

WIP - iCtrl - A Remote Login Software Application

Junhao Liao

Junhao Liao is a fourth year Computer Engineering student at University of Toronto. Previously, he worked as a Teaching Assistant at University of Toronto and as a Software Engineering Intern in Qualcomm's Automotive Display Team. He is interested in software programming and User Experience designs. He is proficient with C, C++ and Python and familiar with JavaScript, PSQL, Intel FPGA Verilog and ARM Assembly(V7). Personal Website: <https://junhao.ca>

Haoran Zhang

Yizhong Xu

Undergraduate student at the University of Toronto (2017-2022). Bachelor's degree of Electrical and Computer Engineering with a Business Minor Degree. Co-op internship at Tencent as a Web Developer starting in July 2020. Passionate about programming. Sophisticated with C/ C++, familiar with JavaScript, PSQL, Intel FPGA Verilog and ARM Assembly(V7). Upcoming graduate students in University of New South Wales, major in Master of Information Technology.

Jiaxing Li

Jiaxing Li graduated from University of Toronto. He studied Computer Engineering for undergraduate. He will join Intel as a software engineer. Jiaxing Li is responsible for creating a C++ software model of the FPGA chip that the rest of the compiler can interact with. The FPGA chip is composed of various building blocks. Examples of building blocks are Look-up tables, Flip Flops, Clocks, Routing, IO and more. For each of our basic blocks that compose the FPGA he - understand the customer usage with silicon design and other teams -align on ports and parameters of the basic blocks - package blocks in higher level IPs, or support IP teams to do that -enable synthesis support and implement compiler legality checks -support internal and customer facing simulations -enable place and route (where applicable) -create logical to physical translations -interact with timing and power data -ensure setting correctness to enable FPGA to function as intended Jiaxing Li has also worked for BiblioCommons, where he is responsible for: - Improve, maintain, and deploy backend services built in Spring Boot (Java). JRuby on Rails (JRUBY) - Monitor and maintain the platform services with tools such as: AWS (S3, Lambda, Glue, Athena, CloudWatch), Ansible, TeamCity, Docker, SQL, Prometheus - Work in team with Jira (Bitbucket, Confluence, Agile)

Hamid S Timorabadi (Assistant Professor, Teaching Stream)

Hamid Timorabadi received his BSc, MSc, and PhD degrees in Electrical Engineering from the University of Toronto. He has worked as a project, design, and test engineer as well as a consultant to industry. His research interests include the application of digital signal processing in energy systems and computer networks. He also has deep interest in engineering education and the use of technology to advance the learning experience of undergraduate students.

Work in Progress: iCtrl - A Remote Login Software Application

Abstract

As a result of the COVID-19 pandemic, the need for remote computer connection has drastically increased. This need is also anticipated to increase in the future as many institutes allow working remotely. This paper reports on a development of a software called “iCtrl”, which started in late 2020. iCtrl is a desktop remote-control application that allows remote connection to an institute’s computer facilities. iCtrl application supports Virtual Network Computing (VNC) connection on both Windows and macOS operating systems. It offers a user-friendly interface to eliminate complicated setup processes. iCtrl has been used for over two years and received positive ratings from undergraduate students in a first-year programming language course and a third-year computer network course. Feedback provided by students suggested a few potential improvements such as cross-platform support and Secure Shell (SSH) connections. This motivated us to design a web-based version for iCtrl in summer 2021 that supported cross-OS platform access, SSH and VNC connection, while providing an industry-leading file management interface design with reference to Google Drive and Microsoft OneDrive. This allowed not only to transfer responsibilities of platform support to the browser developers but also to design a more intuitive interface for the students to upload, retrieve and manage files using the SSH File Transfer Protocol (SFTP). The latest survey noted an 86.2% approval of our application over the traditional methods, and the download count of the application is now 10 times of the old version after we applied engineering practices.

Introduction

During the COVID-19 pandemic, studying and working remotely became regular practice for students, professors, and workers around the world [1]. Consequently, the need for a remote computer connection software has exceedingly increased since the outbreak of COVID-19 pandemic in 2020. This need is anticipated to continue to rise in the future since more organizations allowing remote work [2]. For instance, Shopify, a leading electronic commerce (E-commerce) platform, announced that their employees would work remotely on a permanent basis in May 2020 [3].

For desktop remote connection software, existing free-of-charge products include TigerVNC [4] and MobaXterm [5]. As indicated by a survey that was conducted in students who had taken multiple computer courses (Appendix A), the setup processes for these products are relatively complicated for students, especially for those who have little exposure to industry-level software such as Linux Shell (Appendix A). In addition, some remote connection software, such as MobaXterm, is only accessible on a limited number of operating systems (OS), creating problems in providing consistent setup instructions for students who might use different OS platforms. For web-based remote connection software, a state-of-the-art example is Google’s free Chrome Remote Desktop [6]. However, it requires software installations on both local and remote machines, which makes it complex to be distributed in cooperative environments such as school computer laboratories, which usually impose restrictions on software installations.

The Engineering faculty at University of Toronto already provides remote access to over 100 laboratory machines with Linux installations [7], but students still have to set up the connections on their own. The machine allocation mechanism is not optimal, such that the user cannot check

the loads on the machines before establishing the remote connection [8]. Thus, a machine can be operated at full capacity while others remain idle [8], leading to potential overloading and shutdown of the remote machine.

As a result, the current software for remote connection in the market is either concentrated on a single OS platform (Appendix B), making remote connection complicated to establish (Appendix A), or requires additional content to be downloaded [6], raising security concerns from laboratory system administrators. There is neither unified instructions for remote connection, nor optimised laboratory machine allocation mechanism [8].

This motivated us to develop a web-based remote-control application that users can access on any OS platform without additional downloads and eliminate complicated setup processes, while providing an optimised machine allocation mechanism for the University's laboratory machines. The web-based design is chosen for the following reasons: (1) Translate portability and compatibility burden to web-browsers; (2) Eliminate installation and compatibility issues especially on mobile platforms (3) Provide real-time updates to avoid potential technical issues resulted from version updating delays; (4) Offer a more versatile and user-friendly interface with aid from widely used open-source libraries.

