|  |
| --- |
|  |
| 三层电梯状态机课程设计报告 |
|  |

|  |
| --- |
| 1604031008  罗跃 |

**一**.**状态机图及其课程设计报告。**

if(GetCloseDoorLight())

SetCloseDoorLight();

return;

Idle

DoorOpen

if (GetOpenDoorLight()) SetDoor();

SetOpenDoorLight();\*state = DoorOpen;

if (IsBeamBroken()) SetDoor();

DoorClosing

if (GetCloseDoorLight())

SetDoor();

SetCloseDoorLight();

\*state = DoorClosing;

if (IsDoorClosed(floor))

\*state = Idle;

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

SetMotorPower(0);SetDoor();SetPanelFloorLight();

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

SetMotorPower(0);SetDoor();\*state=DoorOpen;

if (Lib\_FloorNum)

SetCallLight;

SetPanelFloorLight;

if (GetOpenDoorLight())

SetDoor();

if(GetCallLight())SetDoor();

if (floor>0 && !up) SetMotorPower(-1);

if (floor > 0 && up)

SetMotorPower(1)

MovingDown

MovingUp

**二.运行功能测试。**

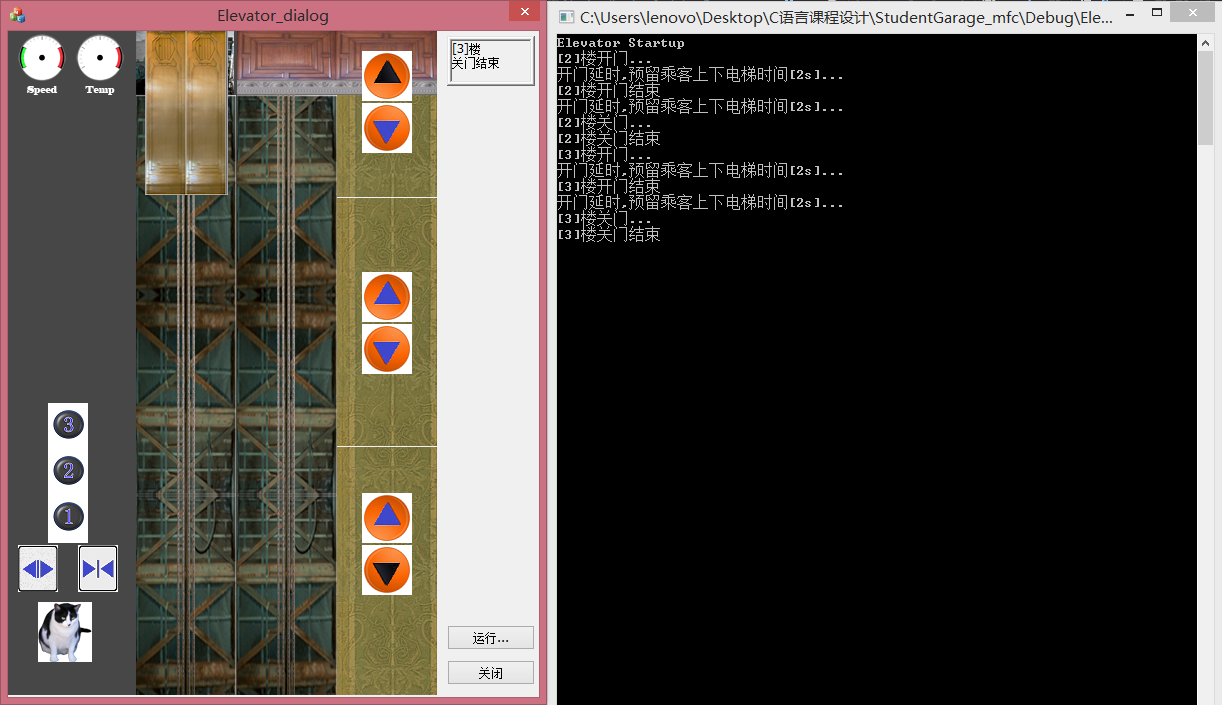
1. **电梯停于1F或2F时，按3F向下呼叫按钮；电梯上升到3F停止，开门/关门。**
2. **电梯停于2F或3F时，按1F向上呼叫按钮；电梯下降到1F停止，开门/关门。**
3. **电梯停于1F时，按2F向上呼叫按钮；电梯上升到2F停止，开门/关门。**
4. **电梯停于3F时，按2F向下呼叫按钮；电梯下降到2F停止，开门/关门。**
5. **电梯停于1F，2F和3F均有按钮呼叫；电梯先上升到2F，开门/关门，然后上升到3F停止，开门/关门。**
6. **电梯停于3F，2F和1F均有按钮呼叫；电梯先下降到2F，开门/关门，然后下降到1F停止，开门/关门。**

