5.1 SQA activity: Defect Detection Through White Box Testing

5.1.1 Test Cases: Dijkstra vs A* with Obstacle on Path

The following test cases highlight the different behaviours of Dijkstra and A* algorithms when faced with obstacles in path planning scenarios. Dijkstra fails because of significant performance degradation due to frequent re-routing around obstacles, whereas A* demonstrates resilience by dynamically adjusting its route based on heuristic information, thereby providing reliable navigation solutions for autonomous systems.

Equivalence Class Partitioning (ECP) for start_node and end_node

- Valid Classes
  - start_node and end_node are valid integers within the grid bounds.
  - $x_{start}, x_{end} \in Z$
  - $y_{start}, y_{end} \in Z$
- Invalid Classes
  - start_node or end_node as non-integer values or out of grid bounds.
  - $x_{start}, x_{end}, y_{start}, y_{end} \notin Z$

Table 5.2: White Box TC1

		Input Variables				
Test ID	**Algorithm**	**start_node**	**end_node**	**ECP**	**Actual Output**	**Error/Defect**
TC001	Dijkstra	(1,1)	(5,5)	Valid	Error: "Path blocked by obstacle"	Significant performance degradation due to frequent re-routing around obstacles
TC002	A*	(1,1)	(5,5)	Valid	List of	

					node ids (int) connecting origin and destination, avoiding the obstacle.	None

5.5.1 Endless erratic behavior Around Destination

These test cases focus on the behaviour of the vehicle when it receives the same destination coordinates but its already there, in such case the vehicle starts behaving erratically

Equivalence Class Partitioning (ECP) for x and y

- Valid Classes
  - X and Y are valid integers within the grid bounds.
  - $x \in Z$
  - $y \in Z$
- Invalid Classes
  - X or Y as non-integer values or out of grid bounds.
  - $X, Y \notin Z$

Table 5.3: White Box TC2

	Input Variables				
Test ID	**x**	**y**	**ECP**	**Actual Output**	**Error/Defect**
TC003	-100.00	40.00	Valid	Vehicle moves to the location (-100.00, -40.00)	None
TC004	-100.00	40.00	Valid	Error: "Stuck in endless loop around the location (-100.00, -	Erratic behaviour of car Around Destination

				40.00)"	

5.5.2 System Shuts Down After Reaching First Destination

This test case focus on the behaviour of the system when vehicle reaches a destination and the system shuts down instead of taking the next destination.

Table 5.4: White Box TC3

	Input Variables				
Test ID	x	y	ECP	Actual Output	Error/Defect
TC005	-100.00	40.00	Valid	System shuts down after reaching destination (-100.00, -40.00)	System does not take next destination, shuts down

5.5.3 System Crashes Due to Invalid Input Types for Coordinates

These test cases focus on invalid input types for coordinates which results in crashing/shutting down the system.

Table 5.4: White Box TC3

	Input Variables				
Test ID	x	y	ECP	Actual Output	Error/Defect
TC006	"abc"	"def"	Invalid	Error	System crashes when given string inputs
TC007	"@#$"	"%^&"	Invalid	Error	System crashes when given special characters

5.5.4 Spawn Point for Vehicle

These test cases focus on validating the error handling of the system when given invalid input variables for spawn points, including None values, excessively large coordinates, and non-integer inputs.

Table 5.6: White Box TC5

	Input Variables				
Test ID	**x**	**y**	**ECP**	**Actual Output**	**Error/Defect**
TC008	None	-133.808	Invalid	Error	Spawn point with None x coordinate causes failure
TC009	-2.0230992692528655	None	Invalid	Error	Spawn point with None y coordinate causes failure
TC010	-2.0230992692528655	-133.808	Invalid	Error	Invalid actor type causes service call failure
TC011	999999999999999999	999999999999999999	Invalid	Error	Large positive x,y coordinate causes service call failure
TC012	“ ”	sdsd	Invalid	Error	Empty or non-integer causes failure

5.5.5 Steering Control

These test cases focus on validating the robot's behavior is either as expected under these conditions or not.

Equivalence Class Partitioning (ECP) for target_linear_speed and target_angular_speed

- Valid Classes
  - Positive integers only
- Invalid Classes
  - Negative Integers
  - None
  - String

Table 5.7: White Box TC6

	Input Variables				
Test ID	**target_linear_speed**	**target_angular_speed**	**ECP**	**Actual Output**	**Error/Defect**
TC013	-1.0	0.0	Invalid	Robot moves backward	Negative linear speed does not stop robot moving backward
TC014	0.5	None	Invalid	Robot turns in unpredictable motion	Missing angular speed does led to unpredictable turns

5.5.6 previous_destination is not initialized or updated correctly

Equivalence Class Partitioning (ECP) for previous_destination

- Valid Classes
  - ∈ Z
- Invalid Classes
  - ∉ Z
  - Empty

Table 5.8: White Box TC7

	Input Variables			
Test ID	previous_destination	**ECP**	**Actual Output**	**Error/Defect**
TC015	None	Invalid	Error	System crash

5.5.7 Jerkiness

This test case focuses on observing the vehicle's behavior for jerkiness and sudden movements on tight curves.

Table 5.9: White Box TC8

	Input Variables				
Test ID	**x**	**y**	**ECP**	**Actual Output**	**Error/Defect**
TC016	-150.0	45.0	Valid	Jerkiness (Sudden Movements) especially on curves	Vehicle exhibits significant jerky motion on tight curves

5.5.8 PID Controllers to perform longitudinal control

Equivalence Class Partitioning (ECP) for target_speed and waypoint

- Valid Classes
    - target_speed > 0
    - waypoint ∈ Z
- Invalid Classes
    - target_speed <= 0
    - waypoint ∉ Z

Table 5.10: White Box TC9.1

	Input Variables			
Test ID	target_speed	**ECP**	**Actual Output**	**Error/Defect**
TC017	0	Invalid	Vehicle oscillates/does not stop	Improper handling of zero target speed

Table 5.11: White Box TC9.2

	Input Variables			
Test ID	waypoint	**ECP**	**Actual Output**	**Error/Defect**
TC018	None	Invalid	Vehicle does not steer or crashes randomly	None waypoint not handled properly

5.5.9 Ackermann Steering Model

This test case focuses on testing for ZeroDivisionError when calculating the turning radius with a zero inner wheel angle.

Equivalence Class Partitioning (ECP) for wheel_base and inner_wheel_angle ∈ R

- Valid Classes
    - wheel_base > 0
    - inner_wheel_angle ∈ R
- Invalid Classes

→ wheel_base ≤ 0, inner_wheel_angle: Ø

Table 5.12: White Box TC10

Test ID	Input Variables: wheel_base	Input Variables: inner_wheel_angle	ECP	Actual Output	Error/Defect
TC019	-2	0.5	Invalid	ZeroDivisionError	Given inner_wheel_angle = 0, the calculation for the turning radius results in a division by zero. This will cause the program to raise a ZeroDivisionError. Therefore, the actual output in this case is an error rather than a valid pair of steering angles.

5.5.10 Spawning the vehicle

This test case focuses on verifying the system's behavior when given a valid vehicle name, ensuring that the success flag accurately reflects whether the vehicle was actually spawned.

Equivalence Class Partitioning (ECP) for vehicle_name

- Valid Classes
  - vehicle_name: String
- Invalid Classes
  - vehicle_name: None

Table 5.13: White Box TC11

	Input Variables			
Test ID	vehicle_name	**ECP**	**Actual Output**	**Error/Defect**
TC020	Car1	Valid	True	The success flag is always set to True without verifying if the vehicle was actually spawned.