In the following sections, key features including user system, remote connection interface, specialised machine allocation mechanism, file management system, and desktop version will be examined in detail. Survey results with positive ratings and increasing active users show a potential in future collaboration with our university to yield a campus-wide software for remote computer connection.

User System

To sign up for iCtrl, users would need to provide an email address which they will later use to receive an account activation email, as well as a username and a secure password. All the communications and credentials are encrypted to ensure the security of user's information. Figure 1 shows the application dashboard after the user logs in, which displays the user's previously stored sessions. Users can add and remove stored sessions, and the changes will be stored in the application server's database.

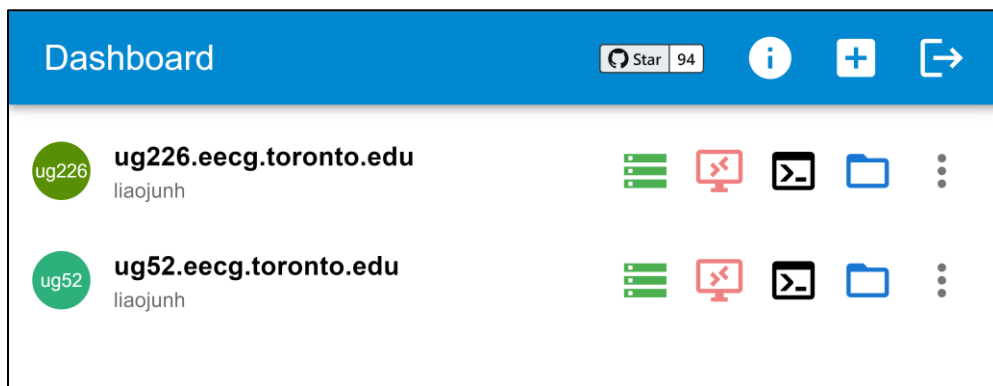


Figure 1: iCtrl Dashboard

Remote Connection

One of the key features the team implemented is the remote connection to the University lab machines. The remote connection feature delivers connection with SSH protocol on terminal command interface, and with VNC protocol on graphical interface by user's preference. Remote connection with SSH protocol transmits the user input in the command line to a remote computer and receives command line output from the remote computer. Remote connection with VNC protocol supports a graphical interface such that the user can operate the remote computer in the same way they operate it locally. As shown in Figure 2, when the user clicks the button for VNC remote connection, the application will open a new window and start the user authentication.

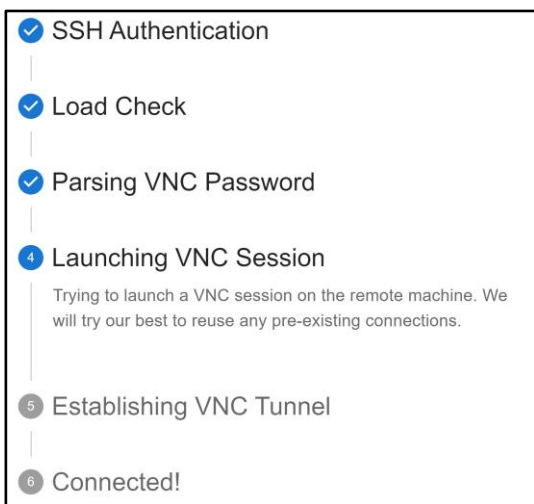


Figure 2: VNC Loading Interface

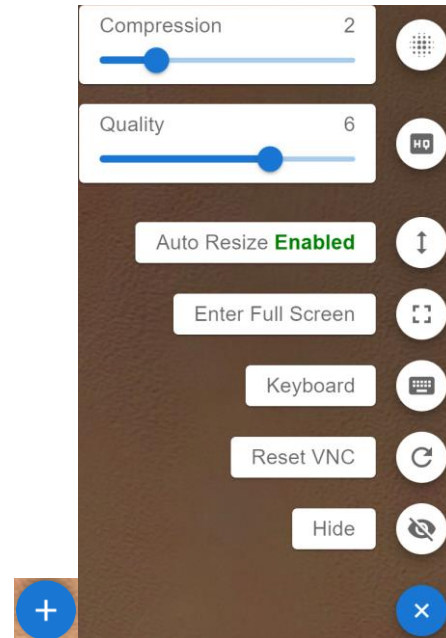


Figure 3: Interface of the VNC "Advanced" Menu

After the VNC remote connection is established, as shown in Figure 3, the user can click the button with '+' at the bottom right corner to open the "Advanced Menu". The user can then adjust the remote connection quality by changing the "compression" and the "quality" with scales from 0 to 9 based on their Internet quality. The auto resize feature is enabled by default and can be disabled, which will adjust the remote window based on the window size of the browser. Clicking the "Enter Full Screen" button will hide the taskbar and other applications and fill the browser window with the VNC virtual screen as if using a computer locally.

When users open a VNC session on their mobile devices, they can also toggle a virtual keyboard bar from the "Advanced Menu". As shown in Figure 4, the keyboard bar provides some common keys that are absent on typical mobile virtual keyboards, which allows the user to complete all computer operations on a phone.



Figure 4: Interface of Keyboard Extension

If the user happens to corrupt a VNC session, forget their password, or want to simulate a “logout” action to apply some shell profile settings, they can click on the “Reset” button in the “Advanced Menu”. As shown in Figure 5, a dialog window will pop up and prompt the user to confirm this potentially dangerous action. When they are determined to reset the configurations, the user will be prompted to enter a new VNC password. If the user ever needs to access their VNC sessions in an environment where our application cannot be used, they can use the same password because we intentionally integrate their passwords directly into the password resetting commands without any processing.

