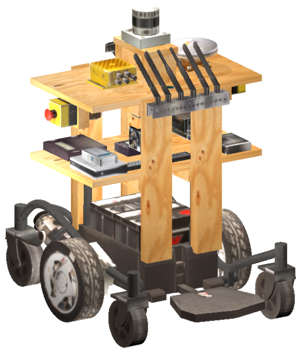
PACMAN

Test Plan



Draft 3

October 21, 2019

Very Good Robotics (VGR)

Clarkson University, EE 416/464 Project Group

Revisions

Version	Primary Authors(s)	Description of Version	Date Completed
1.1	Max Cutugno	Adding more descriptive procedures to the *Features to be Tested* section (2.1, 3.1).	9/29/2019
2.0	Max Cutugno, Avery Oefinger, Adam Romlein	Software and Hardware testing more fully detailed in 2.	10/21/2019

Contents

1. Introduction

The PACMAN (Path Assembling Cartographer, Mapping under Autonomous Navigation) is a vehicle that incorporates both software and hardware components such as cameras, motor controls, LiDAR sensors, accelerometers, GPS receivers, GUI's, etc. It is essential that each of these features/items has a corresponding unit test to check its functionality. At a higher level, these components are interfaced in such a way that the vehicle will be able to run in manual mode or autonomous mode. These modes are critical to the functionality of the PACMAN vehicle; each mode has its own tests to accurately measure its performance.

1.1 Goals

It is imperative that the vehicle's implemented design satisfies the functional and performance requirements for PACMAN (specified in the requirements specification document[1]). Therefore, a list of testing procedures should be carried out to compare measured results to specification constraints. This document aims to describe all these tests pertaining to the software and hardware that is used for the PACMAN vehicle.

1.2 Assumptions

1. When testing the autonomous operation of the vehicle that is tracking a sidewalk:
   1.1. The sidewalk is assumed to always be at least 3 feet in width.
   1.2. The sidewalk is assumed to never incline over 40° (pitch or roll).
   1.3. The sidewalk must have a path where eventually it returns back to its starting point
      1.3.1. If the sidewalk ever splits, the right-most path should eventually lead back to the starting point, or the vehicle will stop at a dead end.

1.3 Risks and Assets

1. Risk - The vehicle in autonomous mode may travel to undesirable locations, either leading to damaging the vehicle itself (falling into a lake, off a cliff, or into a dangerous object), or putting others at risk (running into an active road or a riding into fragile objects or living creatures). Therefore, one must accompany the vehicle during operation as a safe guide; whenever the vehicle is headed into an unsafe location, the safe guide should manually paralyze the vehicle.

2. Asset - Higher level tests (such as testing the operation mode) will be administered on Clarkson University Campus, which has a GPS beacon at disposal. This may allow more accurate GPS data than other means.

1.4 Terms

1. **High-level Test** - A test in which the results are directly comparable with the constraints specified within the requirements specification.
2. **Intermediate-level Test** - A test in which the results are indirectly comparable with the constraints specified within the requirements specification. This may be a part of a high-level test.
3. **Low-level Test** - Also may be referred to unit testing, where each individual component is tested rather than the system as a whole.

1.5 References

1. The detailed document for the Requirements Specification can be found at this Link (Document name for print: *Requirement Specification*).

2. Features To Be Tested

Hardware Testing

This section describes all the hardware features that are to be tested for PACMAN including unit, intermediate-level, and high-level tests. These tests are followed by a brief description of what is to be performed. Testing equipment includes a multimeter.

Note: To safely shut down the system at any time hit one of the E-stops and safely unplug the connector from the batteries to the system.

1. Power
    1.1. Unlatch the E-stops and unplug the barrel connector from the first floor. Test the continuity from the barrel jack when each E-stop is pressed.

P | F	Both E-stops were unlatched. E-stop 1 created an open circuit when pressed
P | F	Both E-stops were unlatched. E-stop 2 created an open circuit when pressed

1.2. Connect charged batteries to the system battery connector. Note the two 12v batteries should be in series connection.

P | F	The motor controller is powered (receiving 24 volts) and LEDs of the motor controller are lit.
P | F	The 24v, 12v, and 5v rails used to power all the other components is zero volts.

