上海交通大學

SHANGHAI JIAO TONG UNIVERSITY

《程序设计》项目报告

项目名称：Clumsy Bird

团队名称：404NOTFOUND

团队成员：高婕　张珂欣　刘欣语

指导教师：鲍杨

9th/06/2020

目　录

1 游戏介绍

我们组项目的灵感来源于游戏《Flappy Bird》。《Flappy Bird》于几年前爆红，规则简单却难度极高，该项目旨在结合《Flappy Bird》的玩法以及手势识别或眨眼识别技术，即玩家通过伸出手指或眨眼取代原游戏中点击屏幕使小鸟上升的操作。为了增加游戏的可玩性，我们做出以下两点改动：一是加入炸弹元素，小鸟在穿越管道的同时必须避开炸弹；二是随着时间的推移，管道移动的速度逐渐变快。

2 项目概述

2.1 问题分析

1.三个界面的搭建与跳转

这个简单小游戏拥有三个界面：初始界面、主游戏界面以及结束界面。为此，我们分别定义了四个主要函数，在函数中实现游戏的各项功能，并在 main 函数中集中实现它们。

三个界面的跳转需要一定的条件。从初始界面到主游戏界面的跳转需要玩家键入空格或向上键，再跳转至结束界面则需要小鸟触碰界面边缘、管道或炸弹。当然，玩家可以随时鼠标点击退出游戏。至于其内部逻辑，只需使前两个函数分别返回一个参数，并用于下一个函数，即可完成跳转。

2.界面元素的载入与交互

该游戏的元素除了最基础的背景以及提供信息的文字外，还需载入小鸟、管道以及炸弹。其中，管道和炸弹都需要实现随机高度（随机出现）以及随时间加速的功能，其逻辑相似，因此我们分别定义了 pipeline 和 bomb 两个类来实现。而小鸟需要实现扇动翅膀的动作、随眨眼动作改变纵坐标的功能。

小鸟与管道、炸弹的交互是用来检测碰撞以获取游戏结束的条件，此处两个碰撞检验的方法并不完全相同，我们将在后面作进一步解释。

3.眨眼功能的实现

我们将通过计算眼睛纵横比（eye aspect ratio (EAR)）的数值，判断眼睛是张开还是闭合，从而检测眨眼动作。使用 dlib 库采集玩家左右眼上的 12 个特征点，当 EAR 小于阈值时触发。

4.手势识别功能的实现

我们通过基于 hsv 颜色空间的肤色识别大致识别手势范围，并进一步通过高斯滤波消除未识别的局部点和过度识别的亮斑。找出手势轮廓，并通过角度计算识别出实时的手势。

2.2 使用工具

2.2.1 Pygame

Pygame 是一套用来写游戏的 Python 模块。基于 SDL 库，可以用 Python 语言创建完全界面化的游戏和多媒体程序。我们通过其官网 https://www.pygame.org/docs/的官方文档进行学习。通常我们使用 pygame 中内置的几个函数来装饰游戏界面，为玩家提供更好的游戏体验。例如，我们使用’ init ’来初始化游戏引擎，’ display ’来创建游戏界面，’ font ’来设置字体，等等。

2.2.2 Opencv

OpenCV 是一个基于 BSD 许可(开源)发行的跨平台计算机视觉库,可以运行在 Linux、Windows、Android 和 Mac OS 操作系统上。它轻量级而且高效——由一系列 C 函数和少量 C++ 类构成,同时提供了 Python、Ruby、MATLAB 等语言的接口,实现了图像处理和计算机视觉方面的很多通用算法。我们在官方文档 http://www.woshicver.com/学习所需的功能,如对视频流的处理、摄像头画面的读取、实时画面上的作图等等。

