Send data

1. API Introduction

Arm object initialization

```
mc = MyCobot("/dev/ttyUSB0", 1000000)
```

Among them, the first parameter ["/dev/ttyUSB0"] is the control serial port device number, which can be modified according to actual conditions. The second parameter [1000000] is the serial port communication baud rate, which cannot be changed.

The API related to controlling the robotic arm is:

send_angle(id, degree, speed)

Function: Control the angle of a joint of the robotic arm.

Parameter explanation:

- [id]: The input range of id is 1~6, corresponding to the joints from bottom to top of the robotic arm.
- [degree]: Servo angle value. The angle range of each servo is different, please check the list below.

Servo ID	Minimum value	Maximum value
1	-168°	168°
2	-135°	135°
3	-150°	150°
4	-145°	145°
5	-165°	165°
6	-180°	180°

- [speed]: Controls the running speed, the range is [1-100], the larger the value, the faster the movement.
- Return value: None.

send_angles(degrees, speed)

Function explanation: Controls the angles of the six joints of the robot arm.

Parameter explanation:

- [degrees]: List of six joint angles, for example, degrees = [0.43, -0.52, 0.17, -1.14, -0.35, -45.08].
- [speed]: Controls the running speed, the range is [1-100]. The larger the value, the faster the movement.
- Return value: None

send_coord(id, coord, speed)

Function explanation: Controls the robot to move to a certain axis.

Parameter explanation:

- [id]: The input range of id is 1~3, corresponding to the robot coordinate axis, 1: X axis, 2: Y axis, 3: Z axis.
- [coord]: Robot coordinate parameter, indicating the input range, please see the list below.

Coordinate	Minimum	Maximum
x	-281 mm	281 mm
y	-281 mm	281 mm
z	-70 mm	412 mm

- [speed]: Controls the running speed, the range is [1-100], the larger the value, the faster the movement.

send_coords(coords, speed)

Function explanation: Controls the robot arm to move toward a certain coordinate.

Parameter explanation:

- [coords]: Robot arm coordinate parameters [x, y, z, rx, ry, rz]. The three parameters x, y, z represent the coordinates of the end of the robot arm, and rx, ry, rz represent the direction of the gripper.

Coordinate	Minimum	Maximum
x	-281 mm	281 mm
y	-281 mm	281 mm
z	-70 mm	412 mm
rx	-180°	180°
ry	-180°	180°
rz	-180°	180°

- [speed]: Controls the running speed, the range is [1-100], the larger the value, the faster the movement.
- Return value: None

set_encoder(id, encoder, speed)

Function: Controls the potential value of a joint of the robot arm.

Parameter explanation:

- [id]: The input range of id is 1~6, corresponding to the joints from bottom to top of the robot arm.
- [encoder]: Servo potential value, value range: 0~4096.

- [speed]: Control running speed, range is [1-100], the larger the value, the faster the movement.
- Return value: None.

set_encoders(encoders, speed)

Function: Control the potential value of the six joints of the robot arm.

Parameter explanation:

- [encoders]: List of potential values of the robot arm joints, for example, encoders = [2048, 2048, 2048, 2048, 2048, 2048].
- [speed]: Control running speed, range is [1-100], the larger the value, the faster the movement.
- Return value: None.

send_radians(radians, speed)

Function: Control the radian value of the six joints of the robot arm.

Parameter explanation:

- [radians]: List of robot joint radian values, for example, radians = [0.008, -0.009, 0.003, -0.02, -0.006, -0.787].
- [speed]: Control running speed, the range is [1-100]. The larger the value, the faster the movement.
- Return value: None.

set_gripper_value(gripper, speed)

Function: Control the release and clamping of the robot gripper.

Parameter explanation:

- [gripper]: The value of the robot joint gripper, the range is: 0~100. 100 is the maximum opening angle.
- [speed]: Control running speed, the range is [1-100]. The larger the value, the faster the movement.
- Return value: None.

2. Code content

Code path: ~/jetcobot_ws/src/jetcobot_ctrl/scripts/send_data.ipynb

```
#!/usr/bin/env python3
#coding=utf-8
import os
import time
from pymycobot.mycobot import MyCobot
from pymycobot.genre import Angle
from pymycobot.genre import Coord
```

Initialize the robot object.

```
mc = MyCobot('/dev/ttyUSB0', 1000000)
speed = 50
```

Control joint 1 to 90 degrees.

```
id = Angle.J1.value
degree = 90
mc.send_angle(id, degree, speed)
```

Control the angles of the six joints of the robot.

```
degrees = [0.43, -0.52, 0.17, -1.14, -0.35, -45.08]
mc.send_angles(degrees, speed)
```

Control the Z axis of the robot. Note: If the current robot posture coordinates are unreachable, the setting fails and the robot will not move.

```
id = Coord.Z.value
coord = 350
mc.send_coord(id, coord, speed)
```

Control the robot to move to a specific coordinate point. Note: If the given coordinates are unreachable, the setting fails and the robot will not move.

```
coords = [49.6, -63.3, 419, -92.11, -45.07, -88.41]
mc.send_coords(coords, speed)
```

Control the potential value of the robot arm. It is rarely used in general.

```
id = Angle.J1.value
encoder = 1500
mc.set_encoder(id, encoder, speed)
```

```
encoders = [2048, 2048, 2048, 2048, 2048, 2048]
mc.set_encoders(encoders, speed)
```

Control the radian value of the robot arm.

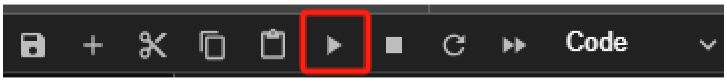
```
radians = [0.008, -0.009, 0.003, -0.02, -0.006, -0.787]
mc.send_radians(radians, speed)
```

Control the gripper angle of the robot arm.

```
gripper = 100
mc.set_gripper_value(gripper, speed)
```

3. Run the program

Click the run button on jupyterlab to run the relevant program.


Code