California State University
Long Beach
Computer Science

# Nao Sorter Specification

*Submitted By:*
Christopher Yip
Anthony Guerra
Chau Trieu
Truc Tran
Fair N. Aboshehwa

*Submitted To:*
Dr. Penzenstadler
Asst. Professor
Dept. of CECS

May 12, 2019

# Contents

# 1 Executive Abstract

The following are test cases to be used to determine whether the functionality of the NAO Robot satisfies our requirements. These test cases were created using the use cases and non-functional requirements in the requirement specifications. Each test case consists of: the criteria of the test, what the test is referenced from, the steps to take in order to execute the test, and the expected outcome. These test cases will be used to see if the system works as intended.

# 2 Test Cases

## 2.1 Unit Level

| | |
|---|---|
| Test Level | Unit Level - Coordinate Accuracy |
| Quality Criterion | Internal and External Quality - Functionality : Accuracy. |
| Description of Test | Ensure the localization function outputs the correct coordinates for the robot. |
| Requirements Reference | Quality requirement : recognize and pick up the right object. |
| Steps of the Test Case | 1. Place the robot in a location with known coordinates (in a room).<br>2. Give the robot coordinates to move to.<br>3. After requests, robot moves to location given.<br>4. Measure actual coordinate of robot with coordinate the robot thinks it's in.<br>. |
| Expected Outcome | The coordinates the robot moves to is within .2 meters of the intended coordinates. |

| | |
|---|---|
| Test Level | Unit Level - Performance Time |
| Quality Criterion | Internal and External Quality - Efficiency: Efficiency. |
| Description of Test | Ensure algorithms are correct and run fast. |
| Requirements Reference | Process requirement: 3 |
| Steps of the Test Case | 1. Check for algorithm correctness (someone other than the one who wrote it).<br>2. Analyze the algorithm and find the running time O notation.<br>3. Check if another working algorithm has better running time (pick the one with the least amount of time takes to run).<br>. |
| Expected Outcome | The system/ robot can respond and provide the output in a fast/ reasonable time. |

| | |
|---|---|
| Test Level | Unit Level - Conciseness |
| Quality Criterion | Internal and External Quality - Usability: Understandability. |
| Description of Test | Every method should be concise and accomplish one task. |
| Requirements Reference | Quality requirement : 4. |
| Steps of the Test Case | 1. Iterate through written methods.<br>2. Verify that the method accomplishes one task rather than a set of tasks.<br>3. Highlight methods that can be split into two or more methods.<br>. |
| Expected Outcome | Methods are sufficiently concise, or they are highlighted so as we can separate them into concise methods. |

| | |
|---|---|
| Test Level | Unit Level - Adequate Comments |
| Quality Criterion | Internal and External Quality - Maintainability: Maintainability. |
| Description of Test | Ensure that the code/methods are commented adequately and readable. |
| Requirements Reference | Quality requirement: 2,3 (easy to use and control, easy maintenance). |
| Steps of the Test Case | 1. Go through each method<br>2. Ensure there is documentation for each method with precise description, input and output type).<br>. |
| Expected Outcome | Code not up to standards is highlighted and can be quickly fixed. |

| | |
|---|---|
| Test Level | Unit Level - Method I/O Testing |
| Quality Criterion | Internal and External Quality -Maintainability: Testability. |
| Description of Test | Methods should give expected output and be able to handle the wrong input. |
| Requirements Reference | Process requirement: 1, 2 |
| Steps of the Test Case | 1. List different inputs to give method.<br>2. Give method the possible inputs one by one.<br>3. Verify the method gives expected output for each input (including wrong input).<br>. |
| Expected Outcome | With valid input, method provides expected value. If the input is invalid, the method handles it accordingly. |

## 2.2 Module Level

| | |
|---|---|
| Test Level | Module Level - Computer Vision Time Efficiency |
| Quality Criterion | Internal and External Quality - Efficiency. |
| Description of Test | Verify that the robot is able to detect recognizable objects within 2 meters in front of it in less than 2 seconds. |
| Requirements Reference | Use Case: Pick Up Object |
| Steps of the Test Case | 1. Cover the robot's camera vision sensor.<br>2. Place an object in front of the robot within 2 meters.<br>3. Uncover the robot's camera vision sensor.<br>4. Note down how long it took for the robot to detect the object.<br>5. Repeat multiple times with different objects, and verify that in all cases, detection time is within two seconds.<br>. |
| Expected Outcome | The robot is able to detect recognizable objects within 2 seconds of being in its vicinity.. |

| | |
|---|---|
| Test Level | Module Level - Object Orientedness of the Software |
| Quality Criterion | Internal and External Quality - Usability: Operability. |
| Description of Test | Ensure that all objects in the software are encapsulated. |
| Requirements Reference | N/A |
| Steps of the Test Case | 1. Verify that all the variables of every class are private.<br>2. Check that every object exposes only necessary functions needed to operate on its hidden data.<br>. |
| Expected Outcome | Every object in the software is fully encapsulated and operable. |

