中国机器人大赛-机器人旅游寻宝项目

——国家级二等奖

林德旸
18 级自动化创新班
3118001018
LaTeX

更新：*2020* 年 *1* 月 *2* 日

摘　　要

此款机器人是一款基于 STM32F103RCT6 主控芯片的四轮车型搬运机器人。该主控芯片来自 ST (意法半导体) 公司, ARM Cortex TM—M3 内核, 32 位微控制器。通过光电传感器循迹板循迹定位，颜色传感器 TCS34725 判别物块颜色, 通过机械臂和一个红外避障传感器来夹取物料，将物料储存在仓库中，到达指定放置地点将其投放, 以完成比赛要求。按照中国机器人大赛“光电车型搬运赛”项目的竞赛要求, 模拟工业自动化过程中的作业任务，将不同颜色但相同形状的物料分类搬运到规定的目标区域。由于篇幅有限，在此仅着重介绍个人设计的模糊 PID 控制算法，其详细的车辆设计方案与具体的程序工程文件已经上传至 Github，如果您想了解更多关于此项项目，请访问：https://github.com/ldylab，在 Repositories 下获取所有比赛项目个人的开源程序，再次感谢！

关键词： 搬运工程、光电车型、机器人、模糊 PID 控制算法

1　项目背景

随着工业自动化进程不断加快, 工业机器人在其中扮演着越来越重要的角色。而搬运机器人 (AGV) 又是工业机器人的代表。它可广泛应用于机械、电子、纺织、卷烟、医疗、食品、造纸等行业的柔性搬运、传输等功能; 也可以用于自动化立体仓库、柔性加工系统、柔性装配系统; 同时可在车站、机场、邮局的物品分捡中作为运输工具。本款机器人是按照中国机器人大赛“光电车型搬运赛”项目的竞赛要求, 模拟工业自动化过程中的作业任务，将不同颜色但相同形状的物料分类搬运到规定的目标区域。

