1.1概述

该脚本旨在使用OpenMV摄像头检测AprilTag，计算其位置和姿态，并通过UART通信发送这些信息。该脚本适用于机器人和其他嵌入式系统中，使用视觉标签进行定位和导航。

1.2依赖库

该脚本依赖以下Python库：

sensor：用于摄像头初始化和图像捕获。

image：用于图像处理和AprilTag检测。

time：用于测量帧率。

math：用于数学计算，包括角度转换。

pyb：用于UART通信。

ustruct：用于处理二进制数据（尽管代码中未直接使用）。

1.3摄像头设置

重置与配置：

sensor.reset()初始化摄像头。

sensor.set\_pixformat(sensor.RGB565)设置像素格式为RGB565。

sensor.set\_framesize(sensor.QVGA)设置分辨率为QVGA（320x240）。

跳过帧：

sensor.skip\_frames(30)跳过前30帧以让摄像头适应光照条件。

禁用自动功能：

sensor.set\_auto\_gain(False)和sensor.set\_auto\_whitebal(False)禁用自动增益和自动白平衡，以防止图像过曝。

摄像头内参：

这些参数对于准确的AprilTag检测至关重要：

焦距：

f\_x和f\_y分别是x和y方向上的焦距，根据摄像头的物理属性计算得出。

主点：

c\_x和c\_y是图像中心的坐标。

1.4图像处理

快照与镜头校正：

img = sensor.snapshot()捕捉图像。

img.lens\_corr(1.8)校正镜头畸变。

AprilTag检测：

for tag in img.find\_apriltags(fx=f\_x, fy=f\_y, cx=c\_x, cy=c\_y)使用先前定义的内参在图像中检测AprilTag。

可视化：

img.draw\_rectangle(tag.rect(), color=(255, 0, 0))在检测到的标签周围绘制一个矩形框。

img.draw\_cross(tag.cx(), tag.cy(), color=(0, 255, 0))在标签中心绘制一个十字。

1.5数据输出

位置与姿态：

脚本计算检测到的标签相对于摄像头的平移（x, y, z）和旋转角度（x, y, z，单位为度）。

print(pos\_deg\_output)输出每个标签的数据。

方向与角度计算：

脚本计算标签在摄像头左侧或右侧，并确定朝向标签转动的角度（theta）。

clockwise设置为0表示左侧，1表示右侧。

1.6UART通信

脚本构建了一个字节数组（FH）通过UART发送。消息包括：

起始字节（0xb3, 0xb3）

方向（clockwise）

角度（theta）

结束字节（0x5b）

该字节数组通过uart.write(FH)发送到连接的设备。

1.7使用说明

与二维码检测不同，AprilTag检测不需要镜头畸变校正。

该脚本配置为标准的OpenMV摄像头，但可以通过调整内参适应其他摄像头。

1.8实现代码

# Untitled - By: wql19 - Sat Jul 6 2024

import sensor, image, time, math, pyb, ustruct

from pyb import UART

sensor.reset()

sensor.set\_pixformat(sensor.RGB565)

sensor.set\_framesize(sensor.QVGA) # we run out of memory if the resolution is much bigger...

sensor.skip\_frames(30)

sensor.set\_auto\_gain(False) # must turn this off to prevent image washout...

sensor.set\_auto\_whitebal(False) # must turn this off to prevent image washout...

clock = time.clock()

uart = UART(3, 115200)

# c\_x 是图像的x中心位置

# c\_y 是图像的y中心位置

f\_x = (2.8 / 3.984) \* 320 # 默认值

f\_y = (2.8 / 2.952) \* 240 # 默认值

c\_x = 320 \* 0.5 # 默认值(image.w \* 0.5)

c\_y = 240 \* 0.5 # 默认值(image.h \* 0.5)

def degrees(radians):

return (180 \* radians) / math.pi

while(True):

clock.tick()

img = sensor.snapshot()

img.lens\_corr(1.8)

for tag in img.find\_apriltags(fx=f\_x, fy=f\_y, cx=c\_x, cy=c\_y): # 默认为TAG36H11

img.draw\_rectangle(tag.rect(), color = (255, 0, 0))

img.draw\_cross(tag.cx(), tag.cy(), color = (0, 255, 0))

pos\_deg\_output = (tag.x\_translation(), tag.y\_translation(), tag.z\_translation(),

degrees(tag.x\_rotation()), degrees(tag.y\_rotation()), degrees(tag.z\_rotation()))

print(pos\_deg\_output)

print(tag.cx(), tag.cy())

print("FPS %f" % clock.fps())

clockwise = 0

theta = 0

if (tag.x\_translation() < -0.1):

clockwise = 0 & 0xFF

theta = int(180\*math.atan2(-tag.x\_translation(), -tag.z\_translation()) / math.pi + 2) & 0xFF

elif (tag.x\_translation() > 0.1):

clockwise = 1 & 0xFF

theta = int(180\*math.atan2(tag.x\_translation(), -tag.z\_translation())/math.pi + 2) & 0xFF

print(clockwise, theta)

FH = bytearray([0xb3,0xb3,clockwise,theta,0x00,0x5b])

uart.write(FH)