Streamlining Computer Maintenance with Automation Scripts

Chase Twyman

Kennesaw Mountain High School

1898 Kennesaw Due West Rd NW, Kennesaw, GA 30152

Acknowledgements

I conducted this research from August 8, 2024, to December 1, 2024, at True-IT Inc. I want to thank my research and internship teachers, Ms. Dennis and Dr. Hunt, and I would like to thank my True-IT Inc. internship mentor, Cathy Stump, who funded the materials for this research. This research would not have been possible without them.

Table of Contents

Streamlining Computer Maintenance with Automation Scripts

Chapter 1: Introduction

According to Mearian (2022), a Windows computer update can take up to six hours to finish due to the large size of these updates. Although most updates take less than an hour, devices that are infrequently used require a consistent connection to install these extensive updates (Mearian, 2022). This explains why when updates first become available, many individuals are compelled to delay important updates to reduce workflow interruptions (Rajivan et al., 2020). While computer updates can be inconvenient for users, uninstalled updates pose more of a significant challenge for computer repair offices (Pourhadi, 2019). Technicians in computer repair offices are responsible for optimizing and keeping computers updated to improve the computer's lifespan, performance, and security (Bika, 2022; Huculak, 2022; Pourhadi, 2019). Due to these upkeep expectations, computer technicians spend a significant amount of time installing necessary updates and applying optimizations on computers, which delays awaiting workplace tasks and decreases workflow efficiency (Mearian, 2022; Pourhadi, 2019; Wingate, 2016). The lack of cost-effective tools makes automating the computer optimization and update process impractical for computer technicians (Pourhadi, 2019). Additionally, the documentation process of computer maintenance can often be tedious (Wingate, 2016). According to Wingate (2016) and Hanna et al. (2014), computer technicians must document who conducted the maintenance, the maintenance activities performed, the time the maintenance was performed, and the verification of completed maintenance tasks.

Currently, computer technicians perform numerous maintenance tasks manually (Bika, 2022). The most common maintenance tasks performed manually are applying updates, installing software, optimizing systems, and keeping documentation of applied maintenance

(Andersen & MoldStud Research Team, 2024; Bika, 2022; Hanna et al., 2014; Pourhadi, 2019; Urbitsch, 2023; Wingate, 2016). Common optimizations that computer technicians can apply to computers are removing temporary files and defragmenting hard drives (Huculak, 2022; Urbitsch, 2023). For the documentation process, computer technicians document who conducted the maintenance, what time the maintenance was conducted, which maintenance tasks were completed successfully and unsuccessfully, and hardware and software specifications (Andersen & MoldStud Research Team, 2024; Hanna et al., 2014; Wingate, 2016).

Purpose of the Study

The purpose of this research is to develop an automation script for a Raspberry Pi Zero 2 W (RPZ2W) to execute maintenance tasks on computers in order to reduce the time that computer technicians spend on maintenance tasks and documentation. This research is important because it provides a cost-effective tool that automates the computer maintenance process without using a physical mouse, keyboard, or monitor to complete tasks. The outcomes are significant to computer technicians in computer repair companies because this study provides a product that will leave more time for other repair and maintenance tasks that require more cognitive and physical skills that cannot be automated (Shin et al., 2024). The goal is to reduce the time computer technicians must spend on computer maintenance tasks by 10% by automating repetitive tasks, such as checking for updates, collecting device hardware and software information, and installing necessary software (such as antivirus if not present). This automation benefits the computer technician and the business they are working under due to the time it saves. This study programs and analyzes the automated RPZ2W's impact on improving workplace performance by considering accuracy, time saved, and number of completed tasks. Usually, before a computer technician works on a computer, a mouse, keyboard, and

graphics cable must be connected before a computer is ready to operate. Still, the automated RPZ2W can start working immediately as soon as this device is plugged into a computer without requiring a mouse, keyboard, or screen to be set up.

Research Questions

Q1: How much can the automated Raspberry Pi Zero 2 W impact workplace performance?

Q1.1: How many computers can the automated Raspberry Pi Zero 2 W conduct maintenance on per workday day?

Q1.2: How much time can technicians save by conducting computer maintenance with the automated Raspberry Pi Zero 2 W?

Q2: How accurate is an automated Raspberry Pi Zero 2 W at automating maintenance tasks?

