ENGR 3421: ROBOTICS I

Python Advanced

Dr. Lin Zhang

Department of Physics and Astronomy
University of Central Arkansas

September 23, 2021

Outline

Outline

Object-Oriented Programming

Object Oriented programming (OOP) is a programming paradigm that relies on the concept of classes and objects. It is used to structure a software program into simple, reusable pieces of code blueprints (usually called classes), which are used to create individual instances of objects.

Function w/o Return

```
def forward(motor1, motor2, speed=1):
    """
    Args:
        motor1: object instantiate from Motor class
        motor2: object instantiate from Motor class
        speed: scalar in range [0,1]
    Return:
        None
    """
    motor1.set_speed(speed)
    motor2.set_speed(speed)
```

Function w/ Return

```
def compute_center(ul_coord, ur_coord, lr_coord, ll_coord):
    """
    Args:
        ul_coord: array with shape (2,) or list with length 2
        ur_coord: [x, y]
        lr_coord: e.g. array(321, 456)
        ll_coord: e.g. [321, 456]
    Return:
        center_coord: coordinate of center of the box represented
        by a list with length of 2.
    """
    mean_x = (ul_coord[0] + ur_coord[0] + lr_coord[0] + ll_coord[0]) / 4
    mean_y = (ul_coord[1] + ur_coord[1] + lr_coord[1] + ll_coord[1]) / 4
    center_coord = [mean_x, mean_y]

    return center_coord
```

Create a Class

```
# define a class
class Robot:
    def __init__(self, name, target):
        self.name = name
        self.target = target

# make an instance
bot = Robot(name='bouncer', target=0)
# check status of the instance
bot.name
bot.target
```

Create a Class

```
class Robot:
    def __init__(self, name, target):
        self.name = name
        self.target = target

    def is_marker_detected(self, image):
        flag = False
        (corners, ids, rejects) = cv2.aruco.detectMarkers(image, d, p)
        if self.target in ids:
            flag=True
        return flag

    def switch_target(self, new_target):
        self.target = new_target
# use functions
bot = Robot(name='bouncer', target=0)
im = np.random.uniform(0,255,(640,480,3))
print("Marker detected: {}".format(bot.is_marker_detected(im)))
bot.target
bot.switch_target(4)
bot.target
```

Inherit a Class

```
class NewRobot(Robot):

    def switch_target(self, new_target):
        assert new_target < 1000
        self.target = new_target

    def make_noise(self):
        print("Robot {} shouted: 'Huray!'".format(self.name))
# use functions
bot = NewRobot(name='bouncer', target=0)
im = np.random.uniform(0,255,(640,480,3))
print("Marker detected: {}".format(bot.is_marker_detected(img)))
bot.target
bot.switch_target(1000)
bot.switch_target(999)
bot.make_noise()
```

Outline

Source Code

- Read documentation.
- Read source code

Locate pip installed package:

```
pip3 show gpiozero
```

NumPy

NumPy is the fundamental package for scientific computing in Python. To install:

```
pip3 install numpy
```

NumPy Getting Started

What's the difference between a Python list and a NumPy array?

While a Python list can contain different data types within a single list, all of the elements in a NumPy array should be homogeneous. NumPy gives you an enormous range of fast and efficient ways of creating arrays and manipulating numerical data inside them.

```
import numpy as np
v = np.array([1, 2, 3, 4, 5, 6])
m = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
print(a[0])
print(m[0])
print(m[0,1])
l = m.tolist()
print(l[0,1])
print(l[0][1])
print(m[0][1])
```

NumPy Array Operations

```
np.zeros(4)
np.ones(4)
np.empty(4)
np.arange(4)
a = np.array([1, 2, 3, 4])
b = np.array([5, 6, 7, 8])
print(a + b) # try adding two list
print(a * b)
print(a / b)
print (a + 4)
```

NumPy Array Shape

```
a = np.arange(12)
b = a.reshape((3,4))
c = a.reshape((1,12))
print(a.shape)
print(b.shape)
print(c.shape)
print("a size:{}, b size: {}, c size: {}".format(a.size, b.size, c.size))
```

Plot NumPy Array

Install Matplotlib before running the following

```
import matplotlib.pyplot as plt
x = np.linspace(0, 5, 20)
y = np.linspace(0, 10, 20)
plt.plot(x, y, 'purple')
plt.plot(x, y, 'o')
plt.show()
```