项目说明文档

数据结构课程设计

——飞跃地平线游戏

小 组 成 员： 2152056王宇轩

小 组 成 员： 2154168王鹏

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

1 项目背景

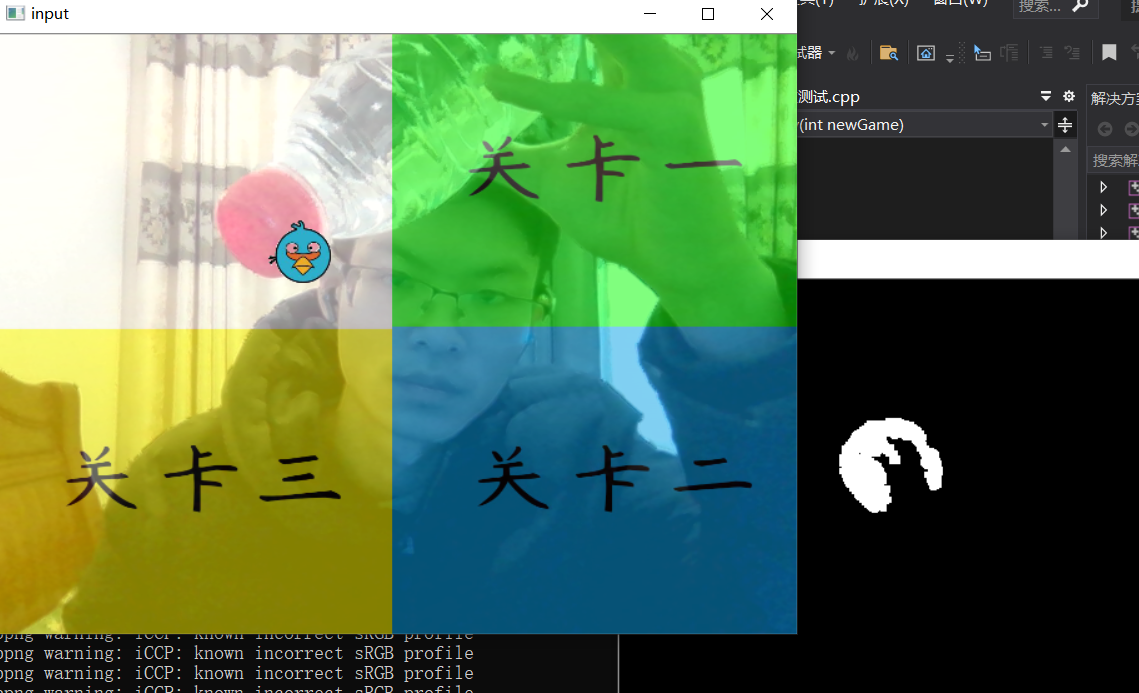
1.1 项目简介

本项目是利用OpenCV库，基于c++开发的一款小游戏。在程序启动后，会调用摄像头，并识别摄像头中出现的红色物体，并将识别到的红色物品作为控制游戏的指针。随后，用户可以通过移动红色物品来选择游戏难度，并通过物品操控游戏中的角色，完成游戏并获取更高的分数。

