Universidade Federal de Santa Catarina
Programa de Pós-Graduação em
Ciência da Computação

Metodologia Computacional para a Simulação Interativa de Fluidos em Estruturas Tubulares

Tiago de Holanda Cunha Nobrega

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Orientador
Dr. rer. nat. Aldo von Wangenheim

Florianópolis, maio de 2010

Catalogação na fonte pela Biblioteca Universitária da
Universidade Federal de Santa Catarina

N745m Nobrega, Tiago de Holanda Cunha
Metodologia computacional para a simulação interativa de fluidos em estruturas tubulares [dissertação] / Tiago de Holanda Cunha Nobrega ; orientador, Aldo von Wangenheim. – Florianópolis, SC, 2010.
143 p.: il., grafs., tabs.

Dissertação (mestrado) – Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Ciência da Computação.

Inclui referências

1. Informática. 2. Ciência da computação. 3. Sistema de Partículas. 4. Simulação (Computadores). 5. Hidrodinamica. I. Wangenheim, Aldo v. (Aldo von). II.Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.

CDU 681

Metodologia Computacional para a Simulação Interativa de Fluidos em Estruturas Tubulares

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação, área de concentração de Sistemas de Conhecimento, e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Coordenador **Orientador**

Dr. Mario Antonio Ribeiro Dantas
Universidade Federal de Santa Catarina

Dr. rer. nat. Aldo von Wangenheim
Universidade Federal de Santa Catarina

Banca examinadora

Dr. rer. nat. Eros Comunello
Universidade Federal de Santa Catarina

Dr. Luiz Felipe de Souza Nobre
Universidade Federal de Santa Catarina

Prof. Joceli Mayer, Ph.D.
Universidade Federal de Santa Catarina

Prof. Mario Costa Sousa, Ph.D.
University of Calgary

Resumo

Este trabalho aborda a simulação de fluidos através de estruturas tubulares arbitrárias. O fluido é simulado através do método baseado em partículas denominado Smoothed Particle Hydrodynamics, enquanto que a estrutura tubular pode ser representada por uma malha de triângulos simples. Este cenário de simulação é utilizado tanto em ferramentas de apoio ao especialista como programas de CAD e simulações cirúrgicas quanto em jogos e na indústria de efeitos especiais, onde o efeito visual é mais importante do que a física. Existe um equilíbrio entre a fidelidade da simulação ao fenômeno físico correspondente e a resposta visual e interatividade com o usuário. O modelo apresentado neste trabalho visa oferecer um ambiente que permita simulações interativas visualmente atraentes em um cenário potencialmente complexo, onde o líquido e a estrutura são representados por milhares de entidades. A interação entre a estrutura e o fluido é simplificada em uma força que utiliza o Eixo Médio Aproximado para guiar as partículas, que são coloridas de acordo com grandezas físicas como velocidade e pressão.

Palavras-chave: Sistemas de partículas, simulação física, Smoothed Particle Hydrodynamics, SPH.

Abstract

This work studies the simulation of fluids through arbitrary tubular structures. The fluid is simulated through the particle-based method Smoothed Particle Hydrodynamics, while the tubular structure can be represented as an ordinary triangle mesh. This simulation scenario is used both in specialist-support tools such as CAD and surgery simulation software and in games and the special effects industry, where the visual effect is more important than the physics. There is a balance between the simulation's fidelity to the corresponding physical phenomenon and the visual response and interactivity with the user. The model presented in this work aims to offer an environment which allows visually appealing interactive simulations in a potentially complex scenario, where the fluid and the structure are represented by thousands of elements. The interaction between the structure and the fluid is simplified in a force that utilizes the Approximate Medial Axis to guide the particles, which are colored according to physical quantities such as velocity and pressure.

Keywords: Particle Systems, Physical Simulation, Smoothed Particle Hydrodynamics, SPH.

À minha família e amigos.
Essa é pra vocês.

"Since then it’s been a book you read in reverse,
So you understand less as the pages turn."

(The Shins, Pink Bullets)

Agradecimentos

À minha família,
que me apóia incondicionalmente,

ao Prof. Aldo von Wangenheim,
pela estrutura propícia à pesquisa,

ao chefe Diego Dias Bispo Carvalho,
pelas várias idéias e sugestões,

ao colega Adiel Mittmann,
fonte profunda de conhecimento e frustrações,

sinceramente agradeço.

Sumário

Lista de Figuras

1 Introdução

Uma simulação é uma imitação de alguma atividade através de simplificações e restrições que acarretam menor custo, tempo e risco. A possibilidade de executar a simulação de uma cirurgia através do uso da ferramenta computacional é atraente pelo aspecto não invasivo e pela liberdade que uma simulação permite ao especialista.

O problema do elevado custo em tempo de processamento que uma simulação cirúrgica de médio porte impõe ao computador vai sendo mitigado conforme a tecnologia de processadores avança. Desta forma, é possível executar hoje uma simulação que há dez anos não era factível. Isto é verdade em todas as áreas de software, como jogos, aplicações de bases de dados e gráficas diversas, etc (BERGEN, 2003).

Conforme a capacidade dos computadores sobe, a expectativa de seus usuários aumenta. Os programadores de aplicações continuam incumbidos de manter a eficiência dos algoritmos e estruturas de dados de seus programas em mente para que seus usuários tenham uma experiência fluida e realista (BERGEN, 2003; COHEN et al., 1995).

Uma simulação interativa do comportamento de fluidos interagindo com estruturas sólidas com formas arbitrárias, que é o foco deste trabalho, é composta por uma série de componentes distintos.

O comportamento do fluido em si deve ser simulado realisticamente.

Talvez este seja o componente mais visualmente evidente — as equações que regem o estado e a evolução do fluido, originárias da Mecânica dos Fluidos, devem ser descritas de forma algorítmica e eficiente.

A interação do fluido com as entidades do mundo, neste caso estruturas sólidas, deve ser modelada e implementada de acordo com a necessidade da aplicação. Métodos variam entre aqueles que incluem as estruturas nas equações de estado do fluido, gerando resultados fisicamente realistas, até aqueles que simplificam as entidades em modelos geométricos básicos, como esferas e triângulos, e tratam a interação como um problema de interferência.

Por ser interativa, o sistema deve permitir alguma resposta visual ao usuário a cada momento. Em um sistema onde o objetivo seja a renderização tridimensional realista da simulação, a visualização pode se tornar um dos gargalos do processo.

Uma seqüência de passos de simulação emerge naturalmente desta lista. Primeiramente, o estado do fluido é considerado e evoluído através da abstração utilizada das interações intermoleculares características. Em seguida, o novo estado do fluido é comparado com o ambiente de forma a corrigir impossibilidades como penetrações e escoamentos. Em sua forma mais simples este caso envolve apenas a *detecção* e o *tratamento* das colisões entre o fluido e o ambiente. Finalmente, o novo estado do sistema todo é apresentado ao usuário pela forma apropriada.

O intuito deste trabalho é precisamente avaliar e modelar de forma eficiente o segundo passo da simulação, que é a interação entre o fluido e as estruturas do ambiente. O seguinte cenário ajudará a ilustrar a importância deste componente inserido em uma simulação cirúrgica hipotética:

Uma seção do abdômen de um paciente foi reconstruída através de uma série de imagens de tomografia computadorizada sob suspeita da formação de um aneurisma na artéria aorta. A simulação resultante consiste na artéria e

órgãos adjacentes, a coluna vertebral e algumas costelas flutuantes. Simula-se também o fluxo sanguíneo através da aorta.

A intenção do usuário especialista é avaliar a presença e a condição do aneurisma e, se necessário, calcular as dimensões e o posicionamento ótimos de uma endoprótese na região afetada pela formação.

Uma simulação deste tipo requer uma rígida fidelidade à anatomia do paciente e ao realismo das interações entre as diferentes estruturas representadas. O líquido sanguíneo, ao fluir pela artéria aorta, deve exercer pressão sobre a parede arterial no sentido de expandi-la. Fluidos são muitas vezes modelados computacionalmente como milhares de pequenas partículas esféricas submetidas às leis da dinâmica dos fluidos (STAM, 1999; MONAGHAN, 1992; MULLER; CHARYPAR; GROSS, 2003), enquanto que a aorta em si pode ser composta de também milhares de pequenos triângulos resultantes da reconstrução a partir da série de imagens (LORENSEN; CLINE, 1987).

O resultado deste cenário — a identificação do momento em que uma partícula de sangue toca um triângulo da artéria (a *detecção* da colisão), e a obtenção da expansão da parede arterial (o *tratamento* da colisão detectada) — deve ser feito a todo instante e para todas as entidades envolvidas. Mesmo para um grupo relativamente pequeno de objetos, o custo computacional pode se elevar exponencialmente. Mais custo computacional significa mais tempo do processador executando esta tarefa, o que significa menos atualizações por segundo e a potencial perda da interatividade e noção do realismo.

Adicionalmente, o especialista deseja *interagir* com a simulação, já que o objetivo da simulação é posicionar uma prótese na artéria para conter a ruptura do aneurisma. Nesta aplicação, o posicionamento da prótese é efetuado através do mouse. Conforme o especialista movimenta o objeto, espera-se que este também interaja com os órgãos e fluidos da simulação e que a momento nenhum o usuário perca a noção de realismo e interatividade.

Requisitos como estes podem transformar o componente gerenciador das colisões no gargalo de performance da aplicação. É essencial que este componente aja da maneira mais inteligente possível, para que exista espaço (na fatia de tempo de um quadro da simulação) suficiente para os outros componentes da aplicação.

Esta dissertação está dividida em quatro grandes partes. A primeira discute estruturas sólidas rígidas representadas por malhas de triângulos e métodos de detecção de colisão. O método escolhido utiliza uma árvore hierárquica onde cada nodo encapsula uma entidade geométrica com um teste de intersecção eficiente (HUBBARD, 1995; GOTTSCHALK; LIN; MANOCHA, 1996; BERGEN, 2003). Três tipos de objetos diferentes são comparados e caixas alinhadas aos eixos (*AABBs* — do inglês *Axis-Aligned Bounding Boxes*) são escolhidas (Capítulo 2).

A segunda parte do trabalho discorre sobre os métodos existentes para a simulação computacional de fluidos (Capítulo 3). O método utilizado, denominado *Smoothed Particle Hidrodynamics*, inicialmente concebido por Gingold e Monaghan (1977), modela uma porção de fluido como um conjunto de *partículas*, e representa a evolução do fluido pela interação entre as partículas utilizando funções de interpolação. O método é interessante por apresentar conservação de massa no fluido, ser altamente paralelizável e permitir a fácil introdução de forças externas.

No caso da interação entre uma estrutura tubular e fluidos, esta força externa pode ser a influência que a estrutura possui sobre o fluido. Em vasos sanguíneos, a expansão e contração das paredes em função da pressão sanguínea gera uma interação complexa pela presença e necessidade de interação de dois sistemas deformáveis — as paredes arteriais e o sangue (MULLER; SCHIRM; TESCHNER, 2004; BLANCO; FEIJóO; URQUIZA, 2007; BLANCO et al., 2009). Como o interesse deste trabalho é a manutenção da interatividade, são consideradas apenas estruturas sólidas rígidas. Para adi-

cionar um componente de influência da estrutura sobre o fluido na ausência de fenômenos como pulsação, uma estratégia que utiliza uma aproximação do eixo médio da estrutura é apresentada na terceira parte do trabalho, no Capítulo 4.

A última parte do trabalho trata da visualização da simulação, demonstrada através da implementação de um protótipo que une todos os componentes discutidos com avaliação de sua performance. Adicionalmente, discute-se as limitações adotadas neste trabalho, uma vez que os tópicos são extensos demais para serem abordados na íntegra, e os trabalhos futuros e conclusões assumidas. São os Capítulos 5 e 6.

1.1 Objetivos

1.1.1 Objetivo Geral

Elaborar uma metodologia computacional para a simulação de fluidos e estruturas sólidas tubulares com formas arbitrárias. O foco será em aplicações de visualização tridimensional de forma que a interação entre o fluido e a estrutura será simplificada para permitir taxas interativas de visualização.

1.1.2 Objetivos Específicos

Elaborar uma metodologia para a detecção de colisões entre estruturas sólidas e fluidos

O tópico de detecção de colisões é bem estudado na literatura e este objetivo visa avaliar e escolher a técnica apropriada para efetuar o passo de detecção de colisões na interação entre estruturas sólidas e fluidos.

Simular de forma eficiente a complexa interação entre estruturas tubulares e fluidos nelas contidos

Esta interação é dita complexa pois, na natureza, são vários os fenômenos que regem o comportamento de um fluido interagindo com paredes sólidas. Exemplos incluem a viscosidade que incentiva o fluido à aderir à parede, fenômenos de expansão e contração para sólidos que não são perfeitamente rígidos, entre outros.

Possibilitar a visualização da evolução de grandezas físicas no fluido ao longo da simulação

O objetivo de uma simulação é a avaliação de algum aspecto crítico à decisão. É interessante possibilitar de forma não apenas quantitativa mas também visual a extração de grandezas físicas sobre o fluido ao longo da simulação, como o aumento da pressão local em um trecho em que a estrutura tubular se estreite.

Considerar decisões de projeto que influenciem na performance, para possibilitar o realismo e a interatividade

Segundo Haines e Akenine-Moller (2002), o mínimo necessário em uma simulação para manter a noção de realismo é de 6 quadros por segundo e 15 quadros para haver a noção de ação e reação com o usuário; Robb, Hanson e Camp (1996) restringem ainda mais este valor para 30 quadros por segundo para simulações médicas. Ainda que o objetivo deste trabalho não seja o desenvolvimento de um software completo para simulações cirúrgicas, é interessante que os produtos aqui gerados (por exemplo, na forma de bibliotecas) não proíbam o realismo e a interatividade de uma eventual aplicação pela ineficiência de suas rotinas.

Validar o sistema obtido através de uma simulação-protótipo com dados reais

Este trabalho não visa validar formal e extensivamente o sistema criado, mas um protótipo que permita uma simulação simples porém realista deve ser implementado.

2 *Detecção de Colisão em Malhas de Triângulos*

O intuito deste trabalho é permitir a simulação realística e interativa de fluidos em estruturas tubulares. Exemplos destas estruturas incluem vasos sanguíneos, encanamentos, corredores, etc. Neste contexto, a estrutura é considerada tubular se for possível distinguir-se uma dimensão consideravelmente maior do que as outras, como o comprimento maior do que a altura e largura. Estas estruturas podem ser modeladas de duas formas gerais.

2.1 Modelagem de Estruturas Tubulares

Na modelagem volumétrica, as estruturas são discretizadas diretamente sobre os *voxels* oriundos do meio de aquisição, como séries de imagens de tomografia computadorizada, ressonância magnética, etc. A maior vantagem deste tipo de modelagem é que a estrutura a ser manipulada está na melhor resolução possível, limitada apenas pelo meio de aquisição. Por este motivo, a modelagem volumétrica é vastamente utilizada em aplicações médicas para visualização e manipulação de estruturas anatômicas (LEVOY, 1988; WESTOVER, 1989; LACROUTE; LEVOY, 1994; VIDAL et al., 2006; KALVIN; LAINE; SONG, 2008).

Em contrapartida, na modelagem por malhas, uma lista de triângulos (ou outro tipo de polígono, mas comumente triângulos) é obtida a partir dos dados volumétricos. A vantagem imediata é o menor consumo de memória na maior parte dos casos visto que os dados originais geralmente podem ser descartados. Assim como métodos volumétricos, as malhas de triângulos possuem um grande histórico de pesquisa na literatura, o que é evidenciado pelos vários tipos diferentes de estruturas de dados existentes para representá-las. Exemplos incluem *Half-Edges* (WEILER, 1985), *Winged-Edges* (BAUMGART, 1975), *Doubly-Linked Face Lists* (AKLEMAN; CHEN, 1999), e *Triangle Soups* (SHIN et al., 2004) convencionais que apenas tratam a malha como uma lista de triângulos sem informação adicional.

Geração de Malhas de Triângulos

Os triângulos que formam a estrutura podem ser modelados diretamente através do uso de ferramentas de CAD como o *Blender* e o *3D Studio Max*, ou podem ser "extraídas" dos dados volumétricos através de métodos específicos. Um destes métodos é utilizado neste trabalho para a obtenção de malhas representativas de seções de artéria — é o método *Marching Cubes*.

O algoritmo *Marching Cubes* foi originalmente concebido por Lorensen e Cline (1987) especialmente para a geração de superfícies anatômicas a partir de dados médicos. Os dados de entrada do algoritmo são as imagens de tomografia ou ressonância e um valor de *threshold* definido pelo usuário. O algoritmo inicia considerando um cubo formado por oito *pixels* adjacentes em duas imagens consecutivas (Figura 2.1) e tomando o valor de cor nestes vértices.

A superfície a ser obtida intersecta o cubo se um ou mais destes *pixels* possuir o valor de cor dentro do *threshold.* A posição do triângulo da superfície dentro do cubo é interpolada pelo valor da cor nos *pixels* e o cubo "marcha" para o próximo octeto de *pixels*. O significado de um *pixel* varia de

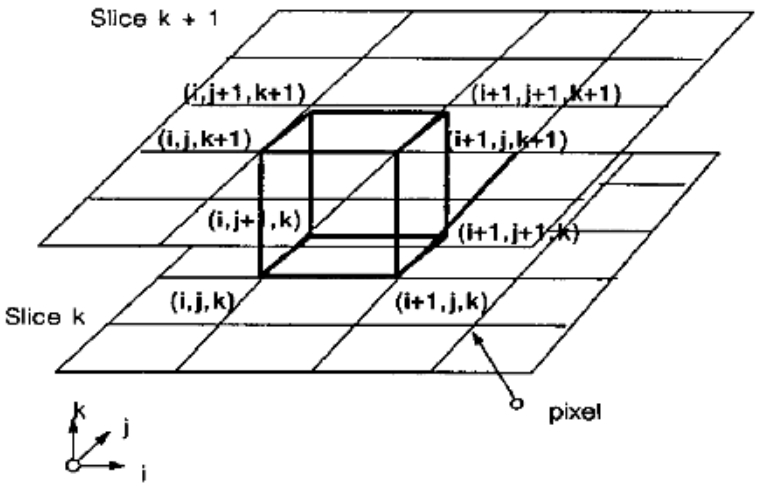


Figura 2.1: Cubo formado por oito *pixels* adjacentes em duas imagens consecutivas (LORENSEN; CLINE, 1987).

acordo com o tipo da imagem.

No caso de imagens médicas neste trabalho trabalhou-se primariamente com imagem de Tomografia Computadorizada (CT). No CT, o equipamento (tomógrafo) expõe o corpo a uma sucessão de raios X. Os diferentes tecidos anatômicos absorvem diferentes quantidades da radiação X — tecidos mais densos (como o fígado) ou compostos por elementos mais pesados (como os ossos, ricos em cálcio) absorvem mais radiação do que os tecidos menos densos (como por exemplo o pulmão, cheio de ar). Cada pixel da imagem resultante corresponde à média da absorção dos tecidos na região, expressa em unidades de *Hounsfield*. Definindo-se um *threshold* baseado nestas medidas, o resultado é a geração da superfície limítrofe da estrutura anatômica de interesse. Diferentes tecidos podem ser reconstruídos simplesmente alterando-se o valor deste *threshold*.

Uma das variáveis que regulam a qualidade da superfície obtida é o tamanho do cubo utilizado. O valor padrão é um cubo com 1 *pixel* de lado, mas

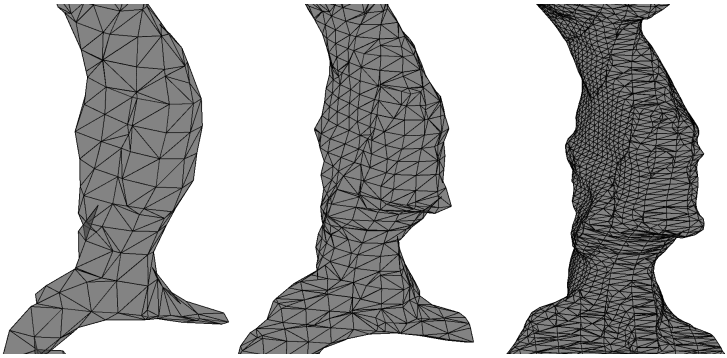


Figura 2.2: Segmento arterial reconstruído com três níveis de qualidade.

é possível utilizar outros valores para gerar superfícies de menor qualidade mas que possuam menos triângulos. Dependendo da aplicação, estas superfícies de resolução mais baixa podem ser abstrações suficientemente boas da estrutura anatômica em questão. De fato, esta é uma vantagem e uma desvantagem do método — é possível extrapolar estruturas de baixa resolução utilizando cubos maiores, mas mesmo a utilização de um cubo na resolução de um *pixel* pode gerar malhas que representem mais de uma estrutura. Isso acontece quando a distância entre uma estrutura e outra é pequena demais, tanto em número de pixels quanto em diferença de tom de cinza. Nestes casos, convém realizar uma etapa prévia de segmentação nas imagens.

A Figura 2.2 ilustra uma superfície representativa da artéria reconstruída com cubos com lado de 3.0, 1.5 e 0.6 milímetros. As reconstruções levam uma média de 1, 4 e 19 segundos, respectivamente, uma vez que cubos menores levam mais tempo para percorrer o espaço todo e tendem a gerar mais triângulos.

2.2 Colisões em Malhas de Triângulos

Segundo Ericson (2004), a detecção de colisão concerne o problema de "determinar se, quando e onde dois objetos entram em contato". Cada pergunta assume um nível incremental de complexidade: responder "se" dois objetos estão em contato é um valor booleano com apenas dois resultados possíveis, enquanto que responder "quando" eles entraram em contato inicialmente representa um valor distinto em um período contínuo de tempo. Similarmente, a resposta de "onde" eles estão em contato pode possuir uma resposta complexa referente à área de intersecção entre os dois objetos (ERICSON, 2004).

Dada uma cena composta de múltiplos objetos de composição arbitrária, a detecção de colisão pode ser separada em duas grandes fases distintas, a *broad phase* e a *narrow phase*.

A *broad phase*, ou *fase abrangente*, visa descartar rapidamente pares de objetos que absolutamente não podem estar colidindo. O objetivo é agrupar objetos que se encontrem "razoavelmente" próximos para posteriormente verificar se eles estão de fato se interceptando. Métodos para a *broad phase* incluem o uso de particionamento espacial com grids uniformes, árvores hierárquicas para o espaço de simulação como *Octrees* e *KD-Trees*, e métodos que exploram a localidade espacial dos objetos como o *Sweep and Prune*. Como a *broad phase* só faz sentido para um número razoável de objetos e este trabalho concerne apenas a interação entre uma estrutura tubular e uma porção de fluido, aqui a *broad phase* é omitida.

Na *narrow phase*, ou *fase restrita*, cada par de objetos que não foi descartado na etapa anterior é completamente testado para a colisão. Em se tratando de colisões entre objetos representados por malhas de triângulos, o jeito mais direto de identificar a colisão consiste em simplesmente testar cada triângulo do primeiro objeto com cada triângulo do segundo. Tal estratégia se torna vi-

sivelmente impraticável com apenas algumas poucas centenas de triângulos em cada objeto.

2.2.1 *Bounding Trees*

Uma otimização simples para a fase restrita é envolver cada objeto dentro de uma entidade geométrica mais simples e com um rápido teste de intersecção. Estas entidades são conhecidas como *Bounding Volumes* e são tidas como condição necessária, porém não suficiente, para colisão. Desta forma, um par de objetos cujos *Bounding Volumes* não intersectem não podem absolutamente estar intersectando.

A escolha da entidade geométrica é um fator dependente da aplicação. Opções incluem esferas (HUBBARD, 1995), caixas alinhadas aos eixos (BERGEN, 2003), caixas orientadas (GOTTSCHALK; LIN; MANOCHA, 1996), cascos convexos, etc. Os fatores mais desejáveis de uma entidade com este fim incluem (ERICSON, 2004):

- Testes de intersecção rápidos
- Bom caimento no objeto
- Cálculo rápido de criação
- Facilidade em transformações, como rotações e translações
- Consumo baixo de memória

Há um equilíbrio entre os fatores de forma que não há entidade "ótima". Por exemplo, de forma geral quanto melhor o caimento da entidade ao objeto, mais demorados são os testes de intersecção. A Figura 2.3 explicita a relação entre os fatores em diversas entidades diferentes.

Caso o teste de intersecção dos *Bounding Volumes* que encapsulam dois objetos sinalize positivo, o problema de testar duas listas grandes de triângu-

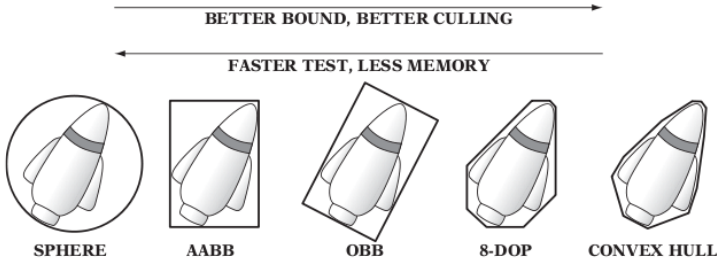


Figura 2.3: Da esquerda para a direita: Esferas, caixas alinhadas, caixas orientadas, *8-dop*, e casco convexo (ERICSON, 2004).

los retorna. A idéia de encapsular a geometria em uma entidade geométrica de teste simples é estendida agora: separa-se a geometria do objeto em dois ou mais grupos, e encapsula-se cada grupo em uma nova entidade. O processo repete-se de forma recursiva de maneira que o resultado final é uma árvore de *Bounding Volumes*, uma *Bounding Volume Hierarchy* ou *Bounding Tree.* A árvore é construída de forma que cada nodo folha contenha um triângulo para que o teste entre duas estruturas geométricas encapsuladas por *Bounding Trees* possa ser feito em tempo médio logarítmico.

Para este trabalho, considerou-se a utilização de três tipos de *Bounding Volumes* diferentes: Esferas, Caixas Alinhadas, e Caixas Orientadas. A árvore de caixas alinhadas, conhecida na literatura por *AABBTree*, foi escolhida pela simplicidade de implementação e a presença de código já existente no grupo.

2.3 Aplicação: Geração do Eixo Médio Aproximado de Estruturas Tubulares

Como será discutido em maiores detalhes no capítulo 4, a detecção de colisão entre uma partícula e uma malha triangular neste trabalho envolve uma grande quantidade de testes de intersecção entre segmentos de reta e *Bounding Trees* e triângulos. O primeiro passo é identificar qual dentre os três

Bounding Volumes estudados é o melhor para este tipo de teste. Esta avaliação foi realizada sobre um problema prático em um protótipo já existente que realiza a extração do *Eixo Médio Aproximado* de estruturas tubulares.

2.3.1 Eixo Médio de Estruturas

O Eixo Médio (em inglês *Centerline*) é uma entidade geométrica que foi inicialmente introduzida por Blum (1967) e é também conhecida como Eixo Simétrico (*Symmetrical Axis* (WAN et al., 2002)). Jiang e Gu (2005) provêem uma definição intuitiva de um Eixo Médio como "uma curva que atravessa o centro de um órgão oco".

Os Eixos Médios possuem muitas aplicações: Na medicina computacional, a extração do Eixo Médio é um passo principal nos processos de colonoscopia (WAN et al., 2002; JIANG; GU, 2005; WILLIAMS et al., 2008) e broncoscopia (LAW; HENG, 2000) virtuais e na prototipação de próteses endovasculares (BIASI et al., 2002; SUBRAMANYAN; CIANCIBELLO; DURGAN, 2004). O problema de encontrar um caminho ótimo de probabilidade de colisão mínima para aplicações de engenharia e arquitetura virtuais também envolve Eixos Médios (BITTER et al., 2000).

