## 

## Đại học quốc gia tp. Hồ Chí minh

TRƯỜNG ĐẠI HỌC CÔNG NGHÊ THÔNG TIN

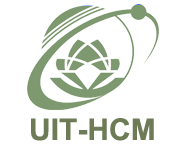
KHOA KHOA HỌC MÁY TÍNH

--------oOo--------

# BÁO CÁO bài tập TRÍ TUỆ NHÂN TẠO

# nâng cao

GAME BẮN TANK

****

**Giáo viên hướng dẫn:** Nguyễn Đình Hiển

**Sinh viên thực hiện:**

1. Bùi Quốc Tý - 12520776
2. Võ Hoài Phong - 12520894
3. Nguyễn Thu Hà - 12520964

Thành phố Hồ Chí Minh, ngày 22, tháng 05, năm 2015.

MỤC LỤC

**CHƯƠNG 1: GIỚI THIỆU BÀI TOÁN** **1**

1. Lịch sử, nguồn gốc, tác giả 1
2. Phát biểu, mô tả bài toán 1
3. Ứng dụng bài toán 2

**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT** **2**

**CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ** **5**

1. Phân tích bài toán 4
2. Cấu trúc dữ liệu và biểu diễn các trạng thái 5
3. Các vấn đề và thuật giải 7
4. Ví dụ minh họa 12

**CHƯƠNG 4: ỨNG DỤNG** 1**3**

Giới thiệu chương trình 13

Cài đặt 13

Ngôn ngữ sử dụng và các công cụ 13

Hướng dẫn cài đặt công cụ 14

Các hàm và code chính 15

Thử nghiệm 21

Kết quả chạy chương trình 21

Nhận xét và đánh giá 26

**CHƯƠNG 5: KẾT LUẬN** 27

**Ý KIẾN CỦA GIÁO VIÊN**

**CHƯƠNG 1: GIỚI THIỆU BÀI TOÁN**

1. ***Lịch sử, nguồn gốc, tác giả.***

* Năm 1985, Namco bất ngờ ra mắt cộng đồng “máy 4 nút” với một tựa game mới có tên là “Battle City” và trò chơi này được chào đón nồng nhiệt trong cộng đồng game thủ lúc bấy giờ.
* Vào năm 1994 phiên bản Super Tank ra đời và được phát hành đầu tiên tại Bắc Mỹ, Châu Âu và Nhật Bản.
* Tại Việt Nam, “Battle City ” là một tựa game thân thuộc được các game thủ gọi là game “Bắn Xe Tăng” và được yêu thích nhất bên cạnh những cái tên như “Mario” hay “Rambo Contra”… năm 2011, Battle City được tái hiện với một hình dạng mới đẹp hơn, sinh động hơn và chắc chắn là hấp dẫn hơn với phiên bản game online.
* Hiện nay có rất nhiều phiên bản game Tank nối tiếp nhau ra đời với các đấu trường cổ điển như Tank War (là một game do hãng MyWorks sản xuất và phát hành miễn phí cho điện thoại Android), Siêu Tăng Hồ Lô Quỷ, …. Đặc biệt vào tháng 2/2014, một phiên bản mới của Battle City ra đời với tên gọi Tank Tank online là thành quả của 24 năm ấp ủ lao động miệt mài, không biết mệt mỏi của đội ngũ kĩ thuật Tamtay Studio.

1. ***Phát biểu và mô tả bài toán.***

* TankVN thuộc thể loại game casual, Game được xây dựng theo hình thức thời gian thực. Người chơi thi đấu với máy tính.
* Bài toán đặt ra là thiết kế các xe tank tự động có khả năng tìm đến vị trí của Player hoặc Nhà chính để phá hủy một cách nhanh và hiệu quả nhất. Khi gặp vật cản phải biết né đi hướng khác hoặc chọn giải pháp phá hủy để đi tiếp.
* Ngoài ra còn phải xây dựng hệ thống Item, cách xuất hiện của các xe tank tự động, âm thanh, tương tác giữa các đối tượng trong game..v..v..
* Như chúng ta đã biết, hiện nay đã có rất rất nhiều game bắn xe tăng khác nhau, dường như trò chơi bắn xe tăng đã trở nên quá phổ biến. Để có thể đáp ứng một cách tốt nhất thị hiếu cho người chơi thì game TankVN cần phải có những tính năng ưu việt hơn các phần mềm khác. Ngoài một số chức năng như các game bắn xe tank khác, game TankVN cần phải xây dựng thêm những điểm mà các game khác không có và làm nổi bật những điểm đó.
* Hiện nay có một sô công cụ, ngôn ngữ, môi trường để phát triển game, nhóm chúng em đã chọn XNA 4.0 để hoàn thành game bắn xe tăng bởi:
* Ngôn ngữ lập trình C# đơn giản, dễ học và hiểu nhanh hơn so với các ngôn ngữ lập trình khác.
* Môi trường XNA làm việc chuyên nghiệp, quản lý tốt khi làm game trên   
  Windows.
* Microsoft tín nhiệm thư viện XNA này.

