Python 语言基础与应用课程实习报告

（曾子欣*，钱运杰，王鸿铖，张瑞宸）

摘要：本小组使用 Python 语言开发了一款多媒体软件：基于 Pygame Zero 模块的单人冒险闯关类小游戏“抗疫英雄”。玩家操作一名一线抗疫的医生，通过移动收集地图中随机出现的免疫细胞，加快疫苗研发进程；同时还要躲避游荡的病毒，利用道具和机关保护自己；最终以被病毒杀死之前获得疫苗为通关胜利条件。游戏共设 10 关，难度随关卡数递增，包括口罩保护、消毒水杀毒、传送门瞬移和病毒突变等游戏元素，另设有排行榜功能记录闯关数和用时，具有较高的策略性和可玩性。

一、选题及创意介绍

2020 年是特殊的一年。疫情对我们的生活产生了巨大的影响，同时也让我们见证了医护人员们的勇敢和奉献。本作取材于现实，以闯关小游戏的形式复现医生争分夺秒研制疫苗，与病毒抗争的过程，旨在致敬那些奋斗在一线的抗疫英雄们。

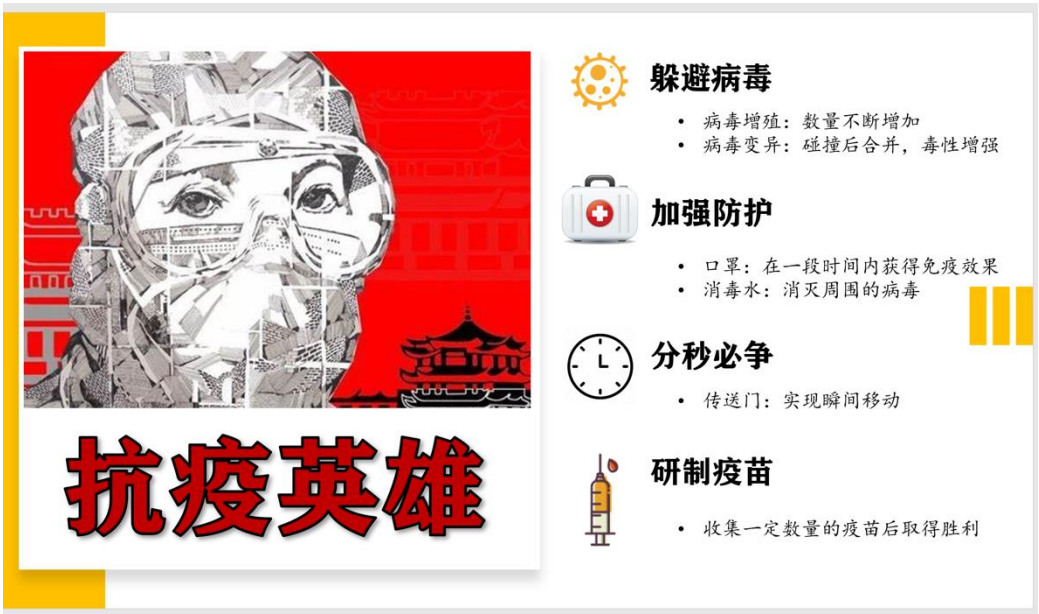

图 1-1 游戏创意介绍

“抗疫英雄”在传统的冒险闯关游戏模式上，结合时事背景尝试了独特的游

戏元素。玩家操作的医生不仅需要躲避游荡的病毒生存下来，还要挑战在尽可能短的时间内收集足够数量的免疫细胞，研发出疫苗获得游戏胜利。为增加游戏挑战性，病毒的伤害和移动速度随关卡数递增，且普通病毒经过碰撞能突变合并为毒性更强的大病毒。同时，玩家除了单纯操纵主角移动，还有丰富的策略用来对抗病毒，如使用口罩进入短暂无敌状态，使用消毒水清除人物附近的病毒，使用传送门进行瞬移提高机动性等等。最后，在成功通关的基础上，玩家可以挑战极限用时下通关，不断刷新排行榜上的成就。

