Automated Visual Verification of Avionics Cockpit Displays

Muhammed Onur Güngör*†, Gökhan İnce†
*_MGEO Test and Verification Directorate_, _ASELSAN Inc._, Ankara, Turkey
mogungor@aselsan.com.tr
†_Faculty of Computer and Informatics Engineering_, _Istanbul Technical University_, İstanbul, Turkey
{mogungor, gokhan.ince}@itu.edu.tr

***Abstract*—The purpose of the study is to design a system to automate the visual verification of avionics cockpit displays using digital cameras. The proposed system captures images from avionics cockpit display systems, and registers to the model of the cockpit display system, while being compliant with international standards, such as DO-178C. In this system, we included three visual verification tasks which are text verification, symbology verification, and color verification. We proposed to use Region Of Interest (ROI) generator, ground truth generator, camera calibration tool, image capture service, image registration service, comparison services, and test execution tool. As a consequence, visual verification test cases have been automated and executed without human intervention. We measured the accuracy of the system using F1 score while detecting text, colors, and objects. We also verified the effectiveness of the proposed system using System Usability Scale (SUS). Experiments show that this system is an effective method for using in automated visual verification.**

***Index Terms*—Automated Test, Avionics HMI Test, Cockpit Display Systems Test**

I. Introduction

Failure of the avionics systems can be vital for the safe operation of aircraft, therefore avionics systems are designed with strict rules. One of the main regulations is European Aviation Safety Agency (EASA) Certification Specification (CS) CS-25. Various standards have been developed to comply with these regulations, such as DO-178C [1] and DO-254 [2]. Testing activities must be performed to comply with these standards. Some of the testing activities constitute the verification of the values in the cockpit display systems. Visual verification test cases are usually performed by manually comparing the values on the cockpit screen. For this reason, it is not only desirable to automate the test cases to eliminate human error, but also desirable to minimize the testing cost [3].

Nguyen et al. group testing frameworks in 4 different categories according to the test case development technique. They are script-based method, capture and replay, random-walk, and automated model-based method [4]. Dyachenko et al. state that it is possible to implement Human-Machine Interface (HMI) verification using computer processing methods, especially image and sound recognition algorithms. Their proposed architecture sets up the test environment and specifies the expected result. Test cases are executed to generate expected values on target screen. Actual values are recorded with a camera and microphone. The software takes the streams and compares them with expected values and generates a test result report [5].

Sartaj et al. provide a model-based technique for automating Cockpit Display Systems (CDS) testing. They proposed a Unified Model Language (UML) profile that is compatible with modern CDS design tools. The offered model is used to generate test cases automatically. They use state-of-the-art automated visual inspection techniques for evaluation on a simulator [6]. Sartaj et al. provide a tool for generating, executing, and evaluating test cases [7]. Tom et al. developed an automation system using GUI screenshots. They developed an editor to write visual scripts. User can select any image for a condition and also user can define an action for this condition [8].

In our study, a hybrid method, which is a mixture of script-based and capture and replay, is proposed. Test cases are not generated automatically as in automated model-based, but suggestions have been made for rapid development of test cases. Automated visual verification is referred in the Dyachenko et al. study, but details about the design are not given. There is no explanation and experimental results about finding ROI, generating expected values, and evaluating them. Unlike the Sartaj et al. approach, the outputs of the CDS design tools are not used to generate test cases automatically in this work. Instead, they are used to generate ground truth images for comparison within the scope of this study. The method we propose enables requirement-based testing instead of model-based testing. Sartaj et al. do not offer a solution for taking the image from target instead of a simulator. In our proposed system, taking the image from the target is discussed in detail.

