ACM SIGGRAPH / Eurographics Symposium on Computer Animation 2020
J. Bender and T. Popa
(Guest Editors)

*Volume 39 (2020), Number 8*

# Higher-Order Time Integration for Deformable Solids

Fabian Löschner, Andreas Longva, Stefan Jeske, Tassilo Kugelstadt and Jan Bender

RWTH Aachen University, Germany

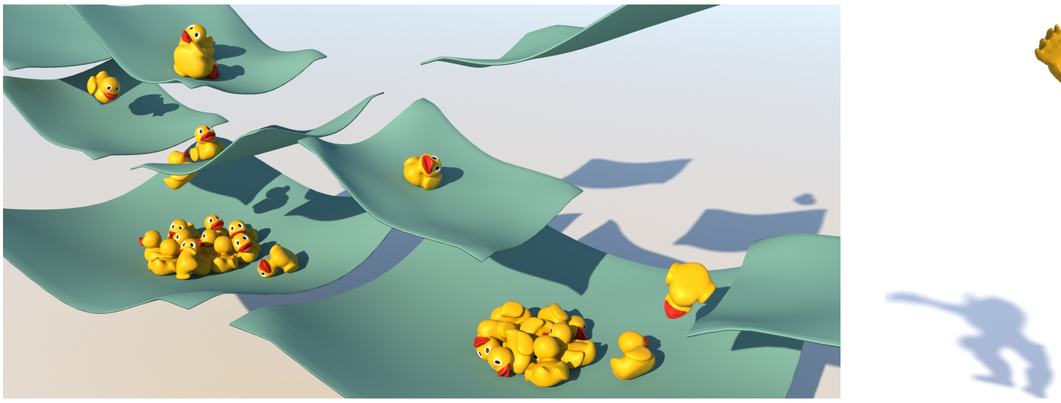


**Figure 1:** *We use higher-order time integration together with our constraint-based contact model to robustly simulate complex scenes with many interactions. This allows us to more accurately capture and control detail inherent to low damping of deformable dynamic motion.*

**Abstract**

*Visually appealing and vivid simulations of deformable solids represent an important aspect of physically based computer animation. For the temporal discretization, it is customary in computer animation to use first-order accurate integration methods, such as Backward Euler, due to their simplicity and robustness. Although there is notable research on second-order methods, their use is not widespread. Many of these well-known methods have significant drawbacks such as severe numerical damping or scene-dependent time step restrictions to ensure stability. In this paper, we discuss the most relevant requirements on such methods in computer animation and motivate the interest beyond first-order accuracy. Keeping these requirements in mind, we investigate several promising methods from the families of diagonally implicit Runge-Kutta (DIRK) and Rosenbrock methods which currently do not appear to have considerable popularity in this field. We show that the usage of such methods improves the visual quality of physical animations. In addition, we demonstrate that they allow distinctly more control over damping at lower computational cost than classical methods. As part of our theoretical contribution, we review aspects of simulations that are often considered more intricate with higher-order methods, such as contact handling. To this end, we derive an implicit linearized contact model based on a predictor-corrector approach that leads to consistent behavior with higher-order integrators as predictors. Our contact model is well suited for the simulation of stiff, nonlinear materials with the integration methods presented in this paper and more common methods such as Backward Euler alike.*

**CCS Concepts**

• ***Computing methodologies*** $\rightarrow$ ***Physical simulation;***

## 1. Introduction

The dynamic simulation of deformable solids is a core component in animation, special effects, virtual prototyping, video games and many other applications in computer graphics. At the heart of this task is the *time integrator*, which is responsible for advancing the simulation from one time step to the next. The choice of time integrator is fundamentally a balancing act between stability and numerical accuracy, and the right choice is highly application-dependent.

In order to help both practitioners and researchers in picking the right integrator for their application, we consider integrators in terms of the following quality criteria:

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd. Published by John Wiley & Sons Ltd. This is the accepted version of the following article: "F. LÖSCHNER, A. LONGVA, S. JESKE, T. KUGELSTADT, J. BENDER: Higher-Order Time Integration for Deformable Solids. In *Computer Graphics Forum* (2020)", which has been published in final form at http://onlinelibrary.wiley.com. This article may be used for non-commercial purposes in accordance with the Wiley Self-Archiving Policy.

- *Stability*, which prevents spurious, unnatural oscillations or total explosions, particularly in challenging scenarios.
- *Reliability*, which indicates whether the subproblems (e.g., nonlinear equations) associated with the integrator can be reliably solved.
- *Performance*, the expected computational cost of the method. For interactive applications especially, it is also of crucial concern to have *predictable* performance.
- *Accuracy*, which measures the degree of convergence towards a ground truth solution, as well as visual plausibility for approximate solutions.
- *Energy conservation*, the ability of the integrator to preserve the inherent underlying energy in the dynamical system.

The first-order Backward Euler integrator is arguably the most popular choice in computer graphics (e.g. [BW98, MG04, MTGG11, KBT17, KKB18]) owing to its reliability even in very demanding scenarios. However, its stability comes at the cost of excessive numerical dissipation, which in many cases precludes fine, local details such as folds and wrinkles in cloth, or produces strongly globally damped motion for otherwise vividly animated objects. While some researchers in graphics have employed selected higher-order time integrators, a more systematic investigation into their use in graphics applications is not available.

Our main contribution is a discussion and comparison of promising higher-order time integrators for the simulation of deformable solids in computer graphics. We study the family of implicit Runge-Kutta (RK) methods (Section 3), and from the vast amount of possible choices, we identify a small selection of promising higher-order methods that are expected to perform well across all our quality criteria (Section 5). We evaluate the selected integrators on problems in numerical benchmark simulations arising from finite element discretizations (Section 5.3), as well as in more complex scenarios with contacts (Sections 5.4 and 5.5). Through our review and test scenarios, we demonstrate that several of the selected integrators, most notably the *SDIRK2* integrator, clearly outperform Backward Euler in most scenarios by remaining stable, yet at the same time much more faithfully reproducing the dynamics of the system without the excessive numerical dissipation that plagues Backward Euler. The drastically reduced numerical damping furthermore enables much greater control over damping behavior, which is crucial to both plausibly and accurately simulate motion for real-world materials.

As discussed later, many existing constraint-based contact handling schemes use a particular linearization of the nonlinear internal forces, essentially precluding the use of higher-order integrators. Our second contribution is a robust predictor-corrector constraint-based contact model that is designed to work well in synergy with higher-order predictors (Section 4). Due to our particular choice of linearization, it avoids unduly affecting the predicted dynamics of the system when correcting for contacts. We demonstrate our contact solver in conjunction with the second-order method SDIRK2 in multiple scenes.

## 2. Related work

Solving ordinary differential equations (ODE) is important in many scientific disciplines. The book of Hairer and Wanner [HW96] offers a good introduction, especially for implicit Runge-Kutta and Rosenbrock methods which are well suited for stiff problems.

In computer graphics the numerical integration of ODEs plays an important role in the simulation of deformable solids. In their seminal work, Terzopoulos et al. [TPBF87] introduce the Backward Euler method to the graphics community. In later works from the 90s most authors prefer explicit methods like symplectic Euler [VCM95] or Runge-Kutta [EWS96] because of their computation speed. However, for stiff materials which are common, e.g., stiff springs in cloth simulations, explicit methods quickly become unstable when large time steps are used. Baraff and Witkin [BW98] demonstrate that the stability issues can be remedied by using implicit methods such as linearized Backward Euler. This enables stable and efficient simulations by using large time steps. However, numerical errors lead to undesired numerical damping which cannot be controlled by the user. Since then, improving the performance, stability, and reducing the numerical damping of implicit integration methods is ongoing research. Several authors propose to use higher-order multi-step methods like BDF2 [HE01, HES03, CK02, BMM17] which suffer less from numerical damping. However, they rely on the polynomial interpolation of states from the previous time steps. Multi-step methods and their stability properties are derived based on the assumption that the interpolated trajectory is sufficiently smooth [HW92]. In general, however, contact forces that are necessary to prevent interpenetration are inherently non-smooth. Violating these assumptions could lead to visual artifacts or even a breakdown of the simulation.

Integrators which preserve the symplectic structure of the mechanical system are attractive candidates for reducing numerical damping [LMOW04, KWT*06, SD08, SG09]. Further examples are Verlet, implicit midpoint [HEE*02] and Newmark methods which use the trapezoidal rule [BMF03, GHDS03]. These methods have good energy conservation properties and the energy oscillates about its correct value [HLW06]. However, it has been pointed out by several authors that these oscillations can be quite drastic such that simulations with large time steps become unstable [XB17, RLK18].

Another approach to prevent numerical damping is the use of asynchronous integrators which use smaller time steps in regions where more accuracy is required [TPS08, SKZF11, ZLB16]. However, these methods require significant implementation overhead and so far only explicit or first-order implicit methods have been explored. Other authors propose integrators that conserve energy exactly. This can be achieved by enforcing a constant energy constraint using Lagrange multipliers [SSF13]. Another approach is to track the total energy of the system and blend between the solutions of implicit midpoint and Backward Euler such that the energy is conserved [DLK18]. Dinev et al. [DLL*18] use an optimization procedure to project the state of the physical system such that energy, linear and angular momentum are conserved. Note that this post-processing step can also be combined with the proposed implicit Runge-Kutta and Rosenbrock methods.

The recently introduced exponential integrators [MSW14, MLT17, CAP17] also provide good stability and energy conservation properties. They solve the linear part of the ODE exactly by computing a matrix exponential and treat the remaining nonlinear part numerically. This is especially beneficial when stiff compo-

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

nents of the system are contained in the linear part. However, these methods are quite involved and they are only applicable when mass lumping is used which is not always possible, e.g., in higher-order or embedded finite element simulations.

In recent years, there has been a lot of research on optimization based integrators. Martin et al. [MTGG11] show that the Backward Euler method can be formulated as an optimization problem. This enables to speed up computations by using efficient optimization methods like alternating local-global solvers [LBBK13, BML*14], Newton's method with sophisticated line search strategies [GSS*15], the Chebyshev Semi-Iterative approach [Wan15], ADMM [OBLN17], L-BFGS [LBK17], or domain decomposition [LGL*19]. Moreover there are specialized material models which are well suited to be used with optimization integrators [KKB18]. It has been pointed out that there is a connection between compliant constraint systems and optimization based integration of Backward Euler [TNGF15]. This connection can be also shown for the position based dynamics method [MMC16]. All of these methods solve Backward Euler as an optimization problem with the goal of solving the involved (non-)linear systems faster and more robustly. They are orthogonal to our work since each stage of the higher-order DIRK methods can be formulated as an optimization problem as well (as long as the forces are derived from a potential energy).

