Modelagem baseada em física
Relatório de estudos

Aluno: Ricardo Bueno Cordeiro
rbc@ime.usp.br
Orientador: Carlos Hitoshi Morimoto
hitoshi@ime.usp.br

1 Introdução

A modelagem baseada em física é uma técnica, que vem sendo utilizada para criar animações realistas. Nessa técnica, gostaríamos de criar uma cena e fazer com que os objetos nela se comportem como se estivessem no mundo real. Para fazermos isso, precisaremos modelar as regras básicas da física, em nossos objetos e em nosso mundo virtual. Massa, velocidade e força são alguns dos tijolos que usaremos nessa modelagem.

Nesse estudo, aprendemos a base para se fazer essa modelagem. Estudamos o sistema de partículas, que é um dos mais simples de todos, estudamos corpos rígidos, a nossa área de estudo mais forte, e também estudos um pouco de corpos deformáveis. Inicialmente vamos lembrar um pouco de física e depois iremos abordar esses três sistemas, respectivamente.

2 Física

Nossa simulação consiste, basicamente, em movimentar objetos de forma realista, a partir de um estado inicial. Isso significa que a informação mais importante que temos, para um objeto, é sua posição. Essa posição é algo que varia com o tempo, e pode ser representada por uma função do tempo $x(t)$.

Da física, sabemos que $\dot{x}(t)$, onde $\dot{x}$ é a primeira derivada de x, é a velocidade, $v(t)$, que um objeto possui em um instante t. Também sabemos que $\ddot{x}(t)$ é a aceleração, $a(t)$, que esse objeto possui no instante t. Temos também que, $a(t) = \dot{v}(t)$.

No início da simulação, tempo t_0, nós temos, para cada objeto, seu estado inicial: $x(t_0)$ e $v(t_0)$. O que queremos é a posição, $x(t)$, para outros valores de t durante a simulação, infelizmente a função $x(t)$ não é conhecida. Porem, sabemos que existe uma relação entre a posição de um objeto e sua velocidade, assim como sabemos que existe uma relação entre a velocidade e a aceleração. Durante a simulação, a aceleração é algo que podemos calcular. Da física temos que $\vec{F} = M \cdot \vec{a}$, onde $\vec{F}$ é a força resultante de todas as forças agindo sobre o objeto e M é a massa do objeto. As forças são grandezas que temos, ou podemos calcular, a qualquer instante t. Elas podem ser a gravidade, uma mola ligando corpos, o contato entre dois objetos ou outras coisas.

Na maioria dos casos, temos a situação: possuímos $x(t_0)$ e queremos calcular $x(t)$, conhecendo $\dot{x}(t)$.

3 Equações diferenciais

No problema acima temos o que é conhecido como uma equação diferencial ordinária (EDO). Dado que nós temos um valor da função $x(t)$, para um valor de $t = t_0$, gostaríamos de calcular o valor de $x(t)$, para um $t = t_0 + \Delta t$.

Existem muitas técnicas numéricas disponíveis para se resolver esse problema, um exemplo é o método de Euler:

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \cdot \dot{x}(t_0)$$

Esse método consiste em dar um passo de tamanho Δt na direção da derivada. Apesar de ser bem simples, e portanto eficiente quando implementado, o método de Euler possui problemas: ele não é preciso e, eventualmente, leva a erros grandes, e também pode ser instável, e estourar o sistema para o infinito.

O núcleo de qualquer simulação baseada em física é o código para resolver as EDOs. Dependendo do método utilizado para se resolver essas equações, pode-se precisar de muito processamento. Existem casos, como em uma animação, em que se deseja ter uma precisão alta da movimentação dos objetos, isso pode forçar a necessidade de se resolver vários passos de uma EDO, por quadro de animação, e se o método utilizado fizer muitos cálculos por passo tomado, ele impossibilita uma animação em tempo real.

3.1 Descontinuidades

Um outro fator importante é que, para o sistema funcionar, essas equações precisam ser diferenciáveis, o que não acontece durante uma simulação intei-

ra. Quando dois objetos se chocam, suas trajetórias sofrem mudanças muito bruscas, causando uma quebra na continuidade de suas funções de deslocamento. No mundo real, quando isso acontece, forças agem entre esses dois corpos alterando suas velocidades e deslocamento de forma continua, porem essa alteração acontece em um espaço de tempo extremamente curto, dando a impressão de ser instantâneo. No mundo real, até o mais sólido dos objetos é deformável o que permite que as forças envolvidas na colisão possam agir por um curto período de tempo. Simular essa interação quase instantânea entre corpos, a cada choque que ocorre, é algo extremamente caro computacionalmente. Esse problema é tratado como uma descontinuidade.