The image on the CDS was captured with a digital camera and registered according to the given model of the CDS. The ROI on the image is determined after the screenshot is captured. These determined regions store information about the absolute position of the expected value in the coordinate plane. Test cases are developed using this information. There are four types of verification, which are text verification, foreground and background color verification, font size verification, and symbology verification. Optical Character Recognition (OCR) procedures are used to read the text in an ROI for text verification. The challenge is that the test developer should be able to supply the absolute position of the relevant word(s) for each test case in a simple and fast manner. Color classification is used for color verification. The difficulty is that due to anti-aliasing, the color of pixels on the edges of characters and objects will be lighter. Color calibration should also be done if the image was captured with a digital camera. Template matching is used for symbology verification. The generation of ground truth symbols is needed for comparison. The main challenge for symbology verification is producing the ground truth symbologies quickly and easily to use in a test case.

Taking into account all of the issues mentioned above, a systematic research should be conducted to accomplish automated visual verification in avionics systems. Architecture and used methods are detailed in Section II. Experimental results are given in Section III. Final words are given in Section IV.

II. Automated Visual Verification System

In this chapter information about proposed system for automated visual verification will be given. An automated visual verification architecture for pages of cockpit display systems is suggested.

A. Architecture

The automated visual verification system environment is shown in Fig. 1.

1) **PC:** PC is used for test execution, ground truth image generation, and ROI definition.
2) **Test Execution Tool:** It runs the test cases and controls the interface cards to create interface messages based on the demands in the test cases.
3) **Ground Truth Image Generator:** It generates ground truth images on a simulator or target hardware for each ROI.
4) **Ground Truth Image Database:** It stores the ground truth images to use in test cases as expected image.
5) **ROI Manager:** It is used to determine ROIs on the cockpit display screens.
6) **ROI Database:** It stores ROIs to use in test cases for defining the exact position of the expected image or text in the coordinate plane.
7) **Web Browser:** A simple web browser for initializing image registration service.
8) **Interface Cards:** Test execution tool uses interface cards to send necessary messages to the cockpit display system to drive screens. These interfaces can be MIL-STD-1553, Serial Port, CAN Bus, Ethernet, ARINC-429 etc.
9) **Comparator:** It processes the comparison request from the test execution tool (2). An image is requested from the image registration server and sent to the submodules according to the comparison type.
    a) **Text Comparison:** Optical Character Recognition (OCR) is used to process the raw image. The expected value is compared with the OCR text output.
    b) **Color Comparison:** The foreground or background color of the ROI in the captured image is compared with the expected color.
    c) **Object Comparison:** The captured symbol in the ROI is compared with the ground truth symbol.
10) **Image Registration:** The comparator accepts the images which have the same perspective to be able to compare them. The image registration module transforms the perspective of the captured images to the perspective of the ground truth image. It not only transforms perspective but also crops out of the screen. Image Registration uses calibration data to register the captured image.
11) **Calibration Tool:** The image registration data, also known as homography, is generated using the calibration tool. Homography is produced by analyzing at least three points on the reference and sensed images. These points and the matching of these points are prompted by the user.
12) **Screen Capture API:** It provides an application program interface to capture the image of the cockpit display system using a digital camera.
13) **Camera:** It is a Digital Single Lens Reflex (DSLR) camera whose AV, TV, and ISO values can be adjusted using its API.
14) **Cockpit Display System:** It is the target device which has the flight screens to be verified.

B. Screen Capturing

There are two ways to get screenshot of the cockpit display system. The first one is taking the screenshots directly through the screen buffer of the graphics card. Exporting this buffer at the software level is a destructive method while considering a real-time system. On the other hand, exporting this buffer at the hardware level is not time and cost efficient while considering different hardware architectures. It is possible to take a screenshot of the cockpit display system using a digital camera. An image equivalent to the image taken from the screen buffer of the graphic card can be captured by calibrating the position and color of the digital camera. Image registration and consequent image processing techniques allow getting the image from the same perspective. The idea of using image registration, which is a technique for matching two distinct images in image processing, as position calibration of the camera is one of the main contributions of this study.

