Future Generation Computer Systems 148 (2023) 139–149

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Concurrent experimentation in NCSLab: A scalable approach for online laboratories

Zhongcheng Lei [a], Hong Zhou [a], Wenshan Hu [a,*], Guo-Ping Liu [b]

[a] Department of Artificial Intelligence and Automation, Wuhan University, Wuhan 430072, China
[b] Center for Control Science and Technology, Southern University of Science and Technology, Shenzhen 518055, China

ARTICLE INFO

Article history:
Received 12 January 2023
Received in revised form 18 May 2023
Accepted 20 May 2023
Available online 27 May 2023

Keywords:
Concurrent experimentation
Massive access
Front-end and back-end separation
Online laboratories
Education

ABSTRACT

The scheduling of user access to test rigs is vital for online laboratories, as a good access scheduling scheme can reduce waiting time and improve the utilization of experimentation systems. To enhance the benefits of online laboratories and also improve user experience, a novel system that is constructed based on front-end and back-end separation has been explored in this paper, which is scalable with a complex four-tier architecture supporting both remote and virtual experiments. A concurrent scheme that allows many users to conduct real-time interactive experiments with a single test rig has been implemented. The new system enables massive access for virtual experimentation and saves users from booking sessions in advance or queuing. To verify the effectiveness of the system, a renewable energy system has been developed, thus, users can concurrently conduct experiments with different control algorithms and monitoring interfaces for research and education purposes. A scalability test was conducted, the results of which showed that 94.67% of 300 users can concurrently access the same virtual laboratory, proving the scalability and concurrency features of the proposed system. The system demonstrated a throughput of 499.4 requests per second without errors during a 30-second performance test with 1600 concurrent users, where each user established 10 connections.

1. Introduction

Online laboratories (remote, virtual, hybrid, digitized laboratories, *etc.*) play a key role in science, technology, engineering, and mathematics (STEM) education, which can promote inquiry learning and enhance problem-solving skills [1–4]. During the past two decades, numerous online laboratories have been set up by researchers throughout the world [5–7]. For example, many remote laboratories have focused on industrial electronics education [8,9]. In [8], remote laboratory applications for engineering research and education were reviewed, which includes large-scale effective application examples in different areas such as electronics and microelectronics, power electronics and electrical drives, and control and automation. In [9], learning methodologies and tools were presented, and remote and virtual laboratories for industrial electronics education were discussed. Even after a decade, the two studies still provide additional insights for remote and virtual laboratory development and application.

The merits of online laboratories that attract the attention of researchers can be divided into two categories:

(1) Merits over other online learning methods [10]. Although online learning methods are trying to promote online education, other online learning methods such as MOOCs (Massive Open Online Courses), can disseminate documentations and tutorials in video forms, while concepts and formulas are still difficult to comprehend thoroughly [11]. As an emerging online learning method, online laboratories are able to share equipment for online experimentation and help students foster experimental skills [12].

(2) Merits over conventional laboratories [13,14]. In conventional laboratories, users need to perform experiments on a specific time and location with costly equipment, which has a relatively low rate of utilization and requires high maintenance cost. While online laboratories offer portability, easy-access, and cost-effectiveness regardless of users' locations and time as long as the Internet is available. For example, during the COVID-19 pandemic, online laboratories can play an important role in engineering education when conventional laboratories are inaccessible due to campus closure [12,15,16].

Therefore, online laboratories can be a complement to both online learning and conventional hands-on laboratories. Well-established online laboratories are able to constitute a crucial part of education without the loss of any level of knowledge [1].

In online laboratories, user access to test rigs is a crucial factor for allowing more users to conduct experiments, which can reduce waiting time and improve the utilization of experimentation systems. Once access has been granted to a specific user, he/she

* Corresponding author.
E-mail address: wenshan.hu@whu.edu.cn (W. Hu).

https://doi.org/10.1016/j.future.2023.05.014
0167-739X/© 2023 Published by Elsevier B.V.

can control over the test rig and conduct experiments in a given period of time. Queuing [16–18] and booking [19–21] are two effective approaches for managing user access to test rigs. However, they lack support for concurrent experimentation, which can impede the application of online laboratories. This limitation can be particularly problematic in scenarios such as classroom teaching and demonstrations, where multiple users may need to conduct experiments with the same test rig simultaneously. For instance, if a teacher is demonstrating an experiment to enhance theoretical knowledge and wants students to follow the example, concurrent experimentation is necessary. Therefore, it is essential to develop a scheduling scheme that enables scalable and concurrent experimentation in online laboratories.

This paper discusses the scheduling scheme that supports scalability and concurrency experimentation and details the design and implementation from the front-end, server, controller and test rig perspectives. A newly developed wind renewable system is developed to demonstrate a case study of concurrent experimentation. The scalability feature is also verified through the concurrent access of 300 virtual students.

The rest of the paper is organized as follows: Section 2 presents a discussion of related work; Section 3 explores the novel four-tier architecture of the system; Section 4 investigates the implementation of the proposed laboratory, in which the front-end technologies, server cluster, controller deployment, and test rig integration are discussed. In Section 5, the concurrent solution that allows for massive access to virtual test rigs is provided, and the corresponding technologies are explored in detail. Section 6 discusses two case studies, and the evaluation and analysis of the new system are presented in Section 7. Section 8 concludes the paper.

2. Related work

Online laboratories have become a crucial tool for facilitating student involvement in STEM education and providing accessible laboratory resources. Conventionally, online laboratories provide one-to-one (1:1) access, which means that each test rig can only be accessed by one user at a time. Thus, to efficiently allocate the test rig resources and guarantee scalability [22], many access mechanisms, such as reservation or booking [19–21,23,24] and queuing [10,16–18] have been designed and implemented. In [25], the different scheduling schemes used by remote laboratories in the literature were analyzed, including booking, queuing, and a mixture of them, which also suits the current context of online laboratories.

Booking systems require users to book a time slot in advance for their experimentation, which have been widely used in online laboratories and enable the scheduling of experimental resources and managing user experiment sessions. For example, NetLab's integrated booking system allows users to book a time session for experiments [19]; GOLDi (Grid of Online Lab Devices Ilmenau) is based on HTML5, and both remote and virtual experiments have been developed. If experiments are not available, the user has to reserve experiments using the booking management system [26]. Apart from booking systems, the queuing mechanism that follows a first-come first-served (FCFS) rule has been developed. Compared to booking systems, the queue mechanism does not need a reservation, so users can conduct experiments immediately or wait in a queue [10,27,28]. To improve the utilization level of remote laboratory resources and decrease waiting time, a combination of queuing and calendar-based booking was proposed in [29,30] to support both modes for gaining access.

The other online laboratory examples are listed in Table 1, in which the access scheduling schemes of test rigs are provided and Apache refers to the Apache HTTP (Hypertext Transfer Protocol) server. It is worth noting that Table 1 is not intended to be exhaustive or complete, but rather to provide a representative sample.

Without the implementation of a scheme that allows users to conduct experiments with the same test rig simultaneously, booking systems or queuing are feasible solutions for experiment scheduling and user management. However, the limitations of booking or queuing are also obvious. For example, the booking time is limited to three hours per week for NetLab [19]. Additionally, booking requires an extra workload for users. For queuing, the user can only experiment with a preferred test rig when it is idle.

In recent years, digitized laboratories that turn existing experiments into datasets and subsequently display them on the iLabs web platform have been proposed, which can provide massively scalable online laboratories for experimentation [3,31,32]. In [33, 34], a method that employs a smart device [35] to integrate a remote laboratory into a MOOC platform was proposed, thus, a ratio of approximately 5–10:1 access was achieved through three actions: short activity time, controller/observer mode, queuing and equipment duplication. It not only requires modification of remote laboratories such that it fits in edX (a MOOC platform), but also requires reducing the experiment duration time for a single user (30 s to a few minutes). For an interactive experiment that needs 20 min to several hours to execute [36], the aforementioned scheme may not be suitable.

