The Duckietown Book

The last version of this book and other documents are available at the URL
http://book.duckietown.org/

TABLE OF CONTENTS

Part 1
The Duckietown project

Chapter 1
What is Duckietown?

1.1. Goals and objectives

Duckietown is a robotics educations and outreach effort.

The most tangible goal of the project is to provide a low-cost educational platform for learning autonomy, consisting of the Duckiebots, an autonomous robot, and the Duckietowns, the infrastructure in which the Duckiebots navigates.

However, we focus on the *learning experience* as a whole, by providing a set of modules teaching plans and other guides, as well as a curated role-play experience.

We have two targets:

1. For **instructors**, we want to create a "class-in-a-box" that allows to offer a modern and engaging learning experience. Currently, this is feasible at the advanced undergraduate and graduate level, though in the future we would like to present the platform as multi-grade experiences.
2. For **self-guided learners**, we want to create a "self-learning experience", that allows to go from zero knowledge of robotics to graduate-level understanding.

In addition, the Duckietown platform has been used as a research platform.

1.2. Results obtained so far

While we are at the early phases of the project, many people have been used the materials in the past year.

1.3. Why the duckies?

Why the duckies?

Compared to other educational robotics projects, the presence of the duckies is what makes this project stand out. Why the duckies?

We want to present robotics in an accessible and friendly way.

1.4. Learn about the platform

The best way to get a sense of how the platform looks is to watch these videos. They show off the capabilities of the platform.

This video is part of the Red Hat documentary:

1.5. Learn about the educational experience

These papers present a more formal description of the technical side of the projecrt as well as the educational side.

This paper [1] describes the course design for Duckietown: learning objectives, teaching methods, etc.

This video is a Duckumentary about the first version of the class, during Spring 2016. The Duckumentary was shot by Chris Welch.

1.6. Learn about the platform

The paper [2] describes the Duckiebot and its software. With 29 authors, we made the record for a robotics conference.

Chapter 2
Duckietown history and future

2.1. The beginnings of Duckietown

Duckietown started as an MIT class during Spring 2016.

2.2. Duckietown around the world

1) Duckietown High School

2.3. Coming up

In 2017, the class will be offered contemporaneously at:

- ETH Zurich
- University of Montreal
- University of Chicago

as well as:

CHAPTER 3

First steps

3.1. How to get started

If you are an instructor, please jump to Chapter 4.

If you are a self-guided learner, please jump to Chapter 5.

If you are a company, and interested in working with Duckietown, please jump to Chapter 6.

3.2. How to keep in touch

3.3. How to contribute

CHAPTER 4
Duckietown for instructors

CHAPTER 5

Duckietown for self-guided learners

Chapter 6
Introduction for companies

Chapter 7
Frequently Asked Questions

7.1. General questions

What is Duckietown?

Duckietown is a low-cost educational and research platform.

Is Duckietown free to use?

Yes. All materials are released according to an open source license.

Is everything ready?

Not quite! Please sign up to our mailing list to get notified when things are a bit more ready.

How can I start?

See the next section, Getting started.

How can I help?

If you would like to help actively, please email duckietown@mit.edu.

7.2. FAQ by students / independent learners

I want to build my own Duckiebot. How do I get started?

7.3. FAQ by instructors

How large a class can it be? I teach large classes.

What is the budget for the robot?

I want to teach a Duckietown class. How do I get started?

Please get in touch with us at duckietown@mit.edu. We will be happy to get you started and sign you up to the Duckietown instructors mailing list.

7.4. FAQ by researchers

Part 2
Setup instructions

CHAPTER 8

Acquiring the parts for the Duckiebot

The trip begins with acquiring the parts. Here, we provide a link to all bits and pieces that are needed to build a Duckiebot and the Duckietowns.

In general, keep in mind that:

- The links might expire, or the prices might vary.
- In general, substitutions are OK for the mechanical components, and not OK for all the rest.

8.1. Bill of materials

Chassis	USD xxx
Camera	USD xxx
Raspberry PI 3	USD 35
Total for minimum configuration	USD ??
Total for fancy configuration	USD ??

8.2. Chassis

We selected the Magician Chassis as the basic chassis for the robot (Figure 1).

We chose it because it has a double-decker configuration, and so we can put the battery in the lower part.

The price for this in the US is about USD 15-30.

Figure 1. The Magician Chassis

8.3. Raspberry PI 3

...

8.4. Camera

...

Chapter 9
Soldering boards

CHAPTER 10
Assembling the Duckiebot

Chapter 11
Reproducing the image

These are the instructions to reproduce the Ubuntu image that we use.

Please note that the image is already available, so you don't need to do this manually.

However, this documentation might be useful if you would like to port the software to a different distribution.

We organize this in three steps:

- Step 1: Downloading and loading the raw ubuntu image
- Step 2: Installation of ros and other dependencies and other configurations

11.1. Step 1: From blank to minimal setup

Resources necessaries:

- Internet connection to download the packages.
- A PC running any Linux with an SD card reader.
- Time: about 20 minutes.

Results:

- A baseline Ubuntu Mate 16.04.2 image with updated software.

