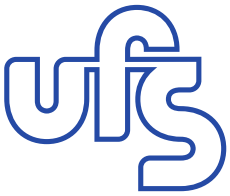


UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

DEPARTAMENTO DE COMPUTAÇÃO

Desenvolvimento de um Compilador de BRDFs em LaTeX para linguagem de shading GLSL, através da técnica Pratt Parsing

Trabalho de Conclusão de Curso

Everton Santos de Andrade Júnior

São Cristóvão – Sergipe

2024

UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

DEPARTAMENTO DE COMPUTAÇÃO

Everton Santos de Andrade Júnior

Desenvolvimento de um Compilador de BRDFs em LaTeX para linguagem de shading GLSL, através da técnica Pratt Parsing

Trabalho de Conclusão de Curso submetido ao Departamento de Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador(a): Beatriz Trinchão Andrade
Coorientador(a): Gastao Florencio Miranda Junior

São Cristóvão – Sergipe

2024

Este trabalho, além de cultural, filosófico e pedagógico
É também medicinal, preventivo e curativo
Servindo entre outras coisas para pano branco e pano preto
Curuba e ferida braba
Piolho, chulé e caspa
Cravo, espinha e berruga
Panarismo e água na pleura
Só não cura o velho chifre
Por que não mata a raiz
Pois fica ela encravada
No fundo do coração
(Falcão)

Resumo

O presente trabalho propõe o desenvolvimento de um compilador de funções de distribuição de reflexão bidirecional (BRDFs) expressas em LaTeX para a linguagem de shading GLSL, utilizando a técnica de parsing de Pratt. O objetivo é automatizar o processo de tradução de funções complexas de materiais, frequentemente descritas em LaTeX, para o código GLSL utilizado em programação de shaders para OpenGL. Ao fornecer essa ferramenta, pretende-se não apenas simplificar o trabalho dos desenvolvedores e pesquisadores na área de computação gráfica, mas também democratizar o acesso e compreensão de modelos de materiais complexos. Além disso, ao permitir que as BRDFs sejam expressas em uma forma mais familiar e acessível, como a notação matemática, o compilador reduz a barreira de entrada para aqueles que não estão familiarizados com linguagens programação. Isso pode facilitar a colaboração interdisciplinar entre profissionais de diferentes áreas, como artistas visuais, designers e cientistas de materiais, que desejam explorar e entender o comportamento visual de materiais em suas aplicações.

Palavras-chave: Compilador, BRDFs, LaTeX, GLSL, Shading, Pratt Parsing.

Lista de ilustrações

Lista de quadros

Lista de tabelas

Lista de abreviaturas e siglas

ABNT	Associação Brasileira de Normas Técnicas
abnTeX	ABsurdas Normas para TeX
DCOMP	Departamento de Computação
UFS	Universidade Federal de Sergipe

Lista de símbolos

Γ	Letra grega Gama
Λ	Lambda
ζ	Letra grega minúscula zeta
$\in$	Pertence

Sumário

1 Introdução

1.1 Contexto

Na computação gráfica, a representação realista de cenas tridimensionais depende fortemente da modelagem da luz. A interação da luminosidade incidente no objeto, bem como os materiais que compõem esses objetos, são aspectos críticos a serem considerados na geração dessas cenas [referencia]. Na prática, essa interação é frequentemente modelada por meio de funções de distribuição de refletância bidirecional, conhecidas como BRDFs.

As BRDFs, essencialmente, calculam a proporção entre a energia luminosa que atinge um ponto na superfície e como essa energia é refletida, transmitida ou absorvida [referencia]. Na renderização, essas funções são implementadas por meio programas especializados das unidades de processamento gráfico (GPUs), esses programmas são chamados de shaders, e cada API de rederização disponibiliza etapas diferentes onde esses executaveis podem ser mudados durante o processo de renderização. Esses shaders concedem a capacidade de cada objeto renderizado ter sua aparência configurada por meio de um código que implementa uma BRDF.

1.2 Motivação

Apesar da disponibilidade de linguagens específicas para a programação de shaders, que possibilitam a modificação procedimentos que representam uma BRDF, a aplicação de BRDFs na geração de shaders requer conhecimento especializado em programação [referencia?]. Essa barreira técnica pode restringir a exploração dos efeitos visuais por profissionais de áreas não relacionadas à programação. Diante disso, surge a necessidade de ferramentas mais acessíveis para a criação de shaders.