Note: Check the blinking pattern of the motor controller based on the figure below and make sure there are no fault messages

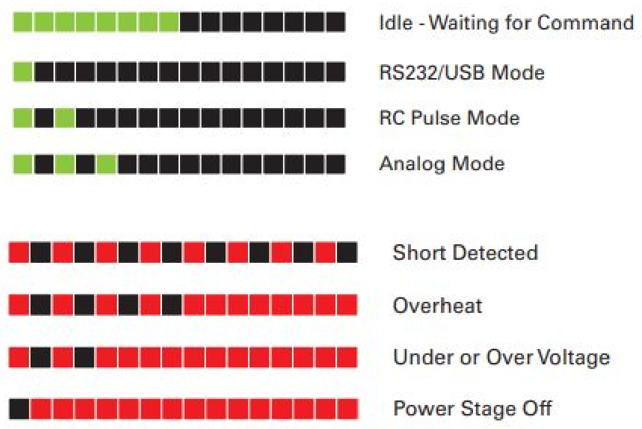
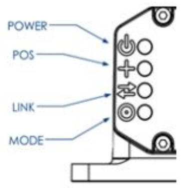
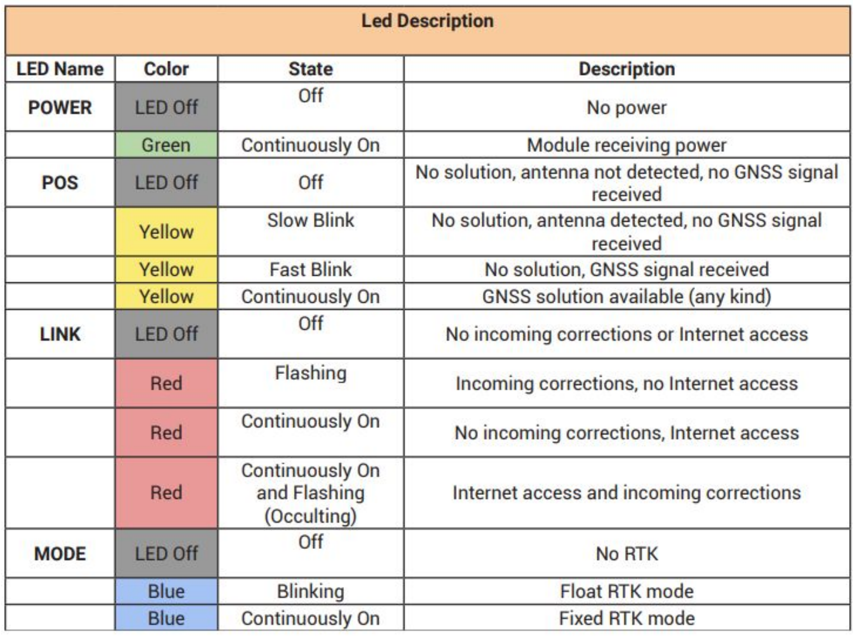


Figure 1 Status LED Flashing Patterns for RoboteQ motor controller

1.3. Make sure E-stops are not latched and connected to the first floor. Press the power button on the second floor of the system.

P | F	The LED of the power button turns on and stays on after release.
P | F	The 24v, 12v, and 5v rails used to power all the other components are their respective voltages.
P | F	The green LED of the LiDAR is continuously on (Figure 2).
P | F	The LEDs of the Piksi GPS are lit.
P | F	The LEDs of the router are blinking.
P | F	The LED of the Jetson Xavier is lit.
P | F	The LED of the USB Hub are lit green.
P | F	The LED of the ethernet Hub are lit green.





Led Description			
LED Name	**Color**	**State**	**Description**
POWER	LED Off	Off	No power
	Green	Continuously On	Module receiving power
POS	LED Off	Off	No solution, antenna not detected, no GNSS signal received
	Yellow	Slow Blink	No solution, antenna detected, no GNSS signal received
	Yellow	Fast Blink	No solution, GNSS signal received
	Yellow	Continuously On	GNSS solution available (any kind)
LINK	LED Off	Off	No incoming corrections or Internet access
	Red	Flashing	Incoming corrections, no Internet access
	Red	Continuously On	No incoming corrections, Internet access
	Red	Continuously On and Flashing (Occulting)	Internet access and incoming corrections
MODE	LED Off	Off	No RTK
	Blue	Blinking	Float RTK mode
	Blue	Continuously On	Fixed RTK mode

Figure 2. Piksi GPS LED Description

Software Testing

This section describes all the features that are to be tested for PACMAN including unit, intermediate-level, and high-level tests. These tests are followed by a brief description of what is to be performed. All tests that specify module should be publishing data to a ROS topic are verified using “rostopic echo”. All bash commands are specified with “$” at the beginning of the line, and are sent from the workspace directory (i.e. “/home/<user>/pacman_ws”). Testing equipment includes a PC.

