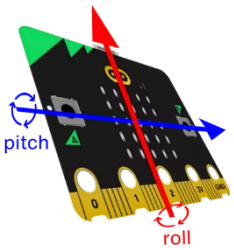
变异的吃豆人（单片机版）实习报告

摘要：以 micro:bit 编程经典案例“重力控制水平仪”为基础，参考经典小游戏吃豆人，我们对游戏规则和游戏呈现方式稍作改变，设计了这款变异的吃豆人（单片机版）小游戏。在原版游戏的基础上，我们为玩家提供不同游戏难度选项，并且对“目标点”的设置有一定修改。受限于单片机的“小巧”，这款小游戏尚有不足，期待进一步完善。我们希望变异的吃豆人（单片机版）可以向大家展示 micro:bit 编程的神奇魅力，并向经典游戏“吃豆人”致敬。

一、选题及创意介绍



利用 micro:bit 设计重力控制水平仪是 micro:bit 编程的经典案例之一。借助 micro:bit 上的加速度计，可以测量上下（纵向）以及左右（横向）的倾斜（如右图），从而实现一个重力控制上下左右滑动的 LED，将单片机的倾斜角度转化为显示面板上 LED 灯的位置，非常直观。

由此，我们设想到可以在 micro:bit 重力控制水平仪的基础上，与简单小游戏“吃豆人”相联系，设计一款通过改变单片机倾斜角度，控制“吃豆人”位置，在单片机控制面板上吃掉“目标点”获得得分的“变异的吃豆人（单片机版）”小游戏。

经过一系列天马行空的讨论，我们对“变异的吃豆人（单片机版）”小游戏有如下规则制定（说明书不容易读懂，建议看视频演示或者直接尝试）：

在一个神秘的迷宫中，住着一群拥有特殊能力的小生物，他们的身体由黄色的圆形和三条黑线组成，就像豆子一样。他们被称为“吃豆人”，因为他们的主要任务就是寻找和吃掉所有的豆子。

吃豆人生活在一个充满危险的世界里，他们需要不断地躲避各种障碍和敌人的袭击，才能顺利地完成任务。但是，他们拥有非凡的速度和机智，可以通过不同的技巧和战术来应对各种挑战。每当他吃掉一个豆子，就会增加一点能量和力量，让他更加强大。

1. 程序初始化完成后，按下按钮 A 可以开始游戏，触摸金属 Logo 可以更改游戏难度，游戏难度从低到高为 1-5，默认为 1，直接按下 A 开始游戏的难度为 1；

2. 通过改变**单片机倾斜角度**，控制“吃豆人”的位置；

3. 游戏过程中若按下**按钮** B，游戏强制终止；

4. 每**吃掉一个“目标点”**，也即“吃豆人”与“目标点”位置重合，得分将+1；

5. 初始时，单片机上将显示 **3 个“目标点”**，每吃掉 2 个“目标点”后，单片机显示面板上将再次生成 2 个“目标点”，使显示面板上“目标点”总数再次变为 3 个；

6. **得分达到 8 分后**，显示面板上将出现“Killer（地雷）”，且“Killer”数量随得分增加而增加；

7. 每间隔一定时间，**“Killer”位置**将改变，游戏难度越高，Killer 移动速度越快；

8. 若**踩到“地雷”**，也即“吃豆人”与“地雷”位置重合，则游戏结束，“U r dead”，单片机显示面板将显示玩家得分。若想再次游戏，按下按钮 A。

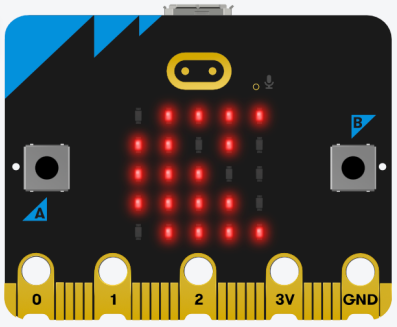
9. 若**得分超过 30**，则游戏成功，“U win”，单片机面板将显示玩家得分。若想再次游戏，按下按钮 A。

