UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Curso de Bacharelado em Ciência da Computação

Trabalho de Conclusão de Curso

Desenvolvendo uma Engine Física com simulação básica de Tecidos

Douglas Vanny Bento

Pelotas, 2014

Douglas Vanny Bento

Desenvolvendo uma Engine Física com simulação básica de Tecidos

Trabalho de Conclusão de Curso apresentado ao Centro de Desenvolvimento Tecnológico da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Bacharel em Ciência da Computação

Orientadora: Profa. Dra. Aline Brum Loreto

Pelotas, 2014

Universidade Federal de Pelotas / Sistema de Bibliotecas
Catalogação na Publicação

B478d Bento, Douglas Vanny

Desenvolvendo uma Engine Física com simulação básica de Tecidos / Douglas Vanny Bento ; Aline Brum Loreto, orientadora. — Pelotas, 2014.

40 f. : il.

Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) — Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 2014.

1. Simulação física. 2. 3D. 3. Partículas. 4. Tecidos. I. Loreto, Aline Brum, orient. II. Título.

CDD : 005

Elaborada por Aline Herbstrith Batista CRB: 10/1737

Douglas Vanny Bento

Desenvolvendo De Uma Engine Física Com Simulação Básica De Tecidos

Trabalho de Conclusão de Curso aprovado, como requisito parcial, para obtenção do grau de Bacharel em Ciência da Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas.

Data da Defesa: 09 de Dezembro de 2014

Banca Examinadora:

Profa. Aline Brum Loreto – Orientador

Prof. Marilton Sanchotene de Aguiar

Prof. Anderson Priebe Ferrugem

Dedico esse trabalho aos meus pais Arno e Gladis, que me apoiaram durante toda minha vida acadêmica.

AGRADECIMENTOS

Primeiramente eu gostaria de agradecer à toda minha família, que não apenas me apoiou, mas foi indispensável para a minha formação acadêmica. Agradeço também a meus colegas pelo apoio moral e pela ajuda que me foi dada durante esses 5 anos de curso nos estudos e execução de trabalhos. Em especial aos colegas Felipe Corrêa e Rafael Becker, que muito me aguentaram durante esse longo tempo. Mando também um abraço para todo o pessoal da DeMorgan e ao Walmor. Não devo deixar também de agradecer a todos meus professores, em especial à minha orientadora Aline Brum Loreto, pela paciência ao me conduzir ao lugar onde hoje estou, e a Anderson Ferrugem, que sempre demonstrou muita preocupação para com todos seus alunos. Finalmente, agradeço a Universidade Federal de Pelotas e a todos seus funcionários, que também foram indispensáveis nessa conquista. Agradeço também ao Governo Brasileiro pelo apoio sempre presente.

One of life's greatest blessings is the freedom to pursue one's goal.
— NYX (SHIN MEGAMI TENSEI: PERSONA 3)

RESUMO

BENTO, Douglas Vanny. **Desenvolvendo uma Engine Física com simulação básica de Tecidos**. 2014. 42 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2014.

A área de computação gráfica hoje em dia tem um grande potencial para evoluir, e esse potencial está de fato sendo aproveitado, como fica claro ao observarmos o avanço da tecnologia nas áreas aplicadas em jogos e filmes.

O presente trabalho descreve a teoria de um motor físico 3D, todo o processo de desenvolvimento do mesmo, descrevendo métodos e algoritmos utilizados, falando-se sobre *Ordinary Diferential Equation* e sua aplicação no Método de Euler; Além de um detalhamento maior ao descrever as características da simulação de tecidos, assim como seus desafios. É realizada uma breve análise do método da *Global Intersection Analysis* (GIA) para lidar com autocontato entre tecidos, se analiza um caso de contato que o GIA nao cobre e então é realizada uma proposta de método que é capaz de evitar 20% desses contatos indesejáveis.

Palavras-chave: Simulação Física, 3D, Partículas, Tecidos.

ABSTRACT

BENTO, Douglas Vanny. **Developing an Physics Engine with Basic Cloth Support**. 2014. 42 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2014.

Computer graphics today has great potential to evolve, and this potential is, in fact, being taken advantage of, as seen by observing the advancement of technology in the area applied in games and movies.

This paper describes the theory behind a 3D physical engine, as well as it's entire development process, detailing methods and algorithms used, talking about Ordinary diferential Equation and its application in the Euler method. Further detailment are made in describing the characteristics of the simulation of cloth, as well as its challenges. A brief analysis of the *Global Intersection Analysis* (GIA) to handle self-contact in cloth, analysing a case of contact not covered by GIA and then a new method proposal is made, being it able to avoid 20% of the undesireable contacts.

Keywords: Physics Simulation, 3D, Particles, Cloth.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ODE	Ordinary Diferential Equation
GIA	Global Intersection Analysis
CG	Computação Gráfica

SUMÁRIO

1 INTRODUÇÃO

Na área de computação gráfica a simulação física de tecidos é um dos tópicos mais populares de pesquisa. De forma geral, existem dois tipos de abordagens ao realizar tal simulação. A primeira usa modelos mais complexos porém fisicamente realistas, enquanto a segunda procura alcançar um resultado aceitável usando sistemas bem mais simplificados e econômicos. O primeiro é muito utilizado em animações de cinema devido ao realismo. O segundo já é utilizado muito em sistemas de tempo real como em jogos por exemplo, onde os cálculos são constantemente refeitos.

Toda essa simulação física ocorre dentro dos chamados motores físicos, também conhecidos como Engines 3D. Exemplos desses são a Cryengine, Unreal Engine e Unity. Elas são implementadas usando equações diferencias como no método implícito de Euler, onde a física é toda simulada em passos de tamanho fixo, tamanho que representa o balanço entre a precisão da simulação e o custo de processamento na máquina, fator muito importante no caso de simulações em tempo real (LIU et al., 2013).

Ao realizar-se uma animação em 3D em tempo real, entra-se em uma situação onde tem-se um tempo limite para renderizar um quadro da animação para que a fluidez não seja prejudicada. Desenhar todos os objetos da cena não é uma tarefa que exige muito processamento, a simulação da física envolve uma quantidade superior de cálculos a serem realizados. Em muitos casos os cálculos são altamente simplificados separando o modelo renderizado na tela do modelo usado para a simulação física de fato, como quando tem-se um objeto altamente complexo, como uma forma humanoide, porém sua área de colisão é vista como um simples conjunto de cilindros, paralelepípedos e esferas pelo motor gráfico.

Devido a natureza da simulação, o balanço entre desempenho e precisão é extremamente importante. É possível alcançar uma ótima precisão se deixarmos o desempenho de lado, porém isso vai contra o principal motivo do uso desse método para fazer a simulação, visto que o mesmo não rodaria em um sistema de tempo real nesse caso. A diversidade entre as plataformas pode revelar ser mais um desafio ao buscar por esse balanço, como por exemplo no caso de um jogo de computador.

Desde o início da pesquisa na área de simulação de tecidos, seus resultados tem sido aplicados nas mais diversas áreas, como em animações computacionais, jogos 3D, vestuários, etc. Em 1995, na animação *"Toy Story"* da Pixar, toda animação de tecido era feita manualmente, sem envolver nenhum tipo de simulação. Já em 1997 a Pixar lançou *"Geri's game"*, a sua primeira animação envolvendo simulação de tecidos (DUFILHO, 1998)

1.1 Trabalhos Relacionados

O primeiro artigo na área de simulação de tecidos baseado num modelo físico foi escrito pelo autor (TERZOPOULOS et al., 1987). Anos depois, em 1992, surgiu um artigo sugerindo a adição de forças de amortecimento com colisões internas (CARIGNAN et al., 1992). Em 1994 foi realizado o uso de sistemas descontínuos de partículas para simular um efeito de panejamento (BREEN; HOUSE; WOZNY, 1994).

A partir de então vieram vários outros artigos que trouxeram cada vez mais resultados satisfatórios. No ano de 1999 foi alcançado um modelo com resultados bastante satisfatórios e capazes de rodar em tempo real usando equações diferencias implícitas e matrizes hessianas (DESBRUN; SCHRÖDER; BARR, 1999). No ano 2000 foi então sugerido um modelo que melhorava tanto a sensação de realismo e a performance em si da execução em tempo real através de melhoras nas equações diferencias implícitas por não utilizar sistemas lineares (KANG et al., 2000).

1.2 Motivação

A pesquisa na área ainda é recente, principalmente quando se trata da simulação em tempo real. Devido a imaturidade desses modelos físicos simplificados, ainda existem muitos problemas de estabilidade na simulação, principalmente ao interagir com outros objetos na cena. O principal desafio desse problema é atingir um resultado satisfatório sem a aplicação de um modelo físico de alta precisão, mas a sua resolução pode permitir simulações com resultados mais fiéis a realidade e a construção de cenas mais complexas com tecidos dentro de um sistema em tempo real.

Outro motivo pelo qual a pesquisa na área é interessante é a redução da simplificação das áreas de colisão na cena. O grau de simplificação dessas regiões atualmente é bem alto, fato que pode ser observado principalmente em jogos de computador, onde frequentemente pode ser visto objetos colidindo quando visualmente eles ainda não teriam se tocado ou ter ocorrido intersecção entre objetos.

Nesses últimos anos o desafio está em simular os mais variados tipos de tecidos. Exemplos disso estão na simulação de tecidos mais rígidos, tecidos mais elásticos e na simulação de roupas (VOLINO et al., 2009; GOLDENTHAL et al., 2007). Os

resultados da pesquisa na área também são usados em áreas similares, como por exemplo a simulação de cabelo (SELLE; LENTINE; FEDKIW, 2008).

