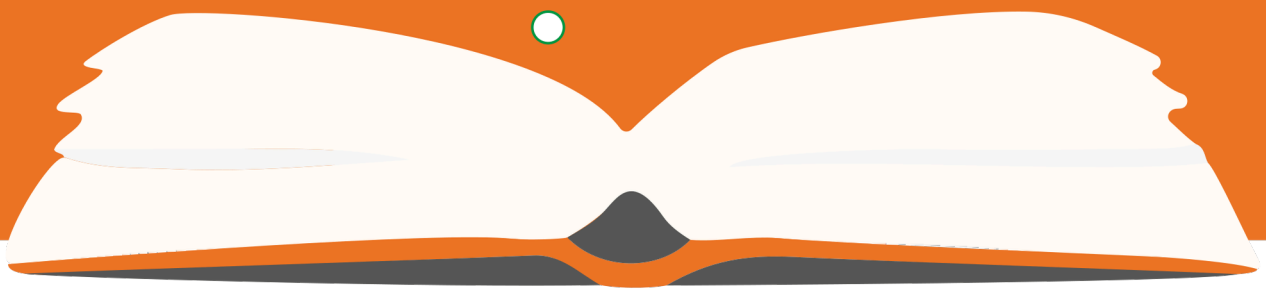
in partnership with

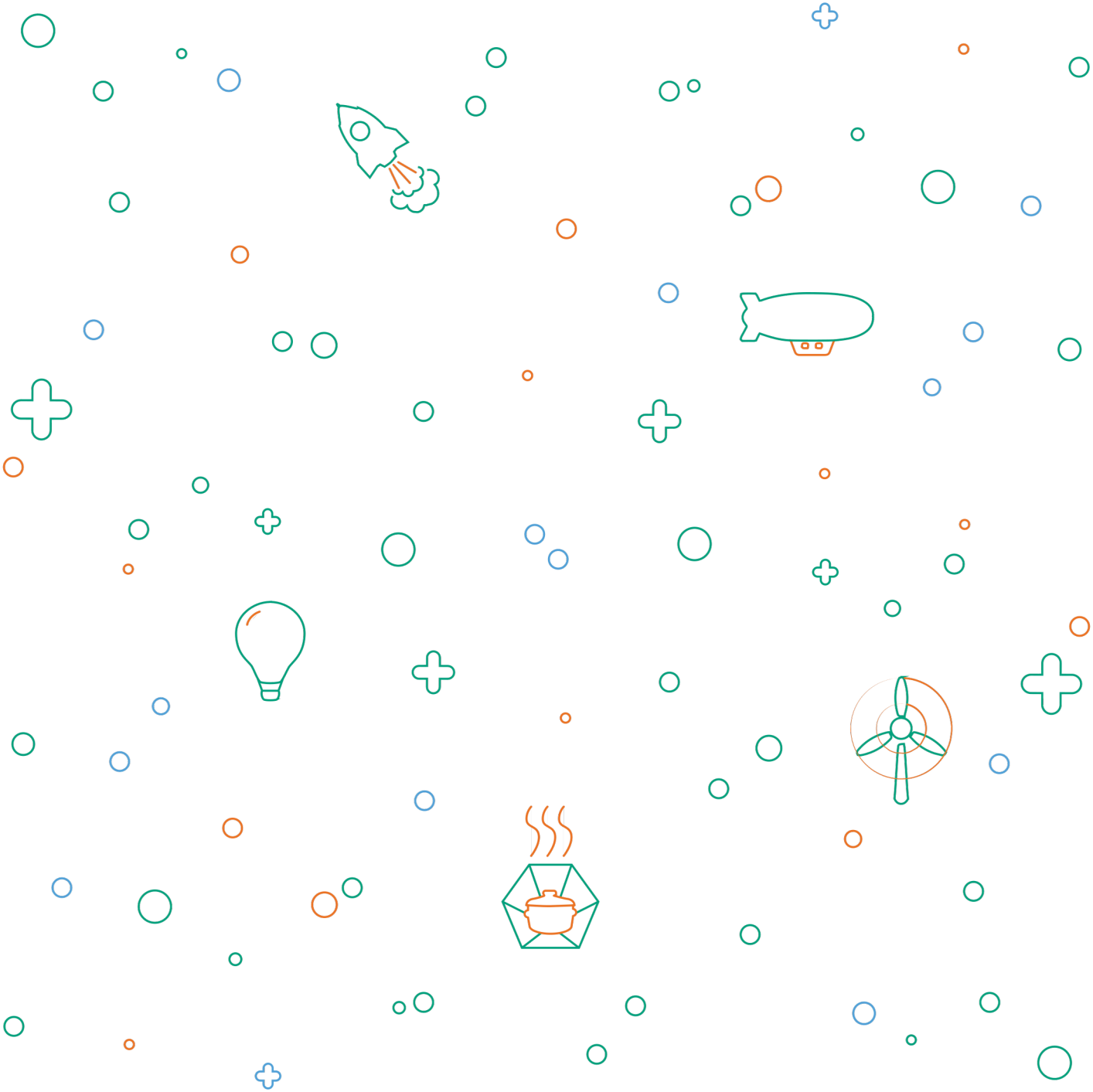
life.augmented

FLY TO MARS!



EDUCATIONAL BOOKLET

MARTIAN ROBOT



vitta science

EDUCATIONAL BOOKLET

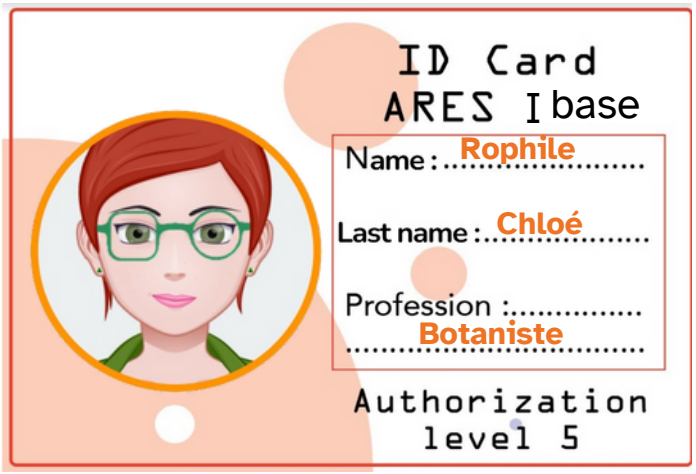
MARTIAN ROBOT

Original edition written by @P7i7Plum3
and Damien Vallot with the support of the
"Survive on Mars" team

Booklet updated in April 2025

Layout and illustration by Laura Venezia and
Diana Khalipina

• 2025 Edition •
Printed in Italy



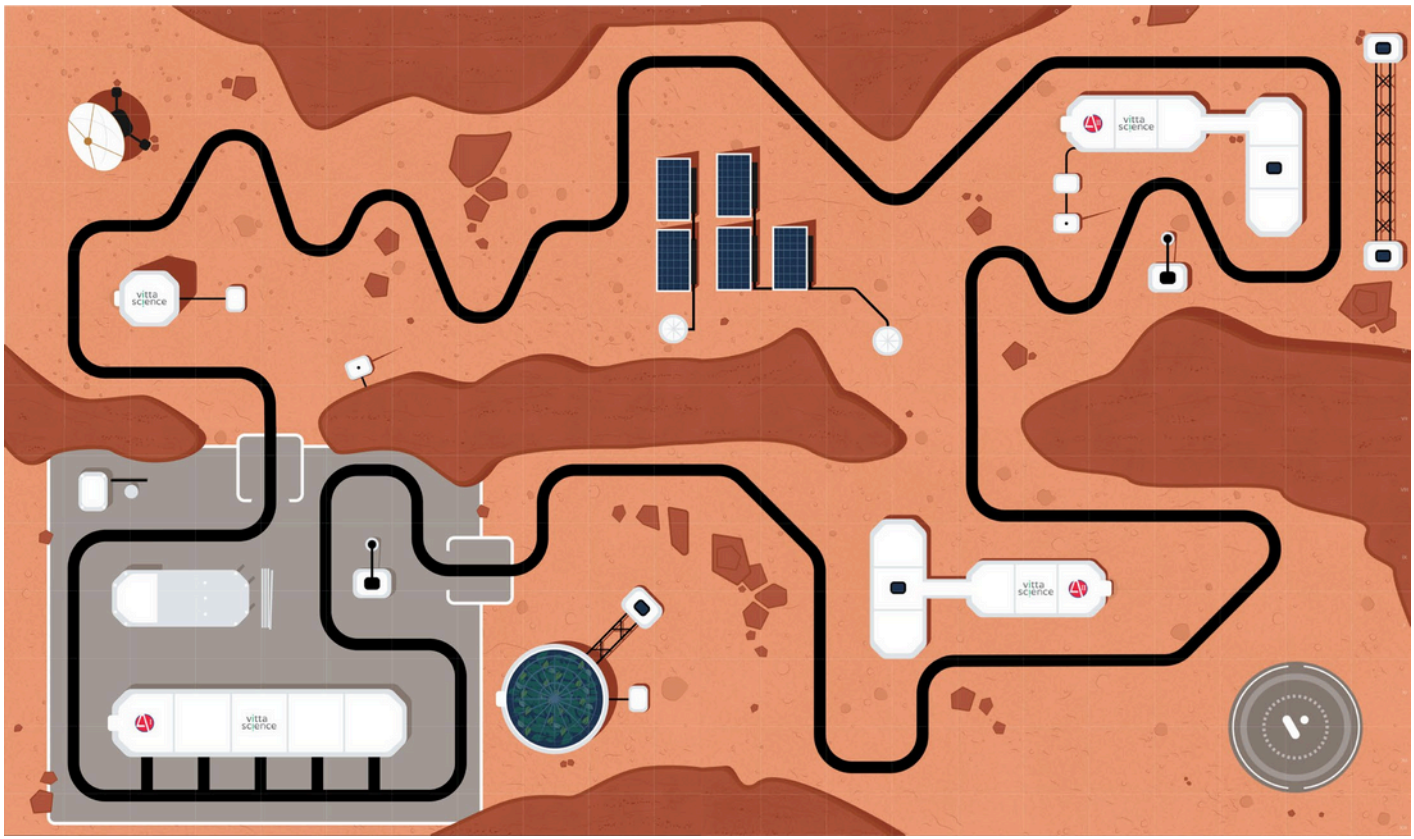
OUR TWO MARTIAN COLONS READY TO TAKE ON ANY CHALLENGE!

The year is 2041.
The success of Ingenuity's mission to Mars in 2021 is still fresh in everyone's memory. This NASA flying robot, deployed alongside the Perseverance rover, validated the concept of a robot attached to a control center on Mars.

Following this success, numerous missions have been successfully completed. Since the installation of the Ares I underground base on Mars in 2039, we have been seeking to colonize its surface. To this end, a swarm of robots was deployed there in 2040 to patrol, analyze, and build the future Ares II base, planned for 2042. The Mars Space Agency (MSA), based on Earth, is leading the entire project with the help of the first Martian colonists on site.

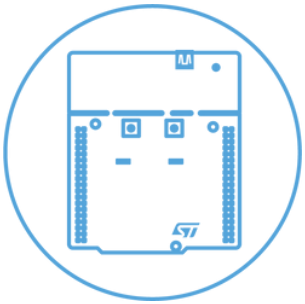
In the missions, we will be dealing with two of our most accomplished colonists from the Ares I base. Indeed, following an incident that occurred on the base, they face several challenges!

THEATER OF OPERATIONS

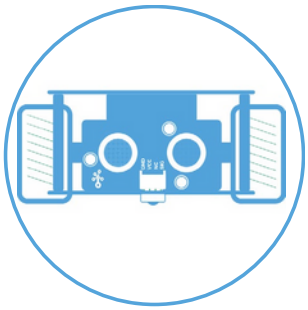


MATERIALS NEEDED TO CONSTRUCT AND USE THE MARTIAN ROBOT KIT

Contenu du KIT :

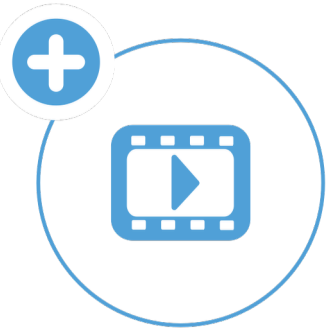


NUCLEO-L476RG board

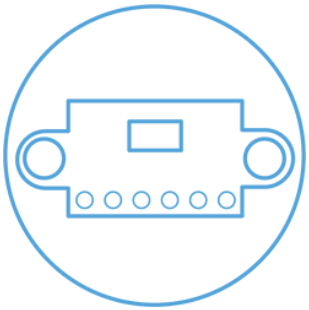


DonutBot Robot

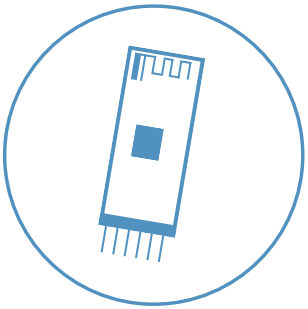
Robot track



Online digital resources



VL53L0X Distance Sensor



Bluetooth Module HM10

Elements for assembly

Provide a computer

WARNINGS REGARDING THE USE OF THE KIT

BE CAREFUL !