2 项目实施情况

2.1 结构设计

其整体设计如下：

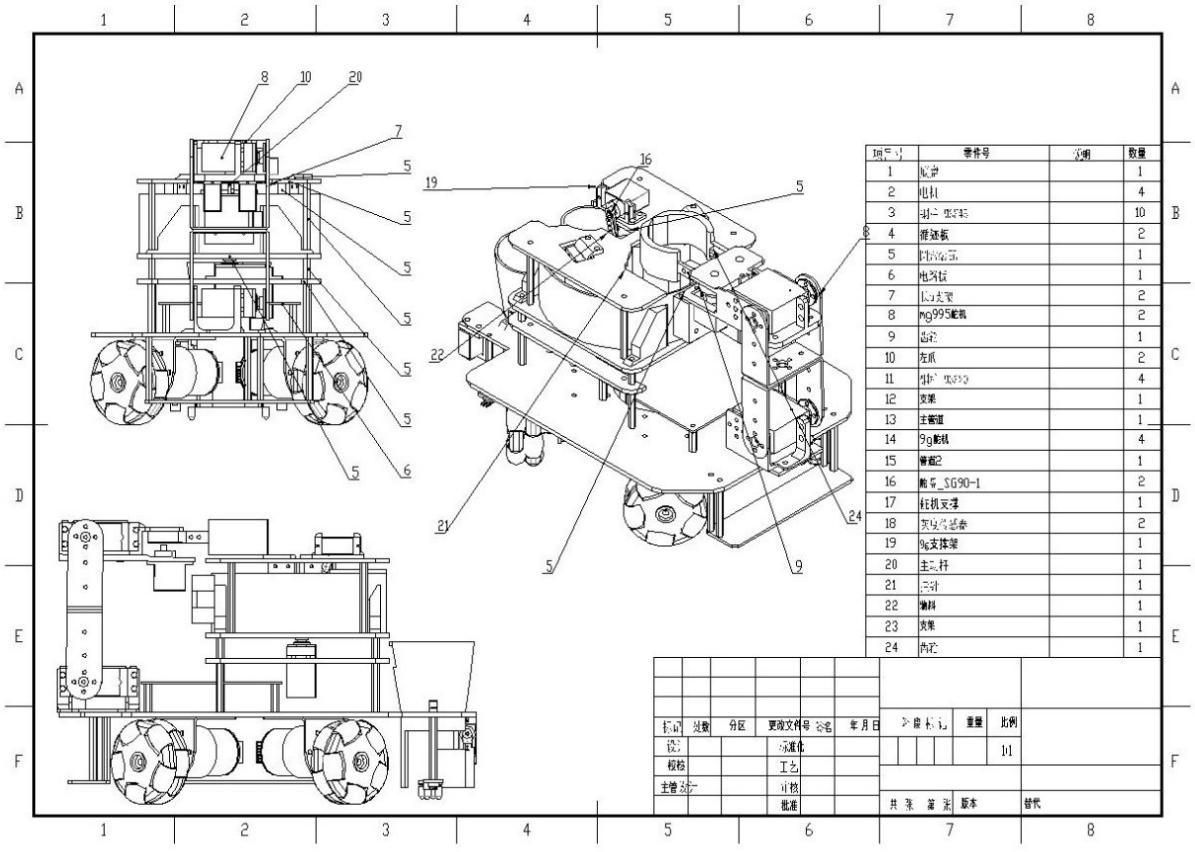

图 1: 整体结构

2.2 电路设计

2.2.1 红外光电对管循迹板

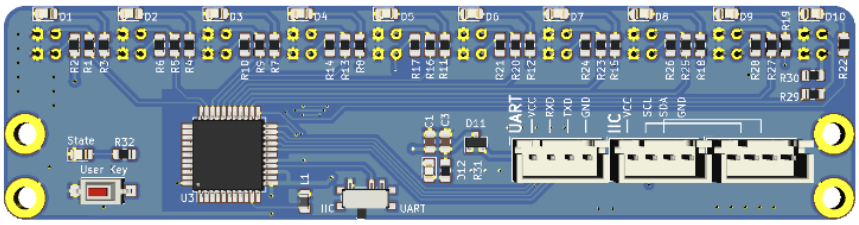

图 2: 动态阈值循迹板

2.2.2 其余电路部分

车辆主控采用 STM32F103，电机驱动采用 TB6612 驱动四路编码器电机，使用 JY-91 九轴陀螺仪确定小车在场内的方向方向，其不再累述。

3 项目个人主要承担工作

个人在此项目中主要开发了基于 BP 神经网络的动态阈值循迹板（在中国机器人大赛-游中国项目中已经有详细介绍，在此不再累述），模糊 PID 控制算法，其保证了机器人在更换电机电机后与不同载重与与不同目标速度时，在使用同一套 PID 参数时依然有着非常不错的控制效果，在比赛中具有非常重要的应用与意义。