Para solucionar esses casos, o solucionador de EDOs não pode ser utilizado. Inicialmente calcula-se o instante em que a descontinuidade acontece, depois os corpos envolvidos são tratadas isoladamente, alterando instantaneamente o seu estado, mudando sua velocidade, posição e outras coisas, depois se volta a utilizar o solucionador de EDOs para prosseguir a simulação.

4 Partículas

O sistema mais simples que podemos ter, em uma modelagem física, é o de partículas. Partículas são objetos que não possuem forma, que não ocupam um lugar no espaço. Elas são pontos no espaço que possuem massa, isso facilita muito uma simulação, pois elas não colidem entre si, não podem rotacionar e não mudam de forma.

Partículas possuem apenas as informações básicas necessárias para se ter uma simulação: posição, velocidade e massa. Por estarem no espaço 3D, a posição e a velocidade são vetores.

4.1 Simulando uma partícula

A cada instante t, uma partícula possui o seu estado, posição e velocidade, representado pelo vetor X, sua derivada é dada pelo vetor $\dot{X}$:

$$X = \begin{bmatrix} p_x \\ p_y \\ p_z \\ v_x \\ v_y \\ v_z \end{bmatrix} \text{ e } \dot{X} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ F_x/M \\ F_y/M \\ F_z/M \end{bmatrix},$$

onde p é o vetor da posição, v é o vetor da velocidade, a derivada achamos a partir do fato que $\dot{p} = v$ e que $\dot{v} = a$, e a aceleração a pode ser calculada por $F = M \cdot a$.

Sabendo um valor para X, no instante t atual, e conhecendo $\dot{X}$, no mesmo instante, podemos calcular X para um instante $t + \Delta t$. A cada passo da simulação atualizamos X

4.2 Modelo: massa + mola

Partículas, por si só, podem se utilizadas para simular gases ou fluídos, mas gostaríamos de poder modelar estruturas mais complexas com essa ferramenta simples. Isso pode ser atingindo unindo um conjunto de partículas por molas, possibilitando a criação de corpos mais complexos como: tecidos ou corpos rígidos.

Imagine que temos uma grade 2D de 10×10 nós. Cada nó é uma partícula, uma partícula é ligada às suas vizinhas por molas. Também podemos ter um cubo em que os vértices são partículas e as arestas são molas.

Quando essas estruturas colidem com algo, as molas aplicam forças nas partículas que elas unem, tentando manter a forma original, isso geralmente permite que o objeto se deforme um pouco, mas com o tempo ele volta a sua forma original.

Infelizmente, quando temos molas muito rígidas, molas que tem uma grande tendência à não mudar de tamanho, nossas EDOs se tornam instáveis, e o sistema pode estourar.

4.2.1 Molas

Uma mola é um objeto que não possui massa, posição ou velocidade. Ela é apenas um objeto que cria uma força entre duas partículas, de repulsão ou atração, que estão conectadas a suas extremidades. Molas aplicam forças, f_a e f_b entre duas partículas, a e b, seguindo a equação:

$$f_a = -\left[k_s(|\Delta x| - r) + k_d \frac{\Delta v \cdot \Delta x}{|\Delta x|}\right] \frac{\Delta x}{|\Delta x|},\ f_b = -f_a,$$

onde k_s e k_d são constantes da mola, Δx e Δv são as diferenças de posição e velocidade das partículas e r é o comprimento de descanso da mola.

5 Corpos rígidos

O tema principal de nossos estudos são os corpos rígidos. Ao contrario de partículas, corpos rígidos possuem forma e volume, e sua massa pode estar distribuída uniformemente em seu volume, ou não. As forças presentes no

sistema podem ser aplicadas a qualquer ponto de um corpo rígido, possivelmente gerando velocidade angular no corpo.

Quando nossos objetos possuem volume, começamos a ter problemas de colisão e de contato. Ambos os problemas tem um alto preço computacional para serem resolvidos, por isso existem muitas técnicas diferentes para tratá-los. Existem técnicas diferentes para cada tipo de modelagem de corpos.

