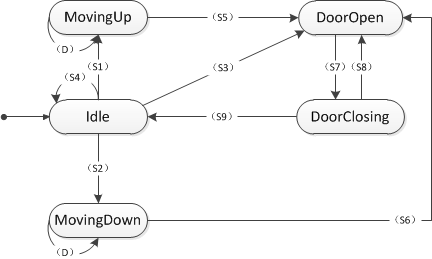
|  |  |  |
| --- | --- | --- |
| 状态函数 | 状态函数 | 检查事件，执行动作（出弧标识），状态变迁 |
| IDLE | stateIDLE | S1 s2 s3 s4 |
| Moving up | statemovingup | S5 d |
| Moving down | statemovingdown | S6 d |
| Door open | statedooropen | S7 |
| Door closing | statedoorclosing | S8 s9 |



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)

E4: 门外up呼叫按钮(Call Light)

E5: 门外down呼叫按钮(Call Light)

思路一：根据状态的变化分析

S1:

电梯上升，IDLE------Moving up

1)检查E3事件：门内按下电梯当前所在层之上的楼层

2)检查E4事件：电梯当前所在层之上的楼层按下up按钮

3)检查E5事件：电梯当前所在层之上的楼层按下down按钮

S2：

电梯下降，IDLE-----Moving down

1)检查E3事件：门内按下电梯当前所在层之下的楼层

2)检查E4事件：电梯当前所在层之下的楼层按下up按钮

3)检查E5事件：电梯当前所在层之下的楼层按下down按钮

S3：

IDLE-----door open

1)检查E1事件：门内按下开门按钮-------开门--------消费门内开门键

2)检查E4事件：电梯所在楼层门外按下up按钮----开门-----消费门外up键

3)检查E5事件：电梯所在楼层门外按下down按钮------开门------消费门外down键

S4：

IDLE------IDLE

1)检查E2事件：无动作------消费门内关门键

S5：

1.Moving up--------door open

1)检查E3事件：电梯到达门内按钮选择的楼层----------开门------消费门内为当前楼层的按钮

2)检查E4事件：电梯到达门外up键被按下的楼层-------开门-------消费门外up键

3)检查E5事件：电梯到达门外down键被按下的楼层-------若该层不是最高层，则电梯不停；若该层为最高层，则电梯停下，开门，消费门外down键

2.Moving up-----Moving up

1)检查E1，E2事件-----运动状态门内开关门按钮失效-----消费门内开关门按钮

S6：

1.Moving down-------door open

1)检查E3事件：电梯到达门内按钮选择的楼层----------开门------消费门内为当前楼层的按钮

