Simulation in Computer Graphics
Particle Motion 1

Matthias Teschner

UNI FREIBURG

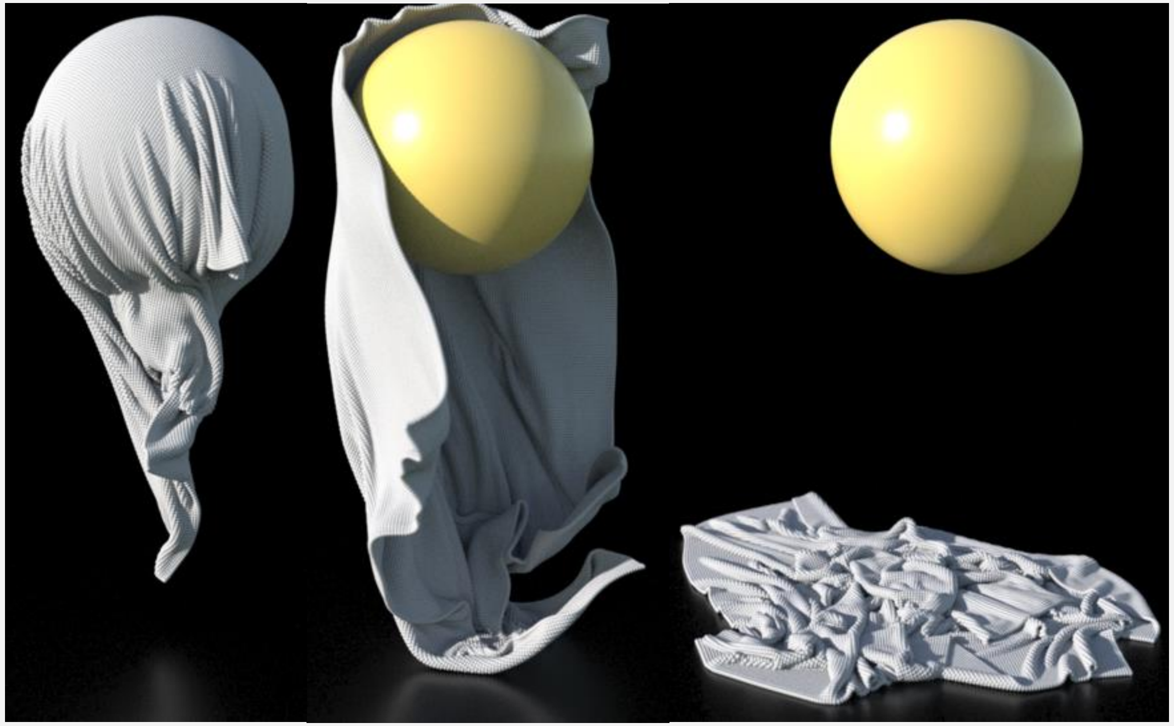
Outline

- Introduction
- Particle motion
- Finite differences
- System of first-order ODEs
- Second-order ODE
- Performance
- Discussion

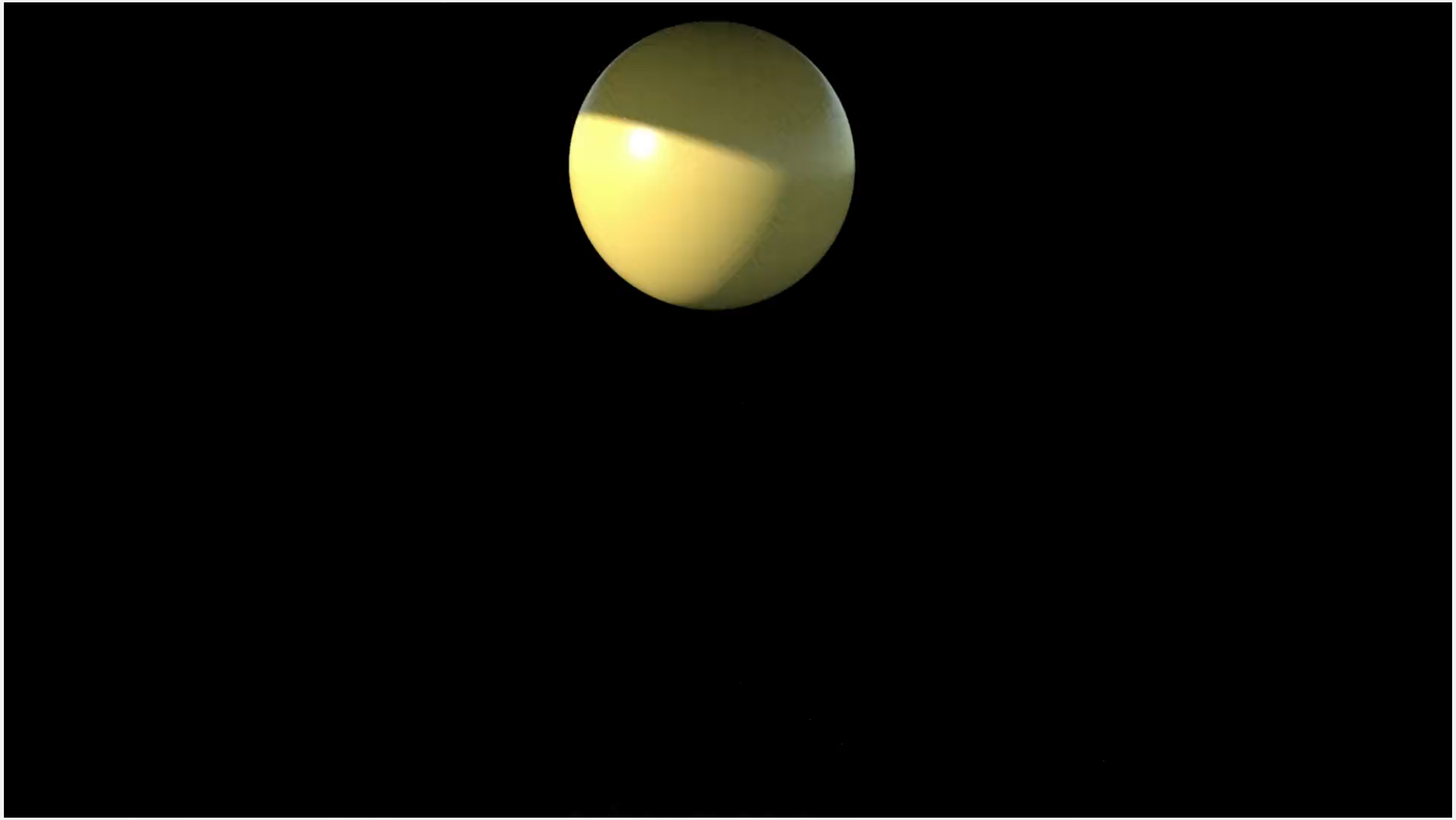
UNI FREIBURG

Goal

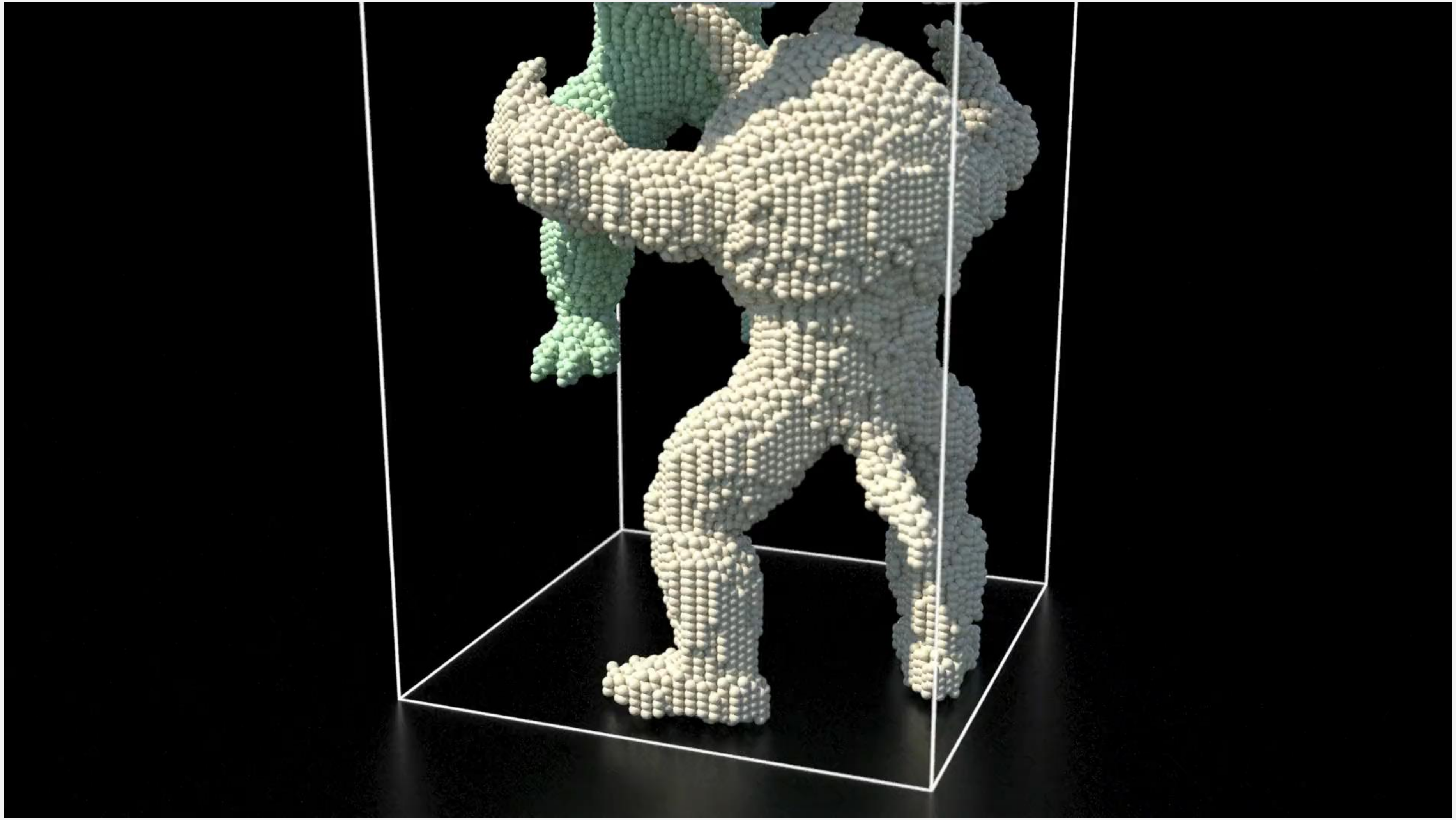
- Dynamic simulation of
  - Rigid bodies
  - Deformable objects
  - Fluids



Goal



Goal

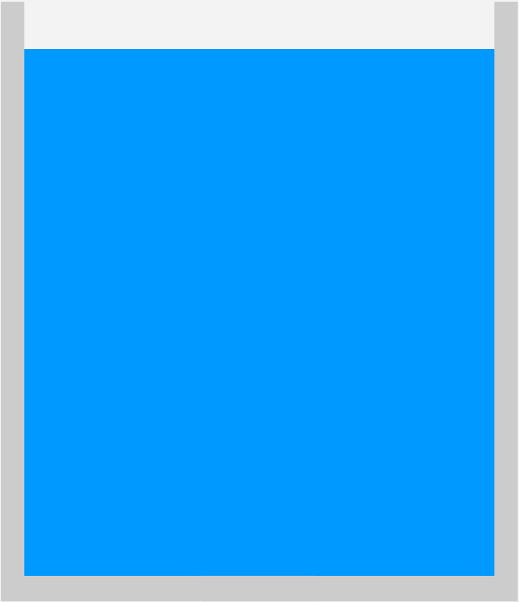


Representation

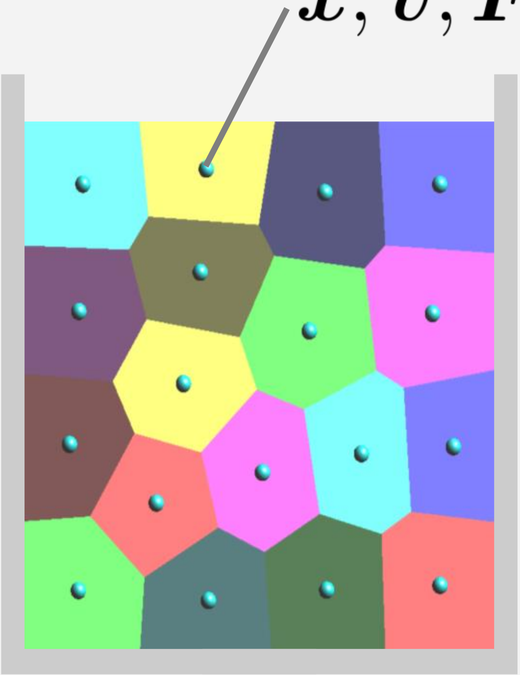
- Subdivision of objects into small parts, i.e. particles
- Particles have properties
  - Mass m, volume V, density ρ
  - Position $\boldsymbol{x}$, velocity $\boldsymbol{v}$, force $\boldsymbol{F}$
- Particles are of arbitrary shape

Fluid Particles

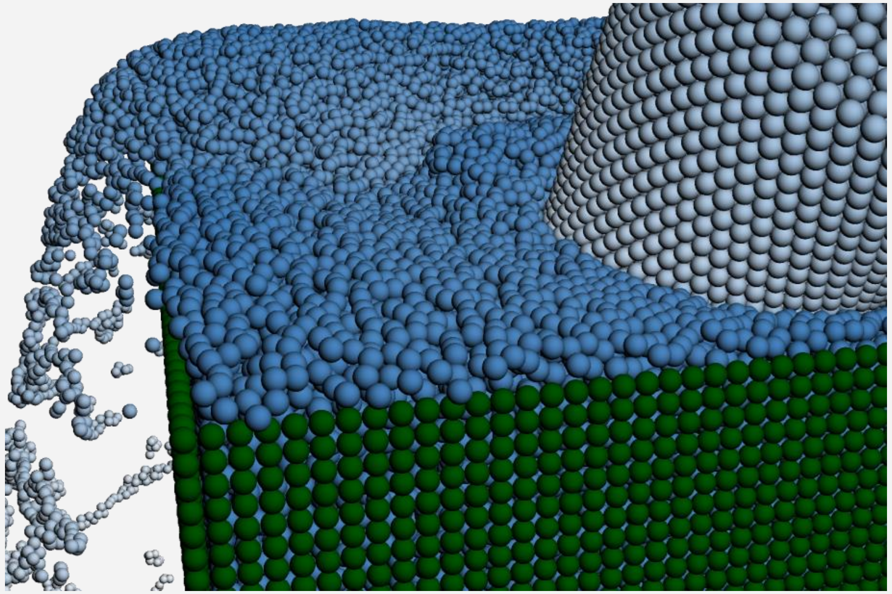
$$\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{F}, m, V, \rho$$



