# 自制全场定位使用说明

Version:1.1 author:lbc

目录

[自制全场定位使用说明 1](#_Toc70088999)

[一、简介 2](#_Toc70089000)

[二、初始参数配置 3](#_Toc70089001)

[三、坐标数据读取 3](#_Toc70089002)

[四、串口指令功能 4](#_Toc70089003)

[1. 复位 4](#_Toc70089004)

[2. 读取坐标 5](#_Toc70089005)

[3. 读取编码器的具体信息 5](#_Toc70089006)

[4. 读取角度的具体信息 6](#_Toc70089007)

[6. 读取flash存储的数据 6](#_Toc70089008)

[7.自动校准陀螺仪零漂 7](#_Toc70089009)

[8.设置两轮子直径 7](#_Toc70089010)

[9.设置CAN id 8](#_Toc70089011)

[五、可能存在的问题 9](#_Toc70089012)

[1.偏航角误差 9](#_Toc70089013)

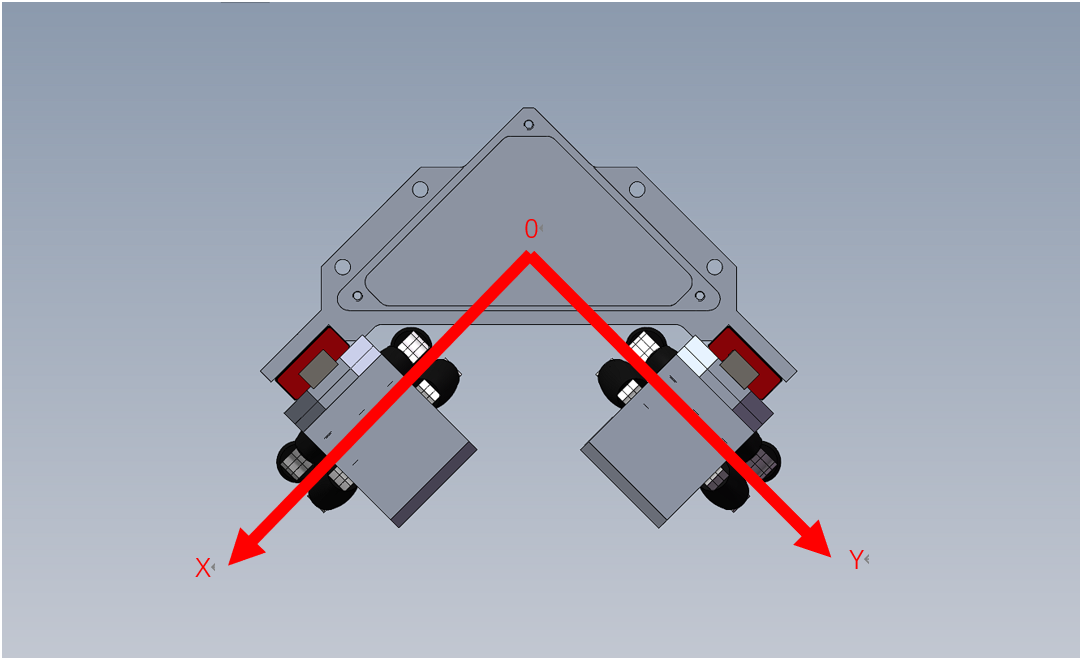
[2.码盘 9](#_Toc70089014)

[六、部分测试数据 10](#_Toc70089015)

## 一、简介

**主要芯片**： STM32F405RGTx芯片，两个as5047p绝对式编码器，一个asm330LHH六轴陀螺仪。

**结构示意图：**



**坐标系**：全场定位坐标系如上图所示，世界坐标系为初始时全场定位坐标系，刚上电时坐标为（0，0），偏航角为0。坐标单位为mm；偏航角单位为度，范围是（-180°，180°）。

**计算频率：**全场定位每5ms进行一次坐标的计算，其中偏航角采用两种方式进行计算，最终得到两组坐标计算的结果，两组坐标数据通过三组CAN消息发送。

**计算方式**：通过两编码器得到两轮子在5ms内位移的变化量dx’,dy’，结合偏航角计算出世界坐标系下全场定位x轴和y轴的变化量dx,dy，对dx和dy进行累加即可得到全场定位的坐标。

目前采用的偏航角两种计算方式：  
①将陀螺仪读到的六轴数据进行零漂处理再进行滤波，然后利用六轴融合计算出偏航角。

②不采用六轴融合，将全场定位的俯仰角和横滚角作为定值处理，计算出偏航角的角速度，然后积分得到偏航角。

二、初始参数配置

全场定位初次使用时按以下步骤进行参数配置1. 烧写程序后重新上电，将底盘置于地面；

2. 执行零漂校准，通过串口发送test\_zero\_data 30，将底盘静止30s，30s后串口将回显陀螺仪的六轴零漂数据并自动将数据写入flash。

3. 写入全场定位两个轮子直径，通过串口发送set\_diameter 50.8 50.8，两个数据分别为x轮和y轮的直径（单位为mm）。实际使用中可能需要对直径作拟合估计。

