## 三层电梯课程设计报告

**1.状态机图及其分解描述**：

将三层升降电梯的运行过程分为空闲，向上，向下，开门，关门五个状态。图附最后。

Idle（空闲）状态及其事件

S1：空闲到空闲，门本来就关着，只要门内关门灯关闭即可。

S2：空闲到上升，检查三种情况，点亮门内的上楼键，门外的上下楼键。

S3：空闲到开门，对应的事件是当前楼层门外呼叫或者门内开门按钮点亮。

S4: 空闲到下降，与S2恰好相反。

MovingUp/Down（上下）状态及其事件

S6/S10由向上升/向下降到开门，情况一是目的楼层到了，自动开门；情况二是若上升/下降的过程中途，楼层同方向被呼叫，则在中途楼层停止，到开门状态。

S11： 在运动过程中出于安全考虑不允许开关门，若有人按了开关门按键，无反应，不执行。

DoorOpen（开门）状态及其事件

S7：由开门到关门的过程。对应三种情况，一是门内人手动关门；二是开门动作结束后，门自动关闭；三是在开门状态下按门内开门按钮电梯无反应，只是灭掉开门按钮的灯。

DoorClosing（关门）状态及其事件

S8：关门到开门，对应门内开门按键被按亮，电梯开门。

S9：关门到空闲，关门结束后，电梯没有下一步动作，处于空闲的状态。

**2.各状态函数设计过程**

共有5个事件，列举如下：

E1: 门内开门按钮(OpenDoorLight)

E2: 门内关门按钮(CloseDoorLight)

E3: 门内楼层按钮(PanelFloorLight)

E4: 门外up呼叫按钮(Call Light)

E5: 门外down呼叫按钮(Call Light)

**空闲状态**：

(S1) 检查E2 GetCloseDoorLight(),并消费按键行为。

(S2) 检查E3、E4、E5事件。（根据布尔变量up的值，以及 IdleWhatFloorToGoTo（）返回值来确定电梯下一步应该是怎样的状属于静态检测。bool up; floor=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。

(S3) 检查E1事件, 开门, 消费开门按钮;

上升，开门，消费门外up按钮；

下降，开门，消费门外down按钮。

(S4) 检查E3 E4 E5事件,(和S2相反）

**运动中状态：**MovingUp/MovingDown 🡪 DoorOpen

(S6) 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，消费门外按钮。

(S11) 检查E1、E2事件，无动作，消费开/关门按钮。

(S10) 检查E3、E4、E5事件。动态检测，目标楼层=GoingDownToFloor();其它与(S2)类似。

**开门状态:** 电梯门打开 🡪 DoorClosing

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

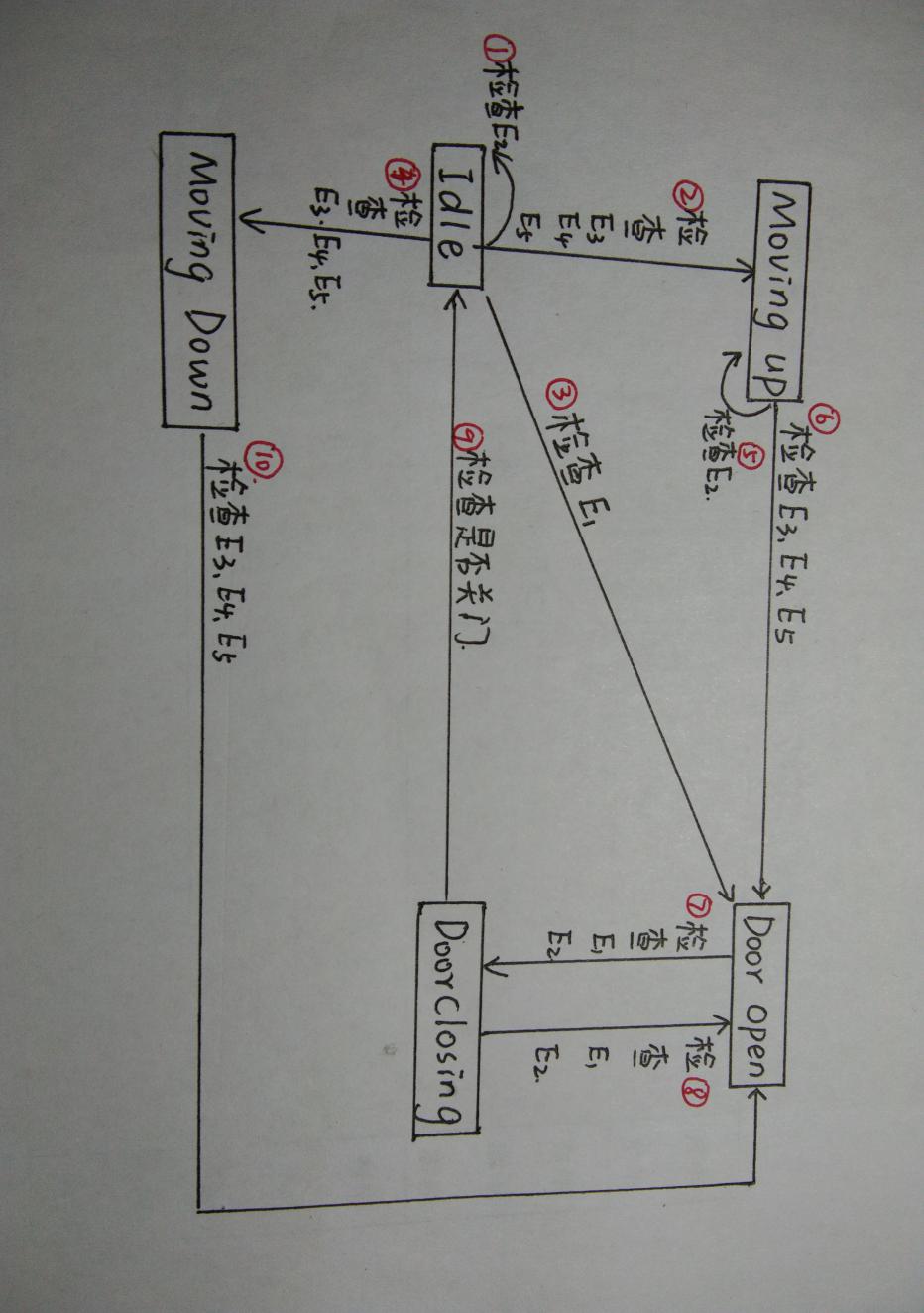
开门结束后，自动进入关门状态。IsDoorOpen();SetDoor();