No meio acadêmico, as BRDFs são, comumente, descritas por uma fórmula escrita em LaTeX, uma abordagem promissora para atender a essa necessidade é o desenvolvimento

de um compilador capaz de traduzir BRDFs em LaTex para shaders, assim democratizando a visualização dessas BRDFs. Dado que as fórmulas são equações matemáticas, precisamos retrigir repretsentação da linguagem de entrada para o compilador afim de garatir um projeto útil em tempo ábil.

1.3 Objetivo

Este trabalho visa projetar e implementar um compilador que, a partir de funções de distribuição de refletância bidirecional escrita como equações em LaTeX, seja capaz de gerar código de shading na linguagem alvo da API OpenGL (referencia). A saída será um shader capaz de reproduzir as características de reflexão da função de refletância original, considerando a precedencia de operadores, em uma superfície tridimensional, ou, ao menos, alcançar uma aproximação satisfatória dessas características, considerando as limitações da linguagem de shading da API principalmente as representações de dados de forma discreta.

1.4 Metodologia

Para alcançar o objetivo, a sequencia das etapas adotadas serão as seguintes.

1. Realizar uma análise abrangente das áreas relacionadas ao desenvolvimento da ferramenta proposta;
2. Investigar o estado da arte no campo da compilação de BRDFs em linguagens de shading;
3. Definir a linguagem de entrada e a linguagem de saída do compilador;
4. Elaborar testes com equações LaTeX de entrada pareado com a saída em shader GLSL esperado;
5. Implementar o compilador utilizando uma linguagem de programação e tecnicas recursivas de parsing
6. Realizar a renderização de cenas utilizando o shader gerado pelo compilador.

2

Revisão Bibliográfica

Para esta seção, será conduzida uma revisão literária abrangente com o objetivo de explorar trabalhos relacionados ao desenvolvimento de compiladores para tradução de BRDFs expressas em LaTeX para a linguagem de shading, empregando, técnicas de parsing. O processo de busca será conduzido em duas etapas distintas. Primeiramente, será realizado um levantamento dos trabalhos existentes nas bases de dados com relevantes periodicos, anais de eventos, artigos e trabalhos.

Por fim, será realizada uma busca por produtos ou ferramentas similares no mercado, utilizando strings de busca específicas em repositórios digitais, especificamente GitHub, e SourceForge. Esses processos de busca permitirão identificar referências relevantes e estabelecer um panorama do estado da arte no campo dos compiladores de BRDFs para shaders, contribuindo para a compreensão do contexto acadêmico e prático no qual este trabalho se insere.

2.1 Mapeamento Sistemático

Com o intuito de obter resultados relevantes para a pesquisa, foram elaboradas frases de busca com base nos termos-chave relacionados ao tema deste trabalho. Assim como, foram criadas questões de pesquisa para guiar a seleção dos trabalhos.

2.1.1 Seleção das Bases

As bases escolhidas foram: ACM Digital Library, IEEE Xplorer Digital Library, Biblioteca Digital Brasileira de Teses e Dissertações (BDTD), Portal de Periódicos da CAPES, Google Acadêmico, esse foram escolhidos por serem acessíveis gratuitamente pela afiliação à Universidade Federal de Sergipe, já o google scholar foi escolhido para agregar pesquisas em outras bases que possam ter trabalhos relevantes. <https://bdtd.ibict.br/> <https://ieeexplore.ieee.org/> <https://www-periodicos-capes-gov-br.ezl.periodicos.capes.gov.br/>

2.1.2 Questões de Pequisa

Foram elaboradas questões de pesquisa específicas, que guiam as frases-chave que refletem os principais aspectos do tema em questão. A partir desse processo, foram identificados e selecionados os trabalhos que melhor atendiam às questões propostas, garantindo maior relevância para o estudo em questão.

1. Quais são as abordagens mais comuns utilizadas na criação de compiladores para tradução de BRDFs expressas em alguma linguagem de texto, com LaTeX, para shaders?
2. Quais as técnicas de parsing que têm sido aplicadas no desenvolvimento de compiladores para linguagens matemáticas como LaTeX?
3. O trabalho utiliza arvores, ou gramáticas livre de contexto para representar uma BRDF?
4. Quais são os principais desafios enfrentados ao traduzir funções matemáticas complexas, como as BRDFs, em shaders?
5. Quais são as ferramentas e recursos disponíveis para auxiliar no desenvolvimento de compiladores para BRDFs e shaders, e como elas podem ser integradas ao processo de desenvolvimento?