To enhance the functionalities and user experience while maintaining features such as plug-in free, web-based architecture, Hypertext Transfer Protocol Secure (HTTPS) security considerations, a hybrid framework, interactive features and three-dimensional (3-D) virtual test rigs, a new Networked Control System Laboratory (NCSLab) online laboratory with a scalable structure has been developed, the detailed design and implementation of which was explored in [12]. As shown in Table 1, the previous version of the NCSLab adopted a queuing scheme for test rig scheduling, thus, one test rig can only be accessed by a single user at a time. Concurrent access to virtual test rigs has been proposed in this paper for virtual experimentation, which can achieve a ratio of x:1 (many-to-one), where x can be easily modified from the administrator interface or the database. As long as the hardware supports more concurrent users, x can be set to larger. From the users' perspective, concurrent experimentation allows them to conduct experiments simultaneously and at their own pace, providing greater flexibility and control over their experiment sessions.

3. Novel architecture of the implemented laboratory

Based on the existing NCSLab framework [17], basic functionalities of the previous system have been achieved in the newly developed system, the system architecture of which is more efficient for developers and can provide better user experience for users.

During the design and implementation of remote laboratories, various technologies and architectures have been adopted by different developers. To support an easy-to-use and flexible framework, the developed NCSLab is based on the component-based React, which is currently the most famous front-end framework and has gained widespread acceptance. A reverse proxy server based on NGINX, along with a web server based on NGINX and PHP (Hypertext Preprocessor) FastCGI (Fast Common Gateway Interface), constitute the back-end of the system, which will be covered in more detail in the section that follows. Fig. 1 shows the web page of the new NCSLab experimentation system, which takes a test rig interface as an example.

Table 1
Online laboratory example list.

Laboratory name	Website	Access scheduling	Laboratory type	Server-side technology	Client-side technology
NetLab	http://netlab.unisa.edu.au	Booking	Remote	LABVIEW	Java
Next-Lab	https://nextlab.golabz.eu	Booking	Remote	PHP&Apache	Drupal, Graasp, *etc.*
VISIR	http://openlabs.bth.se/el-ectronics	Booking	Remote	Apache&C++&LABVIEW	PHP&HTML5
GOLDi	https://www.goldi-labs.net/	Booking	Remote&Virtual	PHP&Apache	HTML5
Labshare Sahara	https://remotelabs.eng.u-ts.edu.au	Booking&Queuing	Remote&Virtual	PHP&Apache	Java
WebLab-Deusto	https://weblab.deusto.es/	Queuing	Remote&Virtual	Python&NGINX	AngularJS, Unity3D, WebGL
LabsLand	https://labsland.com	Queuing	Remote	Apache	AngularJS
iLab	http://icampus.mit.edu/projects/ilabs/	Booking&Queuing	Remote	LABVIEW	Java
Remotelab	https://remotelab.fe.up.pt	Booking	Remote&Virtual	PHP&Apache	LABVIEW
UNILabs	https://unilabs.dia.uned.es	Booking	Remote&Virtual	PHP&Apache	Easy Java/Javascript Simulations
Old NCSLab	https://www.powersim.whu.edu.cn/ncslab/	Queuing	Remote&Virtual	JSP&Java&Tomcat	YUI&HTML5
Proposed	https://www.powersim.whu.edu.cn/react/	Concurrent&Queuing	Remote&Virtual	PHP&NGINX	React&HTML5

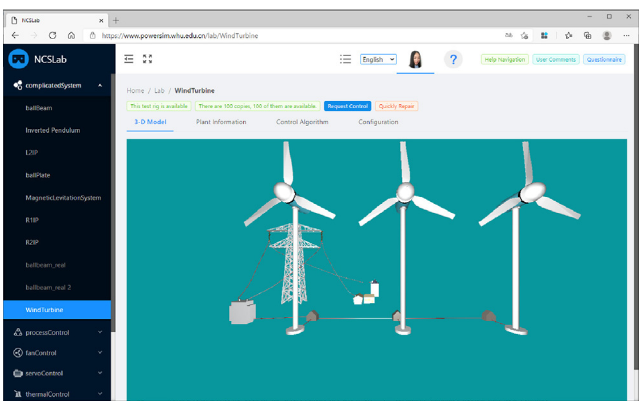


Fig. 1. Web page of the NCSLab experimentation system.

3.1. Four-tier architecture

As discussed in [17], the NCSLab has evolved through four major versions. This new version has been developed and put online at https://www.powersim.whu.edu.cn/react, which is completely different from previous versions in the following aspects: the new version adopts a new front-end framework and the front-end and back-end separation solution for development, which forms a complex architecture with different technologies and different user interfaces. The complex architecture is illustrated in Fig. 2, which can also be concluded into a four-tier architecture as previously discussed in [10] but with richer contents in each tier. The details of the four-tier architecture include the user tier where users use web browsers to access the system, the server cluster tier where six different types of servers are deployed to support experimental services such as parameter tuning and real-time monitoring and control, the controller tier where Windows-based controllers are deployed and the test rig tier where virtual and physical test rigs are integrated. A detailed description will be included in Section 4.

3.2. Front-end and back-end separation framework

Regarding the complex architecture presented in Section 3.1, a front-end and back-end separation approach has been adopted, which has become a common practice [37,38]. This approach enables the front-end and the back-end to be loosely coupled and relatively independent from each other. The merits of the front-end and back-end separation approach are as follows:

(1) For users, a more responsive and dynamic user interface can be provided, as a front-end and back-end separation approach allows for the optimization of each part for its specific function. For example, with React as the front-end technology, a better user experience can be provided since the whole page is not updated with every request.
(2) For developers and researchers, this approach offers a clear roadmap for front-end and back-end development regarding system implementation, which can improve scalability and require less workload for future system updates and upgrades.

In the front-end and back-end separation scheme, the front-end is responsible for processing resources that comprise the user interface (UI) in the web browser, such as HTML, CSS, and JavaScript files. These resources define the layout and appearance of the web application and handle user interactions. The back-end, on the other hand, primarily manages the logic of servers such as the web server and the file server, as well as parsing and processing application programming interfaces (APIs) and performing create, read, update and delete (CRUD) operations on the database.

3.3. Technical summary review and new features

Previous work of the NCSLab has provided accessible and available experimental services from a technical perspective [17]. Previous features include cross platform and universality, graphical interactivity, user customized monitoring interface, secured communication and nonintrusive access and client-side image capture.

In addition to the above mentioned features that have also been implemented in the new system, new features are developed to ensure a more powerful system.

(1) *Concurrent experimentation for virtual test rigs*: Theoretically, a controller can supply as much concurrent access to a virtual test rig limited by the controller capacity, for example, hardware limitations.
(2) *Live video streaming for physical test rigs*: Instead of the previous image refreshing scheme [17], live video streaming has been achieved for better monitoring experience and audio presentation for physical test rigs in NCSLab. The