In summary, implicit higher-order single-step methods, like Runge-Kutta and Rosenbrock, have not gained much attention in the graphics community. They are mentioned in the works of Hauth et al. [HE01, HES03] as being too costly for animation purposes. However, they do not discuss which methods they used nor show any experiments to prove this statement. One other example of Runge-Kutta methods can be found in the work of Xu and Barbič [XB17]. They use a variant of the TR-BDF2 method which is a combination of the trapezoidal rule and BDF2 and they report good stability with low numerical damping. As the method uses BDF2 only internally, it is still a single-step method and therefore does not necessarily suffer from the same problems as a multi-step method. Accordingly, this method can be rewritten as a second-order implicit Runge-Kutta method. In the following we will discuss several modern implicit higher-order Runge-Kutta and Rosenbrock methods which are specifically designed for the integration of stiff systems and show that they are well suited for animation tasks like simulations of deformable solids.

## 3. Runge-Kutta methods

In this section, we review Runge-Kutta methods (RK) applied to the dynamical systems associated with deformable solids, with a special focus on the subclasses DIRK and Rosenbrock methods.

### 3.1. Dynamical systems for deformable solids

We consider the solution of dynamical systems in the time interval $t \in [t^n, t^{n+1}]$, i.e. the solution of a single time step, described by differential equations of the form

$$M\ddot{\boldsymbol{u}}(t) = \boldsymbol{f}_{\text{ext}}(t) + \boldsymbol{f}_{\text{int}}(\boldsymbol{u}(t)) - D\dot{\boldsymbol{u}}(t)\,. \tag{1}$$

Here $M$ is the positive definite mass matrix of the system, $\boldsymbol{f}_{\text{ext}}$ represents the time-dependent external forces acting on the system, $\boldsymbol{f}_{\text{int}}$ represents the state-dependent internal forces of the system and $D$ is a symmetric positive semi-definite matrix that encodes the material damping behavior of the system. For the damping model that we consider, $D$ is constant in the time interval, but it can take different values for different time steps. We focus on dynamical systems that arise from the discretization of deformable solids with finite elements. In this case, $\boldsymbol{u}$ represents the degrees of freedom for the displacement field of the solid. We consider both fully nonlinear and linearized corotational material models, and employ Rayleigh damping $D = \alpha M + \beta K(\boldsymbol{u}^n)$, where $K(\boldsymbol{u}^n)$ is the tangent stiffness matrix evaluated at the beginning of the time step. For nonlinear material models, $K$ is not in general semi-definite. We therefore modify $K$ such that it becomes semi-definite by clamping negative eigenvalues to zero during element matrix assembly, similar to the approach of Teran et al. [TSIF05].

### 3.2. Integration methods

Stability is perhaps the foremost criterion in picking an integrator for most computer graphics applications. Therefore, we restrict ourselves to methods that are L-stable. L-stability essentially means that a given stability function decays to 0 as the time step goes to infinity [HW96], when applied to a linear model problem. In practice, this means that the method has a tendency towards a numerically damped response rather than blowing up when the time step is large. Essentially all real-world materials in graphics applications exhibit some degree of damping in their elastic response. Therefore, we eschew perfect energy conservation in favor of *sufficiently good* energy conservation, and similarly exclude symplectic methods, since their energy conservation properties tend to translate into instabilities for highly nonlinear problems. Finally, we only consider single-step methods. This excludes multi-step integrators like BDF2. Multi-step methods are somewhat less suitable for systems with discontinuous events, such as collisions. They also have higher implementation complexity, as one needs to track the state of the previous simulation step.

#### 3.2.1. Backward Euler

To introduce notation and further motivate the investigation we start with the well-known Backward Euler method. Let us first consider the first-order system of ODEs

$$\dot{\boldsymbol{y}} = \boldsymbol{g}(t, \boldsymbol{y})\,, \tag{2}$$

with $\boldsymbol{g}(t) \in \mathbb{R}^d$. From an initial value $\boldsymbol{y}^n = \boldsymbol{y}(t^n)$ the Backward Euler method approximates the solution of this system at the next step $\boldsymbol{y}^{n+1} = \boldsymbol{y}(t^n + \Delta t)$ as

$$\boldsymbol{y}^{n+1} = \boldsymbol{y}^n + \Delta t\, \boldsymbol{g}(t^{n+1}, \boldsymbol{y}^{n+1})\,, \tag{3}$$

which, in general, requires the solution of a nonlinear system. It is well known that Backward Euler is probably the most stable integration method but this comes at the cost of strong numerical damping.

**Equations of motion** To apply the method to the dynamical system in (1), we rewrite it as a first-order system

$$M\dot{\boldsymbol{v}} = \boldsymbol{f}(t, \boldsymbol{u}, \boldsymbol{v}), \quad \dot{\boldsymbol{u}} = \boldsymbol{v}, \tag{4}$$

© 2020 The Author(s)

where $\boldsymbol{f}(t,\boldsymbol{u},\boldsymbol{v})$ contains all forces in the system. Then, the equations for $\boldsymbol{v}$ and $\boldsymbol{u}$ can be discretized followed by substituting the latter into the former. The resulting implicit system can then be solved using Newton's method which results in the update

$$\left(M-\Delta t\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}}-\Delta t^2\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right)\Delta\boldsymbol{v}_{k+1} = \Delta t\,\boldsymbol{f}(t^{n+1},\boldsymbol{u}_k^{n+1},\boldsymbol{v}_k^{n+1}) - M(\boldsymbol{v}_k^{n+1}-\boldsymbol{v}^n)\,, \tag{5}$$

where $k$ denotes the iteration index and $\Delta\boldsymbol{v}_{k+1}=\boldsymbol{v}_{k+1}^{n+1}-\boldsymbol{v}_k^{n+1}$. In general, the Jacobians $\partial\boldsymbol{f}/\partial\boldsymbol{v}$ and $\partial\boldsymbol{f}/\partial\boldsymbol{u}$ have to be evaluated in every iteration for an exact Newton method.

In the context of deformable solids, the Jacobians in Equation (5) are given by $\frac{\partial f}{\partial \boldsymbol{v}}=-D$ and $\frac{\partial f}{\partial \boldsymbol{x}}(\boldsymbol{u})=-K(\boldsymbol{u})$. As introduced in Section 3.1, we defined the damping matrix $D$ to be constant over a time step. Therefore, it is only evaluated once for our choice of model. This also applies to the methods introduced in the following sections.

### 3.2.2. General Runge-Kutta methods

A large and well-studied class of single-step methods that offers many candidates with high orders of accuracy are Runge-Kutta (RK) methods [HW92]. RK methods compute the solution at the next step $\boldsymbol{y}^{n+1}$ as a linear combination of intermediate stage values that are obtained by evaluating the right hand side $\boldsymbol{g}(t,\boldsymbol{y})$. Any $s$-stage RK method can be written as

$$\begin{aligned}\boldsymbol{y}^{n+1}&=\boldsymbol{y}^n+\Delta t\sum_i^s b_i\boldsymbol{G}_i\,,\\ \boldsymbol{G}_i&=\boldsymbol{g}(t^n+c_i\Delta t,\boldsymbol{y}^n+\Delta t\sum_{j=1}^s a_{ij}\boldsymbol{G}_j)\,,\end{aligned}\tag{6}$$

for $i=1,\dots,s$, and $\boldsymbol{G}_i$ are the aforementioned stage values. In theory, they can be derived with arbitrary order of accuracy and the set of coefficients leave a large design space to enforce different properties on construction. A specific integration scheme is then identified by the set of coefficients $a_{ij}$, $b_i$, $c_i$. The pattern of the coefficient matrix $A=(a_{ij})$ affects the numerical properties of the method and how involved the procedure to compute $\boldsymbol{y}^{k+1}$ is. This leads to the distinction of three main categories of RK methods.

**Explicit methods** A strictly lower triangular matrix $A$ yields fully explicit methods known as explicit RK methods (ERKs). Explicit methods are only conditionally stable depending on the time step size, and therefore only have niche use cases in computer graphics.

**Fully implicit methods** Methods with a (nearly) full matrix $A$ are known as fully implicit RK methods (FIRKs). In general, FIRKs require the solution of nonlinear systems of dimension $sd\times sd$ where $s$ is the number of stages of the method and $d$ the dimension of the ODE system. The cost and difficulty associated with solving these non-linear equations often outweigh their benefits. We did not pursue the direction of FIRKs further.

The third category of methods, known as Diagonally implicit Runge-Kutta (DIRK) methods, will be introduced in the next section.

### 3.2.3. Diagonally implicit Runge-Kutta methods

The third category of Runge-Kutta methods is identified by a lower triangular coefficient matrix $A$, which are called diagonally implicit RK (DIRK) methods. These methods are still implicit but the stages only depend sequentially on each other. They offer a promising combination of computational efficiency and stability, and therefore they are of particular interest for our investigation. For a $s$-stage DIRK method, $s$ systems of size $d\times d$ have to be solved in succession instead of a single large system. Following this definition, the formula for a general RK method given in Equation (6) can be simplified to

$$\begin{aligned}\boldsymbol{y}^{n+1}&=\boldsymbol{y}^n+\Delta t\sum_i^s b_i\boldsymbol{G}_i\,,\\ \boldsymbol{G}_i&=\boldsymbol{g}(t^n+c_i\Delta t,\boldsymbol{y}^n+\Delta t\sum_{j=1}^{i-1} a_{ij}\boldsymbol{G}_j+a_{ii}\Delta t\boldsymbol{G}_i)\,.\end{aligned}\tag{7}$$

A thorough review of the theoretical background, implementation details and numerical experiments w.r.t. DIRK methods was presented by Kennedy and Carpenter [KC16].

