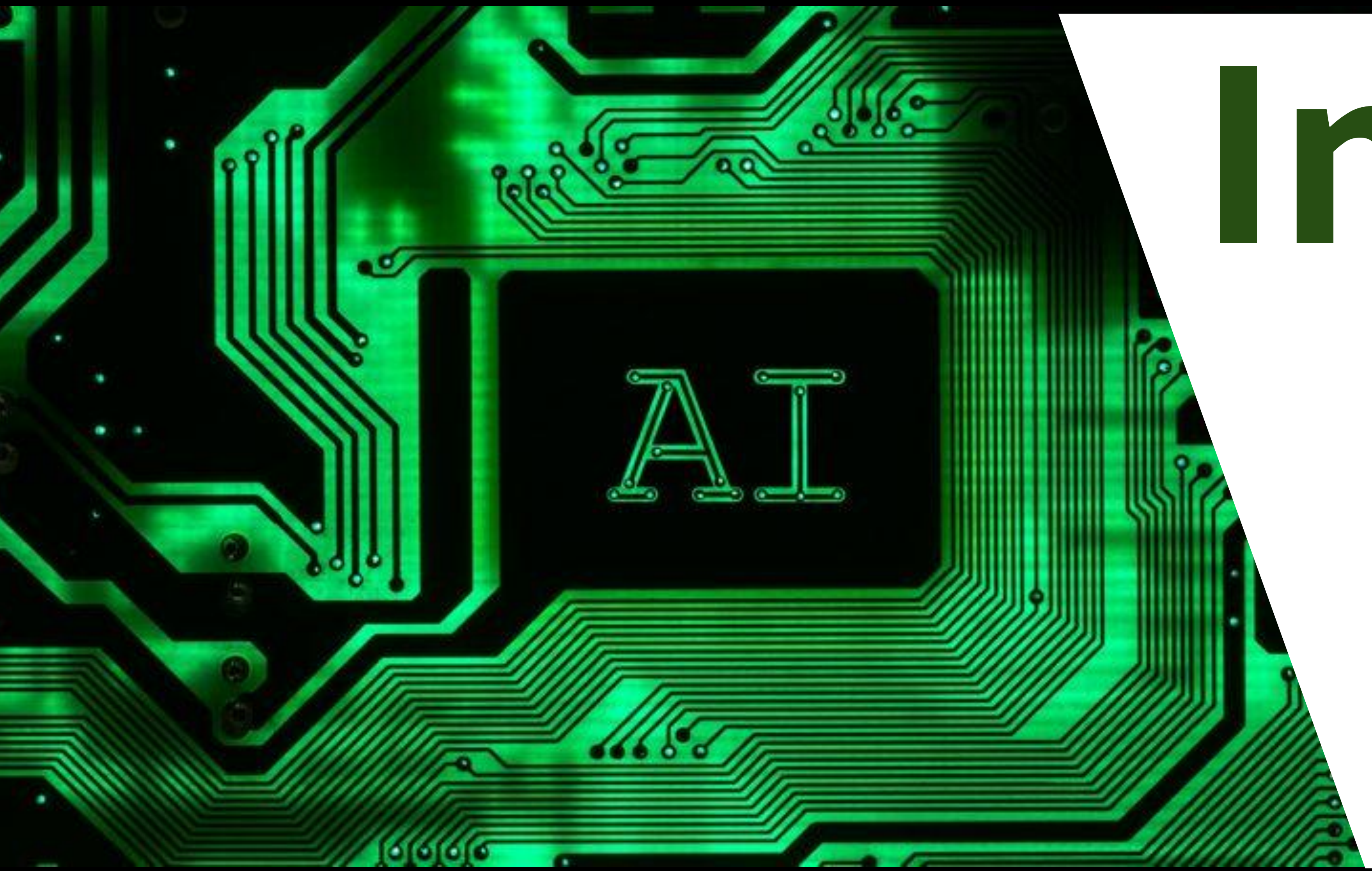
IFS DriverAlert



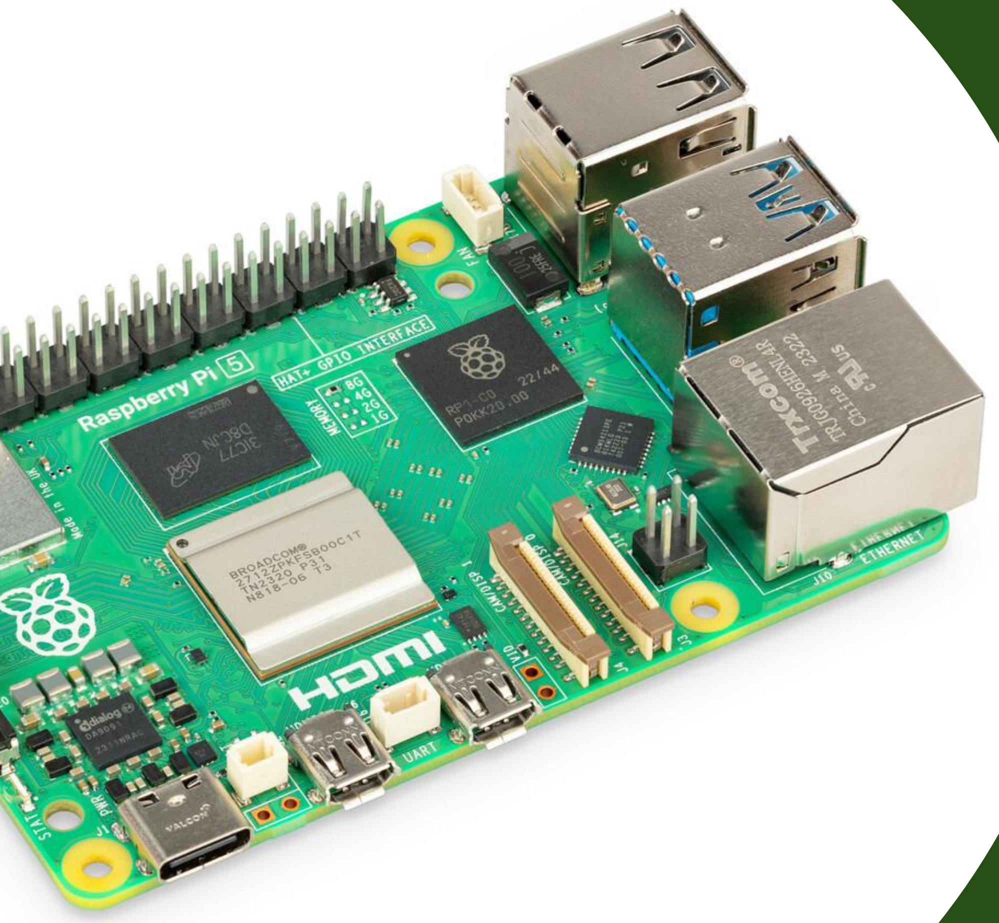
Intelligent Focus System

Feras Aljoudi

Ihab Mohamad

Seonyu Park

Quick Overview



What We Showed Last Scrum:

- Our detection system alerts when detecting drowsiness (eye closure, yawning, and looking away).
- Introduced speed detection.
- Implement speed detection to work with the system.
- Implementing detection modes (On/Auto).

Feedback & Concerns from Last Scrum:

- Having different alert levels based on how many times a sign is detected.
- Having different alert messages.
- Will the system give false alerts if the driver look back while reversing.

Alert System

🟡 **1st Detection**: **Low volume** alert plays 50% of volume for the first detection.

🟠 **2nd Detection within 5 min**: Same alert repeats at a **higher volume**, 75% of volume, for the second detection.

🔴 **3rd or More Detection within 5 min**: Urgent alert plays at **full volume** for a stronger warning.

⏳ **Reset Mechanism**

- If more than 5 minutes pass without repeated signs, the system resets.
- Alert start over from the beginning, at low-volume level.