二、设计方案

游戏设想的运行效果图如下：

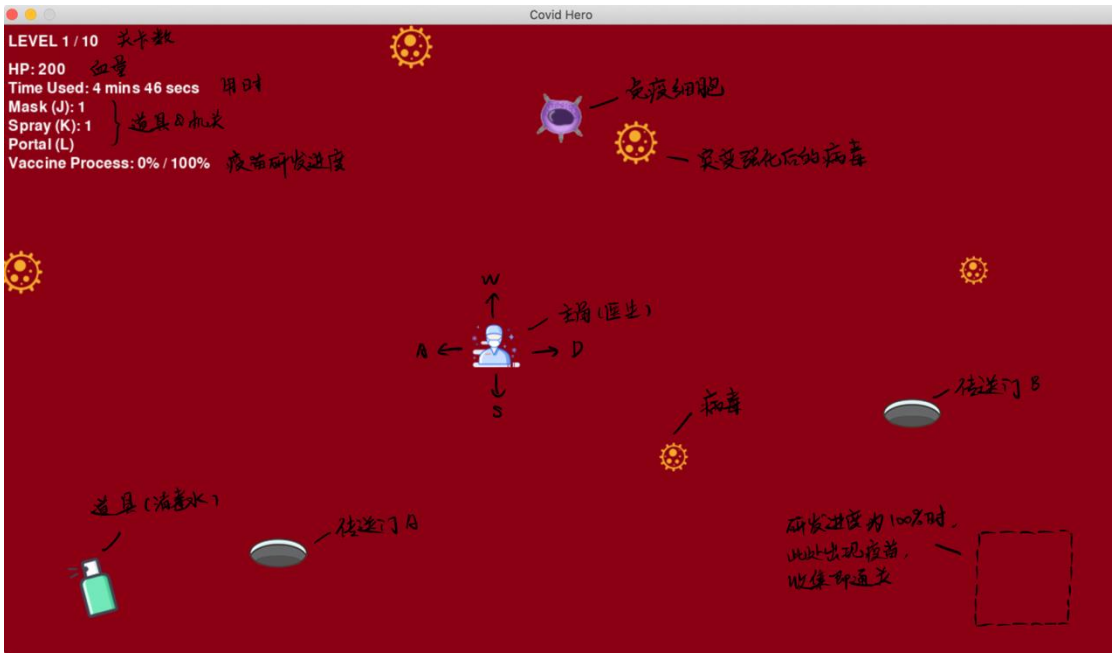

图 2-1 游戏界面

具体的功能模块有：

2.1 主角/医生

1. 移动：WASD 键控制人物上下左右移动；

2. 血量（HP）：与病毒碰撞扣血（没有道具保护情况下），血量为 0 死亡游戏失败。每次通关或重来重置血量，随关卡数增加初始血量按比例上调。

2.2 病毒

1. 数量设定：病毒在全屏任意位置随机生成，移动方向随机。病毒分四个强度等级，初始化时均为一级病毒；
2. 伤害设定：病毒伤害随关卡数和病毒突变等级递增；
3. 突变设定：同等级病毒碰撞后会合并升级，体积和伤害增加。

2.3 道具和机关

1．道具

道具包括口罩（Mask）和消毒水（Spray），在地图随机出现可供拾取，之后按指定键位一次性使用。道具的出现频率，上限和使用冷却时间都有规定。人物所拥有的道具数量和键位提示显示在左上角的道具栏。其中，口罩的作用是指定时间内与病毒碰撞不受伤害；消毒水的作用是立刻杀死人物周围一定半径内的所有病毒；

2. 机关

机关包括传送门（Portal），游戏开始不久随机位置刷出，在该局游戏之后便一直存在。传送门两道成对出现，进入一道可以立即传送到另一道所在位置，是躲避病毒、提高机动性的利器。

2.4 疫苗研制

1. 免疫细胞：地图上随机位置刷出，人物需要收集免疫细胞来推进疫苗研发进度（Vaccine Process）。关卡数越高，研发疫苗所需免疫细胞越多；
2. 疫苗：研发进度达 100%时，在地图右下角固定位置出现疫苗，控制人物前往拾取后即通关，进入下一关。

2.5 其他功能

1. 计时：游戏启动计时开始，闯关计时继续，失败重来计时重置；
2. 关卡：设置一定总关数（10 levels），首先设置 level 0 教程关帮助玩家熟悉游戏操作，之后进入正式关卡，每通一关地图和人物属性重置，全部关卡通过则一

