TED University

Department of Computer Engineering

Senior Design Project

Test Plan Report

Group Members:

- Umay ŞAMLI [52255193822]
- Rümeysa ÖZ [10216424224]
- İhsan Melih ŞİŞMAN [56725508390]
- Fatih Mehmet YİĞİTEL [38651154902]

Supervisor:

- Yücel ÇİMTAY

Jury Members:

- Gökçe Nur YILMAZ
- Saiful ISLAM

Test Plan Report
May 16, 2023

Test Plan Report document graduation project course is offered under CMPE492.

Table of Contents

Introduction

Today, low visibility conditions such as fog and haze pose a major challenge for computer vision systems' accurate and reliable operation. Image enhancement from a single foggy image is a complex and ill-defined problem. This test plan describes testing activities for a computer vision system operating in fog and haze conditions, which we propose with an innovative deep learning-based technique called DehazeNet. DehazeNet uses a deep architecture based on Convolutional Neural Networks (CNN) or Capsule Networks (CapsNet) with specially designed layers to predict the media transmission map and then used to obtain a fog-free image via the atmospheric scattering model.

This test plan describes the test activities of a computer vision system operating in fog and haze conditions. The main objective is to evaluate the performance of the system, which includes object detection, shape recognition, color recognition, and response times at various visibility levels. The plan describes the scope, approach, resources, and timing of testing activities, as well as the features to be tested and the personnel responsible for each task.

To evaluate the effectiveness of DehazeNet, we will analyze the performance of the system under various scene types and lighting conditions at specific fog and haze densities. This will provide insight into the innovative features and components of DehazeNet that allow it to achieve better results compared to existing methods.

This test plan is a fundamental step to ensure the success of the DehazeNet project and further develop the system. We believe we can achieve our goal of developing a solution that offers advanced automation and functionality by significantly improving the performance and reliability of computer vision systems operating in fog and haze conditions. Through this test plan, we can set a new standard for computer vision systems operating in low visibility conditions, laying a solid foundation for future development and optimization of DehazeNet.

1. Scope

This test plan covers the following aspects of the DehazeNet computer vision system operating in fog and haze conditions.

Detection Accuracy in Fog and Haze Conditions: The tests will evaluate the accuracy and reliability of objects detected by the system. This will help to understand DehazeNet's overall detection success with testing under different fog and haze densities.

Shape Recognition at Different Levels of Visibility: By examining Dehaze Net’s ability to recognize shapes in low visibility conditions, we will determine how successfully it can distinguish the shapes of objects of varying complexity and scale.

Color Recognition at Varying Fog and Haze Intensities: The tests will focus on evaluating Dehaze Net’s performance in recognizing and classifying colored objects with accurate colors. This will be ensured by tests on different colors and hues to measure the color accuracy of the system and how well it preserves colors.

Test Environment Setup and Simulation: As part of the test plan, proper simulation and environment preparation are essential to create realistic and controlled test environments under different fog and haze conditions. This provides valuable insight into how tests produce reliable and repeatable results and how DehazeNet will perform in real-world applications.

Testing under this scope will allow us to evaluate how well DehazeNet meets the critical requirements of computer vision systems in low visibility conditions. In addition, the test results will be used to identify the progress of the project and potential areas for future improvements. This test plan will help us achieve our goal of significantly improving the performance and accuracy of systems operating in fog and haze conditions, contributing to the positioning of DehazeNet as a reliable and effective computer vision system.

2. Approach

The test approach consists of two techniques: proactive and reactive. The proactive approach involves initiating the test design process at the earliest possible stage to identify and address defects before creating the software build.

On the other hand, the reactive approach involves starting testing activities only after the design and coding phases have been completed. In Project Reactive approach is used. The testing approach includes various testing methodologies, such as:

- Unit testing: Testing individual components of the computer vision system.
- System and integration testing: Evaluating the combined performance of the system components in different fog and haze conditions.
- Performance testing: Assessing the system's efficiency, response time, and processing speed.
- User acceptance testing: Gathering user feedback on the system's performance and usability.
- Beta testing: Releasing the system to a select group of users for real-world testing and feedback.