10. 由于单片机显示面板只能显示红色，我们用亮度区分“吃豆人”，“目标点”和“地雷”。其中**“吃豆人”最亮，“目标点”次亮，“地雷”最暗**。

二、设计方案和硬件连接

我们考虑并调用了 MicroBit 中的加速度传感器、扬声器、LED 点阵模块，并利用了其自带的按钮与 Logo 触摸板。通过侦测 X 与 Y 方向的加速度分量，来控制 Player 的移动，并将 Player、Killer、Goal 以不同亮度在 LED 上显示出来。对于 Killer 而言，我们设计了 Killer 追逐 Player 的相关算法，并为该算法加入了随机函数，使得每一个 Killer 的每次追逐 Player 的过程是相对随机的，从而使玩家难以找到 Killer 移动的规律。

我们的初版方案并未加入移动的 Killer（因此最初把 Killer 称为 Bomb）。后续的不断完善中，我们为该程序加入了 Killer 的追逐算法，为游戏带来更多乐趣（也导致了难度上升，可能难以通关），同时也加入了难度的选择界面，不断完善了游戏中可能遇到的如中途退出的情况，以求提供更完整的游戏体验。



三、实现方案及代码分析

我们将“吃豆人”设置为 player，“目标点”设置为 goal，“地雷”设置为 bomb。

第一重 While 循环用于检测 button_a 是否被按下，若没有按下，始终显示箭头提示图案；若按下，开始第二重 While 循环。

第二重 While 循环首先调用 movecontrol()，每次循环都基于加速度计探测到的数值重置 player 的位置。再利用条件判断 if 语句，分别调用函数。

一局游戏结束后，退出第二重 While 循环，重新进入第一重 While 循环。

关于实现过程和更加详细的代码分析可以参看下文代码中的注释，我们相信已然比较明晰。

```
# Imports go at the top
from microbit import *
import random, music, speech, time

start = False
Exit =True
points = 0 #初始得分 0
panel = []
# 由于单片机只能显示红色，可以用亮度区分目标点，player 点和地雷
# 设定 player 点亮度 9，目标点亮度 6，地雷亮度 3
pl = 9 #player 亮度 playerlight
gl = 6 #goal 亮度 goallight
bl = 3 #bomb 亮度 bomblight
interval = [4000, 3000, 2000, 1000, 500]
level = 0

#单片机上显示所有点的坐标
lst = [[0,0],[0,1],[0,2],[0,3],[0,4],[1,0],[1,1],[1,2],[1,3],
      [1,4],[2,0],[2,1],[2,2],[2,3],[2,4],[3,0],[3,1],[3,2],
```

```
      [3,3],[3,4],[4,0],[4,1],[4,2],[4,3],[4,4]]
num =
['1234','1324','1342','1423','1432','1243','2134','2143','2314','2341','2413','2431',

'3124','3142','3214','3241','3412','3421','4123','4132','4213','4231','4312','4321']

player = [2,2]
goalset = []
bombset = []
bombTime = time.ticks_ms()

#控制 player 随加速度变化移动位置
def movecontrol():
    global player
    '''每次刷新 player 点位置，先将上一轮的 player 亮度设置为 0，再重新设置新一轮的 player 亮度'''
    display.set_pixel(player[0],player[1],0)
    x = (accelerometer.get_x()+1000)//400
    y = (accelerometer.get_y()+1000)//400
    if x > 4:
        x = 4
    if y > 4:
        y = 4
    if x < 0:
        x = 0
    if y < 0:
        y = 0
    player = [x, y]
    display.set_pixel(x, y, pl)

#随机生成一个目标点并将目标点放进 goalset
def setgoal():
    global panel
    goalindex = random.randint(0, 24)
    currentgoal = lst[goalindex]
    if (currentgoal not in goalset and currentgoal not in bombset
        and currentgoal != player):
        '''保证新加入的目标点和 player，地雷不重合'''
        display.set_pixel(currentgoal[0], currentgoal[1], gl)
        goalset.append(currentgoal)
        panel[currentgoal[0]+1][currentgoal[1]+1] = 1
    else:
        setgoal()