轮游戏胜利，否则返回 level 1 重来；

3. 排行榜：记录每次最高关卡挑战记录及总耗时，游戏失败或全通后显示；

三、实现方案及代码分析

3.1 文件和库说明

1. covidhero.py: 源代码，调用的库有 pgzero, random 和 time；

2. images, music, sounds: 主程序运行所需的图片、音乐和声音文件；

3. record.txt: 排行榜记录。

3.2 自定义类说明

3.2.1 class man（主角/医生类）

```
class man(Actor):
    life = 200  # 生命值
    dis = 0  # 消毒水数量
    mask = 0  # 口罩数量
    target_pt = 200
    temp_pt = 0

    def move(self):  # 判断移动
        global SPEED
        if keyboard[keys.A] and self.left > 0:
            self.left -= SPEED
        if keyboard[keys.D] and self.right < WIDTH:
            self.right += SPEED
        if keyboard[keys.W] and self.top > 0:
            self.top -= SPEED
        if keyboard[keys.S] and self.bottom < HEIGHT:
            self.bottom += SPEED
```

功能：初始化人物参数（血量，道具数量等）；判断人物移动。

实现：继承 Pygame Zero 内置的 Actor 类，由 keyboard 类设置 WASD 按键响应上下左右方向移动。

3.2.2 class V（病毒类）

```
class V(Actor):
    def Init(self, i):  # 细菌初始化
        global gamelevel
        self.dir = random.randint(1, 4)
        x = random.randint(1, 8)
        y = random.randint(1, 6)
        self.tag = i
        self.hurt = 15 + 5 * gamelevel
        self.level = 1
        self.pos = x * 100, y * 100
        self.loop = 60

    def draw_v(self):
        self.draw()

    def move_v(self):  # 细菌移动
        global gamelevel
        if self.loop == 0:
            self.dir = random.randint(1, 4)
            self.loop = 20
            self.move_v()
        else:  # 移动速度随着游戏关卡的增加而增加
            self.loop -= 1
            if self.dir == 1:
                self.left += step / 10 + gamelevel
                if self.left + w > WIDTH:
                    self.left = WIDTH - w
            elif self.dir == 2:
                self.left -= step / 10 + gamelevel
                if self.left < 0:
                    self.left = 0
            elif self.dir == 3:
                self.top += step / 10 + gamelevel
                if self.top + h > HEIGHT:
                    self.top = HEIGHT - h
            else:
                self.top -= step / 10 + gamelevel
                if self.top < 0:
                    self.top = 0
```

功能：初始化病毒参数（伤害，生成位置等）；判断病毒移动。

实现：继承 Pygame Zero 内置的 Actor 类，由 random 模块随机化病毒的初始位置和移动方向，并由条件判断确保病毒在地图范围内移动。

3.2.3 class Game（游戏控制类）

```
class Game:  # 游戏页面跳转：初始页面--规则说明--游戏中--结束（胜利/失败）
    def __init__(self):
        self.gameOn = 1
        self.gameMessage = "Welcome to Covid Hero!\nPRESS SPACE TO START GAME"
        self.play_sound = True

    def checkGameOver(self):
        if doctor.life <= 0 and self.gameOn != 6:
            self.gameOn = 2
            self.gameMessage = "Game Over, You Lose!\nPRESS SPACE TO RESTART\nPRESS R TO CHECK THE RANKING LIST"  # 失败

            global flag
            flag += 1
            if (
                flag == 1 and gamelevel >= 1 and Allgrade
            ):  # 仅在第一次checkGameOver时记录成绩信息；只有通过第一关的成绩才需要记录
                minutes, seconds = Allgrade[gamelevel - 1]
                elap = int(60 * minutes + seconds)
                f = open("record.txt", "a")  # 改写本地文件
                f.write("\n" + str(gamelevel))
                f.write("\n" + str(elap))
                f.close()

        for vaccine in Vaccinelist:
            if doctor.colliderect(vaccine):
                if len(Allgrade) <= gamelevel:
                    elap = time.time() - star  # 获取时间差
                    minutes = int(elap / 60)
                    seconds = int(elap - minutes * 60.0)
                    Allgrade[gamelevel] = minutes, seconds
                    print(Allgrade)

                if gamelevel < 10:
                    self.gameOn = 3  # 进入下一关卡
                    self.gameMessage = "You Win!\nPRESS SPACE TO THE NEXT LEVEL"
                else:
                    self.gameOn = 4  # 游戏结束,胜利
                    flag += 1
                    if flag == 1:  # 仅在第一次checkGameOver时记录成绩信息
                        minutes, seconds = Allgrade[gamelevel]
                        elap = int(60 * minutes + seconds)
                        f = open("record.txt", "a")
                        f.write("\n" + str(gamelevel))
                        f.write("\n" + str(elap))
                        f.close()
```

