was implemented in an actual production line to verify its feasibility.

Keywords: graphical control; process control; communication integration; IoT

Citation: Lee, J.-D.; Hsu, H.-Y.; Li, C.-Y.; Yang, J.-Y. Design and Implementation of Intelligent Automated Production-Line Control System. *Electronics* **2021**, *10*, 2502. https://doi.org/10.3390/electronics10202502

Academic Editors: Jahangir Hossain and Davide Astolfi

Received: 1 September 2021
Accepted: 9 October 2021
Published: 14 October 2021

1. Introduction

Because of the rapid development of sensor components, information technology, and wireless network technology in recent years, the development of industrial network applications has been accelerated. The related applications include, but are not limited to, smart vehicles, smart security, smart logistics, smart medical system, smart entertainment, smart energy, and smart buildings. The Internet of Things (IoT), which is considered to be the central topic in the next wave of industrial revolution and development of information and communication technology industry in the future, offers huge business opportunities and has become a key field for manufacturers to compete.

Recently, many information technologies (e.g., IoT, big data, and cloud computing) have been introduced into traditional industries and have introduced a significant effect on the development of industry 4.0. A cyber-physical system (CPS) [1] is an integrated system that combines the field of computer computing with sensors, actuators, and computer computing, and these are widely adopted in factories. The implementation of IoT devices in the factory enables data collection during the manufacturing process. Along with a growing body of evidence that shows the performance of the physical network manufacturing system, and understands the supporting data-driven decision-making process, there is a large amount of relevant literature by scholars that investigates [2] and analyzes networks of physical, sustainable, intelligent manufacturing systems and sensing technology; and integrated with the sustainable, intelligent manufacturing point of view, we can see that today's smart and sensing technology becomes very important. To fully fathom the challenges and opportunities ahead, we must adopt a holistic view of CPSs that includes self-adaptation, autonomy, efficiency, functionality, reliability, safety, scalability, and usability. In the CPS adaptive literature section, scholars discuss the techniques needed to build a common CPS architecture that provides efficient adaptive features. Based on the literature review of adaptive technology in CPSs by scholars [3,4], these studies discuss the

technologies needed to build a common CPS architecture, and considers that effective mechanisms must be developed to provide self-monitoring capability for CPS subsystems to prevent or correct abnormal behavior in operation and adapt to respond to the occurrence of faults and errors. As well as changing environmental conditions or the state of internal CPSs, the architecture must provide efficient adaptive features. In addition to exploring adaptability, cloud computing and big-data technologies have also emerged in the scientific community in recent years as key enabling technologies to address the current needs of CPSs, the value creation utilizing cloud-based CPS in robotic mobile fulfillment system had been addressed [5]. The large-scale CPS real-time monitoring platform based on big data technology is designed to perform real-time analysis with the goal of monitoring industrial machines in a real work environment. This was verified by implementing the proposed solution on a real industrial-use case involving multiple industrial presses [6].

The real physical phenomena are converted into digital values so that a virtual digital system can communicate with the physical world. For example, in the intelligent factory of Siemens in Germany Amberg, 75% of the workload can be replaced by machines, and the factory has a yield of almost 100% [7]. The research and application of task allocation in a multirobot system [8], which is an intelligent factory, has been developed for many years, but the current multirobot task-assignment mechanism can not easy to implemented in real robot systems directly. The present study explicates the problem of heterogeneous multirobot cooperative-task assignment. A distributed auction algorithm was designed to classify the different abilities of robots using atomic capabilities. According to the distance between the robot and target task, and the matching degree between the robot and task, a task-assignment problem model was established to solve the heterogeneous multitask problem; namely a robot collaborative-task assignment system. The simulation results show that the model is correct and the algorithm is effective. The current research provides a blueprint from existing manufacturing systems to smart factory. An empirical study was conducted at a leading thin-film-transistor liquid-crystal display (TFT-LCD) company [9], which proposed a simulation-based dynamic scheduling and dispatching model for TFT-LCD array manufacturing. The results demonstrated that the capacity loss of a bottleneck workstation and delivery tardiness could be significantly improved by the proposed hybrid genetic algorithm. This contribution provided an insight into how this objective can be achieved by utilizing software-defined networking for the specific use of the smart factory web [10]. To address the heterogeneous and changing demands of the users, the factory has made flexible not only its production assets but also its network in terms of management and configuration. After deep-learning-based quality control, the satisfactory products are forwarded to the production stages, and the unsatisfactory products are separated [11]. In the present research, a visual quality-control automation application is proposed using a camera installed over the assembly line in a smart-factory model. We used the cloud-coupler and plug-and-work techniques to create a new factory or retrofit an existing one that is available in the smart factory web to share capability information toward a new marketplace for manufacturing [12]. In the present study, we not only reduce the technical efforts but also the use of standards to define the interfaces. Cloud–fog computing network architecture can be efficiently used to store, analyze, and utilize the accumulated data in smart factories [13]. We built and evaluated the system model, and developed a testbed based on the proposed architecture. As a result, the cloud developed in the open stack can be used for smart-factory operations, analyze various types of data, and check the status of real-time processing through a dashboard. These processes create the most suitable structure for a smart-factory environment that requires real-time performance. Some problems that industrial big-data systems meet in a smart factory are discussed [14]. The design concerns of the industrial big-data-based system are abstracted from the product life-cycle perspective. The design concerns focus on the requirements from consumers, system builders, and other stakeholders in the smart factory.

Owing to the development of intelligence technology, the production model is oriented from previous mass production to customized demand, and the production line

must be highly adaptable to the customized demand. Many different innovative architectural designs for Industry 4.0 are continuously proposed [15–17]. In addition, a more popular need is required to upgrade the existing industry and improve its automation. Generally, small-scale manufacturers often cannot sufficiently invest to build advanced intelligent production, such as in semiconductor factories and related industries. Often, they can only gradually upgrade their existing machinery. In recent years, production lines have relied more and more on flexible or smart work cells, especially within the context of Industry 4.0. Although many people want to build a smart factory, upgrading the degree of automation of existing factories is a more important requirement. Therefore, machine-to-machine (M2M) and Internet of Things (IoT) technologies have been continuously introduced into the industry to analyze production and diagnose problems in order to improve the factory's automation. Currently, scholars are discussing a pluggable software Orchestrator [18], which can automatically extract instructions from a given input, set the factory process accordingly, and execute production by itself to realize a dynamic resource arrangement. The automation of a robot control system based on an automatic programming environment minimizes user rework and analysis time when there is a frequent need to redistribute the robot task through robot self-observational learning, independent reprogramming, or—through the integration of the M&S environment—maintain and promote model reuse, and to provide an automatic model-generation function to obtain the best solutions [19,20]. At present, even internal personnel training of enterprises has introduced intelligent learning [21]. CNN neural networks with SVM classifiers can extract and share expert knowledge according to film materials, effectively improve the learning process of employees in manufacturing companies, and shorten the time cost of personnel training. In this research, the IAPLCS was successfully developed and applied to the actual automatic production. It will become more convenient to edit and monitor the control process by using the GCS. Users can also get equipment information, including the grain rate, intelligent sensor data, real-time production line monitoring, etc. This will help narrow the gap between research and practice, and allow users to apply the system flexibly, which will helpful to implement different smart manufacturing strategies.

