流星雨

田春迅，姜鑫

摘要：本创意作品通过模块化设计，建立实时系统并计时计次，以随机生成、寻路算法、样式设计实现了随机下落的流星雨，玩家能与其交互，进行躲避和道具使用等操作，最终结算分数。并在此框架之上拓展了三种道具及相应的动画、音乐，最终实现了一款内容完备、过程流畅、容易上手的下落式躲避游戏。

一、 选题及创意介绍

本组以设计一款趣味小游戏为选题，从东方绀珠传中汲取灵感，最终在 micro bit 上实现了一款纵向躲避小游戏。玩家可以操纵一架飞机，躲避屏幕中不停滑过的流星，获取更高的分数。为了平衡游戏难度以及提高玩法趣味性，我们额外设计了三款道具，可以在游戏开始前进行选择，供玩家在游戏过程中灵活使用，化解困境。

二、 设计方案

1. 整体目标：

玩家在选择特定道具后，通过左右按键操纵飞机以及使用道具躲避上方生成的流星雨，失败后结算最终分数。

2. 功能模块设计：

玩家移动机制：通过左右按键控制移动。

流星雨生成：随机生成，随难度增加提高生成密度，同时保证至少有一条出路，能随着时间位置不断下移。

计分机制：游玩时间越长，游戏难度增加，流星雨下落越快，最终得分越高。

道具机制：开局时可以选一个道具，如：被逼到边缘时可以从一边边缘窜到另一边边缘的能力；可以使用一定次数的时缓能力；一键清除所有流星的能力。

三、 实现方案及代码分析

1. 实现方案：

（1） 基础框架：

a. while True 循环，以一个循环（约 30ms）为最小原子时间（tick）运行
b. 5*5 二维列表 img 用不同亮度代表流星雨和飞机的位置并实时更新，用于判定和显示

（2） 基础玩法：

a. 在 while True 循环中不断计次代表时间和得分，达到一定次数后流星下进 1 位，更新 img；每个 tick 内检测玩家有无按动按键，更新 img。img 一旦更新立即检测 plane 与 star 是否重合。如果死亡，则终止循环，结束游戏。

b. 用 list 储存流星位置和长度。流星步进时修改位置，概率生成新流星，调用 randint 随机生成不同位置、长度，通过 dfs 寻路算法检测是否有解后更新 img

（3） **道具实现：**

a. 用列表储存三种道具，用 index 代表当前选择。在循环中检测左右键并修改 index，检测到中心键后确认，更新道具可使用情况，中止循环。
b. Jump/一边边缘窜到另一边：移动机制细化
c. Bomb/一键清除所有流星：在 while True 中检测中心键有无按动和道具可使用情况，如符合则清空流星 list
d. Timedown/时缓能力：在计次步进中增加 timedowning 状态，该状态下流星步进需要的计次数增加，timedowning 随时间流逝减少直至为 0 恢复为普通状态。在 while True 中检测中心键有无按动和道具可使用情况，如符合，timedowning 增加一定量 tick。

2. **代码分析：**

（1） **库引入和初始化**

（2） **道具动画与选择：**

a. animations 储存道具动画，delays 储存播放间隔，switch 函数负责更新道具 index 并播放相应的动画，注意将 wait 设置为 False 防止阻塞。
b. 进入道具选择后根据按键情况 switch（1）或（-1），如果中心键被按动，根据 index 更新相应道具状态。EGG 为彩蛋系统，当检测到摇晃时出现问号，中心键被按动后随机给予 all，任一道具或 none。道具选择结束后倒数，播放音乐。

（3） **死亡函数 `die()`：播放失败动画和音乐，显示分数**

（4） **寻路函数 `searchWay(img, plane)`：**

a. 进行一次回收防止内存溢出
b. 用 dfs 储存待搜索的用 tuple 表示的位置，将 img 拷贝 img2，current 表示当前搜索节点，自下往上左右搜索，并将搜索过的节点在 img2 中标记。如果搜索到顶端就返回 True 如果不能就返回 False。

（5） **星图生成 `summon_stars(stars)`：**

将 list 中的流星在 img 上生成，在处于 timedown 状态时会显示低亮度，并在临近结束时闪烁。

（6） **步进函数 `step(difficulty)`：**

星星参数：位置坐标（loc，y），长度 lenth，以 tuple 形式储存在 stars 中。

a. 将所有星星的 y 坐标减一，如果离开屏幕，则删除。
b. 根据 difficulty 进行一次随机判定是否生成新星星，如判定生成，进行 5 次尝试，随机生成 loc 和 lenth，生成后检查是否与已有的星星重合，如重合，则再次尝试；如成功，则将 summon 更新为 true，并停止召唤。
c. 清空 img 并调用 summon_stars(stars)，并判定玩家是否失败，如失败，将 dead 更新为 True。
d. 如果 b 步成功召唤，c 步未死亡，则调用 searchWay(img, plane)，如果寻路失败，从 stars 弹出 b 步召唤的流星并重新清空 img，调用 summon_stars(stars)，在 img 中放置 plane，显示图像，步进完成。

（7） **进入游戏：**

a. 计次加一，得分加一
b. 如果调用 bomb 且 bomb 可使用，清空 stars 列表，清空屏幕，播放动画音效
c. 如果调用 timedown 且 timedown 可使用，timedowning 增加，并立刻调用 summon_stars(stars)，显示 timedown 状态的特殊亮度
d. 计次部分，如果处在 timedown 状态，实时更新屏幕用于显示闪烁效果，降低步进频率；如果处在正常状态则无需实时更新屏幕，计次调用 step(difficulty)即可
e. 移动机制，按钮被按动时，如果不在角落或者 jump 可使用，则进行相应移动，并判定玩家是否失败。
f. 如果失败，调用 die()，游戏结束

四、 后续工作展望

优化流星生成算法，避免由于概率原因而造成难度突跃。同时还可以进行扩展，设计不同下落速度的流星，以及横向一次生成多个流星，丰富游玩体验。

优化积分系统，增加更多得分项目，比如惊险地躲过可以获得更高分数。

连接硬件，比如像素更高的显示屏以实现更丰富的玩法、更精美的效果与动画。

五、 小组分工合作

田春迅：确立框架，完成代码主体，优化性能使程序稳定运行；海报设计。
姜鑫：测试反馈，动图效果设计，游戏平衡性调整；撰写实习报告；拍摄视频