Introduction to AI Assignment 1: Search Agent

Simple 2D Ball Game

By

資工二乙 408262416

陳嬿婷

Date Submitted: March 20,

2021

Writing

1. Describe your goal formulation and problem formulation.

- Goal formulation :

  讓my opponent變成最大的球球

- Problem formulation
  - 從黑球最大顆的開始吃
  - 避免撞擊其他比my opponent大的球
  - 若沒有黑球吃則去找比my opponent小但是又是裡面最大顆的球為目標去追，直到下一批黑球出現。

2. Describe the search strategy you used, and explain your design concept.

- Search strategy :

  我用的方式是先從 5 顆黑球裡找到最大顆且半徑大於等於 5 的那顆球並以他為 target 朝他前進，直到場面上的黑球半徑皆小於 5 時，再去找 2 到 9 號裡面半徑最大且沒有超過一號球大小的那顆球為 target 去追，過程中若黑球 reset，則繼續以一開始的方式找黑球當 target 直到遊戲結束。

- Design concept :
  - 從 5 顆黑球裡找到最大顆且半徑大於等於 5 的那顆球並以 location 記錄他的位置。用 flag 記錄他有 max 的黑球可以追

```
// find the biggest black ball and return its locatin
for(size_t i = 10; i < 15; i++){
    if(circleRadius[i] > 0 && circleRadius[i] >= maximum_blackball){
        maximum_blackball = circleRadius[i];
        location = i;
        flag = true;
    }
}
```

- 如果沒有黑球可以追，則找 ball 2 到 ball9 中半徑小於 ball1 且裡面最大的為目標用 location 記錄他的位置

```
// if the black ball's radius are all smaller than 5, and the target turn to the ball which is smaller than ball 1 and the biggest in ball 2 to ball 10;
if(!flag){
    for(size_t i = 10; i < 15; i++){
        if(circleRadius[i] < 5){
            for(size_t t = 1; t < 10; t++){
                if(circleRadius[t] < circleRadius[0] && circleRadius[t] > 0 && circleRadius[t] > maximum_ball){
                    maximum_ball = circleRadius[t];
                    location = t;
                    break;
                }
            }
        }
    }
}
```

- 判斷 target 在 ball1 的哪一邊並將其 X 座標、Y 座標傳到 function direction 判斷前進的方向

```
// find the target in the which side of ball 1
int x_target = xCoordinate[location] - selfx;
int y_target = yCoordinate[location] - selfy;
// decide the direction
action = direction(x_target, y_target);
```

- Function direction

```
int direction(int x_target, int y_target)
{
    int action;
    if(x_target == 0 && y_target > 0) //the target ball is above
        action = DOWN;
    else if(x_target == 0 && y_target < 0) //the target ball is under
        action = UP;
    else if(x_target > 0 && y_target == 0) //the target ball is on the right
        action = RIGHT;
    else if(x_target < 0 && y_target == 0) //the target ball is on the left
        action = LEFT;
    else if(x_target > 0 && y_target > 0) //the target ball is at the bottom right
        action = DOWN_RIGHT;
    else if(x_target > 0 && y_target < 0) //the target ball is at the top right
        action = UP_RIGHT;
    else if(x_target < 0 && y_target > 0) //the target ball is at the bottom left
        action = DOWN_LEFT;
    else if(x_target < 0 && y_target < 0) //the target ball is at the up right
        action = UP_LEFT;

    return action;
}
```

3. Describe the challenges you encountered when designing the agent.

- 方向的判斷，在寫 function direction 的時候上下容易搞混，因為忘記 XY 軸的方向。

- 理解各個參數所表示的意思，一開始用假設的猜測 ball 1 的位址但遲遲無法確定假設是否正確，最後以 freopen 的方式去確認假設的是否正確。
- 速度問題，因為速度為 read only 所以不能更改，以致於目前有一個狀況是球衝太快導致在追 target 的時候會有擦邊、吃不到一直在 target 旁轉圈圈直到 black ball reset 才會再找新的 target。
- 尚未找到撞擊到比 ball 1 大的球的方式，所以當 ball 1 一開始若吃得比其他球慢，就比較容易使得遊戲失敗。
- Black ball 大小，因為在測試的時候發現因為慣性的關係，當 target 太小會導致 ball1 一直徘徊在他附近以至於無法去吃其他的 ball，所以才假設 maximum_blackball = 5, 減少吃不到小球的機率。
- 因為上述原因有可能會造成沒有 target，所以才會再去判斷其他球的大小並以他們為 targer，減少徘徊以及被其他大球撞到的機率，而且去追比自己小一點的球也可以較快的超越比 ball1 大的球，使得遊戲成功。
- 剛開始使用*min_element()去找 circleRadius 的最小值，但因為沒有考慮到最小有可能為 0，所以倒置遊戲一直無法成功，後來用途法煉鋼使用 for loop 去找，但因為找最小值吃的效率不好，所以改成找最大值且半徑不能小於 5 的黑球，以增加獲勝的機率。
- 另一個 Search strategy 是找出離 ball1 最近的黑球，並判斷是否有其他球能比 ball1 更快到達黑球(用 時間 = 距離 / 速率 計算) ，如果有就換另一個黑球當 target。一開始知道如何算吃每顆球的 target ball 得距離(用兩點間的距離攻式)，但我想不到該如何使用 xVelocity、yVelocity 算出球移動的速率，所以實做不出來。