Comparative Evaluation of OCR Systems for Testing Game User Interface

Kyotaro Takahashi s1260200 Supervised by Prof. Maxim Mozgovoy

Abstract

In recent years, the demand for automated testing in game development has been increasing, drawing attention to vision-based GUI testing techniques[1],[2]. This study aims to evaluate the practical utility of Optical Character Recognition (OCR) technologies and support the selection of appropriate tools for test automation. Three OCR systems—EasyOCR, PaddleOCR, and Tesseract—were experimentally compared. The results show that EasyOCR and PaddleOCR demonstrated practical performance for tasks such as identifying the locations of UI elements, although their accuracy was insufficient for strict text validation. On the other hand, Tesseract exhibited high recognition accuracy for the characters it successfully detected, but its overall performance was limited by a low character detection rate. Based on these findings, this study highlights the importance of selecting an OCR system appropriate to the visual characteristics of game screens. Additionally, it emphasizes the need for preprocessing techniques and proposes future research directions such as evaluation methods with relaxed accuracy requirements and separate assessments of detection and recognition performance.

1 Introduction

In recent years, the number of game developers has increased significantly, with many projects undertaken by individuals or small teams. In such small-scale development environments, it is often difficult to allocate sufficient resources to testing tasks, making automated testing highly effective for improving productivity in game development.

However, automated UI testing in game applications faces challenges distinct from those encountered in websites or conventional desktop applications. Traditional automated testing methods typically rely on accessing source code or layout files to structurally identify and manipulate GUI elements for testing purposes. In contrast, many UI elements in games are directly rendered onto canvases or similar widgets and often do not appear in layout files[1]. Furthermore, the presence of frequently changing interactive UIs and animations makes stable identification of UI components difficult. Consequently, conventional test approaches that depend on GUI structure may not be suitable for games.

Against this background, vision-based GUI testing techniques have attracted attention. In particular, the application of OCR — which recognizes textual information rendered on the screen — offers a flexible approach that can interpret screen information without reliance on source code[3].

Recently, several high-performance OCR tools have been released, each featuring different approaches and characteristics. However, existing studies have rarely conducted comprehensive cross-comparisons and evaluations of these OCR tools.

This study aims to verify the effectiveness of OCR in the context of automated testing for game development and to clarify the strengths and weaknesses of major OCR tools through performance comparison. The aim is to provide practical guidance for selecting appropriate OCR tools in the automation of game testing.

2 Method

2.1 OCR Systems and Configuration

In this study, we compare the performance of three open-source OCR systems—EasyOCR, PaddleOCR, and Tesseract—in recognizing English text within game UI images. The evaluation focuses on multiple image categories representing varying complexities of fonts and backgrounds to analyze each OCR system's strengths and limitations.

The reason for selecting the three OCR systems is their open-source software. This makes them freely available for use even by individual developers and small development teams and allows for customization when necessary. Additionally, all systems support English text recognition, which aligns with the focus of this study.

Each OCR tool was configured to recognize English text. Specifically, EasyOCR was initialized with easyocr.Reader(['en']). For PaddleOCR, the lang parameter was set to "en", and English-specific detection and recognition models were used:

```
ocr = PaddleOCR(
   lang="en",
   det_model_dir="models/en_PP-OCRv4_det_infer",
   rec_model_dir="models/en_PP-OCRv4_rec_infer"
)
```

For Tesseract, the language was set using the following configuration:

```
custom_oem_psm_config = r'--oem 3 --psm 3 -l eng'
```

This study aims to evaluate the OCR systems' raw performance without any image preprocessing to isolate their inherent recognition capabilities. Thus, no preprocessing was applied to the input images. The OCR results were simply output as text files.
The operating environment was as follows:

OS: Windows 11 Pro
Python: 3.10.5
Tesseract: v5.5.0.20241111
pytesseract: 0.3.13
EasyOCR: 1.7.2
PaddleOCR: 2.9.1
torch: 2.6.0+cu124
torchvision: 0.21.0+cu124
torchaudio: 2.6.0+cu124
opencv-python: 4.11.0.86
numpy: 1.26.4
Pillow: 11.1.0

2.2 Data Collection and Categorization

Images were collected in PNG format with a resolution of 1920×1080 and classified into the following five categories:

Category A: Images containing text in a standard font with black letters on a white monochromatic background.

Category B: Images containing text in a standard font superimposed on game scenery backgrounds.