A maioria dos algoritmos de extração de Eixos Médios assumem que os dados de entrada estão na forma de *Grids* de *voxels*, como séries de tomografia computadorizada e ressonância magnética. Em seu *survey*, Kirbas e Quek (2004) citam que métodos para a extração do Eixo Médio de vasos sanguíneos incluem a segmentação de áreas de interesse com posterior definição de conectividade, segmentação com afunilamento (*thinning*) e extração baseada em descrição de grafo.

Em contrapartida, Pivello, Larrabide e Feijóo (2007) utilizam um método baseado em malhas que explora a conectividade dos elementos para executar a travessia da estrutura de forma eficiente, gerando seções trans-

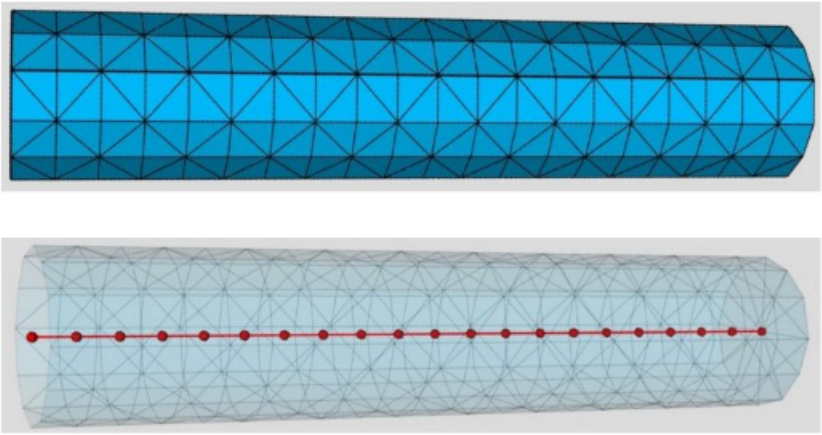


Figura 2.4: Geração do Eixo Médio através de seções tranversais (PIVELLO; LARRABIDE; FEIJóO, 2007).

versais da estrutura e conectando os centróides destas secções para gerar o Eixo Médio final (Figura 2.4).

2.3.2 Geração do Eixo Médio Aproximado

O método utilizado neste trabalho gera um aproximação do Eixo Médio a partir de uma malha de triângulos. O método foi desenvolvido por Felipe Borges Alves, um ex-colaborador do projeto Cyclops, com o intuito de permitir a navegação e obtenção de dados como diâmetro de seções transversais e o comprimento aproximado de regiões de interesse em estruturas tubulares (Figura 2.5).

Uma região de interesse é definida pelo posicionamento manual de dois planos de corte que definirão as extremidades da região. A partir destes planos e da malha de triângulos são adotados os passos descritos nos parágrafos seguintes.

As intersecções entre os planos e a malha são encontradas. A intersec-

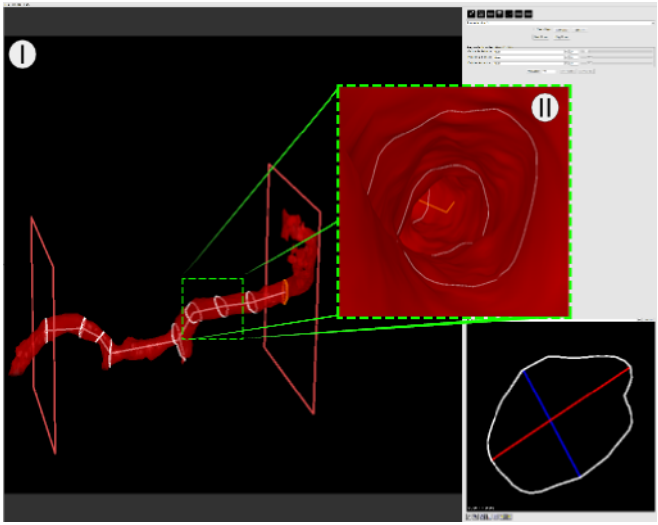


Figura 2.5: O Eixo Médio Aproximado permite a navegação através da estrutura e a medição de seções (SILVA et al., 2009).

ção entre um plano arbitrário e uma malha de triângulos resulta em um ou mais polígonos planares (CARVALHO; SANTOS; WANGENHEIM, 2006) que aqui são chamados de Seções Tranversais, mesmo que o plano não seja perfeitamente perpendicular à estrutura na região de intersecção.

Um segmento de reta é traçado entre os centróides dos dois polígonos encontrados. Se esta linha intersecta a malha, um novo plano é posicionado no ponto médio do segmento de reta utilizando o vetor de direção do segmento como vetor normal (Figura 2.6).

Para cada novo plano posicionado desta forma, sua interseção com a malha é calculada e novos segmentos de reta são gerados entre a nova seção tranversal e as seções anterior e próxima. O processo continua recursivamente até que não sejam encontradas intersecções entre os segmentos de reta e a malha, ou algum outro critério de parada seja alcançado.

Se uma interseção plano-malha resulta em mais de um polígono, uma

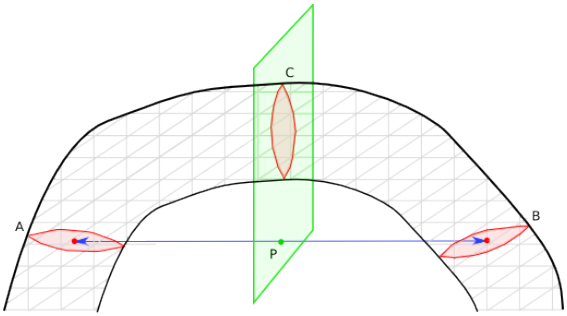
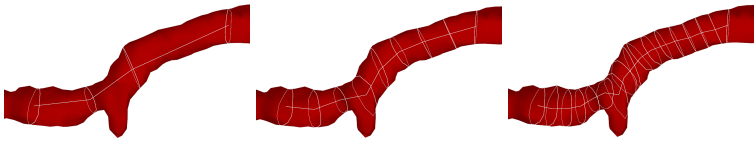


Figura 2.6: Um segmento de reta é traçado entre as seções *A* e *B*, gerando um plano utilizando o ponto médio *P* e o vetor de direção da reta *N*. A intersecção entre este plano e a estrutura gera a nova seção *C*.



(a) (b) (c)

Figura 2.7: Uso de três distâncias máximas distintas no refinamento do Eixo Médio Aproximado.

bifurcação na estrutura foi encontrada. O processo é repetido para cada intersecção da bifurcação, gerando uma árvore de seções tranversais.

Após o término do algoritmo, o Eixo Médio Aproximado gerado pode ser refinado para melhorar sua qualidade. Uma condição que foi identificada como útil é a adoção de um limite superior na distância entre duas seções transversais consecutivas, que aqui é definida como a distância entre os centróides das seções. A Figura 2.7 exemplifica esta condição.

Em 2.7a, nenhum limite máximo foi adotado. O Eixo Médio Aproximado utilizado é o gerado pelo algoritmo que parou quando nenhum segmento de reta gerado tocou a malha. O resultado é um Eixo Médio Aproximado que é composto por apenas algumas seções transversais e segmentos de

retas que em alguns pontos quase tocam a estrutura.

Em contrapartida, nas figuras 2.7b e 2.7c limites progressivamente menores na distância máxima entre duas seções transverais são adotados. Em ambos os casos, se a distância entre duas seções for *maior* do que o limite adotado, um novo plano é posicionado entre as seções usando novamente o vetor de direção do segmento de reta que liga as duas seções como vetor normal. O resultado é uma aproximação que é progressivamente mais representativa da forma real da estrutura, com o custo adicional de um maior número de seções e maior tempo de processamento.

O Eixo Médio Aproximado é composto por um conjunto de seções transversais e um conjunto de segmentos de retas que ligam estas seções, e será utilizado para efetuar a interação entre a estrutura tubular e o fluido simulado no capítulo 4.

2.3.3 Testes de Intersecção na Geração do Eixo Médio Aproximado

Como visto na Subseção 2.3.2, cada iteração do algoritmo de geração do Eixo Médio Aproximado requer testes entre segmentos de retas e planos e malhas de triângulos. Em uma malha de triângulos que não possui informação de conectividade, a única forma de efetuar estes testes é percorrer a malha e testar cada triângulo individualmente.

Para avaliar o custo desta abordagem força bruta, uma seção arterial foi reconstruída a partir de uma série de tomografia computadorizada utilizando o algoritmo *Marching Cubes* com três tamanhos de cubo diferentes (Seção 2.1). As malhas geradas possuem 1096, 18058 e 65556 triângulos. Uma vez que os planos e segmentos de reta que o algoritmo de geração cria tocam a malha na maioria dos casos, foram posicionados planos e segmentos que também o fazem.

Força Bruta	Número de Triângulos		
	1096	18058	65556
Melhor	1.65	16.25	61.26
Pior	1.83	16.83	63.41
Média	1.73	16.46	62.14
Desv.Pad.	0.06	0.24	0.89

Table 2.1: Tempos de teste para Força Bruta, com Planos (em segundos).

Força Bruta	Número de Triângulos		
	1096	18058	65556
Melhor	2.91	28.42	105.95
Pior	2.98	29.06	108.01
Média	2.95	28.84	107.16
Desv.Pad.	0.03	0.24	0.81

Table 2.2: Tempos de teste para Força Bruta, com Segmentos de Retas (em segundos).

O algoritmo geral de teste de intersecção entre um plano ou segmento de reta e uma malha de triângulos representada por uma *Bounding Tree* é descrito no Algoritmo 1. Para um dado plano ou segmento de reta *X*, o algoritmo testa a intersecção com a árvore começando pela raiz. Se o tipo de *Bounding Volume* que a árvore usa não intersecta *X*, então nenhum triângulo da malha que a árvore encapsula o faz, e o algoritmo termina. Caso contrário, se a árvore for uma folha, o triângulo encapsulado nesta folha é adicionado à lista de possíveis intersecções, e o algoritmo termina.

No terceiro caso, quando o *Bounding Volume* da árvore intersecta *X* mas a árvore não é uma folha, o processo se repete recursivamente utilizando cada filho da árvore como novo parâmetro de entrada. O que varia entre cada tipo de *Bounding Volume* utilizado é o algoritmo *IntersectaBV*, presente no Algoritmo 1 na linha marcada com *, que testa o tipo específico de *Bounding Volume* com o plano ou segmento de reta.

O algoritmo foi simplificado de forma a apenas retornar uma lista de

triângulos que *possivelmente* intersectam o plano ou segmento de reta, i.e., triângulos candidatos à intersecção. A razão para tal é que um processamento adicional pode ser necessário — no caso do plano, a seção transversal será gerada a partir dos segmentos de reta da intersecção do plano com cada triângulo. Estes detalhes foram excluídos da listagem do Algoritmo 1 para ilustrar o foco na árvore e nos *Bounding Volumes*, e porque este processamento adicional é comum também à abordagem em força bruta.

```
   Entrada: Bounding Tree Árvore, Plano ou Segmento de Reta X,
            Lista de Triângulos Lista
   Saída: Lista
   EncontraIntersecções(Árvore, X, Lista) :
*  se não IntersectaBV(Árvore.ObjetoGeométrico, X) então
   |  retorna;
   fim
   se ÉFolha(Árvore) então
   |  Insira(Lista, Árvore.Triângulo);
   |  retorna;
   fim
   EncontraIntersecções(Árvore.FilhoEsquerda, X, Lista);
   EncontraIntersecções(Árvore.FilhoDireita, X, Lista);
   retorna;
```

Algoritmo 1: Algoritmo geral de teste de intersecção entre uma *BoundingTree* e um objeto *X*.

O mesmo conjunto de planos e segmentos de reta utilizados no teste de força bruta foram aplicados na artéria envolta por uma *AABBTree*. As Tabelas 2.3 e 2.4 ilustram os tempos de medição obtidos e o ganho sobre a versão em força bruta. Na tabela é possível observar uma anomalia nos testes com segmentos: as intersecções com a artéria composta por 65556 triângulos são calculadas mais rápido do que as intersecções com a artéria composta por 18058 triângulos.

A razão para tal anomalia não foi completamente identificada, mas um motivo possível é que a intersecção entre um segmento de reta e uma ma-

lha de triângulos tende a gerar poucos pontos, pois o segmento ocupa muito pouco do espaço. Desta forma a *AABBTree* pode descartar segmentos após poucos testes de intersecção e descendo pouco na hierarquia. Assim, a disposição das caixas alinhadas que compõem os nodos das árvores deste conjunto em particular favoreceu um descarte rápido de várias seções da artéria. Resultados mais precisos necessitariam de mais conjuntos de dados e uma gama maior de segmentos de teste. Uma vez que a adoção de um tipo de árvore não é o foco deste trabalho e que os resultados da geração do Eixo Médio Aproximado descritos na Tabela 2.5 ilustram um ganho significativo, o assunto não foi estudado mais a fundo.

AABBTree	Número de Triângulos		
	1096	18058	65556
Melhor	0.14	0.62	2.08
Pior	0.73	3.03	6.07
Média	0.33	1.27	3.15
Desv.Pad.	0.21	0.91	1.51
Ganho	5.28	13.0	19.72

Table 2.3: Tempos de teste para *AABBTree*, com Planos (em segundos).

AABBTree	Número de Triângulos		
	1096	18058	65556
Melhor	0.08	0.11	0.09
Pior	0.48	0.54	0.49
Média	0.25	0.26	0.25
Desv.Pad.	0.15	0.17	0.17
Ganho	11.7	110.11	432.83

Table 2.4: Tempos de teste para *AABBTree*, com Segmentos de Retas (em segundos).

Eixo Médio Aproximado	Número de Triângulos		
	1096	18058	65556
Força Bruta	0.46	3.71	27.8
AABBTree	0.06	0.08	0.14
Ganho	7.67	46.38	198.57

Table 2.5: Tempos de teste para Força Bruta (em segundos).

3 Modelos para a Simulação de Fluidos

O objetivo deste capítulo é tanto apresentar brevemente os métodos existentes para a simulação computacional de fluidos quanto ilustrar o método utilizado. Tal exposição é necessária para que fique explícita a maneira como o modelo utilizado para a interação entre o fluido e a estrutura que o contém, que é discutido no capítulo 4, se porta dentro de uma iteração do processo de simulação de fluidos dentro de estruturas tubulares.

Em 1822, Claude Louis Marie Henri Navier derivou as equações que regem o movimento de um fluido viscoso. Ele partiu do trabalho de Euler de 1755, que já havia derivado as equações para fluidos ideais não viscosos. Após isso, em 1845 George Gabriel Stokes derivou as equações na maneira como elas são conhecidas hoje.

Fluidos incompressíveis são fluidos cuja densidade não varia. Líquidos são fluidos compressíveis, mas como a compressão só se dá sobre grande pressão eles são geralmente simplificados como incompressíveis (BRIDSON; FISCHER; GUENDELMAN, 2006).

A evolução do estado de um fluido incompressível no tempo é dada pelas seguintes especializações das equações de Claude Navier e George Stokes (CRAMER, 2002):

$$\frac{\partial \rho}{\partial t} + \nabla.(\rho \vec{v}) = 0, \tag{3.1}$$

e

$$\rho \left(\frac{\partial \vec{v}}{\partial t} + \vec{v}.\nabla \vec{v} \right) = -\nabla p + \vec{f}_{ext} + \mu \nabla^2 \vec{v}. \tag{3.2}$$

A equação 3.1 rege a conservação da massa no líquido, e a equação 3.2 dita a conservação do momento. Em ambas:

ρ é a densidade do líquido no ponto considerado;

p é a pressão do líquido no ponto considerado;

t é o tempo;

v é a velocidade do líquido no ponto considerado;

$\vec{f}_{ext}$ é o vetor das forças externas ao líquido, e.g. a gravidade;

μ é um coeficiente de viscosidade.

Neste trabalho adotam-se as seguintes convenções de notação:

$\frac{\partial f}{\partial x}$ é a derivada de f em relação a x ;

∇f é o *gradiente* de uma função f de n variáveis, e é definido como o vetor $\left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n} \right)$;

$\nabla.f$ é o operador vetorial de divergência de uma função f de n variáveis, e é definido como a soma dos componentes de ∇f;

$\nabla^2 f$ é o operador laplaciano escalar, definido como a divergência do gradiente de f , i.e. $\nabla.\nabla f$. No espaço cartesiano R^nele é definido como a soma das derivadas parciais de segunda ordem não mistas, ou seja, $\nabla^2 f = \frac{\partial^2 f}{\partial x_1^2} + \frac{\partial^2 f}{\partial x_2^2} + \ldots + \frac{\partial^2 f}{\partial x_n^2}$;

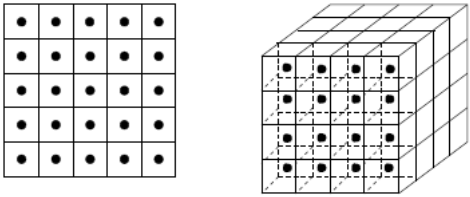


Figura 3.1: Um exemplo do grid estático de células que contém o fluido (STAM, 1999).

$\nabla^2\vec{v}$ é o operador laplaciano vetorial.

Os modelos matemáticos gerados pelas equações 3.1 e 3.2 geralmente não possuem solução analítica simples, de forma que o campo da Dinâmica dos Fluidos Computacional visa encontrar métodos numéricos para simular o comportamento de fluidos. Tais métodos podem ser divididos em dois tipos distintos (BRIDSON; FISCHER; GUENDELMAN, 2006): Métodos baseados em *Grids* (ou *Eulerianos*), e métodos baseados em Partículas (ou *Lagrangeanos*). Neste trabalho adota-se um método baseado em Partículas, de modo que os métodos baseados em *Grids* serão abordados apenas brevemente.

3.1 Métodos baseados em *Grids* para Representação de Fluidos

Os métodos baseados em *Grids* (Malhas), têm sua base na discretização do espaço onde ocorre a simulação em subespaços de tamanho fixo (Células), transformando o espaço inteiro em uma Malha n-dimensional. Nestes métodos, a evolução do fluido é dada pela evolução do espaço de velocidade *v,* o espaço de densidade ρ e o espaço de pressão p, avaliados em cada célula. A figura 3.1 ilustra o grid. A maneira como as três quantidades são avaliadas na malha varia de acordo com o método.

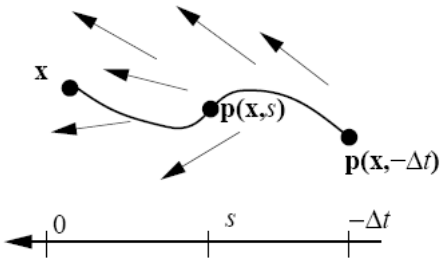


Figura 3.2: Retrocesso no tempo de execução para obtenção da grandeza em "x" (STAM, 1999)

3.1.1 Método Semi-Lagrangeano

Stam (1999) considera o valor das grandezas de interesse no centro de cada célula, e obtém este valor usando o valor que a "porção de líquido" em questão possuía no instante anterior de simulação.

Por exemplo, para obter a velocidade do fluido em uma célula qualquer do grid no instante $t + \Delta t$, o método utiliza o grid para percorrer o campo vetorial de velocidade no sentido inverso ("voltando" no tempo), e obtém a velocidade que aquela porção de líquido possuía no instante t. Segundo Stam, a velocidade na nova célula é igual à velocidade na célula original, no tempo inicial, conforme ilustra a figura 3.2.

Este método é chamado de Semi-Lagrangeano porque os métodos originais lagrangeanos para a simulação de fluidos são baseados em partículas, e não em grids de posições fixas. Neste método, a "partícula" é apenas uma abstração para a resolução do sistema euleriano, não aparecendo necessariamente na solução em código.

Stam aponta que seu método é isento de instabilidade numérica, e o considera extremamente eficiente na simulação realista em tempo real de gases, enquanto Foster e Metaxas (1996) discutem o problema da dissipação de massa em métodos baseados em grids que dificulta seu uso na simulação de

líquidos. A dissipação provém do fato de que o grid por si só não garante que a cada instante toda a massa de líquido está distribuída nas células do grid — é possível que o somatório das massas em cada célula seja menor do que o valor inicial de massa utilizada. Este problema não acontece com modelos baseados em partículas porque cada partícula carrega consigo uma quantidade constante de massa.

3.1.2 *Marker-and-Cell (MAC) Grids*

O método *Marker-and-Cell*, originalmente proposto por Harlow e Welch (1965), tem como característica principal o fato de avaliar diferentes grandezas em diferentes regiões das células, ao contrário do método anterior que sempre as avalia nos centros.

A pressão p continua sendo avaliada no centro de cada célula, porém o vetor de velocidade $\vec{v}$ é decomposto em suas três coordenadas, u, v, e w, e cada coordenada é avaliada em um dos três pares de faces opostos da célula, conforme ilustra a figura 3.3.

O nome do método deriva da necessidade de saber aonde se encontra a superfície do fluido (por exemplo, a superfície do líquido). Para resolver este problema, Harlow e Welch (1965) incluem no conjunto de células um grupo de partículas, chamadas de *Marker Particles* (partículas marcadoras), inicialmente posicionadas em células que sabidamente contém o fluido. Estas partículas se movem com o fluido, utilizando uma interpolação linear das velocidades nas várias faces da célula que contém a partícula para determinar a velocidade da mesma.

Desta forma, a qualquer momento da simulação, células que contém partículas são células com fluido, e células sem partículas estão vazias. Células cheias que possuem vizinhas vazias são células de superfície.

Este método simplifica a obtenção da superfície do líquido, sob custo da

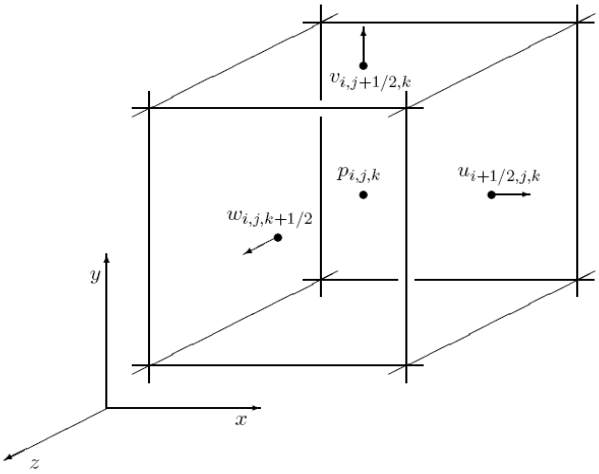


Figura 3.3: Uma célula com a densidade p avaliada em seu centro, e a velocidade $\vec{v}$ decompostas em u, v e w. Os índices i, j e k possuem incrementos de $\frac{1}{2}$ ao longo dos respectivos eixos u, v e w porque a indexação é feita em relação ao centro da célula (BRIDSON; FISCHER; GUENDELMAN, 2006).

perda da garantia da estabilidade numérica de Stam (1999).

3.1.3 Aproximações *HeightField*

As aproximações baseadas em *Heightfields* para grids surgiram do fato de que, para algumas aplicações, é desnecessário e custoso levar em conta todo o volume do fluido considerado para se obter uma simulação realista. Bridson, Fischer e Guendelman (2006) apontam como exemplo a simulação de um lago. Do ponto de vista de um observador externo, só interessa a evolução da superfície do lago.

Nestes métodos, o grid Euleriano é visto como um conjunto de "colunas" de fluido. Ao longo da simulação, a altura de cada coluna varia e o conjunto de alturas h das colunas caracteriza a superfície do fluido. As desvantagens apontadas por Kass e Miller (1990) incluem a inabilidade de simular "respin-

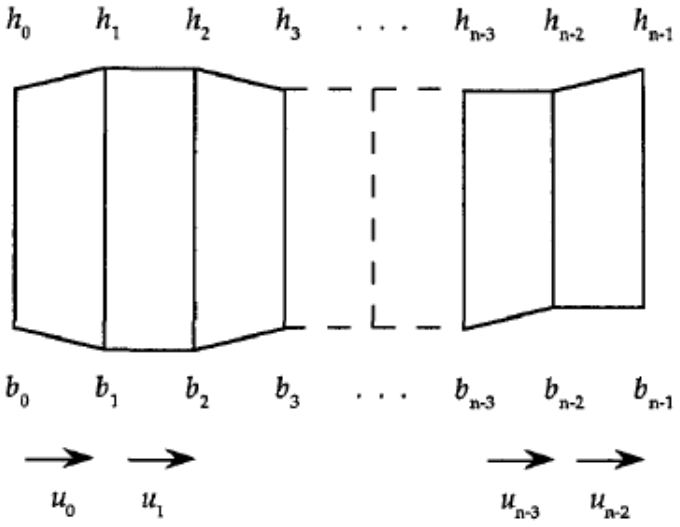


Figura 3.4: *Heightfield* em duas dimensões: as colunas variam de altura *h* ao longo da simulação (KASS; MILLER, 1990).

gos" de água e ondas quebrando. A vantagem marcante é a grande redução da complexidade computacional, conforme indicam Kass e Miller (1990) em seu trabalho relacionado.

Os métodos baseados em grids, por possuírem pontos fixos no espaço onde avaliar o estado do fluido, possuem como vantagem a simplicidade de implementação (de fato, Stam (2003) desafia o leitor a implementar um algoritmo mais simples que o seu, que possui em torno de 100 linhas de código escrito na linguagem C). Estes métodos se estabeleceram na literatura por serem úteis para simulações estáticas e sem variações durante a execução. São, porém, computacionalmente caros e de difícil uso em aplicações que requerem interatividade com o usuário em tempo real.

3.2 Sistemas de Partículas

Sistemas de partículas foram originalmente propostos por Reeves (1983) para modelar objetos que o autor denominou *fuzzy* (“nebulosos”), como o fogo, água, fumaça, etc. As características principais deste tipo de objeto são as seguintes:

1. Objetos deste tipo não são representados por um conjunto de polígonos que delimitam sua superfície, mas por uma nuvem de tipos primitivos denominados *Partículas* que definem o seu volume.

2. Estes objetos não são rígidos - as partículas se movimentam, alteram sua forma, saem e entram do sistema com o passar do tempo.

3. Geralmente há um fator estocástico (aleatório) que influencia o comportamento geral do objeto, isto é, o comportamento do objeto não é determinístico.

Desde então, sistemas de partículas foram refinados para representar uma grande variedade de elementos - sólidos deformáveis, como por exemplo órgãos do corpo humano (JAILLET; SHARIAT; VANDORPE, 1997), fogo e fumaça (STAM, 2000), líquidos (MULLER; CHARYPAR; GROSS, 2003; PREMOZE et al., 2003; MUELLER et al., 2004), entre outros. Cada tipo de elemento possui suas características distintas - objetos sólidos, por exemplo, atribuem menos liberdade de movimento às partículas, enquanto é comum modelar o fogo como um conjunto de partículas onde cada partícula “nasce”, se desloca para cima com algum grau de variação (o elemento estocástico), e “morre”.

Para líquidos, que são o foco deste trabalho, o sistema é definido da seguinte forma: A massa do líquido é fixa, e como a massa é distribuída entre as partículas, o número de partículas é fixo. A evolução do sistema ao longo

do tempo é regida por equações conhecidas da mecânica dos fluidos, de forma que não há necessidade do elemento aleatório.

Sistemas de partículas possuem vantagens sobre a maneira clássica baseada em malhas para simulação de fluidos: São geralmente mais computacionalmente eficientes, por avaliarem os parâmetros que regem o estado do fluido apenas em pontos específicos do espaço, ao contrário de em todo ele. Por serem mais eficientes, eles são mais facilmente implementados em aplicações que requerem interatividade com o usuário. Em contrapartida, uma desvantagem significativa é a dificuldade da geração da superfície do líquido representado pelas partículas.

3.2.1 *Smoothed Particle Hydrodynamics*

O método *Smoothed Particle Hydrodynamics* (SPH) (LUCY, 1977; GINGOLD; MONAGHAN, 1977; MONAGHAN, 1992) foi originalmente concebido para a solução de modelos astrofísicos em três dimensões, mas é genérico a ponto de ter sido adaptado para vários outros campos da engenharia e computação, particularmente a simulação de fluidos compressíveis e não compressíveis.

