《机器人导论实验》报告

|  |  |  |
| --- | --- | --- |
| **第三次实验** | **日期：2022-10-31** | **得分：** |
| **学号：20201060351** | **姓名：张翰林** | **专业：智能科学与技术** |

# 实验准备

学习、理解RZ7886电机转动、控制的原理，为调控转速和方向做准备。

学习、理解编码器的原理，理解编码器将数学表达转换为脉冲信号的过程。

学习、理解PID控制器根据高级语言控制实际电机作用效果的过程，理解其将连续过程转换为离散数据以供使用的原理。

# 实验原理

电机工作原理：以RZ7886直流电机为例，电机两极的电流流向控制电机的转向，电机的转速大体上与外接电压正相关。

编码器原理：编码器通过光电转换或磁电转换，将输出轴的机械位移转换成脉冲信号或数字量，以便于将采集的方波信号供给单片机。

PWM原理：PWM信号是脉宽调制信号，可以转化成电压信号，所转化成的电压信号与PWM的占空比有着直接的关系。

PID原理：因为控制器是通过软件实现其控制算法的，所以必须对模拟调节器进行离散化处理，这样它只需根据采样时刻的偏差值计算控制量。因此，我们需要使用离散的差分方程式代替连续的微分方程。

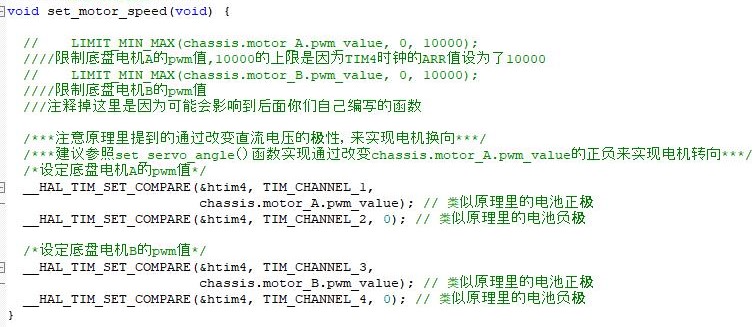
# 实验任务

# 实验过程与实验结果

## 控制智能小车的两个轮子，一个正转，一个反转。并录制一个简短的视频。

根据PPT内容提示，只需要调整电机的极性，即可调整小车的两个轮子的正转或反转。对于本次实验的小车，由于两个小车分别充当左、右轮，本身方向就相反，所以当两个电机相同设置时，其效果就是一个正转一个反转，原版工程便满足这一要求。只是由于原版工程中pwm值过低，使得两直流电机的输出动力太弱，不过这一问题可以简单通过增大pwm value解决。

### (1) 代码改动如下：



(该函数用于控制两后轮的转向)

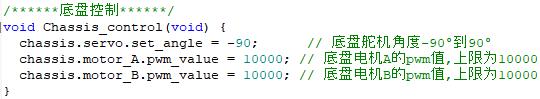
### (2) 实验结果如下：

(视频已删除)

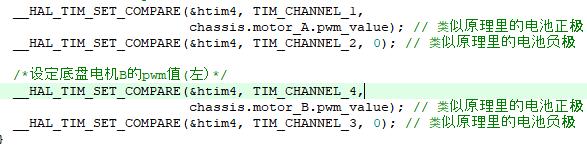
## 编写智能小车代码，控制智能小车在无遥控器控制的状态下进行圆周运动。并录制一个简短的视频。

欲实现圆周运动，则需要前轮保持一个固定角度，两个后轮保持一致的角速度，位于内侧的线速度偏慢。但是实际上由于小车的转动半径并不小，且摩擦力也不是很大，所以出于实际和简单考虑，两后轮的速度直接设为相同即可。

### (1) 代码改动如下：



(将两个电机的pwm值都调高，设定舵机角度为恒定-90°)



(将motor\_B的方向对换，使舵机B也保持正转)

### (2) 实验结果如下：

(视频已删除)

## 编写智能小车代码，控制智能小车在无遥控器控制的状态下随意运动（比如前进→转弯→后退→转弯……)。并录制一个简短的视频。

在本题中，我们组打算效果是，前轮使用sin函数，使得转向角度能够周期性变化，后轮则是周期性地在前进和后退状态间交替运行。

为此，要设定一个计数器，角度的计算和前进后退的判别都根据这个计数值决定。

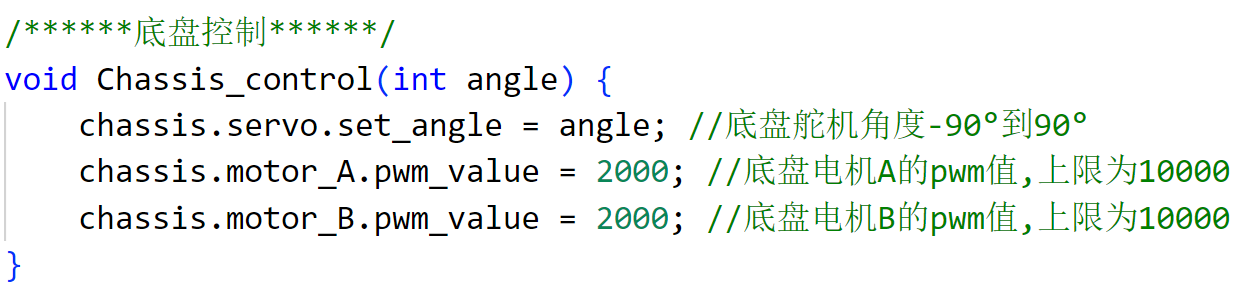
对于角度，首先将计数器i的值作为输入，将sin的输出值\*60作为角度值，这样就限定了舵机角度在[-60,60]之间。同时，将Chassis\_control()函数改为传参类型，接受调用处计算的角度值。