Hypotheses

H1: Using an automated Raspberry Pi Zero 2 W in computer maintenance will increase workplace performance by at least 10%.

H1.1: The automated Raspberry Pi Zero 2 W will be able to conduct maintenance on at least two computers per workday.

H1.2: The automated Raspberry Pi Zero 2 W will reduce the time technicians conduct computer maintenance by 48 minutes (10% of an eight-hour workday).

H2: The automated Raspberry Pi Zero 2 W will be able to perform computer maintenance tasks with an accuracy of at least 99%.

Definition of Key Terms

Automation Script

A program that runs on a machine to collect, send, and modify data to perform tasks that would otherwise need to be done manually (Pourhadi, 2019).

Automated Raspberry Pi Zero 2 W (RPZ2W)

A Raspberry Pi Zero 2 W that has the study's automation script on it.

Raspberry Pi Zero 2 W (RPZ2W)

A lightweight, miniature Linux-based computer that runs its own operating system and has a multifunctional USB port that can execute mouse and keyboard events (Contino, 2024).

Computer Maintenance

Computer maintenance is the tasks technicians conduct after computer repairs to ensure proper functionality. These tasks include applying updates, installing software, optimizing systems, and keeping documentation of applied maintenance (Andersen & MoldStud Research Team, 2024; Bika, 2022; Hanna et al., 2014; Pourhadi, 2019; Urbitsch, 2023; Wingate, 2016).

Overfitting

Overfitting occurs when a model is trained on sample data that is not representative of the general population, limiting the model's ability to successfully complete each task (Al-Bataineh et al., 2021).

Chapter 2: Research Method

Research Design and Methods

I used an engineering research design and a mixed methods approach to develop and test a script for the RPZ2W to automate repetitive computer maintenance tasks. Scripts are programs that can automate repetitive computer maintenance tasks to increase workplace productivity (Pourhadi, 2019). Installing scripts can be inefficient, as computer technicians must utilize a mouse and keyboard to enter credentials, install the automation script, and run the automation script (Hanna et al., 2014; Pourhadi, 2019). This process can be made more efficient by installing the automation script on the Raspberry Pi Zero 2 W (RPZ2W) that utilizes the RPZ2W's

functionalities to perform computer maintenance (Contino, 2024). Unlike automation scripts installed on the Windows computer, the automated RPZ2W can receive power and execute maintenance tasks when plugged into the Windows computer, regardless of whether the Windows computer is signed in or not (Contino, 2024). Therefore, this study develops a script for the RPZ2W to automate common computer technician maintenance tasks.

To create the script, I used Microsoft's Visual Studio Code, a coding software tool, since this software is used to develop scripts in other scripting studies to achieve similar tasks (Hanna et al., 2014). Python was the primary programming language for this script since the packages I found that utilize the multifunctional USB port on the RPZ2W to simulate mouse and keyboard events are coded in Python (thewh1teagle, 2024). The multifunctional USB port on the RPZ2W allows me to replicate computer maintenance tasks as if a technician were conducting them.

To determine the most common and important maintenance tasks to test, I used qualitative data collection of maintenance checklists from computer repair companies and credible resources (Andersen & MoldStud Research Team, 2024; Bika, 2022; Hanna et al., 2014; Pourhadi, 2019; Urbitsch, 2023; Wingate, 2016). The computer maintenance tasks that the RPZ2W conducts are checking for Windows updates, installing software, clearing temporary files, defragmenting hard drives, and documenting results and system hardware and software information, as these tasks are performed by other computer technicians (Andersen & MoldStud Research Team, 2024; Bika, 2022; Hanna et al., 2014; Huculak, 2022; Pourhadi, 2019; Urbitsch, 2023; Wingate, 2016). Since part of the engineering and design process is testing the solution, I used quantitative data collection to measure the effectiveness of using an RPZ2W as an automation tool (Al-Bataineh et al., 2021; Hanna et al., 2014). The quantitative variables used to test the effectiveness of the automated RPZ2W for the engineering and design process are the

duration of each maintenance task and the number of failed and completed maintenance tasks (Andersen & MoldStud Research Team, 2024; Hanna et al., 2014; Wingate, 2016).