2.1.3 Termos de Busca

As frases foram contruidas considerando suas variações equivalentes através de operadores lógicos. Posteriormente, as frases de pesquisa foram adaptadas de acordo com as características individuais de cada base de dados utilizada nas pesquisas. Os termos-chave escolhidos foram: "shader", "BRDF", "compiler", "parser"e "grammar".

Bases	**Termos de Pesquisa**	**Resultados**
IEEE Xplore Digital Library	("Full Text & Metadata":brdf) AND (("Full Text & Metadata":shader) OR ("Full Text & Metadata":shading)) AND (("Full Text & Metadata":compiler) OR ("Full Text & Metadata":parsing) OR ("Full Text & Metadata":parser) OR ("Full Text & Metadata":grammar))	36
BDTD	(Todos os campos:compiler OU Todos os campos:parsing OU Todos os campos:parser OU Todos os campos:compilador) E (Todos os campos:shader OU Todos os campos:shading) E (Todos os campos:brdf)	0
CAPES Periodico	Qualquer campo contém brdf E Qualquer campo contém compi* E shad*	0
ACM Digital Library	AllField:((shader OR shading) AND brdf AND (compiler OR compiling) AND (parser OR grammar OR parsing))	46
Google Academico	("BRDF"AND ("COMPILER"OR "COMPILING") AND("PARSER"OR "PARSING") AND ("SHADER"OR "SHADING"))	69

Tabela 1:

2.1.4 Critérios

Para garantir relevancia dos resultados obtivos, seguimos os critérios de inclusão e exclusão estabelecidos, de forma que os resultados serão filtrados. Ao fim desse procedimento, apenas os resultados com maior compatibilidade com este trabalho seram análisado e descritos de maneira mais detalhada.

2.1.4.1 Critérios de Inclusão

1. Foram incluídos artigos relacionados às palavras-chaves;
2. Foram incluídos artigos que de alguma forma inclua a criação de um compilador ou um parser;
3. Foram incluídos artigos que sintetize uma arvóre como representação de BRDFs

2.1.4.2 Critérios de Exclusão

1. Foram excluídos artigos dos quais dispunham de links incorretos e ou quebrados;
2. Foram excluídos artigos que dispunham de aplicacões muito similares/repetitivas;
3. Foram excluídos artigos que não respodem as questões de pesquisa 2.1.2;
4. Artigo que não tem como entrada a BRDFs no formato de equação, ou seja, está utilizando a representação diretamente como código, também foi excluído.

5. Foram excluídos artigos que não consideram a geração de shaders como saída ou estrutura da BRDF em arvore.

6. Foram excluídos artigos que não citam BRDFs e compilador em seu resumo;

7. Se após a leitura completa, o artigo não concerne os interesse deste trabalho, esse foi excluído.

Bases	**Filtrados**
IEEE Xplore Digital Library	2
BDTD	0
CAPES Periodico	0
ACM Digital Library	1
Google Academico	1

Tabela 2: Resultados da Base após aplicar os critérios

2.1.5 Descrição dos Trabalhos Relacionados

2.1.5.1 genBRDF: Discovering New Analytic BRDFs with Genetic Programming

Neste artigo é introduzido uma framework chamada genBRDF, a qual aplica tecnicas de programação genética para explorar e descobrir novas BRDFs de maneira analitica. O processo inicia utilizando uma BRDF existente, e iterativamente aplica mutações e recombinações de partes das expressões matematicas que compões essas BRDFs a medida que novas gerações surgem. Essas mutações são guiadas por uma função fitness, que seria o inverso de uma função de erro, essa é baseada em um dataset de materiais já medidos. Por meio da avaliação de milhares de expressões, a framework identificas as viávies, que estão na Fronteira de Pareto.

A representação das BRDFs de entrada para o GA, autores geraram uma gramática que inclui constantes e operadores matemáticos comuns encontrados em equações BRDF. A gramática é compilada, e a árvore de sintaxe abstrata resultante passa por modificações realizadas pelo algoritmo genético. Nós na árvore podem ser trocados, substituídos, removidos e novos nós podem ser adicionados. Esse processo, após refinamento e análise, resulta em novas BRDFs.

Alguns dos novos modelos BRDF apresentados no documento incluem aqueles que superam os modelos existentes em termos de precisão e simplicidade.

Esse artigo se concentra principalmente em utilizar programação genética para descobrir automaticamente novos modelos analíticos de BRDF, em vez de compilar diretamente equações BRDF em linguagens de shading. Embora a representação das expressões das BRDFs possam potencialmente inspirar o nosso trabalho, o principal objetivo do artigo difere do objetivo de compilar equações BRDF para linguagem de shading.

