Journal of Robotics, Networking and Artificial Life
Vol. 11(3); March (2026), pp. 214-219
ISSN (Online): 2352-6386; ISSN (Print): 2405-9021
https://alife-robotics.org/jrnal.html

Research Article

MIXVRT: MIX Visual Regression Testing Tool Which Mixes Image Comparison and HTML Code Comparison

Naoki Aridome[1], Nobuya Takahashi[1], Tetsuro Katayama[1*], Yoshihiro Kita[2], Hisaaki Yamaba[1], Kentaro Aburada[1], and Naonobu Okazaki[1]

[1]*Department of Computer Science and Systems Engineering, Faculty of Engineering, University of Miyazaki 1-1 Gakuen-kibanadai nishi, Miyazaki, 889-2192 Japan*
[2]*Department of Information Security, Faculty of Information Systems, Siebold Campus, University of Nagasaki 1-1-1 Manabino, Nagayo-cho, Nishi-Sonogi-gun, Nagasaki, 851-2195 Japan*
Email: aridome@earth.cs.miyazaki-u.ac.jp, ntakahashi@miyazaki-u.ac.jp, kat@cs.miyazaki-u.ac.jp, kita@sun.ac.jp, yamaba@cs.miyazaki-u.ac.jp, aburada@cs.miyazaki-u.ac.jp, oka@cs.miyazaki-u.ac.jp
[*]Corresponding Author

ARTICLE INFO

Article History

Received 03 December 2024
Accepted 25 January 2026

Keywords

MIXVRT
Web pages
Layout defects
Visual regression testing
HTML code

ABSTRACT

Image-based visual regression testing has the problem that it takes time required to find layout defects. Therefore, this paper develops MIXVRT (MIX Visual Regression Testing tool), which mixes image comparison and HTML code comparison. It highlights the layout defects. From evaluation experiments, we have confirmed that the time required to find layout defects can be reduced compared to the conventional methods of image-based visual regression testing, while detecting them accurately without omissions or false detections.

1. Introduction

In recent years, web pages need to be updated more frequently from the perspectives of user experience (UX) and search engine optimization (SEO) [1]. Here, modifications made to web pages may cause not only layout differences based on changes in the HTML code but also layout defects. In this paper, layout defects refer to visual defects, such as collisions or protrusions of screen elements.

As a method for detecting layout defects, there is visual regression testing [2]. In the visual regression testing for web pages, the web pages before and after changes are placed side by side, and layout differences are highlighted to make it easier to spot layout defects. However, in studies on existing visual regression testing [3],[4], these comparisons are limited to images, and there is no mechanism to determine whether the detected layout differences are based on changes in the HTML code or are actual layout defects. As a result, developers need to check all displayed layout differences against the HTML code, and it takes time required to find layout defects.

In this paper, to reduce the time required to find layout defects in web pages, we develop MIXVRT (MIX Visual Regression Testing tool), which mixes image comparison and HTML code comparison. It highlights the layout defects. It also supports four major browsers, Google Chrome, Microsoft Edge, Mozilla Firefox, and Safari, and detects layout defects across multiple browsers.

2. Feature of MIXVRT

MIXVRT detects differences by comparing images of the web pages before and after changes and detects changes by comparing HTML code of the web pages before and after changes and then compares them to detect layout defects. In addition, it highlights the following three areas by surrounding them with red or green boxes.

- Difference Areas
  Difference Areas is the area that MIXVRT detects by comparing the images of the web pages before and after changes at the pixel level, representing the parts removed from the web page before the change and added to the web page after the change.

- Modified Areas
  Modified Areas is the area that MIXVRT detects by comparing the HTML code of the web pages before

