1
Introdução

A procura por métodos de visualização eficientes e robustas de variedades implícitas vem sendo objetivo de estudo de muitos autores (Zhou, 1991; Hoffmann e Zhou, 1991; Paiva et al., 2006; Bordignon et al., 2013; Chen et al., 2014).

As variedades implícitas são representações versáteis de objetos geométricos, e são comumente aplicadas em simulação (Raveendran et al., 2014). Em Computação Gráfica e em Modelagem Geométrica, um importante tema é a visualização exata de variedades implícitas (Paiva et al., 2006).

Em particular, o estudo de variedades implícitas mergulhadas em espaços de dimensão maior do que 3 tem aplicações em geração de rotas para robos, detecção de colisões, entre outras (Zhou, 1991).

Na literatura, há muitos trabalhos relacionados à visualização de variedades de dimensão 2 no $\mathbb{R}^3$, com diferentes enfoques (Paiva et al., 2006). Entretanto, para a visualização interativa de variedades de dimensão 3 no $\mathbb{R}^4$ há muito a ser desenvolvido.

1.1
Objetivos

O principal objetivo deste trabalho é apresentar um novo método de visualização de variedades implícitas de dimensão 3 mergulhadas no $\mathbb{R}^4$. Este método será dependente do observador e se baseia na técnica de *Ray Casting*. Para se obter taxas interativas de renderização, a programação será feita em GPU com OpenGL 4.

1.2
Trabalhos relacionados

O interesse na visualização de objetos matemáticos no $\mathbb{R}^4$ é muito antigo, com trabalhos datando do inicio do seculo XX, como os de (Manning, 1914) e de (Forsyth, 1930). Já nas décadas de 70/80, o livro de (Coxeter, 1973) desenvolveu bases para a geometria no hiperespaço e (Carey S.A.; Burton e

Campbell, 1987) é outro contribuidor às técnicas de computação gráfica para renderização no espaço de dimensões superiores.

Uma abordagem para a visualização de superfície implícita no $\mathbb{R}^4$ é fazer uma aproximação poligonal da superfície implícita, como é feito em (Castelo et al., 2006). Os problemas deste tipo de abordagem, estão na dificuldade de encontrar pontos no objeto e representá-los em uma malha. Outra abordagem é usar *ray tracing* (Hollasch, 1991). Nesse trabalho ele define uma câmera no $\mathbb{R}^4$, gera um cubo de voxels onde são lançados raios desde a câmera para cada voxel, e calcula iluminação, reflexão e refração. Logo, esse cubo é visualizado como se fosse uma câmera em $\mathbb{R}^4$. Todo o processamento é feito em CPU. Entre os trabalhos de *ray tracing*, que consideram otimizações de métodos e algoritmos para conseguir uma visualização interativa de variedades implícicas no $\mathbb{R}^3$ destacamos (Knoll, 2009). Uma terceira abordagem é a renderização baseada em pontos. (Witkin e Heckbert, 1994) renderizam superfícies implícitas colocando pequenos discos opacos nos pontos amostrados na superfície implícita. Outros trabalhos, como de (Balsys e Suffern, 2007) apresentam o algoritmo de visualização baseado em pontos, usando hipercubos para realizar a subdivisão em um volume $4D$. A desvantagem desse trabalho está em requerer uma subdivisão bem fina para ter uma maior quantidade de pontos para cobrir a superfície.

Na tese de (Knoll, 2009) usa-se Aritmética Intervalar e Aritmética Afim para acelerar o cálculo de interseção em *ray Tracing*, mas esse trabalho está feito para superfícies no $\mathbb{R}^3$. (Knoll et al., 2009) descreve a implementação da Aritméticas Intervalar e Afim em GPU, para renderizar superfícies implícitas no $\mathbb{R}^3$. (Sigg, 2006) descreve diferentes enfoques para a representação de superfícies implícitas e métodos de renderização de superfícies dessas representações com uma implementação altamente paralelizada. Esse trabalho somente aborda superfícies no $\mathbb{R}^3$.

1.3 Organização

Este trabalho está organizado da seguinte maneira: o Capítulo 2 apresenta os fundamentos teóricos necessários para entender o método proposto; o Capítulo 3 descreve as duas etapas do método, a primeira etapa que é o pré-processamento em CPU com o uso da Aritmética Intervalar; e a segunda etapa do método, que corresponde ao processamento em GPU; o Capítulo 4 apresenta os resultados obtidos; e finalmente, o Capítulo 5 conclui o trabalho descrevendo as conclusões e propondo trabalhos futuros.