3.1 模糊 PID 控制算法

3.1.1 模糊 PID 控制算法 Matlab 仿真

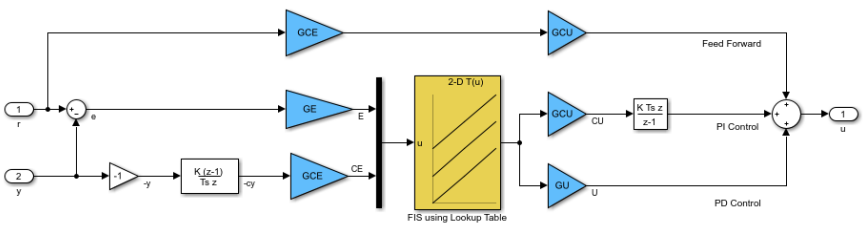

图 3: Simulink 框图

查看 3D 非线性控制面。该表面在 E 和 CE 平面的中心附近具有比线性表面更高的增益，这有助于在误差较小时更快地减小误差。当误差很大时，控制器将变得不那么主动以避免可能的饱和。

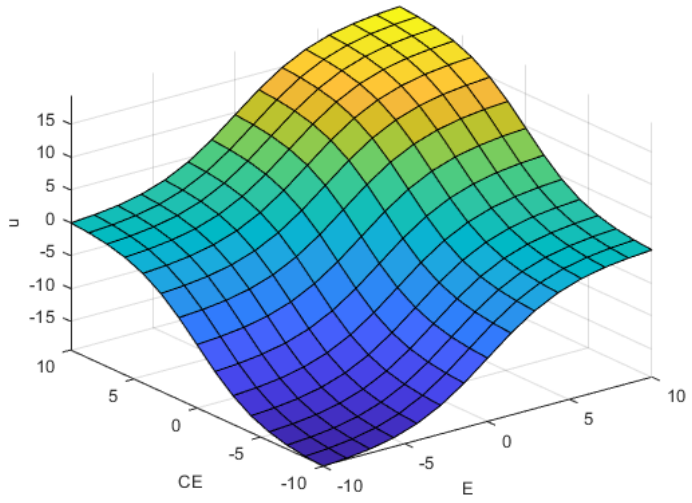

图 4: 非线性控制面

与传统的线性 PID 控制器（具有较大过冲的响应曲线）相比，非线性模糊 PID 控制器可将过冲减少 50%。非线性模糊控制器的两条响应曲线几乎重叠，这表明二维查找表很好地逼近了模糊系统。