O que diferencia um tecido de outro objeto rígido qualquer é justamente a forma como ele reage a uma força ou impulso. Tecidos são objetos muito flexíveis, e isso torna a sua simulação muito mais complexa visto que começa-se a ter que nos preocupar com forças de tensão internas, que acabam resultando, por exemplo, em um efeito do peso no tecido em si mesmo ou a completa alteração de seu estado interno ao colidir com um corpo rígido ou até mesmo com outro tecido.

Atualmente a técnica de combate a autocolisões mais utilizada, a *Global Intersection Analysis* (GIA), cobre bem apenas parte dos casos, sendo ela ainda um problema em casos onde a colisão ocorre próximo às bordas do tecido. Acredita-se que é possivel melhorar os resustados atualmente atingidos com a aplicação de um método paralelo focado nesses casos que não funcionam bem com o GIA. O presente trabalho tem como objetivo estudar técnicas de modelação e simulação, com foco em especial em tecidos, realizando uma análise em cima do problema do autocontato. Os objetivos especificados são:

- Estudo das técnicas de modelação e simulação de tecidos;
- Propor uma solução para o problema do autocontato de tecido nas bordas.

1.3 Estrutura do Texto

Esse trabalho é composto por 4 capítulos, contando o capítulo introdutório atual.

No Capítulo 2 serão abordados conceitos necessários para uma melhor compreensão do trabalho realizado, como a simulação da física e a modelação de objetos no ambiente 3D, como tecidos.

No Capítulo 3 é detalhado todo o desenvolvimento do motor gráfico utilizado nos testes, assim como uma análise do método GIA para tratar as auto-colisões e a proposta de um método sem as restrições presentes no GIA.

Finalmente, o Capítulo 4 conclui o trabalho, demonstrando a visão do autor frente ao trabalho executado.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Tipos de objeto

O presente capítulo apresenta conceitos necessários para a compreensão do trabalho realizado, descrevendo a simulação física e a modelação de objetos no ambiente 3D, em tecidos.

Ao simular-se um ambiente tridimensional pode-se encontrar geralmente 2 tipos de objetos: partículas e modelos 3D.

Partículas são objetos compostos de massa, posição e velocidade linear, porém são desprovidos de rotação. Individualmente elas raramente tem um propósito muito importante, porém quando usadas em conjunto, no chamado sistema de partículas, podem realizar tarefas muito interessantes. Um exemplo disso é a animação Particle Dreams, de (SIMS, 1988), retratada na Figura 1.

Figura 1: Representação de uma cachoeira usando partículas na animação Particle Dreams

Modelos 3D, por sua vez, já possuem orientação e velocidade angular, além de massa, posição e velocidade linear. São usados para representar objetos com vo-

lume, como cubos e esferas, compostos por uma série de vértices ligados com outros vértices vizinhos, assim formando uma série de polígonos. Normalmente um modelo é formado por polígonos de 3 lados, porém é possível também usar 4 ou mais lados, apesar de isso tornar a modelação mais complexa. A Figura 2 retrata um modelo formado por polígonos simples.

Figura 2: Bola de futebol modelada usando polígonos de 5 e 6 lados

Vale reforçar que a forma como algo é modelado em um motor físico e a representação do mesmo ao ser renderizado não necessariamente estão fortemente relacionadas. Um exemplo disso é na modelação de fluidos, que pode ser representado utilizando um sistema de partículas com forças de atração e repulsão entre si, porém seu processo de renderização pode ser realizado separando-os em grupos de partículas baseando-se na proximidade entre elas, e então é apenas desenhado um modelo dinâmico usando as partículas como vértices e unindo arestas entre eles.

A modelação de tecidos é também realizada com o uso de um sistema de partículas em forma matricial bidimensional. Considerando um estado de repouso, o tecido pode ser imaginado como uma grade assim como representado na Figura 3. Cada encontro de arestas representa uma partícula, e o sistema todo representa o tecido completo. Para fazer com que esse conjunto de partículas se comporte como um tecido devem ser aplicadas forças entre as partículas. Considerando um modelo básico de tecido, existem geralmente 2 tipos de conexões em cada partícula, ditas de tipo A e tipo B, representadas na Figura 4. Conexões do tipo A, representadas por linhas e nodos azuis, conectam partículas com 2 arestas de distância na horizontal, vertical ou diagonal. Essas conexões são responsáveis pelas forças de resistência a elasticidade. No caso das conexões do tipo B, representadas por linhas e nodos vermelhos, ligam partículas com apenas uma aresta de distância, também na horizontal, vertical e diagonal. São nelas que se aplicam as forças de compressão e dobra.

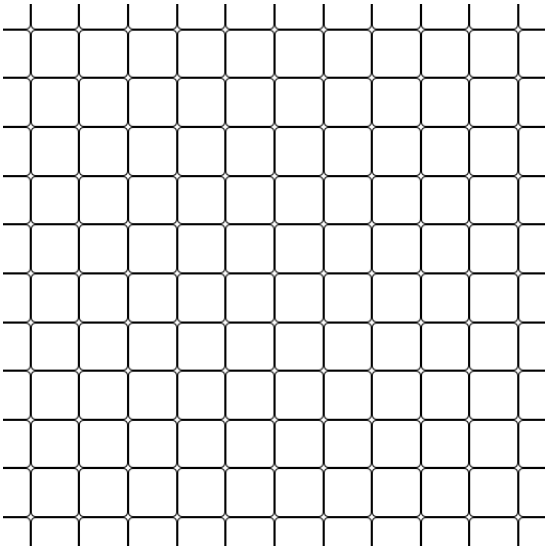

Figura 3: Representação de um sistema de partículas em formato de grade

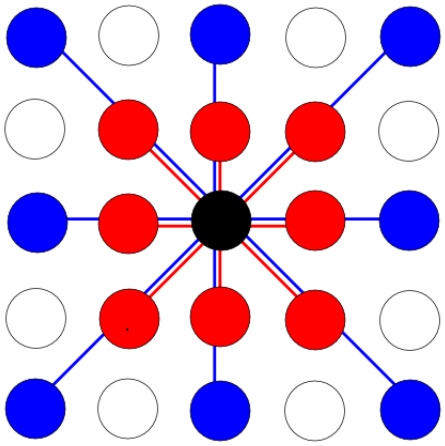

Figura 4: Forças que agem sobre uma partícula em uma malha

2.2 Simulação

A simulação física normalmente ocorre com o uso de Equações Diferenciais Ordinárias (ODE) solucionadas discretamente dando pequenos passos na direção da derivada. ODEs são conjuntos de funções e possivelmente suas respectivas derivadas que trabalham sempre em cima de uma única variável independente. No caso de simulações físicas, essa variável geralmente é o tempo atual da animação (EBERHARDT; WEBER; STRASSER, 1996). Um exemplo é a lei de Newton que define o comportamento da gravidade como aceleração.

$$\ddot{y} = \frac{f(y)}{m} = g \tag{1}$$

A equação acima pode ser representada no formado de um sistema ODE:

$$\begin{bmatrix} \dot{y} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} v \\ g \end{bmatrix} \tag{2}$$

Embasada na definição de derivada, foi criado o Método de Euler que realiza curtas iterações na direção da derivada no ponto para calcular a aplicação de $t + h$ a uma função x, onde $\dot{x} = f(x)$.

$$x(t + h) = x(t) + hf(x(t)) \tag{3}$$

A utilização desse método, porém, gera um acúmulo de erro proporcional ao tamanho do passo definido em h. Isso pode ser conferido na Figura 5, representando o uso do Método de Euler em uma função de translação. O uso da derivada discreta

faz com que, a cada iteração, o ponto se afaste cada vez mais do centro. Esse erro é estudado na chamada Série de Taylor.

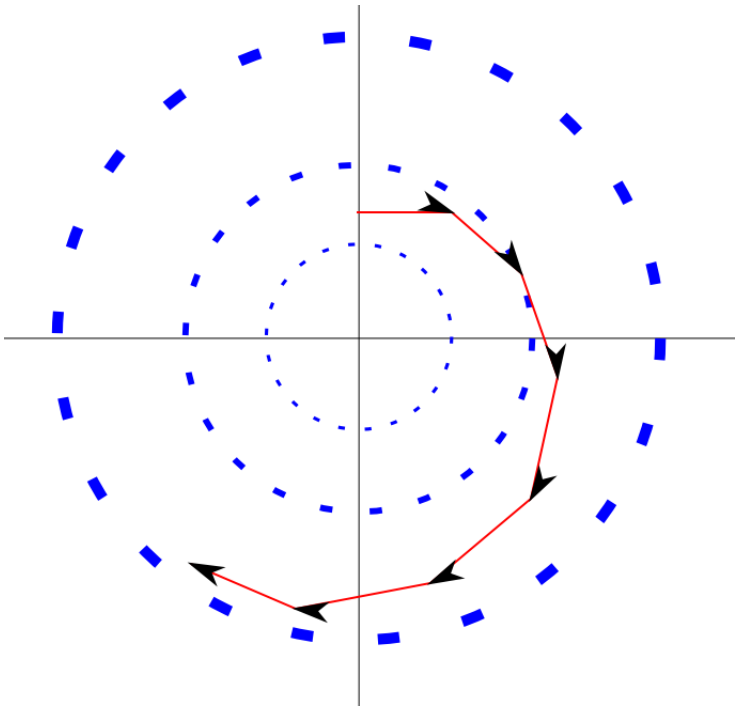

Figura 5: Representação do erro causado pelo uso do método de Euler

A Série de Taylor representa uma função como uma soma infinita de termos calculados em cima de suas derivadas em um único ponto, como indicada na equação 4. O Método de Euler representada em termos da Série de Taylor considera i de 0 a 1 apenas, portanto se diz que o erro acumulado em cada iteração está na ordem de $\mathcal{O}(h^2)$, representada na equação 5.

