11. AprilTag tag code follow

11.1 Introduction

Apriltag tag code tracking is based on apriltag tag code recognition, adding the function of car body movement, controlling the car body so that the camera will keep the tag code moving left and right in the middle of the vision, and based on the principle that objects are larger when near and smaller when far in the camera image, the tag code following function is realized.

11.2 Startup

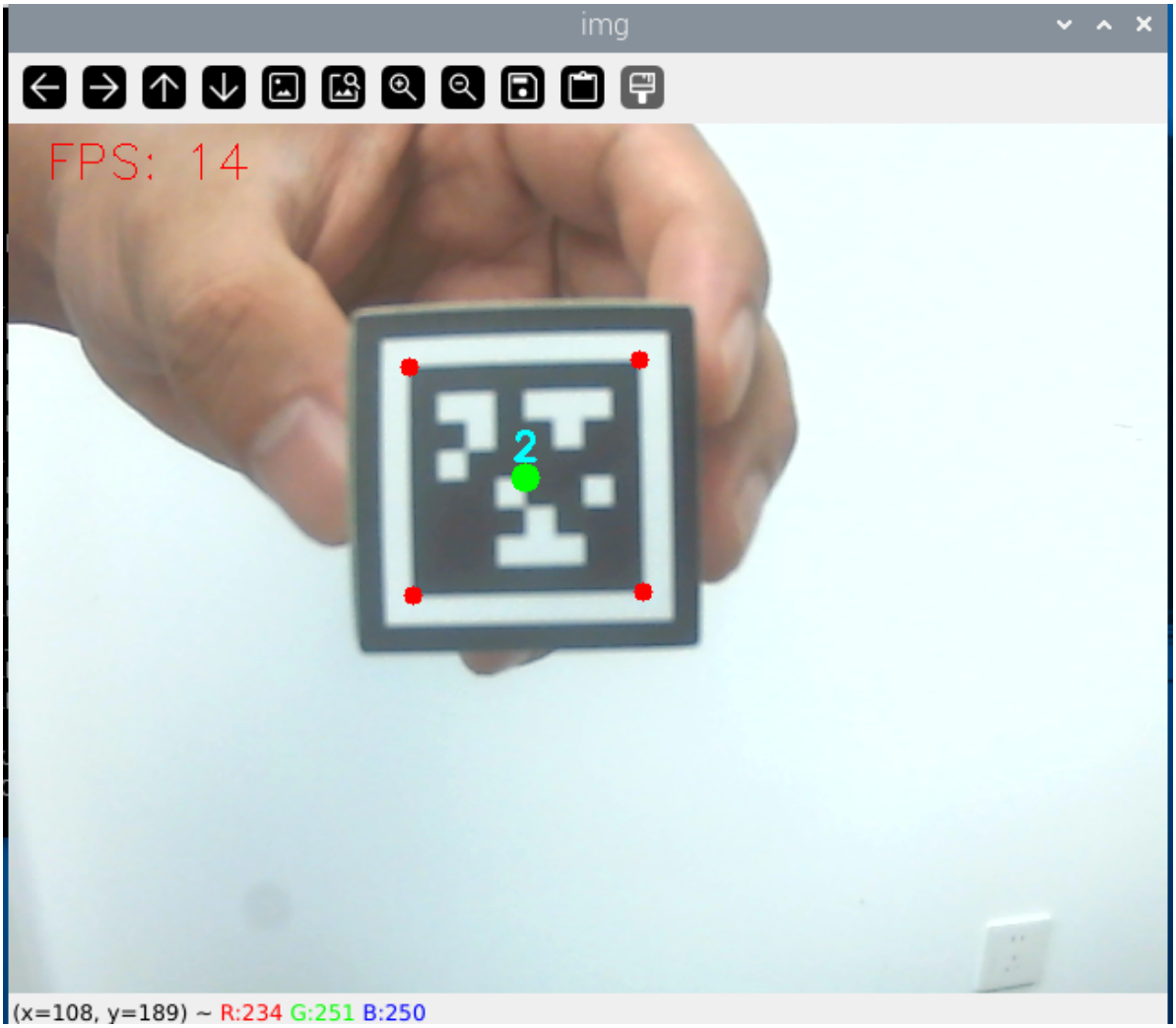
11.2.1 Preparation before program startup

This apriltag tag code uses TAG36H11 format, and the factory has equipped the relevant tag code and attached it to the building block. You need to take out the building block and place it on the camera screen for recognition.

11.2.2 Program description

After the program is started, the camera captures the image, puts the tag code into the camera screen, and the system will recognize and frame the four vertices of the tag code and display the ID number of the tag code. Then slowly move the position of the building block, and the car will move forward and backward, left and right with the building block.

Note: When the building blocks move, the label code should face the camera, and the movement speed should not be too fast to prevent the camera pan/tilt from failing to keep up.