1) Download and uncompress the Ubuntu Mate image

Download the image from the page https://ubuntu-mate.org/download/.

The file we are looking for is:

```
filename: ubuntu-mate-16.04.2-desktop-armhf-raspberry-pi.img.xz
    size: 1.2 GB
  SHA256: dc3afcad68a5de3ba683dc30d2093a3b5b3cd6b2c16c0b5de8d50fede78f75c2
```

Run the command `sha256sum` to make sure you have the right version:

```
laptop $ sha256sum ubuntu-mate-16.04.2-desktop-armhf-raspberry-pi.img.xz
dc3afcad68a5de3ba683dc30d2093a3b5b3cd6b2c16c0b5de8d50fede78f75c2
```

If the string does not correspond exactly, your download was corrupted. Delete the file and try again.

Then decompress using the command `xz`:

```
laptop $ xz -d ubuntu-mate-16.04.2-desktop-armhf-raspberry-pi.img.xz
```

2) Finding your device name for the SD card an unmount it

```
laptop $ df -h
```

Inspect the output for something like `/dev/mmcblk0`, you may see `/dev/mmcblk0pX` of a couple of similar entries for each partition on the card, where `X` is the partition number. If you don't see anything like that, take out the sd card and run the command again and see what disappeared.

Next unmount all the partitions associated with the device probably:

```
laptop $ sudo umount /dev/mmcblk0p1
laptop $ sudo umount /dev/mmcblk0p2
```

3) Burn the image to an SD card

Then burn to disc using the command `dd`:

```
laptop $ sudo dd of=DEVICE if=IMG status=progress bs=4M
```

where `IMG` is the `.img` file you unzipped, and `DEVICE` is the device that represents your SD card reader (NB without partitions. ie, `/dev/mmcblk0` not `/dev/mmcblk0pX`)

4) Verify that the SD card was created correctly

Remove the SD card and plug it in again in the laptop.

Ubuntu will mount two partitions, by the name of `PI_ROOT` and `PI_BOOT`.

5) Installation

Boot the disk in the Raspberry PI.

I chose the following options:

```
language: English
username: ubuntu
password: ubuntu
hostname: duckiebot
```

(LP: I also chose the option to log in automatically)

Then I rebooteed.

6) Update installed software

The WiFi was connected to airport network `duckietown` with password `quackquack`.

Afterwards I upgraded all the software preinstalled with these commands:

```
duckiebot $ sudo apt update
duckiebot $ sudo apt dist-upgrade
```

(Expect dist-upgrade to take quite a long time - e.g. 2hrs)

11.2. Part 2: Dependencies and Configurations

1) Raspi Config

The raspi is not sshable by default, the camera is disabled, and the I2C bus is disabled. We need to fix those things:

```
duckiebot $ sudo raspi-config
 choose 3. Interfacing Options,
 P2 SSH
 change no to yes
 back
 P1 camera
 change no to yes
 back
 P3 I2C
 change no to yes
 back
 finish
```

things to be installed:

2) ROS

first commands are copied from http://wiki.ros.org/kinetic/Installation/Ubuntu

```
duckiebot $ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/so
duckiebot $ sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key 421C365BD9FF1F71781
duckiebot $ sudo apt update
duckiebot $ sudo apt install ros-kinetic-desktop-full
duckiebot $ sudo apt install
ros-$ROS_DISTRO-{tf-conversions,cv-bridge,image-transport,camera-info-manager,theora-image-transport,joy,ima
```

3) Install packages

```
duckiebot $ sudo apt install -y emacs vim byobu build-essential git python python-dev ipython
libblas-dev liblapack-dev libatlas-base-dev gfortran libyaml-cpp-dev python-sklearn i2c-tools
python-smbus
duckiebot $ sudo pip install scipy --upgrade
```

(may need to do the following but might be done already through raspi-config):

```
duckiebot $ sudo usermod -a -G i2c ubuntu
duckibot $ sudo udevadm trigger
```

4) Optional user preferences

To automatically boot into byobu:

```
duckiebot $ byobu-enable
```

This can be disabled with byobu-disable.

5) Wireless configuration

There are the two key to edit files:

- `/etc/network/interfaces` should look like this:

```
 interfaces(5) file used by ifup(8) and ifdown(8)
# Include files from /etc/network/interfaces.d:
#source-directory /etc/network/interfaces.d

auto wlan0

# The loopback network interface
auto lo
iface lo inet loopback

# Wireless network interface
allow-hotplug wlan0
iface wlan0 inet dhcp
wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf
iface default inet dhcp
```

- `/etc/wpa_supplicant/wpa_supplicant.conf` should look like this:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

network={
ssid="duckietown"
psk="quackquack"
proto=RSN
key_mgmt=WPA-PSK
pairwise=CCMP
auth_alg=OPEN
}
network={
   key_mgmt=NONE
}
```

6) SSH config

Add `.authorized_keys` in the image so that we can all do passwordless ssh.

Create the `.authorized_keys` files.

On the PI, download the official key:

```
duckiebot $ cd
duckiebot $ mkdir -p .ssh
duckiebot $ chmod g-rwx,o-rwx .ssh
duckiebot $ wget -O .ssh/authorized_keys https://www.dropbox.com/s/pxyou3qy1p8m4d0/
duckietown_key1.pub?dl=1
```

7) Swap Space

Do the following:

Create an empty file using the dd (device-to-device copy) command:

```
duckiebot $ sudo dd if=/dev/zero of=/swap0 bs=1M count=512
```

This is for a 512 MB swap space.

Format the file for use as swap:

```
duckiebot $ sudo mkswap /swap0
```

Add the swap file to the system configuration:

```
duckiebot $ sudo vi /etc/fstab
```

Add `/swap0 swap swap` at the bottom

Activate the swap space:

```
duckiebot $ sudo swapon -a
```

(Can probably do something similar through `raspi-config`.)

Chapter 12
Reproducing the image for laptops

These are Andrea's notes for the laptops (Duckietops).
The image was Ubuntu Mate 16.04.2.
I chose the following options:

```
language: English
username: ubuntu
password: ubuntu
hostname: duckietop
```

If you choose a different username, you will need to change all the commands later.

12.1. Useful software

Use `etckeeper` to keep track of the configuration in `/etc`:

```
laptop $ sudo apt install etckeeper
```

Install `ssh` to login remotely:

```
laptop $ sudo apt install ssh
```

Use `byobu`:

```
laptop $ sudo apt install byobu
```

Use `vim`:

```
laptop $ sudo apt install vim
```

Use `htop` to monitor CPU usage:

```
laptop $ sudo apt install htop
```

Additional utilities for `git`:

```
laptop $ sudo apt install git-extras
```

Other utilities:

```
laptop $ sudo apt install avahi-utils ecryptfs-utils
```

1) Redshift

This is Flux for Linux. It is an accessibility/lab safety issue: bright screens damage eyes and perturb sleep [3].

Install redshift and run it.

```
laptop $ sudo apt install redshift-gtk
```

Set to "autostart" from the icon.

12.2. Installation of the documentation system

Next, the docs system was installed using the documentation in Section 32.4.

Chapter 13
Linux commands reference

This is a reference for all commands mentioned in the documentation.

13.1. SSH

(To write)

```
$ sudo apt install ssh
```

1) Can you do the following?

- Log in the duckiebot from the laptop;
- Log in the duckiebot from the laptop, without using password;

13.2. SCP

1) Can you do the following?

- Copy a file from the duckiebot to the laptop;
- Copy a file from the laptop to the duckiebot;

13.3. Byobu

(To write)

```
$ sudo apt install byobu
```

13.4. VIM

The editor to choose is VI, or more precisely, `vim` (improved vi).

Install like this:

```
$ sudo apt install vim
```

Documentation:

- A VIM tutorial

13.5. (`htop`)

Use `htop` to monitor CPU usage:

```
sudo apt install htop
```

13.6. Git and related commands

Additional utilities for `git`:

```
$ sudo apt install git-extras
```

This adds `git-ignore`.

(To write)

13.7. Python virtual environments

Install using:

```
$ sudo apt install virtualenv
```

Chapter 14
Burn the Duckiebot image

Prerequisites:

- A laptop with an SD card reader/writer

Result:

- An SD card with image v2.0.

CHAPTER 15
Duckiebot Initialization

Prerequisites:

- A complete Duckiebot in configuration C0.
- An SD card with image v2.0.

Result:

- A Duckiebot that is ready to use

15.1. Login and update base system

If you are in an environment where there is a wireless network with SSID "duckietown" and pwd quackquack then good news! your robot is (should be) already connected to the network. If not you can either connect through an wired LAN or worst case connect a monitor and keyboard.

to ssh into your robot with do:

```
laptop $ ssh ubuntu@duckiebot.local
```

pwd ubuntu

15.2. Learn to love Byobu

By default your robot terminal boots into *byobu*.

Yes, you need to learn to use byobu, unless you know an equivalent program. You will save much time later.

Byobu is "GNU screen" with fancy configuration. Please learn about Byobu here: http://byobu.co/

Quick commands reference, using function keys:

F2: open a new terminal

F3/F4: switch among the terminals

Ctrl-F6: close current terminal

Using control sequences:

```
ctrl-A then C: creates new terminal
ctrl-A then number: switches to terminal
```

To quit a terminal:

```
duckiebot $ exit
```

15.3. Wireless network configuration

The duckiebot is configured by default to connect to a wireless network with SSID "duckietown". If that is not your SSID then you will need to change/add a new clause

to the */etc/wpa_supplicant/wpa_supplicant.conf* file:

```
duckiebot $ sudo vi /etc/wpa_supplicant/wpa_supplicant.conf
```

you will see a block:

```
network={
 ssid="duckietown"
 scan_ssid=1
 psk="quackquack"
 priority=10
}
```

either modify that one or add a new one with your ssid and password (assuming you have a roughly similar wireless network setup - if not then you might need to change some of the other attributes)

15.4. Update basic system

Use:

```
duckiebot $ sudo apt update
duckiebot $ sudo apt dist-upgrade
```

15.5. Camera Test

You can test the camera right away, without setting up ROS.

Use the command:

```
duckiebot $ raspistill -t 1 -o out.jpg
```

Then download out.jpg using scp:

```
laptop $ scp ubuntu@duckiebot.local: /out.jpg out.jpg
```

15.6. Do not change the default shell

If you know what you are doing, you are welcome to install and use additional shells (such as zsh), but please keep **bash** as be the default shell. This is important for some scripts.

(For the record, our favorite shell is zsh with oh-my-zsh.)

15.7. Set hostname

Choose a name for your robot. This is a simple string that will always appear lowercase.

Edit /etc/hostname and put "ROBOT_NAME" instead of "duckiebot".