Presence of small elements, do not ingest (risk of suffocation).

+7

Minimum age

Not suitable for children under 7 years old.

INSTRUCTIONS FOR PROPER SORTING

The following infographic details the sorting instructions for the various components of the kit.
For more information, visit your city's website.

SUMMARY

PAGE
33

PAGE
28

Mission 2:
Motor Control

- Mission 2-1: Move the robot forward
- Mission 2-2: Learn to pivot
- Mission 2-3: Follow a trajectory

Mission 3:
Test the detector

- Mission 3-1: Discovery of the Time of Flight (ToF) distance sensor
- Mission 3-2: Discovery of the optional ultrasonic distance sensor
- Mission 3-3: Distance sensors and motors

PAGE
36

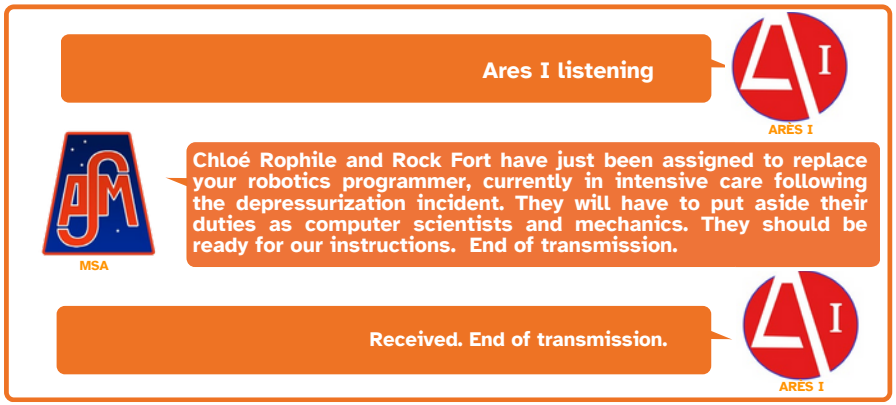
PAGE
39

Mission 4: Real-time control with the Bluetooth module

- Mission 4-1: Communication verification
- Mission 4-2: Testing in real “Martian” conditions: the remote control

Mission 5: Movement in tile mode

- Mission 5-1: Moving in square mode
- Mission 5-2: Detecting obstacles before moving



PAGE 47

PAGE 43

Mission 6: Line Follower

- Mission 6-1: Follow a line
- Mission 6-2: Follow a line with 3 sensors
- Mission 6-3: Follow a line and avoid obstacles

Mission 7: Complex scenario

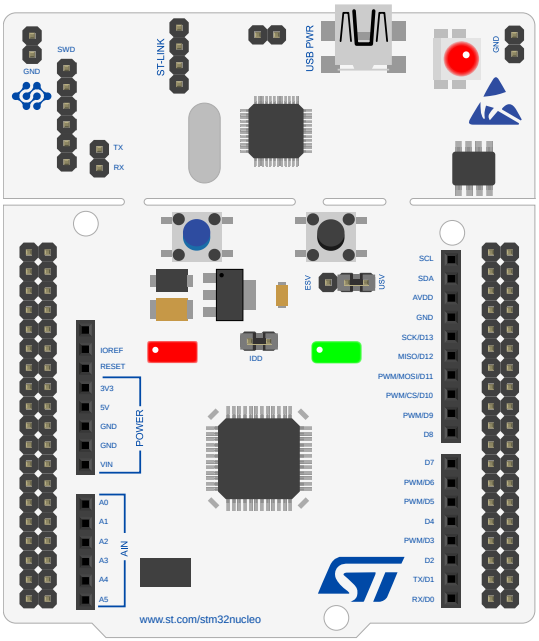
- Mission 7: Finally the routine!

Overview of Vittascience interface and hardware ⏱ 15 min

The Mars robot kit includes all the components needed to build a robot capable of autonomously exploring Mars. It comes with a NUCLEO-L476RG board developed by STMicroelectronics, equipped with an STM32L476RG microcontroller.

• NUCLEO-L476RG card overview

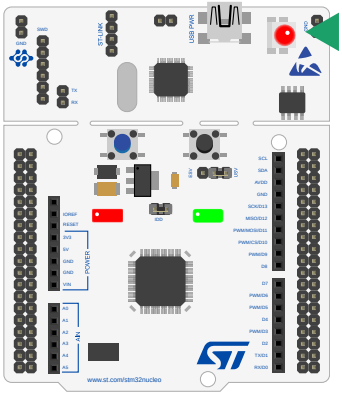
The following illustration shows the NUCLEO-L476RG board, which can also be used as a support for various assemblies with other components, including Grove modules.

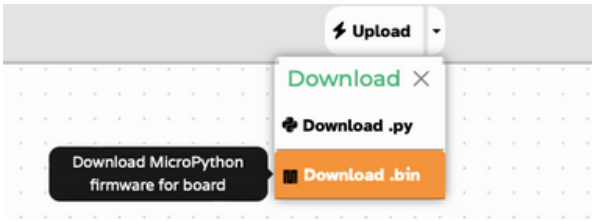


It is programmed via the Vittascience interface in MicroPython. **Before first use, it is necessary to load firmware into its microcontroller so that it can run Python programs** (see box).

MICROPYTHON FIRMWARE LOADING PROCEDURE

To program the NUCLEO-L476RG in MicroPython, either by code or by blocks from the Vittascience website, you must load the appropriate firmware into your STM32L476RG microcontroller. This step only needs to be performed once, upon receipt of the kit.





Here are the steps to follow:

1 • Connect the USB cable to the mini USB port on the board. The LED will light up red.

2 • Download the firmware (.bin) from the Vittascience website or at:
https://stm32python.gitlab.io/fr/docs/Micropython/Telechargement,
then drag and drop it into your board, which appeared as a USB drive named "NOD_L476RG".
Caution: do not unzip the file.

3 • When the download is complete, the LED will turn green.

4 • Unplug the USB cable.

5 • Connect the board to the computer, LED 5 is now lit red. The board is powered.

6 • Use the Vittascience interface to program your card.

Got a problem or a question? We're here to help: contact@vittascience.com

• Board programming ⏱ at the discretion of the supervisor

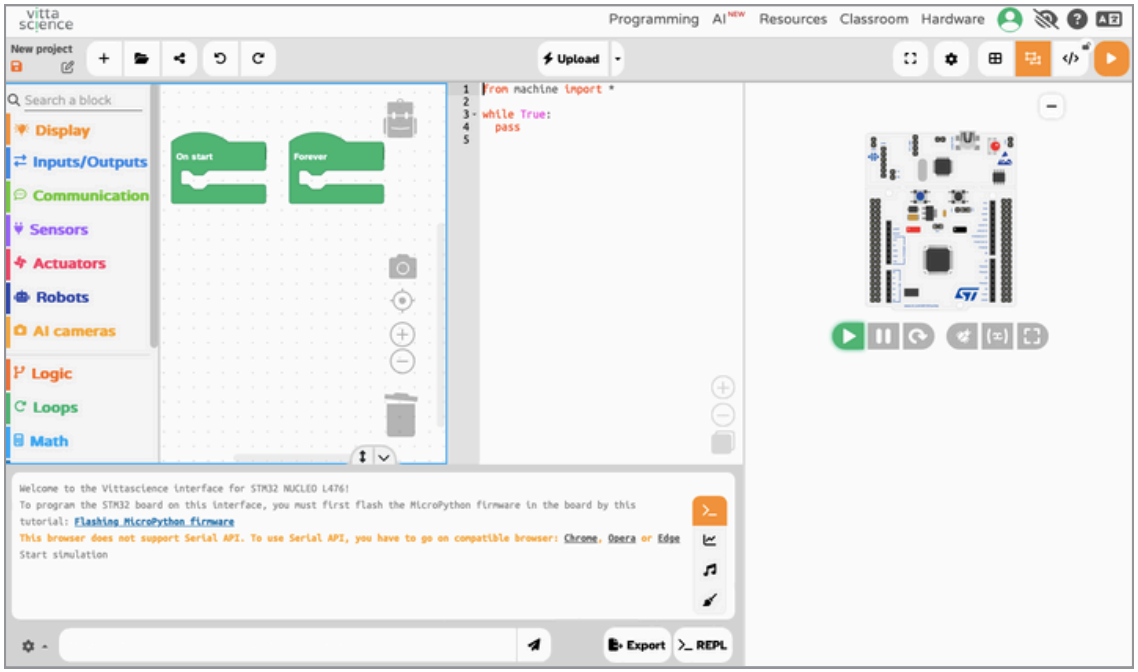
Here we detail how the Vittascience.com online programming interface works.
It's also possible to program the board using Arduino software (C++ language).

1 • Create a Vittascience account

We recommend creating an account on our website. This is not required to use your kit, but it will allow you to save and share your programs, resources, and feedback. To do this, go to Vittascience.com and click the green icon in the top right corner to register.

2 • The interface

The interface allows block programming with parallel transcription in Python language.



Please note : The NUCLEO-L476RG board is required to run the program on the robot. Find resources and programs to learn how to program with it on Vittascience.com.

Transfer the program to the board: 

the code is executed on the board as soon as the transfer is complete.

Port selection: When you connect the board to the computer, the interface automatically detects which port it is connected to. The window that appears allows you to select the correct port if multiple boards are connected to the computer.

Undo or redo:  to go back through the actions performed.

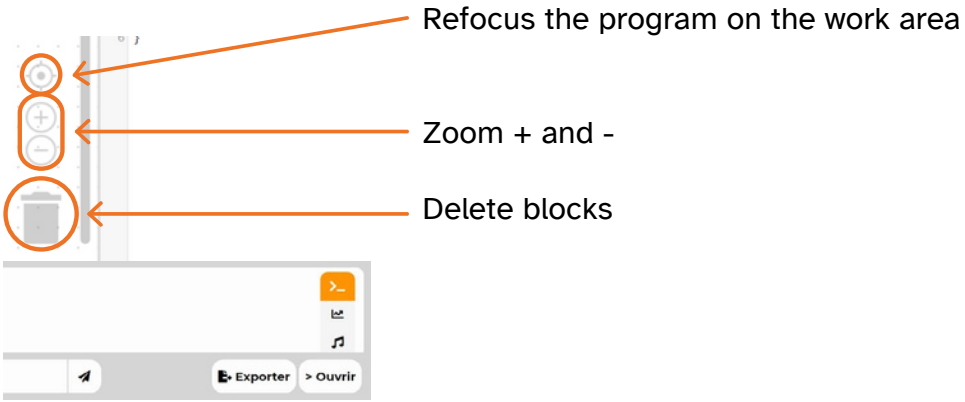
Start a new project:  to start a new blank project, click this button.

Save project:  to save your project, click this button. This button also allows you to share the program with the community.

Open an existing project:  if you want to open programs that have already been created, or have your students work on a framework that you have created, click on this button.

Coding in Python:  if you want to code directly in Python.

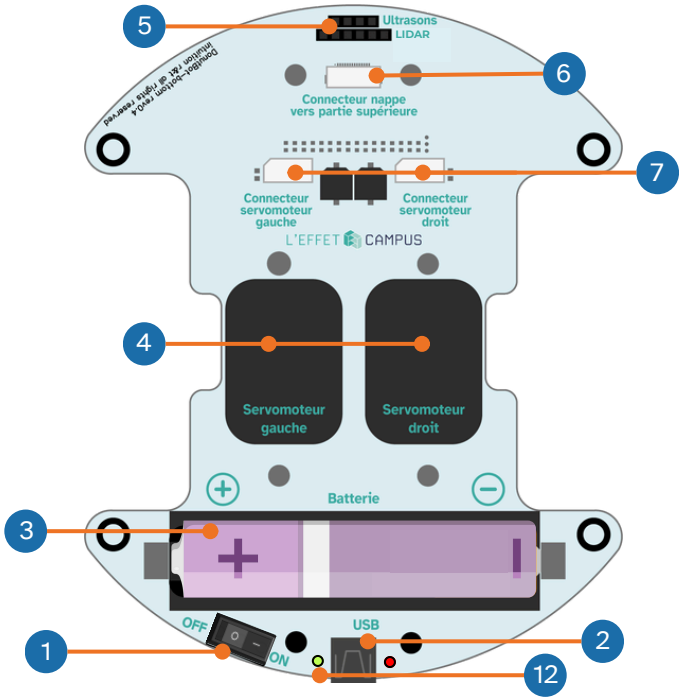
Quick access:



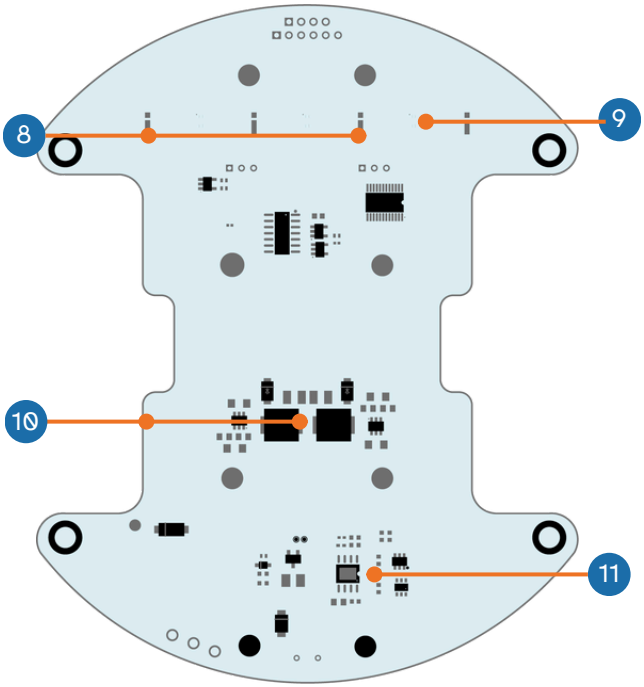
Tip : This programming interface is designed to be very easy to use.
Additional resources are available on the website to help you get started.

