# 10. AprilTag tag code tracking



## 10.1. Introduction

Apriltag tag code tracking is based on apriltag tag code recognition, and adds the function of the car camera pan-tilt movement. The camera will keep the tag code in the middle of the vision and move. Based on this feature, the tag code tracking function is realized.

## 10.2. Startup

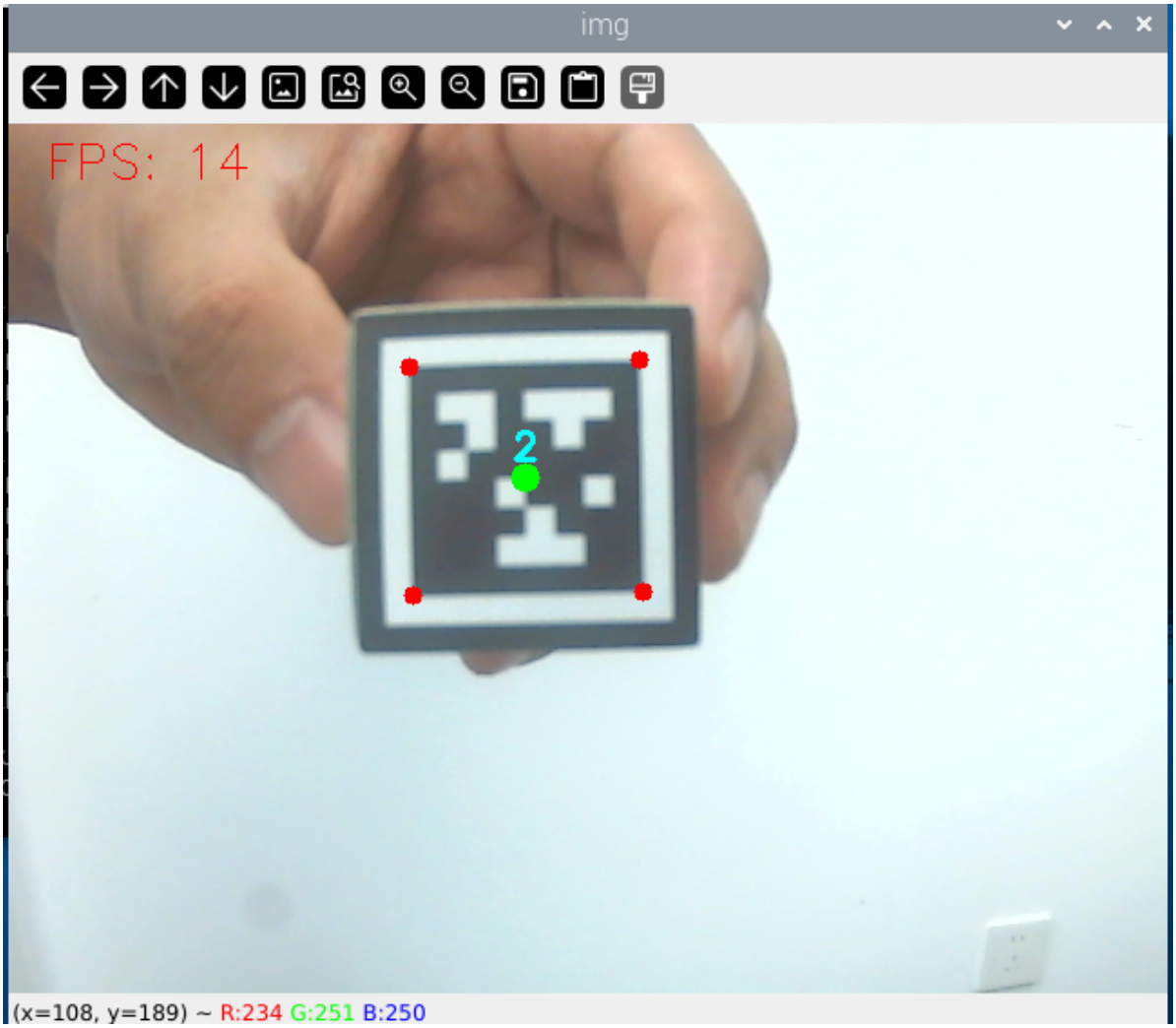
### 10.2.1. Preparation before program startup

This apriltag tag code uses the TAG36H11 format. The factory has equipped the relevant tag code and attached it to the building block. The building block needs to be taken out and placed on the camera screen for recognition.

### 10.2.2. 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 camera pan-tilt will move with the building block.

**Note: When the building block moves, the tag code should face the camera, and the moving speed should not be too fast to avoid the camera pan-tilt not keeping up.**




### 10.2.3, Program Startup

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

```
./docker_ros2.sh
```

The following interface appears, which means 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_tracking.launch.py
```

### 10.2.4, 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_tracking.py

```
#!/usr/bin/env python3
# encoding: utf-8
import cv2 as cv
import time
```

```python
from dt_apriltags import Detector
from yahboomcar_apriltag.vutils import draw_tags
import logging
import yahboomcar_apriltag.logger_config as logger_config
import rclpy
from rclpy.node import Node
from std_msgs.msg import String, Float32MultiArray

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 Raspbot_Lib import Raspbot
import math

class TagTrackingNode(Node):
    def __init__(self):
        super().__init__('tag_tracking_node')
        # 初始化 Raspbot 实例
        self.bot = Raspbot()
        self.bridge = CvBridge()
        self.xservo_pid = PositionalPID(0.6, 0.2, 0.01)  # PID控制器用于X轴
        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.fps = fps.FPS()  # 帧率统计器

        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:
            x, y, w, h = tags[0].bbox
            if math.fabs(180 - (x + w/2)) > 20:#调试方块半径    Debug Block Radius
                self.xservo_pid.SystemOutput = x + w/2
                self.xservo_pid.SetStepSignal(350)
                self.xservo_pid.SetInertiaTime(0.01, 0.1)
                target_valuex =  int(1000+self.xservo_pid.SystemOutput)
                target_servox = int((target_valuex)/10)
                #self.get_logger().info('x = {}'.format([x + w/2]))
                #self.get_logger().info('joints_x = {}
{}'.format([target_servox],[target_valuex]))
                if target_servox > 180:
                    target_servox = 180
                if target_servox < 0:
                    target_servox = 0
                self.bot.Ctrl_Servo(1, target_servox)

            if math.fabs(180 - (y + h/2)) > 20: #调试方块半径    Debug Block Radius
                self.yservo_pid.SystemOutput = y + 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)
                #self.get_logger().info('joints_y = {}
{}'.format([target_servoy],[target_valuey]))
                if target_servoy > 110:
                    target_servoy = 110
                if target_servoy < 0:
                    target_servoy = 0
                self.bot.Ctrl_Servo(2, target_servoy)

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

    tag_tracking_node = TagTrackingNode()

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

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