2012 年　AI 大戦　AI 制作マニュアル vol.1

TeRes

注意

本プロジェクトは開発用ソフトとして、Microsoft Visual C++ 2010 Express を使用しています。したがって、使用できる言語は C、C++に限られることをあらかじめご了承ください。

ゲームの大まかな詳細は公式マニュアルを参照してください。

また、本文で指示している箇所以外ではソースコードを書き換えないでください。思わぬ誤作動を起こすことがあります。

目次

1.ダウンロード、起動の手順

このゲームに使用する AI を用意するには、まず自分の手元でゲームを起動する必要があります。

https://github.com/henteko/2012TeResAI

から ZIP ファイルをダウンロードしてください。

ダウンロードが完了したら、2012TeResAI　2012teresAI の順にフォルダを開き、2012teresAI.vcxproj というファイルを起動してください。これでプロジェクトファイルが起動します。

これだけでは、デバッグしてもゲームは起動しません。よって、以下の手順に従ってください。

(1) メニューバーの**プロジェクト(P)⇒2012teresAI のプロパティ(P)**をクリックし、プロパティページを開く。

(2) 左側のリストから**構成プロパティ→　C/C++　→　全般**を選択する。

(3) 追加のインクルードライブラリの項目を編集し、
DX lib / DxLib_VC / プロジェクトに追加すべきファイル_VC 用
を追加する。

(4) 左側のリストから**リンカー　→　全般**を選択する。

(5) 追加のインクルードディレクトリの項目を(3)と同様に編集する。

(6) OK を選択し、プロパティを閉じる。

これでデバッグすればゲームが起動するようになります。これで基盤の準備は完了です。

2.AI の仕様

まず、このゲームで対戦する AI の仕様ですが、このようなメンバ変数を与えられた構造体となっています。

```
typedef struct{
    char name[100];//名前
    int Graph;//AIの画像
    Action (*moveFunc)(int view[2*VISIBLE+1][2*VISIBLE+1]);
    int x;//座標x
    int y;//座標y
    int s_x;//ドットレベルの座標(滑らかな動き)
    int s_y;//ドットレベルの座標(滑らかな動き)
    int step;//何歩移動したか
    Action act;//AIの行動
    int life;//
    int view[2*VISIBLE+1][2*VISIBLE+1];//
} AI_T;
```

☆name　・・・AI の名前

☆Graph　・・・AI の画像

☆moveFunc　・・・移動関数（後述）の関数ポインタ

x, y　・・・マップ上の AI のマスの位置

s_x, s_y　・・・ドットレベルの AI の位置。描画するためのもの。

step　・・・ドットレベルでの移動距離。moveFunc を呼び出すタイミングを決める

☆act　・・・次に移動する方向を定める変数。Action 型（後述）

☆view　・・・AI の周辺のマップの情報を格納する 2 次元配列（後述）

この中で AI の制作に必要なのは☆のついた 5 つの変数だけです。他は無視しても構いません。

～Action 型について～

moveFunc と act の宣言に Action という見慣れない型名が使用されていますが、これは次のように定義されている列挙型です。

typedef enum{N,E,S,W,STOP} Action;

AIは常に変数actとしてN,E,S,W,STOPのいずれかの値を持っており、これに従った動き（例 : N→北へ移動、STOP→移動しない）をします。そして、AIがマスの中央に来たとき、関数moveFuncが実行され、actの値が変化します。つまり、交差点における移動方向を決定するmoveFuncの内容がAIの核になります。