```
duckiebot $ sudo vi /etc/hostname
```

Also edit /etc/hosts and put "ROBOT_NAME" instead of "duckiebot":

```
duckiebot $ sudo vi /etc/hosts
```

Note: the command “sudo hostname duckiebot” is not enough. The change will not persist. You need to go through the steps above. `` NEVER ADD HOSTNAMES IN /etc/hosts (e.g. duckiebot.local)

Then reboot:

```
$ sudo reboot
```

When you reboot, you should see your new hostname:

```
Ubuntu 16.04.2 LTS
$ROBOT_NAME$ login:
```

Chapter 16
Setup the laptops

16.1. Setup an Ubuntu laptop to ssh to the duckiebot

Connect your laptop to the same network as the robot.

From your laptop, now you should be able to ping the duckiebot:

```
laptop $ ping ROBOTNAME.local
```

(or if you have not changed your hostname then ROBOTNAME=duckiebot)

Do not continue if you cannot do this successfully.

Protip: In general, if you find yourself:

- *typing an IP*
- *typing a password*
- *typing "ssh" more than once*
- *using a screen / USB keyboard *

*it means you should learn more about Linux and networks, and you are setting yourself up for failure. Yes, you "can do without", but with an additional 30 seconds of your time. The 30 seconds you are not saving every time are the difference between being productive roboticists and going crazy. Really, it is impossible to do robotics when you have to think about IPs and passwords... *

Verify that you can ssh to the PI:

```
laptop $ ssh ubuntu@duckiebot.local
```

Say "yes" if you get asked whether you want to add it to a list of known hosts.

Offending key error: If you get something like this:

Warning: the ECDSA host key for ... differs from the key for the IP address '10.0.1.17'

```
Offending key for IP in /Users/<user>/.ssh/known_hosts:<line>
```

then remove line <line> in /.ssh/known_hosts

Now, let's set up **passwordless ssh.**

On the laptop, create the .ssh directory:

```
laptop $ mkdir -p  /.ssh
```

Install the duckietown_key1 by downloading it:

```
laptop $ wget -O  /.ssh/duckietown_key1 "https://www.dropbox.com/s/q23qptu01u7ur3y/
duckietown_key1?dl=1"
```

Changing Key Permission

Edit the permission of the file to make ssh happy. The key file must not be readable or writable from other users or groups.

```
laptop $

chmod 600  /.ssh/duckietown_key1

chmod 600  /.ssh/duckietown_key1
```

Regenerate the public key according to:za

```
laptop $ ssh-keygen -f  /.ssh/duckietown_key1 -y >  /.ssh/duckietown_key1.pub
```

Troubleshooting

If there are issues such as “scheme missing” and the file duckietown_key1 does not exist in /.ssh/ folder but instead downloaded a file named duckietown_key1?dl=1 in the current folder simply rename duckietown_key1?dl=1 to duckietown_key1 and copy it over to the directory /.ssh/.

To move the mislabeled file:

```
laptop $ mv ”duckietown_key1?dl=1”   /.ssh/duckietown_key1
```

On the laptop, now edit /.ssh/config:

```
laptop $ nano  /.ssh/config
```

and add the following lines:

```
Host **ROBOTNAME**
Hostname ROBOTNAME.local
User ubuntu
IdentityFile  /.ssh/duckietown_key1
HostKeyAlgorithms ssh-rsa
```

Now you should be able to ssh passwordlessly from your laptop:

```
laptop $ ssh **duckiebot**
```

This should succeed. Do not continue unless it does.

Note that you can actually auto-complete with tab after ssh.

Chapter 17
Setup Github access

This chapter describes how to create a Github account and setup SSH on the robot and on the laptop.

17.1. Create an SSH key on your machine

This step needs to be repeated twice: once on the Duckiebot, and once on your laptop. You will thus create two private/public key pairs.

An SSH key can be generated with the command:

```
laptop / duckiebot $ ssh-keygen -h
```

The program will prompt you for the filename on which to save the file.

When you do this on the robot, use this name convention:

```
/home/ubuntu/.ssh/username@robot name
```

Of course, substitute your username and hostname.

When doing it on the laptop, use the file name:

```
/home/username/.ssh/username@laptop name
```

The output will look something like this:

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/username/.ssh/id_rsa):
/home/username/.ssh/username@hostname
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/username/.ssh/username@hostname
Your public key has been saved in /home/username/.ssh/username@hostname.pub
The key fingerprint is:
XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX username@hostname
The key's randomart image is:
+--[ RSA 2048]----+
|            .    |
|       o   o  .  |
|      o = o  . o |
|       B . .  * o|
|        S o    O |
|         o o  . E|
|          o o  o |
|           o  +  |
|            .. . |
+-----------------+
```

Note that the program created two files: a file that contains the private key in `/home/username/.ssh/username@hostname` and a file that contains the public key with extension `.pub` called `/home/username/.ssh/username@hostname.pub`.