| | |
|---|---|
| Test Level | Module Level - Space and Time Complexity is Minimized |
| Quality Criterion | Internal and External Quality - Resource Utilization. |
| Description of Test | Verify that every module uses efficient and quick data structure and usage of constant and linear algorithms to operate on those data structures. |
| Requirements Reference | N/A |
| Steps of the Test Case | 1. Have every developer in the team analyze the space and time complexity of every module and note them down.<br>2. Verify that every module's space and time complexity is linear or constant.<br>3. For any module having linear running time or memory usage, verify if the running time or memory usage can be made constant by modifying the algorithm or using more efficient data structures.<br>. |
| Expected Outcome | Space and Time Complexity of every module is minimized. |

| | |
|---|---|
| Test Level | Module Level - Every Class has a single responsibility. |
| Quality Criterion | Internal and External Quality - Functionality. |
| Description of Test | Verify that every class in the software follows the "single responsibility" principle. |
| Requirements Reference | N/A |
| Steps of the Test Case | 1. Verify each module has no unnecessary side effects that propagates to other modules.<br>2. No classes should depend on other classes to complete its operation, unless it's absolutely necessary<br>. |
| Expected Outcome | Every class only operates on its own domain. |

| | |
|---|---|
| Test Level | Module Level - Accuracy of Computer Vision. |
| Quality Criterion | Internal and External Quality - Reliability. |
| Description of Test | Each time a tester passes an image containing a recognizable object image to the object detection function, the output should be the correct label of the object detected. |
| Requirements Reference | N/A |
| Steps of the Test Case | 1. Tester takes passes an image to the object detection function.<br>2. Test if the output is the correct label of the image<br>3. Repeat steps for a variety of positive sample images and negative sample images.<br>4. Calculate percentage correct and percentage of false positives<br>. |
| Expected Outcome | The robot should be able to classify any given image with a recognizable object correctly. |

## 2.3 Integration Level

| | |
|---|---|
| Test Level | Integration. |
| Quality Criterion | Internal and External Quality: Functionality. |
| Description of Test | Verify that the robot picks up the specific-shaped object chosen on the app. |
| Requirements Reference | Use Case: Pick Up Object, Quality Requirements: Recognize and Pick Up the Right Object |
| Steps of the Test Case | 1. On the app connected to a robot, choose to pick up an object.<br>2. Select a specific shape to pick up.<br>. |
| Expected Outcome | The robot finds and picks up only an object of the specified shape. |

| | |
|---|---|
| Test Level | Integration. |
| Quality Criterion | Internal and External Quality: Maintainability. |
| Description of Test | Check that teaching the robot to recognize new shapes can then be implemented in the app as a new shape to be chosen for pick up. |
| Requirements Reference | Deployment Requirements: Upload/Install Software Program to the Robot, Process Requirements: The Software Development Process Must Be Object Oriented |
| Steps of the Test Case | 1. Add the recognition of a new shape to the robot.<br>2. Add the choice to pick up the new shape in the app.<br>3. Select the new shape in the app for the robot to pick up.<br>. |
| Expected Outcome | The robot finds and picks up only an object of the new specified shape. |

| | |
|---|---|
| Test Level | Integration. |
| Quality Criterion | Internal and External Quality: Reliability. |
| Description of Test | Verify that an unpaired robot on a WiFi network can be paired with an operating app on the same network. |
| Requirements Reference | System Constraints Requirements: The Robot and the App Device Must Be Paired. |
| Steps of the Test Case | 1. Connect the robot to an available WiFi network.<br>2. Connect the app device to the same WiFi network.<br>3. Connect the robot to the app.<br>. |
| Expected Outcome | The robot successfully pairs with the app. |

| | |
|---|---|
| Test Level | Integration. |
| Quality Criterion | Internal and External Quality: Usability. |
| Description of Test | Verify that the robot returns an error that the app can appropriately inform the user of when it is unable to carry out a command. |
| Requirements Reference | Use Case: Store Object, Quality Requirements: Easy to Use and Control. |
| Steps of the Test Case | 1. On an app connected to a robot, select the command to Store Object without the robot first picking up an object.<br>. |
| Expected Outcome | The app displays an error message to the user that the robot has not picked up any object to store. |

| | |
|---|---|
| Test Level | Integration. |
| Quality Criterion | Internal and External Quality: Portability. |
| Description of Test | Verify that the software installation on the robot pairs and works with both the iOS and Android versions of the app. |
| Requirements Reference | Deployment Requirements: Mobile App Must Be Compatible with Android and iOS. |
| Steps of the Test Case | 1. Pair the robot with an Android version of the app.<br>2. Run through all available robot commands on the app.<br>3. Pair the robot with an iOS version of the app.<br>4. Run through all available robot commands on the app.<br>. |
| Expected Outcome | Both versions of the app are able to pair successfully with the robot and the robot carries out all commands as expected issued from both versions of the app. |

