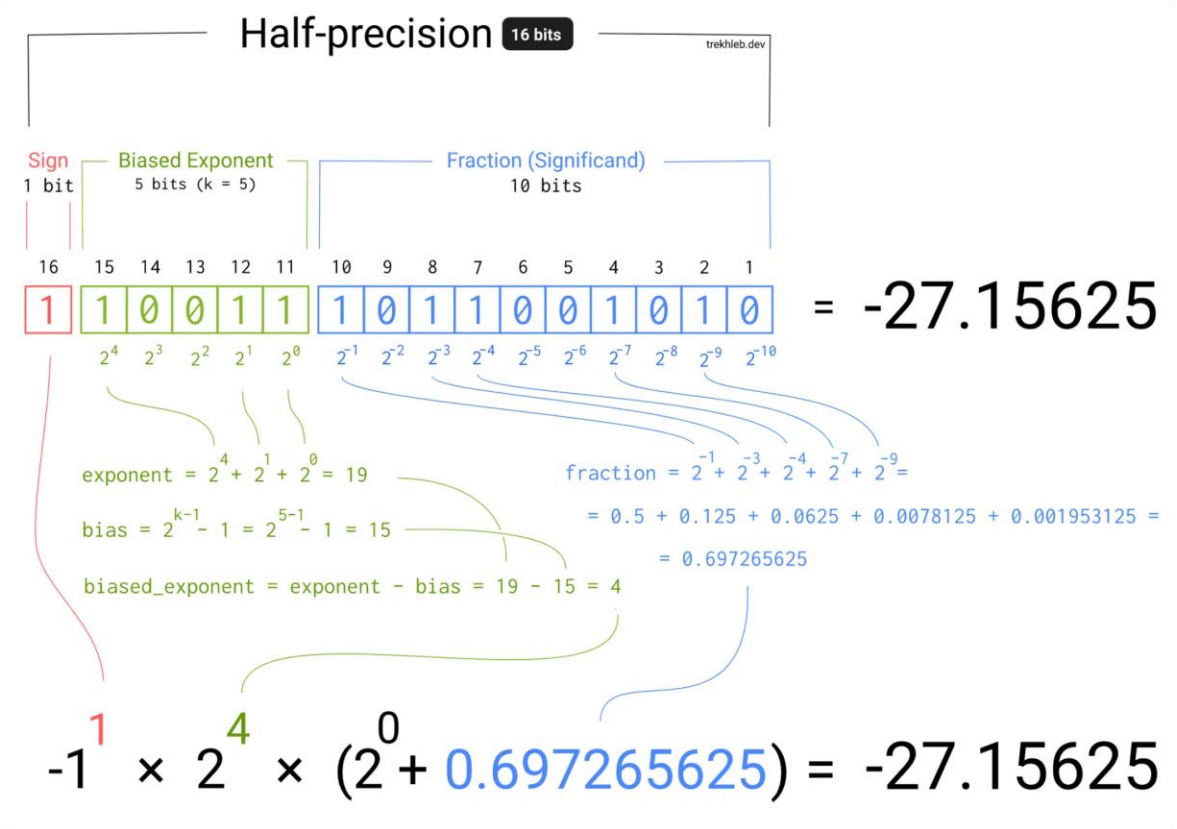
Resolvendo o Problema da CPU

Como o Wolfenstein 3D superou **limitações** de hardware usando **aritmética de ponto fixo** para operações de **ponto flutuante** rápidas.

Aritmética de Ponto Fixo

A técnica que "engana" a **ALU** para realizar **operações** de **ponto flutuante** usando apenas operações inteiras. Permite **frações** enquanto mantém a **velocidade** da CPU.

Layout 8:8: 8 bits para parte inteira, 8 bits para parte fracionária.