No método SPH, as grandezas de interesse do sistema, como densidade e pressão, são discretizadas nas partículas através de uma interpolação ponderada do valor destas grandezas nas partículas vizinhas. Ou seja, o estado de cada partícula depende do estado de suas partículas vizinhas. Generalizando para uma grandeza A na posição espacial r:

$$A(r) = \sum_{j=1}^{n} m_j \frac{A_j}{\rho_j} W(r - r_j, h), \tag{3.3}$$

onde:

$A(r)$ é o valor da grandeza A na posição espacial r;

m_j é a massa da partícula *j;*

A_j é o valor da grandeza A para a partícula j;

ρ_j é a densidade da partícula *j;*

W é a função de suavização utilizada, arbitrária;

r_j é a posição espacial da partícula j;

h é uma constante que define o raio máximo de atuação da função W.

A função de suavização *W (Smoothing Kernel)* é o que caracteriza o método. É intuitivo pensar neste núcleo como uma função que leva todo valor de distância $r - r_j$ do centro de uma esfera de raio h a um valor de interpolação. O método então consiste em determinar as grandezas relevantes ao sistema de fluidos através dos núcleos de suavização, utilizando a equação 3.3. O método SPH foi o escolhido para a simulação do fluido neste trabalho e será discutido com mais detalhes na seção 3.3.

3.2.2 Método *Moving Particle Semi-Implicit (MPS)*

Este método, originalmente proposto por Koshizuka, Nobe e Oka (1998) também utiliza a idéia de que o estado de uma partícula é influenciado por suas partículas vizinhas. Desta forma, existe neste método uma função de peso w que, semelhantemente ao método SPH, interpola um valor a partir de uma distância $r - r_j$. A diferença básica entre os dois métodos é o comportamento da função de peso e o algoritmo de atualização das grandezas do líquido.

Premoze et al. (2003) citam diversas aplicações do método MPS, como transições de fase, fluxo multifásico, fluidos com sedimentação e estruturas elásticas.

3.3 Simulação de Fluidos utilizando *Smoothed Particle Hydrodynamics*

Por praticidade, apresenta-se novamente a equação da conservação do momento de *Navier-Stokes* (3.2):

$$\rho\left(\frac{\partial v}{\partial t} + v.\nabla v\right) = -\nabla p + \vec{f}_{ext} + \mu\nabla^2\vec{v}.$$

Muller, Charypar e Gross (2003) afirmam que a expressão $\frac{\partial v}{\partial t} + v.\nabla v$ pode ser simplificada a $\frac{\partial v}{\partial t}$ pois, ao contrário dos sistemas estáticos baseados em malhas, em sistemas de partículas as mesmas se deslocam junto com o fluido, significando que a variação do espaço vetorial de velocidade do fluido é representado simplesmente pela variação no tempo da velocidade das partículas que o compõem. Desta forma, dividindo esta simplificação de 3.2 por ρ, obtém-se:

$$\frac{\partial v}{\partial t} = \frac{-\nabla p + \rho\vec{f}_{ext} + \mu\nabla^2\vec{v}}{\rho}, \tag{3.4}$$

que lembra intuitivamente a segunda lei de Newton, aplicada para líquidos:

$$\vec{a} = \frac{\vec{F}}{\rho}.$$

A tarefa de um sistema computacional que utiliza o método *Smoothed Particle Hydrodynamics* (SPH) para simular fluidos é utilizar a interpolação do método (equação 3.3) em cada componente de $\vec{F}$ para obter a aceleração das partículas e simular seu movimento. A maneira como o método é aplicado a cada diferente grandeza é o tópico desta seção.

São os núcleos de suavização utilizados no método SPH que descrevem a forma como as partículas interagem umas com as outras e o comportamento

total do fluido sendo simulado. A equação 3.3 na página 47 dita o valor de uma grandeza A qualquer na posição representada pelo ponto r. Muller, Charypar e Gross (2003) afirmam que as derivadas destas grandezas, comuns em equações de fluidos, afetam apenas o núcleo de suavização utilizado. Esta propriedade mostra-se útil na simulação de fluidos pois a equação simplificada utilizada, que é a Equação 3.4, apresenta gradientes e laplacianos de várias das grandezas envolvidas.

Desta forma, pode-se simplificar o gradiente de uma grandeza A, que é o vetor formado pelas derivadas parciais de A, conforme mencionado no início deste capítulo:

$$\nabla A(r) = \sum_{j=1}^{n} m_j \frac{A_j}{\rho_j} \nabla W(r - r_j, h) \tag{3.5}$$

E seu laplaciano, que é a soma das derivadas parciais de segunda ordem não mistas:

$$\nabla^2 A(r) = \sum_{j=1}^{n} m_j \frac{A_j}{\rho_j} \nabla^2 W(r - r_j, h) \tag{3.6}$$

A equação 3.4 na página precedente apresenta a variação da velocidade em um fluido representado por um sistema de partículas ao longo do tempo. O método apresentado por Muller, Charypar e Gross (2003) consiste em resolver a equação 3.4 através de SPH, de forma a obter a variação da velocidade e simular o fluido. Resolver a expressão $-\nabla p + \vec{f}_{ext} + \mu \nabla^2 \vec{v}$ através de SPH significa interpolar cada componente desta expressão utilizando os núcleos de suavização.

Como os núcleos de suavização possuem um papel integral no método, a Seção 3.3.1 irá discutir suas propriedades e os núcleos específicos adotados neste trabalho.

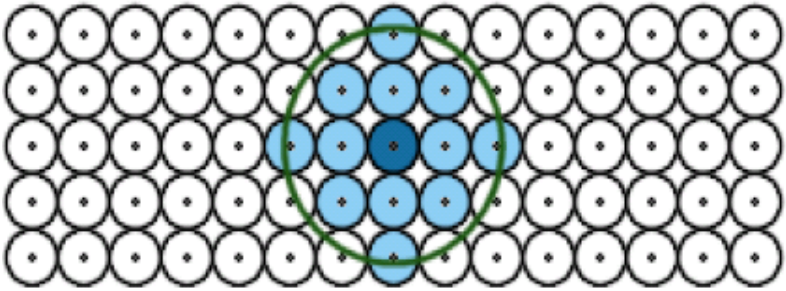


Figura 3.5: O núcleo de suavização pode ser visto como uma esfera ao redor de uma partícula (KELAGER, 2006).

3.3.1 Núcleos de Suavização

A maneira mais simples e intuitiva de abordar os núcleos de suavização é pensar nas partículas no espaço. Se isolarmos uma destas partículas, a qual chamaremos de p, tomarmos uma esfera hipotética de raio h ao redor de p e considerarmos que cada partícula dentro desta esfera hipotética contribui para o comportamento de p, então um núcleo de suavização explicita a maneira pela qual esta contribuição é feita. A região compreendida dentro desta esfera hipotética é chamada de *vizinhança* de p e todas as partículas dentro desta região são ditas *na vizinhança* de p.

Um núcleo de suavização é uma função. Segundo Muller, Charypar e Gross (2003), se esta função assumir as propriedades de reflexão e normalização, ela é de segunda ordem de precisão, o que significa dizer que o erro obtido ao interpolar $A(r)$ utilizando o núcleo é da ordem de $O\left(h^2\right)$ou menor (KELAGER, 2006), onde h é o raio da esfera.

A primeira propriedade diz que o núcleo deve ser uma função reflexiva, ou seja, $W\left(\vec{r},h\right)$ deve ser igual a $W\left(-\vec{r},h\right)$. Isto permite que as partículas sejam relacionadas em qualquer ordem.

A segunda propriedade diz que o núcleo deve ser normalizado, para que

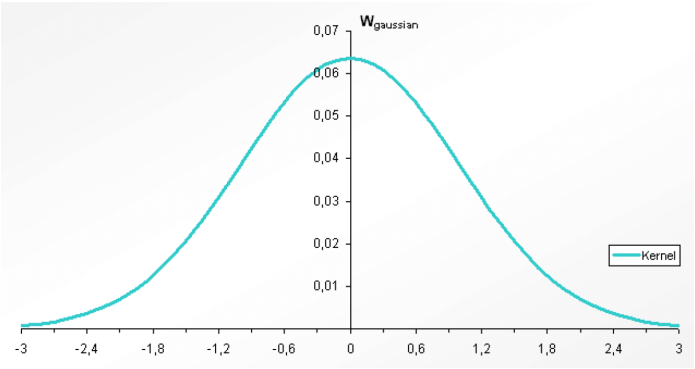


Figura 3.6: O núcleo gaussiano com $h = 1$(KELAGER, 2006).

as constantes presentes na função da grandeza A sejam interpoladas corretamente (GINGOLD; MONAGHAN, 1977), ou seja:

$$\int W(\vec{r},h)\,d\vec{r} = 1$$

Adicionalmente, existe um limite ao módulo de $\vec{r}$, acima do qual a função sempre vale zero. Na analogia de que o núcleo de suavização representa uma esfera, este limite é o raio da mesma, h. Este valor é também chamado de *raio de suporte* do núcleo, e serve para limitar o número de partículas em uma vizinhança. Isto é computacionalmente interessante, visto que uma partícula não dependerá de todas as outras do sistema. A figura 3.6 exemplifica o gráfico do núcleo de suavização gaussiano com $h = 1$.

A escolha do valor do raio de suporte h influencia o comportamento do sistema e não é diretamente disponível na literatura. Valores pequenos demais implicam em poucas partículas em uma região e aumentam a localidade do comportamento da partícula, enquanto que valores grandes demais diminuem a contribuição de partículas próximas ao centro da esfera do núcleo de suavização. Adicionalmente, valores grandes para h significam um aumento no tempo de processamento pelo fato de que cada partícula terá mais vizi-

nhos, em média. Há então um balanceamento a ser feito na escolha do valor de h entre tempo de processamento e estabilidade. Dados o volume total V do fluido a ser simulado, o número n de partículas a serem utilizadas na sua representação, e um número médio x de partículas dentro de uma região de vizinhança qualquer (x é definido arbitrariamente), Kelager (2006) utiliza a seguinte equação para obter o valor do raio de suporte:

$$h = \sqrt[3]{\frac{3Vx}{4\pi n}}.$$

Esta fórmula é obtida do fato que $\frac{n}{V}$ é a densidade das partículas no líquido, ou seja, o número de partículas por unidade de volume. Multiplicando este valor pelo volume de uma esfera de raio h, obtém-se o número de partículas que ocupam aquele volume:

$$x = \left(\frac{n}{V}\right)\frac{4}{3}\pi h^3.$$

Kelager (2006) sugere que x seja testado até o menor número possível que simule o fluido e suas propriedades convincentemente.

Nas seguintes subseções serão apresentados os núcleos de suavização utilizados neste trabalho, conforme o modelo de Muller, Charypar e Gross (2003). Em todos os casos, se $|\vec{r}| > h$, $W(\vec{r},h) = 0$.

Núcleo de Suavização *Poly6*

O primeiro núcleo proposto por Muller, Charypar e Gross (2003) não requerer o cálculo de raízes, computacionalmente caros:

$$W_{poly6}(\vec{r},h) = \frac{315}{64\pi h^9}\left(h^2 - |\vec{r}|^2\right)^3, \tag{3.7}$$

$$\nabla W_{poly6}(\vec{r},h) = -\frac{945}{32\pi h^9}\vec{r}\left(h^2-|\vec{r}|^2\right)^2, \tag{3.8}$$

$$\nabla^2 W_{poly6}(\vec{r},h) = -\frac{945}{32\pi h^9}\left(h^2-|\vec{r}|^2\right)\left(3h^2-7|\vec{r}|^2\right). \tag{3.9}$$

Núcleo de Suavização *Spiky*

Este núcleo é utilizado no cálculo da força de pressão sobre as partículas, e sua utilidade será explicada na subseção 3.3.2:

$$W_{spiky}(\vec{r},h) = \frac{15}{\pi h^6}(h-|\vec{r}|)^3 \tag{3.10}$$

$$\nabla W_{spiky}(\vec{r},h) = -\frac{45}{\pi h^6}\frac{\vec{r}}{|\vec{r}|}(h-|\vec{r}|)^2 \tag{3.11}$$

$$\nabla^2 W_{spiky}(\vec{r},h) = -\frac{90}{\pi h^6}\frac{1}{|\vec{r}|}(h-|\vec{r}|)(h-2|\vec{r}|) \tag{3.12}$$

Núcleo de Suavização *Viscosity*

O terceiro núcleo proposto por Muller, Charypar e Gross (2003) é utilizado no cálculo da força de viscosidade atuando sobre as partículas, por uma razão que será discutida na subseção 3.3.2:

$$W_{viscosity}(\vec{r},h) = \frac{15}{2\pi h^3}-\frac{|\vec{r}|^3}{2h^3}+\frac{|\vec{r}|^2}{h^2}+\frac{h}{2|\vec{r}|}-1 \tag{3.13}$$

$$\nabla W_{viscosity}(\vec{r},h) = \frac{15}{2\pi h^3}\vec{r}\left(\frac{-3|\vec{r}|}{2h^3}+\frac{2}{h^2}-\frac{h}{2|\vec{r}|^3}\right) \tag{3.14}$$

$$\nabla^2 W_{viscosity}(\vec{r},h) = \frac{45}{\pi h^6}(h-|\vec{r}|) \tag{3.15}$$

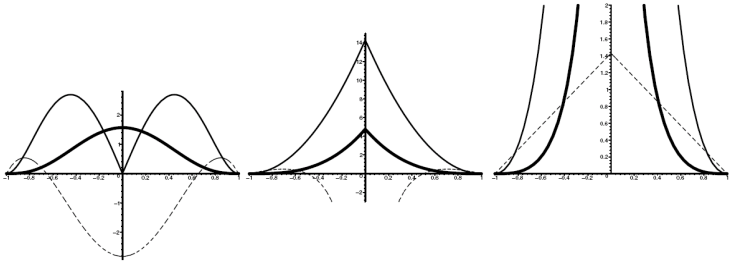


Figura 3.7: Os três núcleos utilizados neste trabalho: *Poly6, Spiky* e *Viscosity*, respectivamente. As linhas grossas representam o valor dos núcleos, as finas o valor dos gradientes, e as tracejadas os laplacianos (MULLER; CHARYPAR; GROSS, 2003).

3.3.2 As grandezas que governam os fluidos em SPH

Finalmente é possível aplicar o método SPH sobre a equação da variação da velocidade no fluido (equação 3.4). Para isso é preciso utilizar os núcleos de suavização em cada grandeza envolvida na expressão $-\nabla p + \vec{f}_{ext} + \mu\nabla^2\vec{v}$.

Observando novamente a equação geral do método (equação 3.3 na página 47),

$$A(r) = \sum_{j=1}^{n} m_j \frac{A_j}{\rho_j} W(r - r_j, h),$$

nota-se que, além do valor da própria grandeza A, a equação também depende da massa das partículas (m) e da densidade das massas (ρ) em uma vizinhança. A massa é conhecida e constante, definida no início da simulação. A densidade das massas varia de acordo com a vizinhança de uma partícula, de forma que esta grandeza sempre deve ser calculada primeiro, antes de ser possível obter as outras. Esta é a única restrição de ordem no método, todas as outras grandezas podem ser calculadas na ordem que for mais conveniente.

Densidade das massas

O valor da densidade das massas em uma partícula pode ser entendido como a quantidade de massa em sua vizinhança. Quanto maior o número de partículas em uma determinada vizinhança, maior a densidade das massas.

A equação para o cálculo da densidade das massas é obtida simplesmente substituindo-se a variável A por ρ na equação 3.3. O resultado desta substituição é:

$$\begin{aligned} A(r) &= \sum_{j=1}^{n} m_j \frac{A_j}{\rho_j} W(r - r_j, h), \\ \rho(r) &= \sum_{j=1}^{n} m_j \frac{\rho_j}{\rho_j} W_{poly6}(r - r_j, h), \\ \rho(r) &= \sum_{j=1}^{n} m_j W_{poly6}(r - r_j, h). \end{aligned} \tag{3.16}$$

Assim, a densidade das massas em uma partícula depende apenas do núcleo de suavização e do valor da massa das partículas em sua vizinhança e de sua própria massa.

Pressão

De posse da densidade das massas, é possível calcular a pressão de cada partícula diretamente, sem a necessidade da interpolação do método SPH. Este valor pode ser obtido através da fórmula do gás ideal (KELAGER, 2006)

$$pV = nRT \tag{3.17}$$

onde V é o volume por unidade de massa, n é o número de partículas de gás em 1 mol, R é a constante universal dos gases, e T é a temperatura.

Se a massa do fluido é constante (ou seja, não existe introdução ou perda de fluido durante a simulação) e a temperatura também, a expressão nRT pode ser abreviada para uma constante k. O resultado se torna

$$pV = k. \tag{3.18}$$

V é o volume em uma unidade de massa. A densidade é definida como a quantidade de massa por volume. Em uma unidade de massa, obtém-se:

$$\begin{aligned} \rho &= \frac{m}{V}, \\ \rho &= \frac{1}{V}, \\ V &= \frac{1}{\rho}. \end{aligned}$$

Substituindo na equação 3.18, obtém-se

$$\begin{aligned} p\frac{1}{\rho} &= k, \\ p &= k\rho. \end{aligned} \tag{3.19}$$

O valor p obtido na equação 3.19 é utilizado posteriormente no cálculo da força de pressão exercida sobre uma partícula (subseção 3.3.2). Ele pode ser visto como o "potencial" que uma partícula apresenta para influenciar suas vizinhas. Quanto maior a densidade das massas de uma partícula, maior o valor da pressão p, e consequentemente maior a força que ela exercerá para repelir suas vizinhas. A figura 3.8 ilustra esta relação.

Desbrun e Gascuel (1996) comentam que o uso da equação 3.19 resulta em forças puramente repulsivas. Em aplicações em astrofísica, a força de

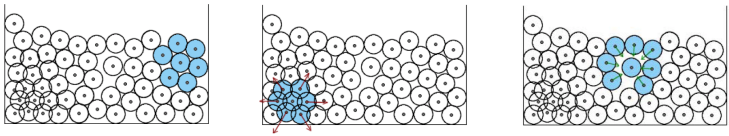


Figura 3.8: A relação entre a densidade das massas e a pressão. No primeiro caso, as partículas destacadas estão em equilíbrio. No segundo, a concentração de partículas aumenta a densidade das massas e consequentemente, a pressão na região. Por fim, no terceiro caso a baixa densidade das massas acarreta em uma pressão baixa, porém existente (KELAGER, 2006).

pressão é comumente combinada com forças gravitacionais que balanceiam a repulsão. Forças de pressão puramente repulsivas fazem sentido em gases, que tendem a se expandir e ocupar todo o espaço, mas líquidos tendem a manter uma coesão interna (KELAGER, 2006). Desbrun e Gascuel (1996) propõem uma alternativa à equação 3.19:

$$p = k(\rho - \rho_0), \tag{3.20}$$

onde ρ_0 é chamado de densidade de repouso e é uma propriedade física do tipo de fluido. O resultado é que partículas distantes tenderão a se atrair, enquanto que partículas muito próximas tenderão a se afastar. Desbrun e Gascuel (1996) identificam duas vantagens significativas na inclusão de ρ_0 à equação 3.19.

Primeiro, se as partículas possuírem a mesma massa (ou seja, se a massa do líquido for igualmente distribuída entre as partículas), elas tenderão a se distribuir uniformemente no espaço. Segundo, como as partículas tenderão a encontrar um estado de equilíbrio entre a atração e a repulsão, a densidade das massas tenderá a ser constante, o que resulta em um volume total aproximadamente constante. Desta forma, o fluido tenderá a assumir este estado de equilíbrio após uma deformação.

Kelager (2006) chama a atenção para um cuidado em especial com a

constante k. Este valor, como mencionado, representa as três constantes nRT da equação 3.17, mas os valores físicos reais dessas grandezas não podem ser utilizados na obtenção de k por gerarem valores absurdamente grandes, fato que também foi observado durante a implementação deste sistema. Desta forma, k deve ser obtido empiricamente, idealmente sendo o maior possível que ainda permita uma simulação realista. Quanto maior este valor, menor deve ser o intervalo de tempo entre iterações da simulação, para evitar instabilidade no fluido.

Uma hipótese para os valores exageradamente grandes de k obtidos por Kelager (2006) é que a lei do gás ideal, além de ser um modelo estritamente teórico, diz respeito apenas a substâncias no estado gasoso (ATKINS, 1999). Como fluidos no estado líquido possuem pressão interna e temperatura intuitivamente mais baixas do que gases, faz sentido que k deva ser amenizada quando simulando líquidos.

De posse da densidade e da pressão nas partículas, pode-se extrair a força de pressão resultante em cada uma delas.

Força de Pressão

Retornando à equação 3.4, a parcela relevante à força de pressão$(-\nabla p)$ é obtida pelo gradiente do núcleo de suavização *Spiky*, da Seção 3.3.1 na página 54:

$$-\nabla p\,(r) = -\sum_{j=1}^{n} m_j \frac{p_j}{\rho_j} \nabla W_{spiky}\,(r - r_j, h)\,. \tag{3.21}$$

No método SPH, a componente da força de pressão utiliza o gradiente do núcleo de suavização. O gráfico do gradiente do núcleo *Poly6* (Figura 3.7 na página 55) mostra que conforme $|\vec{r}|$ se aproxima de 0, $\nabla W\,(\vec{r}, h)$ também se anula. O que isto significa, espacilamente, é que a força de pressão ge-

rada entre duas partículas tenderá a zero conforme elas se aproximam. Este comportamento é exatamente o oposto do intuitivamente esperado pela força de pressão, onde partículas muito próximas deveriam se repelir com uma intensidade maior do que partículas fisicamente distantes. Desbrun e Gascuel (1996) propuseram então o núcleo *Spiky*, cujo gráfico da Figura 3.7 ilustra que $|\nabla W(\vec{r},h)|$ tende a aumentar conforme $|\vec{r}|$ tende a 0. Tal estratégia foi posteriormente adotada por Muller, Charypar e Gross (2003) e Kelager (2006), sendo usada também neste trabalho.

Desbrun e Gascuel (1996) atentam ao fato de a força resultante da equação 3.21 não ser simétrica. Em outras palavras, a força que uma partícula exerce sobre sua vizinha não será igual à força que ela sofre em retorno, a não ser que as densidades das massas de ambas sejam iguais, o que raramente ocorrerá. Como resultado, a lei de ação-reação não é garantida e o resultado não é realista.

Os autores propõem a seguinte variação à equação, que garante a simetria:

$$-\nabla p(\vec{r}_i) = -\rho_i \sum_{j=1}^{n} m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \nabla W(\vec{r} - \vec{r}_j, h),$$

enquanto que Muller, Charypar e Gross (2003) propõem uma outra alternativa, computacionalmente mais eficiente, que utiliza a média das pressões. Esta última é a versão adotada neste trabalho:

$$-\nabla p(\vec{r}_i) = -\sum_{j=1}^{n} \left(\frac{p_i + p_j}{2} \right) \frac{m_j}{\rho_j} \nabla W(\vec{r} - \vec{r}_j, h).$$

Força de Viscosidade

A viscosidade em um fluido representa a capacidade do fluido de resistir à deformação. Conforme o fluido se deforma, suas moléculas atritam entre

si, gerando calor e reduzindo a energia cinética (KELAGER, 2006).

A parcela na equação 3.2 referente à viscosidade é $\mu\nabla^2\vec{v}$. Interpolando com o método SPH obtém-se:

$$\mu\nabla^2\vec{v} = \mu\sum_{j=1}^{n}\vec{v}_j\nabla^2 W\left(r-r_j,h\right). \tag{3.22}$$

Esta equação gera forças assimétricas, tal qual a Equação 3.21, pois a velocidade varia entre as partículas. Uma alternativa é utilizar no lugar da velocidade absoluta a velocidade relativa entre as partículas (MULLER; CHARYPAR; GROSS, 2003):

$$\mu\nabla^2\vec{v} = \mu\sum_{j=1}^{n}\left(\vec{v}_j-\vec{v}_i\right)\nabla^2 W\left(r-r_j,h\right).$$

Muller, Charypar e Gross (2003) afirmam que, se $\nabla^2 W\left(\vec{r}-\vec{r}_j,h\right)$ não for positivo para todo $|\vec{r}|<h$, a força de viscosidade ao invés de agir como um atenuante irá introduzir energia no sistema e *aumentar* a velocidade das partículas. Desta forma, tanto o núcleo *Poly6* quanto o *Spiky* não podem ser utilizados no cálculo da força de viscosidade, por possuírem valores negativos em determinados pontos de seus laplacianos (Figura 3.7 na página 55). Os autores então propuseram o núcleo *Viscosity*, cujo laplaciano é positivo em todo o domínio, como a própria Figura 3.7 ilustra.

Forças externas

A última parcela restante na expressão $-\nabla p+\vec{f}_{ext}+\mu\nabla^2\vec{v}$ diz respeito às forças externas que influenciam o fluido. Neste trabalho, são três as forças externas utilizadas:

$$\vec{f}_{ext} = \vec{f}_{grav}+\vec{f}_{sup}+\vec{f}_{eixo},$$

onde:

$\vec{f}_{grav}$ equivale a força exercida pelo campo gravitacional da Terra (Subseção 3.3.2),

$\vec{f}_{sup}$ é a força exercida na superfície do líquido (Subseção 3.3.2),

$\vec{f}_{eixo}$ é a força exercida pela estrutura que contém o líquido, se aplicável. Esta força é o tópico do capítulo 4.

Força da Gravidade

A força que a aceleração gravitacional exerce sobre o fluido é igual em todas as partículas, e vale (MULLER; CHARYPAR; GROSS, 2003):

$$\vec{f}_i = \rho_i \vec{g}$$

onde $\vec{g}$ é a aceleração da gravidade, geralmente denotada como $\begin{bmatrix} 0.0 & -9.82 & 0.0 \end{bmatrix}^T$.

Força de Superfície

A segunda força externa atuando sobre o fluido é aquela agindo sobre a superfície do mesmo. Esta força aponta sempre para o interior do fluido e serve para suavizar curvaturas acentuadas (figura 3.9) (MORRIS, 2000).

Para simular a força de superfície Muller, Charypar e Gross (2003) utilizam as idéias de Morris (2000), que diz:

$$f_{sup}(r) = -\sigma \nabla^2 c(r) \frac{\vec{n}(r)}{|\vec{n}(r)|}, \tag{3.23}$$

onde:

- σ é um coeficente de tensão que depende das duas substâncias que

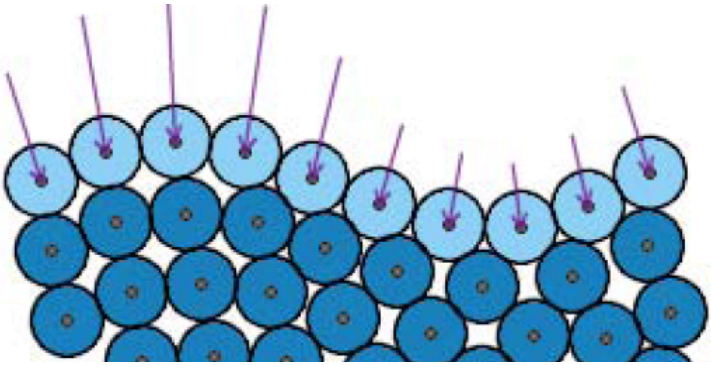


Figura 3.9: A força exercida na superfície do fluido aponta para seu interior.(KELAGER, 2006)

interagem na superfície (como o ar agindo na superfície da água);

- $\vec{n}_i$ é o vetor normal que aponta para dentro do fluido na posição da partícula i;

- c_i é o valor do *campo de cor* na posição da partícula i.