1.2 项目示例

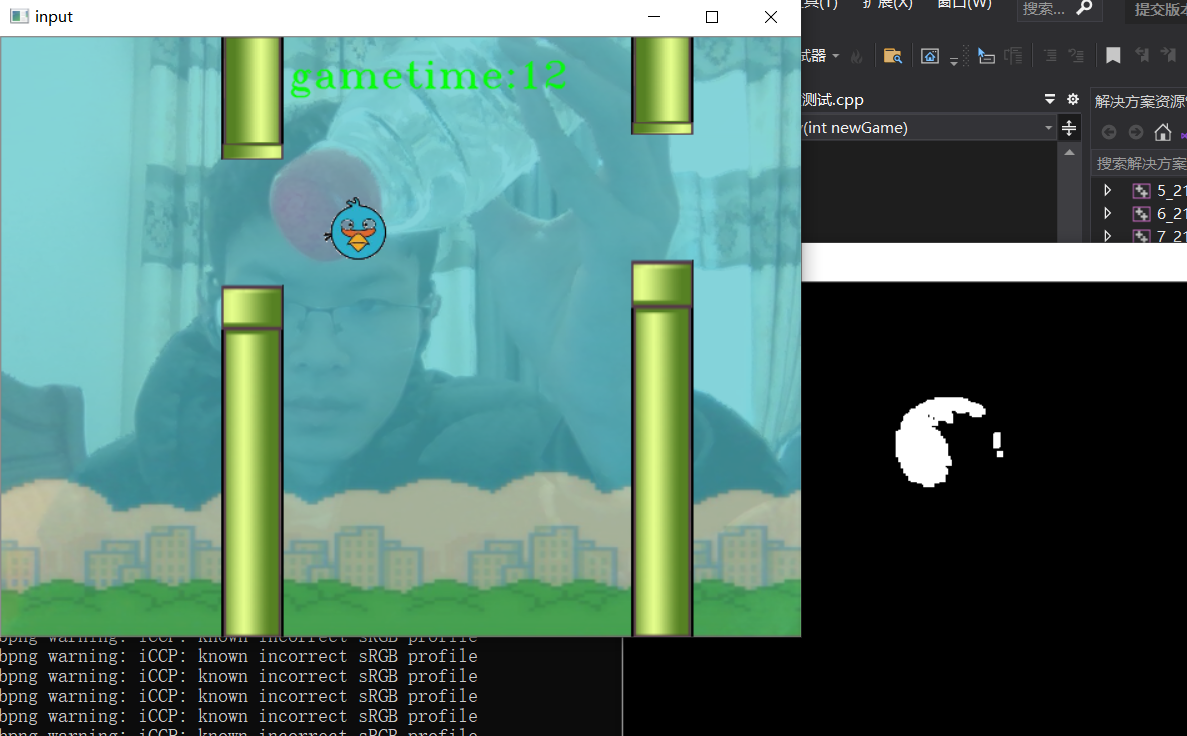
1.2.1 识别红色物品

识别摄像头画面中的红色物体，取其中色块矩形面积最大者作为操控指针，在input界面中，角色的位置保持在红色色块的位置，在output界面中显示所识别到的红色色块的位置。



1.2.2 操控红色物品进行游戏

移动红色物品，程序识别到的红色色块随之移动。游戏界面中的角色维持在红色色块的位置。从游戏界面左侧不断出现有缺口的柱状障碍物，玩家通过移动红色物品以移动角色绕过障碍物，若碰到障碍物则游戏结束。游戏界面上方显示游戏时间，游戏结束后显示本局游戏得分与历史最高得分。



2 设计

2.1 数据结构设计

项目中涉及红色物块识别确立坐标，对角色与障碍物进行碰撞判定的功能，因此使用二维数组保存实体轮廓与信息。

2.2 类结构设计

为高效、统一、便携地调用项目所需的资源，使用一个资源类保存项目中所要使用到的各种图片文件的位置与命名信息。为完成游戏的核心功能，使用一个角色类来表示玩家所操控的角色，使用一个障碍物类来表示游戏中所产生的各个障碍物，使用一个碰撞判断类对二者之间的碰撞情况进行判断，以此完成游戏的核心玩法。使用一个分数类以计算玩家在一局游戏中获得的分数。使用一个ui类以显示游戏过程中所要显示的透明方块图片，角色、障碍物与背景等的图片。

2.3 成员与操作设计

**分数类（Grade）**

**私有成员：**

int grade;//分数

**公有操作：**

int getgrade()

//获取分数

int count(int newGame)

//计算每一局游戏的得分

void showcount(Mat frame,int newGame)

