KILL PINGÜIM – METODOLOGIA PARA DESENVOLVIMENTO DE UM JOGO UTILIZANDO MODO GRÁFICO 13H COM IMAGENS PROPRIETÁRIAS E APLICAÇÃO DE LEIS DA FÍSICA

Celso Eduardo dos Santos, José Luis Lourenço, Reginaldo de Oliveira, Vivian Machado Rodrigues, Silene Fernandes Bicudo

Universidade do Vale do Paraíba – Faculdade de Ciência da Computação
Av. Shishima Hifumi, 2911 CEP 12244-000 São José dos Campos – SP – Brasil
E-mail: eduardo_sme@yahoo.com.br

Resumo: Este artigo apresenta a metodologia para o desenvolvimento de um jogo utilizando linguagem de programação C/C++ com o modo gráfico 13H e imagens num formato proprietário. O principal objetivo do jogo é arremessar um pinguim o mais longe possível. Utilizou-se no desenvolvimento do projeto, o software Turbo C++, o qual foi necessário configurá-lo para utilizar memória HUGE, pois o padrão TINY, pré-definido pelo software, não permite acessar a memória RAM acima dos 64KB do MS-DOS. As imagens e paletas utilizadas foram criadas através de softwares gráficos e são convertidas do padrão BMP para o formato proprietário ZEB ou ZEP por um algoritmo próprio. Esse formato proprietário (ZEB ou ZEP) garante a integridade das imagens, evitando que o usuário as modifique. Para salvar os recordes, é utilizado um arquivo do tipo DAT. Esse arquivo também é criptografado para que nenhum usuário altere facilmente seus registros. Essa criptografia baseia-se em Off-Set 2 para off-set's pares e Off-Set 3 para off-set's ímpares. Utilizamos ainda uma barra de força, recurso encontrado para fornecer o valor da força da tacada dado em Newton. Esse valor será utilizado no cálculo da distância a ser atingida pelo pingüim.

Palavras-chave: Modo 13H, Arremesso do pingüim, Boneco de neve, Linguagem C/C++, Imagens ZEB.
Área do Conhecimento: I – Ciências Exatas e da Terra

Introdução

Desenvolver um jogo gráfico em C/C++ [1] com animações requer muita disponibilidade de memória do equipamento e um tratamento para não ocorrer atrasos nessas animações, tornando-as o mais real possível. O modo convencional para se carregar imagens direto de uma unidade de disco é um dos maiores problemas para animação, pois acarreta lentidão. Aplicações de leis da física também são essenciais para um maior realismo nas animações, em especial no nosso caso para representar a curva (parábola), realizada pelo pingüim. Para desenvolvermos o jogo, utilizamos o modo gráfico 13H, como resolução 320X200X256, que é o melhor formato para utilização de jogos feitos para MS-DOS, obtendo uma boa qualidade de resolução e um alto desempenho nas animações. Além disso, o modo 13H é compatível com praticamente todas as placas de vídeo, sem a necessidade da utilização de driver's externos [2] [3].

Para que o pingüim fosse arremessado de modo a proporcionar realismo no lançamento, utilizamos as fórmulas físicas da aceleração e velocidade, fornecendo a força (Newton), o ângulo de inclinação da tacada e a coordenada inicial do pingüim (posição na tela), bem como, o tempo de aceleração da tacada, a massa do pingüim e a força gravitacional equivalente a da terra, valores que são pré-determinados [4] [5].

Outro recurso muito importante é o que diz respeito ao *Vertical Syncronization* – VSYNC. O Vsync serve pra sincronizar a renderização das imagens com o refresh rate do monitor. Para fazer a sincronização, o chip gráfico renderiza alguns frames "antes" do que deveria, pois temos que aguardar o canhão de elétrons retornar ao início antes de plotar outra imagem [6].

O objetivo do nosso trabalho é apresentar a metodologia para o desenvolvimento de um jogo, que consiste em arremessar um pinguim o mais longe possível, utilizando linguagem de programação C/C++ com o modo gráfico 13H, imagens num formato proprietário e arquivo de recordes criptografado.

Materiais e Métodos

Utilizou-se o software Turbo C++ para desenvolver o projeto, o qual foi posteriormente configurado para utilizar memória HUGE, pois o padrão TINY, pré-definido pelo software, não permite acessar a memória RAM acima dos 64KB do MS-DOS.