✔️ **Implementation:**

- Used 'mpg321', an MP3 line command for Linux based systems, to play alerts.
- Tracks time and detection frequency to adjust alert intensity dynamically.

Different Alert Messages

In response to feedback from our last Scrum, we have implemented a system that plays different alert messages, so that drivers receive varied prompts, making the alerts more effective and engaging.

💬 **Eye Closure Alert Messages:**

- 1st & 2nd times: “Please focus on the road”
- 3rd & more: “Closed eyes detected! Stay focused”

💬 **Yawn Alert Messages:**

- 1st & 2nd times: “Consider taking a break”
- 3rd & more: “Yawning detected! Take a rest soon”

💬 **Looking Away Messages:**

- 1st & 2nd times: “Eyes on the road!”
- 3rd & more: “You’re looking away! Please focus on driving”

Test Plan

🎯 Goals of Testing:

- Ensure the system accurately detects drowsiness signs.
- Verify that alerts trigger correctly based on different conditions.
- Confirm the interaction between components (camera, model, GPS, and alerts).
- Assess the system’s usability and reliability in real-world conditions.

📊 Testing Strategy:

- Decision Table Testing – Defines alert actions for different drowsiness conditions.
- Equivalence Class Testing – Ensures various input conditions are correctly handled in and test_mediapipe.py.
- Integration Testing – Verifies how well detection and alert components work together.

Test Cases

🔎 **Eyes/Yawning Detection:**

- Detect eyes/yawning when they are wide open, half open, and fully closed at different time durations.

🔎 **Looking Away Detection:**

- Detect face position when looking straight, at a small angle, and fully turned at different time durations.

🔎 **Progressive Alerts:**

- Repeat detections within/after 5 minutes to verify increasing/resetting alert intensity.

🔎 **Simultaneous Detections:**

- Test different signs at the same time to ensure system prioritization.

🔎 **Multiple Faces in a Frame:**

- Test when a second face appears in the camera view to confirm driver-only detection.

Test Procedure

📋 **Unit Testing:**

- Test eye detection, mouth detection, and turn detection individually.

📋 **Integration Testing:**

- Ensure camera, detection models, and alert system work together.
- Test if alerts trigger correctly based on detection results.

📋 **Decision Table Testing:**

- Validate that different detection conditions lead to the correct alerts.

