API Library

1. Installation

Install the latest version of the library

```
pip install alphai-api
```

2. Start with Python

- Before start programing, remember to connect to the robot first with following steps: run AlphAI and use the grahical interface to connect either to the simulated robot or to a real robot (Wi-Fi or Bluetooth).
- After that, if you run the code below, you should notice that the software now displays "The robot is controlled by a Python program" in the graphical interface.

```
# module is called alphai
import alphai # OR from alphai import *



# You can control the robot to move with motor() now.
alphai.motor(20,-20,5) # Start turning on itself for 5 seconds
```

Example python script

```
import time
import alphai
alphai.open_api() # Sometimes necessary to ensure that the connection between
the robot and the Python program is established.
alphai.connect_wifi() # Connect to the robot's Wi-Fi network

time.sleep(1) # Wait for 1 second to ensure the connection is established

for i in range(50):
    if alphai.get_blockade(): # If robot is blocked.
        alphai.motor(20,-20,0.2) # Rotate slightly to the right.
    else:
        alphai.motor(30,30,0.3) # Go straight ahead.
```

3. Functions defined in this module

In this segment, we are showing all functions in the library with a short explaination code.

- <1> open_api() --> None | Normally automatically called when the alphai module is imported. It can establish the connection between your python program and the AlphAI software.
- <2> bluetooth_list() --> List[int] | Returns the list of AlphAI robot numbers known by the computer's Bluetooth system, to which it is possible to connect.

```
l = alphai.bluetooth_list() # Get the list of AlphAI robots known by the
computer's Bluetooth system.
print(l) # Print the list of AlphAI robots.
```

- Reminder: If you want to connect the robot with code, you always need to open up AlphAI software first.
- <3> connect_bluetooth(int) --> bool | Attempts a Bluetooth connection with the robot corresponding to the specified number.
- <4> disconnect() --> None | Disconnect from the robot and returns to the connection screen.

```
alphai.connect_bluetooth(l[0]) # Connect to the first robot in the list.
```

Before we disconnect, you could go to AlphAI software and see whether the interface is changed to the dashboard.

```
alphai.disconnect() # Disconnect from the robot and return to the connection
screen.
```

- <5> connect_wifi() --> bool | Tries to connect to the robot to which the computer is connected via Wi-Fi.
- Reminder: You must connect to the robot via Wi-Fi in advance (If your computer is not connected with a real robot's Wi-Fi, this code will connect your computer with the 2D simulator).
- By running this code, you could access to the Alphai software interface.

```
alphai.connect_wifi() # Connect to the robot via Wi-Fi.
```

Before we disconnect, you could go to AlphAI software and see whether the interface is changed to the dashboard.

```
alphai.disconnect() # Disconnect from the robot and return to the connection screen.
```

- <6> connect(str) --> bool | To connect to the simulated AlphAI robot, use the call: connect("Simulation"). Return True if the connection was successful, False otherwise.

```
alphai.connect("Simulation") # Connect to the simulated AlphAI robot.
```

Before we disconnect, you could go to AlphAI software and see whether the interface is changed to the dashboard.

```
alphai.disconnect() # Disconnect from the robot and return to the connection screen.
```

- <7> is_connected() --> bool | Returns True if a robot is connected, False otherwise

```
print(alphai.is_connected()) # Print True or False depending on connection status
```

- <8> motor(left,right,duration:"float") --> None | Control the robot to move: left: the speed of left motors; right: the speed of right motors; duration: time duration (in seconds); the speed is between -50 and 50.

```
alphai.connect("Simulation") # Connect to the simulated AlphAI robot.
alphai.motor(20,-20,5) # Start turning on itself for 5 seconds
```

```
alphai.disconnect() # Disconnect from the robot and return to the connection screen.
```

- <9> wait_for_key() --> List | Wait for user to press or release a key, and returns the list of keyboard keys that is preessed simultaneously

```
l = alphai.wait_for_key() # record the key pressed by the user.
```

```
print(l) # Print the list of keys pressed by the user.
```

Next part is the functions relating to sensors, so we connect to the robot in advance

```
import alphai
alphai.connect_wifi()
print("Is connected?", alphai.is_connected())
```

- <10> get_blockade() --> bool | Return True if the robot is blocked (by and obstacle), and False otherwise

```
print("Is the robot bloced?", alphai.get_blockade()) # Check if the robot is blocked
```

```
Is the robot bloced? False
```

- <11> set_distance(bool) --> None | Enables or disables the ultrasonic sensor.

```
alphai.set_distance(True) # Enable the ultrasonic sensor
```

```
alphai.set_distance(False) # Disable the ultrasonic sensor
```

- <12> get_distance() --> float | Returns the distance in centimetres measured by the ultrasonic sensor.

```
alphai.set_distance(True)
distance = alphai.get_distance() # Get the distance measured by the ultrasonic sensor
print("Distance:", distance) # Print the distance measured by the ultrasonic sensor
```

```
Distance: 66.06647034181991
```

```
alphai.set_distance(False)
```

- <13> set_camera(str) --> None | Activate camera with different resolution: "2x1","4x3","8x6","16x12","32x24","64x48". When the parameter is None, deactivates the camera.

```
alphai.set_camera("2x1") # Activate camera with 2x1 resolution
```

```
alphai.set_camera() # Deactivate camera
```

- <14> get_camera() --> List | Returns the image taken by the robot's camera in the form of an array of integers.

```
alphai.set_camera("2x1")
data = alphai.get_camera() # Get the image taken by the robot's camera
print("Camera data:", data) # Print the image data
```

```
Camera data: [[[235, 186, 190], [239, 208, 212]]]
```

- <15> set_ir(str) --> None | Enabes or disables the robot's infrared line-tracking sensors. The parameter must be "all" or "None".

```
alphai.set_ir("all") # Enable the infrared line-tracking sensors
```

```
alphai.set_ir("None") # Disable the infrared line-tracking sensors
```

- <16> get_infra_red() --> list | Returns the values measured by the robot's 5 infrared sensors in the form of an array of numbers between 0 and 1.

```
alphai.set_ir("all")
data = alphai.get_infra_red() # Get the values measured by the infrared sensors
print("Infrared data:", data) # Print the infrared sensor data
```

```
Infrared data: [1.0, 1.0, 0.9950207741146376, 1.0, 1.0]
```

- <17> get_all_sensors() --> List | Returns the values of all the robot's sensors in the form of a list.
- In the sequence of: obstacle, distance (in meters), infrared sensors, camera.

```
data = alphai.get_all_sensors() # Get the values of all the robot's sensors
print("All sensors data:", data) # Print the values of all the robot's sensors
```

```
All sensors data: [False, 0.660664703418199, array([1., 1., 1., 1., 1.]),
array([[[0.92335781, 0.73221566, 0.74499293],
        [0.93868713, 0.81724157, 0.83171083]]])]
```