Fluid body



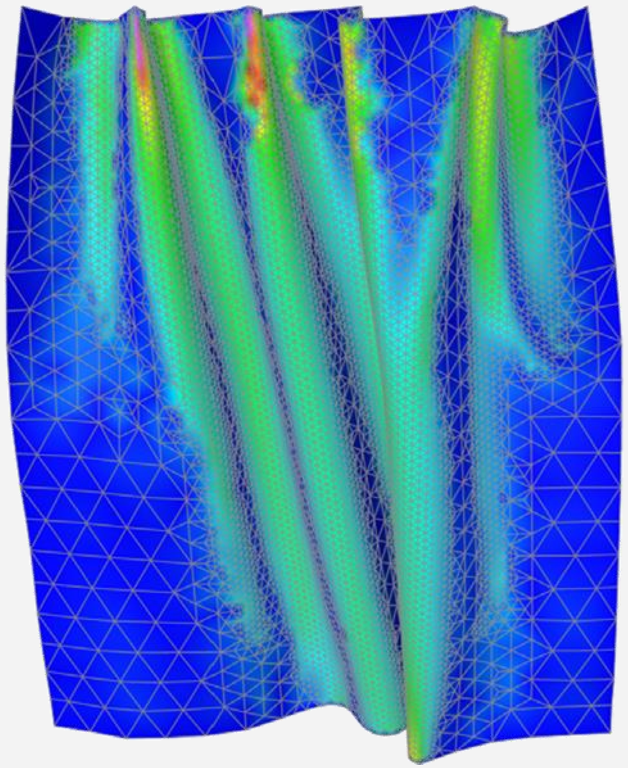
Fluid particles



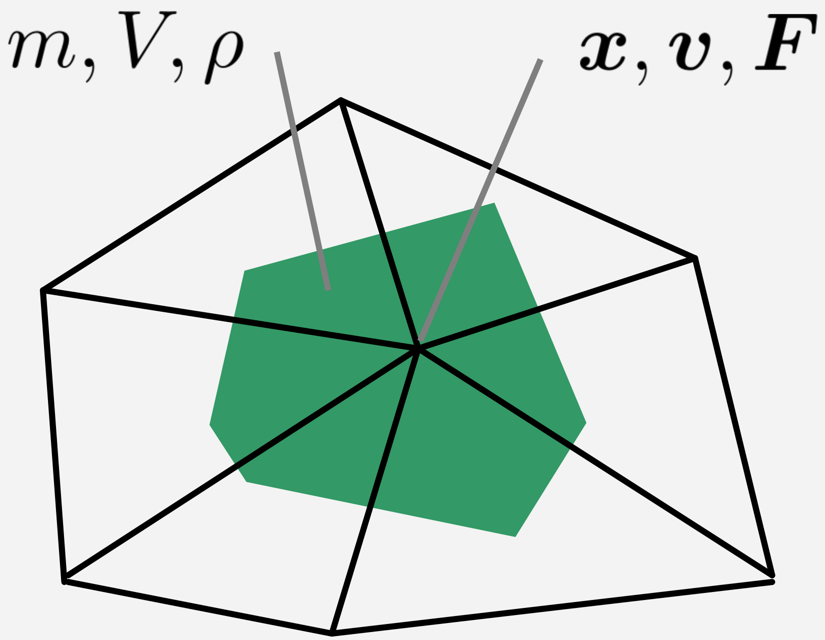
Typical visualization

Cloth Particles

Cloth



Cloth particles

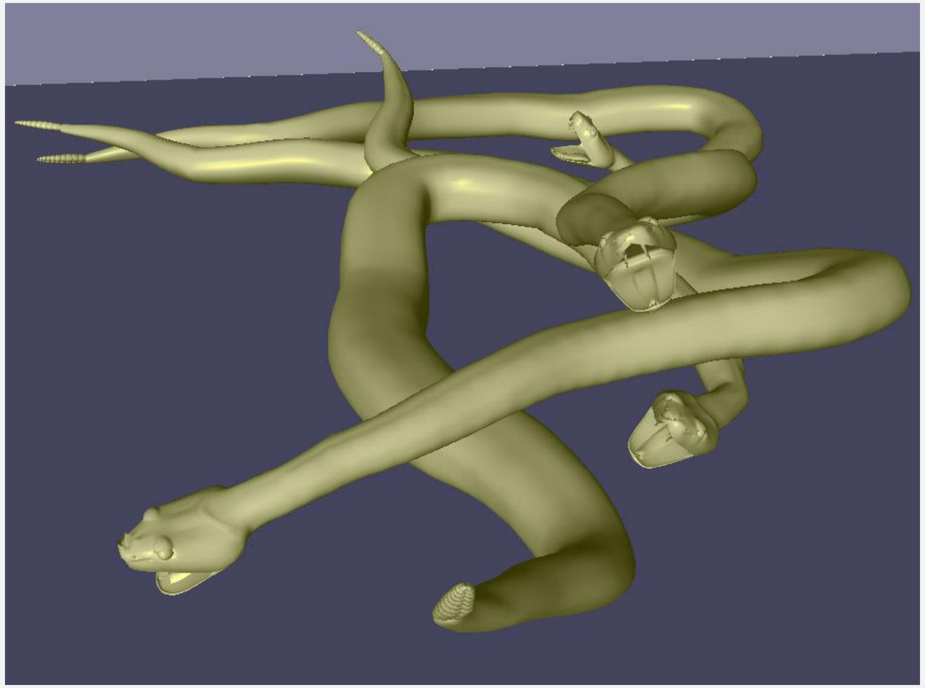


Illustration

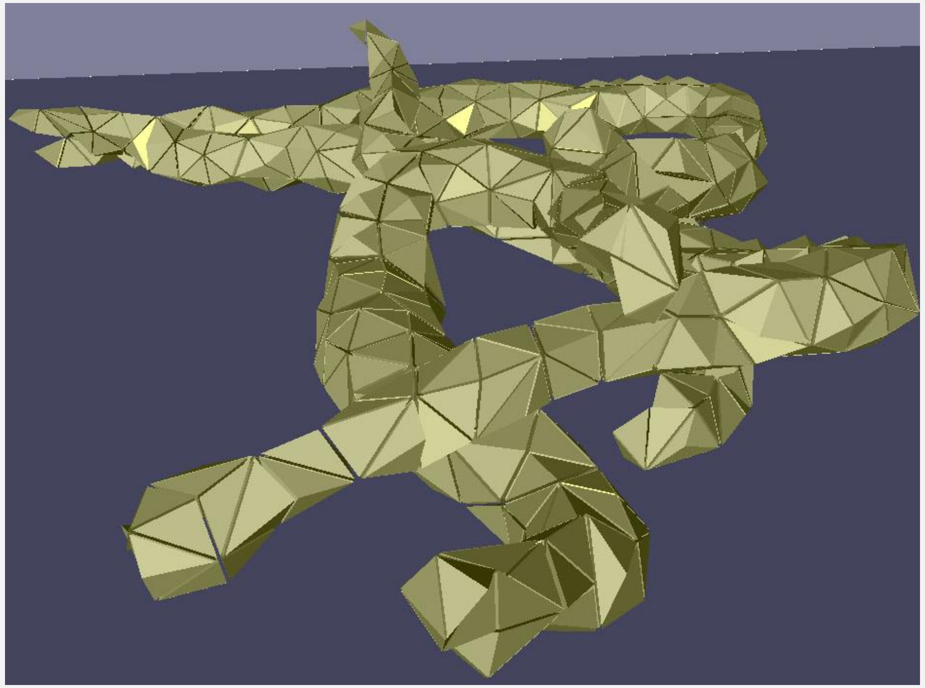
[Bender, Deul, 2013]

UNI FREIBURG

Deformable 3D Particles



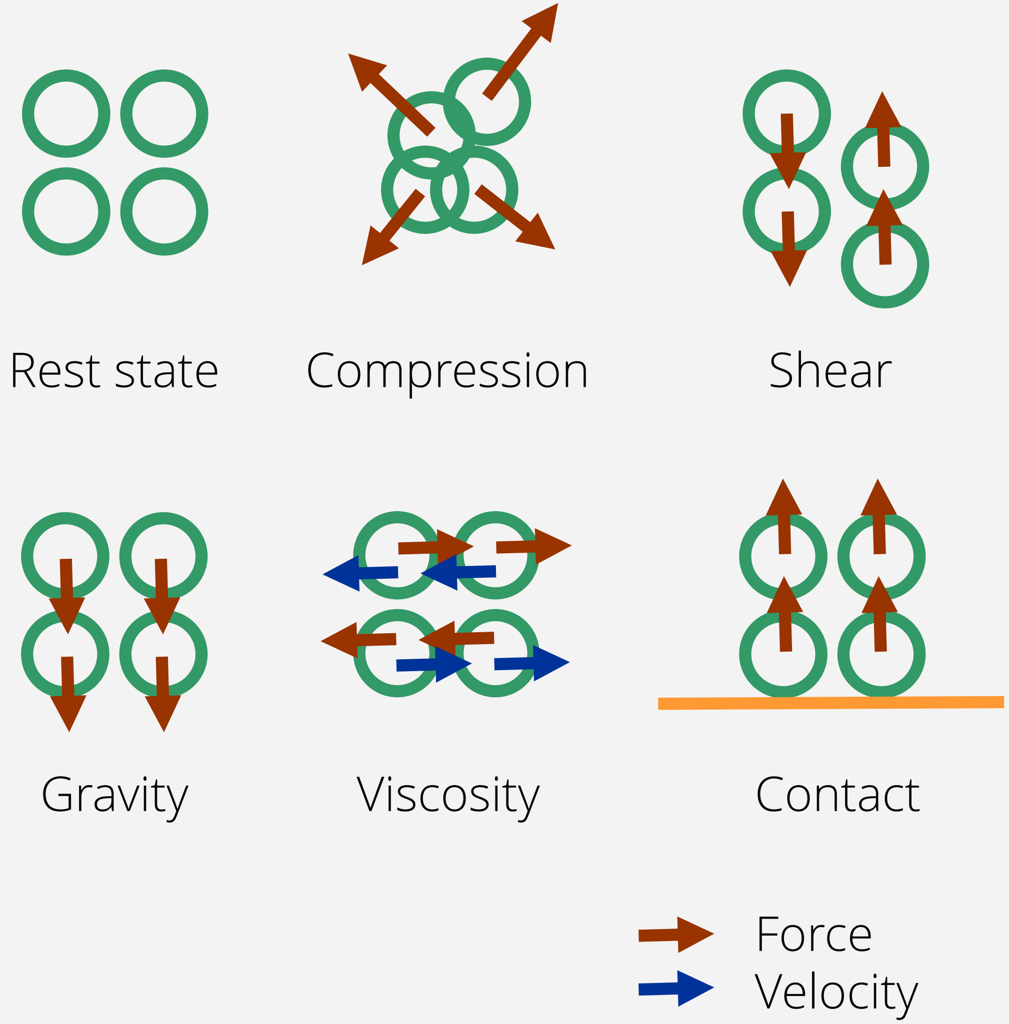
Deformable 3D object



Approximate tetrahedral mesh

Particle Forces

- Result from
  - Distortions, e.g. volume or shape change
  - Gravity
  - Friction, viscosity
  - Contact
  - …



Particle Motion

- Particles change position $\boldsymbol{x}$ with velocity $\boldsymbol{v}$ $\quad \boldsymbol{v} = \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t}$
- Velocity governed by Newton's Second Law
  - Force at a particle equals the time rate of change of its momentum

$$\boldsymbol{F} = \frac{\mathrm{d}}{\mathrm{d}t}(m\boldsymbol{v}) = \frac{\mathrm{d}m}{\mathrm{d}t}\boldsymbol{v} + \frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t}m$$

  - Two governing equations for two unknown functions $\boldsymbol{x}, \boldsymbol{v}$

$$\boldsymbol{F} = m\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t} \qquad \boldsymbol{v} = \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t}$$

Coupled system of first order ODEs

  - Can also be written as

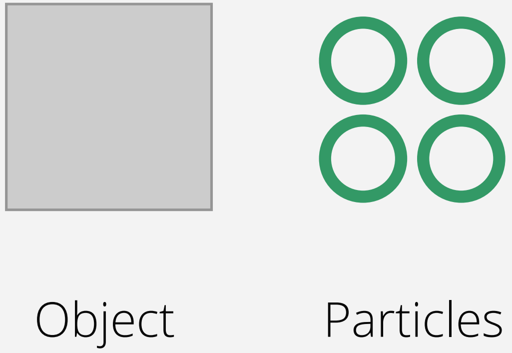
$$\boldsymbol{F} = m\frac{\mathrm{d}^2\boldsymbol{x}}{\mathrm{d}t^2}$$

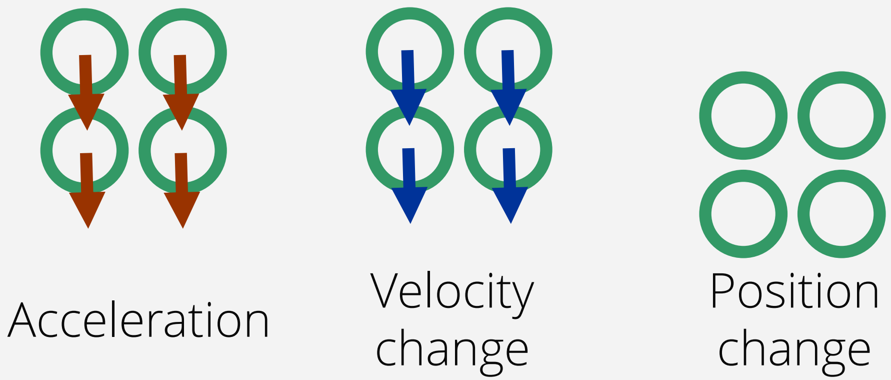
Second order ODE

UNI FREIBURG

Particle-based Simulation

- Object subdivision into particles (spatial discretization)
- Force modeling
- Particle motion
  - Transport / advection





$$\frac{\boldsymbol{F}}{m} = \frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t} = \frac{\mathrm{d}^2\boldsymbol{x}}{\mathrm{d}t^2}$$

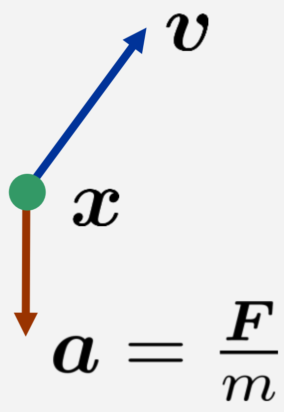
Outline

– Introduction
– Particle motion
– Finite differences
– System of first-order ODEs
– Second-order ODE
– Performance
– Discussion

UNI FREIBURG

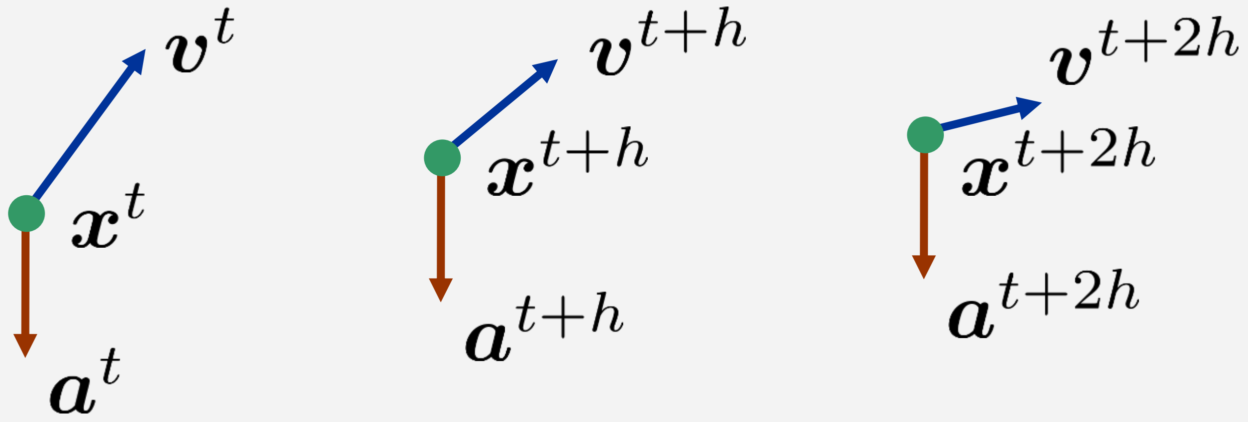
Particle Quantities

- Mass $m \in \mathbb{R}$
- Position $\boldsymbol{x} \in \mathbb{R}^3$
- Velocity $\boldsymbol{v} \in \mathbb{R}^3$
- Force $\boldsymbol{F} \in \mathbb{R}^3$
- Acceleration $\boldsymbol{a} = \frac{\boldsymbol{F}}{m} \in \mathbb{R}^3$



Time Discretization

– Quantities are considered at discrete time points



h is the so-called time step.