Como Funciona

Adição/Subtração

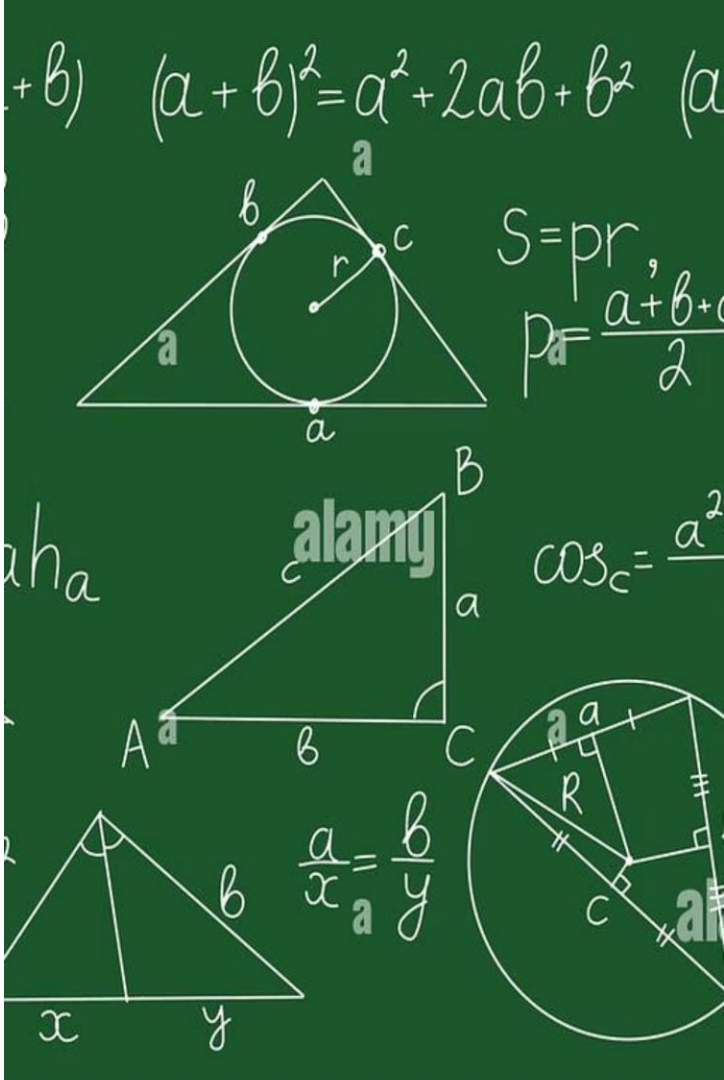
Funciona exatamente como **inteiros**

Shifts

Dividir/multiplicar por potências de 2

Multiplicação

Caso especial com **perda de precisão**



Curiosidade

O uso da aritmética de ponto fixo não se limitava aos jogos de PC. Muitos **consoles de jogos** fabricados na década de 90 e posteriormente não possuíam **unidades de ponto flutuante** para reduzir o **custo de produção** e maximizar o **desempenho do pipeline** da CPU.

O **PlayStation** original da **Sony** (1994) e o **Saturn** da **Sega** (1994) são exemplos disso.

