Comparing different types of rendering strategies in an animated short film

Comparação de estratégias de render em um curta-metragem de animação

Marcelo Tannure
Universidade FUMEC, Brasil
Gustavo Rodrigues Corrêa
Universidade FUMEC, Brasil
João Victor Boechat Gomide
Universidade FUMEC, Brasil

Abstract

This article discusses the development of an animated short film in which different rendering techniques were used, in order to compare them and find the best cost/benefit ratio. The short film is 16 minutes long and features scenes with complex landscapes and different types of framing.

It is known that the rendering process was, until a few years ago, the most time-consuming process in the production of animations and visual effects scenes. Thus, a great effort has been spent on research to develop hardware and software solutions that perform this activity in an affordable time, maintaining the necessary quality to produce images in the shortest possible time.

In the animated short film, V-Ray, Arnold Renderer and Unreal renderers were used in different sequences. Arnold, RenderMan and Unreal renderers were also compared in the same sequence. RenderMan, V-Ray and Arnold are renderers used in post-production and have an extensive portfolio of applications. Unreal opened the possibility of rendering in real time, without loss of quality, with the release of version 4.0 in 2014, and opened new paths for the so-called Virtual Production, by integrating live scenes with animated images in computer graphics, with realism and in real time, a trend that is definitely established with Unreal 5.0, from 2022.

In this article, the production and rendering process of the animated short film will be described, and the different strategies adopted by renderers to obtain the finished image will be discussed.

Keywords: Real time rendering, Render, Motion capture, Ray tracing, Animated short film.

Introdução

Neste artigo, fazemos comparações entre diferentes renderizadores utilizados para a finalização de um curta-metragem de animação com 16 minutos. Este foi um trabalho que se desenrolou ao longo de quase 10 anos, todo ele realizado por professores e alunos dos bacharelados em Computação Gráfica e em Design de Games da Universidade FUMEC. A longa duração da produção foi devido a se contar com um equipe semiprofissional e, na parte final, ao tempo de renderização, processo que será explicado a seguir. Em uma primeira renderização, incluindo somente as sequências do Jardim Botânico de Berlim, que correspondem a pouco mais da metade do tempo da animação, foram gastos sete meses para finalizar o processamento de todas as tomadas, com o hardware disponível.

De acordo com o (Dicionário Online de Português 2022), "renderizar é tornar permanente um trabalho de processamento digital (áudio, imagem etc.) que, após as alterações editadas, resulta num arquivo final; compilar: renderizar imagens fixas em vídeo".

Esse tempo alongado na formação das imagens geradas por computador se tornou um obstáculo à medida que a complexidade das cenas aumentava. Renderizar, para (Andaló 2010), é o processo que executa os cálculos de iluminação, a partir de uma cena com polígonos em 3D, para uma imagem final 2D. De uma forma mais simplificada, renderização determina a maioria das características visuais de um ambiente tridimensional virtual e é nesse processo que são geradas as imagens finais de todos os efeitos aplicados à cena (Dutra 2019). O processo de renderização resulta na criação de uma imagem fotorrealista ou não, obtida do cálculo de uma grande quantidade de informações provenientes de um conjunto imagético.

(Kerlow 2000) descreve de maneira poética, mas muito precisa, o significado da palavra "rederização":

> O mundo da renderização por computador é povoado pela maioria dos atributos do nosso reino visual, onde as formas dos objetos são reveladas pela luz e obscurecidas pela sombra, onde a cor cria ambientes de tranquilidade sutil ou felicidade explosiva, onde as texturas são tão delicadas e líricas como areia fina ou dramática e forte como a malaquita, onde a translucidez incansável da chuva distorce as feições do mundo e o espelho de água transparente os reúne. Renderizar mundos de realidade ou fantasia com o computador pode criar resultados tão artísticos quanto com qualquer outro meio.

Renderizar, neste artigo, é o mesmo que sintetizar uma imagem a partir dos dados coletados pelo computador e processados através de um programa específico. Esses dados podem ser informações de posição de vértices de uma geometria bi ou tridimensional, a cor que é aplicada no ponto, a luz que incide sobre o objeto da cena, efeitos especiais

de fumaça, fogo, água e todo sistema de partículas, dentre outros.

A renderização pode ser realizada de forma pós-processada, quando toda a cena é preparada para ser trabalhada em um tempo de processamento maior que a taxa de quadros por segundo, como sempre foi realizada, ou em tempo real, que tem sido permitida pelos motores gráficos de jogos digitais. Esses motores começaram a ser aplicados para renderizar filmes e programa para televisão, como o Unreal (Cowley 2015) e o Unity (Liptak 2017). O motor Unreal, em especial, tem sido utilizado para criar cenários em paredes de LEDs, como no longa-metragem *Mandalorian* (Farris 2020) e abrindo caminho para uma nova tendência de produção cinematográfica, a da produção virtual (Kadner 2019).

Este trabalho busca compreender quais as etapas e limitações de diferentes softwares de render, aplicados em um curta-metragem de animação. Para isso, foram estudadas algumas ferramentas e buscadas as razões teóricas para essas diferenças de desempenho e qualidade final da imagem.

As sequências da animação do curta-metragem foram separadas em três tipos, de acordo com os cenários e os personagens envolvidos, e foram utilizados os renderizadores (V-Ray 2022), (Arnold 2022) ou (Unreal 2022) em cada um deles. Para uma das sequências, utilizou-se os renderizadores (RenderMan 2022), Arnold e Unreal, para se comparar o resultado de cada uma delas na imagem renderizada.

O texto a seguir está organizado da seguinte forma. Na próxima seção, são apresentadas as sinopses e as etapas de produção do curta-metragem. Em seguida, é discutido o fluxo de trabalho. Na outra seção, são discutidos trabalhos relacionados a render. A seguir, são apresentados os resultados e a metodologia adotada e, por último, as conclusões.

Sinopse e etapas de produção

O curta-metragem produzido, *A Primeira Perda da Minha Vida*, é uma adaptação livre de uma história narrada por Dora Diamant, última companheira do escritor Franz Kafka, após ele falecer, em 1924. Kafka nasceu em Praga, na República Tcheca, mas teve quase toda a sua educação em escolas alemãs. Alimentou, durante sua vida, o desejo de morar em Berlim. Já com tuberculose, mudou-se para esta cidade em setembro de 1923, e lá viveu, na companhia de Dora Diamant, até março de 1924.

