基于T-bot实验指导书——运动学

1.项目名称

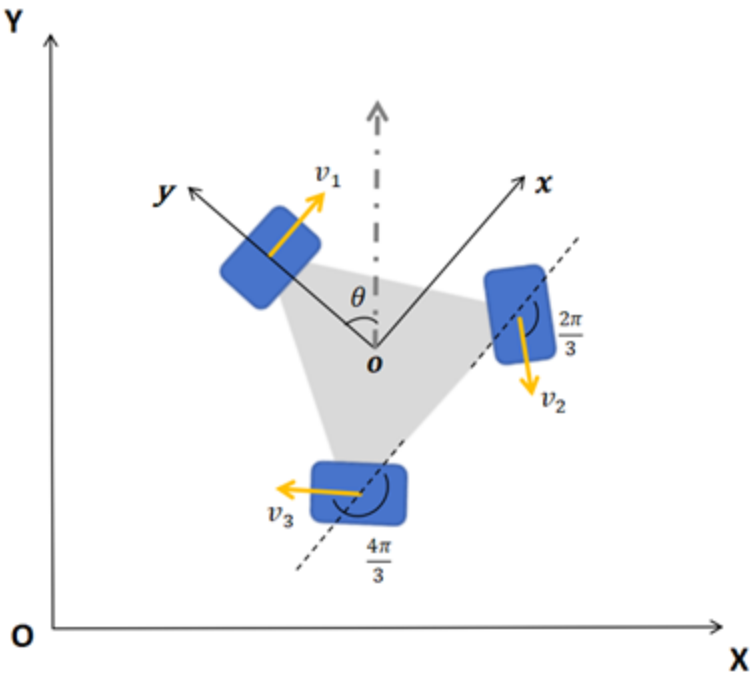
移动机器人正运动学实践

2.背景

运动学是研究物体运动状态及其变化的科学，尤其关注位置、速度、加速度等参数。对于移动机器人而言，运动学模型可以帮助我们分析和预测机器人的运动轨迹，从而实现精确的控制和导航。通过运动学分析，机器人可以在动态环境中避免障碍、规划路径并执行任务。

在本实验中，我们将研究全向轮驱动移动机器人的基本运动学原理。通过建立数学模型，我们将进行仿真实验和实际测试，以验证理论模型的准确性和有效性

3.理论知识



根据上图，建立世界坐标系{XOY},以小车中心为原点建立机器人坐标系{xoy},设每个轮子与车体中心的距离为r，每个轮子与坐标系{xoy}的x轴夹角分别为

$$a_1 = 0, a_2 = \frac{2 * pi}{3}, a_2 = \frac{4 * pi}{3}$$

每个轮子速度与车体速度之间的关系为：

$$v_i = v_x \cos(\alpha_i) + v_y \sin(\alpha_i) + r\omega$$

对于三轮全转车体，可以列出三个轮子的运动学方程：

$$v_1 = v_x \cos(0) + v_y \sin(0) + r\omega = v_x + r\omega$$

$$v_2 = v_x \cos\left(\frac{2\pi}{3}\right) + v_y \sin\left(\frac{2\pi}{3}\right) + r\omega = -\frac{1}{2}v_x + \frac{\sqrt{3}}{2}v_y + r\omega$$

$$v_3 = v_x \cos\left(\frac{4\pi}{3}\right) + v_y \sin\left(\frac{4\pi}{3}\right) + r\omega = -\frac{1}{2}v_x - \frac{\sqrt{3}}{2}v_y + r\omega$$

将这些方程写成矩阵形式：

$$\begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}^o = \begin{pmatrix} 1 & 0 & r \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & r \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & r \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ \omega \end{pmatrix}^O$$

4.实验过程

4.1实验前准备

4.1.1. 准备控制板工程代码

- **选择开发环境**：
  - 确保已选择合适的开发环境，例如 STM32CubeIDE 或 Keil MDK。
- **获取代码库**：
  - 从 GitHub 或其他代码托管平台下载所需的控制板工程代码。
- **配置项目**：
  - 根据硬件配置，调整工程设置和代码中的参数。

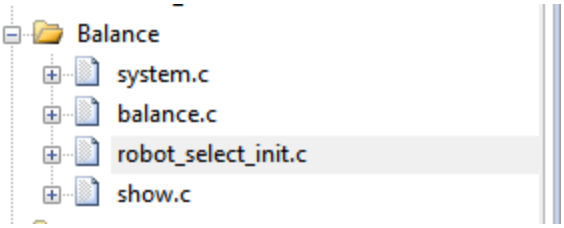
4.1.2. 准备下载器

- **选择下载器**：
  - **ST-Link**：适用于 STM32 系列微控制器。
  - **J-Link**：适用于多种微控制器，支持更广泛的设备。
- **安装驱动**：
  - 对于 ST-Link，安装 STM32 ST-LINK Utility。
  - 对于 J-Link，安装 J-Link Software and Documentation Pack。
- **连接下载器**：
  - 使用相应的连接线将 ST-Link 或 J-Link