One of the images in image registration is considered as a reference image, and the other is a sensed image that is used to register to the reference image [9]. Image registration performs 2-dimensional transformation on the sensed image with respect to reference image. This transformation operates on homogeneous coordinates and it is calculated using (1).

$$\tilde{x}_0 = \tilde{H}\ \tilde{x}, \tag{1}$$

where $\tilde{H}$ is an arbitrary 3×3 matrix used to calculate $\tilde{x}_0$ which is transformed image matrix and $\tilde{x}$ is the original image matrix.

In this study, it is aimed to find $\tilde{H}$ matrix. In order to calculate this matrix, the common features on the ref-

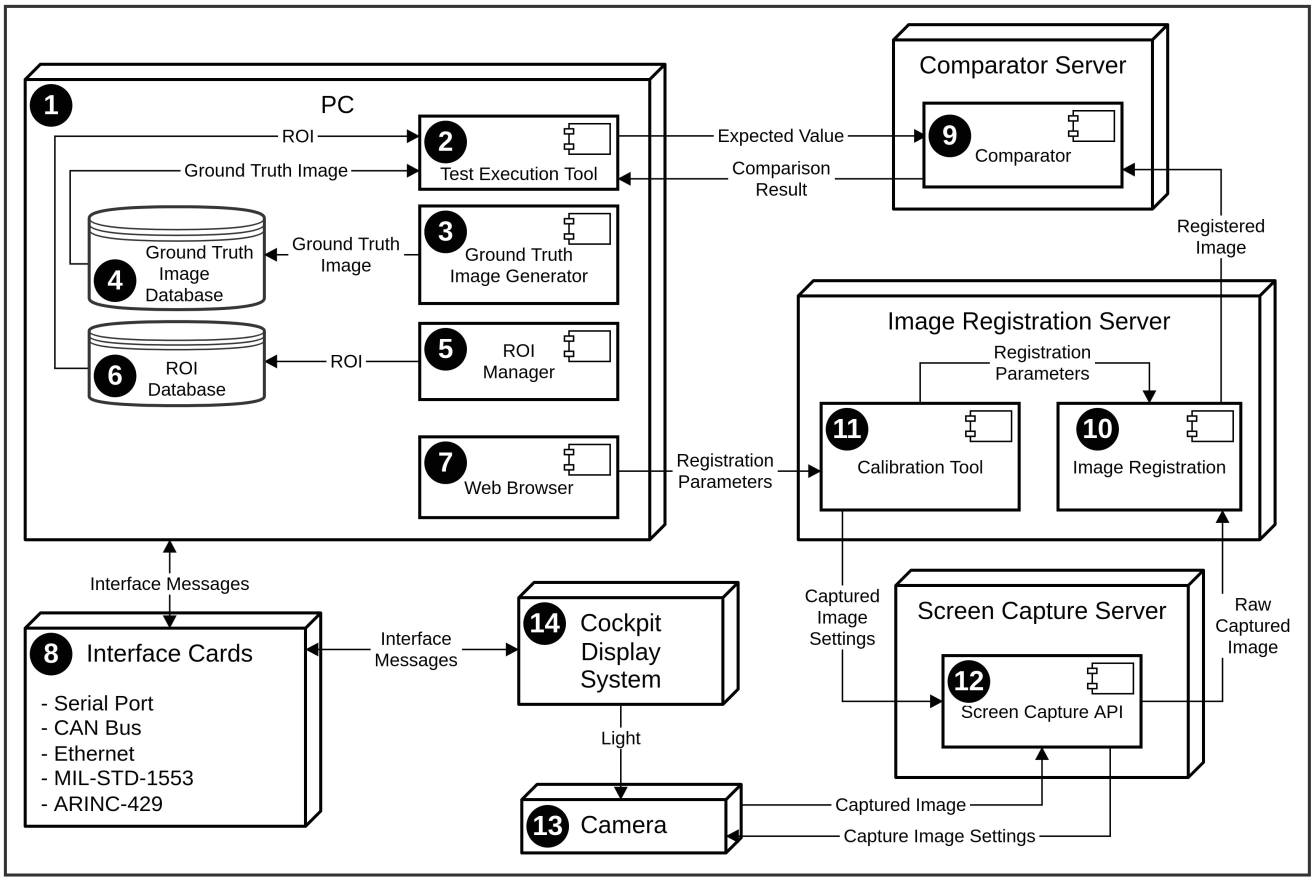

Fig. 1. Automated Visual Verification System

erence image (CAD models of the cockpit display system) and sensed image must be found and matched. Common features can be found automatically using feature detection algorithms [10], [11], [12]. Since these methods, which automatically find features, do not work in a deterministic way, manual determination of these points will be preferred while considering safety-critical systems.

C. Test Case Development and Execution

The data on the cockpit display system should be partitioned since verification activity involves a specific partition on the screen. Each of these partitions is called the ROI. It can be generated manually or automatically. The exported data from graphical modeling tools such as Presagis VAPS XT or Ansys SCADE can be used to create automatically. To create the ROIs manually, the existing pages in the cockpit display system are partitioned according to the software requirements using an ROI manager tool. Manual partitioning provides a more flexible structure and will eliminate the dependency on graphics modeling tools. The positions of the ROI are determined using at least three points. The verification type for an ROI, such as background color verification, text, font verification, etc. should be provided. A previously created ground truth is also provided if an object is to be verified.

Ground truth image is used as an expected image while comparing objects in the test cases. Generating ground truth images is a time-consuming and difficult task to run the entire system, giving all the required inputs and obtaining the expected value as a ground truth image. Instead of utilizing the Operational Flight Program (OFP), a cockpit display design template developed with VAPS XT or SCADE may easily be modified to obtain data using injected codes. In this way, ground truth values can be produced quickly.

ROI and expected value (a text, a color or ground truth image) are sent to the comparator module. The expected value and the actual value are compared by the comparator module.

D. Comparison Methods and Verification

Comparison is a complex task for the machines to automatically perform. It is a simple process for humans, but it is slow and error-prone. There are three types of comparisons which are text comparison, object (image) comparison, and color comparison. Flow diagram for comparison is shown in Fig. 2.

Text comparison is made by reading the ROI field using OCR techniques. ROI given in the test case is cropped from