## 2.4 System Level

| | |
|---|---|
| Test Level | System: Object Detection Accuracy. |
| Quality Criterion | Internal and External Quality: Functionality. |
| Description of Test | Verify that the NAO robot recognizes objects to be picked up with at least a 90 percent accuracy within a distance of two meters. |
| Requirements Reference | Use Case - Pick Up Objects. |
| Steps of the Test Case | 1. Put NAO robot in an environment such as a room.<br>2. Place objects within the robot's vision and within two meters. Objects should be both ones it should pick up and those it shouldn't.<br>3. Note the percentage of objects that it detected correctly.<br>. |
| Expected Outcome | Robot should detect which objects it wants within a ninety percent accuracy. |

| | |
|---|---|
| Test Level | System: Object Detection Time. |
| Quality Criterion | Software Product Quality - Performance Efficiency. |
| Description of Test | Measure time it takes for the robot to detect an object. |
| Requirements Reference | Use Case - Pick Up Objects. |
| Steps of the Test Case | 1. Place object within detection range of NAO robot.<br>2. Measure time it takes to detect object.<br>. |
| Expected Outcome | Detection time should be within five seconds. |

| | |
|---|---|
| Test Level | System: Connection Compatibility. |
| Quality Criterion | Software Product Quality - Compatibility |
| Description of Test | Verify that the robot can be connected on different routers without issue. |
| Requirements Reference | System Constraint Requirements 3,4. |
| Steps of the Test Case | 1. Connect robot to multiple different routers.<br>2. Verify the robot connects successfully.<br>. |
| Expected Outcome | Robot can connect to router successfully. |

| | |
|---|---|
| Test Level | System - Software Upgrades. |
| Quality Criterion | Internal and External Quality - Maintainability. |
| Description of Test | Verify that software updates are established successfully. |
| Requirements Reference | Quality Requirement 3 |
| Steps of the Test Case | 1. Upload software to the robot.<br>2. Verify updated functionality works as established.<br>3. Verify unrelated functionality is not affected.<br>. |
| Expected Outcome | Unrelated functionality should work as before. |

| | |
|---|---|
| Test Level | System - Command Recognition. |
| Quality Criterion | Internal and External Quality - Functionality. |
| Description of Test | Verify that NAO robot receives communication from external app. |
| Requirements Reference | System Constraint Requirement 5. |
| Steps of the Test Case | 1. Use external app to send a communication to the NAO robot.<br>2. See if NAO robot receives the communication.<br>3. Verify the communication received was the one sent.<br>4. Repeat for all communications the external app contains.<br>. |
| Expected Outcome | NAO robot receives the correct communication successfully each time. |

## 2.5 Acceptance Level

| | |
|---|---|
| Test Level | Acceptance - Usability, Intuition. |
| Quality Criterion | Quality In Use - Ease of Use. |
| Description of Test | Ensure new users are able to learn robot actions within an adequate time frame. |
| Requirements Reference | N/A |
| Steps of the Test Case | 1. Provide user with our app, a manual, and a robot.<br>2. Give user tasks to execute with robot.<br>3. Measure how long it takes for user to learn how to operate robot.<br>. |
| Expected Outcome | User should be able to learn how to operate the robot within 10 minutes. |

| | |
|---|---|
| Test Level | Acceptance - Efficiency Performance. |
| Quality Criterion | Quality In Use - Efficiency. |
| Description of Test | Ensure that the user receives a result/ report from Robot within expected time. |
| Requirements Reference | N/A |
| Steps of the Test Case | 1. User select an option in a main menu.<br>2. See if the robot takes action and report a result.<br>3. Measure time its take Robot to complete a task.<br>. |
| Expected Outcome | User should physically receive an object or report from Robot. |

| Test Level | Acceptance - Accuracy. |
|---|---|
| Quality Criterion | Software Product Quality - Accuracy. |
| Description of Test | Ensure Robot action coincides with expected action of the user. |
| Requirements Reference | N/A |
| Steps of the Test Case | 1. Have user state what he/she thinks the robot would do by clicking a button from menu.<br>2. User select that option from menu.<br>3. See if the robot takes right action and report the difference between the expected outcome and the actual outcome.<br>. |
| Expected Outcome | Robot actions should coincide with what the user thinks will happen. |

| Test Level | Acceptance - UI Attractiveness. |
|---|---|
| Quality Criterion | Software Product Quality - Attractiveness. |
| Description of Test | Ensure Robot's functions are useful to users.(all options in main menu are used by the user). |
| Requirements Reference | N/A |
| Steps of the Test Case | 1. Provide users with an app and a robot to use.<br>2. Have users execute various tasks with the robot.<br>3. Have users give a brief review of what they liked or didn't like about using the app and robot.<br>. |
| Expected Outcome | Highlight potential problems within app. |

| | |
|---|---|
| Test Level | Acceptance - Reusability. |
| Quality Criterion | Software Product Quality - Maintainability. |
| Description of Test | Ensure every Robot's function can be repeated as much as the user desires. |
| Requirements Reference | N/A |
| Steps of the Test Case | 1. User select an option in a main menu.<br>2. See if Robot take right action and report a result.<br>3. Repeat as necessary.<br>. |
| Expected Outcome | Robot should process the user's command as many times as user requires. |