Therefore, the current research aims to develop a general automation-equipment control system with communication integration and a graphic control interface that can adapt to the production-line process in order to complete different production requirements.

2. Materials and Methods

Smart production-line conceptual diagram in this IAPLCS is shown in Figure 1, automatically processes, analyzes, communicates, and feeds back constructive information in real time to the user interface when the embedded sensors send messages to the data server through the network. This system represents an integration of the machine-to-machine and machine-to-system structures, and it effectively manages the relationships of automation equipment. In this system, production processes can be selected using mobile phones or desktops. Moreover, the current status of the production line is displayed on the user interface in this system. When a fault or event occurs, the alarm is triggered, and a notification is then sent via email or SMS to relevant personnel.

A GCS for automation equipment with communication integration was developed to address quick changes in the production-line processes. A graphical interface is needed to package the machine communication and commands into graphics to develop and quickly change the process needs. Then, we used a simple flowchart to build the relationships of the production-line control process, as shown in Figure 2. As shown in Figure 2b, there are three most commonly used functions, namely the circle responsible for the beginning and end, the rectangle responsible for the program steps, and the diamond responsible for decision-making. In the flowchart example, Rectangle-0 executes the action of writing Y0 as 0 after the program flow starts. In the next step, Diamond-0 determines whether M50 is 1—if it is, it ends the program; if not, it skips to Rectangle-1 to execute writing Y0 as 1, and jumps to the rectangle to continue execution. When the process needs to be changed, a

block can be dragged to the appropriate position, and a relationship can be developed to complete the modification. In the present study, an object-oriented programming method is adopted to subdivide the system into many classes to develop suitable system architecture. Therefore, this system is more convenient for maintaining or updating the system in the future. The class hierarchy proposed in this research is shown in Figure 3. It is divided into the GCS main interface, communication area, login area, and graphical area. The composition, structure, and functions of each area are described in the next sections.

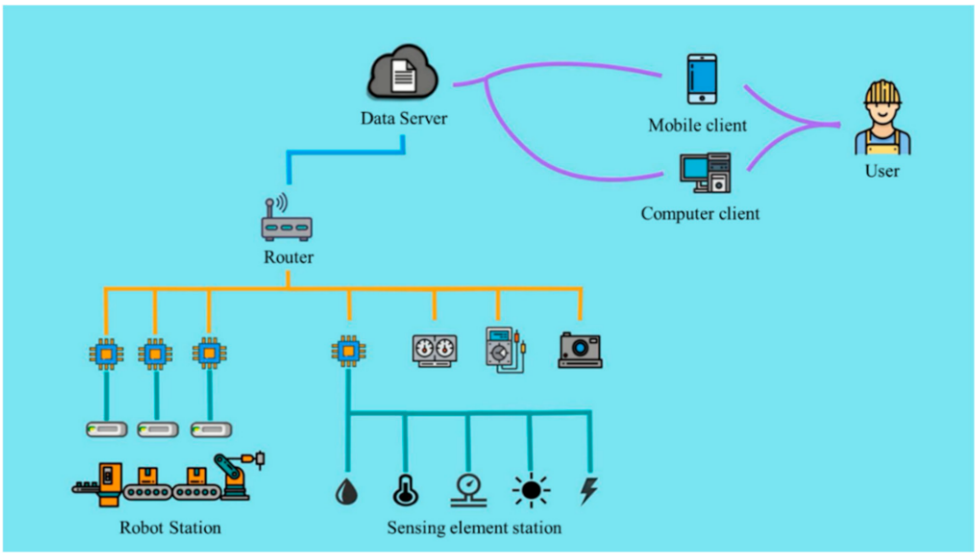
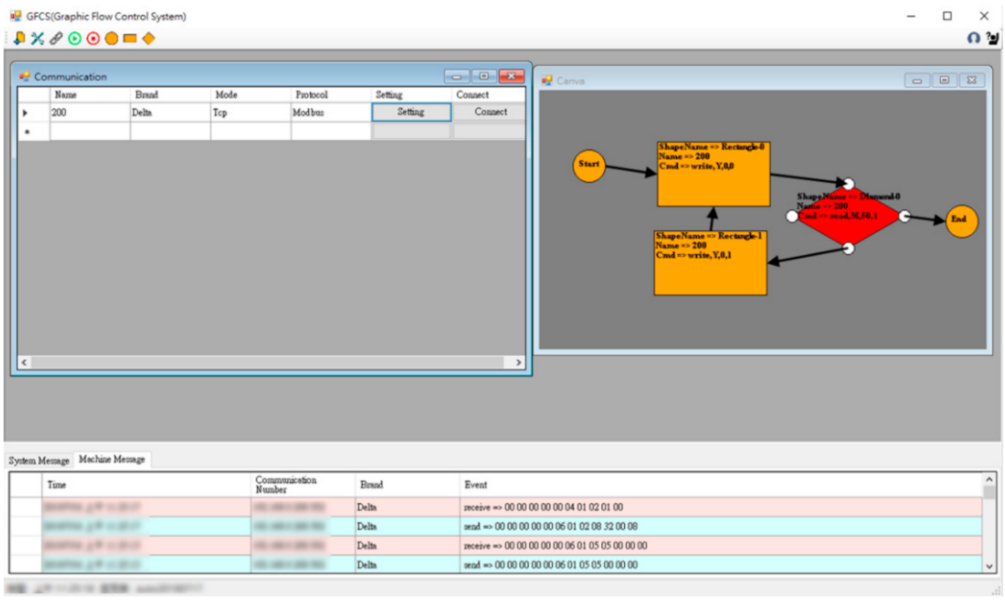
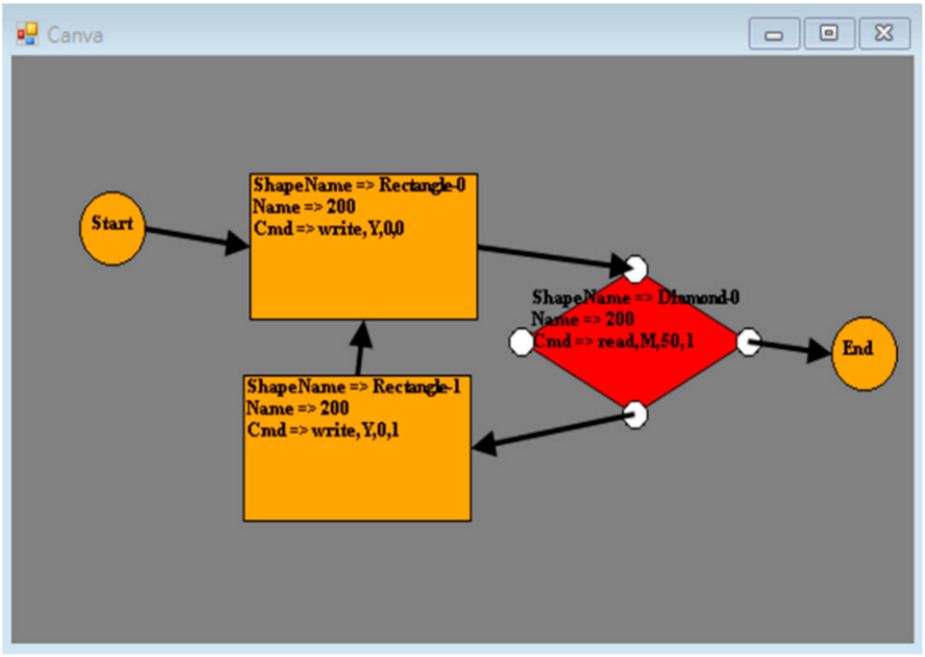


Figure 1. Smart production-line conceptual diagram.



