C语言课程设计

状态和状态机

States and State Machines

1. 设计题目：三层电梯状态机
2. 设计方案：以每个楼层可能的情况判断分析

通过设计各种状态的翻转，运用状态机模型，来实现电梯的正常运转，首先对所有可能发生的事件进行罗列，并进行归类，到达一种好的状态分析。根据日常经验分析，采取把电梯的运动进行归类，把门分为四种状态，门开（Dooropen），门正在关（DoorClosing），电梯上升（Movingup），下降（Movingdown），而把什么都不做视为第一状态（Idle，以及在里面判断是否有外部事件的发生），在这状态下，可以实现向其他四状态的跳转。

1. 状态图

(D) (S5)

DoorOpen

MovingUp

(S3)

(S1) (S7) (S8)

Idle

(S9)

(S4) (S2)

DoorClosing

(S6)

MovingDown

(D)

1. 状态机功能
2. 初始化
3. 移动（下降模块，上升模块）
4. 开关门

关门模块可能发生的Events：

E1：门内开门按钮（OpenDoorLight）

E2：门内关门按钮(CloseDoorLight)

E3：门内楼层按钮(PanelFloorLight)

目前使用三个按钮（1，2，3）

E4：门外up呼叫按钮（CallLight）

E5：门外down呼叫按钮(CallLight)

1. 函数及调用关系

状态机主要函数及自己发挥部分

viod Idlestate(int \*state); //初始化函数

viod movingup(int \*state); //向上升函数

viod movingdown(int \*state); //下降函数

viod Dooropen(int \*state); //门关闭函数

初始化函数根据条件跳转上升下降及开门函数；

上升函数根据条件跳转开门函数；

门开函数根据条件调用门关函数；

三层电梯使用到的函数

//系统是否运行

bool IsElevatorRunning();

//门外Up/Down呼叫按钮灯（CallLight）

bool GetPanelFloorLight(int floor);

viod SetCallLight(int floor,boor up,bools);

//电梯门内楼层按钮灯

bool GetPanelFloorLight(int floor);

// 电梯门内开关门按钮灯

bool GetOpenDoorLight();

void SetOpenDoorLight(bool s);

bool GetCloseDoorLight();

void SetCloseDoorLight(bool s);

// 电梯箱体门

bool IsDoorOpen(int floor);

bool IsDoorClosed(int floor);

void SetDoor(int floor, bool open);

// 设置电机功率，power=1,全速上升；-1，全速下降；0，停止

void SetMotorPower(double power);

// 一定时间无动作，自动到1楼

extern void AutoTo1Floor();

// 取消自动到1楼

extern void CancelTo1Floor();

// 获取电梯箱体当前所在楼层

double GetFloor(); // 浮点数，如1.5，表示电梯箱体处在1层到2层的中间

int GetNearestFloor(); // GetFloor()函数的四舍五入值

if(fabs(GetFloor() - GetNearestFloor()) < Lib\_FloorTolerance) { 到达GetNearestFloor()层 }

// 静态监测,电梯处于空闲状态, 确定下一步的运动方向和所到楼层(目标楼层)

int IdleWhatFloorToGoTo(bool \*up);

int floor; bool up;

floor = IdleWhatFloorToGoTo(&up);

// 动态监测, 电梯正在上升时，检测将要到达停止的最近楼层(目标楼层) int GoingUpToFloor();

// 动态监测, 电梯正在下降时，检测将要到达停止的最近楼层(目标楼层) int GoingDownToFloor();

1. 分别对每个状态进行分析

Idle状态，电梯停止在某楼层，门是关闭的，处于静止状态，等待相关事件的发生，从而转换到下一个状态。

(S1) 检查E3、E4、E5事件。

静态检测，bool up; 目标楼层=IdleWhatFloorToGoTo(&up); 关闭本层门外up按钮，SetCallLight();

即消费门外up按钮，防止下一周期重复处理此按钮行为。

(S2) 同(S1)，消费门外down按钮。

1层以上，一定时间无动作，自动下降到1楼。AutoTo1Floor();[其它 状态， 取消此功能，CancelTo1Floor()]