Para o desenvolvimento do projeto, utilizou-se Programação Orientada a Objetos – POO, que teve o seu papel fundamental, pois, através dela, obtivemos melhor controle de ponteiros, utilizados para alocação de memória. A POO não é nova, sua formulação inicial data de 1960. Porém, somente a partir dos anos 90 é que passou a ser utilizada. Hoje, todas as grandes empresas de desenvolvimento de programas tem desenvolvido os seus software's usando a programação orientada a objeto. Esse tipo de programação difere da programação estruturada. Na programação orientada a objeto, funções e dados estão juntos, formando o objeto. Esta abordagem cria uma nova forma de analisar, projetar e desenvolver programas. De uma forma mais abstrata e genérica, que permite um maior reaproveitamento dos códigos e facilita a manutenção. A programação orientada a objeto não é somente uma nova forma de programar, é uma nova forma de pensar um problema, de forma abstrata, utilizando conceitos do mundo real e não conceitos computacionais. Os conceitos de objetos devem acompanhar todo o ciclo de desenvolvimento de um software [7].

As imagens e paletas utilizadas foram criadas através de softwares gráficos. Após a criação das imagens, estas são convertidas, por um algoritmo, do padrão BMP para o formato proprietário ZEB ou ZEP. Cada imagem ZEB depende de uma paleta ZEP. Porém, pode-se criar uma única paleta ZEP para vários arquivos ZEB, desde que esses contenham as mesmas cores. Cada paleta contém no máximo 256 cores.

Ao invés de se utilizar os padrões para imagens comuns no mercado, como por exemplo, os tipos BMP, GIF e JPG, optou-se pela criação de um novo formato denominado ZEB ou ZEP. A utilização de um formato proprietário tem como principal objetivo a preservação sa integridade das imagens utilizadas.

Tabela 1 – Formato do arquivo ZEB

Off -Set	Nº Bytes	Descrição	Exemplo
0x000	15	Cabeçalho do arquivo	“By –”
0x010	02	Largura da Imagem	320 pixels
0x012	02	Altura da Imagem	200 pixels
0x014	01	Índice da cor na posição 0,0	0x00 (preto)
0x015	01	Índice da cor na posição 0,1	0x01 (azul)
0x016	63998	Cores restantes	...

Tabela 2 – Formato do arquivo ZEP

Off -Set	Nº Bytes	Descrição	Exemplo
0x00	01	Índice da cor	0x00
0x01	01	Intensidade da cor azul	0x00
0x02	01	Intensidade da cor verde	0x00
0x03	01	Intensidade da cor vermelha	0x00
0x04	01	Índice da cor	0x01
0x05	01	Intensidade da cor azul	0xFF
0x06	01	Intensidade da cor verde	0x00
0x07	01	Intensidade da cor vermelha	0x00
0x08	1016	Restante das cores	...

Observou-se que, por diversas vezes, as imagens eram plotadas pela metade na tela. Após vários testes, detectou-se que o problema estava na sincronização da renderização dessas imagens com o *refresh rate* do monitor. Para solucionar o problema, foi criada uma função para simular esse sincronismo. Essa função consiste em aguardar o canhão de elétrons retornar a posição inicial da tela antes de plotar a próxima imagem. Este recurso é conhecido como *Vertical Syncronization* – VSYNC.

Um manual do usuário (MU) on-line foi elaborado, através da criação de várias telas contendo as instruções de como jogar e principalmente a descrição dos marcadores e comandos utilizados.

Para salvar os recordes é utilizado um arquivo do tipo DAT. Também como recurso de segurança, evitando que um usuário modificasse facilmente os registros desse arquivo, de modo a alterar seu recorde, por exemplo, os dados salvos nesse arquivo são criptografados, utilizando uma criptografia simples baseada em Off-Set 2 para off-set's pares e Off-Set 3 para off-set's ímpares. Como exemplo prático, citamos um jogador com as iniciais “SIL” e que tenha atingido uma distância no lançamento de “999”. Como utilizou-se criptografia no arquivo DAT, ao invés de simplesmente salvar “SIL999”, o mesmo está representado como “ULN<;<”.

O “off-set” em questão é a posição do caractere no arquivo DAT, começando em 0x00 que é par, alternando o próximo para 0x01 que é ímpar, e assim por diante. Ou seja, o caractere “S” em ASCII é o número 83 em decimal e como seu off-set é par, será somado mais 2, obtendo o total de 85, que é o caractere “U”. O caractere “I” em ASCII é o número 73 em decimal e como seu off-set é ímpar, será somado mais 3, obtendo um total de 76 ,que é o caractere “L” e assim por diante.