Overview of the Donutbot robot

• **Figure 1: Lower base, front face**

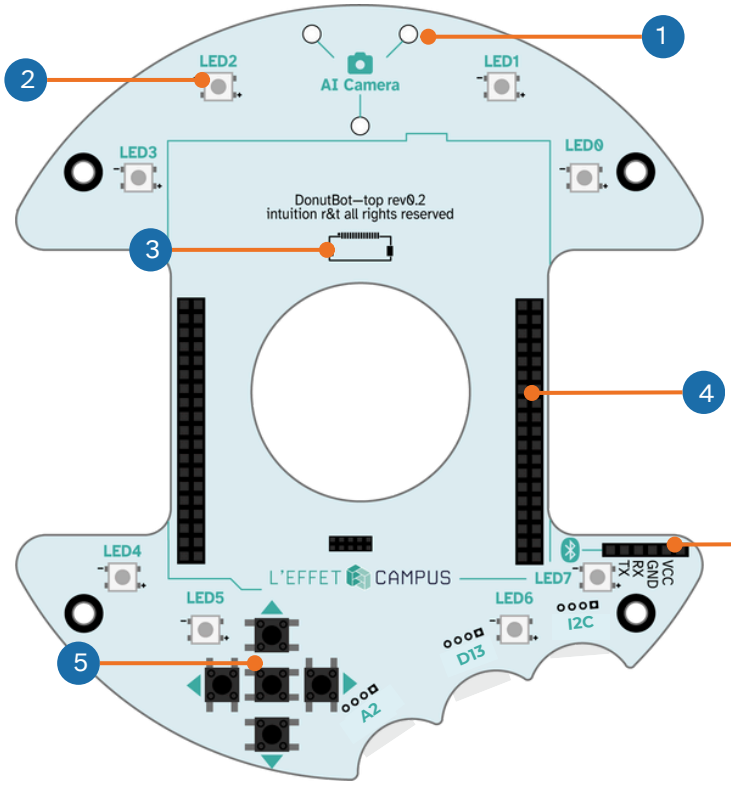


• **Figure 2: Lower base, rear face**

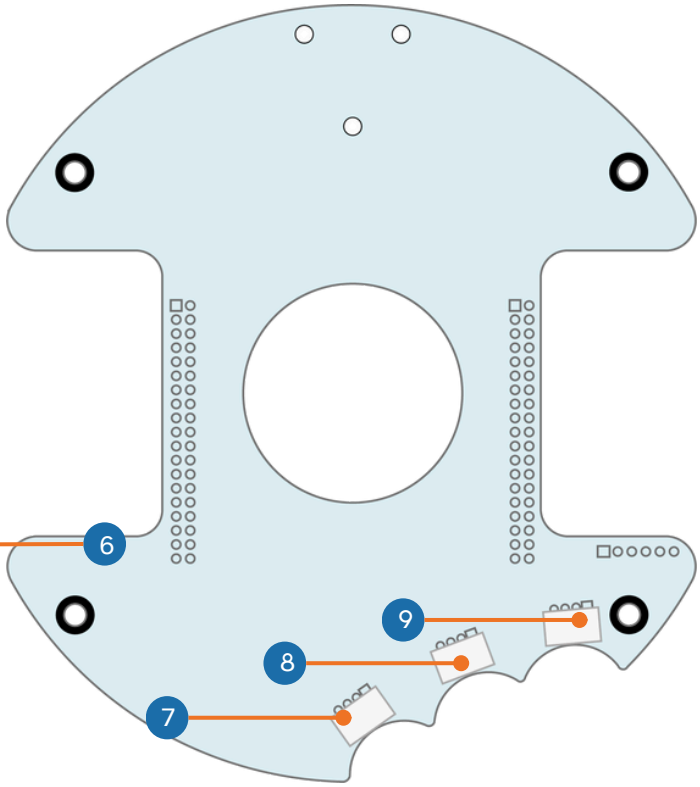


1 • On/Off power switch
2 • 5V USB port for battery charging
3 • Battery holder for 18650 Li-Ion rechargeable battery. Please note that the orientation of this box may be reversed depending on the chassis version.
4 • Servomotor
5 • Distance sensor holder (ultrasonic or ToF)
6 • Ribbon cable connector toward the top of the robot
7 • Servomotor connectors
8 • White LED for line tracking
9 • Color sensor
10 • Robot voltage management system
11 • Battery charge control system
12 • Robot power indicator (green LED) and battery charge indicator (red LED)

• **Figure 3: Upper base, front face**



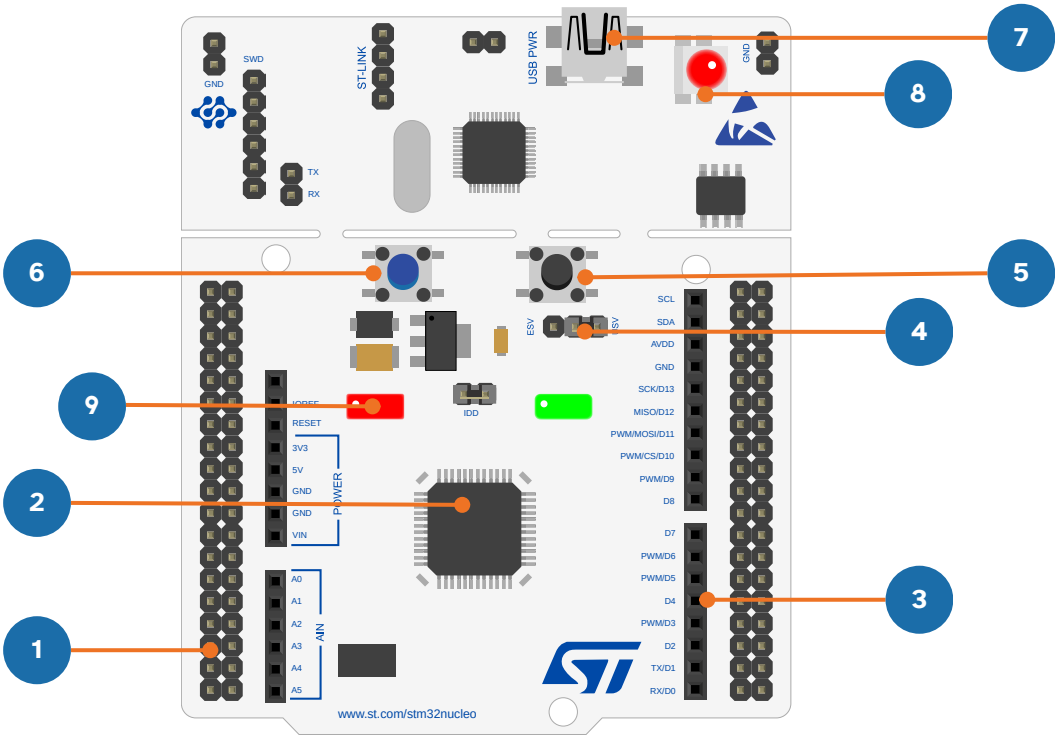
• **Figure 4: Upper base, rear face**



1 • AI camera support
2 • Addressable RGB LEDs (x8)
3 • Ribbon cable connector towards the bottom of the robot
4 • Expansion connector for the NUCLEO-L476RG board
5 • Control buttons
6 • HM10 Bluetooth module connector
7 • Grove digital port connector D13
8 • Grove analog port connector A2
9 • Grove I2C port connector

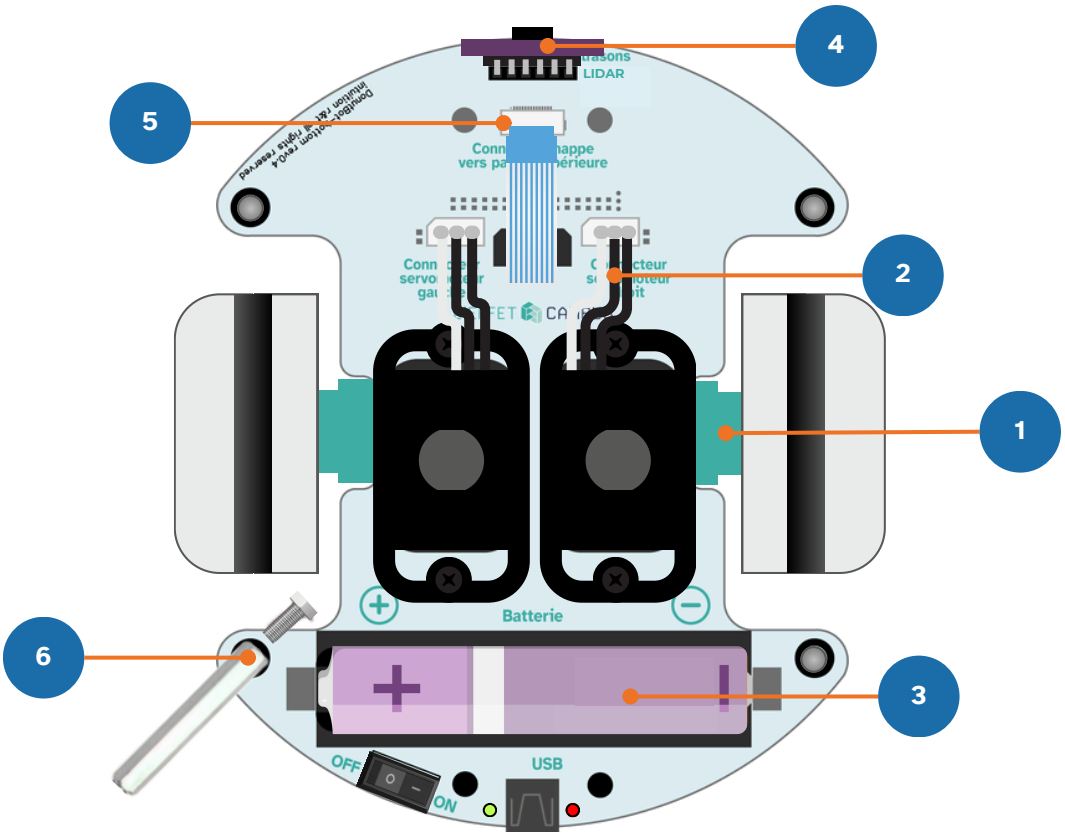
NUCLEO-L476RG Card Overview

• **Figure 7: NUCLEO-L476RG (front panel)**



1 • ST-Morpho expansion connector (male)
2 • STM32L476RG microcontroller chip
3 • Arduino expansion connector (female)
4 • Power and programming connector (male)
5 • RESET button
6 • User button
7 • Mini-USB connector
8 • LED indicating the status of the ST-LINK programming and debugging interface
9 • Programmable user LED

Assembling the DonutBot robot ⏱ 15 min



1 • Attach the wheels to the servomotors using the screws.

2 • Connect the servomotor connectors according to the instructions.