(S3) 检查E1事件, 开门, 消费开门按钮;

上升 (up && E4事件), 开门，消费门外up按钮

下降 (!up && E5事件), 开门，消费门外down按钮

(S4) 检查E2事件,此时门应该是关闭的,因此仅读取关门灯，并关闭关门灯，即消费按键行为，防止下一周期重复处理该按钮的行为。

if(GetCloseDoorLight()) { SetCloseDoorLight(false); return; }

MovingUp/MovingDown 跳转到 DoorOpen

(S5) 检查E3、E4、E5事件。动态检测，目标楼层floor=GoingUpToFloor(); if(fabs(GetFloor() - floor) < Lib\_FloorTolerance) 到达目标楼层，停止，开门

消费门外up按钮; 到了最高层Lib\_FloorNum, 消费门外down按钮。消费门内楼层按钮。

(D) 检查E1、E2事件，无动作，消费开/关门按钮。

(S6) 检查E3、E4、E5事件。动态检测，目标楼层=GoingDownToFloor();其它与 (S5)类似。

DoorClosing◊DoorOpen状态: 电梯门打开 跳转到DoorClosing；

(S7) 检查E2事件，转而关门，GetCloseDoorLight(),SetDoor(); 消费关门按 钮。

开门结束后，自动进入关门状态。IsDoorOpen();SetDoor();

检查E1事件, 无动作，消费开门按钮。

DoorOpen◊DoorClosing状态: 正在关门 跳转到

(S8) 检查E1事件，转而开门。GetOpenDoorLight();SetDoor(); 消费关门按钮。 检查E2事件，无动作，消费关门按钮。

Idle◊DoorClosing状态: 正在关门

(S9) 关门结束后，进入Idle状态。IsDoorClosed();

使用switch语句进行判断， 1层以上，一定时间无动作，自动下降到1楼。AutoTo1Floor();[其它状态，取消此功能，CancelTo1Floor()

分别在某一层可能发生的事件进行判断，并进行相应的处理。没有使用部分函数 代码及注解分析如下：

void Idlestate(int \*state)

{

int floor, nowfloor;

bool up;

nowfloor = GoingDownToFloor();

if (nowfloor == 1) { if (GetCallLight(1, 1)) { \*state = DoorOpen;

SetCallLight(1, 1, false);

printf("Transition: from Idlestate to DoorOpen.\n");

return;

}

}

floor = IdleWhatFloorToGoTo(&up); //静态检测上升到那一楼 if (floor > -1) {

}

if (up == false) { }

//关门灯判断

\*state = MovingDown;

printf("Transition: from Idlestate to MovingUp.\n"); return;

\*state = MovingUp;

printf("Transition: from Idlestate to MovingUp.\n"); return;

if (up == true)

//开始消费

//判断电梯楼层按钮程序

if (up && GetPanelFloorLight(floor)) //向上的按钮按了就向上运{ }

if (!up&&GetPanelFloorLight(floor)) { }

//电梯内上下按钮的程序

if (GetCallLight(floor, up) | GetCallLight(floor, !up)) //楼层上下呼叫 {

// SetCallLight(floor, !up, false); \*state = MovingDown;

printf("Transition: from Idlestate to MovingDown.\n"); return;

// SetCallLight(floor, up, false); \*state = MovingUp; return;

//状态翻转

printf("Transition: from Idlestate to MovingUp.\n");

//减少代码运行时间

//开关门函数处理

if (GetCloseDoorLight()) {

SetCloseDoorLight(false);

} } } \*state = DoorClosing;

printf("Transition: from Idlestate to DoorClosing.\n");

return; //开门灯判断

if (GetOpenDoorLight()) { SetOpenDoorLight(false);

\*state = DoorOpen;

printf("Transition: from Idlestate to DoorOpen.\n");

return; ///对自动下降函数的处理

if(nowfloor > 1) AutoTo1Floor();

//printf("进去了1111\n");

void movingup(int \*state)

{

int floor;

bool up = 1;

CancelTo1Floor(); //取消自动到一楼

floor=GoingUpToFloor(); //判断到那一层

if (fabs(GetFloor()-floor) < Lib\_FloorTolerance)