1. ***Ứng dụng của bài toán.***

* TankVN trong thực tế là một loại trò chơi giải trí, nó giúp xã stress cho người chơi sau những giờ làm việc mệt mỏi.
* Việc phát triển các xe tank tự động trong game là tiền đề để ứng dụng trí tuệ nhân tạo vào các lĩnh vực khác trong cuộc sống.
* Đưa ra giải pháp cho việc sắp xếp, phân chia công việc cho các xe tank tự động một cách hợp lí.

**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT**

***Thuật giải.***

* Về Cơ bản Game sử dụng thuật giải nền tảng là thuật giải tìm kiếm AKT để tìm đường đi đến vị trí cần thiết của các Enemy một cách tự động.
* Thuật giải:
* Đối với nhiều bài toán, việc tìm kiếm đường đi cực tiểu sẽ được định hướng tập trung xung quanh đường đi tốt nhất, nếu sử dụng các thông tin đặc tả về bài toán gọi là các heuristic.
* Đối với việc tìm kiếm đường đi với chi phí cực tiểu, người ta sử dụng hàm đánh giá heuristic như sau:
* Gọi g(n): giá cực tiểu đường đi từ n0→n. Tại đỉnh n, g(n) xác định được.
* Gọi h(n): giá cực tiểu đường đi từ n→dich, h(n) không xác định được ⇒ người ta tìm cách ước lượng giá trị này.
* Đặt **f0(n) = g0(n) + h0(n):** dự đoán chi phí cực tiểu của đường đi từ n0→dich có đi qua đỉnh n.
* g0(n) là chi phí của đường đi từ đỉnh xuất phát đến đỉnh n tại thời điểm đang xét. H0(n) là ước lưọng (dự đoán) chi phí đường đi từ đỉnh n đến đích. Việc chọn giá trị xấp xỉ h0(n) của h(n) không có một phương pháp tổng quát và được xem như một nghệ thuật. Giá trị này sẽ do các chuyên gia đưa ra.
* Lúc này giải thuật tìm kiếm cực tiểu sẽ thay việc xét hàm g bởi hàm f.

Tuy nhiên, người ta cũng chứng minh được 2 kết quả như sau:

**Kết quả 1**: nếu h0(n) có tính chất:  và  thì thủ tục tkct sử dụng hàm f0(n) để chọn phần tử trong open ra xét (thay g(n)) sẽ cho đường đi từ n0→n\*∈dich sao cho 

**Kết quả 2:** giả sử dùng 2 hàm ước lượng  và  thoả tính chất: (giá cực tiểu của đường đi từ m→n) và . Khi đó #close2 ≤ #close1

**Nhận xét:**

 phương án tốt nhất.

 phương án tồi nhất.

* **Thuật toán aKT**

**Input:**

Đồ thị g = (v,e), đỉnh xuất phát n0

Hàm chi phí c: e → r+

C(i, j): xác định chi phí chuyển từ đỉnh i sang đỉnh j với (i, j) ∈ e

H: v → r+; h(n) xác định dự đoán chi phí tối ưu của đường đi từ đỉnh n đến đích. (ký hiệu h thay cho h0, (tương tự g))

Tập các đỉnh đích dich

**Output:**

Đường đi từ đỉnh n0 đến đỉnh n\* ∈ dich

**Procedure AKT :**

Begin

G(n0):= 0;

Push(open, n0);

While open ≠ null do

Begin



If n∈dich then

Exit {xây dựng đường đi cực tiểu}

Push(close, n);

If г(n) ≠ null then

For m∈ г(n) do

If m ∉ close then

Begin

Push(open,m);

Tính f(m);

Cha(m):=n;

End

If fmới(m) > fcũ(n) then

Begin

F(m):= fmới(m);

Cha(m):=n;

End;

End;

**Chương 3: phân tích và thiết kế**

1. ***Phân tích bài toán.***

* Bài toán cho xe tank biết tìm đường đến vị trí cần phá hủy.
* Ý tưởng giải quyết: Sử dụng tọa độ tương đối giữa 4 vị trí xung quanh xe tank tự động so với vị tri cần đến → đánh giá được vị trí nào tốt nhất thì chọn.
* Bài Toán cho xe Tank biết cách bắn.
* Ý tưởng giải quyết:
  + Trong quá trình di chuyển nếu gặp đối tượng Gạch (có thể phá hủy) thì bắn
  + Nếu vị trí có thể bắn được tới Player hoặc nhà chính thì bắn.
  + Bắn ngẫu nhiên → tạo sự bất ngờ cho người chơi.
* Bài toán cho xe Tank biết chọn nên đuổi theo Player hay Tìm phá hủy nhà chính.
* Ý tưởng giải quyết:
* Đối tượng nào gần xe tank tự động hơn thì sẽ tìm đến đối tượng đó.
* Để Game dễ chơi hơn cần thiết kế các xe Tank có độ thông minh ít hơn, giúp game mang màu sắc Game Casual điển hình.
* Tạo các xe tank có thuộc tính di chuyển ngẫu nhiên.
* Các xe tank này có các thuộc tính khác nhau về tốc độ, số Live Point, …