$$f(x) = \sum_{i=0}^{\infty} \frac{f^{(i)}(a)}{i!}(x-a)^i \tag{4}$$

$$x(t+h) = x(t) + hf(x(t)) + \mathcal{O}(h^2) \tag{5}$$

É possível reduzir a ordem do erro se quebrarmos as iterações realizadas usando o Método de Euler para adquirirmos as demais derivadas de x. Pode-se adquirir $\ddot{x}$ se dividirmos pela metade um passo de Euler aplicando os cálculos das equações de 6 a 8. Existem ainda métodos que diminuem ainda mais a ordem do erro, como o Método de Runge-Kutta que atinge $\mathcal{O}(h^5)$, porém deve-se considerar que, quando maior for a quantidade de cálculos realizados por passo, menos passos podem ser dados.

$$k_1 = hf(x(t)) \tag{6}$$

$$k_2 = hf(x(t) + k_1/2) \tag{7}$$

$$x(t+h) = x(t) + k_2 + \mathcal{O}(h^3) \tag{8}$$

2.3 Colisões

Conforme a base teórica escrita por (TERZOPOULOS et al., 1987), colisões são divididas em duas partes. Primeiramente tem-se a detecção de colisões, e então o cálculo da resposta a essa colisão no caso de o primeiro passo ter detectado algo.

O passo de detecção da colisão é responsável não apenas por checar se ocorreu ou não uma colisão, mas também detectar em que ponto ocorreu a colisão, tanto no espaço quando no tempo. Tendo em mãos a informação de em que ponto ocorreu a colisão, deve-se checar a necessidade de voltar no tempo, ou seja, recalcular o movimento com um passo menor até o ponto da colisão, calcular a reação a essa colisão, e então dar mais um passo com o tempo restante.

A forma como a colisão é detectada varia de objeto para objeto. Como no caso de tecidos estamos lidando com sistemas de partículas, deve-se testar a ocorrência de uma colisão para cada uma das partículas. Uma colisão de uma partícula com um objeto rígido ocorre quando essa partícula passa de dentro para fora do objeto ou vice versa. Lembrando que um objeto rígido é composto por vários polígonos, uma forma simples de detectar se um ponto se encontra dentro do objeto é traçando um segmento de reta da localização da partícula em t_1 até t_2 e ver se essa cruza com quaisquer polígonos do objeto. Essa detecção pode ser simplificada dependendo do tipo de objeto, como no caso de uma esfera, onde basta calcular a distância do centro dela ao ponto e comparar com o raio da mesma.

Tendo essas informações, é importante ainda realizar mais um teste, que é diferenciar a colisão de um contato estático. Contatos estáticos ocorrem quando a velocidade da partícula multiplicada pela normal do plano é próxima o suficiente de zero.

Devido ao fato de uma colisão ser algo instantâneo, a sua resposta deve ser calculada instantaneamente também, e isso é alcançado aplicando um impulso ao ponto com duração zero. Esse impulso ocorre fora da integração numérica, fazendo com que ela já não seja mais considerada no cálculo do passo seguinte.

2.3.1 Autocolisões

Dentro do tema de colisões, existe o desafio das autocolisões. Enquanto descobrir se um ponto está do lado de dentro ou de fora de um objeto é uma tarefa fácil, não é possível determinar se uma partícula está do lado certo ou errado do tecido (BARAFF; WITKIN; KASS, 2003). A Figura 6 representa o problema causado pela ausência de

detecção de autocolisões.

Figura 6: Representação do problema da auto colisão. A área vermelha representa a área que deve ficar para fora, e a verde a que deveria ficar para dentro.

Existem diversos métodos para se detectar e calcular as reações de uma auto colisão. (PROVOT, 1997) desenvolveu um método de ordem cúbica que era capaz de garantir a ausência de auto interferência de tecidos dinamicamente. Os autores (VOLINO; THALMANN, 2000) sugeriram um método que, apesar de não garantir consistência do sistema, era capaz de atingir resultados satisfatórios, onde uma heiristica aplicada a cada partícula decide para onde cada uma deve ir. Um método proposto pelo autores (BRIDSON; FEDKIW; ANDERSON, 2002) realiza essa tarefa evitando o uso de interações não físicas, assim evitando efeitos como o de flutuação no empilhamento de tecidos.

3 DESENVOLVIMENTO

3.1 Motor Gráfico

3.1.1 Comportamento geral

O presente capítulo apresenta o desenvolvimento do motor gráfico utilizado nos testes de autocolisão, assim como uma análise do método GIA para tratar as autocolisões. Por fim descreve o método proposto sem as restrições presentes no GIA.

A estrutura básica do código do motor gráfico envolve uma classe de alto nível onde são configuradas diversas variáveis de ambiente e adicionados fatores que participaram da simulação de fato, podendo ser eles objetos ou forças. Em um primeiro momento, foram apenas desenvolvidas estruturas mais básicas das quais uma simulação completa de tecidos depende. Para isso, criaram-se partículas e planos, sendo esses últimos de tamanho limitado. Uma vez adicionados eles no sistema, começa-se a executar iteração por iteração indefinidamente.

Uma vez iniciada a simulação, começam-se a rodas as iterações. Cada iteração é dividida nos seguintes passos.

1. Passo pré-iteração numérica
   Responsável pela aplicação de forças nos objetos, como gravidade ou vento. Essa etapa é também responsável pela simulação da dissipação de energia devido a resistência do ar. É possível também aplicar outros tipos de forças dependendo dos objetos, como forças de resistência impedindo que duas partículas se afastem ou se aproximem demais. A aplicação de uma força não acarreta em nenhuma modificação na velocidade do objeto em si, sendo isso realizado no passo de iteração numérica de fato.

2. Iteração numérica
   A iteração numérica é a verdadeira responsável pelo avanço da simulação em si. Ela tem como parâmetro uma variável h que indica a quantidade de tempo que será avançada na simulação. A grande vantagem dessa separação é que é possível aplicar quaisquer forças sem se preocupar com o quanto será avançado

na simulação em si. A forma como é realizado o avanço depende de qual *Ordinary Diferential Equation* (ODE) está sendo utilizada, assim como discutido na Seção 2.2. No caso do método de Euler, usam-se todas as forças calculadas no passo anterior e se aplica a Segunda Lei de Newton a elas, as dividindo pela sua massa e multiplicando por h, tendo seu resultado acumulado a velocidade atual do objeto. O mesmo método é utilizado para calcular a nova posição, que recebe um incremento da nova velocidade recém calculada, novamente a multiplicando por h.

3. Colisões
   Após calcular a nova posição de um objeto, ocorre a busca por colisões. A detecção de colisões é, indubitavelmente, o passo mais custoso da simulação, visto que deve-se comparar todas possíveis combinações de 2 objetos, assim tendo uma complexidade na ordem de $O(n^2)$. No caso de detectar-se uma colisão, existe o cálculo de sua reação. Esse caso é um onde ocorre uma quebra da regra, visto que deve-se refazer a iteração numérica em passos menores até encontrar o momento da colisão, e então, já fora da iteração, aplicam-se impulsos nos objetos envolvidos no contato. A grande diferença de um impulso para uma força é que ela ocorre em apenas um instante. No caso, ela é aplicada diretamente ao vetor velocidade do objeto em questão. Aplicado o impulso, retorna-se ao passo de iteração numérica novamente para completar a iteração em questão.

A detecção da colisão é realizada tendo um ponto de origem p_1 e um ponto de destino p_2. No caso de um plano, é necessário primeiro saber se o segmento de reta traçado de p_1 a p_2 passa de um lado do plano para o outro, e depois se calcula se o suposto ponto de contato está dentro do plano ou não. Para calcular de que lado do plano está um ponto, simplesmente se calcula $normal \cdot p_x$, sendo $normal$ o vetor normal do plano. Caso o resultado seja positivo, ele está acima, caso contrário, ele está abaixo do plano. Para descobrir o ponto onde ocorre a colisão cp, deve-se utilizar os cálculos anteriormente feitos. Calcular $normal \cdot p_x$ significa projetar o vetor p_x ao vetor $normal$, ou seja ver o tamanho de p_x paralelo ao eixo de $normal$. Subtraindo a projeção de $p2$ a $p1$, tem-se a distância nesse eixo, e dividindo $normal \cdot p_1$ a esse valor, sabe-se em que ponto do trajeto de p_1 a p_2 ocorreu a colisão. O cálculo para saber se o ponto cruza com o plano em si é similar. Além da normal, o plano tem mais 2 vetores $forward$ e $right$, indicando para quais lados ele cresce, podendo-se fazer uma analogia aos vetores unitários x e y. Basta comparar $forward \cdot cp$ com o tamanho em x e $right \cdot cp$ com o em y. O código abaixo demonstra o teste da colisão e a Figura 7 representa esses possíveis resultados do teste.

function DETECTACOLISÃO

▷ Primeiramente transforma-se os pontos de partida p_1 e chegada p_2 em posições relativas a posição do plano para facilitar nos cálculos

$p_1 \leftarrow p_1 - position$

$p_2 \leftarrow p_2 - position$

▷ Faz-se uma projeção dos pontos p_1 e p_2 no vetor normal da face onde se está testando a ocorrência de um contato

$p_1\_side \leftarrow normal \cdot p_1$

$p_2\_side \leftarrow normal \cdot p_2$

if SIGNAL(p_1_side) = SIGNAL(p_2_side) **then**

▷ O ponto de colisão é encontrado usando a razão da projeção de p_1 sobre a normal pela soma de ambas projeções

