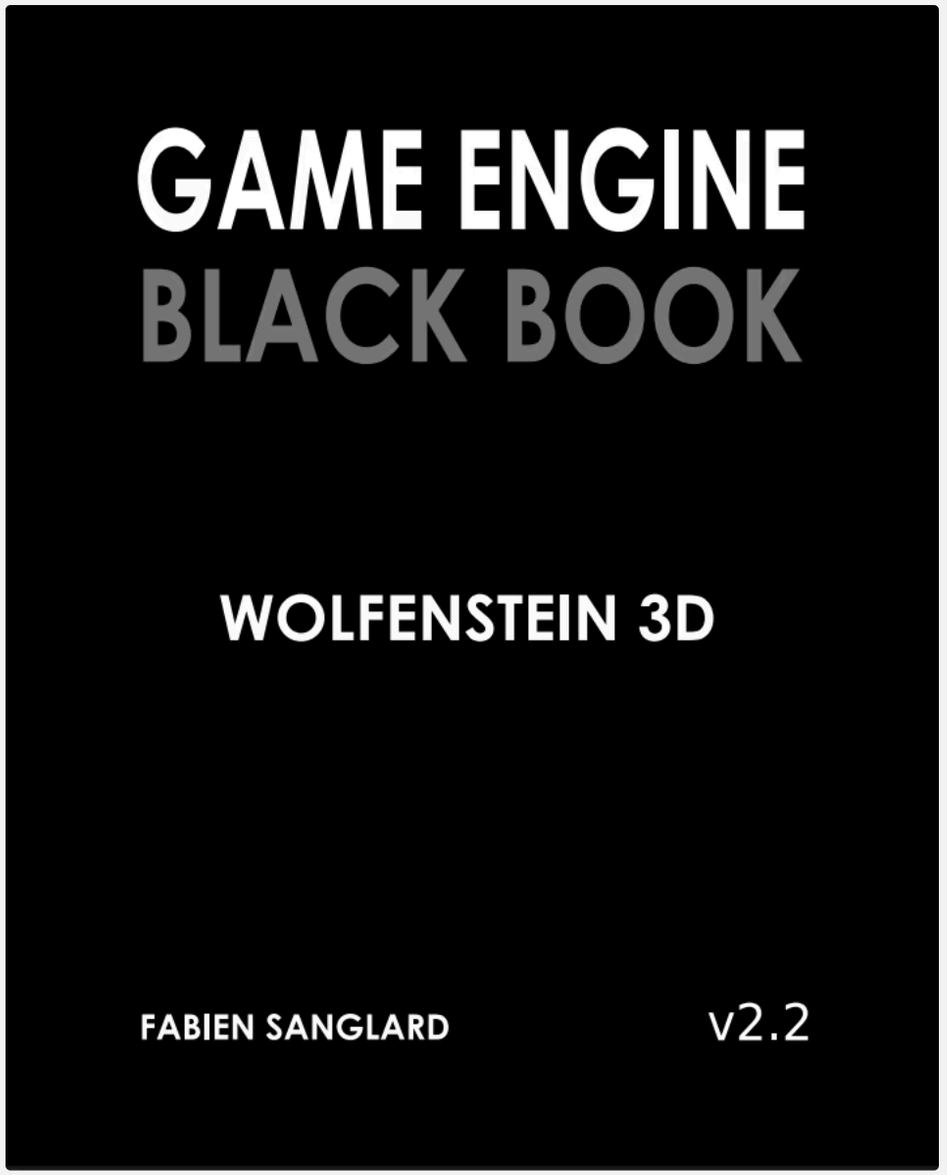
Game Engine Black Book: Wolfenstein 3D

Seções 4.1, 4.2 e 4.3 — Software

feito por Felipe Stein, Kaiky Viglioni, Marco Antonio, Nícolas Medeiros e Sérgio Alexandre



Obtenção do Código Fonte

O código-fonte do motor de jogo de Wolfenstein 3D foi disponibilizado originalmente no servidor FTP da id Software em 21 de julho de 1995. Notavelmente, o arquivo permanece no mesmo URL original décadas depois, o que é um fato raro na web.

```
ftp://ftp.idsoftware.com/idstuff/source/wolfsrc.zip
```