**Equations of motion** We want to apply a general DIRK method to the equations of motion (4). For stage $i$, let $\boldsymbol{U}_i$ and $\boldsymbol{V}_i$ denote the stage values of the displacement and the velocity, respectively. Then, the implicit equations for the stage values are

$$\boldsymbol{U}_i=\tilde{\boldsymbol{v}}_i^n+a_{ii}\Delta t\boldsymbol{V}_i\,, \tag{8a}$$

$$M\boldsymbol{V}_i=\boldsymbol{f}(t^n+c_i\Delta t,\tilde{\boldsymbol{u}}_i^n+a_{ii}\Delta t\boldsymbol{U}_i,\tilde{\boldsymbol{v}}_i^n+a_{ii}\Delta t\boldsymbol{V}_i)\,, \tag{8b}$$

where $\tilde{\boldsymbol{u}}_i^n$ and $\tilde{\boldsymbol{v}}_i^n$ denote the already known part of the system state at stage $i$ given by

$$\tilde{\boldsymbol{u}}_i^n=\boldsymbol{u}^n+\Delta t\textstyle\sum_j^{i-1}a_{ij}\boldsymbol{U}_j \quad\text{and}\quad \tilde{\boldsymbol{v}}_i^n=\boldsymbol{v}^n+\Delta t\sum_j^{i-1}a_{ij}\boldsymbol{V}_j\,. \tag{9}$$

Substituting Equation (8a) into (8b) and applying Newton's method yields the sequence of linear systems

$$\left(M-a_{ii}\Delta t\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}}-(a_{ii}\Delta t)^2\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right)\Delta\boldsymbol{V}_{i,k+1}=-M\boldsymbol{V}_{i,k} + \boldsymbol{f}(t^n+c_i\Delta t,\tilde{\boldsymbol{u}}_i^n+a_{ii}\Delta t\boldsymbol{U}_{i,k},\tilde{\boldsymbol{v}}_i^n+a_{ii}\Delta t\boldsymbol{V}_{i,k})\,, \tag{10}$$

where $k$ denotes the Newton iteration index and $\Delta\boldsymbol{V}_{i,k+1}=\boldsymbol{V}_{i,k+1}-\boldsymbol{V}_{i,k}$. The corresponding displacement stage values are obtained with back-substitution into Equation (8a). The evaluation of the Jacobians is treated, per stage, analogous to Backward Euler, as described in Section 3.2.1. As an initial guess for the Newton iterations, we use $\boldsymbol{V}_{i,0}=0$. Note that the stage values for the velocities have to be interpreted as a form of acceleration, not velocity, which is why the previous stage value is not necessarily a good choice. Alternatively, it is possible to derive a more accurate initial guess using interpolation that could save some iterations (see [HW96]).

### 3.2.4. Rosenbrock methods

In computer animation it is also common, for performance reasons or for solving linear problems (e.g. when using an approximate corotational model), to use a linearization of the Backward Euler method. Rosenbrock methods extend this idea to arbitrary order and can be used to obtain a linearized solution to any nonlinear ODE system. Rosenbrock methods can be seen as a DIRK method

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

that uses only a single Newton iteration per stage with a zero initial value. A computationally efficient formulation can be found in the book by Hairer and Wanner [HW96].

**Equations of motion** We want to apply a Rosenbrock scheme to the equations of motion. This can be done by concatenating the system (4) to a single two-dimensional system over the state $(\boldsymbol{u},\boldsymbol{v})^T$, computing the Jacobians required by the Rosenbrock formulation and plugging them in. Similar to the derivation for the DIRK method, we can define the already known part of the system state at stage $i$ as

$$\hat{\boldsymbol{u}}_i^n = \boldsymbol{u}^n + \textstyle\sum_j^{i-1} a_{ij}\boldsymbol{U}_j \quad \text{and} \quad \hat{\boldsymbol{v}}_i^n = \boldsymbol{v}^n + \textstyle\sum_j^{i-1} a_{ij}\boldsymbol{V}_j. \tag{11}$$

Then, the velocity stage values $\boldsymbol{V}_i$ are given as solution of the linear equation system

$$\begin{aligned}\left(\frac{1}{\gamma_{ii}h}M - \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}} - h\gamma_{ii}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right)\boldsymbol{V}_i &= \boldsymbol{f}(t^n + \alpha_i\Delta t, \hat{\boldsymbol{u}}_i^n, \hat{\boldsymbol{v}}_i^n) \\ &+ M\sum_j^{i-1}\frac{c_{ij}}{\Delta t}\boldsymbol{V}_j + \gamma_i\Delta t\frac{\partial \boldsymbol{f}}{\partial t} + \gamma_{ii}\Delta t(\hat{\boldsymbol{v}}_i^n + \sum_j^{i-1}\frac{c_{ij}}{\Delta t}\boldsymbol{U}_j),\end{aligned} \tag{12}$$

where $a_{ij}$, $c_{ij}$, $m_i$ and $\gamma_{ij}$ and $\gamma_i = \sum_j^i \gamma_{ij}$ are the coefficients identifying a particular Rosenbrock scheme. In the literature, sometimes other formulations of Rosenbrock methods are used such that one has to take care to correctly transform the coefficients for the implementation. The Jacobians $\partial \boldsymbol{f}/\partial \boldsymbol{u}$ and $\partial \boldsymbol{f}/\partial \boldsymbol{v}$ and the derivative $\partial \boldsymbol{f}/\partial t$ (in case of forces with an explicit time dependency) of the right hand side $\boldsymbol{f}$ of the system (4) appear in the formulation above. For Rosenbrock methods, they are all only evaluated once at the beginning of the time step. After solving this system, the displacement stage values can be obtained explicitly using

$$\boldsymbol{U}_i = \gamma_{ii}\Delta t\left(\sum_j^{i-1}\frac{c_{ij}}{\Delta t}\boldsymbol{U}_j + \hat{\boldsymbol{v}}_i^n + \boldsymbol{V}_i\right). \tag{13}$$

## 4. Contact model

To our knowledge, the realization of a robust and stable contact model that incorporates higher-order integrators in a fully nonlinear model is still an open research problem. We instead consider a linearized model, in the hope that it may provide new insights and provide a stepping stone toward fully non-linear models in the future. To maintain stability, it is crucial to ensure that the linearization corresponds to an implicit treatment of the internal forces. However, existing approaches [OTSG09, TNGF15, BCDA11, LDN*18, VJ19] directly employ particularly chosen linearizations, precluding direct use of an arbitrary higher-order integrator. Moreover, the direct linearization of a higher-order integrator cannot be expected to remain stable in challenging scenarios.

We note that very recently, Li et al. [LFS*20] proposed Incremental Potential Contact, which is compatible with time integrators that can be reformulated as an *incremental potential* (IP) problem. However, the stability properties of the barrier-augmented IP for higher-order integrators — especially in the presence of highly non-smooth contact forces — have not yet been investigated.

We propose a predictor-corrector model in which the predictor for the unconstrained displacements and velocities can be arbitrarily chosen — e.g. a higher-order integrator — while the corrector relies on a carefully chosen linearization similar to (linearized) Backward Euler, ensuring robustness and stability. This choice ensures that the contact constraints are satisfied without unduly affecting the dynamics of the system.

We follow an approach similar to the predictor-corrector method used for the computation of the normal forces in the Staggered Projections (SP) [KSJP08] method, which was originally designed for rigid bodies and reduced-order soft bodies. In this context, the predictor corresponds to an unconstrained time integration of the dynamic system including possibly non-linear elastic forces, while the corrector enforces the contact constraints. However, the formulation used by SP is not stable for stiff materials discretized by finite elements, as the elastic forces would essentially be treated explicitly. In other words, the corrector used by SP does not take elastic stiffness into account when enforcing the contact constraints.

The corrector should also in some sense not deviate from the predicted solution more than necessary. In particular, if no correction is necessary, the corrector should leave the predicted solution intact. However, straightforward linearization of the internal forces about the state at time $t^n$, as done by most existing works, yields a corrector that does not have this property. We show that we can construct a corrector with this property by carefully choosing the point at which the elastic forces are linearized. Moreover, our proposed corrector effectively penalizes deviation from the predicted state by considering elastic stiffness, which is instrumental in ensuring stability.

We add contact forces represented by $G(t)^T\boldsymbol{\lambda}(t)$ to the differential equation (1) and concatenate state vectors of the bodies in the system to obtain the velocity-level differential complementarity problem

$$\begin{aligned}M\ddot{\boldsymbol{u}}(t) =& \boldsymbol{f}_{\text{ext}}(t) + \boldsymbol{f}_{\text{int}}(\boldsymbol{u}(t)) - D\dot{\boldsymbol{u}}(t) + G(t)^T\boldsymbol{\lambda}(t) \\ & 0 \le G(t)\,\dot{\boldsymbol{u}}(t) \perp \boldsymbol{\lambda} \ge 0.\end{aligned} \tag{14}$$

Here $G(t)$ encodes the non-negativity constraints for relative velocities between contact points on the bodies in the system.

We denote by $\boldsymbol{u}^p = \boldsymbol{u}^p(t)$ the predicted displacement corresponding to the solution of the differential equation (1) in the time interval $t \in [t^n, t^{n+1}]$ with any of the presented integrators. That is, $\boldsymbol{u}^p$ solves the unconstrained differential equation. Subtracting the left and right sides of the unconstrained differential equation (1) from the differential equation in (14), we obtain the equivalent differential equation

$$M(\ddot{\boldsymbol{u}} - \ddot{\boldsymbol{u}}^p) = \boldsymbol{f}_{\text{int}}(\boldsymbol{u}) - \boldsymbol{f}_{\text{int}}(\boldsymbol{u}^p) - D(\dot{\boldsymbol{u}} - \dot{\boldsymbol{u}}^p) + G^T\boldsymbol{\lambda}, \tag{15}$$

where for brevity we have omitted the explicit time dependencies. To simplify the notation in the following, we redefine $\boldsymbol{u}^p$ such that $\boldsymbol{u}^p \leftarrow \boldsymbol{u}^p(t^{n+1})$. We choose a discretization similar to Backward Euler, but fix the contact constraint matrix $G = G(t^n)$ at the current time step. Since $\boldsymbol{u}^n = \boldsymbol{u}(t^n) = \boldsymbol{u}^p(t^n)$ and $\boldsymbol{v} = \dot{\boldsymbol{u}}$, we obtain

$$\begin{aligned}M\frac{\boldsymbol{v}^{n+1} - \boldsymbol{v}^p}{\Delta t} =& \boldsymbol{f}_{\text{int}}(\boldsymbol{u}^{n+1}) - \boldsymbol{f}_{\text{int}}(\boldsymbol{u}^p) \\ & - D(\boldsymbol{v}^{n+1} - \boldsymbol{v}^p) + G^T\boldsymbol{\lambda}^{n+1}.\end{aligned} \tag{16}$$

© 2020 The Author(s)