2.2.3 Dlib

Dlib 有 c++、Python 的接口。使用 Dlib 可以大大简化开发,比如人脸识别,特征点检测之类的工作。同时也有很多基于 Dlib 开发的应用和开源库,比如 face_recogintion 库等等。我们采用了 shape-predictor shape_predictor_68_face_landmarks.dat 检测器进行检测。该检测器能识别出人脸面部的 68 个特征点,我们提取双眼上的 6 个特征点用于眨眼动作的识别。

2.2.4 Numpy

NumPy 是 Python 中科学计算的基础包。我们通过 https://www.numpy.org.cn/user/官方文档进行学习,并用于皮肤识别。

3 代码解释

3.1 导入类库

为了完成游戏界面的搭建、游戏的操控等功能,导入了 pygame、imutils、dlib、cv2、numpy 等等。

```
import cv2
import dlib
import math
import numpy as np
import os
import pygame
import random
import sys
import time
from imutils import face_utils
from itertools import cycle
from pygame.locals import *
from scipy.spatial import distance
```

3.2 基础框架的搭建

3.2.1 基础设置

首先,设置界面大小以及画面刷新帧数,并导入图片和音频。

然后定义 main 函数,初始化游戏并创建游戏窗口,再创建一个跟踪时间的对象以便实现界面的刷新。

```
pygame.init()
screen = pygame.display.set_mode((width,height))
pygame.display.set_caption('Clumsy Bird')
```

```
fpsclock = pygame.time.Clock()
```
（张珂欣）

最后，播放背景音乐并依次调用三个界面的函数，实现游戏的运行。

```
while True:
    SOUNDS['bgm'].play()
    scoreInfo = welcomescreen()
    if scoreInfo == 1:
        overinfo = maingame1(0)
    elif scoreInfo == 2:
        overinfo = maingame2(0)

    gameoverscreen(overinfo)
```
（张珂欣）

3.2.2 定义初始界面

首先，载入背景图片以及文字信息，设置它们的坐标并显示在画面上。

```
background = pygame.image.load('assets/sprites/background-day.png').convert_alpha()
message = pygame.image.load('assets/sprites/message.png').convert_alpha()
base = pygame.image.load('assets/sprites/base.png').convert_alpha()

#创建字体对象
font1 = pygame.font.Font(None,20)
text1 = font1.render("Typing UP to control by hand.",1,(0,0,0))
font2 = pygame.font.Font(None,20)
text2 = font2.render("Typing DOWN to control by eyes.",1,(0,0,0))

#设置坐标
messagex = int((width - message.get_width())/2)
messagey = int(height * 0.2)
playerx = int(width * 0.1)
playery = int(height * 0.1)
```

```
screen.blit(background,(0,0))
screen.blit(message,(messagex,messagey))
screen.blit(base,(0,400))
screen.blit(text1,(0,0))
screen.blit(text2,(0,20))
```
（张珂欣）

小鸟扇动翅膀的功能将在后面进一步说明。

玩家可以鼠标单击关闭窗口来退出游戏，如键入向上键则用手势控制小鸟，向下键用眨眼控制。

```
for event in pygame.event.get():
    if event.type == pygame.QUIT:
        sys.exit()
    #空格键or向上键开始游戏
    elif event.type == pygame.KEYDOWN and (event.key == K_UP):
        SOUNDS['wing'].play()
        return 1
    elif event.type == pygame.KEYDOWN and (event.key == K_DOWN):
        SOUNDS['wing'].play()
        return 2
```
（张珂欣）

3.2.3 定义主游戏界面

此界面调用了较多类与函数，其具体功能将在后面作进一步说明，在此只叙述几个共有的基本功能。

将管道和炸弹实例化；

设置小鸟飞出界面，游戏结束；

```
if birdy <= 0 or birdy >= 400-player.get_height():
    return scoreinfo
```
（张珂欣）