(**a**)

Figure 2. *Cont*.



(**b**)

Figure 2. (**a**) GCS (graphic control system), (**b**) GCS flowchart.

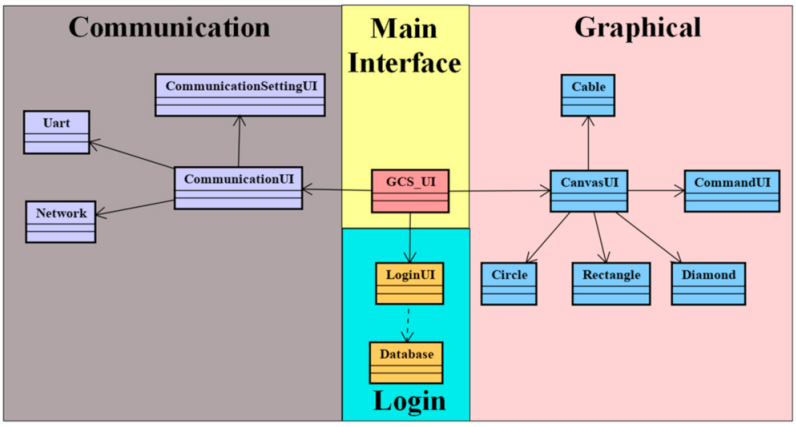


Figure 3. Class hierarchy.

2.1. Main Interface Area

A GCS_UI class is present in the main interface, and this class represents the main control interface of this system. It is responsible for interface management and system-information display of the other small systems.

2.2. Communication Area

The communication area is responsible for all communications of the operating equipment. This area is divided into four classes: Communication UI, CommunicationSettingUI, Uart, and Network. The area is dominated by Communication_UI, which can implement the communication-setting interface (CommunicationSettingUI), and a user can setup the communication parameters required for control. Then CommunicationUI implements the corresponding communication objectives according to the input parameters and stores them in an array, which is provided to the system when the system requires.

Currently, many communication methods are used in the industry. Famous factories and associations have also developed their own communication formats such as Profibus

(Siemens Co., Munich, Germany), EtherCAT (Beckhoff Co., Verl, Germany), Ethernet/IP (Rockwell Automation Co., Milwaukee, WI, USA), CC-Link (CC-Link Partner Association), and Profibus (Profibus Association). However, the current research focuses on the communication of existing equipment in traditional industries, including the communication methods such as TCP, UDP, RS232, or RS485. These methods can be divided into two transmission methods: Network and Uart, which are programmed in individual classes for use. If a device with another communication protocol needs to be controlled, the system can separately add this class.

2.3. Login Area

The login area is responsible for the operation restrictions of the system, and the corresponding operation restrictions are opened via an account permission level. According to its functionality, it is divided into two classes; namely, LoginUI and Database. In LoginUI, users can enter their account and password and wait for the login button to be pressed. After the button is pressed, LoginUI creates a database object and links it to the database for account verification to create system-operation management. In addition to protecting the user accounts and passwords, a verification code and data encryption in LoginUI is necessary to avoid data leakage due to brute-force cracking and packet interception.

2.4. Graphical Area

The graphical area is responsible for the sequence setting of the process and formulation of the control commands. The commands are packaged into functional blocks, and the control commands are sent according to the order of the user needs. Functionality is divided into graphical main interface (CanvasUI), command interface (CommandUI), connection line (Cable), and three functional blocks (Circle, Rectangle, and Diamond). The canvas can create different functional-block diagrams; namely, "Circle", which is responsible for the start and end of the process, "Rectangle", which is responsible for the output of the action commands, and "Diamond" to read and distinguish values. Using the aforementioned three function blocks, flow control can be established. The command is implemented on the function block. Users can enter the command for control on its interface and finally connect it through the cable according to the desired control order.

3. Communication Integration

"Communication" means that the computer or controller sends various data from the transmitting terminal to the receiving end via a communication device. Different types of machines can communicate with one another to achieve coordination or message transmission. Therefore, understanding the principles behind the communication technology is necessary so that the technology can be flexibly applied.

3.1. Communication Protocol

"Communication protocol" means that computers communicate among one another in the same manner. They all must follow certain guidelines and norms so that they can understand the meaning of each other in order to cooperate with one another to achieve a common task. Currently, one of the most commonly used automatic controllers in factories is the programmable logic controller (PLC). Therefore, the communication protocol selected by this research is "Modbus". The next section introduces this protocol. According to the coding method in Uart, this protocol can be divided into "Modbus ASCII" and "Modbus RTU". However, Modbus can also be sent via Ethernet, and other transmission format can be derived, e.g., "Modbus TCP". Although Modbus is divided into ASCII, RTU, and TCP, the formats are similar, with each composed of Start, Address, Function, Data, LRC, and END, as shown in Figure 4.

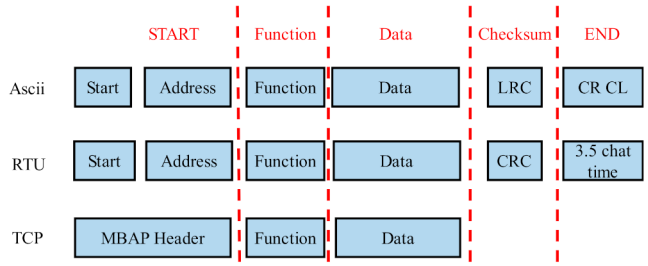


Figure 4. Modbus format.

3.2. Architecture Implementation

Architecture implementation is based on the architecture planned during the design phase, as shown in Figure 5. The communication and transcoding parts are tested in individual programs as classes, and their relationships with one another are established according to the plan. During the program execution, the required object is created according to the communication method selected by the user and equipment. Through cooperation among the objects, communication integration can be achieved.