Category C: Images with decorative fonts (e.g., stylized characters with gloss, gradients, or three-dimensional effects) displayed on solid or near-solid backgrounds.

Category D: Images with handwritten-style fonts on solid or near-solid backgrounds.

Category E: Images with pixel fonts on solid or near-solid backgrounds.

The objective of Category A is to evaluate the fundamental performance of OCR systems under ideal conditions. Category B is intended to evaluate how game backgrounds and icons affect text recognition performance. The objectives of Categories C, D, and E are to evaluate the impact of different font styles and visual effects on OCR performance.

To prepare the images for Category A, a text sample comprising approximately 50 words (around 300 characters) was generated using ChatGPT. The generation prompt was specifically designed to minimize bias toward any particular alphabet letter.
The generated text is presented below:

jumbo vexing dwarf blitz frock nymph glaze whiz block fjord zipneck qualm spiky torch maze pluck brave ghost index jumpy whack zephyr cling vortex blimp chafe drunk glove panic hexal brisk trophy quake snout vivid ample squid zebra knack joust grimy bloat spunk hefty wizkid banjo muckle glyph zonked fjords hazel thump croup webzin glint scurf droop lavish
(58 words, 300 characters)

The text was rendered on white backgrounds to create 16 images, varying by the following conditions:

- Font: Arial or Times New Roman
- Case: All uppercase or all lowercase
- Font size: 12 pt, 18 pt, 24 pt, 48 pt

For Categories B to E, images were collected from PC games available on the Steam platform. The games were launched in full-screen mode with a resolution of 1920×1080. Screenshots of the entire screen were taken using the Windows built-in screenshot function (Windows key + PrintScreen key).

Tables 1 to 4 summarize the text datasets collected from PC games categorized as B to E. Each table includes the game title, number of images, word count, and character count.

Table 1. Overview of the image dataset for Category B

Game Title	Number of Images	Word Count	Character Count
ARK: Survival Ascended	15	16	145
Elin	8	44	208
Grounded	17	29	156
HITMAN	9	83	412
Monster Hunter Wilds	17	75	432
Shadowverse: Worlds Beyond	11	40	260

A total of 77 images, 287 words, and 1,613 characters were collected from 6 game titles in Category B.

Table 2. Overview of the image dataset for Category C

Game Title	Number of Images	Word Count	Character Count
100%おれんじじゅ〜すっ！	4	5	23
60 Parsecs!	12	38	191
Book of Demons	6	18	90
Brawlhalla	6	7	35
Castle Crashers	8	91	436
Dicey Dungeons	2	3	11
Don't Starve Together	9	69	367
Grounded	7	13	93
Hero Siege	3	17	88
Monster Hunter Wilds	4	23	144
Salt and Sanctuary	2	6	33
Shadowverse Worlds Beyond	8	29	185
Skullgirls 2nd Encore	16	35	199
Stardew Valley	6	42	229
Terraria	9	63	298
Ultimate Chicken Horse	6	45	214
Vampire Survivors	2	4	24

A total of 110 images, 508 words, and 2,660 characters were collected from 17 game titles in Category C.

Table 3. Overview of the image dataset for Category D

Game Title	Number of Images	Word Count	Character Count
OMORI	12	86	306
夜廻	8	49	213

A total of 20 images, 135 words, and 519 characters were collected from 2 game titles in Category D.

Table 4. Overview of the image dataset for Category E

Game Title	Number of Images	Word Count	Character Count
Crawl	7	56	288
Nuclear Throne	8	45	228
Vagante	6	102	589
ZERO Sievert	6	73	437
恐怖の世界	9	31	172

A total of 36 images, 307 words, and 1,714 characters were collected from 5 game titles in Category E.

2.3 Evaluation Method

To evaluate the performance of OCR systems, we calculated two metrics for each image: the word match rate and a score based on the Levenshtein distance. Primarily, the reason for including the word match rate as an evaluation metric is that, in game development test automation, accurately locating UI elements often depends on correctly recognizing specific words. In addition, the Levenshtein distance was used to measure overall character-level similarity, providing a robust metric that accounts for cases where the OCR output is longer than the ground truth, thus enabling fair comparisons across varying outputs. The score was computed using the following formula:

$$\text{Score} = 100 \times \left(1 - \frac{Levenshtein\ Distance}{Maximum\ Possible\ Distance}\right)$$