Dora Diamant contou a Max Brodi, editor das obras de Franz Kafka, que, em setembro de 1923, quando Kafka passeava em um parque de Berlim, próximo da casa onde moravam, ele encontrou uma menina que perdera a sua boneca. Depois de uma pesquisa em parques próximos à residência do casal, descobriu-se que o Jardim Botânico poderia ser o lugar onde se desenrolou a história.

A menina chorava pela perda e Kafka disse a ela que a boneca estava escrevendo cartas, que ele recebia. Kafka então passou a se encontrar no Jardim Botânico com a menina, para ler as cartas. Isso durou algumas semanas, até que a menina se conformou com a perda. Segundo Dora, Kafka se dedicava às cartas com a mesma energia que se dedicava a seus escritos literários.

O roteiro do curta-metragem foi escrito pelo diretor do Grupo de Teatro Galpão, Eduardo Moreira, introduzindo algumas adaptações livres na história original, como a visita da Boneca a Olinda, uma cidade histórica do Nordeste do Brasil. A Boneca acaba se apaixonando por um Bonecão do Carnaval de Olinda, uma tradição centenária desta festividade.

O desenho de produção foi realizado escolhendo as locações e os conceitos artísticos de todos os personagens. Os movimentos dos personagens foram capturados a partir da encenação de quatro atores do Grupo Galpão, utilizando a captura de movimento. Todos os personagens foram desenvolvidos, a partir dos *concepts*, com o software (Autodesk Maya 2022). O cenário onde acontece os encontros de Kafka com a menina, o Jardim Botânico de Berlim, foi criado e desenvolvido utilizando o software (Autodesk 3ds Max 2022), com a vegetação sendo construída com o (iToo Software 2022). Os cenários onde se passa as sequências com a boneca e o escritório do Kafka foram criados com o Autodesk Maya. O detalhamento de toda a produção, antes da etapa do render, está no artigo (Gomide et al 2014).

Fluxo de trabalho

O trabalho se inicia com a produção das peças tridimensionais que comporão a cena, no processo de modelagem tridimensional. De maneira simplificada, um modelo 3D representa pontos matemáticos posicionados no espaço tridimensional, ligados entre si, formando a malha de polígonos que representa o objeto volumétrico. Os vértices podem ser manipulados livremente até que se consiga o formato ideal do modelo.

O passo seguinte é a colocação dos materiais e texturas nas superfícies poligonais. Essa etapa determina o tipo de interação com a iluminação, no momento de renderizar. Os materiais e texturas podem ser do tipo procedural, quando um modelo matemático é aplicado nos polígonos e os parâmetros do modelo são ajustados, ou do tipo mapeamento UV. Neste caso, a superfície do objeto tridimensional, que é a malha de polígonos, é considerada bidimensional, associando as coordenadas U e V a uma origem nesta malha. Com a malha representada bidimensionalmente, é possível colocar uma imagem sobre a malha e fazer uma correspondência biunívoca entre essa imagem e cada ponto da malha bidimensional. Feito isso, a superfície do objeto é envelopada pela imagem ou imagens mapeadas sobre ela. No processo de renderização, o mapeamento funciona como a textura e a interação da luz com a superfície é determinada pelo material, ou sombreador (*shader*), escolhido, como, por exemplo, o lambert, o phong ou o blinn, que são os principais dentre muitos tipos. O material, ou sombreador, tem a descrição do espalhamento da luz, da sua reflexão especular e outros parâmetros

para o render, como refração e outros, envolvidos no rastreamento dos raios de luz que ermergem do objeto (*ray tracing*), que é uma das técnicas computacionais utilizadas pelos algoritmos.

A partir do momento que toda a modelagem estiver concluída e com as texturas aplicadas, inicia-se o processo denominado *rigging*. Ele é definido como as estratégias que se adota para que se torne possível a movimentação de personagens ou elementos da cena, de modo que a animação fique convincente. Ele envolve estruturas articuladas, que atuam nas malhas de polígonos, com uma distribuição de peso dessa atuação. Essas estruturas podem deformar e movimentar a malha de polígonos, de acordo com o peso dessa atuação. É certo que um personagem detalhado vai fazer com que o *rigging* se torne mais complexo e, consequentemente, aumentando o tamanho em bits do arquivo, o que reverte em mais cálculos, aumentando o tempo de processamento da renderização.

A animação dos personagens e elementos de cena é a etapa onde todas as informações do roteiro são aplicadas para a realização efetiva do filme. É aqui que tudo toma vida e se pode contar a história. Os aspectos técnicos do processo de animação não vão influenciar o tempo de render ou a qualidade da imagem final, mas vão certamente definir a qualidade do produto em sua perspetiva lúdica.

Seguindo o fluxo de trabalho, em seguida vem o conjunto de técnicas que mais impactam na qualidade da imagem e no tempo gasto para renderizar, que é a iluminação das cenas do projeto. No mundo real, os sistemas de iluminação são muito complexos e atuam sobre os objetos de formas distintas, dependendo do tipo de superfície que eles têm.

A simulação desses efeitos de luzes na computação gráfica é sofisticada e demanda muito fortemente os recursos de hardware. A iluminação, como a interação com as superfícies, é construída apoiada em princípios da Física, que são modelados matematicamente e representados virtualmente, quando são implementados no software. A maioria dos softwares de modelagem 3D atuais contam com uma biblioteca de luzes artificiais que imitam o mundo real, como luzes pontuais, luzes tipo spot, luzes direcionais, luzes de área e outros tipos de luzes físicas, como os domos, para geração de efeitos no ambiente convincentes.

Entretanto, cálculos mais específicos precisam ser feitos para que a iluminação do mundo real possa ser representada dentro do ambiente virtual de forma mais próxima da realidade. Os tratamentos dessas luzes vão envolver características do mundo físico, como a distribuição de fótons na cena, radiosidade, oclusão de ambiente, rastreamento de raio (*raytracing*), coeficiente de refração e outros. Esses atributos impactam diretamente no render e precisam ser trabalhados, para não se perder o realismo da cena, mas também levando em conta a otimização do tempo de produção dessas imagens finais.