```

```
#随机生成一个地雷并将地雷放进 bombset
def setbomb():
    global bombset
    bomb = random.randint(0,24)
    while ((lst[bomb] in goalset) or (lst[bomb] in bombset) or
    (lst[bomb] == player)):
        '''保证新加入的地雷与先前的 player，目标点都不重合'''
        bomb = random.randint(0,24)
    bombset.append(lst[bomb])
    display.set_pixel(lst[bomb][0], lst[bomb][1], bl)

#地雷（怪物）追逐玩家
def bombmove():
    global bombTime, bombset, panel, num, interval, level
    currentTime = time.ticks_ms() #记录时间
    dir = [[1,0],[-1,0],[0,1],[0,-1]] #移动放下：上下左右
    if time.ticks_diff(currentTime, bombTime) >= interval[level]: #判断时间间隔
        for i in range(len(bombset)):
            order = random.choice(num) #随机设置移动顺序，处理 bomb 与 goal 重合的情况
            bomb = bombset[i] #选取一个 bomb
            display.set_pixel(bomb[0], bomb[1], 0) #将上一轮的该 bomb 的亮度调 0
            XorY = random.randint(0,1) #XorY=0 在 x 方向移动；XorY=1 在 y 方向移动
            xDst = player[0] - bomb[0] #判断与 player 在 x 方向的距离
            yDst = player[1] - bomb[1] #判断与 player 在 y 方向的距离
            if xDst == 0: #如果已经和 player 的 x 坐标值相同，就在 y 方向移动
                XorY = 1
            elif yDst == 0: #如果已经和 player 的 y 坐标值相同，就在 x 方向移动
                XorY = 0
            if XorY == 0 and xDst > 0: #在 x 方向移动
                if [bomb[0]+1, bomb[1]] in goalset: #如果移动后的位置和 goal 重合
                    for d in order: #order 是随机生成的移动顺序
                        d = int(d) - 1
                        if panel[bomb[0]+dir[d][0]+1][bomb[1]+dir[d][1]+1] == 0:
                            bomb = [bomb[0]+dir[d][0],bomb[1]+dir[d][1]]
                            break
                else:
                    bomb[0] += 1
            elif XorY == 0 and xDst < 0: #类比
                if [bomb[0]-1, bomb[1]] in goalset:
                    for d in order:
                        d = int(d) - 1
                        if panel[bomb[0]+dir[d][0]+1][bomb[1]+dir[d][1]+1] == 0:
                            bomb = [bomb[0]+dir[d][0],bomb[1]+dir[d][1]]
                            break
```

```
                else:
                    bomb[0] -= 1
            elif XorY == 1 and yDst > 0: #类比
                if [bomb[0], bomb[1]+1] in goalset:
                    for d in order:
                        d = int(d) - 1
                        if panel[bomb[0]+dir[d][0]+1][bomb[1]+dir[d][1]+1] == 0:
                            bomb = [bomb[0]+dir[d][0],bomb[1]+dir[d][1]]
                            break
                else:
                    bomb[1] += 1
            elif XorY == 1 and yDst < 0: #类比
                if [bomb[0], bomb[1]-1] in goalset:
                    for d in order:
                        d = int(d) - 1
                        if panel[bomb[0]+dir[d][0]+1][bomb[1]+dir[d][1]+1] == 0:
                            bomb = [bomb[0]+dir[d][0],bomb[1]+dir[d][1]]
                            break
                else:
                    bomb[1] -= 1
            print(bomb)
            display.set_pixel(bomb[0], bomb[1], bl)
            bombset[i] = bomb
            bombTime = time.ticks_ms()

#初始化设置
def initial():
    global goalset, panel
    panel = [[1,1,1,1,1,1,1],
        [1,0,0,0,0,0,1],
        [1,0,0,0,0,0,1],
        [1,0,0,0,0,0,1],
        [1,0,0,0,0,0,1],
        [1,0,0,0,0,0,1],
        [1,1,1,1,1,1,1]
        ]
