# EMPs算法

### 算法前提

设出发点为S，中心为O，暂存点为H，五个重点顺时针为1，2，3，4，5

A，C，E点可以通过前端的超声波测距判断出来

O点可以通过7路红外传感器判断出来

### 必要函数

已知需要的：

void Home(); //回到出发点

void Stop();

uchar GetColor(void); //result：1-绿，2-白，3-红，4-黑，5-蓝

void Forward();

void Backward();

void LeftTurn(); //转弯需要差速控制，这里为了简洁就不传速度进去了，实现时需要修改

void RightTurn();

void LeftTurn\_45();

void LeftTurn\_90();

void LeftTurn\_90\_45();

void RightTurn\_45();

void RightTurn\_90();

void RightTurn\_90\_45();

void Turn\_180(); //原地掉头

void Forward\_S2O(); //前进，由S点至O点

void Forward\_S2C(); //前进，由S点至C点(用于开始出发至C点)

void Forward\_Get(); //前进，至抓取色块(即O点到A-E的距离)

void Forward\_Get\_Return(); //(掉头后),前进至原点(即A-E到O点的距离)

void Forward\_Put(); //前进，至放下色块(即A-E到1-5的距离)

void Forward\_Target(); //前进，从O点直接至A-E处(即Get+Put的距离，但是中途不停)

void Forward\_Target\_Get(); //前进，从O点直接至F、G，并get颜色准备搬运

void Backward\_Point(); //后退，回到原点

void Catch(); //拾取色块

void Down(); //放下色块

### 路径选择策略

几点注意：

1. 尽量走直线，避免直角弯，避免掉头
2. 走场内黑线，不走外圈，不走内圈
3. 设立暂存点，摆脱场内障碍
4. 需要准确定位，尤其是O点的位置，O点是枢纽点，务必每次都能立刻准确停在此点

* 由出发点直行至C：S-->C，

出发直行省时间，也没有必要在中心点停留判断方向

* 判断C点色块颜色，并拾取
  + 红色：直行至3（C-->3），放下色块，退到O点
  + 白色：倒退至O（C-->O），O-->2，放下色块，退到O点
  + 黑色：倒退至O（C-->O），O-->4，放下色块，退到O点
  + 绿色或蓝色：掉头，行至O（C-->O），再向前走几步（O-->H，防止自己以后在O点掉头时撞到色块），放下色块，退到O点
* 若C为红色、白色或者黑色，行至A点（O-->A），判断A点色块颜色，并拾取
  + 非蓝色：中略，放下色块，退到O点
  + 蓝色：退到O点，转头至H，放下色块，退到O点，之后行至E处判断E的颜色搬运即可（详略）
* 若C为绿色，行至E点，判断E的颜色并搬运（因为A不可能为绿色，所以不会挡路不用再去暂存点）；之后去A点，判断A的颜色并搬运；最后回到H点，（判断H点的颜色？）并搬运
* 若C为蓝色，行至A点，判断A的颜色并搬运；之后去E点，判断E点的颜色并搬运；最后回到H点，（判断H点的颜色？）并搬运
* 回到出发点S，至此，一轮搬运结束

至于F点和G点的十个色块，这里提供两种策略供参考：

1、可以经由O点作为转弯点不断来回搬运，效率可能略低，但是算法上和一轮搬运相似并且简单；

2、也可以走外圈，从F处随机搬运色块，从外圈顺时针走到目标处，放下色块，顺便到G点，从G点随机搬运色块，掉头后，从外圈逆时针走到目标处，放下色块，顺便到F点……如此往复，从顺时针到逆时针，从F到G再从G到F，效率会很高，但是算法很难写，另外也实现动态避障（因为1-5点的10环都在路径上，色块必然会形成障碍）也很困难。