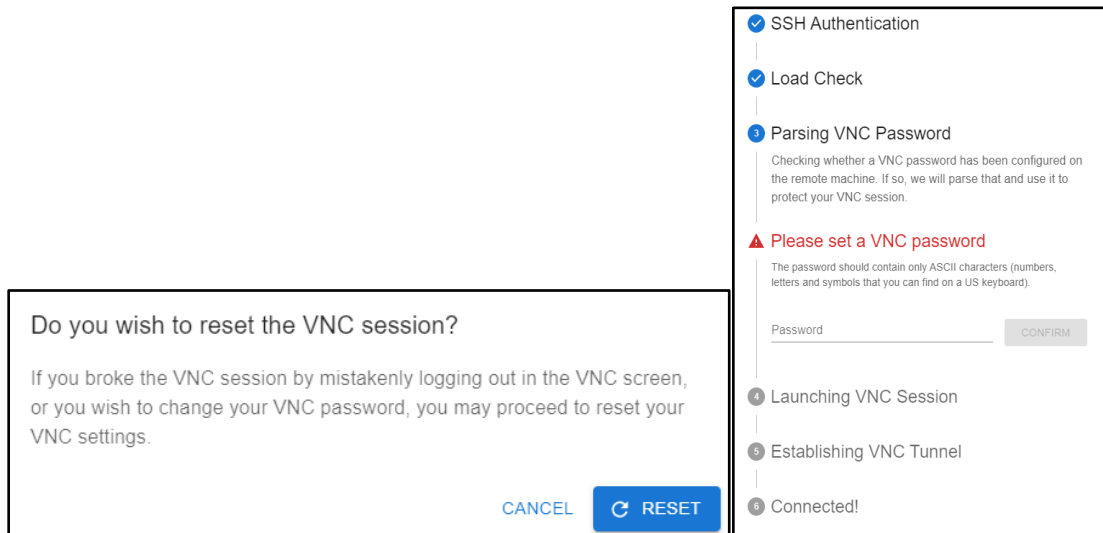


Figure 5: VNC Reset and Password Prompt

Specialised Machine Allocation Mechanism

The “Change Machine” feature is developed to address the uneven workload issue on our university’s laboratory machines. This feature provides automatic workload check prior to making any connection. If there are other users on the same machine currently, a window with machine workload rankings from lightest to heaviest will pop up, such that users can switch from the current machine to any machine in the suggestion list as shown in Figure 6.

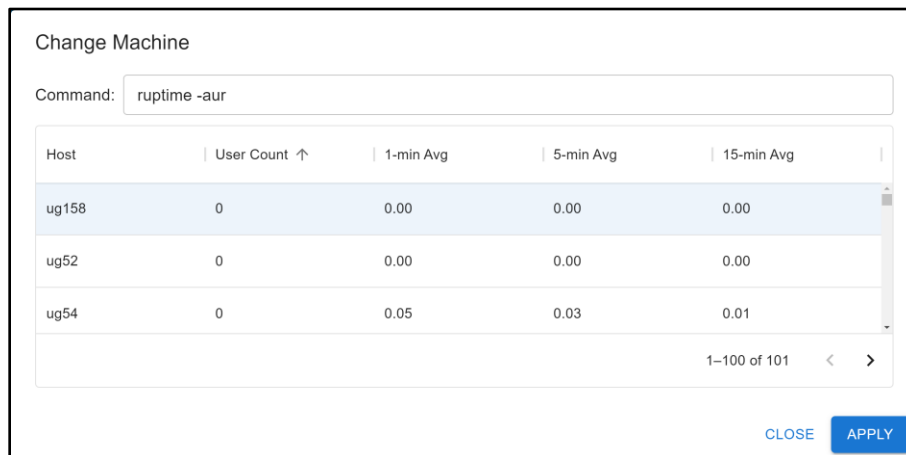


Figure 6: “Change Machine” Window with Hosts Sorted According to Their Loads

The application does not force users to change machines. However, when a new remote session is demanded, the application automatically checks and prompts the user if a high load is detected on the current machine. Since the workload check is dynamic upon live requests, it is optimised at the moment of connection establishment.

File Management Interface

The file management interface aims to simplify daily file operations available in a desktop file manager, with a strong reference from macOS Finder and Google Drive. The underlying protocol is Secure File Transfer Protocol.

All the files in the current working directory will be displayed in a sortable table. When no row is selected, the user can double click on some row to enter or down the entry, depending on whether the entry is a directory or a file. As shown in Figure 7, when a row is selected, the user can double-click on the selected row's file name to rename some file. When multiple rows are selected, the user can click the "Delete" or "Download" button in the top menu bar and the operations will be performed for each entry.

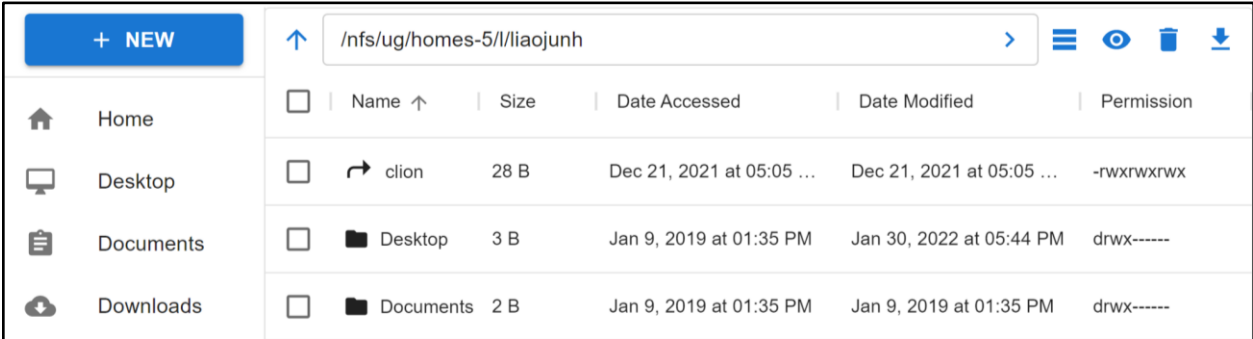


Figure 7: File Management Interface - "File Manager"

If the user wants to upload a file or a folder from the local computer to the target machine, then the user can click on the "New" menu to start the upload. The upload progress will be shown in a pop-up window as shown in Figure 8 so that the user knows whether the file upload has been completed or not.