Atualmente, a forma mais recomendada e rápida de acesso é através do GitHub, onde a id Software migrou seus projetos de código aberto por volta de 2012.

```
$ git clone git@github.com:id-Software/wolf3d.git
```

Primeiro Contato e Estatísticas

Após a descompressão do arquivo original, o código revela uma estrutura predominantemente escrita em C, com porções críticas em Assembly. O uso de Assembly era destinado a otimizações de gargalos e operações de I/O (Input/Output) de baixo nível para vídeo e áudio.

```
$unzip WOLFSRC.1
```

Utilizando ferramentas de análise de código (cloc), observam-se as seguintes proporções:

```
$ cloc-1.64.pl WOLF3D

96 text files.
94 unique files.
27 files ignored.

-------------------------------------------------------------------
Language         files        blank      comment         code
-------------------------------------------------------------------
C++                 26         5750         6201        21169
C/C++ Header        42          802          660         3900
Assembly            10          669          732         2150
DOS Batch            1            1            0            4
-------------------------------------------------------------------
SUM:                79         7222         7593        27223
-------------------------------------------------------------------
```

Primeiro Contato e Estatísticas

Apesar de não ser sempre um fator de comparação válido, analisar o número de linhas do programa é um indicador ótimo de seu tamanho e proporção, com um total de aproximadamente 27.223 linhas de código (SLOC - Source Lines Of Code), o motor do Wolfenstein 3D é considerado pequeno para os padrões modernos.

Em comparação, o navegador Google Chrome possui cerca de 1,7 milhão de linhas, e o kernel Linux ultrapassa as 15 milhões.

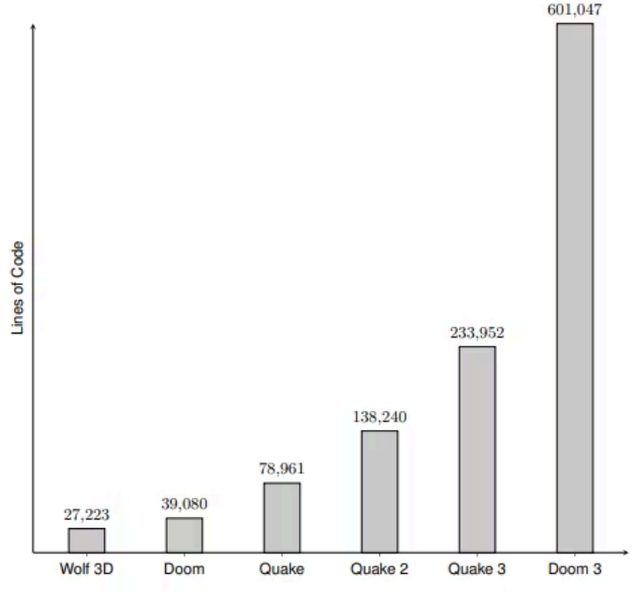


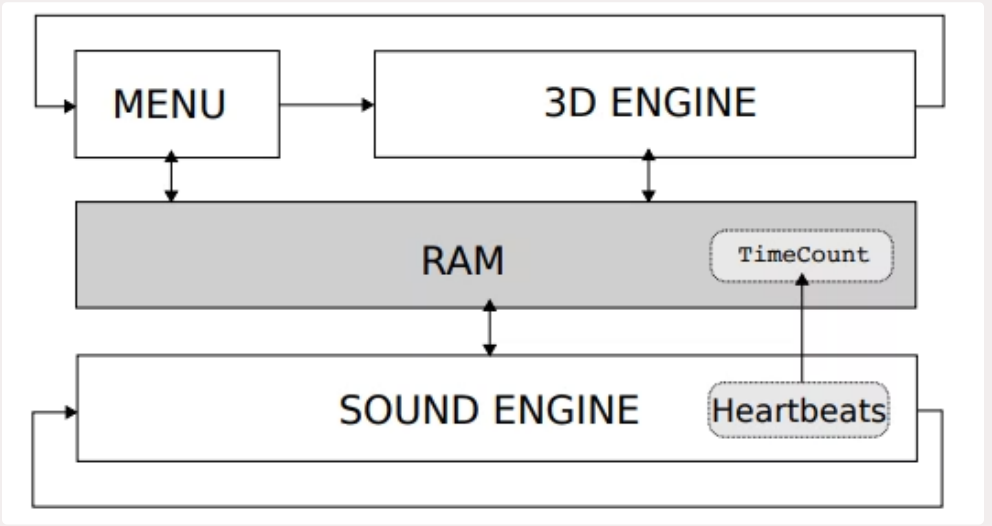
Figure 4.1: Lines of code from id Software game engines.