– Particle simulations are concerned with the computation of unknown future particle quantities $\boldsymbol{x}^{t+h}$, $\boldsymbol{v}^{t+h}$ from known current information $\boldsymbol{x}^t$, $\boldsymbol{v}^t$, $\boldsymbol{a}^t$

Governing Equations

- Newton's Second Law, motion equation
  $$\boldsymbol{a}^t = \frac{\mathrm{d}\boldsymbol{v}^t}{\mathrm{d}t} = \frac{\mathrm{d}^2\boldsymbol{x}^t}{\mathrm{d}t^2}$$
- Ordinary differential equations ODEs
- Describe the behavior of $\boldsymbol{x}^t$ and $\boldsymbol{v}^t$ in terms of their derivatives with respect to time
- Numerical integration is employed to approximatively solve the ODE , i.e. to approximate the unknown functions $\boldsymbol{x}^t$ and $\boldsymbol{v}^t$

Governing Equations

- Initial value problem of second order

$$\frac{\mathrm{d}^2 \boldsymbol{x}^t}{\mathrm{d}t^2} = \boldsymbol{a}^t \quad \boldsymbol{x}^{t_0} = \boldsymbol{x}^{\text{init}} \quad \frac{\mathrm{d}\boldsymbol{x}^{t_0}}{\mathrm{d}t} = \boldsymbol{v}^{\text{init}}$$

- Second-order ODEs can be rewritten as a system of two coupled equations of first order
- Initial value problems of first order

$$\frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t} = \boldsymbol{v}^t \qquad \boldsymbol{x}^{t_0} = \boldsymbol{x}^{\text{init}}$$

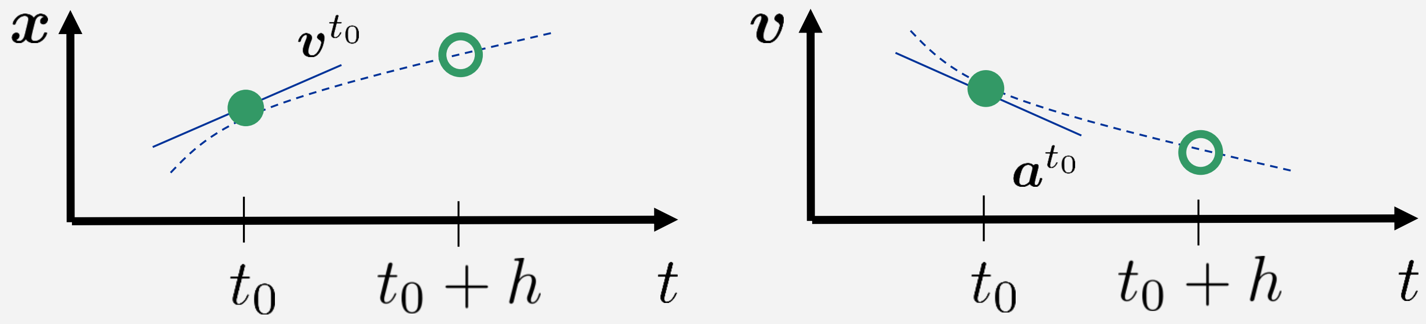
$$\frac{\mathrm{d}\boldsymbol{v}^t}{\mathrm{d}t} = \boldsymbol{a}^t \qquad \boldsymbol{v}^{t_0} = \boldsymbol{v}^{\text{init}}$$

Initial Value Problem of First Order

- Functions $\boldsymbol{x}^t$ and $\boldsymbol{v}^t$ represent the particle motion
- Initial values $\boldsymbol{x}^{t_0}$ and $\boldsymbol{v}^{t_0}$ are given
- First-order differential equations are given

$$\frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t} = \boldsymbol{v}^t \quad \frac{\mathrm{d}\boldsymbol{v}^t}{\mathrm{d}t} = \boldsymbol{a}^t$$

- How to estimate $\boldsymbol{x}^{t_0+h}$ and $\boldsymbol{v}^{t_0+h}$?



Particle Accelerations

- Depend on sets of positions and velocities
- E.g., damped spring $\boldsymbol{a}_1(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{v}_1, \boldsymbol{v}_2)$

$$= \frac{1}{m_1} \frac{k_{12}}{L_{12}} (\|\boldsymbol{x}_2 - \boldsymbol{x}_1\| - L_{12}) \frac{\boldsymbol{x}_2 - \boldsymbol{x}_1}{\|\boldsymbol{x}_2 - \boldsymbol{x}_1\|}$$

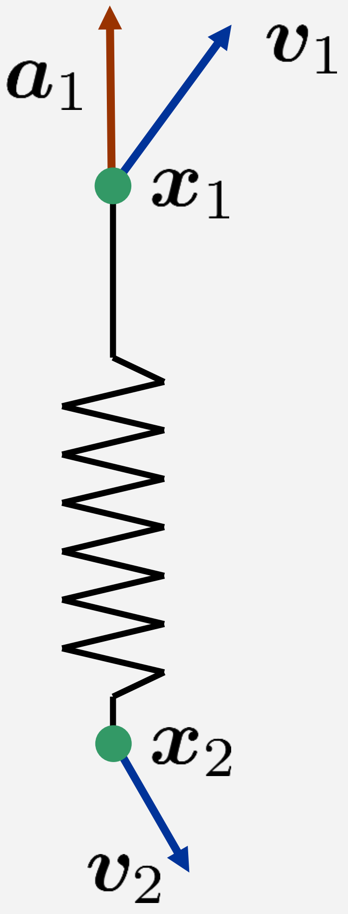
Elastic acceleration

Particle mass | Spring stiffness | Actual spring length | Rest spring length | Normalized direction

$$+ \frac{1}{m_1} \gamma \left((\boldsymbol{v}_2 - \boldsymbol{v}_1) \frac{\boldsymbol{x}_2 - \boldsymbol{x}_1}{\|\boldsymbol{x}_2 - \boldsymbol{x}_1\|} \right) \frac{\boldsymbol{x}_2 - \boldsymbol{x}_1}{\|\boldsymbol{x}_2 - \boldsymbol{x}_1\|}$$

Damping acceleration

Damping parameter | Relative velocity projected onto spring | Normalized direction



Particle Accelerations

- Are typically expensive to compute
  - E.g., sums over adjacent particles
- Might need additional effort
  - E.g., contact handling forces require collision detection

Outline

- Introduction
- Particle motion
- Finite differences
- System of first-order ODEs
- Second-order ODE
- Performance
- Discussion

Finite Differences

– Taylor-series approximation

$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + \frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t} h + O(h^2)$$

O(h^2) – order of the truncation / discretization error

$$\frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t} = \frac{\boldsymbol{x}^{t+h} - \boldsymbol{x}^t}{h} + O(h)$$

O(h) – error order of, e.g., a scheme that employs such approximation

– Continuous ODEs are replaced with discrete finite-difference equations FDEs

$$\frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t} = \boldsymbol{v}^t \quad \Rightarrow \quad \frac{\boldsymbol{x}^{t+h} - \boldsymbol{x}^t}{h} = \boldsymbol{v}^t \quad \Rightarrow \quad \boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\boldsymbol{v}^t$$

$$\frac{\mathrm{d}\boldsymbol{v}^t}{\mathrm{d}t} = \boldsymbol{a}^t \quad \Rightarrow \quad \frac{\boldsymbol{v}^{t+h} - \boldsymbol{v}^t}{h} = \boldsymbol{a}^t \quad \Rightarrow \quad \boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^t$$

ODE | FDE | The first approximate solution of our problem

UNI FREIBURG

Finite Differences

- Line fitting (assuming $\frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t} = \text{const}$ near $\boldsymbol{x}^t$)

$$\boldsymbol{x}^t = \boldsymbol{b}t + \boldsymbol{c}$$

$$\Rightarrow \frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t} = \boldsymbol{b} \Rightarrow \boldsymbol{c} = \boldsymbol{x}^t - \frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t}t$$

$$\boldsymbol{x}^{t+h} = \frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t}(t+h) + \boldsymbol{x}^t - \frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t}t$$

- Resulting in

$$\frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t} = \frac{\boldsymbol{x}^{t+h} - \boldsymbol{x}^t}{h} + O(h)$$

UNI FREIBURG

Outline

- Introduction
- Particle motion
- Finite differences
- System of first-order ODEs
  - Explicit schemes
  - Predictor-corrector schemes
  - Implicit schemes
- ...

UNI FREIBURG

Explicit Euler

- Governing equations

$$\frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t} = \boldsymbol{v}^t \quad \frac{\mathrm{d}\boldsymbol{v}^t}{\mathrm{d}t} = \boldsymbol{a}^t$$

- Initialization $\boldsymbol{x}^{t_0} = \boldsymbol{x}^{\mathrm{init}}$, $\boldsymbol{v}^{t_0} = \boldsymbol{v}^{\mathrm{init}}$, $\boldsymbol{a}^{t_0}$, h
- Explicit Euler update

$$\boldsymbol{x}^{t_0+h} = \boldsymbol{x}^{t_0} + h\frac{\mathrm{d}\boldsymbol{x}^{t_0}}{\mathrm{d}t} + O(h^2) = \boldsymbol{x}^{t_0} + h\boldsymbol{v}^{t_0} + O(h^2)$$

$$\boldsymbol{v}^{t_0+h} = \boldsymbol{v}^{t_0} + h\frac{\mathrm{d}\boldsymbol{v}^{t_0}}{\mathrm{d}t} + O(h^2) = \boldsymbol{v}^{t_0} + h\boldsymbol{a}^{t_0} + O(h^2)$$

Coupled Equations

- Position update depends on velocity
- Velocity update depends on position

$$\boldsymbol{x}^{t_0+h} = \boldsymbol{x}^{t_0} + h\boldsymbol{v}^{t_0}$$

$$\boldsymbol{v}^{t_0+h} = \boldsymbol{v}^{t_0} + h\boldsymbol{a}^{t_0}(\boldsymbol{x}^{t_0}, \boldsymbol{v}^{t_0})$$

$$\boldsymbol{x}^{t_0+2h} = \boldsymbol{x}^{t_0+h} + h\boldsymbol{v}^{t_0+h}$$

$$\boldsymbol{v}^{t_0+2h} = \boldsymbol{v}^{t_0+h} + h\boldsymbol{a}^{t_0+h}(\boldsymbol{x}^{t_0+h}, \boldsymbol{v}^{t_0+h})$$

Accuracy and Stability

- Discretization error is the difference between the solution of the ODE and the solution of the FDE
- The FDE is consistent, if the discretization error vanishes if the time step h approaches zero
- The FDE is stable, if previously introduced errors do not grow within a simulation step
- The FDE is convergent, if the solution of the FDE approaches the solution of the ODE

UNI FREIBURG

Accuracy and Stability

- Although the discretization error is diminished by smaller time steps in consistent schemes, the discretization error is introduced in each step of the FD scheme
- If previously introduced discretization errors are not amplified by the FD scheme, then it is stable
- Consistent and stable schemes are convergent

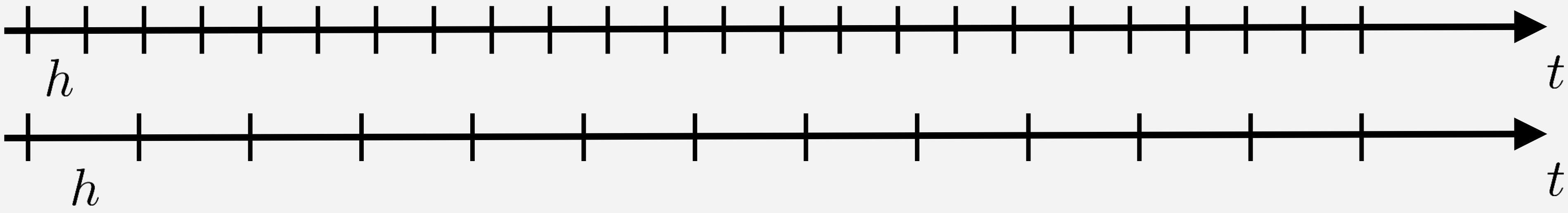
Stability

- If stability is influenced by the time step, the FD scheme is conditionally stable
- If the FD scheme is stable or unstable for arbitrary time steps, it is unconditionally stable or unstable
- ODE, FDE and the parameters influence the stability of a system
- Schemes with improved stability work with larger time steps ⇨ reduced overall computation time

UNI FREIBURG

Time Step

- Larger time steps result in less simulation steps and speed-up the overall computation time of a simulation