//显示得分

**障碍物类（Hinder）**

**私有成员：**

int weight;//宽度

int high;//高度

String src;//障碍物图片文件名

int speed;//移动速度

int direction;//方向 1,2,3,4代表上下左右

int location\_x;//水平位置

int location\_y;//垂直位置

int T; //计时器

int flag;//结束标志

enum udflagenum { downFlag,upFlag };//类内使用的标记枚举

**公有操作：**

int getT()//获取变量

int getflag()//获取变量

int getWeight()//获取变量

int getHigh()//获取变量

String getSrc()//获取变量

int getSpeed()//获取变量

int getDirection()//获取变量

int getLocation\_y()//获取变量

int getLocation\_x()//获取变量

void setT(int n)//修改变量

void setFlag(int n)//修改变量

void setWeight(int w)//修改变量

void setHigh(int h)//修改变量

void setSrc(String s)//修改变量

void setLocation\_x(int n)//修改变量

void setLocation\_y(int n)//修改变量

void setSpeed(int n)//修改变量

Hinder(int w, int h, String s, int v, int d, int x, int y)

//含参构造函数

void updownshow(Mat frame, Point2f center, bool udFlag,Hinder\* up = NULL)

//显示单个障碍物函数

void hindershow(Mat frame, Point2f center, User u1, Hinder\* h2, Hinder\* h3, Hinder\* h4)

//显示障碍物整体函数

**碰撞判定类（isHit）**

**公有操作：**

int hit(Point2f center, Hinder\* hin1, User u1)

//判断角色与指定障碍物块是否碰撞

int hited(Mat frame, Point2f center, User u1, Hinder\* h1, Hinder\* h2, Hinder\* h3, Hinder\* h4,int newGame)

//判断角色是否出现碰撞障碍物

**图标类（Ui）**

**私有成员：**

int flag//关卡选择标记

**公有操作：**

void picture(Mat frame, Point2f center);

//显示一张图片 分四个区域

int choice(int location\_x, int location\_y,int newGame);

//关卡选择的判定

void countdowninst(Mat& frame, Mat& pic, Source& s2,int tag);

//显示游戏开始倒计时图标

void countdown(Mat frame,int newGame);

//执行游戏开始的倒计时

void gameui(Mat frame);

//显示游戏ui

void gamecount(Mat frame,int newGame);

//执行游戏中的计时并显示

**角色类（User）**

**私有成员：**

int weight;//宽度

int high;//高度

String src;//角色图片文件

**公有操作：**

int getWeight();//获取参数

int getHigh();//获取参数

String getSrc();//获取参数

void setWeight(int w);//设置参数

void setHIgh(int h); //设置参数

void setSrc(String s); //设置参数

User(int w, int h, String s);

//含参构造函数

void dead(Mat frame, Point2f center);

//角色死亡操作

void show(Mat frame, Point2f center);

//角色显示操作

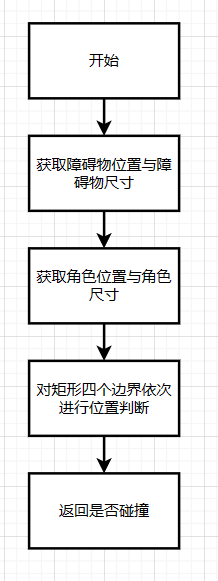
2.4 系统设计

程序开始后，在程序运行期间内全程维持摄像头调用与红色物块识别。程序一开始，窗口界面内给出关卡选择提示，用户根据提示信息选择想要的关卡。完成关卡选择后，界面跳转到游戏游玩界面，执行游戏。游戏结束后，界面显示本局分数与最高分数，随后跳转回关卡选择界面。

