UNIVERSIDADE SANTO AMARO

CURSO DE ENGENHARIA DE SOFTWARE

Kelvin Moraes Souza

Problemas de Otimização da Unreal Engine: Um Estudo Sobre o Desempenho em Jogos Digitais

São Paulo

2025

Kelvin Moraes Souza

Problemas de Otimização da Unreal Engine: Um Estudo Sobre o Desempenho em Jogos Digitais

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Software da Universidade Santo Amaro - UNISA, como requisito parcial para obtenção do título Bacharel em Engenharia de Software.
Orientador: Prof. Msr. Angelo Luiz da Cruz Oliveira.

São Paulo

2025

Kelvin Moraes Souza

Problemas de Otimização da Unreal Engine: Um Estudo Sobre o Desempenho em Jogos Digitais

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Software da Universidade Santo Amaro - UNISA, como requisito parcial para obtenção do título Bacharel em Engenharia de Software.
Orientador: Prof. Msr. Angelo Luiz da Cruz Oliveira.

São Paulo, ___ de _____ de _____

Banca Examinadora:

Prof. Me. Angelo Luiz da Cruz Oliveira - Orientador
Prof. Me. Rafael Guem Murakami
Prof. Me. Nardo Gonçalves dos Santos

Conceito Final

_____

RESUMO

Os jogos digitais é uma parcela grande do mercado de desenvolvimento de software, com a evolução da tecnologia era esperado que esse meio de entretenimento se tornasse mais acessível, graças às novas técnicas de desenvolvimentos que visam facilitar o processo de criação desses softwares. Porém foi observado o oposto, com os últimos lançamentos está cada vez mais comum surgirem jogos com má otimização, e em especial aqueles desenvolvidos na Unreal Engine. Esse trabalho tem como objetivo compreender as possíveis causas e com isso, verificar possíveis soluções para tal problema.

Palavras-chave: Desenvolvimento de jogos, Desempenho de jogos, Otimização de jogos, Unreal Engine, Engine de jogos.

ABSTRACT

The digital games is a big scale of the software development market, with the technology evolution is expected then this means of entertainment be more accessible, thanks to the new development techniques with objective to make more easy the task of make these softwares. But it is the same opposite, with the latest releases it is more common to see games with bad optimization, and especially those developed in Unreal Engine. These works have the objective to understand the possible causes, and with this, verify the possible fixes to this problem.

Keywords: Game development, Game performance, Game Engine, Game optimization, Unreal Engine.

Listas de ilustrações

SUMÁRIO

1. INTRODUÇÃO

Com a ascensão dos videogames como uma das maiores indústrias de entretenimento, passando até mesmo do cinema, a cada dia mais pessoas vêm se interessando e entrando para esse mundo, seja por lazer ou profissionalmente. LIA (2020) trás segundo a análise de mercado, utilizando dados fornecidos pela NEWZOO, que a indústria Brasileira de Jogos Digitais apresentou crescimentos entre 2014 e 2018, registrando mais de U$1,3 Bilhão de movimentação no Brasil, englobando compra de jogos e dentro dos mesmos, e compará, que no mesmo período a indústria cinematográfica arrecadou aproximadamente U$830 milhões.

Devido a essa crescente, um tema que vem se popularizando bastante é a discussão sobre otimização de jogos. A cada lançamento, vem o questionamento: “Será que vou ter o *hardware* necessário para jogar de forma satisfatória?”, sendo uma dúvida bem mais frequente entre usuários de computadores, por sua variedade de combinações de peças, que sempre vem associada a questões socioeconômicas. Segundo AZSAN (2025), os requisitos de um jogo são como uma barreira monetária para o consumidor atribuída ao *hardware* e afetando o acesso, sendo jogos bem otimizados, gerando mais facilidade de acesso para uma maior audiência.

No cenário socioeconômico do Brasil, essa discussão se torna ainda mais relevante devido aos custos do *hardware*. O preço médio para um computador de entrada já é elevado, sendo os topos de linha irreais para a maior parte da população. A partir do momento que os requisitos mínimos se tornam maiores e mais distantes dos *hardwares* de entrada, acontece a exclusão da maior parte dos consumidores.

Ao mesmo tempo que os custos de produção e tempo do desenvolvimento de jogos cresce, o cuidado para um produto bem polido e otimizado diminui. Com o avanço da tecnologia, foram criadas *engines*, conjuntos de componentes de software com funcionalidades e ferramentas para desenvolvimento de um processo específico, que ajudam ao desenvolvedor começar seu projeto com uma base pré-estabelecida de uma forma mais rápida e sem precisar desenvolver a sua própria, porém, muitas das ferramentas podem ser utilizadas de forma incorreta, gerando perda de performance e *bugs*.

Nesse contexto, essa pesquisa é focada na Unreal Engine, por ser uma das maiores e mais utilizadas *engines* do mercado e ter muitos lançamentos recentes apresentando problemas de otimização, buscando entender o que leva a esse problema, descobrir se é causado pela *engine* ou pelo desenvolvimento e buscar possibilidades de melhorias e boas práticas para melhorar ou resolver o problema.

2. OBJETIVOS

2.1 Objetivo geral: Analisar os problemas de otimização na unreal engine em jogos digitais.

2.2 Objetivos específicos:

- Analisar ferramentas da Unreal Engine e seus possíveis impactos na performance;
- Investigar as tecnologias atuais para o desenvolvimento de jogos;
- Analisar os jogos digitais atingidos pelos problemas de desenvolvimento na Unreal Engine;
- Verificar se o mercado utiliza a engine de forma otimizada;
- Buscar possíveis melhorias para otimização de jogos desenvolvidos na Unreal Engine.

3. METODOLOGIA

A metodologia adotada foi a da pesquisa bibliográfica, de caráter exploratório. Realizou-se leituras exploratórias, identificando relações com os conceitos de otimização de *softwares*, *engine* de jogos e Unreal Engine, para posterior leitura analítica em busca de informações que auxiliem na compreensão e construção de possíveis soluções para o problema identificados.

As informações foram coletadas por meio de livros, artigos científicos, notícias de canais fidedignos e documentação referente ao software estudado. Também foi utilizado matérias de sites de notícias da área de tecnologia e vídeos com informações visuais e exemplos práticos.

4. DESENVOLVIMENTO

Para dar início é necessário destacar que o *software* a ser avaliado possui diferentes versões, estando atualmente na sua quinta edição, Unreal Engine 5,

sendo essa a versão com maior relatos dos problemas a serem estudados, segundo a matéria de WEIZENMANN (2025):

> Com a Unreal Engine 5, a Epic Games criou uma engine com recursos que são indiscutivelmente avançados demais para o hardware de GPU atual. Cada nova versão da Unreal Engine 5 promete desempenho otimizado em comparação com as versões anteriores.