17.2. Create a Github account

Our example account is the following:

```
Github name: greta-p
E-mail: greta-p@duckietown.com
```

Create a Github account (Figure 2).

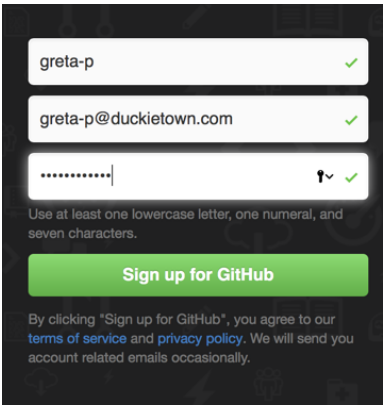

Figure 2

Go to your inbox and verify the email.

Go to settings (Figure 3).

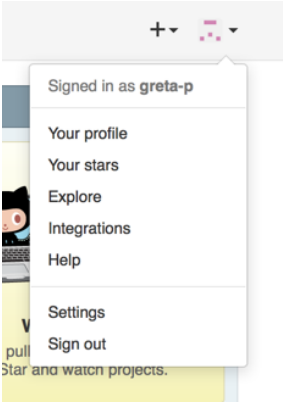

Figure 3

Add the two public SSH keys that you created (on the laptop and on the robot).

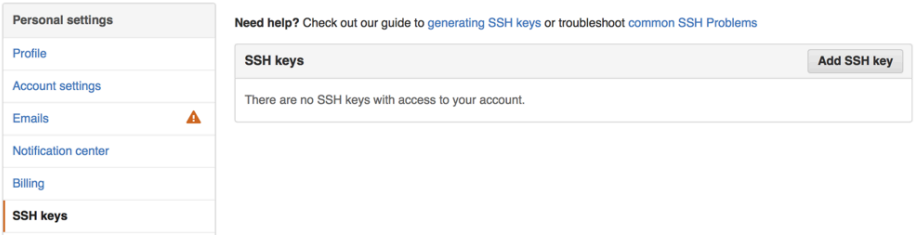

Figure 4

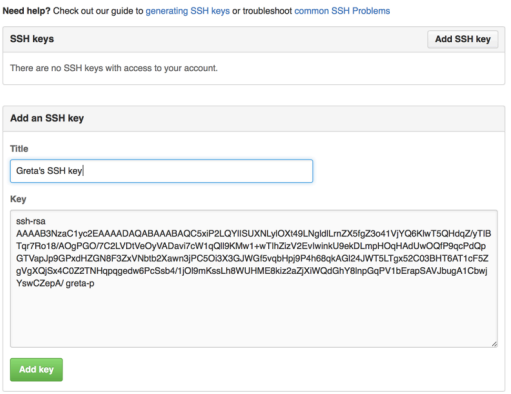

Figure 5

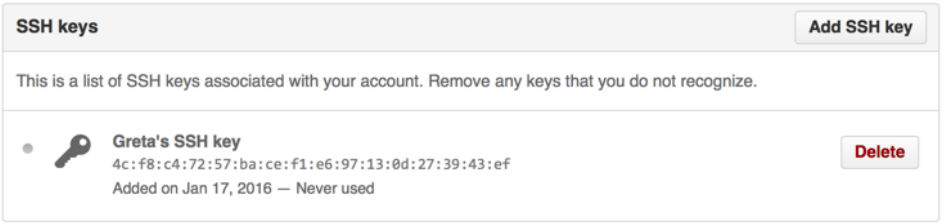

Figure 6

17.3. Creation of SSH config file for your machine

This step should be performed twice, once on the laptop and once on the robot.

First, create the file `/.ssh/config`:

```
laptop / duckiebot $ touch  /.ssh/config
```

Add a line containing "`IdentityFile PRIVATE_KEY_FILE`" (using the filename for the private key).

Check that the config file is correct:

```
laptop / duckiebot $ cat  /.ssh/config
IdentityFile  /.ssh/PRIVATE_KEY_FILE
```

To check that all of this works, use the command `ssh -T git@github.com`. The command tries to connect to Github using the private keys that you specified:

```
laptop / duckiebot $ ssh -T git@github.com
Warning: Permanently added the RSA host key for IP address '192.30.252.128' to the list of
known hosts.
Hi USERNAME! You've successfully authenticated, but GitHub does not provide shell access.
```

If you don't see the greeting, stop.

Repeat what you just did for the Duckiebot on the laptop as well, making sure to change the name of the file containing the private key.

17.4. Setting up global configurations for Git

On your laptop, set up git, with the following commands:

```
laptop / duckiebot $ git config --global user.email "<email>"
laptop / duckiebot $ git config --global user.name "<name>"
```

The email will be public in our repository's history.