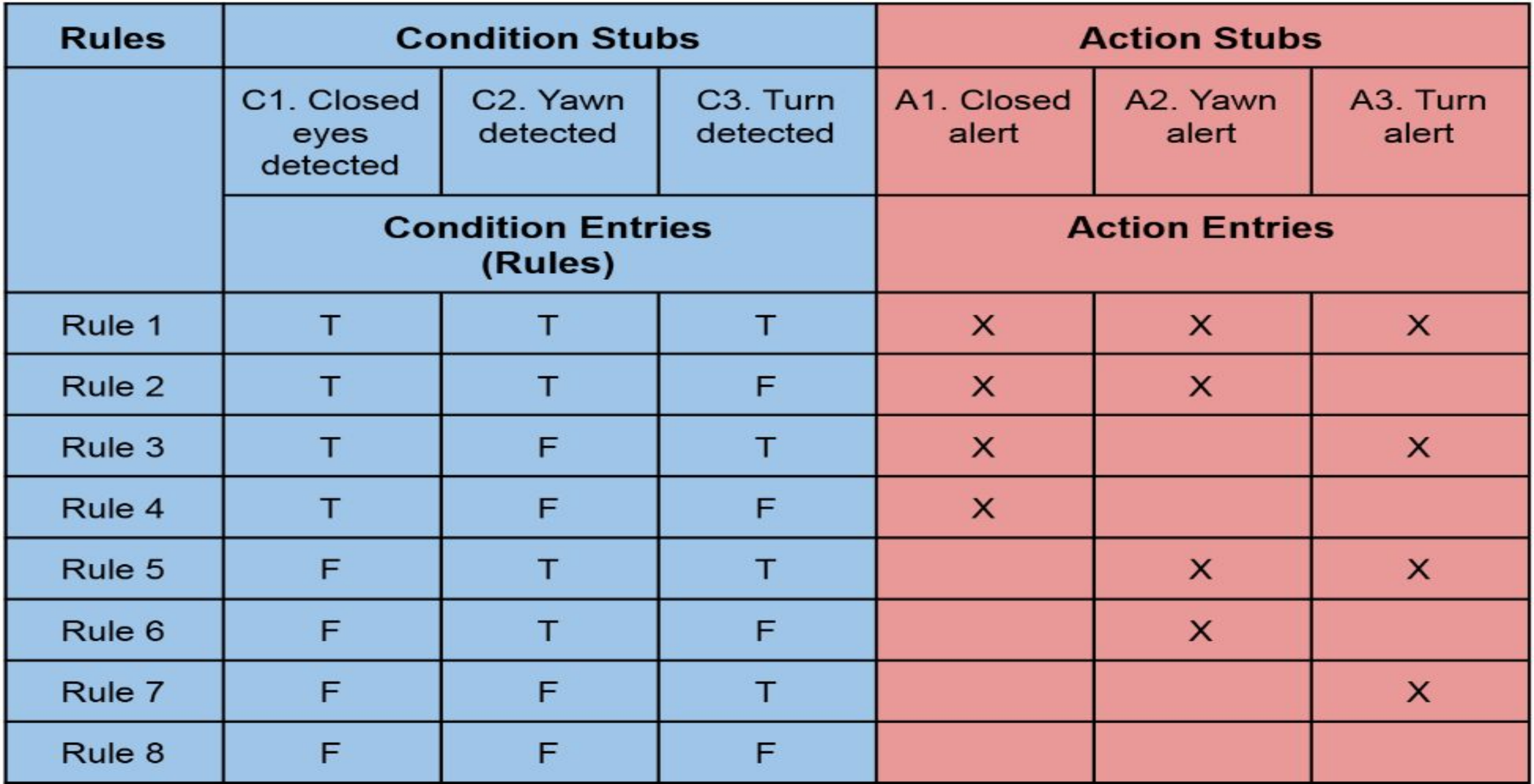
📋 **Threshold-Based Testing:**

- Check if detections trigger alerts at the right time and right ratio.

