



## **VIENA**

Veículo Inteligente Elétrico de Navegação Autónoma  
Documentation Guide

### **Current authors:**

**Bruno Tiberio:** [bruno.tiberio@tecnico.ulisboa.pt](mailto:bruno.tiberio@tecnico.ulisboa.pt)

**Revision: 1**

**September 2, 2018**



The group VIENA would like to acknowledge all past and present contributors that are making this project alive and growing. Following is a list of them in no particular order.

---

<b>IST</b>	For allowing the creation of this project and providing a variety of tools and of course the facilities.
<b>FST Lisboa</b>	Contributor with main parts that made possible the powertrain. Among them are two essential parts, the battery pack and inverter.
<b>Dr. João Fernandes</b>	Co-supervisor of project.
<b>Dr. João Sequeira</b>	Co-supervisor of project.
<b>Dr. Paulo Branco</b>	Co-supervisor of project.
<b>Pedro Costa</b>	Former FST leader. Valuable knowledge contributor to get us started with parts supplied by FST team as well as many suggestions to problems that arise or can arise in future. His experience as former FST leader, saved us may time in problems.
<b>André Agostinho</b>	FST member. Provided us help and information specially with battery pack.
<b>André Antunes</b>	Former FST member. Enlightened us in the very beginning of CAN connection with inverter.
<b>AEIST/ BPI</b>	In partnership, provided funding for the project as one of the winners in an open competition fundings.



# Contents

List of Tables . . . . .	vii
List of Figures . . . . .	ix
Nomenclature . . . . .	xi
Acronyms . . . . .	xiii
<b>1 Introduction</b>	<b>1</b>
1.1 Guide Outline . . . . .	1
1.2 Contributions . . . . .	2
<b>2 Main Server configuration</b>	<b>3</b>
2.1 Requirements . . . . .	3
2.1.1 Preparation . . . . .	3
2.1.2 Remote access . . . . .	4
2.2 Troubleshooting . . . . .	4
<b>3 NovatelOEM4 GPS Library</b>	<b>5</b>
<b>Bibliography</b>	<b>7</b>



# List of Tables





# List of Figures

2.1 arp -a command example output . . . . .	4
---	---



# Nomenclature

${}^b_w R$  Rotation matrix that maps physical quantities defined in subscript frame (in this case  $w$ , world frame) to superscript frame (in this case  $b$ , body frame).



# Acronyms

**IST** Instituto Superior Técnico.

**Rpi** Raspberry PI.

**SBC** Single Board Computer.

**VIENA** Veiculo Inteligente Elétrico de Navegação Autónoma.



# Chapter 1

## Introduction

Instituto Superior Técnico (IST) is currently developing research in autonomous electrical vehicles, namely converting from commercial vehicles with upgraded power management. This provides a motivating/attracting setup for new students and a testbed for industry solutions and it is the main motivation for this work.

The main goal of the project is a conversion of an electrical car (Fiat Elettra) property of IST into an autonomous vehicle as framework for future projects or research in this field and also energy efficiency. Within the conversion, researchers, student and collaborators knowledge acquired during academic cycle is put into test and evaluated in a real situation.

With environmental issues and technological waste in mind, this project has also been focused on the reuse of parts and material from other IST projects by the simple fact that they have been replaced by improved versions or will no longer serve the current goals of those projects. Not only it is given a new propose for those parts but it will also allow the cost reduction.

This guide aims to be auxiliary documentation and future memory source as part of the IST project named Veículo Inteligente Elétrico de Navegação Autónoma (VIENA) and as well as final report for fellowship BL43/2018.

### 1.1 Guide Outline

This documentation guide is organized in four main chapters, server, main sensors, controllers, miscellaneous. In the server is provided information about the used hardware, designed pieces, software configuration needed to get started and connections. In main sensors will be discussed the work done with the current available sensors. New sensor additions should be documented under this chapter. In controllers will be reported mainly hardware controllers and software necessary to connect, configure or communicate with it. Miscellaneous contain other parts that do not fit particularly inside any of those chapters but are necessary or may help in the project.

## 1.2 Contributions

Main code contributions are available under Github repository in <https://github.com/brtiberio/VIENA> and will be kept up to date as soon as possible.



## Chapter 2

# Main Server configuration

In order to control the current features already developed and used in the car it is used a [Raspberry Pi](#) (currently version 3 model B). The Raspberry PI, from now on denoted as Rpi, is a Single Board Computer (SBC) affordable and low power, widely used among community and developed by Raspberry Pi Foundation.

### 2.1 Requirements

For using the SBC it is required the following hardware:

- **Raspberry Pi SBC** Recommended version at least 3 since it has a built in wifi.
- **MicroSD card** Recommended with 8Gb capacity at least.
- **Power Supply** Recommended 2.5 Amps . While developing, it is used as the main power for the SBC.
- **Host computer (any OS)** with internet and ability to read MicroSD cards.
- **Ethernet cable** for connecting to router/switch for internet access. It can also be used an crossover cable connecting directly to PC if it is able to share internet connection.