Also do this, and it doesn't matter if you don't know what it is:

```
laptop / duckiebot $ git config --global push.default simple
```

Chapter 18
Software setup and RC remote control

18.1. Clone the Duckietown repository

You should have already followed the steps in Setup Step 1.9 - Github basics.

```
duckiebot $ git@github.com:duckietown/Software.git duckietown
```

Troubleshooting:

- *For this to succeed you should have a Github account sudo ntpdate -u us.pool.ntp.org already set up.*
- *if it fails with weird errors, probably the time is not set up correctly. Use ntpdate as above.*

18.2. Set up ROS environment on the Duckiebot

```
duckiebot $ cd  /duckietown
```

Now we are ready to make the workspace. First you need to source the baseline ROS indigo environment:

```
duckiebot $ source /opt/ros/kinetic/setup.bash
```

Then, build the workspace (you have to be under the catkin_ws folder to invoke catkin_make)

```
duckiebot $ make build
```

18.3. Add your vehicle to the machines file

On the robot edit the file

```
duckiebot $ vi  /duckietown/catkin_ws/src/duckietown/machines

<launch>
    <arg name="env_script_path" default=" /duckietown/environment.sh"/>
    <arg name="env_script_path" default=" /duckietown/environment.sh"/>
        <machine name="megaman" address="megaman.local" user="ubuntu" env-loader="$(arg
env_script_path)"/>
    <machine name="megaman" address="megaman.local" user="ubuntu" env-loader="$(arg
env_script_path)"/>
        ...
    ...
</launch>
```

Now, duplicate a <machine ... /&rt; line between <launch&rt; and </launch&rt;, and replace the name and address string with the name of your vehicle. In this example, the additional line to add is

```
<machine name="duckiebot" address="duckiebot.local" user="ubuntu" env-loader="$(arg
env_script_path)"/>
```

commit and push the new machines file.

18.4. Run the joystick demo

don't have a joystick? Move to

Plug the joystick to one of the usb port on the RasPi.

SSH into your PI. Go to the duckietown folder and invoke the following scripts:

```
duckiebot $ cd  /duckietown
duckiebot $ source environment.sh
duckiebot $ source set_ros_master.sh
```

The `environment.sh` setups the ROS environment at the terminal (so you can use commands like rosrun and roslaunch). The `set_ros_master.sh` by default sets the PI as its own rosmaster.

Now make sure the motor shield is connected.

Run the command:

```
duckiebot $ roslaunch duckietown joystick.launch veh:=duckiebot
```

If there is no "red" output in the command line then pushing the left joystick knob controls throttle - right controls steering.

This is the expected result of the commands:

left joystick up = forward

left joystick down = backward

right joystick left = turn left (positive theta)

right joystick right = turn right (negative theta)

It is possible you will have to unplug and replug the joystick or just push lots of buttons on your joystick until it wakes up. Also make sure that the mode switch on the top of your joystick is set to "X" not "D".

Close the program using Ctrl-C.

Troubleshooting - robot moves weirdly (forward instead of backward): Either the cables or the motors are inverted. Please refer to the assembly guide for pictures of the correct connections.

Troubleshooting - left joystick does not work: If the green light on the right to the "mode" button is on, click the "mode" button to turn the light off. The "mode" button toggles between left joystick or the cross on the left.

Troubleshooting - robot does not move: The joy_mapper_test.launch assumes that the joystick is at /dev/input/js0. To make sure that the joystick is there, you can do

```
duckiebot $ ls /dev/input/
```

and check if there is a `js0` on the list.

To test whether or not the joystick itself is working properly (without ROS), you can do

```
duckiebot $ jstest /dev/input/js0
```

Move the joysticks and push the buttons and check the printouts.

Troubleshooting - robot moves very weirdly: Check that the joystick has the switch set to the position "x". And the mode light should be off.

shutting down the robot:

```
duckiebot $ sudo shutdown -h now
```

then physically disconnect power cables.

Chapter 19
Proper shutdown procedure for the PI

Ctrl-C in the terminal of joystick.launch when you're done.

To shutdown: DO NOT DISCONNECT THE POWER - the system might get corrupted. Instead, issue the following command:

```
duckiebot $ sudo shutdown -h now
```

and then wait 30 seconds before disconnecting the power.

Also: Disconnect the battery end of the cable, not the one close to the PI, because you might damage it.

Chapter 20
Joystick plus camera output

Chapter 21
Wheel calibration

Chapter 22
Camera calibration

CHAPTER 23

Taking a log

Chapter 24
LED Setup

Part 3
Duckietowns

Chapter 25
Acquiring the parts

Chapter 26
Traffic Light Assembly

Chapter 27
The Duckietown specification

27.1. Topology

1) Topology constraints

27.2. Signs placement

PART 4
Developing software

This part is about how to develop software for the Duckiebot.