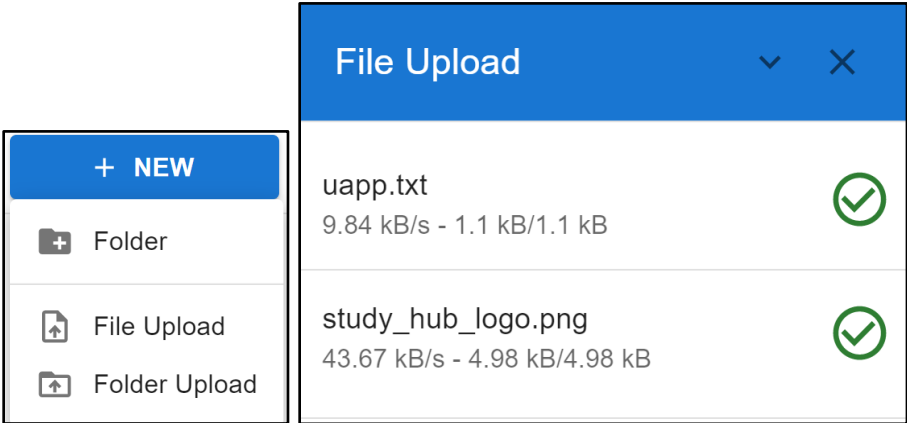


Figure 8: File Upload Panels

As shown in Figure 9, if the user wants to create a new folder in the current working directory, it can also be done by clicking on the “New” menu and selecting the “Folder” option.

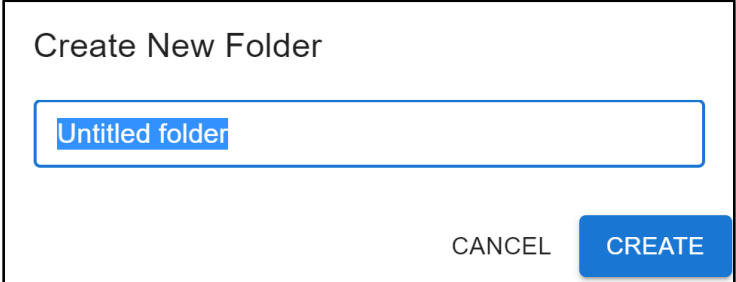


Figure 9: Create New Folder Dialog

As shown in Figure 10, the user can also change the file permission by double clicking on the permission cell of the file. As the user changes the read/write/execute permissions, the client will recalculate the octal representation in real time and display that to the user. The user can also decide whether the permission change should be applied recursively by toggling a switch button. After the user clicks on “APPLY”, `chmod` will run with appropriate parameters as configured by the user.

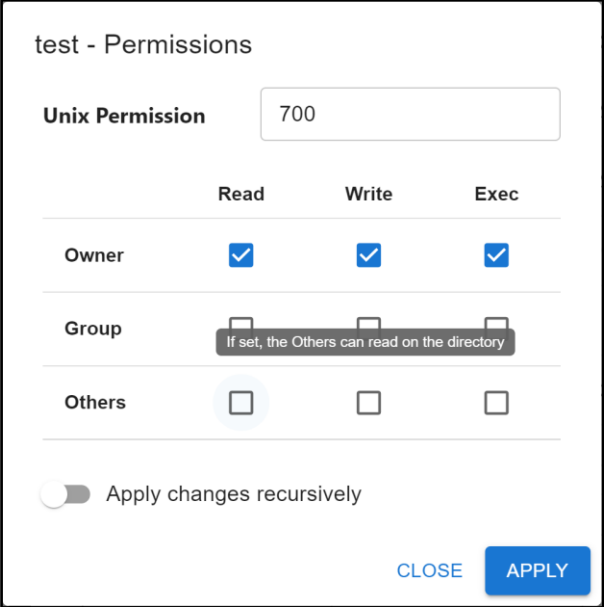


Figure 10: Change File Permission Pop-up Dialog

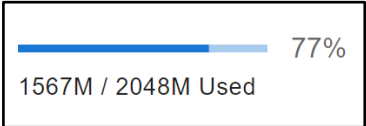


Figure 11: Quota checking display

The quota checking feature is implemented by running the `quota -s` command and parsing the output. The total space allocated by the network’s system administrator is extracted together with the storage that has already been used by the user. By calculating the percentage of the quota used and displaying the value as a partially filled bar, the user can better visualize the quota usage and diagnose any file-related operations more easily as shown in Figure 11.

Desktop version

While the web-based browser version helps to address portability and compatibility issues, a desktop version is also developed to collect early feedback, enhance user information security, and reduce server workload. The desktop version was initially released in August 2021, and the team reached out to multiple course instructors to promote the application. At that early stage of the project, feedback and suggestions from student and course instructors were collected to improve the browser version, which shares the same codebase with the desktop. Unlike the browser version, user information is kept locally in the desktop version, which enhances the security by reducing critical information transmissions over the internet. Also, due to limited resources and funds, our server may lack the capacity to host potentially hundreds of users who may run multiple graphical and terminal sessions at the same time. In case of a server maintenance or shutdown, the desktop version can be downloaded on our GitHub repository's "Release" page and serve as a backup to provide services. Users are also encouraged to switch to the desktop version during peak hours such as assignment deadlines to reduce server workload.

Feedback and Evaluation

On November 26, 2021, a survey was conducted in a third-year computer-networking course, whose students have taken a first-year programming language course and a second-year graphical program design course. The survey questions can be categorised into two sections as shown in Appendix A:

1. Demands for remote connection.
2. Difficulties faced in working remotely.
3. iCtrl application ratings.

The survey showed demands for remote connections have increased especially during the pandemic. As shown in Figure 12, the majority (81.8%) of the students agree that working remotely can increase their productivity given the current resources. This survey shows the potential of iCtrl as more students are likely to continue working remotely even after the pandemic.

