Welcome to the Duckietown (city) operation manual. Here you will find the instructions to get started and other useful information.

- Part A - Preliminaries: What you need to know before you start.
- Part B - Assembly: What you need to know to build a Duckietown.

Watchtowers, the eyes of Duckietown.

- Part C - Troubleshooting: What could go wrong, and how to fix it.
PART A
Preliminaries

Before operating Duckiebots in a Duckietown, there are few things to keep in mind.

- Unit A-1 - Signal layers: How is Duckietown structured?
- Unit A-2 - Appearance Specifications: How should every component of Duckietown look like?
- Unit A-3 - Duckietowns and Duckiebots: What are the minimal components in Duckietown that enable Duckiebot functionalities?
UNIT A-1

Signal layers

KNOWLEDGE AND ACTIVITY GRAPH

- **Requires:** Nothing.
- **Results:** Preliminary knowledge of Duckietown information layers.
- **Next:** The Duckietown appearance specifications.

From a functional perspective, Duckietown cities are an integral part of the robotic ecosystem we call Duckietown. They are designed to send information to the Duckiebots, so these can operate.

Duckietowns are *modular*. They are composed of fundamental building blocks that can be combined to create nearly arbitrary city landscapes. Moreover, many hardware components are the same as those used for the Duckiebots.

At a high level, Duckietown is built with two layers, the *floor* and *signals* layers.

### 1.1. Floor Layer

The floor layer is the substrate on which Duckiebots drive, i.e., the road. Regardless of the geometry of the roads (straight, curve, intersections), roads are made of two lanes; one for each direction of driving. Lanes are obtained by applying lane markings a black background. The lane markings need to adhere to the appearance specifications to be effective.

### 1.2. Signals Layer

The signals layer contains all the signs and other functional objects (e.g., traffic lights or watchtowers) that sit on top of the mats. Objects are functional when they enable some behavior for the Duckiebots. For example, traffic signs are functional because they inform Duckiebots at intersections where they are and what should they look out for to know when to drive on.

### 1.3. Non-functional elements

The citizens of Duckietown like their cities to be colorful and fun, and encourage all ef-
forts at adding non-functional components to the city. Non functional objects can still sit on the floor, e.g., decorative building, but make sure they don’t interfere with the signals or floor layers!

Moreover, although Duckiebot drivers all have their driving licenses and know to focus on the road, the background, i.e., whatever is in the room the Duckietown was assembled, matters too. More information in this regards can be found in the troubleshooting section.
UNIT A-2

Appearance Specifications

KNOWLEDGE AND ACTIVITY GRAPH

Requires: Nothing.
Results: Knowledge of the Duckietown appearance specifications
Next: Duckietowns and Duckiebots, or Duckietown assembly.

This document describes the Duckietown appearance specification. Specifications are a set of rules for which a functional system has been verified. This means that if these rules are followed while building a Duckietown, Duckiebots will (most probably!) work.

Any Duckietown not adhering to the rules described here cannot be considered a Duckietown, and may cause the Duckiebots operating within them to fail in unexpected ways.

Small perturbations to the appearance specifications might affect negatively the performance of Duckiebots, although most algorithms are robust to variations.

2.1. Layer 1 - The Floor Layer

The floor layer is made of interlocking black tiles. Each tile represents one road element: straight, curve, 3-way intersection, 4-way intersection and empty tile. The road elements are positioned in specific orders to create compliant Duckietowns. The road elements are shown in Figure 2.1, note that the left turn and right turn tiles are symmetric: one is the 90 degree rotation of the other.

Each tile is square and measures 61 x 61 cm (2 ft x 2 ft) from the outer edges of the interlocking dents. The thickness of the tiles is not as important as the surface roughness. The objective is having good grip between the Duckiebots and the road in order to minimize slipping of the wheels.

(a) DT17_tile_straight  (b) DT17_tile_curve_left  (c) DT17_tile_curve_right
Figure 2.1. The principal tile types in Duckietown

Figure 2.2. A 3 by 3 city loop (DT17_map_loop3)
The empty tiles can be of any color, although it is discouraged to use the same colors as the road markings (red, white and yellow).

For tiles to become road elements, we need to apply road markings. Road markings can be obtained through the application of tapes of different colors and sizes.

1) Tapes

There are 3 colors of tapes used in Duckietown: white, yellow, and red.
**White tape:**

**Proposition 1.** A Duckiebot on a road never collides with Duckiebots or other Duckietown elements if it never crosses or touches a white tape strip.