2)检查E5事件：电梯到达门外down键被按下的楼层--------------开门-------消费门外down键

3)检查E4事件：电梯到达门外up键被按下的楼层-------若该层不是最低层，则电梯不停；若该层为最低层，则电梯停下，开门，消费门外up键

2.Moving down----Moving down

1)检查E1，E2事件-----运动状态门内开关门按钮失效-----消费门内开关门按钮

S7：

1.Door open-----door closing

1)检查E1 事件：继续开门-----消费开门按钮

2)检查E2事件：关门-----消费关门按钮

2.Door open-----door closing

1)开门结束-----自动关门

S8：

1. Door closing-----door open

检查E1事件：开门-----消费开门按钮

2.Door closing----door closing

检查E2事件：继续关门------消费门内关门按钮

3.Door closing----door open

若有障碍物遮挡------开门

S9：

door closing----IDLE

关门结束

思路二：根据状态分析

IDLE

检查E1事件：门内按下开门按钮-------开门--------消费门内开门键

IDLE-----door open

检查E2事件：无动作------消费门内关门键

IDLE------IDLE

检查E3事件：门内按下电梯当前所在层之上的楼层

电梯上升，IDLE------Moving up

门内按下电梯当前所在层之下的楼层

电梯下降，IDLE-----Moving down

检查E4事件：电梯当前所在层之上的楼层按下up按钮

电梯上升，IDLE------Moving up

电梯当前所在层之下的楼层按下up按钮

电梯下降，IDLE-----Moving down

检查E5事件：电梯当前所在层之上的楼层按下down按钮

电梯上升，IDLE------Moving up

电梯当前所在层之下的楼层按下down按钮

电梯下降，IDLE-----Moving down

Moving up：

检查E1，E2事件：运动状态门内开关门按钮失效-----消费门内开关门按钮

Moving up-----Moving up

检查E3事件：电梯到达门内按钮选择的楼层----------开门------消费门内为当前楼层的按扭

Moving up--------door open

检查E4事件：电梯到达门外up键被按下的楼层-------开门-------消费门外up键

Moving up--------door open

