# C Project Report:*CAR GAME*

## Abstract

## Our project achieves through operate the spirit to drive in the designated area and keep away from the heading enemy cars, but once the spirit and the enemy car collide,game will over.what’s more,our game have both 2D and 2.5D version. In the 2D version,spirit will drive in a group of predefined enenmy cars.While in the 2.5Dversion,the enemy cars will increase their speed as the time goes by,and during the way will drop so many coins,which contributes to your scores.

## Introduction/Problem Statement

Our game is a kind of evade racing game, through the keyboard operation vehicles (hereinafter referred to as the elves are vehicles), to avoid the incoming enemy vehicles. In the game, background movement, the emergence of enemy vehicles and the fall of bonus coins are all carried out at the same time. For this reason, we use multithreading technology.

Our first problem is that judging whether the two spirits meet，we abstract them into two rectangular models, that is, to judge the intersection of two rectangles.

Our second problem is that controlling as well as the enemy car coming and coins dropping，we used multithread program.

## **Group Division**

### Member

wrote the code for the production of the enemy cars and the coins,and mouse click control.

He wrote the functions like “void enemycarcontrol1”to realize the enemy car coming successfully，and “void coin” to realize the coins dropping successfully.And by “transplanting” the Bool function in Java,he realized using mouse to control the botton.

### Member

is responsible of the code for spirit control and the condition of crushed.

He wrote the functions like“BOOL isIntersect”to judge if two rectangles are intersected.

### Member

compiled the expression of the pictures and animation,and the play and entering of the music.

She wrote the funcations like “void linechange” to realize the background movement.

### Member

did the control of the menu and link.

He wrote the functions like “BOOL intmousebotton1”and “BOOL intmousebotton2”and so on.

## Analysis

Problem 1:Control the upper, lower, left and right movements of the vehicle through the keyboard key.

Solution 1: Define the buttons on the keyboard and the upper and lower connections.

Problem 2:Play the background music.

Solution 2: Design the appropriate functions.

Problem 3:Background movement.

Solution 3: Design the appropriate functions.

Problem 4: Mouse click control.

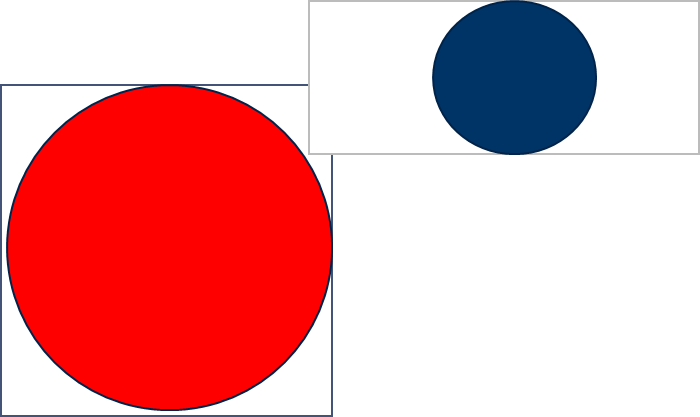
Solution 4:Create the functions to int mouse botton and identify the mouse position.

Problem 5: Menu and linking.

Solution 5: We finally decided to design two programs, one is the main body of the game and the other is a menu. We first open the menu program, and the buttons in the menu can open the game main program.

Problem 6: The end of the game, judging if two cars meet.

Solution 6a: Cheek the distance of two sprites.if the distance <r1+r2,two sprites Intersect.Result:didn't work well.(like the picture 1)



Picture 1

Solution 6b:We abstract the two cars into two rectangular models. Naturally, we first think about the intersection of two rectangles. But we found that two rectangles intersect a lot, which makes coding very complicated. So we began to consider two disjoint rectangles. In geometry, there are four kinds of disjoint situations.(like the picture 2)

Picture 2

Problem7: Control the car as well as the enemy car coming and coins dropping.