若检测到小鸟撞到管道或炸弹，则游戏结束。

需要注意的是，此处碰撞检测的方法不完全相同，管道的碰撞检测原理是，当管道移动到小鸟所在横坐标时（小鸟横坐标不变），通过比较纵坐标判断是否发生碰撞；

```
if width*0.2-pipeline.pipeup.get_width() <= pipeline.pipex <= width*0.2+player.get_width():
    if birdy <= pipeline.pipey+pipeline.pipeup.get_height() or birdy >= pipeline.pipey+pipeline.pipeup.get_height()+130-player.get_height():
        SOUNDS['hit'].play()
        return scoreinfo
```
（张珂欣）

而炸弹碰撞检验的原理是检测小鸟图片的矩形与炸弹图片的矩形是否发生碰撞。这里存在一定误差，但我们默认允许。利用 pygame.Rect()定义小鸟与炸弹的矩形 birdRect 与 bombRect，函数包含四个参数，前两个参数代表图片左上角在界面中的坐标，后两个参数代表矩形的宽度与高度，其中宽度与高度用.get_width()与 get_height()获取。利用.colliderect()函数检测小鸟矩形与炸弹矩形是否碰撞，若碰撞，则游戏结束。

```
birdRect = pygame.Rect(width * 0.2,200,34,24)
birdRect[1] = birdy
bombRect = pygame.Rect(bomb.bombx,bomb.bomby,bomb.bombpic.get_width(),bomb.bombpic.get_height())
if bombRect.colliderect(birdRect) :
    SOUNDS['hit'].play()
    return scoreinfo
```
（刘欣语）

3.2.4 定义结束界面

同样地，载入背景以及文字信息并设置它们的坐标。

利用主游戏界面返回的 score 变量展示玩家最终得分（showscore 函数已提前定义）。

```
showscore(score)
```
（张珂欣）

3.3 控制识别

3.3.1 眨眼识别

我们将通过计算眼睛纵横比（eye aspect ratio (EAR)）的数值，判断眼睛是张开还是闭合，从而检测眨眼动作。

```
def eye_aspect_ratio(eye):
    # print(eye)
    A = distance.euclidean(eye[1], eye[5])
    B = distance.euclidean(eye[2], eye[4])
    C = distance.euclidean(eye[0], eye[3])
    ear = (A + B) / (2.0 * C)
    return ear
```

（高婕）

使用 dlib 库采集玩家左右眼上的 12 个特征点，当 EAR 小于阈值一定时长时触发。

```
for rect in rects:  #遍历人脸
    shape = predictor(gray, rect) #检测特征点
    points = face_utils.shape_to_np(shape)
    leye = points[l_eye_start:l_eye_end + 1] #取出特征点
    reye = points[r_eye_start:r_eye_end + 1]
    lear = eye_aspect_ratio(leye)    #计算ear
    rear = eye_aspect_ratio(reye)
    ear = (lear + rear)/2

    if ear < eye_thresh:
        frame_counter += 1
    else:
        if frame_counter >= eye_consec_frames:
            blink += 1
            birdy -= 20
        frame_counter = 0
```

（高婕）

此外，为了方便观察眨眼识别的精确度，我们在视频画面上标注了检测到的眨眼次数。

```
#打印到图片上 参数分别为 图片，文字，左上角坐标，字体，字体大小，颜色，粗细
cv2.putText(img, 'Blinks:{}'.format(blink), (10, 30), cv2.FONT_HERSHEY_SIMPLEX , 0.5, (0,255,0), 1)
```

（高婕）

3.3.2 手势识别

通过基于 hsv 颜色空间，进行肤色识别。然后利用高斯滤波，消除局部的亮斑和暗点。

```
# 肤色检测，基于hsv颜色空间
    hsv = cv2.cvtColor(roi,cv2.COLOR_BGR2HSV) #转换到hsv空间
    lower_skin = np.array([0,28,70],dtype=np.uint8) #最低阈值
    upper_skin = np.array([20, 255, 255],dtype=np.uint8) #最高阈值
# 进行高斯滤波
    mask = cv2.inRange(hsv,lower_skin,upper_skin) #腐蚀操作，去除毛刺亮点
    mask = cv2.dilate(mask,kernel,iterations=4) #膨胀操作，填补漏洞
    mask = cv2.GaussianBlur(mask,(5,5),100)
```