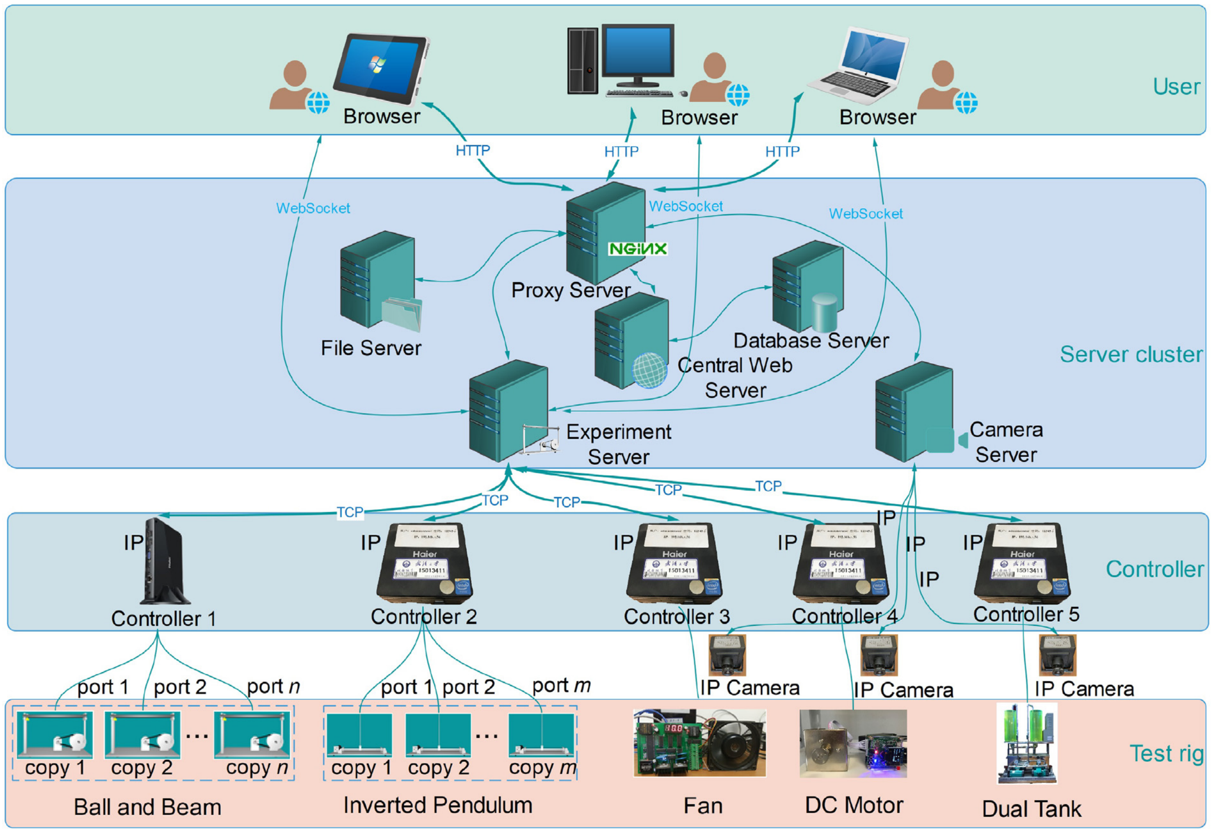


Fig. 2. Architecture of the React-based NCSLab experimentation system.

scheme is achieved based on an open source project that is easy to deploy.

(3) *WebSocket communication for algorithm downloading and real-time experimentation*: The WebSocket protocol is an HTML5 protocol that enables Web client applications to maintain bidirectional communications with server-side processes with low latency, in which the server can actively push data to the client.

4. Implementation of the laboratory

In this section, the detailed design and implementation of the proposed laboratory regarding the perspective of the four-tier architecture are investigated.

4.1. Front-end implementation and UI

Considering that the previous Yahoo UI (YUI) framework is out of date and is no longer actively maintained [39], React, which is an open source mainstream front-end framework, has been selected as the framework for the new NCSLab experimentation system. Currently, React is the most popular front-end framework throughout the world [40], which provides comprehensive documentation, informative tutorials, and community support. Similar to the previous version of the NCSLab based on YUI, the newly developed system is also based on HTML5 and shares features such as cross-platform, graphical interactivity and user customized monitoring interface, which means that users can use any mainstream web browser to conduct experiments free from plug-ins. Moreover, users can also customize their monitoring interface using the provided widgets. The following describes the front-end implementations in detail.

4.1.1. Front-end architecture and used technologies

To construct such a complex architecture that enables a higher level of autonomy for experimentation, many open source technologies are used and seamlessly integrated together to serve the purposes.

The adopted technologies are described as follows:

(1) React [41]: React, a popular front-end JavaScript library for building UIs, was chosen as the main framework for developing the experimentation system.

(2) AntD [42]: A React UI library developed by Alibaba, which provides a collection of reusable UI components for interactive UI design and implementation.

(3) Three.js [43]: Three.js is a lightweight 3-D library for 3-D content rendering on a webpage, which has already been used for 3-D rendering in the NCSLab system [10,17]. As three.js is powerful, open source, easy-to-use and supports web-based interactive 3-D rendering, it continues to be adopted as the 3-D model rendering engine.

(4) WebRTC (Web Real-Time Communications) [44]: To implement the functionality of capturing user images using a user-side camera, WebRTC has been used to provide a web browser application with RTC capability via *getUserMedia. MediaDevices* API. With HTTPS, WebRTC allows users to grant permission for the system to access their cameras (user-side webcams or built-in laptop cameras) and capture user-side images in real-time.

(5) ECharts [45]: ECharts, a powerful and high-performance JavaScript-based charting and visualization library developed by Baidu, was used to provide visual representations of experimental data.

4.1.2. Live video streaming for physical test rigs

Live video streaming has been achieved by using HTML5 features such as WebSocket and canvas, as well as open source software such as *FFmpeg*, which is used to feed the relay from the RTSP (real time streaming protocol) video stream of Internet Protocol (IP) cameras to the mpeg-ts (transport stream) stream, and *JSMepg* [46], which provides jsmpeg to decode the mpeg-ts video and WebSocket relay to create a WebSocket server for WebSocket connection. Compared with HTTP live streaming, *JSMepg* allows low latency streaming through WebSocket compatible with any existing devices.

4.1.3. User side camera for supervision

By using the HTML5 WebRTC API, the user-side camera, whether a webcam or a built-in laptop camera, can be invoked during the experiment in the monitoring interface if permitted by the user. The detailed implementation of a user-side camera is demonstrated in Algorithm 1. Once the user-side camera is enabled and the image capture is attached to the laboratory report, the teacher can easily recognize the experimenter to achieve a much more objective laboratory work assessment [12].

Algorithm 1 Implement a user-side camera widget

Require: A webcam or a built-in laptop camera
Ensure: WebRTC in *componentDidMount()*
  constraints = {audio: false, video: true}
  monitor = this.props
  if *monitor* **then**
    navigator.mediaDevices.getUserMedia(constraints)
    .then(function(stream){
    var video = document.querySelector('video');
    video.srcObject = stream}
  end if
  while *render*() **do**
    if *monitor* **then**
      <Camera />
    else
      < Icon type="camera" size="small"/>
    end if
  end while

4.2. Server cluster deployment

As Fig. 2 shows, there are six different types of servers deployed in the NCSLab system, which are classified according to their functionalities. The six servers work together to provide experimental services that involve login and real-time experimentation under the coordination of the central web server. With the utilization of the NGINX reverse proxy server for load balancing, the loosely coupled server cluster enables seamless expansion, allowing for the addition of new experiment servers or web servers to support more users.

The detailed server cluster deployment is illustrated in Fig. 3. To demonstrate the logical order of request access clearly, an application scenario that details the process flow is considered as follows:

Stage 1 A user enters the uniform resource locator (URL) of the new NCSLab system (https://www.powersim.whu.edu.cn/react) into a web browser, and a request is sent to the *Proxy Server* that is based on NGINX and acts as a high-performance HTTP and reverse proxy server. With proper configuration, all requests can be passed to different servers to provide different services.

Stage 2 When the user navigates to the homepage before logging into the system, the *File Server* where static resources such as CSS, HTML and JavaScript files are stored is reached.

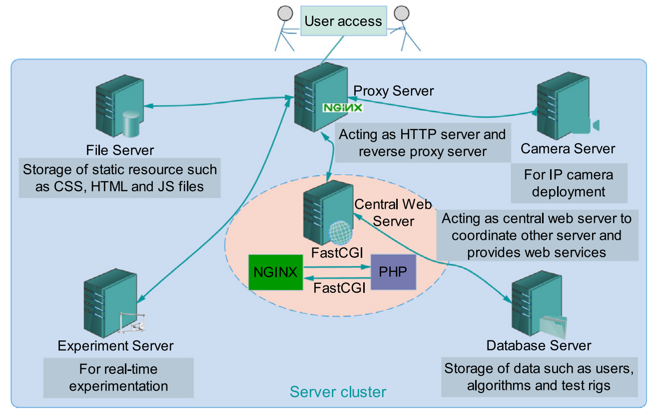


Fig. 3. Server cluster deployment.

Stage 3 When the user attempts to log in and conduct experiments, the *Central Web Server* is accessed to handle different APIs. The *Central Web Server* is composed of NGINX and PHP FastCGI, which offer a high-performance web server architecture capable of effectively managing a large number of requests. The APIs provided by the *Central Web Server* are responsible for handling crucial tasks, such as authentication and authorization. This process involves accessing the *Database Server*, which addresses requests for data such as user accounts, test rigs, and algorithms.

Stage 4 The *Experiment Server* which is called rtlab (real-time laboratory) takes charge of real-time experimentation where algorithms are downloaded and executed in controllers and test rigs are controlled and monitored. Tomcat is adopted for the deployment of the Experiment Server, which communicates directly with the web browser using WebSocket.