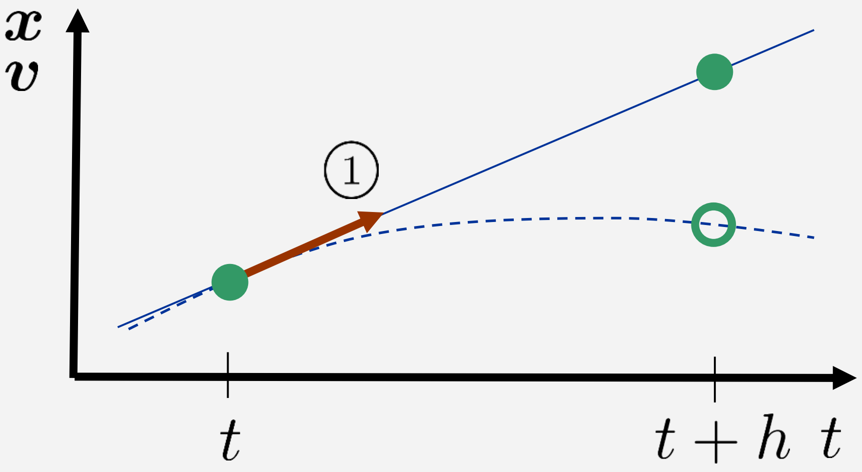
- Different FD schemes allow for different time steps
  - E.g. due to different error orders
  - Computing complexity also differs

Goal

– Stable scheme with maximized ratio between time step and computing complexity per simulation step

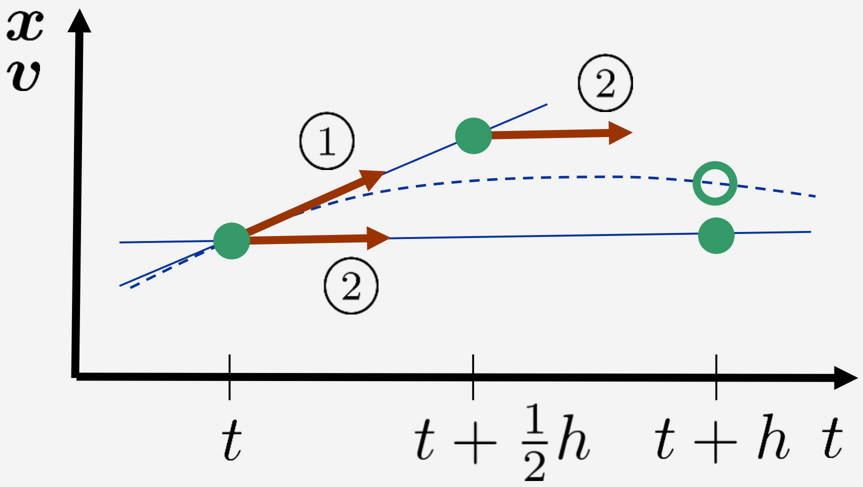
Second-Order Runge Kutta - Midpoint Method

Euler



- One derivative computation ①
- Discretization error $O(h^2)$

Midpoint



- Two derivative computations ① ②
- Requires intermediate positions and velocities
- Discretization error $O(h^3)$

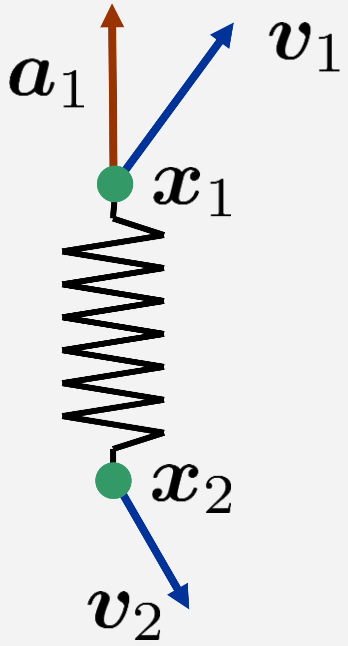
Midpoint Implementation - Spring

- Acceleration at time t: $\boldsymbol{a}_1^t(\boldsymbol{x}_1^t, \boldsymbol{x}_2^t, \boldsymbol{v}_1^t, \boldsymbol{v}_2^t)$
- Intermediate position and velocity at time $t + \frac{h}{2}$:

$$\boldsymbol{x}_1^* = \boldsymbol{x}_1^t + \tfrac{h}{2}\boldsymbol{v}_1^t \quad \boldsymbol{v}_1^* = \boldsymbol{v}_1^t + \tfrac{h}{2}\boldsymbol{a}_1^t \quad \boldsymbol{x}_2^* = \dots \quad \boldsymbol{v}_2^* = \dots$$

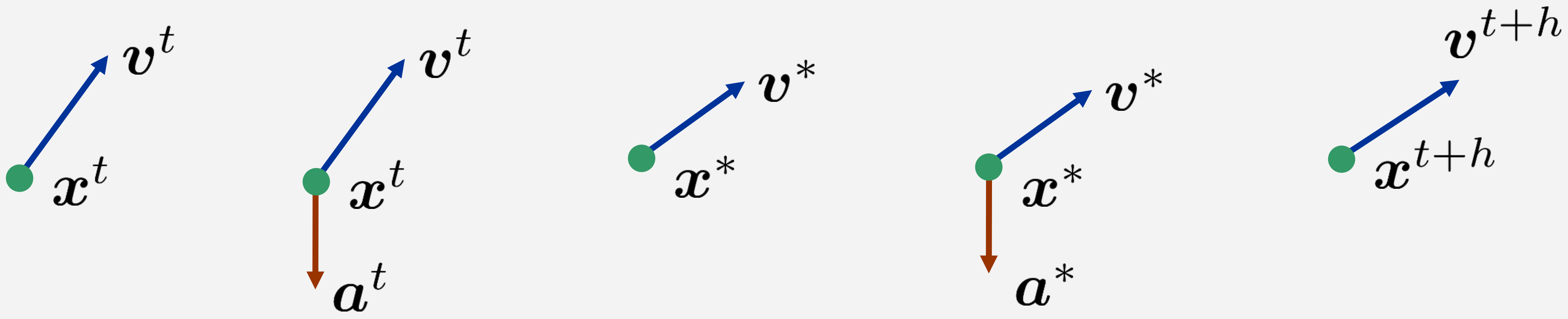
- Intermediate acceleration at time $t + \frac{h}{2}$ using intermediate positions and velocities: $\boldsymbol{a}_1^*(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*, \boldsymbol{v}_1^*, \boldsymbol{v}_2^*)$
- Final position and velocity at time $t + h$

$$\boldsymbol{x}_1^{t+h} = \boldsymbol{x}_1^t + h\boldsymbol{v}_1^* \quad \boldsymbol{v}_1^{t+h} = \boldsymbol{v}_1^t + h\boldsymbol{a}_1^*$$



UNI FREIBURG

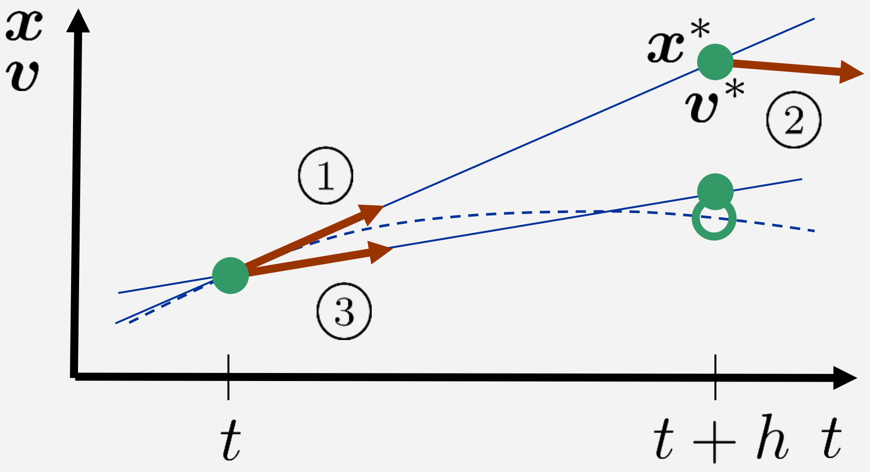
Midpoint Implementation



Current state	Compute all accelerations	Compute all predicted pos. and vel.	Compute all predicted accelerations	Compute all final pos. and vel.

UNI FREIBURG

Second-Order Runge Kutta - Heun



① $\boldsymbol{v}^t, \boldsymbol{a}^t$

② $\boldsymbol{v}^*, \boldsymbol{a}^*$

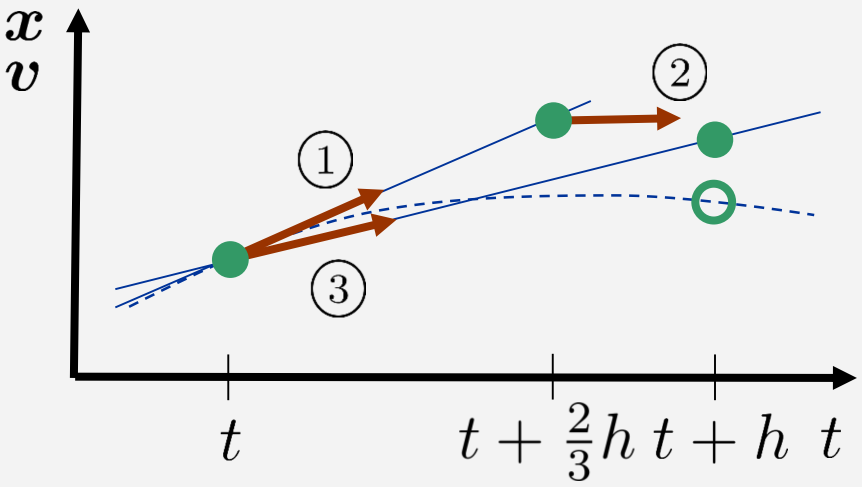
③ $\frac{\boldsymbol{v}^t + \boldsymbol{v}^*}{2}, \frac{\boldsymbol{a}^t + \boldsymbol{a}^*}{2}$

- $\boldsymbol{x}^t, \boldsymbol{v}^t$
- $\boldsymbol{a}^t(\boldsymbol{x}^t, \boldsymbol{v}^t)$ ①
- $\boldsymbol{x}^* = \boldsymbol{x}^t + h\boldsymbol{v}^t \quad \boldsymbol{v}^* = \boldsymbol{v}^t + h\boldsymbol{a}^t$
- $\boldsymbol{a}^*(\boldsymbol{x}^*, \boldsymbol{v}^*)$ ②
- $\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\frac{\boldsymbol{v}^t + \boldsymbol{v}^*}{2}$

  $\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\frac{\boldsymbol{a}^t + \boldsymbol{a}^*}{2}$ ③

UNI FREIBURG

Second-Order Runge Kutta - Ralston

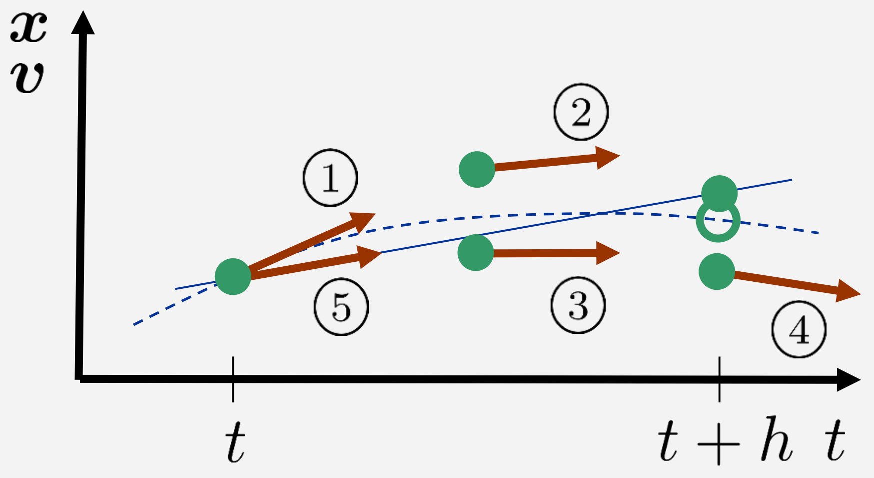


- $\boldsymbol{x}^t, \boldsymbol{v}^t$
- $\boldsymbol{a}^t(\boldsymbol{x}^t, \boldsymbol{v}^t)$ ①
- $\boldsymbol{x}^* = \boldsymbol{x}^t + \frac{2}{3}h\boldsymbol{v}^t \;\; \boldsymbol{v}^* = \boldsymbol{v}^t + \frac{2}{3}h\boldsymbol{a}^t$
- $\boldsymbol{a}^*(\boldsymbol{x}^*, \boldsymbol{v}^*)$ ②
- $\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + \frac{1}{4}h\boldsymbol{v}^t + \frac{3}{4}h\boldsymbol{v}^*$

  $\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + \frac{1}{4}h\boldsymbol{a}^t + \frac{3}{4}h\boldsymbol{a}^*$ ③

UNI FREIBURG

Fourth-Order Runge Kutta - Classic



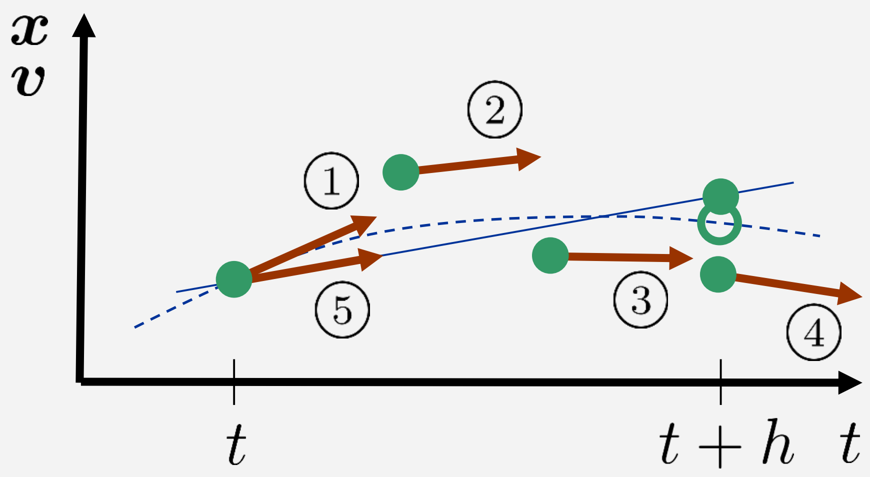
- Four derivative computations
- Discretization error $O(h^5)$