#### 2.1.1 Preparation

This preparation will focus on providing instructions to a minimal headless setup. For this reason, the recommended OS image version is the Raspbian Lite.

For this step it is recommended to be used the [Etcher](#) software for burning the OS image into microSD card. It is assumed the user is familiar with ssh capable software, for example [Putty](#). The next steps are mainly based in the documentation guide provided by [\[1\]](#).

1. Download the OS image, Raspbian Lite version in [here](#)
2. Connect the MicroSD to host computer.

3. Open Etcher and:

- (a) Select OS image or zip file that have been downloaded.
- (b) Select the SD card you wish to write your image to.
- (c) Review your selections and click 'Flash!' to begin writing data to the SD card.
- (d) after successful write, continue to next step.

4. From microSD card, open the boot partition.

5. Create a new blank file with name "ssh" without extension. This will allow to enable the ssh daemon at first boot time.

6. Remove card from host computer and insert on raspberry PI.

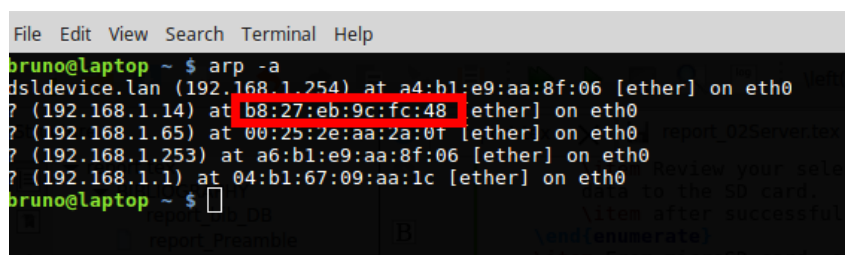
7. Plug in the ethernet cable and connect the Raspberry PI (Rpi) to the same local network as host computer.

8. Plug in the power supply to Rpi.

If everything went as expected, the Rpi will start to boot and prepare the first setup. It will be seen led light blinking indicating activity. After a few minutes, it should be possible to access it remotely via ssh

## 2.1.2 Remote access

The next step is to find the ip address of the Rpi. For example, in linux terminal type `arp -a`. In figure 2.1 is seen an output example. In red stroke is shown the Media Access Control (MAC) address of a Rpi. Every MAC is unique and the first three bytes are fixed in every Rpi wich correspond to organizationally unique identifier [2] associated to Raspberry Pi Foundation, B8:27:EB [3].



```
File Edit View Search Terminal Help
bruno@laptop ~ $ arp -a
dslddevice.lan (192.168.1.254) at a4:b1:e9:aa:8f:06 [ether] on eth0
? (192.168.1.14) at b8:27:eb:9c:fc:48 [ether] on eth0
? (192.168.1.65) at 00:25:ze:aa:2a:0t [ether] on eth0
? (192.168.1.253) at a6:b1:e9:aa:8f:06 [ether] on eth0
? (192.168.1.1) at 04:b1:67:09:aa:1c [ether] on eth0
bruno@laptop ~ $
```

Figure 2.1: arp -a command example output

## 2.2 Troubleshooting

**The command `arp -a` do not show my Rpi**

In that case, try use the `sudo nmap -sS 192.168.1.0/24`, assuming the 192.168.1.0/24 is your local network. This is a time consuming command!

## Chapter 3

# NovatelOEM4 GPS Library

**Date** 24 Jul 2018

**Version** 0.4

**Author** Bruno Tibério

**Contact** [bruno.tiberio@tecnico.ulisboa.pt](mailto:bruno.tiberio@tecnico.ulisboa.pt)

This module contains a few functions to interact with Novatel OEM4 GPS devices. Currently only the most important functions and definitions are configured, but the intention is to make it as much complete as possible.

A simple example can be run by executing the main function which creates a Gps class object and execute the following commands on gps receiver:

- **begin:** on default port or given port by argv[1].
- **sendUnlogall**
- **setCom(baud=115200):** changes baudrate to 115200bps
- **askLog(trigger=2,period=0.1):** ask for log *bestxyz* with trigger ONTIME and period 0.1
- wait for 10 seconds
- **shutdown:** safely disconnects from gps receiver

### Example:

```
$python NovatelOEM4.py
```

Contents:

gps



# Bibliography

- [1] Raspberry Pi Foundation. Installing operating systems images, April 2018. URL <https://www.raspberrypi.org/documentation/installation/installing-images/README.md>. (2018).
- [2] Wikipedia. Mac address. URL [https://en.wikipedia.org/wiki/MAC\\_address](https://en.wikipedia.org/wiki/MAC_address). (2018).
- [3] Wireshark. Oui lookup tool. URL <https://www.wireshark.org/tools/oui-lookup.html>. (2018).

