Curso: Desenvolvendo Jogos 2d Com C# E Microsoft XNA

Conteudista: André Luiz Brazil

Aula 11: TORNANDO A ESPAÇONAVE INIMIGA MAIS INTELIGENTE

META

Tornar a espaçonave inimiga mais ágil e inteligente.

OBJETIVOS

Ao final da aula, você deve ser capaz de:

1. Fazer a espaçonave inimiga tentar escapar dos tiros disparados pelo jogador;
2. Fazer a espaçonave inimiga atirar na nave do jogador;
3. Fazer a espaçonave inimiga se aproximar da nave do jogador.

PRÉ-REQUISITOS

1. Conhecer a ferramenta XNA Game Studio, conceito abordado na aula 2;
2. Possuir um computador com as ferramentas Visual C# e XNA Game Studio instaladas, conforme explicado na aula 3**;**
3. Ter o seu projeto de jogo atualizado conforme o conteúdo da aula 9, que inclui a a verificação de colisões entre os tiros disparados e a espaçonave inimiga.

Introdução

Na aula anterior aprendemos a programar efeitos duradouros dentro do jogo, tais como explosões. Acrescentamos o efeito de explosão quando uma espaçonave inimiga era acertada pelo tiro disparado pela nave do jogador.

Agora vamos trazer um pouco mais de inteligência ao jogo. Aprenderemos como programar algumas táticas para a espaçonave inimiga se tornar mais desafiadora.

Figura 11.1

Fonte: http://www.carteret.edu/IntranetHome/Admissions/images/thinking.jpg (Jefferson, favor redesenhar)

Caixa de Ênfase

Inteligência Artificial é a programação de ações automáticas a serem tomadas pelos objetos do jogo de acordo com os acontecimentos ocorridos durante o jogo.

Fim da Caixa de Ênfase

Como podemos tornar a espaçonave inimiga mais inteligente?

Precisamos programar algumas ações automáticas a serem tomadas pela espaçonave inimiga, com base nas ações que o jogador está tomando durante o jogo.

Quais ações automáticas podem tornar a espaçonave inimiga mais inteligente?

- Se esquivar dos tiros disparados pelo jogador;
- Atirar na espaçonave do jogador;
- Se aproximar da espaçonave do jogador.

Escapando dos tiros

Vamos programar a espaçonave inimiga para tentar se esquivar dos tiros disparados pelo jogador.

Passo 1: Adicione o método **Proximo** à classe Espaçonave, para verificar se um objeto está próximo da espaçonave. Este método é muito importante e será a base para todos os outros testes que iremos fazer.

```
// Testa a proximidade em relação a um objeto
public bool Proximo(int p_proximidade, int p_posicaox_objeto, int
                    p_posicaoy_objeto)
{
   // Testando se o objeto está próximo da espaçonave
   if (_posicaox + _textura.Width + p_proximidade > p_posicaox_objeto &&
       _posicaox < p_posicaox_objeto + p_proximidade &&
       _posicaoy + _textura.Height + p_proximidade > p_posicaoy_objeto &&
       _posicaoy < p_posicaoy_objeto + p_proximidade)
   {
       return true;
   }
   else return false;
}
```

O teste para verificar se o objeto está próximo da espaçonave é bastante semelhante ao teste de colisão, apenas inclui mais um valor de proximidade para verificar quando o objeto está chegando mais perto da espaçonave.

Passo 2: Acrescente um método **Escapar** à classe Espaconave para fazer com que ela se movimente na direção oposta aos tiros disparados no jogo quando estes se aproximarem dela

```
// Fugir de um tiro
public void Escapar(int p_posicaox_objeto, int p_posicaoy_objeto, int
                    p_posicaox_maxima, int p_posicaoy_maxima)
{
      // Testando se o objeto está próximo da espaçonave
      if ( Proximo(50, p_posicaox_objeto, p_posicaoy_objeto) )
      {
          // Tiro abaixo e à esquerda - mover para cima e direita
          if (p_posicaox_objeto >= _posicaox && p_posicaoy_objeto >
               _posicaoy)
          {
              Mover(1, p_posicaox_maxima, p_posicaoy_maxima);
              Mover(4, p_posicaox_maxima, p_posicaoy_maxima);
          }

          // Tiro abaixo e à direita - mover para cima e esquerda
          if (p_posicaox_objeto < _posicaox && p_posicaoy_objeto >
               _posicaoy)
          {
              Mover(1, p_posicaox_maxima, p_posicaoy_maxima);
```

```
                Mover(2, p_posicaox_maxima, p_posicaoy_maxima);
            }
        }
}
```

Repare que o método Escapar recebe quatro parâmetros: as posições x e y do objeto do qual a espaçonave está tentando escapar e as posições-limite x e y da tela.