- $\boldsymbol{x}^t, \boldsymbol{v}^t$
- $\boldsymbol{a}^t(\boldsymbol{x}^t, \boldsymbol{v}^t)$ ①
- $\boldsymbol{x}^* = \boldsymbol{x}^t + \frac{h}{2}\boldsymbol{v}^t \qquad \boldsymbol{v}^* = \boldsymbol{v}^t + \frac{h}{2}\boldsymbol{a}^t$
- $\boldsymbol{a}^*(\boldsymbol{x}^*, \boldsymbol{v}^*)$ ②
- $\boldsymbol{x}^{**} = \boldsymbol{x}^t + \frac{h}{2}\boldsymbol{v}^* \qquad \boldsymbol{v}^{**} = \boldsymbol{v}^t + \frac{h}{2}\boldsymbol{a}^*$
- $\boldsymbol{a}^{**}(\boldsymbol{x}^{**}, \boldsymbol{v}^{**})$ ③
- $\boldsymbol{x}^{***} = \boldsymbol{x}^t + h\boldsymbol{v}^{**} \qquad \boldsymbol{v}^{***} = \boldsymbol{v}^t + h\boldsymbol{a}^{**}$
- $\boldsymbol{a}^{***}(\boldsymbol{x}^{***}, \boldsymbol{v}^{***})$ ④
- $\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\frac{\boldsymbol{v}^t + 2\boldsymbol{v}^* + 2\boldsymbol{v}^{**} + \boldsymbol{v}^{***}}{6}$
- $\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\frac{\boldsymbol{a}^t + 2\boldsymbol{a}^* + 2\boldsymbol{a}^{**} + \boldsymbol{a}^{***}}{6}$ ⑤

UNI FREIBURG

Fourth-Order Runge Kutta – 3/8 Rule

- $\boldsymbol{x}^t, \boldsymbol{v}^t$
- $\boldsymbol{a}^t(\boldsymbol{x}^t, \boldsymbol{v}^t)$ ①
- $\boldsymbol{x}^* = \boldsymbol{x}^t + \frac{1}{3}h\boldsymbol{v}^t \quad \boldsymbol{v}^* = \boldsymbol{v}^t + \frac{1}{3}h\boldsymbol{a}^t$
- $\boldsymbol{a}^*(\boldsymbol{x}^*, \boldsymbol{v}^*)$ ②
- $\boldsymbol{x}^{**} = \boldsymbol{x}^t + \frac{2}{3}h(-\frac{1}{2}\boldsymbol{v}^t + \frac{3}{2}\boldsymbol{v}^*)$
- $\boldsymbol{v}^{**} = \boldsymbol{v}^t + \frac{2}{3}h(-\frac{1}{2}\boldsymbol{a}^t + \frac{3}{2}\boldsymbol{a}^*)$
- $\boldsymbol{a}^{**}(\boldsymbol{x}^{**}, \boldsymbol{v}^{**})$ ③
- $\boldsymbol{x}^{***} = \boldsymbol{x}^t + h(\boldsymbol{v}^t - \boldsymbol{v}^* + \boldsymbol{v}^{**})$
- $\boldsymbol{v}^{***} = \boldsymbol{v}^t + h(\boldsymbol{a}^t - \boldsymbol{a}^* + \boldsymbol{a}^{**})$
- $\boldsymbol{a}^{***}(\boldsymbol{x}^{***}, \boldsymbol{v}^{***})$ ④

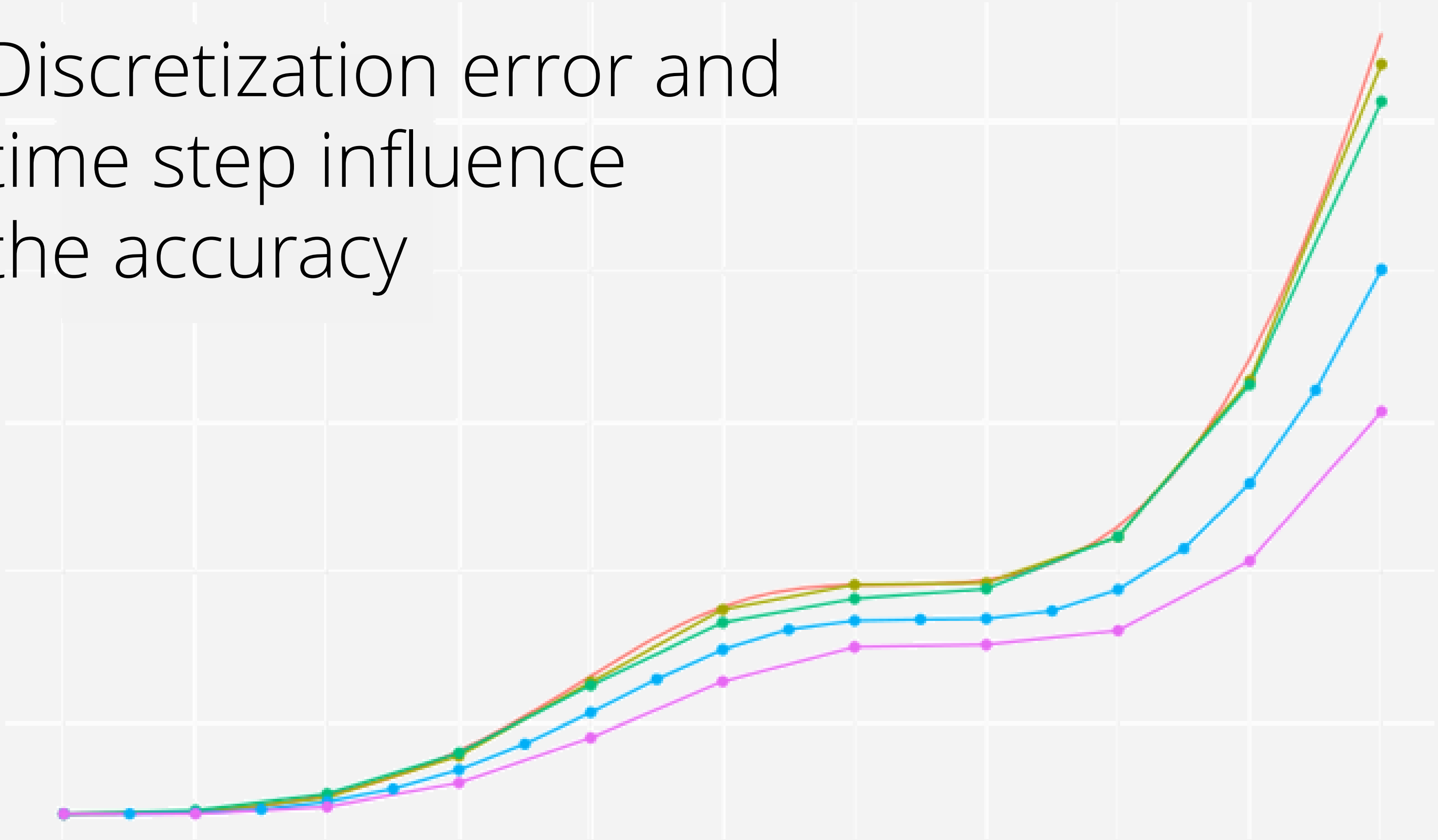


$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\frac{\boldsymbol{v}^t + 3\boldsymbol{v}^* + 3\boldsymbol{v}^{**} + \boldsymbol{v}^{***}}{8}$$

$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\frac{\boldsymbol{a}^t + 3\boldsymbol{a}^* + 3\boldsymbol{a}^{**} + \boldsymbol{a}^{***}}{8} \quad ⑤$$

Accuracy

– Discretization error and time step influence the accuracy



Wikipedia: Runge-Kutta-Verfahren

UNI FREIBURG

Performance

- Computation dominated by derivatives, actually only by the accelerations $\boldsymbol{a}^t, \boldsymbol{a}^*, \boldsymbol{a}^{**}, \boldsymbol{a}^{***}$
- RK4 is four times as expensive as Euler
- RK2 is two times as expensive as Euler
- RK4 is more accurate than RK2 which is more accurate than Euler. Error: $O(h^5) < O(h^3) < O(h^2)$
- RK4 allows larger time steps than RK2 which allows larger times steps than Euler

Performance

- If, e.g., RK4 runs with a time step four times larger than Euler, the overall computation time is the same
  - Comparison: RK4 : Euler
  - Time per simulation step: 4 : 1
  - Simulation steps: 1 : 4
  - Overall computation time: 1 : 1

UNI FREIBURG

Accelerations

- $\boldsymbol{a}^t, \boldsymbol{a}^*, \boldsymbol{a}^{**}, \boldsymbol{a}^{***}$ can be very expensive to compute
- E.g., if the accelerations consider contact forces, collision detection has to be performed four times for different sets of positions $\boldsymbol{x}^t, \boldsymbol{x}^*, \boldsymbol{x}^{**}, \boldsymbol{x}^{***}$

Simulation in Computer Graphics
Particle Motion 2

Matthias Teschner

UNI FREIBURG

Outline

- Introduction
- Particle motion
- Finite differences
- System of first-order ODEs
  - Explicit schemes
  - Predictor-corrector schemes
  - Implicit schemes
- ...

Explicit Adams-Bashforth

- Current and previous accelerations (multistep)
  - Two acceleration computations per step
  - Previous accelerations have to be stored

$$\boldsymbol{v}^* = \boldsymbol{v}^t + \tfrac{h}{2}(3\boldsymbol{a}^t - \boldsymbol{a}^{t-h}) + O(h^3)$$

$$\boldsymbol{v}^* = \boldsymbol{v}^t + \tfrac{h}{12}(23\boldsymbol{a}^t - 16\boldsymbol{a}^{t-h} + 5\boldsymbol{a}^{t-2h}) + O(h^4)$$

$$\boldsymbol{v}^* = \boldsymbol{v}^t + \tfrac{h}{24}(55\boldsymbol{a}^t - 59\boldsymbol{a}^{t-h} + 37\boldsymbol{a}^{t-2h} - 9\boldsymbol{a}^{t-3h}) + O(h^5)$$

$$\boldsymbol{v}^* = \boldsymbol{v}^t + \tfrac{h}{720}(1901\boldsymbol{a}^t - 2774\boldsymbol{a}^{t-h} + 2616\boldsymbol{a}^{t-2h} - 1274\boldsymbol{a}^{t-3h} + 251\boldsymbol{a}^{t-4h}) + O(h^6)$$

$$\boldsymbol{x}^* = \boldsymbol{x}^t + \tfrac{h}{2}(3\boldsymbol{v}^t - \boldsymbol{v}^{t-h}) + O(h^3)$$

$\ldots$

Implicit Adams-Moulton

– Next, current and previous accelerations (multistep)

$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + \tfrac{h}{2}(\boldsymbol{a}^* + \boldsymbol{a}^t) + O(h^3)$$

$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + \tfrac{h}{12}(5\boldsymbol{a}^* + 8\boldsymbol{a}^t - \boldsymbol{a}^{t-h}) + O(h^4)$$

$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + \tfrac{h}{24}(9\boldsymbol{a}^* + 19\boldsymbol{a}^t - 5\boldsymbol{a}^{t-h} + \boldsymbol{a}^{t-2h}) + O(h^5)$$

$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + \tfrac{h}{720}(251\boldsymbol{a}^* + 646\boldsymbol{a}^t - 264\boldsymbol{a}^{t-h} + 106\boldsymbol{a}^{t-2h} - 19\boldsymbol{a}^{t-3h}) + O(h^6)$$

$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + \tfrac{h}{2}(\boldsymbol{v}^* + \boldsymbol{v}^t) + O(h^3)$$

$$\ldots$$

UNI FREIBURG

A Predictor-Corrector Example

- Initialization

$$\boldsymbol{x}^t,\ \boldsymbol{v}^t,\ \boldsymbol{a}^t \qquad \boldsymbol{x}^{t-h} = \boldsymbol{x} - h\boldsymbol{v}^t,\ \boldsymbol{v}^{t-h} = \boldsymbol{v} - h\boldsymbol{a}^t,\ \boldsymbol{a}^{t-h}$$

- Prediction

$$\boldsymbol{x}^* = \boldsymbol{x}^t + \frac{h}{2}(3\boldsymbol{v}^t - \boldsymbol{v}^{t-h}) \qquad \boldsymbol{v}^* = \boldsymbol{v}^t + \frac{h}{2}(3\boldsymbol{a}^t - \boldsymbol{a}^{t-h}) \qquad \boldsymbol{a}^*$$

Accelerations at predicted positions using predicted velocities

- Correction

$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + \frac{h}{2}(\boldsymbol{v}^* + \boldsymbol{v}^t) \qquad \boldsymbol{v}^{t+h} = \boldsymbol{v}^t + \frac{h}{2}(\boldsymbol{a}^* + \boldsymbol{a}^t) \qquad \boldsymbol{a}^{t+h}$$

Discussion

- Two accelerations
- Improved accuracy, larger time steps
  - Not necessarily true for discontinuous functions, e.g., in case of contact handling
- Initialization of previous steps
- Iterative correction steps possible

Outline

- Introduction
- Particle motion
- Finite differences
- System of first-order ODEs
  - Explicit schemes
  - Predictor-corrector schemes
  - Implicit schemes
- ...

UNI FREIBURG

Explicit vs. Implicit Schemes

- Explicit Euler
  $$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\boldsymbol{v}^t$$
  $$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^t$$
- One unknown per equation
- Direct computation of unknowns
- Non-linear equations do not affect the approach
- Non-analytical, procedural forces can be handled

- Implicit Euler
  $$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\boldsymbol{v}^{t+h}$$
  $$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^{t+h}$$
- System of algebraic equations
- Simultaneous computation of unknowns
- Solution of a linear system
- Linearization of non-linear equations

Implicit Schemes

- Challenge
  - Solving a linear system
  - Implementation
- Benefit
  - Largely improved stability
- Issue
  - Reduced accuracy
  - Discretization error plus linearization error plus approximate solution of a linear system

UNI FREIBURG

Implicit Schemes – Example Overview

- Linearization of accelerations

$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^{t+h}(\boldsymbol{x}^{t+h})$$

Here, accelerations depend only on positions.

$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^{t+h}(\boldsymbol{x}^t + h\boldsymbol{v}^{t+h})$$

$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\left(\boldsymbol{a}^t(\boldsymbol{x}^t) + \boldsymbol{J}^t h\boldsymbol{v}^{t+h}\right)$$

J is a 3x3 Jacobi matrix. *h*·*v* is a small displacement. *a*(*x*) + *J*·*h*·*v* is an approximation of the acceleration at position *x* + *h*·*v*.

