**Donkey RC Car Setup**

## A close up of a machine Description automatically generated

**Part List…………………………………………………………………………….2**

**Printing Parts…………………………………………………………………….3**

**SD Card Formatting, Powering Up, Wi-Fi Setup……..………….3**

**Connecting to Pi………………………………………………………………..4**

**Software Install…………………………………………………………………5**

**Creating the Car App…………………………………………………………7**

**Calibration………………………………………………………………………..8**

**Driving……………………………………………………………………………..10**

**Training……………………………………………………………………………11**

**Building Donkey Car**

To build the donkey car you will need the following parts:

1. Exceed Racing Desert Short Course Truck 1/16 Scale Ready to Run 2.4ghz (AA Blue)

<https://www.amazon.com/Exceed-Racing-Desert-Course-2-4ghz/dp/9269802086/ref=as_li_ss_tl?_encoding=UTF8&pd_rd_i=9269802086&pd_rd_r=78JFQN575NX4QN9YW9NA&pd_rd_w=WcZLu&pd_rd_wg=GWXCG&psc=1&refRID=78JFQN575NX4QN9YW9NA&linkCode=sl1&tag=donkeycar-20&linkId=1bc61faa0d090a48af6fbaeeaa069b0d>

1. Element14 Raspberry Pi 3 B+ Motherboard <https://www.amazon.com/ELEMENT-Element14-Raspberry-Pi-Motherboard/dp/B07BDR5PDW?tag=donkeycar-20>
2. HiLetgo 2pcs PCA9685 16 Channel 12-Bit PWM Servo Motor Driver IIC Module for Arduino Robot

<https://www.amazon.com/HiLetgo-PCA9685-Channel-12-Bit-Arduino/dp/B07BRS249H/ref=sr_1_4?keywords=servo+driver&qid=1578327015&sr=8-4>

1. SainSmart Wide Angle Fish-Eye Camera Lenses for Raspberry Pi Arduino

<https://www.amazon.com/gp/product/B00N1YJKFS?tag=donkeycar-20>

1. M3x10 screws <https://www.amazon.com/Screws-Mushroom-Phillips-Self-Tapping-Electronic/dp/B07NQCG6JP?tag=donkeycar-20>
2. M2x6 screws <https://www.amazon.com/uxcell-Stainless-Phillips-Tapping-Screws/dp/B01KXTSW6Q?tag=donkeycar-20>
3. 32GB Micro SD Card <https://www.amazon.com/dp/B06XWMQ81P/ref=twister_B07V2PRSXC?_encoding=UTF8&psc=1>

# Power Bank 10000mQSah,TONV Portable External Cell Phone Li-Polymer Battery’s 2 Input and 2 Output

<https://www.amazon.com/10000mah-TONV-Portable-Li-Polymer-Compatible/dp/B078TFHXVY/ref=sr_1_1?keywords=tonv&qid=1578335174&sr=8-1>

Or

<https://www.amazon.com/10000mAh-External-Portable-Compatible-Electric/dp/B07H87HKKM/ref=sr_1_fkmr4_2?keywords=1000mah+portable+lipo+battery+power+bank+2+input+2+output&qid=1579183624&sr=8-2-fkmr4>

1. Vibration-Damping Sandwich Mount https://www.mcmaster.com/5822k4

**3-D Printing**

**About:** GitHub is an open-source repository for code and other documents, like a cloud

1. Download the 3-D print files for the Donkey Car located on the GitHub <https://github.com/michiganaerospace>
2. You will need to print the top and bottom Donkey body, the back and front adapter, and the ping sensor mount (optional, only if actually using ping sensor)
3. Put the STL files into the slicer of your choice (preferably Slic3r)
4. Print all the parts of the donkey car
5. Remove/clean any residue from the 3d prints
6. Assemble (<http://docs.donkeycar.com/guide/build_hardware/>)

**About:** Raspbian is the Raspberry Pi Operating System and Balena Etcher is an SD card formatter

**SD Card Formatting**

## On a PC, download the latest version of Raspbian, as well as Balena Etcher. Select the Raspbian image and your SD card. Click ‘flash’ to start formatting the SD card. Once the card is formatted, you can slide it into the small SD card slot on the underside of the Raspberry Pi (located at the front).