Observe que o movimento da espaçonave é sempre na direção contrária à posição atual do objeto em relação à espaçonave, ou seja, se o objeto está abaixo e à esquerda da espaçonave, por exemplo, esta se movimentará para cima e para a direita.

Passo 3: Acrescente ao método de atualização do jogo (**Update**) uma instrução para chamar o método **Escapar** da espaçonave inimiga. O melhor local para fazer isso é logo após a movimentação de cada tiro. Veja como ficou:

```
// Movendo os tiros do jogo
foreach (Tiro oTiro in TirosDisparados.GetRange(0,TirosDisparados.Count))
      {
          oTiro.Mover();

          // Inimigo foge dos tiros
          NaveInimiga.Escapar(oTiro.PosicaoX(),
                  oTiro.PosicaoY(),
                  this.Window.ClientBounds.Width,
                  this.Window.ClientBounds.Height);
```

Atividade Prática 1 – Atende ao Objetivo 1

Seguindo os três passos descritos anteriormente, abra o seu projeto de jogo na ferramenta Visual C# e programe o método **Escapar** da classe espaçonave para fazer a espaçonave inimiga tentar escapar dos tiros disparados pela nave do jogador.

Fim da Atividade Prática 1

Atirando contra o jogador

Vamos agora programar a espaçonave inimiga para atacar a espaçonave do jogador atirando contra ela.

Passo 1: Acrescente um método **Atacar** à classe Espaconave para fazer com que ela se decida se irá atacar caso algum objeto esteja se aproximando dela.

```
// Atacar um objeto inimigo
public bool Atacar(int p_posicaox_objeto, int p_posicaoy_objeto)
{
      if (Proximo(200, p_posicaox_objeto, p_posicaoy_objeto))
          return true;
      else
```

```
            return false;
}
```

Repare que o método **Atacar** é apenas uma chamada ao método **Proximo** com o valor 200, ou seja, apenas decide se a espaçonave deve ou não atacar com base na proximidade do objeto inimigo em relação à espaçonave.

Passo 2: Ajuste o método **Atirar** da classe **Espaçonave** para que receba um novo parâmetro, que é a direção do tiro.

```
public void Atirar(int p_direcao_tiro, GraphicsDevice
                   p_dispositivo_grafico, string p_local_textura_tiro,
                   List<Tiro> p_lista_tiros)
{
      Tiro oTiro;
      // Cria um tiro
      oTiro = new Tiro(_potencia_tiro, p_direcao_tiro,
                  _velocidade_tiro, _posicaox, _posicaoy);

      // Carrega a textura do tiro
      oTiro.CarregarTextura(p_dispositivo_grafico, p_local_textura_tiro);

      // Ajusta a posição inicial do tiro
      oTiro.PosicaoX(oTiro.PosicaoX() + (_textura.Width /2));

      if (p_direcao_tiro == 3) // Tiro para baixo
       {
              oTiro.PosicaoY(oTiro.PosicaoY() +
                                  _textura.Height);
       }

      // Acrescenta o tiro na lista de tiros do jogo
      p_lista_tiros.Add(oTiro);
}
```

Observe que foi acrescentado também um ajuste para a posição inicial y do tiro caso a direção do tiro seja para baixo (direção 3). Isto foi feito para que o tiro saia abaixo da textura da espaçonave inimiga e não por dentro dela.

Passo 3: Acrescente ao método de atualização do jogo (**Update**) uma instrução para chamar o método **Atacar** da espaçonave inimiga. O melhor local para fazer isso é logo após a movimentação da espaçonave do jogador. Veja como ficou:

```
if (teclado.IsKeyUp(Keys.Space) && apertou_tiro)
{
    apertou_tiro = false;
}

// Atirando contra o jogador
if (NaveInimiga.Atacar(NaveJogador.PosicaoX(),
                       NaveJogador.PosicaoY()))
```

```
{
        NaveInimiga.Atirar(3,graphics.GraphicsDevice,
         "../../../Content/Imagens/tiro.png",
         TirosDisparados);
}
```

Atividade Prática 2 – Atende ao Objetivo 2

Seguindo os três passos descritos anteriormente, abra o seu projeto de jogo na ferramenta Visual C# e programe o método **Atacar** da classe espaçonave para fazer a espaçonave inimiga tentar atirar contra a nave do jogador.

Fim da Atividade Prática 1

Se aproximando do jogador

Vamos agora programar a espaçonave inimiga para se aproximar da espaçonave do jogador, indo na sua direção quando esta chega perto.