3. Test - Schedule

Level 0: Environment Setup:

Setting up a test environment is a crucial step in ensuring the efficiency and reliability of a dehazing image project. A well-prepared test environment enables developers to validate the performance of their algorithms and identify potential issues. Here is a brief overview of a one-week test environment setup for a dehazing image project:

Day 1-2: Installing necessary software and libraries.

Begin by installing the required software, such as Python and an IDE, along with necessary libraries like OpenCV, NumPy, and any specific dehazing algorithm libraries. Verify that all installations are successful and compatible with the project.

Day 3: Preparing test datasets.

Gather a diverse dataset of hazy images with varying degrees of haze, image sizes, and formats. These images will be used to test the dehazing algorithms and evaluate their performance.

Day 4: Implementing unit tests.

Create unit tests for each component of the dehazing project, including loading and saving images, estimating atmospheric light, calculating the transmission map, and dehazing the image. Use a testing framework like pytest to facilitate the process.

Day 5-6: Conducting test runs and refining tests.

Perform initial test runs using the prepared dataset and unit tests. Analyze the results, identify any issues or bottlenecks, and refine the tests as needed to improve their accuracy and coverage.

Day 7: Finalizing the test environment and documentation.

Finalize the test environment setup, ensuring all components function correctly and efficiently. Document the test environment setup process, including any configuration settings, to facilitate future maintenance and updates.

The process of time: The environment setup chapter took 1 week to complete.

Level 1: Unit Testing:

Unit testing is an indispensable practice in software development, which aims to confirm the accuracy and dependability of individual modules within a project. The primary objective of a dehazing image project is to eliminate haze from images using specialized algorithms.

To perform unit testing for a dehazing image project, the project must be broken down into smaller, testable components. These components may include loading and saving images, determining atmospheric light, calculating the transmission map, and performing the actual dehazing of the image.

Developing unit tests for each of these components enables developers to validate the precision and efficacy of the dehazing algorithms. As the project progresses, these unit tests form a solid foundation for code maintenance and enhancement, ensuring the project's ongoing reliability. Here's a tailored outline of a 2-week plan for unit testing in a dehazing image project:

Week 1, Days 1-3: Identifying testable components Start by identifying the testable components of the dehazing system, such as the dehazing algorithm, image preprocessing, and post-processing functions. Define the expected behavior and inputs/outputs for each component.

Week 1, Days 4-7: Designing and developing unit test cases Create test cases for each component, focusing on edge cases and potential failure points. These test cases should cover a range of inputs, such as images with varying haze intensities and different file formats.

Week 2, Days 1-4: Executing unit test cases Run the unit test cases on each component, documenting any issues or failures that arise. Ensure that each component performs as expected and adheres to the defined inputs/outputs.

Week 2, Days 5-6-7: Analyzing results and adjusting, analyzing the test results, and identifying areas where components can be improved or optimized. Adjust the components as needed, addressing any issues or failures identified during the testing process.

The process of time: The unit testing chapter took 2 weeks to complete.

Level 2: System and Integration Testing:

System and integration testing play pivotal roles in ensuring a dehazing image project's components work harmoniously and the entire system delivers the desired results. Here is a bespoke outline of a 4-week plan for system and integration testing in a dehazing image project:

Week 1: Establishing test scenarios and devising test cases Start by pinpointing various test scenarios that encompass diverse aspects of the dehazing process, such as handling multiple image formats, haze intensities, and image dimensions. Craft test cases that consist of specific inputs, anticipated outputs, and acceptance criteria for each scenario.

Week 2: Developing integration tests Design integration tests that assess the communication between the dehazing system's distinct components, such as image loading, atmospheric light estimation, transmission map computation, and the actual dehazing process. Use a testing framework like pytest to facilitate the process.

Week 3: Executing integration and system testing Carry out the integration and system tests using an assorted dataset of hazy images. Examine the results to detect any issues or inconsistencies in the dehazing process. Rectify any bugs, optimize the algorithms, and iteratively run the tests to confirm that the entire system operates efficiently.

Week 4: Concluding and documenting the testing process Upon finalizing all testing stages, document the test scenarios, cases, results, and any identified issues. Include details on the test environment, configuration settings, and any insights gained throughout the testing process. This documentation will be crucial for future maintenance and updates.