A etapa seguinte se refere, dentro do fluxo de trabalho proposto, aos efeitos dinâmicos que deverão existir nas cenas. Um dos principais é a simulação de partículas, próprias para produção de efeitos como poeira, fumaça, fogo e outros. Em muitos casos, existirão simulações de tecidos, cabelos e água. Efeitos volumétricos, como nuvens e névoas, também se enquadram dentro dessa área. Todos esses efeitos dinâmicos também têm um peso no tempo de render, pois ocuparão boa parte da memória do computador para que os cálculos ocorram corretamente.

A fase seguinte é o render propriamente dito, objeto dessa pesquisa. Depois que todas as etapas anteriores estiverem configuradas satisfatoriamente, a cena deve entrar no processo de renderização. Embora o contexto mais amplo desta fase comece com o sombreamento, na colocação de materiais nas superfícies, a texturização de objetos e iluminação da cena, o processo de renderização termina quando superfícies, materiais, luzes e movimentos são processados em uma imagem final ou sequência de imagens. Deve-se escolher o software que fará o render, porque ele determina a forma em que os parâmetros utilizados na cena serão trabalhados e como as texturas procedurais são descritas. Muitos parâmetros dependem dessa decisão estratégica, como a escolha de materiais, câmeras, texturas, efeitos especiais e, principalmente, as luzes, pois cada software de render adota algoritmos próprios para essas etapas e, muitas vezes, eles não são intercambiáveis.

De acordo com o manual da (Autodesk 2022), a chave para uma renderização bem-sucedida é encontrar o equilíbrio entre a complexidade visual necessária e a velocidade de produção das imagens finais, que determina quantos quadros podem ser renderizados em um período de tempo. A renderização envolve cálculos matemáticos complexos que podem manter o computador ocupado por muito tempo. A produção de imagens renderizadas sempre envolve escolhas que afetam a qualidade das imagens e a velocidade com que elas são renderizadas. Esta etapa será descrita com mais detalhes na seção a seguir.

Para finalizar o fluxo de produção, seja de uma imagem ou de um filme, depois de todo processo de render finalizado, o material é encaminhado para a pós-produção, onde vão ser feitos ajustes de cor, inclusão de efeitos especiais, efeitos de áudio, trilha sonora, composição e montagem.

Fundamentação teórica

A renderização pode acontecer durante a pós-produção, depois de construídas as tomadas, ou em tempo real. A renderização feita apenas na pós-produção é também conhecida como renderização offline. Até meados dos anos 2010, toda renderização eram pós-produzidas, com as cenas sendo processadas em um tempo muito mais longo do que a duração da cena. A renderização em tempo real começou a se tornar viável com a utilização de motores gráficos de jogos digitais para renderizar cenas de vídeo. Ela trabalha com a criação imediata de imagens no computador, ocorrendo em tempo igual ou ligeiramente superior ao tempo da cena. Ela possibilita

a interatividade com a computação gráfica, ao permitir que a reação do espectador afete a imagem que é gerada a seguir, como nos jogos digitais. Este ciclo de reação e renderização acontece em uma taxa rápida o suficiente para que o espectador não veja imagens individuais, mas, ao invés disso, fique imerso em um processo dinâmico (Akenine-Möller et al. 2018).

Hoje, as necessidades por cenas cada vez mais realísticas, ou mesmo as cartunizadas mais complexas, requerem cálculos físicos de polígonos, luzes, sombreadores e ambientações. Os pós-renderizadores podem atender a esses requisitos, mas a um custo de tempo razoável. Por esse motivo é que os departamentos de pesquisa e desenvolvimento da área procuram formas de otimizar o processo de render, melhorando os algoritmos e encontrando meios de aumentar a performance das máquinas para produzirem imagens finais no menor tempo de render possível, sem perda de qualidade.

Em 1965, Gordon Moore estabeleceu uma relação empírica que determinava que em cada 12 meses, depois corrigida para 24 meses, o número de transístores na arquitetura dos computadores dobraria e, consequentemente, o seu poder de processamento, ao mesmo custo monetário (Moore 1965). Com o desenvolvimento dos computadores, sua crescente capacidade de processamento e continua redução de custos, tornou-se viável a busca por imagens 3d que pudessem se aproximar do mundo real, principalmente no que esse refere à iluminação. Os aperfeiçoamentos dos algoritmos dos softwares de render permitiram o uso de recursos mais avançados na elaboração das imagens virtuais.

(Akenini-Moller et al 2018) descreve 4 etapas para um fluxo de trabalho de renderização em tempo real: aplicação, processamento de geometria, rasterização e processamento de pixels. O primeiro se refere ao processamento em CPU (Unidade Central de Processamento), no qual se pode ter controle total sobre os elementos da cena. Outros elementos, como sombreamento dos vértices (*vertice shading*), são calculados na GPU (Unidade Gráfica de Processamento).

Pontos, linhas e triângulos são as primitivas de renderização a partir das quais um modelo é construído. Portanto, podemos manipular esses elementos e os processadores vão atualizar sempre os dados para a condição atual do objeto. A partir do enquadramento de uma câmera virtual sintética no cenário, a aplicação vai fazer os processadores calcularem a animação e reconstruir, quadro a quadro, a imagem com todas as variações de posição espacial dos vértices e triângulos, além dos efeitos de luz sobre as superfícies.

O processamento geométrico se refere a todos os objetos que estão em cena e que deverão fazer parte do render. Tudo aquilo que se encontra fora do campo de visão da câmera sintética e que, portanto, não vai fazer parte da imagem final, será descartado e não processado. Assim, um cubo, por exemplo, que estiver dentro do enquadramento da câmera, terá sua forma, a posição espacial de seus vértices, os sombreamentos, os efeitos da luz sobre sua superfície, calculados e apresentados na imagem resultante. Todo o resto fora do enquadramento não será calculado. Essa característica diminui a carga de processamento da CPU e da GPU, acelerando o processo de render.

