上海交通大學

SHANGHAI JIAO TONG UNIVERSITY

2020 年春　《程序设计》项目报告

安泰喵大作战

——基于 handtracking 的 pygame 小游戏

学生姓名: 朱程麟、程维夏

指导教师: 鲍杨

学院(系): 安泰经济与管理学院

班级: (2019-2020-2)-CS159-2

2020.06

目录

一、项目背景

几年前，微信小程序《飞机大战》风靡网络；几年后，随着计算机深度学习的发展，已能做到用简单代码实现粗略的手势识别与跟踪。本项目希望将以往的简单小游戏与如今新领域的技术想结合，并加入上海交通大学安泰经济与管理学院的特色元素，用 python 代码实现交大特别版“安泰喵”大作战 PLUS。

二、游戏简介

玩家通过手势的左右移动，控制游戏窗口中安泰喵的移动，以躲避 GPA 与 SJTU 的攻击，并可以通过键盘 123 键，切换安泰喵的武器，最终“getGPA”“enterSJTU”并获得分数。

其中 GPA 为“普通敌人”，共 10 滴血，击毁得 100 分，随机在屏幕最上方出现，并以固定速度下落；SJTU 为“精英敌人”，共 30 滴血，击毁得 500 分，一次只出现一只，固定在屏幕最上方的版面里以固定速度游走，会发射子弹。

安泰喵的武器共三种，以不同的爱心数区分，爱心数越多，威力越大，但发射频率会减慢。

当安泰喵与“敌人”碰撞，或是被子弹击中，游戏结束，弹出结束界面，可以点击“-Retry-”重新开始游戏，或点击“ranking list”查看排行榜。

三、使用工具

（一）pygame

Pygame 是由 Pete Shinners 开发的跨平台 Pyth，它建立在 SDL 基础上，允许实时电子游戏研发而无需被低级语言（如机器语言和汇编语言）束缚。基于这样一个设想，所有需要的游戏功能和理念都（主要是图像方面）都完全简化为游戏逻辑本身，所有的资源结构都可以由高级语言提供，如 Python。

在本项目中主要利用 pygame 搭建游戏主框架。

（二）openCV

OpenCV 是一个基于 BSD 许可（开源）发行的跨平台计算机视觉和机器学习软件库，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，被广泛应用于物体识别、人机互动、无人驾驶等领域。

在本项目中主要利用 OpenCV 处理摄像头输入的实时图像，更好的实现手势跟踪的效果。

（三）TensorFlow

TensorFlow 是一个基于数据流编程的符号数学系统，被广泛应用于各类机器学习算法的编程实现，其前身是谷歌的神经网络算法库 DistBelief 。其拥有多层级结构，可部署于各类服务器、PC 终端和网页并支持 GPU 和 TPU 高性能数值计算，被广泛应用于谷歌内部的产品开发和各领域的科学研究。

在本项目中主要利用 TensorFlow 深度学习各种手势动作，得到模型。

四、代码介绍

（一）导入库

```
#pygame部分
import pygame
from pygame.locals import *
import sys
import random
import codecs
#handtracking部分
from utils import detector_utils as detector_utils
import cv2
import tensorflow as tf
import datetime
import argparse
```

（二）设置参数

此处以窗口及自机的参数设置为例，敌机子弹有类似参数，不过多重复。

```
#初始化窗口
pygame.init()
screen = pygame.display.set_mode((screen_width,screen_height))
pygame.display.set_caption("飞机大战lz版")
background = pygame.image.load("photo_new//background.png")
screen_width = 479
screen_height = 700
# 自机
player_image = pygame.image.load("photo_new//player.png")
player_down_image = pygame.image.load("photo_new//player.png")

# 自机构建及真实碰撞体积
player_rect = pygame.Rect(0,0,100,111)
player_img = player_image.subsurface(player_rect)
player_down_img = player_down_image.subsurface(player_rect)
player_pos=[240,500]
player_true_pos = [280,550]
player = Player(player_img,player_rect,player_pos)
player_true_rect = pygame.Rect(45,57,10,10)
player_true = Player(player_img,player_true_rect,player_true_pos)
```

（三）定义类与方法

1. 定义读写文件的函数，方便排行榜记录数据

```
def read_txt(path): # 读取txt文件
    with open(path, "r") as f:
        lines = f.readlines()
    return lines

def write_txt(context, srtim, path): # 重写txt文件
    f = codecs.open(path, srtim, 'utf8')
    f.write(str(context))
    f.close()
```

