As imperfeições causadas pelo Aliasing nos jogos e os métodos de *Anti-Aliasing* utilizados para resolução destas falhas

Luan Pedro R. Coimbra[1], José Carlos R. da Silva[1], Ennio Willian L. Silva[1]

[1]Instituto Federal do Tocantins (IFTO) - Campus Araguatins

{luaniic63@gmail.com, josecrs12@gmail.com, ennio.silva@ifto.edu.br

Abstract. *The present paper provides a brief analysis of the aliasing effects caused by the aliasing problem that frequently occur in electronic game images, as well as the main reasons behind these graphic errors. From this perspective, this paper aims to present the main ways to "solve" this problem, as well as a comparative analysis between these so-called anti-aliasing algorithms that were explained and contrasted according to cost and performance. The analysis showed that these two elements should be aligned for the best cost-benefit.*

Resumo. *O presente artigo traz uma breve análise sobre os efeitos serrilhados causados pelo problema de aliasing que frequentemente ocorrem nas imagens de jogos eletrônicos, bem como os principais motivos por trás destes erros gráficos. Sob esta perspectiva este trabalho tem como objetivo apresentar as principais formas de "solucionar" este problema, assim como uma análise comparativa entre estes algoritmos denominados anti-aliasing que foram explanados e contrastados de acordo com o custo e desempenho. A análise mostrou que estes dois elementos devem estar alinhados para o melhor custo-benefício.*

1. Introdução

Tudo o que é possível ser visto pela tela de nosso computador, sejam elas imagens, uma palavra ou até mesmo uma linha, é constituída por um agrupamento de *pixels*, que são pequenas formas quadradas. Ao observamos alguma figura, como por exemplo, uma forma circular em nosso computador como uma bola, consideramos que as bordas de tal objeto são lisas. No entanto, quando passamos a observar mais minuciosamente em torno dela, podemos notar que as bordas parecem estar desfiguradas como se fossem serras ou dentes. A partir disso é possível perceber que tal imagem não é realmente lisa ou "perfeita". Estas imperfeições que ocorrem quando a amostragem inerente à renderização não tem informação suficiente para uma imagem precisa são denominadas *aliasing*.

> Podemos ver o fenômeno de *aliasing* em outras situações como ao filmarmos uma roda de um carro rodando mais rápido do que a câmera é capaz de capturar, neste caso vemos a roda girar para trás, justamente por que a taxa de amostragem está muito baixa, no nosso caso a resolução da imagem (SCURI.1999, p. 20).

Para contornar os problemas causando pelo *aliasing*, existem os métodos denominados *Anti-Aliasing* (*AA*). Estes procedimentos estão presentes principalmente em jogos computacionais e possuem como objetivo melhorar a renderização de imagens e cenários através da aplicação de filtros de suavização. Ao ser verificada de perto, a impressão é que os contornos estão borrados, mas vendo a uma distância maior, a

qualidade é impecável.

Sob esta perspectiva, este trabalho tem como objetivo apresentar os conceitos inerentes ao *aliasing*, bem como uma análise comparativa entre os principais métodos *anti-aliasing* utilizados em jogos computacionais.

2. Referencial Teórico

2.1. Aliasing

Segundo Teixeira (2010) o *Aliasing* é um fenômeno que provoca o surgimento de elementos em uma imagem quando a taxa de amostragem é muito baixa para capturar as frequências altas de um determinado sinal. Essas frequências altas, ao invés de desaparecerem, surgem em localidades indesejadas na imagem.

Figura 1: A esquerda um pilar e a direita chamas, ambos os frames com aliasing. Fonte: Autor.

De acordo com Scuri (1999) o *aliasing* é uma ocorrência muito comum em amostragens de imagens e para Teixeira (2010) as principais causas de *aliasing* em imagens computacionais são: a descontinuidade nas bordas, distorção de tonalidades e sobreposição das bordas dos objetos com outras superfícies.

2.2. Anti-Aliasing

O *anti-aliasing* ou antisserrilhamento é um procedimento de diminuição do efeito causado pelo *aliasing* também conhecido como serrilhado. Com o objetivo de suavizar as bordas de uma imagem, o *anti-aliasing* é um método que reduz a distorção das imagens, uma vez que a tela de um monitor é composta por pixels que por serem quadrados faz com que as curvas não fiquem perfeitas mostrando assim, uma imagem distorcida (GONÇALVES, 2004).

Figura 2: A esquerda um pilar e a direita chamas, ambos os frames com anti-aliasing. Fonte: Autor.

