1. Image scaling

1.1. OpenCV image scaling

cv2.resize(InputArray src, OutputArray dst, Size, fx, fy, interpolation)

Parameter meaning:

InputArray src: input image

OutputArray ds: output image

Size: output image size

fx, fy: scaling coefficients along the x-axis and y-axis

interpolation: interpolation method, you can choose INTER_NEAREST (nearest neighbor interpolation), INTER_LINEAR (bilinear interpolation (default setting)), INTER_AREA (resample using pixel area relationship), INTER_CUBIC (bicubic interpolation of 4x4 pixel neighborhood), INTER_LANCZOS4 (Lanczos interpolation of 8x8 pixel neighborhood)

Note:

1. The output size format is (width, height)
2. The default interpolation method is: bilinear interpolation

1.2. Actual effect display

Code path:

/home/pi/project_demo/06.Open_source_cv_fundamentals_course/B.Geometric_Transformations/01_Image_Scaling.ipynb

```
import cv2
import matplotlib.pyplot as plt # Python 2D drawing library

# Read the original image
img = cv2.imread('yahboom.jpg')
# Print the image size
print(img.shape)
# Assign the image height and width to x, y respectively
x, y = img.shape[0:2]

# Display the original image
#cv.imshow('OriginalPicture', img)

# Scale to half of the original size, the output size format is (width, height)
img_test1 = cv2.resize(img, (int(y / 2), int(x / 2)))
# cv2.imshow('resize0', img_test1)
```

```
# cv2.waitKey()

# Nearest neighbor interpolation scaling
# Scale to one-fourth of the original
img_test2 = cv2.resize(img, (0, 0), fx=0.25, fy=0.25,
interpolation=cv2.INTER_NEAREST)
# cv.imshow('resize1', img_test2)
# cv.waitKey()
# cv.destroyAllWindows()
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
dst1 = cv2.cvtColor(img_test1, cv2.COLOR_BGR2RGB)
dst2 = cv2.cvtColor(img_test2, cv2.COLOR_BGR2RGB)

# Display original image
plt.imshow(img)
plt.show()
```

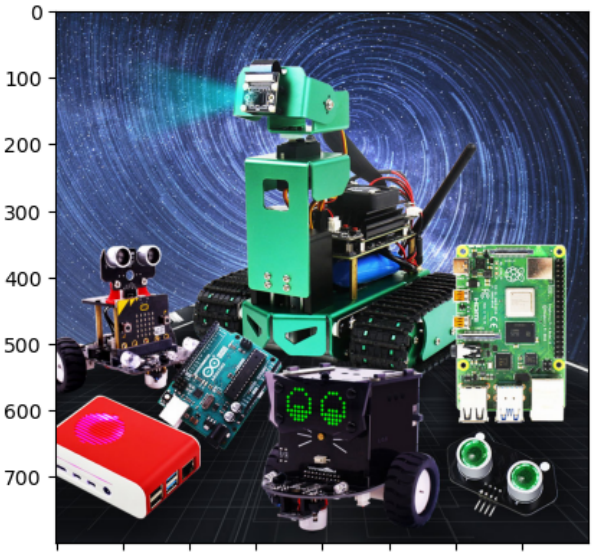
01_Image_Scaling.ipynb

Code

```
# 最近邻插值法缩放 Nearest Neighbor Scaling
# 缩放到原来的四分之一 Zoom to one-quarter of its original size
img_test2 = cv2.resize(img, (0, 0), fx=0.25, fy=0.25, interpolation=cv2.INTER_NEAREST)
# cv.imshow('resize1', img_test2)
# cv.waitKey()
# cv.destroyAllWindows()
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
dst1 = cv2.cvtColor(img_test1, cv2.COLOR_BGR2RGB)
dst2 = cv2.cvtColor(img_test2, cv2.COLOR_BGR2RGB)

# 显示原始图像 Display original image
plt.imshow(img)
plt.show()
```

```
(800, 800, 3)
```

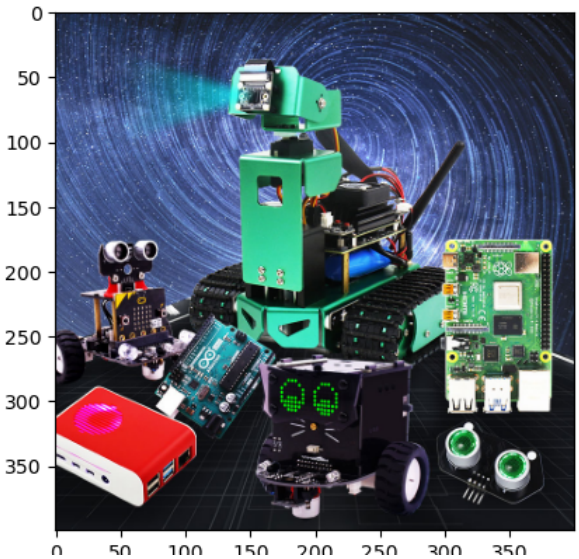



```
# Display zoom 1/2
plt.imshow(dst1)
plt.show()
```

01_Image_Scaling.ipynb

Code

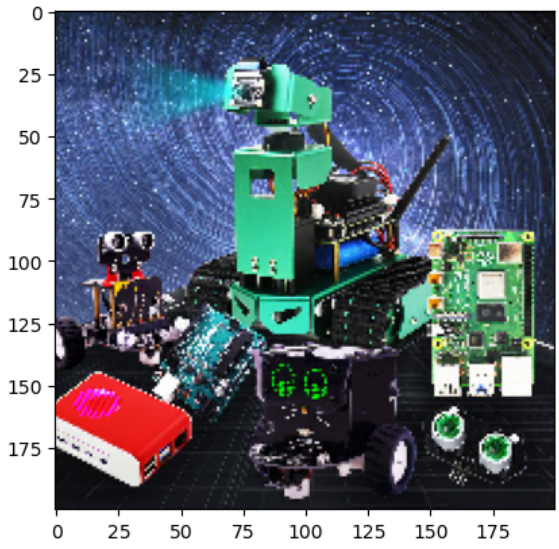
[2]:
```
# 显示缩放1/2 Display Zoom 1/2
plt.imshow(dst1)
plt.show()
```




```
# Display zoom 1/4 neighbor interpolation method zoom
plt.imshow(dst2)
plt.show()
```

Code

[3]:
```
# 显示缩放1/4 邻插值法缩放 Display zoom 1/4 neighbor interpolation zoom
plt.imshow(dst2)
plt.show()
```



1.3.matplotlib: Python's 2D drawing library.

The following is a small example of matplotlib

Reference tutorial: https://www.runoob.com/numpy/numpy-matplotlib.html

```
import numpy as np
from matplotlib import pyplot as plt

x = np.arange(1,11)
y = 2 * x + 5
plt.title("Matplotlib demo")
plt.xlabel("x axis caption")
plt.ylabel("y axis caption")
plt.plot(x,y)
plt.show()
```

Code

The following is a small example of matplotlib Reference tutorial:
https://www.runoob.com/numpy/numpy-matplotlib.html

[4]:
```
import numpy as np
from matplotlib import pyplot as plt

x = np.arange(1,11)
y =  2 * x +  5
plt.title("Matplotlib demo")
plt.xlabel("x axis caption")
plt.ylabel("y axis caption")
plt.plot(x,y)
plt.show()
```