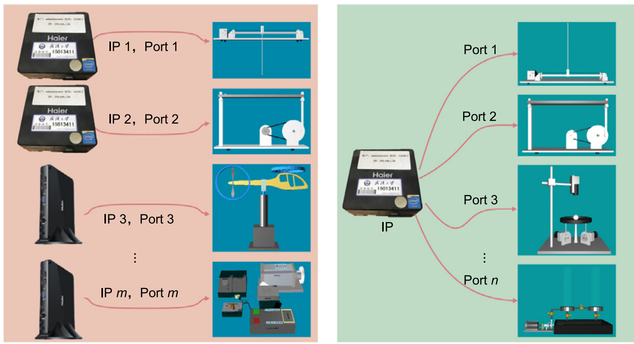
Stage 5 During a real-time experiment with a physical test rig, the live video streaming of the test rig can be monitored with the support of the *Camera Server*. The *Camera Server* is platform independent and can be deployed on any platform, such as Windows and Linux.

To enhance security, HTTPS, which runs HTTP on top of secure sockets layer/transport layer security (SSL/TLS) with the installation of an HTTPS certificate, is adopted. The HTTPS ensures the protection of user data as well as the authentication between the server and the web client.

4.3. Controller deployment

The controllers are low-cost mini personal computers (PCs) based on the Windows operating system. The executable codes generated using MATLAB/Simulink Coder can be executed directly in the controller [47]. The advantage of PC-based systems is easy debugging. In case any malfunctioning occurs, maintenance staff can easily login to the system using a simple remote desktop and track the problems.

In previous versions of the NCSLab system, one controller that is allocated with a dedicated IP address corresponds to one test rig, either a physical or virtual one, thus, IP mapping between controllers and test rigs has been achieved [48]. Instead of IP mapping, port mapping, which is cost effective from the perspective of controller purchasing, has been achieved in the proposed system of this paper, which enables one controller to supply as many test rigs as it can only be limited by its hardware, such



(a) (b)

Fig. 4. Illustration of controller deployment for multiple test rigs. (a) Deployment of previous versions. (b) Current deployment of the React-based system.

as the central processing unit (CPU) and random access memory (RAM), and the number of ports. Fig. 4 illustrates the controller deployment for multiple test rigs. Fig. 4(a) shows the controller deployment of previous versions, and Fig. 4(b) demonstrates the proposed system in this paper. Multiple ports of one controller instead of multiple controllers with more IP addresses are needed, which requires fewer controllers.

4.4. Test rig integration

NCSLab supports both physical and virtual test rigs, as shown in the test rig tier of Fig. 2. Physical test rigs such as fan, direct current (DC) motor, and dual tank are integrated, as well as virtual test rigs such as ball and beam and inverted pendulum. In addition to physical test rigs for remote laboratories, 3-D virtual test rigs are developed and integrated into the NCSLab for interactive experimentation. The process of integrating a test rig for virtual laboratories involves 3-D modeling, mathematical modeling, control algorithm design and implementation, and finally integration into the NCSLab. In virtual laboratories, the 3-D models are dynamically updated and animations are achieved under the control of algorithms, which can be a complement to physical test rigs for remote laboratories.

Although there are currently several physical and virtual test rigs in the new NCSLab system, it can hold as many users to control test rigs as possible, which used to be the number of test rigs at a time. For a specific virtual test rig, the replicas that enable concurrent experimentation are called *copies*. Through copies, concurrent experimentation is possible and enables many users to conduct virtual experiments with the same virtual test rig represented by the 3-D model and mathematical model.

For one physical test rig, only one user can conduct an experiment. Multi-instance remote laboratories in [22] enable remote experimentation for multiple users concurrently. The NCSLab system provides similar experience, which constructs multiple cost-effective test rigs and can be allocated to users automatically and dynamically using a unified queuing system as the virtual test rigs. Thus, the previous queuing scheme remains for physical test rigs. The solution to hold more users is to integrate more physical test rigs and improve virtual experiments to achieve online learning without loss of any level of knowledge [1].

Each virtual test rig copy or physical test rig consumes two ports: one for algorithm downloading and the other for monitoring. The integration of new test rigs into the NCSLab is similar to the old version of the NCSLab [10], which includes hardware design and implementation (for the physical test rig), 3-D modeling (for the virtual test rig), control algorithm implementation, and web-based rendering and motion control.

5. Concurrent solution for usage control of virtual test rigs

Throughout the development and evolution of the NCSLab system, it is insisted that real-time interactive experimental services are provided to users. In previous versions of the NCSLab system, the queuing scheme has been applied to provide experimental services. The users who apply for control privilege (control over a test rig while others cannot access it) are placed into a queue, and as long as the former users finish their experiments or give up their control, the latter user in the queue can control the test rig. However, previous versions of the NCSLab did not offer concurrent experimentation, limiting the capacity to deliver experiments to a greater number of students.

For the new React-based NCSLab system, multiple users can conduct experiments with the same virtual test rig without interfering with each other in the concurrent situation. They can upload and remotely execute their own control algorithms, customize their own monitoring interfaces, and monitor and control their experiment.

The concurrent solution is not only efficient but also cost-effective. Previously, there was a one-to-one correspondence between controllers and IP addresses. As one controller occupies one IP address, if an increasing number of test rigs are integrated, the IP addresses of the local area network (LAN) with one router will no longer be available. The newly implemented system occupies one controller for virtual test rigs, and different ports correspond to different copies, thus, multiple ports are required instead of multiple controllers and IP addresses.

The details on concurrent design and implementation are explored as follows.

5.1. Monitoring interface

When a user selects a test rig for experimentation, the test rig interface will provide information regarding the *3-D Model*, *Plant Information*, *Control Algorithm*, and *Configuration* to the user. Moreover, the user would also be informed of the usage information of the corresponding test rig, which includes the status (whether available or not), the total number of copies and the number of copies that remained available, as shown in Fig. 1. Once the user requests the control and control has been granted, the usage information will change in real-time. In this case, the total number of copies, the occupied copy, and the time left for experimentation will be provided to the user. For instance, the message "*There are 100 copies, the No. 1 copy is occupied by you. 29 min left*" reminds the user of the information of the test rig. Once the control has been released, more copies would be available for other users.

Fig. 5 illustrates the usage control scheme for virtual and physical test rigs in the new version of the NCSLab. In addition to the FCFS queuing scheme, the concurrent and queuing scheme has been implemented. For a virtual test rig, the first m users (C_1 to C_m) as shown on the right-hand side of Fig. 5, can conduct experiments simultaneously, which constitutes a concurrent scenario. When all copies of the virtual test rig are not idle, users who request control of the same test rig are put in a queue (Q_1 to Q_n in the upper left part of Fig. 5). Control is automatically granted to the first queuing user once the former user completes the experiment and releases the control or gave up control, or when a timeout occurs for the experimentation. For physical test rigs, the previous queuing scheme (P_1 to P_k in the lower left part of Fig. 5) remains to be adopted.

5.2. Database

To support concurrent experimentation, the structure of the database of the new NCSLab system must be reorganized. For

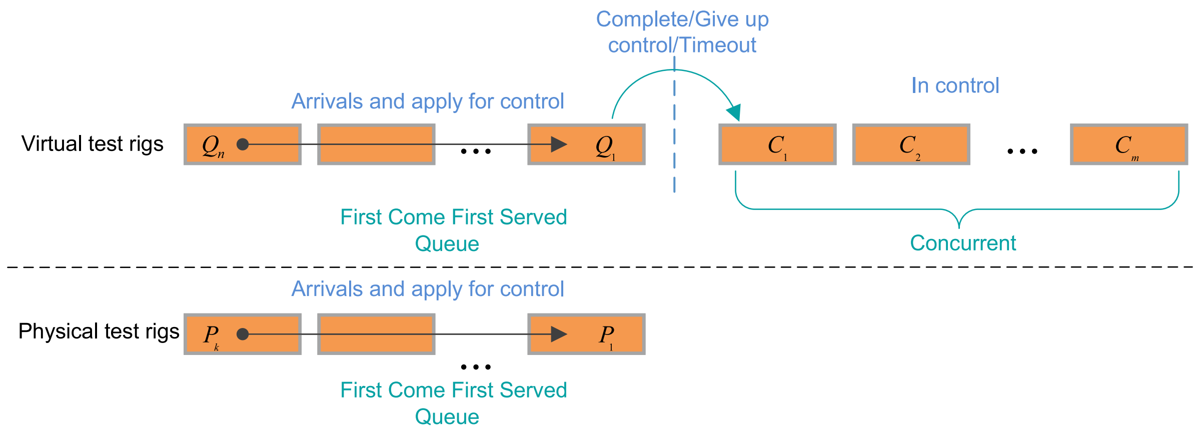


Fig. 5. Usage control scheme for virtual and physical test rigs in the new version of NCSLab.

example, for the *test_rigs* database table, as there are two types of test rigs, the data of the *plantType* field can either be "1" or "2". "1" stands for virtual test rigs, and "2" stands for physical test rigs. For physical test rigs (*plantType=2*), as only one user can conduct the experiment at a time, the copy number is set to "1". For virtual test rigs (*plantType=1*), the copy number can be modified to meet the requirements, for example, if the copy number has been set to 100, it means 100 different users can experiment with the same test rig concurrently.