While the demands for remote connections are high, students face a few difficulties in establishing those connections on their own. The difficulty is mainly due to unfamiliarity with shell commands which have no graphical output, setup remote connections and file transfer. To address those problems, iCtrl is able to provide both graphical and shell interface that allows users to flexibly choose the environment that they are most familiar with. Users can access laboratory machines by simply entering their username and password while the rest of the progress is automatically handled by iCtrl, which seamlessly sets up the connection by hiding complicated connections in the background. Along with user profile management, no password

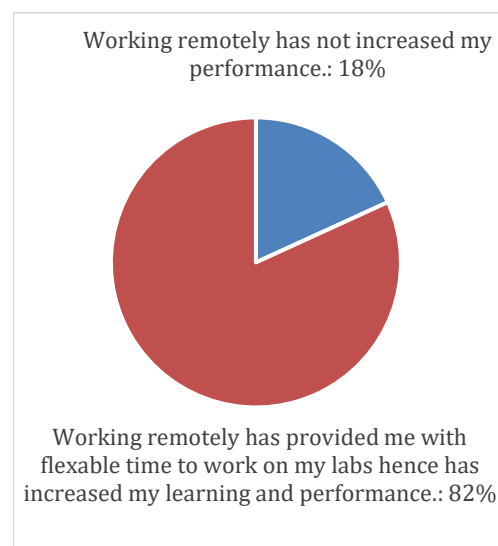


Figure 12: Productivity in remote work vs. in-person work

memorization is required for later usage. iCtrl also features file transfer to upload and download files through the web application between local computers and remote laboratory machines. Feedback and ratings of iCtrl is mostly positive. The survey shows iCtrl not only delivers user-friendly experience, but also provides vital features in programming tasks, which contributes to continuous increase in new users. When graphical interfaces are required, for instance, in the graphical programming course, 47.7% of the students have chosen iCtrl as their default way to conduct graphical related programming tasks. Also, as shown in Figure 13, 55.2% students have given a perfect score to iCtrl, which indicates its popularity and robustness among students.

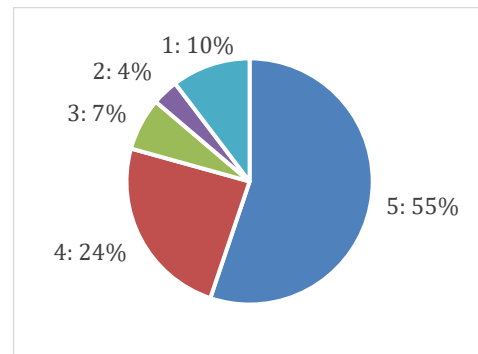


Figure 13: iCtrl rating on a scale of 5, where 5 is the highest rating

Future work

The further collaboration with our university can allow us to integrate iCtrl application software with the official course websites. Once the team is granted access to the official course website server, the entire application can be migrated to provide seamless experience for all students. The iCtrl application interface is designed such that to match the current university's course website both in functionalities and UI interface. This will eliminate students accessing multiple websites and facilitates ease of usage. The most challenging part will be integration of users' credentials as students already have a valid user account on the course website as well as on the iCtrl. The step to accomplish this task also depends on the database design for the course website. Therefore, the integration process is expected to extend over a few months after access is granted.

A better approach to receive feedback from students is to embed a diagnostic data collector into our application. It will allow us to collect usage data and bug reports of different features. We can then use those data to decide which features to further enhance, and the bug reports should shorten the period of identifying any technical defects in our application. The collector will automatically upload the required data to our dedicated server upon users' agreements, which helps us to collect students' feedbacks more effectively compared to the survey method.

Online compilers with text editors can be offered to increase coding efficiency. There are a few situations where no graphical interface is required to complete a coding task. In response to non-graphical interface usage, the current approach is terminal connection that only allows the user to interact in a Shell application. The compilation has to be done manually. Therefore, the application can integrate a few compilers that support the majority of common programming languages such as C++, Python, and Java. In addition, a modern text editor can boost the efficiency of writing code directly from the web application. This combination should serve as an ultimate coding environment for all professional users.