Sistema de Coordenadas

64

Blocos

Mapa de 64x64 blocos

8

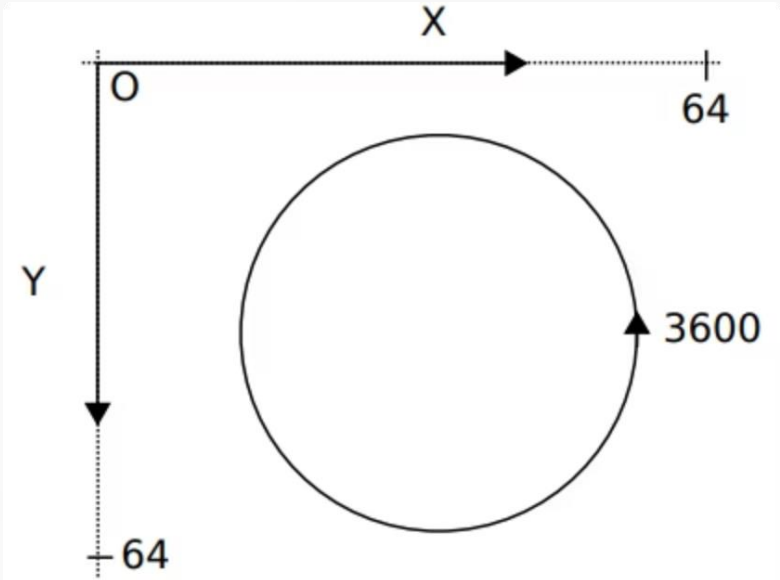
Pés

Cada bloco tem 8 pés

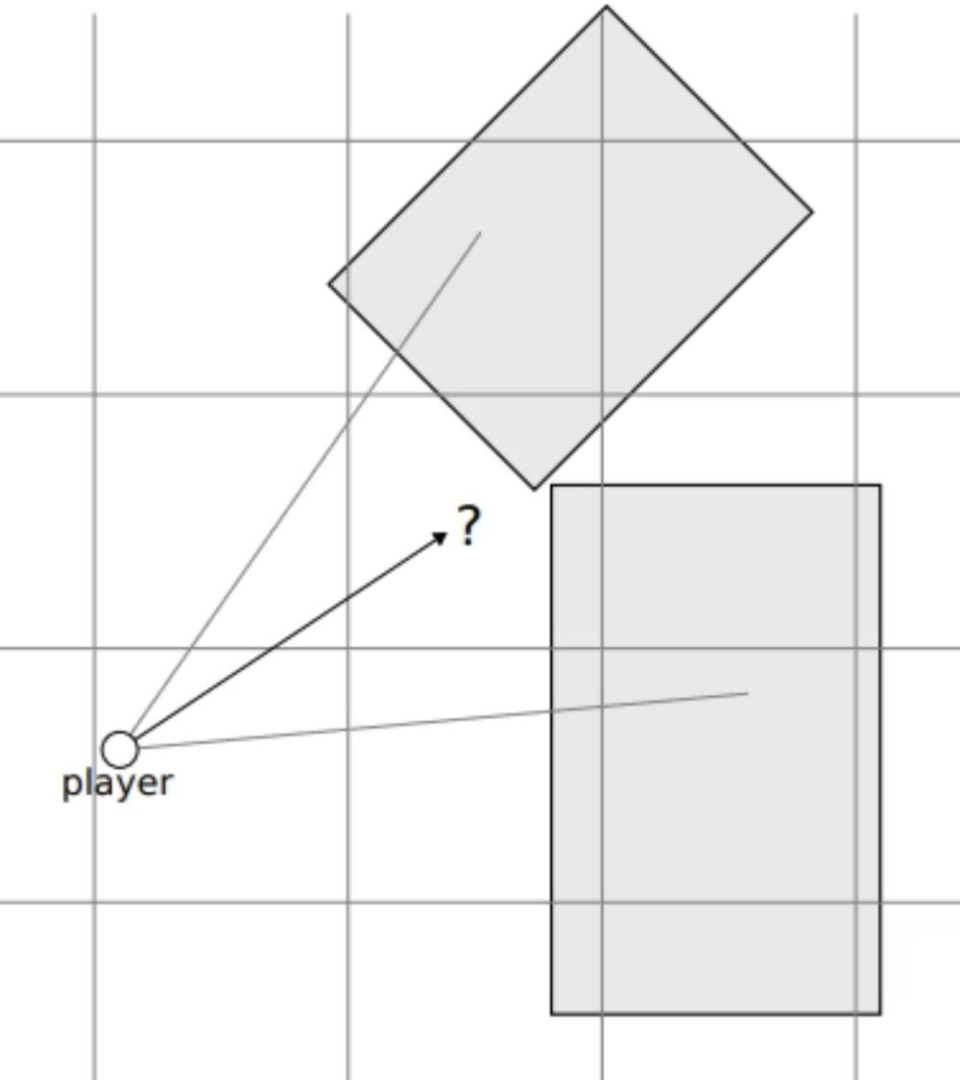
512

Pés

Mapas de 512x512 pés



Origem no canto superior esquerdo. Ângulos em int representando décimos de graus [0, 3600].