A rasterização é definida por (Akenine-Möller et al 2018) como o reconhecimento de entrada de três vértices de cada polígono, formando um triângulo, e encontra todos os pixels que são considerados dentro desse triângulo, ou seja, transforma uma entrada vetorial em uma saída de pixels. Essa rasterização é aplicada a todas as primitivas que estão dentro do enquadramento da câmera sintética, na qual todos os pixels que compõem as primitivas calculadas são encontrados e enviados ao passo seguinte do fluxo de trabalho, ou seja, o processamento de pixels.

Nesse último passo, é calculado a cor de cada pixel de cada objeto visível na imagem. Aos triângulos das geometrias são associadas as texturas definidas no primeiro estágio do fluxo de trabalho, mostrando a sua visibilidade, por meio do algoritmo z-buffer. Por meio do z-buffer, a cena é calculada por planos perpendiculares ao eixo óptico da câmera, levando em consideração a distância em z da câmera. Se um elemento está mais próximo e ele faz a oclusão de outro elemento mais distante, o novo elemento substitui o anterior na imagem. Cada informação de cor de cada pixel é alocada em coordenadas espaciais X e Y, portanto bidimensional. Assim o algoritmo z-buffer introduz a coordenada Z, para que a distância do pixel da câmera possa ser calculada. Isso dá à imagem gerada uma informação de profundidade que vai garantir um realismo maior às imagens geradas e permite, ainda, que cada objeto possa ser processado separadamente e a imagem final é exibida na tela (Akenini-Moller et al. 2018).

Outros aspectos se incorporam no sistema de render, como os cálculos de rastreamento de raio que hoje é utilizado em qualquer software de renderização. No conceito original, de acordo com (Lopes 2013), rastreamento de raio “é um conjunto de algoritmos que determinam a visibilidade dos objetos de uma cena, operando ao nível de precisão de uma imagem. O algoritmo fundamental de rastreamento de raio considera raios com origem no centro de projeção, em que cada um dos raios passa por um ponto correspondente a um pixel da imagem, sobre o plano de projeção da cena a representar. Se um raio não intersectar nenhum objeto da cena, é atribuída ao pixel da imagem por onde o raio passa, a cor de fundo da cena. Caso contrário, determinam-se as interseções do raio com os objetos da cena e, ao pixel correspondente ao raio, é atribuída a cor do ponto de interseção mais próximo da origem do raio”.

Simular as propriedades físicas da luz em computação gráfica é um dos aspectos mais complexos de se implementar e para se obter um resultado satisfatório é necessário recorrer a um estudo mais profundo sobre o comportamento da luz natural ou artificial quando ela incide sobre alguma superfície.

Em um passado recente, esse processo era severamente prejudicado pela limitação de hardware,

mas com o desenvolvimento dos computadores e a consequente redução de custos, as imagens 3D puderam então se aproximar do mundo real, em especial no que diz respeito à iluminação (Andaló et al. 2010).

Existem vários aspectos de iluminação em um ambiente 3D que devem ser levados em consideração para que a superfície iluminada se comporte dentro do esperado. Dentre estes temos os reflexos especulares e difusos, que são a concentração de luz em superfícies polidas e dispersão em superfícies rugosas, a translucidez dos objetos, que vão determinar qual quantidade de luz poderá atravessar a superfície, a transparência e refração, nas quais é determinado como a luz se comportará ao penetrar nos objetos, e outros fenômenos naturais que devem ser observados para serem replicados nos ambientes digitais.

Embora o render não envolva somente o cálculo do efeito de luz – inclui também a disposição dos elementos da cena, a eliminação de linhas e superfícies ocultas, projeção em perspetiva, para a apresentação final da imagem numa grade de pixels -, é a modelagem através da simulação da luminosidade que garante a aparência realista no ambiente virtual (Dutra 2019).

O software de renderização deve fornecer as ferramentas para o cálculo da imagem final simulando diferentes distâncias focais da lente da câmera sintética, produzindo a profundidade de campo da imagem. Na profundidade de campo, estará em foco aquilo que estiver dentro da distância focal da lente da câmera e se aplica um desfoque naquilo que estiver fora (Demers, 2022). O software também deve fornecer uma biblioteca de sombreadores e texturas procedurais, que darão à cena condições de se parecer com uma cena do mundo real.

Na maioria dos softwares de modelagem tridimensional, existe uma lista extensa de materiais disponíveis pré-configurados para criação de aparências para objetos de todos os tipos. Os materiais Lambert descrevem superfícies opacas e sem brilho. Do lado oposto, estão os materiais Phong, que determinam superfícies lisas e brilhosas. Esses dois shaders são a base para todos os outros materiais que podem ser configurados para se ter a aparência desejada.

O anti-aliasing também é aplicado, para evitar que as bordas dos objetos fiquem serrilhadas na imagem. O anti-aliasing introduz um pequeno desfoque das bordas na imagem final, evitando, desta forma, que elas tenham a aparência quebrada entre os pixels.

Devido à natureza complexa e cumulativa do processo de renderização, geralmente há uma transição considerável de ida e volta entre os estágios, antes do processo ser concluído, e a implementação dessas técnicas em programas diferentes pode exigir pequenas variações na sequência de realização dessas etapas.

(Kerlow 2000) propõe um fluxo de trabalho para o render composto por cinco etapas. A primeira etapa no processo de renderização consiste em fazer com que os modelos ou objetos poligonais, presentes na cena, sejam renderizados diretamente do módulo de modelagem do programa. Esses modelos geralmente incluem personagens, cenários e adereços. A segunda etapa se refere à manipulação da câmera sintética no espaço XYZ, para que seja possível enquadrar a parte do cenário no qual estamos interessados. Nesta fase podemos trabalhar a posição e rotação da câmera, além da abertura da lente, a profundidade de campo e ajustar as proporções e a resolução da imagem.

A terceira etapa é o momento no qual organizamos a disposição das fontes de luz para a iluminação da cena, baseado num arranjo previamente desenvolvido no layout da cena ou ao longo do processo em si, dando ao profissional de iluminação a oportunidade de reorganizar e reposicionar as luzes no espaço tridimensional.