1. ***Cấu trúc dữ liệu và cách biểu diễn các trạng thái của bài toán.***

* Biến GLOBAL.xe chứa các thông tin về Player (vị trí, livepoint, tốc độ, …)
* Biến GLOBAL.PosObjectProtect chứa vị trí của đối tượng nhà chính giúp xe tank tự động xác định được vị trí cần đến.
* Tập CLOSE[] : Chứa các vị trí mà xe tank đã đi qua.
* Tập CellBiKet[] : Chứa các vị trí của các ô Bê Tông mà xe Tank đã gặp.
* Tập OPEN[4] Chứa 4 vị trí xung quanh của xe tank tự động ở mỗi thời điểm xét.
* Hàm IsMeetObject() trả về giá trị True nếu Player hoặc nhà chính nằm trên hướng di chuyển và trong tầm bắn đạn của xe tank tự động.
* Hàm IsNearObject() trả về giá trị True nếu Player hoặc nhà chính nằm gần với xe tank tự động 1 khoảng cách 6 ô vuông.

enum tankstate

{

up,

down,

left,

right,

stop

}

tankstate state;

* Chứa trạng thái của xe tank ( đang di chuyển lên, xuông,trái,phải hay đứng yên?).

public enum typemoving

{

Ngang,

Doc

}

* Chứa trạng thái xu hướng di chuyển của xe tank theo hàng ngang hay dọc (vấn đề này được trình bày ở bên dưới).

public enum TankType

{

intelligent,

normal,

normal2livepoint,

normalspeed

}

* Chứa kiểu của xe Tank (loại thông minh, thường, tốc độ, thiết giáp).
* Biến Biket = True nếu xe Tank va chạm với Bê Tông, ngược lại bằng False.

1. ***Các vấn đề và thuật giải.***

* Bài toán đặt ra là tìm đường đi ngắn nhất từ vị trí ban đầu của xe tank tự động đến vị trí xe Tank Player hoặc vị trí Nhà Chính để phá hủy.
* Ý tưởng giải quyết:
* Ở mỗi thời điểm, xe tank có 4 phương án để đi tiếp, bao gồm (trên, dưới, trái, phải). → Chọn ra phương án di chuyển hiệu quả nhất.
* Áp dụng thuật toán AKT, trong đó G(n) là chi phí đã đi nhưng do chúng ta chỉ xét 4 vị trí (trên, dưới, trái, phải) của xe tank nên không cần sử dụng giá trị này.
* Hàm đánh giá F(n) được so sánh như sau:
* Nếu xe Tank không ở vị trí kẹt thì độ ưu tiên của 1 vị trí A được tính bằng công thức:

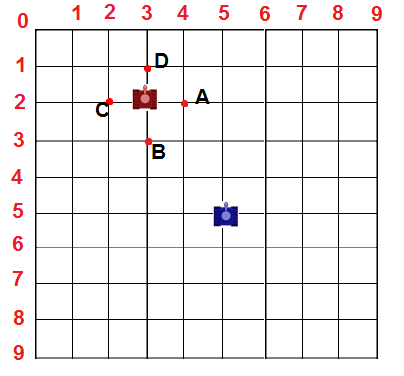
H(A) = |P.x – A.x| + |P.y – A.y|(Trong đó P là vị trí Player)

* Nếu Xe tank ở vị trí kẹt thì độ ưu tiên cũng được đánh giá bằng công thức H(n) phía trên, nhưng trong trường hợp H(A) = H(B) thì ta có thêm 1 hàm đánh giá khác là:

H’(A) =

*H’(B)=*

* Tức là so sánh coi điểm nào gần Player hơn thì chọn.
* Lý do tại sao ta phải xây dựng 2 hàm đánh giá H(n) và H’(n) như vậy? Hãy xem ví dụ dưới đây:

******

* Nếu ta chỉ sử dụng hàm H’(n) ngay từ đầu. tức là đo độ dài từ 4 vị trí quanh xe tank tự động (màu đỏ) đến xe tank Player (màu xanh), và chọn vị trí gần nhất, thì xe tank sẽ đi theo hình ZigZag → nhìn không tự nhiên như xe tank di chuyển.
* Do đó ta sẽ sử dụng hàm dánh giá H(N) = |P.x – N.x| + |P.y – N.y| trước. Với hàm này thì theo ví dụ trên, 2 vị trí A và B nếu đều không có vật cản (di chuyển đến được) nên sẽ có giá trị H(N) như nhau (2 giá trị A,B này dĩ nhiên có độ tốt lớn hơn so với 2 vị trí C,D).
* Vì xe chưa ở vị trí kẹt nên ta chọn 1 trong 2 vị trí A hay B đều được, nếu chọn A thì xe sẽ có xu hướng di chuyển theo hàng ngang, còn chọn B sẽ có xu hướng di chuyển theo hàng dọc. Để tạo sự di chuyển tự nhiên hơn nữa cho xe tank tự động, ta sử dụng cơ chế Switch 2 xu hướng di chuyển trên như sau:

if (TimeRandomTypeMoving > 1)