功能：游戏状态判断（如进行中或结束）；排行榜记录成绩信息。

实现：在 Game 类中定义初始参数 gameOn 作为不同游戏状态的控制门，与 draw() 和 update()函数配合，实现不同状态下相应游戏页面及提示信息的展示和事件响应。具体如表 3-1：

gameOn	状态
1	标题
1.1	规则说明
1.5	游戏中
2	结束（失败）
3	结束（胜利）
4	结束（全通关）
6	排行榜展示

表 3-1 游戏状态一览

此外，使用 time 模块统计游戏用时，创建空白 txt 文件记录游戏进度，通过 open(), write()方法将历史用时和最高通关数写入文件保存。

3.2.4 class rank（排名类）

```python
class rank:  # 游戏成绩
    def __init__(self, level, time):
        self.level = level  # 通过的关卡数量
        self.time = time  # 用时

    def __lt__(self, other):
        return (
            self.level > other.level
            if self.level != other.level
            else self.time < other.time
        )
        # 若通过的关卡数相同，则比较用时
```

功能：排行榜上的通关记录排序。

实现：以通过的最高关卡数和最短用时分别为第一、第二标准排序，由内置的__lt__(self, other)函数实现 sort 功能。

3.3 函数说明

3.3.1 reset 函数

```python
def reset():
    global gamelevel,star
    # 重开游戏之后对医生初始化
    doctor.life = 200 + 20 * gamelevel
    doctor.pos = 50, 100
    doctor.target_pt = 200 + 100 * gamelevel
    doctor.temp_pt = 0
    flag = 0
    file_flag = 0
    star = time.time()
    gamelevel += 1
    Alldis.clear()
    Allmask.clear()
    doctor.dis=0
    doctor.mask=0
    # 对病毒，传送门，疫苗初始化
    global num
    num = 6
    viruses.clear()
    for i in range(num):
        vi = V("virus_1")
        vi.Init(i)
        viruses.append(vi)

    Allgate.clear()
    gate_location.clear()
    Vaccinelist.clear()
```

每局游戏开始前运行 reset 函数，将人物属性、病毒、道具和疫苗等参数初始化，由 clear()方法将列表中已保存的元素清除。

3.3.2 draw 函数

draw 函数的功能是绘制静态游戏页面、对象以及提示信息，主要通过调用 screen 类和 draw()方法来实现。具体绘制内容包括：

1.不同游戏状态对应的静态页面，见前文表 3-1；

```
def draw():
    if game.gameOn == 1:  # 游戏开始
        screen.clear()
        screen.blit("startpage", (0, 0))
    elif game.gameOn == 1.1:  # 规则
        screen.clear()
        screen.blit("rules-2", (0, 0))
    elif game.gameOn == 2: # 失败
        screen.clear()
        screen.blit("deadpage", (0, 0))
        screen.draw.text(
            game.gameMessage, color="white", center=(HEIGHT * 6 / 7, WIDTH / 2)
        )
    elif game.gameOn == 3: # 胜利
        screen.clear()
        screen.blit("win_stage", (0, 0))
        screen.draw.text(
            game.gameMessage, color="black", center=(HEIGHT * 3 / 4, WIDTH / 2)
        )
    elif game.gameOn == 4: # 最后通关
        screen.clear()
        screen.blit("endpage", (0, 0))
```