- Linear system with unknown velocities

$$(\boldsymbol{I} - h^2\boldsymbol{J}^t)\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^t$$

- Position update

$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\boldsymbol{v}^{t+h}$$

Linearization

- $f_{x+\Delta x} = f_x + \frac{\partial f_x}{\partial x}\Delta x + O((\Delta x)^2)$ f: 1D field of scalar values
- $f_{\boldsymbol{x}+\Delta\boldsymbol{x}} = f_{\boldsymbol{x}} + \nabla f_{\boldsymbol{x}} \cdot \Delta\boldsymbol{x} + O(\|\Delta\boldsymbol{x}\|^2)$ f: 3D field of scalar values

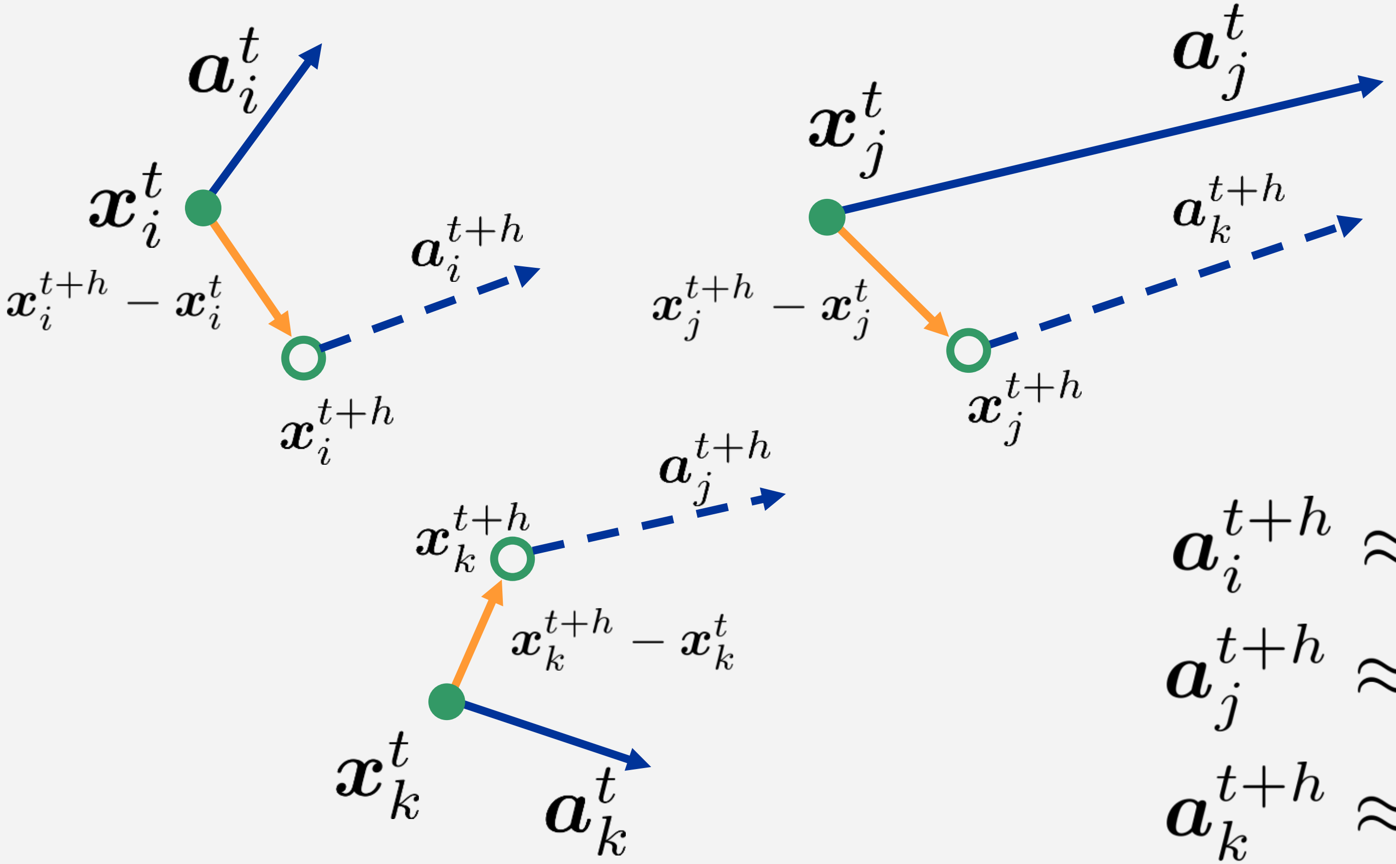
  $$\nabla f_{\boldsymbol{x}} = \left(\frac{\partial f_{\boldsymbol{x}}}{\partial x_1}, \frac{\partial f_{\boldsymbol{x}}}{\partial x_2}, \ldots, \frac{\partial f_{\boldsymbol{x}}}{\partial x_n}\right)^{\mathrm{T}}$$ Gradient

- $\boldsymbol{a}_{\boldsymbol{x}+\Delta\boldsymbol{x}} = \boldsymbol{a}_{\boldsymbol{x}} + \boldsymbol{J}_{\boldsymbol{x}}\Delta\boldsymbol{x} + O(\|\Delta\boldsymbol{x}\|^2)$ a: 3D field of 3D values

  $$\boldsymbol{J}_{\boldsymbol{x}} = \begin{pmatrix} \frac{\partial a_{x_x}}{\partial x_x} & \frac{\partial a_{x_x}}{\partial x_y} & \frac{\partial a_{x_x}}{\partial x_z} \\ \frac{\partial a_{x_y}}{\partial x_x} & \frac{\partial a_{x_y}}{\partial x_y} & \frac{\partial a_{x_y}}{\partial x_z} \\ \frac{\partial a_{x_z}}{\partial x_x} & \frac{\partial a_{x_z}}{\partial x_y} & \frac{\partial a_{x_z}}{\partial x_z} \end{pmatrix}$$ Jacobi matrix

  $$\boldsymbol{a}_{\boldsymbol{x}} = \begin{pmatrix} a_{x_x} \\ a_{x_y} \\ a_{x_z} \end{pmatrix} \quad \boldsymbol{x} = \begin{pmatrix} x_x \\ x_y \\ x_z \end{pmatrix}$$

Jacobi Matrix - Application



$$\boldsymbol{a}_i^{t+h} \approx \boldsymbol{a}_i^t + \boldsymbol{J}_{\boldsymbol{x}_i^t}(\boldsymbol{x}_i^{t+h} - \boldsymbol{x}_i^t)$$
$$\boldsymbol{a}_j^{t+h} \approx \boldsymbol{a}_j^t + \boldsymbol{J}_{\boldsymbol{x}_j^t}(\boldsymbol{x}_j^{t+h} - \boldsymbol{x}_j^t)$$
$$\boldsymbol{a}_k^{t+h} \approx \boldsymbol{a}_k^t + \boldsymbol{J}_{\boldsymbol{x}_k^t}(\boldsymbol{x}_k^{t+h} - \boldsymbol{x}_k^t)$$

Linearization

- Approximation of the acceleration at position $\boldsymbol{x}^{t+h}$ using the acceleration at position $\boldsymbol{x}^t$, the Jacobi matrix $\boldsymbol{J}^t$ of the acceleration at position $\boldsymbol{x}^t$ and the small displacement $\boldsymbol{x}^{t+h} - \boldsymbol{x}^t = h\boldsymbol{v}^{t+h}$:

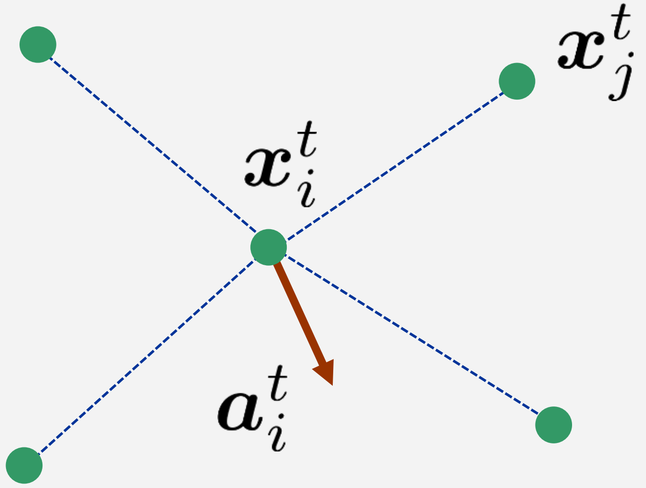
$$\boldsymbol{a}^{t+h}(\boldsymbol{x}^{t+h}) = \boldsymbol{a}^{t+h}(\boldsymbol{x}^t + h\boldsymbol{v}^{t+h}) \approx \boldsymbol{a}^t(\boldsymbol{x}^t) + \boldsymbol{J}^t h\boldsymbol{v}^{t+h}$$

- Equation $\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^{t+h}(\boldsymbol{x}^{t+h})$ with unknown velocities and positions can be rewritten with unknown velocities only: $(\boldsymbol{I} - h^2\boldsymbol{J}^t)\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^t$

UNI FREIBURG

Particle System

- Set of particles with, e.g., interconnecting springs
- Force at a particle depends on particle and its neighbors
- E.g. $\boldsymbol{a}_i^t = \frac{1}{m_i} \sum_j k_{ij} \frac{|\boldsymbol{x}_j^t - \boldsymbol{x}_i^t| - L_{ij}}{L_{ij}} \frac{\boldsymbol{x}_j^t - \boldsymbol{x}_i^t}{|\boldsymbol{x}_j^t - \boldsymbol{x}_i^t|}$
  - Position $\boldsymbol{x}_i^t$, acc. $\boldsymbol{a}_i^t$ and mass m_i of particle i at time t
  - Rest distance L_{ij} and stiffness k_{ij} between particles i and j



UNI FREIBURG

Notation

$$\boldsymbol{x}^t = \begin{pmatrix} x^t_{1,x} \\ x^t_{1,y} \\ x^t_{1,z} \\ x^t_{2,x} \\ x^t_{2,y} \\ x^t_{2,z} \\ \vdots \\ x^t_{n,x} \\ x^t_{n,y} \\ x^t_{n,z} \end{pmatrix} \quad \boldsymbol{v}^t = \begin{pmatrix} v^t_{1,x} \\ v^t_{1,y} \\ v^t_{1,z} \\ v^t_{2,x} \\ v^t_{2,y} \\ v^t_{2,z} \\ \vdots \\ v^t_{n,x} \\ v^t_{n,y} \\ v^t_{n,z} \end{pmatrix} \quad \boldsymbol{a}^t = \begin{pmatrix} a^t_{1,x} \\ a^t_{1,y} \\ a^t_{1,z} \\ a^t_{2,x} \\ a^t_{2,y} \\ a^t_{2,z} \\ \vdots \\ a^t_{n,x} \\ a^t_{n,y} \\ a^t_{n,z} \end{pmatrix}$$

UNI FREIBURG

Linear System – Implicit Euler

- Linear system for n particles
  - $(\boldsymbol{I}_{3n\times 3n} - h^2 \boldsymbol{J}^t)\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^t$
- Jacobian
  - $\boldsymbol{J}^t \in \mathbb{R}^{3n\times 3n}$
  - Spatial derivatives of all accelerations with respect to all positions

Jacobian - Example

– E.g., $\boldsymbol{a}_i^t = \frac{1}{m_i}\frac{k_{ij}}{L_{ij}}(|\boldsymbol{x}_j^t - \boldsymbol{x}_i^t| - L_{ij})\frac{\boldsymbol{x}_j^t - \boldsymbol{x}_i^t}{|\boldsymbol{x}_j^t - \boldsymbol{x}_i^t|} = \frac{1}{m_i}\frac{k_{ij}}{L_{ij}}\left(\boldsymbol{x}_j^t - \boldsymbol{x}_i^t - L_{ij}\frac{\boldsymbol{x}_j^t - \boldsymbol{x}_i^t}{|\boldsymbol{x}_j^t - \boldsymbol{x}_i^t|}\right)$ depends on two positions $\boldsymbol{x}_i^t$ and $\boldsymbol{x}_j^t$

$$\boldsymbol{J}_{i,i}^t = \frac{\partial \boldsymbol{a}_i^t}{\partial \boldsymbol{x}_i^t} = \begin{pmatrix} \frac{\partial a_{i,x}}{\partial x_{i,x}} & \frac{\partial a_{i,x}}{\partial x_{i,y}} & \frac{\partial a_{i,x}}{\partial x_{i,z}} \\ \frac{\partial a_{i,y}}{\partial x_{i,x}} & \frac{\partial a_{i,y}}{\partial x_{i,y}} & \frac{\partial a_{i,y}}{\partial x_{i,z}} \\ \frac{\partial a_{i,z}}{\partial x_{i,x}} & \frac{\partial a_{i,z}}{\partial x_{i,y}} & \frac{\partial a_{i,z}}{\partial x_{i,z}} \end{pmatrix}$$

$$= \frac{\partial}{\partial \boldsymbol{x}_i^t}\frac{1}{m_i}\frac{k_{ij}}{L_{ij}}\left(\boldsymbol{x}_j^t - \boldsymbol{x}_i^t - L_{ij}\frac{\boldsymbol{x}_j^t - \boldsymbol{x}_i^t}{|\boldsymbol{x}_j^t - \boldsymbol{x}_i^t|}\right)$$

$$= \frac{1}{m_i}\frac{k_{ij}}{L_{ij}}\left(-\boldsymbol{I} + \frac{L_{ij}}{|\boldsymbol{x}_j^t - \boldsymbol{x}_i^t|}\left(\boldsymbol{I} - \frac{1}{|\boldsymbol{x}_j^t - \boldsymbol{x}_i^t|^2}(\boldsymbol{x}_j^t - \boldsymbol{x}_i^t)(\boldsymbol{x}_j^t - \boldsymbol{x}_i^t)^{\mathrm{T}}\right)\right)$$

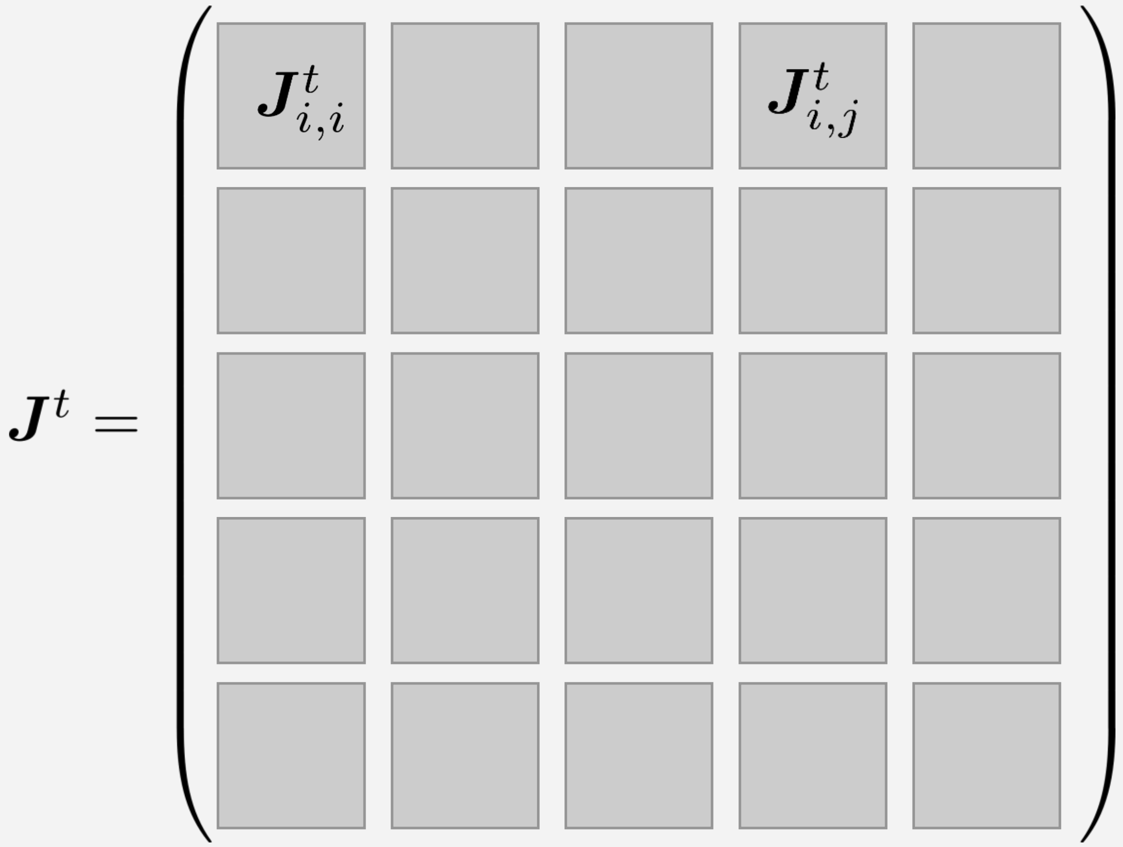
$$\boldsymbol{J}_{i,j}^t = -\frac{m_i}{m_j}\boldsymbol{J}_{i,i}^t$$

Mueller et al. , Real-time Physics. SIGGRAPH 2008.