3 实现

3.1 碰撞检测功能的实现

3.1.1 碰撞检测功能流程图



3.1.2 碰撞检测功能核心代码

int x = hin1->getLocation\_x();

int y = hin1->getLocation\_y();

int w = hin1->getWeight();

int h = hin1->getHigh();//获取障碍物的边界构成一个矩形框

Rect rect(x, y, w, h);

if (center.x > rect.x + rect.width) { return 0; }

if (center.y > rect.y + rect.height) { return 0; }

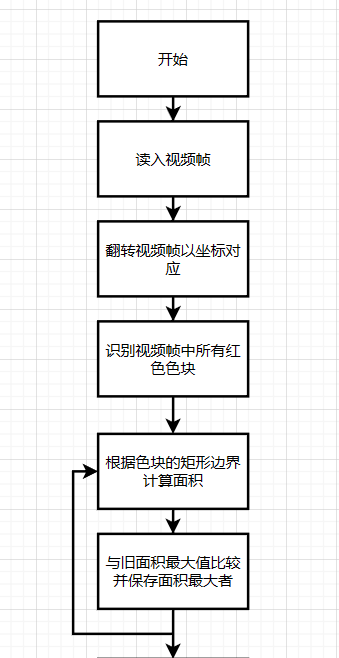
if (center.x + u1.getWeight() < rect.x) { return 0; }

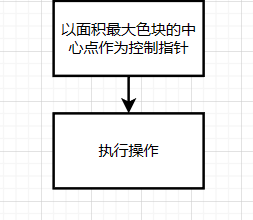
if (center.y + u1.getHigh() < rect.y) { return 0; }

else return 1;

3.2 识别红色物品功能的实现

3.2.1 识别红色物品功能流程图





3.2.2 识别红色物品功能核心代码

while (capture.read(frame))//循环读入视频帧

{

flip(frame, frame, 1);//视频镜像翻转

//blur(frame, dst, Size(5,5));

inRange(frame, Scalar(0, 0, 213), Scalar(255, 128, 255), dst);

//彩色图像分割（输入图像，下限，上限，输出图像）

//开操作

morphologyEx(dst, dst, MORPH\_OPEN, kernel);

//获取边界

findContours(dst, contours, hireachy, RETR\_EXTERNAL, CHAIN\_APPROX\_SIMPLE, Point(0, 0));

//框选面积最大的边界

if (contours.size() > 0)

{

double maxArea = 0;

for (int i = 0; i < contours.size(); i++)

{

double area = contourArea(contours[static\_cast<int>(i)]);

if (area > maxArea)

{

maxArea = area;

rect = boundingRect(contours[static\_cast<int>(i)]);

minEnclosingCircle(contours[static\_cast<int>(i)], center, radius);

}

}

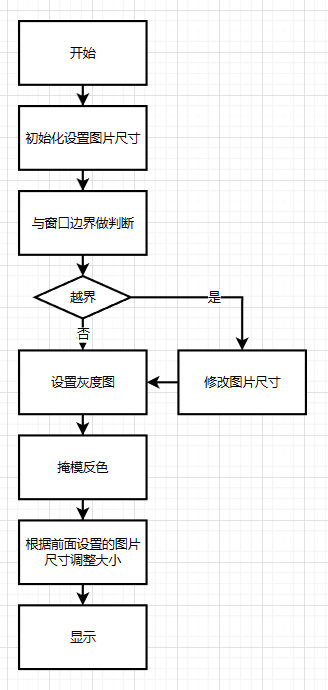
}

doThings();

}

3.3 角色显示功能的实现

3.3.1 角色显示功能流程图



3.3.2 角色显示功能核心代码

int high\_t = getHigh();

int weight\_t = getWeight();

if (center.x + this->getWeight() >= frame.size().width)

weight\_t = frame.size().width - center.x;

if (center.y + this->getHigh() >= frame.size().height)

high\_t = frame.size().height - center.y;

//灰度图

Mat grayPng = imread(src, 0);

threshold(grayPng, grayPng, 180, 255, CV\_THRESH\_BINARY);

//掩模反色

Mat mask = 255 - grayPng;

resize(mask, mask, Size(weight, high));

mask = mask(Range(0, high\_t), Range(0, weight\_t));

Mat png = imread(src);

//调整为固定大小

resize(png, png, Size(weight, high));