$cp \leftarrow p_1 - normal \cdot (p_1) * (p_2 - p_1)/(normal \cdot (p_2) - normal \cdot (p_1))$

▷ Tendo o ponto de possível colisão encontrado, conferimos se ele está dentro do plano comparando ele com os vetores frente e direita, lados para os quais o plano cresce, retornando imediatamente o resultado do teste

return $forward \cdot (cp - position) <= x \ \ \& \ \ right \cdot (cp - position) <= y$

else

return $false$

end if

end function

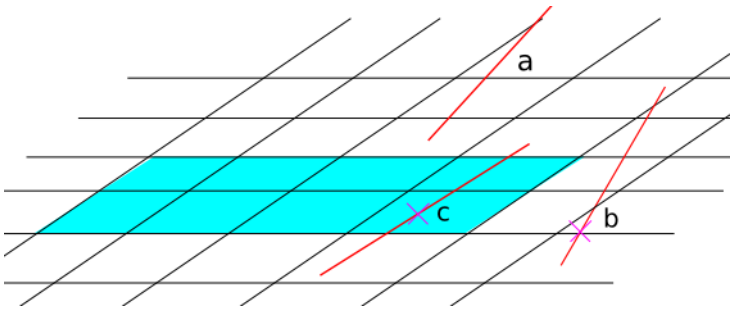

Figura 7: Representação dos possíveis casos ao testar uma colisão com um plano. O segmento a fica do mesmo lado do plano durante toda a trajetória, assim não ocorrendo contato. O segmento b troca de lado, porém sem cruzar ele, assim não ocorrendo contato também. O segmento c, porém, tem seu ponto de intersecção dentro do plano, assim ocorrendo o contato.

Detectada uma colisão, é importante descobrir em que ponto a colisão ocorreu. A forma como isso é realizado é similar a como se acha a raiz de uma função, onde a raiz seria o ponto de contato. Como já garantiu-se que existe uma colisão no trajeto sendo testado, usa-se o método da bisseção para encontrar tal ponto. Para isso, volta-se ao estado original da partícula antes do movimento, ponto a, avança-se $h/2$ para ter um ponto médio do movimento b, e usa-se o avanço anterior h já calculado em c. Caso

seja detectado uma colisão entre a e b, b vira o novo c e é calculado um novo b usando metade do tamanho da iteração. Caso não seja detectada nada, b vira o novo a e o processo é repetido da mesma forma, acumulando o valor de $h/2$. Vale acrescentar que é possível que o teste inicial de contato pode ter resultados errados, porém que isso não chega a causar nenhuma falha no comportamento do sistema. Isso é comum quando a partícula está em um movimento curvo, passando perto da borda de algo. Traçar um segmento de reta do ponto h_1 a h_2 pode detectar uma colisão, porém, ao aplicar o método da bisseção, tal colisão nunca é encontrada, fazendo com que ela "ache" o zero no final do movimento, assim dispensando qualquer cálculo a mais. A Figura 8 representa esse caso.

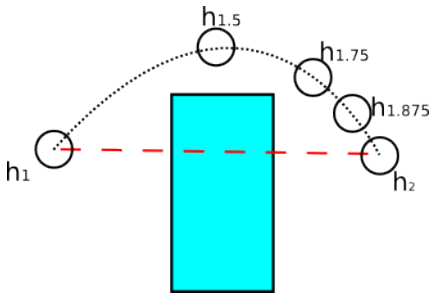

Figura 8: Erro na detecção inicial da colisão, demonstrando como que isso é corrigido automaticamente.

Tendo encontrado o momento exato onde a partícula entra em contato com o plano, começa-se o cálculo da reação. Primeiramente, é importante diferenciar contatos estáticos e contatos elásticos. Para isso, pega-se a velocidade da partícula e quebra-se em velocidade normal vn e velocidade tangencial vt, sendo a primeira paralela ao vetor normal do plano, e a segunda perpendicular ao mesmo. A diferenciação é realizada usando a magnitude do vetor de velocidade normal. Na teoria, tem-se um contato estático quando vn tem tamanho zero, porém isso causa problemas na integração numérica, visto que as chances de encontrar o valor 0 são extremamente baixas devido ao fato de tudo ser calculado usando variáveis de ponto flutuante, portanto deve-se comparar o valor com algum numero mágico bem baixo, sendo ele nesse caso 0.01. No caso de detectar-se um contato elástico, ou seja, termos um vn alto, deve-se simplesmente espelhar o vetor de velocidade pelo plano. Neste espelhamento, porém, entra uma variável importante da partícula, que é o quociente elástico k. A elasticidade da partícula é um valor geralmente no intervalo $(0:1)$ e é responsável pela dissipação de energia no momento da colisão. Finalmente, a nova velocidade se da pelo cálculo de:

$$vt + k * normal * (\frac{vn}{||vn||}) \tag{9}$$

Contatos estáticos não levam em consideração o quociente elástico da partícula.

Em compensação, deve-se passar a se preocupar com forças de atrito, e por causa delas o tratamento deve ser realizado de duas formas diferentes dependendo da velocidade tangencial da partícula. Em ambos os casos deve ocorrer uma compensação da velocidade normal da partícula, que apesar de muito baixa, ela não é zero, e por causa disso adicionasse uma levíssima força da direção da normal. Para a velocidade tangencial, faz-se um teste muito similar ao realizado em vn, que se resume a ver se ela é suficientemente grande ou não. Caso tenhamos vt com um valor alto, simplesmente multiplica-se ele pelo quociente de atrito da face onde a partícula reside, sendo ele também no intervalo $(0 : 1)$. Caso vt seja um valor muito baixo, ele simplesmente é removido da velocidade, assim atingindo um estado de repouso. A Figura 9 demonstra uma simulação com quatro partículas, explicitando suas respectivas trajetórias.

3.1.2 Tecidos

Tendo o comportamento das partículas completamente modelado, o foco passou a ser na modelação do tecido em si. O modelo mais utilizado atualmente é o chamado Mass-Spring Model, onde se tem um sistema grande de partículas com forças elásticas ligando elas. Essas forças impedem elas tanto de se afastarem demais quanto se aproximarem demais, como se cada partícula fosse um nó de uma costura que mantém o conjunto todo ligado. O que define o tecido, porém, não são as partículas em si, mas sim as formas que são geradas da ligação dessas partículas. Devido a facilidade que é trabalhar com polígonos de três lados quando na terceira dimensão, são formados triângulos ligando essas partículas de forma que toda área do tecido seja preenchida e não haja intersecção entre dois ou mais triângulos. A forma de um tecido varía muito, podendo ser simplesmente uma toalha, cuja modelação seria simplesmente uma série de partículas no formato de grade; mas também pode ser uma forma mais complexa, como em uma camisa, onde a distribuição dos nós e arestas se torna bem menos homogênea, ou como uma fronha, onde não existem bordas, cada nó tendo sempre vizinhos para todos os lados. Com o objetivo de facilitar a modelação, optou-se por fazer simplesmente um tecido homogêneo.

A modelação de um pedaço de tecido deve abordar de forma diferente uma série de fatores, devido ao fato de que não estamos trabalhando com um único objeto, mas sim com uma abstração que é um conjunto de partículas representando um único objeto maior. A primeira delas é a massa. Considerando que todo tecido tem uma massa m, esse valor é distribuído proporcionalmente entre os vários triângulos que o formam, e uma vez tendo esses valores, cada triângulo tem tua massa individual distribuída entre as partículas que o formam. A distribuição inicial dessas partículas também deve ser levado em conta, visto que ela deve trabalhar em conjunto com outro fator, que é a variável l, responsável pela distância entre cada duas partículas em um estado de repouso localizado. A ideia que o tecido sempre busque alcançar esse estado de re-

