基于 S12 的模糊控制调试心得

夜随风舞 2008-8-5

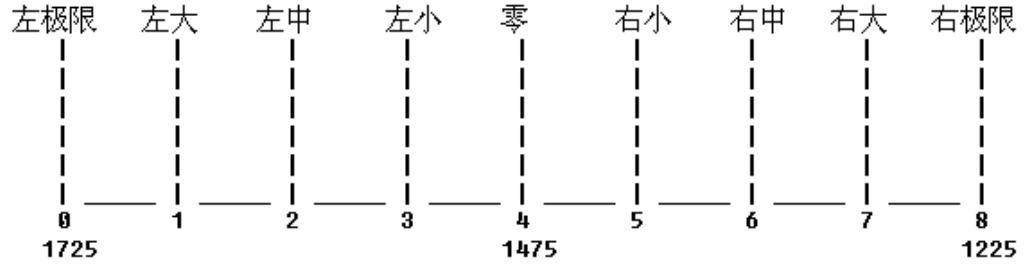
小可不才在博客发了一篇有关 S12 的模糊控制的日志（《MC9S12DG128 模糊控制崩溃之旅》）。并跟一些志同道合的网友展开了较为深入的讨论。现应部分网友同仁的要求将我调试成功的心得体会写在如下，由于本人也只是懂得一些皮毛，而且有很多问题并未深究，难免有不妥或错误之处，还请各位包涵并指正，谢谢！

在此以飞思卡尔智能小车的方向控制为例，文中可能会提到一些变量，但不会对该变量作深入的探究，敬请原谅！

首先必须明确你的被控量是什么？它的变化范围（即论域）多大？它是不是可以由 S12 的模糊机输出？隶属度函数是什么类型的？

然后是模糊控制的两个输入量是什么？它们的论域是多大？是否要量化？隶属度函数是什么类型的？

在这里，被控量是小车转角，向上回朔是舵机的转角，再向上就是用来控制舵机的 PWM 波的占空比，也就是说你的被控量是 PWM 波的占空比；一旦你的舵机安装方式确定后，小车前轮的左右极限转角也就定死了，也就是说你的 PWM 波的占空比的变化范围也就确定了，即被控量的论域确定了，以我的为例是 1225 到 1725，显然 S12 的模糊机无法直接输出这么大的数值，在此对于输出量的隶属度函数我选用的是单值的，对应 9 个等级（0-8），即对应 9 个 PWM 占空比值，即对应 9 个角度值。模糊语言等级、输出等级以及实际 PWM 占空比值的对应关系如下。



该对应关系在代码中就是：

```
//输出角度等级的隶属度函数
const uchar OUTPUT_MFS[9]={
                            0,1,2,3,4,5,6,7,8
                            };
```

如果说你要的输出量是一个浮点型的，即带小数。我通常采用的方法是将输出量同比增大 10 倍或是 100 倍，如果允许增大那么多倍的话。这样你的输出就可以是 uchar 的拉，要还原成小数的话，只要在后续的运算中相应地除以 10 或 100.

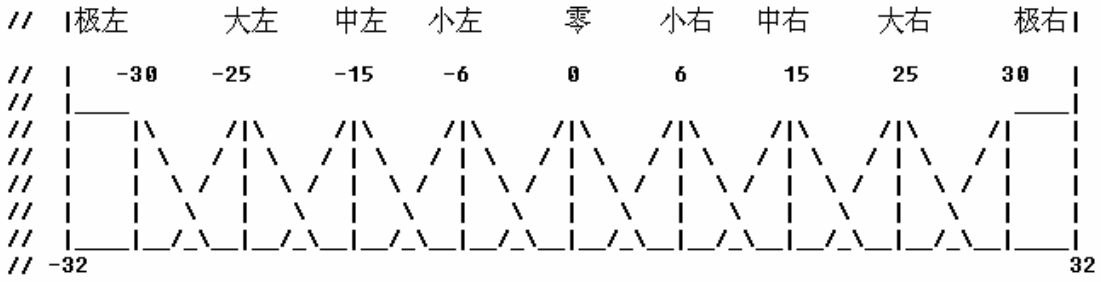
输出确定之后，你可以先不管模糊控制的中间过程，而是应该先确定你的控制输入量。在此选择了赛道黑线的弯曲程度（curve）以及车体的位置（av_loca）。经实际测量发现它们的论域分别是 curve：-32—32；av_loca：0—80。当然我在测量时没有穷尽所有的情况，所以 curve 的论域可能比上面还要大，但是实际上用上面的论域来做控制以及足够了，因此在程序中加了一段防止论域溢出的代码。

```
//防止越界
        if(curve < -32) curve = -32;
        else if(curve > 32) curve = 32;
```