This study has some limitations. Firstly, this study only applies to Windows 10 and Windows 11 computers, limiting the population of computers. Furthermore, these computers must have at least a USB-A port and an HDMI port for the automated RPZ2W to automate tasks.

There are multiple assumptions in this study. This study assumes that the common maintenance tasks that computer technicians perform are applying Windows updates, installing software, optimizing systems, and keeping documentation of applied maintenance (Andersen & MoldStud Research Team, 2024; Bika, 2022; Hanna et al., 2014; Pourhadi, 2019; Urbitsch, 2023; Wingate, 2016). Although these are the most common maintenance tasks performed, other computer technicians will likely perform different or more maintenance tasks that are not automated by my study's script. Secondly, to keep a consistent metric, this study assumes that computer technicians work the average eight-hour shift since technicians have day-to-day responsibilities and can be called in when there is a significant technical problem (Bika, 2022).

Materials and Instruments

This research uses a Raspberry Pi Zero 2 W (RPZ2W), a lightweight computer capable of emulating various USB devices (mouse, keyboard, storage device) and analyzing screen input from its camera port by using an HDMI (High-Definition Multimedia Interface) to CSI (Camera Serial Interface) adapter by Waveshare (HDMI to CSI Adapter, n.d.; Contino, 2024). This adapter converts the HDMI signal from the host computer into video data that is readable to the RPZ2W. To reiterate, this study develops an automation script for the RPZ2W to automate computer maintenance tasks more effectively (Contino, 2024). Unlike automation scripts installed on the Windows computer, the automated RPZ2W can receive power and execute

maintenance tasks when plugged into the Windows computer, regardless of whether the Windows computer is signed in or not (Contino, 2024). For the power supply and mouse and keyboard output, the RPZ2W uses two USB type A to USB mini adapters, one to transfer power, and the other to transfer mouse and keyboard events (Contino, 2024). The device is connected to any Windows 11 or Windows 10 desktops and laptops to perform maintenance tasks.

To develop the automation script for the RPZ2W, I used Microsoft's Visual Studio Code since this software is used to develop scripts in other scripting studies to achieve similar tasks (Hanna et al., 2014). With Visual Studio Code, I created the automation script using the Python coding language since the Zero-HID Python package utilized the RPZ2W's functionalities to emulate emulating mouse and keyboard events (thewh1teagle, 2024).

The resources used to build the device for this study are the Raspberry Pi Zero 2 W and the HDMI-CSI adapter. The Raspberry Pi Zero 2 W costs $15, and the adapter costs $25 (HDMI to CSI Adapter, n.d.; Contino, 2024). Therefore, this device is economically feasible in computer repair companies as this device was cheaper than the combined cost of a mouse, keyboard, and monitor at my computer repair internship workspace.

Operational Definition of Variables

Effectiveness

Measuring the automated RPZ2W's accuracy and efficiency determines its effectiveness. If the automated RPZ2W has an accuracy of 99% and saves technicians 10% more time in their eight-hour workday by automating repetitive computer maintenance tasks, then the automated RPZ2W would be effective (Hanna et al., 2014; Pourhadi, 2019).

Data Collection, Processing, and Analysis

This study analyzes video data from the host computer through an HDMI to CSI adapter.

The HDMI to CSI input collects button location data on the host computer as well as data from the computer's System Information page to collect identifying computer information such as MAC (Media Access Control) Address, operating system type, processor information, memory size, slots of memory installed, storage size, network adapter settings (including Bluetooth), desktop name, and device manufacturer (Andersen & MoldStud Research Team, 2024; Hanna et al., 2014; Wingate, 2016). The procedure to create this RPZ2W device would be first to use the Raspberry Pi Imager to install the configuration of the Raspberry Pi OS Lite (Bookworm) 64-bit using a 64-gigabyte microSD card. The custom configuration would involve enabling SSH (Secure Shell) and setting a username and password for the RPZ2W. Then, I configured the RPZ2W by entering commands into its terminal based on the zero-hid documentation (thewh1teagle, 2024).