Os métodos *anti-aliasing* tem como propósito minimizar o efeito de serrilhado,

criando uma ilusão visual na qual as bordas gráficas da imagem aparentam bem polidas e perfeitas a olho nu. "O *anti-aliasing* (suaviza as bordas de uma imagem), sombras volumosas, mapeamento de imagens, efeitos de movimento, transparência, reflexos, texturas em 3D" (ARAÚJO et al, 2008, p. 73).

Essencialmente, esse método trabalha com a variedade das cores das bordas pixeladas de maneira bem suave, criando um efeito que, quando analisado pelo olho humano, passa a impressão das linhas diagonais serem perfeitas, porém os pixels continuam como escadas já que não é possível pintar um pixel pela metade.

Sendo assim, resolver completamente tal problema causado pelo *aliasing* é fisicamente impossível, pois o método de *anti-aliasing* é apenas um meio de enganar os olhos de quem vê tal imagem a uma certa distância. Basicamente esse método tem apenas a ação de minimizar o efeito de serrilhado causado a determinadas imagens.

2.3. As evoluções nos aspectos visuais dos jogos eletrônicos.

Um dos assuntos mais falados no mundo dos jogos é sobre importância dos gráficos para tornar a imersão e a jogabilidade mais realista. Mas, para chegar a tal resultado demonstrado nos dias de hoje, os desenvolvedores tiveram um árduo trabalho. Um dos principais problemas remontam o início da era de desenvolvimento de jogos, eram as configurações bem limitadas dos computadores, as quais eram muito inferiores aos atuais.

> Os videogames antigos, principalmente aqueles que rodavam nas plataformas de até 16 bits, ou seja, os consoles lançados os anos 70 e 80, cuja imagem tinha poucos pixels e cores, apresentavam muitas limitações, permitindo a suposição de que contrato de interface fosse menos efetivo (PFUTZENREUTER, 2014, p.14).

Como os próprios jogos antigos já eram feitos em 8 ou 16 bits, todos naturalmente não possuíam o problema de *aliasing*, uma vez que os *frames* dos jogos eram todos pixelados. Entretanto, com o passar dos anos este fato mudou. Os jogos ganharam gráficos mais realistas e juntamente com este avanço, surgiu o problema do serrilhamento. Para a resolução deste problema foi criado o *anti-aliasing* no intuito de incrementar a riqueza visual dos jogos eletrônicos.

2.4. Técnicas de Anti-Aliasing utilizadas em jogos

O *Anti-Aliasing* (*AA*) é subdividido em dois tipos de filtragem. O primeiro tipo de filtro consiste em aumentar a taxa de amostragem. Os filtros que utilizam este algoritmo são: *SSAA*, *MSAA* (*Super Sampling Anti-Aliasing e Multi Sampling Anti-Aliasing* respectivamente).

O *Super-Sampling Anti-Aliasing* ou simplesmente "*SSAA*" como é conhecido, é um dos vários métodos de aplicação do *anti-aliasing*. É uma maneira alternativa de executar as correções dos defeitos causados pelo *aliasing* de bordas serrilhadas com um filtro que traz a suavização das imagens presentes, renderizadas em programas e games das mais variadas plataformas encontradas.

Segundo Castro (2015) *SSAA* é a técnica original de *AA*, na qual a imagem é renderizada em uma escala de resolução maior do que a original, e na qual posteriormente é aplicado o *downsampling*. Esta escala é relacionada aos níveis disponibilizados, o mais básico é o SSAA 2x. Como exemplo desta técnica, temos uma imagem na resolução de

800x600, que passará a ser renderizada em 1600x1200 e depois reduzida para 800x600 tornando-se assim uma imagem mais detalhada e menos serrilhada.

Dessa forma, significa que para cada pixel presente na imagem, foram convertidos quatro pontos e depois os mesmos foram combinados. O *downsampling* é o procedimento de redução da taxa de quadros de uma imagem. O método é meramente feito, separando uma determinada proporção de amostragem da imagem a cada pixel. "A operação de *downsampling* é onde começa a acontecer a redução de dados necessários para armazenar a informação da imagem ou, em outras palavras, começa a acontecer a compressão" (AGOSTINI, 2001, p.3). O resultado que isso traz, é uma imagem com variações bem mais suaves entre os contornos dos objetos, no caso, os pixels que anteriormente estavam com efeito de serrilhado, mas vale ressaltar que existem perdas nesse processo e que podem se tornar irreversíveis.

Os mecanismos específicos do *SSAA* são identificados de acordo com a resolução utilizada pelo mesmo. Como por exemplo: 2x significa que o método duplicou a resolução dos pixels na horizontal e preservou a resolução vertical sem alterações, dessa forma resultando em que cada pixel final presente na imagem é feito a partir do intermédio de dois pixels e quanto maior for a numeração demonstrada no final, melhor será o resultado trazendo uma imagem cada vez mais nítida.