在建立输入量的隶属度函数时，我选用的是三角函数。至于为什么要选三角函数，它相对于其他类型的函数有什么优缺点并未深究。当划分好模糊等级后，便是量化输入，在此推荐将输入量经行量化，为什么要量化？我的理解是：由于有两个输入量，且它们的论域不一样，为了保证它们在进行模糊计算时的横、纵坐标的变化范围能保持一致，这样有利于计算前、后沿斜率。而且很多时候输入量是小数，那就更要量化了。仅是我个人的体会，或许不对。对于 curve 的量化过程如下：

```
        fuzzy_input_1 = (uchar)((curve + 32) << 2);
```

同样要防止越界，因为 uchar 型的范围是 0—255. 由于如何建立一个较好的隶属度函数，实际上是比较难的，在此只是一个示意。



有关有网友提出精确的输入是如何对应到论域中去的。我的理解是这样：以上图为例，如果输入的精确值是 6，那么它肯定对应就是小右这个等级，但是如果是 4，那就要看 4 为小右的比例大还是零的比例大了。也就是说跟你的隶属度

函数有关了。

好了，关于输入、输出量总结一下：确定它的论域；划分好它的模糊等级，通常跟主观经验和要求有关；做好量化及防止越界工作；隶属度函数的建立实际上很难，同样跟主观经验和要求有关，最好能用 MATLAB 仿真一下。

下面是我对于 C 语言的模糊代码格式的一些理解。

```
uchar FUZ_INS_1[18];
uchar FUZ_OUTS_1[9];
```

这样的写法是表示在FUZ_INS为首地址后预留18个字节或FUZ_OUT后预留9个字节的空间，为后续的模糊机推理时预备调度空间。

```
const uchar INPUT_MFS_1[18][4]={
                                0x00,0x10,0x00,0x20,
                                0x0c,0x20,0x10,0x10,
                                0x28,0x60,0x09,0x09,
                                0x5c,0x74,0x15,0x15,
                                0x6c,0x94,0x0d,0x0d,
                                0x8c,0xa4,0x15,0x15,
                                0xa0,0xd8,0x09,0x09,
                                0xd4,0xf4,0x10,0x10,
                                0xf0,0xff,0x20,0x00,

                                0x00,0x2d,0x00,0x11,
                                0x1e,0x3c,0x11,0x11,
                                0x2d,0x4b,0x11,0x11,
                                0x3c,0x5a,0x11,0x11,
                                0x4b,0xa5,0x05,0x05,
                                0x96,0xb4,0x11,0x11,
                                0xa5,0xc3,0x11,0x11,
                                0xb4,0xd2,0x11,0x11,
                                0xc3,0xf0,0x11,0x00
                               };
```

以上是两个输入量的隶属度函数的代码表示格式，实际上并不需要一定写成16进制的格式，写代码是直接写成10进制也可以。一行数据的含义是：起点、终点，前沿斜率、后沿斜率。

关于模糊规则库，它的建立很大程度上是根据调试经验来做的，不过能用MATLAB 仿真一下最好。

```
const uchar RULE_START_1[406]={
                                (9*0)+0, (9*1)+0, 0xfe, (9*0)+0+18, 0xfe,
//no.1
                                (9*0)+0, (9*1)+1, 0xfe, (9*0)+0+18, 0xfe,
//no.2
                                (9*0)+0, (9*1)+2, 0xfe, (9*0)+0+18, 0xfe,
//no.3
                                (9*0)+0, (9*1)+3, 0xfe, (9*0)+1+18, 0xfe,
//no.4
                                (9*0)+0, (9*1)+4, 0xfe, (9*0)+2+18, 0xfe,
//no.5
                                (9*0)+0, (9*1)+5, 0xfe, (9*0)+2+18, 0xfe,
//no.6
                                (9*0)+0, (9*1)+6, 0xfe, (9*0)+3+18, 0xfe,
//no.7
                                (9*0)+0, (9*1)+7, 0xfe, (9*0)+3+18, 0xfe,
//no.8
                                (9*0)+0, (9*1)+8, 0xfe, (9*0)+4+18, 0xfe,
//no.9
```

我的理解：“0xfe”是前件与后件的分隔符，“0xff”是规则库结束符。对于上面某一行规则的第二个“0xfe”是否要该成“0xff”，暂时还没有做实验，不过我最终程序中的格式是“0xfe”。还有诸如“(9*0)+0”是完全合法的。其他等我做完试验会立即发布最新调试心得。

时间仓促，今天就写到这里吧。希望各位同仁不吝赐教！谢谢！