The process of time: The system and integration testing chapter took 4 weeks to complete.

Level 3: Performance Testing:

Performance testing is essential in a dehazing image project to ensure that the system operates efficiently and achieves the desired quality standards under a variety of conditions. Here is a tailor-made outline of a 2-week plan for performance testing in a dehazing image project:

Week 1, Days 1-3: Defining performance metrics and benchmarks Start by identifying crucial performance metrics, such as response time, resource utilization, and throughput. Set benchmark values for these metrics to evaluate the dehazing system's performance against predetermined goals.

Week 1, Days 4-7: Designing performance test scenarios and cases Create test scenarios that encompass a broad range of conditions, including different image formats, haze intensities, and image dimensions. Develop test cases for each scenario, specifying particular inputs, expected outputs, and performance criteria.

Week 2, Days 1-2-3-4: Analyzing results and optimizing the system Examine the performance test results, pinpoint any bottlenecks or areas needing improvement, and refine the dehazing algorithms accordingly. Iteratively re-run the tests to confirm that the system meets the established performance benchmarks.

Week 2, Day 5-6-7: Finalizing and documenting the performance testing process upon completing the performance testing phase, document the test scenarios, cases, results, and any identified issues or enhancements. Include information about the test environment, configuration settings, and insights gained throughout the process. This documentation will prove invaluable for future maintenance and updates.

The process of time: The performance testing chapter took 2 weeks to complete.

Level 4: User Acceptance Testing:

User Acceptance Testing (UAT) is a vital phase of a dehazing image project in which the system is assessed for usability and functioning from the perspective of the end user. Here's a customized framework for a 2-week UAT strategy in a dehazing image project:

Week 1, Days 1-3: Defining UAT criteria and objectives Start by identifying the user acceptance criteria based on the system's functional requirements and user expectations. The criteria might include ease of use, speed of dehazing, quality of dehazed images, and compatibility with various image formats.

Week 1, Days 4-7: Designing and developing UAT test cases Create test cases that mimic real-world user interactions with the dehazing system. These could include loading a hazy image, applying the dehazing function, and saving the dehazed image. Ensure the test cases cover all user stories and functionalities.

Week 2, Days 1-4: Executing UAT test cases Run the UAT test cases with a group of end-users or user representatives. Observe their interactions with the system, capture their feedback, and document any issues or challenges they encounter.

Week 2, Days 5-6: Analyzing feedback and adjusting Analyze the user feedback and test results, identify areas for improvement, and adjust the dehazing algorithms or user interface accordingly. The goal is to enhance user satisfaction and the overall user experience.

Week 2, Day 7: Finalizing UAT and documenting the process upon completion of UAT, document the entire process, including test cases, user feedback, identified issues, and implemented adjustments. This documentation will be invaluable for future iterations and updates.

The process of time: The user acceptance testing chapter took 2 weeks to complete.

Level 5: Beta Testing:

Beta testing is an essential phase in a dehazing image project, during which real-world users evaluate the system's performance, usability, and functionality. Here's a customized outline of a 4-week plan for beta testing in a dehazing image project:

Week 1: Preparing for beta testing Select a diverse group of potential users who will participate in the beta testing process. Prepare a beta testing guide that outlines the objectives, test scenarios, and instructions on how to provide feedback on their experience.

Week 2: Distributing the beta version and gathering feedback Release the beta version of the dehazing system to the selected users. Encourage them to use the system in various scenarios, such as dehazing images with different haze intensities, image formats, and sizes.

Week 3: Analyzing feedback and making improvements Collect and analyze user feedback to identify common issues, trends, and areas for improvement. Adjust the dehazing algorithms or user interface as needed to enhance the overall user experience and address any identified issues.

Week 4: Re-testing and finalizing the beta testing process Distribute the updated beta version to the same group of users for re-testing. Ask them to verify if the previously identified issues have been resolved and if there are any new concerns. Analyze their feedback and make any final adjustments to the system.

Week 4: Documenting the beta testing process upon completion of the beta testing phase, document the entire process, including participant selection, test scenarios, user feedback, identified issues, and implemented adjustments. This documentation will be invaluable for future updates and ongoing system improvements.

The process of time: The beta testing chapter took 4 weeks to complete.