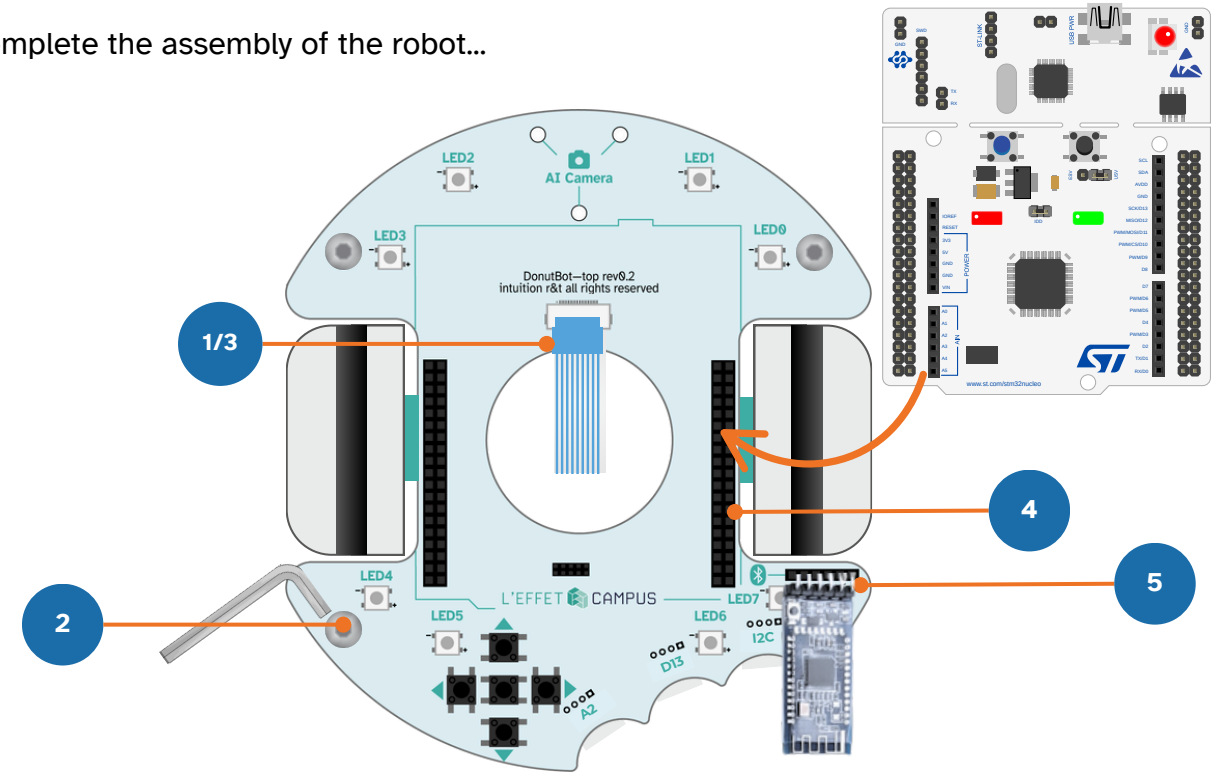
3 • Connect the battery if necessary. Pay attention to the battery polarity!

4 • Position the distance sensor in its dedicated slot. The measuring module should face outwards.

5 • Secure one end of the cable into its connector (lift the latch and then lower it to lock the cable).

6 • Start attaching some of the spacers (1 screw + spacer) into the designated holes.

To complete the assembly of the robot...



1 • Route the cable through the central recessed area of the upper part and place it on the spacers. The upper side with the connectors of the NUCLEO-L476RG card must be positioned upwards.

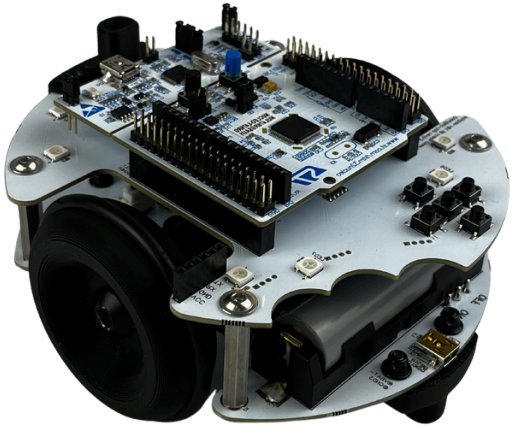
2 • Screw the upper part onto the spacers using the 4 remaining screws.

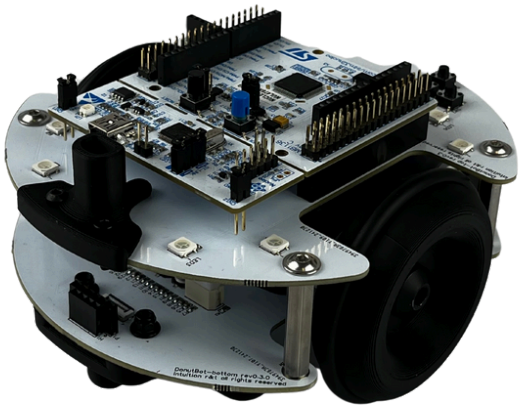
3 • Connect the cable to its dedicated slot on the upper part of the robot (you must lift the latch then lower it to lock the cable).

4 • Position the NUCLEO-L476RG board in its slot, respecting the orientation (use the outline drawn on the robot's chassis as a guide).
Be sure to check that the connectors are properly aligned!

5 • Secure the Bluetooth module, if necessary, in its connector. Be sure to respect the orientation! The VCC and GND markings must match in pairs between the module and the chassis.

Photos of the assembled Martian robot





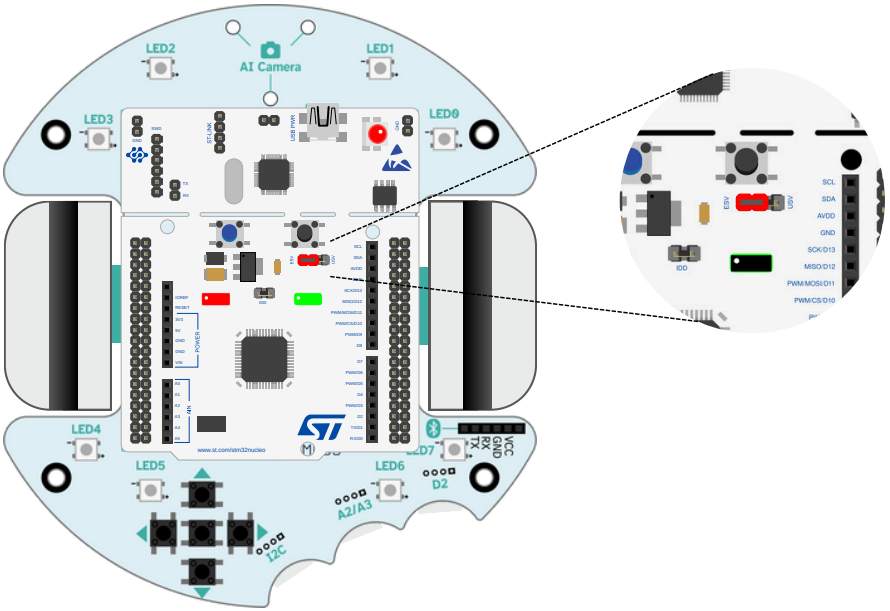
The photos above will give you visual confirmation that you have not made a mistake in following the assembly instructions.

Now it's time to move on to programming the robot!

Programming the Donutbot robot

Configuration for programming in MicroPython with the Vittascience interface

1 • Turn the robot's power switch to "ON."
2 • Set the jumper (shown in red in the figure below) to the "E5V" position.
3 • Connect the board to your computer using the USB cable.
4 • Open the Vittascience programming interface: vittascience.com/l476.
5 • Click the "Upload" button to upload the program to the board.



Starting the robot

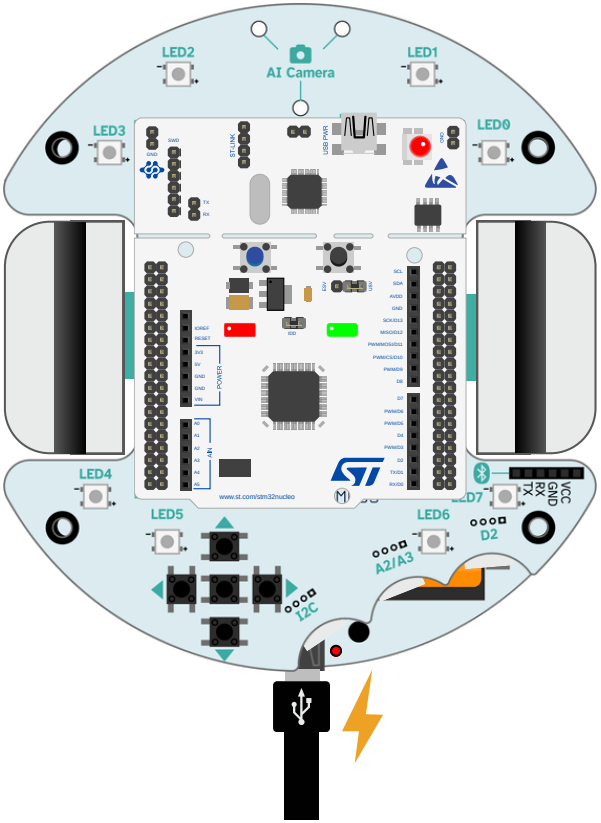
1 • Disconnect any USB cables connected to your robot.
2 • Verify that the jumper (in red) in the figure above is in the "E5V" position. The NUCLEO-L476RG board is powered directly by the robot's battery.
3 • Set the robot's power switch to "ON." The LED on its base should light up green.

The robot will begin executing the last program you downloaded to the NULCEO-L476RG board using the Vittascience programming interface.
If your robot's behavior is not as expected, check your program and make sure the battery is sufficiently charged.

Charging the Donutbot robot batteries

1 • Connect the USB cable to the robot's 5V USB port on the bottom of the DonutBot chassis.
2 • The charging LED will turn red and stay on until the battery is fully charged.
3 • The light will turn off when the battery is fully charged.

Your robot is now ready to use. Unplug the cable and turn the power switch on the robot to "ON."



Mission • 1 ⏱ 20 min

Turn on the LEDs and interact with the robot

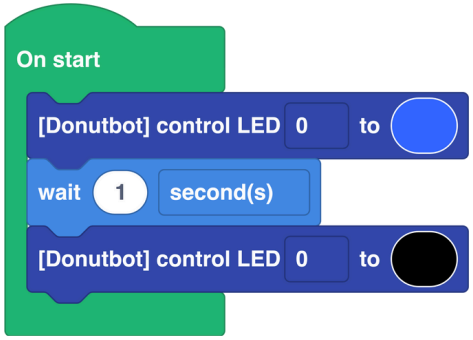
For each mission, you'll find the programming code for each block corresponding to the program. Python programming is also available on the Vittascience interface. To save space, we've chosen not to display it for all missions in this booklet.

- **Mission 1-1 : Turn on the robot's LEDs**

To begin this first mission, we will turn on the robot's LEDs. This very simple mission allows you to master program transfer and the Vittascience platform interface.

The blocks to use for this mission are in the left menu, to begin with only the following two categories will be useful:

Robots Inputs/Outputs




```
from machine import *
import neopixel
import utime

""" DonutBot robot """

# Neopixel on D12
d12 = Pin('D12', Pin.OUT)
npDonutbot = neopixel.NeoPixel(d12, 8)

npDonutbot[0] = (51, 51, 255)
npDonutbot.write()
utime.sleep(1)
npDonutbot[0] = (0, 0, 0)
npDonutbot.write()

while True:
  pass
```

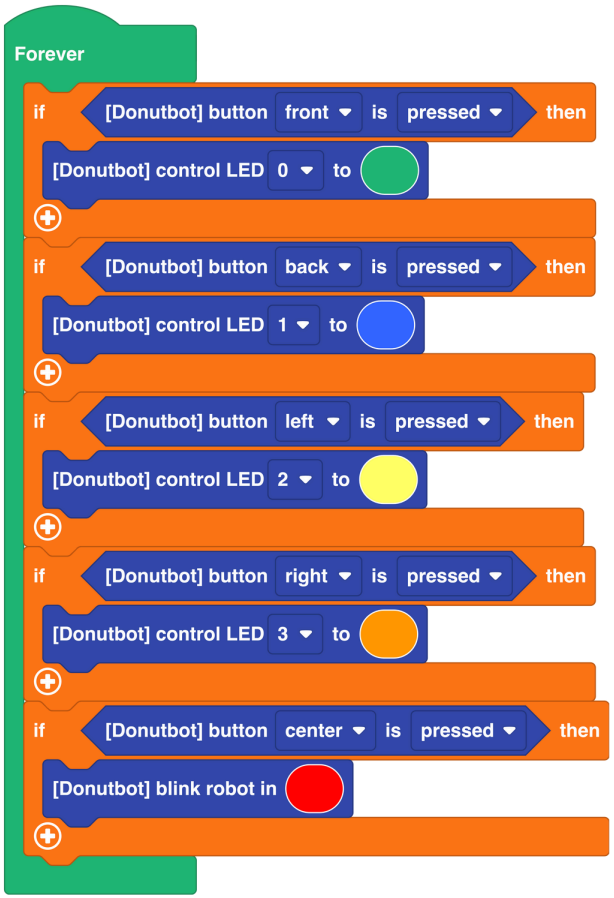
Access the program online:

https://en.vittascience.com/l476/?link=6807e6d554b1d&mode=blocks

• Mission 1-2 : Using the buttons

We've just learned how to turn on an LED on the robot using the board. Now let's learn how to interact with it using the buttons. We'll make the LEDs light up when we press the robot's buttons.