Levenshtein distance is a metric that measures the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one string into another. In this study, the Levenshtein distance was calculated between the ground truth text—representing the characters shown in the image—and the text output by the OCR system. To ensure consistency in comparison, both the ground truth text and the OCR output were formatted with a single space inserted between each unit. For the ground truth, this unit was defined as an individual word. For the OCR output, a space was inserted between each text segment recognized and output as a unit by the OCR systems, regardless of whether it consisted of one or more words. The ground truth text was arranged in Latin script reading order: top-to-bottom, then left-to-right. Similarly, the OCR

output segments were reordered based on their spatial positions to reconstruct a complete recognized text string. The maximum possible distance was defined as the length of the longer string between the ground truth and the OCR result. This formulation ensures that a score of 100 is achieved only when the OCR output fully matches the ground truth. The word match rate is defined as the number of correctly recognized words divided by the total number of target words present in the image.

For each image, multiple words were evaluated simultaneously. If two adjacent words were merged into a single word due to a missing space in the OCR output, both words were treated as incorrect and excluded from the count of matched words.

Symbols, including punctuation marks, were treated as individual words for evaluation purposes. The evaluation was case-sensitive, requiring an exact match of uppercase and lowercase letters for a word to be considered correct.

Furthermore, all test samples were included in the evaluation, including cases where OCR results were empty or significantly inaccurate, to ensure a comprehensive assessment.

In Category B, designed to evaluate the impact of game backgrounds and icons on OCR performance, characters misrecognized as text due to scenery elements were also included in the evaluation.

3 Results

3.1 Evaluation Results for Category A

The results of Category A are summarized in the following tables.

Table 5. Average word match rates per font and case

Font	Case	Easy OCR	Paddle OCR	Tesseract
Arial	Lowercase	99.14	82.33	97.84
	Uppercase	100.00	93.97	100.00
Times New Roman	Lowercase	88.79	90.09	95.69
	Uppercase	94.83	91.81	99.57

Table 6. Average scores per font and case

Font	Case	Easy OCR	Paddle OCR	Tesseract
Arial	Lowercase	99.75	85.34	99.58
	Uppercase	100.00	94.92	100.00
Times New Roman	Lowercase	96.92	93.00	99.25
	Uppercase	98.83	92.75	99.92

The tables show that both the word match rate and the score followed the order: Tesseract > EasyOCR > PaddleOCR. This outcome is largely due to two specific issues observed with PaddleOCR. First, when the font size was 12pt, PaddleOCR often failed to recognize multiple lines of text. Second, it frequently recognized an additional period at the end of a text line, even though no such character was present in the image. When these two specific failure patterns were excluded and the evaluation was limited to sporadic spelling errors, PaddleOCR outperformed EasyOCR in accuracy. Furthermore, across all three OCR systems, recognition accuracy declined as font size decreased, with a significant drop observed at 12pt. This drop was more pronounced in lowercase text, indicating a greater sensitivity to size in such cases.

3.2 Evaluation Results for Category B

The results of Category B are summarized in the following tables.

Table 7. Average word match rates per game title (Category B)

Game Title	Easy OCR	Paddle OCR	Tesseract
ARK: Survival Ascended	56.25	0.00	0.00
Elin	81.82	63.64	20.45
Grounded	93.10	96.55	27.59
HITMAN	89.16	92.77	62.65
Monster Hunter Wilds	85.33	86.67	53.33
Shadowverse: Worlds Beyond	100.00	100.00	15.00
Average of per-title match rates	84.28	73.27	29.84

Table 8. Average scores per game title (Category B)

Game Title	Easy OCR	Paddle OCR	Tesseract
ARK: Survival Ascended	90.83	5.00	2.96

Elin	95.75	77.11	32.47
Grounded	98.76	99.35	23.53
HITMAN	96.75	91.54	49.62
Monster Hunter Wilds	97.73	97.58	47.80
Shadowverse: Worlds Beyond	100.00	100.00	15.43
Average of per-title scores	96.64	78.43	28.64

Table 9. Average number of characters misrecognized from game backgrounds and icons per game title

Game Title	Easy OCR	Paddle OCR	Tesseract
ARK: Survival Ascended	0.07	0.07	2.00
Elin	0.88	0.13	25.25
Grounded	0.29	1.71	8.76
HITMAN	0.22	2.44	3.67
Monster Hunter Wilds	0.41	0.65	7.12
Shadowverse: Worlds Beyond	0.45	1.45	8.18
Average of per-title number	0.39	1.08	9.16

The tables indicate that EasyOCR outperformed PaddleOCR and Tesseract. One primary reason is that both PaddleOCR and Tesseract often failed to detect text when the contrast between the text and background was low, which resulted in lower word match rates and scores. In addition, PaddleOCR frequently misrecognized nearby icons as characters when they appeared adjacent to text, while Tesseract tended to misinterpret background elements near detected text as additional characters.