Here are some facts about the white tapes:
- White tapes must be solid (not dashed);
- The width of the white tape is roughly **4.8 cm** (1.88 inches);
- The white tape is always placed on the right hand side of a lane. We assume that the Duckiebots drive on the right hand side of the road.

---

For curved roads, the white lane marker is formed by five pieces of white tape, while the inner corner is formed by three pieces, placed according to the specifications in the image below, where the edge pieces are matched to adjacent straight or curved tiles (Figure 2.5).

---

![Figure 2.5. The specification for a curved road tile](image)

**Yellow tape:**

Here are some facts about the yellow tapes:
- Yellow tape must be dashed (not solid);
- Each piece should be **5 cm** long and placed with a **2.5 cm** gap between each piece;
• The width of the yellow tape is roughly 2.4cm (0.94 inches);
• The yellow tape is always placed on the left hand side of a lane, i.e., in the center of the road. We assume that the Duckiebots drive on the right hand side of the road.

Yellow tapes on curves: see curved road image (Figure 2.5) in white tape section. Pieces at tile edges should be in center of lane, piece at the middle of the curve should be approximately 20.5 cm from middle of inner center white piece of tape, with approximated circular arc in between.

*Red tape:*

Red tapes MAY only appear on **intersection** tiles. The width of the red tape must be the same as the white roll (roughly 4.8cm or 1.88 inches) and should cross the entire lane perpendicular to the road. The placement of red tape should always be **under** yellow and white tape, as shown, e.g., in Figure 2.3 or Figure 2.4.

A Duckiebot navigates Duckietown by a sequence of:

• Navigating one or more straight tiles until a red tape appears,
• Waiting for the coordination signal,
• Executing an intersection traversal,
• Re-localizing in a straight tile.

**Proposition 2.** If the Duckiebot stops before or ON the red strip, no collisions are possible.

2) **Topological Constraints During Map Construction**

Here are some topological rule constraints that must be met:

1. An intersection can NOT be adjacent to a curved road tile or another intersection tile.
2. Any two adjacent non-empty tiles must have a feasible path from one to the other of length two: if they are adjacent, they must be connected.

Some examples of **non-conforming** topologies are shown in Figure 2.6.

(a) Topology violates rule 2 since the bottom two curved tiles are adjacent but not connected
(b) Topology violates rule 1 since curved tiles are adjacent to intersection tiles

(c) Topology violates rule 2 since left-most tiles are adjacent but not connected

Figure 2.6. Some non-conforming Duckietown map topologies

Note: The tile types described here are experimental. Use at your own risk!

A parking lot is a place for Duckiebots to go when they are tired and need a rest. A parking lot introduces three additional tile types:

1. **Parking lot entry tile**: This is similar to a straight tile except with a red stop in the middle. The parking lot sign (Figure 2.8o - parking) will be visible from this stop line.

2. **Parking spot tiles**:

3. **Parking spot access tiles**:

The following are the rules for a conforming parking lot:

1. One “parking spot” has size one tile.

2. From each parking spot, there is a path to go to the parking lot entry tile that does not intersect any other parking spot. (i.e. when a Duckiebot is parked, nobody will disturb it).

3. From any position in any parking spot, a Duckiebot can see at least two orthogonal lines or a sign with an April tag.

Note: The tile type described here is experimental. Use at your own risk!

A “launch tile” is used to introduce a new Duckiebot into a Duckietown in a controllable way. The launch file should be placed adjacent to a turn tile so that a Duckiebot may “merge” into Duckietown once the initialization procedure is complete. A “yield” sign should be visible from the launch tile.

### 2.2. Layer 2 - Signage and Lights

Generally, it is advisable to adhere signal layer elements to the tiles with double-sided tape. **Under no circumstances should any tape be obscured by the base of the stands.** At
least a 0.5 cm free (black) space should separate any line from a signal layer elements’ base.

2.3. Traffic Signs

Traffic signage in Duckietown is obtained through the union of a traffic sign and an AprilTag, as shown in Figure 2.7:

![Traffic sign and AprilTag](image)

We call the symbol above *traffic sign*, while the code below is an AprilTag.

Figure 2.7. A traffic sign in Duckietown (do not print this one out!)
**APPEARANCE SPECIFICATIONS**

1) **Specifications**

For traffic signage to be compliant:
- The center of the traffic signs is 13 cm height from the floor layer;
- The AprilTag is 6.5 cm sq.;
- There is a white border of roughly 0.8 cm around them;
- The signage stands perpendicular to the ground, and the angle of the sign with the road is 90°.
- The signal is flat (no deformation / folding) and without wrinkles. This can be obtained, e.g., by printing the signs on thick paper.

