Dynamic Simulation

TODOS

- What is dynamic simulation?
- What is the standard approach?
- What are the challenges?
- Enter constraints!

Simulating effects such as incompressibility, inextensibility and joints between atriculated rigid bodies can be achieved in elasticity-based simulations by using high stiffness values. High stiffness values lead to large forces which in turn cause numerical issues in the solver.

We demonstrate these issues based on the example of maintaining a desired distance between two points using a stiff spring [5]. Let $\boldsymbol{x_1}, \boldsymbol{x_2}$ be the positions, $\boldsymbol{v_1}, \boldsymbol{v_2}$ the velocities and $\boldsymbol{a_1}, \boldsymbol{a_2}$ be the accelerations of the two particles. Let $\bar{l}$ be the rest length and $l = \|\boldsymbol{x_1} - \boldsymbol{x_2}\|$ be the current length of the spring with stiffness k. It can be shown that the force that the spring applies at each particle is equal to $\boldsymbol{f_1} = -\boldsymbol{f_2} = \lambda \boldsymbol{u}$, where $\boldsymbol{u} = (\boldsymbol{x_1} - \boldsymbol{x_2})/l$ and $\lambda = -\frac{\delta V}{\delta l} = k(\bar{l} - l)$.

Once the forces, accelerations, velocities and positions are combined into vectors $\boldsymbol{f}, \boldsymbol{a}, \boldsymbol{v}, \boldsymbol{x}$, respectively, the motions of the system can be modeled via Newton's Ordinary Differential Equation (ODE) $\boldsymbol{f} = \boldsymbol{M}\boldsymbol{a}$, where $\boldsymbol{M}$ is a $n_d \times n_d$ diagonal matrix and n_d is the total number of independent degrees of freedom for the particles.

This system can be integrated via the symplectic Euler method as follows:

$$\boldsymbol{v_{n+1}} = \boldsymbol{v_n} + h\boldsymbol{a_n}$$
$$\boldsymbol{x_{n+1}} = \boldsymbol{x_n} + h\boldsymbol{v_{n+1}}$$

As the stiffness k of the spring increases, so does the magnitude of the acceleration $\boldsymbol{a}$. Consequently, the integration diverges unless the timestep is prohibitively small. The stability issues are often addressed by switching to an implicit integration scheme, such as the backward Euler method [2]. Replacing current accelerations with future accelerations requires the solution of the following linear system of equations (LSE):

$$(\boldsymbol{M} - h^2\boldsymbol{K})\boldsymbol{v_{n+1}} = \boldsymbol{p} + h\boldsymbol{f}$$

where $\boldsymbol{p} = \boldsymbol{M}\boldsymbol{v_n}$ is the momentum, and $\boldsymbol{K} = \frac{\delta \boldsymbol{f}}{\delta \boldsymbol{x}}$ is the stiffness matrix. Note that $\boldsymbol{K}$ is typically non-singular since elastic forces are invariant under rigid body transforms. When using large stiffness k for springs, the entries of $\boldsymbol{K}$ are large (due to large restorative forces for stiff springs) and dominate the entries of the system matrix $\boldsymbol{H} = \boldsymbol{M} - h^2\boldsymbol{K}$. In these cases, $\boldsymbol{H}$ will be almost non-singular as well, leading to numerical issues and poor convergence for many solvers. Additionally, implicit integration introduces noticable numerical damping [4].

This system results from performing the implicit integration and solving the non-linear system via linearization using the Taylor expansion. Positions can be expressed in terms of velocities and eliminated from the system.

Constraint-based Dynamics

Hard Constraints

The problem of maintaining hard distance constraints between particles can be formulated as a Differential Algebraic Equation (DAE) [6, 1]. In this framework, the standard ODE $\boldsymbol{f} = \boldsymbol{M}\boldsymbol{a}$ is handled together with algebraic equations that model the constraints on the system. Distance constraints are typically implemented using holonomic constraints of the form $\boldsymbol{\phi(x)} = 0$. Note that the distance constraint $\phi(x)$ is formulated in terms of the particle positions, whereas the ODE works on accelerations or velocities. Consequently, the constraints need to be differentiated once or twice so that they can be combined with the ODE in terms of velocties or accelerations, respectively. In xPBD, we go the other way! The ODE is tanslated so that it is in terms of positions, so that it can be handled together with the constraints. Is there a reason nobody bothered to do this before? What are the challenges here? Is this exactly what xPBD is? Is there a way to view the simplifications made in terms of the other frameworks?. Using $\boldsymbol{J} = \frac{\delta \boldsymbol{\phi}}{\delta \boldsymbol{x}}$, where $\boldsymbol{J}$ is a $n_c \times n_d$ matrix and n_c is the number of scalar constraints, this leads to the following constraint formulations:

$$\boldsymbol{J}\boldsymbol{v} = 0$$
$$\boldsymbol{J}\boldsymbol{a} = \boldsymbol{c}(\boldsymbol{v})$$

for some $\boldsymbol{c}(\boldsymbol{v})$. If you check [6], see that $c(v)$ also depends on the positions q. That should be indicated! Additionally, constraint forces are required in order to link the algebraic constraint equations with the ODE describing the motion of the system. It can be shown that the constraint forces $\boldsymbol{f_c}$ applied to the particles have to be in the following form in order to avoid adding linear and angular momentum to the system [1]:

$$\boldsymbol{f_c} = \boldsymbol{J^T}\boldsymbol{\lambda}$$

where the λ are the Lagrange multipliers of the constraints. With external forces $\boldsymbol{f_e}$, the DAE can now be expressed as follows [6]:

$$\begin{pmatrix} \boldsymbol{M} & \boldsymbol{J^T} \\ \boldsymbol{J} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{a} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \boldsymbol{f_e} \\ \boldsymbol{c(v)} \end{pmatrix}$$

Note that the lower block-row of the system drives towards accelerations that satisfy the constraints imposed by $\boldsymbol{\phi(x)}$ (or, striclty speaking, the differentiations thereof) exactly. This is indicated by the lower-right zero block in the system matrix in either formulation. Aren't $\dot{q} = v$ and $\dot{v} = a$ also part of the differential equation? Because $c(v)$ and f_e also depend on q!