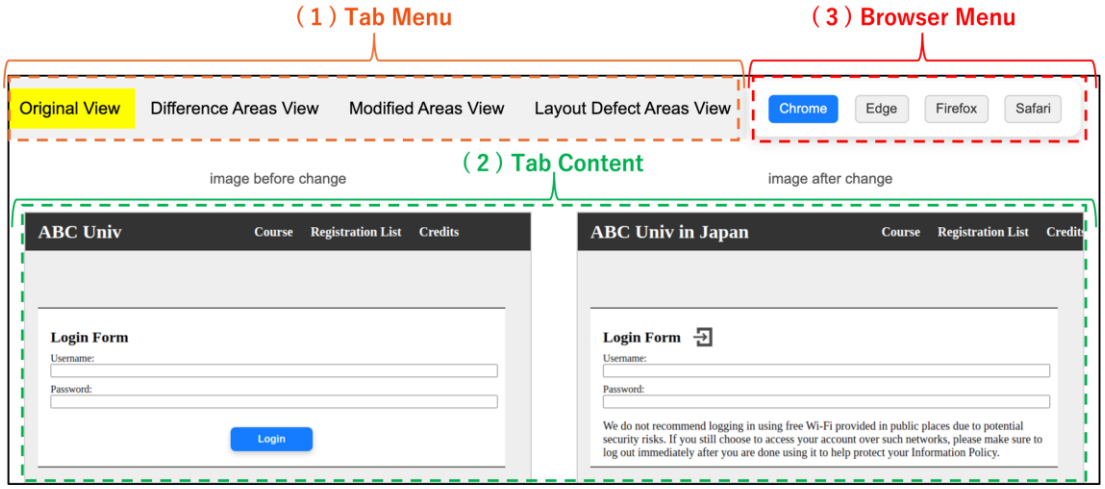

Figure 1 Appearance of MIXVRT. Four tab views (Original View, Difference Areas View, Modified Areas View, and Layout Defect Areas View) and the Browser Menu for switching between Google Chrome, Microsoft Edge, Mozilla Firefox, and Safari.

and after changes, representing the parts of screen elements affected by changes in the body or style elements. In this paper, Modified Areas is regarded as the area of layout differences based on changes in the HTML code.

- Layout Defect Areas
  Layout Defect Areas is the area that MIXVRT detects by excluding the Modified Areas from the Difference Areas, filtering out misalignments of screen elements, and detecting the remaining layout differences.

3. Appearance of MIXVRT

Figure 1 shows the appearance of MIXVRT. The appearance of MIXVRT consists of the following three components, where the numbers correspond to those shown in Figure 1.

(1) Tab Menu

MIXVRT has four tab buttons: the Original View, Difference Areas View, Modified Areas View, and Layout Defect Areas View. When a tab button is selected, the selected tab is highlighted in yellow, and the corresponding Tab Content is displayed.

(2) Tab Content

MIXVRT displays the images of the web pages before and after the change according to the tab button selected in the Tab Menu.

(3) Browser Menu

MIXVRT has four buttons: Google Chrome, Microsoft Edge, Mozilla Firefox, and Safari. When a button is selected, the selected button is highlighted in blue, and the Tab Content switches to that of the corresponding browser.

4. Structure of MIXVRT

Figure 2 shows the structure of MIXVRT. It consists of the following five processing modules.

4.1. Data Manager

The Data Manager transfers and stores data between each processing module. It also outputs the images of the web pages before and after changes to the Tab Content according to the selections in the Tab Menu and Browser Menu shown in Figure 1. In this paper, the term “data” refers to the images and HTML code of web pages before and after processing.

4.2. Data Getter

The Data Getter obtains the image and HTML code of the web page in each of the four major browsers supported by MIXVRT, using Selenium WebDriver [5] based on the URL provided as input. By performing this process for both the URL of the web page before the change and the URL of the web page after the change, the Data Getter acquires the images and HTML code of the web pages before and after changes for each browser.

4.3. Image Comparator

The Image Comparator compares the images of the web pages before and after changes at the pixel level to detect differences. Among the detected differences, it surrounds the areas removed from the image before the change with red boxes and the areas added to the image after the change with green boxes, thereby generating the images of the web pages before and after changes with the Difference Areas highlighted.