2.人物属性栏，位于左上角，在 update 函数配合下动态显示当前关卡数、游戏总用时、人物血量、拥有道具数和疫苗研发进度；

```
        if gamelevel == 0:
            screen.draw.text(
                "LEVEL %d (Tutorial Level)\n" % gamelevel, (10, 10), color="white"
            )
        else:
            screen.draw.text("LEVEL %d / 10\n" % gamelevel, (10, 10), color="white")
        screen.draw.text("HP: %d\n" % doctor.life, (10, 40), color="white")
        screen.draw.text("Mask (J): %d\n" % doctor.mask, (10, 80), color="white")
        screen.draw.text("Spray (K): %d\n" % doctor.dis, (10, 100), color="white")
        screen.draw.text("Portal (L)", (10, 120), color="white")
```

3.游戏中需要用到的各类对象，即主角/医生、病毒、道具、免疫细胞和疫苗。

```
257         doctor.draw()
258         for v in viruses:
259             v.draw_v()
260         for m in Allmask:
261             m.draw()
262         for m in Alldis:
263             m.draw()
264         for m in Allgate:
265             m.draw()
266         for m in Allcell:
267             m.draw()
268 
269         for vaccine in Vaccinelist:
270             vaccine.draw()
271             screen.draw.text("VACCINE IS READY!", (500, 60), color="white")
```

3.3.3 update 函数

update 函数主要包含对更新 draw 函数状态的各类事件判断，具体功能有：1.对象生成

```
394             # 生成一个病毒
395             if luck % (220 - 20 * gamelevel) == 0:
396                 sounds.item_generate.play()
397                 vi = V("virus_1")
398                 vi.Init(index)
399                 index += 1
400                 num += 1
401                 viruses.append(vi)
```

```
440             # 生成积分道具
441             if luck % 100 == 0 and len(Allcell) == 0:
442                 sounds.item_generate.play()
443                 if luck <= 400:
444                     m = Actor("macrophage")
445                 elif luck <= 800:
446                     m = Actor("natural_killer")
447                 elif luck <= 1200:
448                     m = Actor("cellt")
449                 elif luck <= 1600:
450                     m = Actor("cellb")
451                 elif luck <= 2000:
452                     m = Actor("dendritic_cell")
453                 x = random.randint(1, 8)
454                 y = random.randint(1, 6)
455                 m.pos = x * 100, y * 100
456                 Allcell.append(m)
```

由 random 模块随机 x, y 坐标确定生成位置，通过 luck 变量确定生成概率/频率。一个对象生成后，通过 append 方法将其保存到列表里。

2.道具使用

所有道具均创建一个相应的空列表存储。

(1) 口罩

```
        # 使用口罩
        if doctor.mask > 0 and keyboard[keys.J] and UsingMask == 0:
            sounds.wear_mask.play()
            doctor.mask -= 1
            UsingMask = 1
            maskstar = time.time()
        maskuse = time.time()
        masktime = int(maskuse - maskstar)
        if masktime > 3 and UsingMask:
            UsingMask = 0
```

口罩的保护功能写入了医生和病毒的碰撞判断里，只有在碰到病毒没戴口罩的情况才扣血；

(2) 消毒水

```
        # 使用消毒水，每次使用CD=0.5s
        if keyboard[keys.K] and doctor.dis >= 1 and UsingDis == 0:
            sounds.spray.play()
            sounds.spray.play()
            UsingDis = 1
            doctor.dis -= 1
            disstar = time.time()
```

消毒水是通过 distance 和 remove 方法消除医生附近范围的病毒；

(3) 传送门

```
        if luck % 100 == 0 and len(Allgate) < 2:
            sounds.portal_appear.play()
            m1 = Actor("gate")
            x = random.randint(120, 360)
            y = random.randint(160, 650)
            m1.pos = x, y
            Allgate.append(m1)
            gate_location.append(x)
            gate_location.append(y)

            m2 = Actor("gate")
            x = random.randint(840, 1080)
            y = random.randint(160, 650)
            m2.pos = x, y
            Allgate.append(m2)
            gate_location.append(x)
            gate_location.append(y)
```