Solution7: We have learnt multithread program. We learnt that every program running on the system is a process. Each process contains one or more threads. The process may also be a dynamic execution of the whole program or part of the program. A thread is a set of instructions, or a special segment of a program, which can be executed independently in a program. It can also be interpreted as the context of code operation. So threads are basically lightweight processes, which are responsible for performing multiple tasks in a single program. The operation system is usually responsible for the scheduling and execution of multiple threads. Multithreading is a single sequence control process in a program. It runs multiple threads simultaneously in a single program to do different jobs, called multithreading.

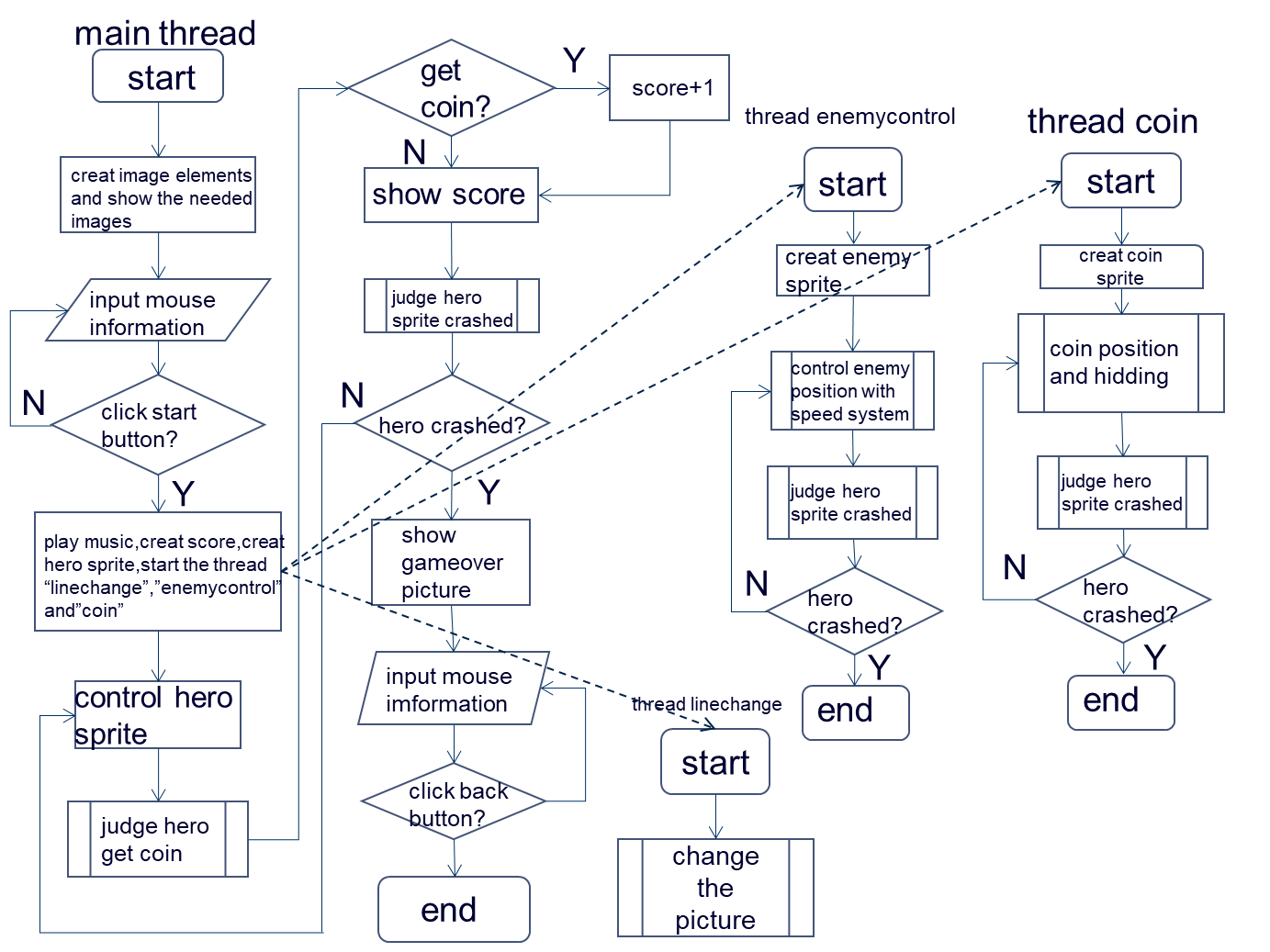
And we also know that there are many risks to use multithread program，just like：

Threads may bring more "bug" to programs, which mean that we have to be careful to use them.