5.1 Simulando um corpo rígido

Um corpo rígido, por possuir volume, precisa armazenar duas informações extras em seu estado, alem de sua posição e velocidade: orientação, representada pela matriz 3×3 R, e velocidade angular, representada pelo vetor ω, esse vetor representa o eixo em que a rotação ocorre, seguindo a regra da mão direita, e sua intensidade representa a velocidade da rotação em torno desse eixo.

Para simplificar as equações, o que guardamos é o momento linear, P do corpo, e o momento angular, L. Isso nos leva a termos o seguinte vetor de estado, $E(t)$, de um corpo rígido, em um instante t:

$$E(t) = \begin{bmatrix} x(t) \\ R(t) \\ P(t) \\ L(t) \end{bmatrix},$$

de onde podemos calcular $v(t) = \frac{P(t)}{M}$, $I(t) = R(t)I_{body}R(t)^T$ e $\omega(t) = I(t)^{-1}L(t)$. Onde $I(t)$ e I_{body} são matrizes que representam a distribuição da massa no volume do corpo.

A derivada de $E(t)$, $\dot{E}(t)$, segue abaixo:

$$\dot{E}(t) = \begin{bmatrix} v(t) \\ \omega(t)^* R(t) \\ F(t) \\ \tau(t) \end{bmatrix},$$

onde $\tau(t)$ é o torque que o objeto esta sofrendo e a operação $\omega(t)^*$ é a matriz:

$$\begin{pmatrix} 0 & -\omega_z(t) & \omega_y(t) \\ \omega_z(t) & 0 & -\omega_x(t) \\ -\omega_y(t) & \omega_x(t) & 0 \end{pmatrix}$$

Esses resultados, suas origens e o calculo das grandezas envolvidas podem ser achados em [4].

As equações acima permitem realizarmos uma simulação da movimentação de corpos rígidos. Mas ainda faltam muitas coisas para conseguirmos termos uma simulação realista de corpos rígidos, ainda precisamos ter uma forma de simular a interação entre os objetos que existem.

5.2 Pontos de contato

Considerando que nossos objetos são poliedros, todas as colisões podem ser reduzidas a colisões entre pontos dos objetos. Os casos que mais acontecem são o de vértice/plano ou aresta/aresta, nesses casos temos bem definidos os pontos, nos dois objetos, em que o contato ocorreu, e temos também uma normal, $\vec{n}$, associada a esse contato. Considerando que os corpos são chamados de A e B, essa normal está saindo de B.

Contato do tipo aresta/plano podem ser tratadas como dois contatos vértice/plano, utilizando as extremidades da intersecção como os pontos de contato. Quando temos contato plano/plano, tratamos eles como sendo vários contatos vértice/plano, onde os pontos utilizados podem ser os vértices do polígono formado pela região de contato entre os planos.

Os dois casos restantes são raros de se acontecer e não possuem uma normal bem definida: vértice/vértice e vértice/aresta. Ambas situações são, em geral, casos instantâneos e de curta duração, portanto sua existência não influencia muito no efeito final da simulação. Mas de qualquer forma devem ser tratados. O cálculo correto da força de contato nesse ponto é um problema NP-Completo, e para tratar esse caso aceita-se a perda de precisão. Escolhe-se um plano que passa pelo ponto de contato e, se possível, que não toca nenhuma outra parte dos corpos. Esse plano é tratado como se fosse uma superfície do objeto B, criando assim um contato do tipo vértice/plano.

A criação desses pontos de contato pode ser achada em [1].

No caso de termos objetos que possuem partes curvas, representadas por superfícies estritamente convexas e fechadas, podemos calcular pontos de contato através da função característica $\chi(t)$, que será mostrada na seção 5.4.

5.2.1 Teste de colisão

Na literatura temos muitos algoritmos disponíveis para se calcular intersecção entre objetos no espaço.