//如果超出边界，则裁剪掉超出部分

png = png(Range(0, high\_t), Range(0, weight\_t));

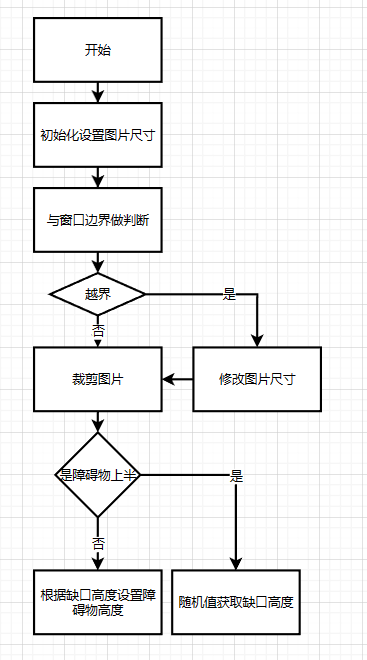
Mat imROI;

imROI = frame(Rect(center.x, center.y, png.cols, png.rows));

png.copyTo(imROI, mask);

3.4 障碍物显示功能的实现

3.4.1 障碍物显示功能流程图



3.4.2 障碍物显示功能核心代码

int high\_t = this->getHigh();

int weight\_t = this->getWeight();

if (this->location\_x + this->getWeight() >= frame.size().width)

weight\_t = frame.size().width - location\_x;

if (this->location\_y + this->getHigh() >= frame.size().height)

high\_t = frame.size().height - location\_y;

if (high\_t == 0 || weight\_t == 0)

return;

//User u1 = User(50, 50, "C:/Users/lenovo/Desktop/23.png");

Rect roi(this->location\_x, this->location\_y, weight\_t, high\_t);

Mat frame\_roi = frame(roi);

Mat pic = imread(this->src);

cv::resize(pic, pic, Size(this->weight, this->high));

pic = pic(Range(0, high\_t), Range(0, weight\_t));

//pic2 = pic2(Range::all(), Range(0, 30)); 当达到边界的时候需要裁剪

cv::addWeighted(frame\_roi, 0, pic, 1, 1, frame\_roi);

this->setLocation\_x(location\_x + speed);

if (this->getLocation\_x() >= frame.size().width)

{

this->setLocation\_x(0);

if (udFlag) {

int ran = rand() % (frame.size().height - 100);//预留60个像素给小鸟通过

this->setHigh(ran);

}

else {

this->setHigh(frame.size().height - up->getHigh() - 100);

this->setLocation\_y(frame.size().height - this->getHigh());