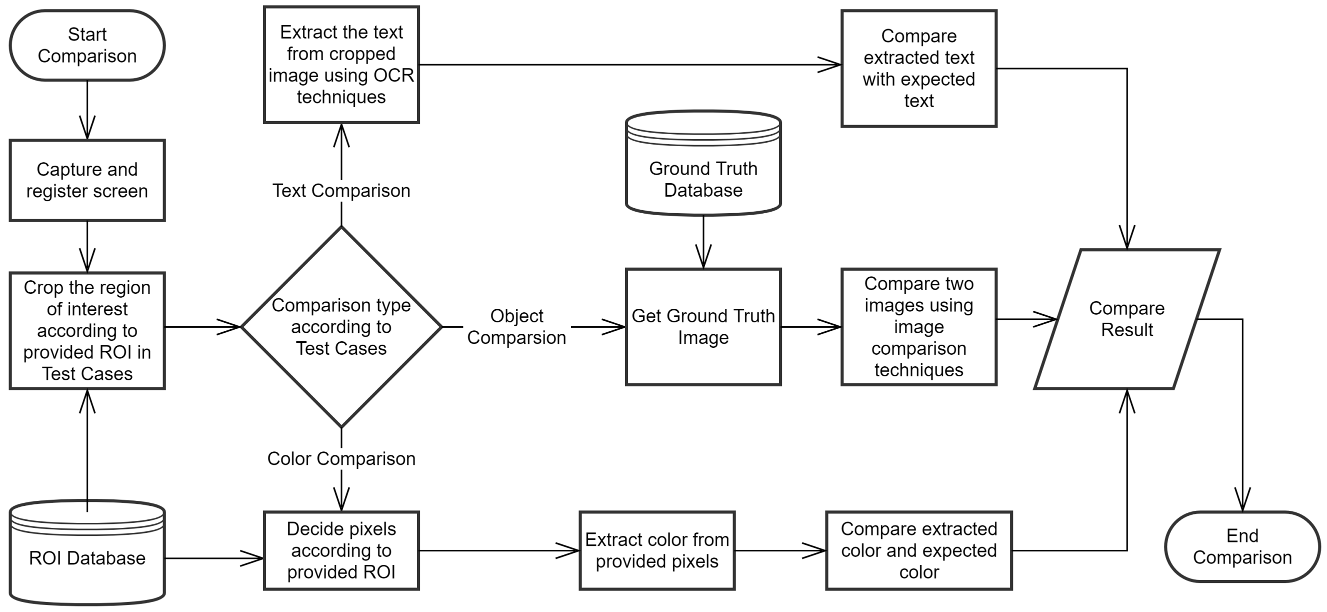
Start Comparison
Capture and register screen
Crop the region of interest according to provided ROI in Test Cases
ROI Database
Comparison type according to Test Cases
Text Comparison
Object Comparsion
Color Comparison
Extract the text from cropped image using OCR techniques
Compare extracted text with expected text
Ground Truth Database
Get Ground Truth Image
Compare two images using image comparison techniques
Compare Result
Decide pixels according to provided ROI
Extract color from provided pixels
Compare extracted color and expected color
End Comparison

Fig. 2. Comparison Flow Diagram

the registered screenshot. This cropped image is given to the OCR framework. The text output from the OCR framework is compared with the expected value and the result is returned. State-of-the-art OCR techniques are used to extract text from an image to compare with expected text. Tesseract [13], which is also supported by Google, offers an advanced framework. Some parameters can be provided to show improved performance. These parameters are the number of lines, language, white list characters, etc.

Object in the ROI field is compared with ground truth image by using several image comparison techniques. The ROI given in the test case is first cropped from the registered screenshot. This cropped image and ground truth image are given to the image comparison framework. The output from the image comparison framework is returned. Template matching algorithm is used to compare symbologies.

Color comparison is needed when verifying the background or foreground color of the text or color of the object in the ROI. The ROI given in the test case is first cropped from the registered screenshot. The foreground and background pixels of a text must be identified if their colors are to be verified. The pixels of the object must be identified if the color of an object is to be verified. These pixels are given to the color comparison framework. The output from the color comparison framework is returned. Support Vector Machine (SVM) is used for color classification. There are 16 colors to identify in the scope of this study. The fixed-size rectangles formed with these colors are positioned randomly on the screen to train the SVM model. These rectangles are partitioned and labeled. In this way, 786432 pixels and the label of these pixels are determined. Model accuracy on test set is found to be 1.0.

III. Experiments and Results

In this chapter information about the experiments and their results will be given. Experiments were carried out for assessing the performance and usability of the system. How successfully text recognition, color recognition and object comparison can be made has been demonstrated by $F1$ scores with the performance tests. Effectiveness and satisfaction of the designed system are measured with System Usability Scale (SUS) [14].

A. Experiments

In order to measure the performance of the proposed system, two different test datasets are prepared using Primary Flight Display (PFD) page of the Cockpit Display System by simulating flight. Dataset 1 (DS1) is prepared for verification of text, text size, and foreground and background color of the text and Dataset 2 (DS2) is prepared for symbology verification. DS1 consists of 208 ROIs which have text with different foreground and background colors and text sizes. DS2 is a data set consisting of 30 images belonging to 6 different objects in CDS. It consists of 5 images of each object taken with different AV, TV, and ISO configuration. The configuration list is given Table I.