Para se calcular a distância e altura que o pingüim atingirá no arremesso, necessita-se que alguns valores constantes e variáveis sejam passados por função. Como constantes tem-se: a força gravitacional (g) com valor aproximadamente igual ao da Terra, de 9,8 m/s^2; a massa (M) do pingüim estipulada em 4.4kg; o tempo (t) de aceleração da tacada, ou seja,

tempo que a força (fornecida pela barra de força) atua em contato com o pingüim, estipulado em 1.3s e a posição inicial X_0 (distância), fixado em 230, pois o pingüim, horizontalmente, sempre estará na mesma posição inicial na tela.

Como variáveis tem-se: o ângulo (Ang) de inclinação, cujo valor será obtido através da posição em que o pingüim estiver na hora da tacada, podendo variar entre 0 e 60 graus (quanto mais se retarda a tacada, menor será o ângulo, desde que o pingüim esteja na área delimitada); a força (F) da tacada, cujo valor será obtido através da barra de força, quanto mais carregada, maior será esta força, desde que obedecidos os limites inferior e superior, podendo variar de 0 a 335 N (Newton) e a posição inicial Y_0 (altura), cujo valor será obtido através da distância do chão em que o pingüim estiver na hora da tacada, podendo variar de 135 a 200.

Para se calcular a força, velocidade, tempo e distâncias utilizou-se várias fórmulas, cujo resultado final foi a curva do arremesso do pingüim. Resumidamente tem-se a velocidade inicial:

$$Vi = (F / M) * t$$

A velocidade inicial decomposta em sua componente X: $V_0X = Vi * \cos(Ang)$

A velocidade inicial decomposta em sua componente Y: $V_0Y = Vi * \text{sen}(Ang)$

A distância total: $X = X_0 + V_0X * t$

A altura total:
$Y = (Y_0 + V_0Y * t) - (g * (t * t)) / 2)$

E o tempo total:
$T (Y=0) = (V_0Y + V_0Y) / g$

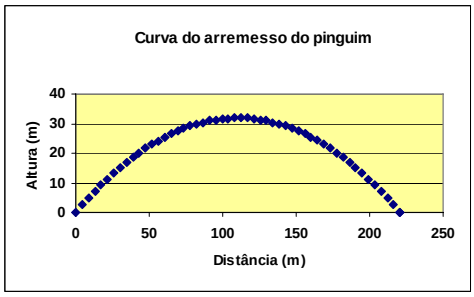

Figura 1: Gráfico da simulação de um arremesso.

A barra de força foi o recurso encontrado para fornecer o valor da força da tacada, dada em Newton, que será utilizado no cálculo da distância a ser atingida pelo pingüim.

Resultados

Foi desenvolvido um jogo utilizando linguagem de programação C/C++ com modo gráfico 13H, onde se obteve uma boa qualidade de resolução das imagens e das animações, visto que essa é uma linguagem de baixo nível.

As interfaces entre memória de armazenamento em disco, memória RAM e memória de vídeo aconteceram de maneira sincronizada, não deixando falhas nas transições das imagens, proporcionando um realismo ainda maior.

Colocando o jogo em execução, primeiramente temos a tela de créditos do jogo que permanece durante 5 segundos. Nela é possível observar o nome fictício da empresa que criou o jogo, o nome do jogo, o nome dos integrantes do grupo que desenvolveu o projeto, o nome da orientadora do projeto e a identificação da turma a qual o grupo faz parte.

Após a tela de créditos, será exibida a tela com as opções do jogo, e permanecerá até que seja escolhida uma das opções. As escolhas poderão ser “NOVO JOGO”, para iniciar o jogo propriamente dito, “RECORDES”, para mostrar os cinco maiores pontuadores do jogo, “COMO JOGAR”, para ter acesso ao manual do usuário (MU) ou “SAIR”, para sair do jogo. A escolha será feita através das teclas seta pra cima e pra baixo e confirmada com a tecla ENTER. Também é possível sair utilizando a tecla ESC.

A tela “CARREGANDO” será exibida quando for escolhida a opção “NOVO JOGO” da tela de opções. Ela permanecerá visível durante aproximadamente 2 segundos com o texto “Carregando” escrito com o efeito “pegando fogo”. Logo em seguida inicia-se o jogo propriamente dito, montando-se um cenário gelado com o pingüim no alto de uma montanha e logo abaixo, na mesma direção, o boneco de neve para rebatê-lo.