Before any software testing, send the following two commands:

```
$ cd /home/$USER/pacman_ws
$ source ./devel/setup.bash
```

1. Modular Tests:

1.1. Test Swiftnav Duro GPS Receiver.

```
$ roslaunch pacman_launch piksi_multi_rover.launch
```

P | F	GPS driver is publishing NavSatFix.msg data to topic gps/navsat_bestfix

1.2. Test OS1-16 LiDAR.

```
$ roslaunch pacman_launch os1.launch
```

P | F	LiDAR driver is publishing point cloud data in the form of PointCloud2.msg data to topic /os1/pointcloud.
P | F	LiDAR driver is publishing IMU data in the form of Odometry.msg data to topic os1/imu.

1.3. Test Operator GUI.

1.3.1. Send a debug message to vehicle computer.

1.3.2. Receive and print a debug message from vehicle computer.

1.4. Test RoboteQ MDC2230 Motor Controller.

```
$ roslaunch pacman_launch motor_controller.launch
```

P | F	Motor Controller is publishing Image.msg data to topic webcam/image_raw
P | F	Motor Controller can be commanded to go a specific angular and linear velocity via message Twist.msg through topic /cmd_vel.

1.5. Test battery monitor.

1.5.1. Measure the error that deviates from the constraints listed in Performance Requirement[1] 3.4.

2. High-level test - Manual Operation Mode.

2.1. Have operator control movement through the GUI of vehicle:

2.1.1. Have Vehicle Facilitator place vehicle in a 20-20 meter area.

2.1.2. Have Vehicle Facilitator make sure all E-Stops are disengaged.

2.1.3. Have Vehicle Facilitator turn Vehicle on using power switch.

2.1.4. Have the operator turn on Base Station PC and connect through ssh to the vehicle computer.

P | F	SSH connection succeeds with return code 0.

2.1.5. Start all sensors and master in ROS.

```
$ bash ./src/runscripts/jetson/tmux_core.sh
$ bash ./src/runscripts/jetson/tmux_sensors.sh
```

P | F	All tmux window commands succeed with no runtime errors.

2.1.6. Launch all analytics

```
$ bash ./src/runscripts/jetson/tmux_analytics.sh
```

P | F	All tmux window commands succeed with no runtime errors.

2.1.7. Launch navigation stack

```
$ bash ./src/runscripts/jetson/tmux_navigation.sh
```

P | F	All tmux window commands succeed with no runtime errors and odometry is published.

2.1.8. Through the remote control, have the operator turn the vehicle in a 1 meter circle path.

2.1.9. Through the remote control, have the operator control the vehicle to follow a 1 meter side square path.

2.1.10. Have the operator disconnect through ssh to the vehicle computer and turn off Base Station PC.

2.1.11. Have Vehicle Facilitator turn Vehicle off using power switch.

2.2. Under this operation, measure the error that deviates from the constraints listed in Performance Requirements[1] 3.1 to 3.3, and 3.5.

3. High-level test - Autonomous Operation Mode.

3.1. Have the vehicle placed on a sidewalk and start operation:

3.1.1. Have Vehicle Facilitator place vehicle in the center of a sidewalk.

3.1.2. Follow 2.1.2. - 2.1.6.

3.1.3. Through the terminal or GUI, initiate autonomous mode operation.

P | F	Message “Autonomous mode activated!” appears in std out and GUI.

3.1.4. Have the vehicle autonomously follow the sidewalk path (which is looped) until it reaches back to its starting point (where it should halt movement and end operation).

3.1.5. Have occasionally obstacles (non-threatening such as traffic cones) within the path of the autonomous vehicle.

3.1.5.1. If the autonomous vehicle fails to stop at an obstacle, have vehicle facilitator press down the E-Stop button on the vehicle to stop progression. Also, the operator must also disable operation through the GUI within the base station.

3.1.6. Follow 2.1.9. - 2.1.10.

3.2. Have operator GUI monitor the movement of vehicle throughout 3.1-3.2:

3.2.1. Have the vehicle computer send LiDAR data back to the operator GUI.

3.2.2. Have the vehicle computer send its current global position back to the operator GUI.

3.2.3. Have the vehicle computer send a reroute message to the operator GUI when an obstacle arises within path.

3.3. Under this operation, measure the error that deviates from the constraints listed in Performance Requirements[1] 3.1 to 3.3, and 3.5.

3. Features Not To Be Tested

This section describes all the features that are not to be tested. There are no features in this category for this project.

4. Approach

This section describes the testing approaches to carrying out tests on PACMAN features.

4.1 Test Objectives

Testing the vehicle is done with the intention to reveal software and hardware defects / bugs. When bugs are found, it is important to patch these bugs using test results or debugging information obtained from the testing session. Moreover, it is necessary to analyze recorded test data to avoid future defects. Therefore, the testing process aims to detect, fix, and build upon defects to ensure PACMAN meets its specifications.

4.2 Types of Testing

Common types of tests that will be performed are listed below.