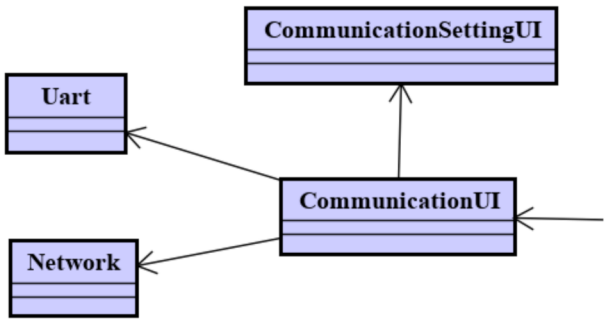


Figure 5. Communication-system architecture at the design stage.

3.3. Communication Integration, Implementation, and Discussion

During the communication integration processes, we must allow establishment of the preliminary design of the communication-class structure to be achieved. However, no transcoding function is available for communications in this structure. Therefore, the idea for modifying this problem is to add three different classes, such as AbsModbus, Delta, and Mitsubishi, as shown in Figure 6. First, users call out the communication-setting interface form (CommunicationSettionUI) via the main interface (CommunicationUI). Here the users can input the required communication parameters. According to this form, the user creates corresponding objects, such as Network or Uart, and transcoding objects (Delta or Mitsubishi). When communication is needed, the communication object calls the transcoding object as a parameter and then converts it to a correct command code. Finally, the commands are sent through the communication object.

Following the previous established structure, when a one-to-one communication structure is built, the required communication objects and transcoding can be implemented and allowed to cooperate with each other to achieve control of the message transmission. However, transmission cannot be correctly executed in a one-to-many communication structure. The reason is that many of the communication and transcoding objects are created, which can create object-matching difficulties among one another.

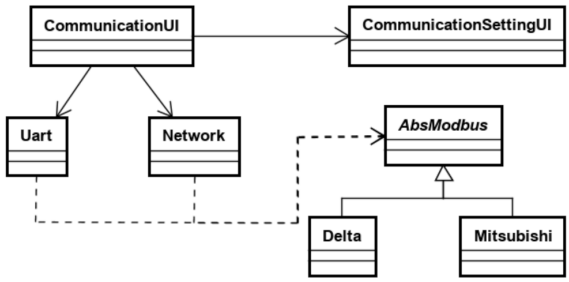


Figure 6. Communication-system architecture.

The CommunicationUI class can be used to record the processes of the object relationship, and it can call up the record to recognize the relative object. However, the programming of CommunicationUI can become more complicated. Therefore, few modifications have been made, and the improved structure is shown in Figure 7.

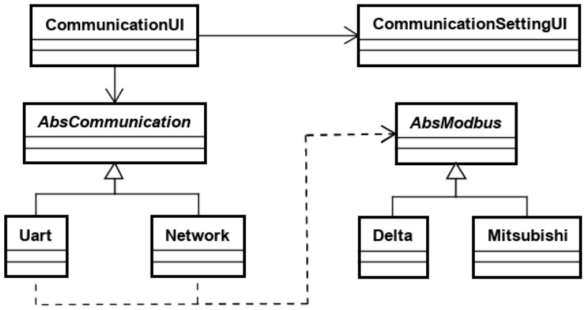


Figure 7. Improved communication-system architecture.

In the communication part, the AbsCommunication abstract class is adopted to define the basic communication format, i.e., connecting, disconnecting, sending, and receiving values.

The Uart and Network classes inherit the abstract class and create the functions that actually use these two communication methods, such as connecting, disconnecting, sending, and receiving values by the Uart and Network. Similarly, the abstract-classification method integrates different equipment-control protocol in the transcoding part.

4. GCS

A basic flowchart in the GCS can be constructed, as shown in Figure 8. According to the flowchart rules, the three common graphic functions in the GCS can be created. Among them, the circle represents the start and end processes, the rectangle represents a process step, and the diamond represents a decision-making process.

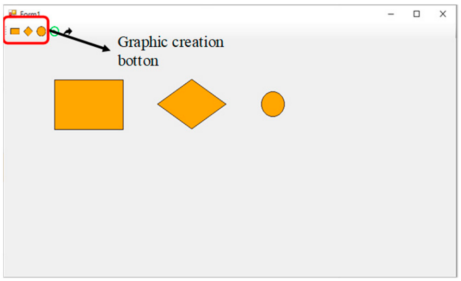


Figure 8. Graphic creation test.

4.1. Graphical Detection and Drag Test

The graphic drag test is shown in Figure 9. During the construction of graphics in GCS, the mouse-drag function must adjust the order of the function block. Therefore, the mouse-detection and dragging functions of the graphics are necessary. Finally, the correct graphics can be dragged to achieve the required control flow.

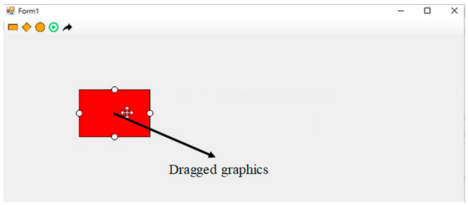


Figure 9. Graphic drag test.

4.2. Sequence-Control Test

In GCS, the control process depends on the arrow to move to the next step, as shown in Figure 10. The main purpose is to perform sequence control in automated production and thus allow the user to easily learn the operating steps via every graphic connection with the other graphics. The relevant graphic function turns to a red color to highlight the step currently being processed.

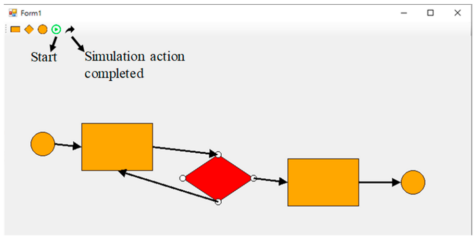


Figure 10. Sequence-control test.

4.3. Architecture Implementation

Graphical-control architecture implementation is established, as shown in Figure 11, according to the architecture planned during the design phase. The classes and their connections can be set up according to the action functions and planned structure. During the program execution, different graphics are created according to the actual requirements. Users can create graphic functions and related connections to control the production line according to the designed program flow.

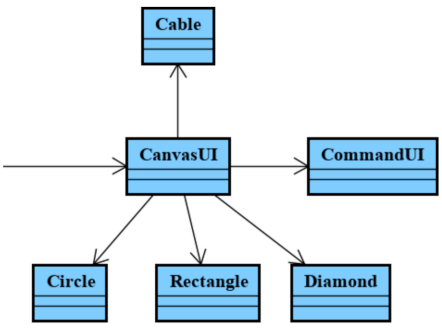


Figure 11. Graphical-control architecture during the design phase.

4.4. Integration Implementation and Discussion

When the GCS is programmed, it must follow the classes designed during the design stage. In addition, the actual process of this framework must follow the open–closed

principle. According to the architecture shown in Figure 11, when the process function adds new requirements, the user must add more conditional operators in CanvasUI to ensure that the system can respond to the correct graphics, which may appear easy for coding but the program becomes more complicated. Moreover, when the function of a certain class is used, class decisions among many similar classes need to be performed. Once the number of classes increases, the computation burdens become higher. In a high-coupling case among classes, many bugs are easily generated when the program is modified. Generally, this problem can be solved using the "Abstract" method. The use of an abstract feature that only defines the function format but not the processing method can clearly define the function format of the graphics. Once a new graphic needs to be added, it can directly inherit the class and perform the function call. The corresponding class can be called without determining the type. Therefore, an additional abstract class, namely, AbsShape, is adopted, and the graphic control at the design stage is modified, as shown in Figure 12.