To program the RPZ2W, I used Microsoft Visual Studio Code and saved the project file to a GitHub repository. I used the Python programming language inside of Visual Studio Code to create the basis of my project since the packages required to utilize the RPZ2W's multifunctional USB to emulate mouse and keyboard outputs are coded in Python (thewh1teagle, 2024). First, I imported the Zero-HID Python package to emulate a mouse and keyboard output from the RPZ2W, py-tesseract for optical character recognition, FFMPEG for capturing frames of the RPZ2W's video stream, and Python's CSV library for analyzing pixel colors. To identify the host computer's OS version, I used OCR (optical character recognition) to get the text displayed by the Windows search bar since Windows 10 and 11 have different search bars. Then, the RPZ2W checks for Windows updates and navigates to its emulated USB mass storage device to install the antivirus software. Next, the RPZ2W optimizes the computer by clearing temporary files and defragmenting the hard drive (Huculak, 2022; Urbitsch, 2023). Then, the automated

RPZ2W opens System Information and uses OCR to collect identifying computer information such as MAC (Media Access Control) Address, operating system type, processor information, memory size, slots of memory installed, storage size, network adapter settings, and the computer's manufacturer to create a detailed patch report. Finally, after the computer reboots, the RPZ2W checks for additional Windows Updates before printing out a unique patch report for that computer. Additionally, the patch report will include the duration of maintenance work and the number of maintenance tasks that were successfully completed. Detailed information about the specific lines of code used in the automation script can be found on my GitHub repository (Twyman, 2024).

Chapter 3: Findings and Results

Results

The two research questions of this study determine the effectiveness and significance of implementing an RPZ2W to conduct maintenance tasks in the workplace. Q1.1 was addressed by collecting the average time the RPZ2W spent conducting maintenance tasks on computers on a convenience sample of 12 computers at a computer repair office. The following table shows the time spent on each computer.

Table 1

Time Spent Conducting Maintenance on 8 Computers

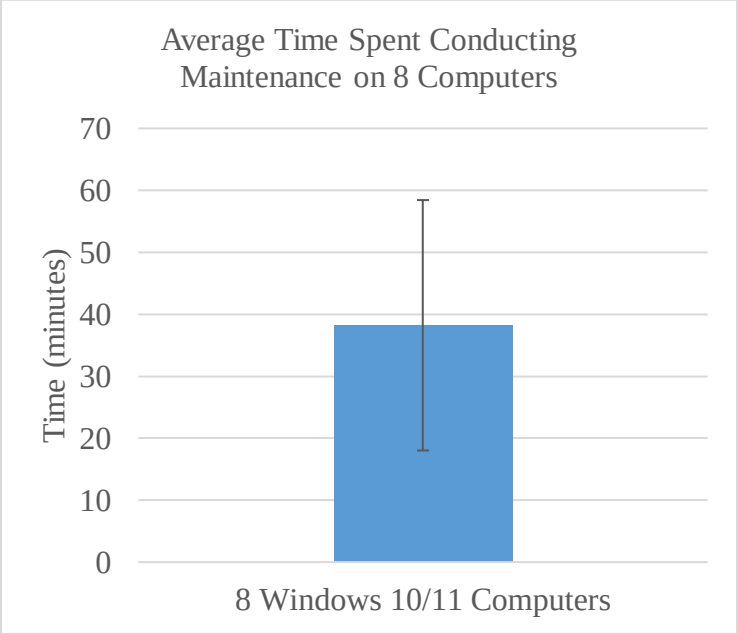
Computer (Make and Model)	Time (minutes:seconds)
Computer 1	20:13
Computer 2	25:01
Computer 3	10:05
Computer 4	47:51
Computer 5	68:19
Computer 6	38:22
Computer 7	67:49
Computer 8	28:08
Average Time	38:13.5

Standard Deviation	20.21

Note. Each computer was subjected to the same type and amount of maintenance tasks.

Figure 1

Bar Graph of Average Time Spent Conducting Computer Maintenance



The average time to conduct computer maintenance was used to approximate the number of computers in an eight-hour workday, a 21-workday month, and a 260-workday year. Because this device can continue to operate outside of working hours, I took the ceiling of the average instead of the floor. Therefore, computer technicians can conduct maintenance on approximately 13 computers per eight-hour workday. Although each computer was chosen from a convenience sample rather than a random sample, each computer was unique, with different hardware and processing power. Therefore, taking a convenience sample of computers did not significantly affect the validity of my data. The time spent on conducting maintenance for each computer was collected by the automated RPZ2W itself and started from when the automated RPZ2W ran its first maintenance task rather than when it was powered on. Since each computer has a different port layout, the time spent attaching the RP2W to each computer was variable and, thus, was not

included in the computer maintenance duration. Not factoring in the time it takes to set up the RPZ2W is a delimitation of this study.