Porém não foram descartadas informações referente à versão anterior do *software*, se confirmado que essas continuam sendo válidas para a versão atual, sendo assim, dados de artigos referentes a Unreal 4 são comparados com a documentação e informações disponíveis da Unreal 5, caso se prove inalterado ele é considerado válido.

Também é válido salientar que, devido a escassez de estudos científicos de processos específicos da indústria de jogos e com o fim de complementar as informações, serão analisadas literaturas cinzentas como entrevistas, teses, dissertações e estudos etnográficos.

4.1. Função de uma Engine para desenvolvimento de jogos

Ao iniciar o desenvolvimento de um jogo, é comum que os desenvolvedores escolham a utilização de uma *engine* para o desenvolvimento, sejam proprietárias como a Capcom e sua RE Engine ou terceiras como a Unity, Godot, Unreal ou outras.

Essa escolha impacta diretamente em aspectos do desenvolvimento e do produto final, como qualidade gráfica, eficiência do desenvolvimento, requisitos mínimos e outros diversos fatores.

Em seu livro, THORN (2011) compara que a engine é para o jogo e seu desenvolvedor, como o coração para um corpo. Ou seja, é uma parte essencial, mas que por si só, não representa o todo. O autor traz como exemplo o processo de fazer um sucessor de um jogo aclamado, comentando que muitas das lógicas, como a física ou o código que faz um efeito sonoro em uma ação específica, poderiam ser reutilizadas ao invés de refeitas do zero. JUNGHERR e SCHLARB (2022) descrevem essa tecnologia como produtos de *softwares* que oferecem ferramentas e soluções pré estabelecidas para tarefas fundamentais de programação no processo de desenvolvimento de jogos e ambientes virtuais.

Pode-se concluir então, que a *game engine* é uma espécie de facilitador do

processo, cortando a necessidade de criar do zero programações comuns e repetitivas, GREGORY (2009) identifica diversos processos de desenvolvimento de jogos que são administráveis pelas *game engines*, alguns deles são: Gráficos, colisão, físicas, animações de personagens e inteligência artificial.

4.1.1. Porque implementar uma Engine

Hoje em dia as empresas de jogos não precisam passar pelo processo de criar do zero ou manter atualizada suas engines, já que outras empresas se especializaram no desenvolvimento e aprimoramento desses *softwares*, como a Epic Games com a Unreal Engine. Segundo JUNGHERR e SCHLARB (2022) ao desenvolver, fornecer e manter a infraestrutura e padrões para o desenvolvimento, as empresas de *engine*, tem contribuído para o grande sucesso comercial da indústria de jogos.

SILVA e RODRIGUES (2024) comentam que a escolha de uma *engine* estabelecida no mercado pode influenciar o tempo e custos para o desenvolvimento, podendo assim reduzir o prazo para finalizar o mesmo. Concluindo então, que ao utilizar uma *engine* de terceiros as empresas economizam tempo e dinheiro, se isentando de ter que desenvolver e dar manutenção á esses *softwares*. Ao analisar os fatores para a performance de mercado dos jogos digitais, ALEEM et al (2016, p. 2, tradução nossa) ressalta: “As organizações do setor de jogos digitais precisam responder rapidamente às mudanças no ambiente de negócios e tecnológico, e se não responderem adequadamente, não sobreviverão por muito tempo”. Afirmação essa que aponta para um dos motivos da implementação desses *softwares* de terceiro, visto que as *game engines* estão sempre acompanhando as novas tecnologias e tendências de mercado.

Essa prática de uso de *engines* de terceiros pode abrir margem para o mal uso, por falta de conhecimento técnico ao utilizar o *software* ou uso de ferramentas que não foram devidamente feitas com as necessidades daquela empresa, podendo não atender às demandas específicas da forma mais otimizada possível. Porém seria injusto falar que o mesmo não acontece com engines proprietárias, segundo GUGELMIN (2025):

> Monster Hunter Wilds está se provando um jogo muito popular que quebrou todos recordes anteriores da série no PC. Ao mesmo tempo, a versão para a plataforma tem sido apontada como problemática e chegou marcada por problemas de desempenho e

crashes frequentes.

Isso em um jogo desenvolvido com a RE Engine, que é de autoria da própria empresa e mesmo assim, entregou seus dois últimos lançamentos com problemas de performance, sendo eles, Monster Hunter Wilds e Dragon's Dogma 2. O que abre um alerta para investigar não somente problemas de otimização na Unreal, mas também, comparar com outras *engines* do mercado e descobrir um possível padrão de desenvolvimento nos tempos atuais.

4.2. Analisando a Unreal Engine

Para definir, inicialmente, os principais problemas que levam aos jogos da Unreal Engine serem reconhecidos por terem uma otimização insatisfatória e requisitos altos de *hardware*, foi elaborado um levantamento abordando otimização no desenvolvimento de jogos feitos pela *engine* em artigos e notícias. Também foi desenvolvida uma análise minuciosa da documentação do *software*, avaliando a clareza, quantidade de informações e ferramentas junto com os processos de otimização dessas.

Para a EVANSON (2025), ao ser questionado em um evento, o Diretor Executivo Tim Sweeney da Epic Games, empresa responsável pelo desenvolvimento da *engine*, comenta que o motivo principal para os jogos desenvolvidos na Unreal Engine 5 não desempenharem bem em certos computadores ou placas de vídeos é o processo de desenvolvimento, segundo ele, a maioria das empresas desenvolvem seus jogos com base em computadores de topo de linha, só se atentando para testar em hardwares mais fracos durante a etapa final do processo de desenvolvimento. Afirmação essa que abre margem para a investigação dos processos de desenvolvimento da indústria de jogos.