3.3 Evaluation Results for Category C

The results of Category C are summarized in the following tables.

Table 10. Average word match rates per game title (Category C)

Game Title	Easy OCR	Paddle OCR	Tesseract
100%おれんじじゅ〜すっ！	0.00	60.00	0.00
60 Parsecs!	84.21	86.84	0.00
Book of Demons	94.44	77.78	27.78
Brawlhalla	100.00	100.00	0.00
Castle Crashers	13.19	54.95	29.67
Dicey Dungeons	0.00	33.33	0.00
Don't Starve Together	73.91	97.10	66.67
Grounded	92.31	92.31	46.15
Hero Siege	35.29	5.88	0.00
Monster Hunter Wilds	78.26	78.26	56.52
Salt and Sanctuary	100.00	100.00	0.00
Shadowverse: Worlds Beyond	79.31	89.66	17.24
Skullgirls 2nd Encore	17.14	37.14	14.29
Stardew Valley	80.95	78.57	28.57
Terraria	85.71	82.54	65.08
Ultimate Chicken Horse	82.22	93.33	20.00
Vampire Survivors	50.00	25.00	0.00
Average of per-title match rates	62.76	70.16	21.88

Table 11. Average scores per game title (Category C)

Game Title	Easy OCR	Paddle OCR	Tesseract
100%おれんじじゅ〜すっ！	68.33	87.50	0.00
60 Parsecs!	98.14	98.76	0.25
Book of Demons	95.00	98.04	29.33
Brawlhalla	100.00	100.00	0.00
Castle Crashers	40.12	77.05	42.35
Dicey Dungeons	0.00	50.00	0.00

Don't Starve Together	93.48	97.61	61.07
Grounded	98.70	95.24	55.23
Hero Siege	69.60	51.16	21.52
Monster Hunter Wilds	89.21	86.49	23.95
Salt and Sanctuary	100.00	100.00	0.00
Shadowverse: Worlds Beyond	92.12	97.78	13.79
Skullgirls 2nd Encore	54.19	75.68	2.83
Stardew Valley	91.28	93.18	24.97
Terraria	90.89	91.75	77.77
Ultimate Chicken Horse	79.01	97.06	14.95
Vampire Survivors	84.38	84.38	0.00
Average of per-title scores	79.09	87.16	21.65

The results of the Category C experiment demonstrated the superiority of PaddleOCR. The performance gap between EasyOCR and PaddleOCR was primarily attributed to differences in recognition accuracy. In particular, the evaluation scores of EasyOCR and Tesseract were adversely affected by vertically stretched decorative fonts. These distortions often led to frequent misrecognitions, such as interpreting the letter f as l, or W as V. Additionally, the distinction between uppercase and lowercase letters was especially difficult in Category C, resulting in reduced performance across all OCR systems.

3.4 Evaluation Results for Category D

The results of Category D are summarized in the following tables.

Table 12. Average word match rates per game title (Category D)

Game Title	Easy OCR	Paddle OCR	Tesseract
OMORI	70.93	91.86	70.93
夜廻	59.18	87.76	40.82

Table 13. Average scores per game title (Category D)

Game Title	Easy OCR	Paddle OCR	Tesseract
OMORI	83.63	88.48	61.15
夜廻	86.28	97.08	29.01

Results from both games demonstrated the superiority of PaddleOCR. In *OMORI*, Tesseract achieved a remarkably high word detection rate compared to other games. The text in this game was concentrated at the bottom of the screen and displayed in white on a black background.

3.5 Evaluation Results for Category E

The results of Category E are summarized in the following tables.