> Naquela época, nossos editores não tinham corretores ortográficos, e eu sempre tive uma ortografia ruim. A palavra "coluna" aparece dezenas de vezes no código-fonte. Depois que liberei o código-fonte, um dos e-mails que me marcou dizia:
>
> ’É "COLUMN", burro do C%&#!!’
>
> **John Carmack - Programmer**

A Visão Geral (Big Picture)

A arquitetura do motor é dividida em três blocos principais que se comunicam via memória compartilhada:

1. **Menu 2D: Responsável pela interface de configuração.**
2. **Motor 3D (Renderer): Onde o processamento principal ocorre.**
3. **Sistema de Som: Executado de forma concorrente aos outros dois.**



O sistema de som é orientado por interrupções, agindo como o "batimento cardíaco"do motor, controlando o tempo global através da variável TimeCount.

Otimização para 386

Embora compilado para 16 bits, o jogo detecta se a CPU é um Intel 386 através da função Patch386(). Se detectado, o motor "patcheia"seu próprio código em tempo de execução, substituindo divisões de 32 bits da biblioteca do Borland C por instruções nativas que utilizam os registradores eax e edx, resultando em um ganho significativo de performance.

```
void main ( void ){
CheckForEpisodes ();
Patch386 ();
InitGame ();
DemoLoop ();
}
```

O Loop Principal

O ponto de entrada do programa em void main() inicializa os gerenciadores e entra no loop de demonstração. Um detalhe técnico importante da época é que, por rodar em Modo Real, os tipos de dados diferem do padrão de 32 bits: int e word possuem 16 bits, enquanto long possui 32 bits.

```
void InitGame () {
 MM_Startup () ; // Gerenciador de Memoria
 VW_Startup () ; // Gerenciador de Video
 IN_Startup () ; // Gerenciador de Entrada
 SD_Startup () ; // Gerenciador de Som
 CA_Startup () ; // Gerenciador de Cache
}
```

O Sistema de Áudio e Interrupções

Como o MS-DOS não oferecia suporte a processos ou threads, a única forma de executar áudio simultaneamente ao processamento gráfico era através de ISR (Interrupt Service Routines). O motor instala uma rotina na Tabela de Vetores de Interrupção, que pode ser disparada em frequências de 140Hz a 7000Hz, dependendo do hardware de som utilizado (PC Speaker ou Sound Blaster).

```
static void SDL_SetTimerSpeed ( void ) {
 word rate ;
 void interrupt (* isr )( void );

 if (( DigiMode == sds_PC ) && DigiPlaying ) {
 rate = TickBase * 100; // 7000 Hz
 isr = SDL_t0ExtremeAsmService ;
 } else {
 rate = TickBase * 2; // 140 Hz
 isr = SDL_t0SlowAsmService ;
 }
 setvect (8 , isr ); // Instala a interrupcao
 }
```

Conclusão

A arquitetura de Wolfenstein 3D reflete os desafios de programação do início dos anos 90: a necessidade de lidar com limitações do modo real, a falta de multitarefa nativa do sistema operacional e a dependência de otimizações manuais em Assembly para garantir a fluidez em hardware limitado.

FIM

Referências bibliográficas:

SANGLARD, Fabien. **Game Engine Black Book: Wolfenstein 3D**. v2.2. [S. l.]: Fabien Sanglard, 2022.