O *campo de cor* é um quantidade adicional que sempre vale exatamente 1 nas posições do espaço que contém fluido e 0 em todos os outros pontos. Assim, as partículas representam os pontos não-nulos deste campo. Com a interpolação do método SPH, este campo vale

$$\begin{aligned} c(r) &= \sum_j c_j \frac{m_j}{\rho_j} W(r - r_j, h), \\ &= \sum_j \frac{m_j}{\rho_j} W(r - r_j, h). \end{aligned}$$

E o gradiente deste valor, que vale

$$\begin{aligned}\vec{n}(r) &= \nabla c(\vec{r}), \\ &= \sum_j \frac{m_j}{\rho_j} \nabla W(r - r_j, h),\end{aligned}$$

é o vetor normal que aponta para dentro da superfície do fluido na posição r_i (MULLER; CHARYPAR; GROSS, 2003). Este vetor só possui módulo diferente de zero na região próxima da superfície, de forma que $|\vec{n}|$ tende a zero conforme as partículas se distanciam das bordas do fluido. E conforme $|\vec{n}|$ tende a zero, $\frac{\vec{n}}{|\vec{n}|}$ tende a um vetor com módulo infinitamente grande, introduzindo instabilidade no fluido (KELAGER, 2006). Uma maneira de evitar esta instabilidade é calcular a força de superfície apenas para partículas onde $|\vec{n}| > \varepsilon$, onde ε é algum limite observado empiricamente. Kelager (2006) recomenda o uso de

$$\varepsilon = \sqrt{\frac{\rho}{x}}$$

onde ρ é a densidade do material e x é o número médio de partículas em uma vizinhança. Outra possibilidade é o limite de Morris (2000) , que vale

$$\varepsilon = \frac{0.01}{h},$$

onde h é o raio de suporte dos núcleos de suavização. Finalmente, o laplaciano do campo de cor c vale

$$\nabla^2 c(r) = \sum_j \frac{m_j}{\rho_j} \nabla^2 W(r - r_j, h).$$

Agora é possível obter o valor da força de superfície atuando sobre as partículas:

$$
\begin{aligned}
f_{sup}(r) &= -\sigma\nabla^2 c(r)\frac{\vec{n}(r)}{|\vec{n}(r)|}, \\
f_{sup}(r) &= -\sigma(\sum_j \frac{m_j}{\rho_j}\nabla^2 W(r-r_j,h))\frac{(\sum_j \frac{m_j}{\rho_j}\nabla W(r-r_j,h))}{|(\sum_j \frac{m_j}{\rho_j}\nabla W(r-r_j,h))|}. \quad (3.24)
\end{aligned}
$$

3.3.3 Integração Temporal

O resultado da aplicação do método SPH sobre a equação de conservação de momento (Equação 3.4 na página 49) é um vetor de aceleração $\vec{a}$ para cada partícula. O passo seguinte é aplicar este vetor sobre as partículas para atualizar suas velocidades e posições e assim simular a dinâmica do fluido.

Idealmente, todo este cálculo seria feito instantâneamente e a posição de cada partícula seria atualizada continuamente, em um intervalo de tempo absolutamente pequeno. Como isto não é possível, é preciso aplicar o vetor de aceleração obtido em um instante qualquer durante um intervalo de tempo consideravelmente grande, para "anular" pelo menos uma parcela do custo do processamento do estado do fluido.

A escolha do tamanho deste intervalo de tempo (ou tamanho do *frame*) deve ser cuidadosa: Valores pequenos demais significarão uma simulação lenta, e valores grandes demais introduzirão instabilidade na simulação.

O método mais simples de aplicar a aceleração na atualização da velocidade e posição das partículas é o esquema de Euler (KELAGER, 2006). Nele:

$$
\begin{aligned}
\vec{v}_{t+\triangle t} &= \vec{v}_t + \triangle t\vec{a}_t, \\
R_{t+\triangle t} &= R_t + \triangle t\vec{v}_t.
\end{aligned}
$$

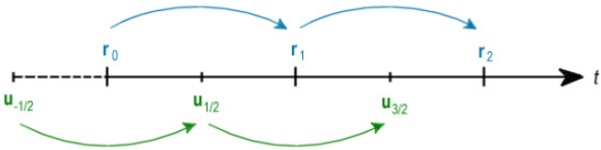


Figura 3.10: No método *Leap-Frog*, a velocidade (u) "pula" por cima da posição (r), e vice-versa (KELAGER, 2006).

onde R é a posição da partícula, $\vec{v}$ a sua velocidade e $\vec{a}$ sua aceleração. As equações são simples de implementar e eficientes, mas dependem de intervalos de tempo muito pequenos para manterem a estabilidade (HUT; MAKINO, 2006).

Vários outros métodos foram propostos na literatura, e neste trabalho utiliza-se um método denominado *Leap-Frog*. A característica principal do método *Leap-Frog* é que a posição de uma partícula é definida em intervalos de tempo t_i,t_{i+1},t_{i+2}, como no método de Euler, mas a velocidade é definida na metade destes intervalos, i.e. $t_{i-\frac{1}{2}}$, $t_{i+\frac{1}{2}}$,$t_{i+\frac{3}{2}}$ (HUT; MAKINO, 2006). O nome deste método vem do fato que a posição e a velocidade "pulam" uma sobre a outra (Figura 3.10).

Assim:

$$\begin{aligned} \vec{v}_{i+\frac{1}{2}} &= \vec{v}_{i-\frac{1}{2}} + \triangle t \vec{a}_i, \\ R_i &= R_{i-1} + \triangle t \vec{v}_{i-\frac{1}{2}}. \end{aligned}$$

Esta maneira de utilizar a aceleração para atualizar a velocidade e posição da partícula é mais estável, mas requer definir a velocidade em "meio-intervalos", que não são diretamente intuitivos. Hut e Makino (2006) apresentam uma outra maneira de reescrever as equações do método, agora considerando todas as grandezas em intervalos inteiros:

$$\vec{v}_{i+1} = \vec{v}_i + \frac{1}{2}(\vec{a}_i + \vec{a}_{i+1}) \triangle t, \quad (3.25)$$

$$R_{i+1} = R_i + \vec{v}_i \triangle + \frac{1}{2}\vec{a}_i(\triangle t)^2. \quad (3.26)$$

Busca de Vizinhança

A Seção 3.3.1 definiu a vizinhança de uma partícula p como a região esférica de raio h ao redor de p. As partículas contidas nesta região são ditas vizinhas de p e apenas elas afetam o estado de p no que diz respeito às grandezas discutidas ao longo desta seção. Desta forma, o primeiro passo de cada iteração da simulação geralmente consiste na definição da vizinhança de cada partícula.

Como as partículas movem-se livremente no espaço, cada uma é potencialmente vizinha de todas as outras, numa busca $O(n^2)$. Esta complexidade torna rapidamente impraticável o uso de um número de partículas acima de alguns milhares para aplicações interativas.

Uma divisão espacial simples é o uso de um grid tridimensional para conter as partículas. Como a região de influência possui raio conhecido e, no caso deste trabalho, fixo, se cada célula do grid possuir raio h então as candidatas em potencial para a vizinhança de uma partícula p estão todas contidas na célula de p e nas células imediatamente vizinhas (MULLER; CHARYPAR; GROSS, 2003), como ilustra a Figura 3.11. O custo da busca total por passo de simulação agora é $O(mn)$, onde n é o número total de partículas no sistema e m é o número médio de partículas por célula do grid.

3.3.4 Aceleração com a Utilização de Placas Gráficas

O custo do processo de simulação de fluidos por SPH vem principalmente da necessidade da utilização de um grande número de partículas para

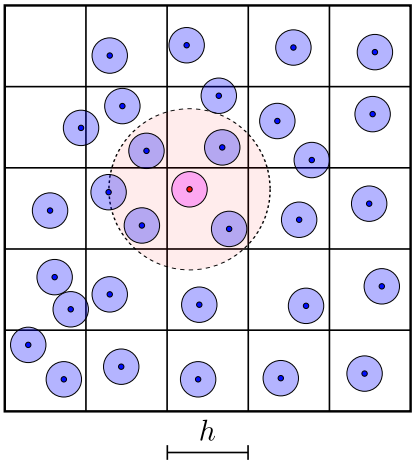


Figura 3.11: Uma vez que as células do grid possuem tamanho de lado igual ao raio de suporte h, as potenciais vizinhas de partícula destacada se encontram na sua célula e nas imediatamente próximas.

uma simulação convincente. Quanto mais partículas, maior é o tempo de processamento.

Felizmente, o processo é altamente paralelizável. Todos os componentes do lado direito da Equação 3.2 na página 40 bem como o passo de integração podem ser realizados para cada partícula independentemente das outras. De fato, a implementação utilizada neste trabalho explora o paralelismo em computadores com mais de uma CPU distribuindo o processamento em um número de threads.

Uma outra abordagem popular para a exploração do paralelismo do método SPH é o uso das placas gráficas (GPUs) no processo de simulação. O hardware gráfico foi inicialmente concebido e projetado para o processamento e geração de pixels na tela, num processo largamente independente. A tendência de utilizar o hardware gráfico para processamento de atividades mais gerais como simulações físicas, criptografia, processamento de cadeias de DNA e outros já está relativamente estabelecido na indústria e na acade-

mia e recebe o nome geral de GPGPU (General-purpose Programming on Graphics Processing Units).

Amada et al. (2004) estiveram entre os primeiros autores a transpor o método SPH para a GPU. A restrição de sua abordagem era que a vizinhança de cada partícula (i.e., sua lista de vizinhos), era construída em CPU e transferida para a GPU. Com o gargalo identificado no passo em CPU e na transferência de memória, os autores apontam ganhos de 2 vezes.

Harada, Koshizuka e Kawaguchi (2007) resolvem o problema da busca de vizinhos utilizando uma textura tridimensional separada em várias texturas bidimensionais para representar um grid de voxels sobre o espaço de simulação. Utilizando máscaras de cores e os buffers de profundidade e stencil os autores conseguem armazenar até quatro partículas por voxel da textura. A vizinhança de uma partícula é definida então pelas partículas presentes nos voxels vizinhos ao voxel que a contém. Os autores reportam ganhos de até 28 vezes (Tabela 3.1).

Número de partículas	Tempo CPU	Tempo GPU	Ganho
1.024	15,6	3,9	4.0x
4.096	43,6	5,45	8.0x
16.386	206,2	14,8	13.9x
65.536	1018,6	58,6	17.3x
262.144	6725,6	235,9	28.5x

Tabela 3.1: Ganhos da implementação em GPU sobre a implementação em CPU, em milisegundos adaptado de (HARADA; KOSHIZUKA; KAWAGUCHI, 2007).

A desvantagem do uso de GPUs é o aumento do tempo médio de implementação, visto que o hardware foi inicialmente projetado para atividades específicas de computação gráfica e não possui necessariamente as facilidades de linguagens de programação de propósito geral. Linguagens tradicionais de shaders como Cg e GLSL utilizam um paradigma que é orientado a vértices de polígonos e texturas que pode ser difícil de adaptar ao problema em ques-

tão. Adicionalmente, não há acesso a recursos externos como discos rígidos, rede, teclado, etc.

De particular dificuldade é o uso de estruturas hierárquicas como as Bounding Trees do Capítulo 2 em conjunto com o sistema de partículas. De fato, os trabalhos anteriormente citados lidam com ambientes simples como caixas e cilindros fechados. Esta é a razão pela qual toda a implementação deste trabalho foi feita em CPU na linguagem C++.

Ainda assim, o caminho da GPU foi inicialmente explorado numa tentativa de acelerar o processamento da simulação e possibilitar um número maior de partículas. A linguagem CUDA (Compute Unified Device Architecture) foi concebida pela empresa NVIDIA como uma tentativa de facilitar a programação em placas gráficas ao expôr uma sintaxe parecida com a linguagem C. De certa forma o uso de CUDA não significa mais GPGPU, uma vez que a linguagem foi concebida para a programação de propósito mais geral e não o caso específico de processamento gráfico. Um protótipo simples em uma caixa fechada foi implementado nesta linguagem CUDA para efeitos de comparação. Apesar dos resultados terem sido encorajadores — a implementação em CUDA leva em média 0.004 segundos para calcular uma iteração em um conjunto de 4000 partículas, enquanto que 1000 partículas levam 0.02 segundos na implementação em CPU — o tempo de implementação fez com que o desenvolvimento se mantivesse em CPU.

A performance deste protótipo é ilustrada no gráfico da Figura 3.12 que mostra a evolução do tempo médio em segundos por iteração ao longo de 500 iterações. Após o estado inicial com as partículas dispostas em um cubo (Figura 3.13a) nota-se um pico inicial que ocorre quando o fluido choca-se contra o piso da caixa (Figura 3.13b). Nesta situação, as partículas aglomeram-se em um volume pequeno de espaço, aumentando a vizinhança de cada uma delas e consequentemente o tempo de processamento sobe. Este aglomerado, juntamente com o choque contra o piso, acarreta uma expulsão das partículas

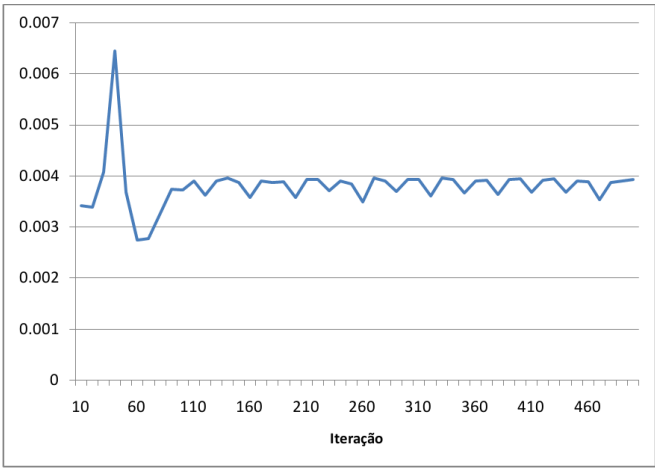


Figura 3.12: Tempo em segundos por iteração para o protótipo na caixa fechada em CUDA.

em todas as direções (o "respingo" ilustrado na Figura 3.13c) numa situação onde elas já não possuem mais tantas vizinhas — é o vale logo após o pico no gráfico. Após isso a simulação entra num estado de estabilidade rapidamente (Figura 3.13d).

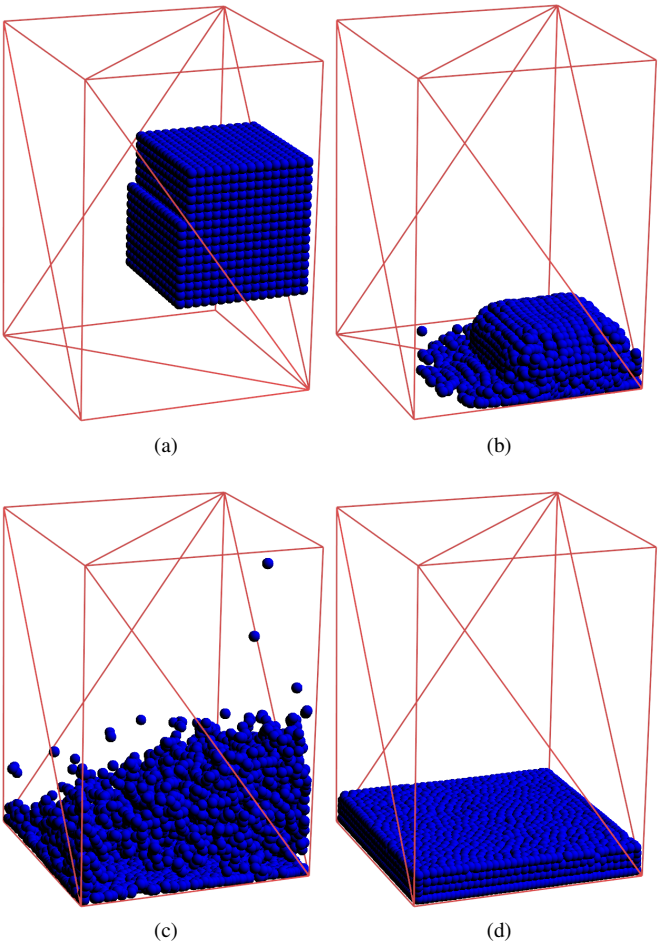


Figura 3.13: Quatro instantes de simulação do protótipo em CUDA.

4 Interação entre Fluidos e Estruturas Tubulares

O foco deste capítulo é unir os dois capítulos anteriores na interação entre fluidos representados computacionalmente por sistemas de partículas e estruturas tubulares representadas por malhas de triângulos. Inicialmente a necessidade desta representação é ilustrada, juntamente com algumas abordagens existentes na literatura, tanto do ponto de vista do realismo quanto da interatividade na Seção 4.1. Após isso, o restante do capítulo é dedicado à explanação do procedimento elaborado para a modelagem da interação que utiliza o Eixo Médio Aproximado discutido no Capítulo 2.

4.1 Necessidade da Simulação da Interação entre Fluidos e Estruturas Tubulares

Um exemplo da importância da representação da interação entre fluidos e estruturas tubulares está no sistema circulatório humano. O sangue e os vasos sanguíneos possuem propriedades químicas e biológicas que geram uma interação complexa, e o uso da Dinâmica dos Fluidos Computacional na simulação de tais fluxos complexos pode prover ao especialista informações necessárias à determinação de como patologias são formadas, como elas evoluem, e como elas podem ser tratadas (DESCHAMPS et al., 2004). Blanco et

al. (2009) afirmam que lesões ateroescleróticas, resultantes da alteração em camadas dos vasos arteriais, estão relacionadas à regiões onde há a presença de pertúbios no fluxo sanguíneo regular. A interação entre o tecido arterial deformável e o fluido sanguíneo é ainda um componente essencial de simuladores cirúrgicos completos (MULLER; SCHIRM; TESCHNER, 2004).

4.1.1 Dinâmica do Sistema Vascular Humano

A contração do músculo cardíaco, o miocárdio, causa um aumento na pressão dentro dos compartimentos do coração e ejeta o líquido sanguíneo para fora do coração através das artérias aorta e pulmonar. Esta contração é denominada sístole, enquanto que o consequente relaxamento dos átrios e ventrículos para a nova recepção de sangue é denominado diástole. Durante a sístole uma onda de alta pressão é transmitida ao longo do sistema vascular, seguida por uma onda de baixa pressão durante a diástole.

Este movimento de contração-expansão é o principal responsável pelo transporte de sangue através do sistema vascular, uma vez que os vasos sanguíneos não participam ativamente neste transporte. Apesar deste transporte por parte dos vasos ser primariamente passivo, as artérias e veias podem regular seus diâmetros internos através de vasoconstrição e vasodilatação do músculo liso presente. Este controle age na termorregulação e na manutenção da pressão arterial média.

4.1.2 Abordagens para a Simulação Computacional do Sistema Vascular

De uma forma geral, as abordagens que trabalham primariamente com grids tendem a focar no realismo e fidelidade anatômica da simulação. Deschamps et al. (2004) adotam um abordagem aonde os elementos da superfície vascular são inicialmente identificados juntamente com uma função sinali-

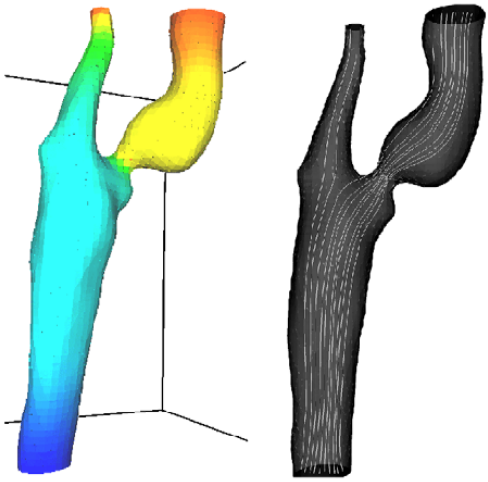


Figura 4.1: Superfície arterial simulada como uma série de elementos embutidos em um grid tridimensional, ilustrando a pressão na superfície (esquerda) e streamlines dentro da carótida (direita) (DESCHAMPS et al., 2004).

zada de distância, que atribui para cada elemento do espaço a sua distância à superfície do vaso. Este conjunto é então embutido em um grid cartesiano regular para identificar as células do grid que intersectam os elementos de superfície.

Este processo é semelhante em idéia ao algoritmo Marching Cubes, entretanto os autores utilizam este grid cartesiano com as informações de superfície diretamente na simulação do sangue (Figura 4.1). Não há menção do tempo de processamento da simulação sanguínea, mas os autores citam que a geração do grid cartesiano leva minutos.

Blanco, Feijóo e Urquiza (2007) utilizam uma abordagem que acopla modelos 3D e 1D para a simulação de grandes regiões do sistema vascular. Neste acoplamento, a maior parte da área de simulação é representada por um modelo simplificado em 1D enquanto que as regiões de interesse específicas, como regiões estenóticas e aneurismas, são representadas em 3D (Figura 4.2).

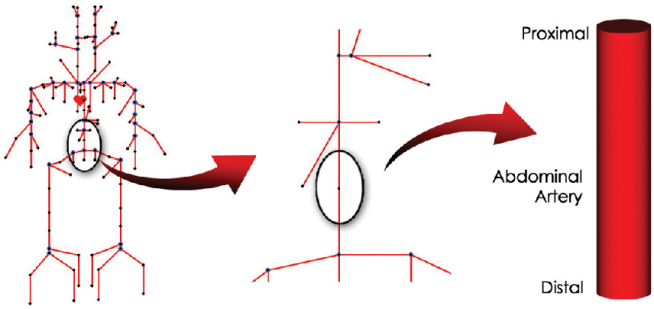


Figura 4.2: Modelagem 3D da região de interesse de um modelo 1D (BLANCO; FEIJóO; URQUIZA, 2007).

O sangue e os vasos são representados por um modelo híbrido denominado *Arbitrary Lagrangian-Eulerian* (HIRT; AMSDEN; COOK, 1974; DONEA; HUERTA; RODRíGUEZ-FERRAN, 2004) que visa atenuar as deficiências tanto da abordagem Lagrangeana, onde os pontos do material são a referência, quanto a abordagem Euleriana, que utiliza os pontos do espaço. No apêndice contido em (BLANCO et al., 2009) o autor cita que a simulação de cada batida cardíaca utilizando 8 CPUs leva entre 20 e 100 horas, dependendo do tamanho do caso de estudo.

Utilizando SPH para a simulação sanguínea e Elementos Finitos para a deformação do vaso, Mueller et al. (2004) implementam a interação entre sólidos deformáveis e fluidos em um simulador cirúrgico virtual que enfatiza os efeitos visuais e a interatividade (Figura 4.3) . De particular interesse é a modelagem da interação entre a parede arterial e as partículas de fluido com o uso de pseudo-partículas nas superfícies dos triângulos da malha. Estas partículas exercem forças sobre as partículas do fluido que modelam tanto repulsão, para a garantia de que as partículas não atravessem a parede arterial, quando adesão, para simular a viscosidade natural que atrai as partículas para as paredes.

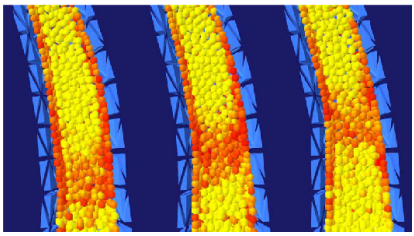


Figura 4.3: Partículas de sangue simuladas com SPH e parede arterial representada por uma malha tetraédrica com Elementos Finitos (MUELLER et al., 2004).

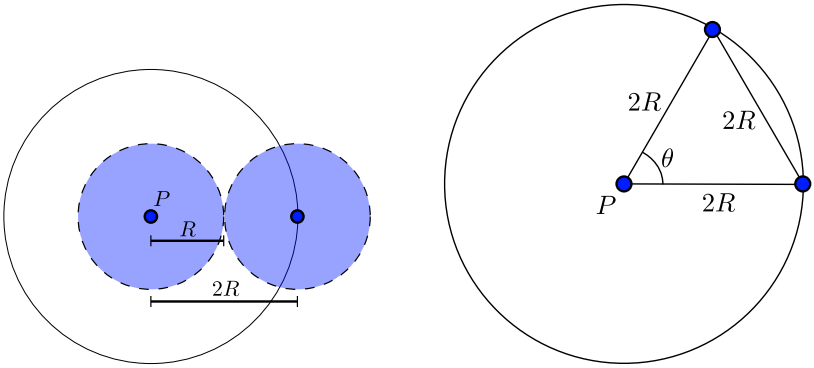
4.2 Força do Eixo Médio Aproximado

Como o foco deste trabalho é a manutenção da interatividade da simulação, aqui a interação entre o fluido e sua estrutura tubular é representada por um processo simplificado e eficiente que utiliza o Eixo Médio Aproximado discutido no Capítulo 2. Este eixo servirá tanto para o posicionamento inicial das partículas ao longo do interior da estrutura quanto para gerar uma força que representará o agente impulsionador do sistema, como o coração do sistema vascular.

4.2.1 Posicionamento Inicial das Partículas

Para o posicionamento inicial das partículas ao longo do Eixo Médio Aproximado adota-se neste trabalho uma estratégia que quebra o problema em unidades menores. De posse do volume de fluido a ser simulado e do número de partículas desejado é possível estimar o raio inicial de cada partícula, R. Este raio vai guiar a distância que cada partícula deve possuir de suas vizinhas imediatas.

Consideraremos, inicialmente, o problema de posicionar partículas ao longo de uma circunferência. Se o raio de cada partícula for R, então as partículas devem ser posicionadas a distâncias $2R$ de suas vizinhas imediatas



(a) As bordas tracejadas representam partículas de raio R. As partículas mais próximas à partícula P e que não a sobrepõem se encontram sobre a circunferência de raio $2R$.

(b) Os centros da partícula P e de duas partículas vizinhas na circunferência formam um triângulo equilátero de forma que $\theta = 60°$.

Figura 4.4: Nas duas figuras os pontos azuis cheios representam o centro de partículas de raio R.

para que não haja sobreposição. O caso mais simples é uma única partícula — neste caso, a circunferência terá raio zero.

A menor circunferência que envolve esta partícula da circunferência de raio zero e mantém a não-sobreposição possui raio $2R$ (Figura 4.4a). Na Figura 4.4b, o centro da circunferência e as duas partículas ilustradas formam um triângulo equilátero de lado $2R$. É possível calcular a posição de partículas nesta nova circunferência utilizando coordenadas polares, bastando encontrar o valor do ângulo θ que para o triângulo equilátero é $60°$.

A próxima circunferência possui raio $4R$, pois ela deve manter a distância de $2R$ da anterior. Desta vez o triângulo formado é isósceles e o ângulo pode ser facilmente derivado da altura, conforme mostra a Figura 4.5. Todas as circunferências seguintes podem ser obtidas incrementando o raio em $2R$ por vez e derivando o ângulo.

O conjunto de partículas posicionadas em circunferências concêntricas

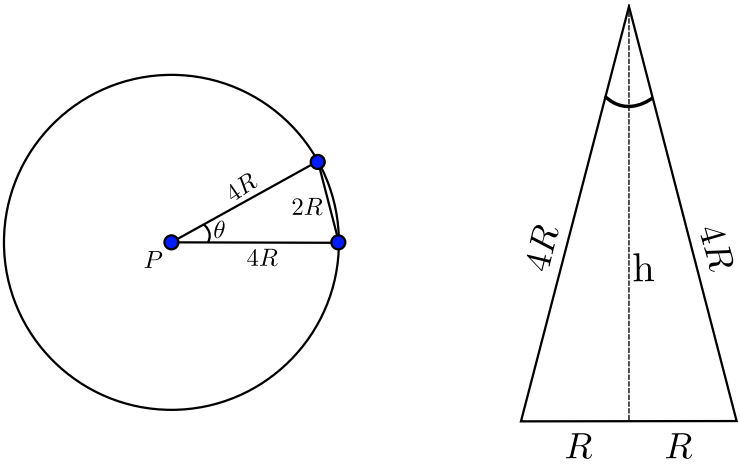


Figura 4.5: Os centros da partícula *P* e de duas partículas vizinhas na circunferência formam um triângulo isósceles com lados 4*R*, 4*R* e 2*R*. O ângulo θ obtido deste triângulo pode ser usado para posicionar novas partículas na circunferência.

que mantém a não-sobreposição (Figura 4.6(A)) é neste trabalho chamado de um *disco* de partículas. Para posicionar uma quantidade qualquer de partículas ao longo de um segmento de reta basta posicionar vários discos em sequência formando um *cilindro* (Figura 4.6(B)). Estes discos mantém a não-sobreposição através do uso da distância 2*R* entre dois discos consecutivos. Finalmente, para posicionar uma quantidade de partículas ao longo de um conjunto de segmentos de reta basta posicionar um cilindro em cada segmento (Figura 4.6(C)).