2. 定义自机、敌机、子弹（以自机为例）

```
#飞机
class Player(pygame.sprite.Sprite):
    def __init__(self, player_image, player_rect, init_pos):
        pygame.sprite.Sprite.__init__(self)
        self.image = player_image    #飞机
        self.rect = player_rect
        self.rect.topleft = init_pos #飞机左上角坐标
        self.speed = 10 # 飞机速度
        self.bullets = pygame.sprite.Group()
        self.rockets = pygame.sprite.Group()
        self.image_index = 0 #图片索引
        self.is_hit = False #是否被击中
#发射子弹
    def shoot(self, bullet_img):
        bullet = Bullet(bullet_img, self.rect.midtop)
        self.bullets.add(bullet)
    def shootrocket(self, rocket_img):
        rocket = Rocket(rocket_img, self.rect.midtop)
        self.rockets.add(rocket)
#飞机移动
    def moveUp(self):
        if self.rect.top >= 0:
            self.rect.top += -self.speed
    def moveDown(self):
        if self.rect.top <= screen_height-self.rect.height:
            self.rect.top += self.speed
    def moveLeft(self):
        if self.rect.left >= 0:
            self.rect.left += -self.speed
    def moveRight(self):
        if self.rect.left <= screen_width - self.rect.width:
            self.rect.left += self.speed
```

3. 定义相机

```
def cam():
    global cap, im_width, im_height
    cap = cv2.VideoCapture(args.video_source)
    cap.set(cv2.CAP_PROP_FRAME_WIDTH, args.width)
    cap.set(cv2.CAP_PROP_FRAME_HEIGHT, args.height)
    detection_graph, sess = detector_utils.load_inference_graph()
    im_width, im_height = (cap.get(3), cap.get(4))
    num_hands_detect = 1
    cv2.namedWindow('Single-Threaded Detection', cv2.WINDOW_NORMAL)
    num_left, num_right, num_top, num_bottom=0, 0, 0, 0
```

（四）脚本编写

1. 生成敌机（以 boss 为例）

```
# 生成Boss
if len(bossass) == 0:
      if boss_frequency % 200 == 0: # boss出现的最低频率
            boss_pos = [screen_width/2,50]
            boss = Boss(boss_img,boss_down_img,boss_pos)
            bossass.add(boss) # 添加到group中
      boss_frequency += 1
      if boss_frequency >= 200:
            boss_frequency = 0
for boss in bossass:                  # 随机游走
      if boss.rect.left <= 20 and boss.rect.right <= screen_width:
            boss.moveright2()         # 碰到边缘换方向
            n = 0
      elif boss.rect.right >= screen_width-20 and boss.rect.left >= 0:
            boss.moveleft2()
            n = 1
      if n == 1:
            boss.moveright()
      else:
            boss.moveleft()
      if boss_shoot_frequency % 30 == 0:    # 发射子弹
            boss.shoot(bullet0_img)
      boss_shoot_frequency += 1
      if boss_shoot_frequency >= 30:
            boss_shoot_frequency = 0
for bullet in boss.bossbullets: # 子弹group
      bullet.downmove()
      if pygame.sprite.collide_circle(bullet,player_true):
            player.is_hit = True
            cv2.destroyAllWindows() # 关闭摄像头
            break
      if bullet.rect.bottom <= 0:
            boss.bossbullets.remove(bullet) # 到屏幕底端的子弹消失
for boss in bossass:    # 血槽
      pygame.draw.line(screen, (79,148,205), (boss.rect.left, boss.rect.top),
                              (boss.rect.left + boss.rect.width*boss.blood/30, boss.rect.top),4)
```

2. 切换武器

```
    if not player.is_hit:
          # 武器切换
          if weapon == 0:
                if shoot_frequency % 10 == 0:    #控制发射速度
                      player.shoot(bullet_img)
                shoot_frequency += 1
                if shoot_frequency >= 15:
                      shoot_frequency = 0
          elif weapon == 1:
                if shoot_frequency % 15 == 0:
                      player.shoot(bullet2_img)
                shoot_frequency += 1
                if shoot_frequency >= 20:
                      shoot_frequency = 0
          elif weapon == 2:
                if shoot_frequency % 25 == 0:
                      player.shootrocket(rocket_img)
                shoot_frequency += 1
                if shoot_frequency >= 25:
                      shoot_frequency = 0
```

3. 碰撞机制（以 boss，自机为例）

```
# 自机
if not player.is_hit:
      screen.blit(player_image,player.rect)
else:
      for i in range(100):
            pygame.display.update()
      running = False # 游戏结束
```

```
# boss
for b in bossass:
    for bu in player.bullets:
        if pygame.sprite.collide_mask(b, bu):
            b.blood += -2     # 血槽变化
            score += 10       # 分数变化
            pygame.draw.line(screen, (79,148,205), (b.rect.left, b.rect.top),
                                        (b.rect.left + b.rect.width*b.blood/30, b.rect.top),4)
            player.bullets.remove(bu)
    if b.blood <= 0:              # boss坠毁
        bossass_down.add(b)
        bossass.remove(b)
    for boss_down in bossass_down: # 坠毁照片
        for _ in range(200):
            screen.blit(boss_down.down_img,boss_down.rect)
        bossass_down.remove(boss_down)
        score += 500
```

