Modeling the World as a Mesh of Springs

Solving for Motion where there is a Spring

Modeling a String as a Group of Masses Connected by Springs

Computing Forces in 1D

From the Physics Notes: What does a Second Order solution look like in a Program?

\[ F_{spring} = -k(y - D_b) \]

\[ \Delta V = \sum_{m} F \Delta t = \frac{-W - k(y - D_b)}{m} \Delta t \]

\[ F_{i-1} = k(D_{i-1} - D_b) \]

\[ F_x = \text{Mass} \times \text{Gravity} \]

\[ F_{i+1} = k(D_{i+1} - D_b) \]

\[ X: \Delta x = x_{i+1} - x_{i-1} \]

\[ Y: \Delta y = y_{i+1} - y_{i-1} \]

\[ X: \Delta x = x_{i+1} - x_{i-1} \]

\[ Y: \Delta y = y_{i+1} - y_{i-1} \]

\[ X: \Delta x = x_{i+1} - x_{i-1} \]

\[ Y: \Delta y = y_{i+1} - y_{i-1} \]

\[ X: \Delta x = x_{i+1} - x_{i-1} \]

\[ Y: \Delta y = y_{i+1} - y_{i-1} \]

\[ X: \Delta x = x_{i+1} - x_{i-1} \]

\[ Y: \Delta y = y_{i+1} - y_{i-1} \]

\[ X: \Delta x = x_{i+1} - x_{i-1} \]

\[ Y: \Delta y = y_{i+1} - y_{i-1} \]

\[ X: \Delta x = x_{i+1} - x_{i-1} \]

\[ Y: \Delta y = y_{i+1} - y_{i-1} \]

\[ X: \Delta x = x_{i+1} - x_{i-1} \]

\[ Y: \Delta y = y_{i+1} - y_{i-1} \]
Solve for Each State as a Whole, not as Individual Links:

Correct Second Order solution:

GetOneBodyDerivs( Links, i, &vx1[i], &vy1[i], &ax1[i], &ay1[i] );

for( int i = 0; i < NUMLINKS; i++ )
{
    Links[i].y   = Links[i].y    + DT * ( vy1[i] + vy2[i] ) / 2.;
    Links[i].x   = Links[i].x    + DT * ( vx1[i] + vx2[i] ) / 2.;
    Links[i].vy = Links[i].vy + DT * ( ay1[i] + ay2[i] ) / 2.;
    Links[i].vx = Links[i].vx + DT * ( ax1[i] + ax2[i] ) / 2.;
}

GetOneBodyDerivs( TmpLinks, i,  &vx2[i], &vy2[i], &ax2[i], &ay2[i] );

for( int i = 0; i < NUMLINKS; i++ )
{
    TmpLinks[i].y   = Links[i].y    + DT * vy1[i];
    TmpLinks[i].x   = Links[i].x    + DT * vx1[i];
    TmpLinks[i].vy = Links[i].vy + DT * ay1[i];
    TmpLinks[i].vx = Links[i].vx + DT * ax1[i];
}

Solve for Each State as a Whole, not as Individual Links:

GetLinkVelAcc( ), I

GetLinkVelAcc( ), II

Changing Variables on-the-fly in the String Project
Simulating a String

Placing a Physical Barrier in the Scene

First Order Instability

Vector from circle center to the lumped mass

If the lumped mass is inside the circle

Unit vector from circle center to the lumped mass

Push the lumped mass from inside the circle to the circle's surface

Keep just the tangential velocity

Less Damping

Modeling Cloth

\[
\begin{align*}
\text{if (DoCircle)} & \{ \\
\quad \text{for (i = 0; i < NUMLINKS; i++)} & \\
\quad \text{float dx = Links[i].x - CIRCX; } & \\
\quad \text{float dy = Links[i].y - CIRCY; } & \\
\quad \text{float rsqd = dx*dx + dy*dy; } & \\
\quad \text{if (rsqd < CIRCR*CIRCR)} & \\
\quad \quad \text{float r = sqrt(rsqd);} & \\
\quad \quad \text{dx /= r; } & \\
\quad \quad \text{dy /= r;} & \\
\quad \quad \text{Links[i].x = CIRCX + CIRCR * dx;} & \\
\quad \quad \text{Links[i].y = CIRCY + CIRCR * dy;} & \\
\quad \quad \text{Links[i].vx *= dy;} & \\
\quad \quad \text{Links[i].vy *= -dx;} & \\
\quad \}
\end{align*}
\]
We Can Also use this Same Method to Model and Analyze Rigid Objects