The formulation (16) is still nonlinear and difficult to solve when taking the constraints into account. We define $\Delta\boldsymbol{u} = \boldsymbol{u}^{n+1} - \boldsymbol{u}^p$ and write $\boldsymbol{u}^{n+1} = \boldsymbol{u}^p + \Delta\boldsymbol{u}$, and subsequently make the particular choice of linearization

$$\boldsymbol{f}_{\text{int}}(\boldsymbol{u}^{n+1}) = \boldsymbol{f}_{\text{int}}(\boldsymbol{u}^p + \Delta\boldsymbol{u}) \approx \boldsymbol{f}_{\text{int}}(\boldsymbol{u}^p) + \frac{\partial \boldsymbol{f}_{\text{int}}}{\partial \boldsymbol{u}}(\boldsymbol{u}^p)\Delta\boldsymbol{u}. \quad (17)$$

Denoting by $K = -\frac{\partial \boldsymbol{f}_{\text{int}}}{\partial \boldsymbol{u}}(\boldsymbol{u}^p)$ the stiffness matrix evaluated at the predicted state, we observe that with the above choice of linearization applied to (16), the internal force terms vanish entirely:

$$\begin{aligned} M\frac{\boldsymbol{v}^{n+1} - \boldsymbol{v}^p}{\Delta t} &= -K(\boldsymbol{u}^{n+1} - \boldsymbol{u}^p) \\ &\quad - D(\boldsymbol{v}^{n+1} - \boldsymbol{v}^p) + G^T\boldsymbol{\lambda}^{n+1}. \end{aligned} \quad (18)$$

This is an important property, because it implies that when $\boldsymbol{\lambda}^{n+1} = 0$, we have that $\boldsymbol{v}^{n+1} = \boldsymbol{v}^p$. More generally, if $(G^T\boldsymbol{\lambda}^{n+1})_I = 0$ for some index subset $I$ corresponding to a particular body, then $\boldsymbol{v}_I^{n+1} = \boldsymbol{v}_I^p$. In other words, the correction process leaves bodies untouched if no contact forces are necessary, preserving the convergence properties of the integration method. Similarly, body parts that are not in contact and not significantly affected by stiff internal forces due to contact in other body parts are also largely unaffected by the correction process.

With $\boldsymbol{u}^{n+1} = \boldsymbol{u}^n + \Delta t\boldsymbol{v}^{n+1}$, we can now formulate the discrete constrained dynamics as

$$C\boldsymbol{v}^{n+1} - G^T\tilde{\boldsymbol{\lambda}}^{n+1} - c = 0, \qquad 0 \leq G\boldsymbol{v}^{n+1} \perp \tilde{\boldsymbol{\lambda}}^{n+1} \geq 0 \quad (19)$$

with $C := M + \Delta tD + (\Delta t)^2K$, $\tilde{\boldsymbol{\lambda}}^{n+1} := \Delta t\boldsymbol{\lambda}^{n+1}$ and $\boldsymbol{c} := M\boldsymbol{v}^p + \Delta tK(\boldsymbol{u}^p - \boldsymbol{u}^n) + \Delta tD\boldsymbol{v}^p$. We now assume that $K$ and $D$ are positive semi-definite, which implies that $C$ is positive definite. If $K$ is not positive definite, we replace it with a projected version as described in Section 3.1. In this case, it is well known that the conditions defined by (19) are the first-order optimality (KKT) conditions [NW06] for the Quadratic Program (QP) given by

$$\begin{aligned} &\min \frac{1}{2}\boldsymbol{v}^T C\boldsymbol{v} - \boldsymbol{c}^T\boldsymbol{v} \\ &\text{s.t.} \quad G\boldsymbol{v} \geq 0. \end{aligned} \quad (20)$$

Note that superscripts have been omitted in the above QP. We here consider velocities as the primal variable and construct a quadratic program directly for $\boldsymbol{v}$, so that $\tilde{\boldsymbol{\lambda}}$ takes the role of dual variables. This is in contrast to the formulation used by, e.g., Staggered Projections, in which the dual formulation is used. However, the dual formulation would require that the matrix $C$ in the QP be replaced by the Schur complement $GC^{-1}G^T$, which is intractable to form explicitly, and moreover would almost certainly be very dense. The primal formulation presented above, on the other hand, retains the sparsity of the finite element formulation.

### 4.1. Scaling

Optimization algorithms generally perform better when the problem data is scaled well. This is particularly true for first-order methods. To improve scaling of the problem, we therefore formulate a preconditioned variant of the contact QP (20). First, we note that we can freely replace the constraints by $EG\boldsymbol{v} \geq 0$ for any choice of $E$ diagonal and positive definite. We choose $E_{ii} = (\max_j |G_{ij}|)^{-1}$, so that the largest absolute value in the $i$th row is 1. This essentially normalizes the constraints. This is particularly effective if the constraint formulation used depends on, e.g., the area of triangles on the contact surface, in which case large differences in triangle size may lead to poor scaling of the constraints.

Next, we write $\boldsymbol{v} = P^{1/2}\hat{\boldsymbol{v}}$, where $P = \operatorname{diag}(C)^{-1}$ is the inverse of the diagonal matrix consisting of the diagonal entries of $C$. This is just the usual Jacobi preconditioner, which is particularly effective in the presence of elements of different sizes in the finite element discretization. The preconditioned QP formulation finally becomes

$$\begin{aligned} &\min \frac{1}{2}\hat{\boldsymbol{v}}^T P^{1/2}CP^{1/2}\hat{\boldsymbol{v}} - \boldsymbol{c}^T P^{1/2}\hat{\boldsymbol{v}} \\ &\text{s.t.} \quad EGP^{1/2}\hat{\boldsymbol{v}} \geq 0. \end{aligned} \quad (21)$$

### 4.2. Implementation details

We solve the preconditioned QP (21) with OSQP [SBG*20], a first-order solver for fast approximate solutions to QPs that robustly handles the case when $G$ is singular or close to singular, which it might very well be if some constraints are (nearly) redundant. In our current implementation, we only use vertex-point constraints, i.e. we constrain the relative velocity at a vertex of one mesh that is in contact with the other body to be non-negative. For two bodies $A$ and $B$, we consider contacts both ways; vertices on $A$ against $B$ and vertices on $B$ against A.

## 5. Evaluation

The following sections contain the evaluation of a selection of integration methods. In Sections 5.1 and 5.2, we first introduce the specific methods that we want to investigate. This is followed by the description and evaluation of experiments without contacts in Section 5.3 which is concluded with an intermediate summary for the investigated methods. Next, we focus on the interaction of the higher-order integrators with our contact model, starting with a simple example suitable for quantitative analysis in Section 5.4. Finally, we present more complex scenes in Section 5.5, demonstrating the practical usability of higher-order methods in computer animation.

### 5.1. DIRK methods

For this paper we collected several DIRK methods from the literature that, on paper, sound like good candidates to fulfill the desired properties introduced in Section 1. The candidates include methods of specific types of DIRK methods, in particular singly diagonally implicit RK (SDIRK) methods where all diagonal entries $a_{ii}$ of the coefficient matrix $A$ have the same value and DIRK methods with an explicit first stage known as EDIRK or ESDIRK methods. For more details on the properties of these methods, we refer again to [KC16]. In the following discussion we will consider the following DIRK methods:

- SDIRK2 (two-stage, second-order) and SDIRK3 (three-stage, third-order) were the first methods designated as DIRK methods [Ale77]. It was noted that both methods are stiffly-accurate and L-stable [KC16].

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

- The TR-BDF2 integrator originally proposed by Banks et al. [BCF*85] performs a fractional substep from $t^n$ to $t^n + \gamma\Delta t$ with the trapezoidal rule for $\gamma \in (0,1)$ and then uses a variant of the BDF2 integrator with the states at $t^n$ and $t^{n+\gamma}$. It can be written as a family of EDIRK methods (three-stage, second-order, see [BDR17]). Banks et al. suggested $\gamma = 2 - \sqrt{2}$ and proved that the method is L-stable in this case. The specific case of $\gamma = \frac{1}{2}$ was later introduced to nonlinear structural dynamics by Bathe [Bat07] (henceforth denoted TR-BDF2(B)), and also recently employed in computer graphics by Xu and Barbič [XB17] for better control of damping behavior when animating deformable bodies. However, this variant is not L-stable [BC00]. In our evaluation we will focus on the latter variant, TR-BDF2(B), due to its previous usage in our field. In the future, however, it might be interesting to investigate the practical implications of the stronger theoretical stability properties of the original TR-BDF2 method.
- SDIRK-NCS23 (two-stage, third-order) and SDIRK-NC34 (three-stage, fourth-order) were the first DIRK methods that were investigated for their nonlinear stability and are algebraically stable and A-stable. It was noted that SDIRK-NCS23 is not stiffly-accurate [KC16].
- SDIRK(3,3,4,5) and SDIRK(4,3,4,7) are derived as methods suited well for ODEs with oscillating solutions [FGR97]. They follow the naming scheme DIRK($s$,$p$,$p_{disp}$,$p_{diss}$) where $s$ is the number of stages, $p$ the classical order of accuracy, $p_{disp}$ the dispersion order and $p_{diss}$ the dissipation order. Both methods are A-stable and were tested with a low dimensional, nonlinear oscillatory problem in the original publication.

SDIRK methods have the property that the Newton iteration matrix at each stage is the same with the exception of the change in the derivative of $\boldsymbol{g}$ (and $\boldsymbol{f}$ for the equations of motion), which is a property that can be exploited by e.g., Quasi-Newton methods. We also briefly investigated the low dissipation, low dispersion methods ILDDDIRK22 (two-stage, second-order) et al. (up to fifth order) [GS20], but we found them to be far less reliable than other methods.

### 5.2. Rosenbrock methods

For this paper we investigated two Rosenbrock methods. ROS3PL is a four-stage, third-order method that is L-stable and stiffly accurate. The authors derived it as a W-method that permits inexact Jacobians and claim that it is robust against order reduction [CLW09]. In addition, it has an embedded method for error estimation.

More recent publications have shown that previously used conditions to prevent order reduction for very stiff problems were not sufficient. Rang [Ran15] derives new variants of commonly used third-order methods including ROS3PL and, amongst others, presents ROS3PRL2 that is shown to preserve third-order accuracy in numerical experiments. In our experiments, however, we were not able to obtain stable results with this method. In the publication it was not indicated if the newly derived method has the same stability properties as the original ROS3PL. Still, we want to further investigate this in the future as the premise of this modified method sounds interesting.

