TRƯỜNG ĐẠI HỌC CÔNG NGHỆ
ĐẠI HỌC QUỐC GIA HÀ NỘI

Mô phỏng hiện tượng bày đàn

Giảng viên: Nguyễn Hồng Thịnh

Nhóm thực hiện: Nhóm 2

21021600 - Lương Quốc Khánh

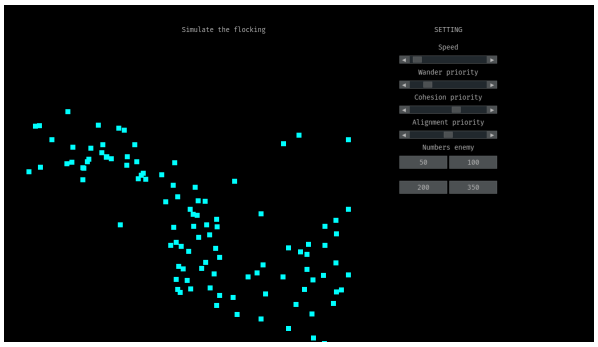
21020698 - Nguyễn Đức Minh

21020688 - Đỗ Huy

21021587 - Phạm Minh Hiếu

1 Giới thiệu

Lập trình Python sử dụng thư viện pygame mô phỏng hiện tượng bầy đàn. Thông qua chương trình ta có thể tìm hiểu, điều chỉnh về những tác động làm ảnh hưởng đến tập cá thể. Trong ứng dụng này, chúng tôi có các cá thể là những hình ô vuông màu xanh được mô phỏng bên trái màn hình, bên phải là màn hình điều chỉnh độ lớn của các tác động cũng như tăng giảm số lượng cá thể biểu diễn và tốc độ di chuyển của chúng.



Hình 1: Màn hình làm việc.

Các tác động xử lý:

1. Sự ngẫu nhiên: cá thể di chuyển ngẫu nhiên, không theo quy luật nào.
2. Sự kết hợp: cá thể di chuyển về hướng của bầy đàn.
3. Sự giãn cách: cá thể di chuyển tránh va chạm với các cá thể khác.

Sự kết hợp các tác động giúp tạo nên chương trình của chúng tôi. Chúng tôi mong rằng nó sẽ hữu ích và khơi gợi sự hứng thú của bạn về chủ đề này.

2 Các tác động.

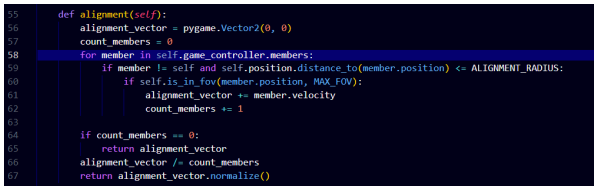
2.1 Tác động ngẫu nhiên.

Sự ngẫu nhiên có thể xảy ra bất cứ lúc nào và không có dự đoán trước theo một quy luật nào đó. Bằng cách tạo một hướng đi mới ngẫu nhiên kết hợp với hướng đi ban đầu rồi chuẩn hóa chúng tôi biểu diễn được được di chuyển ngẫu nhiên của cá thể.

```
def wander(self):
    deltaTime = 0.01
    self.wander_target += pygame.Vector2(random.uniform(-.5, .5) * WANDER_JITTER * deltaTime, random.uniform(-.5, .5) * WANDER_JITTER * deltaTime)
    self.wander_target = self.wander_target.normalize() * WANDER_RADIUS

    target_in_local_space = self.wander_target + pygame.Vector2(0, WANDER_DISTANCE)
    target_in_world_space = self.position + target_in_local_space.rotate(self.velocity.angle_to(pygame.Vector2(1, 0)))
    return target_in_world_space.normalize()
```

Hình 2: Xử lý dữ liệu trên python.



Hình 3: Xử lý dữ liệu trên python.

2.2 Tác động kết hợp.

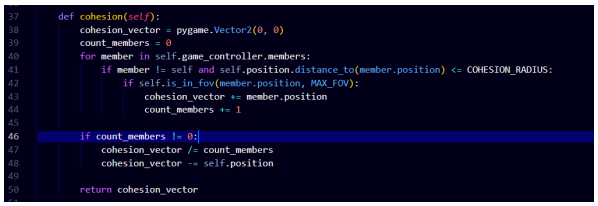
Bằng cách tổng hợp hướng di chuyển của cá thể đồng loại xung quanh và nằm trong hướng quan sát rồi chia cho số lượng cá thể rồi chuẩn hóa nó, chúng tôi tính được vector vận tốc cần di chuyển.

Trong đó:

- alignmentVector là vận tốc mới cần căn chỉnh.
- ALIGNMENT RADIUS là bán kính ảnh hưởng.
- countMembers là số lượng cá thể thỏa mãn.
- self.gameController.members là tập các cá thể.

2.3 Tác động giãn cách

Bằng cách tổng hợp vị trí của các cá thể nằm trong phạm vi xét sau đó lấy trung bình vị trí tổng hợp trừ đi vị trí hiện tại ta được vector hướng. Tiếp tục chuẩn hóa ta được kết quả cần tính.



Hình 4: Xử lý dữ liệu trên python.

Trong đó:

- cohesionVector là vận tốc mới cần căn chỉnh.
- COHESION RADIUS là bán kính ảnh hưởng.
- countMembers là số lượng cá thể thỏa mãn.
- self.gameController.members là tập các cá thể.