CHAPTER 28
Create a ROS package

Chapter 29
Integrate package in the architecture

Part 5
Modeling

TODO:

Chapter 30
Kinematics of Duckiebot

TODO:

Part 6
How to contribute

Chapter 31
Accounts

31.1. Complete list of accounts

Currently, Duckietown has the following accounts:

- Github: for source code, and issue tracking;
- Slack: a forum for wide communication;
- Twist: to be used for instructors coordination;
- Google Drive: to be used for instructors coordination, maintaining TODOs, etc;
- Dropbox Folders (part of Andrea's personal accounts): to be abandoned;
- Vimeo, for storing the videos;
- The `duckietown-teaching` mailing list, for low-rate communication with instructors;
- We also have a list of addresses, of people signed up on the website, that we didn't use yet;
- The Facebook page.

31.2. For Fall 2017

As a student in Fall 2017, these are the accounts that you need:

- A Github account and membership in the Duckietown organization.
- A Slack account, for team discussion and organization.

As an instructor/TA for the Fall 2017 class, in addition to the accounts above, these are the accounts that you need:

- Twist: for class organization (such as TAs, logistics);
- Google Docs, used to maintain TODOs.

31.3. For other contributors

If you are an international contributor:

- Sign up on Slack, to keep up with the project.
- (optional) Get Github permissions if you do frequent updates to the repositories.

Chapter 32
Contributing to the documentation

32.1. Where the documentation is

All the documentation is in the repository `duckietown/duckuments`.

The documentation is written as a series of small files in Markdown format.

It is then processed by a series of scripts to create this output:

- a publication-quality PDF;
- an online HTML version, split in multiple pages and with comments boxes.

32.2. Editing links

The simplest way to contribute to the documentation is to click any of the "✎" icons next to the headers.

They link to the "edit" page in Github. There, one can make and commit the edits in only a few seconds.

32.3. Comments

In the multiple-page version, each page also includes a comment box powered by a service called Disqus. This provides a way for people to write comments with a very low barrier. (We would periodically remove the comments.)

32.4. Installing dependencies for compiling the documentation

Let `DUCKUMENTS` be the base directory for the documentation.

Download the `duckuments` repo in that directory:

```
$ git clone git@github.com:duckietown/duckuments.git $DUCKUMENTS
```

Cd into directory:

```
$ cd $DUCKUMENTS
```

1) Setup a virtual environment

On Ubuntu 16.04, create a virtual environment usign `virtualenv` (`sudo apt install virtualenv` if needed):

```
$ virtualenv --system-site-packages deploy
```

In other distributions you might need to use `venv`:

```
$ venv deploy
```

Activate the virtual environment:

```
$ source $DUCKUMENTS/deploy/bin/activate
```

Install some dependencies:

```
$ sudo apt install libxml2-dev libxslt1-dev
$ sudo apt install libffi6 libffi-dev
$ sudo apt install python-dev python-numpy python-matplotlib
```

Clone the `mcdp` external repository:

```
$ cd $DUCKUMENTS
$ git clone -b duckuments git@github.com:AndreaCensi/mcdp.git
```

Install it and its dependencies:

```
$ cd $DUCKUMENTS/mcdp
$ python setup.py develop
```

(If you get a permission error here, it means you have not properly activated the virtualenv)

Depending on your system, you might need to install these other dependencies: (It should not be necessary on Ubuntu 16 given the `apt` commands above.)

```
$ cd $DUCKUMENTS
$ pip install numpy matplotlib
```

Ensure the latest version (>6) of `nodejs` is installed.

Run:

```
$ nodejs --version
6.xx
```

If the version is 4 or less, remove `nodejs`:

```
$ sudo apt remove nodejs
```

Install `nodejs` using the instructions at this page.

Next, install the necessary Javascript libraries using `npm`:

```
$ cd $DUCKUMENTS
$ npm install MathJax-node jsdom@9.3 less
```

Install PrinceXML from this page.

Download STIX fonts from this site.

Unzip and copy the ttf to `/.fonts`:

```
$ cp -R STIXv2.0.0  /.fonts
```

and then rebuild the font cache using:

```
$ fc-cache -fv
```

32.5. Troubleshooting installation problems

1) Installing (nodejs) packages

The only pain point in the installation procedure has been the installation of `nodejs` packages using `npm`. For some reason, they cannot be installated globally (`npm install -g`).

Do not use `sudo` for installation. It will cause problems.

If you use `sudo`, you probably have to delete a bunch of directories, such as: `RBROOT/node_modules`, `/.npm`, and `/.node_modules`, if they exist.

32.6. Compiling the documentation

Make sure you have deployed and activated the virtual environment. Then:

```
$ cd $DUCKUMENTS
$ make duckuments-dist
```

This creates the directory `duckuments-dist`, which contains another checked out copy of the repository, but with the branch `gh-pages`, which is the branch that is published by Github using the "Github Pages" mechanism.

At this point, please make sure that you have these two `.git` folders:

```
$DUCKUMENTS/.git
$DUCKUMENTS/duckuments-dist/.git
```

To compile the docs, go in the `DUCKUMENTS` directory and run `make compile`:

```
$ cd $DUCKUMENTS
$ make clean compile
```

If you want to do incremental compilation, you can omit the `clean` and just use:

```
$ make compile
```

To compile the PDF, use:

```
$ make compile-pdf
```

This creates the following files:

- `duckuments-dist/master/duckiebook.html` is a single-page HTML of everything.
- `duckuments-dist/master/duckiebook.pdf` is the PDF version.
- `duckuments-dist/master/duckiebook/index.html` is the first page of the version with each chapter on a different page.