11.2.3, Program Startup

Open a terminal and enter the following command to enter docker,

```
./docker_ros2.sh
```

The following interface appears, indicating that you have successfully entered docker

```
pi@yahboom:~ $ ./docker_ros2.sh
access control disabled, clients can connect from any host
root@yahboom:/#
```

Enter the following command in the docker terminal to start the program

```
ros2 launch yahboomcar_apriltag apriltag_follow.launch.py
```

11.2.4, Source code

After entering the docker container, the source code of this function is located at,

Code path:
/root/yahboomcar_ws/src/yahboomcar_apriltag/yahboomcar_apriltag/apriltag_follow.py

```
import rclpy
from rclpy.node import Node
from sensor_msgs.msg import Image
from cv_bridge import CvBridge
```

```
import cv2
import yahboomcar_apriltag.fps as fps
import numpy as np
from yahboomcar_apriltag.vutils import draw_tags
from dt_apriltags import Detector
from yahboomcar_apriltag.PID import PositionalPID
from yahboomcar_apriltag.common import *
from Raspbot_Lib import Raspbot
import math

class TagFollowNode(Node):
    def __init__(self):
        super().__init__('tag_follow_node')
        self.twist = Twist()
        self.pub_cmdVel = self.create_publisher(Twist,"/cmd_vel",1)
        self.h=0
        self.scale = 1000
        self.FollowLinePID = (60, 0, 20)
        self.linear = 0.05
        self.PID_init()
        # 初始化 Raspbot 实例
        self.bot = Raspbot()
        self.fps = fps.FPS()  # 帧率统计器
        self.bridge = CvBridge()
        self.yservo_pid = PositionalPID(0.8, 0.6, 0.01)  # PID控制器用于Y轴
        self.numx=self.numy=1
        target_servox = 90
        target_servoy = 25
        self.bot.Ctrl_Servo(1,target_servox)
        self.bot.Ctrl_Servo(2,target_servoy)
        self.at_detector = Detector(searchpath=['apriltags'],
                                    families='tag36h11',
                                    nthreads=8,
                                    quad_decimate=2.0,
                                    quad_sigma=0.0,
                                    refine_edges=1,
                                    decode_sharpening=0.25,
                                    debug=0)
        self.subscription = self.create_subscription(
            Image,
            '/image_raw',
            self.image_callback,
            100)
        self.subscription

    def image_callback(self, ros_image):
        # cv_bridge
        try:
            cv_image = self.bridge.imgmsg_to_cv2(ros_image,
desired_encoding='bgr8')
        except Exception as e:
            self.get_logger().error(f"Failed to convert image: {e}")
            return

        # 使用 AprilTags 检测器
```

```
        tags = self.at_detector.detect(cv2.cvtColor(cv_image,
cv2.COLOR_BGR2GRAY), False, None, 0.025)
        tags = sorted(tags, key=lambda tag: tag.tag_id)

        # 绘制标签
        result_image = draw_tags(cv_image, tags, corners_color=(0, 0, 255),
center_color=(0, 255, 0))


        # 处理 AprilTags
        if len(tags) == 1:
            _, y, _, self.h = tags[0].bbox
            center_x =tags[0].center[0]
            if math.fabs(180 - (y + self.h/2)) > 20: #调试方块半径    Debug Block
Radius
                self.yservo_pid.SystemOutput = y + self.h/2
                self.yservo_pid.SetStepSignal(220)
                self.yservo_pid.SetInertiaTime(0.01, 0.1)
                target_valuey = int(650+self.yservo_pid.SystemOutput)
                target_servoy = int((target_valuey)/10)
                if target_servoy > 110:
                    target_servoy = 110
                if target_servoy < 0:
                    target_servoy = 0
                #self.bot.Ctrl_Servo(2, target_servoy)
            self.twist.linear.x = self.linear
            [z_Pid, _] = self.PID_controller.update([((center_x) - 320)*1.0/16,
0])
            self.twist.angular.z = +z_Pid
            #self.get_logger().info('center = {} angular =
{}'.format([(center_x)],[self.twist.angular.z]))
            #self.get_logger().info('h = {} '.format([h]))
            if(80<self.h<110):#调试目标半径70~90 Debug target radius 70~90
                self.twist.angular.z =0.0
                self.twist.linear.x= 0.0
            elif(self.h>110):#调试目标半径90 Debug target radius 90
                self.twist.linear.x= -self.linear
            elif(20<self.h<80):
                self.twist.linear.x= self.linear
            else:
                self.twist.angular.z =0.0
                self.twist.linear.x= 0.0
        else:
            self.twist.angular.z =0.0
            self.twist.linear.x= 0.0
        self.get_logger().info('h = {}linear = {} angular ={}'.format([self.h],
[self.twist.linear.x],[self.twist.angular.z]))
        #print(self.twist.linear.x,self.twist.angular.z)
        self.pub_cmdVel.publish(self.twist)

        # 更新并显示 FPS
        self.fps.update()
        self.fps.show_fps(result_image)
        cv2.imshow("result_image", result_image)
        key = cv2.waitKey(1)
```

```
        if key != -1:
            cv2.destroyAllWindows()

    def PID_init(self):
        self.PID_controller = simplePID(
            [0, 0],
            [self.FollowLinePID[0] / 1.0 / (self.scale), 0],
            [self.FollowLinePID[1] / 1.0 / (self.scale), 0],
            [self.FollowLinePID[2] / 1.0 / (self.scale), 0])

def main(args=None):
    rclpy.init(args=args)

    tag_follow_node = TagFollowNode()

    try:
        rclpy.spin(tag_follow_node)
    except KeyboardInterrupt:
        tag_follow_node.bot.Ctrl_Servo(1, 90)
        tag_follow_node.bot.Ctrl_Servo(2, 25)
        pass

if __name__ == '__main__':
    main()
```