The termination of threads needs to consider its impact on program execution.

The common block model data is shared among multiple threads, which prevents the occurrence of thread deadlock.

## Design

This phase tries to answer ***how*** the problem solution will meet the above enumerated requirements based on the known constraints.Some flowcharts should be used to illustrate your project design.

## Implementation

1. Header file：

①#include <stdlib.h>

The stdlib header file is the standard library standard header file.

②#include <stdio.h>

Stdio is"standard input & output".

③#include <yzkgame.h>

We downloaded a game engine called YZK on the Internet, and of course it's empty. We write the game code in it. This header provides us with YZKGAME­\_API BOOL or YZKGAME\_API void showMouseCursor and other functions.

④#include <time.h>

Time.h is the date and time header file in C/C++. Functions that require time.

2.Function

2.5D version

①void linechange(void\*arglist);

We use this function to make the line on the road change，so it will be like the background is moving.

Details:

void linechange(void\*arglist)

{

int line = 1;

createImage(line, "line1.png"); Use the already designed road line pictures

for (;;)

In this cycle, we keep calling “line” that equal to 1, Because of the different location of the line in the picture, the circular call will form animation effect.

{

pauseGame(100);

setImageSource(1, "line2.png");

pauseGame(100);

setImageSource(1, "line3.png");

pauseGame(100);

setImageSource(1, "line1.png");

}

}

②void enemycarcontrol1(void\*arglist);

We use this function to make enemy vehicles randomly appear.

void enemycarcontrol1(void\*arglist)

{

int enemyx, enemyy, speed, difficulty;

Input the position and speed of an enemy vehicle

createSprite(3, "enemycar");

playSpriteAnimate(3, "drive");

srand((unsigned)time(NULL));

while (1)

In this cycle, as time goes on, the speed of the vehicle will increase and the difficulty will increase.

{ first，we set the original location、speed、difficulty

enemyy = 700;

speed = 1;

difficulty = 0;

for (;;)

{

enemyx = rand() % 460 + 281;

for (;enemyy > -200;)

{

setSpritePosition(3, enemyx, enemyy);

enemyy = enemyy - speed;

pauseGame(3);

}

difficulty = difficulty + 1;//make enemy car faster

if (difficulty > 5)

{

speed++;

difficulty = 0;

}

if (speed > 4)//make sure enemy car can't be too fast

actually，we tried not to use this cycle，but soon，the car will be so fast that we can’t see them at all.

{

speed = 4;

}

enemyy = 700;

if (isSpriteIntersect(2, 3))

{

break;

}

}

while (1)

{

pauseGame(5);

if (isMouseLeftButtonDown())

{

int mousex = getMousePositionX();

int mousey = getMousePositionY();

if (ismouseonbutton2(mousex, mousey, 4))

{

break;

}

}

}

}

}

③void coin(void\*arglist);

We use this function to control the drop of gold coins

void coin(void\*arglist)

{

int coinx, coiny;

input the the position of the gold coin

createSprite(4, "coin");

call the picture of the coin

playSpriteAnimate(4, "rotate");

srand((unsigned)time(NULL));

while (1)

{In this cycle, the gold coin will continue to fall

for (coiny = 700;;)

{

coinx = (6000 + rand()) % 460 + 281;

for (;coiny > -200;)

{

setSpritePosition(4, coinx, coiny);

coiny -= 2;

pauseGame(2);

}

showSprite(4);

if (isSpriteIntersect(2, 3))

Judging whether the two rectangles intersect, that is, whether the game is over or not, thus ending the falling gold coins.

{

break;

}

coiny = 700;

}

while (1)

{

pauseGame(5);

if (isMouseLeftButtonDown())

{

int mousex = getMousePositionX();

int mousey = getMousePositionY();

if (ismouseonbutton2(mousex, mousey, 4))

{

break;

}

}

}

}

}

④BOOL isIntersect(int x1, int y1, int width1, int height1, int x2, int y2, int width2, int height2);

BOOL isIntersect(int x1, int y1, int width1, int height1, int x2, int y2, int width2, int height2)

A function to judge whether two cars meet.