Level 6: Final Analysis and Report:

The final analysis and report phase of a dehazing image project involves a thorough review of the entire development and testing process, assessing the project's success, and documenting the findings. Here's a bespoke outline of a 2-week plan for the final analysis and report in a dehazing image project:

Week 1, Days 1-4: Gathering data and evaluating the project Collect data from all development and testing phases, including unit testing, integration testing, performance testing, user acceptance testing, and beta testing. Assess the project's success based on predefined criteria such as meeting functional requirements, user satisfaction, and performance benchmarks.

Week 1, Days 5-7: Identifying areas for improvement and future work Analyze the collected data and feedback to pinpoint areas for improvement or future enhancements. Consider aspects like algorithm optimization, user interface refinements, compatibility with additional image formats, and integration with other systems or platforms.

Week 2, Days 1-4: Compiling lessons learned, and best practices Review the entire project timeline and identify lessons learned, best practices, and areas where the development process could have been improved. These insights can be invaluable for future projects or ongoing system maintenance and updates.

Week 2, Days 5-6: Creating the final report Develop a concise final report that includes the following sections:

a. Project overview: A brief description of the project's goals, scope, and objectives.

b. Development process: A summary of the development and testing phases, with an emphasis on the methodologies and tools used.

c. Results and analysis: An evaluation of the project's success in meeting its objectives, including an assessment of the dehazing system's performance, and functionality.

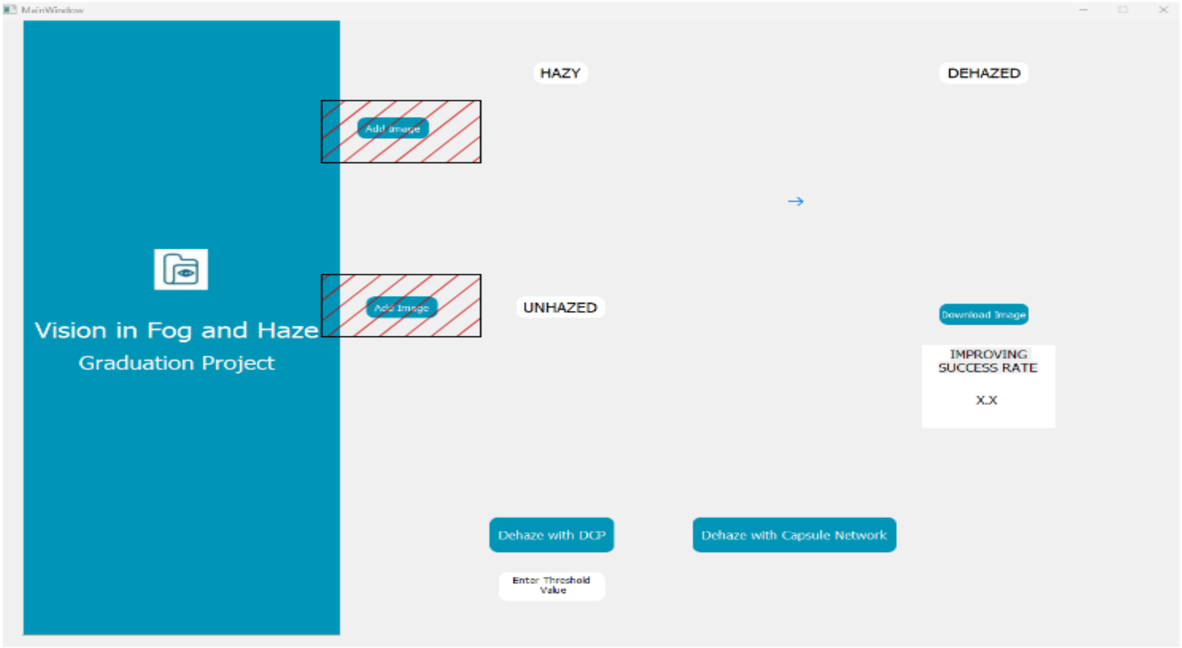
Week 2, Day 7: Distributing the final report Share the final report with the supervisor, the jury members, and any other interested parties. This documentation will be crucial for future projects and ongoing system improvement.

The process of time: The final analysis and report chapter took 2 weeks to complete.