4.2.1. Documentação e Ferramentas da Unreal Engine

Após a análise da documentação, puderam ser observados vários tópicos de otimização, não só em termos gerais de lógicas de programação e boas práticas, como também para tópicos específicos, por exemplo, para otimizar as ferramentas disponíveis como a de efeitos visuais "Niagara" que conta com conteúdos de *debug* e otimização. Esses tópicos são anexados a subtópicos através de *links,* oferecendo várias camadas para o desenvolvedor poder se aprofundar em busca de

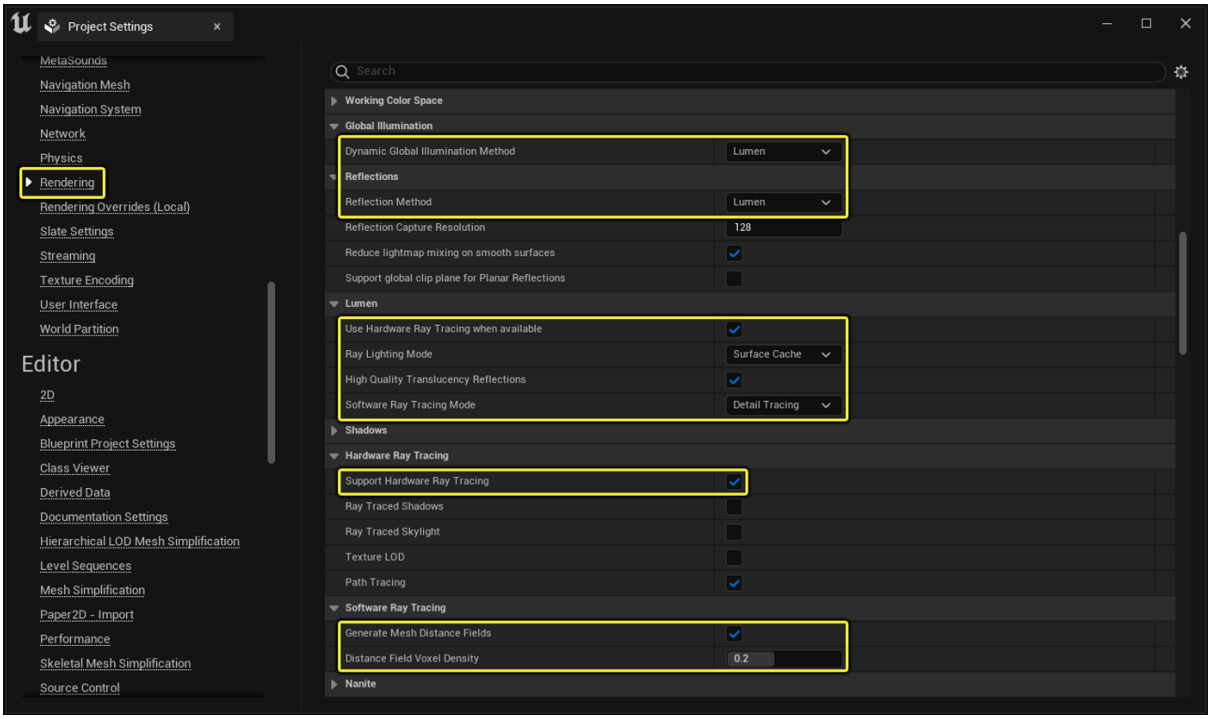
conhecimento e boas práticas. Possui uma interface de fácil navegação e explicações acompanhadas de imagens para detalhar e exemplificar. Apesar disso, é necessário a visão aprofundada de um desenvolvedor para atestar que na documentação consta todos os conteúdos necessários para a otimização nas diversas etapas de produção de um *software*.

O que pode ser verificado é as diferentes ferramentas que a *engine* oferece para otimizar o desenvolvimento do jogo e como é abordada a otimização dessas, já que por ser uma ferramenta que necessita de conhecimento técnico, podem ser um grande problema de otimização se mal implementadas e por não ser de autoria do desenvolvedor é necessário o aprendizado junto a uma boa documentação. Em sua tese, SATALIN (2024) faz a comparação da ferramenta de sistemas de luzes complexas da Unreal com a vida real, dizendo que ambas são caras, mas no caso de computadores, são caras em termos de processamento, precisando de conhecimento para usá-las de forma efetiva. Consultando a documentação pode ser destacada algumas ferramentas e suas funções.

4.2.1.1. Lumen

Lumen é uma ferramenta de renderização de iluminação global e reflexos, que utiliza da propagação da luz dinâmica para simular o contato da luz com os objetos presentes no ambiente. Ela conta também com a opção de *ray tracing*, tecnologia que no momento já apresenta uma barreira de acessibilidade e performance, já que só está disponível em gerações de placas de vídeos lançadas a partir dos últimos seis anos, apresentando performance satisfatória predominantemente para as opções de topo de linha, como será abordado a frente (Epic Games, 2025). A documentação traz as configurações do Lumen, como sua implementação para a renderização da iluminação global e reflexos, além de possibilidades de personalização, com os devidos impactos na performance, como visto na Figura 1.

Figura 1 – Configurações do Lumen



Fonte: Página de Epic Games - Lumen Global Illumination and Reflections. (2025) https://dev.epicgames.com/documentation/en-us/unreal-engine/lumen-global-illumination-and-reflections-in-unreal-engine

Segundo SOBCHYSHAK et al. (2025) em sua análise técnica da ferramenta, foi feito testes de *benchmarks* notando que a propagação de luz dinâmica da Lumen, custa de 15 a 20 por cento de performance em um computador intermediário, ressaltando que isso faz da otimização de *hardware* algo crucial. Essa porcentagem para *hardwares* de entrada pode ser o suficiente para tornar a experiência do jogo insatisfatória, como será abordado mais à frente.

4.2.1.2. Nanite

Outra ferramenta é o Nanite, que é o sistema de geometria virtualizada da Unreal, permite *renderizar* objetos 3Ds com alta quantidade de pixels e detalhes, com seu formato de arquivo altamente comprimido. Um de seus benefícios é que ele aplica automaticamente a técnica de *Level of Detail* (LOD), que se conciste em uma forma de otimização gráfica onde os modelos 3Ds são simplificados conforme a câmera se afasta deles, fazendo assim que diminua a carga do processamento, e por isso, um melhor desempenho. Na documentação é recomendado seu uso sempre que possível, apresenta como ele pode ser utilizado e suas configurações. Sendo assim, essa ferramenta permite a criação de cenários bem detalhados sem

afetar a performance, oferecendo praticidade para o desenvolvedor que não precisa manualmente aplicar o LOD e reduz o uso de memória pela sua compressão (Epic Games, 2025). Porém segundo LI (2024) é levantado uma limitação em relação à demanda de *hardware* ser muito alta, causado pela necessidade de muitos cálculos e transmissões de dados, o que vai de contra a afirmação da documentação de fornecer um melhor desempenho, levantando a possibilidade de ser uma implementação para cenários com *hardwares* específicos de maior capacidade de processamento, sendo assim, de maior performance e custo.

> O Nanite obteve um bom resultado ao otimizar o número de triângulos e chamadas de desenho. No entanto, em termos de desempenho, o Nanite é muito inferior às configurações padrão.
> Na configuração padrão, o QPS da câmera permanecia estável em 120 QPS, praticamente sem alterações. Porém, após a desativação do sombreamento direto, o QPS médio caiu drasticamente para cerca de 50. Ativar o Nanite não resolve o problema, mas o agrava. O FPS médio caiu mais 10. LI (2024, p. 34, tradução nossa).