2.1.5.2 Slang: language mechanisms for extensible real-time shading systems

O artigo descreve a linguagem Slang, uma extensão da amplamente utilizada linguagem de shading HLSL, projetada para melhorar o suporte à modularidade e extensibilidade. A abordagem de design da Slang é baseada em dois princípios fundamentais: manter a compatibilidade com o HLSL existente sempre que possível e introduzir recursos com precedentes em linguagens de programação mainstream para facilitar a familiaridade e intuição dos desenvolvedores.

O autor destaca que cada extensão da Slang visa fornecer uma trajetória incremental para a adoção a partir do código HLSL existente, evitando a necessidade de uma migração completa. Algumas dessas extensões são: funções generica, struct genericas, tipos que implementam uma dada interface assim como interfaces funcionam em Java nas para struct. Exemplo de funçao generica escrita em Slang:

```
float3 integrateSingleRay<B:IBxDF>(B bxdf,
SurfaceGeometry geom, float3 wi, float3 wo, float3 Li)
{ return bxdf.eval(wo, wi) * Li * max(0, dot(wi, geom.n)); }
```

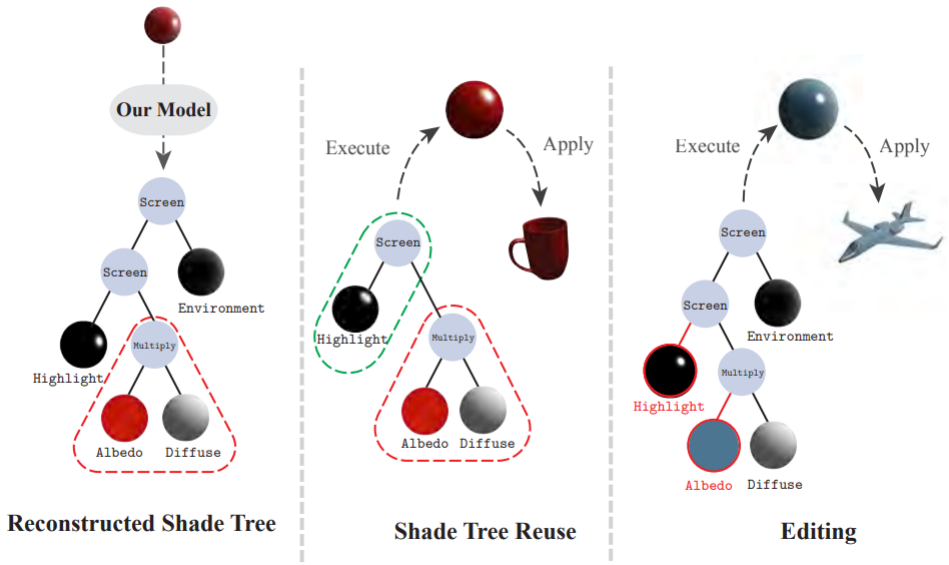
Enquanto o artigo se concentra na extensão de linguagens de shading existentes para melhorar a eficiência e a extensibilidade dos sistemas de shading em tempo real, o nosso trabalho se concentra na compilação de equações BRDF em linguagens de shading para explorar e descobrir novos modelos analíticos, mesmo para pessoas que não tem o conhecimento ténico da linguagem de shading espeficica. Embora ambos os projetos façam uso de shading e compilação, as abordagens e focos são diferentes.

2.1.5.3 Tree-Structured Shading Decomposition

O artigo propõe uma abordagem para inferir uma representação de BRDF estruturada em árvore a partir de uma única imagem para a sombreamento de objetos. Em vez de usar representações paramétricas ou medidas para modelar o sombreamento, como é comum, é proposta uma abordagem que utiliza uma representação em árvore de shading, combinando nós de sombreamento básicos e métodos de composição para decompor o sombreamento da superfície do objeto.

Essa representação permite que usuários inexperientes editem o sombreamento do objeto de maneira eficiente e intuitiva. Para abordar o desafio de inferir a árvore de sombreamento, é proposta uma abordagem híbrida que combina um modelo de inferência auto-regressivo para gerar uma estimativa aproximada da estrutura da árvore com um algoritmo de otimização para ajustar a árvore inferida. Experimentos são realizados em diversas imagens para demonstrar a eficácia da abordagem proposta.