图 5: 对比控制

3.2 模糊 PID 实际设计

其主要 C 语言程序设计如下所示（节选）：

```
//模糊规则表参数设定
#define IS_Kp 1
#define IS_Ki 2
#define IS_Kd 3

#define NL   -3
#define NM   -2
#define NS   -1
#define ZE   0
#define PS   1
#define PM   2
#define PL   3

//模糊规则表
static const float fuzzyRuleKp[7][7]=
{
  PL, PL, PM, PM, PS, PS, ZE,
  PL, PL, PM, PM, PS, ZE, ZE,
  PM, PM, PM, PS, ZE, NS, NM,
  PM, PS, PS, ZE, NS, NM, NM,
  PS, PS, ZE, NS, NS, NM, NM,
  ZE, ZE, NS, NM, NM, NM, NL,
  ZE, NS, NS, NM, NM, NL, NL
};

static const float fuzzyRuleKi[7][7]=
{
  NL, NL, NL, NM, NM, ZE, ZE,
  NL, NL, NM, NM, NS, ZE, ZE,
  NM, NM, NS, NS, ZE, PS, PS,
  NM, NS, NS, ZE, PS, PS, PM,
  NS, NS, ZE, PS, PS, PM, PM,
```

```
  ZE, ZE, PS, PM, PM, PL, PL,
  ZE, ZE, PS, PM, PL, PL, PL
};

static const float fuzzyRuleKd[7][7]=
{
  PS, PS, ZE, ZE, ZE, PL, PL,
  NS, NS, NS, NS, ZE, NS, PM,
  NL, NL, NM, NS, ZE, PS, PM,
  NL, NM, NM, NS, ZE, PS, PM,
  NL, NM, NS, NS, ZE, PS, PS,
  NM, NS, NS, NS, ZE, PS, PS,
  PS, ZE, ZE, ZE, ZE, PL, PL
};

//模糊PID的参数计算
PID fuzzy(float e,float ec)
{
  float etemp,ectemp;
  float eLefttemp,ecLefttemp;
  float eRighttemp ,ecRighttemp;

  int eLeftIndex,ecLeftIndex;
  int eRightIndex,ecRightIndex;
  PID fuzzy_PID;
  etemp = e > 3.0 ? 0.0 : (e < - 3.0 ? 0.0 : (e >= 0.0 ? (e
     >= 2.0 ? 2.5: (e >= 1.0 ? 1.5 : 0.5)) : (e >= -1.0 ?
     -0.5 : (e >= -2.0 ? -1.5 : (e >= -3.0 ? -2.5 : 0.0) ))))
     ;

  //一次三角函数的模糊化
  eLeftIndex = (int)e;
  eRightIndex = eLeftIndex;
  eLeftIndex = (int)((etemp-0.5) + 3);
  eRightIndex = (int)((etemp+0.5) + 3);
```

```
65
66      eLefttemp =etemp == 0.0 ? 0.0:((etemp+0.5)-e);
67      eRighttemp=etemp == 0.0 ? 0.0:( e-(etemp-0.5));
68
69      ectemp = ec > 3.0 ? 0.0 : (ec < - 3.0 ? 0.0 : (ec >= 0.0 ?
            (ec >= 2.0 ? 2.5: (ec >= 1.0 ? 1.5 : 0.5)) : (ec >= -1.0
             ? -0.5 : (ec >= -2.0 ? -1.5 : (ec >= -3.0 ? -2.5 : 0.0)
             ))));
70
71      ecLeftIndex = (int)((ectemp-0.5) + 3);
72      ecRightIndex = (int)((ectemp+0.5) + 3);
73
74      ecLefttemp =ectemp == 0.0 ? 0.0:((ectemp+0.5)-ec);
75      ecRighttemp=ectemp == 0.0 ? 0.0:( ec-(ectemp-0.5));
76
77      /*************************************反模糊
            *************************************/
78      //反模糊解算参数设定
79      fuzzy_PID.Kp = (eLefttemp * ecLefttemp *  fuzzyRuleKp[
            eLeftIndex][ecLeftIndex]
80      + eLefttemp * ecRighttemp * fuzzyRuleKp[eLeftIndex][
            ecRightIndex]
81      + eRighttemp * ecLefttemp * fuzzyRuleKp[eRightIndex][
            ecLeftIndex]
82      + eRighttemp * ecRighttemp * fuzzyRuleKp[eRightIndex][
            ecRightIndex]);
83
84      fuzzy_PID.Ki = (eLefttemp * ecLefttemp * fuzzyRuleKi[
            eLeftIndex][ecLeftIndex]
85      + eLefttemp * ecRighttemp * fuzzyRuleKi[eLeftIndex][
            ecRightIndex]
86      + eRighttemp * ecLefttemp * fuzzyRuleKi[eRightIndex][
            ecLeftIndex]
87      + eRighttemp * ecRighttemp * fuzzyRuleKi[eRightIndex][
            ecRightIndex]);
```

```
88
89    fuzzy_PID.Kd = (eLefttemp * ecLefttemp *    fuzzyRuleKd[
          eLeftIndex][ecLeftIndex]
90    + eLefttemp * ecRighttemp * fuzzyRuleKd[eLeftIndex][
          ecRightIndex]
91    + eRighttemp * ecLefttemp * fuzzyRuleKd[eRightIndex][
          ecLeftIndex]
92    + eRighttemp * ecRighttemp * fuzzyRuleKd[eRightIndex][
          ecRightIndex]);
93
94    return fuzzy_PID;
95  }
96
97  //位置式PIDFuzzy计算
98  float posi_pid(PID *pid, float target, float current)
99  {
100   PID OUT = {0, 0, 0}; //初始化模糊PID
101
102   pid->target_speed = target;
103   pid->current_speed = current;
104
105   pid->error_now = pid->target_speed - pid->current_speed;//
          目标值 - 实际值 = 误差值
106   pid->error_delta = pid->error_now - pid->error_last;//误差
          变化率用于Pd的调节
107   pid->error_sum += pid->error_now;//误差值的累加
108   pid->error_last = pid->error_now;//上一次的误差值
109
110   if(pid->Ki == 0)
111   {
112     pid->set_point = (pid->Kp+OUT.Kp)*pid->error_now + (pid->
          Kd+OUT.Kd)*pid->error_delta;
113   }
114   else if(pid->Ki != 0)
115   {
```

```
116       pid->set_point = (pid->Kp+OUT.Kp)*pid->error_now + (pid->
              Kd+OUT.Kd)*pid->error_delta + (pid->Ki + OUT.Ki)*pid->
              error_sum;
117     }
118
119     //对于输出值的限定
120     if(pid->set_point > pid->output_maximun)
121     {
122       pid->set_point = pid->output_maximun;
123     }
124     else if(pid->set_point < pid->output_minimun)
125     {
126       pid->set_point = pid->output_minimun;
127     }
128     return pid->set_point;
129 }
```