A quarta etapa é o momento em que as características das superfícies dos objetos são definidas, como cor, textura, brilho, refletividade e transparência. Esta fase é crucial para o resultado da renderização, pois tem um grande impacto na qualidade e refinamento da imagem final. A quinta etapa consiste em definir um método de sombreamento e gerar a imagem final renderizada. Vale aqui lembrar que as características da superfície e o sombreamento são duas etapas distintas, mas estão intrinsecamente relacionadas e muitas vezes se sobrepõem. O sombreamento das superfícies vai proporcionar como elas interagirão com a luz que as atinge. O efeito resultante é o descrito na quarta etapa

Dependendo do tipo de imagem que se quer gerar, esse processo pode ser tornar muito complexo e muitas vezes é necessário renderizar diferentes componentes da cena separadamente. Isso é chamando de renderização em camadas (*render layers*). Um exemplo simples desse método consistiria em renderizar o fundo sozinho (*background*) e, em seguida, renderizar os elementos do primeiro plano e, finalmente, combiná-los utilizando softwares de composição (Kerlow 2000).

Uma das técnicas de criação de imagens que podem acelerar a renderização é conhecida como Remoção de Superfícies Escondidas (*Hidden Surface Removal*), onde linhas e superfícies que estão ocultas, não sendo observadas pela câmera virtual, não farão parte do pacote de dados oferecidos ao processamento, resultando da sua remoção do cálculo da cena renderizada final.

A ocultação de geometrias, ou parte delas, ainda assim, não é a solução definitiva pra acelerar o render. Nesse caso o benefício seria, então, uma redução de cálculos de polígonos, que reduziria a quantidade de processamento. De fato o processamento se tornaria mais rápido, mas não o suficiente para se obter um render em tempo real.

Um dos primeiros métodos de síntese de imagens bem-sucedidos começou com uma ideia da literatura da física. Ao projetar lentes, os físicos tradicionalmente traçaram o caminho percorrido pelos raios de luz começando em uma fonte de luz, passando pela lente e indo além dela. Este processo de seguir os raios de luz foi chamado de rastreamento de raio (*ray tracing*)

(Glassner, 1989). Este processo é baseado no modelo da câmera do tipo *pinhole* (buraco de agulha) e sua matriz de projeção é mais simplificada, mas pode incluir posteriormente outros efeitos, como as aberrações introduzidas pela espessura da lente. Apenas os fótons que eventualmente atingem os olhos são os que realmente contribuem para a formação da imagem. “Essa luz define as características de cada objeto e a forma dele, através de suas linhas de contorno, sua superfície, seu volume e sua cor, revela-se pela luz que eles emitem, sendo percebidos pelos diversos sensores dentro do olho” (Gomide 2014).

Os fótons que estão deixando a fonte de luz, atingindo os objetos e sensibilizando a retina dos olhos descrevem o rastreamento de raios. Esse processo é descrito por (Glassner 1989) como rastreamento de raio direto.

Mas o processo de rastreamento de raio direto não é o método mais eficiente para se criar uma imagem porque, seguindo o fóton do seu ponto gerador, pode-se inutilizar a trajetória de um deles que não está atingindo nenhum objeto, como, por exemplo, saindo por uma janela. Para tornar o processo mais eficaz o caminho reverso é considerado mais adequado para que o cálculo de rastreamento de raio possa efetivamente tornar o render mais rápido. Dessa forma, os raios que interessam à cena são aqueles que realmente atingiram os objetos e são enquadrados e, então, seu caminho é seguido até a fonte de luz. Esse é rastreamento de raio indireto ou reverso, também conhecido como *path tracing*. Para (Glassner 1989) esse processo, associado ao rastreamento de raio direto, contribui eficazmente para a geração de imagens realistas produzidas no ambiente virtual.

Um outro processo de renderização, conhecido como z-buffer, considera, de acordo com (Kerlow 2000) técnicas que abordam o espaço de imagem, mas incorpora também o espaço de objetos. Essas informações de profundidade são mantidas em um buffer e disponibilizada para o processo de renderização à medida que os cálculos de remoção de superfície são realizados.

Tanto o rastreamento de raio quanto o Z-Buffer são levados em consideração no momento de cálculo, mas o resultado não depende somente desses fatores. Existe um outro efeito natural que precisa ser simulado para tornar a cena mais realista. Esse efeito é chamado de radiosidade e Iluminação global.

A iluminação global pode criar imagens que são fisicamente mais precisas do que qualquer outra técnica de renderização, porque calcula a iluminação indireta em objetos, incluindo reflexão difusa e especular entre superfícies e transmissão de luz de outros objetos (Kerlow 2000). Iluminação indireta se refere ao fóton que é rebatido por uma superfície e alcança outra, ou seja, os objetos não estão sendo iluminados diretamente pela fonte de luz. No software de render, se leva em consideração um número máximo de rebatimento (*bounce*) dos raios nos objetos.

A radiosidade se concentra na inter-reflexão difusa entre as superfícies e normalmente divide a geometria em áreas de polígonos de acordo com a maneira como a luz os afeta. Os polígonos são classificados em fontes de luz, superfícies de recepção de luz e superfícies de bloqueio de luz e a radiosidade calcula a quantidade de luz que é transferida de uma superfície para outra. Ainda dentro da análise de (Kerlow 2000), a distância entre as superfícies e sua posição angular são dois fatores importantes para estabelecer a quantidade de energia luminosa que pode ser transferida. Antes que a luz se dissipe no espaço, grande parte dela é refletida entre as superfícies se elas forem paralelas. Mas menos energia é transferida se as superfícies forem perpendiculares e nenhuma é transferida se elas estiverem de costas uma para a outra.

Um dos efeitos mais marcantes obtidos com a radiosidade é o sangramento de cores (*color bleeding*), que ocorre quando objetos coloridos passam um pouco de sua cor através da luz difusa para os objetos vizinhos. Quando se renderiza com a radiosidade bem configurada, uma esfera vermelha brilhante, por exemplo, lançará uma inter-reflexão difusa vermelha em uma parte branca adjacente.