4. Features to be Tested

a. User Interface Test

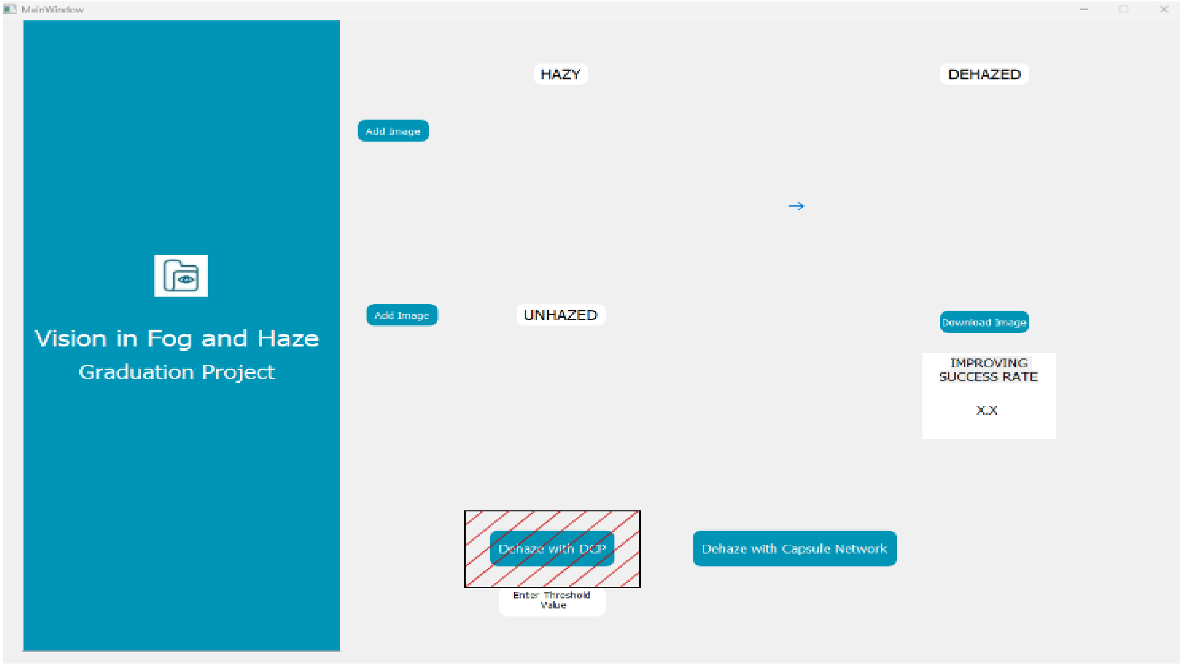
Test 1



- Tests 'add image' buttons are working correctly and putting the photos to the true Directory of the computer.