Table 14. Average word match rates per game title (Category E)

Game Title	Easy OCR	Paddle OCR	Tesseract
Crawl	25.00	26.79	32.14
Nuclear Throne	33.33	51.11	13.33
Vagante	49.02	53.92	66.67
ZERO Sievert	75.34	94.52	69.86
恐怖の世界	80.65	93.55	54.84
Average of per-title match rates	52.67	63.98	47.37

Table 15. Average scores per game title (Category E)

Game Title	Easy OCR	Paddle OCR	Tesseract
Crawl	66.26	74.20	61.29
Nuclear Throne	59.65	78.32	20.11
Vagante	84.32	91.47	60.97
ZERO Sievert	95.60	98.72	80.67
恐怖の世界	93.25	98.48	61.11
Average of per-title scores	79.82	88.24	56.83

The experimental results demonstrated the superiority of PaddleOCR in Category E. Across all OCR systems, there was a notable increase in specific

misrecognition patterns, including substitutions such as s → 3, O → Q, f → F, a → o, v → y, and G → C.

3.6 Cross-Category Recognition Patterns

While the previous sections focused on performance within individual categories, this section examines the OCR systems' behaviors that were consistently observed across various conditions.

First, EasyOCR exhibited relatively low recognition accuracy for symbols compared to the other OCR systems. For example, exclamation marks (!) were frequently misrecognized as lowercase l, and periods (.) were often recognized as colons (:). Additionally, EasyOCR showed a greater tendency to either omit spaces or insert them incorrectly within words. However, across nearly all images, EasyOCR consistently achieved the highest overall text detection rate, not only for the target text but also for non-target characters present in the images.

Next, PaddleOCR frequently added a nonexistent period (.) after the last word when multiple words appeared in a horizontal sequence. Moreover, a recognition pattern commonly observed in Category A—where vertically aligned text led to the omission of entire lines—was also found in three images from other categories. Although the number of such cases was small, the impact of these omissions was substantial enough to warrant attention. In addition to these pattern-based tendencies, it is worth noting that although blurred text was not included as part of the target characters in this study, PaddleOCR demonstrated remarkably high recognition accuracy for such text compared to the other OCR systems.

Lastly, Tesseract recognized significantly fewer words than the other OCR engines in all categories except Category A, and it frequently failed to detect any text at all in some images. Particularly in Categories B and C, the number of words detected by Tesseract dropped significantly in images with low contrast between text and background. Furthermore, even under high-contrast conditions, several cases were observed in which Tesseract failed to detect any text in images that contained only one or two words consisting of approximately five to six characters. When focusing solely on the characters that were successfully detected, Tesseract achieved the highest character-level match rate, followed by PaddleOCR and then EasyOCR. Nevertheless, its recognition accuracy for digits was notably lower than that of the other systems. In Category E, the digit recognition rate fell below 50%.

4 Discussion

4.1 Analysis of Performance Differences and Misrecognition Patterns

First, a notable characteristic of EasyOCR was its consistently high detection performance for text embedded in game UI elements, outperforming the other two OCR systems. This superiority was particularly evident in Categories B and C, where low contrast between text and background or unusual font shapes led to decreased detection rates for the other systems, whereas EasyOCR maintained high detection accuracy under these challenging conditions. Furthermore, in nearly all of the images used in this study, EasyOCR achieved the highest overall text detection rate, including non-target characters. These results suggest that EasyOCR exhibits strong robustness in text detection across game UIs with diverse design elements.

Next, PaddleOCR also demonstrated high-level performance in both text detection and recognition accuracy within game UIs, achieving the highest overall evaluation scores in Categories C through E. Even in Category B, where both PaddleOCR and Tesseract scored nearly zero for the game title *ARK: Survival Ascended*, the performance of PaddleOCR was comparable to that of EasyOCR when that outlier was excluded. Additionally, the impact of font shape variation on PaddleOCR's performance was the smallest among the three systems, further suggesting its robustness in handling design variability.

Finally, regarding Tesseract, while it achieved the highest accuracy rate for detected characters, it frequently failed to detect the target characters in the first place. As a result, its overall performance across Categories B to E was significantly lower, reaching only about 30% of the maximum possible score. The primary cause of these detection failures appeared to be low text-background contrast. This trend was especially pronounced in Category C, where white text was successfully detected, while colored text was often missed. Another possible contributing factor was character density: even with identical font, color, and background, characters in dense clusters were more likely to be detected than isolated ones. However, since the experimental results for density effects were less conclusive than those for contrast, further investigation is needed.