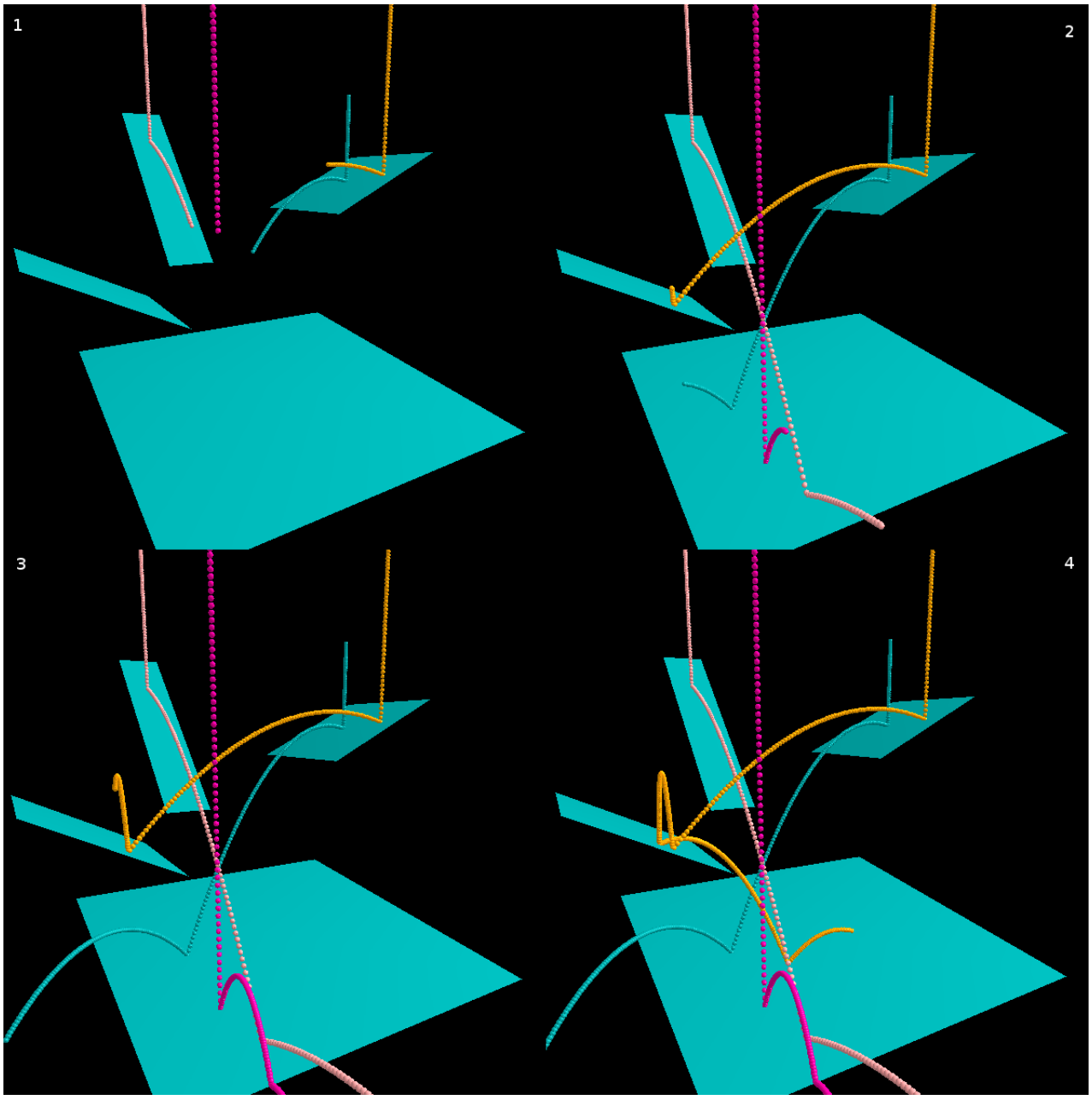

Figura 9: Simulando partículas caindo em planos imóveis.

pouso total. Quando tem-se, por exemplo, uma toalha parcialmente em cima de uma mesa, a parte que está para fora constantemente puxa a que está na mesa, enquanto o atrito da mesa a impede de rolar para fora. Ao passarmos esse comportamento para a simulação, teremos partículas na mesa e partículas fora dela, estando as primeiras em completo repouso e as demais sofrendo uma ação constante da gravidade. Ao botarmos na conta essas forças elásticas ligando as partículas, aquelas que estavam inicialmente em repouso deixarão de estar, para assim compensar o peso das demais. Caso seja o suficiente, a força da gravidade passara a ser completamente compensada pela força de atração das demais partículas, e assim o sistema inteiro entrará em repouso.

Existe, porém, um fator a mais que participa dessa conta, que é o quociente de elasticidade do tecido k_s. Ela representa a intensidade da tensão aplicada à aresta, que fara com que o tecido busque retornar ao seu estado de repouso. Um valor mais alto representaria um tecido mais firme, como talvez um pedaço de papel. O oposto já representaria um tecido mais mole, como um pedaço de plástico, desconsiderando fatores de deformidade na mola.

O passo de integração numérica considera cada item da simulação individualmente, independente de ele fazer parte de um sistema maior ou não, portanto é necessário calcular as forças para cada partícula antes disso. O cálculo dessa força ocorre da seguinte forma: para cada partícula p_i, são listadas todas as suas vizinhas p_j e, para cada uma delas, é calculada a distância x_{ij} entre elas. Faz-se uma comparação entre $||x_{ij}||$ e a distância de repouso l e, caso a diferença seja diferente de zero, calcula-se a força de reação dada na equação abaixo.

$$f = k_s(||x_{ij}|| - l)\frac{x_{ij}}{||x_{ij}||} \tag{10}$$

Essa força é aplicada em ambas direções, uma vez quando se está testando os vizinhos de x_i e outra nas de x_j.

Outro fator importante na simulação de tecidos é a quantidade de partículas a serem usadas. Similar a escolha da frequência de passos que o motor gráfico da por segundo, uma quantidade grande de partículas acaba resultando em uma animação mais suave e realista, porém também mais custosa, enquanto uma quantidade baixa pode tornar a simulação falha, porém bem mais barata de ser processada. As Figuras 10 e 11 representam um caso onde essa escolha tem um impacto na simulação. Na primeira imagem, está se usando poucas partículas, e isso acaba causando uma intersecção entre o bloco ciano e a ligação entre as partículas, problema que não ocorre de forma tão grave na segunda imagem, onde as partículas estão bem mais próximas umas das outras, apesar de ainda sim ocorrer a intersecção.

Nessa parte do desenvolvimento foi encontrado um problema não inicialmente pre-

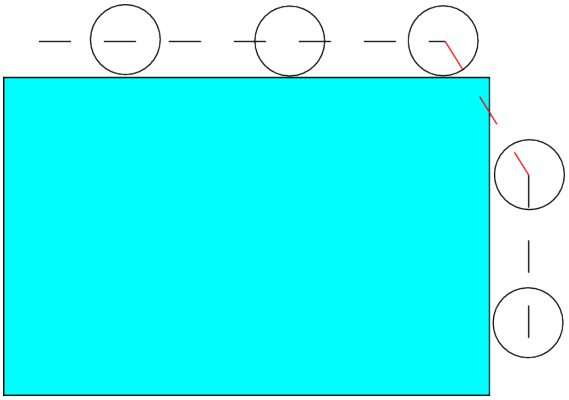

Figura 10: Tecido com uma grande intersecção na borda de um objeto

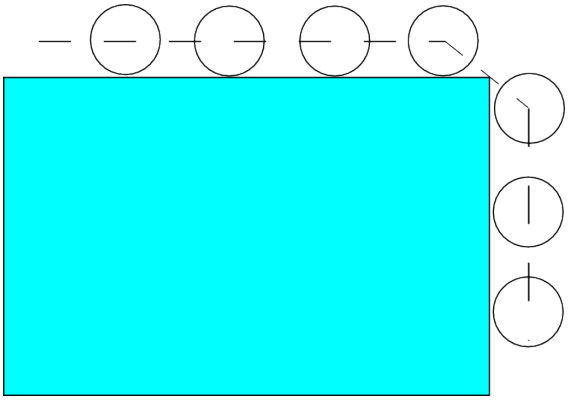

Figura 11: Tecido com uma pequena intersecção na borda de um objeto

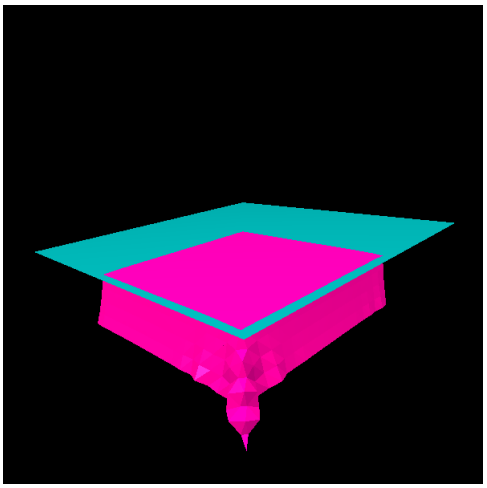

Figura 12: Tecido simulado com 121 partículas

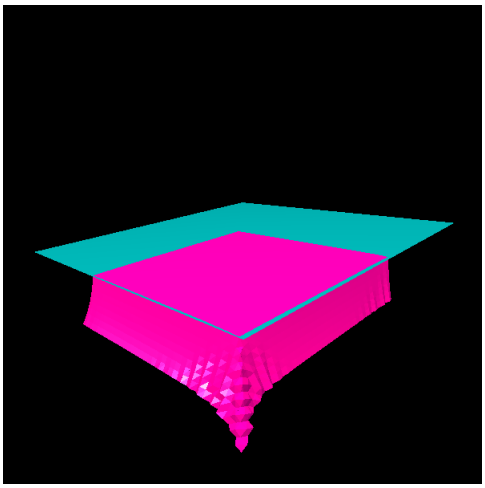

Figura 13: Tecido simulado com 484 partículas

visto. Ao simular o tecido, mantendo duas de suas pontas fixas, notou-se que o tecido entrou em um estado onde ele jamais alcançava o completo repouso, alternando infinitamente entre o estado inicial e um estado com ele esticado para baixo. Isso estava acontecendo devido a um acúmulo de energia potencial que não era dissipada em momento algum da simulação. Foi nesse ponto do desenvolvimento que a resistência do ar foi adicionada, assim permitindo que o tecido ficasse em repouso como demonstrado na Figura 14.

Ao resolver esse desafio, porém, notou-se que ele gerou uma falta de realismo à animação. Ao enxergar um tecido inicialmente paralelo ao plano z de lado, ele se tornava completamente invisível devido a completa ausência de forças perpendiculares a esse plano. Isso foi facilmente resolvido adicionando o fator externo do vento como representado na Figura 15, que está a se comportar de forma aleatória assim evitando também uma situação de repouso com o tecido inclinado.