2) **Types**

The allowable traffic signs are as in Figure 2.8.

(a) stop  (b) yield  (c) no-right-turn  (d) no-left-turn  
(e) do-not-enter  (f) oneway-right  (g) oneway-left  (h) 4-way-intersect  
(i) right-T-intersect  (j) left-T-intersect  (k) T-intersection  (l) pedestrian
3) Placement

Signs may appear on the opposite side and at the corner of the adjacent tile from which they are viewed. In the absence of any signs, it is assumed that all network flows are allowed so a sign MUST be placed and visible whenever this is not the case.

Signs must only be placed on empty tiles, or next to one of the other tile types if on the border of a map. As mentioned, it is important to not overlap the base of the sign stand with any road marking.

The sign placements for four different cases are shown in Figure 2.9. At intersections, from each stop line 2 signs should be clearly visible: 1) the intersection type (traffic light or stop sign) and 2) the intersection topology (3-way with correct orientation, or 4-way).
On straight and curved roads, additional signs can be added as desired. Their placement is indicated in Figure 2.9c - straight road and Figure 2.9d - curved road. The signs should be placed at the border between two tiles and should face towards oncoming traffic as indicated.

In these figures the arrow is the direction of the sign.

2.4. Traffic Lights

Requires: The assembly procedure for building the a traffic light is found in Unit B-2 - Traffic Lights Assembly.

1) Placement

The lights must be at a height of 20 cm above the center of the intersection tile. The computational stack of the traffic light should be mounted in the appropriate housing outside the allowable driving region. The cabling should be housed in the appropriate structure as detailed in Unit B-2 - Traffic Lights Assembly. The traffic light pillar stands should be positioned in such a way that the embedded traffic sign stands respect the above specifications for traffic light stands.
UNIT A-3
Duckietowns and Duckiebots

**KNOWLEDGE AND ACTIVITY GRAPH**

<table>
<thead>
<tr>
<th>Requires:</th>
<th>Knowledge of the Duckietown appearance specifications.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results:</td>
<td>Knowledge of the relation between Duckietown and Duckiebots, or, what can Duckiebots do in different Duckietowns.</td>
</tr>
<tr>
<td>Next:</td>
<td>Duckietown assembly.</td>
</tr>
</tbody>
</table>

Not all functionalities of the Duckiebots require all Duckietown city elements to work. If you are wondering what learning experiences you can explore with what Duckietown, you are in the right place. We define Duckietown configurations of increasing complexity depending on what Duckiebot functionalities they support.

### 3.1. Only traffic signs (no city)

Without a Duckietown, it will not be possible for Duckiebots to showcase most of their behaviors. Nonetheless, traffic signs have AprilTags, which allow to play with the AprilTag detection and relative pose estimation pipeline, and in turn with the camera calibration and system identification procedures.

### 3.2. City loops

The fundamental Duckiebot behaviors can be explored with a very simple Duckietown. We call these simple cities: *loops*.

City loops are closed road patterns, without intersections, that meet the appearance specifications. They can be made only of floor layer elements, with no signals.

City loops enable to play with the:

- **perception pipeline**: what happens when data is obtained in form of measurements (i.e., images in the DB17 and DB18 configurations), and how it is used to extract information, leading the Duckiebots to generate a belief of their position and orientation in the lanes.
- **lane control**: the process that Duckiebots apply to transform beliefs provided by the perception pipeline in decisions on how to steer in order to stay inside the lane.
- **traffic management**: the functionality for which Duckiebots stay at a safety distance from other Duckiebots driving in front of them.

When city loops are augmented with traffic signs, e.g., road names, it is then possible to explore the functionality of:

- **AprilTag detection and relative pose estimation**: how Duckiebots detect AprilTags, and determine their position and orientation relative to them. This information can be used to localize in the map.

If you acquired a Duckietown Starter Pack, you have a city loop with traffic signs.
3.3. **Navigable cities**

The more complex Duckiebot behaviors require intersections and other city elements such as traffic lights to work. We define cities that include intersections as *navigable* cities.

Navigable cities are city loops connected by intersections, meeting the appearance specification. Navigable cities require traffic signs.

Navigable cities enable testing of Duckiebot behaviors. A complete list (for one or more Duckiebots) of behaviors can be found in the demo section of the Duckiebot operation manual.

3.4. **Robotarium**

Robotariums (Robotic aquariums) are experimental Duckietowns designed for continuous operations. Robotariums require a maintenance area in addition to a Duckietown, providing essential functions like automatic charging. More information on Robotariums can be found in the Robotarium book.
To assemble a Duckietown, follow this workflow:

0.5. Read the appearance specifications
You can find them here: Unit A-2 - Appearance Specifications.

0.6. Design a map
According to the specifications presented in Unit A-2 - Appearance Specifications, design your Duckietown.