To do this we will need new blocks which are located in the following menus:



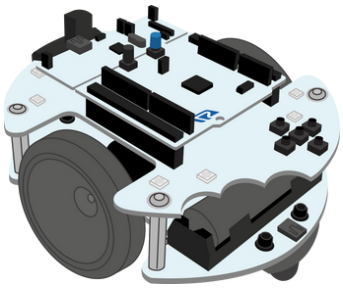
Access the program online:

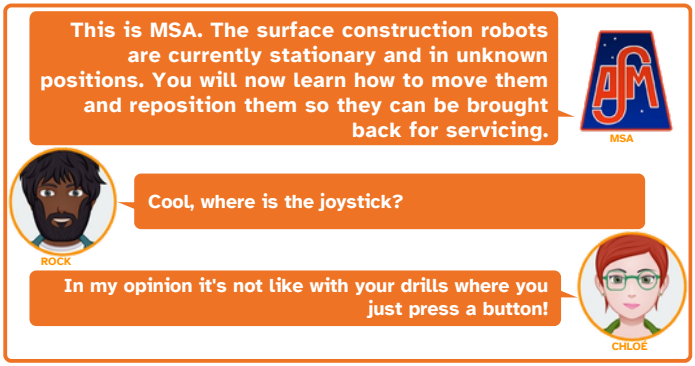
https://en.vittascience.com/l476/?link=6808a19248558&mode=blocks

Mission • 2 ⏱ 40 min
Engine control

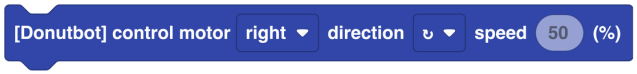
• Mission 2-1 : Move the robot forward

It's time to get our little robot moving. Let's start with the simple movements of "moving forward" and "turning."





The robot's wheels can be controlled in several ways, using different code blocks:



Block 1: Allows you to control each motor, right or left, independently, and to control the direction of rotation and the speed.

Block 2: Allows you to control the direction and speed of rotation of both motors simultaneously.

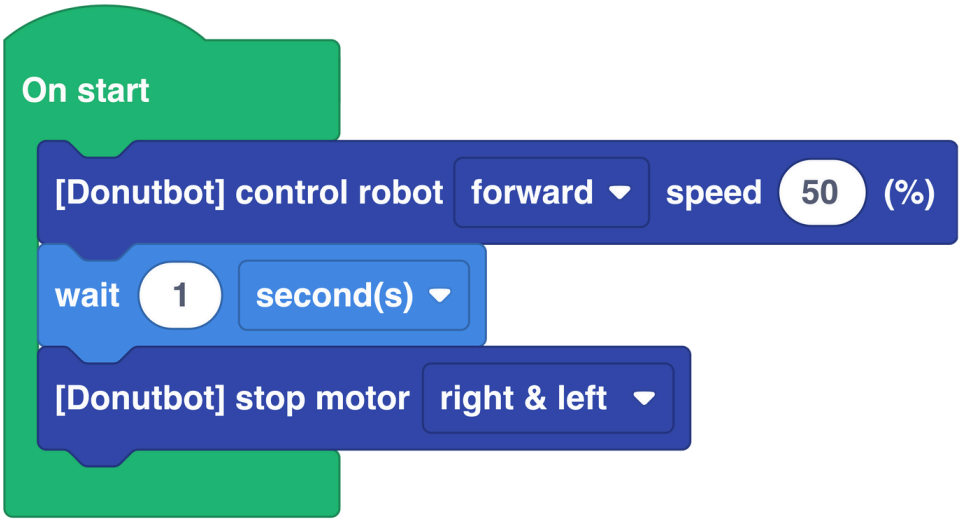
Block 3: Allows you to activate a motor to rotate right or left and control its power.

Block 4: Allows you to stop one of the two motors, or both simultaneously.

Block 5: Allows you to release the motor and free the wheel to spin.

For our first program, we'll move the robot forward for 1 second.
We'll use blocks located in:

Here is the block programming code for this program:



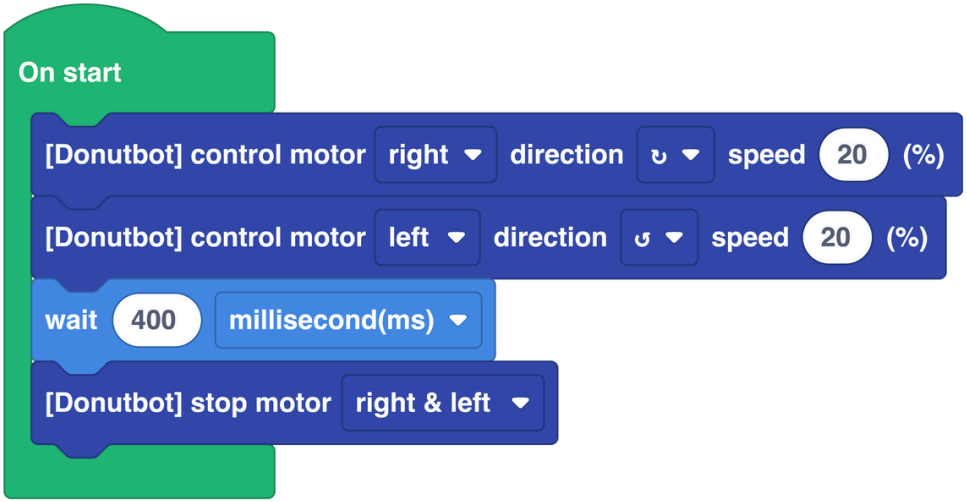
Access the program online:
https://en.vittascience.com/l476/?link=6808a4886ce1c&mode=blocks

• Mission 2-2 : Learn to pivot

A robot moving straight ahead is great, but it's dangerous. We need to teach it how to turn! To do this, we need to control each motor independently.
We have two options for turning: the "tank" method or the "car" method. The tank rotates its right and left tracks in opposite directions, allowing it to rotate on the spot. The car with the differential system generally has one wheel spinning faster than the other to follow a curved path. Each technique has its advantages and disadvantages. The "tank" technique allows for tighter turns at the expense of overall travel speed. For the "car" method, it's the opposite.

In our case, we will be moving in a hostile environment (Mars), so we will prioritize the robot's maneuverability with the "tank" method to make tight 90° turns. To do this, we will use blocks found in:

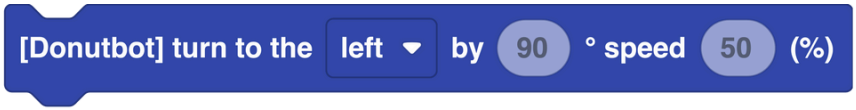
Here is the block programming code for this program:



Access the program online:
https://en.vittascience.com/l476/?link=6808ab714a3ce&mode=blocks

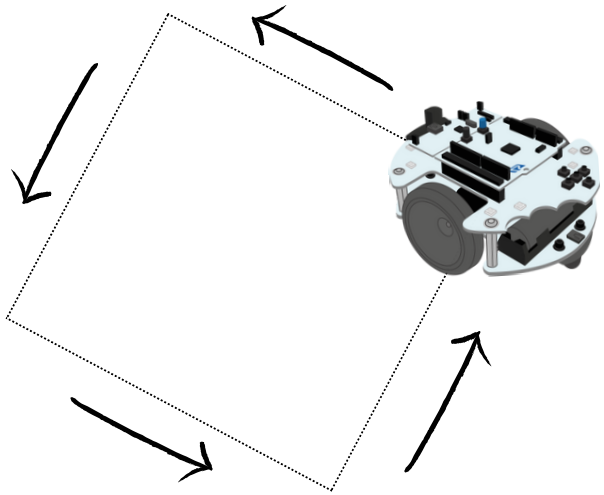
This program probably won't produce a perfect 90° turn. To do this, you'll need to adjust the timings either through trial and error or using a proportionality chart. Don't hesitate to reduce speed to better control the direction.

You can also use the following block to determine a rotation angle:

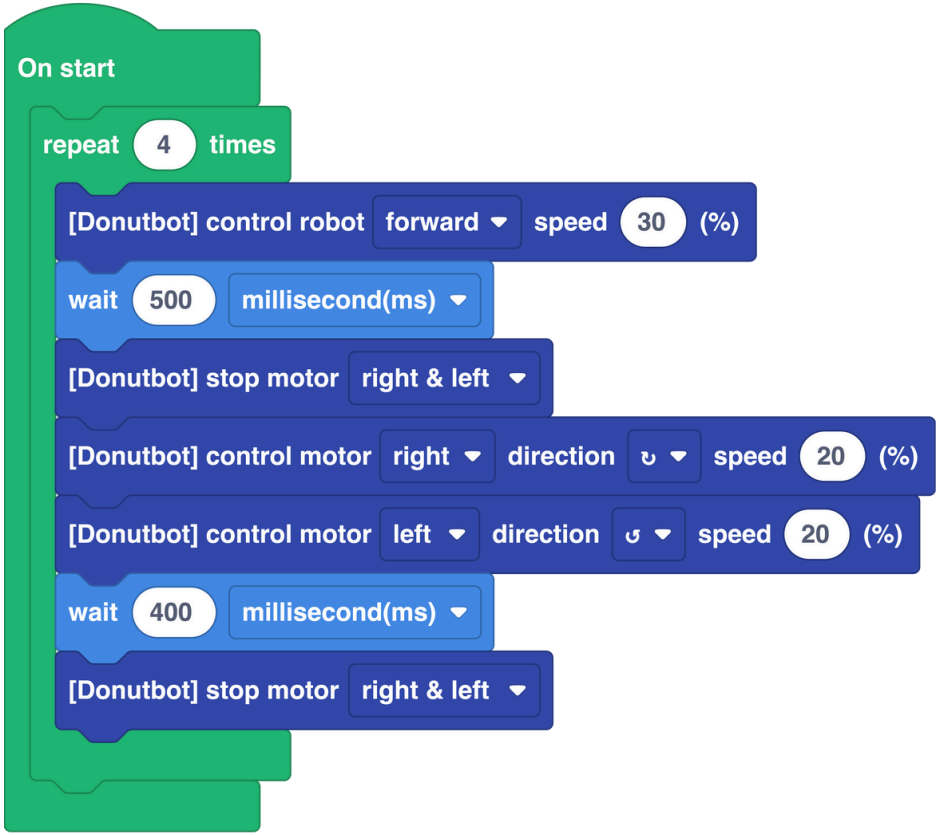


- **Mission 2-3 : Follow a trajectory**

We now have complete control over our robot and can move it wherever we want. You know how to move forward, backward, and turn 90°.
Now try to follow a square path! To do this, we'll use blocks located in:



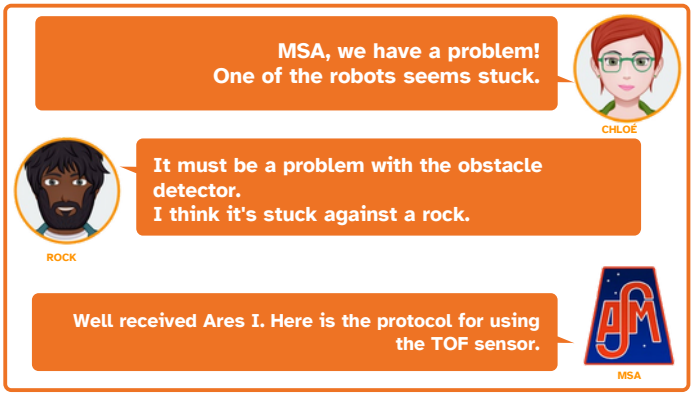
Here is the block programming code for this program:



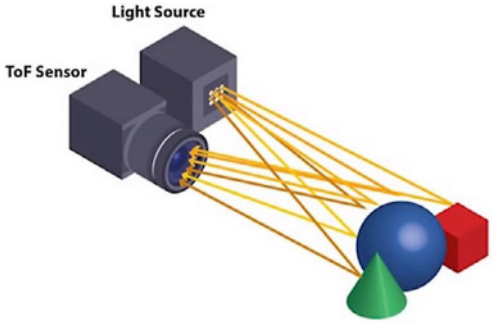
Access the program online:
https://en.vittascience.com/l476/?link=6808abde0c6c2&mode=blocks

Mission • 3 ⏱ 40 min
Test the obstacle detector

- **Mission 3-1 : Discovery of the Time of Flight (ToF) distance sensor**

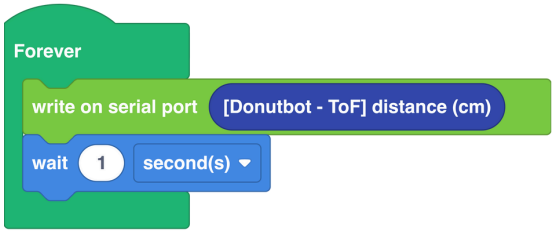


A "Time of Flight" sensor, abbreviated ToF, detects obstacles in front of it and determines their distance.



Its operating principle is as follows:
The ToF sensor sends out light waves (infrared laser) to probe the space in front of it. Since their propagation speed (that of light in air) is known, all that's left for its electronics to do is calculate the light's round-trip time to estimate the obstacle's distance.