**7．电梯上升途中或下降途中，任何反方向按钮呼叫均无效。**

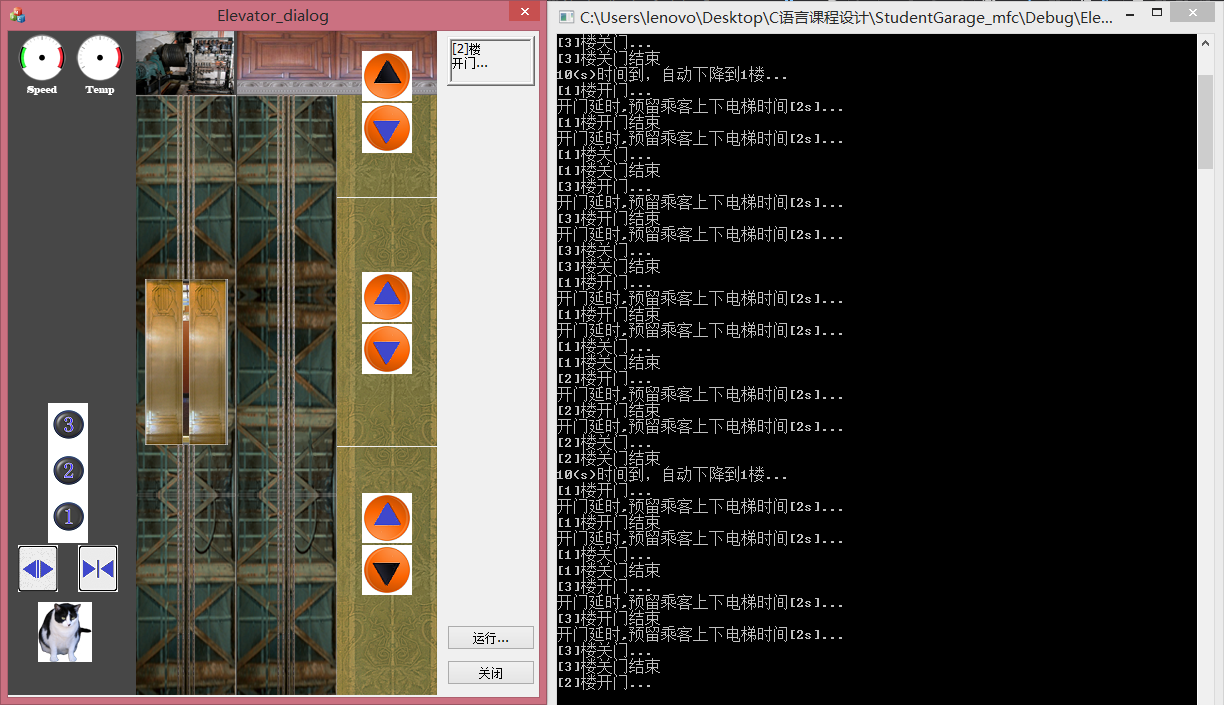
1. **电梯停于1F，按3F向下呼叫按钮，然后立即按2F向下呼叫按钮；电梯上升到3F停止，开门/关门，然后下降到2F停止，开门/关门。**
2. **电梯停于2F，按门内楼层按钮3，然后门内楼层按钮1；电梯上手到3F停止，开门/关门，然后下降到1F，开门/关门。**
3. **电梯停于1F，按门内楼层按钮3，当电梯上升在1F到2F中间以下，按2F向上呼叫按钮；电梯先上升到2F，开门/关门；然后再到3F，开门/关门。**
4. **电梯停于1F，按门内楼层按钮3，当电梯上升在1F到2F中间以上，按2F向上呼叫按钮；电梯先上升到3F，开门/关门；然后再到2F，开门/关门。**
5. **电梯在2F以上，10s无动作，自动降到1楼。**
6. **所有停止，开门/关门后，对应楼层的同方向门外呼叫按钮灯（最高楼向下呼叫按钮，最底层向上呼叫按钮）和门内楼层按钮灯关闭。**
7. **运动状态，开关门按钮失效。正在开门，开门按钮失效；正在关门，关门按钮失效。**
8. **电梯停于1F，按2F向下呼叫按钮和向上呼叫按钮以及3F的向下呼叫按钮；电梯上升到2F停止，开门/关门，2F的向上呼叫按钮灯关闭，开门/关门， 2F的向下呼叫按钮灯关闭，然后上升到3F停止，开门/关门，3F的向下呼叫按钮关闭。**
9. **空闲状态，门是关闭的，因此按关门按钮失效。**
10. **开关门结束，延时2秒用于乘客上下电梯【延时功能在库函数中实现，不用在状态函数中实现】，然后进入关门状态。**
11. **开门结束前，按关门按钮，转而进入关门状态；关门结束前，按开门按钮，转而进入开门状态。**