Manter a não-sobreposição entre dois cilindros consecutivos que possuem orientação arbitrária requer o cálculo da menor distância entre eles. Para simplificar, este trabalho posiciona cilindros de altura menor do que os segmentos de reta correspondentes, como ilustrado na Figura 4.6). Embora esta medida não garanta a sobreposição inter-cilindros, nos experimentos não foi observado nenhum efeito indesejável significativo.

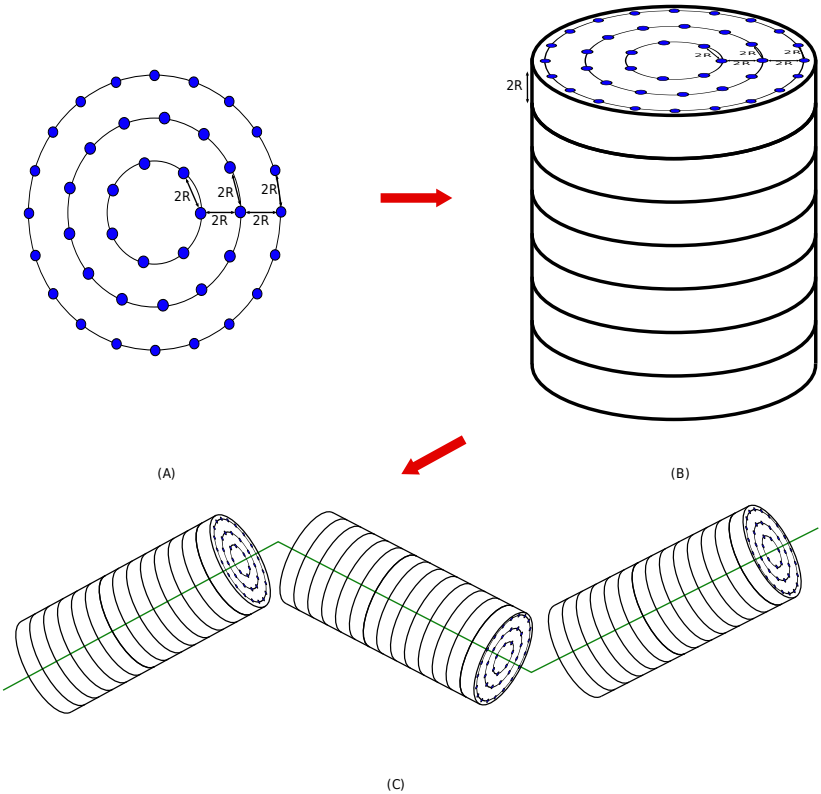


Figura 4.6: As partículas são inicialmente dispostas em circunferências que formam *discos* (A), que empilhados formam *cilindros* (B). Em cada segmento de reta do Eixo Médio Aproximado é posicionado um cilindro de partículas (C).

Um problema decorrente desta estratégia de posicionamento de partículas é que os cilindros seguem o Eixo Médio Aproximado e não respeitam a forma real da malha. A consequência disso é que é possível que partículas sejam criadas no *exterior* da estrutura tubular, pois o cilindro no local é maior do que a estrutura. Esta situação é exemplificada na Figura 4.7. Para tentar mitigar este efeito, dois processamentos adicionais são efetuados em cada disco de partículas.

O primeiro desloca todo o disco de forma a alinhar o seu centro com o centro da seção transversal da estrutura tubular na região (polígono de cor verde na Figura 4.7). Esta medida tende a diminuir o número de partículas fora da estrutura, particularmente em regiões onde o Eixo Médio Aproximado se aproxima da parede da estrutura.

A segunda medida testa cada partícula individualmente para verificar se elas se encontram no interior do polígono da seção transversal. Partículas que se encontram fora da estrutura são então descartadas. A desvantagem de ambas as medidas é que a seção transversal precisa ser calculada para cada disco, aumentando o tempo de processamento. Como este posicionamento ocorre só no início da simulação, o custo é pago somente uma vez.

4.2.2 Evolução e Atualização da Força

O processo de geração do Eixo Médio Aproximado separa a estrutura tubular em questão em uma série de regiões volumétricas delimitadas pelas seções transversais de corte e as paredes da estrutura (são as regiões demarcadas de R_O a R_6 na Figura 4.8). A cada instante de simulação cada partícula sofre a influência da região que a contém. Os parágrafos a seguir explicitam a notação adotada neste texto para definir esta influência.

O resultado do algoritmo de geração do Eixo Médio Aproximado é um conjunto de segmentos de reta, L, e um conjunto de polígonos de seção trans-

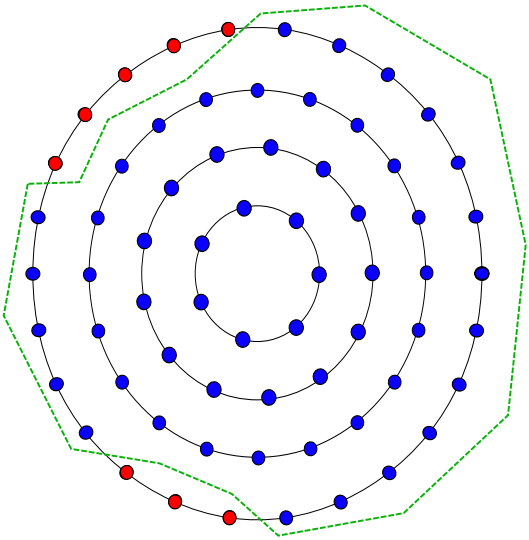


Figura 4.7: Como os discos de partículas não seguem a forma arbitrária da malha, é possível que as partículas criadas se encontrem fora da estrutura. O polígono em verde tracejado representa a seção transversal formada pela intersecção da estrutura com o plano do disco.

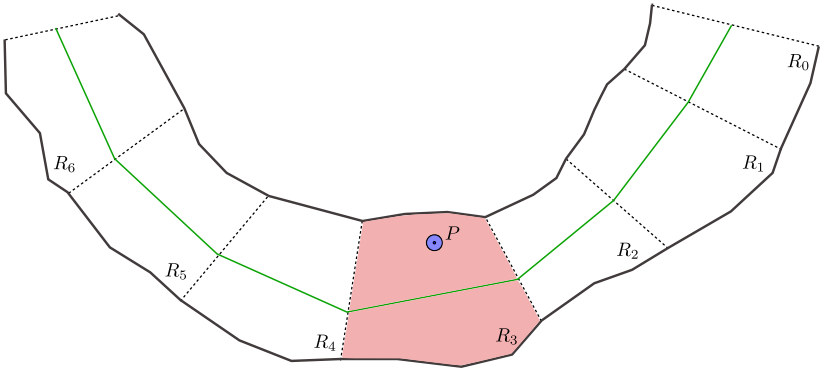


Figura 4.8: Os espaços demarcados $R_{0..6}$ são Regiões, delimitadas pela estrutura tubular e os polígonos de corte.

versal, S. Ambos conjuntos são ordenados, e dois polígonos consecutivos de S delimitam uma região que possui um segmento de reta correspondente em L. Este segmento de reta servirá como uma aproximação da direção do fluxo naquela região.

A cada instante da simulação uma partícula sofre uma força cuja direção é regida pelo segmento de reta da região que contém esta partícula. A tarefa então consiste em identificar a região que contém cada partícula a cada passo da simulação. Isto é feito identificando a região que contém cada partícula na inicialização do sistema, e atualizando este valor ao longo da simulação.

Identificação da Região Inicial

Na inicialização do sistema a região inicial de cada partícula precisa ser identificada e armazenada. Como cada segmento de reta corresponde a um cilindro de partículas tal qual descrito na Seção 4.2.1 poderia-se assumir que a região inicial de uma partícula é a região do segmento de reta do cilindro que a contém. Entretanto, as seções transversais que separam as regiões possuem orientação arbitrária, o que possibilita que algumas partículas do cilindro de uma região estejam de fato localizadas em uma região vizinha. Na Figura 4.9, a região hachurada representa a porção do cilindro da direita que na verdade pertence à região à esquerda do plano.

A forma como a identificação é feita depende de uma característica particular dos conjuntos L e S — O polígono de seção transversal S_i está contido em um plano que, por convenção deste trabalho, possui vetor normal que aponta para o mesmo "lado" que a direção do segmento de reta L_i, ou seja, se $\vec{v}_i$ é o vetor normal do plano e $\vec{l}_i$ é a direção do segmento L_i, então $\vec{v}_i.\vec{l}_i > 0$. A Figura 4.10 ilustra esta condição.

Para localizar a região inicial de uma partícula é necessário então localizar o par de planos consecutivos onde a partícula esteja no lado *positivo* do

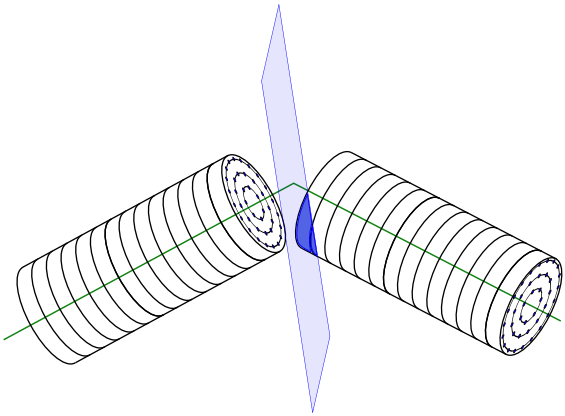


Figura 4.9: Os planos que separam as regiões podem tocar os cilindros de partículas.

primeiro plano e no lado *negativo* do segundo (Figura 4.11), adotando-se a convenção de que o lado positivo de um plano é aquele para o qual o vetor normal do plano aponta.

A vantagem de utilizar planos para a identificação da região inicial é que o algoritmo é simplificado e os cálculos são bastante eficientes. Entretanto um cuidado inicial é necessário e ilustrado na Figura 4.12. Na figura, a estrutura tubular faz uma curva em U, de forma que a condição de planos consecutivos com sinais contrários aplica-se tanto aos planos limítrofes de R_5 quanto aos planos de R_1, uma vez que os planos são infinitos e estendem-se além dos limites internos da estrutura. Para sanar este problema a estratégia adotada neste trabalho consiste em assumir que a região R_i com a menor distância entre P e o centróide de S_i é a correta.

Evolução da Força

A cada passo da simulação, as posições antiga e nova de cada partícula definem a sua *trajetória* durante aquele intervalo de tempo. Checando a in-

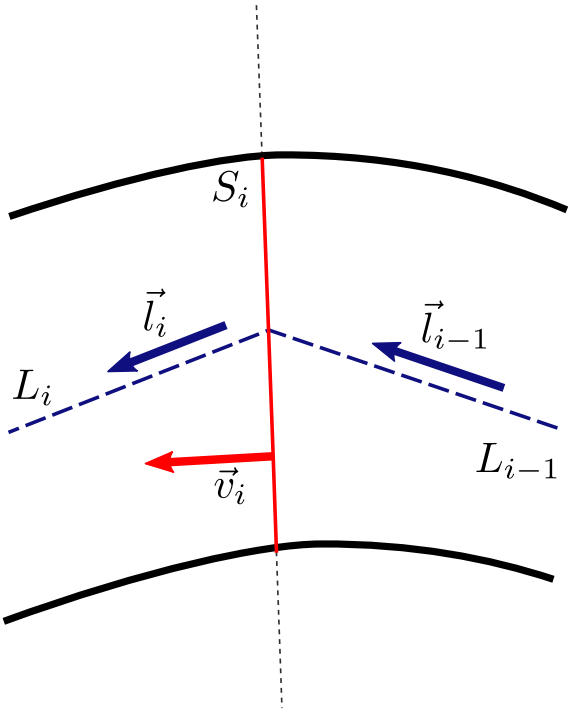


Figura 4.10: Os polígonos de seção transversal e os segmentos de reta são ordenados de forma que $\vec{v}_i$ (em vermelho) aponte para o mesmo "lado" que $\vec{l}_i$ (em azul, à esquerda).

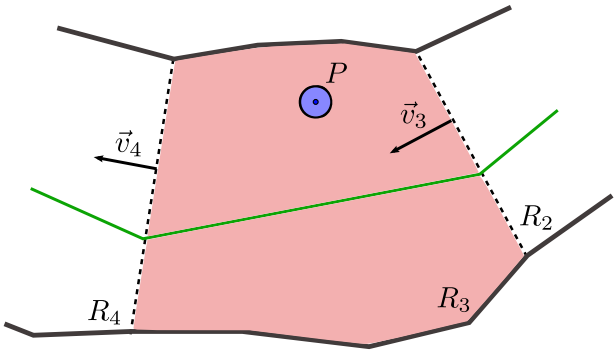


Figura 4.11: A partícula P encontra-se no lado positivo do plano de R_3, e no lado negativo de R_4, estando portanto contida em R_3.

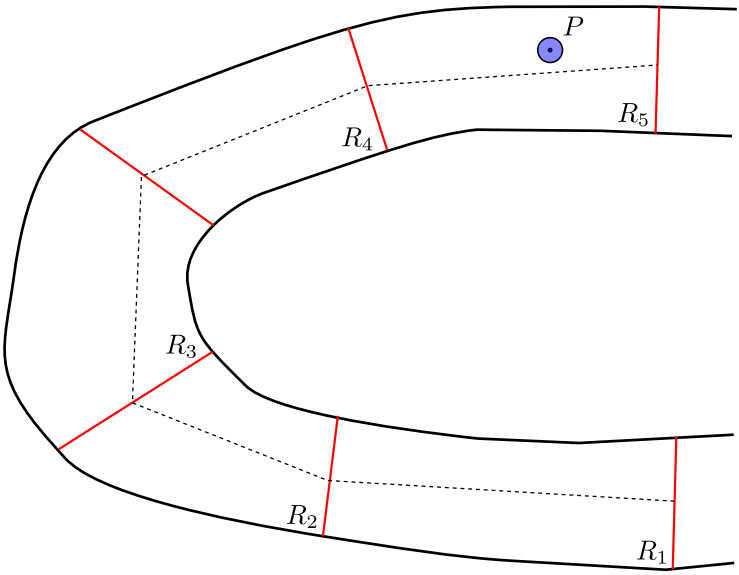


Figura 4.12: A partícula P encontra-se na região R_5, mas pode ser identificada como estando em R_1 também.

tersecção desta trajetória (que é um segmento de reta) com cada polígono do conjunto de seções transversais S é possível verificar se a partícula atravessou a fronteira entre duas regiões durante o passo, ou se continua na região calculada no passo anterior. Se a trajetória de uma partícula atravessou o polígono S_i e partindo do pressuposto que os conjuntos são ordenados de forma que o primeiro vértice do segmento de reta L_i corresponda ao centróide do polígono S_i, então a nova força a agir sobre a partícula terá a direção de L_i.

Adicionalmente, uma vez que os conjuntos são ordenados, é razoável esperar que uma partícula que cruzou o polígono S_i por último irá cruzar o polígono S_{i+1} a seguir, e não qualquer outro polígono, uma vez que a força da região tenderá a impelir a partícula de S_i a S_{i+1}. Assim, ao invés de checar a intersecção da trajetória da partícula com cada polígono de S, bastaria checar apenas S_{i+1}. Na prática, a dinâmica do fluido e a colisão das partículas com a estrutura podem fazer com que esta não siga estritamente a direção de L_i,

podendo se deslocar até no contra-fluxo. Isto significa que a partícula pode voltar e atravessar S_i novamente, mas no sentido contrário. Esta situação precisa ser verificada pois neste caso a nova força a agir sobre a partícula possui a direção de L_{i-1}.

Para exemplificar, considere a situação da Figura 4.13. Nela, a partícula P encontra-se na região R_3 e sobre a influência de uma força que possui a direção de $\vec{l}_3$. Quando sua posição for atualizada durante o próximo passo de simulação três cenários são possíveis:

- A partícula não se desloca o bastante para sair de R_3, de forma que a força sobre ela continua possuindo a direção de $\vec{l}_3$,
- A partícula segue para a esquerda o bastante para cruzar S_4. Agora, sua nova região é R_4 e a força exercida sobre ela possui a direção de $\vec{l}_4$,
- A partícula segue para a direita, cruzando S_3. Agora, sua nova região é R_2 e a força exercida sobre ela possui a direção de $\vec{l}_2$.

Estas três situações possíveis assumem que a partícula não irá se deslocar o suficiente em um único passo de simulação para cruzar mais de um polígono de seção transversal, e de fato esta é uma limitação do método. A solução consiste em verificar os polígonos seguintes mas não foi implementada porque nos casos observados o deslocamento da partícula era sempre muito inferior ao tamanho das regiões.

Como apenas as intersecções da trajetória da partícula com os polígonos S_i e S_{i+1} precisam ser testadas, o processo torna-se eficiente e independente do número de regiões no Eixo Médio Aproximado. Como o conjunto S possui um elemento a mais do que L (uma vez que o último polígono de seção transversal encerra o Eixo Médio Aproximado e não possui segmento de reta correspondente), neste trabalho adotou-se a estratégia de que, quando uma partícula atravessa o último polígono, ela é imediatamente transportada para

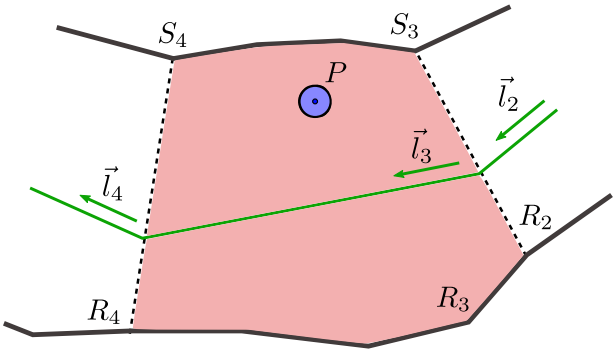
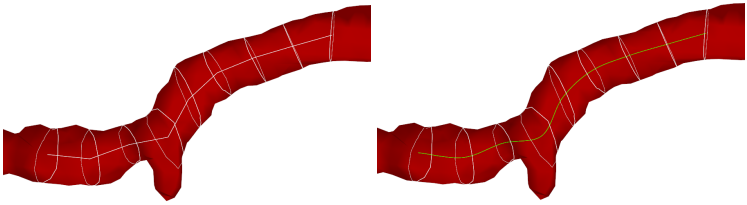


Figura 4.13: A partícula P encontra-se em R_3, podendo cruzar S_4 ou S_3 a seguir.

o primeiro, S_0, e tem sua região devidamente atualizada. Desta forma as partículas mantém-se dentro da região de interesse delimitada pelo Eixo Médio Aproximado.



(a) (b)

Figura 4.14: Eixo Médio Aproximado composto por segmentos de reta (esquerda) e um curva de Catmull-Rom (direita).

4.2.3 Suavidade do Eixo Médio Aproximado

A utilização de segmentos de reta para o Eixo Médio Aproximado gera um campo de forças com descontinuidades na primeira derivada. Isto é facilmente observável nas fronteiras entre regiões, onde a força varia bruscamente de direção. Estas descontinuidades geram artefatos e reduzem a estabilidade da simulação (MUELLER et al., 2004). Intuitivamente, é possível verificar a redução na estabilidade pela inércia gerada na partícula enquanto esta atravessa uma região. Ao cruzar a fronteira para uma nova região, a partícula tenderá a manter a direção anterior utilizada em vários passos consecutivos (Figura 4.10).

Utilizando os centróides dos polígonos de seção transversal é possível gerar uma curva que suavize o problema da descontinuidade. Neste trabalho adotam-se curvas de Catmull-Rom (CATMULL; ROM, 1974), que interpolam todos os pontos de controle e são utilizadas em automatização de trajetória de câmeras virtuais (FOLEY et al., 1996). A Figura 4.14 ilustra o Eixo Médio aproximado por uma curva em contraste ao Eixo Médio composto por segmentos.

Evolução da Força

O procedimento para a identificação e evolução da força que agirá sobre as partículas em função do Eixo Médio Aproximado suavizado é similar ao descrito na seção 4.2.2. A região inicial de cada partícula ainda é identificada pela sua distância aos planos dos polígonos de seção transversal e as regiões são atualizadas conforme a partícula se desloca e atravessa os planos limítrofes da região (Figura 4.15. O que difere é a forma como a força é calculada dentro de uma região.

No Eixo Médio Aproximado composto por segmentos de reta, a força dentro de uma região possui a direção do segmento de reta daquela região. Como o Eixo agora é uma curva, não existe um segmento de reta único e a partícula deveria, a cada instante, tender a seguir a curva.

Um valor que pode ser utilizado para a direção da força é o vetor tangente da curva em seu ponto mais próximo da partícula. Segundo Haines e Akenine-Moller (2002), o vetor tangente de uma curva de Catmull-Rom em um ponto P_i é simplesmente $P_{i+1} - P_{i-1}$, ou seja, o vetor que vai do ponto anterior a P_i ao seu ponto seguinte. A Figura 4.16 exemplifica esta noção para uma partícula, enquanto a Figura 4.17 ilustra a direção da força agindo sobre partículas situadas em vários pontos diferentes do espaço.

4.3 Detecção de Colisão

O segundo componente da interação entre o fluido e a estrutura tubular que o contém é o tratamento de colisão, que impede que as partículas atravessem as paredes da estrutura de maneira não realista. É fácil constatar que este problema é não-trivial quando se considera que o fluido é composto por milhares de partículas que estão contidas dentro de uma estrutura com potencialmente milhares de triângulos. O capítulo 2 abordou a representação de

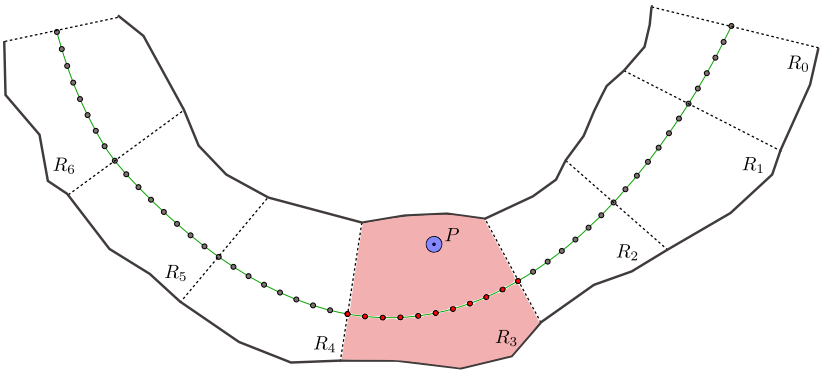


Figura 4.15: As mesmas regiões da Figura 4.8, agora contendo os pontos da curva.

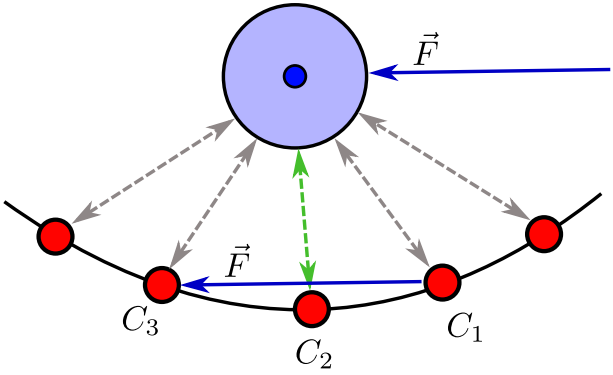


Figura 4.16: C_2 é o ponto da curva mais próximo da partícula, de forma que o vetor $\vec{F}$ da força a ser aplicada possui a direção $C_3 - C_1$, que aproxima a tangente da curva em C_2.

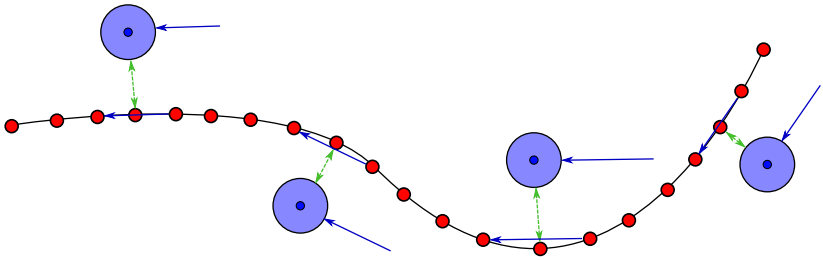


Figura 4.17: Exemplos da evolução da direção da força aplicada sobre as partículas ao longo da trajetória da estrutura.

estruturas tubulares e uma forma de acelerar testes de intersecção através do uso de árvores para subdivisão espacial. Estas árvores foram então utilizadas no processo de geração do Eixo Médio Aproximado.

Estas mesmas árvores podem agora ser usadas para acelerar o processo de detecção de colisão entre uma partícula de fluido e uma malha de triângulos. De fato, se considerarmos que as posições de uma partícula antes e depois da integração da velocidade e da atualização da posição compõem um segmento de reta que representa sua trajetória ao longo da iteração de simulação, o algoritmo 1 na página 36 pode ser utilizado com pouquíssimas modificações. A idéia ainda é descer na hierarquia da árvore para encontrar a lista de triângulos que intercepta a trajetória da partícula e depois encontrar nesta lista qual o triângulo que intercepta a trajetória primeiro.

O problema é que agora, a "trajetória" não é mais um segmento de reta mas sim uma cápsula (um cilindro com pontas esféricas), conforme ilustra a Figura 4.19a. Testar a intersecção entre uma cápsula e uma caixa ou um triângulo é mais difícil computacionalmente. Felizmente o problema pode ser simplificado para que o teste seja efetuado com segmentos de reta novamente.

Ericson (2004) descreve um teste de intersecção entre uma esfera que se move e uma caixa alinhada que utiliza a soma de Minkowski. A soma de Minkowski de duas formas geométricas é a forma resultante da adição

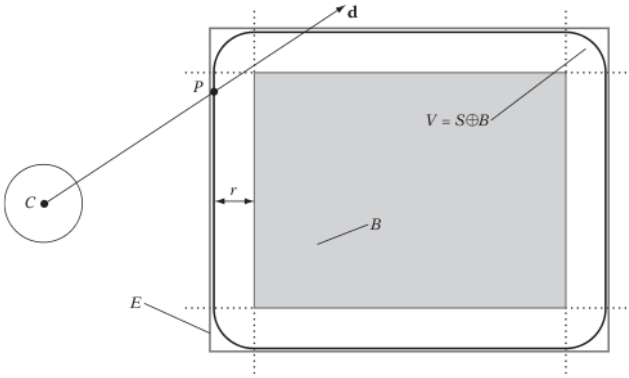
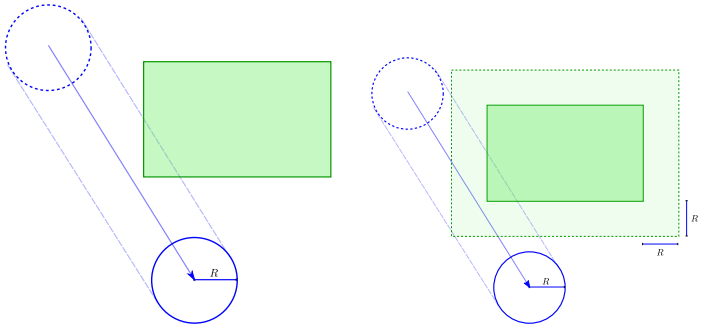


Figura 4.18: Soma de Minkowski entre a caixa e a esfera no teste de intersecção (ERICSON, 2004).

de todos os pontos da primeira forma na segunda forma. No caso de uma esfera e uma caixa alinhada o resultado é uma caixa alinhada com as quinas arredondadas, como ilustra a Figura 4.18. O autor efetua então o teste entre esta nova forma e o segmento de reta da trajetória do centro da partícula.