É possível ainda identificar mais um problema, mas esse não está relacionado a

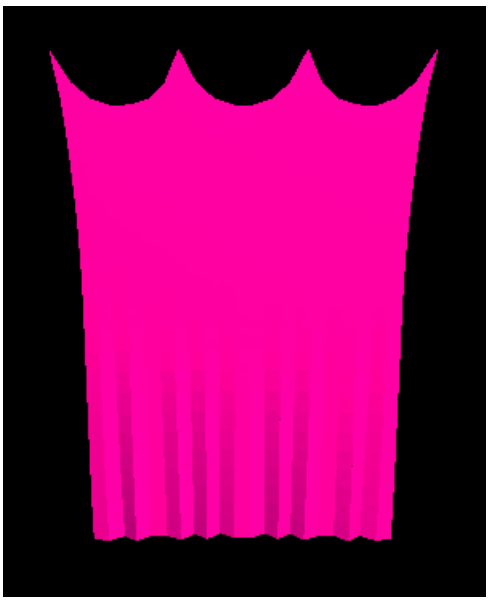

Figura 14: Tecido pendurado pelas pontas em repouso

Figura 15: Tecido pendurado pelas pontas sofrendo ação do vento

simulação, mas sim a renderização do tecido. Não foi aplicada nenhuma técnica especial para lidar com isso, porém seria possível fazer a renderização utilizando vários segmentos de curvas B-spline, assim como foi realizado por (WEIL, 1986).

3.1.3 Detectando e respondendo a autocolisões

Ao iniciar os testes na simulação de tecidos, algo que rapidamente se nota é que, em situações muito extremas, o movimento do tecido pode acabar fazendo com que ele passe a cruzar consigo mesmo, fazendo com que a parte de trás dele acabe indo para frente. Isso é o que chamamos de autocolisão. A detecção e tratamento dessas colisões ocorrem de forma muito diferente de quando estamos lidando com objetos tradicionais.

Primeiramente, detectar a colisão é um desafio que se torna muito maior do que simplesmente detectar se existe um cruzamento entre objetos. O maior problema é, após a detecção, saber para que lado deve ser a força de reação. Muitas das soluções, como a sugerida por (BRIDSON; FEDKIW; ANDERSON, 2002), envolvem consulta a um histórico de posições para saber de que lado cada partícula deve estar, porém isso se torna um problema quando o contato não pode ser desfeito imediatamente, geralmente devido a fatores externos ao próprio tecido. Se o tecido mantêm-se por muito tempo em uma posição errada, o método pode passar a acreditar que aquela é a posição certa devido ao histórico de posições anteriores. Um exemplo desse tipo de situação é quando o tecido está sendo usado como roupa para uma forma humana. A modelação de uma figura humana ocorre usando uma série de formas compridas e finas ligadas entre si por juntas, e quando se vai animar esse corpo, é comum que essas formas acabem se interseccionando, e caso exista um tecido envolvendo esses objetos, eles vão acabar sendo forçados a se manter em um estado extremo. A Figura 16 demonstra um caso como esse.

A detecção do cruzamento no tecido é um passo que tem em comum em grande maioria dos métodos. O tecido tem suas partículas interligadas formando uma malha de triângulos, assim como representado na Figura 17, e então se buscam colisões entre cada dois triângulos. Essa detecção é o passo mais pesado de todo o processo, e por isso é muito importante fazer ela da forma mais eficiente possível. Em uma primeira tentativa aplicou-se o teste de rotação para testar a intersecção. Tendo dois triângulos, o algoritmo usa as 3 arestas de um dos triângulos como segmentos de reta e procura ver se alguma delas cruza o segundo triângulo. Primeiramente se calcula se a origem e o destino da aresta estão de lados diferentes do plano formado pelo triângulo, e se calcula um possível ponto de colisão p_c. Tendo esse ponto, então, deve-se testar se ele está dentro ou fora do triângulo, e é dessa parte que o algoritmo tira o nome. Tendo os 3 vértices do segundo triângulo p_1, p_2 e p_3, criam-se vetores v_{12}, v_{23} e v_{31}, onde v_{ij} e o vetor de origem em p_i e destino em p_j. São puxados também vetores

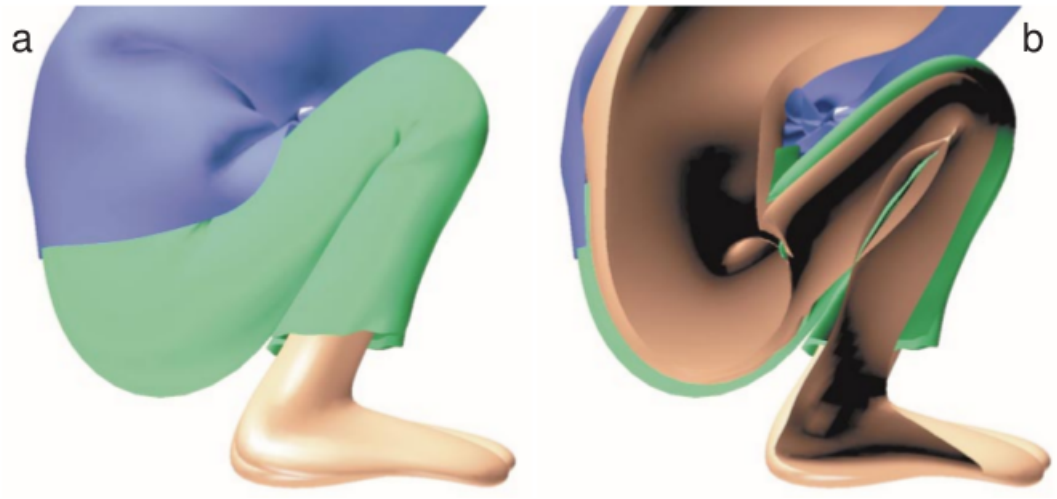

Figura 16: Tecido em uma situação forçada de auto colisão, sem ser capaz de se restituir imediatamente

v_{1c}, v_{2c} e v_{3c} dos 3 vértices para o ponto de colisão, e então se olha para o ângulo formado entre cada dois vértices v_{ij} de mesma origem. Caso todos estejam abertos no mesmo sentido, então o ponto se encontra dentro do triângulo, caso contrário, ele se encontra fora. A Figura 18 demonstra visualmente a aplicação desse método.

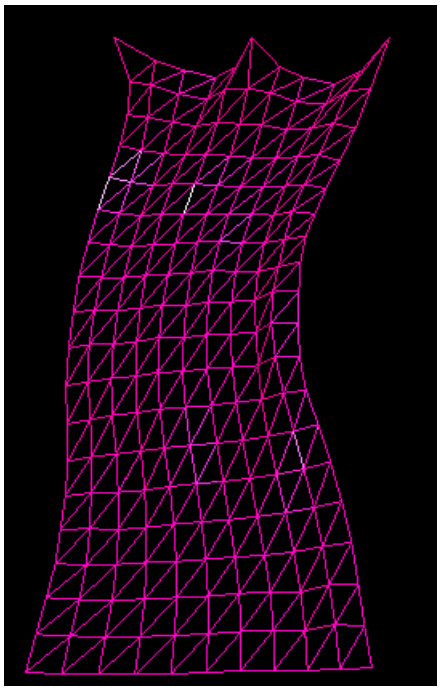

Figura 17: Representação de um tecido como uma malha de triângulos

O desempenho do algoritmo, porém, estava extremamente baixo, portanto se optou por utilizar o algoritmo de Möller-Trumbore. Uma diferença desse algoritmo para o algoritmo anterior é que ele não trabalha com segmentos de reta, mas sim com retas infinitas, devido a isso é necessário executar o algoritmo duas vezes para ter um

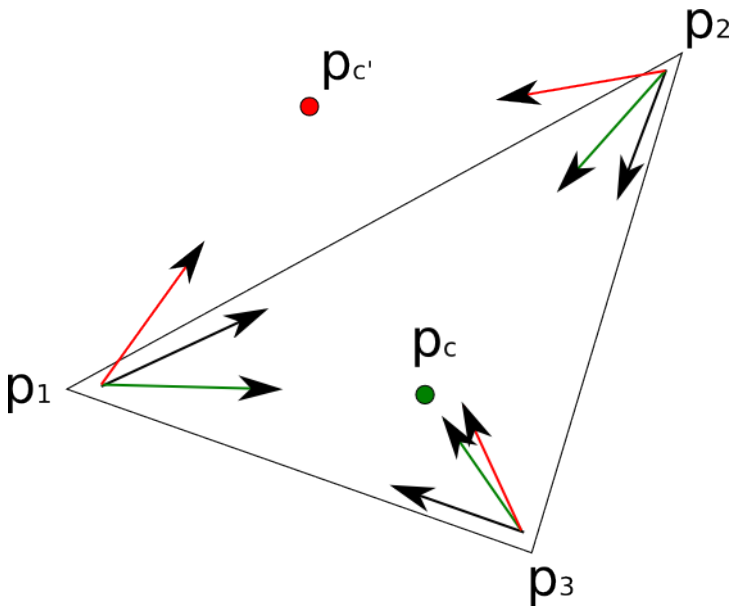

Figura 18: Aplicação do teste de rotação em um triângulo. O ponto p_c, que se encontra dentro do triângulo, tem seus vetores v_{ic}, indicados pela cor verde, estão todos abertos para o mesmo lado, quando comparados aos seus respectivos vértices v_{ij}, sendo o ângulo entre de mesmo sinal no domínio [-180° : 180°). Já no caso do ponto vermelho $p_{c'}$, os vetores $v_{2c'}$ e $v_{3c'}$ estão com uma abertura positiva, enquanto o $v_{1c'}$ tem uma abertura negativa, assim fazendo com que se detecte que ele está do lado de fora do triângulo.

resultado correto. Primeiramente se testa cada uma das arestas que formam o primeiro triângulo e se buscam colisões no segundo triângulo. Caso seja detectado uma colisão, é necessário ainda fazer o caminho contrário, buscar uma colisão de alguma das arestas do segundo triângulo no primeiro. Caso ambos testes detectem uma colisão, então tem-se uma colisão entre ambos triângulos. Apesar da aplicação dupla do algoritmo, atingiu-se resultados muito mais eficientes com eles.