Determinação de Superfície Visível

> "O problema mais difícil de 3D: determinar qual superfície desenhar em cada pixel e descartar polígonos não visíveis."

VSD escala diretamente com complexidade da cena, tornando-se fator limitante em mundos realistas.

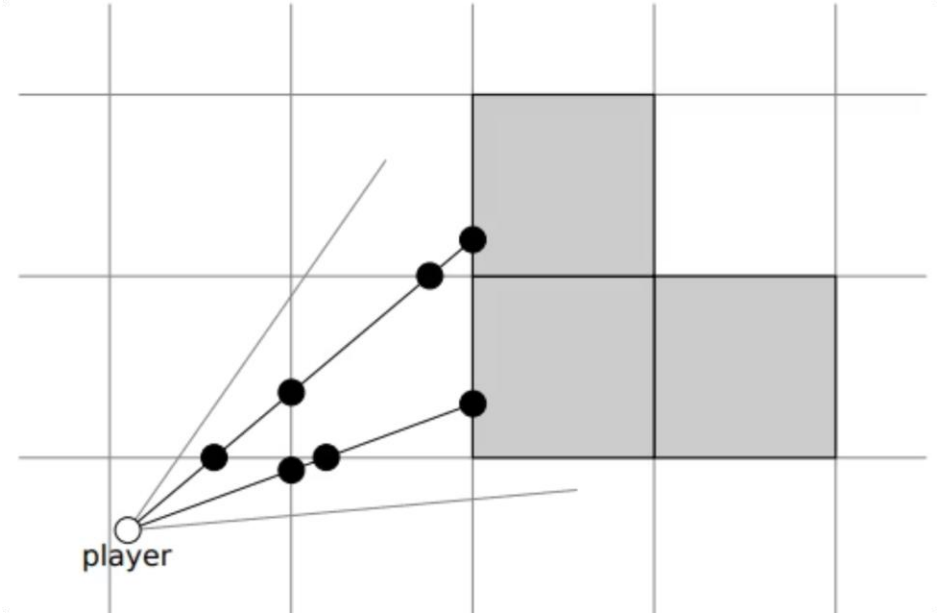
RayCasting com Grid

Restrições do Mundo

- Blocos quadrados alinhados
- Distribuição uniforme em grid
- Paredes perpendiculares 8x8x8 pés (±2,5m)
- Aparência de corredor/labirinto

Vantagens

100% de precisão com baixa sobrecarga de tempo de execução. Verifica colisões apenas quando "raio" cruza grid.



Call Apogee

O jogo possuía paredes secretas que podiam ser empurradas pelo jogador, escondendo corredores, salas secretas, tesouros e atalhos. Os jogadores normalmente descobriam essas áreas ao testar paredes suspeitas, observar diferenças nas texturas, procurar espaços vazios no mapa, ouvir sons diferentes e explorar cuidadosamente os labirintos do jogo.

Após o lançamento do jogo, muitos jogadores começaram a utilizar programas de trapaça e técnicas de engenharia reversa, que permitiam revelar o mapa completo do jogo. Com isso, tornou-se muito mais fácil localizar a sala secreta do “Aardwolf”.

•“Aardwolf” foi escolhido por ser o primeiro arquivo de imagem no dicionário das máquinas NeXT usadas pela id Software.

•O nome acabou se tornando um mascote interno da equipe de desenvolvimento.