{

if(rd.Next(0, 10) < 5)

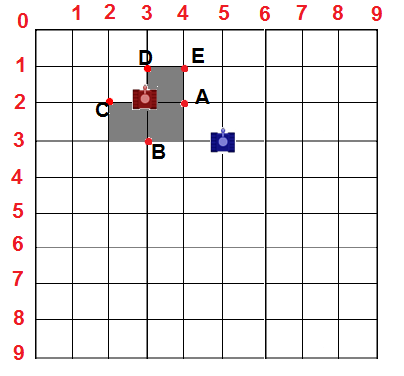
ChangeTypeMoving();

TimeRandomTypemoving = 0;

}

// rd là biến Random

* Trường hợp Tank bị kẹt (trạng thái khi xe tank va chạm bê tông):

******

* Trường hợp ví dụ trên, 2 vị trí A và B xe tank không thể di chuyển tới được do gặp vật cản (Bê Tông).
* Xét 2 điểm C và D. Nếu chỉ dùng hàm H(n) như phía trên với xu hướng đi ngang thì xe sẽ di chuyển qua C rồi lại trở lại vị trí ban đầu → không thoát ra khỏi được vật cản.
* Do đó sử dụng hàm H’(n) sẽ giải quyết được vấn đề trên. Xe sẽ chọn điểm D (do ở gần Player hơn) và sau đó di chuyển qua điểm E (xu hướng di chuyển ngang) → thoát được vật cản.
* Tuy nhiên một số trường hợp vật cản quá lớn hoặc phức tạp thì xe vẫn bị mắc kẹt. Do đó ta bổ sung thêm 2 Tập Mảng:
* Tập CLOSE: chứa các ô đã đi qua và không đi lại những ô đó. Tuy nhiên vì Player cũng di chuyển liên tục nên xe tank đôi khi vẫn phải quay lại vị trí đã đi để tiếp cận player dễ dàng hơn. Do đó ta sẽ tiến hành xóa tập CLOSE sau mỗi 6 lần di chuyển:

if (close.Count > 6)

{

close.RemoveAll(item => item != null);

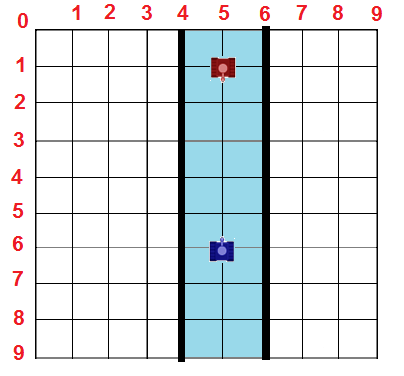
}

* Tập CellBiKet chứa các ô bị kẹt (Tức là Bê Tông) để không xét đến ở các lần sau. Xe di chuyển càng nhiều → càng có nhiều kinh nghiệm.
* Bài toán cho xe Tank Bắn
* Nếu xe gặp đối tượng gạch thì bắn.

if (va chạm(Gạch))

Fire();

* Nếu Xe tank gặp Player hoặc nhà chính ở hướng chính diện và trong tầm quan sát thì bắn.

******

* + Khung màu xanh chính là tầm quan sát của xe tank, gồm có dãy các ô chính diện xe tank và 2 dãy ô ở 2 bên.

if (IsMeetObject())

Fire();

* Bài toán cho xe Tank biết chọn nên đuổi theo Player hay Tìm phá hủy nhà chính.
* Xét xem xe tank gần với Player hơn hay Nhà Chính hơn thì đuổi theo đối tượng gần hơn đó.

if (Distance(this,player) < Distance(this,ProtectObject))

MoveTo(player);

else MoveTo(ProtectObject);

* Bài toán xây dựng các xe tank có độ thông minh thấp hơn, di chuyển ngẫu nhiên để giảm độ khó cho game và làm game đa dạng hơn.
* Thuật toán di chuyển của xe tank thường:
* Thuật toán trên được thực hiện với chu trình 0,2s một lần.
* Khi các xe Tank này ở gần Player một khoảng cách là 6 ô gạch thì sẽ chuyển thành loại xe Tank thông minh.

B1: If (Time>0.2s) thì chuyển qua bước 2

B2: a = random(1,10), nếu a < 5 thì qua bước B2.1. Nếu a>=5 thì qua bước B2.2

B2.1: b = random(1,10). Nếu b<5 thì di chuyển qua Trái. Néu b>=5 thì di chuyển qua Phải.

B2.2: b=random(1,10). Nếu b<5 thì di chuyển Lên.

Nếu b>=5 thì di chuyển xuống.