**Powering up the Donkey Car**

## To turn on the donkey car you will need two batteries. One is a Ni-MH battery that fits into the bottom slot of the donkey car and has some pins that hold it in. There is a small connector right by the slot that the Ni-MH battery will plug into. In order to power the Pi, there is a Li-Po battery pack which will plug into the Pi thru a USB to micro USB cable. Once you have these two batteries plugged in the donkey car will be ready to roll or setup.

**Setup the Wi-Fi Connection for Pi**

## To setup Wi-Fi access to the donkey car you will want to power up the Pi on the vehicle. Connect the Pi to a monitor thru the HDMI port on the Pi and plug in a USB mouse and keyboard. Click on the Wi-Fi icon in the top right corner of the Raspberry Pi screen. Select your network and put in the password if necessary

**About**: The Raspberry Pi is a microcomputer with many different capabilities and applications

**Connecting to the Pi**

**About:** SSH (Secure Shell) is a way to access another computer over a network

## If you followed the above instructions to add Wi-Fi access, your Pi should now be connected to your Wi-Fi network. Now you need to find its IP address so you can connect to it via SSH.

To find the IP address, you will want to plug in an HDMI monitor and USB keyboard/mouse into the Pi, if it isn’t already. Boot it. Login with:

**About:** An IP address is a string of numbers that are used to identify individual computers and devices

* Username: **pi**
* Password: **1234**

Then try the command in the Pi terminal:

**About:** A terminal (aka console, command line) is an application in which the user can execute commands directly

ifconfig wlan0

## If this has a valid IPv4 address, 4 groups of numbers separated by dots, then you can try that with your SSH command. If you don't see anything like that, then your Wi-Fi config might have a mistake. You can try to fix with:

**Tip:** You can also use the *Network Analyzer* app on your phone to find the IP address of the Pi on the network

sudo nano /etc/wpa\_supplicant/wpa\_supplicant.conf

In the Pi terminal, type:

ping raspberrypi.local or ping pi-<your car number>.local

Hopefully, one of those methods worked and you have the IP address. On Mac and Linux, you can open Terminal. On Windows, use **PuTTY**,or on Windows 10 you may have SSH via the command prompt.

**Tip:** On most computers, you will want to use the PuTTY app to SSH. See the Quick Start Guide for more details on SSH’ing with PuTTY.

If you have a command prompt, you can try:

**ssh** **pi**@**raspberrypi**.**local**

or **ssh** **pi**@<**your** pi ip address>

**About:** PuTTY is an open source software that enables users to use different network protocols, such as SSH.

**Install Software on Host PC**

**About:** Anaconda is an open source distribution of the Python programming language

1. Install [miniconda Python 3.7 64 bit](https://conda.io/miniconda.html).
2. Open the Anaconda prompt window via Start Menu | Anaconda Prompt
3. Type git. If the command is not found, then install [Git](https://git-scm.com/download/win) 64 bit
4. Install Notepad++

**About:** Git allows us to access and use the latest code from a variety of repositories

1. Install PuTTY 64 bit
2. Install Google Chrome

In the Anaconda prompt window:

* Make a directory you would like to use as the head of your projects and go to it

mkdir projects

cd projects

* Get the latest donkey software from GitHub

git clone https://github.com/autorope/donkeycar

cd donkeycar

git checkout master

* If this is not your first install, update Conda and remove old donkey software

conda update -n base -c defaults conda

conda env remove -n donkey

* Create the Python Anaconda environment

conda env create -f install\envs\windows.yml

conda activate donkey

pip install -e .[pc]

* Create your local working directory:

donkey createcar --path ~/mycar

Note: After closing the Anaconda Prompt, when you open it again, you will need to type conda activate donkey to re-enable the mappings to donkey specific Python libraries

**Install Software on Donkey Car**

If you haven’t already done so:

1. Plug Raspberry Pi into an HDMI monitor, power supply, USB keyboard, and USB mouse.
2. Connect to a Wi-Fi source

Start the software installation by opening the Pi terminal and typing the specified code:

* Updating and upgrading the Pi’s current software

**Fun Fact:** ‘apt’ stands for ‘advanced package tool’; apt-get is used to install, remove, and perform operations on installed software packages

**Fun Fact:** ‘sudo’ stands for ‘superuser do’

|  |
| --- |
| sudo apt update  sudo apt-get upgrade |

1. Changing Raspberry Pi configuration

|  |
| --- |
| sudo raspi-config |

* Change default password to 1234
* Enable interfacing options – I2C
* Enable interfacing options – Camera
* Enable interfacing options – SSH
* Advanced Options – Expand filesystem
* Hit enter then reboot

1. Install dependencies

Type the following code or copy and paste from: <http://docs.donkeycar.com/guide/robot_sbc/setup_raspberry_pi/#step-5-connecting-to-the-pi>

|  |
| --- |
| sudo apt-get install build-essential python3 python3-dev python3-pip python3-virtualenv python3-numpy python3-picamera python3-pandas python3-rpi.gpio i2c-tools avahi-utils joystick libopenjp2-7-dev libtiff5-dev gfortran libatlas-base-dev libopenblas-dev libhdf5-serial-dev git ntp |

1. Setup virtual environment

Type the following code or copy and paste from: <http://docs.donkeycar.com/guide/robot_sbc/setup_raspberry_pi/#step-5-connecting-to-the-pi>

|  |
| --- |
| python3 -m virtualenv -p python3 env --system-site-packages  echo "source env/bin/activate" >> ~/.bashrc  source ~/.bashrc |

1. Install Donkey Car Python code

Type the following code or copy and paste from: <http://docs.donkeycar.com/guide/robot_sbc/setup_raspberry_pi/#step-5-connecting-to-the-pi>

* Make a directory you would like to be head of your projects, and go to it

|  |
| --- |
| mkdir projects  cd projects |

* Get latest Donkey Car from GitHub

|  |
| --- |
| git clone https://github.com/autorope/donkeycar  cd donkeycar  git checkout master  pip install -e .[pi]  pip install tensorflow==1.13.1 |

**Create Car Application**

If you’re not already, go into the Pi terminal.

* Type the following to create a set of files to control your car:

|  |
| --- |
| donkey createcar --path ~/mycar |

* To look at the configuration file in the newly created directory, use the command:

|  |
| --- |
| **nano** **myconfig**.py |

Each line has a comment mark. The commented text shows the default value. When you want to make an edit to overwrite the default, uncomment the line by removing the # and any spaces before the first character of the option.

For example:

# STEERING\_LEFT\_PWM = 460

becomes

STEERING\_LEFT\_PWM = 500

**About:** I2C is a protocol for two-wire interfacing to connect low-speed devices like microcontrollers (like Pi’s)

* Configure I2C

Type the following:

|  |
| --- |
| sudo apt-**get** install i2c-tools  sudo i2cdetect -y 1 |

The i2cdetect command should show you a grid of addresses as shown below. If this isn’t what shows up, check the wiring on the Pi.

0 1 2 3 4 5 6 7 8 9 a b c d e f

**00: -- -- -- -- -- -- -- -- -- -- -- -- --**

**10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --**

**20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --**

**30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --**

**40: 40 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --**

**50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --**

**60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --**

**70: 70 -- -- -- -- -- -- --**

**Calibrate Car**

* To edit your configuration file use this code

|  |
| --- |
| nano ~/mycar/myconfig.py |

## Steering Calibration

Make sure your car is off the ground to prevent a runaway situation.