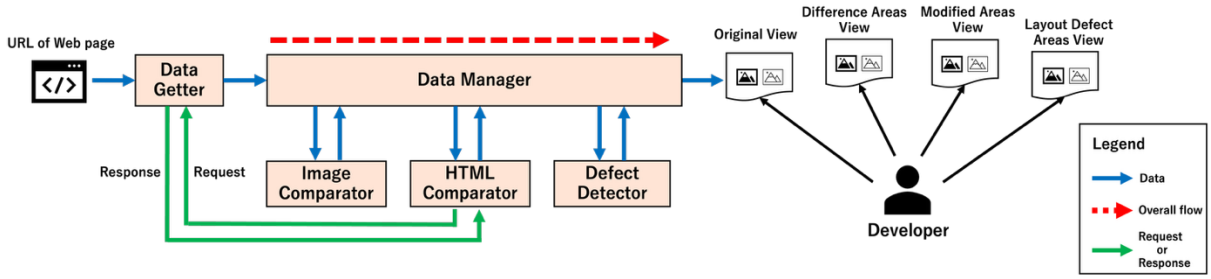

Figure 2 Structure of MIXVRT.

4.4. *HTML Comparator*

The HTML Comparator performs a line-by-line comparison between the HTML code of the web pages before and after changes and generates a diff code. Based on this diff code, it appends HTML code that surrounds the modified screen elements with red or green boxes to both the HTML code before and after changes. The two augmented HTML codes are then hosted as web pages on a local server, and their URLs are sent to the Data Getter as requests for image capture. Using these URLs, the Data Getter acquires, for each browser, the images of the web pages before and after changes with the Modified Areas highlighted and returns a total of eight images to the HTML Comparator.

This processing targets static layout changes in Web pages after browser rendering and does not handle dynamic or script-controlled screen elements. These are left for future work.

4.5. *Defect Detector*

The Defect Detector detects Layout Defect Areas by comparing a web page image highlighting the Difference Areas with a web page image highlighting the Modified Areas. It then generates the images of the web pages before and after changes with the Layout Defect Areas highlighted.

Figure 3 shows an example of detecting Layout Defect Areas using red box comparisons between the Difference Areas and the Modified Areas, and Figure 4 shows an example using green box comparisons between the Difference Areas and the Modified Areas, respectively. The Defect Detector first compares the red boxes of the Difference Areas and the Modified Areas, as shown in Figure 3, or compares the green boxes of the Difference Areas and the Modified Areas, as shown in Figure 4, to detect the Layout Defect Areas. Then, if the overlap ratio shown in Eq. (1) is less than 50%, it is determined that the difference is not a layout difference based on changes in the HTML code, and the red or green box of the Difference Areas is determined as a Layout Defect Area.

$$\frac{Area\ of\ overlap\ between\ boxes}{Area\ of\ the\ smaller\ box} \times 100 \qquad (1)$$

The threshold of 50% is empirically determined to balance omissions and false detections. Preliminary experiments with several threshold values indicate that lowering the threshold tends to increase missed layout defects, whereas raising it tends to increase false detections.

Next, a filtering process is applied to prevent misalignments of screen elements from being mistakenly detected as Layout Defect Areas. Figure 5 shows an example of filtering misalignments of screen elements. As shown in Figure 5, the red and green boxes of the Difference Areas that were determined as Layout Defect Areas are compared. If the similarity between the image regions within the red and green boxes is 85% or higher, the area is determined to be a misalignment of screen elements, and the compared red and green boxes are filtered out from the Layout Defect Areas. Note that the Structural Similarity Index Measure (SSIM) [6] is used to calculate the similarity. In addition, the similarity threshold of 85% is set to account for pixel-level changes caused by misalignments, even in screen elements whose HTML code remains unchanged.

Through the above process, the Layout Defect Areas are detected. Figure 6 shows an example of Layout Defect Areas highlighted after the filtering process.

5. Evaluation

To evaluate the effectiveness of MIXVRT, we conduct an experiment to compare MIXVRT with the conventional image-based visual regression testing tool reg-suit [7] (hereinafter referred to as the conventional method) in terms of the time required to find layout defects and the detection accuracy of layout defects. The subjects are four students majoring in computer science at the University of Miyazaki (hereinafter referred to as Subjects A-D).

In the experiment, we prepare two web pages before the change, referred to as Web page α and Web page β, respectively. Web page α consists of 261 lines of code, and Web page β consists of 477 lines of code. Each of these web pages is then intentionally modified to embed three layout defects. After modification, Web page α consists of 295 lines of code, and Web page β consists of 489 lines of code. These are referred to as Web page α and Web page β after the change, respectively.