UNI FREIBURG

Jacobian

- $\boldsymbol{J}^t \in \mathbb{R}^{3n \times 3n}$ is built from 3x3 matrices $\boldsymbol{J}^t_{i,j} \in \mathbb{R}^{3 \times 3}$
- If position $\boldsymbol{x}^t_j$ influences acceleration $\boldsymbol{a}^t_i$, then $\boldsymbol{J}^t_{i,j} \neq \boldsymbol{0}$
- Otherwise, $\boldsymbol{J}^t_{i,j} = \boldsymbol{0}$



Solver

- $\underbrace{(\boldsymbol{I}_{3n\times 3n} - h^2\boldsymbol{J}^t)}_{\boldsymbol{A}}\boldsymbol{v}^{t+h} = \underbrace{\boldsymbol{v}^t + h\boldsymbol{a}^t}_{\boldsymbol{s}}$
- Iterative. Start with a guess, e.g. $\boldsymbol{v}^0 = \boldsymbol{v}^t$
- Iterative updates $\boldsymbol{v}^0 \rightarrow \boldsymbol{v}^1 \rightarrow \ldots \rightarrow \boldsymbol{v}^l$
- Result $\boldsymbol{v}^{t+h} = \boldsymbol{v}^l$

Here, superscript indicates the iteration.

UNI FREIBURG

Solver – Conjugate Gradient

$$l = 0$$

$$\boldsymbol{d}^l = \boldsymbol{r}^l = \boldsymbol{s} - \boldsymbol{A}\boldsymbol{v}^l$$

$$\alpha^l = \frac{\boldsymbol{r}^l \cdot \boldsymbol{r}^l}{\boldsymbol{d}^l \cdot (\boldsymbol{A}\boldsymbol{d}^l)}$$ Scaling factor for the solution update.

$$\boldsymbol{v}^{l+1} = \boldsymbol{v}^l + \alpha^l \boldsymbol{d}^l$$ Update of the solution with a scaled direction.

$$\boldsymbol{r}^{l+1} = \boldsymbol{r}^l - \alpha^l \boldsymbol{A}\boldsymbol{d}^l$$ Residual. Exit loop, when sufficiently small.

$$\boldsymbol{d}^{l+1} = \boldsymbol{r}^{l+1} + \frac{\boldsymbol{r}^{l+1} \cdot \boldsymbol{r}^{l+1}}{\boldsymbol{r}^l \cdot \boldsymbol{r}^l} \boldsymbol{d}^l$$ Direction for the solution update.

$$l = l + 1$$ Iteration count.

UNI FREIBURG

Solver – Conjugate Gradient

- Works (converges) for symmetric, positive-definite matrices
- Exact solution of an $n \times n$ system in n steps
- Frequently used for deformable objects
- Typically used with a fixed iteration count, e.g. 3-5

Solver - Jacobi

- $\boldsymbol{v}^{l+1} = \boldsymbol{v}^l + \omega \boldsymbol{D}^{-1}(\boldsymbol{s} - \boldsymbol{A}\boldsymbol{v}^l)$
- $\boldsymbol{D}$ diagonal elements of $\boldsymbol{A}$
- ω determines convergence and convergence rate
- $0 \leq \omega \leq 2$, in practical settings typically $\omega = 0.5$
- Per-component update

$$\begin{aligned} v_i^{l+1} &= (1-\omega)v_i^l + \tfrac{\omega}{A_{ii}}(s_i - \textstyle\sum_{j\neq i} A_{ij}v_j^l) \\ &= (1-\omega)v_i^l + \tfrac{\omega}{A_{ii}}(s_i - (\boldsymbol{A}\boldsymbol{v}^l)_i + A_{ii}v_i^l) \\ &= v_i^l + \tfrac{\omega}{A_{ii}}(s_i - (\boldsymbol{A}\boldsymbol{v}^l)_i) \end{aligned}$$

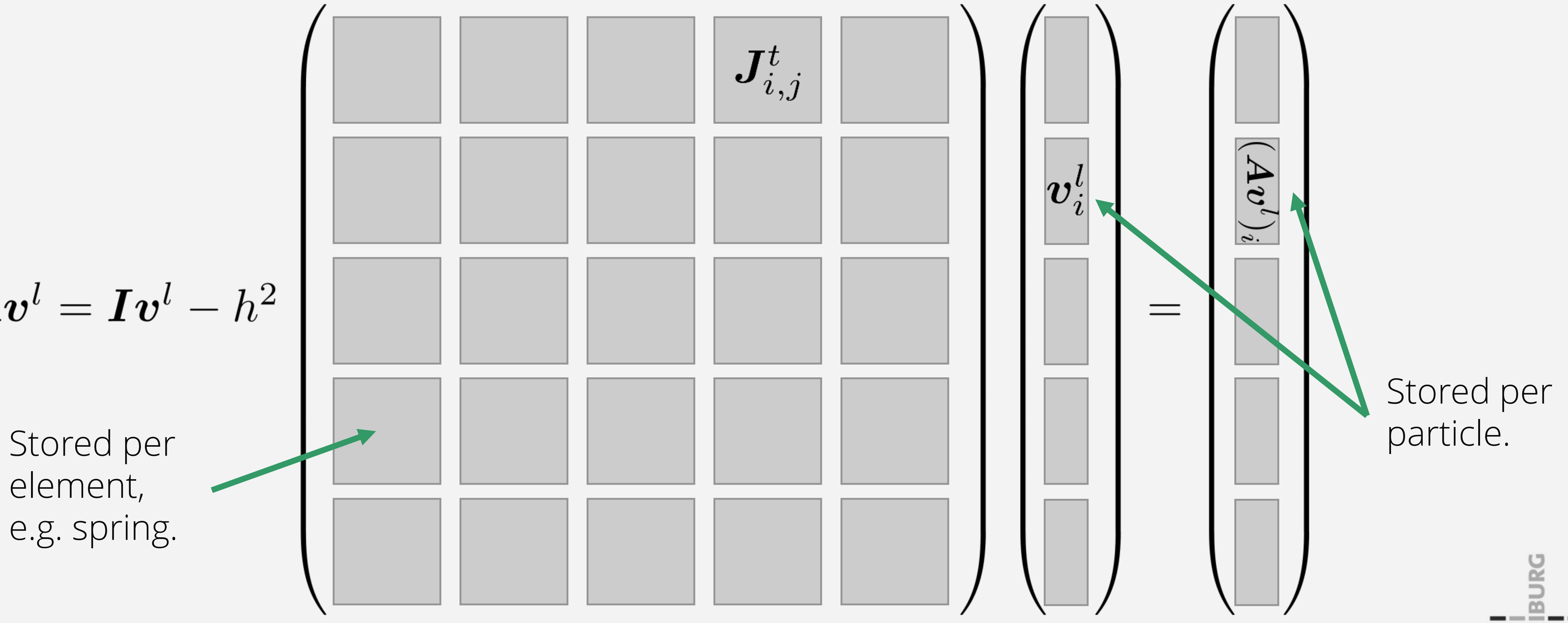
Solver - Implementation

- $\boldsymbol{A} = \boldsymbol{I}_{3n\times 3n} - h^2 \boldsymbol{J}^t$ is not explicitly built or stored
- $\boldsymbol{s} = \boldsymbol{v}^t + h\boldsymbol{a}^t$ is not explicitly built or stored
- Instead
  - Per-particle information is stored at particles, e.g. $\boldsymbol{s}_i$
  - Per-element information is stored at elements, e.g. $\boldsymbol{J}^t_{i,i}$ for an elastic spring between i and j, $\boldsymbol{J}^t_{i,j} = \boldsymbol{J}^t_{j,i} = -\frac{m_i}{m_j}\boldsymbol{J}^t_{i,i}$ and $\boldsymbol{J}^t_{j,j} = \frac{m_i}{m_j}\boldsymbol{J}^t_{i,i}$ can be reconstructed
  - Matrix-free implementation of solver steps

Solver - Implementation

– $\boldsymbol{A}\boldsymbol{v}^l$ is computed and stored per particle

$$\boldsymbol{A}\boldsymbol{v}^l = \boldsymbol{I}\boldsymbol{v}^l - h^2$$



Solver - Implementation

- $\boldsymbol{A}\boldsymbol{v}^l$ is computed by iterating over elements
- E.g., spring connects particles j and k

For each particle:

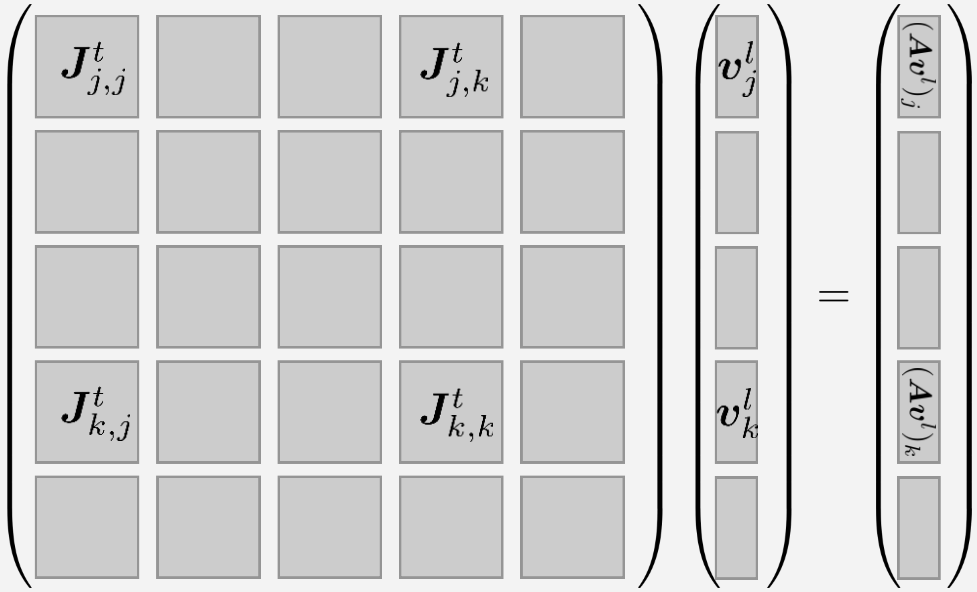
$$(\boldsymbol{A}\boldsymbol{v}^l)_i = \boldsymbol{v}_i^l$$

For each spring:

$$(\boldsymbol{A}\boldsymbol{v}^l)_j += -h^2 \boldsymbol{J}_{j,j}^t \boldsymbol{v}_j^l + h^2 \underbrace{\frac{m_j}{m_k} \boldsymbol{J}_{j,j}^t}_{-\boldsymbol{J}_{j,k}^t} \boldsymbol{v}_k^l$$

$$(\boldsymbol{A}\boldsymbol{v}^l)_k += -h^2 \underbrace{\frac{m_j}{m_k} \boldsymbol{J}_{j,j}^t}_{\boldsymbol{J}_{k,k}^t} \boldsymbol{v}_k^l + h^2 \underbrace{\frac{m_j}{m_k} \boldsymbol{J}_{j,j}^t}_{-\boldsymbol{J}_{k,j}^t} \boldsymbol{v}_j^l$$

$$\boldsymbol{A}\boldsymbol{v}^l = \boldsymbol{I}\boldsymbol{v}^l - h^2$$



Solver - Discussion

- Jacobi vs. Conjugate Gradient CG:
  - CG converges faster
  - Jacobi is good-natured, e.g. in case of clamping intermediate solutions to implement constraints
- Implementation, e.g., in a particle-spring model
  - Matrix-free
  - All solver information is stored at particles and springs
  - All solver steps are realized by iterating over particles and springs

Implicit Schemes – Summary

- Implicit Euler

$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^{t+h}(\boldsymbol{x}^{t+h})$$

- Linearization

$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\left(\boldsymbol{a}^t(\boldsymbol{x}^t) + \boldsymbol{J}^t h\boldsymbol{v}^{t+h}\right)$$

- Solve a linear system for velocities

$$(\boldsymbol{I} - h^2\boldsymbol{J}^t)\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^t$$

- Update positions according to implicit Euler

$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\boldsymbol{v}^{t+h}$$

UNI FREIBURG

Semi-implicit Euler (Euler-Cromer)

- Explicit Euler for the velocity update

  $$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\boldsymbol{a}^t$$

- Implicit Euler for the position update

  $$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\boldsymbol{v}^{t+h}$$

- No linear system

UNI FREIBURG

Simulation in Computer Graphics
Particle Motion 3

Matthias Teschner

UNI FREIBURG

Outline

- Introduction
- Particle motion
- Finite differences
- System of first-order ODEs
- Second-order ODE
- Performance
- Discussion

UNI FREIBURG

Initial Value Problem of Second Order

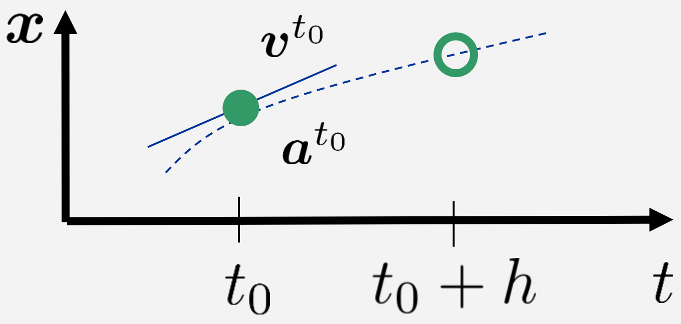
– Function $\boldsymbol{x}^t$ represents the particle motion

– Second-order differential equation is given

$$\frac{\mathrm{d}^2\boldsymbol{x}^t}{\mathrm{d}t^2} = \boldsymbol{a}^t$$

– Initial values $\boldsymbol{x}^{t_0}$ and $\boldsymbol{v}^{t_0}$ are given

– How to estimate $\boldsymbol{x}^{t_0+h}$?