Figura 2: Cenário onde decorre o jogo.

Se escolhida a opção “COMO JOGAR” da tela de opções, aparecerá uma tela contendo um “help on-line”. Como jogar, como funciona o jogo, botões a serem utilizados e regras são algumas

informações que poderão ser obtidas através dela. Ela é constituída de duas imagens sobrepostas, uma imagem de fundo, e outra com as informações de como jogar. São várias telas, que poderão ser alternadas através das teclas direcionais: esquerda para retroceder e direita para avançar. Para sair e voltar à tela de opções pressiona-se ESC. Existe ainda uma terceira imagem como rodapé, com as indicações das teclas a serem utilizadas, já descritas anteriormente.

Se escolhida a opção "RECORDES" da tela de opções, aparecerá uma tela que também é constituída de duas imagens sobrepostas, uma imagem de fundo, e outra com os registros que serão carregados a partir do arquivo *record.dat.* Cada caractere alfa-numérico que constitui os recordes é uma imagem "plotada" na tela. Nela poderão ser observados os registros dos 05 maiores pontuadores do jogo (os cinco jogadores que atingiram a maior distância no arremesso do pingüim)

A barra de força, recurso já citado anteriormente, funciona tendo um tempo pré-determinado de 8 segundos para que o jogador fique pressionando sucessiva e alternadamente as teclas direcionais "SETA PARA A ESQUERDA" e "SETA PARA A DIREITA", fazendo com que "carregue" essa barra. Quanto mais rápidas forem as repetições, mais a barra carregará e mais força terá a tacada, pois ao mesmo tempo existirá uma força contraria definida no jogo fazendo com que a barra de força descarregue. Somente essa rapidez nas repetições conseguirá vencer essa força contraria, antes do término do tempo.

Essa barra possui também limites inferior e superior. Se a "força contraria" vencer a do jogador, fazendo com que não atinja o limite inferior, o "boneco de neve" errará a tacada, e como vingança o pingüim o explodirá com uma dinamite. Caso ultrapasse o limite superior, definido pela cor vermelha, o pingüim irá quebrar a tela do computador.

Discussão

Através da utilização do modo gráfico 13H, obteve-se uma melhor qualidade na resolução das imagens e das animações do projeto Kill Pingüim, comparando-o a outros jogos desenvolvidos na linguagem de programação C/C++ padrão.

Conclusão

A linguagem C/C++ mostrou-se eficiente para impressão de gráficos no vídeo.

O uso de interfaces para transição da imagem da memória de armazenamento para memória RAM, em seguida para memória de vídeo e só então impressa no vídeo tornou as animações bem reais, sem falhas e com alta qualidade.

Utilizando o conceito de POO, as linhas de código diminuíram, tendo então um melhor aproveitamento, e deixando-o mais estável, principalmente no uso de ponteiros, sem que houvesse estouro de memória.

O uso de fórmulas físicas foi essencial para as animações, em especial para representar a curva do arremesso do pingüim.

É importante ressaltar a possibilidade de este projeto ser utilizado como modelo no ensino de disciplinas como lógica de programação ou programação em C, na própria UNIVAP, ou instituições de ensino que tiverem interesse.

Referências

[1] SCHILDT, Herbert – C Completo e Total Ed. MAKRON Books, Terceira Edição 827p ISBN: 8534605955

[2] Modo gráfico VGA 13h
Disponível em:
http://www.elrincondelc.com/decas/ug/sp-vga.html acessado em 01/03/2005

[3] Modo gráfico VGA 13h (A to Z)
Disponível em:
http://guideme.itgo.com/atozofc/ch32.pdf
acessado em 01/03/2005

[4] Biscuola, Gualter J. e Maiali, André C. – Física Ed. Saraiva, Terceira Edição, Volume único, 1998 652p ISBN: 8502021281

[5] Angle Conversions
Disponível em
http://www.teacherschoice.com.au/Maths_Library/Angles/Angles.htm acessado em 03/05/2005

[6] VSYNC – Vertical Syncronization
Disponível em:
http://sp4br75.digiweb.psi.br/programacion_2D/Accessing_mode_13h_in_djgpp.htm acessado em 25/03/2005

[7] Apostila de Programação em C++
André Duarte Bueno, UFSC-LMPT-NPC
http://www.lmpt.ufsc.br/~andre – email: andre@lmpt.ufsc.br – Versão 0.4 – 22 de agosto de 2002.
Disponível em: http://www.apostilando.com acessado em 25/03/2005