{ The case of four kinds of disjoint rectangles

BOOL b1 = y2 > y1 + height1;

BOOL b2 = y2 + height2 < y1;

BOOL b3 = x2 + width2 < x1;

BOOL b4 = x2 > x1 + width1;

BOOL Intersect = (b1 == FALSE && b2 == FALSE && b3 == FALSE && b4 == FALSE);

return Intersect;

}

BOOL isSpriteIntersect(int sprite1, int sprite2);

BOOL ismouseonbutton1(int x, int y, int buttonx, int buttony, int buttonwidth, int buttonheight);

BOOL ismouseonbutton2(int x, int y, int imgnum);

void gameMain(void)

{

int scorenumber = 0;

set the original score

char score[10] = { 0 };

int enemycarsprite = 3;

int bgImg = 0;

int gameover = 2;

set the key

int button1 = 3;

int button2 = 4;

int button3 = 5;

int carsprite = 2;

setGameTitle("赛车");

set the size of game window

setGameSize(1200, 900);

call the picture of the start、button and so on

createImage(bgImg, "start.png");

createImage(button1, "button1.png");

setImagePosition(button1, 550, 200);

createImage(gameover, "gameover.png");

createImage(button2, "button2.png");

setImagePosition(button2, 450, 200);

createImage(button3, "button3.png");

setImagePosition(button3, 650, 200);

hideImage(2);

hideImage(4);

hideImage(5);

for (;;)

{

if (isMouseLeftButtonDown())

{

int mousex = getMousePositionX();

int mousey = getMousePositionY();

if (ismouseonbutton2(mousex, mousey, button1))

{

break;

}

}

}

hideImage(button1);

play the background music

playSound("网络歌手-天国与地狱.mp3", 1);

createText(1, "0");

setTextFontSize(1, 70);

setTextPosition(1, 1000, 800);

createSprite(carsprite, "car");

setImageSource(bgImg, "road1.png");

asyncRun(linechange, (void\*)999);

setSpritePosition(carsprite, 650, 50);

playSpriteAnimate(carsprite, "drive");

asyncRun(enemycarcontrol1, (void\*)888);

asyncRun(coin, (void\*)887);

while (1)

{

scorenumber = 0;

hideImage(2);

hideImage(4);

hideImage(5);

showImage(0);

showImage(1);

showSprite(2);

showSprite(3);

showSprite(4);

playSpriteAnimate(carsprite, "drive");

while (1)

{

int ch = getPressedKeyCode();

int x = getSpriteX(carsprite);

int l = getSpriteWidth(carsprite);

if (RP\_KEY\_LEFT == ch)

{

if (x > 100)

{

x -= 3;

setSpritePosition(carsprite, x, 50);

}

}

else if (RP\_KEY\_RIGHT == ch)

{

if (x + l < 1100)

{

x += 3;

setSpritePosition(carsprite, x, 50);

}

}

if (isSpriteIntersect(2, 4) && isSpriteVisible(4))

{

hideSprite(4);

scorenumber++;

}

if (isSpriteIntersect(carsprite, enemycarsprite))

{

playSpriteAnimate(carsprite, "bang");

pauseGame(3000);

hideImage(0);

hideImage(1);

hideSprite(2);

hideSprite(3);

hideSprite(4);

break;

}

sprintf(score, "%d", scorenumber);

setText(1, score);

pauseGame(5);

}

showImage(2);

showImage(4);

showImage(5);

while (1)

{

pauseGame(5);

if (isMouseLeftButtonDown())

{

int mousex = getMousePositionX();

int mousey = getMousePositionY();

if (ismouseonbutton2(mousex, mousey, button2))

{

hideImage(4);

hideImage(5);

break;

}

}

}

}

}

This code is cheek if mouse is on the button

BOOL isSpriteIntersect(int sprite1, int sprite2)