Figura 1 – Exemplo de decomposição de BRDFs em nós de uma arvóre



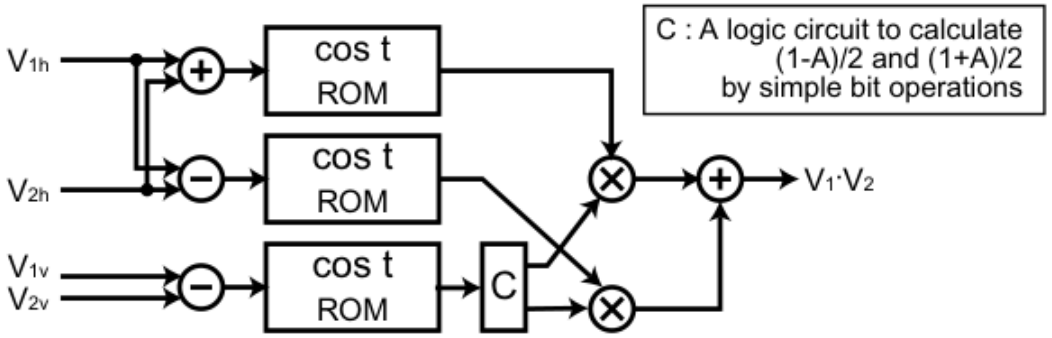
Fonte: **??**, p. 24)

Assim como o nosso trabalho, esse artigo se concentra em facilitar o processo para usuários inexperientes, pois ambos visam fornecer ferramentas acessíveis para manipular representações de sombreamento sem exigir conhecimento avançado em programação de shading. Esse artigo também emprega uma representação em árvore, embora para um propósito diferente. Enquanto o nosso trabalho utiliza árvores para representar expressões matemáticas de BRDFs, esse artigo utiliza a decomposição em nós de árvores para representar o shading parcial de objetos.

2.1.5.4 A Real-Time Configurable Shader Based on Lookup Tables

Este trabalho propõe uma arquitetura de hardware que permite cálculos de shading por pixel em tempo real, utilizando lookup-tables. Para isso, são projetados circuitos configurável baseado em lookup-tables, memórias de acesso aleatório (RAMs) e memórias somente leitura (ROMs). Varios circuitos base foram projetados visando realizar cálculos de shading, considerando as operações mais comuns, por exemplo, circuitos para calcular produto interno entre dois vetores, circuitos de rotação de um vetor por um angulo. Ademais, é usado interpolação em um sistema de coordenadas polares em vez da interpolação vetorial convencional com normalização, com o objetivo de reduzir o tamanho dos circuitos e melhorar o desempenho.

Figura 2 – Exemplo de circuito de produto interno entre vetores



Fonte: **??**, p. 24)

Além disso, o circuito suporta diversas BRDFs, como Blinn-Phong, Cook-Torrance, Ward e modelos baseados em microfacetes, com tabelas de lookup específicas para cada modelo. O uso de tabelas de pesquisa permite a representação organizada da parametrização das BRDFs, tornando o processo de transformação de BRDF para shaders mais acessível. Assim como este trabalho, a abordagem facilita a geração de shaders a partir da descrição de BRDFs, apesar da metologia ser diferente.

2.2 Pesquisa por Repositórios online

Também foram analisados repositórios no github e SourceForge, cada um com uma string de busca especifica. Os repositórios encontrados foram filtrados baseados em seus resumos, caso não haja a menção da criação de um compilador, ou não citar uma transformação de BRDF para outra estrutura, esse repositório será excluido.

Plataformas	**Termos de Pesquisa**	**Resultados**
GitHub	in:readme (GLSL AND BRDF AND (compiler OR compilation) AND (shader OR shading))	15
SourceForge	compiler bdrf	0

Tabela 3:

Após ler por completo os resumos dos repostiórios do GitHub, é evidente que nenhum desses projetos é relacionado com o proposto neste trabalhos, apesar de comentar sobre BRDFs, esses projetos não implementam compiladores, não fazem parsing de equações de BRDFs e nem mesmo geram shaders a partir de BRDFs.

3
Conceitos

3.1 BRDFs

<https://www.youtube.com/watch?v=kPIqO929pIc&list=PL2zRqk16wsdpyQNZ6WFlGQtDICpzzQ9
index=3>

3.2 Compilador

3.3 Análise Léxica

3.4 Análise Sintática

3.4.1 Pratt Parsing

3.4.1.1 precedencia de expressões

3.5 Shaders

Referências