Um algoritmo proposto se aproveita do fato que, durante uma simulação, um teste de colisão entre os objetos é muito semelhante ao teste de colisão que foi feito anteriormente. O que o algoritmo faz é: para cada par de objetos testados, ele guarda uma testemunha do resultado do teste. No caso

de estarmos testando poliedros convexos, sabemos que dois não se tocam se existir um plano entre eles, de tal maneira que cada poliedro se encontra em um dos semi-espaços definido pelo plano, podemos dizer ainda que esse plano é uma face de um dos poliedros. Com sorte podemos dizer que, no próximo passo da simulação, esse mesmo plano pode ser utilizado como testemunha de que os objetos não se tocaram. Uma testemunha pode dizer que os objetos não se tocam, como também pode dizer que os objetos estão se chocando.

Essa técnica diminui, consideravelmente, o tempo gasto com o calculo de colisão entre todos os objetos da cena.

5.2.2 Detecção do instante de colisão

Quando estamos fazendo uma simulação, sempre tomamos passos discretos de tempo. Geralmente estamos em um instante t_0, que não possui contato ou interpenetração, e passamos para um instante t_1, onde temos interpenetração. Quando isso acontece, precisamos calcular o instante t_c da colisão. Se estivermos gerando uma animação, precisamos achar todos os instantes em que ocorre colisão, devemos tratar essas colisões, e depois avançarmos para o instante t_1.

Para resolver isso, [3] recomenda a utilização de um tipo de busca binária entre os instantes, mas essa busca utilização aproximações para se determinar o instante t_p que será usado para realizar a bipartição do intervalo. Para se calcular o valor de t_p existe o algoritmo de *regula falsa*, que consiste em fazer uma interpolação linear das distâncias, entre os dois objetos, diretamente na faixa entre t_0 e t_1. Também se pode usar o método de Newton, para se achar t_p, ele necessita de medidas de distância e de velocidade, entre os pontos de contato, e esses dois valores podem estão disponíveis nas funções características $\chi(t)$ e $\dot{\chi}(t)$.

5.3 Colisão

Uma situação que sempre encontraremos em uma simulação é a colisão entre dois objetos. Como dito na seção 3.1, quando isso acontece, precisamos parar o solucionador de EDOs, no instante em que ocorreu a colisão, e tratarmos a colisão como um caso especial.

Uma vez determinado os pontos de colisão entre objetos, aplica-se neles uma força instantânea, chamada de impulso. Um impulso é uma força com a unidade de momento linear, ele afeta diretamente o momento linear e o momento angular dos dois objetos envolvidos na colisão. Esse impulso, para o caso de uma colisão sem atrito entre os corpos, é sempre no sentido da normal $\vec{J} = j \cdot \vec{n}$. A matemática por trás do calculo de j é trabalhosa,

porem simples e baseia-se nas velocidades dos pontos de contato. Ela pode ser achada no capítulo de corpos rígidos de [4].

O cálculo de j permite um parâmetro configurável ϵ, $0 \leq \epsilon \leq 1$. Esse valor define o quanto perfeita é a colisão. Quando ele vale 1, a colisão é perfeitamente elástica e não existe perda de energia cinética, os objetos possivelmente irão se afastar. Mas para o valor 0, toda a energia cinética no sentido do impulso se perde, o que resulta nos objetos estarem, agora, em contato, possivelmente deslizando em suas superfícies.

5.4 Forças de contato

Um outro caso de corpos se tocando, é o caso em que a velocidade relativa entre os pontos de contato não leva à interpenetração ou ao afastamento deles, ou seja, eles estão em repouso ou deslizando. Nesse caso, surgem forças entre os corpos que apenas mantêm essa situação. Essa forças de contato são contínuas, atuando por todo o tempo em que os objetos se tocam, e portanto podem ser calculadas e deixadas para o solucionador de EDOs simular seu efeito no sistema.

Para um ponto de contato i, $1 \leq i \leq n$, cria-se uma função característica $\chi_i(t)$, que mede a posição relativa entre os dois corpos, também temos $\dot{\chi}_i(t)$ e $\ddot{\chi}_i(t)$, que representam, respectivamente, a velocidade e aceleração relativa entre os dois pontos.

Essa função característica representa se os corpos estão afastados, $\chi_i(t) > 0$, se eles estão se tocando, $\chi_i(t) = 0$, ou se eles estão interpenetrando, $\chi(t) < 0$. Para $\dot{\chi}_i(t)$ nós sabemos se os corpos estão se afastando, parados ou se aproximando, para $\ddot{\chi}_i(t)$ sabemos qual o efeito das forças atuais no sistema.