传送门瞬移通过修改 doctor 对象的 pos 参数实现。

3.碰撞事件

```
        # 医生碰撞病毒扣血
        for v in viruses:
            v.move_v()
            if doctor.colliderect(v) and UsingMask == 0:
                doctor.life -= v.hurt
                if v.hurt >= 30 or doctor.life <= 100:
                    sounds.die.play()
                else:
                    sounds.hurt.play()
                viruses.remove(v)

        # 病毒碰撞和合并
        for v in viruses:
            for v2 in viruses:
                if v.tag != v2.tag:
                    if v.colliderect(v2) and v.level == v2.level:
                        sounds.virus_level_up.play()
                        v.level += 1
                        v.hurt *= 1.2
                        viruses.remove(v2)
                        num -= 1
                        if v.level == 2:
                            v.image = "virus_2"
                        elif v.level == 3:
                            v.image = "virus_3"
                        elif v.level == 4:
                            v.image = "virus_4"
```

由来自 Actor 类的 colliderect 方法和 if 语句实现碰撞判定；对于两个同等级病毒的碰撞，通过改变 level 及 image 参数将其升级为高级病毒。

4.游戏静态页面切换

```
    global rule_s
    if keyboard.space and game.gameOn == 1:
        game.gameOn = 1.1  # 游戏规则说明
        rule_s = time.time()
    if keyboard.space and game.gameOn == 1.1:
        if time.time() - rule_s > 0.5:
            game.gameOn = 1.5
    if keyboard.space and game.gameOn == 3:
        game.gameOn = 1.5
        reset()
    if keyboard.space and game.gameOn == 2:
        game.gameOn = 1
        gamelevel = 0
        reset()
    if keyboard[keys.R] and game.gameOn in (2, 4):
        game.gameOn = 6
    if keyboard.space and game.gameOn == 6:
        game.gameOn = 1
        gamelevel = 0
        reset()
```

通过 keyboard 类设置 SPACE 键切换，如首次登入游戏按空格开始游戏，游戏结束按空格跳转到新的一关等。

5.胜利/失败页面音效播放：

```
# 事件触发音效播放
if game.play_sound:
    if game.gameOn == 2:
        sounds.game_lose.play()
        game.play_sound = False
    if game.gameOn == 3:
        sounds.game_win.play()
        game.play_sound = False
    if game.gameOn == 4:
        sounds.game_victory.play()
        game.play_sound = False

if game.gameOn == 1.5 and game.play_sound == False:
    game.play_sound = True
```

如前文所述，通过切换 gameOn 参数实现不同游戏结果播放不同音效。由于 sounds 类默认循环播放，通过设置一个名为 play_sound 的 flag 增加额外 if 判断实现单次播放。

四、后续工作展望

4.1 已有功能优化

1. 提高病毒和免疫细胞生成的规律性，控制游戏时长稳定；
2. 美化背景图片，设置多个不同风格关卡地图和 bgm；
3. 数值平衡性补丁，包括人物血量、病毒伤害和移动速度等。
4. 减小 global 声明的使用，将主程序、draw 和 update 函数中的部分代码整合成自定义类封装，以提高基础设施代码的可读性和可维护性。

4.2 新增功能

1．添加人物受伤、死亡和获得疫苗通关的动画效果；
2. 加入事件触发提示，如扣血提示，下个免疫细胞出现倒计时，消毒水作用半径

提示；

3. 添加新的道具种类，包括一次性使用类、可存储使用次数类和被动 buff 类；

4. 首次通关后自由选择关卡难度。

五、小组分工合作

小组主要通过定期开线上腾讯会议和日常微信群讨论与交流。具体分工如下表所示：

成员	职责和贡献
曾子欣*	游戏规则制订；功能优化；游戏页面与海报设计
钱运杰	关卡设计；排行榜；道具优化
王鸿铖	主角、病毒类实现；功能实现与优化；视频制作剪辑
张瑞宸	游戏创意；主函数框架；音效；报告撰写