A radiosidade é ainda complexa de ser configurada e quando o número de vezes que a luz é permitida para refletir dentro do ambiente é limitado, ela não se torna forte para resolver todos os detalhes do ambiente, porque não existe amostragem de radiosidade suficiente. Isso pode causar ruídos em toda a imagem, principalmente nas áreas mais escuras. Esse artefato introduzido pela radiosidade, curiosamente, produz uma aparência semelhante à do grão da emulsão no filme cinematográfico e os diretores de fotografia veem como uma vantagem a ser abordada esteticamente em suas produções (Kerlow 2000). Essa granulação da imagem causada por esse efeito pode ter uma representação artística útil, mas, na maioria das vezes, é preciso removê-la, já que na realidade eles não ocorrem.

Os softwares de renderização offline abordados nesse estudo tem a capacidade de remover os grãos através de um processamento interno que utiliza algoritmos de retirada de ruído, chamados *denoise*. Para executar esse algoritmo, o renderizador deve gerar a imagem granulada primeiro para, em seguida, a converter em uma imagem sem o ruído. Simplificadamente, o algoritmo analisa cada grão da cena e os compara com os pixels mais próximos e induz um desfoque no grão baseado na cor predominante da região. O resultado de tempo final deve ser levado em consideração para se determinar em quanto tempo realmente o render ficará pronto.

Metodologia e resultados

A partir do desenvolvimento do curta-metragem de animação intitulado *A primeira perda da minha vida*, esta pesquisa utiliza algumas cenas onde os elementos 3D afetam mais fortemente o processo de render e aplica alguns quadros desse curta metragem aos softwares de render escolhidos. Dessa forma, se pode obter parâmetros de comparação relacionados ao tempo que cada renderizador demorou para formar a imagem e às diferenças na qualidade das imagens geradas.

Para os testes, separou-se o curta-metragem em três grupos de sequência. O primeiro grupo são as sequências no Jardim Botânico. Para esse conjunto de sequências, se utilizou o renderizador V-Ray. As sequências no escritório de Kafka formam o segundo grupo, renderizadas com o Arnold. O terceiro grupo, das sequências com a Boneca, foi todo renderizado utilizando o motor gráfico Unreal. O resutado dos diferentes renderizadores no curta-metragem de animação estarão disponíveis online assim que a animação passar pelo circuito de festivais.

A sequência no quarto da Menina, com a Boneca, foi escolhida para aplicar diferentes renderizadores, que são o RenderMan, o Arnold e o Unreal. Isso foi feito para comparar os diferentes renderizadores na mesma sequência.

O hardware utilizado para renderizar as cenas foi um computador Intel com 2 processadores i7 XEON CPU ES-2630 2,4 GHz, 64 gigabytes de memória RAM, uma placa de vídeo Nvidia Geforce RTX 2070 e um SSD de 1 TB para executar os comandos do sistema operacional e dos softwares utilizados.

Para obter uma renderização viável de toda a animação, foi necessário equilibrar vários parâmetros dentro do software de renderização. Isso inclui o algoritmo usado para calcular os primeiros raios de rebatimento, bem como outros rebatimentos; colocando limites sobre eles; e a configuração dos níveis de nitidez. Também foi equilibrada a qualidade do antialias versus o tempo de renderização junto com os limites de ruído.

O render utilizando o Arnold gastou um tempo de aproximadamente 5 minutos para apenas 1 quadro de imagem. Esse resultado pode ser considerado rápido em relação ao Renderman, que demandou cerca de 36 minutos para conclusão de cada quadro no quarto da Menina. Para os dois renderizados utilizou-se apenas a CPU, porque as licenças utilizadas eram educacionais.

Apesar da grande diferença de tempo entre o Arnold e o Renderman, a imagem final de ambos ficou muito bem definida, apesar de alguns problemas terem sido detectados no Renderman, principalmente nos materias com transluscência, como a cortina do quarto. No Arnold, alguns problemas surgiram na geração de sombras de oclusão.

O tempo de render na Unreal Engine 5 foi consideravelmente mais rápido que no Arnold, podendo ser classificado como um render em tempo real. A Unreal Engine 5 utiliza a GPU para o render, o que aumenta a velocidade com que o software pode finalizar a imagem. Entretanto, a imagem final ainda mantém uma aparência convencional de jogos, com um foco doce e um brilho artificial, para mascarar problemas de definição da imagem. Um problema observado se relaciona às sombras de oclusão, que são menos eficazes no Unreal.

No V-Ray, uma das escolhas mais significativas para este trabalho foi usar *Brute Force* e *Light Cache* como mecanismo primário e secundário, respectivamente, para calcular a Iluminação Global. Enquanto o *Brute Force* recalcula os reflexos de luz primária para cada ponto sombreado separadamente e independentemente, resultando em um método muito preciso e fotorrealista, o *Light Cache* compensa os seguintes saltos, usando um método de aproximação, equilibrando o tempo de renderização, em troca de precisão. O tempo de render dos primeiros quadros da animação, com um grande plano geral do Jardim Botânico, foi de aproximadamente 45 minutos por quadro, mesmo utilizando a GPU. Isso se deve à quantidade de detalhes no enquadramento, que aumenta com as árvores com muitas folhas e toda a vegetação. Para os quadros mais próximos da estufa, quando a grua já está mais baixa, o tempo de render foi de 15 minutos.

Para melhorar o rendimento da renderização, todos os renderizadores deste trabalho utilizam o rastreamento de raio reverso. O algoritmo de *path tracing* ainda seleciona os raios de luz que seguem direções semelhantes e os agrupa para serem um único elemento a ser calculado. Isso permite que esses renderizadores otimizem o processo, pois eles irão, além de desprezar os raios que não colidem com nenhum objeto, reduzir a quantidade de raios, diminuindo o tempo de cálculo. Como efeito positivo do rastreamento de raio reverso, somente o que está sendo enquadrado pela câmera será processado durante a renderização.