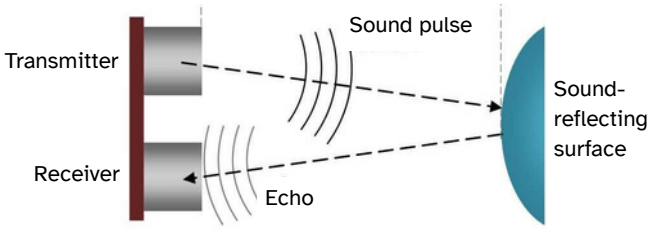
For this task, we will display the distance of an object in the computer's serial console. To do this, we will use blocks located in:



Access the program online:
https://en.vittascience.com/l476/?link=6808ac49d58b3&mode=blocks

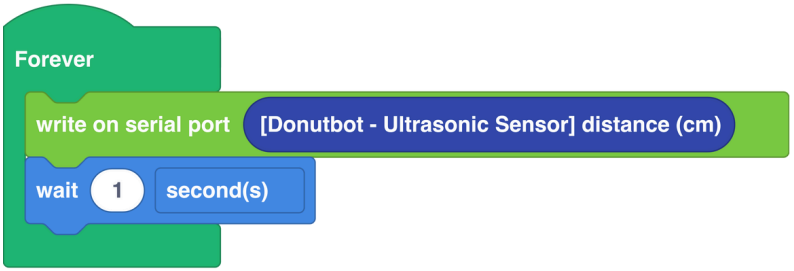
• Mission 3-2 : Discover the ultrasonic distance sensor (available as an option)

An ultrasonic sensor detects obstacles in front of it and determines their distance. It works like the Time of Flight sensor, but is slower and less accurate.



Its operating principle is as follows:
A transmitter produces a sound pulse.
This pulse propagates until it encounters an obstacle (at a distance d), which partially reflects it back to a receiver (echo).
When the echo reaches the receiver, the sound has traveled approximately a distance 2d (unknown) in a time t (known), at the speed V (known) of sound in air.
From this, we deduce d.

For this task, we will display the distance of an object in the computer's serial console. To do this, we will use blocks located in:



Access the program online:
https://en.vittascience.com/l476/?link=6808ac91bc783&mode=blocks

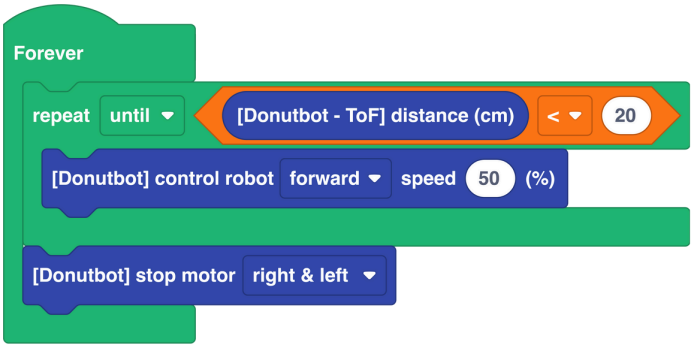
• Mission 3-3 : Distance sensors and motors

We'll use the ToF sensor (or ultrasonic sensor) to move the robot.

The robot will have to move until its distance from an obstacle reaches a chosen value (for example, 20 cm or less). To do this, we'll use blocks located in:

In this assignment, we'll use logical comparators, which allow us to test whether a piece of information is true or false. Depending on the result, we'll decide whether to perform a particular action. In our case, the question will be:

“Is the distance between my robot and the obstacle less than 20 cm?”



If you use a centimeter to measure the distance obtained after the robot has moved, there may be a slight difference with the sensor's reading, possibly for two reasons:

- A protocol error: you should not measure at the wheels, but rather at the distance sensor.
- The robot's inertia, which cannot stop instantly; the higher its speed, the longer its braking distance.
- The NUCLEO-L476RG's microcontroller needs fractions of a second to process the information and send the braking command, which will also not be executed instantly.

These latencies can result in a few millimeters or even centimeters of excess travel, depending on the case.

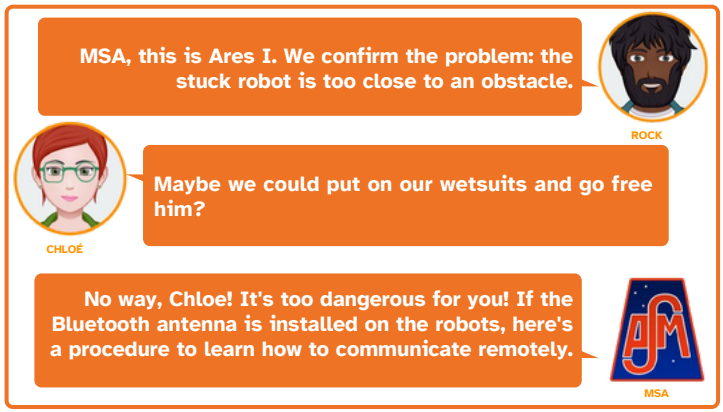
Note: It is entirely possible to use the ultrasonic sensor instead of the ToF sensor; the program will be identical except for the name of the sensor used.

Access the program online:

https://en.vittascience.com/l476/?link=6808acf337cbe&mode=blocks

Mission • 4 ⏱ 40 min

Real-time control with the Bluetooth module

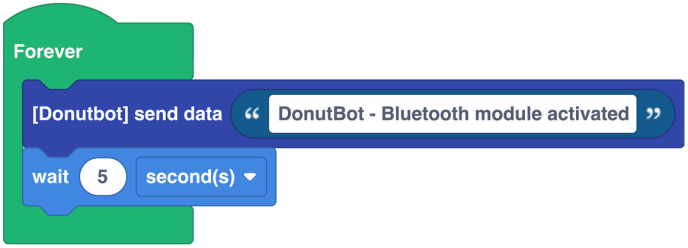


HM-10 Bluetooth module to attach to the robot.
Be careful to respect the pin orientation!

We will now control the robot using wireless communication. The idea is for the robot to receive information sent via Bluetooth by the operator and react according to the instructions transmitted.

• Mission 4-1 : Communication verification

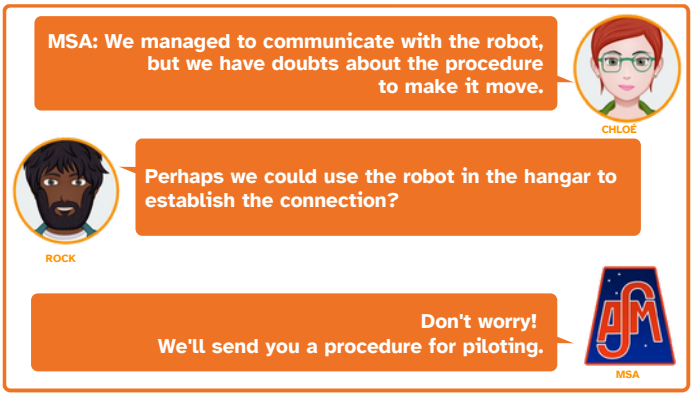
In the first mission, we will establish communication with the robot. We need a smartphone to receive the signal emitted by the robot. To do this, we will use blocks located in:



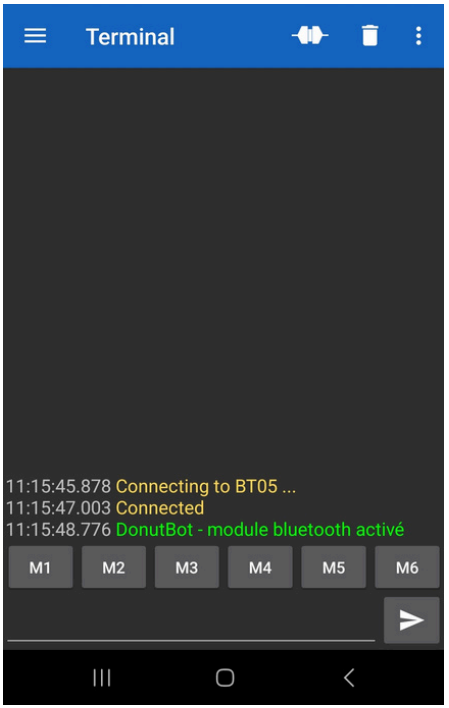
Access the program online:
https://en.vittascience.com/l476/?link=6808adb968d54&mode=blocks

Bluetooth technology allows data to be exchanged between two devices (in this case, your smartphone and the robot) using radio waves with a frequency around 2.4 GHz.
This technology is widely used in everyday life: hands-free car kits, wireless headphones or earbuds, smartwatches, etc.

On the smartphone, we need an application like Serial Bluetooth Terminal (Android) or BLE Tools (iOS).



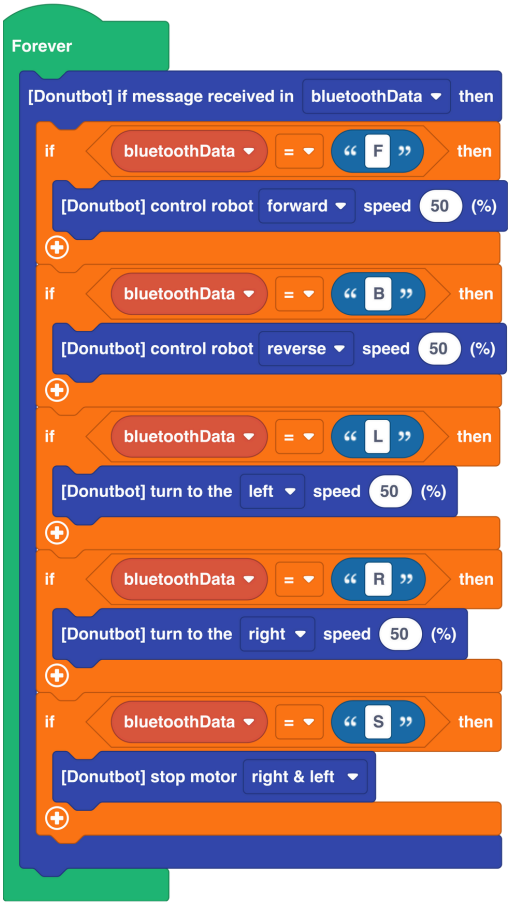
The screenshot opposite shows the Serial Bluetooth Terminal application receiving the data sent by the robot.



- **Mission 4-2 : Testing the remote control in real "Martian" conditions**

This new exercise will involve piloting our little robot in the hangar. To do this, we'll use the programs created in the previous missions. The goal will be to move the robot forward, turn, reverse, and stop using the various commands sent.
To do this, we will use blocks that are located in:

It's also possible to create your own app for a more visually appealing look.
We suggest using one of two websites: Thunkable or App Inventor.



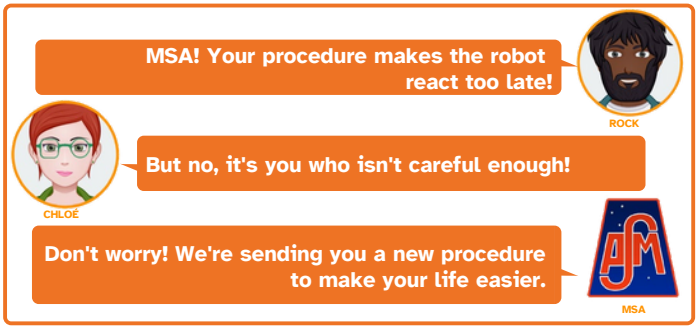
It is clear that controlling the robot with this system is complex because there is a latency period.

A more efficient solution is to program it to manage its movements autonomously based on the rotation speed of its motors as well as information about its environment, obtained by the sensors with which it is equipped.

On command, the robot will have to adapt to best accomplish its task. For example, if it is given the instruction "Go 100 meters north!", it will have to use its programming to reach the rendezvous point, which is located 100 meters to the north.

Access the program online:
https://en.vittascience.com/l476/?link=6808ae1b5d57d&mode=blocks

Mission • 5 ⏱ 50 min
Moving in square mode



- **Mission 5-1 : Moving in square mode**

For this mission, we'll program the robot's movements in square mode using the buttons. The robot will make small movements as if it were on a chessboard or checkers board.
This will simplify programming.
For example, we can move 3 squares forward and 2 squares to the right.
To do this, we'll use blocks located in:

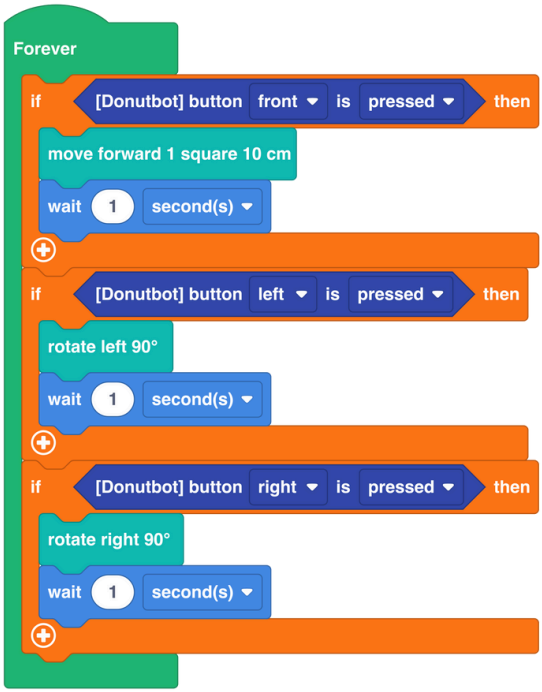
Robots Logic

Access the program online:
https://en.vittascience.com/l476/?link=6808af91dc594&mode=blocks

Another, more advanced method is to use functions.
To make things easier, we'll create three functions: one for moving forward, one for turning right, and one for turning left.
We'll use the previous missions with the 90° rotation, which is ideal for moving one square. We'll also measure how long it takes to move one square forward when moving in a straight line.