～viewについて～

・viewはAIの視界の中にあるマップデータを格納したint型2次元配列です。

・AIが見渡せるマスの数はヘッダーファイルData.hで以下のように定義されており、

#define VISIBLE 10

AIは10マス先まで見渡すことができます。

・viewはAIのいるマスから上下左右に10マス先までのステージのマップデータを保存しているので、要素数は21*21となります。

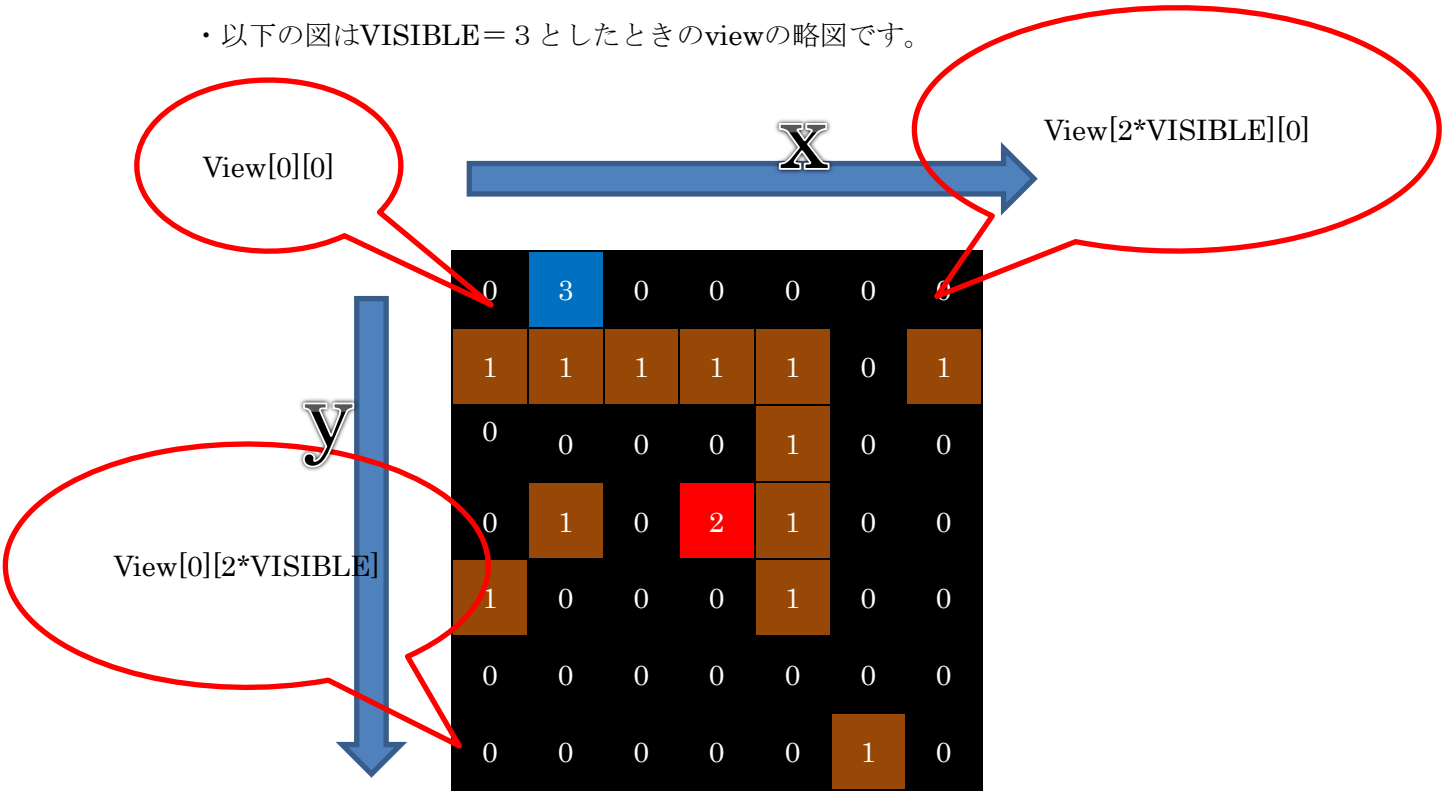
・viewの要素へアクセスするときはAIから見て最も左上の要素をview[0][0]とし、添え字はview[x][y]（ｘ：右向きが正、ｙ：下向きが正）となります。

・AI自身の位置はviewで表されたマップデータの中央、つまりview[10][10]ですが、この位置を表すための定数CENTERが用意されています。

よって、view[CENTER][CENTER]でもAI自身の位置を表すことができます。

・マップデータの中身は0,1,2,3のいずれかであり、それぞれ通路（通過できる）、壁（通過できない）、AI、鬼を表します。

・以下の図はVISIBLE＝３としたときのviewの略図です。



2.自分の AI の追加

基盤には 2 つのサンプル AI が搭載されており、3 つ目以降の AI をあなたが追加することになります。

～AI 追加の流れ～

1) cpp ファイルの追加
2) 初期化関数、移動関数の追加
3) 関数 intro の修正
4) AI 数の修正

(1)cpp ファイルの追加

まず、ソースファイルにあなたの作る AI の中身を持った cpp ファイルを追加します。

左側のリストからソースファイルフォルダを右クリック → 追加 → 新しい項目 → C++ファイルを選択、名前を入力して**追加**をクリック。

(2)初期化関数、移動関数の追加

追加されたソースファイルに aiSample.cpp または aiTest.cpp の内容をそのままコピペしてください。

コピペしたソースの中に次の 2 つの関数があります(aiSample.cpp の場合)。

aiSampleInit(AI_T &myAi)

aiSample(int view[2*VISIBLE+1][2*VISIBLE+1])

これらは順に初期化関数、移動関数といいます。上記の関数名の緑字の部分を自分の AI の名前に書き直してください。

これで cpp ファイルの追加は完了です。

(3)関数 intro の修正

次に、ソースファイルの中の Intro.cpp を開いてください。

関数 intro の中で、参加する AI の登録を行います。

[書き加えの手順]

図 1 の矢印の位置に

init_ai++;

と書き、ソースコード内にコメントとして書かれているテンプレどおりの文章を挿入し、該当部分を自分の AI の初期化関数と行動関数の名前に書き換えてください（既にある AI のサンプルを参照)。また、それ以降さらに AI を追加する場合は追加した文章の下に書き加えてください。

