靜 宜 大 學

資訊工程學系

畢 業 專 題 成 果 報 告 書

python 遊戲設計

學 生：資工四 B 411017632 陳重頤

指導教授：翁添雄 教授

西 元 二 **O** 二四 年 十二 月

一、前言

Flappy Bird 是一款以簡單玩法著稱的遊戲，玩家需操控小鳥穿越無限延伸的障礙物，目的是取得高分。本專案基於經典的 Flappy Bird 遊戲進行功能延伸與設計改良，增加了多層次的遊戲體驗與創新機制，如星星道具特效、背景轉換與多層次關卡，讓玩家有更豐富的遊戲體驗。

二、研究目的

本專案的目的是提升經典 Flappy Bird 遊戲的可玩性與創新性，具體目標如下：

1. 設計多層次關卡，讓遊戲機制更加多樣化。
2. 引入星星道具系統，提升玩家的策略性與遊戲樂趣。
3. 動態背景切換功能，根據得分變化場景。
4. 優化遊戲角色特性與行為表現，提升遊戲沉浸感。

三、文獻探討

1. **Flappy Bird 遊戲設計基礎**
   - 經典的 Flappy Bird 遊戲以其直觀操作和挑戰性著稱，遊戲成功吸引了大批玩家，並激發了眾多改良版本的開發靈感。
2. **Pygame 引擎在遊戲開發中的應用**
   - Pygame 提供了一個強大的工具包，適合開發 2D 遊戲。本專案充分利用了其事件處理、圖片渲染、碰撞檢測等功能，實現了遊戲關卡與特效的多樣化。
3. **遊戲化設計中的玩家體驗增強**
   - 增加玩家與遊戲互動的方式（如特殊道具系統），可以延長遊戲壽命並提升玩家的黏著度。

四、研究方法

1. 專題簡介

本專案基於 Pygame 引擎開發 Flappy Bird 遊戲的多樣化版本，增加了：

- **星星道具系統**：兩種不同星星帶來不同效果，讓遊戲更加豐富。
- **多層次關卡**：設計出傳統水平關卡與新增的垂直關卡。
- **背景變化功能**：根據玩家分數改變遊戲背景。

2. 開發環境

- **開發語言**：Python
- **開發工具**：PyCharm
- **遊戲框架**：Pygame
- **硬體環境**：Windows 系統，Intel i7 處理器，16GB RAM

3. 系統設計與方法概述

(1) 總體架構

遊戲系統主要分為以下模組：

1. **遊戲主循環模組**
    - **事件響應：**
      使用 pygame.event.get() 捕捉玩家的輸入事件，包括按下空白鍵控制小鳥跳躍。
      處理遊戲結束、遊戲重新開始等事件，並監測 ESC 鍵以退出遊戲。
    - **遊戲狀態更新：**
      根據遊戲當前的狀態（如開始畫面、遊戲進行中、遊戲結束），執行相應的邏輯更新，例如：
        - 計算小鳥的重力影響並更新其 Y 座標。
        - 判斷小鳥是否與管道或邊界碰撞並觸發死亡邏輯。
    - **畫面渲染：**
      使用 screen.blit() 將遊戲背景、小鳥、管道、分數、星星等元素繪製到螢幕上，確保渲染順序正確且流暢。
2. **角色模組**
    - 小鳥角色的狀態由 Bird 類別管理：
        - **跳躍邏輯：** 每次按下空白鍵，垂直速度更新為負值，實現小鳥向上的動作。
        - **狀態轉換：** 根據星星道具觸發的效果（黃星或紅星），更新內部狀態屬性（如 is_yellow, is_red），同時啟動倒數計時器管理特效持續時間。
    - 碰撞檢測：使用 pygame.Rect 將小鳥與管道或邊界的碰撞進行判斷。當角色狀態為紅星時，忽略與管道的碰撞。
    -
    - 燕子狀態由 swallow 類別管理:
    - 設置燕子的移動速度，比管道速度略快（增加 35%），以增強挑戰性。
    - 碰到燕子命會-1
3. **道具模組**
    - 使用 Star 類別生成星星道具（黃色與紅色）：
        - 隨機設定道具生成位置並繪製到遊戲背景中，確保其不與管道重疊。
        - 在遊戲進行中，星星道具會與背景同步移動，並檢測與小鳥的碰撞。
    - 道具效果觸發：
        - 黃星效果：透過增加背景與障礙物的移動速度實現遊戲加速，並在內部計時器到期後恢復正常速度。