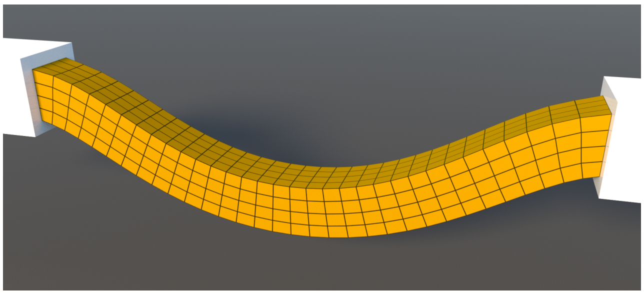


**Figure 2:** *The double-clamped beam experiment: discretized with* $4 \times 4 \times 32 = 512$ *trilinear hexahedral elements, the beam swings under the influence of gravity.*

### 5.3. Experiments without contacts

To gain an overview of the practical performance of the selected schemes for stiff deformable bodies without interference of our particular contact model, we initially restrict ourselves to experiments without contacts. We implemented the integration methods and our in-house FEM framework in Rust using the “nalgebra” library for dense matrix-vector operations [C*19]. For the solution of nonlinear systems, we use Newton’s method with a line search based on the residual norm. To solve the involved linear systems, we used the sparse direct solvers provided in the Intel MKL.

As material models, we employed the Stable Neo-Hookean model (as introduced in [SDGK18]) and an approximate corotated linear elastic model (based on [MG04]). For the Stable Neo-Hookean model, we observed that for most of our experiments it was not necessary to perform a projection of the stiffness matrix to ensure positive definiteness. This saves the performance overhead of the projection operation itself, but also positively affects the number of Newton iterations as element matrix based projections lead to an inexact Jacobian if the current iterate is still far from the solution. As a fallback for the rare case that the Cholesky solver detects an indefinite system, we switch to an LU based solver.

For the corotated linear elastic material, we used a formulation with inversion handling and an approximation that fixes the rotation at the beginning of the time step. An alternative for single-step methods with multiple stages would be to recompute the rotation for every evaluation of the right hand side. In initial experiments with DIRK methods, however, this prevented convergence of the Newton iterations unless a more accurate stiffness matrix is provided, i.e. one which incorporates the derivative of the rotations (see e.g. [Bar12]). Such a fully nonlinear corotational model, however, is not common practice in computer animation, as this would be much less efficient than nonlinear material models with better properties.

In some experiments we explicitly add damping. As we want to focus on the integrator and the contact model, we have chosen to use a simple Rayleigh damping model as introduced in Section 3.1. However, it would alternatively be possible to use more advanced models (e.g. [XB17]).

**Double-clamped beam** We start our experimental evaluation with a simple and controllable example of a double-clamped cantilever

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

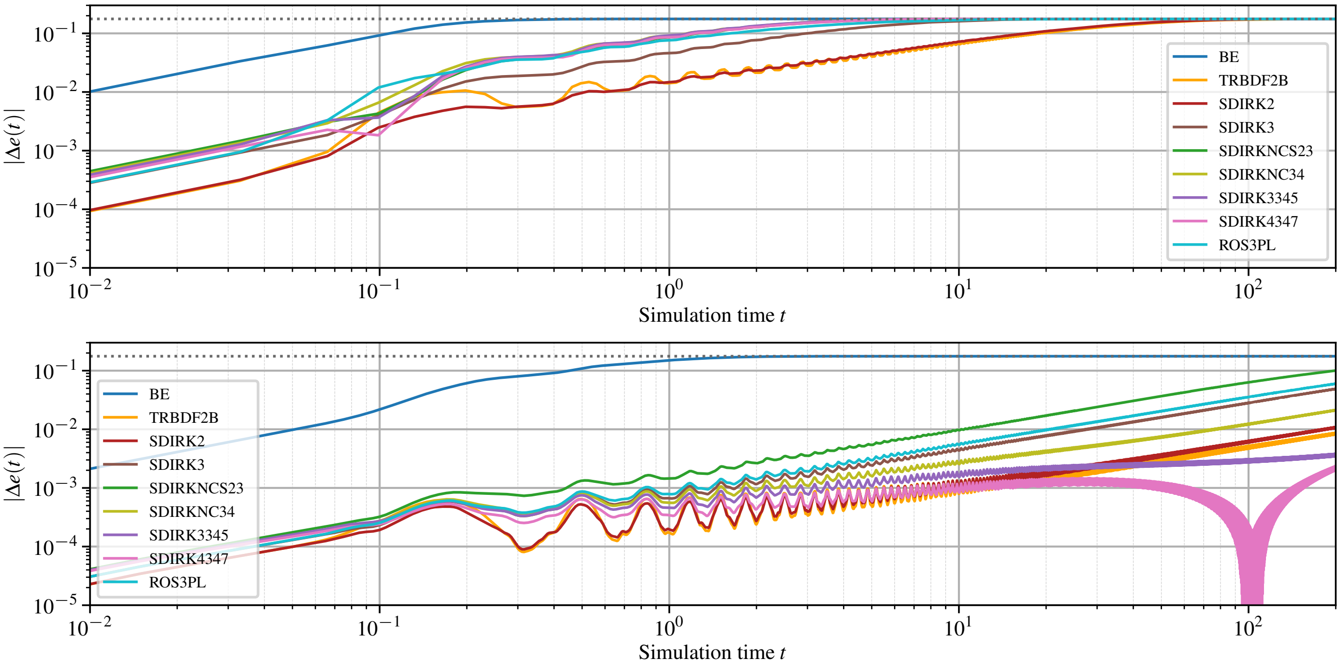


**Figure 3:** *Change in total specific energy over time for the double-clamped beam experiment (lower is better) with $\Delta t = 33\,\text{ms}$ (top) and $\Delta t = 5\,\text{ms}$ (bottom). The dotted line indicates the energy loss when BE reaches a steady state solution due to numerical dissipation.*

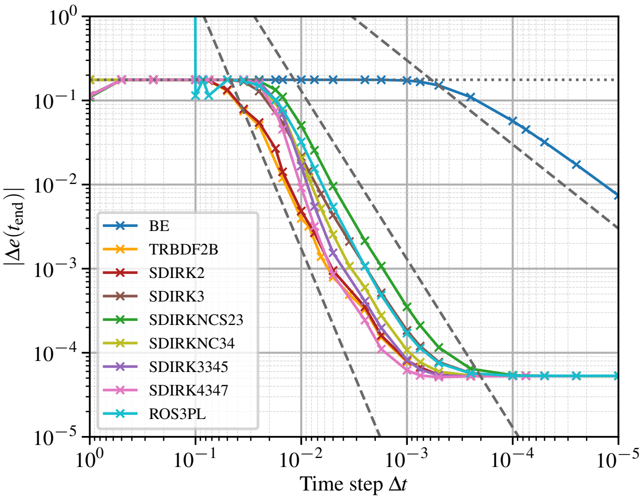
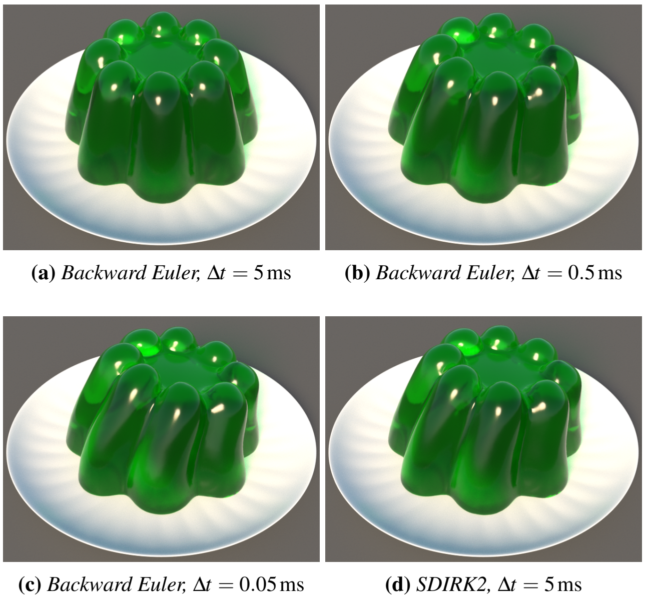


**Figure 4:** *Change in total specific energy (lower is better) at the end of the double-clamped beam experiment for different step sizes. The dotted line indicates the energy loss when Backward Euler reaches its steady state solution. The dashed lines from left to right indicate slopes of ideal third-, second- and first-order convergence.*

beam. A rectangular beam ($\rho = 1000\,{}^{\text{kg}}/_{\text{m}^3}$, $E = 1 \times 10^5\,\text{Pa}$, $\nu = 0.4$) is fixed with homogenous boundary conditions on both ends and is subjected to gravity as shown in Figure 2. Without damping (numerical or manually specified), the beam should oscillate indefinitely. To measure the error from this ideal condition at time $t$, we compute the total specific energy of the beam $e_{\text{tot}}(t)$, which consists of kinetic, potential and strain energy per unit mass. Due to numerical damping or stability issues, $e_{\text{tot}}(t)$ might deviate from the initial total energy $e_{\text{tot}}(t^0)$. Therefore, we define the absolute energy error $|\Delta e(t)| = |e_{\text{tot}}(t) - e_{\text{tot}}(t^0)|$ and record this for every time step.