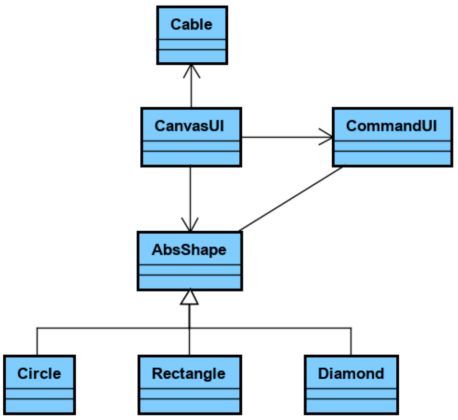


Figure 12. GCS architecture modification.

5. Intelligent Automatic Production-Line Control System

Following the trend in Industry 4.0 and smart manufacturing, this intelligent production line has been developed in our laboratory to realize communication integration, data collection, and visualization. It is made up of five equipment workstations: Feed Station, Laser-Engraving Station, Automated Optical Inspection Station, Gluing Station, and Assembly and Warehousing Station (Figure 13). A luxometer was adopted as the IoT sensor in the redbox in Figure 13 to ensure that the visual inspection was correct. If other sensors are needed, such as luminosity, temperature, color, pressure, speed, etc., in the same way, the sensor data will be collected and displayed in the IAPLCS.

The web interface of this system was designed using Laravel Framework. For security reasons, users must sign up and be authorized before accessing this system (Figure 14). The information about this production line is displayed on the screen according to the function options selected by the user, including data upload, control interface, utilization, IoT monitoring, and graphical interface. Figure 15 shows that the data-upload function facilitates either creating, editing, or deleting the filename or path of a new project script. Figure 16 shows that starting or stopping the selected project is controlled by the control interface, which is used to schedule the production line or adjust the condition of the production line. To quickly improve efficiency and better understand the operation status of the entire production line, the working time of each workstation was converted into utilization rate and cycle time and recorded on the utilization-rate display interface (Figure 17). Figure 18 shows that IoT sensors are installed on site, and the data are uploaded via a Wi-Fi network and directly transferred to the database for further analysis. The IoT data are also visualized in the graphic interface for the users to receive the real-time IoT situation (Figure 19). To track the machines' performance across the factory floor or to solve problems promptly, the remote real-time monitoring system is well-established on the interface (Figure 20).

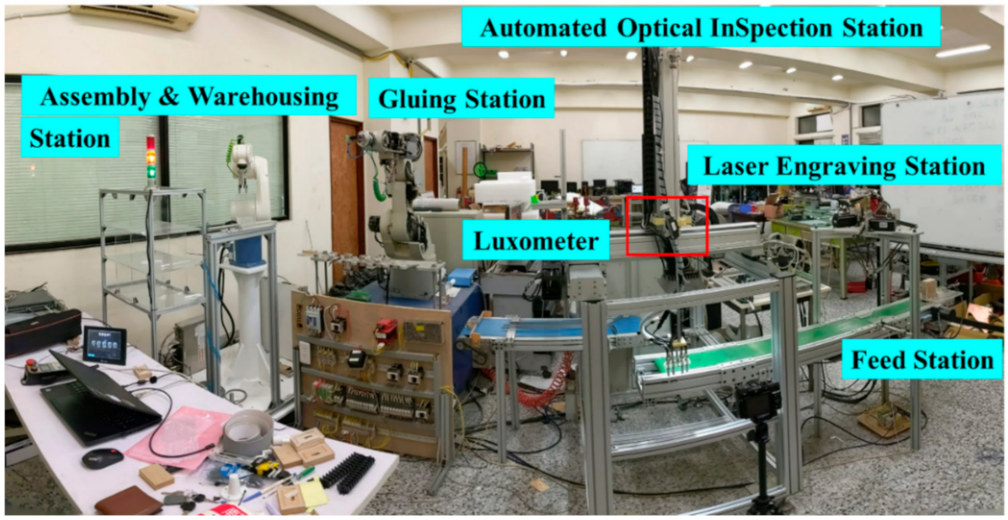


Figure 13. Real production line.

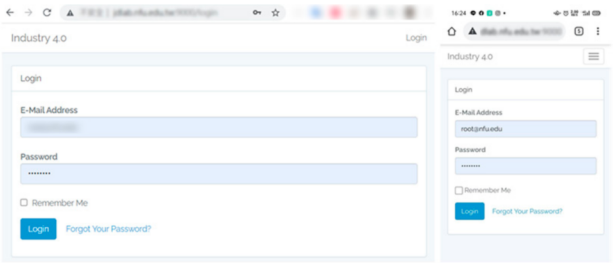


Figure 14. Login permission.

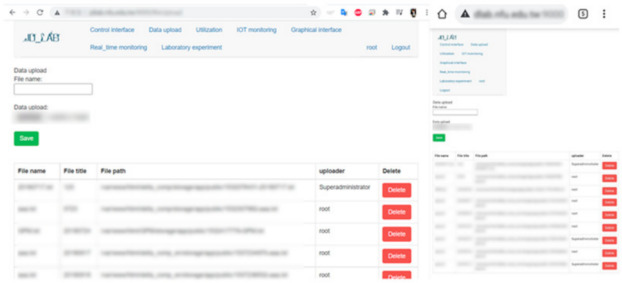


Figure 15. File-upload interface.

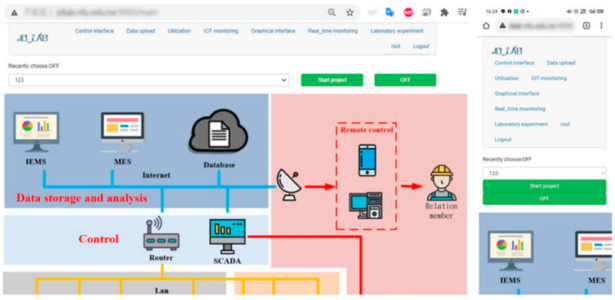


Figure 16. Project-control interface.

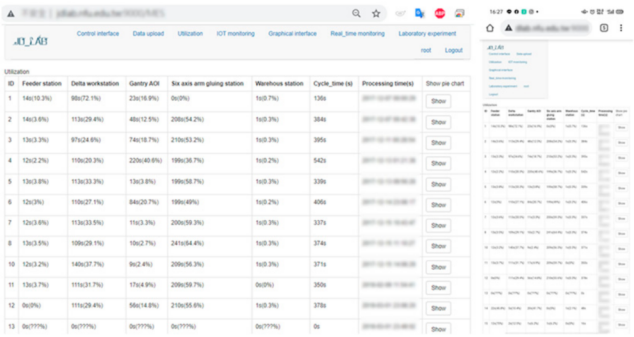


Figure 17. Utilization-rate display interface.