（高婕）

利用 cv2 画出轮廓，数出凹凸点。

```
hull = cv2.convexHull(approx,returnPoints=False)
defects = cv2.convexityDefects(approx,hull)
if defects is not None:  #防止出错
    l=0 #凹凸点个数初始值定为0
    for i in range(defects.shape[0]):
        s,e,f,d, = defects[i,0]
        start = tuple(approx[s][0])
        end = tuple(approx[e][0])
        far = tuple(approx[f][0])
        pt = (100,100)

        a = math.sqrt((end[0]-start[0])**2+(end[1]-start[1])**2)
        b = math.sqrt((far[0] - start[0]) ** 2 + (far[1] - start[1]) ** 2)
        c = math.sqrt((end[0]-far[0])**2+(end[1]-far[1])**2)
        s = (a+b+c)/2
        ar = math.sqrt(s*(s-a)*(s-b)*(s-c))
```
（高婕）

3.4 游戏主要功能

3.4.1 小鸟翅膀扇动、随机颜色的实现

小鸟扇动翅膀动作的实现基于 itertools 库中的 cycle 模块。

首先我们创建三个变量：

```
INDEX = cycle('0121')
index = 0
iterloop = 0
```
（张珂欣）

然后，随着画面的刷新（我们设置一秒钟刷新 60 次）不断调整变量的值：

```
if (iterloop + 1) % 5 == 0:
    index = int(next(INDEX))
iterloop = (iterloop + 1) % 30
```
（张珂欣）

在 1 秒钟之内，iterloop 由 0 变化到 29，共变化两轮，则有 12 次满足 if 条件，故小鸟的编号将变化 12 次。小鸟编号的变化按照“0121、0121、0121……”不断循环，0 代表翅膀朝上的小鸟，2 代表翅膀朝下的小鸟，而 1 则代表翅膀水平的小鸟，即中间过渡状态，故该循环可以很好地实现小鸟扇动翅膀的连续动作。

随机的小鸟颜色通过 random 库中的 randint 模块实现：

```
randindex = random.randint(0,2)
```

```
player = pygame.image.load(players[randindex][index]).convert_alpha()
screen.blit(player,(width * 0.2, birdy))
```
（张珂欣）

3.4.2 音效

在导入所需.wav 和.ogg 格式音频的条件下，利用 sound.play()函数播放音乐。

```
#插入声音
if 'win' in sys.platform:
    soundExt = '.wav'
else:
    soundExt = '.ogg'

SOUNDS['die']    = pygame.mixer.Sound('assets/audio/die' + soundExt)
SOUNDS['hit']    = pygame.mixer.Sound('assets/audio/hit' + soundExt)
SOUNDS['point']  = pygame.mixer.Sound('assets/audio/point' + soundExt)
SOUNDS['wing']   = pygame.mixer.Sound('assets/audio/wing' + soundExt)
SOUNDS['bgm']    = pygame.mixer.Sound('assets/audio/bgm' + soundExt)
```

（刘欣语）

在启动初始界面的同时播放 bgm，在初始界面和游戏界面转换/小鸟上升、小鸟撞到管道/炸弹、小鸟得分时分别播放“swing”、“hit”和“point”的音频。

3.4.3 管道、炸弹的出现和移动

```
class pipeline(object):
    def __init__(self):
        self.pipex = 288
        self.pipey = random.uniform(-220,-70)
        self.pipeup = pygame.image.load("assets/sprites/pipeup.png").convert_alpha()
        self.pipedown = pygame.image.load("assets/sprites/pipedown.png").convert_alpha()

    #定义管道移动函数
    def updatepipeline(self):
        self.pipex -= 3
        if 15 < t2-t1 < 20:
            self.pipex -= 3.000000000000000000001
        if 20 < t2-t1 < 25:
            self.pipex -= 3.01
        if 25 < t2-t1 < 30:
            self.pipex -= 3.02
        if t2-t1 >30:
            self.pipex -= 6
        if self.pipex < -50:
            self.pipex = 288
            self.pipey = random.uniform(-220,-70)
```