Buscou-se utilizar os mesmos parâmetros em cada renderizador. No RenderMan é possível optar pelo uso de *path tracing* ou seis tipos diferentes de rastreamento de raios. Dentro dessa gama de escolhas, o artista deverá levar em consideração o tipo de imagem que deverá ser gerada. Estes 7 tipos de rastreamento de raios estão armazenados em um integrador e cada um deles tem características específicas, como o PxrOcclusion, para renderizar apenas as sombras de oclusão, ou o PxrVCM, mais indicado para cenas que contêm superfícies que vão gerar um eleito óptico chamado de cáustico. Na cena escolhida para esse trabalho, o integrador foi o PxrPathTracer pois ele foi o que representou a cena com mais fidelidade, tanto na iluminação como no comportamento dos materiais.

Na figura 1 se observa o resultado com o renderizador RenderMan, com a imagem à esquerda e o histograma à direita. O Renderman apresenta um bom cálculo das sombras diretas e indiretas e produziu uma iluminação indireta de boa qualidade. As texturas aplicadas por esse renderizador estão bem definidas. Na configuração que utilizamos, se observa a imagem com contraste, com o facho de luz da iluminação externa e definido e o efeito de translucidez na cortina.

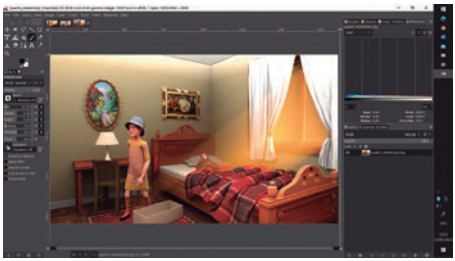

Figura 1 – imagem do quarto renderizada com o RenderMan, à esquerda, e o histograma, à direita.

A arquitetura do renderizador Arnold inclui o rastreamento de raio reverso em seu sistema, com a finalidade de facilitar seu manuseio e configuração. São poucos parâmetros a serem trabalhados

Na figura 2 se observa um quadro da imagem do quarto da Menina, com o histograma ao lado, obtidas com o Arnold. O resultado apresentou um bom contraste, menor que do RenderMan, detalhes da translucidez da cortina e o facho de luz bem definido provocado pela iluminação externa.

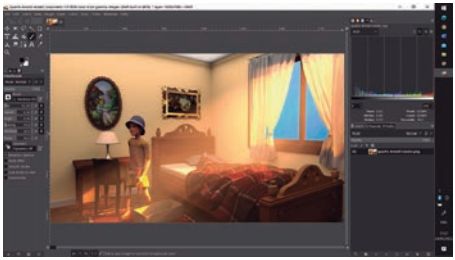

Figura 2 – imagem do quarto renderizada com o Arnold, à esquerda, e o histograma, à direita.

O Unreal Engine 5 trabalha com o rastreamento de raio de maneira semelhante aos outros renderizadores, mas ela precisa de uma definição mais específica de qual comportamento determinado material deve ter. Todas as superfícies lisas e polidas precisam ter um objeto para capturar os reflexos da cena e projetar sobre essas superfícies. O Unreal Engine 5 utiliza a placa de vídeo do computador para realizar esses cálculos e isso dá a esse software uma vantagem no que se refere à velocidade de processamento de render. Por esse motivo o Unreal está atraindo a atenção de grandes produtores de filmes pela rapidez na criação de imagens digitais.

Na figura 3 temos o resultado de render com o Unreal Engine. Observamos um bom contraste, as cores intensas, menos translucidez na cortina e o facho de luz com a iluminação externa quase inexistente. Isso se deveu a utilizarmos os mesmos parâmetros para os diferentes renderizadores. No entanto, ajustes feitos diretamente nos parâmetros do Unreal permitem melhorar ainda mais a qualidade da imagem. O render de cada sequência em que foi utilizado foi feito em menos de um minuto. Foi a melhor relação custo/benefício encontrada de todos os softwares.

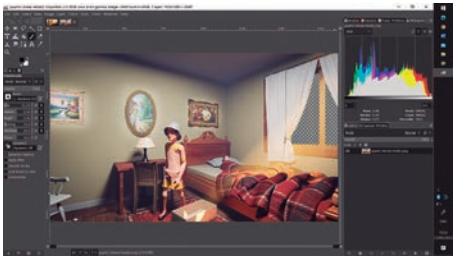

Figura 3 – imagem do quarto renderizada com o Unreal Engine, à esquerda, e o histograma, à direita.

Os renderizadores físicos possuem um algoritmo que trata os objetos poligonais como de alta poligonagem na imagem final de render, mesmo que esses objetos tenham uma baixa contagem de polígonos. Isso é muito eficiente, visto que é possível trabalhar com uma cena leve que não vai exigir muito processamento, principalmente das placas de vídeos, mas, na renderização, esses mesmos objetos são suavizados e tratados como objeto de densidade poligonal bem maior. Esse recurso ainda não está presente no Unreal Engine 5, o que prejudica a qualidade da imagem final. Para se ter uma imagem que não carregue essa deficiência, é necessário importar as geometrias já com uma alta contagem poligonal, prejudicando um pouco a performance do render em tempo real, já que agora o aplicativo irá calcular uma quantidade muito maior de polígonos. Nesse caso, o Unreal Engine 5 traz uma solução viável chamada Nanite, que pode tratar modelos muito densos de forma mais suave, sem muitos atrasos no render, que ainda continua em tempo real. Houve uma perda na aparência dos objetos cromados e dourados dentro do Unreal Engine. Esses materiais parecem não refletir adequadamente o ambiente.

A composição final dos renders, com essas configurações de luz e materiais revela que os renderizadores físicos (V-Ray, Arnold e RenderMan) ainda estão processando a iluminação de uma forma mais eficaz.

Conclusão.

Os renderizadores pós-processados ainda apresentam o melhor resultado de qualidade de imagem ao serem aplicados. As maiores desvantagens são a necessidade de hardware com muita capacidade de processamento, elevando o custo da produção, assim como o planejamento do tempo que se vai passar renderizando, de acordo com o hardware disponível. No caso do curta-metragem de animação, as cenas realizadas com o V-Ray, no Jardim Botânico, utilizando o iToo para gerar a vegetação, traz um resultado muito realista e o renderizador se apresentou como o de melhor qualidade da imagem, apesar do tempo longo de render.