5.3. Experiment server rtlab

For each real-time experiment, an experimental service is established between the experiment server rtlab and the corresponding test rig. For concurrent situations, experimental service instances are established for each copy of the test rig. Each copy is independent from the others, which obtains their own control algorithm, signals and parameters for monitoring and control.

Listing 1 shows the *JavaScript Object Notation (JSON)* file of the WebSocket request for algorithm downloading when a user clicks the "conduct an experiment" button of the ball and beam system in the monitoring interface, which will be sent to the experiment server. Then, WebSocket communication will be established between the client-side web browser and the experiment server. The WebSocket protocol enables interactions between the web browser and the experiment server and facilitates real-time data transfer. Compared to the HTTP request, once a handshake request (start experiment) is sent and the initial connection has been established, the experiment server can actively send content to the web browser without being first requested, and allow messages to be passed back and forth while keeping the connection open.

The WebSocket request identifies the experiment server URL, IP address of the controller, and ID of the control algorithm, thus, the request can be interpreted as follows: "Start connection with the experiment server *www.powersim.whu.e-du.cn/rtlab1*, and download the algorithm (id = 105, stepSize = 0.01, packageSize = 50) to the controller (ip = 192.168. 46.143, downloadPort = 18001, monitorPort = 19001) to conduct the experiment with the virtual ball and beam system (plantId = 2, plantType = 1, copyNum = 1)". For concurrent situations, a different copy of the virtual ball and beam system with the same/different control algorithms can be downloaded to the same controller with different *downloadPort* and *monitorPort*.

Listing 1: *JSON* file for algorithm downloading and execution

```
{
  "serverUrl": "www.powersim.whu.edu.cn
     /rtlab1",
  "ip": "192.168.46.143",
  "plantId": 2,
  "plantType": 1,
  "copyNum": 1,
  "id": 105,
  "stepSize": 0.01,
  "packageSize": 50,
  "downloadPort": 18101,
  "monitorPort": 19101
}
```

5.4. Controller

The controller is where algorithms are downloaded and executed. In concurrent situations, a program that opens several ports is executed on the controller, for example, 500 ports from 18000 to 18499 for downloading and 500 from 19000 to 19499 for monitoring, which actively waits and accepts the algorithm downloading and executing commands. Fig. 4 demonstrates the IP and ports of the corresponding controller. Once a request for *conducting an experiment* has been received, the algorithm will be downloaded to the expected downloading port. As long as the user starts to monitor the experiment, the assigned monitoring port begins to serve its purpose.

The concurrent scheme benefits both users and developers. From the perspective of concurrent users, the corresponding test rig is currently occupied by themselves as multiple identical test rig copies exist for their experimentation. From the point view of developers, fewer controllers are needed and the system can hold many more concurrent users at a time while providing more available rigs without any effect on the usage duration of each user.

6. Case studies

Without loss of generality, a concurrent experiment was carried out with a renewable energy system using the newly developed system to illustrate the effectiveness of the concurrent design and implementation, in which an induction machine was used as a wind turbine generator. The control algorithms of the

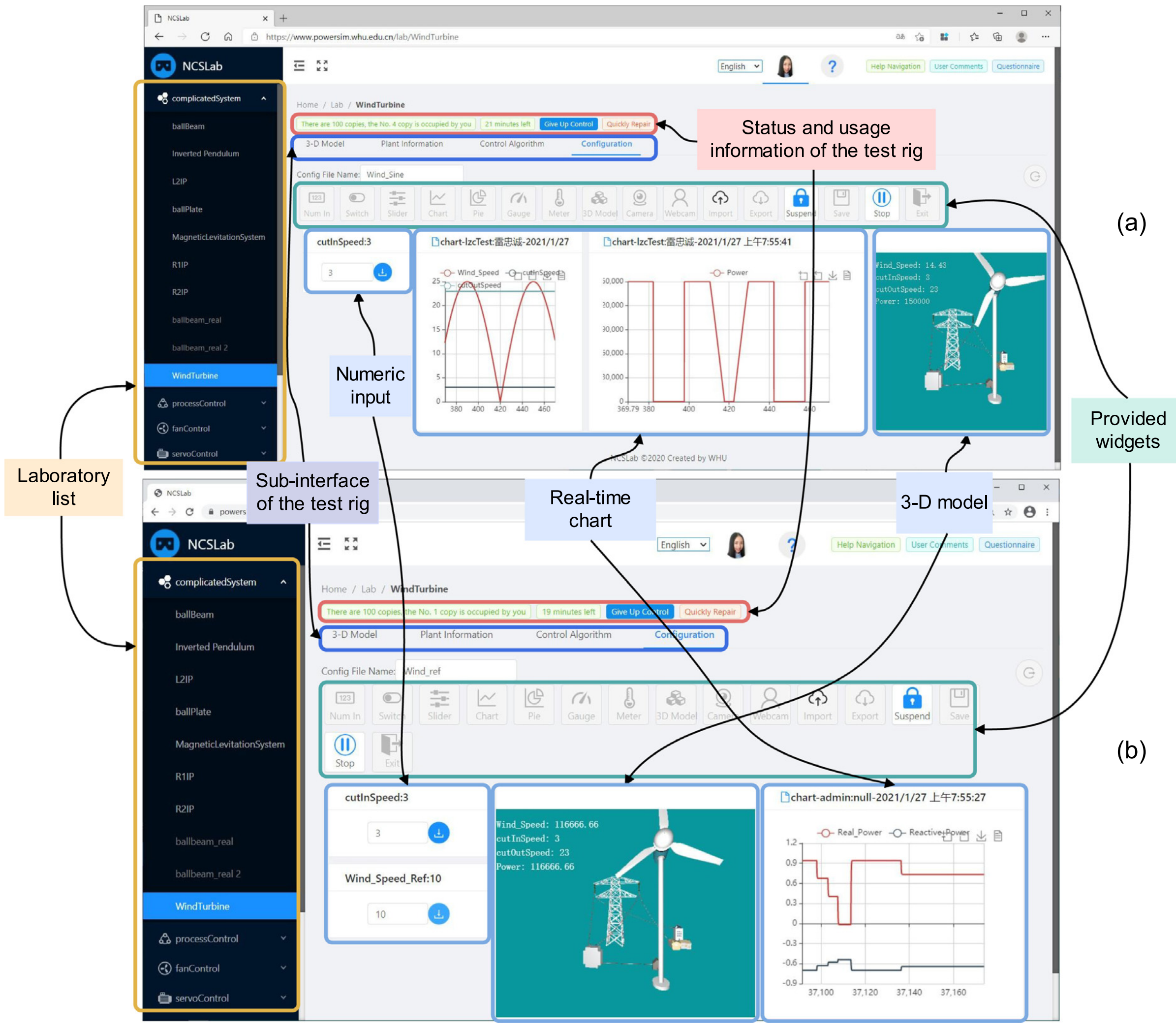


Fig. 6. Different concurrent monitoring interfaces of the NCSLab experimentation system with a wind renewable system. (a) Experiment with Microsoft Edge; (b) Experiment with Google Chrome.

wind renewable system are based on a control algorithm provided by MATLAB/Simulink,[1] with some modifications for control and demonstration purposes in this paper. The copy number of the virtual system has been set to 100, which means 100 different users can conduct their experiments concurrently with one wind renewable system.