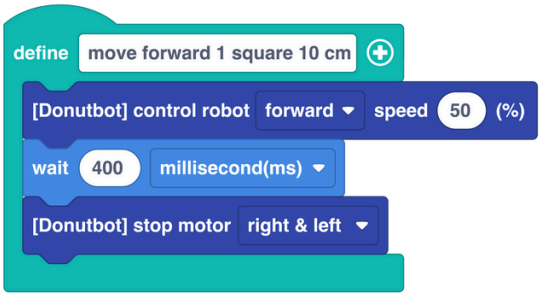
There are no major changes compared to the previous missions except that instead of putting the blocks in the "Repeat forever" or "On startup" loop, we will put them in a "Function" block which we will give a recognizable name so as not to get lost in the code. We still use a condition block so that the robot executes the function when the correct button is pressed, as well as a one-second delay so that we have time to press.

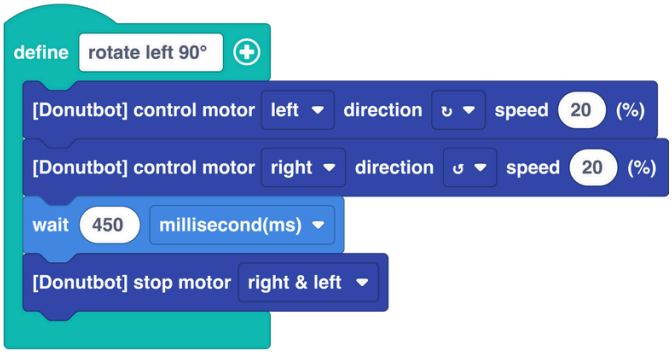
To do this, we will use blocks that are located in:

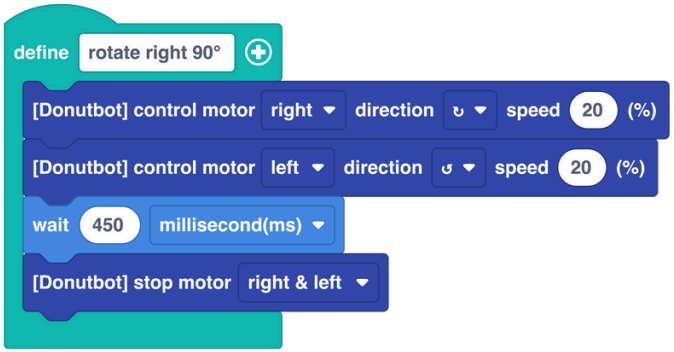


Functions simplify programming. They can be called from within the main program so they execute in one go.
Functions make the program more readable by preventing us from repeating lines of code in the main program many times.

Access the program online:
https://en.vittascience.com/l476/?link=6808d81eb0b4f&mode=blocks



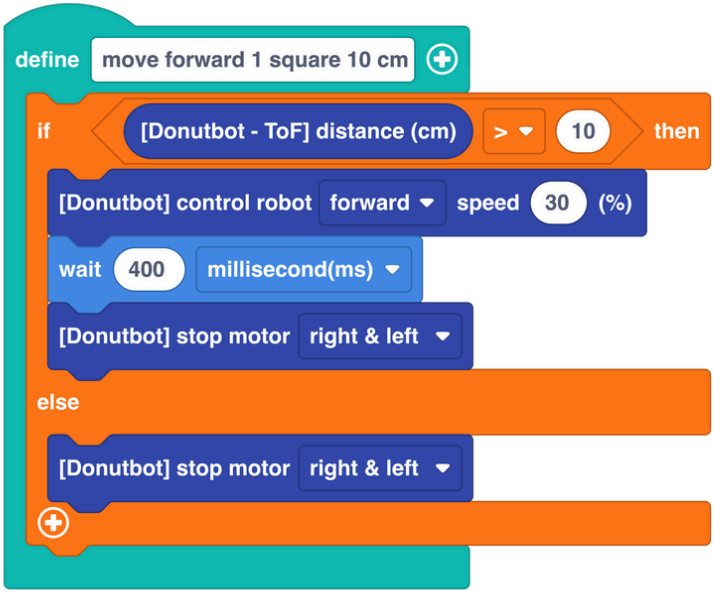




- **Mission 5-2 : Obstacle detection before movement**

Moving without looking where we're going can quickly cause problems. In everyday life, our 3D vision allows us to spot obstacles and assess their distance. For our robot, the Time of Flight sensor or the ultrasonic sensor will act as our eyes. The objective of this mission will be to move while checking before each move whether the destination square is free.
If not, we will decide not to move onto the cluttered square and display a flashing alert on the robot's LEDs. We will only modify the move function so that it detects obstacles. To do this, we will use blocks found in:

Here, only the changes in the “Move forward 10 cm” function are shown:



We have already made good progress, our robot is now able to move on its own, avoiding obstacles, or rather ignoring them!

Access the program online:
https://en.vittascience.com/l476/?link=6808d8631fb7e&mode=blocks

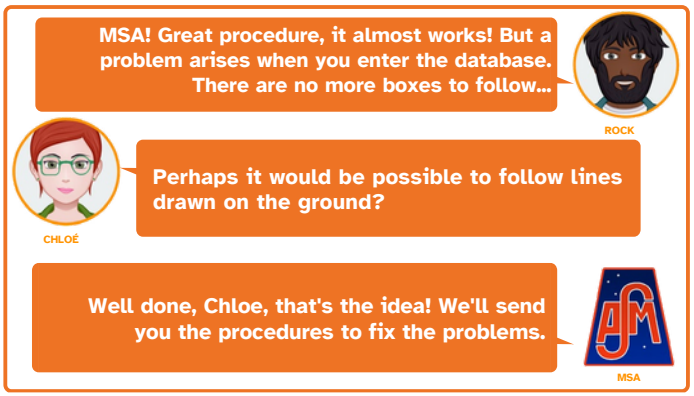
Mission • 6 ⏱ 50 min
Line follower

• Mission 6-1 : Follow a line

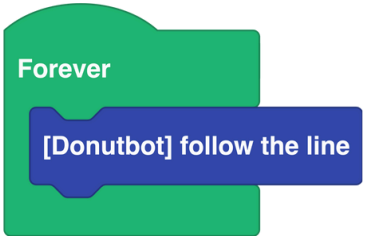
Line tracking is ensured by three color sensors located under the robot's chassis.

Four white LEDs emit light, and the color sensors act as phototransistors that detect the reflection of white light off the ground.

Since black absorbs the radiation, the receiver does not detect the transmitted signal.
When the sensor is above the line, it will return a value of 1. Otherwise, if the signal is reflected (e.g., off white), the sensor returns a value of 0.



To begin, we will use one of the blocks found in: Robots



Access the program online:
https://en.vittascience.com/l476/?link=6808d8c24aa31&mode=blocks

• Mission 6-2 : Follow a line with 3 sensors

Now let's try to break down the program behind the 'Follow Line' block.

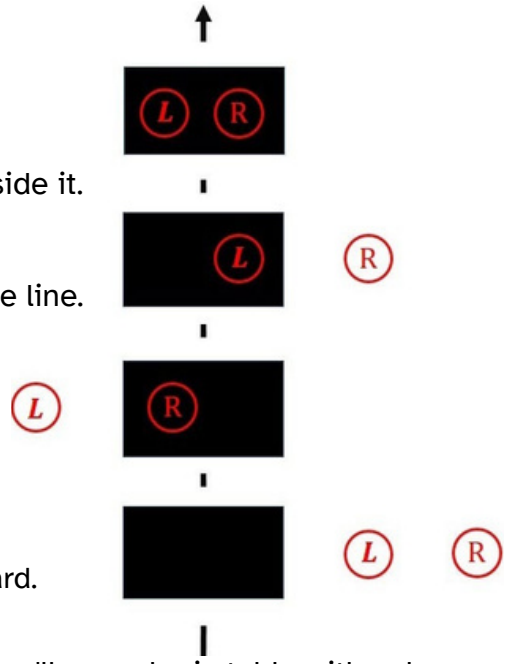
First of all, to make it easier to understand, here is what we would get if there were two sensors to follow the line with a robot.

• **Case 1:** Both sensors are above the black line.
→ Returns 1.1

• **Case 2:** The L sensor is above the line and the R sensor is outside it.
→ Returns 1.0

• **Case 3:** The L sensor is outside it and the R sensor is above the line.
→ Returns 0.1

• **Case 4:** Both sensors are outside the black line.
→ Returns 0.0



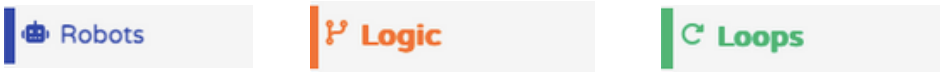
The figure opposite illustrates the operation of the sensors and the information that could be recovered from the NUCLEO-L476RG board.

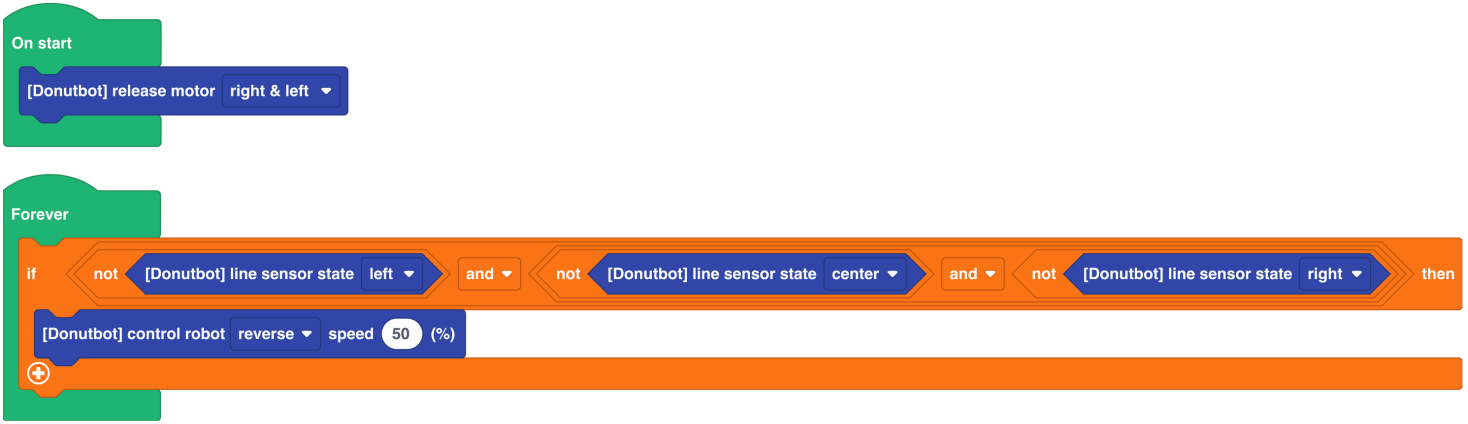
To make it easier to understand the DonutBot robot's 3 sensors, we'll use a logic table with columns for sensors and columns for motor operation. We'll use the code 1 for "no line / color = white" and 1 for "presence of line / color = black."

Left sensor	Central sensor	Right sensor	How engines work
0	0	0	Search the line
0	0	1	Rotate right
0	1	0	Move forward
1	0	0	Rotate left

We see that in the first case, the line is no longer detected and a strategy must be put in place to find it. This could be, for example, "move forward 5 cm then turn in circles until the line is detected" or simply "move back" because we went too fast and lost the line.

Let's program this! To do this, we will use blocks found in:

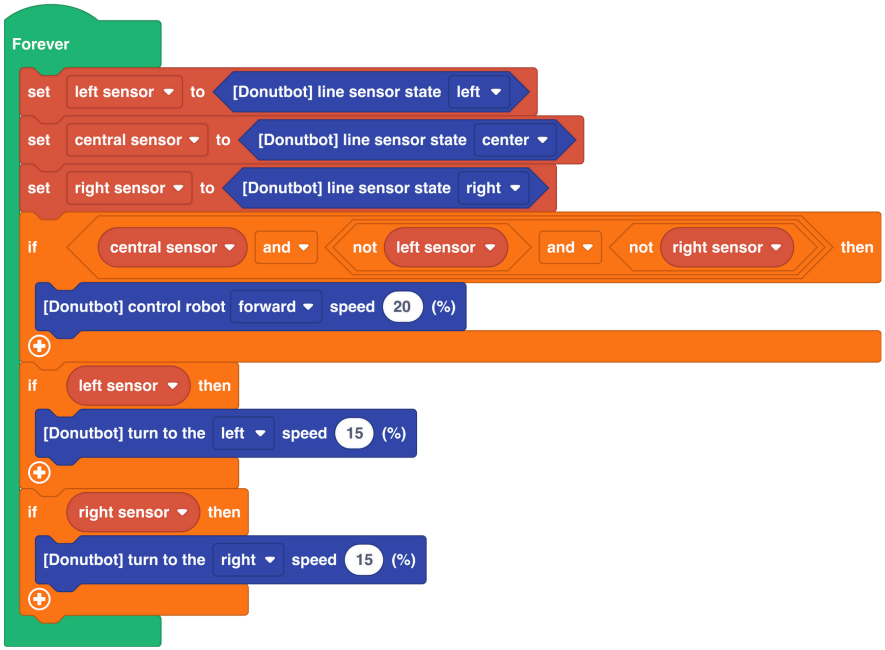




In the "On Start" block of the program, you can add an instruction "release the right and left motors."
This is a mini initialization that ensures that you start with the motor off. You can also add a pause to give you time to properly position the robot before it starts.