* Ngoài ra chúng ta chỉ xét chuyển động tiếp theo khi xe tank đến các đỉnh hình vuông của bản đồ, mỗi hình vuông có kích thước 26x26 nên để xét xem xe tank đã đến được đỉnh của các hình vuông này chưa, ta dùng hàm điều kiện sau:

if ((this.Position.X % 26 == 0) && (this.Position.Y % 26 == 0))

1. ***Ví dụ minh họa.***

******

B1: Xe đang ở trạng thái bị kẹt và Distance(A,player) < Distance(X,player) → đi lên vị trí A.

B2: Distance(C,player) < Distance(B,player) → đi đến vị trí C.

B3: Distance(E,player) < Distnace(D,player) → đi đến vị trí E.

Tương tự như vậy. xe sẽ đi theo quỹ đạo:

A => C => E => F => G => H => I.

**CHƯƠNG 4: ỨNG DỤNG**

1. ***Giới thiệu chương trình.***
2. ***Cài đặt.***
   1. ***Ngôn ngữ sử dụng và các công cụ.***

* Ngôn ngữ chúng em sử dụng là ngôn ngữ C# và kết hợp với công cụ XNA 4.0.
  1. ***Cài đặt các công cụ, cài đặt chương trình demo và hướng dẫn sử dụng.***
* Để máy tính có thể mở file thực thi của Game cần phải cài đặt bộ XNA Redistributable 4.0. Download tại:

<http://www.microsoft.com/en-us/download/details.aspx?id=20914>

* Để mở và biên dịch source của Game cần cài đặt Game for Windows Live sau đó cài thêm thư viện XNA 4.0

Link down Game For Windows Live:

<http://go.microsoft.com/fwlink/?LinkId=98609>

Link down XNA 4.0:

[http://www.fshare.vn/file/T6W3957RJT](http://sinhvienit.net/forum/url/?http%3A%2F%2Fwww.fshare.vn%2Ffile%2FT6W3957RJT)

Hoặc có thể cài đặt ở các file đính kèm theo tài liệu này.

* Hướng dẫn Luật chơi Game:
  + Ở level 1, người chơi sẽ được cho 3 Live Point, nhiệm vụ của người chơi là phải điều khiển xe Tank bảo vệ nhà chính khỏi bị các xe tank tự động phá hủy.
  + Các phím điều khiển bao gồm:
    - Các phím mũi tên lên, xuống, trái, phải.
    - Phím “F” để bắn.
  + Các loại vật cản trong Game:



* + -  Gạch: Có thể phá hủy được.
    -  Bê Tông: Không thể phá hủy được.
    -  Nước: Xe Tank không di chuyển qua được nhưng có thể bắn xuyên qua địa hình này.
    -  Bụi cây: Tác dụng che mắt người chơi.
  + Các loại xe tank trong Game:



* + -  Xe Tank Thường: di chuyển ngẫu nhiên.
    -  Xe Tank tốc độ: Như xe tank thường nhưng có tốc độ di chuyển gấp đôi.
    -  Xe Tank thiết giáp: như trên nhưng có 2 livepoint.
    -  Xe Tank thông minh: có khả năng tìm được 1 cách nhanh nhất đến Player hoặc nhà chính và phá hủy.
  + Các Item hỗ trợ trong Game:



* + -  Teleport: Dịch chuyển đến 1 vị trí bất kì.
    - Hủy diệt: tiêu diệt toàn bộ Enemy trên bản đồ.
    - Đóng băng: làm các Enemy trên bản đồ ngừng hoạt động trong 1 thời gian ngắn.
    - Tăng Live Point: cộng thêm 1 Live Point cho Tank player.
    -  Tốc độ: tăng tốc độ cho Tank player, bị mất tác dụng khi bị bắn trúng.
    - Khiên bảo vệ: dùng bảo vệ cho Tank player trong 1 thời gian ngắn.
    -  Bảo vệ nhà: bảo vệ nhà chính trong thời gian ngắn.
    -  Nâng cấp: dùng nâng cấp Tank player, lần đầu giúp bắn nhanh hơn, lần 2 giúp bắn được 2 viên đạn 1 lúc.
  1. ***Các hàm, đoạn code chính.***
* Hàm DanhGia1(). Tương ứng với hàm H(n) trình bày ở trên, sử dụng khi xe tank không bị kẹt.

float DanhGia1(Vector2 Position1, Vector2 Position2)

{

float X, Y;

if (Position2.X >= Position1.X) X = Position2.X - Position1.X;

else X = Position1.X - Position2.X;

if (Position2.Y >= Position1.Y) Y = Position2.Y - Position1.Y;

else Y = Position1.Y - Position2.Y;

float temp = X + Y;

return temp;

}

* Hàm DanhGia2(). Tương ứng với hàm H’(n) trình bày ở trên, sử dụng khi xe Tank bị kẹt.

float DanhGia2(Vector2 Position1, Vector2 Position2)

{

return (float)Math.Sqrt(pow((Position2.X - Position1.X), 2)+

pow((Position2.Y - Position1.Y), 2));