Passo 1: Acrescente um método **Aproximar** à classe Espaconave para fazer com que ela se movimente na direção da nave do jogador quando esta ficar próxima dela.

```
// Se aproximar do objeto
public void Aproximar(int p_posicaox_objeto, int p_posicaoy_objeto, int
                      p_posicaox_maxima, int p_posicaoy_maxima)
{
  // Testando se o objeto está longe da espaçonave
  if (Proximo(250, p_posicaox_objeto, p_posicaoy_objeto))
  {
     // Objeto abaixo e à esquerda - mover para baixo e esquerda
     if (p_posicaox_objeto >= _posicaox && p_posicaoy_objeto > _posicaoy)
     {
              Mover(3, p_posicaox_maxima, p_posicaoy_maxima);
              Mover(2, p_posicaox_maxima, p_posicaoy_maxima);
     }

     // Objeto abaixo e à direita - mover baixo e direita
     if (p_posicaox_objeto < _posicaox && p_posicaoy_objeto > _posicaoy)
     {
              Mover(3, p_posicaox_maxima, p_posicaoy_maxima);
              Mover(4, p_posicaox_maxima, p_posicaoy_maxima);
          }
     }
}
```

Repare que o método **Aproximar** é bastante semelhante ao método **Escapar**, sendo que a espaçonave, neste caso, se movimenta na mesma direção que o objeto, indo de encontro a ele quando este se aproxima.

Passo 2: Acrescente ao método de atualização do jogo (**Update**) uma instrução para chamar o método **Aproximar** da espaçonave inimiga. O melhor local para fazer isso é logo após a movimentação da espaçonave do jogador. Veja como ficou:

```
if (teclado.IsKeyUp(Keys.Space) && apertou_tiro)
{
   apertou_tiro = false;
}

// Aproximando o inimigo do jogador
NaveInimiga.Aproximar(NaveJogador.PosicaoX(),
                      NaveJogador.PosicaoY(),
                      this.Window.ClientBounds.Width,
                      this.Window.ClientBounds.Height);

// Atirando contra o jogador
if (NaveInimiga.Atacar(NaveJogador.PosicaoX(),NaveJogador.PosicaoY()))
{
    NaveInimiga.Atirar(3,graphics.GraphicsDevice,
                       "../../../Content/Imagens/tiro.png", TirosDisparados);
}
```

Atividade Prática 3 – Atende ao Objetivo 3

Seguindo os dois passos descritos anteriormente, abra o seu projeto de jogo na ferramenta Visual C# e programe o método **Aproximar** da classe espaçonave para fazer a espaçonave inimiga agir na ofensiva, se aproximando da nave do jogador quando esta chega perto dela.

Fim da Atividade Prática 3

Atividade Prática 4 – Atende ao Objetivo 1

Existe uma pequena falha no código do jogo. Repare que a espaçonave inimiga tenta escapar dos seus próprios tiros. Ajuste o código do jogo de forma que a espaçonave inimiga tente escapar apenas dos tiros disparados pela espaçonave do jogador.

Dica: Será necessário criar na classe Tiro um atributo para indicar quem disparou o tiro.

Fim da Atividade Prática 4

CAIXA DE FÓRUM **Informação sobre Fórum**

Figura 11.2

Fonte: http://www.stockxpert.com/browse_image/view/28331341/?ref=sxc_hu (Jefferson- favor redesenhar)

Você teve alguma dificuldade para programar a inteligência artificial da espaçonave inimiga dentro do jogo? Entre no fórum da semana e compartilhe suas dúvidas e experiências com os seus amigos.

FIM DE CAIXA DE FÓRUM

CAIXA DE ATIVIDADE **Informação sobre Atividade on-line**

Figura 11.3
Fonte: http://www.sxc.hu/photo/1000794 (Jefferson : favor redesenhar)

Agora que você já está com o seu código de projeto do jogo ajustado para tornar a espaçonave inimiga mais inteligente, vá à sala de aula virtual e resolva as atividades propostas pelo tutor.

FIM CAIXA DE ATIVIDADE

Resumo

- **Inteligência Artificial** é a programação de ações automáticas a serem tomadas pelos objetos do jogo de acordo com os acontecimentos ocorridos durante o jogo.
- Para tornar a espaçonave inimiga mais inteligente, podemos programar as seguintes ações automáticas com base nas ações que o jogador está tomando durante o jogo:
    - Se esquivar dos tiros disparados pelo jogador;
    - Atirar na espaçonave do jogador;
    - Se aproximar da espaçonave do jogador.
- O método mais importante a ser criado na classe Espaçonave é o método **Proximo**. Ele serve para verificar se um objeto está próximo da espaçonave. Este método é muito importante e será a base para todos os outros testes que iremos fazer. Este método é bastante semelhante ao método **ColisaoSimples** do jogo.
- Também precisaremos criar os métodos **Escapar, Atacar e Aproximar**, para que a espaçonave tome as ações descritas acima.
    - No método **Escapar**, a espaçonave se movimenta na direção contrária à do objeto em questão;
    - No método **Atacar**, a espaçonave atira contra o objeto em questão, tentando acerta-lo;

- No método **Aproximar**, a espaçonave se move na mesma direção que o objeto em questão.

Fim do resumo