{

int x1 = getSpriteX(sprite1);

int y1 = getSpriteY(sprite1);

int x2 = getSpriteX(sprite2);

int y2 = getSpriteY(sprite2);

int width1 = getSpriteWidth(sprite1);

int height1 = getSpriteHeight(sprite1);

int width2 = 200;

int height2 = 200;

return isIntersect(x1, y1, width1, height1, x2, y2, width2, height2);

}

BOOL ismouseonbutton1(int x, int y, int buttonx, int buttony, int buttonwidth, int buttonheight)

{

int upy = buttony + buttonheight;

int downy = buttony;

int leftx = buttonx;

int rightx = buttonx + buttonwidth;

return x >= leftx && x <= rightx && y >= downy && y <= upy;

}

BOOL ismouseonbutton2(int x, int y, int imgnum)

{

return ismouseonbutton1(x, y, getImageX(imgnum), getImageY(imgnum), getImageWidth(imgnum), getImageHeight(imgnum));

}

int main(void)

{

rpInit(gameMain);

return 0;

}

**2D version：**

void gameMain(void)

{

int car = 0;

int enemynum;

int enemyx;

int enemyy=900;

int bgpic = 0;

createImage(0, "background.png");

setGameTitle("2D");

setGameSize(1200, 900);

createSprite(car, "car");

We use this way to create enemy cars

for (enemynum =500;enemynum <= ;enemynum++)

{

createSprite(enemynum, "enemy");

}

setSpritePosition(car, 550, 200);

playSpriteAnimate(car, "drive");

for (enemynum = 500;enemynum <=;enemynum++)

{

playSpriteAnimate(enemynum, "drive");

}

while (1)

{

int ch = getPressedKeyCode();

int x = getSpriteX(car);

int y = getSpriteY(car);

int l = getSpriteWidth(car);

int h = getSpriteHeight(car);

int judge = 0;

if (RP\_KEY\_LEFT == ch)

{

if (x > 100)

{

x -= 5;

setSpritePosition(car, x, y);

}

}

else if (RP\_KEY\_RIGHT == ch)

{

if (x + l < 1100)

x += 5;

setSpritePosition(car, x, y);

}

else if (RP\_KEY\_DOWN == ch)

{

if (y > 0)

{

y -= 5;

setSpritePosition(car, x, y);

}

}

else if (RP\_KEY\_UP == ch)

{

if (y + h < 900)

{

y += 5;

setSpritePosition(car, x, y);

}

}

We use this way to create our map

setSpritePosition(1000,550 , enemyy+0);

setSpritePosition(1001,600 , enemyy+0);

setSpritePosition(1002,650 , enemyy+50);

setSpritePosition(1003,300 , enemyy+30);

setSpritePosition(1004,450 , enemyy+100);

setSpritePosition(1005, 300, enemyy+100);

setSpritePosition(1006,100 , enemyy+700);

setSpritePosition(1007,100 , enemyy+850);

setSpritePosition(1008,150 , enemyy+200);

setSpritePosition(1009,150 , enemyy+350);

......

enemyy-=2;

pauseGame(10);

for (enemynum = 1;enemynum <= 173;enemynum++)

{

if (isSpriteIntersect(car, enemynum))

{

judge = 1;

break;

}

}

if (judge == 1)

{

playSpriteAnimate(car, "bang");

break;

}

}

pauseGame(5000);

}

BOOL isIntersect(int x1, int y1, int width1, int height1, int x2, int y2, int width2, int height2)

{

BOOL b1 = y2 > y1 + height1;

BOOL b2 = y2 + height2 < y1;

BOOL b3 = x2 + width2 < x1;

BOOL b4 = x2 > x1 + width1;

BOOL Intersect = (b1 == FALSE && b2 == FALSE && b3 == FALSE && b4 == FALSE);

return Intersect;

}

BOOL isSpriteIntersect(int sprite1, int sprite2)

{

int x1 = getSpriteX(sprite1);

int y1 = getSpriteY(sprite1);

int x2 = getSpriteX(sprite2);

int y2 = getSpriteY(sprite2);

int width1 = getSpriteWidth(sprite1);

int height1 = getSpriteHeight(sprite1);

int width2 = 50;

int height2 = 50;

return isIntersect(x1, y1, width1, height1, x2, y2, width2, height2);