Motivation

- Schemes for coupled first-order ODEs update $\boldsymbol{x}$ and $\boldsymbol{v}$ simultaneously
- Schemes for second-order ODEs update $\boldsymbol{x}$, but not necessarily $\boldsymbol{v}$

Verlet

– Taylor approximations of $\boldsymbol{x}^{t+h}$ and $\boldsymbol{x}^{t-h}$

$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\boldsymbol{v}^t + \frac{h^2}{2}\boldsymbol{a}^t + \frac{h^3}{6}\frac{\mathrm{d}^3\boldsymbol{x}^t}{\mathrm{d}t^3} + O(h^4)$$

$$\boldsymbol{x}^{t-h} = \boldsymbol{x}^t - h\boldsymbol{v}^t + \frac{h^2}{2}\boldsymbol{a}^t - \frac{h^3}{6}\frac{\mathrm{d}^3\boldsymbol{x}^t}{\mathrm{d}t^3} + O(h^4)$$

– Adding both approximations

$$\boldsymbol{x}^{t+h} = 2\boldsymbol{x}^t - \boldsymbol{x}^{t-h} + h^2\boldsymbol{a}^t + O(h^4)$$

$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\frac{\boldsymbol{x}^t - \boldsymbol{x}^{t-h}}{h} + h^2\boldsymbol{a}^t + O(h^4)$$

Verlet - Discussion

- One acceleration computation per step
  - Same computation cost as explicit Euler
- Discretization error of order 4
  - More accurate than explicit Euler
- Larger time step and improved performance compared to explicit Euler

Verlet - Discussion

- Velocity representation not necessarily required
- But:
  - Velocity typically used for collision handling and damping
  - E.g. $\boldsymbol{v}^{t+h} = \frac{\boldsymbol{x}^{t+h} - \boldsymbol{x}^{t}}{h} + O(h)$

Leap-Frog

$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\boldsymbol{v}^{t+\frac{h}{2}}$$

$$\boldsymbol{v}^{t+\frac{3h}{2}} = \boldsymbol{v}^{t+\frac{h}{2}} + h\boldsymbol{a}^{t+h}$$

– Implementation, e.g.

$$\boldsymbol{v}^{t+\frac{h}{2}} = \frac{\boldsymbol{x}^{t+h} - \boldsymbol{x}^t}{h} \quad \boldsymbol{v}^{t-\frac{h}{2}} = \frac{\boldsymbol{x}^t - \boldsymbol{x}^{t-h}}{h}$$

$$\Rightarrow (\boldsymbol{v}^{t+\frac{h}{2}} - \boldsymbol{v}^{t-\frac{h}{2}})h = \boldsymbol{x}^{t+h} - \boldsymbol{x}^t - \boldsymbol{x}^t + \boldsymbol{x}^{t-h}$$

$$\Rightarrow \boldsymbol{a}^t h^2 = \boldsymbol{x}^{t+h} - \boldsymbol{x}^t - \boldsymbol{x}^t + \boldsymbol{x}^{t-h}$$

$$\Rightarrow \boldsymbol{x}^{t+h} = 2\boldsymbol{x}^t - \boldsymbol{x}^{t-h} + \boldsymbol{a}^t h^2$$ Verlet

UNI FREIBURG

Velocity Verlet

- Same accuracy for position and velocity

$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\boldsymbol{v}^t + \frac{h^2}{2}\boldsymbol{a}^t + O(h^3)$$
$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + \frac{h}{2}\left(\boldsymbol{a}^t + \boldsymbol{a}^{t+h}\right) + O(h^3)$$

- One acceleration computation per step

Beeman

$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + h\boldsymbol{v}^t + h^2\left(\tfrac{2}{3}\boldsymbol{a}^t - \tfrac{1}{6}\boldsymbol{a}^{t-h}\right) + O(h^4)$$

$$\boldsymbol{v}^{t+h} = \boldsymbol{v}^t + h\left(\tfrac{5}{12}\boldsymbol{a}^{t+h} + \tfrac{2}{3}\boldsymbol{a}^t - \tfrac{1}{12}\boldsymbol{a}^{t-h}\right) + O(h^4)$$

- One acceleration computation per step
- Improved accuracy compared to Velocity Verlet
- Possibly larger time step

UNI FREIBURG

Gear

- Taylor approximation

$$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + \frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t}\frac{h}{1!} + \frac{\mathrm{d}^2\boldsymbol{x}^t}{\mathrm{d}t^2}\frac{h^2}{2!} + \frac{\mathrm{d}^3\boldsymbol{x}^t}{\mathrm{d}t^3}\frac{h^3}{3!} + \frac{\mathrm{d}^4\boldsymbol{x}^t}{\mathrm{d}t^4}\frac{h^4}{4!} + \frac{\mathrm{d}^5\boldsymbol{x}^t}{\mathrm{d}t^5}\frac{h^5}{5!} + \dots$$

- Notation

$$\boldsymbol{r}_k^t = \frac{\mathrm{d}\boldsymbol{x}^k}{\mathrm{d}t^k}\frac{h^k}{k!}$$

$$\boldsymbol{r}_0^{t+h} = \boldsymbol{r}_0^t + \boldsymbol{r}_1^t + \boldsymbol{r}_2^t + \boldsymbol{r}_3^t + \boldsymbol{r}_4^t + \boldsymbol{r}_5^t + \dots$$

UNI FREIBURG

Gear

- $$\boldsymbol{x}^{t+h} = \boldsymbol{x}^t + \frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t}\frac{h}{1!} + \frac{\mathrm{d}^2\boldsymbol{x}^t}{\mathrm{d}t^2}\frac{h^2}{2!} + \frac{\mathrm{d}^3\boldsymbol{x}^t}{\mathrm{d}t^3}\frac{h^3}{3!} + \frac{\mathrm{d}^4\boldsymbol{x}^t}{\mathrm{d}t^4}\frac{h^4}{4!} + \frac{\mathrm{d}^5\boldsymbol{x}^t}{\mathrm{d}t^5}\frac{h^5}{5!} + \dots$$
  $$\boldsymbol{r}_0^{t+h} = \boldsymbol{r}_0^t + \boldsymbol{r}_1^t + \boldsymbol{r}_2^t + \boldsymbol{r}_3^t + \boldsymbol{r}_4^t + \boldsymbol{r}_5^t + \dots$$
- $$h\frac{\mathrm{d}\boldsymbol{x}^{t+h}}{\mathrm{d}t} = h\frac{\mathrm{d}\boldsymbol{x}^t}{\mathrm{d}t} + h\frac{\mathrm{d}^2\boldsymbol{x}^t}{\mathrm{d}t^2}\frac{h}{1!} + h\frac{\mathrm{d}^3\boldsymbol{x}^t}{\mathrm{d}t^3}\frac{h^2}{2!} + h\frac{\mathrm{d}^4\boldsymbol{x}^t}{\mathrm{d}t^4}\frac{h^3}{3!} + h\frac{\mathrm{d}^5\boldsymbol{x}^t}{\mathrm{d}t^5}\frac{h^4}{4!} + \dots$$
  $$\boldsymbol{r}_1^{t+h} = \boldsymbol{r}_1^t + 2\boldsymbol{r}_2^t + 3\boldsymbol{r}_3^t + 4\boldsymbol{r}_4^t + 5\boldsymbol{r}_5^t + \dots$$
- $$\frac{h^2}{2}\frac{\mathrm{d}^2\boldsymbol{x}^{t+h}}{\mathrm{d}t^2} = \frac{h^2}{2}\frac{\mathrm{d}^2\boldsymbol{x}^t}{\mathrm{d}t^2} + \frac{h^2}{2}\frac{\mathrm{d}^3\boldsymbol{x}^t}{\mathrm{d}t^3}\frac{h}{1!} + \frac{h^2}{2}\frac{\mathrm{d}^4\boldsymbol{x}^t}{\mathrm{d}t^4}\frac{h^2}{2!} + \frac{h^2}{2}\frac{\mathrm{d}^5\boldsymbol{x}^t}{\mathrm{d}t^5}\frac{h^3}{3!} + \dots$$
  $$\boldsymbol{r}_2^{t+h} = \boldsymbol{r}_2^t + 3\boldsymbol{r}_3^t + 6\boldsymbol{r}_4^t + 10\boldsymbol{r}_5^t + \dots$$

UNI FREIBURG

Gear - Prediction

$$
\begin{aligned}
\boldsymbol{x}^{t+h} = \boldsymbol{r}_0^{t+h} &= \boldsymbol{r}_0^t + \boldsymbol{r}_1^t + \boldsymbol{r}_2^t + \boldsymbol{r}_3^t + \boldsymbol{r}_4^t + \boldsymbol{r}_5^t \\
h\boldsymbol{v}^{t+h} = \boldsymbol{r}_1^{t+h} &= \boldsymbol{r}_1^t + 2\boldsymbol{r}_2^t + 3\boldsymbol{r}_3^t + 4\boldsymbol{r}_4^t + 5\boldsymbol{r}_5^t \\
\frac{h^2}{2}\boldsymbol{a}^{t+h} = \boldsymbol{r}_2^{t+h} &= \boldsymbol{r}_2^t + 3\boldsymbol{r}_3^t + 6\boldsymbol{r}_4^t + 10\boldsymbol{r}_5^t \\
\boldsymbol{r}_3^{t+h} &= \boldsymbol{r}_3^t + 4\boldsymbol{r}_4^t + 10\boldsymbol{r}_5^t \\
\boldsymbol{r}_4^{t+h} &= \boldsymbol{r}_4^t + 5\boldsymbol{r}_5^t \\
\boldsymbol{r}_5^{t+h} &= \boldsymbol{r}_5^t
\end{aligned}
$$

UNI FREIBURG

Gear - Correction

- Error / inconsistency between the predicted acceleration $\frac{2}{h^2}\boldsymbol{r}_2^{t+h}$ at time $t+h$ and the acceleration $\boldsymbol{a}^{t+h}(\boldsymbol{r}_0^{t+h}, \frac{1}{h}\boldsymbol{r}_1^{t+h})$ at predicted positions $\boldsymbol{r}_0^{t+h}$ and velocities $\frac{1}{h}\boldsymbol{r}_1^{t+h}$:

  $$\boldsymbol{\epsilon}^{t+h} = \boldsymbol{r}_2^{t+h} - \frac{h^2}{2}\boldsymbol{a}^{t+h}$$

- Correction:

  $$\boldsymbol{r}_k^{t+h} = \boldsymbol{r}_k^{t+h} - c_k \boldsymbol{\epsilon}^{t+h}$$

  with coefficients

  $$c_0 = \frac{3}{20}, c_1 = \frac{251}{360}, c_2 = 1, c_3 = \frac{11}{18}, c_4 = \frac{1}{6}, c_5 = \frac{1}{60}$$

Gear - Implementation

- Initialization: $\boldsymbol{r}_0^{t_0} = \boldsymbol{x}^{t_0} \quad \boldsymbol{r}_1^{t_0} = h\boldsymbol{v}^{t_0} \quad \boldsymbol{r}_2^{t_0} = \frac{h^2}{2}\boldsymbol{a}^{t_0}$

  $\boldsymbol{r}_3^{t_0} = \boldsymbol{r}_4^{t_0} = \boldsymbol{r}_5^{t_0} = 0$
- Prediction: $\boldsymbol{r}_0^{t+h} = \boldsymbol{r}_0^t + \ldots + \boldsymbol{r}_5^t$

  $\boldsymbol{r}_k^{t+h} = \ldots$
- Error: $\boldsymbol{\epsilon}^{t+h} = \boldsymbol{r}_2^{t+h} - \frac{h^2}{2}\boldsymbol{a}^{t+h}$
- Correction: $\boldsymbol{r}_k^{t+h} = \boldsymbol{r}_k^{t+h} - c_k\boldsymbol{\epsilon}^{t+h}$

UNI FREIBURG

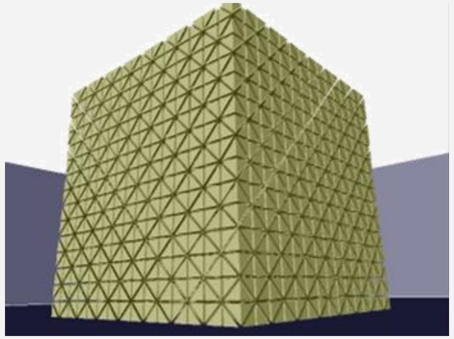
Outline

- Introduction
- Particle motion
- Finite differences
- System of first-order ODEs
- Second-order ODE
- Performance
- Discussion

UNI FREIBURG

Comparison

– Deformable cube on a plane (4k particles, 17k tetrahedra, 22k edges), spring forces, volume preservation, gravity, contact



Scheme	Error order	Time step [ms]	Computation time [ms]	Ratio
Explicit Euler	1	0.5	9.5	0.05
RK 2	2	3.8	18.9	0.20
Implicit Euler	1	49.0	172.0	0.28
RK 4	4	17.0	50.0	0.34
Verlet	3	11.5	9.5	1.21

Time Step

- Larger time steps are generally advantageous for the performance
- However, the time step size is limited: $h \leq \frac{d}{|\boldsymbol{v}|}$
- A particle should not move farther than its size in one simulation step, e.g. its diameter d: $h|\boldsymbol{v}| \leq d$

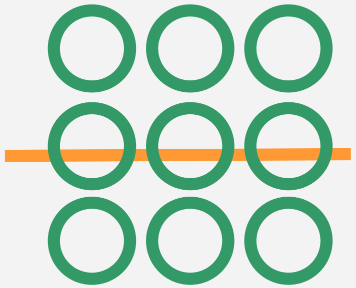
Time Step

- Critical states that can be avoided by a time step limit
  - Inverted elements
  - Unresolvable contacts

Rest state

Inverted elements

State at *t*:
No contact



State at *t*+*h*:
Contact

Outline

– Introduction
– Particle motion
– Finite differences
– System of first-order ODEs
– Second-order ODE
– Performance
– Discussion

Explicit Schemes

- Error order determines accuracy
- Improved accuracy may correspond to an improved stability for larger time steps
- Improved accuracy may correspond to higher costs
- Time steps are comparatively small
- Stability is generally an issue

Implicit Schemes

- Generally stable and robust
- Handle larger time steps
- Less accurate (scheme, linearization, solver)
  - Typically artificial damping / viscosity
- Decreasing accuracy for larger time steps
  - Same as for explicit schemes, but explicit schemes get unstable, while implicit schemes stay stable