**经测试，运行无误，结果如下：**

**Idle MovingUp**

****

**Idle MovingDown**

****

**Idle DoorOpen**

****

**Idle Idle**

****

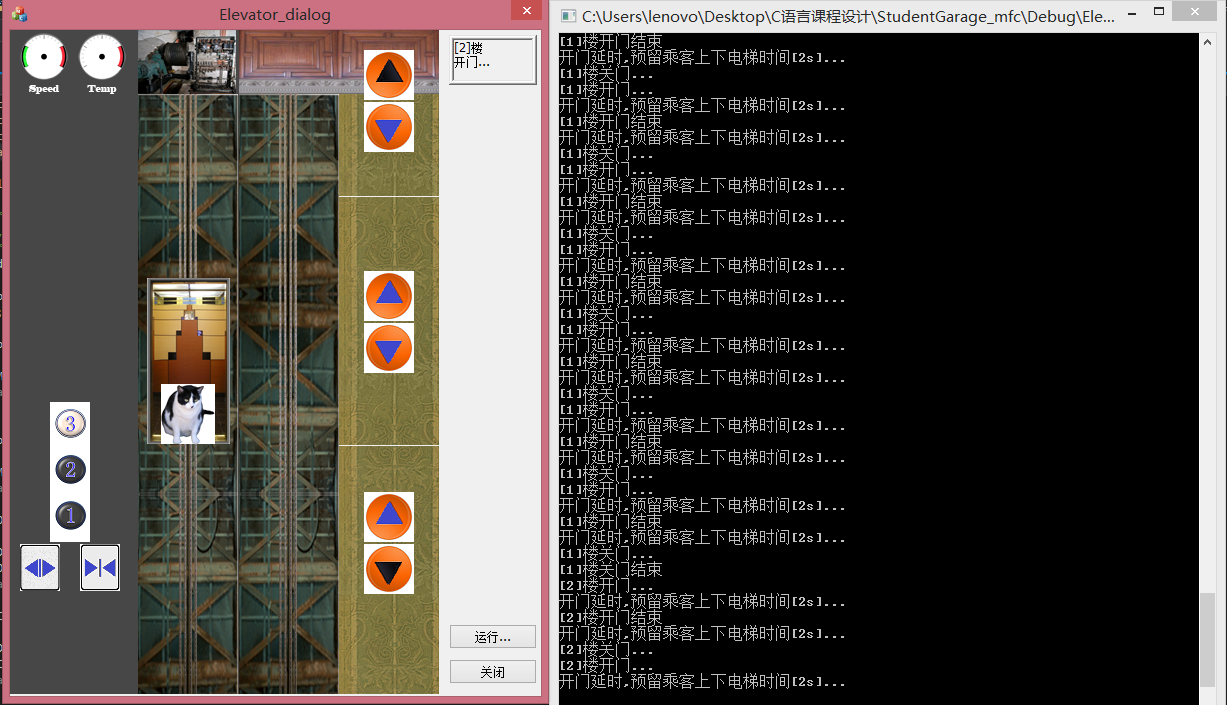
**MovingUp DoorOpen**

****

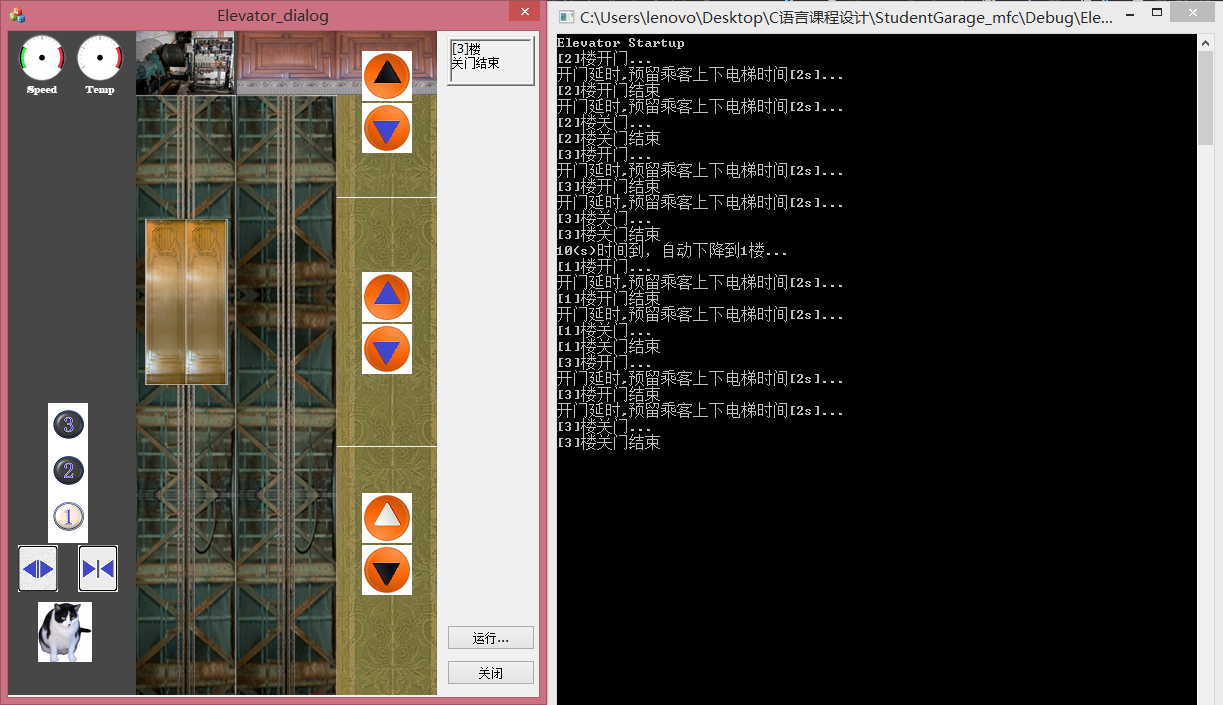
**MovingDown DoorOpen**

****

**DoorOpen DoorClosing**

****

**DoorClosing Idle**

****

**代码如下：**

#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);

if (floor > 0 && up) //Event

{

SetMotorPower(1); //Transition

\*state = MovingUp; //进入MovingDown状态

}

floor = IdleWhatFloorToGoTo(&up);

if (floor>0 && !up) //Event

{

SetMotorPower(-1); //Transition

\*state = MovingDown; //进入MovingDown状态

}

if (GetOpenDoorLight()) //Event

{

SetDoor(GetNearestFloor(), true); //消费开门按键

SetOpenDoorLight(false);