Fig. 6 illustrates concurrent experimentation with the wind renewable system in the NCSLab involving different users with diverse monitoring interfaces, which also employs different control algorithms. It can be seen that different monitoring interfaces are customized by the two users to monitor and control the experiments. In Fig. 6(a), user 1 occupies the No. 4 copy of the wind renewable system after obtaining the control privilege using Microsoft Edge, while in Fig. 6(b), the No. 1 copy is occupied by user 2 using Google Chrome. The cut-in speed, cut-out speed, and wind turbine power output curves are depicted in Fig. 6(a), while the curves of real and reactive power can be seen in Fig. 6(b).

From Fig. 6, it can be seen that at least four users concurrently perform their virtual experiments with the wind renewable system. From the perspective of users, they all control the virtual wind renewable system, and other concurrent users do not affect their experiments at all, which proves that the structure is scalable and that the concurrent solution is highly efficient and cost-effective. The copy number can be easily modified in the database to hold more concurrent users as long as the controller can bear. The wind renewable system can be used for research and education purposes, with which users can explore the wind renewable system by tuning the wind speed, cut-in/cut-out speed, *etc.*

To demonstrate the scalability feature and the system's capability, a concurrent access experimentation test with 300 simulated users has been conducted. Testing concurrent access with massive real users is challenging. Considering the requirements of the test, in this paper, a test program has been built to test the concurrent scenario. To build such a program, part of the procedures of conducting an experiment must be considered as follows:

(1) Each user must log in to the system successfully;
(2) Each user has to be granted control of the test rig;
(3) Each user must synchronously and successfully download the executable control algorithm to the simulator.

Thus, the steps and configurations of the test program are as follows:

Step 1: Batch registration and login.

Step 2: Control request.

[1] https://ww2.mathworks.cn/help/sps/ug/three-phase-asynchronous-wind-turbine-generator.html

Step 3: Algorithm downloading.

Once algorithms have been successfully downloaded to the remote controller, it indicates that the users are able to configure a monitoring interface and start the experiment, which means massive concurrent experiments are achieved.

The test PC that initiates the concurrent test to simulate concurrent access is based on the Windows operating system with a 3.59 GHz AMD Ryzen 7 3700X 8-Core CPU and 16 GB RAM, and the corresponding controller is a Windows-based server with a 2.1 GHz Intel Xeon Gold 6130 CPU and 32 GB RAM. The test results show that when 300 test requests (representing 300 virtual students) were launched, 296 control responses and 284 algorithm downloading responses were obtained, respectively. It can be concluded that even with 300 concurrent users, 94.67% of the users can still concurrently access the same virtual laboratory, which demonstrates the scalability feature.

7. Evaluation and analysis

Since late 2019, the newly developed NCSLab system has been applied to classroom teaching and after-class experimentation in several courses at different universities. Three courses with students of different backgrounds are selected as examples: students in their fourth year with Automation background at Wuhan University (WHU) participated in the class module *System Identification* during the Autumn semester of 2019∼2020, students in their second year with Energy and Power Engineering background at WHU participated in the class module *Classic Control Theory* during the Spring semester of 2019∼2020, and students in their third year with Electronics and Information Science background at Henan Agricultural University (HAU) participated in the class module *Classic Control Theory* during the Spring semester of 2020∼2021. Similar to the previous arrangement [17], students were asked to finish their assignments with the NCSLab system. Due to the COVID-19 pandemic, students participated in the course delivered online and finished their laboratory assignments at geographically diverse locations throughout China in the 2019∼2020 Spring semester.

For the three class modules, students were invited to engage in an online survey regarding the user experience with NCSLab, which includes a five-point Likert scale ranging from "5" that stands for "strongly agree" to "1" that represents "strongly disagree". For consistent evaluation, several questions are similar to those in the old NCSLab system [17]. Finally, 31 out of 52 students in the 2019∼2020 Autumn semester, 20 out of 58 in the 2019∼2020 Spring semester, and 75 out of 75 in the 2020∼2021 Spring semester gave their responses. The results of nine selected questions in the 2019∼2020 Autumn semester are illustrated in Fig. 7(a), from which it can be seen that students think the system is of great educational value (*Q7*, *Q8*, and *Q9*), and the usability (*Q1*, *Q2*, *Q3*, and *Q4*) and stability (*Q5* and *Q6*) are good from their experience. It is worth mentioning that there was no negative feedback (strongly disagree or disagree) regarding the nine questions. Apart from the survey results included in [12], several other survey results from the 2019∼2020 Spring semester at WHU and the 2020∼2021 Spring semester at HAU are depicted in Fig. 7(b) and Fig. 7(c), respectively. The majority of surveyed students at both universities responded positively to the implementation of 3-D virtual test rigs for experimentation, with 90% at WHU and 96% at HAU agreeing that it was necessary (*Q1*). The use of virtual test rigs also helped students understand the experiments, with 100% at WHU and 98.67% at HAU responding positively (*Q2*), and 95% at WHU and 96% at HAU indicating satisfaction with the experimentation process (*Q3*). Moreover, 75% of surveyed students at WHU and 88% of those at HAU expressed interest

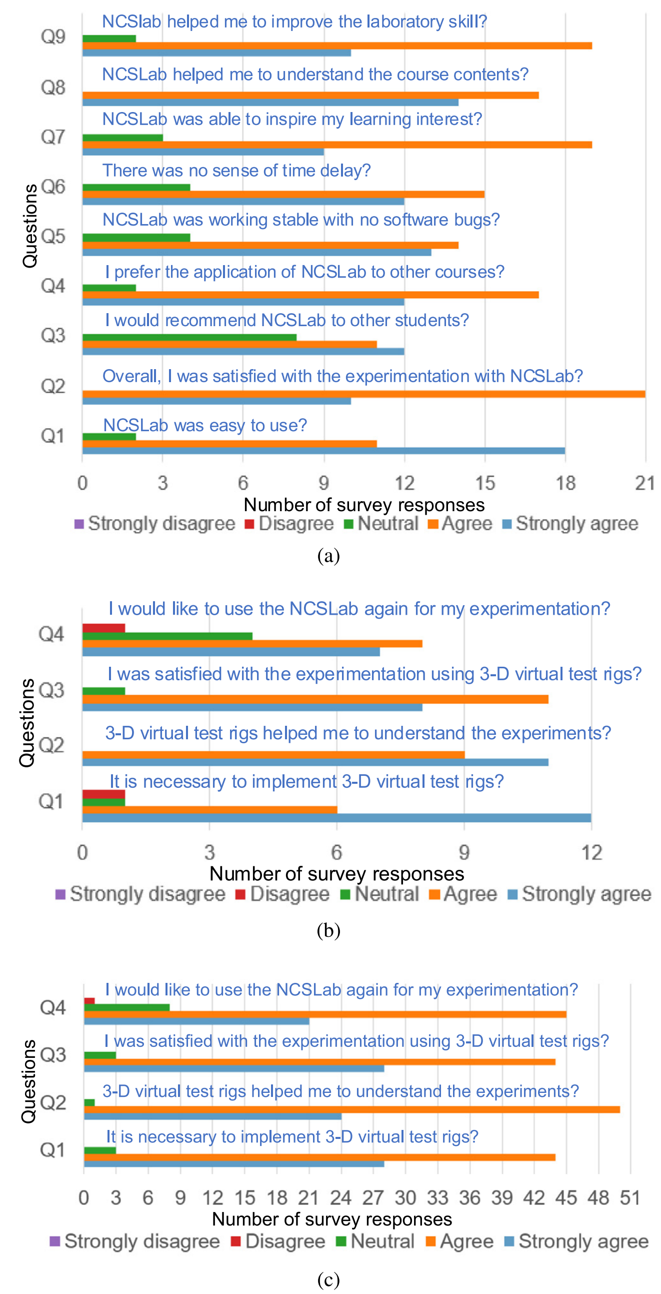


Fig. 7. Questionnaire analysis results. (a) 2019∼2020 Autumn semester at WHU; (b) 2019∼2020 Spring semester at WHU; (c) 2020∼2021 Spring semester at HAU.

in using NCSLab again for their experiments (*Q4*). These survey results demonstrate the potential benefits of using virtual test rigs in the context of online experimentation.