表 5-1 小组分工

组会记录和讨论过程如下图：

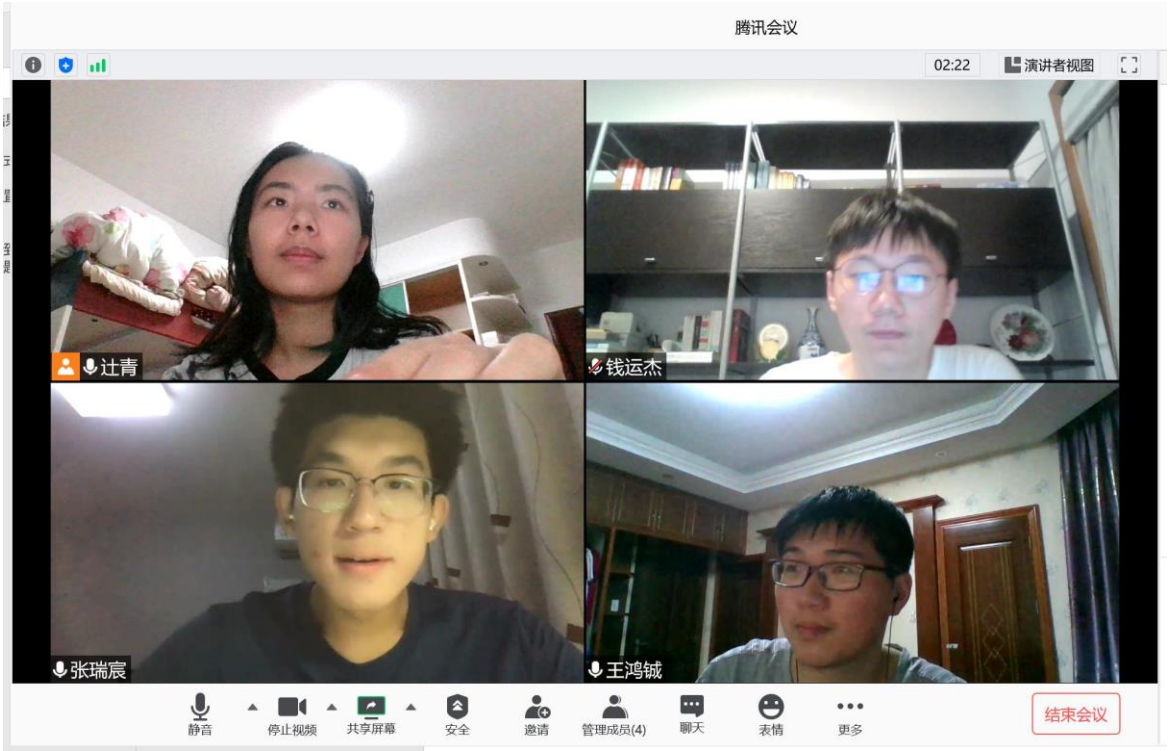

图 5-1 视频会议记录

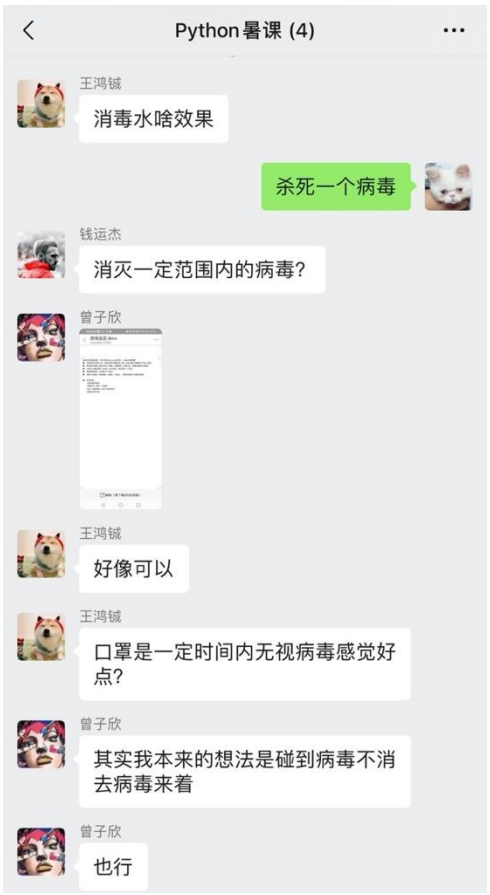

图 5-2 群组讨论记录

致谢

感谢陈斌老师的悉心指导，以及助教们两周以来认真负责的批改作业和飞书群答疑！

参考资料

[1] pgzero 官方文档：pygame-zero.readthedocs.io

[2]音效来源：www.epidemicsound.com

[3]部分图标来源：www.iconfont.cn