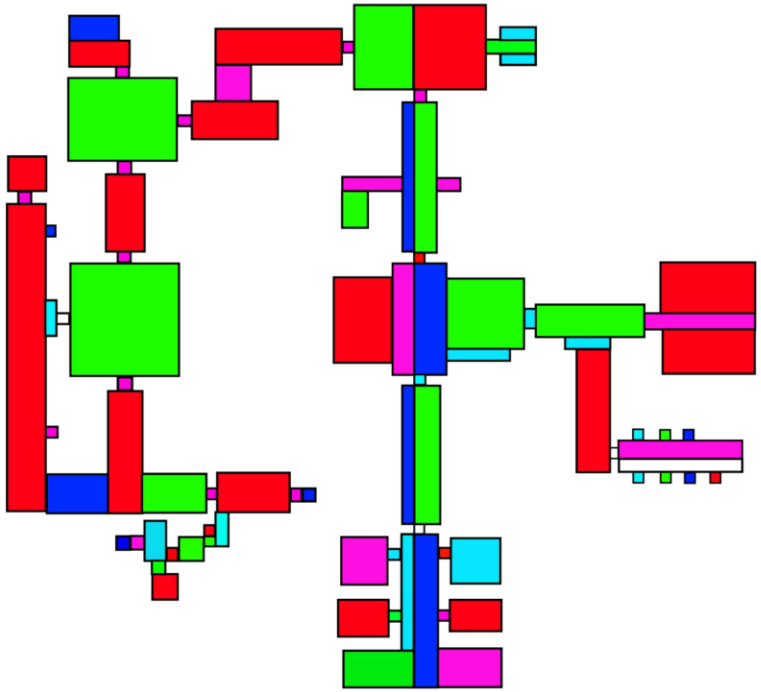
Super Nintendo vs RayCasting

PROBLEMA

- O processador 5A22 era fraco para raycasting
- A primeira versão tinha baixo FPS
- o desempenho ficou inviável

Solução: BSP (Binary Space Partitioning)

- O mapa era dividido repetidamente em regiões menores para acelerar a ordenação dos elementos da cena
- As paredes eram desenhadas “de perto para longe”
- Reduziu o processamento necessário e melhorou a taxa de quadros do jogo.
- O BSP se tornou uma das bases das futuras engines 3D.



DDA – Digital Differential Analyzer

Para cada coluna da Tela

Calcular ângulo do raio e vetores de direção; inicializar variáveis de passo (xstep, ystep)

Enquanto a parede não for encontrada

Identificar interseção mais próxima

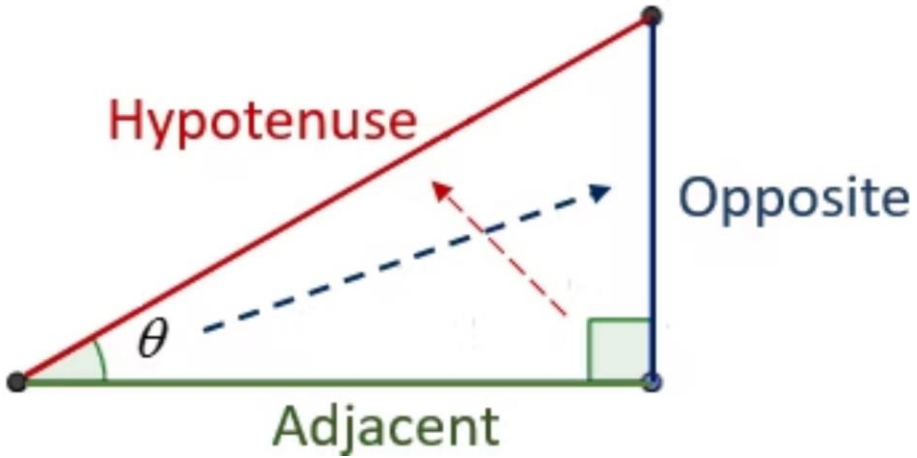
Saltar para a fronteira da grade(horizontal ou vertical)