3 Xử lý UI

Thông qua thư viện pygameGui trong python chúng tôi thiết lập những thanh kéo (slider) để điều chỉnh chỉ số ảnh hưởng của các tác động với cá thể, nút bấm (button) để điều chỉnh số lượng cá thể trong mô phỏng.

```
title = ["Speed", "Wander priority", "Cohesion priority", "Alignment priority", "Numbers enemy"]
game_controller = GameController()
text_label = UILabel(relative_rect=pygame.Rect((300, 0), (300, 100)),
                     text="Simulate the flocking",
                     manager=ui_manager)
text_label = UILabel(relative_rect=pygame.Rect((800, 0), (200, 100)),
                     text="SETTING",
                     manager=ui_manager)
for i in range(0, 5):
    text_label = UILabel(relative_rect=pygame.Rect((800, 50*i + 75), (200, 20)),
                         text=title[i],
                         manager=ui_manager)
slider_0 = pygame_gui.elements.UIHorizontalSlider(relative_rect=pygame.Rect((800, 100), (200, 20)),
                                                  start_value=VALUE[1],
                                                  value_range=(0, 4),
                                                  manager=ui_manager)
slider_1 = pygame_gui.elements.UIHorizontalSlider(relative_rect=pygame.Rect((800, 150), (200, 20)),
                                                  start_value=VALUE[2],
                                                  value_range=(0.25, 4),
                                                  manager=ui_manager)
slider_2 = pygame_gui.elements.UIHorizontalSlider(relative_rect=pygame.Rect((800, 200), (200, 20)),
                                                  start_value=VALUE[3],
                                                  value_range=(0, 4),
                                                  manager=ui_manager)
slider_3 = pygame_gui.elements.UIHorizontalSlider(relative_rect=pygame.Rect((800, 250), (200, 20)),
                                                  start_value=VALUE[4],
                                                  value_range=(0, 4),
                                                  manager=ui_manager)
button = UIButton(relative_rect=pygame.Rect((800, 300), (100, 30)),
                  text="50",
                  manager=ui_manager)
button1 = UIButton(relative_rect=pygame.Rect((900, 300), (100, 30)),
                   text="100",
                   manager=ui_manager)
button2 = UIButton(relative_rect=pygame.Rect((800, 350), (100, 30)),
                   text="200",
                   manager=ui_manager)
button3 = UIButton(relative_rect=pygame.Rect((900, 350), (100, 30)),
                   text="350",
                   manager=ui_manager)
```

Hình 5: Cài đặt các nút bấm và thanh kéo.

```
    def handle_events(self):
        global NUMBER_OF_MEMBERS
        for event in pygame.event.get():
            if event.type == QUIT:
                pygame.quit()
                exit()
            elif event.type == pygame.USEREVENT:
                if event.user_type == pygame_gui.UI_HORIZONTAL_SLIDER_MOVED:
                    if event.ui_element == slider_1:
                        VALUE[2] = event.value
                    elif event.ui_element == slider_2:
                        VALUE[3] = event.value
                    elif event.ui_element == slider_3:
                        VALUE[4] = event.value
                    elif event.ui_element == slider_0:
                        VALUE[1] = (int)(event.value)
                elif event.user_type == pygame_gui.UI_BUTTON_PRESSED:
                    if event.ui_element == button:
                        NUMBER_OF_MEMBERS = 50
                        self.clear_members()
                        self.init_members()
                    elif event.ui_element == button1:
                        NUMBER_OF_MEMBERS = 100
                        self.clear_members()
                        self.init_members()
                    elif event.ui_element == button2:
                        NUMBER_OF_MEMBERS = 200
                        self.clear_members()
                        self.init_members()
                    elif event.ui_element == button3:
                        NUMBER_OF_MEMBERS = 350
                        self.clear_members()
                        self.init_members()

                ui_manager.process_events(event)
```

Hình 6: Xử lý logic.

4 Kết luận

Trong quá trình thực hiện mô phỏng hiện tượng bày đàn sử dụng thư viện pygame, chúng tôi đã thành công trong việc tạo ra một chương trình linh hoạt và sinh động, giúp hiểu rõ hơn về cách các cá thể trong bầy đàn tương tác với nhau. Bằng cách kết hợp các tác động như ngẫu nhiên, kết hợp và giãn cách, chúng tôi đã tạo ra một mô phỏng sống động về hành vi của các cá thể trong một bầy đàn.

Hình ảnh minh họa trong màn hình làm việc (Hình 1) đã thực sự thể hiện sự linh hoạt và tính chân thực của mô phỏng của chúng tôi. Việc điều chỉnh các tham số như bán kính ảnh hưởng và số lượng cá thể đã giúp chúng tôi quan sát được sự biến đổi đáng kể trong hành vi của bầy đàn.

Các tác động như ngẫu nhiên, kết hợp và giãn cách đã được hiện thực một cách chặt chẽ và thực tế, giúp chúng tôi đạt được mục tiêu mô phỏng một cách chân thực và hấp dẫn. Hy vọng rằng chương trình mô phỏng của chúng tôi sẽ là nguồn cảm hứng cho các nghiên cứu về hiện tượng bày đàn và sẽ thúc đẩy sự hiểu biết về tương tác giữa các cá thể trong tự nhiên.