Figure 1: 'Add Image' Button Explained

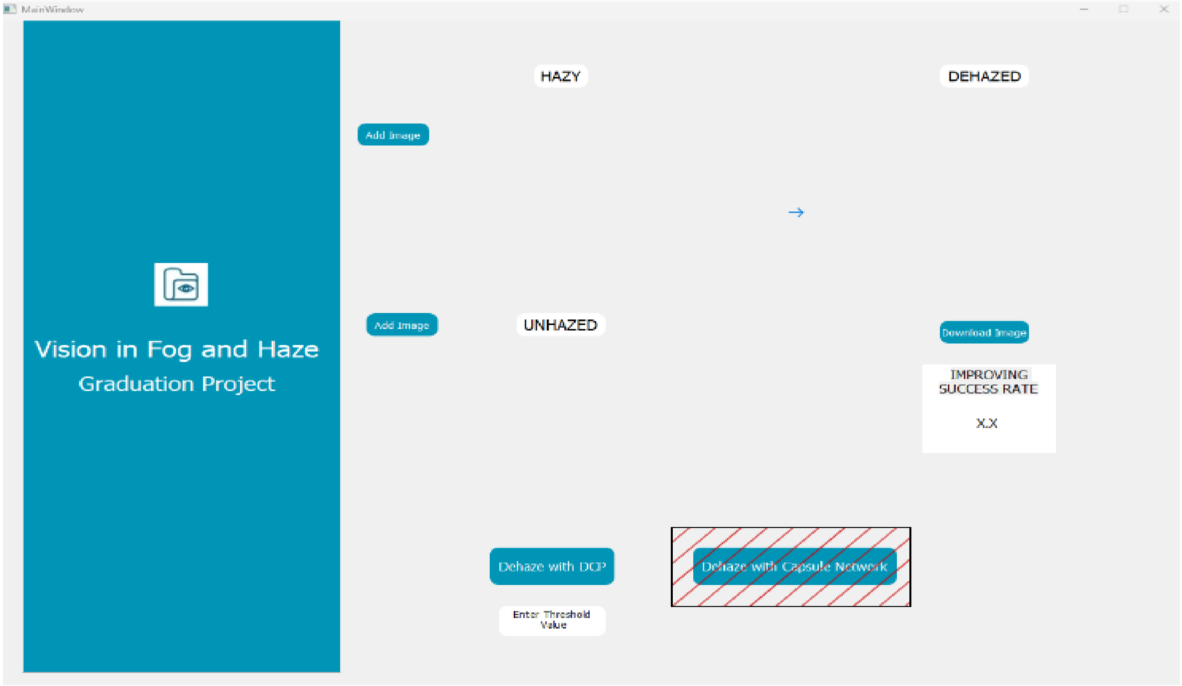
Test 2



- Tests 'Dehaze with DCP' button. Looks the button realy has the image and gives to DCP dehazing function as input. Also look is the button realy has the trashold function from part that we enter the trashold.

Figure 2: 'Dehaze with DCP' Button Explained

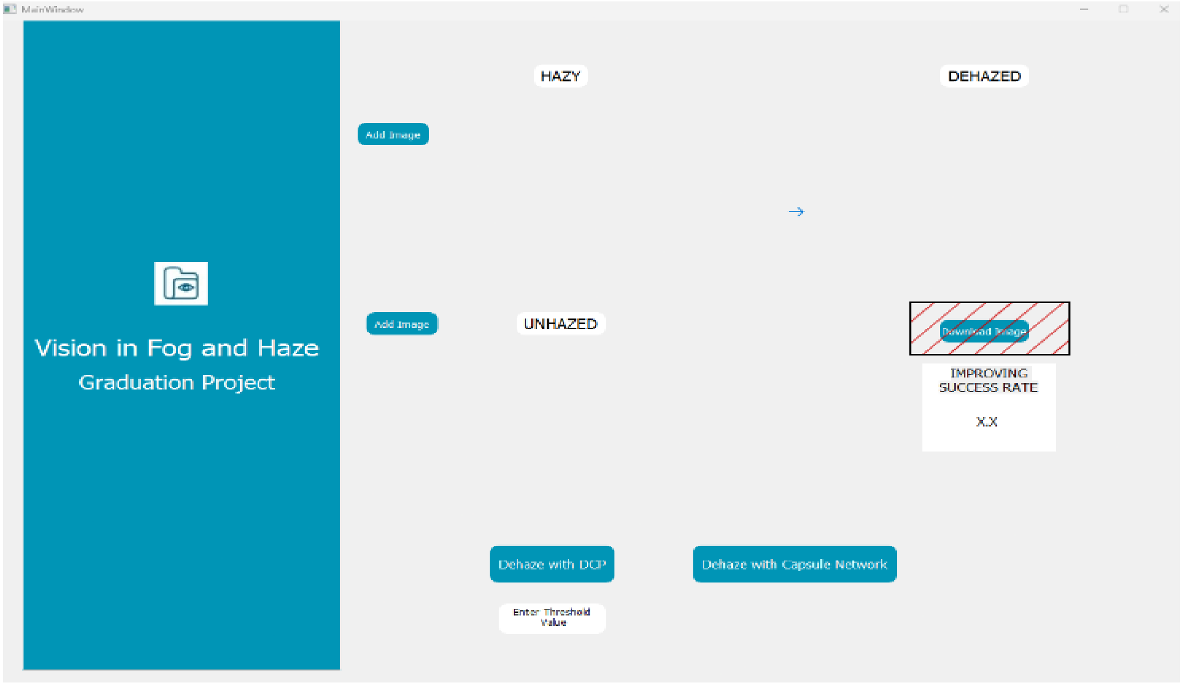
Test 3



- Tests 'Dehaze with Capsule Network' Button. Looks is the button realy has the given photo and gives the photo as input to the function of capusle Network.