Verificar se a célula atingida é uma parede ou uma porta

Base Matemática

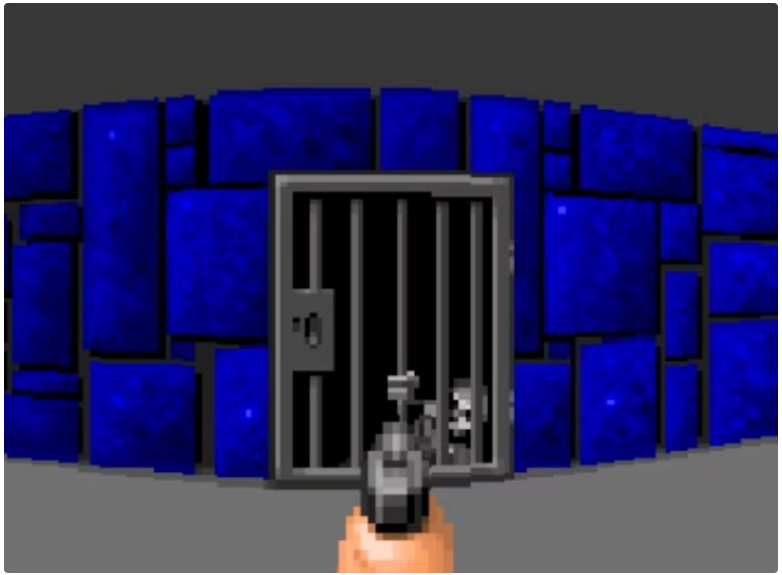
Cálculo do deslocamento Raio

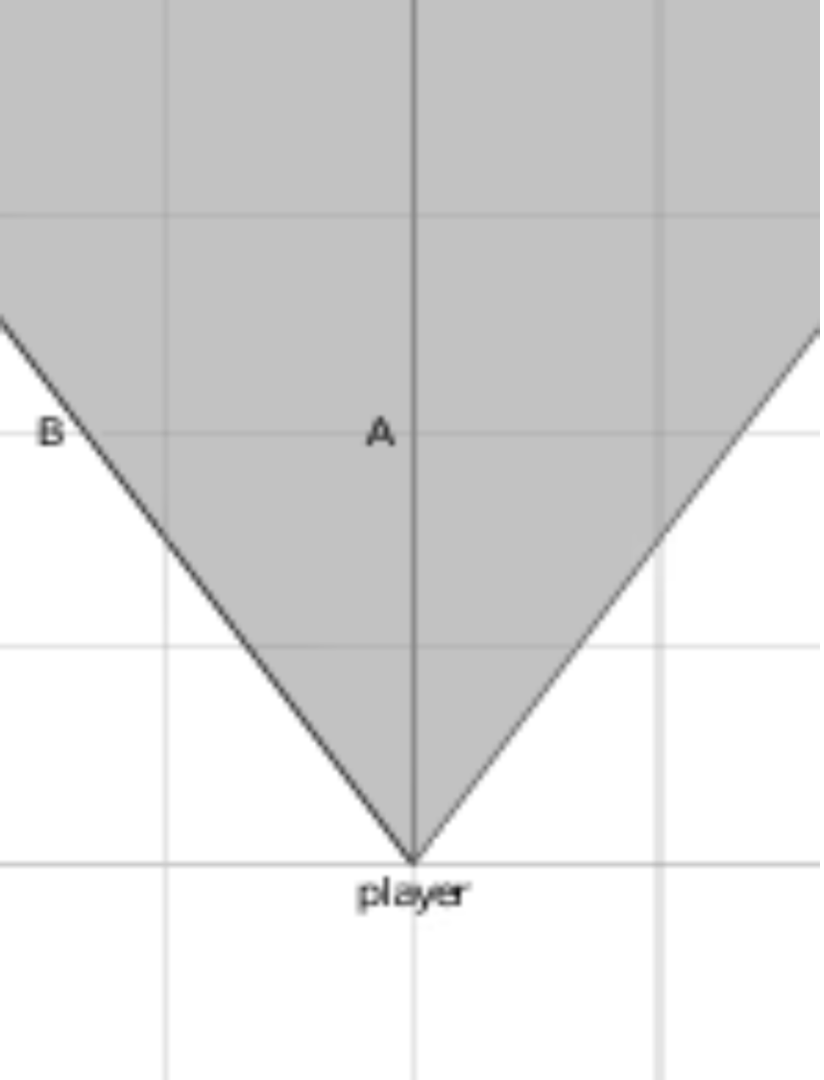
- 1 unidade em x; tan(θ) em y
- 1 unidade em y; tan(90° - θ) em x
- Uso de uma tabela para agilizar o cálculo trigonométrico (capítulo 4.11.1)