\*state = DoorOpen; //进入DoorOpen状态

}

if (GetCallLight(GetNearestFloor(), true)) //Event

{

SetDoor(GetNearestFloor(), true); //消费开门按键

SetCallLight(GetNearestFloor(), true, false); //消费门外up/down按键

\*state = DoorOpen; //进入DoorOpen状态

}

if (GetCallLight(GetNearestFloor(), false)) //Event

{

SetDoor(GetNearestFloor(), true); //消费开门按键

SetCallLight(GetNearestFloor(), false, false);

\*state = DoorOpen; //进入DoorOpen状态

}

if (GetCloseDoorLight()) //Event 消费关门灯按键

{

SetCloseDoorLight(false);

return;

}

}

/\* MovingUp状态函数 \*/

void StateMovingUp(int \*state)

{

int floor;

floor = GoingUpToFloor();

if (fabs(GetFloor() - floor) < Lib\_FloorTolerance) //Event 判断是否到达目标楼层

{

SetMotorPower(0); //Transition 停止

SetDoor(GetNearestFloor(), true);

\*state = DoorOpen;

if (GetNearestFloor() != Lib\_FloorNum) //Event 获取当前楼层

SetCallLight(GetNearestFloor(), true, false); //消费门外up按钮

else (SetCallLight(GetNearestFloor(), false, false)); //消费门外down按钮

}

SetPanelFloorLight(GetNearestFloor(), false); //消费门内楼层按钮

if (GetOpenDoorLight()) //Event

SetOpenDoorLight(false); //消费开门按键

if (GetCloseDoorLight()) //Event

SetCloseDoorLight(false); //消费关门按键

}

/\* IMovingDown状态函数 \*/

void StateMovingDown(int \*state)

{

int floor;

floor = GoingDownToFloor();

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

{

SetMotorPower(0); //Transition

SetDoor(GetNearestFloor(), true);

\*state = DoorOpen; //进入DoorOpen状态

}

if (GetCallLight(GetNearestFloor(), true)) //Event

SetCallLight(GetNearestFloor(), true, false); //消费down按钮

else if (Lib\_FloorNum) //Event

{

SetCallLight(GetNearestFloor(), false, false);

}

SetPanelFloorLight(GetNearestFloor(), false); //消费门内楼层按钮

SetCallLight;

if (Lib\_FloorNum) //Event

SetCallLight;

SetPanelFloorLight;

}

/\* DoorOpen状态函数 \*/

void StateDoorOpen(int \*state)

{

int floor;

floor = GetNearestFloor();

if (GetCloseDoorLight()) //Event

{

SetDoor(floor, false);

SetCloseDoorLight(false);

\*state = DoorClosing; //进入DoorClosing状态

}

if (IsDoorOpen(floor)) //Event

{

SetDoor(floor, false); //自动进行关门

\*state = DoorClosing; //进入DoorClosing状态

}

if (GetCloseDoorLight()) //Event

SetOpenDoorLight(false); //消费开门按钮

}

/\*DoorClosing状态函数 \*/

void StateDoorClosing(int \*state)

{

int floor;

floor = GetNearestFloor();

if (GetOpenDoorLight()) //Event

{

SetDoor(floor, true);

SetOpenDoorLight(false);

\*state = DoorOpen; //进入DoorOpen状态

}

if (GetCloseDoorLight()) //Event

SetCloseDoorLight(false); //消费关门按钮

if (IsBeamBroken()) //Event 是否红外探测到遮挡物

{

SetDoor(floor, true);

\*state = DoorOpen; //进入DoorOpen状态

}

if (IsDoorClosed(floor)) //Event

\*state = Idle; //进入Idle状态

}

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

\* 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");

}

}

}