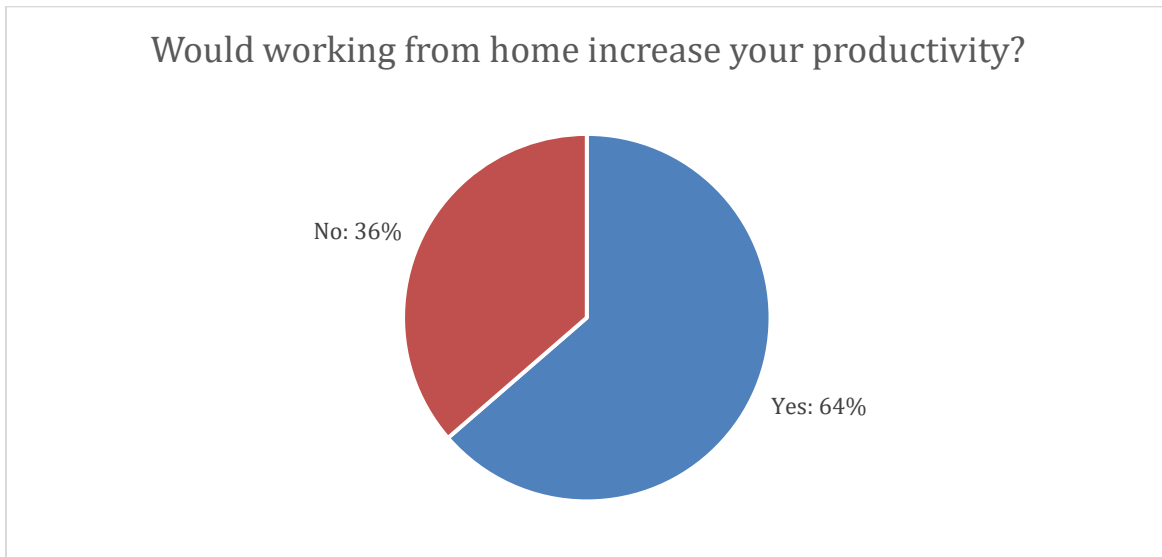
Reference

- [1] M. Kuchenbuch, G. D'Onofrio, E. Wirrell, Y. Jiang, S. Dupont, Z. M. Grinspan, S. Auvin, J. M. Wilmshurst, A. Arzimanoglou, J. H. Cross, N. Specchio, and R. Nabbout, "An accelerated shift in the use of remote systems in epilepsy due to the COVID-19 pandemic," *Epilepsy & Behavior*, vol. 112, p. 107376, 2020.
- [2] D. Garrote Sanchez, N. Gomez Parra, C. Ozden, B. Rijkers, M. Viollaz, and H. Winkler, "Who on earth can work from home?," *The World Bank Research Observer*, vol. 36, no. 1, pp. 67–100, 2021.
- [3] What Does Working Remotely Mean for the Planet? — Sustainability (2021). [Online]. Available: <https://www.shopify.ca/blog/working-remotely-for-the-planet>. [Accessed October 12, 2021]
- [4] TigerVNC. Available: [Online]. Available: <https://tigervnc.org/>. [Accessed October 12, 2021]
- [5] MobaXterm free Xserver and tabbed SSH client for Windows. [Online]. Available: <https://mobaxterm.mobatek.net/>. [Accessed October 12, 2021]
- [6] Chrome Remote Desktop. [Online]. Available: <https://remotedesktop.google.com/>. [Accessed October 12, 2021]
- [7] Remote Access - Current Engineering Undergraduates. [Online]. Available: <https://undergrad.engineering.utoronto.ca/undergrad-resources/engineering-computing-facility-ecf/remote-access/>. [Accessed October 12, 2021]
- [8] V. Betz, "Finding a lightly loaded UG machine for VNC sessions," unpublished, private communication, October 12,. [Online]. Available: https://drive.google.com/file/d/1GwIms8ur_PtVuRUruQmq9OVnfuehYmLx/view?usp=sharing [Accessed October 12, 2021]

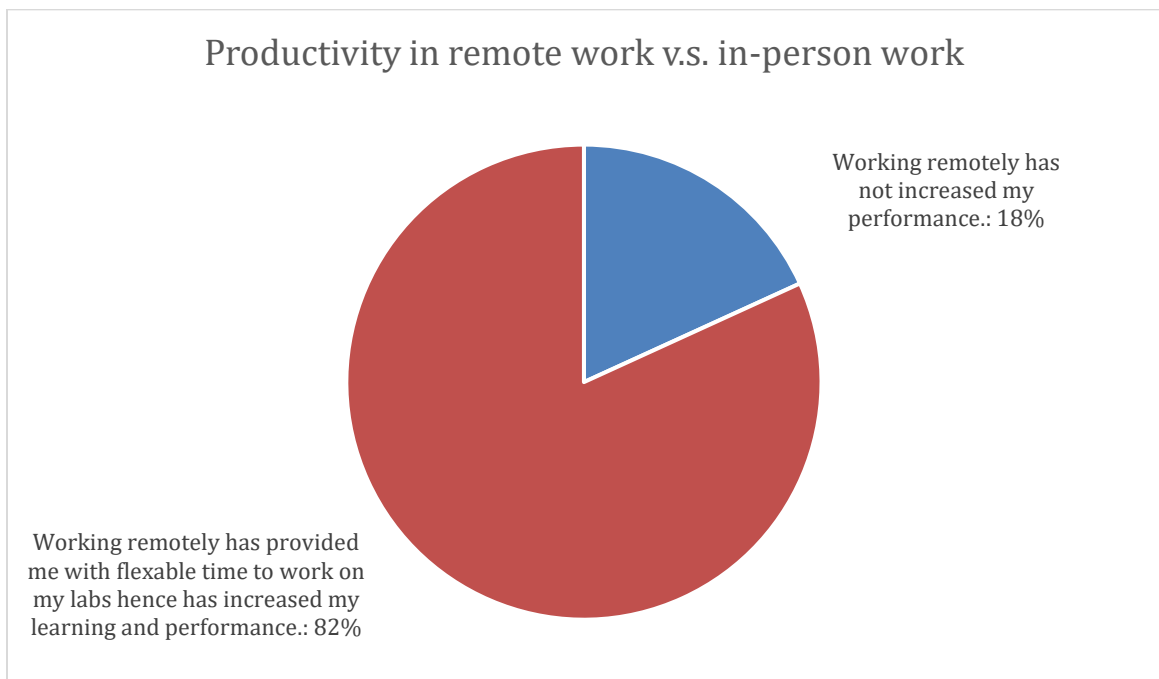
Appendix A: Remote Connection Software Survey

This survey was distributed to the students of a third-year undergraduate computer-networking class, who had learnt C-Language programming on Linux computers in a first-year course.