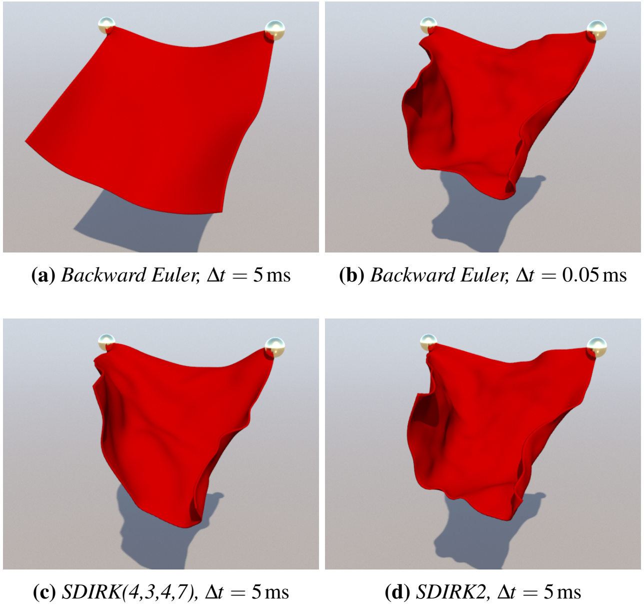
Figure 3 shows the energy error for two simulations over a span of 200 s with step sizes of $\Delta t = 33\,\text{ms}$ and $\Delta t = 5\,\text{ms}$. It is evident that the energy loss for Backward Euler is nearly two orders of magnitude larger than for all other methods. Furthermore, we can see that the SDIRK2 and TR-BDF2(B) methods clearly achieve the smallest error during the entire simulation for a time step of $\Delta t = 33\,\text{ms}$. TR-BDF2(B), however, exhibits stronger oscillations than SDIRK2 in the beginning and seems to conserve the energy slightly better over very long simulation durations. Still, visual inspection of the simulation results does not reveal significant differences between the two methods except for a slight phase shift towards the end. With a time step of $\Delta t = 5\,\text{ms}$, only the three-stage method SDIRK(3,3,4,5) and the four-stage method SDIRK(4,3,4,7) achieve a smaller error over longer periods of time. However, at around 100 s simulation time, the SDIRK(4,3,4,7) method started to gain energy again. This behavior can probably be attributed to the small scale oscillations of the energy that are observed for all higher-order methods. Usually, these tiny oscillations are not noticeable, but for a method like SDIRK(4,3,4,7), which is optimized to have low dissipation, numerical errors can accumulate from these oscillations and eventually lead to an increase in energy. Interestingly, none of the other methods with an order of three or higher performed better than SDIRK2 and TR-BDF2(B). In general, it is not surprising that this can happen, as the error constant of a method which affects the solution error at a specific step size, is



**(a)** *Backward Euler,* $\Delta t = 5\,\mathrm{ms}$ **(b)** *Backward Euler,* $\Delta t = 0.5\,\mathrm{ms}$

**(c)** *Backward Euler,* $\Delta t = 0.05\,\mathrm{ms}$ **(d)** *SDIRK2,* $\Delta t = 5\,\mathrm{ms}$

**Figure 5:** *A plate of jello, wobbling after the plate was nudged at the beginning of the simulation.*



**(a)** *Backward Euler,* $\Delta t = 5\,\mathrm{ms}$ **(b)** *Backward Euler,* $\Delta t = 0.05\,\mathrm{ms}$

**(c)** *SDIRK(4,3,4,7),* $\Delta t = 5\,\mathrm{ms}$ **(d)** *SDIRK2,* $\Delta t = 5\,\mathrm{ms}$

**Figure 6:** *Comparison between different integration methods with different time step sizes for a cloth-like solid consisting of* $96 \times 96 \times 1$ *trilinear hex elements.*

not directly related to the order of a method. The ROS3PL method also performed quite well for a linearized method on a nonlinear material.

To be able to see the effect of the order of a method, it is necessary to compare the error for different step sizes. Therefore, we run the same scene for 10 s with various step sizes $\Delta t \in [1 \times 10^5\,\mathrm{s}, 1\,\mathrm{s}]$ and evaluate the energy error at the end of the simulation. Figure 4 shows this error against the respective step size of the simulation. The first-order convergence of the Backward Euler method is clearly visible and again, all other methods show a significantly smaller error for the same step size over the whole range. In accordance with the previous experiment, both the SDIRK2 method and the TR-BDF2(B) method perform significantly better than all other methods in the range between $\Delta t = 1 \times 10^{-1}\,\mathrm{s}$ and $1 \times 10^{-2}\,\mathrm{s}$. Only starting from a step size at around $5 \times 10^{-2}\,\mathrm{s}$ they are marginally outperformed by the SDIRK(4,3,4,7) method. SDIRK(4,3,4,7) and SDIRK(3,3,4,5) are the only methods that show an order of three or higher, although this is of little value for most animation applications due to their higher computational cost and a larger error until the aforementioned small step size. All other higher-order methods show an order of convergence between two and three, even the methods with a classical order of three or four. As a last observation, the energy error converges for all higher-order methods to a limit towards a step size of $\Delta t = 1 \times 10^{-4}\,\mathrm{s}$. This is related to the fact that the accuracy of the discrete energy also depends on the resolution of the spatial discretization. A finer spatial discretization would further decrease this limit.

**Wobbling jello** For a more practical application in computer animation, we want to simulate jello (see Figure 5). The jello (corotated material, $E = 5.5 \times 10^5\,\mathrm{Pa}$, $\nu = 0.38$) is positioned on a plate which is nudged in the beginning of the simulation. Without any damping we expect the jello to wobble without a reduction in amplitude over time. With a time step of 5 ms for Backward Euler, after only 5 s the movement is already very slow and after 10 s it is visually indistinguishable from a steady state solution. At the same time the solution produced by the SDIRK2 method still wobbles vividly after 10 s with hardly any visible energy loss. Reducing the time step for Backward Euler to 0.5 ms still results in strong damping. Only for a time step of 0.05 ms we get visually similar behavior as with SDIRK2 at $\Delta t = 5\,\mathrm{ms}$. However, as shown in Table 1, computing the solution using SDIRK2 at $\Delta t = 5\,\mathrm{ms}$ was more than 55 times faster than using BE with a time step of 0.05 ms. Note that the simulation using TR-BDF2(B) was around 15% slower than with SDIRK2 at the same step size. This can be attributed purely to an implementation detail, where we recompute the forces of the previous time step that are required by the method due to its explicit first stage. Caching them would result in similar performance of the two methods at the cost of additional memory consumption for TR-BDF2(B).

**Thick cloth blowing in the wind** Figure 6 and the supplemental video show an undamped piece of thick cloth blowing in the wind ($\rho = 1000\,\mathrm{kg}/\mathrm{m}^3$, $E = 3 \times 10^5\,\mathrm{Pa}$, $\nu = 0.4$). The cloth is modeled by a single layer of $96 \times 96$ trilinear hex elements. Due to the numerical dissipation of Backward Euler with $\Delta t = 5\,\mathrm{ms}$, almost all details — such as wrinkles and folds — are lost. In comparison, the higher-order integrators SDIRK2 and SDIRK(4,3,4,7) show much more detail at the same step size. SDIRK2 in particular looks visually comparable to Backward Euler at $\Delta t = 0.05\,\mathrm{ms}$ while being almost 17 times faster (Table 1). SDIRK2 and TR-BDF2(B) require between 2.6 to 3.5 times as many Newton iterations as Backward Euler with the same time step due to their higher frequency de-

© 2020 The Author(s)

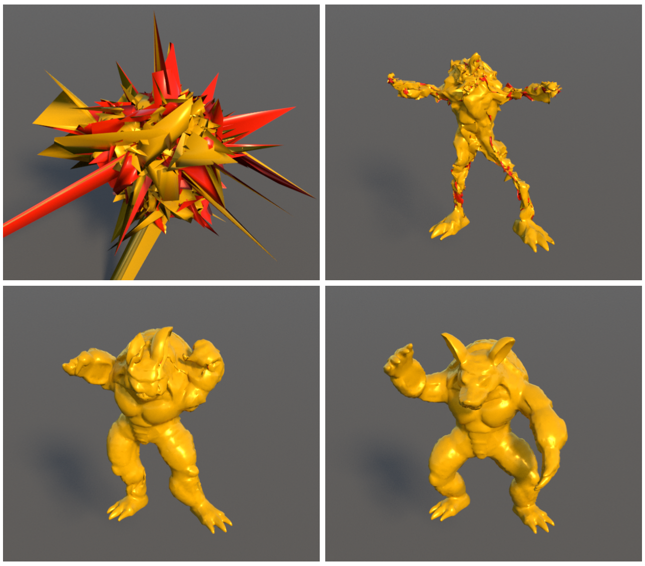


**Figure 7:** *The armadillo recovers from an extreme deformation with inverted and degenerate elements.*

formations. In this experiment TR-BDF2(B) required slightly less Newton iterations than SDIRK2 on average, however, we did not observe this as a consistent trend in other experiments. In conclusion, relative to the results of the second-order integrators, the increased computational cost of the other methods does not seem to be justified in this experiment.

In the supplemental video, the same scene is repeated with the SDIRK2 integrator and Rayleigh damping ($\beta = 2 \times 10^{-4}$ s) compared against the Backward Euler integrator with no damping. The small amount of damping cancels out much of the high-frequent oscillation present in the damping-free scene, yet retains much of the detail in the cloth. We conclude that the inherent lower dissipation associated with the SDIRK2 integrator enables more realistic and user-controllable damping behavior.

**Extreme deformation** We study the ability of an armadillo ($\rho = 1000\,{}^{\mathrm{kg}}/_{\mathrm{m}^3}$, $E = 1 \times 10^6$ Pa, $\nu = 0.4$) to recover from an initial state with randomly generated displacements that are far more severe than those found in typical applications, and can thus be considered a form of worst-case scenario (see Figure 7). Here we observed that recovery for strongly nonlinear materials proved difficult, as the root-finding procedure (Newton's method) with line search based on the residual norm would get stuck in non-global critical points. We were only able to solve the Newton problem for Backward Euler with reduced time steps, whereas we were not able to make progress with Newton iterations for the higher-order integrators for reasonable step sizes.

Figure 7 and the supplemental video depict the progression of the scene and comparison of integrators for a corotational linear elasticity. Between Backward Euler, ROS3PL, SDIRK4347 and SDIRK2, only Backward Euler was found to be long-term stable without material damping. This is primarily a problem due to the employed corotational formulation itself, since keeping the rotations fixed means that the model is not rotation-invariant. Therefore, rapidly changing rotations lead to increasing energy in the system. Only a small amount of material Rayleigh damping ($\beta = 10^{-3}$ s) was sufficient to stabilize all the considered higher-order integrators. Whereas Backward Euler quickly damps out almost all of the stored potential energy, the higher-order integrators preserve much more of the energy, resulting in livelier motion after recovery.