4. 排行榜数据记录

```
j = 0
arrayscore = read_txt("score.txt")[0].split("/")# 读取排行榜数据
for i in range(len(arrayscore)):          # 与之前成绩比较
    if score > int(arrayscore[i]):
        j = arrayscore[i]
        arrayscore[i] = str(score)
        score = 0
    if int(j) > int(arrayscore[i]):
        k = arrayscore[i]
        arrayscore[i] = str(j)
        j = k
for p in range(len(arrayscore)):          # 重写排行榜数据
    if p == 0:
        write_txt(arrayscore[p]+"/","w","score.txt")
    elif p == len(arrayscore)-1:
        write_txt(arrayscore[p],"a","score.txt")
    else:
        write_txt(arrayscore[p]+"/","a","score.txt")
```

5.排行榜界面（部分）

```
def gameranking():        # 排行榜界面函数
    screen2 = pygame.display.set_mode((screen_width,screen_height)) # 背景
    screen2.fill(0)
    screen2.blit(background,(0,0))
    arrayscore = read_txt("score.txt")[0].split("/") # 读取数据
    for i in range(1,len(arrayscore)+1):       # 打印数据
        font = pygame.font.Font(None,30)
        text = font.render(str(i) + " " +arrayscore[i-1],True,(150,150,150))
        text_rect = text.get_rect()
        text_rect.centerx = screen2.get_rect().centerx
        text_rect.centery = 130 + 30*i
        screen.blit(text,text_rect)
```

6. 结束界面（部分）

```
screen.blit(background,(0,0)) # 背景图
gameover_font = pygame.font.Font(None,80)    # 字体，字号
gameover_text = gameover_font.render("GAME OVER",True,(16,78,139)) # 文本，颜色
gameover_text_RECT = gameover_text.get_rect()
gameover_text_RECT.centerx = screen.get_rect().centerx    # x轴坐标
gameover_text_RECT.centery = screen.get_rect().centery    # y轴坐标
screen.blit(gameover_text,gameover_text_RECT ) # 显示
```

```
while True:
      clock = pygame.time.Clock()
      clock.tick(25) # 刷新频率
      for event in pygame.event.get():
            if event.type == pygame.QUIT: # 退出游戏
                  pygame.quit()
                  sys.exit()
            elif event.type == pygame.MOUSEBUTTONDOWN:
                  if screen.get_rect().centerx-70<=event.pos[0] and screen.get_rect().centerx+50>=event.pos[0] and
screen.get_rect().centery+120<=event.pos[1] and screen.get_rect().centery+160>=event.pos[1]: # 定义鼠标有效点击区域
                        startGame() # 重新开始
                  elif screen.get_rect().centerx+80<=event.pos[0] and screen.get_rect().centerx+200>=event.pos[0] and
screen.get_rect().centery+250<=event.pos[1] and screen.get_rect().centery+350>=event.pos[1]:
                        gameranking() # 排行榜
      pygame.display.update() # 刷新界面
```