（刘欣语）

定义一个管道类，设定管道位置坐标的初始值并载入上、下管道的图片。在这里，管道横坐标的初始值为 288，即界面的最右端；纵坐标的初始值考虑的是上管道的情况，由于管道图片本身长度为 320，因此设定上管道纵坐标的初始值为-220 到-70 间的随机数。

在管道类中定义管道移动函数，实现管道移动的基本方法是让管道的横坐标以一个恒定值每次递减，并在其右侧边界的横坐标小于 0 时重新设定其横坐标，使其出现在界面右最端。由此呈现出的视觉效果即管道不断出现并向左匀速移动。

在游戏界面开始的 0 到 15 秒，管道的横坐标每次递减 3，管道匀速缓慢向左移动；此后每五秒，管道向左移动的幅度逐渐变大，直到 30 秒后管道纵坐标每次递减 6，由此实现管道移动速度的变快。

```
class bomb(object):
    def __init__(self):
        self.bombx = 460
        self.bomby = random.uniform(30,350)
        self.bombpic = pygame.image.load("assets/sprites/bomb.png").convert_alpha()

    #定义炸弹移动函数
    def updatebomb(self):
        self.bombx -= 3
        if 15 < t2-t1 < 20:
            self.bombx -= 3.000000000000000000001
        if 20 < t2-t1 < 25:
            self.bombx -= 3.01
        if 25 < t2-t1 < 30:
            self.bombx -= 3.02
        if t2-t1 >30:
            self.bombx -= 6
        if self.bombx < -50:
            self.bombx = 288
            self.bomby = random.uniform(30,350)
```

（刘欣语）

完全相同的方法定义一个炸弹类并定定义炸弹移动函数，设定合适的炸弹横坐标的初始值，使得炸弹在两个管道之间出现，并使炸弹的移动速度与管道完全吻合。

```
def createmap():
    background = pygame.image.load("assets/sprites/background-day.png").convert_alpha()
    base = pygame.image.load("assets/sprites/base.png").convert_alpha()
    gap = 130
    screen.blit(background,(0,0))
    screen.blit(pipeline.pipeup,(pipeline.pipex, pipeline.pipey))
    screen.blit(pipeline.pipedown,(pipeline.pipex, 320+gap+pipeline.pipey))
    screen.blit(bomb.bombpic,(bomb.bombx,bomb.bomby))
    screen.blit(base,(0,400))

    pipeline.updatepipeline()
    bomb.updatebomb()

    pygame.display.update()
```

（刘欣语）

定义背景函数依次显示背景、上下管道、炸弹及地基。

利用 screen.blit()函数显示图片，函数中包括两个参数，第一个参数为图片，第二个参数为图片左上角在界面中的位置坐标。

利用 pygame.display.update()更新界面。

3.4.4 小鸟纵坐标的实现

3.5.1 手势识别模式

小鸟以恒定速度下落，检测到手势识别时，小鸟改为以恒定速度上升。

```
if l==1:
    cv2.putText(frame,'Up',(0,50),font,2,(0,0,255),1,cv2.LINE_AA)
    birdy -= 8
else:
    cv2.putText(frame,'Down',(0,50),font,2,(0,0,255),1,cv2.LINE_AA)
    birdy += 6
```
（高婕）

