Boids

Author: B1030400012 謝承翰

3大規則

- separation
- alignment
- cohension

程式實現

Bird物件屬性:

- (x, y)座標
- 飛行角度
- 速率
- 視野範圍

Bird物件方法:

- fly

飛行

```
def fly(self, canv, dt, boundaryx, boundaryy):
        self.vx = self.base_speed * math.cos(self.angle)
        self.vy = self.base_speed * math.sin(self.angle)
        self.x += self.vx * dt
        self.y += self.vy * dt
        self.x %= boundaryx
        self.y %= boundaryy
        canv.delete(self.name)
        self.draw(canv)
```

將3大規則套用在Bird物件上

separation 程式實現

```
def sep(this, others, minidist = 30):
        avg_x = 0
        avg_y = 0
        n = 0
        for other in others:
                if this.dist(other) < minidist:
                        avg_x += other.x
                        avg_y += other.y
                        n += 1
        if n == 0: return
        avg_x /= n
        avg_y /= n
        dd = math.atan2(avg_y - this.y, avg_x - this.x)
        d = this.angle - dd
        this.angle += 0.005 * d
```

平均位置

目標角度

改變自身飛行角度

alignment 程式實現

```
def alig(this, others):
        avg_ang = 0
        for other in others:
                avg_ang += other.angle
        avg_ang /= len(others)
        this.angle -= 0.07 * (this.angle - avg_ang)
```

平均飛行角度

改變自身飛行角度

cohension 程式實現

```
def cohen(this, others):
        avg_x = 0
        avg_y = 0
        for other in others:
                avg_x += other.x
                avg_y += other.y
        avg_x /= len(others)
        avg_y /= len(others)
        dd = math.atan2(avg_y - this.y, avg_x -this.x)
        d = this.angle - dd
        this.angle -= 0.015 * d
```