Figure 3:'Dehaze with Capsule Network' Button Explained

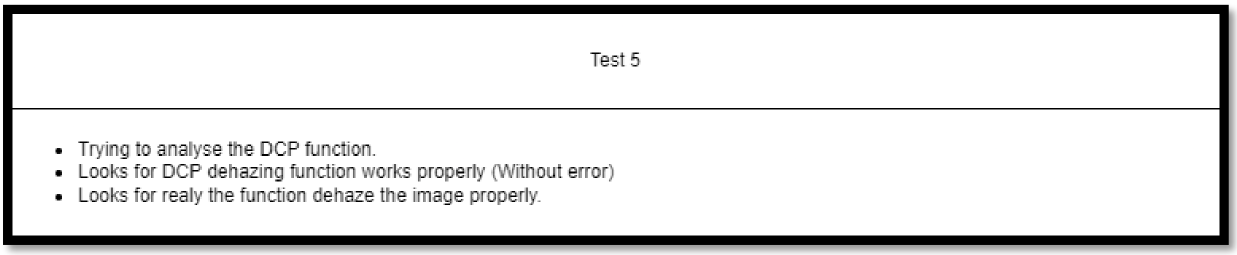
Test 4



- Tests do we realy able to downoad the dehazed image to the spesific folder by clicking 'Download Image' button.

Figure 4: 'Download Image'' Button Explained

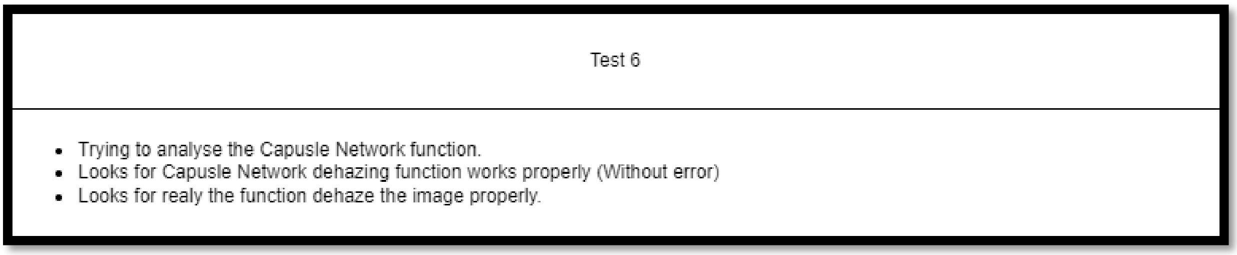
b. Image Dehazing Functions Test



Test 5

- Trying to analyse the DCP function.
- Looks for DCP dehazing function works properly (Without error)
- Looks for realy the function dehaze the image properly.

Figure 5: Test 5 Explained

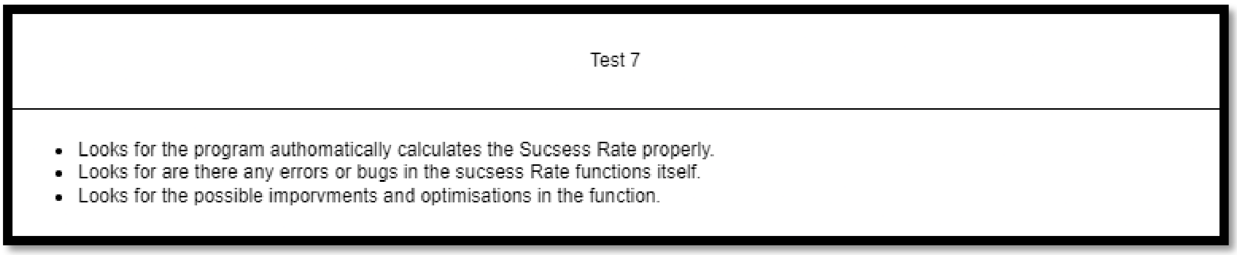


Test 6

- Trying to analyse the Capusle Network function.
- Looks for Capusle Network dehazing function works properly (Without error)
- Looks for realy the function dehaze the image properly.