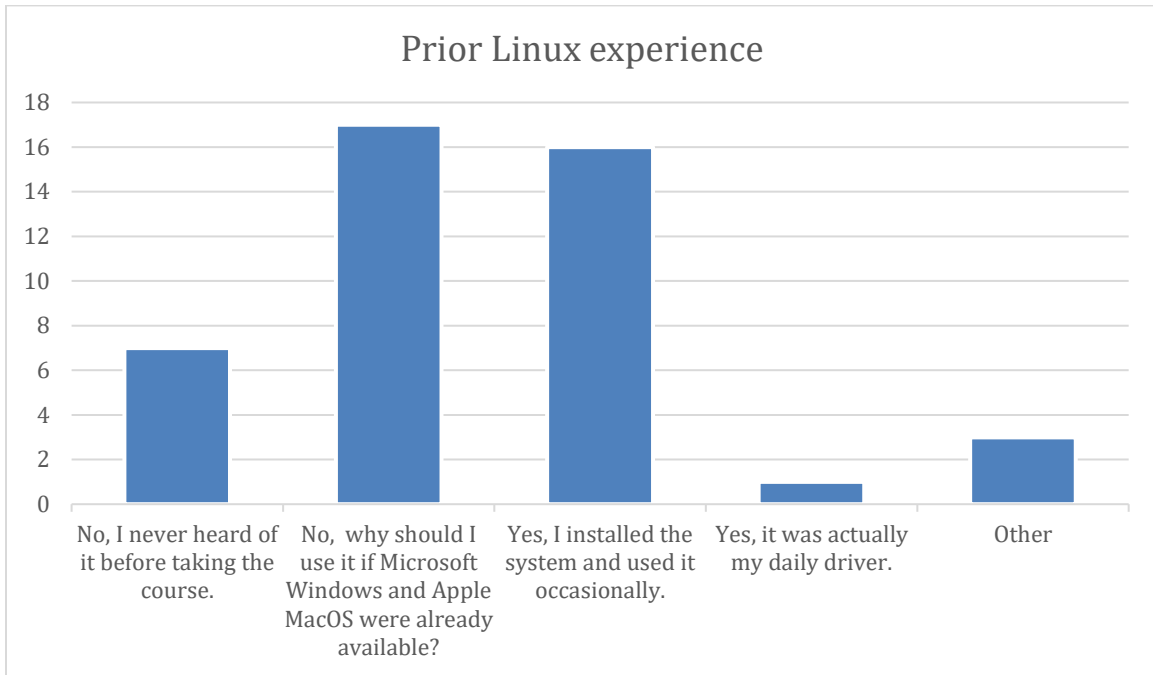
1.1 Would working from home (Remotely) increase your productivity and allow you to learn more effectively?



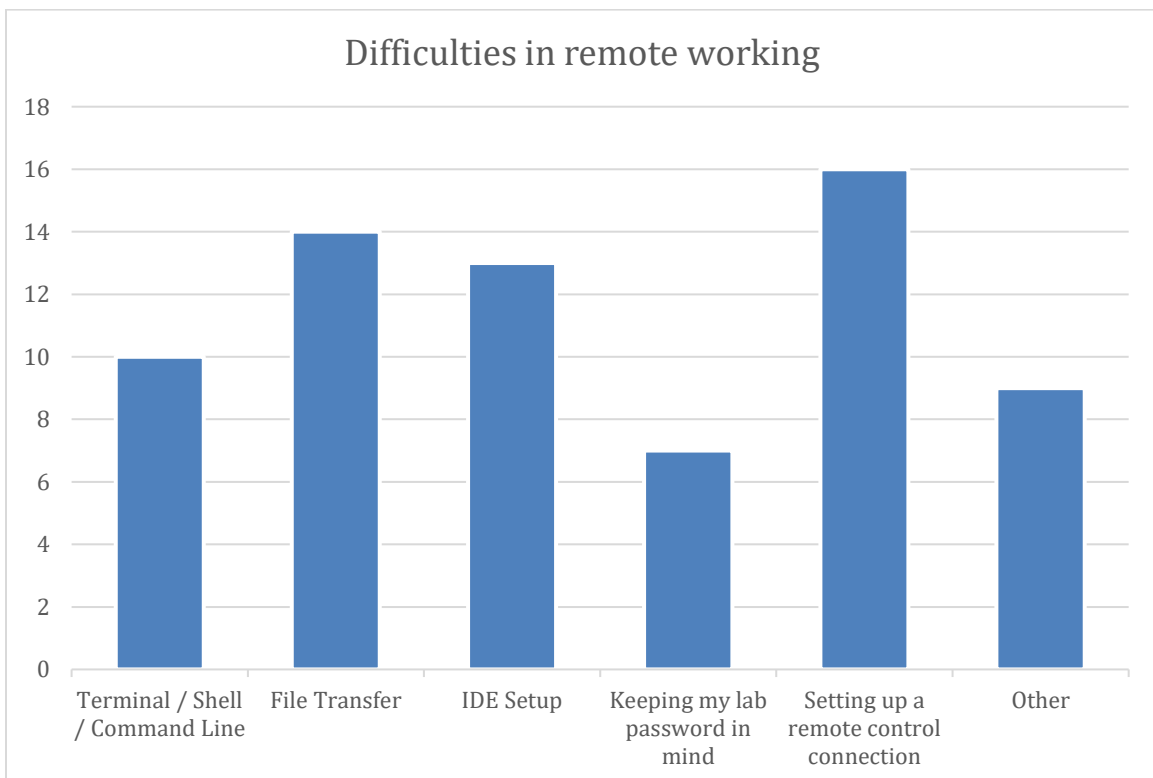
1.2 Given the currently available resources, how do you compare your productivity between coding remotely and working in the lab in-person?



