|  |
| --- |
|  |

Samson Mano

saminnx@gmail.com

<https://sites.google.com/site/samsoninfinite/>

[https://github.com/Samson-Mano](%20https://github.com/Samson-Mano)

Constraint dynamics

Formulation of XPBD for compliant constraint dynamics

# Constraint dynamics

Constrained dynamics is a fundamental concept in physics simulations, particularly in the realms of computer graphics where it is employed to realistically model the behavior of both rigid and soft bodies. Constrained dynamics refers to the study and simulation of physical systems whose motion is restricted in some way by constraints. A constraint can be anything that limits the freedom of a system to move, such as joints, springs, or collision with other bodies.

## Position – Based Dynamics (PBD)

David Baraff, Andrew Witkin: Robotics Institute Carnegie Mellon University

The equation of a general systems is,

Let the global force vector is given by Q and the constraint force is given by ,

The global equation governing the system becomes,

is the constraint reaction force.

The global notation for the constraint equation w.r.t q is given by C(q). By applying chain rule,

The matrix below is called the Jacobian of C

Differentiating above w.r.t time again gives

Substituting for in the above equation gives,

Setting to zero gives the below expression,

Or

Using the method of Lagrange multipliers, the direction of the optimal global constraint force is given by,

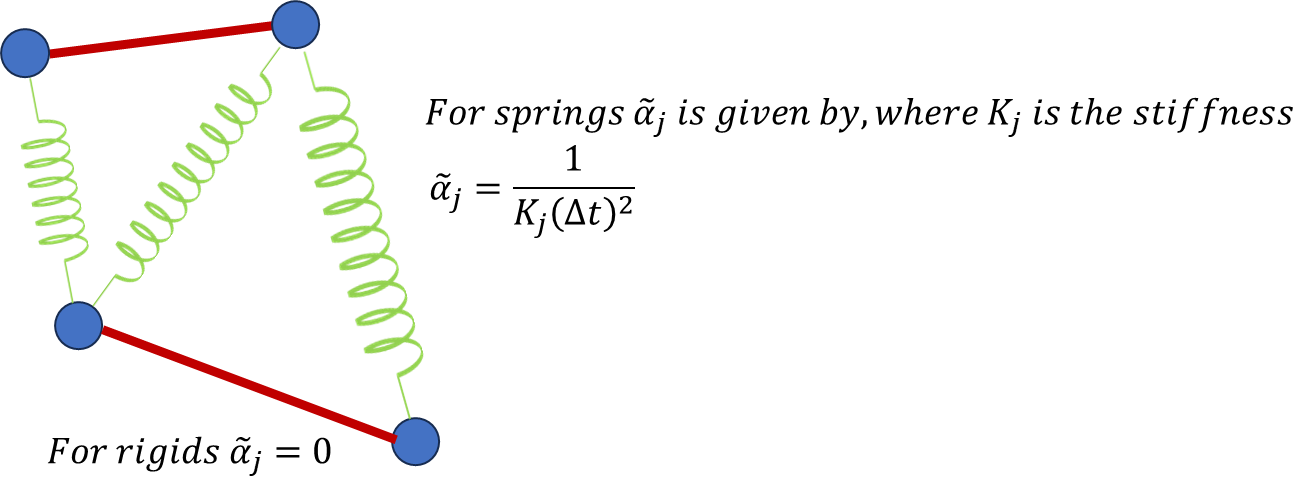
Substituting,

Like the inertial force, the damping and spring restoring force can be derived in the same way,

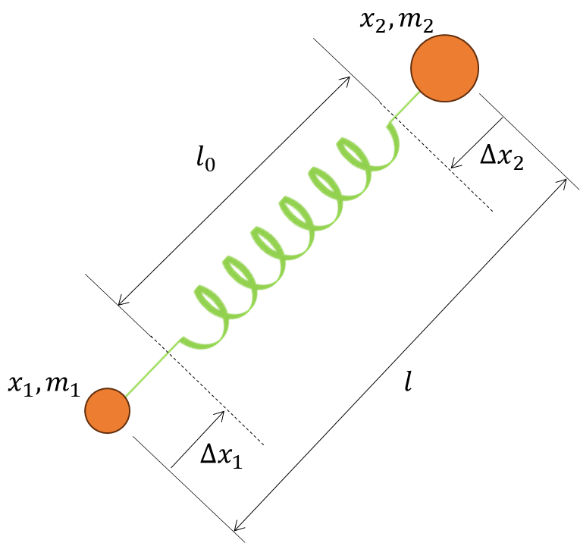
The final constraint equation for spring mass system is given by,