32.7. Deploying the documentation

This is now done by a bot.

To deploy the documentation, jump into the `DUCKUMENTS/duckuments-dist` directory.

Run the command `git branch`. If the out does not say that you are on the branch `gh-pages`, then one of the steps before was done incorrectly.

```
$ cd $DUCKUMENTS/duckuments-dist
$ git branch
...
* gh-pages
...
```

Now, after triple checking that you are in the `gh-pages` branch, you can use `git status` to see the files that were added or modified, and simply use `git add`, `git commit` and `git push` to push the files to Github.

CHAPTER 33

Features of the documentation writing system

33.1. Embedded LaTeX

You can use LaTeX math, environment, and references. For example, take a look at

$$x^2 = \int_0^t f(\tau)\ \text{d}\tau$$

or refer to Proposition 1.

Proposition 1. (Proposition example) This is an example proposition: $2x = x + x$.

The above was written as in Figure 7.

```
You can use $\LaTeX$ math, environment, and references.
For example, take a look at

\[
    x^2 = \int_0^t f(\tau)\ \text{d}\tau
\]

or refer to [](#prop:example).

\begin{proposition}[Proposition example]\label{prop:example}
This is an example proposition: $2x = x + x$.
\end{proposition}
```

Figure 7. Use of LaTeX code.

33.2. Other interesting features

33.3. Limitations

There are some limitations:

- Please use the string `&#36;` to write the dollar symbol $, otherwise it gets confused with LaTeX math materials. Also notice that you should probably use “USD” to refer to U.S. dollars

Chapter 34
The Fall 2017 Global Experience

This is the first time that a class is taught jointly across 3 continents!

There are 4 universities involved in the joint teaching for the term:

- ETH Zürich (ETHZ), with instructors Emilio Frazzoli, Andrea Censi, Jacopo Tani.
- University of Montreal (UdeM), with instructor Liam Paull.
- TTI Chicago (TTI), with instructor Matthew Walter.
- National C T University (NCTU), with instructor Nick Wang.

This part of the Duckiebook describes all the information that is needed by the students of the three institutions.

34.1. General remarks

1) The first rule of Duckietown

The first rule of Duckietown is: you don't talk about Duckietown *using email*!

Instead, we use a communication platform called Slack.

There is one exception: inquiries about "meta" level issues, such as course enrollment and other official bureacratic issues can be communicated via email.

2) The second rule of Duckietown

The second rule of Duckietown is: be kind and respectful, and have fun.

34.2. Synchronization between classes

At ETHZ, UdeM, TTI, the class will be more-or-less synchronized. The materials are the same; there is some slight variation in the ordering.

Moreover, there will be some common groups for the projects.

The NCTU class is undergraduate level. Students will learn slightly simplified materials. They will not collaborate directly with the classes.

34.3. Accounts for students

To participate in Duckietown, students must use two accounts: Slack and Github.

1) Slack

2) Github

34.4. Accounts for all instructors and TAs

There are two more accounts required for instructors and TAs

1) Twist

TODO:

2) Google docs

TODO:

In particular:

- This is the schedule: ...
- This is the calendar in which to annotate everything: ...

34.5. Other accounts for organizers

1) Duckietown-teaching

34.6. Additional information for ETH Zürich students

This section describes information specific for ETH Zürich students.

1) Website

All really important information, such as deadlines, is in the authoritative website:

2) Duckiebox distribution

3) Lab access

4) The local TAs

34.7. Additional information for UdeM students

34.8. Additional information for TTI students

34.9. Additional information for NCTU students

CHAPTER 35
Bibliography

[1] Jacopo Tani, Liam Paull, Maria Zuber, Daniela Rus, Jonathan How, John Leonard, and Andrea Censi. **Duckietown: an innovative way to teach autonomy.** In *EduRobotics 2016*. Athens, Greece, December 2016. pdf

[2] Liam Paull, Jacopo Tani, Heejin Ahn, Javier Alonso-Mora, Luca Carlone, Michal Cap, Yu Fan Chen, Changhyun Choi, Jeff Dusek, Daniel Hoehener, Shih-Yuan Liu, Michael Novitzky, Igor Franzoni Okuyama, Jason Pazis, Guy Rosman, Valerio Varricchio, Hsueh-Cheng Wang, Dmitry Yershov, Hang Zhao, Michael Benjamin, Christopher Carr, Maria Zuber, Sertac Karaman, Emilio Frazzoli, Domitilla Del Vecchio, Daniela Rus, Jonathan How, John Leonard, and Andrea Censi. **Duckietown: an open, inexpensive and flexible platform for autonomy education and research.** In *IEEE International Conference on Robotics and Automation (ICRA)*. Singapore, May 2017. pdf

[3] Tosini, G., Ferguson, I., Tsubota, K. . Effects of blue light on the circadian system and eye physiology. Molecular Vision, 22, 61–72, 2016 (online).