Figure 6: Test 6 Explained

c. Dehazing Success Rate Test



Test 7

- Looks for the program authomatically calculates the Sucsess Rate properly.
- Looks for are there any errors or bugs in the sucsess Rate functions itself.
- Looks for the possible imporvments and optimisations in the function.

Figure 7: Test 7 Explained

5. Testing Methodology

a. Overview

Traditional software development approaches assume that software requirements remain consistent throughout the project.

However, as complexity grows, requirements often change and evolve. To address this, we have adopted an agile approach for our web application project.

We follow incremental development, where new features are continuously added. Given time constraints, we prioritize efficiency.

Hence, we employ incremental testing, testing each new component and feature to ensure they function as intended.

This approach enables us to identify and fix any bugs before the next release, ensuring a more reliable and stable system.

6. Conclusion

In conclusion, the comprehensive testing activities undertaken for the dehazing image project have played a pivotal role in evaluating and optimizing the computer vision system. As the project concludes, it is of utmost importance to prepare a detailed final report that encompasses all aspects of the testing process. This final report should serve as a testament to the rigorous efforts invested in ensuring the system's robustness, usability, and effectiveness in addressing real-world fog and haze conditions.

The final report should provide a clear summary of the test results, painting an accurate picture of the system's performance throughout the various testing phases. These phases include unit testing, integration testing, performance testing, user acceptance testing, and beta testing. By analyzing the results and identifying any trends or patterns, stakeholders can gain valuable insights into the system's strengths and areas for improvement.

Another crucial aspect of the final report is the documentation of issues encountered during the testing process. By maintaining a record of these issues, the project team can gain a deeper understanding of potential challenges and weaknesses in the system. This information is invaluable in guiding future enhancements and ensuring the continuous improvement of the dehazing technology.

The final report should also contain recommendations for system improvements based on the findings from the testing activities. These recommendations may include optimizing the dehazing algorithms, refining the user interface, improving compatibility with various image formats, or integrating the system with other computer vision technologies. By incorporating these recommendations, the computer vision system can be further enhanced to better tackle fog and haze conditions, ultimately meeting, and exceeding the desired quality standards.

Moreover, the final report will serve as a valuable resource for the project team and stakeholders, enabling them to make informed decisions regarding the system's future development and maintenance. The insights gained from the testing process and the final report will contribute to the ongoing optimization of the computer vision system, ensuring that it remains effective and relevant in addressing the challenges presented by fog and haze conditions.

In summary, the final report is a vital component of the dehazing image project, as it consolidates the knowledge and experience gained during the testing activities. By documenting test results, issues encountered, and recommendations for system improvements, the final report ensures that the computer vision system is optimized for performance in fog and haze conditions, meeting the desired quality standards and providing an enhanced user experience. This comprehensive conclusion of the project sets the foundation for future advancements in the field of dehazing technology and computer vision systems.

7. References

- He, K., Sun, J., & Tang, X. (2014). Single image haze removal using dark channel prior. IEEE Transactions on Pattern Analysis and Machine Intelligence, 33(12), 2341-2353.

- Ren, W., Liu, S., Zhang, H., Pan, J., Cao, X., & Yang, M. H. (2016). Single image dehazing via multi-scale convolutional neural networks. In European conference on computer vision (pp. 154-169). Springer, Cham.

- Sabini, G., Perona, P., & Recht, B. (2018). FOG: A dataset for studying generalization in computer vision. arXiv preprint arXiv:1805.04788.

- Hinton, G. E., Sabour, S., & Frosst, N. (2018). Matrix capsules with EM routing. In International Conference on Learning Representations.

- Tan, R. T. (2008). Visibility in bad weather from a single image. In 2008 IEEE Conference on Computer Vision and Pattern Recognition (pp. 1-8). IEEE.

- Li, Y., Guo, J., & Gu, S. (2017). A review on single image dehazing methods. Journal of Visual Communication and Image Representation, 46, 180-190.

- Zhang, X., Wang, T., Qi, J., Ehgoetz Martens, K. A., & Redmill, K. A. (2018). DehazeNet: An end-to-end system for single image haze removal. IEEE Transactions on Image Processing, 27(7), 3508-3522.