4. 设置三个can消息的id，通过串口发送set\_can\_id 204 205 206，三个数据分别为发送的x坐标，y坐标，偏航角的消息id。

配置完成后重新上电

## 三、坐标数据读取

全场定位上电后，5V、3V3和VIN的LED灯常亮表示供电正常；初始化时间大约2s，2s后串口回显初始化成功消息，之后便可读取坐标数据。

自制全场定位通过CAN总线发送坐标数据。每5ms发送三组can标准帧数据包，每组数据为can\_msg的联合体格式，包含有两个float类型数据。具体内容如下

typedef union{

    char ch[8];

    uint8\_t ui8[8];

    uint16\_t ui16[4];

    int16\_t i16[4];

    int in[2];

    float fl[2];

    double df;

} can\_msg;

|  |  |  |
| --- | --- | --- |
| CAN消息ID | 浮点数fl[0] | 浮点数fl[1] |
| X\_id（自行设置） | 方法1的X坐标 | 方法2的X坐标 |
| Y\_id（自行设置） | 方法1的Y坐标 | 方法2的Y坐标 |
| Angle\_id（自行设置） | 方法1的偏航角 | 方法2的偏航角 |

接收数据示例(假设三个id号分别为204，205，206)：

1. 在can\_utils.c的can id过滤中增加需要接收的三个id号

uint32\_t std\_id[] = {230, 324, 325, 89, 90, 204, 205, 206};

2. 在can\_func.c的初始化函数can\_func\_init()中添加接收三组数据和回调函数

can\_callback\_add(204, test\_position\_x);

can\_callback\_add(205, test\_position\_y);

can\_callback\_add(206, test\_position\_yaw);

3. 定义三个回调函数，在回调函数中读取坐标信息

void test\_position\_x(can\_msg \*data)

{

  Mode1\_pos\_x = data->fl[0];

  Mode2\_pos\_x = data->fl[1];

}

void test\_position\_y(can\_msg \*data)

{

  Mode1\_pos\_y = data->fl[0];

  Mode2\_pos\_y = data->fl[1];

}

void test\_position\_yaw(can\_msg \*data)

{

  Mode1\_angle = data->fl[0];

  Mode2\_angle = data->fl[1];

}

更新：程序主要在git两个分支上（DR专用和TR专用），两分支基础代码一样，发送的消息包略有不同。

DR专用：坐标计算进行了定位中心转换，中心距离参数在calculate.c文件里的L1和L2，还未集成到flash中，实际使用时需手动修改代码参数重新烧录。消息包如下：

|  |  |  |
| --- | --- | --- |
| CAN消息ID | 浮点数fl[0] | 浮点数fl[1] |
| X\_id（自行设置） | 方法1的X坐标(mm) | 方法2的X坐标(mm) |
| Y\_id（自行设置） | 方法1的Y坐标(mm) | 方法2的Y坐标(mm) |
| Angle\_id（自行设置） | 方法1的偏航角(degree) | 方法2的偏航角(mm) |

TR专用：未进行定位中心转换，额外计算底盘速度。消息包如下：

|  |  |  |
| --- | --- | --- |
| CAN消息ID | 浮点数fl[0] | 浮点数fl[1] |
| X\_id（自行设置） | 方法2的X轴速度(m/s) | 方法2的X坐标(m) |
| Y\_id（自行设置） | 方法2的Y轴速度(m/s) | 方法2的Y坐标(m) |
| Angle\_id（自行设置） | 方法2的偏航角速度(dps) | 方法2的偏航角(degree) |

待完善TODO:两分支代码功能整合，参数统一写入flash，陀螺仪解算偏航角仍有待改善。

## 四、串口指令功能

### 1. 复位

|  |  |
| --- | --- |
| 指令 | 说明 |
| reset | 全场定位重新初始化 |



### 2. 读取坐标

|  |  |
| --- | --- |
| 指令 | 说明 |
| readpos | 打印当前全场定位的两组坐标值 |



### 3. 读取编码器的具体信息

|  |  |
| --- | --- |
| 指令 | 说明 |
| wheelx\_info | X轮数据 |
| wheely\_info | Y轮数据 |