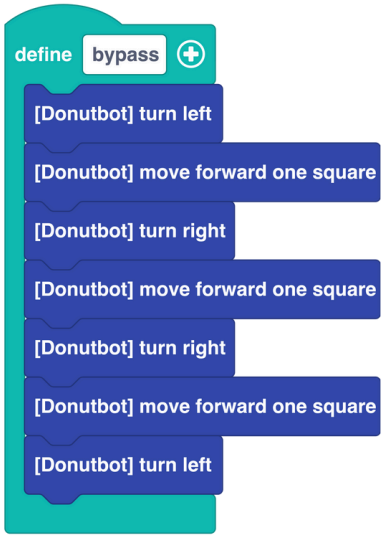
We observe that the program will become more substantial and will call upon the state of the sensor for each instruction. In order to simplify programming, we will use variables.

Access the program online:
https://en.vittascience.com/l476/?link=6808d9184a73b&mode=blocks

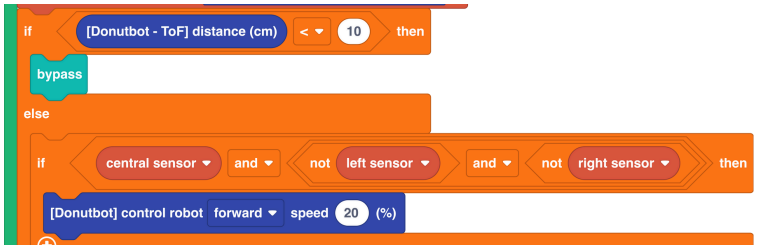


• Mission 6-3 : Follow a line and avoid obstacles

For this mission, we'll imagine the robot detects an object while following its line. In this scenario, it will have to stop, go around it, and then return to the line to continue behind the obstacle. We'll use functions like in mission 5: "Move forward 10 cm" and "Turn right" or "Turn left" to form an obstacle avoidance function. Let's try to set this up; we'll need the following blocks:



Only the bypass function is detailed opposite. The rest of the programming is identical to the previous mission.



Access the program online:

https://en.vittascience.com/l476/?link=6808d9bdcf2c8&mode=blocks

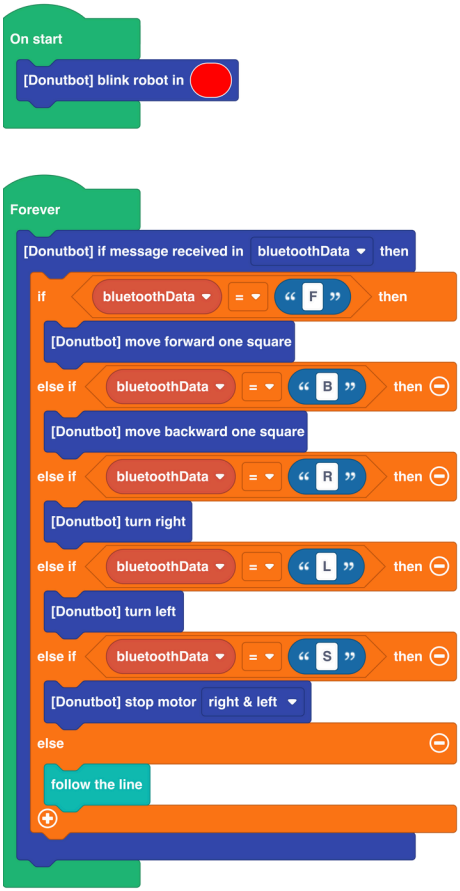
Mission • 7 ⏱ at the discretion of the supervisor

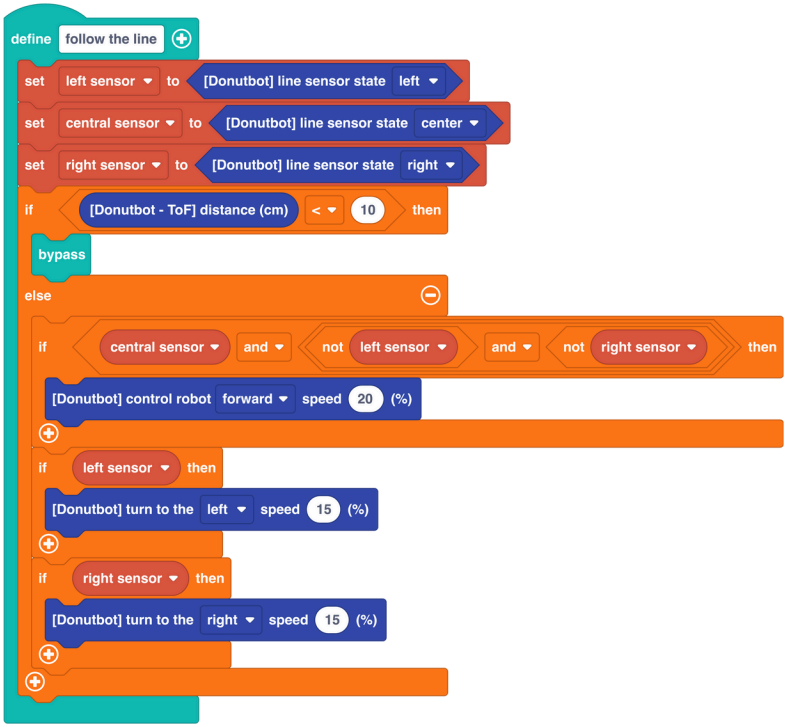
Finally the routine!

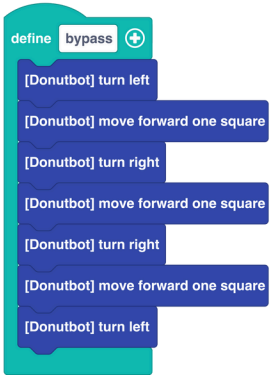
Complex situation:
The robot follows the loop continuously, but a sudden drop in temperature occurs. This event forces it to stop all activity and seek shelter.

To do this, we switch to manual control and plan a return to the Ares I center or to the specified coordinates.
Movements will be secured using the ToF sensor to verify that each movement is valid.
Remote control can be done via smartphone and Bluetooth module. Several types of movements will need to be planned.

The code is presented here in its entirety. While not overly complex, it is getting quite large!







Access the program online:

https://en.vittascience.com/l476/?link=6808da08bdcdd&mode=blocks

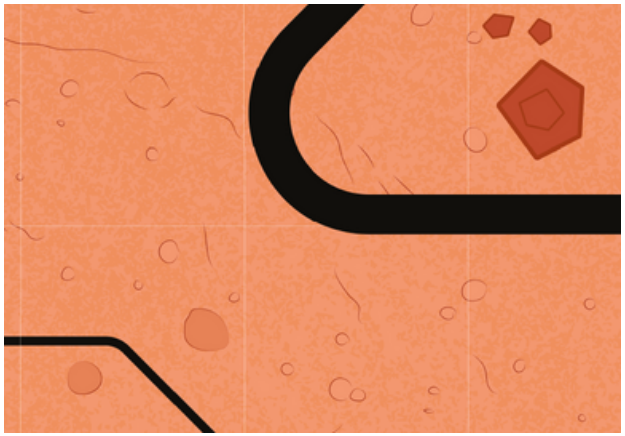
To go further

In order to complete the programming in this booklet, it is possible to add the following activities during the year or at other levels in parallel.

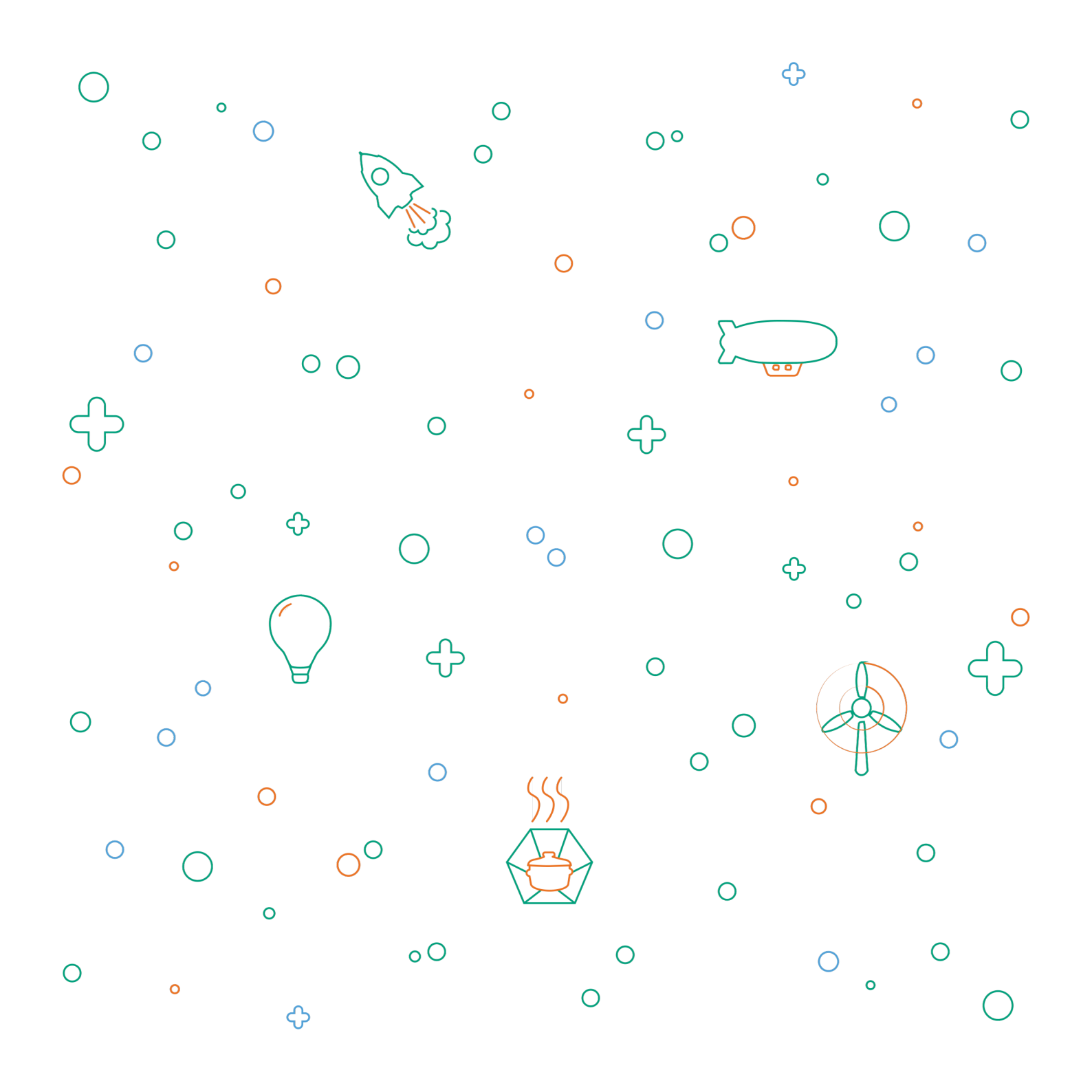
Creation of 3D objects:
- Mini crater that adapts to the path
- Rock
- Martian base
- Solar panel

A larger project would involve designing a base to house the robot. One could start with the idea of a garage with an automatic door, which would allow the use of a distance sensor to detect the robot's presence and control the opening or closing of the base's entrance airlock. This project would involve programming, as well as design and manufacturing using machine tools, 3D printers, or laser cutters.

To close the Martian robot track, print this sheet in A4 format and place it on the track.
Find it on the website https://en.vittascience.com/learn/tutorial.php?id=1399/



Notes



Here is the accompanying booklet for the Mars rover.
It details each step and each workshop required to complete the experiment. This booklet is not exhaustive; the imagination and resources available on the website are there to help you deepen your experience! This booklet is subject to change; we invite you to consult the online version, which is regularly updated at: www.vittascience.com/learn (search for "Martian rover" to find the user guide).

vitta science | life.augmented

STEM your way
Innovation depends on you

IPCEI on Microelectronics