**Summary** Amongst the higher-oder methods, two methods gave outstanding results in comparison to the others: the second-order SDIRK2 and TR-BDF2(B) methods. In our numerical experiment with the double-clamped cantilever, only SDIRK(4,3,4,7) and SDIRK(3,3,4,5) performed better over very long periods of time. Convergence orders higher than two were only achieved by the corresponding methods at step sizes that are too small for practical use. In addition, in the qualitative experiments, visual inspection shows that none of these methods produces significantly better results than the second-order methods. With this in mind, the higher computational cost of the four- and three-stage methods does not seem justifiable in this area of application. In contrast, the SDIRK2 and TR-BDF2(B) methods already yield much higher quality results than Backward Euler, while offering more control over material damping at a comparatatively moderate increase in computationl cost. With the present results, however, we were not able to pick a clear winner between SDIRK2 and TR-BDF2(B) which produced visually very similar results. For the following experiments, we chose to only showcase the results obtained with SDIRK2, as we prefer the absence of an explicit first stage. In the future we would like to investigate the stability of SDIRK2 and TR-BDF2 more, as our experiment with the deformed armadillo was not conclusive, due the indicated shortcomings related to the nonlinear solver.

### 5.4. Evaluation of the contact model

The correction step of our contact model is based on a linerization of the internal forces. In order to examine how this first-order approximation influences the higher-order integrators, we perform a simple experiment with a bouncing ball. The ball ($\rho = 1000\,{}^{\mathrm{kg}}/_{\mathrm{m}^3}$, $E = 6 \times 10^5$ Pa, $\nu = 0.48$) with a diameter of 1 m is dropped from a height of 3 m under the influence of gravity. We do not manually apply damping. Note that the maximum jump height of the ball might not be constant as energy is transformed into oscillatory modes. The total energy, however, should ideally be conserved and can therefore be used as a metric. We simulate the ball over a span of 10 s with Backward Euler and SDIRK2 and time step sizes of $\Delta t = 5$ ms and $\Delta t = 0.5$ ms. The total specific energy of the ball is visualized in Figure 8.

For both step sizes, SDIRK2 dissipates visibly less energy per collision, as well as between each collisions, than Backward Euler. It is also evident that a reduction of the timestep significantly reduces the dissipation during the contact phase. Still, the advantage of the second-order method is less prounounced than in previous experiments. In future work, we would like to improve our contact model in this respect. There are several possible approaches for this, e.g. improving the design of the corrector itself, performing iterative prediction and correction with relaxation or combining the model with an energy tracking based approach (such as [DLL*18]).

© 2020 The Author(s)

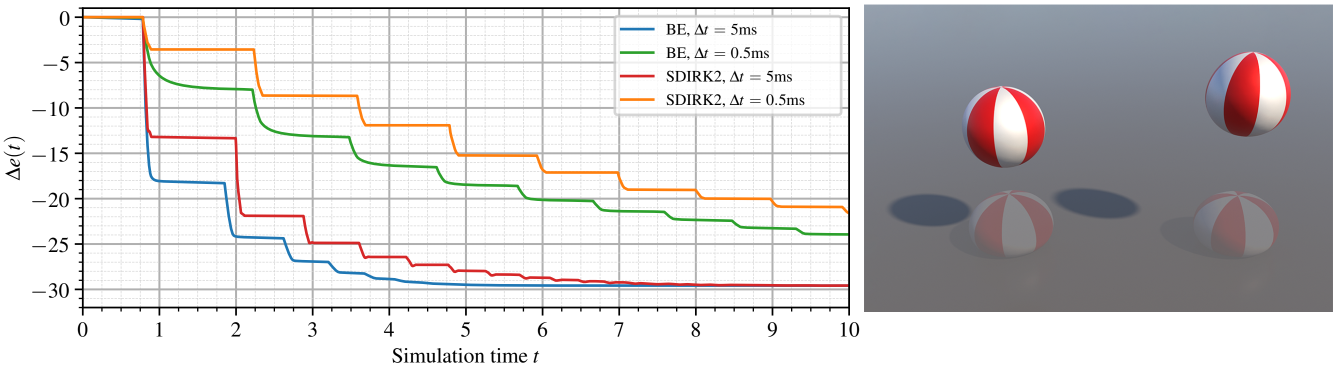


**Figure 8:** *Jumping ball experiment. Left: Loss of the ball's total specific energy over time. Each collision with the floor dissipates energy. Furthermore, Backward Euler dissipates significant amounts of energy even when the ball is not in contact with the floor. Right: The ball simulated with Backward Euler does not jump as high as with SDIRK2. The transparent balls show the collision with the ground.*

Nevertheless, in practice, the influence of the contact model might not be as severe as in this example, e.g. considering scenes where energy is transferred by collisions to a main object of interest, as shown in the following section.

### 5.5. Complex scenes

The following scenes demonstrate the practical interaction between higher-order implicit time integration methods and our contact model by showing collisions of multiple deformable bodies. For the previously stated reasons, we only present results obtained using the SDIRK2 method. Due to the dissipation introduced by our contact model (see Section 5.4), we did not explicitly add Rayleigh damping in the correction process (i.e. $D = 0$), but only as part of the predictor.

**Suspended Armadillo** The scene, shown in Figure 1 (right) and the supplemental video, shows multiple beach balls thrown in sequence colliding with a suspended armadillo ($\rho = 500\,\mathrm{kg/m^3}$, $E = 4 \times 10^4\,\mathrm{Pa}$, $\nu = 0.48$). We are able to qualitatively observe the transfer of kinetic energy from the flying beach balls to the armadillo as well as excellent energy conservation for the armadillo itself after the collisions. Because of the much smaller amount of numerical energy dissipation, we added explicit stiffness damping ($\beta = 4 \times 10^{-4}\,\mathrm{s}$) to the material of the armadillo in order to obtain a visually more pleasing result.

**Duck Trampoline Party** This scene, shown in Figure 1 (left) and the supplemental video, consists of 30 rubber ducks ($\rho = 250\,\mathrm{kg/m^3}$, $E = 5 \times 10^4\,\mathrm{Pa}$, $\nu = 0.4$) which are emitted into a maze of 10 thin, cloth-like trampolines ($\rho = 1000\,\mathrm{kg/m^3}$, $E = 1 \times 10^6\,\mathrm{Pa}$, $\nu = 0.4$), being held in position with homogenous boundary conditions at the corner vertices. While falling through the maze, the ducks slide and bounce across the trampolines, repeatedly colliding with trampolines and other ducks. This scene demonstrates that our constraint-based contact handling is able to robustly handle large amounts of contacts in conjunction with an implicit higher-order time integration scheme. In addition, the long lasting motion of the trampolines again demonstrates the improved energy conservation properties of the SDIRK2 method. Although the scene was stable without material damping, we again use Rayleigh damping for the ducks ($\alpha = 5 \times 10^{-3}\,\mathrm{1/s}, \beta = 5 \times 10^{-3}\,\mathrm{s}$) and cloth ($\alpha = 5 \times 10^{-4}\,\mathrm{1/s}, \beta = 5 \times 10^{-4}\,\mathrm{s}$) to obtain a visually more pleasing motion of the bodies.

## 6. Conclusion and Future Work

We have investigated a number of implicit Runge-Kutta-type methods for computer graphics applications. The results suggest that second-order DIRK methods — in particular SDIRK2 and TR-BDF2(B) — offer an attractive alternative to Backward Euler for many graphics applications. Comparing SDIRK2 and TR-BDF2(B), we would tend to prefer SDIRK2 as it does not feature an explicit stage and might therefore be stable for a wider range of problems than considered in this work. In contrast to the aforementioned second-order methods, we would not recommend the usage of the third- and fourth-order methods that we tested. They did not deliver a similar gain in accuracy and quality for practical ranges of step sizes and simulation durations, especially considering their higher computational cost.

Compared to Backward Euler, the substantially increased accuracy and reduced numerical dissipation allow much more faithful

| Scene | Integrator | $\Delta t$ (ms) | real s/sim s | Newt. iter. |
|---|---|---|---|---|
| Jello | BE | 5 | 21.3 | 1.00 |
| (corotated) | BE | 0.5 | 191.4 | 1.00 |
| | BE | 0.05 | 1907.3 | 1.00 |
| | SDIRK2 | 5 | 34.42 | 2.00 |
| | TR-BDF2(B) | 5 | 40.8 | 2.00 |
| Cloth | BE | 5 | 37.92 | 2.51 |
| (without | BE | 0.05 | 1530.42 | 1.01 |
| damping) | SDIRK2 | 5 | 100.32 | 6.90 |
| | TR-BDF2(B) | 5 | 102.46 | 6.59 |
| | SDIRK(4,3,4,7) | 5 | 533.58 | 20.78 |
| Susp. armadillo | SDIRK2 | 5 | 242.4 | 5.48 |

**Table 1:** *Integration method, step size, computation time ("real s") per simulated* 1 s*, and average Newton iterations for simulations from the supplemental video, obtained on a system with an Intel i9 9900K processor.*

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

reproduction of material behavior. When the underlying Newton solver is driven to convergence, the implicit integrators investigated offer excellent stability, and we believe therefore that they are especially well-suited for high-fidelity simulation. Due to the ability to use larger time steps for the same visual results, SDIRK2 is often substantially faster than Backward Euler, and therefore satisfies all criteria set out in the introduction except for *Reliability* in the most demanding scenarios (in particular our extreme deformation test case). However, the problems we observed in this experiment are closely related to our present design of the nonlinear solver based on the residual norm.

When formulated as a root-finding problem, the nonlinear problems that must be solved for the integration methods may get stuck in critical points that are not a solution to the nonlinear equations in the most challenging scenarios. The issue at heart is essentially the problem of finding a *global* minimum of the residual norm. Sometimes this problem can be alleviated by performing sub-stepping, i.e. when the Newton iterations do not converge, the time step is repeated with a subdivision into smaller time steps. However, this does not guarantee convergence and changes the discrete dynamics of the system. In contrast, optimization-based methods are attractive because any local minimum is a solution to the nonlinear problem. We therefore believe that it would be worthwhile to reformulate, e.g., SDIRK2 as an optimization problem, which combined with an effective solver should resolve the main problem of reliably solving the nonlinear problem, at which point all five quality criteria that we set out in the introduction would be fulfilled.

While our constraint-based contact model is well-suited for use with any higher-order integrator due to its carefully designed predictor-corrector structure, it also has limitations. Its stability is in large part due to the linearization used, which is similar to Backward Euler. As a result, it generally dissipates energy near the contact surface. Moreover, our contact model currently does not include a friction model, and it makes no attempt to recover from penetrations. While this works surprisingly well due to the tendency of colliding deformable bodies to push each other apart due to internal forces, it would be a natural extension to include position stabilization and a compatible friction model.

## Acknowledgements

We thank Dominik Michels for his valuable feedback and discussions. This work is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) — BE 5132/3-2.