1. **Unit-testing** - Testing each component separately at a low level.
2. **Graphical User Interface Testing** - Testing end-to-end / human-machine experience.

3. **Integration Testing** - Used to verify different hardware or software modules work together.
4. **Functional Testing** - Testing for functional requirements specified by specifications document[1].
5. **Non-Functional Testing** - Testing for performance or measurable constraints specified by the specifications document[1].

5. Artifacts

This section states the test artifacts that will be created while designing and testing the system.

5.1 Test Design Specification (TDS)

The TDS is a referenced document that contains the test conditions, detailed approach, and high level test cases. The test conditions are the constraints of the tests performed. The detailed approach is a clear and informative procedure for testing. The high level test cases sections includes information that defines the general functionality of top level blocks.

5.2 Test Case Specification (TCS)

The TCS is a referenced document that specifies the detailed summary of what scenarios will be tested on the system.

5.3 Validation Checklist

The validation checklist items must all be successfully completed before the system is subject to the testing phase.

Validate	**Y/N**
All components in the system are connected	
There are no loose wires or hazards connected to system	
System can be turned on and off	
Safety shut down procedure works	

System gps and localization data can be collected and read	
System camera sensors can segment images collected and determine paths	

5.4 Test Reports

After the testing process two reports will be generated: Issue Summary Report (ISR) and Test Summary Report (TSR).

Issue Summary Report - The IRS will report all the open issues with the testing and testing procedure. Each issue will include a percentage of severity affecting the system.

Test Summary Report - The TSR will report the testing results of the system. Will include items such as the number of tests performed, checklist of tests that passed and any additional comments for each test.

6. Elements Of Test

This section describes all the resources, roles, staffing and other attributes that contribute to the testing process.

6.1 Staffing

To conduct a PACMAN test, it is important to have the necessary staff to carry out the roles and responsibilities specified in the next sub-section. It is required that the testing team knows how to set up, conduct, monitor, record, and halt any test. Therefore it is essential that the testing team knows the human-machine interfaces of the PACMAN hardware and software. Our team, VGR, is a committed group of five testing individuals; It is however only necessary to have at least three individuals carrying out any single test. Each of our team will be debriefed on the testing approach and the monitoring/control interfaces before carrying out any test. It is worth noting that our testing team is also our implementation team; most features and interfaces should be familiar to each person. Other Clarkson University faculty may aid in our testing endeavours.

6.2 Roles & Responsibilities

This subsection outlines roles and their corresponding responsibilities when performing a PACMAN test.

1. **Test administrator** - Overlooks whole testing operation.
    - 1.1. Is the minimum required personally when performing a unit test; This role will setup, monitor, and record unit tests.
    - 1.2. For a high-level test, this role will additionally ensure proper communication between different roles and lead test.
    - 1.3. Will halt a test if necessary (test becomes unsafe or vehicle exceeds constraint).
2. **Test Operator** - Remotely monitors the digital progress of a vehicle at all times during operation (is needed when performing high and intermediate-level tests).
    - 2.1. This individual stays at a static base station using a remote PC / mobile device.
        - 2.1.1. The remote device has the installed GUI that communicates directly with the vehicle computer. Operator must be able to record results and debug relevant debug information from GUI while operating in autonomous mode.
        - 2.1.2. The operator must control the vehicle in manual mode.
    - 2.2. Must update the status of the vehicle to the test administrator every minute.
        - 2.2.1. Must alert test administrator when vehicle is in operation, when it stopped, and when it is finished with operation.
        - 2.2.2. If the operator sees evidence of malfunction from the GUI, then he or she must report to the administrator and halt vehicle operation.
3. **Vehicle Facilitator and Assistant** - Follows the vehicle during operation and monitors the physical progress of a vehicle at all times during operation (is needed when performing high and intermediate-level tests).
    - 3.1. This individual must follow at the same pace of the vehicle, but not obstruct its path (unless test specifies to).
        - 3.1.1. Must be at a distance where the vehicle does not detect the testing individual and trigger a reroute (in autonomous mode).
        - 3.1.2. Must follow vehicle in a way that it could reach it at time where the vehicle is headed into an undesirable location.
    - 3.2. Must update the status of the vehicle to the test administrator every minute.
        - 3.2.1. Must alert test administrator when vehicle is in operation, when it stopped, and when it is finished with operation.

3.2.2. If the operator sees evidence of malfunction of the vehicle, then he or she must report to the administrator and halt vehicle operation using a physical kill switch.

6.3 Schedule

This subsection describes past PACMAN test milestones. No milestones have been recorded yet.

6.4 Resources

This subsection lists all the resources that are utilized and available during PACMAN testing. No resources have been used at this point.