No caso do teste de intersecção com a caixa alinhada, o objetivo do uso da árvore é um rápido descarte de intersecções impossíveis — falsos-positivos são toleráveis porque ao final da hierarquia os triângulos nos nodos folha ainda serão testados. Ao invés de efetuar o teste de intersecção entre uma caixa e uma cápsula representativa da trajetória de uma partícula de raio R, o teste é feito entre o segmento de reta que representa a trajetória do centro da partícula e uma caixa aumentada em R unidades em todas as dimensões (Figura 4.19b). Esta caixa é a menor caixa alinhada que contém a soma de Minkowski entre as duas formas, de maneira que os falso-positivos ocorrem apenas nas quinas da caixa.

Um teste de intersecção robusto entre a trajetória de uma partícula e um triângulo é computacionalmente caro (ERICSON, 2004) e requer testes individuais com as arestas do triângulo e possivelmente com esferas centradas nos vértices do mesmo. Este trabalho adota uma abordagem simplificada —



(a) (b)

Figura 4.19: A caixa de teste é aumentada em todas as direções pelo raio da esfera e utilizada no teste de intersecção.

seja a posição inicial da partícula C, seu raio R e o vetor de direção de sua trajetória $\vec{v}$. Se C está disposto de forma a se encontrar do lado *positivo* do plano que contém o triângulo de teste, então o ponto D da partícula que irá tocar o plano antes de qualquer outro é dado por $D = C - R\vec{n}$, onde $\vec{n}$ é o vetor normal do plano (Figura 4.20).

O teste é realizado então entre o plano e um segmento de reta com origem em D e mesma direção $\vec{v}$. Há intersecção caso o ponto resultante deste teste se encontre dentro do triângulo. Este método pode falhar exatamente quando D for o primeiro ponto a tocar o plano, mas não o triângulo, como mostra a Figura 4.21. Como a solução completa deste problema introduz testes adicionais custosos, neste trabalho o triângulo original é expandido por um fator constante de forma que D seja de fato o primeiro ponto a colidir.

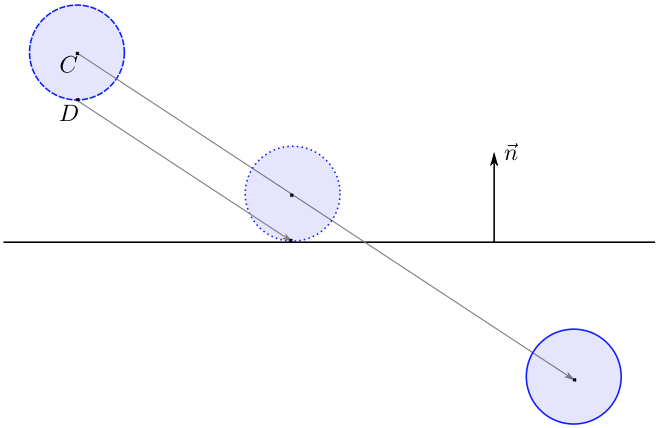


Figura 4.20: O ponto *D* é o primeiro ponto da esfera a tocar o plano com vetor normal $\vec{n}$.

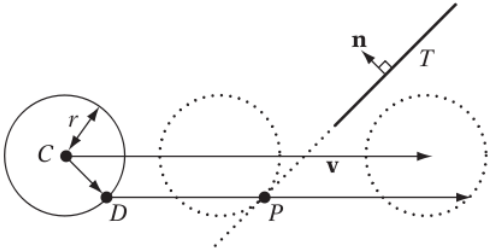


Figura 4.21: Fraqueza do teste de intersecção utilizando apenas a trajetória do ponto *D*(ERICSON, 2004).

5 *Visualização de Resultados*

5.1 Visualização via Rampas de Cor

A função das rampas de cores é mapear cores a um intervalo de valores escalares. Uma rampa comumente usada é a rampa que interpola entre cores "frias" e "quentes". Dado um intervalo de valores, valores relativamente baixos são representados por azul, valores altos são vermelhos e os valores intermediários são verdes. Para suavizar a transição entre as extremidades e o meio, as cores ciano (azul + verde) e amarelo (verde + vermelho) são adicionadas (Figura 5.2).

Para ilustrar, considere as duas séries de imagens a seguir. A primeira (Figura 5.3) mostra três diferentes representações do mesmo instante de simulação na região da artéria que possui um alargamento, com o fluido seguindo da direita para a esquerda da região.

A Figura 5.3a mostra todas as partículas renderizadas com a mesma cor sólida (azul). Na Figura 5.3b, as partículas são coloridas a partir de uma escala de cor do valor de sua *pressão.* Como a pressão que uma partícula

Figura 5.1: Quatro faixas sólidas de cores.

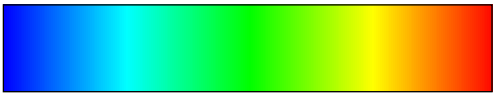


Figura 5.2: Rampa quente-frio com 5 cores.

sofre está diretamente relacionada à sua densidade e quantidade de partículas em sua vizinhança imediata (vide o Capítulo 3), o resultado é que as partículas que caem no alargamento ficam “espremidas” pelo fluxo e recebem coloração avermelhada, representando uma região cuja pressão é visivelmente maior do que no restante do espaço de simulação.

O tráfego de partículas próximas ao alargamento possui um impacto direto em suas *velocidades*, como mostra a Figura 5.3c. As partículas que estão sobre alta pressão estão quase estacionárias e são coloridas de azul. O fluxo das partículas mais à direita do espaço é avermelhado porque estas são as partículas que acabaram de entrar no sistema e ainda possuem um grau de liberdade de movimento relativamente alto. Entretanto, conforme as partículas se aproximam da região com o acúmulo suas velocidades diminuem gradativamente para amarelo, verde, e azul.

A Figura 5.4 ilustra uma situação diferente. Cada sub-figura mostra o estado da simulação de uma quantidade de fluido em diferentes pontos no tempo. A força do Eixo Médio Aproximado é exercida da direita-embaixo para a esquerda-acima. Na Figura 5.4a, as partículas acabam de chegar no início de uma ladeira para cima, vindo de uma região relativamente reta que permitiu que as partículas adquirissem considerável velocidade e momento. Como resultado, as partículas se chocam diretamente com a parede da artéria, e aquelas que se encontram mais à frente do conjunto são esmagadas por suas vizinhas de trás, resultando novamente em uma região de maior pressão.

Após os choque inicial, a força do Eixo Médio Aproximado começa a mover as partículas para cima, seguindo o fluxo, como mostra a Figura 5.4b. Adiante na artéria há um afunilamento na parede que delimita a vazão,

resultando em uma nova região de alta pressão (Figura 5.4c).

5.2 Avaliação de Performance

5.2.1 Passos de Simulação

Cada iteração de simulação é composta por uma série de passos distintos. Cada passo possui sua complexidade particular e a análise do custo de cada passo permitirá identificar os gargalos do sistema. Inicialmente cada passo será apresentado com referências ao capítulo em que é abordado neste texto.

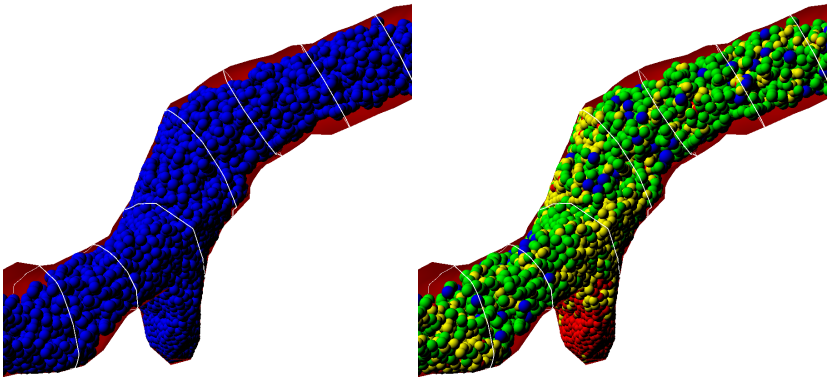
Inicialização do Sistema

A inicialização do sistema requer a estrutura tubular já representada por uma Bounding Tree, o Eixo Médio Aproximado gerado por dois planos e os parâmetros do sistema como número de partículas desejado, o volume de fluido a representar, os coeficientes de pressão e viscosidade do material simulado, etc.

As partículas são criadas e posicionadas ao longo do Eixo Médio Aproximado de acordo com o algoritmo baseado em cilindros descrito na Seção 4.2.1 na página 77. Adicionalmente, a região inicial de cada partícula em relação ao Eixo Médio Aproximado é calculada e armazenada. O grid para o cálculo da vizinhança de cada partícula é criado, inicialmente vazio.

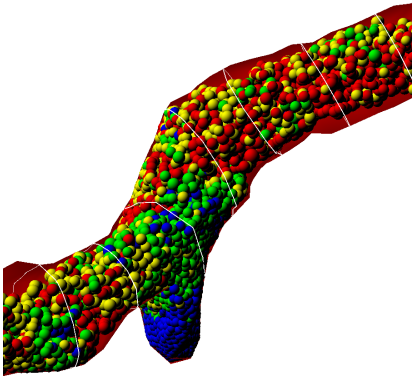
Busca de Vizinhança

Neste passo as partículas são todas inseridas em um grid tridimensional onde as células são cubos com lado igual ao tamanho do raio de suporte dos núcleos de suavização utilizados, h (vide Seção 3.3.3 na página 67). A posição de cada partícula é guardada para que depois seja possível removê-las do



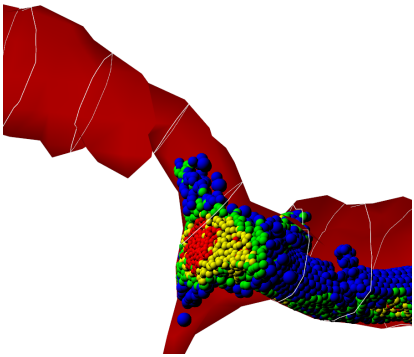
(a) Cor fixa.

(b) Cor indicativa da pressão.

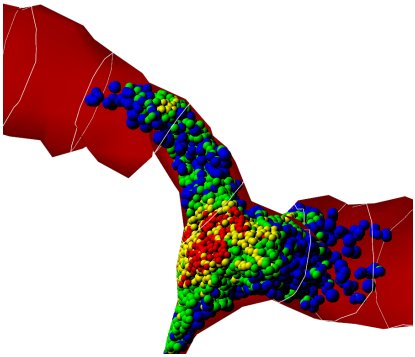


(c) Cor indicativa da velocidade.

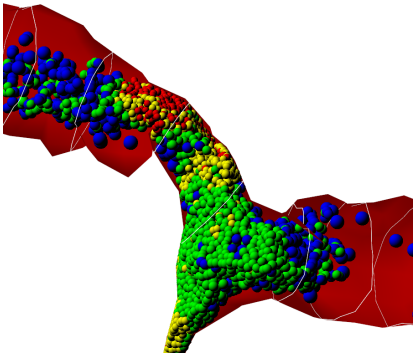
Figura 5.3: O mesmo instante de simulação, com três representações diferentes.



(a) Partículas chegam na inclinação.



(b) Partículas são empurradas para cima pela força do Eixo Médio Aproximado.



(c) Partículas aglutinam-se no afunilamento.

Figura 5.4: Três momentos da simulação.

grid, ao final do passo de simulação.

Após a inserção os vizinhos de cada partícula são definidos. Os candidatos à vizinhança de uma partícula P estão todos contidos na célula de P ou nas células imediatamente adjacentes, uma vez que os núcleos de suavização são todos nulos para valores de distância acima de h. Assim, para cada partícula P_j candidata em potencial à vizinhança de P a propriedade $|P_j - P| \leq h$ é verificada.

Uma vez que o cálculo das forças agindo em cada partícula depende da densidade e da pressão de suas vizinhas, estes valores são cálculados também neste passo, seguindo diretamente as fórmulas da Seções 3.3.2 na página 56 e 3.3.2 na página 56.

Cálculo das Forças do Fluido

Após a definição das vizinhanças é possível calcular a força que age sobre cada partícula em função de suas vizinhas. A contribuição de cada partícula vizinha de P sobre as forças de pressão, viscosidade e superfície (Seções 3.3.2 na página 59, 3.3.2 na página 60 e 3.3.2 na página 62, respectivamente) é calculada diretamente com as equações e acumulada em uma força total. Apesar da força de superfície ser discutida no Capítulo 3 na página 39 como uma força externa, ela é calculada também neste passo por depender apenas da vizinhança de cada partícula.

Cálculo das Forças Externas

Neste passo as forças externas são acumuladas à força total obtida até agora para cada partícula. Aqui há a ação da força do Eixo Médio Aproximado exatamente da forma descrita nas Seções 4.2.2 na página 81 e 4.2.3 na página 89. A força de gravidade é também adicionada aqui, caso esteja sendo utilizada.

Integração Temporal

Aqui a aceleração, velocidade e posição de cada partícula são atualizadas de acordo com a força total acumulada e o esquema Leap-Frog para integração (Seção 3.3.3 na página 65).

Detecção e Tratamento de Colisão

Potencialmente um dos passos mais custosos de simulação é a detecção de colisão entre as partículas e a estrutura tubular. A trajetória de cada partícula, definida por sua posição no início da iteração de simulação e sua nova posição obtida no passo de Integração Temporal, é testada contra a Bounding Tree que encapsula os triângulos da malha. A detecção e o tratamento são contemplados de forma geral no Capítulo 2 na página 23 e no caso específico do fluido com a estrutura tubular na Seção 4.3 na página 90.

Renderização

Finalmente, o sistema todo é renderizado e mostrado ao usuário. A renderização pode ser feita através de segmentos de reta que representam a posição da partícula e a força do Eixo Médio Aproxima agindo sobre ela (mais rápido), ou através de esferas coloridas de acordo com alguma grandeza de interesse (mais lento).

No caso da renderização de esferas sobre grandezas, os limites máximo e mínimo da grandeza em questão são obtidos percorrendo o conjunto inteiro de partículas, e então a cada partícula é atribuída uma cor de acordo com o valor de sua grandeza no intervalo definido. Este método é mais custoso porque requer uma travessia adicional sobre o conjunto de partículas e os cálculos do intervalo.

5.2.2 Protótipo para Visualização da Pressão

Este experimento visa avaliar a performance do caso apresentado na Figura 5.4. As partículas fluem da direita para a esquerda, e o tempo gasto por cada passo é medido ao longo de 200 iterações, que mostrou-se um número razoável para alcançar alguma estabilidade na simulação.

Parâmetros de Simulação

O segmento arterial foi reconstruído a partir de 20 imagens de tomografia com espaçamento de pixel de 0.6 milímetros e distância entre imagens 2.5 milímetros usando o algoritmo Marching Cubes (Seção 2.1 na página 24) com cubos de lado 3.0 milímetros. A malha resultante possui 1906 faces. A simulação foi executada numa máquina AMD Athlon X2 4400+ com 2GB de memória RAM rodando Gentoo Linux com kernel versão 2.6.26-3. A placa de vídeo utilizada é uma NVIDIA GeForce GTS 150 com driver versão 177.82, e o código é todo escrito em C++ com o compilador GCC versão 4.3.4. Para explorar as duas CPUs presentes, em cada passo da simulação as partículas são distribuídas para as threads disponíveis, de forma que a aplicação toda é multithreaded (com exceção da Renderização).

Talvez a parte mais difícil da simulação seja a definição de parâmetros corretos para os vários atributos do fluido. Geralmente estes atributos precisam ser ajustados manualmente até que o resultado seja visualmente satisfatório, e como eles são numerosos e tendem a influenciar um no outro, este ajuste pode levar tempo. Os parâmetros utilizados e que não variam entre as diferentes simulações são listados na Tabela 5.1.

4000 Partículas

O primeiro teste foi realizado com 4000 partículas, com parâmetros adicionais estabelecidos como mostra a Tabela 5.2. O tempo de processamento

Atributo	Símbolo	Valor
Densidade de Repouso	ρ_0	998.29
Viscosidade	μ	3.5
Tensão de Superfície	σ	1.0
Coeficiente de Pressão	k	6.0
Partículas na Vizinhança	–	20
Número de Regiões do Eixo	–	12
Número de Pontos na Curva	–	100

Tabela 5.1: Parâmetros comuns a todas as simulações.

médio de cada 10 iterações foi calculado para um total de 200 iterações e é mostrado na Tabela A.1 na página 128. Conforme o gráfico da Figura 5.5 ilustra, as iterações com o menor tempo de processamento ocorrem logo no início, possivelmente por causa da distribuição semi-regular das partículas pelo algoritmo de posicionamento inicial de partículas do Capítulo 4.

Atributo	Símbolo	Valor
Número de Partículas	–	4000
Volume de Fluido	V	0.16
Massa por Partícula	m	0.04
Raio de Suporte	h	0.057

Tabela 5.2: Parâmetros para a simulação da pressão com 4000 partículas.

Ainda observando o gráfico, o passo que em média consome a maior parte do tempo é o passo de Busca de Vizinhança, cuja média ao longo do período é de 0.04375 segundos por iteração, seguido de perto pelo passo de Colisão, com média de 0.03585 segundos. Após estes passos, os mais demorados em ordem são os passos de Forças Internas do Fluidos, Forças Externas, a Renderização e finalmente a Integração, que é o passo mais rápido levando em média 0.00068 segundos por iteração e variando muito pouco ao longo do período. A simulação ocorre a uma média de quase 10 iterações por segundo.

Os passos de Busca de Vizinhança e Colisão tendem a variar mais em função da configuração do fluido no momento, e a evolução destes dois pas-

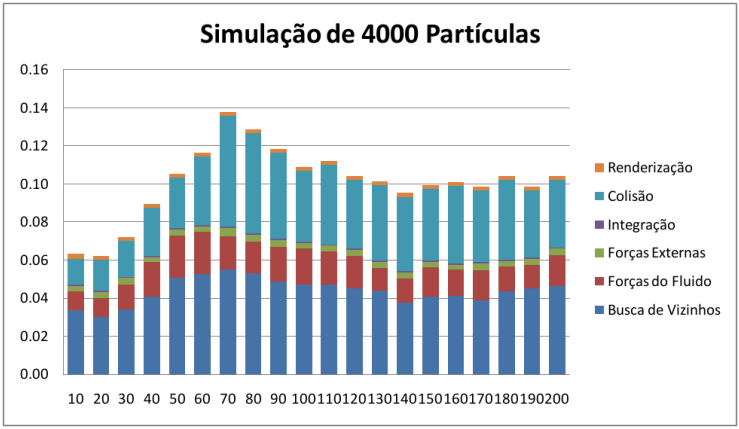


Figura 5.5: Gráfico de barras empilhadas para a simulação com 4000 partículas.

sos é mostrada no gráfico da Figura 5.6. A Colisão tende a levar menos tempo do que a Busca de Vizinhança, e esta situação se inverte pela primeira vez em torno da iteração de número 70, que é também o pico no tempo de processamento total. A Figura 5.7 mostra o estado da simulação no instante das iterações 50, 70, 100 e 150. Nos quatro instantes a simulação apresenta um aglutinamento de partículas na região onde a subida se inicia, entretanto o tempo de simulação total da iteração 70, 0.13776 segundos, é consideravelmente maior e se destaca no gráfico.

A razão para tal é possível de ser vista pela renderização da pressão. Em todos os quatro instantes há regiões de alta pressão avermelhadas. Estas regiões de alta pressão causam um aumento em muitos dos passos de simulação — maior pressão significa maior quantidade de partículas no local, o que causa mais vizinhos candidatos e efetivos por partícula, e isto aumenta o tempo de Busca de Vizinhança. Mais vizinhos efetivos aumenta o tempo do cálculo das Forças Internas do fluido, que é um somatório sobre os vizinhos. E se a região de maior pressão for próxima à parede da artéria, são mais partículas que vão acabar caindo no passo de Tratamento de Colisão. A figura da

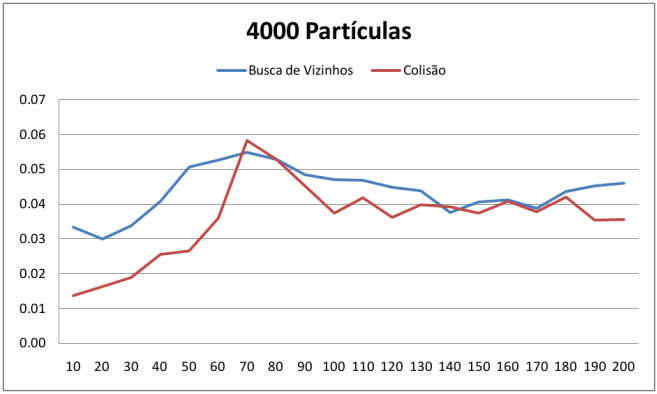


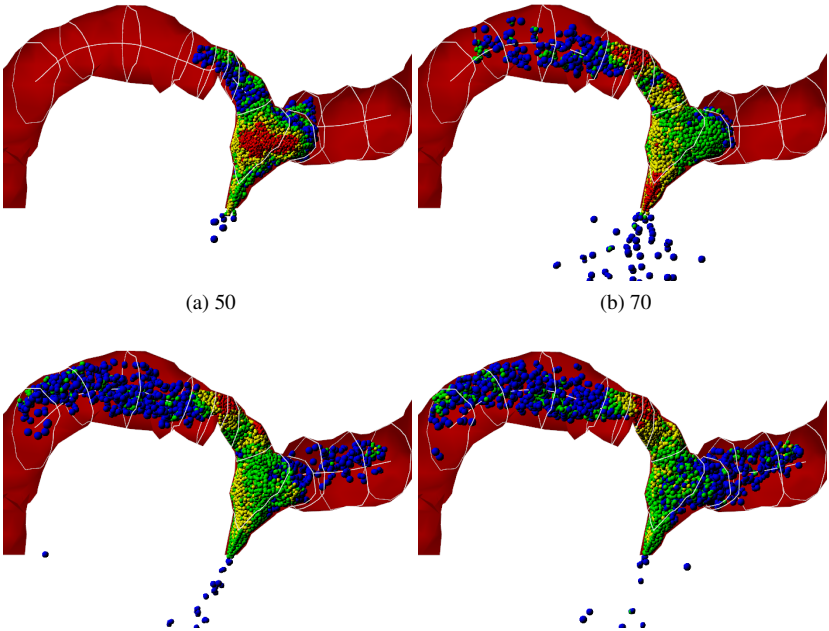
Figura 5.6: Tempo de processamento Busca de Vizinhos x Colisão, 4000 partículas.

iteração 70 mostra duas regiões de pressão alta — os afunilamentos na parte superior da artéria e na parte inferior, onde a reconstrução gerou uma malha aberta propositalmente.

1000 Partículas

Na simulação com 1000 partículas adotou-se 0.08 para o volume total de fluido, pois o valor anterior de 0.16 deixava as partículas grandes demais, o que era visualmente desinteressante. O restante dos parâmetros diferentes são ilustrados na Tabela 5.3, enquanto que o gráfico da Figura 5.8 mostra novamente o tempo médio por grupo de 10 iterações, para um total de 200 iterações. A simulação roda em uma média de 36 iterações por segundo, e os tempos todos estão na Tabela A.2 na página 129.

Nota-se novamente que o processamento acontece mais rapidamente no início da simulação, mas ao contrário do caso com 4000 partículas agora o passo que domina o tempo de processamento é o de Colisão. De fato, o tempo médio para este passo é de 0.01458 segundos, enquanto que o segundo colocado, a Busca de Vizinhança, leva em média 0.00749, quase a metade.



(a) 50 (b) 70

(c) 100 (d) 150

Figura 5.7: Snapshots da simulação com 4000 partículas das iterações 50, 70, 100 e 150.

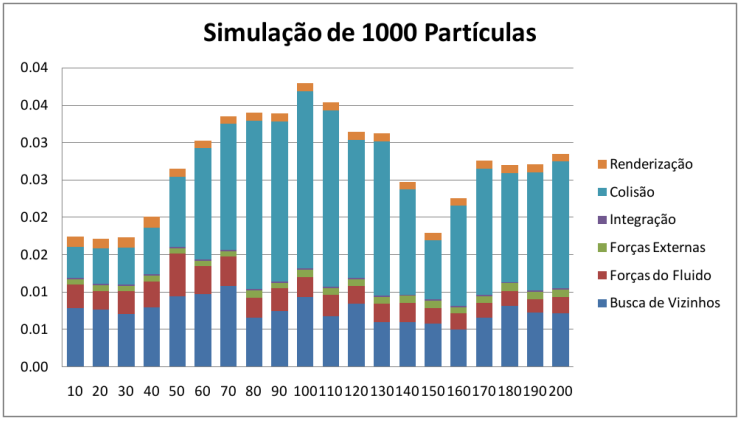


Figura 5.8: Gráfico de barras empilhadas para a simulação com 1000 partículas.

O cálculo das Forças Externas e a Integração são novamente os passos mais rápidos, ambos levando menos de um milissegundo para terminarem.

Atributo	Símbolo	Valor
Número de Partículas	–	1000
Volume de Fluido	V	0.08
Massa por Partícula	m	0.08
Raio de Suporte	h	0.43

Tabela 5.3: Parâmetros para a simulação da pressão com 1000 partículas.

O grande culpado do custo agora é o passo de Colisão. O gráfico da Figura 5.9 ilustra a variação deste passo, contrastado com o segundo colocado, a Busca de Vizinhos. A variação no tempo da Colisão é tanta que para entendê-la ilustra-se novamente alguns pontos da simulação, na Figura 5.10.

Os maiores fatores que influenciam no tempo que o passo de Colisão vai levar são o tamanho da trajetória das partículas, e a quantidade das partículas que de fato tocam triângulos. O tamanho da trajetória influi na largura da travessia da Bounding Tree, pois mais nodos (caixas alinhadas) serão tocados. As partículas que tocam triângulos, mesmo que com trajetórias pequenas,

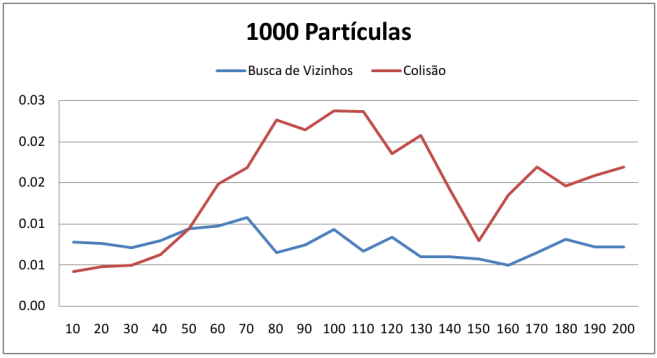


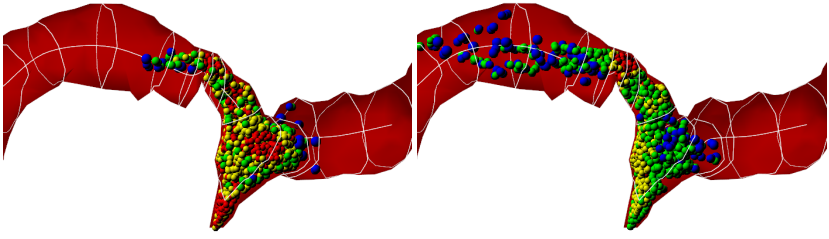
Figura 5.9: Tempo de processamento Busca de Vizinhos x Colisão, 1000 partículas.

requerem a chamada do procedimento de Tratamento de Colisão, descrito no Capítulo 4, que é muito mais custoso do que apenas a Detecção.