4.2.1.3. Virtual Shadow Maps

Trabalhando como um complemento das tecnologias anteriores, o Virtual Shadow Maps é um método de mapeamento de sombra que trabalha em conjunto com a Nanite e Lumen para entregar um sombreamento de alta qualidade, com facilidade de implementação por necessitar de poucos ajustes e segundo a documentação com custos de desempenho razoáveis e controláveis (Epic Games, 2025). Já nos testes feitos pelo Unreal DevOP (2022) pode ser observado perdas de performance ao utilizar o Virtual Shadow Maps sem a implementação do Nanite, tornando assim uma ferramenta situacional, e dependente de outra para a melhor implementação com custo de performance considerável.

4.3. Performance

A métrica utilizada para medir a performance de um jogo são os quadros por segundos (QPS), que mede a taxa de atualização da imagem a ser exibida, influenciando na percepção de fluidez do usuário. Comumente em computadores é atribuída a faixa de 60 QPS como base para boa fluidez e meta a ser alcançada para uma melhor experiência, sendo os 30 QPS uma experiência aceitável em casos de maior demanda de hardware e a depender da cadência do jogo, tendo a experiência comparável à cinematográfica.

Uma boa performance é alcançada quando um computador atende aos

requisitos necessários para lidar com as exigências de determinado *software*, sendo então necessário que ao desenvolver tal programa, haja uma atenção em fazer com que os recursos computacionais sejam o mais bem otimizados possíveis, garantindo assim uma maior compatibilidade com diferentes *hardwares*.

Como visto anteriormente, diversas ferramentas da Unreal Engine têm potencial para melhorar a performance, assim como a capacidade de prejudicá-la em determinados cenários. Porém essa responsabilidade não está limitada às ferramentas que estão sendo disponibilizadas para facilitar a produção do *software*, diversas outras práticas no processo de desenvolvimento podem influenciar como determinado jogo pode performar, como as novas tecnologias de jogos.

4.3.1. Novas tecnologias de desenvolvimentos

Nos últimos anos saíram diferentes tecnologias com o intuito de melhorar a performance de jogos em hardwares de menor potência, sendo elas o *upscaling* e *frame generation*. Essas inovações foram anunciadas com um intervalo de tempo e em diferentes gerações de placas de vídeos, tanto pela AMD, quanto pela Nvidia. Em questão de *upscaling*, do lado na Nvidia foi introduzido a partir da série RTX 20 o Deep Learning Super Sampling Super Resolution (DLSS), exclusivo para essas placas e suas próximas gerações. No lado da AMD foi lançada para competir o FidelityFX Super Resolution (FSR), que diferente da tecnologia da Nvidia, essa não tem exclusividade de placas, funcionando até para a sua concorrente.

Alguns anos depois, foi lançado por parte da Nvidia como novidade para sua série RTX 40 e funcionando também para as posteriores e a anterior, RTX 30, o frame generation, mais uma vez a AMD anunciou a própria tecnologia, que novamente não tem exclusividade de hardware da marca.

4.3.1.1. Super Resolution e FidelityFX Super Resolution

Dentre as duas tecnologias citadas, os *upscaling* são os protagonistas em quesito de desempenho, não só por estarem em circulação a mais tempo, mas também, por englobar uma maior disponibilidade em placas de vídeo do mercado.

Segundo a NVIDIA (2025a) o DLSS aumenta a performance escalando uma imagem de baixa resolução com inteligência artificial para entregar uma de alta resolução. Na prática isso significa que ao configurar um jogo em uma resolução como 1080p e ativar o *Super Resolution*, o mesmo estará sendo renderizado em

uma resolução menor, como 720p, e o *upscaling* faz a conversão para a resolução escolhida inicialmente, quase sem perda de qualidade. Seguindo a lógica computacional, quanto menos resolução tem um *software* gráfico, menor será seu uso de *hardware*, sendo o *upscaling* ideal até mesmo para telas maiores como as 4k, sem abrir mão da performance que resoluções como essa requisitam.

Sendo assim, a técnica de *upscaling* consegue em teoria melhorar o desempenho e a qualidade de imagem de jogos. Desde o seu lançamento tanto o FSR, quanto o DLSS já passaram por inúmeras atualizações e versões, estando atualmente em um patamar onde é quase que obrigatório o uso dessas tecnologias para uma melhor experiência, SEN e BHUSHAN (2024, p. 1, tradução nossa) comentam que:

> Com a crescente demanda por maior fidelidade gráfica, surgiu a necessidade de diferentes técnicas de renderização em tempo real. À medida que as demandas computacionais para renderização gráficas de alta qualidade, com grandes quantidades de polígonos aumentava, surgiram os métodos modernos de upscaling. Esses upscales podem pegar a imagem base e aumentar para resoluções maiores em tempo real. Isso oferece um grande impacto na taxa de quadros da renderização em tempo real.

Ambas as tecnologias concorrentes apresentam diferentes funcionamentos e vantagens, SEN e BHUSHAN (2024) explica as diferenças em sua comparação entre FSR e DLSS, a tecnologia da AMD utiliza a técnica tradicional de *upscaling*, sendo o algoritmo do FSR capaz de produzir uma imagem de alta qualidade aperfeiçoando a experiência do usuário enquanto mantém a performance suave. Já para Nvidia explica que é utilizado o aprendizado profundo, utilizando de várias técnicas de inteligência artificial para o aprimoramento da imagem, apesar de entregar resultados superiores em comparação com os métodos tradicionais, eles vem com um custo computacional significativo.

Enquanto a Nvidia utiliza *hardware* dedicado para reconstruir a imagem com seus núcleos de inteligência artificial, a AMD utilizava de *software*, isso até o anúncio da sua nova série 9000, onde foi lançado exclusivamente o FSR 4 para competir diretamente com o DLSS, utilizando também o aprendizado profundo (AMD, 2025). Sendo assim, as versões anteriores do FSR não chegam ao nível de performance e qualidade gráfica do DLSS, porém tem a vantagem de poder ser utilizado por GPUs mais antigas e de outras fabricantes.

4.3.1.2. Frame Generation

A segunda tecnologia é o *frame generation*, sua função é aumentar os quadros base de um jogo, gerando mais fluidez. No lado da Nvidia essa tecnologia foi implementada no lançamento das RTX 40, porém também está presente na anterior, RTX 30 (NVIDIA, 2025b). Na AMD elas estão presente a partir da série 6000, tendo suporte também para placas de outras produtoras do mercado (LAIRD, 2023).