In addition to the user survey, an open source software, Apache JMeter, was used for performance and load testing to evaluate the system's performance. A GET http request was sent to the NCSLab homepage (https://www.powersim.whu.e-du.cn/react), with the following configurations of *Thread Properties*: *Number of Threads (users)* simulated for five rounds (Round ID from 1 to 5 in Table 2) were 1000, 1500, 1600, 1800 and 2000, respectively. The *Ramp-up period (seconds)* was set to 30 s, which introduced a delay of $30/X$ seconds before starting the next user (where X represents the number of simulated users), and the *Loop*

Table 2
Performance test results using JMeter.

Round ID	Samples	Average	Median	90% Line	95% Line	Min	Max	Std. Dev.	Error %	Throughput
1	10000	44	32	83	111	3	1176	56.01	0.00%	331.7/sec
2	15000	128	117	196	262	3	1292	62.03	0.00%	494.8/sec
3	16000	194	115	214	965	3	2306	276.55	0.00%	499.4/sec
4	16163	447	117	442	2767	3	26117	1638.69	0.72%	276.4/sec
5	14438	850	118	2055	3613	3	29633	2913.92	6.91%	255.2/sec

Count was set to 10, which simulates each user connecting to the NCSLab 10 times. The results are shown in Table 2, in which times are in milliseconds. The 90% Line (90th Percentile) or 95% Line (95th Percentile) means 90% or 95% of the samples took no more than this time [49]. It can be concluded that the system can handle at least 1600 concurrent users (Round 3) in 30 s with each user connecting 10 times without any errors, resulting in a throughput of 499.4 requests per second. However, for 1800 users (Round 4), the error rate increased to 0.72% and the number of samples decreased to 16163 instead of 18000. It is worth noting that the tests conducted were specific to the current deployment of NCSLab, and introducing servers with varying capacities in the deployment may lead to variations in the results. As the four-tier architecture shown in Fig. 2 is scalable and extensible, the system can effectively accommodate a higher number of concurrent users by seamlessly integrating additional servers, such as an extra web server.

The proposed approach has some limitations that should be acknowledged. One of these limitations concerns the scalability of the system, which could still pose a challenge. Although the system has been tested with up to 300 simulated concurrent users, testing with real students is challenging as students may use the systems at different times. Therefore, powerful hardware, including experiment servers and controllers, may be required to improve the system's capacity and enhance its capability to accommodate more real concurrent users. Another limitation is the systems' reliance on the stability and speed of the internet connection. Therefore, a poor internet connection may result in slower response times and potentially affect the overall user experience, especially for 3-D virtual test rigs.

8. Conclusion and future work

In this paper, an experimentation platform that provides remote and virtual experimental services with a scalable structure is discussed. The system has been designed and implemented with a novel architecture integrating many open source technologies, such as React, AntD and Echarts, based on the front-end and back-end separation principle, which provides a better user experience for users and eases the design and implementation for developers. Through the implementation of the user interface, database, controller and experiment server, the concurrent scheme has been achieved, improving the utilization of the system and enabling multiple users to conduct experiments with the same virtual test rig concurrently, which is highly efficient and cost-effective. The implementation is effective for scenarios such as concurrent experimentation in a classroom teaching and demonstration. The new system has been applied to teaching and experimentation at WHU and HAU, and feedback from students is collected and analyzed, the results of which verify the effectiveness and general acceptance of the new NCSLab and also the use of 3-D virtual test rigs. The concurrent test results with 300 virtual students demonstrate the scalability and concurrency features of the proposed system. The detailed methodologies and technologies presented in this paper could be used for other online laboratories to build universal, interactive and user-friendly systems and solve scheduling issues with a concurrent mechanism.

In the future, optimization for mobile devices and large-scale applications in research and education will be carried out. Future work will also include the integration of more test rigs and control algorithms. Future deployment may take into account a cloud-based approach that allows for parallel experimentation on online laboratories.

Funding statement

This work was supported in part by the National Natural Science Foundation of China] under Grant 62103308 and Grant 62073247, in part by the Fundamental Research Funds for the Central Universities, China under Grant 2042023kf0095, in part by the China Postdoctoral Science Foundation under Grant 2022T150496, and in part by Wuhan University Experiment Technology Project Funding, China under Grant WHU-2022-SYJS-10.

CRediT authorship contribution statement

Zhongcheng Lei: Writing – original draft, Conceptualization, Investigation, Funding acquisition. **Hong Zhou:** Supervision, Project administration. **Wenshan Hu:** Writing – review & editing, Software, Funding acquisition. **Guo-Ping Liu:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] T. De Jong, M.C. Linn, Z.C. Zacharia, Physical and virtual laboratories in science and engineering education, Science 340 (6130) (2013) 305–308.

[2] M. Barker, et al., The global impact of science gateways, virtual research environments and virtual laboratories, Future Gener. Comput. Syst. 95 (2019) 240–248.

[3] L. de la Torre, L.T. Neustock, G. Herring, J. Chacon, F. Garcia, L. Hesselink, Automatic generation and easy deployment of digitized laboratories, IEEE Trans. Ind. Inform. 16 (12) (2020) 7328–7337.

[4] Z. Lei, H. Zhou, W. Hu, G.-P. Liu, Flipping laboratories toward future experimentation systems: The blended use of hands-on, pocket, and online laboratories, IEEE Ind. Electron. Mag. http://dx.doi.org/10.1109/MIE.2022.3190823 in press.

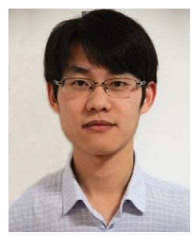
[5] V.-M. Aitor, J. García-Zubía, I. Angulo, L. Rodríguez-Gil, Toward widespread remote laboratories: Evaluating the effectiveness of a replication-based architecture for real-world multiinstitutional usage, IEEE Access 10 (2022) 86298–86317.

[6] J. Sáenz, L. de la Torre, J. Chacón, S. Dormido, A study of strategies for developing online laboratories, IEEE Trans. Learn. Technol. 14 (6) (2021) 777–787.

[7] S. Guan, H. Zhao, S. Tan, L. Xu, Conducting experiments with real-time cloud computing models: A virtual laboratory for massive open online courses on computer control systems, IEEE Control Syst. Mag. 41 (6) (2021) 97–107.

[8] L. Gomes, S. Bogosyan, Current trends in remote laboratories, IEEE Trans. Ind. Electron. 56 (12) (2009) 4744–4756.