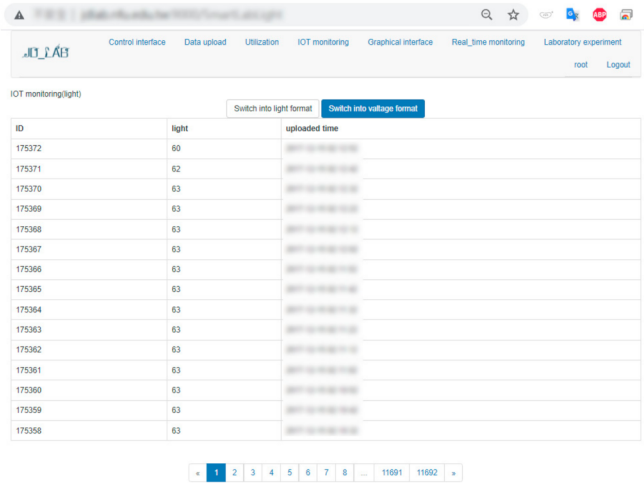


Figure 18. IoT sensor data-collection interface.

Figure 19. IoT graphical interface.

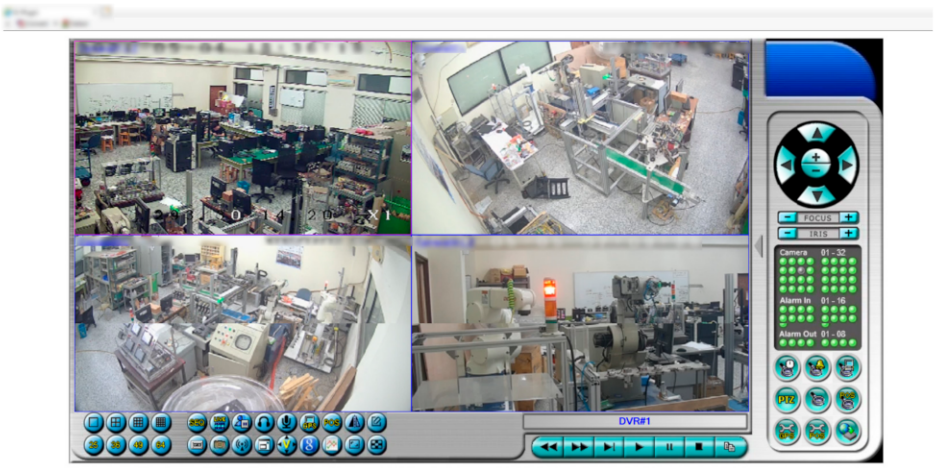


Figure 20. Remote monitoring interface.

6. Integration and Experiment

In the experiment, the automatic production line, human machine interface (HMI), and warning lights with different generally used communication methods, including RS232, RS485, and TCP, were controlled by IAPLCS. Figure 21 shows the experimental-system architecture diagram. An individual PLC sends the command to HMI, production line, and warning light when a recommendation is received from GCS. The user only changes the GCS graphical interface process, which automatically satisfies effective changing of the production schedule.

According to previous tests in the design stage, a class diagram can be reconstructed, as shown in Figure 22. GCS_UI is used to control other small system interfaces. Then, the small system interfaces are used to control different classes of function. They can separate a complicated system into several structures and allow easier system maintenance or upgrading. Moreover, public static classes are added for communication between the two systems introduced earlier. Communication UI saves the established AbsCommunication in Collect and prepares it for the other systems to use. When the system operating parameters are all set, GCS_UI calls CanvasUI to start graphical control after the start button is pressed. According to the requirements of the program, the communication parameters are set by CommunicationSettingUI, and the corresponding action signals are extracted from Collect and sent to AbsCommunication for transcoding, transmission, return comparison, and other functions. AbsCommunication is composed of the following four communication methods: UartModbus, NetworkModbus, Uart, and Network. In addition, new communication methods can also be expanded, such as CC-link, Devicenet, etc. We have also built a new AbsModbus, which includes devices from two manufacturers, Mitsubishi and Delta, which are used to process Modbus-related communication protocols. Besides this, devices with Modbus communication protocol and different data addresses can also be expanded according to the architecture shown in Figure 22.

The central control program (CCP) interface is shown in Figure 23. After CCP is opened, the user must first set some basic communication parameters, connect the device to the IP and corresponding port, and click the automatic function button after completion. CCP then automatically runs and connects to the server and PLC. After the connection is successful, it waits for the start signal from the web page. When the remote start signal is received, the subprogram is started. In the subprogram, the project schedule is downloaded from the file transfer protocol (FTP) server and loaded into the execution window. CCP reads the project command code in the execution window and converts it into Modbus TCP to control the PLC. During this process, it records each command and response time. The utilization rate can be calculated using these data and transmitted to the database.

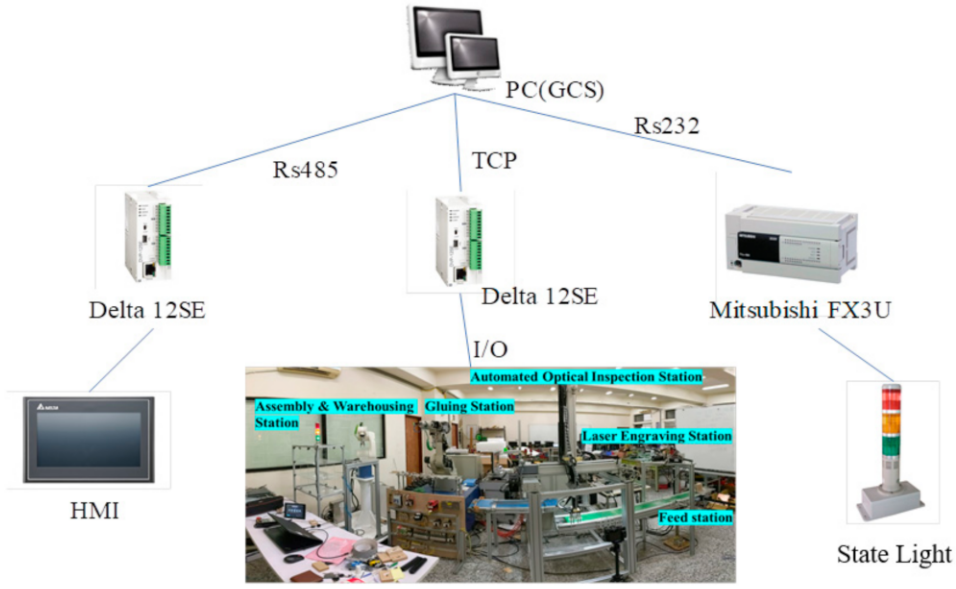


Figure 21. System architecture.