Essa tecnologia utiliza a interpolação de quadros, gerando um quadro falso entre outros dois quadros verdadeiros, porém na prática essa tecnologia é situacional, BAO et al. (2019) descreve que usando essa técnica, vídeos com alta taxa de quadros conseguem evitar artefatos visuais. O problema acontece na situação oposta, quanto menos quadros base se tem, maior os defeitos visuais e a latência, dando assim o feedback visual mais fluido, porém comandos com maior tempo de resposta e uma qualidade visual inferior. Com base nessa afirmação, pode-se concluir que essa tecnologia é voltada para *hardwares* que já se sairiam bem em um jogo conseguir um adicional de fluidez. Não sendo então, uma tecnologia para otimização, pelo contrário, já que conta com maior uso da placa de vídeo.

4.3.1.3. Ray Tracing

A tecnologia de *ray tracing* foi lançada anteriormente às duas citadas acima, é uma afirmação plausível que as tecnologias já comentadas vieram em prol de tentar tornar essa mais viável, já que o DLSS foi lançado para mitigar o impacto de desempenho gerado por ela, MALICK et al (2025) explora a aplicação do DLSS para otimizar o *ray tracing* disponibilizado pela Unreal Engine e é observado uma melhoria significativa de performance, os mesmos comentam que muitas vezes o *ray tracing* sozinho é insuficiente para atender aos QPS sem comprometer a qualidade visual. O motivo de citá-la por último é devido a sua natureza, enquanto o *upscaling* e *frame generation* podem apresentar melhoria na performance, colaborando para a possível otimização de jogos, o *ray tracing* vai pelo lado oposto, afinal, é uma tecnologia com alto custo de *hardware*.

O que é o *ray tracing* afinal? *Ray tracing* ou traçado de raios é um método de renderização gráfica, HAINES e AKENINE-MÖLLER (2019, p.9 tradução nossa) explica que:

> O traçado de raios usa o mecanismo de lançamento de raios para coletar recursivamente as contribuições de luz de objetos reflexivos e refrativos. Por exemplo, quando um espelho é encontrado, um raio é lançado de um ponto de impacto no espelho na direção da reflexão. Qualquer coisa que este raio de reflexão intercepte afeta o sombreamento final do espelho. Da mesma forma, objetos transparentes ou de vidro podem gerar tanto um raio de reflexão quanto um raio de refração. Este processo ocorre recursivamente, com cada novo raio potencialmente gerando raios de reflexão e refração adicionais.

O objetivo dessa técnica é simular uma luz mais realista com reflexos e refrações, por exemplo espelhos, poças de água e iluminações mais fiéis à realidade. No entanto é necessária uma placa de vídeo das séries RTX da Nvidia, ou para a AMD a partir das 6000, com a ressalva que apesar da disponibilidade dessa tecnologia, *hardwares* compatíveis do segmentos de entrada (menor custo e performance) e de gerações anteriores, irão apresentar maior custo de desempenhos. Como pode ser observado nos seguintes testes feito utilizando a placa de menor custo da primeira geração com suporte da Nvidia (2060), em testes realizados pelo SCHIESSER (2024):

Figura 2 – Testes de performance RTX 2060

Fonte: SCHIESSER (2024). Nvidia's RTX 2060 Was Never Fast Enough for Ray Tracing. https://www.techspot.com/review/2924-nvidia-not-fast-enough-ray-tracing

Felizmente no cenário atual os lançamentos contam com essa tecnologia como uma opção e não uma necessidade, tendo então poucos jogos que vem nativamente com a única opção de rasterização o *ray tracing.* Essa inovação reflete na busca do mercado por cada vez mais realismo em seus jogos como será abordado adiante.

4.3.2. Uso do Hardware

Ao analisar os últimos lançamentos de jogos nas plataformas de PC, não exclusivamente da Unreal Engine, observa-se requisitos cada vez maiores e algumas práticas estranhas nos requisitos mínimos, como podemos ver na Figura 3:

Figura 3 – Levantamento de requisitos

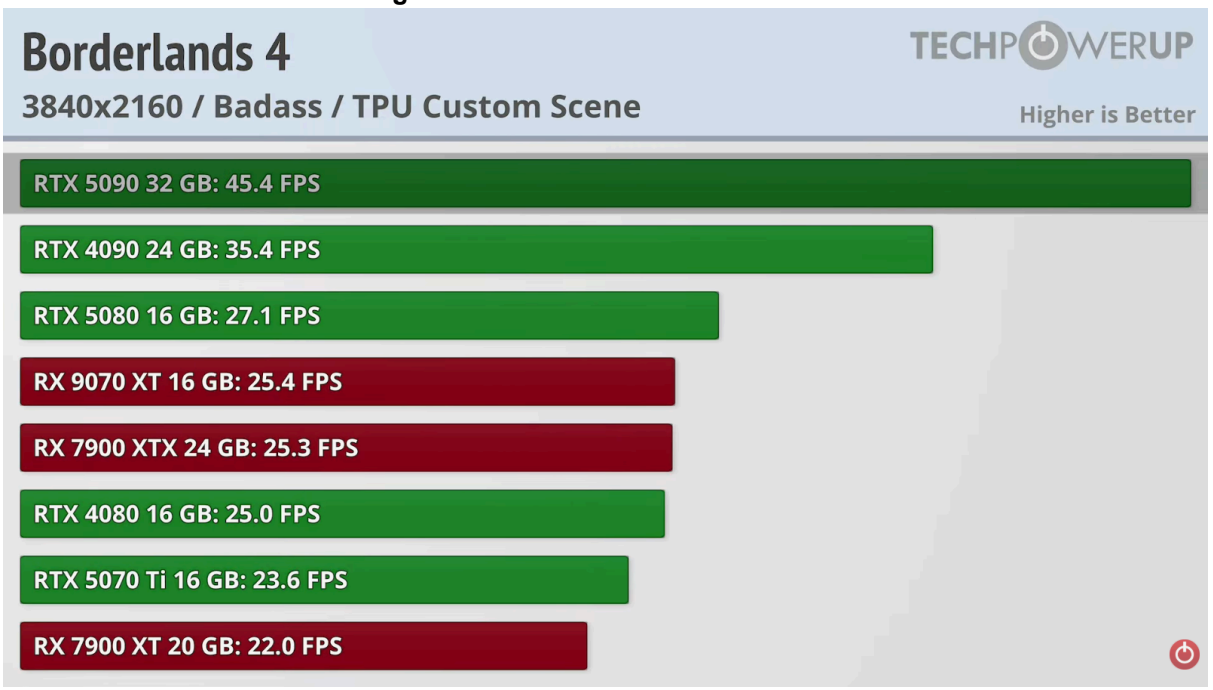
Fonte: Steam (2025). Capturas de tela realizadas pelo autor.

Na imagem foram levantados dois jogos feitos na Unreal Engine 5 e dois de diferentes *engines*, observa-se que não é um caso isolado da Unreal o alto uso de hardware e é possível ver práticas que antes não eram convencionais, como requisitos mínimos visando 30 QPS ao invés de 60, uso obrigatório de *upscaling*, *ray tracing* obrigatório e uso de *frame generation* para atingir os 60 QPS, que como visto anteriormente é uma prática não recomendada.

