# **设计报告**

班级:1604031

学号:16040310078

姓名:黄鑫

1. 程序

#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; bool down;

up = true;

down = false;

if ((GetCallLight || GetPanelFloorLight) == true)

{

int floor;

floor = IdleWhatFloorToGoTo(&up);//Event

if (floor>0 && up)

{

SetMotorPower(1);

\*state = MovingUp;

}

else if (floor > 0 && !up)

{

SetMotorPower(-1);

\*state = MovingDown;

}

}

else if (GetOpenDoorLight() == true)

{

SetDoor(GetFloor(), true);

SetOpenDoorLight(false);

\*state = DoorOpen;

}

if (GetCallLight(GetFloor(), up) || GetCallLight(GetFloor(), down))

{

SetDoor(GetFloor(), true);

SetCallLight(GetFloor(), up, false);

SetCallLight(GetFloor(), down, false);

\*state = DoorOpen;

}

if (GetCloseDoorLight())

{

SetCloseDoorLight(false);

return;

}

}

void StateMovingUp(int \*state)

{

bool up; bool down;

up = true;

down = false;

int floor;

floor = GoingUpToFloor();

if ((GetCallLight || GetPanelFloorLight) == true)

{

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

{

SetMotorPower(0);

SetDoor(floor, true);

\*state = DoorOpen;

if (floor == Lib\_FloorNum)

{

SetCallLight(floor, down, false);

}

if (GetCallLight(floor,up))

{

SetCallLight(floor, up, false);

}

else

{

SetCallLight(floor, down, false);

}

SetPanelFloorLight(GetNearestFloor(), false);//消费门内楼层按钮为当前楼层按钮

}

}

if ((GetOpenDoorLight || GetCloseDoorLight) == true)

{

SetOpenDoorLight(false);

SetCloseDoorLight(false);

}

}

void StateMovingDown(int \*state)

{

bool up; bool down;

up = true;

down = false;

int floor;

floor = GoingDownToFloor();

if ((GetCallLight || GetPanelFloorLight) == true)

{

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

{

SetMotorPower(0);

SetDoor(floor, true);

\*state = DoorOpen;

if (floor == 1)

{

SetCallLight(floor, up, false);

}

else

{

SetCallLight(floor, down, false);

}

SetPanelFloorLight(GetNearestFloor(), false);

}

}

if ((GetOpenDoorLight || GetCloseDoorLight) == true)

{

SetOpenDoorLight(false);

SetCloseDoorLight(false);

}

}

void StateDoorOpen(int \*state)

{

bool up; bool down;

up = true;

down = false;

if (GetCloseDoorLight() == true)

{

SetDoor(GetNearestFloor(), false);

SetCloseDoorLight(false);

\*state = DoorClosing;

}

if (IsDoorOpen(GetNearestFloor()) == true)

{

SetDoor(GetNearestFloor(), false);

\*state = DoorClosing;

}

if (GetOpenDoorLight() == true)

{

SetOpenDoorLight(false);

}

}

void StateDoorClosing(int \*state)

{

bool up; bool down;

up = true;

down = false;

if (GetOpenDoorLight() == true)

{

SetDoor(GetNearestFloor(), true);

SetOpenDoorLight(false);

\*state = DoorOpen;

}

if (GetCloseDoorLight() == true)

{

SetCloseDoorLight(false);

}

if (IsBeamBroken() == true)

{

SetDoor(GetNearestFloor(), true);

\*state = DoorOpen;

}

if (IsDoorClosed(GetNearestFloor()) == true)

{

\*state = Idle;

}

}

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

\* 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");

}

}

}

二、流程图

Movingup

Movingdown

DoorOpen

Doorclose

DoorClose

Door

Idle

if(floor>0&&up)

SetMotorPower(1)

If(floor>0&&!up)

SetMotorPower(-1)

If(fabs(getfloor-floor)<Lib\_floor))

Setdoor()

If(fabs(getfloor-floor)<Lib\_floor))SetMotorPower(0);SetDoor();

If(GetOpenDoorLight()==true)SetDoor(,true)

If(GetCloseDoorLight()==true) SetDoor(,false);

If(GetOpenDoorLight/GetCallLight==true)

DoorOpen

1. 实现功能

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。

8、

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。

9、

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。

10、

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。

11、

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。

12、

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。

13、

电梯在2F以上，10s无动作，自动降到1楼。

14、

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。

15、

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。

16、

空闲状态，门是关闭的，因此按关门按钮失效。

17、

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。

18、

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。