Uma vez detectado onde ocorreram as colisões, deve-se começar a calcular as reações à colisão em si. Atualmente a aproximação mais utilizada é a GIA (*Global Intersection Analysis*) descrita por (BARAFF; WITKIN; KASS, 2003), que é capaz de desfazer cruzamentos entre tecidos em qualquer ponto da simulação, devido ao fato de utilizar apenas as informações do quadro atual da animação, dispensando quaisquer informações de histórico. Seu método funciona da seguinte forma. Sempre que ocorre uma intersecção, é executado um algoritmo de preenchimento visando mapear toda a área que cruzou a outra malha. Isso é realizado em ambas as regiões envolvidas no contato, e uma vez identificada as áreas, são aplicadas forças de atração entre ambas, de forma que o contato é desfeito. Sempre ao executar o algoritmo de preenchimento, porém, não se sabe ainda qual lado é o que cruzou o objeto, e qual não o cruzou. Devido a isso, o algoritmo é aplicado em ambas direções, tanto para fora quanto para dentro, e o que acabar primeiro é o que será considerado a parte que está do lado

errado, ou seja, ele sempre assume que o lado menor é o que está errado. A Figura 19 demonstra duas aplicações do algoritmo.

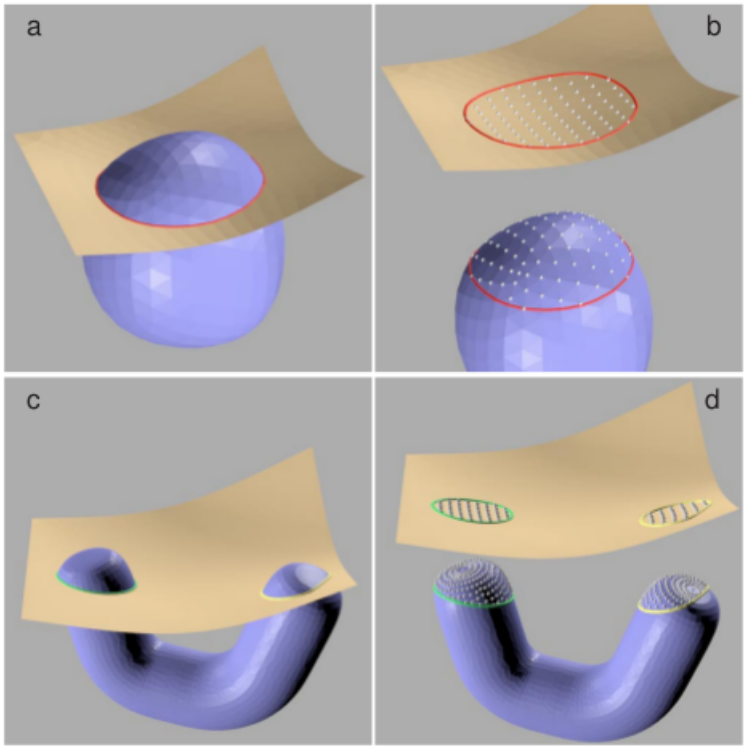

Figura 19: Representação das áreas de intersecção geradas pelo GIA

O GIA, porém, tem uma limitação. A aplicação dele envolve um algoritimo de preenchimento que divide o tecido em áreas “do lado certo” e áreas “do lado errado”, porém muitas vezes podem ocorrer colisões onde não é possível fazer essa divisão tão claramente. Isso ocorre, por exemplo, quando tem-se a borda de um tecido cruzando a si mesmo. Caso tente-se aplicar o GIA nesse caso, o preenchimento irá cobrir todo o tecido, tornando impossível calcular quaisquer forças de reação.

No motor implementado, porém, procurou-se desenvolver um método que fosse capaz de lidar com esses tipos de conflitos onde o GIA não se aplica. Se analisarmos esses casos, o fator que eles tem em comum é o fato de o cruzamento estar sempre ocorrendo próximo a uma borda, ou seja, uma possível reação seria fazer com que a área de contato buscasse se afastar dela. Devido ao estado imprevisível do tecido, porém, é aplicada uma força não na direção oposta à da borda, mas sim aos vizinhos que estão nesse caminho para a borda. Tendo isso em mente, então, pensou-se em 3 variações dessa ideia. O diagrama da Figura 20 busca facilitar a compreensão dos métodos.

1. A primeira variação envolve aplicar uma força às partículas que formam os triângulos nos quais foram detectados o contato no sentido oposto a da partícula vizinha mais próxima da borda p^{-}.

2. A segunda variação é aplicar uma força às partículas que formam os triângulos nos quais foram detectados o contato no sentido da partícula vizinha mais distante da borda p^+.

3. A última variação é aplicar uma força às partículas que formam os triângulos nos quais foram detectados o contato no sentido da posição média entre a partícula vizinha mais distante p^+ e a mais próxima da borda p^-.

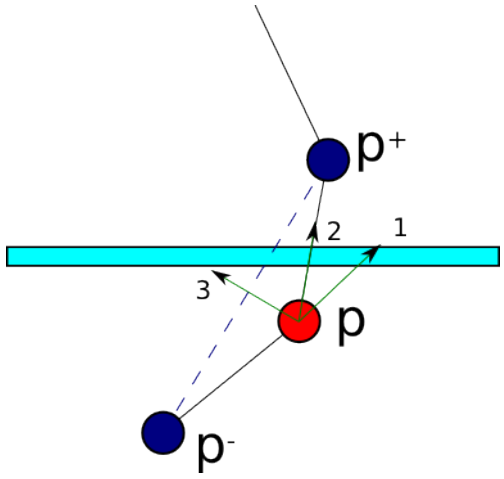

Figura 20: Possíveis forças a serem aplicadas numa partícula para desfazer um contato

Para cada um dos métodos, foram realizados testes com o mesmo estado inicial em todos, porém variando a elasticidade do tecido, variando ela entre 0.5 e 0.9, e o tamanho de cada passo dado na integração numérica, dando passos de 0.005, 0.01 e 0.02 segundos. Os testes foram realizados em 3 tipos de cenários diferentes, com diferentes intensidades de vento, com forças de 0.1, 0.09 e 0.11 Newton. Além das 3 abordagens, foram também aplicados os mesmos testes sem qualquer reação a contatos, assim totalizando 72 testes diferentes. A Figura 21 representa uma das execuções dos testes, já as Tabelas 1, 2 e 3 indicam as proporções de iterações nas quais foram detectados contatos sobre a quantidade total dos mesmos para cada um dos testes.

Fazendo uma análise dos resultados, é possível notar primeiramente que o método 2 foi completamente incapaz de melhorar os resultados em relação à completa ausência de reação, tendo uma ocorrência maior de colisões em 15 dos 18 testes individuais. O primeiro método não foi tão mal quanto o segundo, porém ele representa uma melhora muito baixa, tendo resultados negativos em 8 dos testes e positivos em 10 deles, fazendo com que ele não valha a pena devido ao fato de que a completa ausência de detecção de colisões já torna a simulação bem mais leve. O método 3, em compensação, já foi capaz de demonstrar resultados bem mais satisfatórios. Ele

teve resultados negativos em apenas dois dos testes, e a melhora dos resultados positivos é um número pouco baixo. No cenário com um vento de 0.1N atingiu-se um ganho médio de 10%, e nos demais ficou próximo de 20%.