运行结果如下

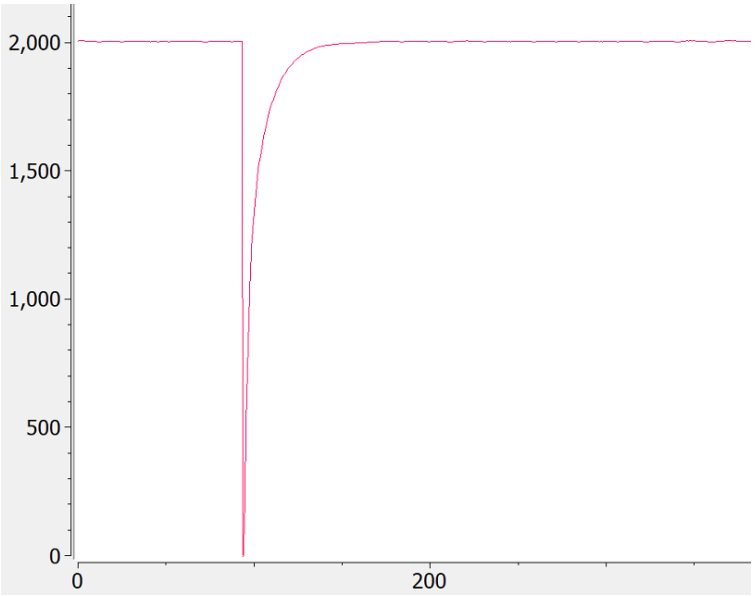

图 6: 同组参数目标速度 =2000

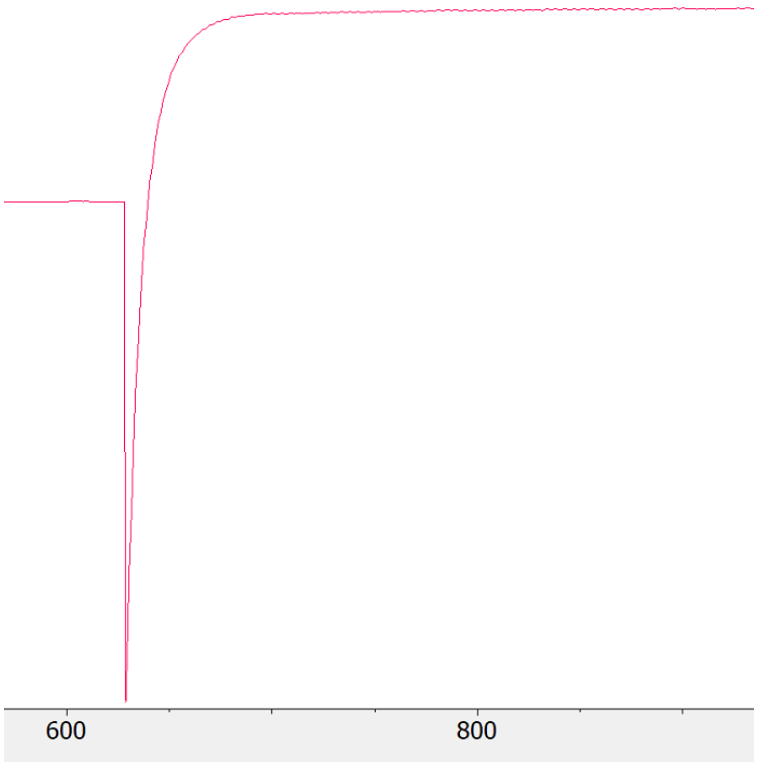

图 7: 同组参数目标速度 =4000

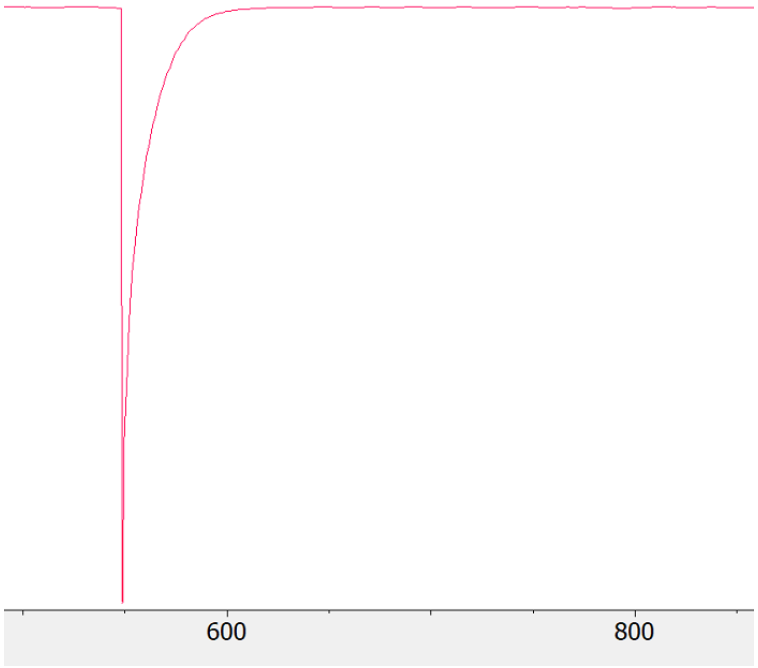

图 8: 同组参数目标速度 =6000

测试结果表明其在速度较大范围的变化和其电机的负载量不同时，其依然有着接近完美的 PID 曲线，而在实际比赛之中也对于控制整体的精度和稳定性起到了非常大的作用。

4 个人总结

通过这一个比赛个人学习到的更多是对于数学知识的应用，同时也意识到仿真对于实际结果的重要意义，也其实在此之后个人更加系统的学习的 Matlab 的 Simulink 模块，对于之后的动力仿真与算法验证都起到了非常大的作用。

参考文献

[1] 祝相泉, 黄海龙, 田昊. 无刷直流电机模糊 PID 控制 [J/OL]. 辽宁工业大学学报 (自然科学版),2020(01):1-4[2020-01-01].http://kns.cnki.net/kcms/detail/21.1567.T.20191227.1041.020.html.

[2] Elham Yazdani Bejarbaneh,Ahmad Bagheri,Behnam Yazdani Bejarbaneh,Salinda Buyamin,Saeed Nezamivand Chegini. A new adjusting technique for PID type fuzzy logic controller using PSOSCALF optimization algorithm[J]. Applied Soft Computing Journal,2019.