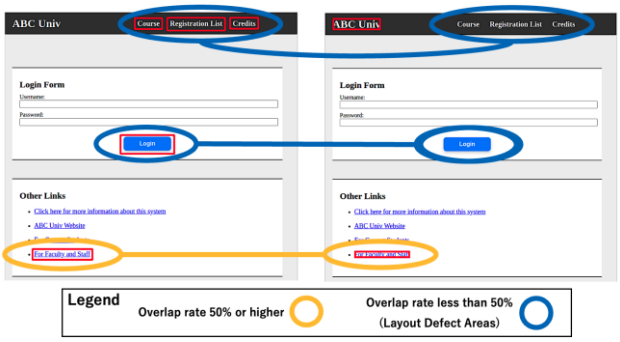

Figure 3 An example of detecting Layout Defect Areas using red box comparisons. (Left: before image with Difference Areas, Right: before image with Modified Areas.)

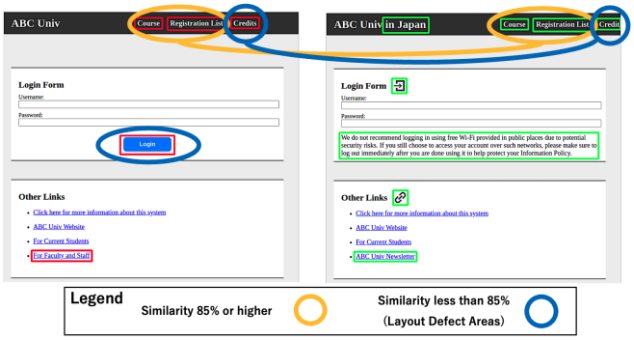

Figure 5 An example of filtering misalignments of screen elements. (Left: before image with Difference Areas, Right: after image with Difference Areas.)

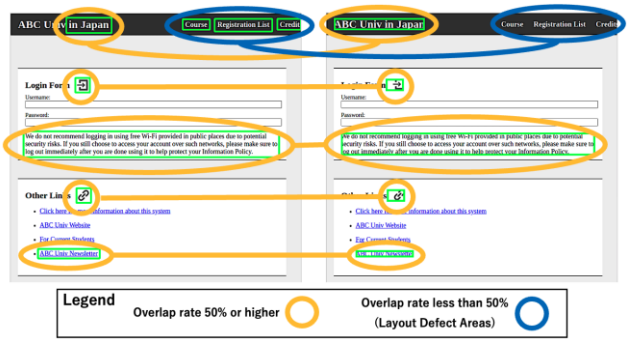

Figure 4 An example of detecting Layout Defect Areas using green box comparisons. (Left: after image with Difference Areas, Right: after image with Modified Areas.)

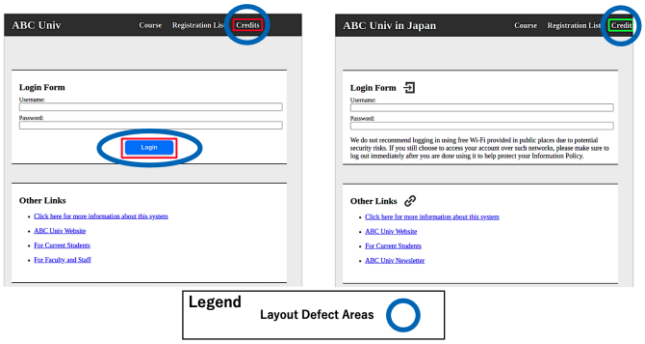

Figure 6 An example of Layout Defect Areas highlighted after the filtering process. (Left: before image with Layout Defect Areas, Right: after image with Layout Defect Areas.)

5.1. Experiment Method

The subjects conduct experiments under the following two cases using the versions of Web page α before and after changes, as well as the versions of Web page β before and after changes.

- Case A
  The time from when the subjects view Layout Defect Areas View of MIXVRT until they judge that they have found.

- Case B
  The time from when the subjects view the difference confirmation screen of the conventional tool, reg-suit, until they judge that they have found all layout defects is measured as the detection time.

In this evaluation, we evaluate the detection accuracy of layout defects by comparing the layout defects reported by each subject with the ground-truth layout defects intentionally embedded by the authors. A reported layout defect is regarded as correct only when both the affected screen element and its state match the corresponding ground truth; otherwise, it is counted as an omission or a false detection.