4.2实验步骤

step1.熟悉工程

打开工程文件，观察工程的文件组成，本实验主要关心balance.c,robot_init.c,show.c三个源文件。 其中balance.c是运动学相关的文件，robot_init.c是移动车体相关参数设置文件，show.c是oled的显示文件。



底层控制模块采用stm32+freertos进行控制系统搭建，系统上电后，创建了四个任务，分别是运动控制任务，IMU数据读取任务，显示任务，数据处理任务。

```
void start_task(void *pvParameters)
{
    taskENTER_CRITICAL(); //Enter the critical area

    //Create the task
    xTaskCreate(Balance_task,  "Balance_task",  BALANCE_STK_SIZE,  NULL, BALANCE_TASK_PRIO,  NUI
    //Vehicle motion control task
    xTaskCreate(MPU6050_task, "MPU6050_task", MPU6050_STK_SIZE, NULL, MPU6050_TASK_PRIO, NULL);
    //IMU data read task
    xTaskCreate(show_task,     "show_task",     SHOW_STK_SIZE,     NULL, SHOW_TASK_PRIO,     NUI
    //The OLED display displays tasks
    xTaskCreate(led_task,      "led_task",      LED_STK_SIZE,      NULL, LED_TASK_PRIO,      NUI
    //LED light flashing task
//    xTaskCreate(pstwo_task,    "PSTWO_task",    PS2_STK_SIZE,      NULL, PS2_TASK_PRIO,      N
//Read the PS2 controller task
    xTaskCreate(data_task,     "DATA_task",     DATA_STK_SIZE,     NULL, DATA_TASK_PRIO,     NUI
    //Usartx3, Usartx1 and CAN send data task //
    vTaskDelete(StartTask_Handler);
    //Delete the start task

    taskEXIT_CRITICAL();
    //Exit the critical section//ÍË³öÁÙ½çÇø
}
```

step2.测量小车物理参数

本次实验的重点是修改控制任务，添加合适的运动学模型，使得全向移动底盘能运动。首先对于机器人的参数进行配置，如下函数描述，这个函数的参数就是机器人的参数，如电机的减速比，轴距，编码器的精度等等。这些参数需要实际测量，或者通过说明书手册获知。

```
/**************************************************************************
Function: Initialize cart parameters
Input   : wheelspacing, axlespacing, omni_rotation_radiaus, motor_gear_ratio, Number_of_encoder_lines, tyre_diameter
Output  : none
函数功能：初始化小车参数
入口参数：轮距 轴距 自转半径 电机减速比 电机编码器精度 轮胎直径
返回  值：无
**************************************************************************/
void Robot_Init(double wheelspacing, float axlespacing, float omni_turn_radiaus, float gearratio,float Accuracy,float tyre_diameter)
```

请你根据你们自己装配的移动底盘，测量确定轮距，轴距，电机减速比，电机编码器的精度，轮直径参数，并填入以下表格

名称	值
轮直径	
轴距	
电机减速比	
编码器精度	

Step3.完成运动学代码编写

Driver_motor为需要修改的运动学函数，请添加运动学代码，控制小车运动

```
void Drive_Motor(float Vx,float Vy,float Vz)
{
    float amplitude=3.5; //Wheel target speed limit
    //Speed smoothing is enabled when moving the omnidirectional trolley
    Smooth_control(Vx,Vy,Vz); //Smoothing the input speed //
    //Get the smoothed data
    Vx=smooth_control.VX;
    Vy=smooth_control.VY;
    Vz=smooth_control.VZ;
    //Omni car
    //Inverse kinematics
    MOTOR_A.Target   =   Vy + Omni_turn_radiaus*Vz;
    MOTOR_B.Target   =  -X_PARAMETER*Vx - Y_PARAMETER*Vy + Omni_turn_radiaus*Vz;
    MOTOR_C.Target   =  +X_PARAMETER*Vx - Y_PARAMETER*Vy + Omni_turn_radiaus*Vz;

    //Wheel (motor) target speed limit //
    MOTOR_A.Target=target_limit_float(MOTOR_A.Target,-amplitude,amplitude);
    MOTOR_B.Target=target_limit_float(MOTOR_B.Target,-amplitude,amplitude);
    MOTOR_C.Target=target_limit_float(MOTOR_C.Target,-amplitude,amplitude);
    MOTOR_D.Target=0;   //Out of use
}
```

Step4.控制移动底盘运动

接入蓝牙遥控模块，手机安装apk文件(位于学习通--资源---工具目录)，通过蓝牙遥控，观察小车是否运动