1. Turn on your car.
2. Find the servo (steering, red/white/black wires) cable on your car and see what channel it's plugged into on the PCA board. It should be 1 or 0.
3. In the Pi terminal, type donkey calibrate --channel <your\_steering\_channel> --bus=1
4. Enter 360 and you should see the wheels on your car move slightly. If not enter 400 or 300.
5. Next enter values +/- 10 from your starting value to find the PWM setting that makes your car turn all the way left and all the way right. Remember these values.
6. Enter these values in myconfig.py file as STEERING\_RIGHT\_PWM and STEERING\_LEFT\_PWM.

## Throttle Calibration

1. Find the cable coming from your ESC (brown/yellow/red wires) and see what channel it goes into on the PCA board. This is your throttle channel.
2. In the Pi terminal, type donkey calibrate --channel <your\_throttle\_channel> --bus=1
3. Enter 370 when prompted for a PWM value.
4. You should hear your ESC beep, indicating that it's calibrated.
5. Enter 400 and you should see your car’s wheels start to go forward. If not, it’s likely that this is reverse. Try entering 330 instead.
6. Keep trying different values until you've found a reasonable max speed, and remember this PWM value.

Reverse on RC cars is a little tricky because the ESC must receive a reverse pulse, zero pulse, and another reverse pulse to start to go backwards. To calibrate a reverse PWM setting:

1. Enter the reverse value (likely 330), then the zero throttle value (370), then the reverse value again.
2. Enter values +/- 10 of the reverse value to find a reasonable reverse speed. Remember this reverse PWM value.

Now open your myconfig.py file and enter the PWM values for your car into the throttle controller part:

* THROTTLE\_FORWARD\_PWM = PWM value for full throttle forward
* THROTTLE\_STOPPED\_PWM = PWM value for zero throttle
* THROTTLE\_REVERSE\_PWM = PWM value at full reverse throttle

**About:** PWM stands for pulse width modulation, which is a method of reducing the average power delivered by an electrical signal. This is done by rapidly switching the power supply from off to on, which regulates speed.

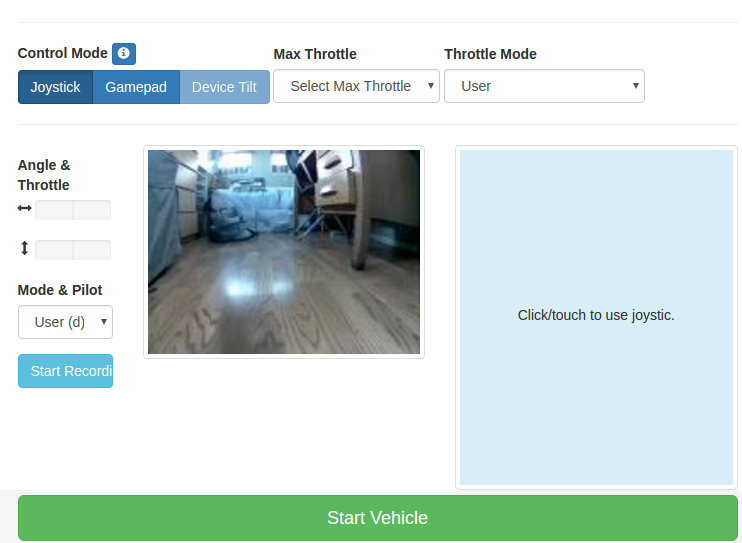
**Tip:** Open two terminal windows, one for the config file, and one for the calibration commands; that way you can put the values in as you get them

**Drive your car**

If you are not already, please [SSH into your vehicle](http://docs.donkeycar.com/guide/robot_sbc/setup_raspberry_pi/#step-5-connecting-to-the-pi), then open your car's folder and start your car. Type the following:

python ~/mycar/manage.py drive

This script will start the drive loop in your car, which includes a feature that enables you to use a web controller. It can be accessed on a computer or any electronic device that has access to the internet. You can control your car from a web browser at the URL: <your car's hostname>:8887

This is what the web controller application should look like:

**Tip:** Make sure to use Google Chrome or Safari; Microsoft Edge often does not display the camera feed correctly

**Driving with Web Controller**

In the browser, there are a couple options for driving:

1. Joystick – This is referring to the blue square to the right of the camera feed. It can be used as a touchpad, but beware of its sensitivity.
2. **Gamepad** – This is what we will primarily use, since the gamepad option allows you to use a gaming controller, such as the Logitech F710.
3. Device Tilt – The idea behind this is being able to use a device like a phone and tilt from side to side, however this is not a very reliable method.