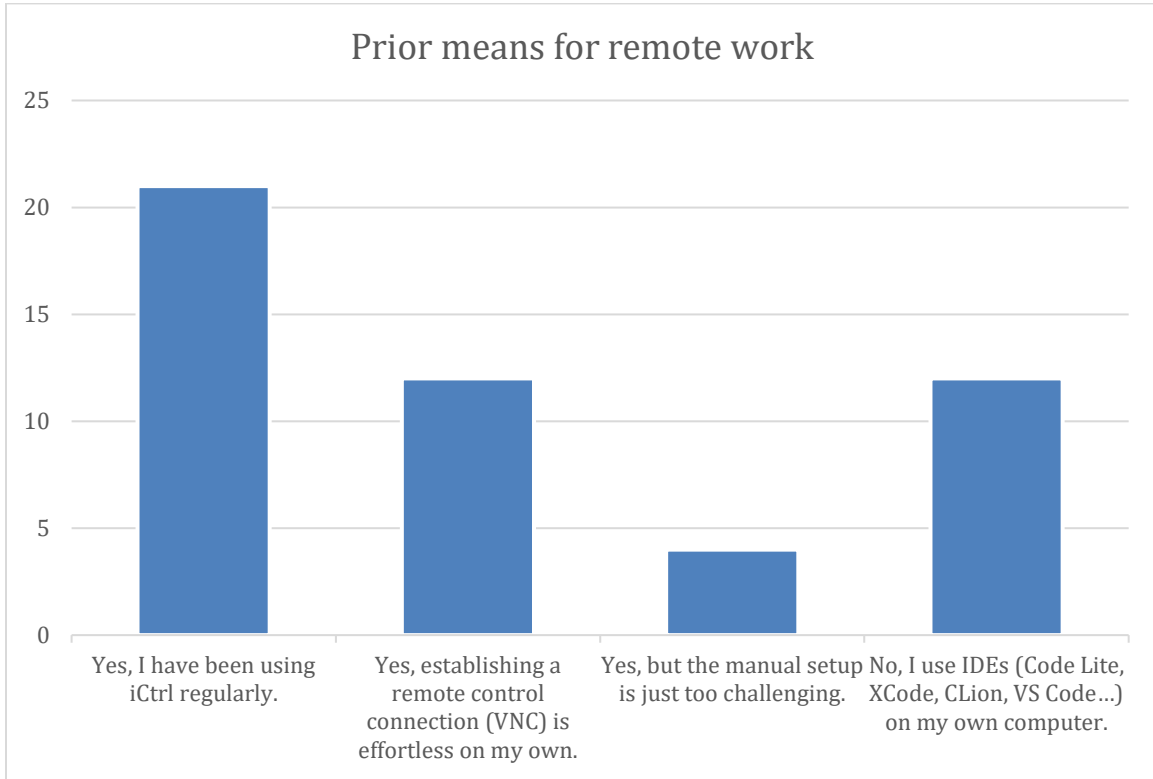
2.1 Did you use Linux before taking any programming courses?



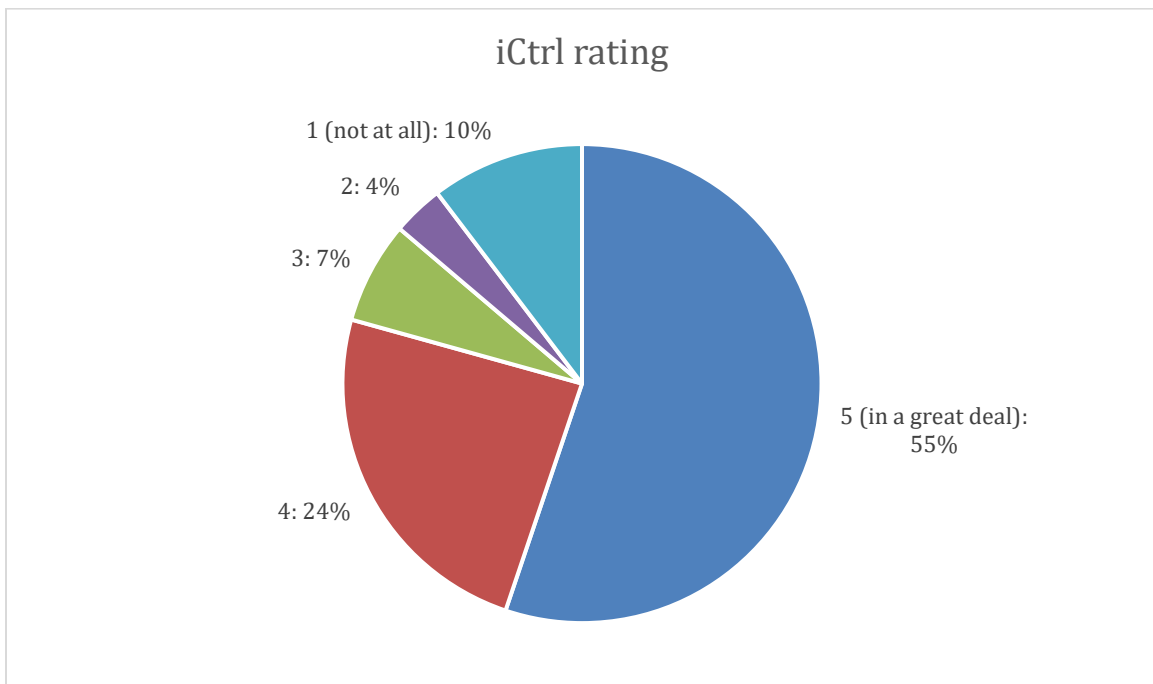
2.2 What are some difficulties in remotely working from home?



3.1 Have you ever tried to write code using a remote-control connection (VNC) in this course?



3.2 If you have used iCtrl, then to what degree iCtrl has increased your performance and learning?



3.3 If you have used iCtrl before, any comments or suggestions?

Respondent #	Response
44	Everything is perfect
42	I haven't used it.
39	Very easy to use, hard to find. I recommended it to a classmate and I took us 30 min to find the link to download
38	very good
36	iCtrl makes using remote connection effortless. Grateful that it exists.
33	I'm not sure its the same but I believe this is a python script written by a TA right? I used this in ECE297 and it helped me a great deal.
28	no
22	I've used it only a few times, but the times I have used it it works really well and has a very intuitive UI. Good job.
21	Add a 'kill all' button so I don't need open terminal to terminate all my previous VNC processes.

Appendix B: Market Research in Remote Control Software Supported OS Platforms

Remote Control Software	Supported OS Platforms
MobaXterm	Windows
PuTTY / KiTTY	Windows
WinSCP	Windows
RealVN	Windows, macOS
TigerVNC	Windows, macOS

Table B.1: Top 5 Free Remote-Control Software with their Supported OS Platforms