📋 **Real-World Testing:**

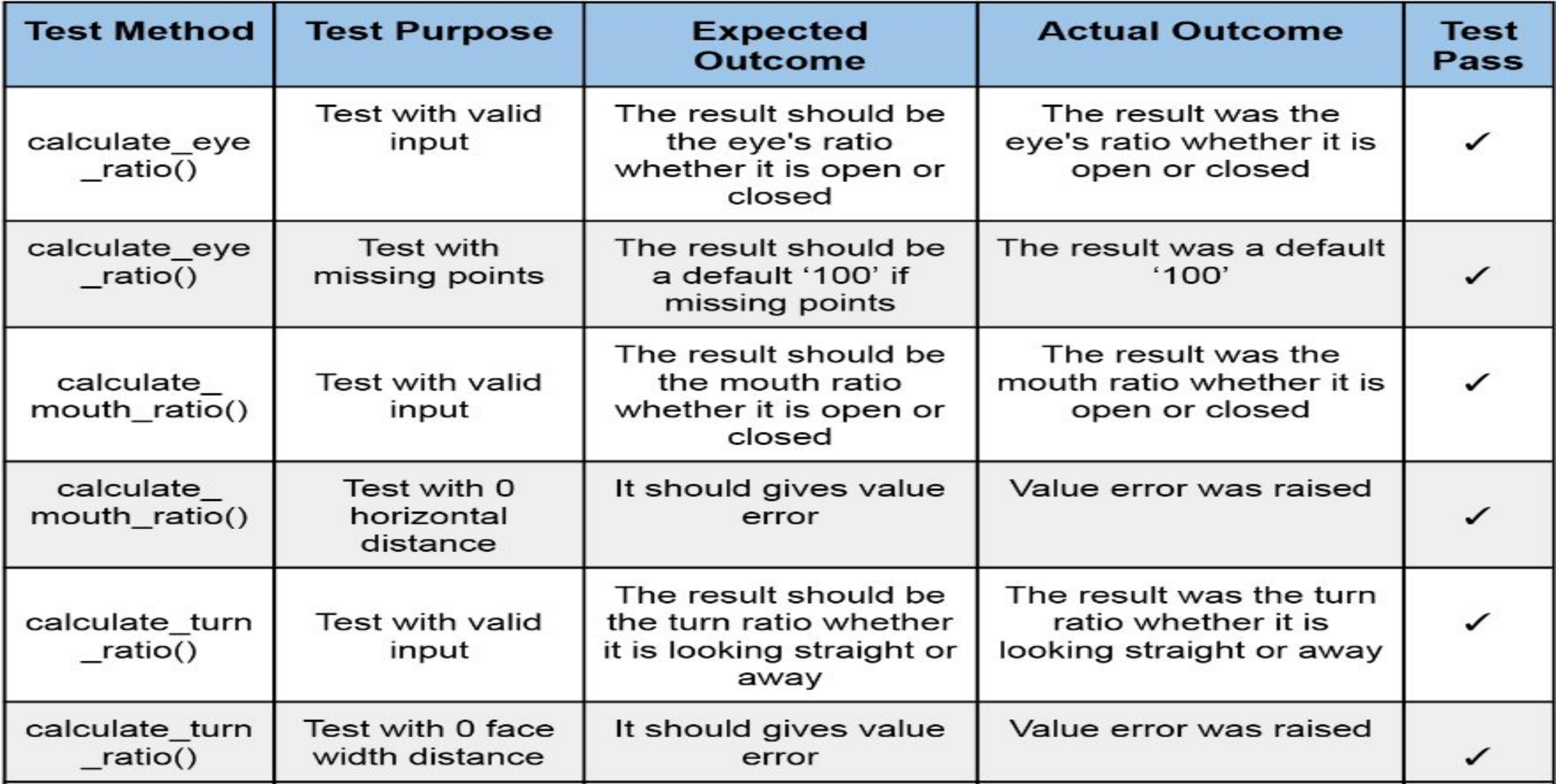
- Ensure system continues working when one feature is blocked (e.g., sunglasses).
- Run tests in a car with different people and varying lighting conditions.
- Verify GPS speed detection and system activation at 20 km/h.

Decision Table Testing



Rules	Condition Stubs			Action Stubs		
	C1. Closed eyes detected	C2. Yawn detected	C3. Turn detected	A1. Closed alert	A2. Yawn alert	A3. Turn alert
	Condition Entries (Rules)			Action Entries		
Rule 1	T	T	T	X	X	X
Rule 2	T	T	F	X	X	
Rule 3	T	F	T	X		X
Rule 4	T	F	F	X		
Rule 5	F	T	T		X	X
Rule 6	F	T	F		X	
Rule 7	F	F	T			X
Rule 8	F	F	F			

Unit Testing



Test Method	Test Purpose	Expected Outcome	Actual Outcome	Test Pass
calculate_eye_ratio()	Test with valid input	The result should be the eye's ratio whether it is open or closed	The result was the eye's ratio whether it is open or closed	✓
calculate_eye_ratio()	Test with missing points	The result should be a default '100' if missing points	The result was a default '100'	✓
calculate_mouth_ratio()	Test with valid input	The result should be the mouth ratio whether it is open or closed	The result was the mouth ratio whether it is open or closed	✓
calculate_mouth_ratio()	Test with 0 horizontal distance	It should gives value error	Value error was raised	✓
calculate_turn_ratio()	Test with valid input	The result should be the turn ratio whether it is looking straight or away	The result was the turn ratio whether it is looking straight or away	✓
calculate_turn_ratio()	Test with 0 face width distance	It should gives value error	Value error was raised	✓

System Behavior – Sequence Diagram

The below Sequence Diagram represents how different components of the system interact after all enhancements have been added.

IFS_SequenceDiagram.pdf

Demo

Thank you