[9] J.J. Rodriguez-Andina, L. Gomes, S. Bogosyan, Current trends in industrial electronics education, IEEE Trans. Ind. Electron. 57 (10) (2010) 3245–3252.
[10] Z. Lei, et al., Unified 3-D interactive human-centered system for online experimentation: Current deployment and future perspectives, IEEE Trans. Ind. Inform. 17 (7) (2021) 4777–4787.
[11] I. Chirikov, T. Semenova, N. Maloshonok, E. Bettinger, R.F. Kizilcec, Online education platforms scale college stem instruction with equivalent learning outcomes at lower cost, Sci. Adv. 6 (15) (2020) eaay5324.
[12] Z. Lei, H. Zhou, W. Hu, G.-P. Liu, Unified and flexible online experimental framework for control engineering education, IEEE Trans. Ind. Electron. 69 (1) (2022) 835–844.
[13] Y. Larbaoui, A. Naddami, A. Fahli, Design and structure of an electronic switching system for remote circuits structuring and online measurements, Int. J. Online Biomed. Eng. 18 (6) (2022) 30–46.
[14] D. May, B. Morkos, A. Jackson, N.J. Hunsu, A. Ingalls, F. Beyette, Rapid transition of traditionally hands-on labs to online instruction in engineering courses, Eur. J. Eng. Educ. http://dx.doi.org/10.1080/03043797.2022.2046707 in press.
[15] M.A. Zaman, L.T. Neustock, L. Hesselink, Ilabs as an online laboratory platform: A case study at stanford university during the COVID-19 pandemic, in: Proc. IEEE Global Eng. Educ. Conf, EDUCON, 2021, pp. 1615–1623.
[16] P.C. Nicolete, F. Herpich, E.T. de Oliveira Júnior, L.M.R. Tarouco, J.B. da Silva, Analysis of student motivation in the use of a physics augmented remote lab during the Covid-19 pandemic, in: Proc. IEEE Global Eng. Educ. Conf, EDUCON, 2021, pp. 1040–1047.
[17] W. Hu, et al., Plug-in free web-based 3-D interactive laboratory for control engineering education, IEEE Trans. Ind. Electron. 64 (5) (2017) 3808–3818.
[18] L. Angrisani, F. Bonavolontà, M. D'Arco, A. Liccardo, A flexible remote laboratory with programmable device under test, Measurement 156 (2020) 107584.
[19] A. Nafalski, M. Miłosz, H. Considine, International sharing of the remote laboratory netlab, Int. J. Online Biomed. Eng. 16 (11) (2020) 16–31.
[20] A. Van den Beemt, S. Groothuijsen, L. Ozkan, W. Hendrix, Remote labs in higher engineering education: engaging students with active learning pedagogy, J. Comput. High. Educ. http://dx.doi.org/10.1007/s12528-022-09331-4 in press.
[21] A.M. Abd El-Haleem, M.G. Anany, M.M. Elmesalawy, E.S.E.-D. Bakr, A matching game-based laboratory learning system for resources management in remote laboratories, IEEE Access 11 (2023) 6246–6260.
[22] A. Villar-Martínez, L. Rodríguez-Gil, I. Angulo, P. Orduña, J. García-Zubía, D. López-De-Ipiña, Improving the scalability and replicability of embedded systems remote laboratories through a cost-effective architecture, IEEE Access 7 (2019) 164164–164185.
[23] R. Costa, F. Pérola, C. Felgueiras, μLab a remote laboratory to teach and learn the atmega328p μc, in: Proc. IEEE Global Eng. Educ. Conf, EDUCON, 2020, pp. 12–13.
[24] L. de la Torre, J. Sàenz, D. Chaos, J. Sánchez, S. Dormido, A master course on automatic control based on the use of online labs, IFAC-PapersOnLine 53 (2) (2020) 17542–17547.
[25] P. Orduña, J. García-Zubia, Scheduling schemes among internet laboratories ecosystems, in: Proc. 2011 8th Int. Conf. Remote Eng. Virtual Instrum, 2011, pp. 1–6.
[26] A. Parkhomenko, A. Parkhomenko, G. Tabunshchyk, K. Henke, H.-D. Wuttke, The remote labs as an effective tool of inclusive engineering education, in: Int. Conf. Perspect. Technol. Methods MEMS Des, MEMSTECH, 2018, pp. 209–214.
[27] P. Orduña, L. Rodriguez-Gil, I. Angulo, U. Hernandez, A. Villar, J. Garcia-Zubia, Weblablib: new approach for creating remote laboratories, in: Proc. Int. Conf. Remote Eng. Virtual Instrum, 2019, pp. 477–488.
[28] A. Maiti, A.A. Kist, A.D. Maxwell, Automata-based generic model for interoperating context-aware ad-hoc devices in internet of things, IEEE Internet Things J. 5 (5) (2018) 3837–3852.
[29] Y. Khazri, A. Fahli, M. Moussetad, A. Naddami, Design and implementation of a reservation system and a new queuing for remote labs, Int. J. Online Biomed. Eng. 15 (12) (2019).
[30] D. Lowe, Integrating reservations and queuing in remote laboratory scheduling, IEEE Trans. Learn. Technol. 6 (1) (2013) 73–84.
[31] G.K. Herring, L.T. Neustock, L. Hesselink, Learning from the unexpected: Statistics and uncertainty in massively scalable online laboratories (msol), in: Proc. IEEE Global Eng. Educ. Conf, EDUCON, 2018, pp. 815–824.
[32] L.T. Neustock, G.K. Herring, L. Hesselink, Remote experimentation with massively scalable online laboratories, in: Online Engineering & Internet of Things, Springer, Berlin, Germany, pp. 258–265.
[33] F. Moghaddam, et al., Massive open online labs (MOOLs): An innovative solution to achieving sdgs in the global south, in: Proc. 2019 5th Exp. Int. Conf. Exp. At, 2019, pp. 394–398.
[34] C. Salzmann, D. Gillet, Next steps in supporting more students in MOOL for control education, IFAC-PapersOnLine 51 (4) (2018) 184–189.
[35] C. Salzmann, W. Halimi, D. Gillet, S. Govaerts, Deploying large-scale online labs with smart devices, in: Cyber-Physical Laboratories in Engineering and Science Education, Springer, 2018, pp. 43–78.
[36] V.J. Harward, et al., The iLab shared architecture: A web services infrastructure to build communities of internet accessible laboratories, Proc. IEEE 96 (6) (2008) 931–950.
[37] Z. Teng, P. Daogang, S. Bin, X. Chao, Z. Hao, A fault diagnosis system for GT control system based on DB and separation of front and back ends, in: Power Syst. Green Energy Conf, 2021, pp. 665–669.
[38] R. Ren, X. e. Sun, L. Hu, A new method for hosting and sharing MATLAB web app, Sci. Rep. 12 (1) (2022) 21645.
[39] YUI, https://clarle.github.io/yui3/. (Online; Accessed 20 March 2023).
[40] F. Žemla, J. Cigánek, Design and implementation of the application for the irrigation system, in: Proc. Int. Conf. Cybern. Informatics, 2022, pp. 1–6.
[41] React, https://react.dev. (Online; Accessed 20 March 2023).
[42] Ant Design 5.0, https://ant.design. (Online; Accessed 20 March 2023).
[43] Three.js, https://threejs.org. (Online; Accessed 20 March 2023).
[44] WebRTC, https://webrtc.org/. (Online; Accessed 20 March 2023).
[45] D. Li, et al., ECharts: A declarative framework for rapid construction of web-based visualization, Vis. Inform. 2 (2) (2018) 136–146.
[46] JSMpeg, https://jsmpeg.com. (Online; Accessed 20 March 2023).
[47] Z. Lei, H. Zhou, W. Hu, G.-P. Liu, Controller effect in online laboratories-an overview, IEEE Trans. Learn. Technol. http://dx.doi.org/10.1109/TLT.2023.3267491 in press.
[48] Z. Lei, et al., 3-D interactive control laboratory for classroom demonstration and online experimentation in engineering education, IEEE Trans. Educ. 64 (3) (2021) 276–282.
[49] Apache JMeter, Aggregate report, 2023, https://jmeter.apache.org/usermanual/component_reference.html#Aggregate_Report. (Online; Accessed 15 March 2023).



Zhongcheng Lei received the B.S. degree in automation and Ph.D. degree in mechatronic engineering from Wuhan University, Wuhan, China, in 2014 and 2019, respectively.

He is currently an Associate Researcher with the School of Electrical Engineering and Automation, Wuhan University. His current research interests include networked control systems, web-based remote and virtual laboratories, and digital twins.

Hong Zhou received the B.S. degree in industrial automation from the Central South University of Technology, Changsha, China, in 1982, the M.Sc. degree in industrial automation from Chongqing University, Chongqing, China, in 1988, and the Ph.D. degree in mechanical engineering from Wuhan University, Wuhan, China, in 2006.

He has been a Professor with Department of Automation (now the Department of Artificial Intelligence and Automation), Wuhan University since 2000. His research interests include wireless power transfer, smart grid, networked control systems, and digital twins.

Wenshan Hu received the B.S and M.Sc. degrees in control theory and applications from Wuhan University, Wuhan, China, in 2002 and 2004, respectively, and the Ph.D. degree in control engineering from the University of Glamorgan, Pontypridd, U.K., in 2008.

He is currently a Professor in the Department of Artificial Intelligence and Automation, Wuhan University. His research interests include network-based control laboratories, wireless power transfer, and digital twins.

Guo-Ping Liu received the BEng and MEng degrees from the Central South University of Technology in 1982 and 1985, respectively, and the Ph.D. degree from the University of Manchester in 1992. He is a professor with the Southern University of Science and Technology. His research interests include networked multi-agent control, nonlinear identification and intelligent control, multi-objective optimal control, and industrial advanced control applications.

Professor Liu was the general chair of the 2007 IEEE International Conference on Networking, Sensing and Control, the 2011 International Conference on Intelligent Control and Information Processing, and the 2012 UKACC International Conference on Control. He served as an Editor-in-Chief of the International Journal of Automation and Computing in 2004–2021. He is a member of the Academy of Europe, a Fellow of IEEE, a Fellow of IET and a Fellow of CAA.