Além do requerimento de alto nível de *hardware*, é comum lançamentos saírem performando mal nos próprios requisitos divulgados pela empresa, como o

caso do recente Borderlands 4, desenvolvido também na Unreal Engine 5, que conta com uma performance insatisfatória até mesmo nos *hardwares* mais avançados do mercado de computadores pessoais e consoles. Segundo o teste do TechPowerUp (2025) podemos observar que nativamente, a melhor placa de video atual do mercado não chega na média do 60 QPS em 4k:

Figura 4 – Benchmark Borderlands 4



Fonte: TechPowerUp Borderlands 4 Performance Benchmark Review (2025). https://www.techpowerup.com/review/borderlands-4-performance-benchmark

É uma configuração exagerada, tendo em vista que *upscaling* é altamente recomendado hoje em dia e com o devido DLSS a performance fica satisfatória, porém, demonstra o quanto os jogos estão sendo desenvolvido visando o uso dessas tecnologias, afinal essa é a placa de vídeo com maior performance do mercado, custando em média R$ 19.000,00 nos dias atuais e não consegue performar bem nativamente.

Outra prática que vem se tornado comum são as empresas lançarem seus jogos sem a devida otimização e ao decorrer do tempo de vida do jogo lançar pequenas atualizações até corrigir a performance. Ao sofrer diversas avaliações negativas, a Capcom, desenvolvedora do Monster Hunter Wilds, mencionado anteriormente, se pronunciou em suas redes sociais agendando uma grande atualização visando corrigir a performance do jogo, isso após quase um ano de

lançamento. Essa prática não é isolada e demonstra que o problema não está centralizado nas *engines* e sim no processo de desenvolvimento, onde falta a devida atenção no processo de otimização.

4.4. A indústria de jogos

Compreender os processos internos da indústria de jogos é uma tarefa complexa devido à natureza sigilosa e ao uso de *softwares* com código fechado. Desse modo, são escassas as informações fundamentadas, sendo geralmente obtidas por meio de análises técnicas do mercado, vazamentos de dados, e depoimentos de ex-funcionários. Ao se dedicar em trazer uma análise extensa de literatura cinzenta, com o intuito de fornecer um panorama dos problemas da indústria de jogos, focando nos particulares enfrentados pelos desenvolvedores e suas causas raízes, POLITOWSKI et al. (2020, tradução nossa) comenta:

> "De fato, pouco se sabe sobre os problemas enfrentados diariamente pelos desenvolvedores de jogos. Os estúdios de jogos são reservados e possuem uma cultura de código fechado. [...] No entanto, ao contrário das outras indústrias de software, os desenvolvedores de jogos compartilham informações sobre seus projetos de jogos na forma de "histórias de guerra".

Por "histórias de guerra" os autores desejam passar os problemas evidenciado por eles durante o trabalho, como por exemplo, a prática de *Crunch*, EDHOLM et al. (2017, tradução nossa) explica:

> Crunch é um termo utilizado na indústria de jogos para descrever períodos extremos de cargas de trabalho. Durante esses períodos, os funcionários trabalham 12 horas por dia durante diversas semanas, ou até meses.

Isso ocorre para garantir que os jogos sairão dentro do prazo definido pelos acionistas, PETRILLO et al. (2009) aponta o chamado *feature creep,* que se refere a uma prática da indústria onde novas funcionalidades são adicionadas durante a etapa de desenvolvimento, aumentando o tamanho do projeto, mas mantendo o prazo inicial, o que corrobora para o *crunch*.

Com o passar das gerações, grande parte da indústria, sempre buscou alcançar a maior fidelidade gráfica possível, visando entregar um produto mais realista possível. O que acaba sendo uma justificativa para o aumento exponencial do uso da Unreal Engine 5, como pode ser observado por NASCIMENTO e ARAKAKI. (2024) "Unreal Engine 5 se tornou a principal escolha para

desenvolvedores que buscavam gráficos de nova geração, com títulos como Fortnite migrando para essa versão devido ao impacto das tecnologias Nanite e Lumen". Essa alta busca por fidelidade gráfica afeta diretamente a performance e requisitos de *hardware*, com a ressalva que, não é diretamente proporcional, é possível ter fidelidade gráfica com uma boa performance e um uso aceitável de hardware, porém demanda um desenvolvimento com atenção às etapas de otimização, o que muita das vezes é ignorada ou adiada para cumpir os prasos de lançamentos dos produtos como reforçado pelas práticas de *crunch*.

Pode ser feito um paralelo com as tecnologias estudadas anteriormente disponibilizadas pela Unreal Engine e desenvolvedores de placas de vídeos, onde o Nanite, Lumen e Virtual Shadow Maps trazem gráficos de alta fidelidade poupando o trabalho que seriam alcançá-los sem o uso da *engine*, porém ao custo de performance, que como visto, tais implementações apresentam grande custo de *hardware* e nesse ponto vem a implementação do *upscaling* e *frame generation* para tentar compensar essa falta de performance base, porém, ambas também apresentam ainda mais custo de *hardware*, causado assim o aumento de requisitos desses.

4.5. Desenvolvedores independentes

Todos os problemas abordados vem de uma indústria sustentada a base de uma busca por hiper-realismo, projetos super ambiciosos com tempos de produção irrealistas e movidos simplesmente a alcançar o maior número de vendas, sem se importar em lançar um produto não finalizado, como comentado por POLITOWSKI et al (2020, p. 1, tradução nossa) "No entanto, a indústria de jogos continua a gerar lucros enquanto os jogadores continuam a comprar os seus jogos, reforçando um ciclo de más práticas". Uma possível fuga para o consumidor dessa mídia de entretenimento está no mercado de jogos independentes (*indies*), composto por desenvolvedores com pouco orçamento, mas que tem amor ao que faz e muitas vezes, buscam entregar experiências mais únicas do que qualquer empresa de grande porte do mercado arriscaria.

> Mais especificamente, os jogos independentes exercem uma influência substancial na indústria de jogos por meio de suas características criativas distintas. Esses jogos frequentemente demonstram liberdade artística, permitindo que os desenvolvedores explorem estilos de arte não convencionais e abordem temas diversos. GOH et al(2023, p. 3, tradução nossa)

Jogos independentes priorizam a criatividade, integridade e engajamento da comunidade, deixando de ser só um produto e passando a ser um reflexo de seus criadores, fidelidade gráfica e realismo são evoluções técnicas incríveis, porém não deveria ser a base para o mercado de jogos. A paixão por essa área leva a diversas pessoas a se interessar por programação simplesmente com o objetivo de criar sua própria obra, como o caso do Eric Barone, desenvolvedor de Stardew Valley, que sozinho conseguiu criar um dos jogos *indies* de maior sucesso vendendo 500 mil cópias em 15 dias (NOGUEIRA, 2024). Além de tudo isso, esses jogos apresentam baixo custo de *hardware*, dado a sua natureza de menor escopo, sendo então uma opção acessível para aqueles que não tem condições para investir em um computador mais potente.