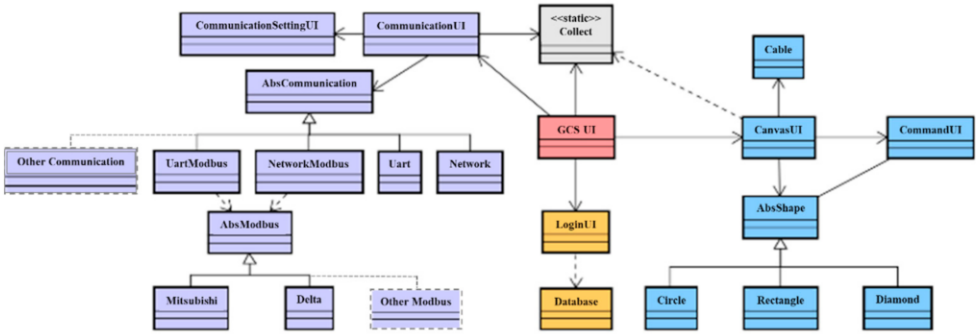


Figure 22. Class hierarchy of the final system.

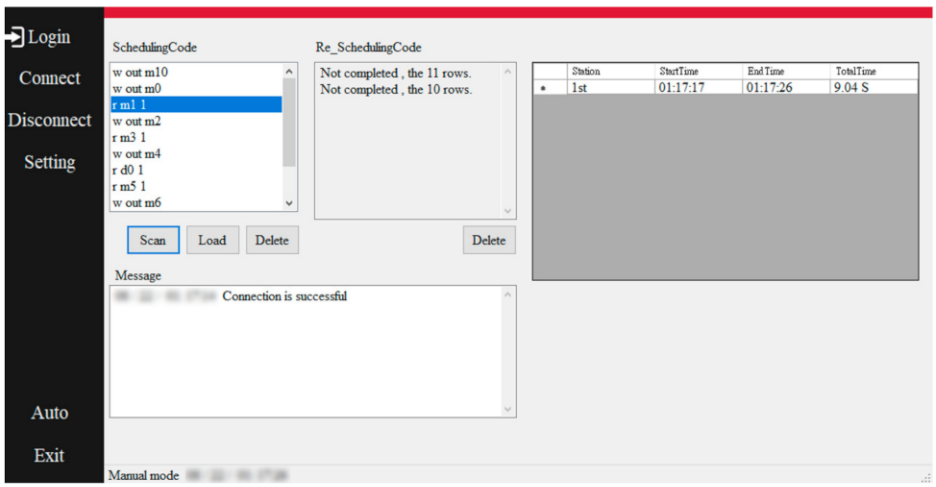


Figure 23. Central control program interface.

As previously mentioned, through the web-page production, graphic control design, communication-system structure, and CCP, IAPLCS is established, and this system can be applied to the self-designed production line. The production-line layout is shown in

Figure 24. The material list and final products are shown in Figure 25. Figure 26a shows the first station, i.e., the Feeding Station. When the production line starts, Base Material A is automatically conveyed from the Feeding Station to the Laser-Engraving Station, which is the second station. The laser-engraving machine engraves the uploaded and scanned picture to the control interface in Base Material A. Figure 26b shows the third station, i.e., the Automated Optical Inspection Station, which is mainly used for inspection of the workpiece to determine whether the product is well-produced. Figure 26c shows the fourth station, i.e., the Gluing System. After Base Material A is laser-engraved, it is allocated for subsequent assembly operations. Figure 26d shows the fifth station, which is the Assembly and Warehousing Station. On the assembly platform, Cover Material B is automatically picked up and assembled onto Base Material A. After this process, the product is reserved for storage.

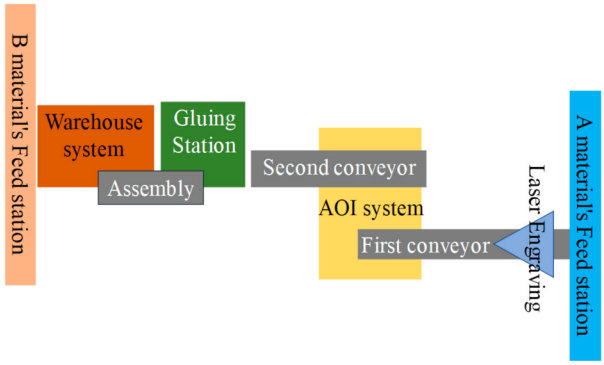


Figure 24. The 2D production-line layout.

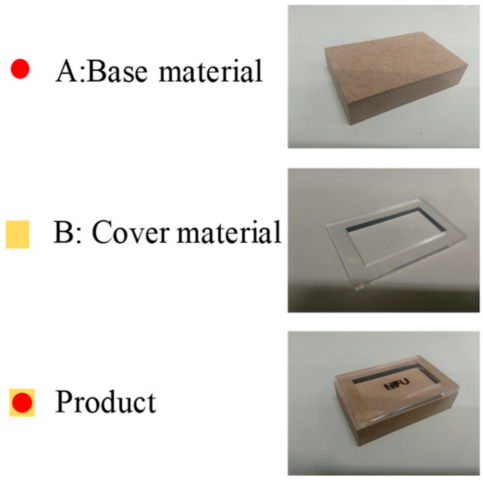


Figure 25. Materials.

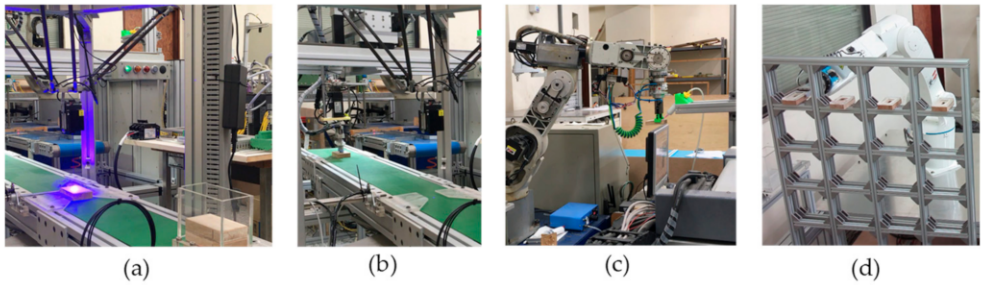


Figure 26. (**a**). Stations 1 and 2. (**b**) Station 3. (**c**) Station 4. (**d**) Station 5.

7. Discussion

With the development of Industry 4.0, production in the factories will be more flexible and must be more intelligent. Related research topics about automated production control systems [18]; automation of product, design, management, and control systems [2]; intelligent manufacturing [20,21]; and CPS [3,6] have been proposed. Whether the research topic is intelligent manufacturing, CPS, or automated production control systems, most people tend to study methods and introductions and do less with the combination and application of the actual production line. However, in this study, actual automation equipment had been successfully integrated in a production line. The system has a flexible and modular communication integration architecture. Users only need to add corresponding communication protocols in the communication architecture to complete the expansion, data acquisition, and visualization of new devices and sensors. In addition, GCS greatly enhances the convenience of user operations and makes the workflow smoother and more convenient.