In [6], the DAE is approached by eliminating the λ from the system entirely and constructing an ODE in terms of positions and velocities. In [5], the authors suggest applying implicit integration schemes to the system directly by constructing the following Karush-Kuhn-Tucker (KKT) equation system:

$$\begin{pmatrix} \boldsymbol{M} & \boldsymbol{-J^T} \\ \boldsymbol{J} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{v} \\ \boldsymbol{\mu} \end{pmatrix} = \begin{pmatrix} \boldsymbol{p} + h\boldsymbol{f_e} \\ 0 \end{pmatrix}$$

If internal forces are taken into account, the upper-left matrix $\boldsymbol{M}$ is replaced by the matrix $\boldsymbol{H}$ from above.

Reverse-engineering how the authors arrived at this system is quite enlightening. Start out from the equations of motion [6]

$$\dot{\boldsymbol{v}} = \boldsymbol{M^{-1}}(\boldsymbol{f} - \boldsymbol{J^T})\boldsymbol{\lambda}$$

and perform implicit integration:

$$\begin{aligned} \boldsymbol{v_{n+1}} &= \boldsymbol{v_n} + h\boldsymbol{M^{-1}}(\boldsymbol{f_e}(\boldsymbol{x_{n+1}}) - \boldsymbol{J^T}(\boldsymbol{x_{n+1}})\boldsymbol{\lambda}) \\ \boldsymbol{M}\boldsymbol{v_{n+1}} &= \boldsymbol{p} + h\boldsymbol{f_e}(\boldsymbol{x_{n+1}}) - h\boldsymbol{J^T}(\boldsymbol{x_{n+1}})\boldsymbol{\lambda} \\ \boldsymbol{M}\boldsymbol{v_{n+1}} + h\boldsymbol{J^T}(\boldsymbol{x_{n+1}})\boldsymbol{\lambda} &= \boldsymbol{p} + h\boldsymbol{f_e}(\boldsymbol{x_{n+1}}) \\ \boldsymbol{M}\boldsymbol{v_{n+1}} + \boldsymbol{J^T}(\boldsymbol{x_{n+1}})\boldsymbol{\mu} &= \boldsymbol{p} + h\boldsymbol{f_e}(\boldsymbol{x_{n+1}}) \end{aligned}$$

If we assume that $\boldsymbol{f_e}$ and the constraint gradients $\boldsymbol{J}$ are constant across the time step, we arrive at the formulation from the paper. For the external forces, which are usually only comprised of gravitational forces, this is not a big deal. Is this what authors mean when they say that the constraints are effectively linearized during one solve, e.g. second page of [3]?

Note that the system matrix is sparse, which can be exploited by sparse-matrix solvers in order to solve the system efficiently [1]. Alternatively, the Schur complement can be constructed since the mass matrix in the upper left block is invertible. This leads to a smaller, albeit less sparse system [5]:

$$\boldsymbol{J}\boldsymbol{M^{-1}}\boldsymbol{J^T}\boldsymbol{\mu} = -\boldsymbol{J}\boldsymbol{M^{-1}}(\boldsymbol{p} + h\boldsymbol{f_e})$$

If the constraints are not redundant, $\boldsymbol{J}\boldsymbol{M^{-1}}\boldsymbol{J^T}$ is non-singular and symmetric positive definite [1], which are desirable properties for many solvers. According to [4], the common approaches for linearizing the constraint forces and stabilizing the constraints $\boldsymbol{\phi(x)} = 0$ are notoriously unstable. Additionally, instabilities in the traverse direction of the constraints occur when the tensile force with respect to particle masses is large when using hard constraints [5].

References

[1] David Baraff. Linear-time dynamics using lagrange multipliers. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, page 137–146, New York, NY, USA, 1996. Association for Computing Machinery.

[2] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, page 43–54, New York, NY, USA, 1998. Association for Computing Machinery.

[3] Matthias Müller, Miles Macklin, Nuttapong Chentanez, Stefan Jeschke, and Tae-Yong Kim. Detailed rigid body simulation with extended position based dynamics. *Computer Graphics Forum*, 39(8):101–112, 2020.

[4] Martin Servin, Claude Lacoursière, and Niklas Melin. Interactive simulation of elastic deformable materials. *Proc. SIGRAD*, 01 2006.

[5] Maxime Tournier, Matthieu Nesme, Benjamin Gilles, and François Faure. Stable constrained dynamics. *ACM Trans. Graph.*, 34(4), jul 2015.

[6] Linda R. Petzold Uri M. Ascher, Hongsheng Chin and Sebastian Reich. Stabilization of constrained mechanical systems with daes and invariant manifolds. *Mechanics of Structures and Machines*, 23(2):135–157, 1995.