0.7. Build it!
Once you are sure that all the specifications are met, build it.
You can refer to these resources for assembly tips on the various city elements:
- Traffic signs;
- Traffic lights;
- Watchtowers

0.8. Bring it to life!
Populate your city with tons of Duckies!

Figure 0.1. Just another day in Duckietown: the city of joy and relaxed introspection
1.1. Build a Map

Before beginning with sign assembly you should design a map that adheres to the specification.

An example that was used for the 2017 version of the class is here: (unknown ref fall2017_info/fall2017-map)

The full set of currently existing signs is available here: pdf docx
The set of tags used for the 2017 map are available here: pdf docx

1.2. Making New Signage

If you find that what is available in the database in insufficient for your needs, then you will need to add to the existing database.

To do so you will have to load the original AprilTags file available here: pdf ps
Which tag you should use depends on what type of sign you are trying add. The ranges of tags are specified in Table 1.1.

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Size</th>
<th>Family</th>
<th>ID Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traffic signs</td>
<td>6.5cm x 6.5cm</td>
<td>36h11</td>
<td>1-199</td>
</tr>
<tr>
<td>Traffic lights</td>
<td>6.5cm x 6.5cm</td>
<td>36h11</td>
<td>200-299</td>
</tr>
<tr>
<td>Localization</td>
<td>6.5cm x 6.5cm</td>
<td>36h11</td>
<td>300-399</td>
</tr>
<tr>
<td>Street Name Signs</td>
<td>6.5cm x 6.5cm</td>
<td>36h11</td>
<td>400-587</td>
</tr>
</tbody>
</table>

First, find the last sign of the type that you are trying to make in the signs and tags doc. You will use the next available ID after this one.
Construct the new sign by first copying and pasting an existing sign of similar type, and then replacing/adding the new AprilTag. To add the new april tag, use a screen capture method to crop precisely around the tag at the top and sides and include the sign id at the bottom. Then paste the tag into your word file under your desired and resize it exactly 6.5cm (2.56inches).

If you make a new road name sign, you may need to change the font size of the name so that it appears on one line (this is why we like people with names like “ROY” and “RUS”).

Important: You must also add your new sign to the April Tags DB in the software repo. Add a new block like the ones that already exist or modify the one with the appropriate tag id:

- tag_id: NEW_TAG_ID
  - tag_type: in {TrafficSign, Light, Localization, StreetName}
  - street_name: either NEW_STREET_NAME or blank
  - vehicle_name: currently not used
  - traffic_sign_type: either TRAFFIC_SIGN_TYPE or blank

The value of NEW_STREET_NAME is up to you to decide (have fun with it!). The value of TRAFFIC_SIGN_TYPE should be one of the signs in Figure 2.8

When finished, regenerate the PDF version of the Word file, and commit everything to the repo (via a pull request of course).

Note: It is also possible of course to start you own completely different signs and tags database, but make sure that you specify in the april_tags code which database to load from.

### 1.3. Traffic Signs Assembly

1) Print the signals

First, you should print out the pdf version of the signs and tags file on the thickest card stock available. Cut the signs out with a straight edge and a very sharp knife, leaving a small border of white around the sign.

2) Assemble the stands

Assemble the wooden base of the signals, according to the following instructions. Then use the provided sticker to affix the the paper sign to the wooden base.

### 1.4. Placement

For placement of signs see Subsection 2.3.3 - Placement.
UNIT B-2
Traffic Lights Assembly

KNOWLEDGE AND ACTIVITY GRAPH

**Requires:** Material: Traffic light components. To obtain them contact info@duckietown.org.

**Requires:** An initialized SD-card.

**Requires:** Tools: (strong) wood glue or hot glue gun, tape, double-sided tape.

**Results:** Traffic light.

Traffic lights are useful to coordinate traffic at intersections. Traffic lights can be used at three or four way intersections. Hardware wise, Traffic lights are Duckiebots without wheels, and a beautiful different chassis.

Reminder: for traffic lights to be recognized by Duckiebots, appropriate signage must be placed at intersections (traffic light traffic sign instead of stop sign).

This section describes the physical assembly and installation of traffic lights.

2.1. Overview
Traffic lights are composed of two wooden boxes on the diagonal direction of an intersection. One of them is equipped with the computational stack and connection to the camera.

2.2. Assembly of the traffic light parts
This section shows how to assemble the components from the laser cut traffic light parts.