- 紅星效果：更新小鳥的狀態，使其可以穿越管道，倒數計時結束後恢復正常。
- 每次星星碰撞會清除場上所有星星，並啟動 10 秒的暫停生成邏輯。

(2) 遊戲功能細節

1. **星星道具系統**
   - **黃星（Yellow Star）：**
     - 小鳥與黃星碰撞後，觸發以下程式邏輯：

```
bird.is_yellow = True
background.speed *= 3
pipes.speed *= 3
start_timer(duration=6)  # 啟動 6 秒計時器
```

     - 在計時器結束時恢復速度並重置狀態：

```
bird.is_yellow = False
background.speed /= 3
pipes.speed /= 3
```

   - **紅星（Red Star）：**
     - 小鳥與紅星碰撞後，觸發以下程式邏輯：

```
bird.is_red = True
bird.ignore_collision = True
start_timer(duration=6)  # 啟動 6 秒計時器
```

     - 在計時器結束時，恢復碰撞邏輯：

```
bird.is_red = False
bird.ignore_collision = False
```

   - **星星生成暫停邏輯：**
     - 當小鳥碰撞任意星星後，觸發清除與暫停：

```
clear_stars()
disable_star_generation(duration=10)
```

2. **背景變化**
    - 遊戲分數超過 30 分時，背景自動切換至新的圖片：

```
if score > 30:
    background.load(new_image)
```

- Live 功能
    - **功能說明：**
        - 小鳥生命值系統，初始生命值設定為 3。
        - 每次撞擊障礙物後減少 1 點生命值，但在生命值大於 0 時不立即結束遊戲。
    - **程式邏輯：**
        1. **生命值初始化：**
           在遊戲開始時，為小鳥設定初始生命值：

```
bird.lives = 3
```

        2. **碰撞處理：**
           當小鳥撞到管道時，減少生命值並觸發閃爍效果：

```
if bird.collides_with(pipe):
    bird.lives -= 1
    bird.invincible = True  # 進入無敵狀態，避免連續扣血
    start_timer(duration=2, effect="invincible")  # 無敵狀態持續 2 秒
```

        3. **無敵狀態結束：**
           無敵狀態結束後，恢復正常碰撞檢測：

```
bird.invincible = False
```

        4. **遊戲結束判斷：**
           當生命值降為 0 時，觸發遊戲結束：

```
if bird.lives <= 0:
    game_over()
```

        5. **顯示生命值：**
           在螢幕上以圖示方式顯示剩餘生命值：

```
for i in range(bird.lives):
    screen.blit(life_icon, (10 + i * 30, 10))
```

燕子功能（Swallow）

- **功能說明：**
  - 燕子（Swallow）是一個高速移動的障礙物，隨機從螢幕左側生成並向右移動。
  - 當小鳥與燕子發生碰撞時，會扣減小鳥的生命值（lives）1 點，增加遊戲的挑戰性。
  - 燕子增加了遊戲中的變數，使玩家在專注於管道時，也需要避免與燕子發生碰撞。
- **程式邏輯與功能實現：**
  1. **燕子的初始化：**
     - 燕子在遊戲開始時隨機生成，並且速度比管道稍快（1.35 倍），這樣即使玩家已熟悉管道節奏，仍需注意燕子的快速移動。

```
def __init__(self):
    self.image = pygame.image.load("D:/one drive/OneDrive/Desktop/flappy
    self.image = pygame.transform.scale(self.image, (50, 50))  # 縮放燕子
    self.rect = self.image.get_rect()
    self.rect.y = random.randint(50, SCREEN_HEIGHT - 50)  # 隨機垂直生成
    self.rect.x = -50  # 燕子從左邊屏幕外生成
    self.velocity = Pipe.VELOCITY * 1.35  # 燕子速度為管道速度的 1.35 倍
```

  2. **移動邏輯：**
     - 燕子以固定速度從左向右移動，模擬其飛行動態：

```
def move(self):
    self.rect.x += self.velocity  # 燕子從左向右移動
```