Assim, se há poucas partículas tocando as paredes da artéria, a travessia na Bounding Tree vai terminar logo porque os nodos mais profundos encapsulam os triângulos que compõem a parede. Em contrapartida, muitas partículas "coladas" na parede requerem uma travessia longa e testes contra triângulos. Na Figura 5.10, no instante da iteração 150 há poucas partículas de fato tocando as paredes (lembrando que a parede que cobriria as partículas não é renderizada, mas existe) — há muitas partículas reentrando no sistema pela direita (partículas azuis) e definitivamente menos partículas no afunilamente inferior, onde todas tocam a parede arterial. Nos outros casos da Figura, há mais partículas no agrupamento central, nos afunilamentos, etc.

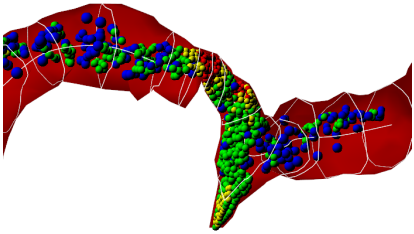
8000 Partículas

A configuração da simulação para 8000 partículas é marcadamente similar ao caso com 4000. O volume é o mesmo, o que faz com que as partículas sejam menores em tamanho. A Tabela 5.4 mostra os parâmetros usados, e a Tabela A.3 na página 130 os tempos totais. O gráfico da Figura 5.11 lista a

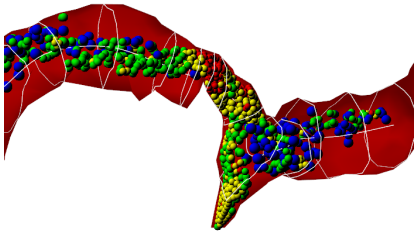


(a) 60

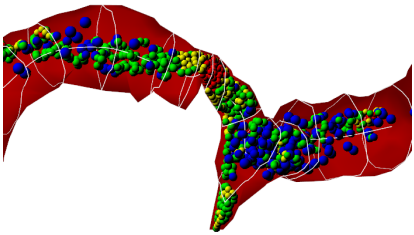
(b) 80



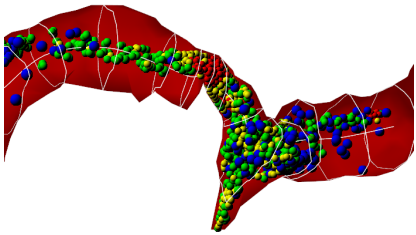
(c) 110



(d) 120



(e) 150



(f) 180

Figura 5.10: Snapshots da simulação com 1000 partículas das iterações 60, 80, 110, 120, 150 e 180.

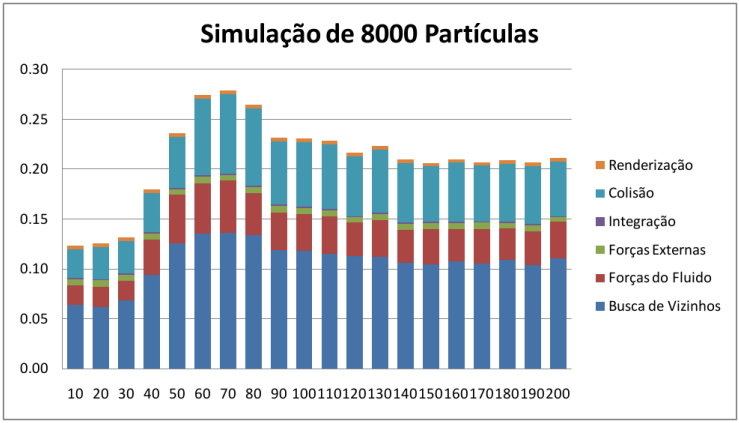


Figura 5.11: Gráfico de barras empilhadas para a simulação com 1000 partículas.

evolução dos passos ao longo do tempo. A simulação roda a uma média de 4.7 iterações por segundo.

Atributo	Símbolo	Valor
Número de Partículas	–	8000
Volume de Fluido	V	0.16
Massa por Partícula	m	0.02
Raio de Suporte	h	0.045

Tabela 5.4: Parâmetros para a simulação da pressão com 8000 partículas.

Agora, o passo de Busca de Vizinhos domina novamente, seguido pela Colisão. A situação se inverte em relação ao caso com 1000 partículas: A Busca de Vizinhos, com média de 0.10705 leva quase o dobro do tempo da Colisão, a 0.05682 segundos por iteração. A evolução dos dois passos é marcadamente similar (Figura 5.12). Alguns quadros da simulação são ilustrados na Figura 5.13.

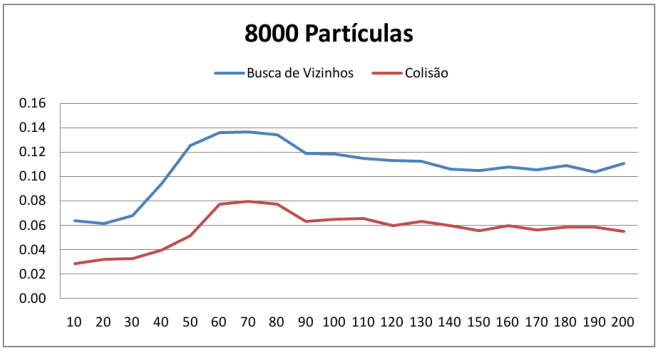
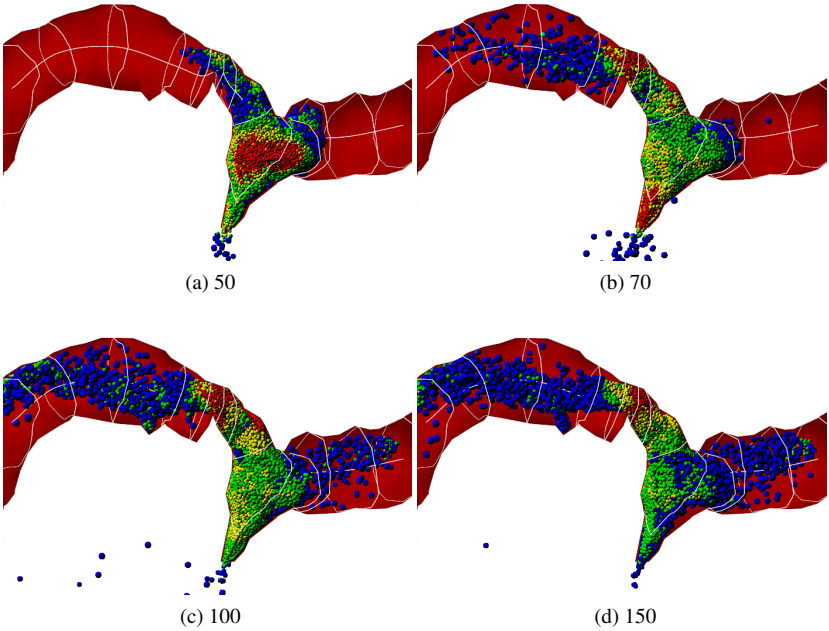


Figura 5.12: Tempo de processamento Busca de Vizinhos x Colisão, 1000 partículas.



(a) 50 (b) 70

(c) 100 (d) 150

Figura 5.13: Snapshots da simulação com 8000 partículas das iterações 50, 70, 100 e 150.

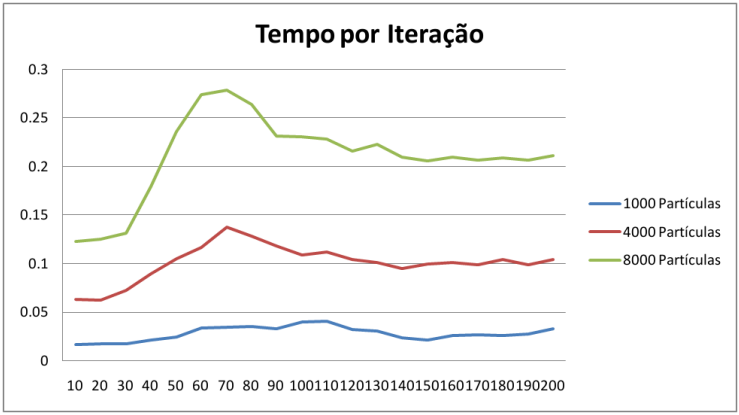


Figura 5.14: Tempo total para 1000, 4000 e 8000 partículas.

Discussão

A simulação com 1000 partículas tende a cair em dois casos: Ou as partículas são muito grandes e visualmente desagradáveis, ou o volume de fluido usado é menor e o ambiente parece mais "vazio". Com 8000 partículas há uma noção de fluidez maior (visto que o fluido parece mais com um líquido e menos com um conjunto de bolas), mas o custo deixa a simulação executando a menos de 5 quadros por segundo. Um meio termo interessante é o uso de 4000 partículas. A Figura 5.14 ilustra o tempo total médio por conjunto de 10 iterações para os três casos.

Um fato interessante é como passo predominante muda entre as configurações - não é sempre um dos passos que domina o tempo de processamento. Os dois passos que mais consomem tempo, a Busca de Vizinhos e a Colisão, invertem suas posições entre os casos 1000 e 8000 partículas. A Tabela 5.5 e o gráfico da Figura 5.15 ilustram as porcentagens que os passos consomem do tempo total de interação. Conclui-se por esta Tabela e a discussão apresentada nas subseções anteriores que, quando o ambiente de simulação é grande o bastante para que as partículas não precisem ficar "espremidas" uma contra

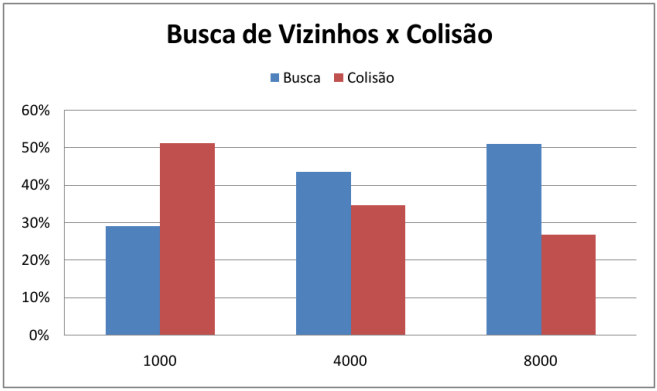


Figura 5.15: Porcentagem média que os passos de Busca de Vizinhos e Colisão consomem das iterações, para os três casos.

as outras, a Busca de Vizinhos será feita mais rapidamente do que a Colisão. Em contrapartida, em ambientes apertados, a Colisão ganha da Busca de Vizinhos.

	1000	4000	8000
Busca de Vizinhos	29.10%	43.69%	50.98%
Forças do Fluido	11.07%	15.62%	16.77%
Forças Externas	3.39%	3.18%	3.00%
Integração	0.67%	0.70%	0.63%
Colisão	51.30%	34.69%	26.79%
Renderização	4.47%	2.12%	1.84%

Tabela 5.5: Porcentagem média que os passos consomem das iterações, para os três casos.

Finalmente, como a força do Eixo Médio Aproximado é uma contribuição deste trabalho, a Figura mostra a evolução do passo de Forças Externas para os três casos. Mesmo usando curvas ao invés de segmentos de retas, o cálculo da Força do Eixo Médio Aproximado sobre cada partícula consome em torno de 3% do tempo total da iteração. A Figura 5.16 mostra a evolução deste passo nos três casos.

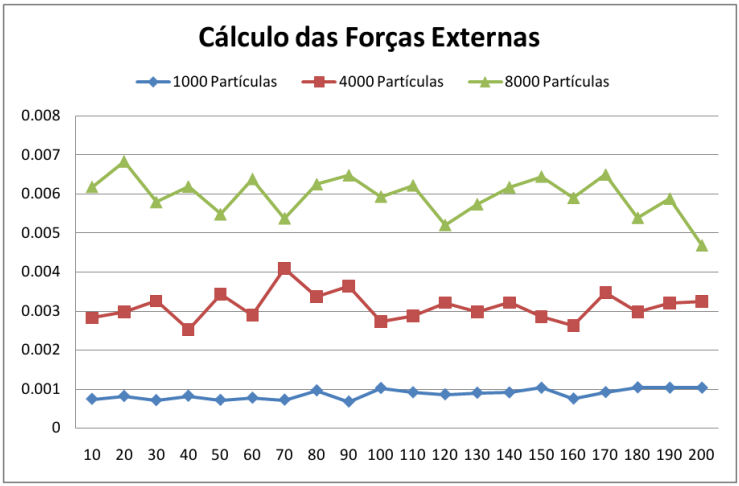


Figura 5.16: Porcentagem média que os passos de Busca de Vizinhos e Colisão consomem das iterações, para os três casos.

6 *Conclusões e Trabalhos Futuros*

Este capítulo iniciará discutindo as limitações assumidas neste trabalho, uma vez que o tópico é grande e é impraticável abordá-lo por completo. Cada item desta lista de limitações é um potencial trabalho futuro, com mérito de estudo por si só.

6.1 Interação Fluido-Estrutura

Posicionamento inicial das partículas

O algoritmo para o posicionamento inicial das partículas ao longo do Eixo Médio Aproximado, que é descrito no Capítulo 4, possui fraquezas por não levar em conta a forma da malha de triângulos. A Seção 4.2.1 discute este problema e as duas medidas adotadas — transladar cada disco de partículas para o centro da seção transversal da estrutura na região, e testar cada partícula individualmente descartando as que se encontrem fora da estrutura. Estas partículas descartadas são, idealmente, recriadas posteriormente no processo, em algum outro disco.

Embora estas medidas atenuem o problema na maioria dos casos, é possível que um número suficiente de partículas seja descartado ao longo do po-

sicionamento dos discos, sobretudo se o volume de fluido desejado for muito grande em relação ao espaço de simulação. Por consequência, sobra um número muito grande de partículas para serem posicionadas nos discos finais, e isto pode não ser possível. Esta é uma limitação clara decorrente da estratégia de simplificar uma estrutura tubular arbitrariamente complexa como um conjunto de cilindros que seguem uma aproximação do Eixo Médio.

Deformações na Estrutura Tubular

O principal motivo para a existência da força do Eixo Médio Aproximado (Capítulo 4) é a ausência de deformações na malha tubular. Um modelo de deformação é necessário para aprimorar o realismo da simulação e permitir a troca de energia entre a estrutura deformável e o fluido. Modelos de deformação de superfícies no contexto de simulações cirúrgicas incluem Modelos de Elementos Finitos (MULLER; SCHIRM; TESCHNER, 2004; MOSEGAARD, 2003; BRO-NIELSEN, 1998) e modelos de Massa-Mola (MOSEGAARD, 2006; CARVALHO, 2007).

Modelo de Referência

A avaliação do resultado final obtido neste trabalho deu-se primariamente através da avaliação do tempo de processamento dos vários passos abordados e da inspecção visual da simulação gerada. Este segundo aspecto é demasiadamente qualitativo e só permitiria a validação do modelo por parte de um especialista em um nível superficial.

Para sanar esta deficiência ainda na ausência de um modelo numérico fisicamente correto, poderia ser criado um modelo de referência teórico com a ajuda de um especialista da área do cenário simulado. Este modelo descreveria o comportamento esperado do sistema em função dos parâmetros de entrada, e serviria como um passo inicial para a validação do sistema.

Outros Tipos de Força Tubular

O modelo adotado neste trabalho utiliza uma força que tende a seguir exatamente a direção do Eixo Médio Aproximado, composto tanto por segmentos de retas quando por uma curva interpolada. Poderia-se prever outros tipos de forças baseadas no Eixo Médio aproximado — um exemplo seria uma força semelhante ao modelo de pressão adotado. Neste caso, partículas próximas ao Eixo Médio Aproximados sofreriam uma força que as empurraria para frente e para longe do Eixo, em direção às paredes da estrutura. Em contrapartida, a partir de uma certa distância as partículas começariam a sofrer uma força trazendo-as próximas novamente ao Eixo Médio Aproximado.

6.2 Aceleração com Placas Gráficas

Evolução do Fluido

Como mencionado no final do Capítulo 3, toda a simulação utilizada neste trabalho foi implementada em CPU. Abordagens em GPU existem e se tornam cada vez mais populares à medida que a tecnologia de hardware e software das placas gráficas melhora e se torna mais barata. Um trabalho futuro viável é o transporte de toda a simulação, inclusive no que diz respeito à interação com a estrutura tubular e a força do Eixo Médio Aproximado, para a GPU. Tal medida possibilitaria resoluções mais altas tanto na estrutura tubular quando no número de partículas, aumentando a percepção de fluidez e o apelo visual. Estruturas de aceleração espacial como Bounding Trees já foram transportadas com sucesso para a GPU, e Thrane e Simonsen (2005), por exemplo, reportaram uma melhoria de até 9 vezes utilizando as próprias Bounding Trees, versus implementações com grids uniformes e KD-trees, numa aplicação de raytracing.

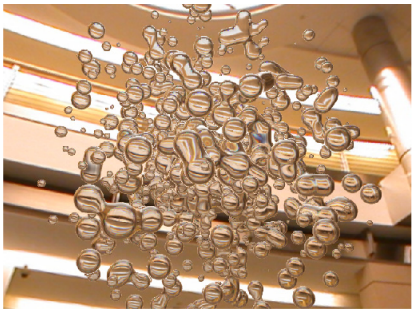


Figura 6.1: Raycasting de metaballs de Kanamori, Szego e Nishita (2008).

Renderização

Além da representação das partículas diretamente como esferas, muitos trabalhos focam na obtenção da superfície do fluido para aumento do realismo, como o uso do algoritmo Marching Cubes com SPH para obtenção dos isovalores (MULLER; CHARYPAR; GROSS, 2003; PAIVA et al., 2006).

A renderização em si é provavelmente o passo mais apropriado para as placas gráficas, uma vez que este é seu propósito inicial. Kanamori, Szego e Nishita (2008) utilizam Raycasting totalmente em GPU sobre superfícies implícitas denominadas Metaballs para a renderização eficiente de fluidos (Figura 6.1). De fato, muitas das abordagens atuais de geração de superfície de sistemas baseados em partículas podem ser transportadas para a GPU — um exemplo é a abordagem de Müller, Schirm e Duthaler (2007), que gera uma malha bidimensional no espaço da tela projetando a nuvem de pontos formada pelas partículas sobre a câmera de visualização (Figura 6.2). O modelo foi implementado em CPU, mas os próprios autores apontam para o uso da GPU como uma trabalho futuro.

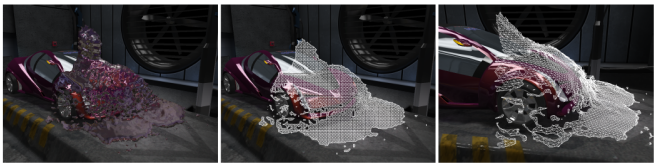


Figura 6.2: Geração de malha bidimensional da superfície visível do fluido deMüller, Schirm e Duthaler (2007).

6.3 Dinâmica do Fluido

Fluidos Não-Newtonianos

Em um fluido Newtoniano o coeficiente de viscosidade, definido como a resistência do material à deformação e ilustrado nas equações do Capítulo 3 pelo símbolo μ, é constante. Fluidos não-Newtonianos são aqueles aonde o coeficiente de viscosidade varia de forma não-linear de acordo com o estresse excercido pelo ambiente, a pressão, a temperatura, entre outros fatores. Um exemplo de fluido não-Newtoniano é a areia movediça, que apresenta maior resistência sob movimentos bruscos e rápidos.

O sangue também é um fluido não-Newtoniano. Neste trabalho o coeficiente de viscosidade foi tido como constante durante toda a simulação, de forma que uma simulação mais fiel do líquido sanguíneo exigiria a incorporação desta propriedade de variação de viscosidade.

Abordagens para simulação de fluidos não-Newtonianos via SPH incluem o uso de tensores de tensão baseados na variação da velocidade relativa entre partículas vizinhas para o cálculo da viscosidade na simulação de fluxos de lama (SHAO; LO, 2003) e no derretimento de objetos sólidos (PAIVA et al., 2006) (Figura 6.3).

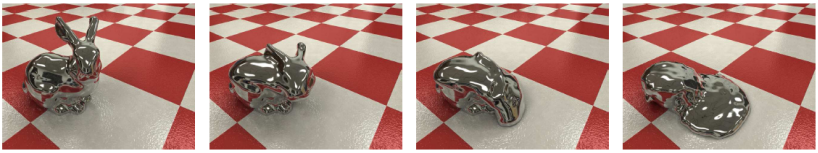


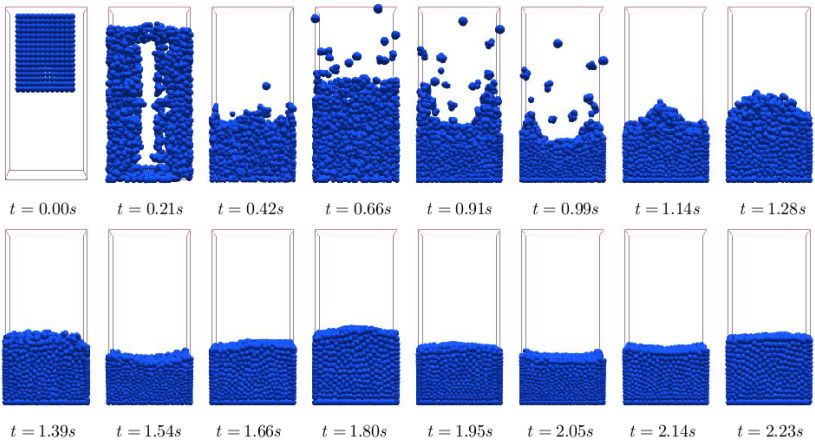
Figura 6.3: Simulação de fluido não-Newtoniano para derretimento de sólidos (PAIVA et al., 2006).

Verificação da Incompressibilidade

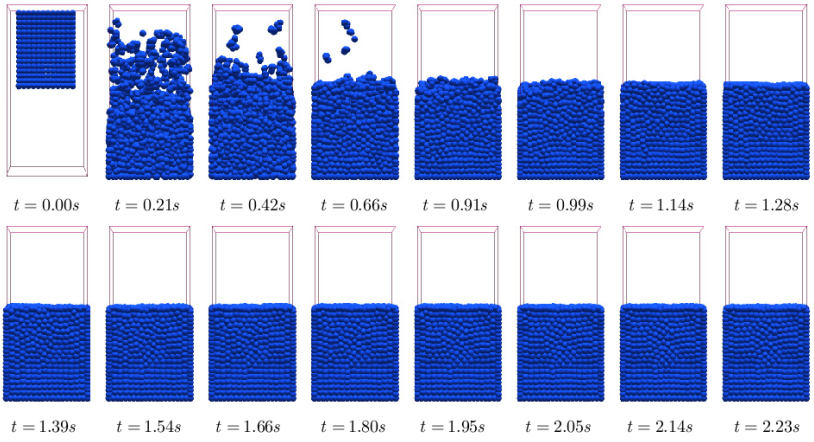
A abordagem descrita no Capítulo 3 na página 39 para a definição da densidade das massas e da pressão das partículas utiliza a equação do gás ideal, o que resulta em alta compressibilidade e oscilações que são indesejáveis na simulação de fluidos (BECKER; TESCHNER, 2007). Este problema é considerável e é um dos principais fatores influentes na performance do protótipo do Capítulo 5, uma vez que a compressibilidade aumenta a vizinhança das partículas, o que afeta quase todos os passos. Garantir a incompressibilidade no método SPH é um problema difícil (KEISER et al., 2005) e com várias abordagens diferentes na literatura.

A maneira mais direta apresentada por Kelager (2006) consiste em alterar o valor da constante k utilizada na obtenção da pressão agindo sobre as partículas (Equação 3.20 na página 58). O autor sugere diminuir o intervalo de tempo Δt por um fator de 10 e aumentar o valor de k da Equação 3.20 até o máximo valor permitido que ainda mantenha a estabilidade da simulação. O autor ilustra este ponto em um cenário onde Δt e k são variados desta maneira enquanto todos os outros parâmetros são mantidos constantes. Tal cenário é ilustrado na Figura 6.4.

Parte do problema é decorrente do fato da densidade das massas ser uma função direta das massas na vizinhança de uma partícula (Equação 3.16 na página 56). O resultado é que partículas na superfície do líquido possuirão densidade das massas mais baixa por possuírem menos vizinhos. Para sa-



(a) $\Delta t = 0.01s$ e $k = 3J$.



(b) $\Delta t = 0.001s$ e $k = 100J$.

Figura 6.4: Cenário ilustrativo do problema da incompressibilidade elaborado por Kelager (2006).

nar este problema, Paiva et al. (2006) fixam o valor da densidade no início da simulação e calculam a variação desta densidade em cada partícula como uma função da velocidade relativa de seus vizinhos. Sigalotti, Daza e Donoso (2006) utilizam um valor de raio de suporte h variável para garantir densidade constante na superfície e ainda modelar efeitos de pequena escala nas extremidades. Valores locais são também explorados por Monaghan (1994), que atribui um limite superior para a razão entre a velocidade de fluxo local e uma constante utilizada na obtenção da pressão (ANTOCI; GALLATI; SIBILLA, 2007).

Parâmetros de Simulação

Os parâmetros de simulação utilizados no Capítulo 5 são todos descobertos empiricamente, ou seja, são testados em configurações diferentes até apresentarem resultados convincentes. Para que a simulação possa futuramente ser útil ao especialista em ambientes como simulações de cirurgia virtual e diagnóstico, é preciso que a simulação siga a realidade o máximo possível. Idealmente, os parâmetros deveriam ser baseados nos dados de entrada, como as séries de imagens de tomografia computadorizada, e esta é uma limitação clara do trabalho.

Caracterização Sanguínea

Além da possibilidade de simulação de fluidos não-newtonianos e da necessidade de parametrização em função dos dados de entrada, existem outros componentes da fisiologia sanguíneo que eventualmente podem necessitar de contemplação para melhor a fidelidade da simulação. Para começar, o sangue não é um fluido uniforme — ele possui uma fase sólida, composta por elementos celulares, e uma fase líquida, que é o plasma.

Adicionalmente, e esta não é uma propriedade exclusiva do sangue, o

escoamente de fluidos toma propriedades complexas quando a velocidade excede um valor crítico. Sob alta velocidade, surgem dentro do fluido correntes circulares locais (vórtices) que geram um grande aumento na resistência do fluido ao escoamento. A combinação de fatores que indicam se um escoamento é turbulento é o chamado Número de Reynolds, que atualmente não toma parte das equações de hidrodinâmica adotadas.

6.4 Conclusões

Neste trabalho foi apresentada uma abordagem para a simulação simplificada de fluidos através de estruturas tubulares. O fluido foi representado por um sistema de partículas onde cada partícula representa uma "porção" do líquido, e a sua dinâmica foi simulada utilizando o método de interpolação Smoothed Particle Hydrodynamics. A estrutura tubular foi representada por uma malha de triângulos, da qual foi extraída uma aproximação do Eixo Médio através de um algoritmo que utiliza Bounding Trees para rápido cálculo. Estas mesmas Bounding Trees são utilizadas na detecção de colisão entre as partículas do fluido e os triângulos da malha.

O Eixo Médio Aproximado representa uma força que empurra as partículas ao longo da estrutura tubular, tal qual o componente físico que na realidade empurraria o fluido adiante, como o bombeamento cardíaco, o estreitamento arterial, um motor ou bomba em encanamentos, etc. A força pode ser suavizada com o uso de curvas ao invés de segmentos de reta e possui a vantagem do custo de cálculo ser baixo e dependente apenas no número de partículas, e não no número de triângulos da malha. As partículas são renderizadas como esferas que podem ser coloridas de acordo com grandezas extraídas das propriedades do fluido, como pressão, densidade e velocidade, para maior apelo visual.

Embora a abordagem do Eixo Médio Aproximado em si seja computa-

cionalmente barata, a simulação de milhares de partículas dentro de uma estrutura composta por milhares de triângulos é custosa. Os objetivos de apelo visual (que aumenta com o número de partículas e triângulos) e interatividade (que diminui com o aumento no tempo de processamento por iteração) são antagônicos e geram a necessidade de um compromisso. Uma possibilidade atraente é a exploração do paralelismo inerente dos passos de simulação através do uso de placas gráficas para a simulação. Tal estratégia é um trabalho futuro promissor.