Table 2

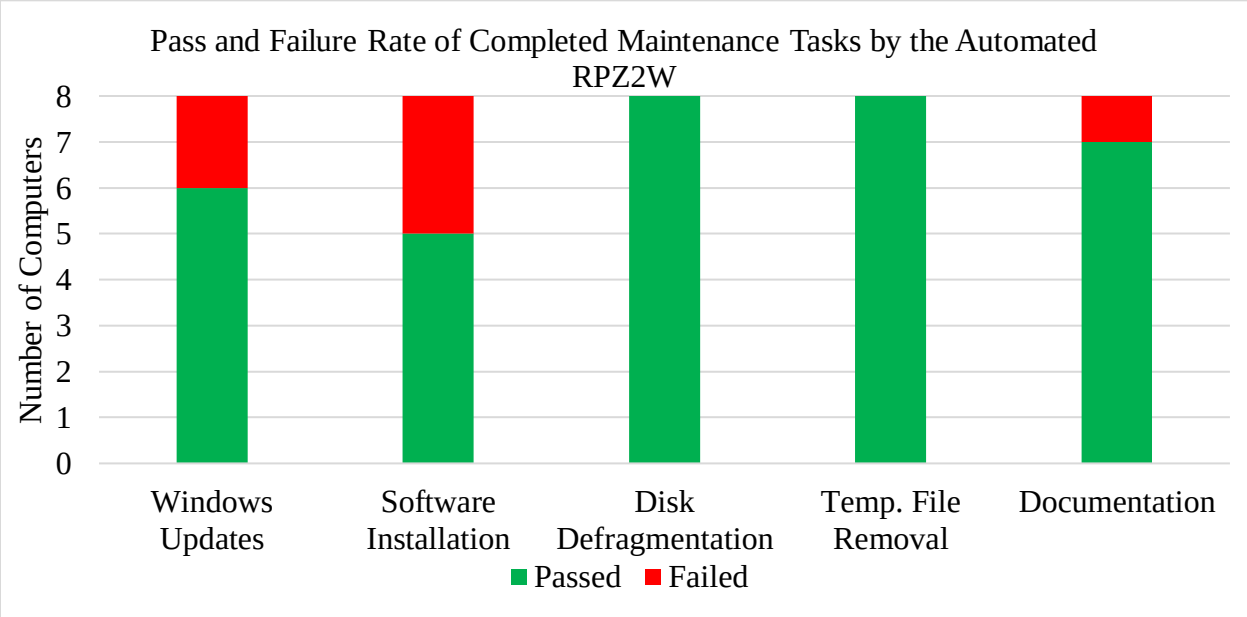
Pass and Failure Rate of Conducted Maintenance Tasks by the Automated RPZ2W

Maintenance Task	Tasks		Task Pass%
	Passed	Failed	
Windows Updates	6	2	75%
Software Installation	5	3	62.5%
Hard Drive Defragmentation	8	0	100%
Temporary File Removal	8	0	100%
Hardware and Software Documentation	7	1	87.5%
Mean			85%
Standard Deviation			16.30

Note. Passed tasks are completed tasks with the expected result, and failed tasks are incomplete or provide unexpected results.

Figure 2

Pass and Failure Rate of Conducted Maintenance Tasks by the Automated RPZ2W



Software installation had the lowest passing rate while performing disk defragmentation and removing temporary files had the highest passing rate. When the automated RPZ2W fails a

maintenance task, that failure is included in its generated maintenance report. As this device is the only kind of external Windows computer maintenance tool, there is little to compare its performance to other than its pass rate. Generally, a pass rate of at least 99% is considered acceptable since one of the aspects of automated tools is the lack of human intervention (Hanna et al., 2014). This statistical test assumes that a completed task is a task that executes completely with total accuracy, and a failed task is a task that is not completed or is completed incorrectly. The automated RPZ2W had an accuracy of 85% for passed maintenance tasks.