5.2. Evaluation of time required to find layout defects

Table 1 shows the time required to find layout defects in the two cases. As shown in Table 1, MIXVRT reduced the detection time by 16 minutes and 45 seconds (approximately 87.4%) for Web page α and by 15 minutes and 3 seconds (approximately 83.8%) for Web page β compared to the conventional method.

These results indicate that the use of MIXVRT can reduce the time required to find layout defects compared to the conventional method.

5.3. Evaluation of detection accuracy of layout defects

In Case B with the conventional method, some subjects excessively detected layout defects, while others detected fewer than the actual number. This result is due to human judgment errors in distinguishing layout defects, even though the subjects compared the displayed differences and referred to both the HTML code before and after changes to identify the affected ranges of HTML elements caused by changes in the body or style elements. In contrast, in Case A using MIXVRT, the subjects were able to detect layout defects accurately without omissions or false detections.

From these results, it can be concluded that MIXVRT is more effective for detecting layout defects than the conventional method.

6. Related Works

In studies on existing visual regression testing, layout differences are detected by performing pixel-level image comparisons for corresponding regions or screen elements [3],[4]. These existing studies propose that using image-based visual regression testing allows developers to efficiently identify layout defects. However, since these approaches rely solely on image comparison, they cannot determine whether the detected layout differences are based on changes in the HTML code or are actual layout defects. As a result, verifying the consistency between HTML code differences and layout differences requires additional time. In contrast, MIXVRT can highlight only the Layout Defect Areas by excluding the Modified Areas detected based on HTML code comparison from the Difference Areas detected based on image comparison. Therefore, compared to existing studies, MIXVRT is more effective in detecting layout defects.

As an automated tool for checking web page layouts, ReDeCheck [8] compares the DOM (Document Object Model) structures of the web pages before and after changes for multiple viewport widths, outputting a list of structural differences that may indicate unnoticed layout defects. Moreover, LAYOUT DR [9] is a tool that automatically repairs layout defects in web pages. It uses ReDeCheck to detect layout defects and presents layout repair suggestions based on the detected layout defects. In contrast, MIXVRT visually highlights Difference Areas, Modification Areas, and Layout Defect Areas by surrounding them with red or green boxes, enabling developers to instantly identify problematic regions. This improves the efficiency of verification and correction tasks after defect detection. However, MIXVRT cannot detect layout defects across multiple viewport widths, which is a limitation compared to ReDeCheck and LAYOUT DR. Addressing this limitation is one of the future challenges for further improving the practicality of MIXVRT.

7. Conclusion

In this paper, to reduce the time required to find layout defects in web pages, we develop MIXVRT which mixes image comparison and HTML code comparison. Experiments applying MIXVRT to two web pages demonstrated that MIXVRT reduced the time required to find layout defects by more than 80%. Furthermore, we have confirmed that MIXVRT detected all layout defects accurately without omissions or false detections compared to the conventional method. These results indicate that MIXVRT is effective for reducing the time find to layout defects compared to the conventional method.

The following are future issues in this paper.

- Supporting web pages where the content changes based on user actions

Table 1. The time required to find layout defects in Case A and Case B for Web pages α and β.

	Web page α		Web page β	
Subjects	**Case A**	**Case B**	**Case A**	**Case B**
A	-	18m 32s	3m 0s	-
B	-	19m 47s	2m 48s	-
C	2m 29s	-	-	17m 53s
D	2m 20s	-	-	18m 1s
Average	2m 24.5s	19m 9.5s	2m 54s	17m 57s
Diff	**16m 45s**		**15m 3s**	

- Supporting multiple viewport widths
- Supporting layout defects caused by dynamic or script-controlled screen elements
- Proving the high usefulness of MIXVRT through evaluation experiments with more diverse experimental web pages and more subjects