}

}

4 测试

4.1 功能测试

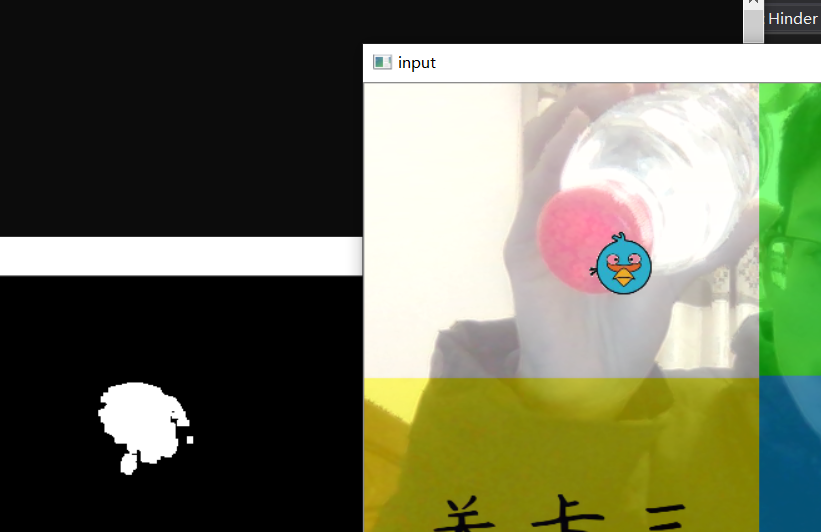
4.1.1 识别红色色块功能测试

**测试用例**：红色瓶盖

**预期结果**：

input窗口中角色位置保持在红色色块位置，output窗口中显示识别到的红色色块

**实验结果**



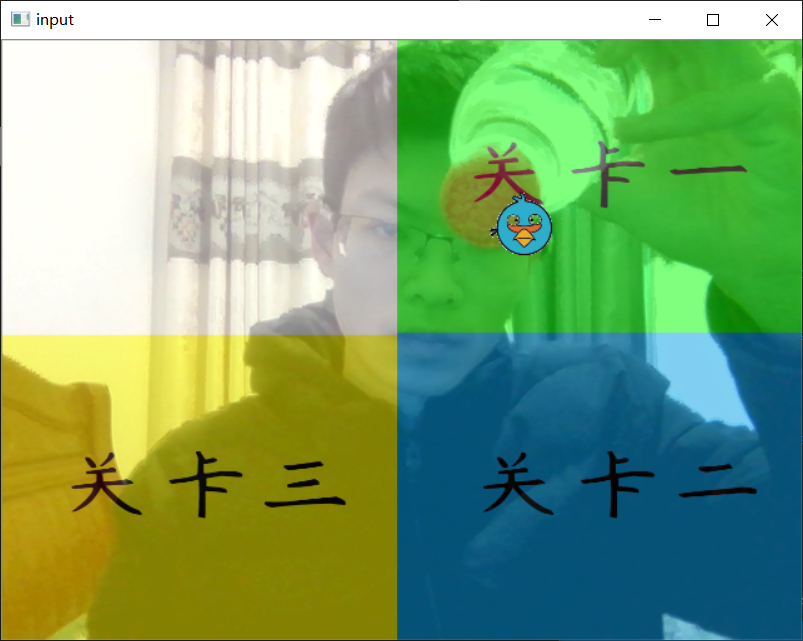
4.1.2 关卡选择功能测试

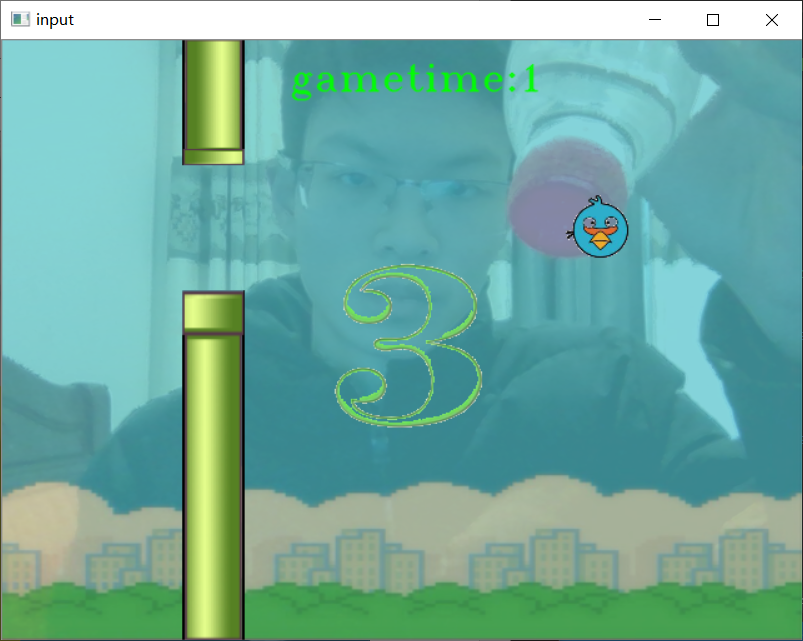
**测试用例：**移动色块选择关卡1

**预期结果：**

进入关卡1

**实验结果：**







4.1.3 游戏功能测试

**测试用例：**移动色块进行游戏

**预期结果：**

程序正常运行

**实验结果：**