检查E5事件：电梯到达门外down键被按下的楼层-------若该层不是最高层，则电梯不停；若该层为最高层，则电梯停下，开门，消费门外down键

Moving up--------door open

Moving down：

检查E1，E2事件：运动状态门内开关门按钮失效-----消费门内开关门按钮

Moving down----Moving down

检查E3事件：电梯到达门内按钮选择的楼层----------开门------消费门内为当前楼层的按钮

Moving down-------door open

检查E4事件：电梯到达门外up键被按下的楼层-------若该层不是最低层，则电梯不停；若该层为最低层，则电梯停下，开门，消费门外up键

Moving down-------door open

检查E5事件：电梯到达门外down键被按下的楼层--------------开门-------消费门外down键

Moving down-------door open

Door open：

检查E1 事件：继续开门-----消费开门按钮

Door open-----door open

检查E2事件：关门-----消费关门按钮

Door open-----door closing

开门结束-----自动关门

Door open-----door closing

Door closing：

检查E1事件：开门-----消费开门按钮

Door closing-----door open

检查E2事件：继续关门------消费门内关门按钮

Door closing----door closing

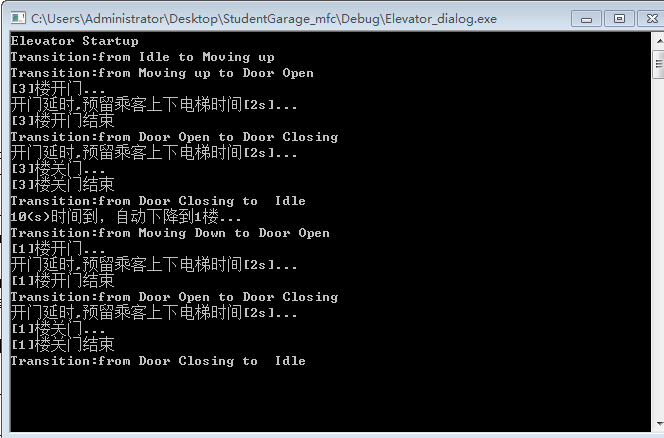
若有障碍物遮挡------开门

Door closing----door open

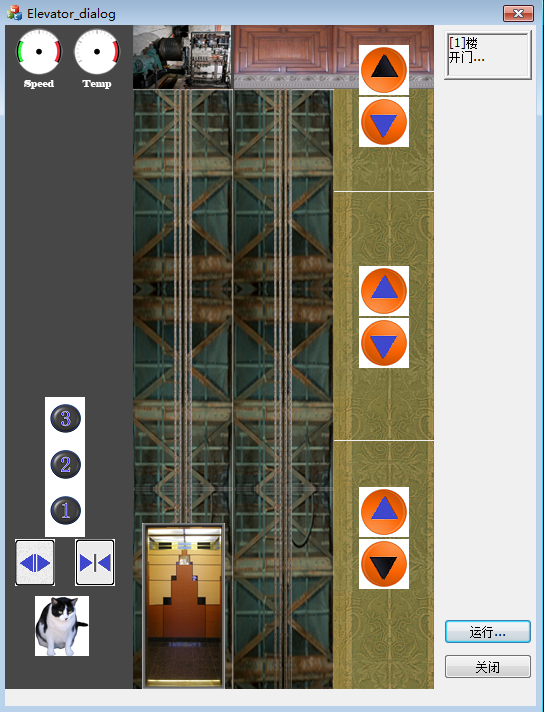
关门结束：door closing----IDLE

检测结果

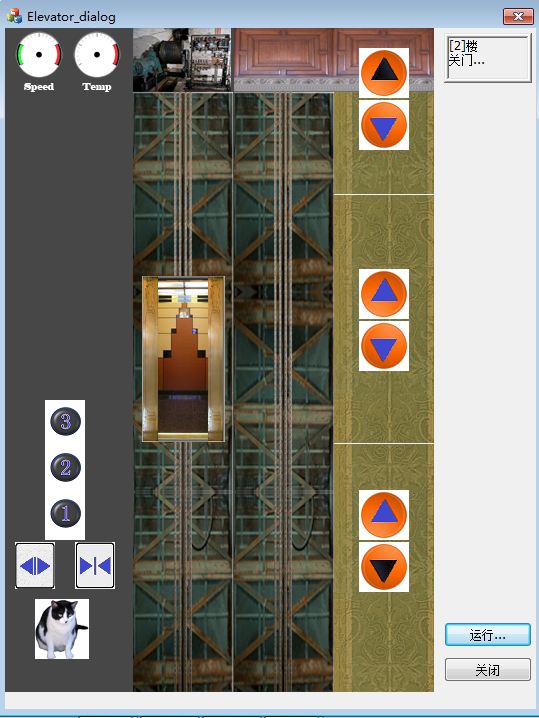
1电梯停于1F，按3F向下呼叫按钮；电梯上升到3F停止，开门/关门。



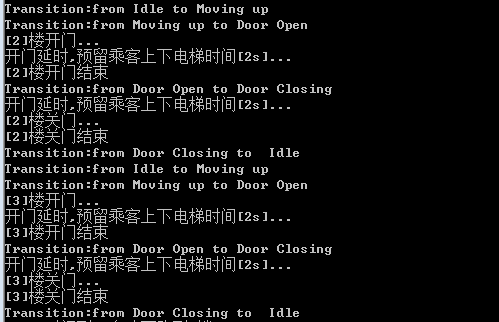
2电梯停于3F时，按1F向上呼叫按钮；电梯下降到1F停止，开门/关门。



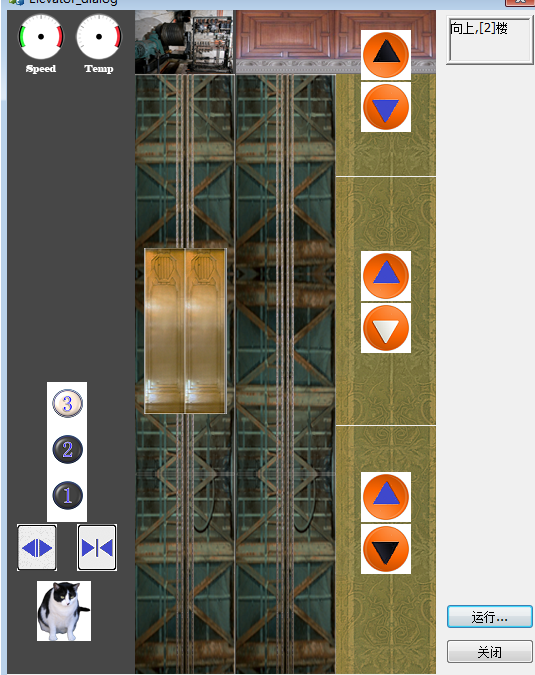
3电梯停于3F时，按2F向下呼叫按钮；电梯下降到2F停止，开门/关门。

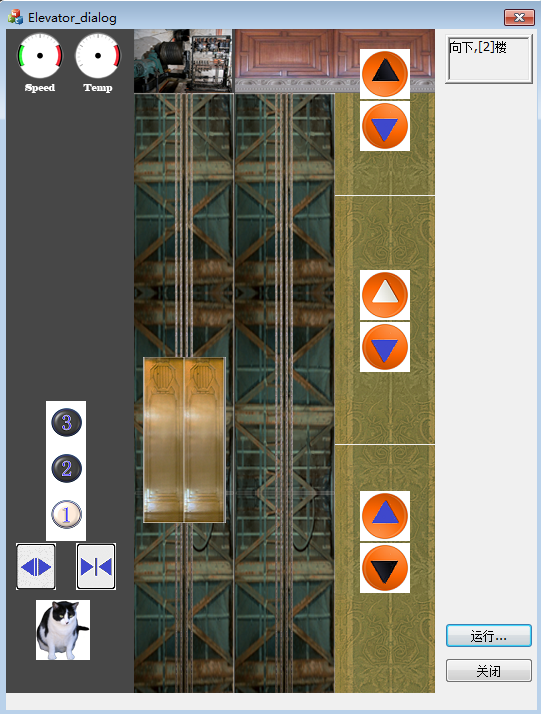


4电梯停于1F，2F和3F均有按钮呼叫；电梯先上升到2F，开门/关门，然后上升到3F停止，开门/关门。

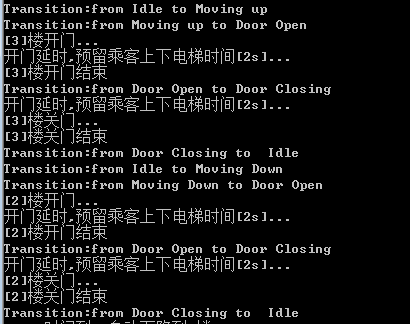