5. Conclusão

Analisar a indústria de jogos é uma tarefa árdua, devido a sua falta de informações e artigos científicos sobre aspectos específicos. Ainda assim, são observadas algumas práticas negativas recorrentes ao longo dos anos, que torna possível correlacionar e responder aos objetivos propostos.

Com base na análise realizada, constatou-se que a Unreal Engine 5 apresenta limitações de desempenho em jogos de computador, muito disso em função à sua busca por fidelidade gráfica fornecida de forma automatizada por meio de suas ferramentas, já que os resultados indicam que o uso inadequado pode impactar negativamente a performance, a engine também oferece recursos que podem aumentar os requisitos finais para a execução dos softwares, como *ray tracing*. No entanto, verificou-se também que problemas semelhantes ocorrem em jogos desenvolvidos em outras *engines*, o que pode indicar uma tendência mais ampla relacionada aos processos de desenvolvimentos no mercado atual.

Abordando as novas tecnologias levanta a suspeita de que os desenvolvedores podem estar deixando de priorizar a otimização de seus jogos, visando que o *upscaling* e *frame generation* façam esse trabalho sozinho. Essa suspeita é fundamentada nas informações levantadas em relação aos escopos e curtos prazos, que ocasionam a prática do *crunch*, sendo então, não por descaso dos desenvolvedores e sim por pressão das empresas em lançarem seus produtos o mais rápido possível. Assim sendo, priorizar os processos de otimização, adotar prazos mais realistas e ter um bom planejamento de escopo dos projetos, são

práticas que podem mudar esse cenário atual, entregando um produto mais polido e que faz o melhor uso possível do *hardware*.

Por último foi apresentado uma possível alternativa aos jogos de grande investimento, o mercado de jogos independentes, apresentando as suas qualidades e maior acessibilidade para jogadores de computador. Esse mercado foi apresentado para contrastar com as práticas negativas observadas pela indústria e trazer uma reflexão de que pode-se encontrar qualidade em produtos de desenvolvedores com menor orçamento.

REFERÊNCIAS

ALEEM, Saiqa; CAPRETZ, Luiz Fernando; AHMED, Faheem. **Empirical investigation of key business factors for digital game performance**. *Entertainment Computing*, v. 13, p. 1 - 19. 2016. Disponível em: https://arxiv.org/abs/1511.04422. Acesso em: 16 set. 2025.

AMD. **Tecnologias AMD FSR™ Desempenho impressionante. Fidelidade máxima**. 2025. Disponível em: https://www.amd.com/pt/products/graphics/technologies/fidelityfx/super-resolution.html. Acesso em: 20 nov. 2025.

AZSAN, Farbod. **The Essentials of Video Game Optimization**. Polydin Studio. 2025. Disponível em: https://polydin.com/video-game-optimization/. Acesso em: 10 set. 2025.

BAO, Wenbo; et al. **Depth-Aware Video Frame Interpolation.** Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). p. 3703 - 3712. 2019. Disponivel em: https://openaccess.thecvf.com/content_CVPR_2019/html/Bao_Depth-Aware_Video_Frame_Interpolation_CVPR_2019_paper.html Acesso em: 19 nov. 2025.

EDHOLM, Henrik; LIDSTRÖM, Mikaela; STEGHÖFER, Jan-Philipp. **Crunch Time: The Reasons and Effects of Unpaid Overtime in the Games Industry.** 2017. Dept. of Computer Science and Engineering Chalmers - University of Gothenburg Gothenburg, Sweden. Disponível em: https://ieeexplore.ieee.org/document/7965428. Acesso em: 20 nov. 2025.

EPIC GAMES. **Unreal Engine 5.6 Documentation**. 2025. Disponível em: https://dev.epicgames.com/documentation/pt-br/unreal-engine/unreal-engine-5-6-documentation. Acesso em: 10 set. 2025.

EVANSON, Nick. **Epic's CEO Tim Sweeney wades in on the UE performance debate: "The primary reason Unreal Engine 5-based games don't run smoothly on certain PCs or GPUs is the development process"**. *PC Gamer*, 28 ago. 2025. Disponível em: https://www.pcgamer.com/hardware/epics-ceo-tim-sweeney-wades-in-on-the-ue-performance-debate-the-primary-reason-unreal-engine-5-based-games-dont-run-smooth

ly-on-certain-pcs-or-gpus-is-the-development-process. Acesso em: 10 set. 2025.

GOH, Edward; AL-TABBAA, Omar; KHAN, Zaheer. **Unravelling the complexity of the Video Game Industry: An integrative framework and future research directions**. Telematics and Informatics Reports, v. 12, 2023. Disponível em: https://www.sciencedirect.com/science/article/pii/S2772503023000609. Acesso em: 25 nov. 2025.

GREGORY, Jason. **Game Engine Architecture**. A K Peters, Ltd. Wellesley, Massachusetts. 2009.

GUGELMIN, Felipe. **Monster Hunter Wilds: Capcom dá dicas de como melhorar o game no PC.** Adrenaline. 2025. Disponível em: https://www.adrenaline.com.br/games/monster-hunter-wilds-capcom-da-dicas-de-como-melhorar-o-game-no-pc. Acesso em 16 nov. 2025.

HAINES, Eric; AKENINE-MÖLLER, Tomas. **Ray Tracing Gems: High-Quality and Real-Time Rendering with DXR and Other APIs**. Berkeley, CA: Apress, 2019. 607 p. ISBN 978-1-4842-4426-5.

JUNGHERR, Andreas; SCHLARB, Damien B. **The Extended Reach of Game Engine Companies: How Companies Like Epic Games and Unity Technologies Provide Platforms for Extended Reality Applications and the Metaverse**. SAGE Journals 2022. Disponível em: https://journals.sagepub.com/doi/full/10.1177/20563051221107641. Acesso em: 19 set. 2025.

LAIRD, Jeremy. **AMD details its Nvidia DLSS 3 Frame Generation-fighting tech**. PC Gamer, 2023. Disponível em: https://www.pcgamer.com/amd-details-its-nvidia-dlss-3-frame-generation-fighting-tech/. Acesso em 20 nov. 2025.