3.5.2 眨眼识别模式

小鸟进行自由落体运动，检测到眨眼识别时，小鸟额外获得一个向上的速度。

```
birdy = 50 + 5*(t2 - t1)**2
```

```
if frame_counter >= eye_consec_frames:
    blink += 1
    birdy -= 20
frame_counter = 0
```
（高婕）

3.5 得分

首先，得分的检测非常简单，当管道移出画面即横坐标为 0 时，游戏还没有结束，则说明小鸟穿过了一个管道，故得分加一。

```
if pipeline.pipex == 0:
    SOUNDS['point'].play()
    scoreinfo += 1
```
（张珂欣）

其次是 showscore 函数的定义，用于在结束界面显示得分。由于分数的位数不确定，而我们的图片只有 0 到 9 的基础数字，故我们首先创造一个列表储存每一个数位上的数字，然后计算整个得分图片的宽度，并根据该宽度确定整个图片的坐标使其居中：

```
def showscore(score):
    digits = [int(x) for x in list(str(score))]
    totalwidth = 0

    for digit in digits:
        totalwidth += numbers[digit].get_width()

    scorex = (width-totalwidth) / 2
```
（张珂欣）

最后在画面上显示得分：

```
    for digit in digits:
        screen.blit(numbers[digit],(scorex, height*0.1))
        scorex += numbers[digit].get_width()
```
（张珂欣）

4 局限与改进

4.1 局限

代码较为繁琐、冗长。由于时间的限制，没有办法很详尽地了解第三方类库所能实现的

功能，因此在一些功能的实现上代码的编写不够简洁。在整合代码的过程中，由于不同成员完成负责板块代码编写时的思路不同，代码显得较为杂乱，条理不清。

得分的检测不准确，代码经多次修改调试后仍然无法做到通过管道数与得分相同，若采用“<=”则分数会因为画面不断刷新变得很大，若采用“==”则有时电脑检测不到得分。

识别精度不够。由于参数的设置不够精确导致手势识别和眨眼识别的准确率不够高，且是否识别到手势与小鸟移动之间存在一定的时间滞后，玩家的游戏体验有待提高。

4.2 改进

项目最开始采用眨眼识别控制，然而由于眨眼识别精度较低，实际游戏体验并不好，于是我们开发了用手势控制的模式，获得了较好的游戏体验，但是眨眼识别模式仍留有遗憾，希望未来可以运用机器学习训练模型，达成较好的识别效果。此外，尽管我们对游戏当中的一系列参数（如障碍物移动速度、小鸟灵敏度等等）进行了测试和调整，但也有可能未能调成最优参数，没有带来最好的游戏体验，玩家可以在源代码中尝试修改优化。

5 致谢

感谢 Github，CSDN，哔哩哔哩等网站，这些平台为我们在编写代码的过程中提供思路，让我们能够解决所遇到的问题。感谢愿意贡献资源、分享经验的所有人。

感谢鲍杨老师和助教这一学期以来的辛勤付出，在我们遇到问题的时候耐心解答，激发我们学习 python 的兴趣。《程序设计》这门课使作为编程初学者的我们受益匪浅，为我们提供了一种全新的解决问题的方式，相信在之后的日子里大家会尝试用编程解决更多实际问题。

6 个人贡献简述

高婕：

① 参与项目确立的讨论、撰写项目方案
② 自主学习 cv2、dlib 的使用，编写手势识别、眨眼识别、小鸟纵坐标的程序
③ 参与编程部分的报告撰写
④ 剪辑 demo 视频

刘欣语：

① 参与项目确立的讨论、撰写项目方案
② 自主学习 pygame 的使用，编写代码（具体负责部分已在代码解释中标出）
③ 参与项目报告的撰写，整合、格式修改
④ 录制 demo——游戏展示

张珂欣：

① 参与项目确立的讨论、撰写项目方案
② 自主学习 pygame 的使用，编写代码（具体负责部分已在代码解释中标出）
③ 参与项目报告的撰写及代码贴图展示
④ 录制 demo——项目简介