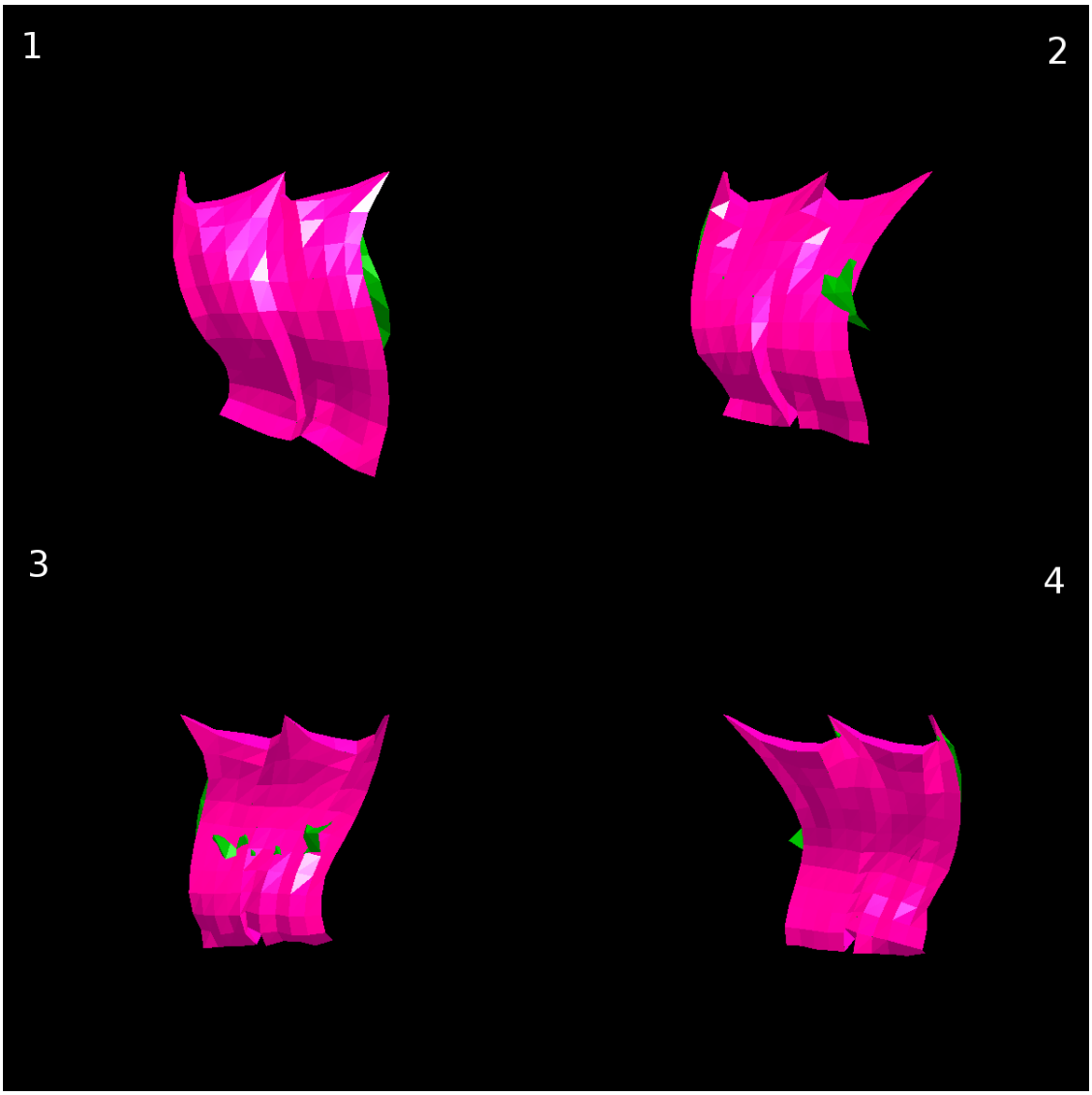

Figura 21: Representação da animação usada para testes. Nela o tecido é largado de um estato de repouso paralelo ao eixo z, preso em 6 pontos. Um vento forte e constante é aplicado a ele, o rodeando constantemente. Essa animação foi escolhida por provocar autocolisões próximas as bordas do tecido constantemente.

Tabela 1: Proporção de quadros com autocolisão com vento = 0.09

Passos por segundo	Elasticidade	Sem reação	Modo 1	Modo 2	Modo 3
50	0.9	0.18363	0.16766	0.19162	0.17365
50	0.5	0.15569	0.18563	0.15968	0.1018
100	0.9	0.19381	0.19281	0.20679	0.15784
100	0.5	0.15285	0.15884	0.14186	0.1019
200	0.9	0.169	0.1905	0.21	0.1625
200	0.5	0.1465	0.1595	0.1615	0.1015

Tabela 2: Proporção de quadros com autocolisão com vento = 0.1

Passos por segundo	Elasticidade	Sem reação	Modo 1	Modo 2	Modo 3
50	0.9	0.16367	0.16766	0.16766	0.15569
50	0.5	0.16367	0.16966	0.18962	0.17365
100	0.9	0.19181	0.18981	0.19381	0.14286
100	0.5	0.18382	0.16683	0.18581	0.14386
200	0.9	0.171	0.1555	0.1845	0.1585
200	0.5	0.1695	0.1675	0.207	0.1505

Tabela 3: Proporção de quadros com autocolisão com vento = 0.11

Passos por segundo	Elasticidade	Sem reação	Modo 1	Modo 2	Modo 3
50	0.9	0.17964	0.1976	0.23752	0.18762
50	0.5	0.25749	0.20758	0.2475	0.18164
100	0.9	0.24476	0.16583	0.26773	0.14985
100	0.5	0.22378	0.21678	0.23976	0.20579
200	0.9	0.195	0.201	0.198	0.174
200	0.5	0.246	0.2355	0.2215	0.2025

4 CONCLUSÃO

O trabalho tinha como objetivo o estudo das tecnicas de modelação e de simulação de tecidos, além da proposta de uma solução para problemas de autocontato proximo às bordas.

O início do processo envolveu a modelação da estrutura base do motor gráfico, onde foi necessário modelar não apenas regras físicas, mas também a integração numérica através da aplicação de ODEs. Ao adicionar-se suporte partículas, passou a ser necessário fazer um tratamento de colisões, que se mostrou um processo bem complexo. Essa dificuldade vem muito da dificuldade no processo de depuração do código, no qual tem-se um conjunto alto de variáveis sob as quais não tem-se controle nenhum a partir do momento que se inicializa a simulação. Outro fator é a complexidade das fórmulas envolvidas. Traduzir uma equação matemática para código não é algo complexo, mas a manipulação dessa equação é algo necessário no processo de depuração, e quando se é necessário fazer isso, é difícil de enxergar o que deve ser mudado para atingir um melhor resultado.

Ao se iniciarem os testes com tecidos o problema do tratamento das colisões foi algo que se mostrou rapidamente. Diferente de um objeto rígido, um tecido é algo muito mais maleável, o tornando muito mais instável, e essa instabilidade em sua forma acaba causando uma série de auto-contatos. Esse trabalho buscou uma forma de diminuir a gravidade desse problema, causado principalmente pela intersecção de partes do tecido próximo às bordas, área na qual ainda ocorrem instabilidades mesmo nos métodos mais atuais. Acredita-se que os resultados alcançados foram capazes de melhorar significativamente o resultado da simulação nesses casos, atingindo-se uma redução de até 20% na contagem de quadros da animação onde ocorrem conflitos. A capacidade de evolução cientifica na área é algo que se evidenciou durante o processo, sendo interessante realizar mais pesquisas na aplicação de métodos similares a esse em motores que utilizam o GIA ou outros métodos de simulação que dividam as mesmas dificuldades.

REFERÊNCIAS

BARAFF, D.; WITKIN, A.; KASS, M. Untangling cloth. In: ACM TRANSACTIONS ON GRAPHICS (TOG), 2003. **Anais...** [S.l.: s.n.], 2003. v.22, n.3, p.862–870.

BREEN, D. E.; HOUSE, D. H.; WOZNY, M. J. Predicting the drape of woven cloth using interacting particles. In: COMPUTER GRAPHICS AND INTERACTIVE TECHNIQUES, 21., 1994. **Proceedings...** [S.l.: s.n.], 1994. p.365–372.

BRIDSON, R.; FEDKIW, R.; ANDERSON, J. Robust treatment of collisions, contact and friction for cloth animation. In: ACM TRANSACTIONS ON GRAPHICS (TOG), 2002. **Anais...** [S.l.: s.n.], 2002. v.21, n.3, p.594–603.

CARIGNAN, M.; YANG, Y.; THALMANN, N. M.; THALMANN, D. Dressing animated synthetic actors with complex deformable clothes. In: ACM SIGGRAPH COMPUTER GRAPHICS, 1992. **Anais...** [S.l.: s.n.], 1992. v.26, n.2, p.99–104.

DESBRUN, M.; SCHRÖDER, P.; BARR, A. Interactive animation of structured deformable objects. In: GRAPHICS INTERFACE, 1999. **Anais...** [S.l.: s.n.], 1999. v.99, n.5, p.10.

DUFILHO, K. Geri's game. In: ACM SIGGRAPH 98 ELECTRONIC ART AND ANIMATION CATALOG, 1998. **Anais...** [S.l.: s.n.], 1998. p.131.

EBERHARDT, B.; WEBER, A.; STRASSER, W. A fast, flexible, particle-system model for cloth draping. **Computer Graphics and Applications, IEEE**, [S.l.], v.16, n.5, p.52–59, 1996.

GOLDENTHAL, R.; HARMON, D.; FATTAL, R.; BERCOVIER, M.; GRINSPUN, E. Efficient simulation of inextensible cloth. In: ACM TRANSACTIONS ON GRAPHICS (TOG), 2007. **Anais...** [S.l.: s.n.], 2007. v.26, n.3, p.49.

KANG, Y.-M.; CHOI, J.-H.; CHO, H.-G.; PARK, C.-J. Fast and stable animation of cloth with an approximated implicit method. In: COMPUTER GRAPHICS INTERNATIONAL, 2000. PROCEEDINGS, 2000. **Anais...** [S.l.: s.n.], 2000. p.247–255.

LIU, T.; BARGTEIL, A. W.; O'BRIEN, J. F.; KAVAN, L. Fast simulation of mass-spring systems. **ACM Transactions on Graphics (TOG)**, [S.l.], v.32, n.6, p.214, 2013.

PROVOT, X. **Collision and self-collision handling in cloth model dedicated to design garments**. [S.l.]: Springer, 1997.

SELLE, A.; LENTINE, M.; FEDKIW, R. A mass spring model for hair simulation. In: ACM TRANSACTIONS ON GRAPHICS (TOG), 2008. **Anais...** [S.l.: s.n.], 2008. v.27, n.3, p.64.

SIMS, K. Particle Dreams. **Video Siggraph Video Review**, [S.l.], v.38, p.39, 1988.

TERZOPOULOS, D.; PLATT, J.; BARR, A.; FLEISCHER, K. Elastically deformable models. In: ACM SIGGRAPH COMPUTER GRAPHICS, 1987. **Anais...** [S.l.: s.n.], 1987. v.21, n.4, p.205–214.

VOLINO, P.; MAGNENAT-THALMANN, N.; FAURE, F. et al. A simple approach to nonlinear tensile stiffness for accurate cloth simulation. **ACM Transactions on Graphics**, [S.l.], v.28, n.4, 2009.

VOLINO, P.; THALMANN, N. M. Accurate collision response on polygonal meshes. In: COMPUTER ANIMATION 2000. PROCEEDINGS, 2000. **Anais...** [S.l.: s.n.], 2000. p.154–163.

WEIL, J. The synthesis of cloth objects. **ACM Siggraph Computer Graphics**, [S.l.], v.20, n.4, p.49–54, 1986.