TABLE I
Image Configuration

Configuration	AV	TV	ISO
Config1	3.5	30	100
Config2	3.5	30	200
Config3	3.5	50	200
Config4	4.0	25	100
Config5	4.5	25	100

B. Results on the Recognition Performance

The texts of 200 out of 208 ROIs were correctly recognized and the text sizes of 190 out of 208 ROIs were correctly recognized as a result of the experiment using DS1. The table of confusion obtained for the foreground and background color recognition as a result of the experiment using DS1 is given in Table II and Table III. Foreground color recognition accuracy is 0.40865. Although the color recognition accuracy of the SVM model is 1.0, it does not have sufficient $F1$ score because of anti-aliasing algorithm on the text. Studies in this area are still ongoing. Background color recognition accuracy is 0.81731. The results are better than foreground color recognition since the background color has no anti-aliasing effect. $F1$ score of all colors could not be calculated because not all colors are used in the data set, they are shown in the table as N/A.

TABLE II
FOREGROUND COLOR RECOGNITION

Color	TP	TN	FP	FN	F_1
Amber	0	177	0	31	0
Black	0	199	2	9	0
Brown	0	170	38	0	0
Cloud	0	146	62	0	0
Cyan	0	206	0	2	0
Gray	0	177	2	31	0
Green	0	193	0	15	0
Light Blue	0	207	1	0	0
Magenta	0	207	0	1	0
Red	2	169	4	33	0.10
White	83	106	18	1	0.90

TABLE III
BACKGROUND COLOR RECOGNITION

Color	TP	TN	FP	FN	F_1
Amber	5	208	0	0	1
Black	45	126	7	30	0.71
Dark Gray	116	54	30	8	0.80
Red	2	206	0	0	1
White	2	206	0	0	1

The normalized cross-correlation for the object recognition as a result of the experiment using DS2 is given in Table IV. C_n stands for calculated normalized cross-correlation between ground truth image and captured image for image configuration Config$_n$ in the table.

TABLE IV
OBJECT RECOGNITION

Object	**C$_1$**	**C$_2$**	**C$_3$**	**C$_4$**	**C$_5$**
Plane	0.86	0.86	0.93	0.84	0.84
Arrow	0.94	0.93	0.93	0.94	0.94
Stop	0.98	0.98	0.98	0.98	0.98
Compass	0.96	0.95	0.96	0.96	0.96
Plane	0.96	0.96	0.96	0.96	0.96
Target	0.97	0.92	0.96	0.97	0.96

C. Results on the Usability Performance

The usability of the system was measured with the SUS questions directed to 13 testers at different experience levels. The average age of the participants was 29.69±3.62. 46.15 percent of the participants were female and 53.85 percent were male. Participants were comprised of those with bachelor's degree in Computer Engineering, bachelor's degree in Electrical and Electronics, and bachelor's in Statistics, and their percentages were 38.46, 46.15, and 15.39, respectively. The average work experience of the participants was 6.77±4.04 years. The average test experience of the participants was 5.85±3.67 years. All participants used at least one test tool for one year in their working life.

First of all, a 1-hour training on how the system and tools work was given to the participants. The participants were given 3 different types of tasks to perform using all the test tools in the system. These tasks are; 1) Camera Calibration, 2) ROI Identification and Ground Truth Generation, and 3) Developing and Executing a Test Case with Defined ROIs and Ground Truths. After each task, 3 questions were asked about the difficulty of the task, the adequacy of time and the adequacy of the technical support they received. The averages of the answers for each task are given in Fig. 3.