}

int main(void)

{

rpInit(gameMain);

return 0;

}

And this is the code of map

#include <stdlib.h>

#include <stdio.h>

#include <yzkgame.h>

#pragma comment( linker, "/subsystem:\"console\" /entry:\"mainCRTStartup\"" )

#pragma comment(lib, "YZKGame.lib")

BOOL ismouseonbutton1(int x, int y, int buttonx, int buttony, int buttonwidth, int buttonheight);

BOOL ismouseonbutton2(int x, int y, int imgnum);

void gameMain(void)

{

int bgpic = 0;

int button1 = 1;

int button2 = 2;

setGameTitle("car race");

setGameSize(1200, 900);

createImage(bgpic, "menu.png");

createImage(button1, "button1.png");

createImage(button2, "button2.png");

setImagePosition(button1, 450, 300);

setImagePosition(button2, 450, 450);

for(;;)

{

if (isMouseLeftButtonDown())

{

int mousex = getMousePositionX();

int mousey = getMousePositionY();

if (ismouseonbutton2(mousex, mousey, button1))

{

system("C:\\cgame\\2dvision\\level1\\level1.exe");

}

if (ismouseonbutton2(mousex, mousey, button2))

{

system("C:\\cgame\\2.5dvision\\cargame.exe");

}

}

pauseGame(10);

}

}

BOOL ismouseonbutton1(int x, int y, int buttonx, int buttony, int buttonwidth, int buttonheight)

{

int upy = buttony + buttonheight;

int downy = buttony;

int leftx = buttonx;

int rightx = buttonx + buttonwidth;

return x >= leftx && x <= rightx && y >= downy && y <= upy;

}

BOOL ismouseonbutton2(int x, int y, int imgnum)

{

return ismouseonbutton1(x, y, getImageX(imgnum), getImageY(imgnum), getImageWidth(imgnum), getImageHeight(imgnum));

}

int main(void)

{

rpInit(gameMain);

return 0;