Para objetos poliédricos, podemos calcular a função diretamente como:

$$\chi_i(t) = \vec{n}_i \cdot (p_{i_a}(t) - p_{i_b}(t)),$$

onde, $\vec{n}_i$ é a normal no ponto de contato i, $p_{i_a}(t)$ e $p_{i_b}(t)$ são as posições dos pontos de colisão em função do tempo. A partir de $\chi_i(t)$ é fácil calcular $\dot{\chi}(t)$ e $\ddot{\chi}(t)$.

Em [2] podemos achar as construções para a função de característica e suas derivadas para objetos representados por superfícies estritamente convexas e fechadas. A complicação dessas funções é que os pontos comparados entre as superfícies podem mudar de posição na superfície com o passar do tempo.

A força de contato, $\vec{F}_i$, que queremos achar é no mesmo sentido que a normal do contato e pode ser escrita como $\vec{F}_i = f_i \cdot \vec{n}_i$.

Como estamos calculando forças de contato, podemos admitir que $\chi_i(t) = 0$, pois os corpos estão em contato nesse ponto, e que $\dot{\chi}_i(t) = 0$, pois os corpos não estão colidindo e não estão se afastando. As forças de contato tem que garantir que $\ddot{\chi}_i(t) \geq 0$. Outra restrição é que as forças apenas empurrem os corpos, levando a $f_i \geq 0$.

A aceleração relativa é uma função linear de f_i, $1 \leq i \leq n$, e pode ser escrita em forma de uma inequação linear que depende de todas as forças de contato no momento t. Isso leva a reescrever o problema de seguinte forma:

$$A\vec{f} = \vec{b} \text{ e } \vec{f} \geq 0,$$

onde A é uma matriz construída com a partir das inequações de $\ddot{\chi}_i(t)$, $\vec{f}$ é um vetor com a magnitude de todas as forças de contato no instante atual e $\vec{b}$ é o efeito de aceleração que todas as outras forças, presentes no sistema, impõe nos corpos. A criação desse sistema pode ser encontrada em [1], mas não foi possível achar uma definição muito concreta da criação da matriz A.

Uma outra restrição para o sistema é que $\ddot{\chi}_i(t) f_i(t) = 0$. Ela garante que para um ponto de contato i nós temos dois possíveis casos: ($\ddot{\chi}_i(t) > 0$ e $f_i = 0$) ou ($\ddot{\chi}_i(t) = 0$ e $f_i >= 0$), em um dos casos os pontos estão se afastando e portanto não existe força de contato, no outro caso a força existe para garantir que a aceleração seja nula. Isso leva ao sistema de restrições:

$$A\vec{f} = \vec{b},\ \vec{f} \geq 0 \text{ e } \vec{f}^T A\vec{f} - \vec{f}^T\vec{b}.$$

Em [1] temos a apresentação de uma heurística para se resolver o problema acima. O principal problema que essa heurística tenta resolver é o de achar o conjunto de pontos que vão se afastar, e portanto não possuem força de contato, e o conjunto de pontos que precisam da força de contato. Quando esses dois grupos são achados o problema se torna viável para solução através de programação linear.

Esse sistema se torna complexo a partir do momento em que representamos a ultima restrição em uma forma genérica. Em [2] é proposto um algoritmo que trabalha com a restrição em sua forma original $\ddot{\chi}_i(t) f_i(t) = 0$, e isso leva a uma solução polinomial para o problema em simulações sem atrito. No caso de um sistema com atrito estático, em que os objetos não estão deslizando, é provado que o algoritmo funciona, mas não é provado que ele termina a execução. No caso em que os objetos possuem atrito dinâmico, o algoritmo pode falhar em alguns pontos, o que exige a aplicação de um impulso entre os objetos envolvidos.

Esse problema pode ser alterado para poder suportar a existência de pontos de contato fixos, como em objetos articulados.

5.5 Colisão com contato

Uma outra situação que existe é um caso híbrido dos dois já citados. Temos alguns corpos em contato e um outro corpo se choca a esse grupo.

Existem dois tipos de soluções para esse problema: propagação ou simultânea. Na propagação, se calcula o impulso em cada um dos pontos de contato, um de cada vez. No modelo simultâneo, calcula-se o impulso em todos os pontos de contato de uma só vez e depois aplica ele aos objetos. Em alguns casos a propagação pode levar um tempo para convergir devido à troca de impulsos entre os corpos.