Evaluation of Findings

The duration of the computer maintenance was the other data recorded by the RPZ2W, which had millisecond precision. Instead of manually documenting how long each maintenance session took, the RPZ2W starts a timer from when it is initiated to when the last maintenance task is completed. Using the RPZ2W onboard computer chip to measure the duration meant there was no human error in the data collected. The quality of the time data was extremely accurate and reliable. The results obtained were expected based on the previous literature about data-driven scripting for automating computer maintenance (Hanna et al., 2014). The OCR was 100% accurate during this research. I used the data collected by the OCR to generate documentation. When an external window obscured the system information window about the computer hardware and software information, the OCR could not read the software and hardware information and, therefore, provided an incomplete documentation report.

The number of computers that the RPZ2W could conduct maintenance on in an eight-hour workday was very reliable but not accurate. Each day, a different number of computers need computer maintenance, and the RPZ2W will most likely not be in operation during those eight hours. Since the standard deviation of the duration of maintenance tasks for eight

computers was 20.21, the data collected about the average time it took to conduct computer maintenance was accurate but unreliable since each computer may not need to go through each maintenance task. For instance, if a computer has the latest Windows updates, certain maintenance tasks will not be performed, shortening the duration of maintenance. Recording the average time spent on computer maintenance was the strongest metric for increasing workplace performance, and the data gave good insight into how much time the RPZ2W could save per year instead of per day.

Chapter 4: Implications, Recommendations, and Conclusions

This study develops a Python program to automate maintenance tasks on the RPZ2W. The program utilizes Python packages that run on the RPZ2W's operating system. The program can send mouse movements and keyboard strokes and obtain video input from the host computer using the RPZ2W's USB emulation and HDMI to CSI adapter. The maintenance tasks the automated RPZ2W completes are checking for updates, installing software, disk defragmentation, clearing temporary files, and collecting system hardware and software information.

Implications

Q1 evaluated how much the automated RPZ2W could impact computer technicians' workplace performance. Q1 could be answered by using the answers from Q1.1 and Q1.2. To measure the impact of the RPZ2W, Q1.1 finds the number of computers that a computer technician can conduct maintenance on using the automated RPZ2W, and Q1.2 finds the amount of time technicians can save by measuring the amount of time it takes for the automated RPZ2W to conduct five maintenance tasks. The more automated computer maintenance, the more time these technicians can save (Pourhadi, 2019). The statistical analysis of Table 1 answered Q1.1

and Q1.2 since Table 1 measured the duration of the automated RPZ2W conducting computer maintenance. The results from the statistical analysis of Table 1 show that the automated RPZ2W conducts computer maintenance in an average time of 38 minutes and 13.5 seconds. The population of this study applies to computer repair technicians who work eight-hour workdays. Therefore, the automated RPZ2W can conduct maintenance on 13 Windows 10 and 11 computers per eight-hour workday while saving technicians 7.96% more time from their workday for each automated computer maintenance. These findings answer Q1.1 and Q1.2 and support H1.1 and H1.2.

Q2 evaluated how accurate the automated RPZ2W is when implemented in computer repair workplaces. For an automation script to be effective, it should require as little human intervention as possible (Hanna et al., 2014). Therefore, the accuracy of the automated RPZ2W should be as close to 100% as possible. Table 2 evaluates the accuracy of the automated RPZ2W by evaluating the accuracy of the automated RPZ2W for each maintenance task. Since the accuracy of the automated RPZ2W was 85%, it answered Q2 but did not support H2.

There were limitations that may have affected the interpretation of the results for Q1.1 and Q1.2. For example, there were time-based and condition-based maintenance techniques used in the data-oriented script that did not require all maintenance tasks to be run for each conducted computer maintenance (Hanna et al., 2014; Yazdi, 204). The variations in the number of maintenance tasks conducted induced a large standard deviation in the duration of maintenance (20.12), which would affect the amount of time computer technicians save each day. For Q2, outside factors, such as pop-ups, affected the accuracy of the automated RPZ2W as the RPZ2W was not able to give the right inputs into the right windows and, therefore, failed that maintenance task.

Recommendations

This study's device is best used for computer technicians who spend a significant amount of time conducting computer maintenance since this device saves computer technicians an average of 38 minutes and 13.5 seconds per computer. Future studies can further develop this type of maintenance device by adding more conditionals to the maintenance script to improve accuracy and reliability (Hanna et al., 2014; Twyman, 2024). I recommend that future studies create a maintenance script that runs on other types of computers, such as Mac devices, to expand the population of computers on this device that can conduct maintenance on.