检查E1事件, 无动作，消费开门按钮。

**关门状态:**

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

(S9) 关门结束后，进入Idle状态。IsDoorClosed();



#include "stdafx.h"

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include "ElevatorLib.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 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。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void StateIdle(int \*state)

{

int floor;bool up;

floor=IdleWhatFloorToGoTo(&up);//检测将要到的目标楼层

if(floor>0&&up)

{

SetMotorPower(1);

\*state=MovingUp;

printf("transition:from Idle to MovingUp");

}

if(floor>0&&!up);

{

SetMotorPower(-1);

\*state=MovingDown;

printf("transition:from Idle to MovingDown");

}

if(GetOpenDoorLight())//检查E1事件

{

SetDoor(floor,1);

SetOpenDoorLight(0);

\*state=DoorOpen;

printf("transition:from Idle to DoorOpen");

}

if(GetCallLight())//检查E4/E5事件

{

SetDoor(floor,1);

SetCallLight(floor,1,0);

SetCallLight(floor,0,0);

\*state=DoorOpen;

printf("transition:from Idle to DoorOpen");

}

if(GetCloseDoorLight())//关门

{

SetCloseDoorLight(0);

return;

\*state=Idle;

}

}

void StateMovingUp(int \*state)

{

int floor;bool up;

floor=GoingUpToFloor();//动态检测目的楼层

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

{

SetMotorPower(0);

SetDoor(floor,1);

\*state=DoorOpen;

SetCallLight(floor,1,0);

if(floor==Lib\_FloorTolerance)

{

SetCallLight(floor,0,0);

}

SetPanelFloorLight(floor,0);

if(GetOpenDoorLight()||GetCloseDoorLight())

{

SetOpenDoorLight(0);

SetCloseDoorLight(0);

}

printf("transition:from MovingUp to DoorOpen");

}

}

void StateMovingDown(int \*state)

{

int floor; bool up;

floor = GoingDownToFloor();

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

{

SetMotorPower(0);

SetDoor(floor, 1);

\*state = DoorOpen;

SetCallLight(floor, 0, 0);

if (floor == 1)

{

SetUpLight(floor, 1, 0);

}

SetPanelFloorLight(floor, 0);

}

if (GetOpenLight()) || GetCloseDoorLight())

{

SetOpenDoorLight(0);

SetCloseDoorLight(0);

}

}

void StateDoorOpen(int \*state)

{

int floor;bool up;

floor=GetNearstFloor();//获取当前楼层

if(GetCloseDoorLight())

{

SetDoor(floor,1);

SetCloseDoorLight(0);

\*state=DoorClosing;

}

if(IsDoorOpen())

{

SetDoor(floor,1);

\*state=DoorClosing;

}

if(GetOpenDoorLight())

{

SetOpenDoorLight(0);

}

printf("transition:from DoorOpen to DoorClosing");

}

void StateDoorClosing(int \*state);

{

int floor;bool up;

floor=GetNearestFloor();

if(GetOpenDoorLight())

{

SetDoor(floor,1);

SetOpenDoorLight(0);

\*state=DoorOpen;

}

if(GetCloseDoorLight())

{

SetCloseDoorLight(0);

}

if(IsBeamBroken())

{

SetDoor(floor,1);

\*state=DoorOpen;

}

printf("transition:from DoorClosing to DoorOpen");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 状态机，每隔一定时间(如，100ms)被调用一次，采集系统的运行状态

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void main\_control(int \*state)

{

if(IsElevatorRunning()) // 仿真正在运行

{

switch(\*state)

{

case Idle:

// Idle状态，一定时间无动作，自动到一楼

if(GetNearestFloor() !=1 ) {

AutoTo1Floor();

}

StateIdle(state);

break;

case MovingUp:

CancelTo1Floor(); // 其它状态，取消自动到一楼

StateMovingUp(state);

break;

case MovingDown:

CancelTo1Floor();

StateMovingDown(state);

break;

case DoorOpen:

CancelTo1Floor();

StateDoorOpen(state);

break;

case DoorClosing:

CancelTo1Floor();

StateDoorClosing(state);

break;

default:

printf("没有这种状态!!!\n");

}

}

}

**16040310027 雷莹**