三层电梯状态机实验设计报告

16040310010 杨洋

1. **流程图描述各个状态函数**

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

if (GetOpenDoorLight())

if(GetCallLight(GetNearestFloor(), true))

if(GetCallLight(GetNearestFloor(),false))

if (floor>0&&up)

if (floor>0&&!up)

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

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

if(GetCloseDoorLight())

if(IsDoorOpen(GetNearestFloor)))

if(GetOpenDoorLight())

if(IsBeamBroken()==true)

if(IsDoorClosed(GetNearestFloor()))

1. 状态机代码

好好#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);//确定目标楼层

if (floor>0&&up)//如果要向上运行

{

SetMotorPower(1);

\*state = MovingUp;

}

if (floor>0&&!up)//如果要向下运行

{

SetMotorPower(-1);

\*state = MovingDown;

}

if (GetOpenDoorLight())//门内按了开门灯

{

SetOpenDoorLight(false);//消费开门灯

SetDoor(GetNearestFloor(), true);

\*state = DoorOpen;

}

if (GetCallLight(GetNearestFloor(), true))//如果门外按了向上

{

SetDoor(GetNearestFloor(), true);

SetCallLight(GetNearestFloor(), true, false);

\*state = DoorOpen;

}

if (GetCallLight(GetNearestFloor(), false))//如果门外按了向下

{

SetDoor(GetNearestFloor(), true);

SetCallLight(GetNearestFloor(), false, false);

\*state = DoorOpen;

}

if (GetCloseDoorLight())//在门关闭时消费门内关门键

{

SetCloseDoorLight(false);

return;

}

}

void StateMovingUp(int \*state)

{

int floor;

floor = GoingUpToFloor();

SetOpenDoorLight(false);

SetCloseDoorLight(false);

if (fabs(GetFloor() - floor) < Lib\_FloorTolerance)//检测是否到目标楼层

{

SetMotorPower(0);

SetDoor(GetNearestFloor(), true);

SetCallLight(GetNearestFloor(), true, false);

SetCallLight(GetNearestFloor(), false, false);

SetPanelFloorLight(GetNearestFloor(), false);

if (GetNearestFloor() == Lib\_FloorNum)//到了最高层，消费门外down按钮

SetCallLight(GetNearestFloor(), false, false);

\*state = DoorOpen;

}

}

void StateMovingDown(int \*state)

{

int floor;

floor = GoingDownToFloor();

SetOpenDoorLight(false);

SetCloseDoorLight(false);

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

{

SetMotorPower(0);

SetDoor(GetNearestFloor(), true);

SetCallLight(GetNearestFloor(), false, false);

SetCallLight(GetNearestFloor(), true, false);

SetPanelFloorLight(GetNearestFloor(), false);

\*state = DoorOpen;

}

}

void StateDoorOpen(int \*state)

{

if (GetCloseDoorLight())

{

SetDoor(GetNearestFloor(), false);

SetCloseDoorLight(false);

\*state = DoorClosing;

}

if (IsDoorOpen(GetNearestFloor()))//如果当前楼层门已经打开

{

SetDoor(GetNearestFloor(), false);//自动进行关门

\*state = DoorClosing;

}

if (GetOpenDoorLight())//如果正在开门时还按了开门键

SetOpenDoorLight(false);//消费开门键

}

void StateDoorClosing(int \*state)

{

if (GetOpenDoorLight() )//正在关门时按了开门键

{

SetDoor(GetNearestFloor(), true);

SetOpenDoorLight(false);

\*state = DoorOpen;

}

if (GetCloseDoorLight() )//正在关门时按了关门灯

SetCloseDoorLight(false);

if (IsBeamBroken())//遇到障碍物

{

SetDoor(GetNearestFloor(), true);

\*state = DoorOpen;

}

if (IsDoorClosed(GetNearestFloor()))

\*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");

}

}

}实现的功能及测试结果

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。

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。

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。

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。

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。

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。

电梯上升途中或下降途中，任何反方向按钮呼叫均无效。

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。

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。

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。

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。

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。

电梯在2F以上，10s无动作，自动降到1楼。

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。

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。

空闲状态，门是关闭的，因此按关门按钮失效。

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。

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。