V-Cycle of Function 3:

Requirements:

- SW01: The system must be able to detect the pupil of the conductor's eye
- SW02: The system must be able to detect the direction of the movement of the pupil of the conductor's eye
- SW03: The video stream must provide more readable and easy results for the treatment
- SW04: The system must be able to follow the movement of the pupil as long as it is moving
- SW05: Estimate the lighting state according to the movement of the pupil
- HW01 SW06: Interfacing of the camera with the car's system (CAN Bus)
- SW07: The software must follow the AUTOSAR Architecture
- HW02 SW08: Using the OpenCV library on the embedded system
- HW03 SW09: Implement the complete application inside the embedded system
- HW04 SW10: The data must be viewable in an IHM representing a dashboard

Unclear requirements:

- SW03
- SW05
- HW01
- HW04

System requirements:

- SW01, SW02 and SW04: Using Linux operating system on personal computer implemented with OpenCV to perform eye tracking and follow the direction of the eye's pupil.
- SW08, SW09: The Embedded System must perform the same applications described is SW01, SW02 and SW04.
- SW10: The movement of the pupil (headlights) must be represented in the dashboard.
- HW01, HW04: The Pi camera implemented in the raspberry card must treat the data in the implemented program and send the output of the treatment (headlights direction) via Can bus (IHM collects the data).
- HW02 HW03: Functional system embedded inside of the Raspberry Pi 3.

System test:

- SW01, SW02, SW04:
    - Input: User's pupil of the eye from computer's camera.
    - Output: Eye tracking data following the user's eyes' direction.
- SW08, SW09:
    - Input: User's pupil of the eye from Pi camera.
    - Output: Eye tracking data following the user's eyes' direction.
- SW10:
    - Input: Eye tracking data.
    - Output: Direction of the headlight graphically represented in the virtual dashboard.

High Level Design:

SW01, SW02, SW04: As long as the conductor's eye is open, the headlight follows the movement of the eye until it reaches angle limits. If the conductor's closes his eyes, the headlight aims straight forward (lighting angle covers 90°).

Functions:

- Function 1: video_capture(). Reads video from Pi camera and returns it.
- Function 2: video_treatment(). Reads video from video_capture() and returns eye tracking data.
- Function 3: headlight_direction(). Reads eye tracking data from video-treatment() and activates actuators accordingly.

Integration tests:

- Function 1:
  - Input: Data from Pi camera.
  - Output: Real time recording of the video.
- Function 2:
  - Input: video_capture(), the real time recording.
  - Output: Eye tracking data from the real time recording.
- Function 3:
  - Input: video-treatment(), eye tracking data.
  - Output: Actuators' commands and control.

Low Level Design:

Function 1:

```
While true

Camera open

Return Camera
```

Function 2:

```
While true

Read input

HEADLIGHT_STRAIGHT=TRUE

If FACE_DETECTED

  While EYES_OPEN

    If EYES_LEFT

      HEADLIGHT_LEFT = TRUE

    If EYES_RIGHT

      HEADLIGHT_RIGHT = TRUE

   If EYES_UP

      HEADLIGHT_UP = TRUE
```

If EYES_DOWN

HEADLIGHT_DOWN = TRUE

Function 3:

Read input data

If HEADLIGHT_LEFT turn actuator left

If HEADLIGHT_RIGHT turn actuator RIGHT

If HEADLIGHT_UP turn actuator UP

If HEADLIGHT_DOWN turn actuator DOWN

Unit Tests:

Function 1: From the camera must return the video recording.

Function 2: From the video recording must return all the data regarding eye direction.

Function 3: From the eye direction data must return commands for the actuators.