**Warning:** The small parts with the hole in the middle, i.e., the ones in the left of Figure 2.1, are not all equal. Some have a round hole, others a polygonal hole. Double check you are using the right ones in the process (compare with the pics).

All parts should be glued together as showed in the pictures for enhanced structural stability.

1) Tube holder with big ground plate
Figure 2.3.

Figure 2.4.

Figure 2.5.
2) Tube holder with small ground plate

Figure 2.9.

Figure 2.10.

Figure 2.11.
Figure 2.12.

Figure 2.13.

Figure 2.14.
Figure 2.15.

3) Traffic light LED housing
Figure 2.21.

4) Ground module cover

Figure 2.22.

Figure 2.23.

5) Joint module
Figure 2.24.

Figure 2.25.

Figure 2.26.
Now that you have assembled the traffic light chassis, you are ready to add the electronics.

2.3. Components of the traffic light

Now that you have assembled the traffic light chassis, you are ready to add the electronics.
Figure 2.32. Parts of a traffic light needed to complete these instructions.

These components are needed for one traffic light:

- Tube holder with big ground plate
- Tube holder with small ground plate (Duckietown)
- Cable with soldered LED strip
- Joint module (2x)
- Traffic light LED housing
- Raspberry Pi base plate
- Ground module cover (Duckietown)
- Camera mount
- Camera mount cover
- Short tube
- Medium tube (2x)
- Long tube with hole at the side
- Raspberry Pi
- Raspberry Pi shield
- M2.5x10 MF Nylon spacers (8x)
- M2.5x8 Nylon screws (4x)
- SD card with Duckietown software
- USB cable
• Ethernet cable
Additionally, the traffic light structure can host:
• Traffic sign stands (4x)
• Traffic sign stand supports (4x).

2.4. Assembling the Traffic Light

1) Put the LEDs into the housing

Bend the LED strip at an angle to reduce the chance that the exposed soldered wires short. The exposed part of the wires should not be in contact, especially when turning on the power.

Warning: the actual traffic light in your hands might vary slightly from the pictures above. In particular, the electrical cables could have different colors or be soldered in different positions. Take note of what each color cable is soldered to, as same will go with same on the other end.
Figure 2.33. Bended LED strip cable

Figure 2.34. Cable with soldered LED strip LED housing
Carefully push the LEDs into the designated holes.
Fix the LEDs with some tape, don't use glue.

2) Connect the tubes
Figure 2.39. Medium tubes and LED housing.

Stick the tubes into the sides of the LED housing and pull the cable through one side.

Figure 2.40.

Add the joint modules on the side of the tube without the cable.

Figure 2.41.
Mount the other joint module on the long tube, such that it aligns with the hole. You can add additional tape under the joint modules to prevent them from slipping down.
Pull the cable through longer tube and stick the tube into the joint module.
Put the tubes into the tube holders.
3) Connect the Raspberry Pi

Use the spacers and the screws to mount the Raspberry Pi on the Raspberry Pi ground plate as shown in Figure 2.51.

Plug the shield on top of the Raspberry Pi.
Insert the SD card.
Connect the LED cable to the shield.
Connect the Ethernet cable.
Connect the USB cable.
If done correctly the LEDs should be on.

Close the ground module with the case.
4) **Add traffic sign stands**
5) Fully assembled traffic light
Place the traffic light at an intersection such that the LEDs are exactly in the middle and are facing each incoming lane perpendicularly.

You can verify the position is correct by verifying that Duckiebots at the red stop lines can see only one light blinking, and no reflections of LEDs facing other directions.

You can finally use the provided double-sided tape pads to fix the traffic light to the tiles.

6) SD-card image Preparation

At hardware and software level, traffic lights are Duckiebots without wheels. In initializing the SD-card of your traffic light, treat is as if it were a Duckiebot: Unit C-5 - Duckiebot Initialization. Don’t forget the `--compress` option for 16GB sd cards.

- For Robotarium users: since traffic lights are coupled to watchtowers, please use the watchtower setup: `hostname: watchtowerXX linuxname: mom password: MomWatches`
- However, if you just want to use it as a traffic light, use the trafficlight setup: `hostname: trafficlightXX linuxname: tlo password: quackquack`

After this step, you should have a traffic light image name.

```
tlo@trafficlight01.local
```

or

```
mom@watchtower01.local
```

7) Launch Traffic Lights

To download and run the trafficlight docker image, simply run:

```
docker -H $hostname.local run -dit --privileged --name trafficlight --network=host duckietown/traffic-lights:master18
```
PART C

Troubleshooting

Let’s discuss some common nuisances and, when possible, their solutions.