Nos testes com a sequência do quarto da Menina, o RenderMan e o Arnold apresentaram qualidade melhor de imagem que o Unreal Engine, mas que não fazem um diferencial significativo, se utilizarmos os parâmetros na configuração do software e do hardware otimizados. O tempo de renderização do Unreal Engine, com ajustes mais adequados para o motor, o torna imbatível em praticidade e rapidez de resultados, com uma qualidade de imagem muito satisfatória. Tivéssemos utilizado o Unreal Engine em todo o trabalho, adotando o fluxo exigido em todas as cenas, e teríamos o curta-metragem finalizado em um tempo muito menor e com a qualidade que pretendíamos.

O Unreal será adotado como o renderizador de preferência nos próximos trabalhos e já vem sendo utilizado em diversas animações realizadas nos bacharelados de Computação Gráfica e de Design de Games da Universidade FUMEC.

Agradecimentos

Os autores agradecem o apoio financeiro do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), da Fundação de Amparo à Pesquisa do Estado de Minas Gerais (Fapemig) e do Fundo Estadual de Cultura da Secretaria da Cultura do Estado de Minas Gerais.

Bibliografia

Akenine-Möller, T., Haines, E., Hoffman, N., Pesce, A., Iwanicki, M., & Hillaire, S. (2018). Real-Time Rendering Ray-Tracing Addendum, Fourth Edition. 1198 pages.

Andaló, F., Vieira, M. H., & Merino, E. (2010). Iluminando objetos 3D: iluminação tradicional versus iluminação realista. Design e Tecnologia, 1(02), 44. https://doi.org/10.23972/det2010iss02pp44-54 , acessado em 20 de abril de 2002.

Arnold. 2022. Sítio oficial do renderizar Arnold, em https://arnoldrenderer.com/ , acessado em 18 de abril de 2022.

Autodesk. 2022. Disponível na documentação do software Autodesk Maya 2022, em https://help.autodesk.com/view/MAYAUL/2022/ENU/?guid=GUID-FDCD0C68-2496-4405-A785-3AA93E9A3B25, acessado em 20 de abril de 2022.

Autodesk Maya. 2022. Sítio oficial do software Maya, em https://www.autodesk.com/products/maya/overview?term=1-YEAR&tab=subscription , acessado em 20 de abril de 2022.

Audesk 3ds Max. 2022. Sítio oficial do software 3ds Max, em https://www.autodesk.com/products/3ds-max/overview?term=1-YEAR&tab=subscription, acessado em 20 de abril de 2022.

Autodesk Arnold. 2022 Sítio do renderizador Arnold, em https://www.autodesk.com/products/arnold/overview , acessado em 20 de abril de 2022.

Cowley, D.. 2015. SIGGRAPH Recognizes 'A Boy and His Kite' with Best Real-Time Graphics and Interactivity Award, disponível em https://www.unrealengine.com/en-US/blog/siggraph-unreal-kite-best-real-time-graphics-interactivity-award, acessado em 20 de abril de 2022.

Dicionário Online de Português. 2022. Verbete renderizar disponível em https://www.dicio.com.br/renderizar/, acessado em 18 de abril de 2022.

Dutra da Silveira Neto, W. 2019. Técnicas de modelagens e renderização em softwares tridimensionais. DAPesquisa, Florianópolis, v. 2, n. 4, p. 388-402. DOI: 10.5965/1808312902042007388. Disponível em: https://revistas.udesc.br/index.php/dapesquisa/article/view/16629. Acesso em: 18 abril de 2022.

Farris, J.. 2020. Forging new paths for filmmakers on "The Mandalorian", disponível no site do Unreal Engine, em https://www.unrealengine.com/en-US/blog/forging-new-paths-for-filmmakers-on-the-mandalorian, acessado em 20 de abril de 2022.

Glassner, A., 1989. An Introduction to Ray Tracing, Morgan Kaufmann Publishers.

Gomide, J. V. B.; Tannure, M. ; Ribeiro, H. L. . 2014. Produção de Animação com a Captura de Movimento: um. Estudo de Caso. In: Avanca|Cinema 2014, Avanca.

Gomide, J.V.B (2014). Imagem Digital Aplicada: Uma abordagem para estudantes e profissionais. São Paulo: Elsevier Editora.

iToo Software. 2022. Sítio oficial do iToo Software em https://www.itoosoft.com/ , acessado em 20 de abril de 2022.

Kadner, N. 2019. The Virtual Production Field Guide. Epic Games, 98 p, disponível em https://cdn2.unrealengine.com/Unreal+Engine%2FvpfieldguideV1.2.02-5d28ccec9909ff626e42c619bcbe8ed2bf83138d.pdf, acessado em 20 de abril de 2022.

Kerlow, I.A.. 2000. The Art of 3-D Computer Animation and Imaging, John Willey and Sons.

Liptak, Andrew (November 30, 2017). "How Neill Blomkamp and Unity are shaping the future of filmmaking with Adam: The Mirror". Dispinível em https://www.theverge.com/2017/10/4/16409734/unity-neill-blomkamp-oats-studios-mirror-cinemachine-short-film , acessado em 18 de abril de 2022.

Lopes, J.M.B. Ray Tracing.2013. Universidade Técnica de Lisboa.39 p.

Moore, Gordon E.. 1965. "Cramming more components onto integrated circuits" (PDF). intel.com. Electronics Magazine. Acessa em 20 de abril de 2022.

RenderMan. 2022. Sítio oficial do software, desenvolvido pela Pixar, em https://renderman.pixar.com/ , acessado em 18 de abril de 2022.

Unreal. 2022. Curso "An In-Depth Look At Real-Time Rendering" do fabricante do software, Epic Games, em https://dev.epicgames.com/community/learning/courses/EGR/an-in-depth-look-at-real-time-rendering/edk/an-in-depth-look-at-real-time-rendering , acessado em 18 de abril de 2022.

V-Ray. 2022. Informações sobre a versão estudantil do software V-Ray, no sítio do seu fabricante, Chaos, em https://www.chaos.com/education/buy-online?gclid=CjwKCAjw4ayUBhA4EiwATWyBrn3PL4K7Tj7APAgwA-Zfgi546_3_I2lArF8lrODxw_qmydD2eILnUxoCCgAQAvD_BwE , acessado em 18 de abril de 2022.