O *MSAA* funciona de forma diferente. Ele reduz o poder de processamento necessário em comparação ao *SSAA*, pois ele detecta as bordas dos polígonos nos objetos 3D e antes da renderização aumenta a amostragem na divisão entre eles. A maior vantagem do *MSAA* é que ele gera uma imagem mais limpa em comparação aos demais e menos custoso que o *SSAA*. A desvantagem é que algumas *engines* com uso elevado de *shaders* têm problemas ao usar *MSAA*, já que novas bordas são criadas para deformar uma superfície plana.

Para Castro (2015) este tipo de *anti-aliasing* é uma versão "menos potente" do *SSAA*. Diferente do *SuperSampling* ele processa um pixel por vez melhorando o processo e diminuindo o consumo de recursos. Por outro lado, quando há pixels com um serrilhado muito deformado, este método não consegue tratá-lo e quando isto ocorre, o *SSAA* é utilizado somente nestas partes em específico. Tendo em vista que menos recursos são utilizados a imagem não obterá os mesmos resultados quando comparado o uso do *SuperSampling*, entretanto consegue corrigir grande parte das imperfeições.

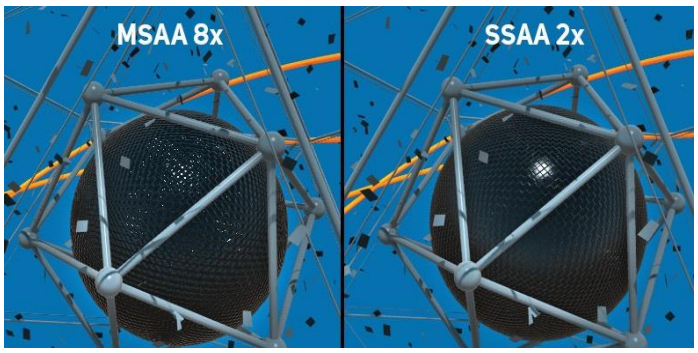

Figura 3: A esquerda um frame com MSAA aplicado e a direita um frame com SSAA aplicado. Fonte: <https://forum.unity.com/threads/released-flowfire-super-sampling-ssaa-visually-best-anti-aliasing.332628/>

O segundo tipo de filtro consiste em borrar (*blur*) os serrilhados que também é chamado de *AA* pós processamento. As técnicas que utilizam este algoritmo são: *FXAA* e *SMAA* (*Fast Approximate Anti-Aliasing*, *Subpixel Morphological Anti-Aliasing*.

O *FXAA* consiste em um dos modos de pós-renderização que trabalha com “filtro de borrão”. A priori, ele detecta o contraste das bordas serrilhadas no frame capturado e em seguida "borra" nas proximidades da borda, o que resulta numa redução notável dos serrilhados. Este algoritmo não faz nenhuma diferenciação entre borda e textura e com isto, também "borra" as texturas. É importante frisar que o *FXAA* é a forma de *AA* mais barata e é principalmente utilizada nos consoles (PS4, XBOX, etc.).

> Fast Approximate Anti-Aliasing (FXAA) – trata-se de uma versão post-process de AA, ou seja, o AA é aplicado apenas depois da imagem renderizada, ao contrário das técnicas melhores como por exemplo MSAA. Tem uma grande compatibilidade visto que se trata de um efeito aplicado posteriormente. Contudo a melhoria não é muito substancial comparado com as outras técnicas. (CASTRO, 2015, p. 36.)

O *SMAA* pode ser considerado a união do *FXAA* e *MSAA*, ou seja, ele é um híbrido entre pré e pós processamento. Do *FXAA* é utilizado o filtro de pré-processamento baseado em borrão, já a detecção de serrilhados é baseada na técnica *MSAA*. Mesmo com a união destas técnicas distintas o preço de implementação é relativamente baixo. O custo-benefício é o melhor dentre as técnicas apresentadas anteriormente, e por isso é considerado por muitos usuários a melhor técnica de *AA* uma vez reduz muito bem os serrilhados e ao mesmo tempo não borra demasiadamente a imagem.