Efeito Fish Eye

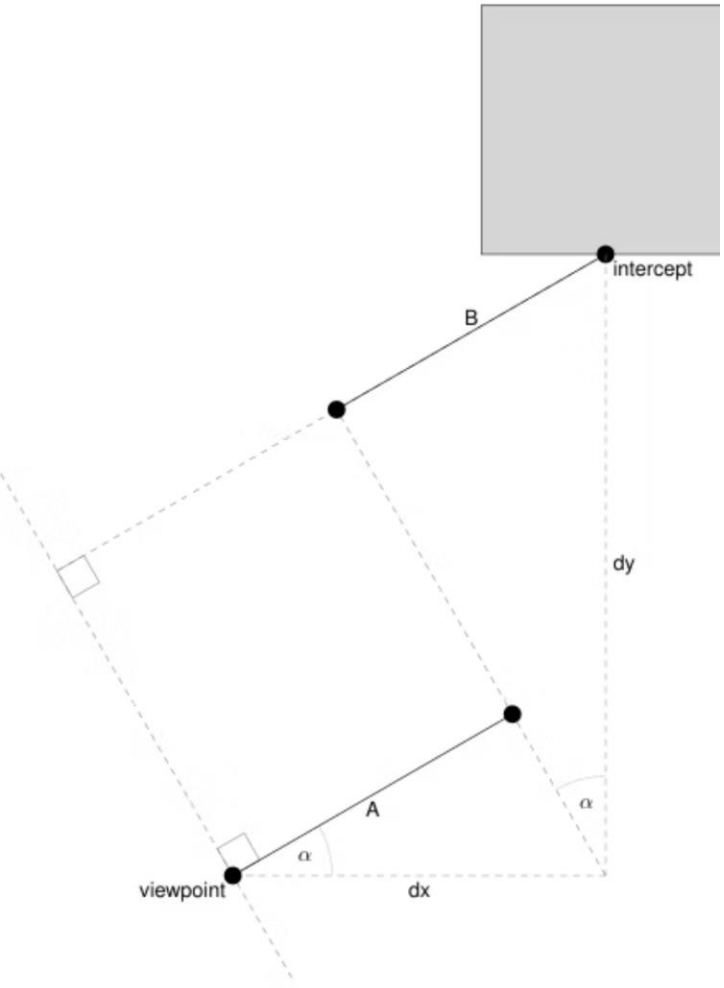
Distância euclidiana gera fish eye pois considera a projeção da cena na câmera, gerando uma percepção distorcida.





Efeito Fish Eye

A distância euclidiana mede corretamente o caminho do raio até a parede, porém os raios laterais percorrem distâncias maiores que os centrais. Como resultado, o motor interpreta as paredes laterais como mais distantes, renderizando-as menores e causando a distorção conhecida como efeito *fish-eye*.



Cálculo de Altura da Coluna

01

Componente A

$A = dx \cdot \cos(\alpha)$

02

Componente B

$B = dy \cdot \sin(\alpha)$

03

Distância Projetada

$z = A + B = dx \cdot \cos(\alpha) - dy \cdot \sin(\alpha)$

A equação calcula a distância projetada da parede em relação à câmera do jogador, considerando apenas o que está à frente da visão e ignorando o deslocamento lateral dos raios diagonais.

Resultado Final

Desempenho

Algoritmo otimizado em assembly

Linhas Retas

Correção elimina distorção fisheye

Precisão

100% de acurácia matemática