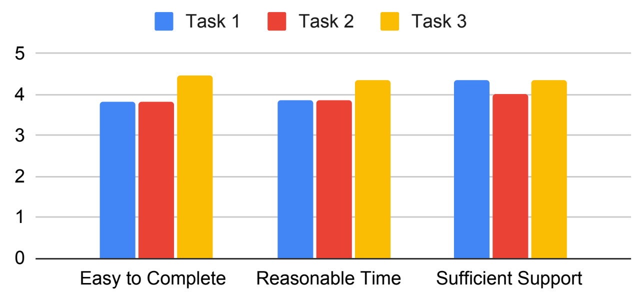

Fig. 3. The Average of the Answers for Tasks

At the end of these tasks, the SUS questionnaire was applied to the participants. The SUS score of our proposed system is 71.92. A SUS score of 70 and above is considered acceptable for the usability of a designed system [14]. Participants thought that it is easy to develop tests with the proposed system but they thought that the preparations made before test case development are more difficult.

IV. CONCLUSION

It is aimed to perform automated visual verification for cockpit display systems on images captured with a professional camera within the scope of this study. The captured images have been transformed to the same perspective as the cockpit screen using the image registration technique. In this way, it is ensured that the position of each ROI is fixed regardless of the perspective of the captured image. The cockpit screen has been successfully partitioned into regions and labeled with the developed ROI Manager tool. ROIs, generated using ROI Manager, were used in the Test

Case Development tool for defining the ROI. Ground Truth Generator tool is used to generate ground truth image for an ROI that needs symbology verification. Text and color values are given directly through the Test Case Development Tool when text or color verification will be performed. State-of-the-art OCR techniques, Tesseract, are used for text recognition. The text verification was performed flawlessly. A simple SVM model was created for color recognition by training 786432 pixels produced within the scope of this study. Template matching was used for object recognition.

In the future, color recognition, especially text foreground color recognition, should be improved with various image processing techniques. The SUS score can be increased by developing more user-friendly methods for camera calibration and ROI creation.

REFERENCES

[1] *RTCA DO-178C, Software Considerations in Airborne Systems and Equipment Certification*. RTCA Incorporated, 2011.

[2] *RTCA DO-254 Design Assurance Guidance for Airborne Electronic Hardware*. RTCA Incorporated, 2000.

[3] E. Dustin, T. Garrett, and B. Gauf, *Implementing automated software testing: How to save time and lower costs while raising quality*. Pearson Education, 2009.

[4] B. N. Nguyen, B. Robbins, I. Banerjee, and A. Memon, “Guitar: an innovative tool for automated testing of gui-driven software,” *Automated software engineering*, vol. 21, no. 1, pp. 65–105, 2014.

[5] S. Dyachenko, D. Ilyashenko, and E. Neretin, “Overview of automation tools for avionics verification,” in *Journal of Physics: Conference Series*, vol. 1958, no. 1. IOP Publishing, 2021, p. 012012.

[6] H. Sartaj, M. Z. Iqbal, and M. U. Khan, “Testing cockpit display systems of aircraft using a model-based approach,” *Software and Systems Modeling*, pp. 1–26, 2021.

[7] ——, “Cdst: A toolkit for testing cockpit display systems of avionics,” *arXiv preprint arXiv:2001.07869*, 2020.

[8] T. Yeh, T.-H. Chang, and R. C. Miller, “Sikuli: using gui screenshots for search and automation,” in *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, 2009, pp. 183–192.

[9] S. S. Bisht, B. Gupta, and P. Rahi, “Image registration concept and techniques: a review,” *J Eng Res App*, vol. 4, pp. 30–5, 2014.

[10] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2. Ieee, 1999, pp. 1150–1157.

[11] H. Bay, T. Tuytelaars, and L. Van Gool, “Surf: Speeded up robust features,” in *European conference on computer vision*. Springer, 2006, pp. 404–417.

[12] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf,” in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.

[13] R. W. Smith, “An overview of the tesseract ocr engine,” in *Ninth international conference on document analysis and recognition (ICDAR 2007)*, vol. 2. IEEE, 2007, pp. 629–633.

[14] J. Brooke, “Sus: a retrospective,” *Journal of usability studies*, vol. 8, no. 2, pp. 29–40, 2013.