4.2 Practical Implications and Mitigation Strategies

First, both EasyOCR and PaddleOCR demonstrated practical potential depending on the role OCR plays in game testing. This study adopted a strict evaluation method, including case sensitivity, punctuation, and spaces. Despite these rigorous criteria, both OCR systems achieved average scores ranging from 80 to 90% across all categories. These results suggest that when the target UI text is in English, these OCR systems can be effectively used to automatically locate UI elements during game testing.

If the role of OCR in testing is to enable UI interaction, detecting the closest matching characters is sufficient to fulfill this purpose. However, if the OCR's function is to verify the correct display of text or to detect typographical errors, the current accuracy levels are insufficient for practical use.

To bridge this gap, it is necessary to utilize the distinct characteristics of EasyOCR and PaddleOCR appropriately. EasyOCR is recommended for scenarios with low contrast between text and background or when the text size is small. In other cases, PaddleOCR is preferable. Additionally, applying preprocessing techniques such as binarization or region extraction on game screen images during testing may further improve accuracy and should be explored.

Regarding Tesseract, the results of this study showed generally low evaluation scores, indicating that it is currently inadequate for practical use. Nevertheless, given its high recognition accuracy for detected characters, Tesseract has the potential to be the optimal OCR system among the three if the word detection rate can be improved. Since Tesseract is highly sensitive to text-background contrast, binarization of game screen images is likely to be effective. Moreover, due to its misrecognition tendencies, Tesseract is less suitable for scenes where text appears scattered in various directions. Therefore, restricting its application to scenes like visual novels, in which text is displayed exclusively at the bottom of the screen with a uniform-colored background, is crucial for enhancing word detection accuracy.

4.3 Limitations and Future Work

The experimental results highlighted a noticeable gap between the current OCR performance and practical requirements. To bridge this gap, it became clear that applying preprocessing techniques to game screen images is necessary. Consequently, conducting performance comparisons after applying such preprocessing will be essential in future studies.

In this study, case sensitivity, punctuation, and spaces were included in the evaluation metrics to maintain strict accuracy requirements. However, excluding distinctions between uppercase and lowercase letters, punctuation marks, and spaces could provide additional insight into OCR performance, especially in applications where such strictness is unnecessary. Therefore, introducing evaluation metrics that omit these factors is a potential future improvement.

Moreover, it is important to separately evaluate text detection rates and recognition accuracy. In particular, Tesseract showed high recognition accuracy only on the detected text, but its overall effectiveness is limited by a low detection rate. Distinguishing these two aspects in evaluation will enable a clearer understanding of each OCR system's strengths and weaknesses, and better guide their application in practical scenarios.

5 Conclusions

This study aimed to provide practical guidelines for selecting OCR tools in the context of automated testing for game development by conducting a comparative evaluation of text recognition accuracy on game screens using EasyOCR, PaddleOCR, and Tesseract. The results showed that EasyOCR and PaddleOCR demonstrated practical performance for the role of locating UI elements. Based on the overall performance, PaddleOCR emerges as the primary candidate for general use. However, in scenarios where the contrast between text and background is low or where the character size is small, EasyOCR tends to provide more stable recognition results. Therefore, EasyOCR is recommended under such conditions.

To obtain further insights for OCR tool selection, future work should explore image preprocessing techniques and conduct performance comparisons that incorporate such preprocessing. Additionally, it is considered effective to introduce evaluation metrics that exclude distinctions between uppercase and lowercase letters, punctuation, and spaces, as well as to separately assess text detection rates and recognition accuracy. These approaches are expected to provide more practical insights for applying OCR effectively.

Acknowledgement

I am grateful to Professor Mozgovoy for his invaluable guidance and support.

References

[1] S. Yu, C. Fang, Z. Tuo, Q. Zhang, C. Chen, Z. Chen, and Z. Su, "Vision-based mobile app GUI testing: A survey," arXiv preprint arXiv:2310.13518, Oct. 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2310.13518

[2] X. Liang, J. Qi, Y. Gao, C. Peng, and P. Yang, "Automated Game GUI Text Glitch Detection Based on Computer Vision," in *Proc. of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*, San Francisco, CA, USA, 2023, pp. Article No. 311, 1–13. doi: 10.1145/3611643.3613867.

[3] M. Mozgovoy and E. Pyshkin, "Pragmatic Approach to Automated Testing of Mobile Applications with Non-Native Graphic User Interface," *International Journal on Advances in Software*, vol. 11, no. 3–4, pp. 239–246, Dec. 2018.