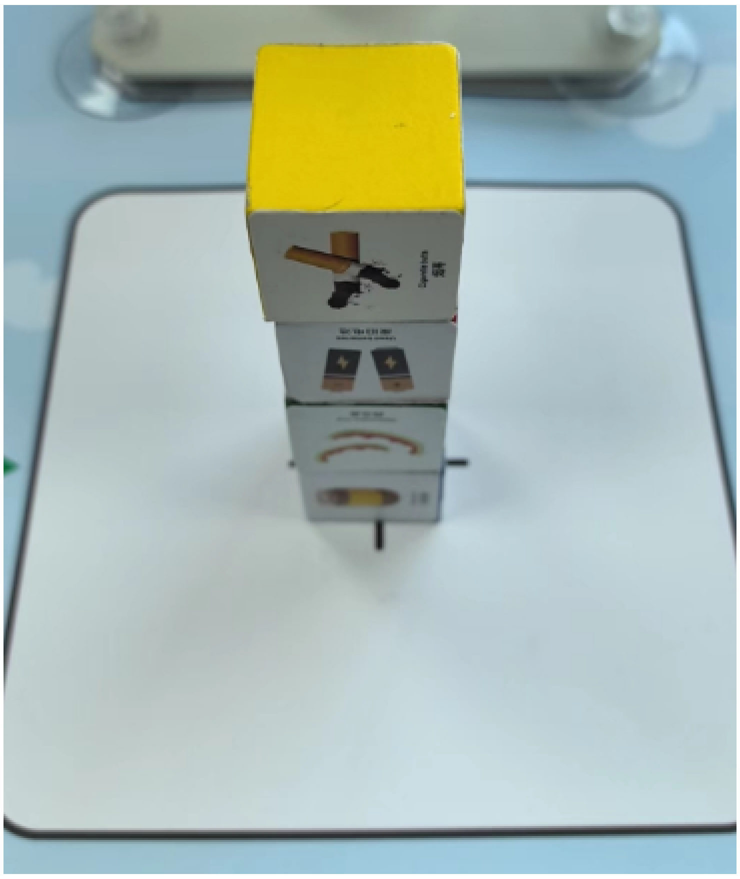
Nature Porter

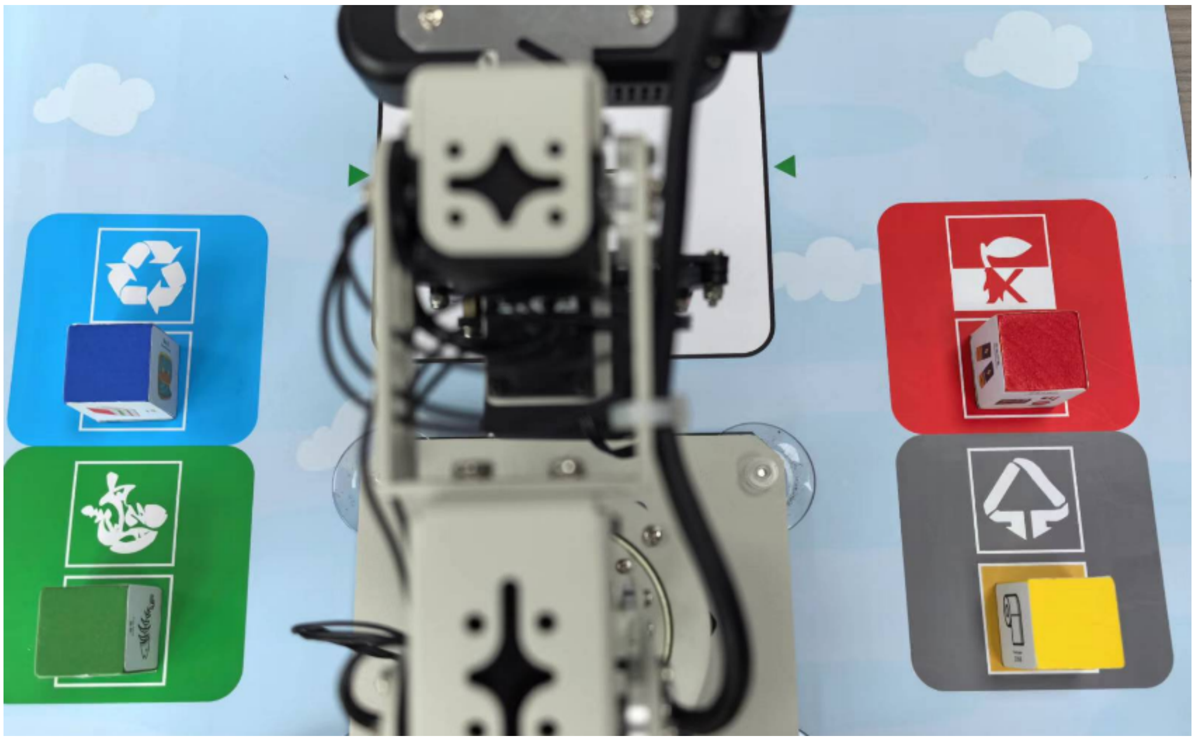
1. Introduction to the game

The purpose of this experiment is to stack four blocks of different colors from bottom to top in the order of blue, green, red and yellow and place them on the cross block in the middle. Then run the code, and the robot arm will follow the order of grabbing the block of the fourth layer and placing it in the yellow area, grabbing the block of the third layer and placing it in the red area, grabbing the block of the second layer and placing it in the green area, and grabbing the block of the bottom layer and placing it in the blue area.

The way to place the building blocks is as shown in the figure below:



After executing the code, the robot arm will move the building blocks to the corresponding position, and the final effect is as shown in the figure below:



2. Code content

Code path:

```
~/dofbot_ws/src/dofbot_ctrl/scripts/10.move_block.ipynb
```

```
#!/usr/bin/env python3
#coding=utf-8
import time
from Arm_Lib import Arm_Device

# Create a robot object
Arm = Arm_Device()
time.sleep(.1)

from dofbot_utils.robot_controller import Robot_Controller
robot = Robot_Controller()
```

```
# Define the function of clamping blocks, enable=1: clamp, =0: release
def arm_clamp_block(enable):
    if enable == 0:
        Arm.Arm_serial_servo_write(6, 60, 400)
    else:
        Arm.Arm_serial_servo_write(6, 135, 400)
    time.sleep(.5)


# Define the function of moving the robot arm and control the movement of servos
1-5 at the same time, p=[S1,S2,S3,S4,S5]
def arm_move(p, s_time = 500):
    for i in range(5):
        id = i + 1
        if id == 5:
            time.sleep(.1)
            Arm.Arm_serial_servo_write(id, p[i], int(s_time*1.2))
        else :
            Arm.Arm_serial_servo_write(id, p[i], s_time)
        time.sleep(.01)