Os dois modelos geram resultados diferentes, quando envolvendo mais de um ponto de contato.

6 Corpos deformáveis

Corpos deformáveis são objetos que podem ter sua forma alterada pela aplicação de forças externas. Eles são relativamente mais complexos que corpos rígidos. As equações que regem seu comportamento são derivadas da mecânica de Lagrange. Essa modelagem permite unificar a descrição dos objetos com o seu comportamento físico. Quando nosso objeto é discretizado, o sistema é regido pela seguinte equação diferencial:

$$M\ddot{q}(t) + D\dot{q}(t) + Kq(t) = R,$$

onde R é vetor das forças externas agindo sobre os nós do objeto discretizado, M é a matriz de massa dos nós discretizados do objeto, D é a matriz de dissipação, K é a matriz de rigidez e $q(t)$ é a função de deslocamento dos nós do objeto. Cada um dos membros dessa equação representa a forma com que as forças se distribuem dentro do objeto.

Resolvendo com integração o sistema acima podemos simular um objeto deformável. A técnicas estudadas variam no processo de discretização dos objetos e na resolução da equação diferencial.

Uma técnica para a discretização de superfícies é mostrada em [5], ela pode ser utilizada para realizarmos simulações realistas de tecidos ou papéis. A modelagem permite que alguns parâmetros sejam independentes e fáceis de se configurar, como: facilidade para esticar o tecido, dobrá-lo ou entortá-lo.

Em [6], é apresentada uma técnica para representar objetos através de sólidos simples e fáceis de se representar como: esferas ou cilindros. A partir dessas formas, aplica-se uma transformação de deformação global no objeto e uma transformação de deformação local. Isso permite que uma grande

variedade de formas possam ser criadas. Esse mesmo artigo mostra como as equações do movimento ficam alterados por essa modelagem.

Em [7] e [8], são utilizadas superquadráticas para representar os sólidos, aplicando as mesmas deformações descritas acima, mas a solução das equações diferencias é feita de uma maneira diferente. Ela é alterada de tal forma que o deslocamento dos nós seja causada por vibrações. Para varias freqüências diferentes os objetos se deformam de uma maneira diferente. Em [7] temos uma boa descrição das superquadráticas, das deformações são implementadas nela, do método de discretização aplicado no objeto e da solução da equação de movimento acima. Em [8] temos um estudo que compara a perda e o ganha, em qualidade da simulação e custo computacional dela.

7 Conclusão

Com esse estudo, tivemos uma visão geral da modelagem baseada em física. Aprendemos os conceitos básicos para se fazer uma simulação, desde sistemas simples até sistemas complexos.

Apesar da matemática envolvida ser complexa, é algo que não requer conhecimentos avançados de cálculo ou se métodos numéricos para um implementação ser feita.

Alguns conceito foram entendidos, ou pelo menos suas conseqüências, porem suas origens ficaram duvidosas.

Os conceitos aprendidos sobre corpos rígidos permitem criarmos um simulador para gerar animações. Esse é o próximo passo planejado para uma pesquisa de corpos rígidos.

8 Referências

1. D. Baraff. *Analytical methods for dynamic simulation of non-penetrating rigid bodies.* Computer Graphics 23(3): 223-232, 1989.

2. D. Baraff. *Curved surfaces and coherence for non-penetrating rigid body simulation.* Computer Graphics 24(4): 19-28, 1990.

3. D. Baraff. *Fast contact force computation for nonpenetrating rigid bodies.* Computer Graphics Proceedings, Annual Conference Series: 23-34, 1994.

4. D. Baraff, A. Witkin. *Physically Based Modeling: Principles and Practice.* SIGGRAPH '97, 1997.

5. D. Terzopoulos, J. Platt, A. Barr, K. Fleischer. *Elastically deformable models.* Computer Graphics, 21(4): 205-214, 1987.

6. D. Terzopoulos, D. Metaxas. *Dynamic deformation of solid primitives with constraints.* Computer Graphics Proceedings, Annual Conference Series: 309-312, 1992.

7. I. Essa, S. Sclaroff, A. Pentland. *Physically-based modeling for graphics and vision.* Directions in Geometric Computing, 1993.

8. P. Chapman, D. Wills. *Towards a unified physical model for virtual environments.* 4^{th} UK VR-SIG Conference, 1997.