Conclusions

In conclusion, the device that this study developed could automate computer maintenance on Windows 10 and 11 computers in an average time of 38 minutes and 13.5 seconds for each computer (7.96% of eight hours). Additionally, the automated RPZ2W can conduct computer maintenance on 13 computers each day. These findings supported hypotheses H1.1 and H1.2 as the automated RPZ2W could conduct computer maintenance on at least two computers each day and reduce the amount of time computer technicians spend on computer maintenance by at least 10% each day for two computers. This study was not able to support H2 as the automated RPZ2W executed with an 85% accuracy. Therefore, future studies can build off of this study by creating automated maintenance scripts that have more conditionals to improve the automation script's accuracy. Ultimately, this study's automation script can be used to streamline computer maintenance.

References

Al-Bataineh, O. I., Grishina, A., & Moonen, L. (2021). Towards more reliable automated program repair by integrating static analysis techniques. *IEEE 21st International Conference on Software Quality, Reliability and Security (QRS)*, 654-663. https://doi.org/10.1109/QRS54544.2021.00075

Andersen, G. & MoldStud Research Team (2024, February 8). The importance of documentation in computer technician services. *MoldStud.* https://moldstud.com/articles/p-the-importance-of-documentation-in-computer-technician-services

Bika, N. (2022, April 25). *Computer technician job description*. Workable. https://resources.workable.com/computer-technician-job-description

Contino, N. (2024, October 31). *Raspberry Pi documentation: Raspberry Pi hardware.* https://www.raspberrypi.com/documentation/computers/raspberry-pi.html

Hanna, M., El-Haggar, N., & Sami, M. (2014). A review of scripting techniques used in automated software testing. *(IJACSA) International Journal of Advanced Computer Science and Applications, 5*(1), 194-202. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=712f5653c4f53cea5e1f62b6668452dfb86a0feb#page=209

HDMI to CSI Adapter. (n.d.). Waveshare. Retrieved October 13, 2024, from https://www.waveshare.com/wiki/HDMI_to_CSI_Adapter

Huculak, M. (2022, December 21). *20 tips and tricks to increase PC performance on Windows 10*. Windows Central. https://www.windowscentral.com/tips-tricks-increase-pc-performance-windows-10

Mearian, L. (2022, February 7). *Research shows Windows updates can take six hours to complete*. Computerworld. https://www.computerworld.com/article/1617577/research-shows-windows-updates-can-take-six-hours-to-complete.html

Pourhadi, S. (2019, May 17). *Top five challenges for IT technicians*. VScope. https://www.vscope.net/blog/challenges-for-it-technicians/

Rajivan, P., Aharonov-Majar, E., & Gonzalez, C. (2020). Update now or later? Effects of experience, cost, and risk preference on update decisions. *Journal of Cybersecurity, 6*(1), 1-12. https://doi.org/10.1093/cybsec/tyaa002

Shin, H., Rothrock, L., & Prabhu, V. (2024). Evaluating technicians' workload and performance in diagnosis for corrective maintenance. *Sensors*, *24*(6), 1943–1943. https://doi.org/10.3390/s24061943

thewh1teagle (2024). *Zero-HID: HID python library for emulating mouse and keyboard on PI*. PyPI. https://pypi.org/project/zero-hid/

Twyman, C. (2024). *Zero-HID.* GitHub Repository https://github.com/GameWire9/zero-hid

Urbitsch, M. (2023). *Windows-10-11-maintenance-script.* GitHub Repository. https://github.com/Trustiatis/Windows-10-11-maintenance-script/tree/main

Windows updates: Everything you need to know about Windows updates. (n.d.). Lenovo. Retrieved October 13, 2024, from https://www.lenovo.com/us/en/glossary/windows-updates/

Wingate, G. (Ed.). (2016). *Pharmaceutical computer systems validation* (2nd ed., pp. 203–235). CRC Press. https://doi.org/10.3109/9781420088953

Yazdi, M. (2024). Maintenance strategies and optimization techniques. *Springer Series in Reliability Engineering*, 43-58. https://doi.org/10.1007/978-3-031-53514-7_3