**Unit Testing Plan  
for Autonomous Object Tracking System**

**• Test Plan**

**Project Team**

**Jonghyeok Lee**

Date

**2022-12-22**

**이종혁 개인 프로젝트**

**Table of Contents**

[1 Introduction 3](#_Toc120161399)

[1.1 Objectives 3](#_Toc120161400)

[1.2 Scope 3](#_Toc120161401)

[1.3 Terms & Acronyms 3](#_Toc120161402)

[1.4 Revision history 3](#_Toc120161403)

[1.5 References 3](#_Toc120161404)

[2 General Information 4](#_Toc120161405)

[3 Unit verification strategy 4](#_Toc120161406)

[4 Units to be tested or not to be tested 5](#_Toc120161407)

[5 Method 7](#_Toc120161408)

[5.1 Test Method 7](#_Toc120161409)

[5.2 Test Case Generation Method 7](#_Toc120161410)

[6 Verification criteria 7](#_Toc120161411)

[6.1 Test 7](#_Toc120161412)

[6.1.1 Test Entry Criteria 8](#_Toc120161413)

[6.1.2 Test Completion Criteria 8](#_Toc120161414)

[7 Testing environment 8](#_Toc120161415)

[8 Tool 8](#_Toc120161416)

1. Introduction
   1. Objectives

본 문서는 자율주행 물체 추적 로봇 프로젝트의 테스팅 계획을 설명한다. 해당 문서의 작성 목적은 로봇 시스템을 구성하는 기능의 수행 조건, 입력에 대한 출력에 대하여 기대 값을 설명하고 성공적인 테스트를 수행하는데 이용하도록 한다.

* 1. Scope

자율주행 물체 추적 로봇 시스템의 구조는 크게 5가지로 구성된다. 물체 추적, 라인 추적, 위험 예방, 우선 순위 판단, 모터 제어의 기능이 그것이며, 본 문서는 해당 기능들로 구성된 자율주행 물체추적 시스템을 설명한다. 해당 기능들은 카메라, 초음파, 적외선 센서 등 HW 모듈에서 수신한 데이터를 바탕으로 구현된 SW를 통해 구현된다.

* 1. Terms & Acronyms

HW : Hardware

SW : Software

ARS : Autonomous Robot System

FOV : Field of view (Camera, Ultrasonic Sensor)

Camera : Pixy2 Camera

* 1. Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Description** | **Author** |
| 1.0 | 2022-12-05 | Unit Test Plan 초안 | 이종혁 |

* 1. References

[1] Software Requirement Specification :   
<https://github.com/sortinghyeok/SRS_Autonomous_Object_Tracking_Robot>

[2] Source Code :  
https://github.com/sortinghyeok/System\_Implementation\_Autonomous\_Object\_Tracking\_Robot

1. General Information

|  |  |
| --- | --- |
| **Category** | **Description** |
| **Project Name** | Autonomous Object Tracking System |
| **Project Manager /Software Engineer** | 이종혁 |
| **Test Manager** | 이종혁 |
| **Test Engineer** | 이종혁 |

1. Unit verification strategy

소프트웨어 단위 테스팅 레벨에서는 소프트웨어 단위 테스트가 검증 목적으로 사용된다.

테스팅에 이용할 소스코드는 Git을 이용해 형상관리를 진행하며, 깃 내용을 Pull하여 구글 테스트 프레임워크로 단위 테스팅을 진행한다. 테스팅이 완료된 소스코드는 다시 깃에 재 업로드 한다.

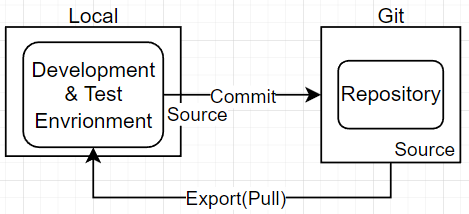
단계 1) 소스코드가 형상관리 툴에 Commit 이 되면 코드 리뷰 수행

단계 2) Git Repository에서 관리하는 코드를 export 하여 Unit 테스팅 진행

단계 3) 테스트 결과 결함이 있는 코드를 수정 후 재 테스트

단계 4) 테스트 완료한 코드를 다시 Git으로 Commit

아래 그림에서 테스트 전략의 도식을 확인할 수 있다.