对于后轮转向，首先将Chassis\_work()函数改为传参类型，在被调用处可以被控制前进还是后退。i<1000时前进；1000<i<2000时后退，大于2000时将i归零，即完成循环。然后在Chassis\_work()内部，根据参数的值决定两电机是正转还是反转。最后完成了相关结果。

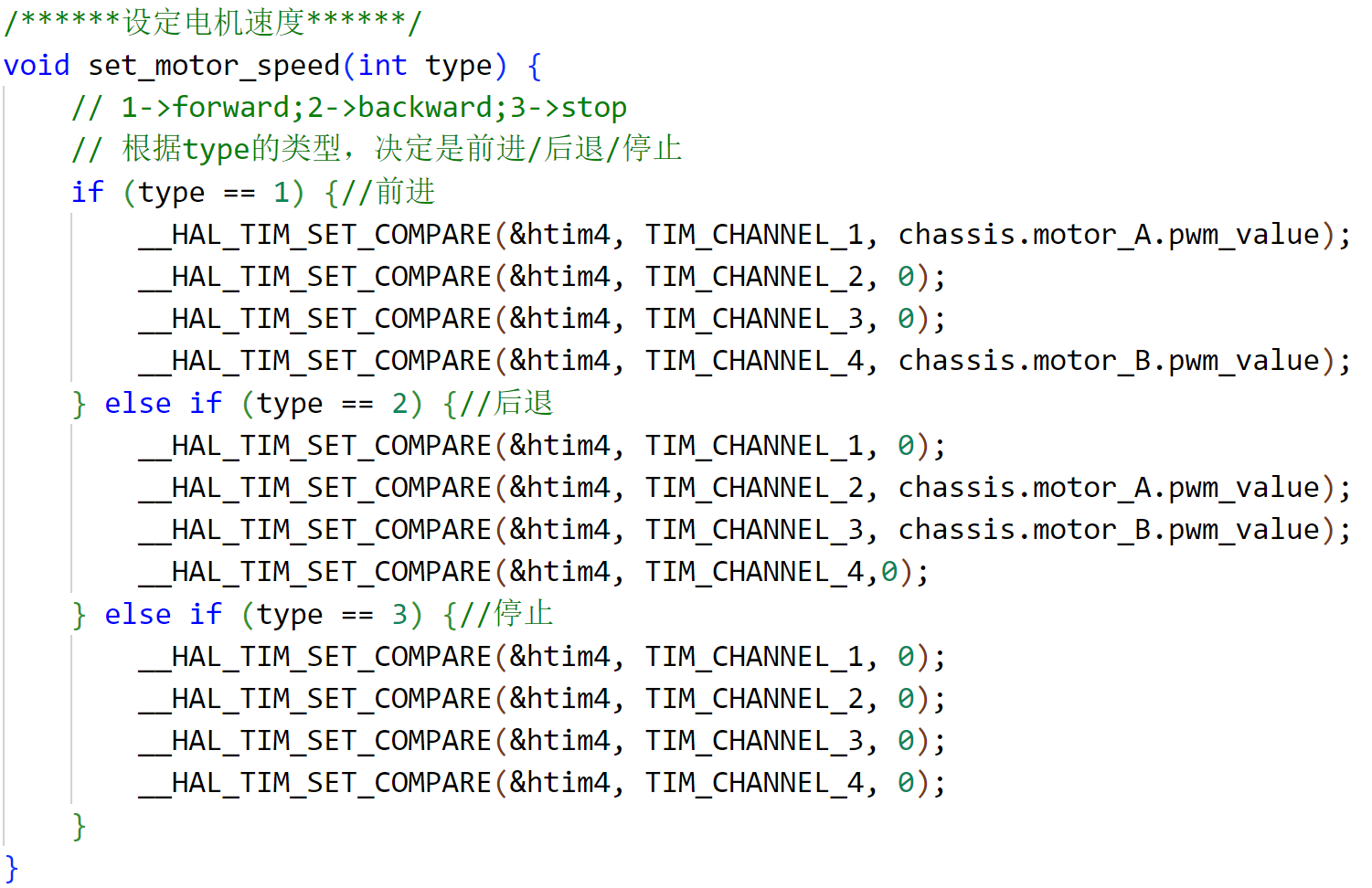
### (1) 代码改动如下：



(主函数，angle为根据计时器计算的角度，i为计时器)



(Chassis\_control函数改动，使实际舵机角度能动态变化，安全起见降低了速度)



(set\_motor\_speed改动，能根据type的不同改变后轮的转向)

### (2) 实验结果如下：

(视频已删除)

# 实验心得

前两题的完成都比较顺利，只要了解了PWM原理以及实际控制电机的函数之后，只需要稍微进行改动即可。

第三题中，由于实际控制舵机角度和电机转速的是pwm值，pwm值的计算代码为：

chassis.servo.pwm\_value=1500+(chassis.servo.set\_angle/90)\*1000

这会导致一个很严重的问题：(chassis.servo.set\_angle/90)的数据类型为int，当chassis.servo.set\_angle不为90或-90时，其结果始终为0，使得pwm\_value的对应角度只有-90、0、90三个，正是因为这个不起眼的一点导致了我们一直无法实现转向。

随后修改了这里的代码，改为如下的形式，问题解决。

chassis.servo.pwm\_value=1500+(int)(chassis.servo.set\_angle/90.0\*1000);