## eXtended Position – Based Dynamics (XPBD)

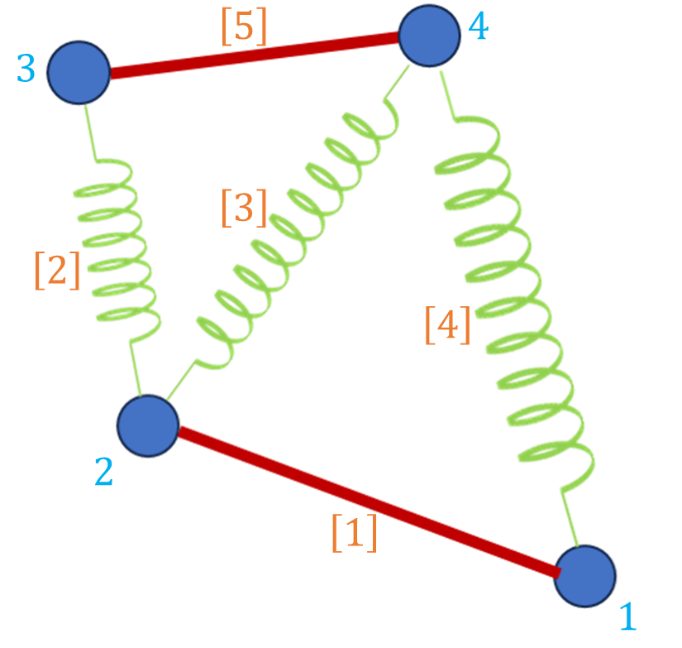
Equations from the paper XPBD: Position – Based Simulation of Compliant Constrained Dynamics, Miles Macklin, Matthias Muller, Nuttapong Chentanez.



## Distance Constraint



## XPBD Algorithm



Set the distance constraints for all the elements.

1. Time integration

In the first step the particle positions are advected using a simple Euler method to obtain predicted positions x∗:

where aext are accelerations due to external forces.

1. Solver loop
   1. Compute Lagrange multipliers

XPBD

When using XPBD, we do not directly compute a Lagrange multiplier in each iteration. Instead, we start with λi=0 and then in each iteration we compute a change of the Lagrange multiplier as

where,

Ki is the spring stiffness.

When the element is rigid.

* 1. Constraint gradients

To compute the Lagrange multipliers, the constraint gradients are required which are computed as:

* 1. Position correction:

The position correction for a particle is determined using the change of the Lagrange multiplier:

* 1. Correction is applied to update x\* as

1. Position and Velocity update:

The final positions and velocities are computed as:

## XPBD Algorithm

|  |
| --- |
| Algorithm: XPBD: Position-Based Simulation of Compliant Constrained Dynamics by Miles Macklin, Matthias M¨uller, Nuttapong Chentanez.  Input: Initial positions (xi0), velocities (vi0), masses (mi), external force function fext, time step Δt, solver iteration count  Output: Updated positions and velocities of particles  1. Initialize each particle i  **For each** particle i **do**  xi = xi0 // Set initial position  vi = vi0 // Set initial velocity  wi = 1 / mi // Compute inverse mass  **End For**  2. Simulation Loop  **Repeat**  // Update velocities with external force  **For each** particle i **do**  vi = vi + Δt \* wi \* fext(t, xi)  **End For**  // Update positions based on new velocities  **For each** particle i **do**  pi = xi + Δt \* vi  **End For**  // Solve constraints specified times  **For** 1 to solveriteration **do**  projectConstraints(C\_1, C\_2, ..., C\_M, p\_1, p\_2, ..., p\_N)  **End For**  // Correct velocities and update positions  **For each** particle i **do**  vi = (pi - xi) / Δt  xi = pi  End For  **Until** simulation ends |