5电梯上升途中或下降途中，任何反方向按钮呼叫均无效。

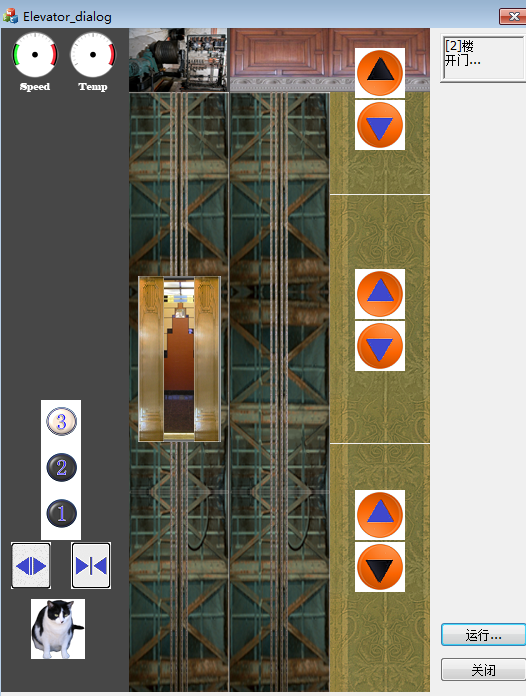




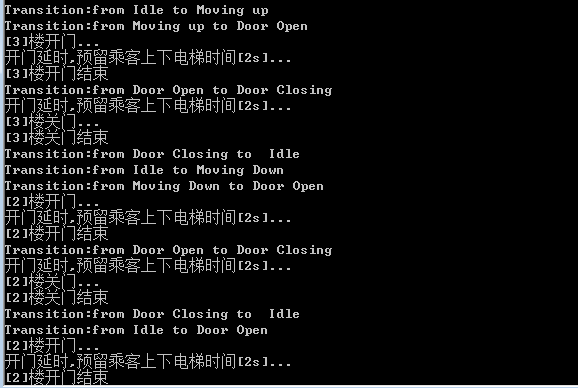
6电梯停于1F，按3F向下呼叫按钮，然后立即按2F向下呼叫按钮；电梯上升到3F停止，开门/关门，然后下降到2F停止，开门/关门。



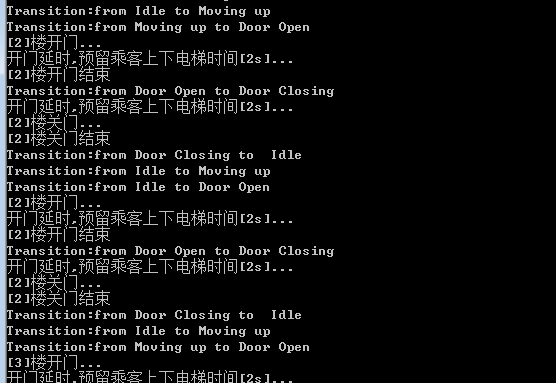
7电梯停于1F，按门内楼层按钮3，当电梯上升在1F到2F中间以下，按2F向上呼叫按钮；电梯先上升到2F，开门/关门；然后再到3F，开门/关门。



8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。



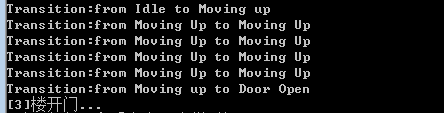
9电梯停于1F，按2F向下呼叫按钮和向上呼叫按钮以及3F的向下呼叫按钮；电梯上升到2F停止，开门/关门，2F的向上呼叫按钮灯关闭，开门/关门， 2F的向下呼叫按钮灯关闭，然后上升到3F停止，开门/关门，3F的向下呼叫按钮关闭。



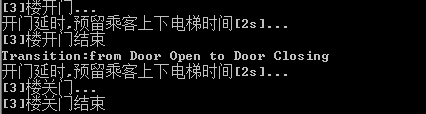
10电梯在2F以上，10s无动作，自动降到1楼。



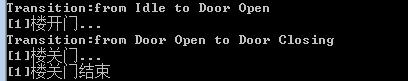
11运动状态，开关门按钮失效。正在开门，开门按钮失效；正在关门，关门按钮失效。



12开关门结束，延时2秒用于乘客上下电梯，然后进入关门状态。



13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。



代码：

#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)

{

bool up;

int floor;

floor = IdleWhatFloorToGoTo(&up);//静态检测将要到达的目标楼层

if (floor>0 && up)

{

SetMotorPower(1);//检验门内楼层按钮，进入上升状态

\*state = MovingUp;

printf("Transition:from Idle to Moving up\n");

}

if (floor>0 && !up)

{

SetMotorPower(-1);//检验门内楼层按钮，进入下降状态

\*state = MovingDown;

printf("Transition:from Idle to Moving Down\n");

}

if (GetOpenDoorLight())

{

SetDoor(GetNearestFloor(), 1);