平均位置

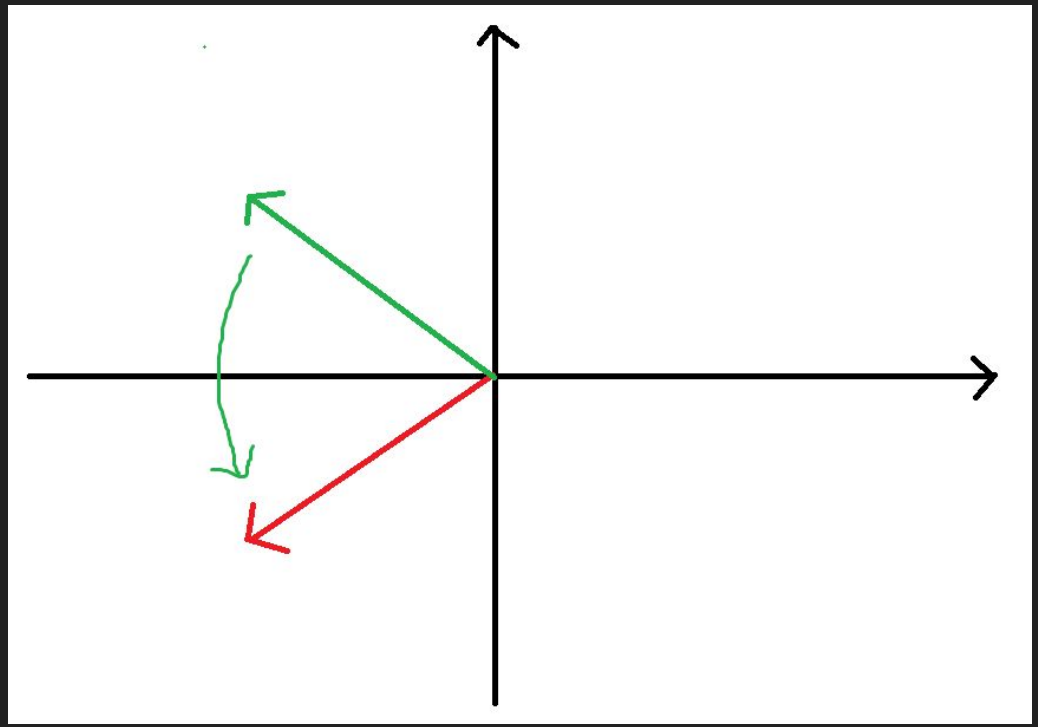
目標角度

改變自身飛行角度

最終效果

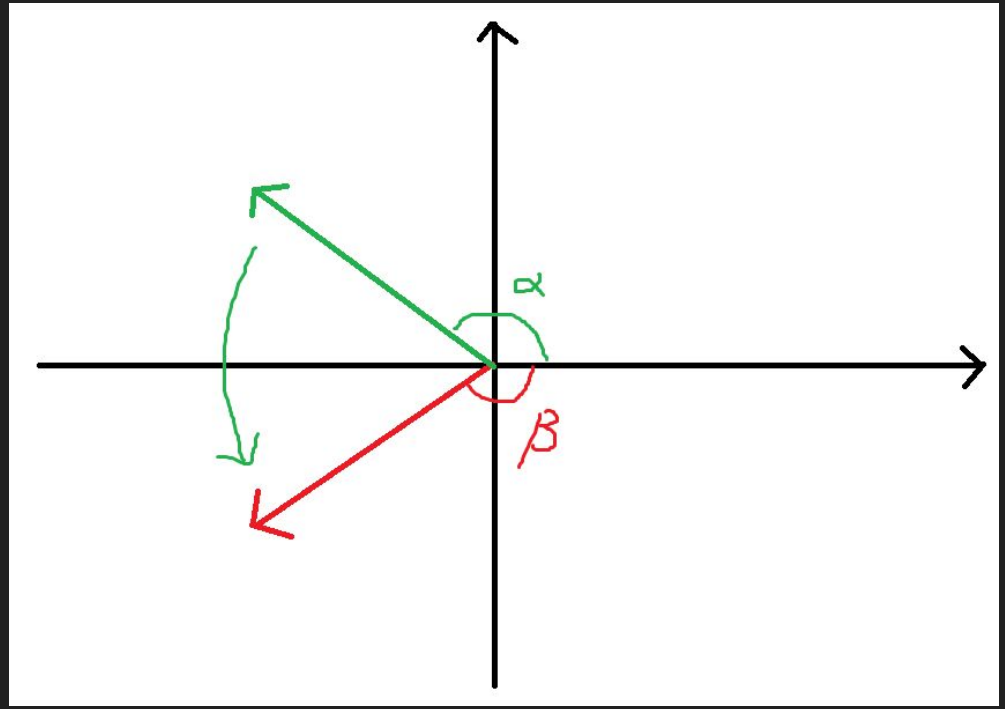
問題

考慮如下角度, 綠色為目前飛行角度, 紅色為目標角度



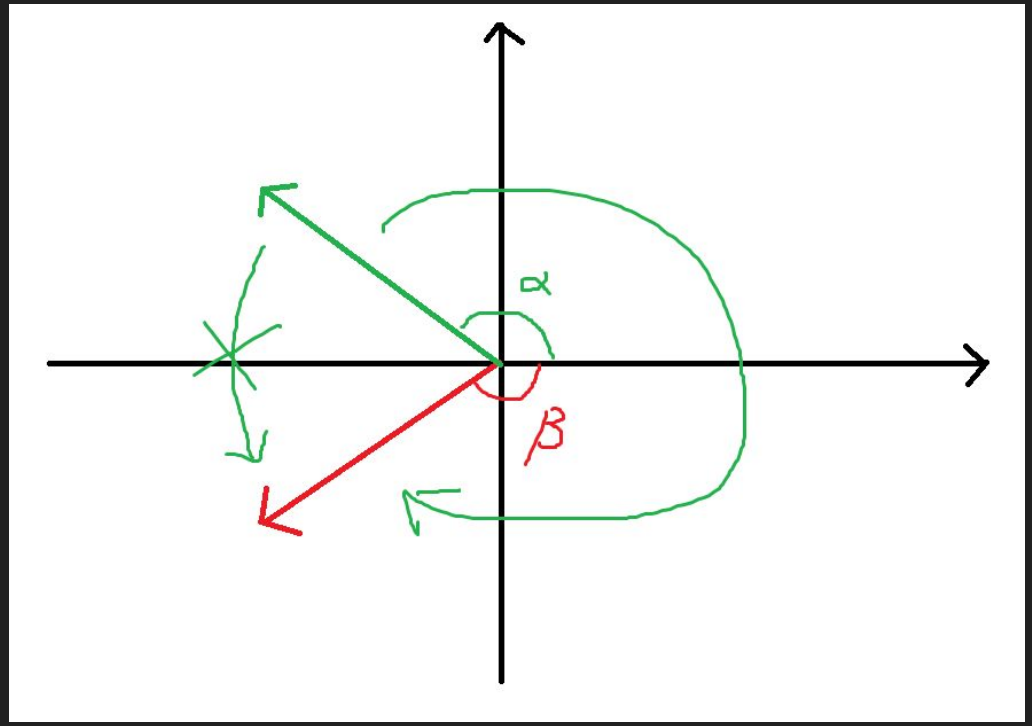
問題

atan2(y, x)的值域是(-π, π]，紅色向量的角度<0



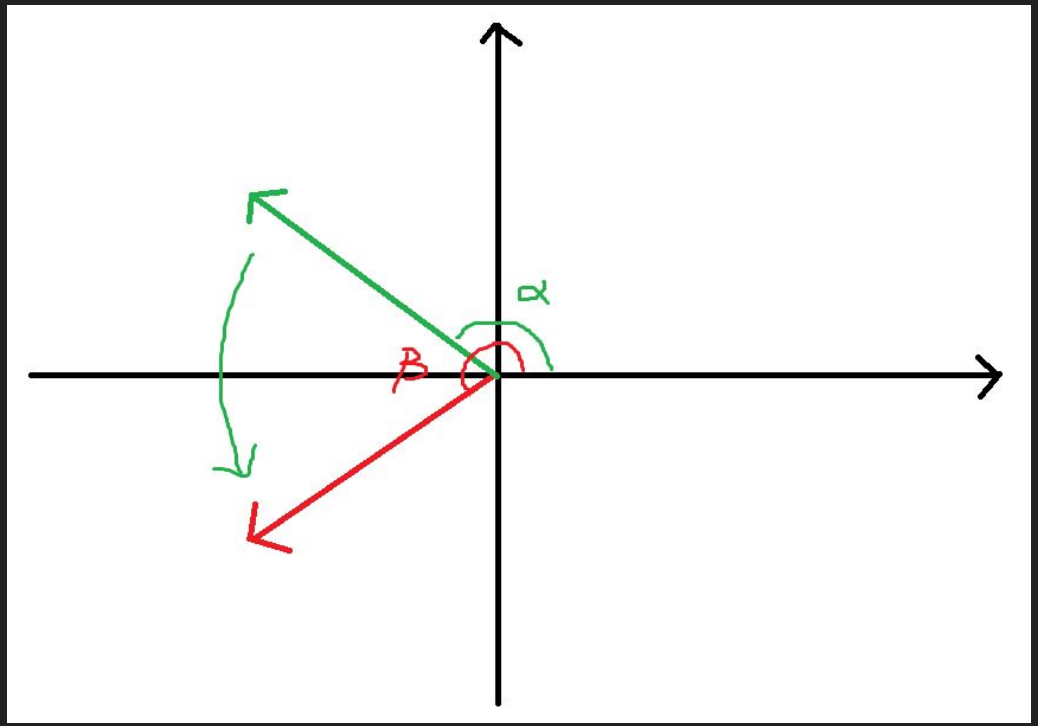
問題

α = α-(α-β), (α-β) > 0, 結果就是往非預期的方向轉。



how to solve it?

將atan2(y, x)的值域從(-π, π]映射到[0, 2π]。**α = α-(α-β), (α-β) < 0**



separation 程式實現 ver.2

```
def sep(this, others, minidist = 30):
        avg_x = 0
        avg_y = 0
        n = 0
        for other in others:
                if this.dist(other) < minidist:
                        avg_x += other.x
                        avg_y += other.y
                        n += 1
        if n == 0: return
        avg_x /= n
        avg_y /= n
        dd = math.atan2(avg_y - this.y, avg_x - this.x)
        if dd < 0: dd += 2*math.pi
        d = this.angle - dd
        this.angle += 0.005 * d
```

平均位置

目標角度

改變自身飛行角度

alignment 程式實現(沒變)

```
def alig(this, others):
        avg_ang = 0
        for other in others:
                avg_ang += other.angle
        avg_ang /= len(others)
        this.angle -= 0.07 * (this.angle - avg_ang)
```

平均飛行角度

改變自身飛行角度

cohension 程式實現 ver.2

```
def cohen(this, others):
        avg_x = 0
        avg_y = 0
        for other in others:
                avg_x += other.x
                avg_y += other.y
        avg_x /= len(others)
        avg_y /= len(others)