}

* Hàm IsMeetObject() trả về giá trị True nếu Player hoặc nhà chính nằm ở phía trước và nằm trong vùng quan sát của xe Tank tự động.

bool IsMeetObject()

{

bool MeetPlayer=false;

bool MeetProtect=false;

// Kiểm tra điều kiện gặp tank player

if ((State == tankstate.up) &&

(GLOBAL.xe.Position.Y < Position.Y) &&

(GLOBAL.xe.Position.X >= Position.X - 26) &&

(GLOBAL.xe.Position.X <= Position.X + 26))

MeetPlayer = true;

if ((State == tankstate.down) &&

(GLOBAL.xe.Position.Y > Position.Y) &&

(GLOBAL.xe.Position.X >= Position.X - 26) &&

(GLOBAL.xe.Position.X <= Position.X + 26))

MeetPlayer = true;

if ((State == tankstate.left) &&

(GLOBAL.xe.Position.X < Position.X) &&

(GLOBAL.xe.Position.Y >= Position.Y - 26) &&

(GLOBAL.xe.Position.Y <= Position.Y + 26))

MeetPlayer = true;

if ((State == tankstate.right) &&

(GLOBAL.xe.Position.X > Position.X) &&

(GLOBAL.xe.Position.Y >= Position.Y - 26) &&

(GLOBAL.xe.Position.Y <= Position.Y + 26))

MeetPlayer = true;

// Kiểm tra điều kiện gặp nhà chính

if ((State == tankstate.up) &&

(GLOBAL.PosObjectProtect.Y < Position.Y) &&

(GLOBAL.PosObjectProtect.X >= Position.X - 26) &&

(GLOBAL.PosObjectProtect.X <= Position.X + 26))

MeetProtect = true;

if ((State == tankstate.down) &&

(GLOBAL.PosObjectProtect.Y > Position.Y) &&

(GLOBAL.PosObjectProtect.X >= Position.X - 26) &&

(GLOBAL.PosObjectProtect.X <= Position.X + 26))

MeetProtect = true;

if ((State == tankstate.left) &&

(GLOBAL.PosObjectProtect.X < Position.X) &&

(GLOBAL.PosObjectProtect.Y >= Position.Y - 26) &&

(GLOBAL.PosObjectProtect.Y <= Position.Y + 26))

MeetProtect = true;

if((State == tankstate.right) &&

(GLOBAL.PosObjectProtect.X > Position.X) &&

(GLOBAL.PosObjectProtect.Y >= Position.Y - 26) &&

(GLOBAL.PosObjectProtect.Y <= Position.Y + 26))

MeetProtect = true;

if ((MeetProtect == true) || (MeetPlayer == true)) return true;

return false;

}

* Hàm IsNearObject() Trả giá trị True nếu xe Tank tự động nằm gần Player hoặc nhà chính một khoảng cách nhỏ hơn 6 ô gạch.

bool IsNearObject()

{

return ((DanhGia2(Position, GLOBAL.xe.Position) < 26\*6) ||

(DanhGia2(Position, GLOBAL.PosObjectProtect) < 26\*6));

}

* Hàm SapXep(), tác dụng sắp xếp trong tập OPEN (chứa 4 vị trí xung quanh xe tank tự động) theo thứ tự giảm dần về độ ưu tiên.
  + if (biket==false) thì dùng hàm DanhGia1().
    - Nếu kiểu di chuyển là Typemoving.Ngang thì xe di chuyển theo xu hướng chiều ngang, ngược lại là chiều dọc (đã trình bày ở trên).
  + if (biket==true) thì dùng hàm DanhGia2().
  + Position2 là vị trí đích cần đến, ở đây cụ thể là vị trí Player hoặc nhà chính.

void sapxep(List<Vector2> open, Vector2 Position2) {

Vector2 temp;

Position2.X = (int)Math.Round((decimal)Position2.X / 26) \* 26;

Position2.Y = (int)Math.Round((decimal)Position2.Y / 26) \* 26;

for (int i = 0; i < open.Count - 1; i++) {

for (int j = i + 1; j < open.Count; j++) {

if (!BiKet) {

if ((TypeMoving == typemoving.Ngang)) {

if (DanhGia1(open[i], Position2) >= DanhGia1(open[j], Position2)) {

temp = open[i];

open[i] = open[j];

open[j] = temp;

}

}

if ((TypeMoving == typemoving.Doc)) {

if (DanhGia1(open[i], Position2) > DanhGia1(open[j], Position2)) {

temp = open[i];

open[i] = open[j];

open[j] = temp;

}

}

} else if (DanhGia1(open[i], Position2) > DanhGia1(open[j], Position2)) {

temp = open[i];

open[i] = open[j];

open[j] = temp;

} else if (DanhGia1(open[i], Position2) ==

DanhGia1(open[j], Position2)) {

if (DanhGia2(open[i], Position2) >

DanhGia2(open[j], Position2)) {

temp = open[i];

open[i] = open[j];

open[j] = temp;