1. Units to be tested or not to be tested

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Unit functions** | **Description** | **테스팅 수행 여부** |
| ARS\_Init.cpp | enable\_camera  \_interface | Pixy camera의 드라이버를 이용하기 위한 initialization | N |
| ARS\_Init.cpp | enable\_ultrasonic  \_sensor\_interface | 임베디드 보드의 초음파 센서와 연결된 pin의 상태를 읽고, pin에 대한 input/output 모드를 결정한다. | N |
| ARS\_Init.cpp | enable\_motor\_  signal | 임베디드 보드의 DC 모터와 연결된 pin의 상태를 읽고, pin에 대한 input/output 모드를 결정한다. | N |
| ARS\_Init.cpp | enable\_infrared\_  sensor\_interface | 임베디드 보드의 IR 센서와 연결된 pin의 상태를 읽고, pin에 대한 input/output 모드를 결정한다. | N |
| ARS\_Process.cpp | get\_distance | 초음파 센서로부터 값을 입력 받고, 해당 거리 값을 Circular Queue에 저장한다. | Y |
| ARS\_Process.cpp | stack\_distance | Circular Queue에 10개의 거리 데이터를 순차적으로 누적한다. 데이터 누적의 시작점은 이전에 값을 저장한 인덱스의 다음 자리가 된다. | Y |
| ARS\_Process.cpp | target\_camera\_  data\_parser | 카메라 센서로부터 입력 받은 다음, 해당 값을 x, y, width, height, signature 등으로 parsing한다. | Y |
| ARS\_Process.cpp | mean\_filter | 신규 입력된 거리 값을 Circular Queue에 누적한 후, 전체 Circular Queue에 대한 평균 값을 구해낸다. | Y |
| ARS\_Process.cpp | line\_location\_  process | 좌측, 중앙, 우측의 IR 센서로부터 값을 입력 받아 구조체를 형성하여 반환한다. | N |
| ARS\_Process.cpp | sensor\_data\_  process | 라인 정보 구조체, 파싱된 타겟 정보 구조체를 생성한다. 또한 전방 물체에 대한 Circular Queue에 거리 값을 누적하여 그 시점에서 FOV Distance 배열 상태를 저장한다.  최종적으로 라인 정보, 타겟 정보, 거리 값 배열을 하나의 구조체로 형성하여 반환한다. | Y |
| ARS\_Process.cpp | get\_angle | 로봇 전방 물체에 대하여 로봇의 좌, 우측 중 어느 방향에 있는지 파악한 후, DC모터의 회전에 적용할 각도를 구해낸다. | Y |
| ARS\_Process.cpp | object\_tracking  \_process | 매개변수로 입력 받은 Sensor Data를 이용하여, 물체 추적에 따른 모터의 속력, 회전 값을 산출한다.  해당 값들은 구조체의 멤버로 포함되어 반환된다. | Y |
| ARS\_Process.cpp | line\_tracking\_  process | 매개변수로 입력 받은 Sensor Data를 이용하여, 라인 추적에 필요한 속력과 방향 값을 산출한다.  해당 값들은 구조체의 멤버로 포함되어 반환된다. | Y |
| ARS\_Process.cpp | hazard\_  prevention  \_process | 매개변수로 입력 받은 Sensor Data를  이용하여, 위험 예방에 필요한 속력과 방향 값을 산출한다.  해당 값들은 구조체의 멤버로 포함되어 반환된다. | Y |
| ARS\_Process.cpp | priority\_maker | 센서 데이터를 매개변수로 입력 받아, 이를 이용해 물체 추적, 라인 추적, 위험 예방 프로세스에 대응하는 구조체를 각각 생성한다.  각 구조체에 담긴 값들을 읽어 해당 기능이 수행 가능한지 판단하며 다음에 실행될 최 우선순위 기능을 선정한다.  최우선순위로 선정된 기능의 속력과 방향 값을 멤버로 하여 구조체로 생산한다. | Y |
| ARS\_Process.cpp | set\_reader | 최우선 순위 기능에 대응하는 구조체의 멤버 변수를 읽어 들여 이상치가 없는지 확인한다. 만일 이상치가 확인될 경우 값을 조정한다.  조정한 값을 바탕으로 모터의 속력, 방향 값을 포함한 구조체를 반환한다. | Y |
| ARS\_Process.cpp | active\_motor\_  interface | 조정된 속력, 방향 값을 입력 받아 모터 원하는 방향과 속력으로 구동되도록 pwm 값을 적용한다. | Y |

1. Method
   1. Test Method

|  |  |  |
| --- | --- | --- |
| **Method** | **Description** | **Remark** |
| 요구사항 기반 테스팅 | 소프트웨어 단위의 요구사항에 해당하는 소프트웨어 설계 문서의 설계 내용을 기반으로 테스팅 수행 |  |

* 1. Test Case Generation Method

|  |  |  |
| --- | --- | --- |
| **Method** | **Description** | **Remark** |
| 요구사항 분석 | Process Specification에서 정의한 Process  Description 내용을 기반으로 테스트 케이스를 생성 |  |
| 동등분할 | Process Specification에서 정의한 Input 에 대하여 Domain을 나누고, 각각의 Domain에 대하여 입력값을 만들어 테스트 케이스를 생성 |  |
| 경계값 분석 | Process Specification에서 정의한 Input 에 대하여 Domain을 나누고, Domain의 경계를 기준으로 2개씩의 입력 값을 만들어 테스트 케이스를 생성 |  |

1. Verification criteria
   1. Test
      1. Test Entry Criteria

물체 추적 로봇 시스템의 설계 및 구현이 완료되고, SW 시스템의 구현의 초안이 90퍼센트 이상 갖춰진 시점에서 테스트를 시작한다.

* + 1. Test Completion Criteria(언제까지 할 것인지)

|  |  |  |
| --- | --- | --- |
| **Category** | **Description** | **Remark** |
| Requirement coverage | 요구사항 커버리지 : 100%   * 설계 문서를 기반으로 하여, 테스트를 수행할 모든 Unit에 대하여 커버리지를 충족할 경우 | 만약 왼쪽의 커버리지를 충족시키지 못할 경우, 적절한 근거를 제시하여야 한다. |
| Structural coverage | 개발 단계 대상  - Statement coverage: 100%  - Branch coverage: 70%  생산 단계 대상  - Statement coverage: 100%  - Branch coverage: 100% | 만약 왼쪽의 커버리지를 충족시키지 못할 경우, 적절한 근거를 제시하여야 한다. |
| Defect | 모든 테스팅에 대하여 수행 결과는 Pass를 만족해야 한다. | 만약 해당 조건을 만족하지 못할 경우, 적절한 근거를 제시해야 한다. |

1. Testing environment

|  |  |  |
| --- | --- | --- |
| **Category** | **Description** | **Remark** |
| Unit Test | Host PC (Desktop)  - CPU : AMD Ryzen 5 3500X 6-Core  Processor 3.59 GHz  - Memory : 16GB RAM  - OS : Microsoft Windows 11 version 22H2 |  |

1. Tool

|  |  |  |
| --- | --- | --- |
| **Category** | **Description** | **Remark** |
| Unit Test | Google Unit Test Framework |  |