References

1. CAES Office of Information Technology: “Why is web content so important?”, https://oit.caes.uga.edu/why-is-web-content-so-important/. (Accessed 2025-11-01)
2. Visual regression testing: awesome-regression-testing, https://github.com/mojoaxel/awesome-regression-testing. (Accessed 2025-11-01)
3. Haruto Tanno, Yu Adachi, Yu Yoshimura, et al.: “Region-based Detection of Essential Differences in Image-based Visual Regression Testing”, Journal of Information Processing, Vol.28, pp.268-278, 2020.
4. Akihiro Tsukakoshi: “Empirical Evaluation of a Visual Regression Testing Difference Detection Method Using Hierarchical Structures of GUI Elements” (in Japanese), Software Quality Profession, Vol.5, pp.207-214, 2020.
5. Selenium: WebDriver, https://www.selenium.dev/documentation/webdriver/. (Accessed 2025-11-01)
6. MathWorks: ssim (Structural Similarity Index Measure), https://jp.mathworks.com/help/images/ref/ssim.html. (Accessed 2025-11-01)
7. REG VIZ: reg-suit, https://reg-viz.github.io/reg-suit/. (Accessed 2025-11-01)
8. Thomas A. Walsh, Gregory M. Kapfhammer, and Phil McMinn: “ReDeCheck: An Automatic Layout Failure Checking Tool for Responsively Designed Web Pages”, ISSTA 2017 Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis, pp.360-363, 2017.
9. Althomali, I., Kapfhammer, G.M., McMinn, P.: “Automated Repair of Responsive Web Page Layouts”, 2022 IEEE Conference on Software Testing, Verification and Validation (ICST), pp.140-150, 2022.

Authors Introduction

Mr. Naoki Aridome

Naoki Aridome received the Bachelor's degree in engineering (computer science and systems engineering) from the University of Miyazaki, Japan in 2024. He is currently a Master's student in Graduate School of Engineering at the University of Miyazaki, Japan. His research interests software testing and software quality.

Dr. Nobuya Takahashi

Nobuya Takahashi received B.S. and M.S. degrees in engineering from the University of Miyazaki, Japan, in 1994 and 1996, respectively, and Ph.D. degree in engineering from Kumamoto University, Japan, in 2006. He is currently an Assistant Professor with the Faculty of Engineering, University of Miyazaki, Japan. His research interests include linear control and machine learning. He is a member of SICE and JSME.

Prof. Tetsuro Katayama

Tetsuro Katayama received a Ph.D. degree in engineering from Kyushu University, Fukuoka, Japan, in 1996. From 1996 to 2000, he has been a Research Associate at the Graduate School of Information Science, Nara Institute of Science and Technology, Japan. Since 2000 he has been an Associate Professor at the Faculty of Engineering, Miyazaki University, Japan. He is currently a Professor with the Faculty of Engineering, University of Miyazaki, Japan. His research interests include software testing and quality. He is a member of the IPSJ, IEICE, and JSSST.

Prof. Yoshihiro Kita

Yoshihiro Kita received a Ph.D. degree in systems engineering from the University of Miyazaki, Japan, in 2011. He is currently an Associate Professor with the Faculty of Information Systems, University of Nagasaki, Japan. His research interests include software testing and biometrics authentication.

Prof. Hisaaki Yamaba

Hisaaki Yamaba received the B.S. and M.S. degrees in chemical engineering from the Tokyo Institute of Technology, Japan, in 1988 and 1990, respectively, and the Ph D. degree in systems engineering from the University of Miyazaki, Japan in 2011. He is currently an Associate Professor with the Faculty of Engineering, University of Miyazaki, Japan. His research interests include network security and user authentication. He is a member of SICE and SCEJ.

Prof. Kentaro Aburada

Kentaro Aburada received the B.S., M.S, and Ph.D. degrees in computer science and system engineering from the University of Miyazaki, Japan, in 2003, 2005, and 2009, respectively. He is currently a Professor with the Faculty of Engineering, University of Miyazaki, Japan. His research interests include computer networks and security. He is a member of IPSJ and IEICE.

Prof. Naonobu Okazaki

Naonobu Okazaki received his B.S, M.S., and Ph.D. degrees in electrical and communication engineering from Tohoku University, Japan, in 1986, 1988 and 1992, respectively. He joined the Information Technology Research and Development Center, Mitsubishi Electric Corporation in 1991. He is currently a Professor with the Faculty of Engineering, University of Miyazaki since 2002. His research interests include mobile network and network security. He is a member of IPSJ, IEICE and IEEE.