8. Conclusions

This study has successfully developed IAPLCS and applied it to an actual automated production. GCS, which controls the production line, is divided into multiple classes according to the employed functions, and the communication and commands of the controlled equipment are all included. IAPLCS allows the system to be easily upgraded and can reduce the effect caused by machine replacement or adjustment of the working-process order. In GCS, many complicated operating procedures are packaged into functional blocks, and they are arranged in order according to the designed control process. The connection line represents the relationship among the modules and allows users to more clearly understand and modify the control process. Finally, IAPLCS has been successfully implemented to control an actual production line to complete a customized production. In addition to the ease in use of GCS to change the control process, users can control the production line and obtain production information, including the utilization rate, IoT sensor data, and real-time production information through the internet. In this study, the whole software, hardware and system integration are self-developed. Different equipment can be integrated easily in IAPLCS to be used in different actual production line applications.

Author Contributions: Conceptualization, J.-D.L.; Methodology, J.-D.L. and H.-Y.H.; Software, H.-Y.H.; validation, C.-Y.L. and J.-Y.Y.; Writing—original draft, J.-D.L., C.-Y.L. and J.-Y.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Ministry of Science and Technology of Taiwan under grant number MOST-110-2221-E-150-037 (form August 2021 to July 2021).

Data Availability Statement: The author at the Mechatronics and Automation Laboratory, located at the Department of Automation Engineering, National Formosa University, Taiwan, was the subject of the experiments. The author consented to participate in this research study.

Acknowledgments: The authors would like to acknowledge the financial support of the Ministry of Science and Technology of Taiwan, R.O.C. through grant number MOST 110-2221-E-150-037.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Explore the Application of CPS and Cloud Networking in Smart Manufacturing. Available online: https://mymkc.com/article/content/22456 (accessed on 22 September 2021).
2. Andronie, M.; Lăzăroiu, G.; Ștefănescu, R.; Uță, C.; Dijmărescu, I. Sustainable, Smart, and Sensing Technologies for Cyber-Physical Manufacturing Systems: A Systematic Literature Review. Sustainability. *Symmetry* **2021**, *13*, 5495.
3. Zeadally, S.; Sanislav, T.; Mois, G.D. Self-adaptation techniques in cyber-physical systems (CPSs). *IEEE Access* **2019**, *7*, 171126–171139. [CrossRef]
4. Möstl, M.; Schlatow, J.; Ernst, R.; Dutt, N.; Nassar, A.; Rahmani, A.; Herkersdorf, A. Platform-centric self-awareness as a key enabler for controlling changes in CPS. *P IEEE* **2018**, *106*, 1543–1567. [CrossRef]

5. Keung, K.L.; Lee, C.K.; Ji, P.; Ng, K.K. Cloud-based cyber-physical robotic mobile fulfillment systems: A case study of collision avoidance. *IEEE Access* **2020**, *8*, 89318–89336. [CrossRef]
6. Canizo, M.; Conde, A.; Charramendieta, S.; Minon, R.; Cid-Fuentes, R.G.; Onieva, E. Implementation of a large-scale platform for cyber-physical system real-time monitoring. *IEEE Access* **2019**, *7*, 52455–52466. [CrossRef]
7. Siemens' Super Money Printing Machine. Available online: https://www.cw.com.tw/article/5063557 (accessed on 22 September 2021).
8. Huang, Y.; Zhang, Y.; Xiao, H. Multi-robot system task allocation mechanism for smart factory. In Proceedings of the 2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, 24–26 May 2019.
9. Hong, Z. Smart Manufacturing System Based on Industry 3.5—The Empirical Study of TFT-LCD Array Dynamic Scheduling and Dispatching. Doctoral Dissertation, National Tsing Hua University, Hsinchu City, Taiwan, 2019.
10. Kobzan, T.; Heymann, S.; Schriegel, S.; Jasperneite, J. Utilizing SDN infrastructure to provide smart services from the factory to the cloud. In Proceedings of the2019 15th IEEE International Workshop on Factory Communication Systems (WFCS), Sundsvall, Sweden, 27–29 May 2019.
11. Ozdemir, R.; Koc, M. A quality control application on a smart factory prototype using deep learning methods. In Proceedings of the2019 IEEE CSIT, Lviv, Ukraine, 17–20 September 2019.
12. Heymann, S.; Stojanovci, L.; Watson, K.; Nam, S.; Song, B.; Gschossmann, H.; Jasperneite, J. Cloud-based plug and work architecture of the IIC testbed smart factory web. In Proceedings of the 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), Torino, Italy, 4–7 September 2018.
13. Ahn, D.J.; Jeong, J.; Lee, S. A novel cloud-based fog computing network architecture for smart factory big data applications. In Proceedings of the 2018 South-Eastern European Design Automation, Computer Engineering, Computer Networks and Society Media Conference (SEEDA_CECNSM), Kastoria, Greece, 22–24 September 2018.
14. Zhao, H.; Hou, J. Design concerns for industrial big data system in the smart factory domain: From product lifecycle view. In Proceedings of the 2018 23rd International Conference on Engineering of Complex Computer Systems (ICECCS), Melbourne, VIC, Australia, 12–14 December 2018.
15. Durocher, D.B.; Hussey, M.R. Advancing technology in upgrading or replacing vintage low-voltage motor control centers. *IEEE Trans. Ind. Appl.* **2019**, *55*, 6576–6584. [CrossRef]
16. Wei, L.; Min, L. China's tourism industrial upgrading and transformation strategies and countermeasures in the background of big data. In Proceedings of the 2020 International Conference on Big Data and Informatization Education (ICBDIE), Zhangjiajie, China, 23–25 April 2020.
17. Rivera, E.; Vasquez, R. A phased paper machine drive & control system upgrade. In Proceedings of the 2018 IEEE IAS Pulp, Paper and Forest Industries Conference (PPFIC), Melbourne, Australia, 12–14 December 2018.
18. Pisarić, M.; Dimitrieski, V.; Vještica, M.; Krajoski, G.; Kapetina, M. Towards a Flexible Smart Factory with a Dynamic Resource Orchestration. *Appl. Sci.* **2021**, *11*, 7956. [CrossRef]
19. Ji, S.; Lee, S.; Yoo, S.; Suh, I.; Kwon, I.; Park, F.C.; Kim, H. Learning-Based Automation of Robotic Assembly for Smart Manufacturing. *P IEEE* **2021**, *109*, 423–440. [CrossRef]
20. Kim, B.S.; Jin, Y.; Nam, S. An integrative user-level customized modeling and simulation environment for smart manufacturing. *IEEE Access* **2019**, *7*, 186637–186645. [CrossRef]
21. Patalas-Maliszewska, J.; Halikowski, D. A model for generating workplace procedures using a CNN-SVM architecture. *Symmetry* **2019**, *11*, 1151. [CrossRef]