### 4. 读取角度的具体信息

|  |  |
| --- | --- |
| 指令 | 说明 |
| angel\_info | 陀螺仪解算的欧拉角具体信息 |



### 6. 读取flash存储的数据

|  |  |
| --- | --- |
| 指令 | 说明 |
| read\_flash | 打印写到flash中的数据 |

|  |  |
| --- | --- |
| **Flash数据** | **数据含义** |
| flash[0] | X轴角速度零漂 |
| flash[1] | Y轴角速度零漂 |
| flash[2] | Z轴角速度零漂 |
| flash[3] | X轴平地静态加速度 |
| flash[4] | Y轴平地静态加速度 |
| flash[5] | Z轴平地静态加速度 |
| flash[6] | X轮估计直径 |
| flash[7] | Y轮估计直径 |
| flash[8] | X坐标消息CAN id |
| flash[9] | Y坐标消息CAN id |
| flash[10] | 角度坐标消息CAN id |



### 7.自动校准陀螺仪零漂

|  |  |
| --- | --- |
| 指令 | 说明 |
| test\_zero\_data <param> | <param>为校准的时间（int型，单位秒）  一般设置校准时间60s即可  将底盘放于平地静置，发送该指令，校准完成后自动将数据写入flash |



### 8.设置两轮子直径

|  |  |
| --- | --- |
| 指令 | 说明 |
| set\_diameter <param1> <param2> | <param1>:x轮直径(float型,单位mm)  <param2>:y轮直径(float型,单位mm)  调用该指令将轮子直径数据写入flash |



### 9.设置CAN id

|  |  |
| --- | --- |
| 指令 | 说明 |
| set\_can\_id <param1> <param2> <param3> | <param1>:X坐标数据id  <param2>:Y坐标数据id  <param3>:偏航角数据id  将三个id 写入flash |



查看其余指令以及增删指令：代码cmd\_func.c文件

## 五、可能存在的问题

### 1.偏航角误差

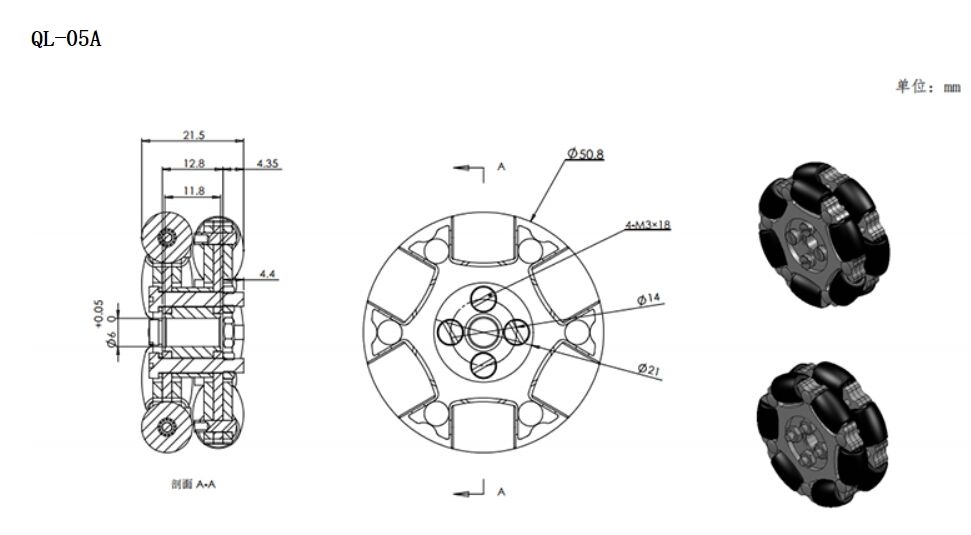
使用六轴融合计算偏航角时，当底盘有较大的加速度且带有自转时，偏航角的误差将会增大，坐标误差随即增大，方法1更适用于底盘简单平稳运动的情况。

方法2是计算过程中没有考虑加速度计，不受底盘运动加速度的影响，但只适用于水平地面的定位。

陀螺仪的零漂会受到环境状况的影响，若底盘静止状态下偏航角的误差随时间不断增大，则需要进行一次零漂校准；使用前最好也进行一次零漂校准。

全场定位在初始化的2s内以及零漂校准的一段时间内必须保持水平静止；偏航角的误差会随时间以及自转圈数增大而增大，坐标计算的误差主要来自偏航角的误差；剧烈的车体震动以及车体的撞击对偏航角的计算也有一定的影响

### 2.码盘



全场定位两全向轮直径的机械尺寸为50.8mm。实际使用中到目标点全场定位的计算坐标和实际坐标会有一定的偏差，需要重新测量全向轮的直径，算出直径的最佳估计。

## 六、部分测试数据





（跑定点测试待补充）