}

## Testing and Debugging

2.5D vision

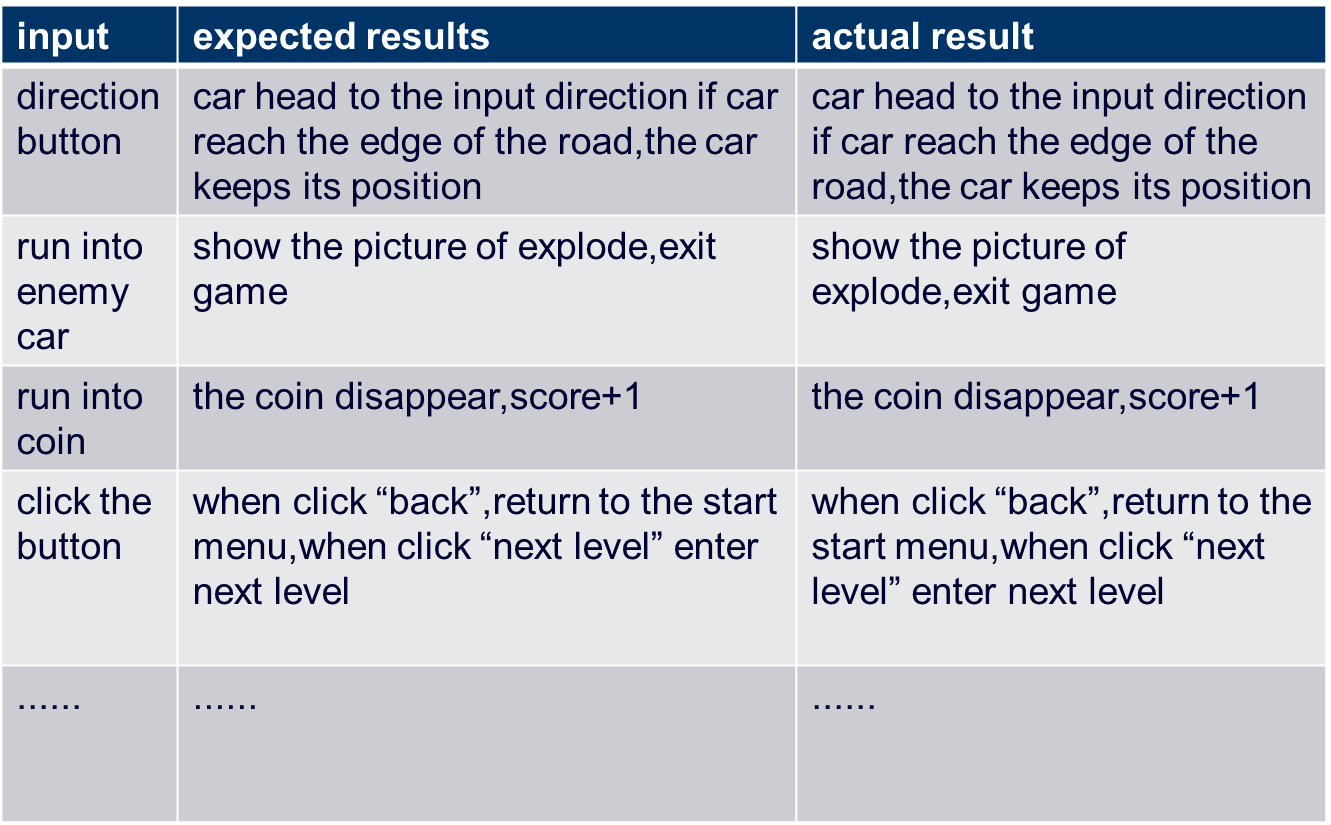
|  |  |  |
| --- | --- | --- |
| **input** | **expected results** | **actual result** |
| ← | Car move left | Car move left |
| → | Car move right | Car move right |
| ↑ | Car don’t move | Car don’t move |
| ↓ | Car don’t move | Car don’t move |
| A | Car don’t move | Car don’t move |
| D | Car don’t move | Car don’t move |
| W | Car don’t move | Car don’t move |
| S | Car don’t move | Car don’t move |
| Reach the edge of the road ← | Car keeps its position | Car keeps its position |
| Reach the edge of the road  → | Car keeps its position | Car keeps its position |
| run into enemy car | show the picture of explode,game over | show the picture of explode,game over |
| run into coin | the coin disappear,score+1 | the coin disappear,score+1 |
| click button | start game | start game |
| Click other part | No reaction | No reaction |

2D vision

|  |  |  |
| --- | --- | --- |
| ← | Car move left | Car move left |
| → | Car move right | Car move right |
| ↑ | Car move up | Car move up |
| ↓ | Car move down | Car move down |
| A | Car don’t move | Car don’t move |
| D | Car don’t move | Car don’t move |
| W | Car don’t move | Car don’t move |
| S | Car don’t move | Car don’t move |
| Reach the edge of the road ← | Car keeps its position | Car keeps its position |
| Reach the edge of the road → | Car keeps its position | Car keeps its position |
| Reach the edge of the road ↑ | Car keeps its position | Car keeps its position |
| Reach the edge of the road ↓ | Car keeps its position | Car keeps its position |
| crash enemy | show the picture of explode,game over | show the picture of explode,game over |
| Reach goal | show the picture congratulation and next level button | show the picture congratulation and next level button |
| click button start | start game | start game |
| click button next level | Open next level | Open next level |
| Click other part | No reaction | No reaction |

Menu

|  |  |  |
| --- | --- | --- |
| click button 2.5D vision | Open 2.5D vision game | Open 2.5D vision game |
| click button 2D vision | Open 2D vision game | Open 2D vision game |
| Click other part | No reaction | No reaction |



## Result&Conclusion

We have the following functions: we can freely control the movement of the vehicle, all the music and pictures are normally called, the enemy vehicles can appear randomly, eating gold coins can increase the score.

For multithreading, we intended to transmit data at first, but failed. In the end, professor AT also came to this point，in our multithreading, if we add numerical transmission, it will cause errors.

For the shortage, we also designed the score list. But due to time, we failed to complete this part of the code. In this regard, our team members have some introspection, and our time arrangement is inadequate.