7. 手势跟踪

```
# 手掌识别的参数输入
parser = argparse.ArgumentParser()
parser.add_argument('-sth','--scorethreshold',dest='score_thresh',type=float,default=0.2,help='Score threshold for displaying bounding boxes')
parser.add_argument('-src','--source',dest='video_source',default=0,help='Device index of the camera.')
parser.add_argument('-wd','--width',dest='width',type=int,default=320,help='Width of the frames in the video stream.')
parser.add_argument('-ht','--height',dest='height',type=int,default=180,help='Height of the frames in the video stream.')
parser.add_argument('-ds','--display',dest='display',type=int,default=1,help='Display the detected images using OpenCV. This reduces FPS')
parser.add_argument('-num-w','--num-workers',dest='num_workers',type=int,default=4,help='Number of workers.')
parser.add_argument('-q-size','--queue-size',dest='queue_size',type=int,default=5,help='Size of the queue.')
args = parser.parse_args()

 # 识别手掌
 num_left,num_right,num_top,num_bottom=0,0,0,0
 while True:
       num_hands_detect = 1
       ret, image_np = cap.read()
       try:
             image_np = cv2.cvtColor(image_np, cv2.COLOR_BGR2RGB) # 颜色处理
       except:
             print("Error converting to RGB")
       #获取位置
       boxes, scores = detector_utils.detect_objects(image_np,detection_graph, sess) # 调用
       detector_utils.draw_box_on_image(num_hands_detect, args.score_thresh,scores, boxes, im_width, im_height,image_np)
       left,right,top,bottom = detector_utils.record_location(num_hands_detect, args.score_thresh,scores, boxes, im_width, im_height,image_np)

       cv2.imshow('Single-Threaded Detection', cv2.cvtColor(image_np, cv2.COLOR_RGB2BGR)) # 打开窗口
       break
 # 判断方向
 hor=(left+right)/2
 ver=(top+bottom)/2

 if (ver*im_width+hor*im_height) < im_width*im_height:
       if im_width*ver > im_height*hor:
             num_right+=1
       elif im_width*ver < im_height*hor:
             num_bottom+=1
 elif (ver*im_width+hor*im_height) > im_width*im_height:
       if im_width*ver > im_height*hor:
             num_top+=1
       elif im_width*ver < im_height*hor:
             num_left+=1

 num_max=max(num_left,num_right,num_top,num_bottom)
 if num_max != 0:     # 飞机移动
       if num_left == num_max:
             hand_direction = 'left'
       elif num_right == num_max:
             hand_direction = 'right'
       elif num_top == num_max:
             hand_direction = 'down'
       elif num_bottom == num_max:
             hand_direction = 'up'
 num_left,num_right,num_top,num_bottom=0,0,0,0
```

五、 各成员工作描述

（朱程麟）：作为队长，我决定了这次 python 项目的方向，就是做一个有图像识别或者声音识别的弹幕游戏。在立项之后，我用 pygame 构建了主要的游戏框架，并在基础的弹幕游戏元素外添加了武器切换、血条、真实判定点等

元素。在队员成功找到并实现 handtracking 的包之后，我负责将手势识别与游戏本体对接并结合，从而实现“手势识别控制飞机移动”这一目的。

（程维夏）：我主要负责本项目库的模块，从检索何时的库，到解决安装上的各种问题，再到具体使用库中的各类函数与方法，其中包括 github 上 handtracking 包，speechrecgnition 语音识别，pyaudio 音量检测等。并负责游戏的所有美工处理和优化，用手绘和 photoshop 技术准备此项目的全部图片。此外，我同时也负责了项目报告的撰写以及项目视频的拍摄、剪辑与后续工作。

六、 问题与不足

一）手势识别时会有个别帧对于手势定位不准确。

二）手势跟踪模块的加入使游戏资源占用，略有卡顿现象。

三）本项目原想加入通过语音识别更换武器的部分，但由于加入后画面帧数过于低，所以舍弃此部分。

```
from pyaudio import PyAudio, paInt16 # 导入pyaudio
import pygame
from pygame.locals import *
import struct

class Voice(pygame.sprite.Sprite):
      def __init__(self):
            self.NUM_SAMPLES = 1000    # pyAudio内部缓存的块的大小
            self.LEVEL = 1500    # 声音保存的阈值

            # 开启声音输入
            pa = PyAudio()
            SAMPLING_RATE = int(pa.get_device_info_by_index(0)['defaultSampleRate'])
            self.stream = pa.open(format=paInt16, channels=1, rate=SAMPLING_RATE, input=True, frames_per_buffer=self.NUM_SAMPLES)

def change_weapon():
      volume = 0
      weapon=0
      voice = Voice()
      # 读入NUM_SAMPLES个取样
      string_audio_data = voice.stream.read(voice.NUM_SAMPLES)
      k = max(struct.unpack('1000h', string_audio_data))
      if k > 3000:        # 音量累计超过一定数则更换武器
            volume += 1
      if volume % 30 == 0:
            weapon = (weapon+1)%3
            volume = 0
      print(weapon)
```

七、 未来展望

基于上述问题，我们小组认为，未来可以对于手势跟踪与游戏结合的模块进行优化，更好的增加人机互动，并可以将手势跟踪的想法应用到更多的场合之中。同时，我们也希望将来可以把此次没有实现的声音识别完善到位，增加游戏的可玩性。此外，游戏本身也可以增加一些元素，如关卡制度、随机事件发生、不同的自机等，从而使游戏变得更为有趣。相信通过不断的创新，我们能够拓展出更多 python 模块的应用场景，让这个世界变得更有趣、更方便、更美好！

很感谢鲍杨老师在这个学期里对我们的教学与指导，让我们从接触python，到学习python，再到应用python。这些过程中，我们获得的到的不只是书本上有限的内容，更多的是学习到自己探索知识的方法，真真切切的体会到程序设计的乐趣与实用。相信未来，我们也将保持这份初心，学习与应用python这门美妙的语言！