Adicionalmente, a dinâmica do fluido foi representada apenas por um modelo que assume que o fluido é Newtoniano e incompressível. A primeira suposição é falha na representação de fluidos como o sangue, que possui propriedades não-Newtonianas. Isto impossibilita uma simulação fidedigna à realidade.

A segunda suposição falha entre outros motivos pelo modelo de pressão utilizado, que é simples de implementar mas gera forças que não garantem a incompressibilidade. Infelizmente não houve tempo hábil para a investigação de outras abordagens (algumas das quais descritas na Seção 6.3). Além da representação não-realista que a abordagem acarreta em alguns cenários, a falta da incompressibilidade faz com que as partículas se aglutinem em situações como o choque contra superfícies. Este agrupamento faz com que cada partícula possua um número maior de vizinhos, e isto tem um impacto direto na performance dos passos de definição de vizinhança e cálculo das forças internas do fluido.

Apesar destas desvantagens e limitações, a conclusão tomada é que há mérito na exploração deste tipo de abordagem, sobretudo porque soluções para os problemas observados existem na literatura e poderiam em sua maioria serem integrados ao sistema modularmente.

APÊNDICE A – Tabelas de Tempo de Processamento

4000 Particulas	10	20	30	40	50	60	70	80	90	100	110
Busca de Vizinhos	0.03355	0.03004	0.03392	0.04098	0.05077	0.05279	0.05495	0.05296	0.04858	0.04703	0.04697
Forças do Fluido	0.01000	0.01012	0.01334	0.01787	0.02192	0.02197	0.01772	0.01692	0.01834	0.01921	0.01772
Forças Externas	0.00282	0.00298	0.00326	0.00252	0.00343	0.00289	0.00410	0.00337	0.00364	0.00272	0.00288
Integração	0.00068	0.00067	0.00068	0.00069	0.00066	0.00068	0.00068	0.00068	0.00068	0.00069	0.00069
Colisão	0.01371	0.01642	0.01900	0.02554	0.02651	0.03607	0.05830	0.05278	0.04519	0.03737	0.04181
Renderização	0.00245	0.00200	0.00203	0.00201	0.00203	0.00201	0.00202	0.00202	0.00204	0.00204	0.00204
Total	0.06320	0.06223	0.07222	0.08961	0.10533	0.11641	0.13776	0.12874	0.11846	0.10906	0.11210

4000 Particulas	120	130	140	150	160	170	180	190	200	Média	Dsv. Pad.
Busca de Vizinhos	0.04495	0.04393	0.03777	0.04061	0.04126	0.03888	0.04364	0.04528	0.04615	0.04375	0.00676
Forças do Fluido	0.01713	0.01208	0.01240	0.01573	0.01365	0.01589	0.01289	0.01209	0.01656	0.01568	0.00350
Forças Externas	0.00321	0.00298	0.00321	0.00285	0.00262	0.00348	0.00298	0.00320	0.00324	0.00312	0.00037
Integração	0.00069	0.00069	0.00069	0.00069	0.00069	0.00069	0.00069	0.00068	0.00069	0.00068	0.00001
Colisão	0.03625	0.03982	0.03918	0.03751	0.04078	0.03777	0.04207	0.03534	0.03558	0.03585	0.01114
Renderização	0.00204	0.00204	0.00204	0.00205	0.00204	0.00205	0.00204	0.00205	0.00203	0.00205	0.00009
Total	0.10428	0.10154	0.09528	0.09944	0.10103	0.09875	0.10430	0.09864	0.10425	0.10113	0.01897

Tabela A.1: Tempo em segundos para a simulação da evolução da pressão com 4000 partículas.

1000 Particulas	10	20	30	40	50	60	70	80	90	100	110
Busca de Vizinhos	0.0077747	0.0075631	0.007053	0.0079339	0.0093909	0.009708	0.010753	0.0064694	0.0073775	0.0092537	0.0066729
Forças do Fluido	0.0031524	0.0025257	0.0029914	0.0034155	0.0057151	0.0036766	0.0039793	0.002726	0.0031387	0.0026548	0.0028826
Forças Externas	0.0007415	0.0008162	0.0007087	0.000822	0.0007156	0.0007715	0.000721	0.0009621	0.0006722	0.0010231	0.000912
Integração	0.0001691	0.0001623	0.0001676	0.0001751	0.0001782	0.0001756	0.0001753	0.0001665	0.000178	0.0001619	0.0001679
Colisão	0.0041584	0.0047203	0.004969	0.0062403	0.0094102	0.0148765	0.0168502	0.0225948	0.0214463	0.0237489	0.0236764
Renderização	0.0013397	0.0013403	0.0013826	0.0014489	0.0010233	0.0010286	0.0010318	0.0010357	0.0010371	0.0010456	0.0010443
Total	0.01734	0.01713	0.01727	0.02004	0.02643	0.03024	0.03351	0.03395	0.03385	0.03789	0.03536

1000 Particulas	120	130	140	150	160	170	180	190	200	Média	Dsv. Pad.
Busca de Vizinhos	0.0083915	0.0059382	0.0059152	0.005695	0.0049214	0.0064863	0.008091	0.0071938	0.0071582	0.00749	0.00148
Forças do Fluido	0.002414	0.0024307	0.0026142	0.0021013	0.0022053	0.0019913	0.0019897	0.0017451	0.0021174	0.00282	0.00090
Forças Externas	0.0008586	0.0008997	0.0009084	0.0010292	0.000749	0.0009161	0.0010392	0.0010313	0.0010314	0.00087	0.00013
Integração	0.0001718	0.00017	0.0001694	0.0001654	0.0001749	0.0001739	0.0001617	0.0001613	0.0001652	0.00017	0.00001
Colisão	0.0184986	0.0207214	0.0140544	0.0078809	0.0134487	0.0169383	0.0146203	0.0158861	0.0169467	0.01458	0.00645
Renderização	0.0010394	0.0010431	0.0010437	0.0010462	0.0010416	0.0010481	0.0010353	0.0010392	0.0010423	0.00111	0.00014
Total	0.03137	0.03120	0.02471	0.01792	0.02254	0.02755	0.02694	0.02706	0.02846	0.02704	0.00656

Tabela A.2: Tempo em segundos para a simulação da evolução da pressão com 1000 partículas.

8000 Particulas	10	20	30	40	50	60	70	80	90	100	110
Busca de Vizinhos	0.06367	0.06155	0.06811	0.09413	0.12534	0.13552	0.13632	0.13410	0.11877	0.11794	0.11467
Forças do Fluido	0.01978	0.02005	0.02008	0.03489	0.04907	0.05047	0.05257	0.04178	0.03800	0.03709	0.03766
Forças Externas	0.00618	0.00683	0.00578	0.00618	0.00547	0.00638	0.00537	0.00624	0.00648	0.00593	0.00621
Integração	0.00122	0.00123	0.00125	0.00128	0.00137	0.00131	0.00129	0.00125	0.00123	0.00123	0.00125
Colisão	0.02860	0.03206	0.03269	0.03951	0.05143	0.07705	0.07949	0.07738	0.06332	0.06460	0.06525
Renderização	0.00391	0.00366	0.00360	0.00370	0.00362	0.00359	0.00367	0.00362	0.00363	0.00369	0.00365
Total	0.12336	0.12538	0.13151	0.17969	0.23631	0.27432	0.27870	0.26439	0.23141	0.23047	0.22870

8000 Particulas	120	130	140	150	160	170	180	190	200	Média	Dsv. Pad.
Busca de Vizinhos	0.11274	0.11212	0.10573	0.10476	0.10746	0.10523	0.10868	0.10383	0.11039	0.10705	0.02149
Forças do Fluido	0.03363	0.03703	0.03342	0.03475	0.03254	0.03459	0.03170	0.03382	0.03661	0.03548	0.00893
Forças Externas	0.00520	0.00573	0.00617	0.00644	0.00589	0.00649	0.00538	0.00588	0.00467	0.00595	0.00053
Integração	0.00128	0.00124	0.00124	0.00123	0.00123	0.00123	0.00123	0.00124	0.00124	0.00125	0.00004
Colisão	0.05968	0.06329	0.05969	0.05547	0.05933	0.05590	0.05821	0.05853	0.05481	0.05682	0.01436
Renderização	0.00364	0.00364	0.00369	0.00363	0.00368	0.00364	0.00363	0.00363	0.00365	0.00366	0.00007
Total	0.21618	0.22305	0.20993	0.20628	0.21015	0.20708	0.20884	0.20692	0.21137	0.21020	0.04336

Tabela A.3: Tempo em segundos para a simulação da evolução da pressão com 8000 partículas.

Referências Bibliográficas

AKLEMAN, Ergun; CHEN, Jianer. Guaranteeing 2-manifold property for meshes. In: *SMI '99: Proceedings of the International Conference on Shape Modeling and Applications*. Washington, DC, USA: IEEE Computer Society, 1999. ISBN 076950065X. Disponível em: <http://portal.acm.org/citation.cfm?id=829509.830281>.

AMADA, T.; IMURA, M.; YASUMURO, Y.; MANABE, Y.; CHIHARA, K. Particle-based fluid simulation on gpu. *ACM Workshop on General-Purpose Computing on Graphics Processors and SIGGRAPH 2004 Poster Session*, 2004.

ANTOCI, Carla; GALLATI, Mario; SIBILLA, Stefano. Numerical simulation of fluid-structure interaction by sph. *Comput. Struct.*, Pergamon Press, Inc., Elmsford, NY, USA, v. 85, p. 879–890, June 2007. ISSN 0045-7949. Disponível em: <http://portal.acm.org/citation-.cfm?id=1243516.1243764>.

ATKINS, P. W. *Fisico-Quimica*. 6. ed. [S.l.]: LTC, 1999.

BAUMGART, Bruce G. Winged-edge polyhedron representation for computer vision. In: *National Computer Conference*. [s.n.], 1975. Disponível em: <http://www.baumgart.org/winged-edge/winged-edge.html>.

BECKER, Markus; TESCHNER, Matthias. Weakly compressible sph for free surface flows. In: *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2007. p. 209–217. ISBN 978-1-59593-624-4.

BERGEN, Gino van den. *Collision Detection in Interactive 3D Environments (The Morgan Kaufmann Series in Interactive 3D Technology)*. Morgan Kaufmann, 2003. Hardcover. ISBN 155860801X. Disponível em: <http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20\&path=ASIN/155860801X>.

BIASI, Herculano De; WANGENHEIM, Aldo von; SILVEIRA, Pierre G.; COMUNELLO, Eros. 3d reconstruction of abdominal aortic aneurysms. In: *Computer-Based Medical Systems 2002*. Los Alamitos, CA, USA: IEEE Computer Society, 2002. v. 00.

BITTER, I.; SATO, M.; BENDER, M.; MCDONNELL, K. T.; KAUFMAN, A.; WAN, Ming. Ceasar: a smooth, accurate and robust centerline extraction algorithm. In: *Visualization 2000. Proceedings*. [s.n.], 2000. p. 45–52. Disponível em: <http://dx.doi.org/10.1109/VISUAL.2000% -.885675>.

BLANCO, P.J.; FEIJóO, R.A.; URQUIZA, S.A. A unified variational approach for coupling 3d-1d models and its blood flow applications. *Computer Methods in Applied Mechanics and Engineering*, v. 196, n. 41-44, p. 4391 – 4410, 2007. ISSN 0045-7825. Disponível em: <http://www.sciencedirect.com/science/article/B6V29-4NTRT0S-4/2-/0d6058918630ff6d29e8d57ae3d55443>.

BLANCO, P.J.; PIVELLO, M.R.; URQUIZA, S.A.; FEIJóO, R.A. On the potentialities of 3d-1d coupled models in hemodynamics simulations. *Journal of Biomechanics*, v. 42, n. 7, p. 919 – 930, 2009. ISSN 0021-9290. Disponível em: <http://www.sciencedirect.com/science/article/B6T82-4VT0H4S-2/2/5d20f7df2aec899bd915fc4d0da9aa17>.

BLUM, Harry. A Transformation for Extracting New Descriptors of Shape. In: WATHEN-DUNN, Weiant (Ed.). *Models for the Perception of Speech and Visual Form*. Cambridge: MIT Press, 1967. p. 362–380.

BRIDSON, Robert; FISCHER, Matthias M.; GUENDELMAN, Eran. Fluid simulation. In: *SIGGRAPH 2006 Course Notes*. [S.l.: s.n.], 2006.

BRO-NIELSEN, M. Finite element modeling in surgery simulation. *Proceedings of the IEEE*, v. 86, n. 3, p. 490–503, Mar 1998. ISSN 0018-9219.

CARVALHO, Diego Dias Bispo. *Modelo computacional para representação gráfica do comportamento elástico de superfícies deformáveis e interação com sólidos para aplicações em tempo real*. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, Programa de Pós-Graduação em Ciências da Computação, 2007.

CARVALHO, Diego D. B.; SANTOS, Thiago R. dos; WANGENHEIM, Aldo von. Measuring arterial diameters for surgery assistance, patient customized endovascular prosthesis design and post-surgery evaluation. *Computed-Based Medical Systems (CBMS)*, IEEE Computer Society, Los Alamitos, CA, USA, v. 0, p. 225–230, 2006. ISSN 1063-7125.

CATMULL, Edwin; ROM, Raphael. A class of local interpolating splines. *Computer Aided Geometric Design*, Academic Press, p. 317–326, 1974.

COHEN, Jonathan D.; LIN, Ming C.; MANOCHA, Dinesh; PONAMGI, Madhav K. I-collide: An interactive and exact collision detection system for large-scale environments. In: *Symposium on Interactive 3D Graphics*. [s.n.], 1995. p. 189–196, 218. Disponível em: <http://citeseer.ist.psu.edu-/cohen95icollide.html>.

CRAMER, M. S. *Navier-Stokes Equations - Incompressible Flows*. 2002. Disponível em: <http://www.navier-stokes.net/nsinc.htm>.

DESBRUN, Mathieu; GASCUEL, Marie P. Smoothed particles: a new paradigm for animating highly deformable bodies. In: *Proceedings of the Eurographics workshop on Computer animation and simulation '96*. New York, NY, USA: Springer-Verlag New York, Inc., 1996. p. 61–76.

DESCHAMPS, T.; SCHWARTZ, P.; TREBOTICH, D.; COLELLA, P.; SALONER, D.; MALLADI, R. Vessel segmentation and blood flow simulation using level-sets and embedded boundary methods. *International Congress Series*, v. 1268, p. 75 – 80, 2004. ISSN 0531-5131. CARS 2004 - Computer Assisted Radiology and Surgery. Proceedings of the 18th International Congress and Exhibition. Disponível em: <http://www.sciencedirect.com/science/article/B7581-4CHRSVD-R/2-/a78860143312a4790eca4156e3d83255>.

DONEA, J.; HUERTA, J.-Ph. Ponthot Antonio; RODRíGUEZ-FERRAN, A. Arbitrary lagrangian–eulerian methods. In: STEIN, E.; BORST, R. de; HUGHES, T.J.R. (Ed.). *Encyclopedia of computational mechanics, Volume 1, Fundamentals*. [S.l.]: John Wiley, 2004.

ERICSON, Christer. *Real-Time Collision Detection (The Morgan Kaufmann Series in Interactive 3D Technology)*. Morgan Kaufmann, 2004. Hardcover. ISBN 1558607323. Disponível em: <http://www.amazon.ca/exec/obidos-/redirect?tag=citeulike09-20\&path=ASIN/1558607323>.

FOLEY, J. D.; DAM, A. Van; FEINER, S. K.; HUGHES, J. F. *Computer Graphics: Principles and Practices. Second Edition in C*. 2nd. ed. [S.l.]: Addison-Wesley Publishing Company, 1996. (Addison-Wesley Systems Programming Series).

FOSTER, Nick; METAXAS, Dimitri. Realistic animation of liquids. *Graphical models and image processing: GMIP*, v. 58, n. 5, p. 471–483, 1996. Disponível em: <citeseer.ist.psu.edu/foster95realistic.html>.

GINGOLD, R. A.; MONAGHAN, J. J. Smoothed particle hydrodynamics - theory and application to non-spherical stars. *mnras*, v. 181, p. 375–389, November 1977.

GOTTSCHALK, S.; LIN, M. C.; MANOCHA, D. Obbtree: a hierarchical structure for rapid interference detection. In: *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press, 1996. p. 171–180. ISBN 0897917464. Disponível em: <http://dx.doi.org/10.1145/237170.237244>.

HAINES, Eric; AKENINE-MOLLER, Tomas. *Real-Time Rendering (2nd Edition)*. AK Peters, Ltd., 2002. Hardcover. ISBN 1568811829. Disponível em: <http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20\&path=ASIN/1568811829>.

HARADA, Takahiro; KOSHIZUKA, Seiichi; KAWAGUCHI, Yoichiro. Smoothed particle hydrodynamics on gpus. In: . [S.l.: s.n.], 2007. p. 63–70.

HARLOW, F. H.; WELCH, J. E. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. In: *The Physics of Fluids*. [S.l.: s.n.], 1965. p. 2182–2189.

HIRT, C. W.; AMSDEN, A. A.; COOK, J. L. An arbitrary lagrangian-eulerian computing method for all flow speeds. *Journal of Computational Physics*, v. 14, n. 3, p. 227 – 253, 1974. ISSN 0021-9991. Disponível em: <http://www.sciencedirect.com/science/article/B6WHY-4DDR5HS-11F/2-/1d0414be3e07426aba8148b91b360abe>.

HUBBARD, Philip M. Collision detection for interactive graphics applications. *IEEE Transactions on Visualization and Computer Graphics*, v. 1, n. 3, p. 218–230, 1995. Disponível em: <http://citeseer.ist.psu.edu-/165547.html>.

HUT, Piet; MAKINO, Jun. The art of computer science. November 2006.

JAILLET, F.; SHARIAT, B.; VANDORPE, D. *Deformable volume object modeling with a particle-based system for medical applications*. 1997. Disponível em: <citeseer.ist.psu.edu/jaillet97deformable.html>.

JIANG, Guangxiang; GU, Lixu. An automatic and fast centerline extraction algorithm for virtual colonoscopy. In: *IEEE-EMBS 2005. 27th Annual International Conference of the Engineering in Medicine and Biology Society, 2005.* [s.n.], 2005. p. 5149–5152. Disponível em: <http://dx.doi.org/10.1109/IEMBS.2005.1615636>.

KALVIN, Alan D.; LAINE, Andrew F.; SONG, Ting. Faster, higher quality volume visualization for 3d medical imaging. In: MIGA, Michael I.; CLEARY, Kevin R. (Ed.). SPIE, 2008. v. 6918, n. 1, p. 691830. Disponível em: <http://link.aip.org/link/?PSI/6918/691830/1>.

KANAMORI, Yoshihiro; SZEGO, Zoltan; NISHITA, Tomoyuki. Gpu-based fast ray casting for a large number of metaballs. *Computer Graphics Forum*, v. 27, n. 2, p. 351–360, 2008.

KASS, Michael; MILLER, Gavin. Rapid, stable fluid dynamics for computer graphics. In: *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press, 1990. p. 49–57.

KEISER, R.; ADAMS, B.; GASSER, D.; BAZZI, P.; DUTRE, P.; GROSS, M. A unified lagrangian approach to solid-fluid animation. In: *Point-Based Graphics, 2005. Eurographics/IEEE VGTC Symposium Proceedings*. [S.l.: s.n.], 2005. p. 125–148. ISSN 1511-7813.

KELAGER, Micky. Lagrangian fluid dynamics using smoothed particle hydrodynamics. January 2006. Disponível em: <www.opentissue.org>.

KIRBAS, Cemil; QUEK, Francis. A review of vessel extraction techniques and algorithms. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 36, n. 2, p. 81–121, 2004. ISSN 0360-0300.

KOSHIZUKA, Seiichi; NOBE, Atsushi; OKA, Yoshiaki. Numerical analysis of breaking waves using the moving particle semi-implicit method. *International Journal for Numerical Methods in Fluids*, v. 26, p. 751–769, 1998.

LACROUTE, Philippe; LEVOY, Marc. Fast volume rendering using a shear-warp factorization of the viewing transformation. In: *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1994. p. 451–458. ISBN 0-89791-667-0.

LAW, Tsui Ying; HENG, Pheng-Ann. Automatic centerline extraction for 3d virtual bronchoscopy. In: *MICCAI '00: Proceedings of the Third International Conference on Medical Image Computing and Computer-Assisted Intervention*. London, UK: Springer-Verlag, 2000. p. 786–795. ISBN 3-540-41189-5.

LEVOY, Marc. Display of surfaces from volume data. *IEEE Comput. Graph. Appl.*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 8, n. 3, p. 29–37, 1988. ISSN 0272-1716.

LORENSEN, William E.; CLINE, Harvey E. Marching cubes: A high resolution 3d surface construction algorithm. In: *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press, 1987. v. 21, n. 4, p. 163–169. ISSN 0097-8930. Disponível em: <http://portal.acm.org/citation-.cfm?id=37422>.

LUCY, Leon B. A numerical approach to testing the fission hypothesis. *aj*, v. 82, n. 12, p. 1013–1924, December 1977.

MONAGHAN, J. J. Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics*, p. 543–574, 1992.

MONAGHAN, J. J. Simulating free surface flows with sph. *Journal of Computational Physics*, v. 110, n. 2, p. 399 – 406, 1994. ISSN 0021-9991. Disponível em: <http://www.sciencedirect.com/science/article/B6WHY-45PTPT7-K/2/b0d6650b12e33347d9fd15c7465c2e39>.

MORRIS, J. P. Simulating surface tension with smoothed particle hydrodynamics. *International Journal for Numerical Methods in Fluids*, v. 33, p. 333–353, June 2000.

MOSEGAARD, Jesper. *Realtime Cardiac Surgery Simulation*. Dissertação (Mestrado) — Department of Computer Science, University of Aarhus, Denmark, March 2003.

MOSEGAARD, Jesper. *Cardiac Surgery Simulation - Graphics Hardware meets Congenital Heart Disease*. Tese (Doutorado) — Department of Computer Science, University of Aarhus, Denmark, October 2006.

MUELLER, Matthias; SCHIRM, Simon; TESCHNER, Matthias; HEIDELBERGER, Bruno; GROSS, Markus. *Interaction of Fluids with Deformable Solids*. 2004. Disponível em: <citeseer.ist.psu.edu-/mueller04interaction.html>.

MULLER, Matthias; CHARYPAR, David; GROSS, Markus. Particle-based fluid simulation for interactive applications. In: *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2003. p. 154–159.

MüLLER, Matthias; SCHIRM, Simon; DUTHALER, Stephan. Screen space meshes. In: *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2007. p. 9–15. ISBN 978-1-59593-624-4.

MULLER, Matthias; SCHIRM, Simon; TESCHNER, Matthias. Interactive blood simulation for virtual surgery based on smoothed particle hydrodynamics. *Technol. Health Care*, IOS Press, Amsterdam, The Netherlands, The Netherlands, v. 12, n. 1, p. 25–31, 2004.

PAIVA, A.; PETRONETTO, F.; LEWINER, T.; TAVARES, G. Particle-based non-newtonian fluid animation for melting objects. In: *Computer Graphics and Image Processing, 2006. SIBGRAPI '06. 19th Brazilian Symposium on*. [S.l.: s.n.], 2006. p. 78–85. ISSN 1530-1834.

PIVELLO, Márcio Ricardo; LARRABIDE, Ignacio; FEIJóO, Raúl A. An algorithm for generating the centerline of a branched tubular geometry. In: *COBEM 2007: 19th International Congress of Mechanical Engineering*. [S.l.: s.n.], 2007.

PREMOZE, S.; TASDIZEN, T.; BIGLER, J.; LEFOHN, A.; WHITAKER, R. *Particle-based simulation of fluids*. 2003. Disponível em: <citeseer.ist-.psu.edu/premoze03particlebased.html>.

REEVES, W. T. Particle systems: a technique for modeling a class of fuzzy objects. *ACM Trans. Graph.*, ACM Press, New York, NY, USA, v. 2, n. 2, p. 91–108, 1983.

ROBB, Richard A.; HANSON, Dennis P.; CAMP, Jon J. Computer-aided surgery planning and rehearsal at mayo clinic. *Computer*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 29, n. 1, p. 39–47, January 1996. ISSN 0018-9162. Disponível em: <http://portal.acm.org/citation-.cfm?id=619005.620393>.

SHAO, Songdong; LO, Edmond Y. M. Incompressible sph method for simulating newtonian and non-newtonian flows with a free surface. *Advances in Water Resources*, v. 26, n. 7, p. 787 – 800, 2003. ISSN 0309-1708. Disponível em: <http://www.sciencedirect.com/science/article/B6VCF-48HXX4F-1/2/380162b8c29394cdf101a2dc3672598c>.

SHIN, Hayong; PARK, J.C.; CHOI, B.K.; CHUNG, Y.C.; RHEE, S. Efficient topology construction from triangle soup. In: . [S.l.: s.n.], 2004. p. 359–364.

SIGALOTTI, L.; DAZA, J.; DONOSO, A. Modelling free surface flows with smoothed particle hydrodynamics. *Condensed Matter Physics 9*, p. 359–366, 2006.

SILVA, André Ferreira Bem; NOBREGA, Tiago H. C.; CARVALHO, Diego Dias Bispo; INáCIO, Renan Teston; WANGENHEIM, Aldo Von. Framework for interactive medical imaging applications. In: NAVAUX, Philippe Olivier Alexandre (Ed.). *Proceedings of Colibri – Colloquium of Computation: Brazil / INRIA, Cooperations, Advances and Challenges*. Bento Gonçalves, Brazil: Sociedade Brasileira de Computação - SBC, 2009.

STAM, Jos. Stable fluids. In: *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1999. p. 121–128.

STAM, Jos. Interacting with smoke and fire in real time. *Commun. ACM*, ACM Press, New York, NY, USA, v. 43, n. 7, p. 76–83, 2000.

STAM, J. *Real-time fluid dynamics for games*. 2003. Disponível em: <citeseer.ist.psu.edu/stam03realtime.html>.

SUBRAMANYAN, Krishna; CIANCIBELLO, Les; DURGAN, Jacob. Abdominal aortic stent graft planning with automatically extracted vessel centerlines/cross-sections in multislice ct. In: *CARS*. [S.l.: s.n.], 2004. p. 183–188.

THRANE, Niels; SIMONSEN, Lars Ole. *A Comparison of Acceleration Structures for GPU Assisted Ray Tracing*. Dissertação (Mestrado) — University of Aarhus, 2005.

VIDAL, F. P.; BELLO, Fernando; BRODLIE, Ken W.; JOHN, Nigel W.; GOULD, Derek; PHILLIPS, R.; AVIS, N. J. Principles and applications of computer graphics in medicine. *Comput. Graph. Forum*, v. 25, n. 1, p. 113–137, 2006.

WAN, Ming; LIANG, Zhengrong; KE, Qi; HONG, Lichan; BITTER, I.; KAUFMAN, A. Automatic centerline extraction for virtual colonoscopy. *Medical Imaging, IEEE Transactions on*, v. 21, n. 12, p. 1450–1460, 2002. Disponível em: <http://dx.doi.org/10.1109/TMI.2002.806409>.

WEILER, Kevin. Edge-based data structures for solid modeling in curved-surface environments. *IEEE Comput. Graph. Appl.*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 5, n. 1, p. 21–40, 1985. ISSN 0272-1716.

WESTOVER, Lee. Interactive volume rendering. In: *VVS '89: Proceedings of the 1989 Chapel Hill workshop on Volume visualization*. New York, NY, USA: ACM, 1989. p. 9–16.

WILLIAMS, David; GRIMM, Soren; COTO, Ernesto; ROUDSARI, Abdul; HATZAKIS, Haralambos. Volumetric curved planar reformation for virtual endoscopy. *IEEE Transactions on Visualization and Computer Graphics*, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 14, n. 1, p. 109–119, 2008. ISSN 1077-2626.