}

}

}

}

}

* Hàm DiChuyen(). Sử dụng thuật toán AKT để điểu khiển xe tank chuyển động.
  + Thêm vị trí hiện tại vào CLOSE.
  + Add 4 vị trí xung quanh vào OPEN.
  + Nếu 1 trong 4 vị trí này đã có trong CLOSE hoặc CellBiKet thì xóa nó đi.
  + Sắp xếp OPEN theo thứ tự giảm dần độ ưu tiên.
  + Chọn OPEN đầu tiên ra và di chuyển đến vị tri đó.

void DiChuyen(Vector2 Position) {

//open.Add(this.Position);

if ((this.Position.X % 26 == 0) && (this.Position.Y % 26 == 0)) {

open.RemoveAll(item => item != null);

Vector2 tren = new Vector2(this.Position.X, this.Position.Y - 26);

Vector2 duoi = new Vector2(this.Position.X, this.Position.Y + 26);

Vector2 trai = new Vector2(this.Position.X - 26, this.Position.Y);

Vector2 phai = new Vector2(this.Position.X + 26, this.Position.Y);

if (!GLOBAL.IsInsideList(this.Position, close))

close.Add(this.Position);

open.Add(tren);

open.Add(duoi);

open.Add(trai);

open.Add(phai);

sapxep(open,Position);

while ((GLOBAL.IsInsideList(open[0], close)) ||

(GLOBAL.IsInsideList(open[0], CellBiKet))) {

open.Remove(open[0]);

if (open.Count == 0) break;

}

if (open.Count > 0) {

if (open[0] == tren) {

Up();

} else if (open[0] == duoi) {

Down();

} else if (open[0] == trai) {

Left();

} else if (open[0] == phai) {

Right();

}

}

}

}

* Thuật toán giúp xe Tank biết được nên đuổi theo Player hay tìm phá nhà chính:

if (DanhGia2(this.Position, GLOBAL.xe.Position) <

DanhGia2(this.Position, GLOBAL.PosObjectProtect))

DiChuyen(GLOBAL.xe.Position);

else

DiChuyen(GLOBAL.PosObjectProtect);

* Thuật toán di chuyển ngẫu nhiên của xe Tank loại thường:

TimeNormalChangeMove += gameTime.ElapsedGameTime.TotalSeconds;

if ((this.Position.X % 26 == 0) &&

(this.Position.Y % 26 == 0) &&

(TimeNormalChangeMove>0.2)) {

int a = rd.Next(1,10);

if (a < 5) {

int b = rd.Next(1, 10);

if (b < 5) Left();

else Right();

} else {

int b = rd.Next(1, 10);

if (b < 5) Down();

else Up();

}

}

if (TimeNormalChangeMove > 0.2) TimeNormalChangeMove = 0;

* Hàm xử lý va chạm giữa 2 rectangle (hình chữ nhật) với nhau.

public static bool IsInside(Rectangle A, Rectangle B)

{

Return (A.X >= B.X && B.X + B.Width >= A.X + A.Width) &&

(A.Y >= B.Y && B.Y + B.Height >= A.Y + A.Height);

}

public static bool IsOutside(Rectangle A, Rectangle B)

{

return (A.X + A.Width <= B.X || A.X >= B.X + B.Width) ||

(A.Y + A.Height <= B.Y || A.Y >= B.Y + B.Height);

}

public static bool IsCut(Rectangle A, Rectangle B)

{

return (!IsInside(A, B) && (!IsOutside(A, B)));

}

* Tuy nhiên khi các viên đạn (Bullet) va chạm với nhau xét bằng các hàm trên sẽ không chính xác nên Game còn sử dụng 1 hàm va chạm khác tính theo Pixel.
  + Giá trị trả về True nếu 2 hình ảnh có 2 điểm pixel chạm vào nhau.

public static bool vachampixel(Rectangle A, Color[] dataA, Rectangle B, Color[] dataB) {

int top = Math.Max(A.Top, B.Top);

int bottom = Math.Min(A.Bottom, B.Bottom);

int left = Math.Max(A.Left, B.Left);

int right = Math.Min(A.Right, B.Right);

Color colorA = new Color(0, 0, 0, 0);

Color colorB = new Color(0, 0, 0, 0);

if (dataA == null || dataB == null) return false;

for (int y = top; y < bottom; y++) {

for (int x = left; x < right; x++) {

// Lấy màu của 2 điểm ảnh tại một điểm

int temp;

temp = (x - A.Left) + (y - A.Top) \* A.Width;

if ((temp >= 0) && (temp < dataA.Length))

colorA = dataA[temp];

temp = (x - B.Left) + (y - B.Top) \* B.Width;

if ((temp >= 0) && (temp < dataB.Length))

colorB = dataB[temp];

// Nếu cả 2 điểm đều có độ trong suốt khác không thì xảy ra va chạm

if (colorA.A != 0 && colorB.A != 0) {

return true;