    display.set_pixel(player[0], player[1], pl)
    i = 0
    while i < 3:
        '''初始时随机生成三个目标点'''
        index = random.randint(0, 24)
        if lst[index] not in goalset and lst[index] != player:
            '''保证三个目标点各不相同而且和 player 不重合'''
            goalset.append(lst[index])
```

```
            panel[lst[index][0]+1][lst[index][1]+1] = 1
            display.set_pixel(lst[index][0],lst[index][1],gl)
            i += 1

#当 player 与某一个目标点重合调用 eat()
def eat():
    global goalset, player, points
    music.play(['C5:1']) #声音提示吃到目标点
    #print(player) #打印吃掉的点，在调试代码时帮助很大
    goalset.remove(player)
    panel[player[0]+1][player[1]+1] = 0
    points += 1

#当 player 与某一个地雷重合调用 dead()
def dead():
    music.play(['D4:1']) #声音提示踩到地雷
    '''结束流程：显示骷髅头，speech 嘲讽，滚动 U r dead，显示得分'''
    display.show(Image.SKULL)
    sleep(400)
    speech.say('ha, ha, You are Dead')
    display.scroll('U r dead', delay=80)
    display.scroll('Got')
    display.scroll(points)
    sleep(2000)
    display.clear()

#当得分超过 30，本轮游戏胜利，调用 win()
def win():
    music.play(music.PRELUDE, wait=False) #在悦耳的音乐中迎接胜利吧
    '''结束流程：显示笑脸，滚动 U win，显示得分'''
    display.show(Image.HAPPY)
    sleep(2000)
    display.scroll('U win', delay=80)
    display.scroll('Got')
    display.scroll(points)
    sleep(2000)
    display.clear()

#按 pin_logo 选择难度，调整 bomb 移动的时间间隔
#难度默认为最低=1，每按一次难度+1
def levelAdjust():
    global interval, level
    if pin_logo.is_touched():
        music.play(['E4:1'])
```

```
        level += 1
        if level > 4:
            level = 0
        display.show(level+1)

music.play(music.RINGTONE, wait=False) #欢迎音乐
display.show(Image.PACMAN)
sleep(500)
while Exit:
    levelAdjust()
    if button_a.was_pressed():
        movecontrol()
        display.clear()
        bombset = []
        goalset = []
        points = 0
        initial()
        start = True
        Exit = False
        #setbomb() #此行供测试使用
    while start:
        movecontrol()
        bombmove()
        if button_b.was_pressed():
            #按 button_b 可以强制退出，显示已有得分
            display.scroll('Got')
            display.scroll(points)
            sleep(500)
            goalset = []
            bombset = []
            start = False
            Exit = True
            display.clear()
            display.show(Image.PACMAN)
            break
        if player in goalset: #吃到目标点
            eat()
            print(points)
        if player in bombset: #踩到地雷
            start = False
            dead()
            Exit = True
            display.show(Image.PACMAN)
            break
```

```
237.         if len(goalset) == 1: #如果吃掉了两个目标点，就更新 goalset
238.             setgoal()
239.             setgoal()
240.         if points//8 - len(bombset): #每多吃掉 8 个目标点，增设一个地雷
241.             setbomb()
242.             bombTime = time.ticks_ms()
243.         if points == 30: #points 达到 30，成功通关
244.             win()
245.             Exit = True
246.             start = False
247.             display.show(Image.PACMAN)
248.             break
249.
```

四、后续工作展望

由于单片机上显示面板的大小有限，“地雷”数量不能无限增多，得分达到 30 即可通关。而且单片机上只能显示红色的 LED 灯，不利于区分“吃豆人”、“目标点”和“地雷”。后续希望能将单片机和其他显示工具结合起来，在更大且呈现颜色更多的显示器上呈现游戏，提高游戏难度上限，带给玩家更好体验和更多刺激。

五、小组分工合作

源代码由 Jzl 和 Lsy 共同完成。Poster 和运行和介绍视频由 Jzl 制作，实习报告主要由 Lsy 主笔。但实际上，整个小游戏是两个人共同构思、共同创作的结果。我们共同合作，互相帮忙 debug，才能将这款变异的吃豆人（单片机版）呈现给大家。