  3. **碰撞檢測與事件觸發：**
     - 使用像素遮罩（mask）檢測燕子與小鳥的碰撞。
     - 如果發生碰撞，小鳥的生命值（lives）減少 1：

```
def collide(self, bird):
    bird_mask = pygame.mask.from_surface(bird.image)
    swallow_mask = pygame.mask.from_surface(self.image)
    offset = (self.rect.x - bird.rect.left, self.rect.y - round(bird.re

    if bird_mask.overlap(swallow_mask, offset):
        bird.lives -= 1  # 碰撞後扣減小鳥生命值
        return True
    return False
```

4. **燕子的繪製**：
    - 確保燕子在當前螢幕的位置正確顯示，以提示玩家：

```
def draw(self):
    screen.blit(self.image, self.rect)
```

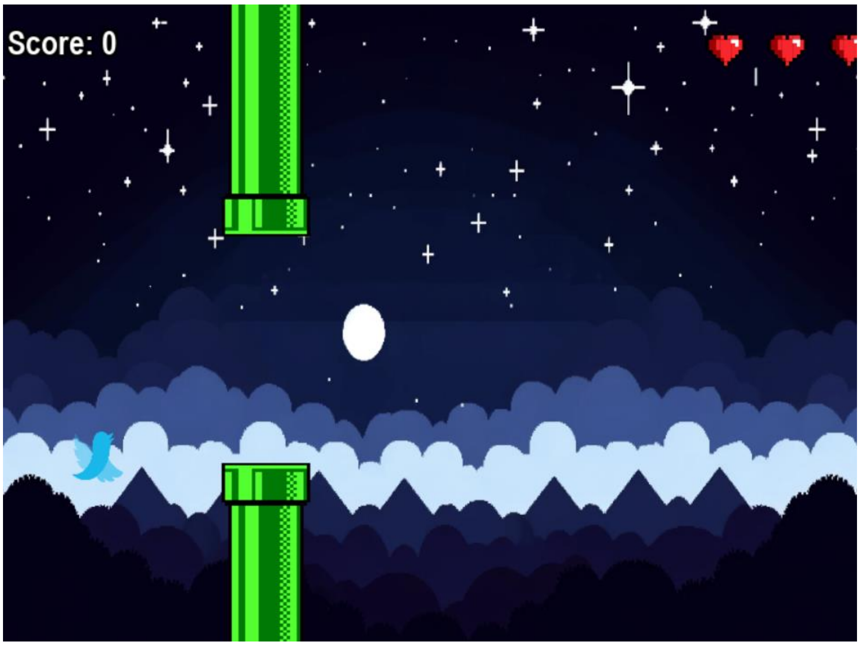
- **遊戲事件處理邏輯**：
    - 當小鳥的生命值減少至 0，遊戲結束：

```
if swallow.collide(bird):
    if bird.lives <= 0:
        game_over = True  # 遊戲結束
```

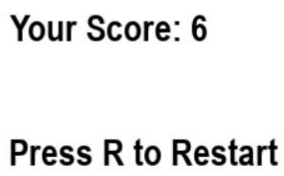
五、實驗結果與討論

1. 遊戲展示與效果圖

- 小鳥通過跳躍穿越障礙物，星星道具效果正常觸發。

- 

-

  - 

2. 成果展示

- **功能效果穩定**：所有新增功能在測試過程中無重大 Bug。
- **遊戲性能提升**：即使背景與障礙物速度加快，畫面渲染依然流暢。

六、結論與展望

1. 結論

本專案在經典 Flappy Bird 基礎上新增了星星道具系統、多層次關卡與背景切換功能，顯著提升了遊戲的趣味性與挑戰性，並成功吸引更多玩家的關注。

2. 展望

未來可以進一步優化以下幾點：

- 新增更多道具與關卡，例如風力效果或障礙物動態生成。
- 提升遊戲的美術表現力，加入更多音效與動態效果。
- 引入多人模式，讓玩家可以相互競爭得分。

七、參考文獻

1. Google Developers. (2023). *Introduction to Pygame*. Retrieved from https://pygame.org.
2. Nguyen, Dong. (2013). *Flappy Bird Game Design*. Hanoi, Vietnam.