CS775 Open Project - Design Document : Soft Body Locomotion

Mayank Meghwanshi (110050012), Shivam H Prasad (110050041)

March 30, 2014

1 Objective

Modelling and Simulating soft body characters and their locomotion.[1][Siggraph 2012]

2 Implementation

2.1 Soft Body Simulation and Modelling

2.1.1 Tetrahedral Mesh

We'll use **Tetrahedral Mesh** to represent a soft body character. Tetrahedral mesh will consist of an array of vertices in 3D and an array of tetrahedral elements which contain an array of 4 vertex indices. We'll use **TETGEN** to generate tetrahedral mesh from input geometry of character. Tetgen takes as input all the vertices and faces of a body which then it converts to a tetrahedral mesh.

2.1.2 Finite Element Method

Our implementation will use Co-rotational linear FEM to simulate the soft body creature. FEM is used for computing the internal forces and stiffness matrix for the entire tetrahedral mesh. For computation of FEM we will use VEGA FEM library.

For FEM we first need to calculate element stiffness matrix in the rest configuration. Stiffness matrix depends upon the material of the object and relates internal displacement to the elastic forces. Its origins come from vibration theory of materials.

Now we need to calculate internal forces after some displacement from initial position of each element.

$$\hat{f}_e = -\hat{B}^T \hat{D} \hat{B} (\hat{p} - \hat{R}\hat{x})$$

where, $\hat{f}_e$ is internal elastic force, p is the state of creature, $\hat{x}$ indicates the nodal position in the rest shape and $\hat{R}$ transforms the element from reference coordinates to deformed coordinates. $\hat{B}$ is the strain-displacement matrix in the deformed coordinates and $\hat{D}$ is the stress-strain matrix. Both $\hat{B}$ and $\hat{D}$ depends upon material properties and displacement from rest position.

2.1.3 Damping Force

For calculation of damping force we generate a global stiffness matrix using individual stiffness matrices calculated by FEM method above. Damping force is then calculated using

$$f_d = -(\mu M + \lambda K)\dot{p}$$

where M is the mass matrix, K is global stiffness matrix and $\dot{p}$ is rate of change of state of creature. μ and λ are the damping coefficients set to 0 and 0.2 respectively which makes system lightly over-damped.

2.1.4 Muscle Force

Muscle will be implemented as list of segments which are represented as vectors in 3D space with compressible length. Each segment acts as a spring with compressible length with some stiffness k due to which it exerts force on nearby FEM elements.

$$f = k(l_d - l)$$

where l_d is the desired length and l is the current length.

Then we transform the muscle force using $\hat{R}$ which then gives a force vector aligned in direction of muscle contraction at deformed position.

Each FEM element is affected by multiple muscles, so we take a weighted sum of all the muscle forces that influence a particular FEM element. Weight from each muscle depend upon distance of element from the muscle fibre.

This muscle force on each element is then used do calculate muscle force on each of element's faces using area weighted face normals. These forces are then used to calculate muscle force on each vertex.

$$f_m = A(l_d - l)$$

where A is used to express the relation between muscle force on each vertex and muscle contraction.

2.1.5 Force Equation

Dynamic equation of motion which guides the character,

$$M\ddot{p} = f_x + f_e + f_d + f_m$$

where f_x, f_e, f_d, f_m are external, elastic, damping and muscle forces respectively. After substituting computed forces equation becomes,

$$M\ddot{p} = f_x - K(p - Rx) - (\mu M + \lambda K)\dot{p} + A(l_d - l)$$

Using Δt time-step for integration we rewrite above equation,

$$\tilde{M}\dot{p}^{n+1} = M\dot{p}^n + \Delta t(f_x^n - K(p^n - Rx^n) + A(l_d - l^n))$$
$$p^{n+1} = p^n + \Delta t\dot{p}^{n+1}$$

where n indicates discretized time index and $\tilde{M} = M + \Delta t(\mu M + \lambda K) + \Delta t^2 K$.

2.2 Locomotion Control

2.2.1 Optimization

To control the locomotion we formulate an optimization problem with the objective function G. Through this optimization we have to find out desired length of muscle fibre given current position to calculate next position of each node. This optimization is also used to calculate contact forces such as $f_\perp$ and $f_\parallel$. We use three control mechanisms to solve optimization problem.

Momentum Control.

$$G(l_d, f_\perp, f_\parallel) = \|\dot{L}(\dot{p}^{n+1}, \dot{p}^n) - \bar{\dot{L}}\|^2$$

where, $\dot{L}(\dot{p}^{n+1}, \dot{p}^n)$ is the sum of change in linear momentum of each element in mesh on going from current position to next position and $\bar{\dot{L}}$ is the change of linear momentum of Center of Mass of the character.

$$G(l_d, f_\perp, f_\parallel) = \|\dot{H}(\dot{p}^{n+1}, \dot{p}^n, \dot{p}^n) - \bar{\dot{H}}\|^2$$

where, $\dot{H}(\dot{p}^{n+1}, \dot{p}^n, \dot{p}^n)$ is the sum of change in angular momentum of each element in mesh on going from current position to next position and $\bar{\dot{H}}$ is the desired change of angular momentum.

Base Control.

$$G(l_d, f_\perp, f_\parallel) = \|\dot{A}(\dot{p}^{n+1}, \dot{p}^n) - \bar{\dot{A}}\|^2$$

where, $\dot{A}(\dot{p}^{n+1}, \dot{p}^n)$ is the change in projected contact area due to change in position of object and $\bar{\dot{A}}$ is the desired change in contact area.

2.2.2 QPCC

To solve the above constraints which form a quadratic optimization problem with complementary constraints, we will implement the graph expansion algorithm mentioned in the paper. We will use the pseudo-code provided in supplementary document with the paper which solves a QPCC by converting it to small QP problems. For solving QP problems the paper assumes a solver. For that we will be using MOSEK QP solver.

2.3 Rendering

We will render the tetrahedral mesh at every time step using OpenGL. We will be using VEGA FEM library's implementation for rendering, If time permits we will try to make a renderer ourselves.

3 Desired Results

From the implementation of QPCC we will get a values of muscle segment lengths and contact forces values. Using these we can modify the position of character at each time step. Thus we will expect to get balance, sliding, jumping and rolling motion of a character.

References

[1] Jie Tan, Greg Turk, and C. Karen Liu. Soft body locomotion. *ACM Trans. Graph.*, 31(4):26:1–26:11, July 2012.