図 1

```
//AIを変更するときにいじる場所

//テンプレ
//extern void !初期化関数名!(AI_T &myAi);
//!初期化関数名!(ai[!aiの番号!]);
//extern Action !移動関数名!(int view[2*VISIBLE+1][2*VISIBLE+1]);
//init_ai->moveFunc = !移動関数名!;
//!!![重要]!!!!一番初め(aiSample)以外、最初にinit_ai++;を書く!!!!!!!!!!
//関数名などが被ると駄目です。名前は早い者勝ちで(基本自分のハンドルネーム)

//AI0 (aiSample)
extern void aiSampleInit(AI_T &myAi);
aiSampleInit(*init_ai);
extern Action aiSample(int view[2*VISIBLE+1][2*VISIBLE+1]);
init_ai->moveFunc = aiSample;
//AI0

//AI1 (aiTest)
init_ai++;
extern void aiTestInit(AI_T &myAi);
aiTestInit(*init_ai);
extern Action aiTest(int view[2*VISIBLE+1][2*VISIBLE+1]);
init_ai->moveFunc = aiTest;
//AI1
```

←ここに追加

```
//位置などの決定
for(int i=0;i<AI_NUM;i++,ai++){//AIの初期化
```

これで Intro の準備は完了です。

(4)AI 数の修正

最後に、AI の総数を定義する部分を書き換えます。ヘッダーファイルの Data.h を開き、その中の

#define AI_NUM 2

という文章を見てください。これは AI の総数が 2 であることを表しています。よって、AI_NUM の後ろの数字をサンプルとあなたが追加した AI の数の和に書き変えてください。

これでようやく新たな AI の登録が完了しました。

3.AI の初期化関数について

新たに追加した cpp ファイルの中の次の関数を見てください。

void AI 名 Init(AI_T &myAi)

この関数は初期化関数といい、AI の初期化、具体的にはこの AI の位置に表示するグラフィックと AI の名前を決定する関数です。

関数の中身は次の 2 行となっており、それぞれ AI の画像、名前の決定を行っています。緑字の部分をあなたの AI に合わせて書き換えてください。

```
myAi.Graph = LoadGraph("AI_image/AIの画像ファイル名");
strcpy_s(myAi.name, "AIの名前");
```

この関数をいじる前に、AI_image フォルダにあなたの AI のグラフィックとして使用する画像ファイルを入れてください。

画像の大きさは 32x32 程度。対応する拡張子については DX ライブラリのホームページを参照してください。

使用する画像については著作権に配慮すること。

4.AI の移動関数について

さて、ここまでの操作によって、サンプル AI と同じ動作をする、あなたの名義の AI ができたはずです。ここからはあなたの AI の独自の動きを作っていきます。

cpp ファイルの中の、この関数を見てください。

Action AI 名(int view[2*VISIBLE+1][2*VISIBLE+1])

この関数は移動関数といい、Action 型の値（N,E,S,W,STOP のいずれか）を返します。AI は変数 act にこの関数の返り値を代入し、次の行動を決定します。

例）return N; → 北へ1マス移動。
return E; → 東へ1マス移動。
return S; → 南へ1マス移動。
return W; → 西へ1マス移動。
return STOP; → その場で停止。

この関数の引数は周辺のマップデータを格納したint型2次元配列viewだけです。つまり、行動決定の手がかりはこの配列だけということになります。いかに地形、鬼の位置、他のAIの位置から安全なルートを導き出せるかが勝利へのカギとなります。

例）

・view[x][y]の場所に鬼がいたら、北へ移動する。

```
if(view[x][y]==3) return N;
```

・view[x][y]の場所に鬼がおり、鬼が自分よりも南にいたら北へ移動する。

```
if(view[x][y]==3 && y>CENTER) return N;
```

・自分の北側に壁があったら、西へ移動する。

```
if(view[CENTER][CENTER-1]==1) return W;
```

・自分の北側に壁がなければ北へ移動し、
北側に壁があり、西側に壁がなければ西へ移動し、
北側と西側に壁があり、東側に壁がなければ東へ移動し、
北側と西側と東側に壁があるとき南へ移動する。

```
if(view[CENTER][CENTER-1]!=1) return N;
else{
  if(view[CENTER-1][CENTER]!=1) return W;
  else if(view[CENTER+1][CENTER]!=1) return E;
  else return S;
}
```

5.サンプルAIの解説

aiSampleがどのような動きをしているのか、見てみましょう。

aiSampleの動作を大まかに分けると、

(1) viewの全要素を探索して自分の周囲に鬼がいるか確認。

(2) ((1)で鬼がいなかった場合)四方へランダムに移動。

(3) ((1)で鬼がいた場合)鬼の位置を記録。

(4) 記録した鬼の位置から遠ざかるような方向へ移動。

これだけ！ソースファイルを見て具体的にどうやってるのか見てみよう！