        dd = math.atan2(avg_y - this.y, avg_x -this.x)
        if dd < 0: dd += 2*math.pi
        d = this.angle - dd
        this.angle -= 0.015 * d
```

平均位置

目標角度

改變自身飛行角度

最終效果

程式實現

Bird物件屬性:

- (x, y)座標
- ~~飛行角度~~
- ~~速率~~
- 速度分量
- 最大速率、最小速率
- 視野範圍

Bird物件方法:

- fly

separation 程式實現

```
def sep(this, others, miniDist = 30):
        dx, dy = 0, 0
        for other in others:
                if this.dist(other) < miniDist:
                        dx += (this.x - other.x)
                        dy += (this.y - other.y)
        this.vx += dx * 0.08
        this.vy += dy * 0.08
```

累加其他Bird造成的改變

改變x分量速度

改變y分量速度

alignment 程式實現

```
def alig(this, others):
        avg_vx = 0
        avg_vy = 0
        for other in others:
                avg_vx += other.vx
                avg_vy += other.vy
        avg_vx /= len(others)
        avg_vy /= len(others)
        this.vx += (avg_vx - this.vx) * 0.2
        this.vy += (avg_vy - this.vy) * 0.2
```

平均分量速度

改變x分量速度

改變y分量速度

cohension 程式實現

```
def cohen(this, others):
        avg_x = 0
        avg_y = 0
        for other in others:
                avg_x += other.x
                avg_y += other.y
        avg_x /= len(others)
        avg_y /= len(others)
        this.vx += (avg_x - this.x) * 0.04
        this.vy += (avg_y - this.y) * 0.04
```

平均位置

改變x分量速度

改變y分量速度

最終效果

可以改進的地方

- Data-Oriented Programming, 改善效能
- 目標
- 障礙物
- 個體差異(視野範圍、速率、三大規則的權重)

報告結束