{

if (GetCallLight(floor, up) | GetCallLight(floor, !up)) 层上下呼叫

{

SetCallLight(floor, up, false); SetCallLight(floor, !up, false);

}

if (GetPanelFloorLight(floor)) //灭灯楼层按钮

SetPanelFloorLight(floor, false);

SetMotorPower(0);

SetDoor(floor, 1);

\*state = DoorOpen;

printf("Transition: from Movingup to DowmOpen.\n");

return;

}

else

SetMotorPower(1);

void movingdown(int \*state)

{

int floor;

CancelTo1Floor();

floor = GoingDownToFloor();

if (GetPanelFloorLight(floor)) SetPanelFloorLight(floor, false);

{

if (fabs(GetFloor() - floor) < Lib\_FloorTolerance)

{ if (GetCallLight(floor, up) | GetCallLight(floor, !up))//楼层上下呼叫

SetCallLight(floor,up,false);

SetCallLight(floor, !up, false);

} if (GetPanelFloorLight(floor)) { SetPanelFloorLight(floor, false);

SetMotorPower(0);

SetDoor(floor, 1);

\*state = DoorOpen;

printf("Transition: from movingdown to DoorOpen.\n");

return; }

else SetMotorPower(-1);}

void Dooropen(int \*state)

{

int floor;

floor = GetNearestFloor();

CancelTo1Floor();

if (GetCloseDoorLight()) //有关门按键就关门

{SetDoor(floor, false);

SetCloseDoorLight(false);

\*state = DoorClosing;

return;

}

else if (IsDoorOpen(floor)) //如果门是关闭的就进去，返回门是否是 打开的，true表示门是打开的，否则门是关闭的或者正在打开/关闭

{

SetDoor(floor, false);

\*state = DoorClosing;

return;

}

if (GetOpenDoorLight()) //如果开门按钮打开

{

SetDoor(floor, true);

SetOpenDoorLight(false);

}}

void Doorclosing(int \*state)

{int floor;

floor = GetNearestFloor(); //判断到最近的楼层

CancelTo1Floor();

if (GetCloseDoorLight())

{SetCloseDoorLight(false);

SetDoor(floor, false);}

else if (GetOpenDoorLight())

{SetDoor(floor, true); //开门

SetOpenDoorLight(false);

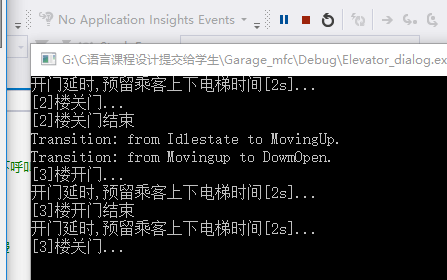
\*state = DoorOpen; return; }

else if (IsDoorClosed(floor))

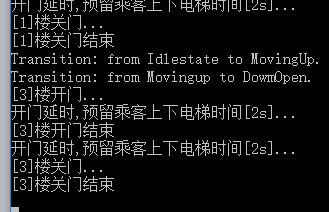
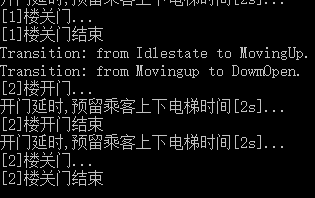
{\*state = Idle; return;}}

7.系统运行测试结果分析

测试一：电梯内楼层按键的使用及演示



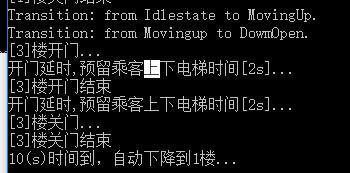
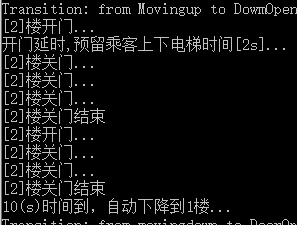
测试二：楼层外上下按钮的使用及演示



测试三：开关门按钮的使用及演示



测试四：自动到一楼



综合测试分析