    time.sleep(s_time/1000)

# The robot moves up
def arm_move_up():
    Arm.Arm_serial_servo_write(2, 90, 1500)
    Arm.Arm_serial_servo_write(3, 90, 1500)
    Arm.Arm_serial_servo_write(4, 90, 1500)
    time.sleep(.1)
```

```
# Define variable parameters for different positions
p_mould = robot.P_LOOK_AT
p_top = robot.P_TOP
p_Brown = robot.P_CENTER
p_Yellow = robot.P_YELLOW
p_Red = robot.P_RED
p_Green = robot.P_GREEN
p_Blue = robot.P_BLUE
p_layer_4 = robot.P_CENTER_4
p_layer_3 = robot.P_CENTER_3
p_layer_2 = robot.P_CENTER_2
p_layer_1 = robot.P_CENTER
```

```
# Move the robot arm to a position ready for grasping
arm_clamp_block(0)
arm_move(p_mould, 1000)
time.sleep(1)
```

```
# Move the fourth layer of blocks to the yellow area
arm_move(p_top, 1000)
arm_move(p_layer_4, 1000)
arm_clamp_block(1)
arm_move(p_top, 1000)
arm_move(p_Yellow, 1000)
arm_clamp_block(0)
time.sleep(.1)
arm_move_up()
arm_move(p_mould, 1100)
```

```
# Move the third layer of blocks to the red area
arm_move(p_top, 1000)
arm_move(p_layer_3, 1000)
arm_clamp_block(1)
arm_move(p_top, 1000)
arm_move(p_Red, 1000)
arm_clamp_block(0)
time.sleep(.1)
arm_move_up()
arm_move(p_mould, 1100)
```

```
# Move the second layer of blocks to the green area
arm_move(p_top, 1000)
arm_move(p_layer_2, 1000)
arm_clamp_block(1)
arm_move(p_top, 1000)
arm_move(p_Green, 1000)
arm_clamp_block(0)
time.sleep(.1)
arm_move_up()
arm_move(p_mould, 1100)
```

```
# Move the first layer of blocks to the blue area
arm_move(p_top, 1000)
arm_move(p_layer_1, 1000)
arm_clamp_block(1)
arm_move(p_top, 1000)
arm_move(p_Blue, 1000)
arm_clamp_block(0)
time.sleep(.1)
arm_move_up()
arm_move(p_mould, 1100)
```

```
del Arm # Release the Arm object
```

Open the program file from jupyter lab and click the Run the entire notebook button on the jupyter lab toolbar to see the effect of the robot arm's natural transporter function.