A B A K Ó S
PUC Minas Instituto de Ciências Exatas e Informática

PUC Minas – Instituto de Ciências Exatas e Informática

Avaliação do Impacto no Desempenho do Método normalize_feature_vector

Lucas Torres Toledo
Francielle Moura Limonge
26 de setembro de 2024

Resumo

A normalização de vetores de características é uma operação essencial em várias áreas da computação, como aprendizado de máquina, gráficos computacionais e simulações científicas. Através da normalização, um vetor é ajustado para ter magnitude 1, preservando suas direções, o que é importante para operações como o cálculo de distâncias e projeções. Este trabalho tem como objetivo otimizar a função de normalização de vetores, utilizando o método *normalize_feature_vector* em três cenários distintos: utilizando uma tabela de consulta (lookup table), aplicando a otimização do Quake III (Newton-Raphson), e utilizando instruções de hardware (SSE rsqrtss). O objetivo principal é comparar o impacto no desempenho de cada técnica e discutir suas vantagens e desvantagens com base em testes de execução.

1 INTRODUÇÃO

A normalização de vetores de características tem uma vasta aplicação, principalmente em algoritmos de aprendizado de máquina e processamento de gráficos 3D. O processo de normalizar um vetor garante que ele tenha magnitude 1, o que facilita a comparação e a interpretação dos dados.

O objetivo deste trabalho é avaliar o impacto no desempenho ao aplicar três abordagens de otimização para normalizar vetores: a utilização de uma tabela de consulta (lookup table), o método de Newton-Raphson utilizado no Quake III, e a instrução de hardware *SSE rsqrtss*.

2 CARACTERIZAÇÃO DO PROBLEMA

A normalização de um vetor $\mathbf{v}$ de dimensão n é definida como:

$$\mathbf{v}_{normalizado} = \frac{\mathbf{v}}{||\mathbf{v}||} = \frac{\mathbf{v}}{\sqrt{\sum_{i=1}^{n} v_i^2}}$$

Neste problema, o cálculo da raiz quadrada inversa $\frac{1}{\sqrt{x}}$ é o ponto central de otimização. A performance da normalização depende do método escolhido para esse cálculo.

3 DESCRIÇÃO DAS ABORDAGENS DE OTIMIZAÇÃO

3.1 Tabela de Consulta (Lookup Table)

Essa abordagem armazena valores pré-calculados da raiz quadrada inversa em uma tabela, de forma que, ao normalizar o vetor, basta consultar o valor na tabela.

3.2 Otimização do Quake III (Newton-Raphson)

Essa técnica, amplamente conhecida no desenvolvimento de jogos, utiliza manipulação de bits e refinamento iterativo para calcular a raiz quadrada inversa com alta precisão e eficiência.

3.3 Instruções de Hardware (SSE rsqrtss)

O uso de instruções de hardware, como *SSE rsqrtss*, permite o cálculo da raiz quadrada inversa diretamente no nível do processador, aproveitando o paralelismo do hardware.

4 RESULTADOS DOS TESTES DE DESEMPENHO

Os testes foram conduzidos em um ambiente Linux com a seguinte configuração de hardware:

- Processador: Intel Core i7-10700, 2.90 GHz
- Memória: 16 GB RAM
- Sistema Operacional: Ubuntu 20.04 LTS

Métrica	Tabela de Consulta	Newton-Raphson	SSE rsqrtss
Tempo de usuário (s)	0.35	0.20	0.15
Tempo de sistema (s)	0.02	0.01	0.01
Memória utilizada (kB)	11200	10800	10500

Tabela 1 – Comparação de desempenho entre as abordagens de normalização

5 ANÁLISE E DISCUSSÃO

Os resultados indicam que a abordagem com instruções de hardware *SSE rsqrtss* foi a mais rápida, oferecendo uma redução significativa no tempo de execução. No entanto, depende da compatibilidade do hardware. A técnica de Newton-Raphson apresentou um bom equilíbrio entre precisão e desempenho, enquanto a tabela de consulta mostrou-se a menos eficiente em termos de tempo de execução.

6 COMO UTILIZAR O SISTEMA

Este sistema foi desenvolvido para avaliar e otimizar o desempenho de normalizações de vetores de características utilizando três métodos distintos: a Tabela de Consulta (Lookup Table), o método de Newton-Raphson (utilizado no Quake III), e instruções de hardware (*SSE rsqrtss*).

6.1 Execução dos Testes

Para utilizar o sistema e realizar a avaliação de desempenho, siga os seguintes passos:

1. **Ambiente de Execução**: O código foi desenvolvido para ser executado em um ambiente Linux. Certifique-se de ter um processador compatível com as instruções *SSE*, caso opte por testar essa abordagem.

2. **Compilação**: Compile o código-fonte com um compilador C/C++ que suporte otimizações de baixo nível, como o GCC (GNU Compiler Collection), garantindo a ativação de instruções SIMD (Single Instruction, Multiple Data) com o seguinte comando:

```
gcc -O3 -msse -o normalize normalize.c
```

3. **Execução**: Execute o programa compilado para que os resultados de desempenho sejam

registrados. O sistema reportará os tempos de execução e o uso de memória de cada uma das abordagens:

```
./normalize
```

4. **Análise dos Resultados**: Após a execução, o sistema exibirá métricas de tempo de usuário, tempo de sistema e memória utilizada por cada abordagem (Tabela de Consulta, Newton-Raphson, e *SSE rsqrtss*). Compare os resultados e selecione o método mais apropriado para suas necessidades com base no desempenho e na compatibilidade do hardware.

6.2 Personalização

Caso deseje realizar testes com diferentes vetores de características ou alterar o tamanho dos vetores, o sistema permite a personalização dessas variáveis no arquivo-fonte. Acesse o código e modifique as seções apropriadas para testar novos cenários ou otimizações.

7 CONCLUSÃO

A análise revelou que, para ambientes compatíveis com SSE, essa abordagem é a mais eficiente. Em cenários onde a portabilidade e precisão são mais importantes, o método de Newton-Raphson é recomendado. A técnica de tabela de consulta, embora viável, não apresentou o melhor desempenho.

8 REFERÊNCIAS

- WATSON, D. Optimizing Game Algorithms: A Historical Perspective. 2004.
- NVIDIA Corporation. GPU Computing Gems Emerald Edition. 2011.