SetOpenDoorLight(0);//检验门内开门按钮，进入开门状态

\*state = DoorOpen;

printf("Transition:from Idle to Door Open\n");

}

if (GetCallLight(GetNearestFloor(), 1))

{

SetDoor(GetNearestFloor(), 1);

SetCallLight(GetNearestFloor(), 1, 0); //检验门外向上呼叫按钮，进入开门状态

\*state = DoorOpen;

printf("Transition:from Idle to Door Open\n");

}

if (GetCallLight(GetNearestFloor(), 0))

{

SetDoor(GetNearestFloor(), 1);

SetCallLight(GetNearestFloor(), 0, 0);//检验门外呼叫按钮，进入开门状态

\*state = DoorOpen;

printf("Transition:from Idle to Door Open\n");

}

if (GetCloseDoorLight())

{

SetCloseDoorLight(0);//检验门内关门按钮，无响应

return;

printf("Transition:from Idle to Idle\n");

}

}

void StateMovingUp(int \*state)

{

int floor;

floor = GoingUpToFloor();//动态检测，目标楼层

if (fabs(GetFloor() - floor) < Lib\_FloorTolerance)//到达目标楼层

{

SetMotorPower(0);//关闭电机

SetDoor(floor, 1);//开门

SetPanelFloorLight(floor, 0);//消费门内楼层按钮为当前楼层的按钮

SetCallLight(floor, 1, 0);//消费当前楼层门外向上呼叫按钮

if (floor == Lib\_FloorNum)

{

SetCallLight(floor, 0, 0);//到了最高层，消费最高层门外向下呼叫按钮

}

\*state = DoorOpen;//进入开门状态

printf("Transition:from Moving up to Door Open\n");

}

if (GetOpenDoorLight() || GetCloseDoorLight())

{

SetOpenDoorLight(0);//运动过程中，门内开关门按钮失效

SetCloseDoorLight(0);

printf("Transition:from Moving Up to Moving Up\n");

}

}

void StateMovingDown(int \*state)

{

int floor;

floor = GoingDownToFloor();//动态检测目标楼层

if (fabs(GetFloor() - floor) < Lib\_FloorTolerance)//到达目标楼层

{

SetMotorPower(0);//关闭电机

SetDoor(floor, 1);//开门

SetCallLight(floor, 0, 0);//消费当前楼层门外向下呼叫按钮

if (floor == 1)

SetCallLight(floor, 1, 0);//到达一层，消费一层门外向上呼叫按钮

SetPanelFloorLight(floor, 0);//消费门内楼层按钮为当前楼层的按钮

\*state = DoorOpen;//进入开门状态

printf("Transition:from Moving Down to Door Open\n");

}

if (GetOpenDoorLight() || GetCloseDoorLight())

{

SetOpenDoorLight(0);

SetCloseDoorLight(0);//运动状态，门内开关门按钮失效

printf("Transition:from Moving Down to Moving Down\n");

}

}

void StateDoorOpen(int \*state)

{

int floor;

floor = GetNearestFloor();//静态检测当前楼层

if (GetCloseDoorLight())//检测关门按钮

{

SetDoor(floor, 0);//关闭电梯门

SetCloseDoorLight(0);//消费关门按钮

\*state = DoorClosing;//进入关门状态

printf("Transition:from Door Open to Door Closing\n");

}

if (IsDoorOpen(floor))//开门结束后

{

SetDoor(floor, 0);//自动进行关门

\*state = DoorClosing;//进入关门状态

printf("Transition:from Door Open to Door Closing\n");

}

if (GetOpenDoorLight())//检验开门按钮

{

SetOpenDoorLight(0);//正在开门，开门按钮失效

printf("Transition:from Door Open to Door Open\n");

}

}

void StateDoorClosing(int \*state)

{

int floor;

floor = GetNearestFloor();//静态检测当前楼层

if (GetOpenDoorLight())//检验开门按钮

{

SetDoor(floor, 1);//开门

SetOpenDoorLight(0);//消费开门按钮

\*state = DoorOpen;//进入开门状态

printf("Transition:from Door Closing to Door Open\n");

}

if (GetCloseDoorLight())//检验关门按钮

{

SetCloseDoorLight(0);//正在关门，关门按钮失效

printf("Transition:from Door Closing to Door Closing\n");

}

if (IsBeamBroken())//检测是否有障碍物

{

SetDoor(floor, 1);//开门

\*state = DoorOpen;//进入开门状态

printf("Transition:from Door Closing to Door Open\n");

}

if (IsDoorClosed(floor))//关门结束

{

\*state = Idle;//进入ldle状态

printf("Transition:from Door Closing to Idle\n");

}

}

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

\* 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");

}

}

}