Figura 4: A esquerda o frame sem Anti-Aliasing, ao meio o frame com FXAA, a direita o frame com SMAA. Fonte: <https://forums.anandtech.com/threads/latest-guild-wars-2-gpu-performance-data.2265540/page-4#post-33876153>

3. Conclusão

Existem vários algoritmos e/ou técnicas para tratarem os serrilhados nas imagens, e estas estão divididas em duas categorias, os que aplicam o *Anti-Aliasing* antes de processarem as imagens e os que utilizam após o processamento destas. Cada uma das categorias citadas possui vantagens e desvantagens. Os algoritmos que utilizam filtros de pré-processamento aplicam melhorias nos serrilhados de forma excepcional se comparada ao de pós-processamento, entretanto devido a essa “melhora excepcional” uma quantidade de poder de processamento enorme é necessária para que ele seja aplicado, tornando-se assim muito caro para ser aplicado.

Por outro lado, os filtros de pós-processamento requerem um poder de processamento gráfico bem menor, uma vez que estes não eliminam os serrilhados, mas sim borram (*blur*) as bordas que contém serrilhados para que haja a impressão de que eles de fato, não estejam ali. Entretanto, este método tem várias imperfeições, e uma delas é que o algoritmo não distingue as texturas da imagem. Mesmo com estas falhas, este é o que tem o melhor custo benefício em relação aos algoritmos de pré-processamento, pois é

bem mais "leve" e por isso comumente utilizados nos consoles (PS4, XBOX, etc.).

E no meio termo entre pré e pós processamento há o *SMAA*, um hibrido que utiliza parte de ambos os algoritmos. Nele o *AA* é aplicado de forma pré-processada, ou seja, ele aumenta a resolução da imagem e posteriormente faz *downsampling*, melhorando os serrilhados sem borrá-los (*blurring*) e a detecção dos serrilhados é feita através dos polígonos da imagem. Ambos são utilizados de maneira que não impactem no custo-benefício de forma agressiva, uma vez que este algoritmo cria uma imagem com qualidade superior à dos algoritmos de pós-processamento e tem o método de detecção de *aliasing* dos algoritmos de pré-processamento. Esta combinação juntamente ao preço não tão alto é o que faz este algoritmo de anti-aliasing ser considerado o melhor dentre os demonstrados.

Referências

AGOSTINI, Luciano; BAMPI, Sergio. **Arquitetura Integrada para Conversor de Espaço de Cores e Downsampler para a Compressão de Imagens JPEG**. In: VII Workshop IBERCHIP**.** 2001. Disponível em:<https://www.researchgate.net/profile/Luciano_Agostini/publication/242088577_Projet o_de_uma_Arquitetura_de_DCT_1D_para_a_Compressao_de_Imagens_JPEG/links/00b7d53965eed3a6ec000000.pdf> Acesso em: 18 fev. 2018.

ARAÚJO, Rogério Henrique C.; SILVA, Wilton Lacerda. **Visualização Síncrona de Processos com o OpenGL**. Ciência & Desenvolvimento-Revista Eletrônica da FAINOR, v. 1, n. 1, p. Pág. 72-77, 2008. Disponível em:<http://srv02.fainor.com.br/revista/index.php/memorias/article/download/21/17> Acesso em: 28 fev. 2018

CASTRO, Diogo Ferreira De. **Estudo Focado no Utilizador de Software Gratuito para Modelação e Visualização 3D Realista**. 2015. Dissertação de mestrado. Disponível em: <http://recipp.ipp.pt/bitstream/10400.22/8142/1/DM_DiogoCastro_2015_MEI.pdf> Acesso em: 27 fev. 2018.

PFUTZENREUTER, Edson P. **Aspectos da Imagem nos Videogames/Aspects of the Image in Video games**. Revista Hipertexto (descontinuada), v. 4, n. 3, p. 73-86, 2014. Disponível em:<http://www.latec.ufrj.br/revistas/index.php?journal=hipertexto&page=article&op=download&path%5B%5D=656&path%5B%5D=673> Acesso em: 28 fev. 2018

SCURI, Antonio Escaño. **Fundamentos da imagem digital**. Pontifícia Universidade Católica do Rio de Janeiro, 1999. Disponível em:<https://sites.google.com/site/gcamarac/bcc-326---processamento-de-imagens/apostila20imagem20digital.pdf> Acesso em 27 dez. 2017.

TEIXEIRA, João Marcelo XN et al**. Improving ray tracing anti-aliasing performance through image gradient analysis. In:** Computing Systems (WSCAD- SCC), 2010 11th Symposium on. IEEE, 2010. p. 144-151. Disponível em:<https://www.researchgate.net/profile/Veronica_Teichrieb/publication/224198219_Improving_Ray_Tracing_Anti-aliasing_Performance_through_Image_Gradient_Analysis/links/57868e1408aef321de2c6d9b.pdf> Acesso em 27 dez. 2017.