}

}

}

// Va chạm không xảy ra

return false;

}

* Ngoài ra còn có các xử lý sau:
  + Nếu gặp object thì bắn.
  + Nếu gần Object thì trở thành thông minh.

if (IsMeetObject()) {

Fire();

}

if (IsNearObject()) Type = TankType.intelligent;

1. ***Thử nghiệm.***
   1. ***Kết quả chạy chương trình.***

**

Hình minh họa: Menu của game (với các Button Play, Music, About, và Exit)

**

Hình minh họa: Chọn cấp độ chơi



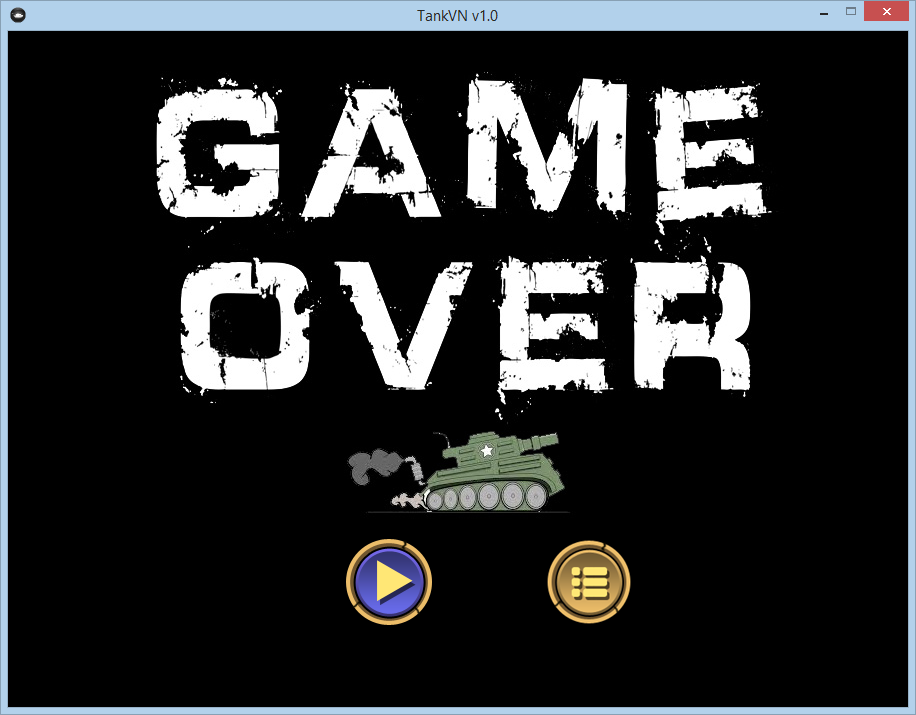
Hình minh họa: Tank tự động tìm cách tiếp cận phá hủy nhà chính.



Hiệu ứng khi ăn các Item hỗ trợ



Hình minh họa: Qua màn chơi mới



Hình minh họa: Kết thúc trò chơi khi hết livepoint hoặc nhà chính bị phá hủy

* 1. *Nhận xét và đánh giá.*
* Ưu điểm:
* Giao diện đẹp, dễ sử dụng.
* Game có nhiều màn chơi với mức độ khó tăng dần tránh gây nhàm chán cho người chơi.
* Áp dụng trí tuệ nhân tạo vào game làm cho game hay và thật hơn.
* Hệ thống Item đa dạng.
* Âm nhạc và hiệu ứng âm thanh hấp dẫn cũng là một lợi thế.
* Khuyết điểm:
* Chưa có chức năng lưu điểm người chơi cao nhất.
* Chưa có chức năng chơi 2 người.
* Còn một vài lỗi khi chơi như 2 xe tank có thể di chuyển trùng lên nhau.
* Và một số khuyết điểm khác cần người chơi phát hiện và cập nhật.

**CHƯƠNG 5: KẾT LUẬN**

1. ***Kết quả đạt được.***

* Học thêm được kiến thức về XNA.
* Cách đưa trí tuệ nhân tạo vào Game.
* Cách làm việc nhóm.

1. ***Hạn chế.***

* Làm việc đôi khi còn chưa hiệu quả.
* Game còn mất phải một số lỗi logic.
* Thiết kế các Class chưa thực sự khoa học, trong sáng.
* Vì là lần đầu làm game trên XNA nên còn nhiều thiếu sót.

1. ***Hướng phát triển.***

* Nếu có thời gian và được đầu tư sẽ phát triển tiếp các chức năng 2 người chơi, chức năng tạo map.
* Sửa các lỗi logic trong game.
* Xa hơn nữa có thể đưa game thành game online.

**TÀI LIỆU THAM KHẢO**

* [https:](https://sites.google.com/site/netbookxna)**[//](https://sites.google.com/site/netbookxna)**[sites.google.com/site/netbookxna](https://sites.google.com/site/netbookxna)
* <http://stackoverflow.com>
* <http://xvna.forumb.biz>