**Features**

* Recording - Press Start Recording to start recording images, steering angles and throttle values.
* Throttle mode - Option to set the throttle as constant. This is used in races if you have a pilot that will steer but doesn't control throttle.
* Mode & Pilot - Choose this if the pilot should control the angle and/or throttle. When we want to autonomously drive, we will switch this option to “Local Pilot” instead of “User”
* Max throttle - Limits the maximum throttle. This can help reduce the sensitivity of the throttle, since the cars tend to be very fast.

**Keyboard Shortcuts (if using keyboard, put into Gamepad mode)**

* space : stop car and stop recording
* r : toggle recording
* i : increase throttle
* k : decrease throttle
* j : turn left
* l : turn right

**Driving with Physical Joystick Controller**

Since we will be using the Logitech F710 controller, plug the USB nano receiver into your laptop, and open the web controller. Select the ‘Gamepad’ option, press the start button on the controller (may have to click mode too), and start driving. Make sure the slider on the back of the controller is set to ‘X’.

**Training Autopilot**

**Collect Data**

1. Practice driving around the track a couple times.
2. When you're confident you can drive 10 laps without mistake, restart the python mange.py process to create a new tub session. Press Start Recording if using web controller. The joystick will auto record with any non-zero throttle.
3. If you crash or run off the track press Stop Car immediately to stop recording.
4. After you've collected 10-20 laps of good data (5-20k images) you can stop your car with Ctrl-c in the SSH session for your car.
5. The data you've collected is in the ‘data’ folder in the most recent tub folder.

**Transfer Data**

Since the Raspberry Pi is not very powerful, we need to transfer the data to a personal computer, such as a laptop, to train. Transferring can be done two ways: manual with a flash drive, or by using SSH. We have found manual to work better, but rsync is always available.

Manual Version: Plug a monitor and keyboard into the pi, then use a flash drive to copy the tub data from the Pi’s ‘data’ folder to the drive, then from the drive to the host PC’s ‘data’ folder

SSH Version: In a new SSH terminal session on your host PC use rsync to copy your cars folder from the Raspberry Pi. You can transfer the data by typing into the current SSH session terminal:

rsync -r pi@<your\_pi\_ip\_address>:~/mycar/data/ ~/mycar/data/

or

rsync -rv --show-progress --partial pi@<your\_pi\_ip\_address>:~/mycar/data/ ~/mycar/data/

**Training Data**

Manual Version

1. After copying the data, open the Anaconda terminal
   * Make sure anaconda and donkey software is downloaded on the PC
2. In the Anaconda terminal, type conda activate donkey followed by python ~/mycar/manage.py train -–tub<your tub> --model ~/mycar/models/Pilot1.h5
   * May have to specify no tub by using python ~/mycar/manage.py train -–model ./mycar/models/pilot.h5 if you get the “Usage:” error; just make sure that tub is the only one in the folder
   * Make sure to check your tubs since it can be picky about your tub location
3. It will train (may take 10-30 minutes); once done, copy the model file back to USB and to the pi’s ‘models’ folder – then use python ~/mycar/manage.py drive –-model ~/mycar/models/Pilot1.h5
4. In the web browser, make sure ‘Local pilot’ is selected and it should start driving itself.
5. Make sure whatever battery you train with, you use during autonomous

SSH Version

1. Once transferred, train the data by typing python ~/mycar/manage.py train --tub <tub folder names comma separated> --model ./models/mypilot.h5 into the current SSH session terminal
2. Copy data back to car by typing rsync -rv --show-progress --partial ~/mycar/models/ pi@<your\_ip\_address>:~/mycar/models/
3. Then you can start driving by typing python ~/mycar/manage.py drive --model ~/mycar/models/mypilot.h5