LI, Tianshu. **Real–time performance comparison of environments created using traditional geometry rendering versus Unreal Nanite technology in virtual reality.** Purdue University Graduate School, 26 abr. 2024. Disponível em: https://hammer.purdue.edu/articles/thesis/_b_Real_time_performance_comparison_of_environments_created_using_traditional_geometry_rendering_versus_Unreal_Nanite_technology_in_virtual_reality_b_/25676631?file=45875235. Acesso em: 18 set.

2025.

LIA, Raphael Vieira da Silva. **NÃO SÃO SÓ JOGUINHOS: CRESCIMENTO E ESPACIALIDADE DA INDÚSTRIA DE JOGOS DIGITAIS NO BRASIL**. UFRJ - Congresso ibero-americano interdisciplinar de economia criativa. 2020.

MALICK, Saad; NAGAR, Shrey; LAMBA, Dr. Shalini. **Optimizing Distributed Ray Tracing in Unreal Engine Using AIBased Techniques: An Experimental Approach**. 2025. International Research Journal of Education and Technology. Disponivel em: https://www.irjweb.com/Optimizing%20Distributed%20Ray%20Tracing%20in%20Unreal%20Engine%20Using%20AI-Based%20Techniques%20An%20Experimental%20Approach.pdf. Acesso em 19 nov. 2025.

NASCIMENTO, Wérik Prado; ARAKAKI, Pedro Henrique Yudi; **UM ESTUDO HISTÓRICO SOBRE O DESENVOLVIMENTO DE JOGOS DIGITAIS**. 2024. Orientador: Prof. Dr. Henrique Dezani. Trabalho de graduação (Análise e desenvolvimento de sistemas) – Faculdade de Tecnologia de São José do Rio Preto, São José do Rio Preto, 2024. Disponivel em: https://ric.cps.sp.gov.br/bitstream/123456789/33872/1/analiseedesenvolvimentodesistemas_2024_2_werikpradonascimento_umestudohistoricoobreodesenvolvimentodejogosdigitai.pdf. Acesso em: 17 nov. 2025.

NOGUEIRA, Rodolfo. **GameDev: Eric Barone e a História do Desenvolvimento de Stardew Valley!**. Um Gamer. 2024. Disponível em: https://umgamer.com/pt-br/articles/gamedev-eric-barone-e-a-historia-do-desenvolvimento-de-stardew-valley. Acesso em: 21 nov. 2025.

NVIDIA. **NVIDIA DLSS.** 2025a. NVIDIA Developer. Disponível em: https://developer.nvidia.com/rtx/dlss?sortBy=developer_learning_library%2Fsort%2Ffeatured%3Adesc%2Ctitle%3Aasc . Acesso em: 20 nov. 2025.

NVIDIA. **NVIDIA DLSS 4.** 2025b. Disponível em: https://www.nvidia.com/en-us/geforce/technologies/dlss. Acesso em 20 nov. 2025.

PETRILLO, Fábio; et al. **What went wrong? A survey of problems in game development.** 2009. Comput. Entertain. 7, Article 13 (February 2009). p. 1 - 22. Disponível em: https://doi.org/10.1145/1486508.1486521. Acesso em: 17 nov. 2025.

POLITOWSKI, Cristiano; et al. **Game Industry Problems: an Extensive Analysis on the Gray Literature**. 2020. Concordia University, Montreal, Quebec, Canada. Disponível em: https://www.researchgate.net/publication/344159967_Game_Industry_Problems_an_Extensive_Analysis_on_the_Gray_Literature. Acesso em: 17 nov. 2025.

SATALIN, Anton. **Improving the game development process on Unreal Engine 5**. Hame University of Applied Sciences. 2024. Disponível em: https://www.theseus.fi/handle/10024/864317. Acesso em: 10 set. 2025.

SEN, Shubhojit; BHUSHAN, Bharat. **Image Quality Comparison Between Nvidia's Deep Learning Super Sampling and AMD's FidelityFX Super Resolution**. 2024. International BIT Conference (BITCON). Disponível em: https://ieeexplore.ieee.org/document/10984458. Acesso em: 19 nov. 2025

SILVA, Lucas Rodrigues da. **Análise de desempenho de diferentes engines no desenvolvimento de jogos com implementação de wizard para recomendação das engines.** 2024. Orientador: Luciene Cavalcanti Rodrigues. Trabalho de Conclusão de Curso (Curso Superior de Tecnologia em Informática para Negócios) – Faculdade de Tecnologia de São José do Rio Preto, São José do Rio Preto, 2024. Disponível em: https://ric.cps.sp.gov.br/handle/123456789/23428. Acesso em: 19 set. 2025.

SOBCHYSHAK, Oleksandra; BERREZUETA-GUZMAN, Santiago; WAGNER, Stefan. **Pushing the boundaries of immersion and storytelling: A technical review of unreal engine**. 2025. Technical University of Munich, Heilbronn, Germany Disponível em: https://arxiv.org/pdf/2507.08142. Acesso em: 16 set. 2025.

STEAM. **Loja Steam – Página inicial**. Valve Corporation. Disponível em: https://store.steampowered.com/. Acesso em: 17 nov. 2025.

TECHPOWERUP. **Borderlands 4 Performance Benchmark Review - 30+ GPUs Tested.** 2025. Disponível em: https://www.techpowerup.com/review/borderlands-4-performance-benchmark/5.html. Acesso em 16 nov. 2025.

THORN, Alan. **Game engine design and implementation.** Jones & Bartlett Publishers, 2011.

UNREAL DEVOP. **Unreal Engine 5 - Virtual Shadow Map Performance (Explained)**. YouTube. 2022. Disponível em: https://www.youtube.com/watch?v=lJgOlSNVVW4. Acesso em: 17 set. 2025.

WEIZENMANN, Henrique. **Veja diferença de desempenho entre Unreal Engine 5.5, 5.6 e 5.7.** Adrenaline. 2025. Disponível em: https://www.adrenaline.com.br/softwares/veja-diferenca-de-desempenho-entre-unreal-engine-5-5-5-6-e-5-7. Acesso em 15 nov. 2025.

Glossário

Benchmarks: Procedimentos utilizados para comparar e avaliar o desempenho de softwares.

Bugs: Erros ou falhas em um software que podem comprometer seu funcionamento.

Crashes: Falhas críticas que levam o software ou jogo a encerrar abruptamente.

Debug: Processo de identificar, analisar e corrigir erros em um software.

Gráficos: No contexto de jogos digitais, refere-se ao visual geral, incluindo qualidade de imagem, texturas e efeitos visuais.

Hardware: Parte física do computador, composta por componentes como processador, placa de vídeo, memória, monitor, entre outros.

Renderizar: Processo de transformar instruções digitais em imagens exibidas na tela.

Software: Parte lógica de um sistema computacional, composta por programas e sistemas que executam tarefas específicas.