## References

[Ale77] Alexander R.: Diagonally Implicit Runge–Kutta Methods for Stiff O.D.E.'s. *SIAM Journal on Numerical Analysis 14*, 6 (1977), 1006–1021. 6

[Bar12] Barbič J.: *Exact Corotational Linear FEM Stiffness Matrix*. Tech. rep., University of Southern California, 2012. 7

[Bat07] Bathe K.-J.: Conserving energy and momentum in nonlinear dynamics: A simple implicit time integration scheme. *Computers & Structures 85*, 7-8 (2007), 437–445. 7

[BC00] Butcher J., Chen D.: A new type of singly-implicit Runge–Kutta method. *Applied Numerical Mathematics 34*, 2-3 (jul 2000), 179–188. 7

[BCDA11] Bertails-Descoubes F., Cadoux F., Daviet G., Acary V.: A nonsmooth Newton solver for capturing exact Coulomb friction in fiber assemblies. *ACM Transactions on Graphics 30*, 1 (2011), 1–14. 5

[BCF*85] Bank R., Coughran W., Fichtner W., Grosse E., Rose D., Smith R.: Transient Simulation of Silicon Devices and Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 4*, 4 (1985), 436–451. 7

[BDR17] Bonaventura L., Della Rocca A.: Unconditionally strong stability preserving extensions of the TR-BDF2 method. *Journal of Scientific Computing 70*, 2 (2017), 859–895. 7

[BMF03] Bridson R., Marino S., Fedkiw R.: Simulation of Clothing with Folds and Wrinkles. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2003), Eurographics Association, pp. 28–36. 2

[BML*14] Bouaziz S., Martin S., Liu T., Kavan L., Pauly M.: Projective Dynamics: Fusing Constraint Projections for Fast Simulation. *ACM Transactions on Graphics 33*, 4 (2014), 1–11. 3

[BMM17] Bender J., Müller M., Macklin M.: A Survey on Position Based Dynamics, 2017. In *EUROGRAPHICS 2017 Tutorials* (2017), Eurographics Association. 2

[BW98] Baraff D., Witkin A.: Large Steps in Cloth Simulation. In *ACM Conference on Computer Graphics and Interactive Techniques* (1998), ACM, pp. 43–54. 2

[C*19] Crozet S., et al.: nalgebra: a linear algebra library for Rust, 2019. URL: https://nalgebra.org. 7

[CAP17] Chen Y. J., Ascher U., Pai D.: Exponential Rosenbrock-Euler Integrators for Elastodynamic Simulation. *IEEE Transactions on Visualization and Computer Graphics* (2017). 2

[CK02] Choi K.-J., Ko H.-S.: Stable but Responsive Cloth. *ACM Transactions on Graphics 21*, 3 (July 2002), 604–611. 2

[CLW09] Clemens M., Lang J., Wimmer G.: Adaptivity in Space and Time for Magnetoquasistatics. *Journal of Computational Mathematics 27*, 5 (2009), 642–656. 7

[DLK18] Dinev D., Liu T., Kavan L.: Stabilizing Integrators for Real-Time Physics. *ACM Transactions on Graphics 37*, 1 (2018), 9. 2

[DLL*18] Dinev D., Liu T., Li J., Thomaszewski B., Kavan L.: FEPR: Fast Energy Projection for Real-Time Simulation of Deformable Objects. *ACM Transactions on Graphics 37*, 4 (2018). 2, 10

[EWS96] Eberhardt B., Weber A., Strasser W.: A Fast, Flexible, Particle-System Model for Cloth Draping. *IEEE Computer Graphics and Applications 16*, 5 (1996), 52–59. 2

[FGR97] Franco J., Gómez I., Rández L.: SDIRK methods for stiff ODEs with oscillating solutions. *Journal of Computational and Applied Mathematics 81*, 2 (1997), 197 – 209. 7

[GHDS03] Grinspun E., Hirani A. N., Desbrun M., Schröder P.: Discrete Shells. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2003), Eurographics Association, pp. 62–67. 2

[GS20] Giri S., Sen S.: A new class of diagonally implicit Runge–Kutta methods with zero dissipation and minimized dispersion error. *Journal of Computational and Applied Mathematics 376* (oct 2020), 112841. 7

[GSS*15] Gast T., Schroeder C., Stomakhin A., Jiang C., Teran J.: Optimization Integrator for Large Time Steps. *IEEE Transactions on Visualization and Computer Graphics 21*, 10 (2015), 1103–1115. 3

[HE01] Hauth M., Etzmuss O.: A High Performance Solver for the Animation of Deformable Objects using Advanced Numerical Methods. *Computer Graphics Forum 20*, 3 (2001), 319–328. 2, 3

[HEE*02] Hauth M., Etzmuss O., Eberhardt B., Klein R., Sarlette R., Sattler M., Daubert K., Kautz J.: Cloth Animation and Rendering. In *Eurographics 2002 - Tutorials* (2002), Eurographics Association. 2

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

[HES03] Hauth M., Etzmuss O., Strasser W.: Analysis of numerical methods for the simulation of deformable models. *The Visual Computer 19*, 7-8 (2003), 581–600. 2, 3

[HLW06] Hairer E., Lubich C., Wanner G.: *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*, 2nd ed. Springer Berlin Heidelberg, 2006. 2

[HW92] Hairer E., Wanner G.: *Solving Ordinary Differential Equations I: Nonstiff Problems*, 3rd ed. Springer Berlin Heidelberg, 1992. 2, 4

[HW96] Hairer E., Wanner G.: *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, 2nd ed. Springer Berlin Heidelberg, 1996. 2, 3, 4, 5

[KBT17] Koschier D., Bender J., Thuerey N.: Robust eXtended Finite Elements for Complex Cutting of Deformables. *ACM Transactions on Graphics 36*, 4 (2017), 55:1–55:13. 2

[KC16] Kennedy C., Carpenter M.: *Diagonally Implicit Runge-Kutta Methods for Ordinary Differential Equations. A Review*. Tech. Rep. TM-2016-219173, NASA, 2016. 4, 6, 7

[KKB18] Kugelstadt T., Koschier D., Bender J.: Fast Corotated FEM using Operator Splitting. In *Computer Graphics Forum* (2018), vol. 37. 2, 3

[KSJP08] Kaufman D. M., Sueda S., James D. L., Pai D. K.: Staggered Projections for Frictional Contact in Multibody Systems. In *ACM SIGGRAPH Asia 2008 Papers* (New York, NY, USA, 2008), ACM. 5

[KWT*06] Kharevych L., Weiwei, Tong Y., Kanso E., Marsden J. E., Schröder P., Desbrun M.: Geometric, Variational Integrators for Computer Animation. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2006), Eurographics Association, pp. 43–51. 2

[LBBK13] Liu T., Bargteil A. W., Brien J. F. O., Kavan L.: Fast Simulation of Mass-Spring Systems. *ACM Transactions on Graphics 32*, 6 (2013), 214:1–214:7. 3

[LBK17] Liu T., Bouaziz S., Kavan L.: Quasi-Newton Methods for Real-Time Simulation of Hyperelastic Materials. *ACM Transactions on Graphics 36*, 3 (May 2017), 23:1–23:16. 3

[LDN*18] Li J., Daviet G., Narain R., Bertails-Descoubes F., Overby M., Brown G. E., Boissieux L.: An implicit frictional contact solver for adaptive cloth simulation. *ACM Transactions on Graphics 37*, 4 (2018), 1–15. 5

[LFS*20] Li M., Ferguson Z., Schneider T., Langlois T., Zorin D., Panozzo D., Jiang C., Kaufman D. M.: Incremental Potential Contact: Intersection- and Inversion-free Large Deformation Dynamics. *ACM Transactions on Graphics 39*, 4 (2020). 5

[LGL*19] Li M., Gao M., Langlois T., Jiang C., Kaufman D. M.: Decomposed Optimization Time Integrator for Large-Step Elastodynamics. *ACM Transactions on Graphics 38*, 4 (2019). 3

[LMOW04] Lew A., Marsden J. E., Ortiz M., West M.: Variational time integrators. *International Journal for Numerical Methods in Engineering 60*, 1 (2004), 153–212. 2

[MG04] Müller M., Gross M.: Interactive Virtual Materials. In *Proceedings of Graphics Interface* (2004), p. 239–246. 2, 7

[MLT17] Michels D., Luan V. T., Tokman M.: A Stiffly Accurate Integrator for Elastodynamic Problems. *ACM Transactions on Graphics 36*, 4 (2017). 2

[MMC16] Macklin M., Müller M., Chentanez N.: XPBD: Position-based Simulation of Compliant Constrained Dynamics. In *ACM Motion in Games* (2016), ACM, pp. 49–54. 3

[MSW14] Michels D. L., Sobottka G. A., Weber A. G.: Exponential Integrators for Stiff Elastodynamic Problems. *ACM Transactions on Graphics 33*, 1 (Feb. 2014), 7:1–7:20. 2

[MTGG11] Martin S., Thomaszewski B., Grinspun E., Gross M.: Example-based elastic materials. *ACM Transactions on Graphics 30*, 4 (July 2011), 1. 2, 3

[NW06] Nocedal J., Wright S.: *Numerical optimization*. Springer Science & Business Media, 2006. 6

[OBLN17] Overby M., Brown G. E., Li J., Narain R.: ADMM⊇Projective Dynamics: Fast Simulation of Hyperelastic Models with Dynamic Constraints. *IEEE Transactions on Visualization and Computer Graphics 23*, 10 (Oct. 2017), 2222–2234. 3

[OTSG09] Otaduy M. A., Tamstorf R., Steinemann D., Gross M.: Implicit contact handling for deformable objects. In *Computer Graphics Forum* (2009), vol. 28, Wiley Online Library, pp. 559–568. 5

[Ran15] Rang J.: Improved traditional Rosenbrock–Wanner methods for stiff ODEs and DAEs. *Journal of Computational and Applied Mathematics 286* (2015), 128–144. 7

[RLK18] Rojas J., Liu T., Kavan L.: Average Vector Field Integration for St. Venant-Kirchhoff Deformable Models. *IEEE Transactions on Visualization and Computer Graphics* (2018). 2

[SBG*20] Stellato B., Banjac G., Goulart P., Bemporad A., Boyd S.: OSQP: An operator splitting solver for quadratic programs. *Mathematical Programming Computation* (2020), 1–36. 6

[SD08] Stern A., Desbrun M.: Chapter 12: Discrete Geometric Mechanics for Variational Time Integrators. In *Discrete Differential Geometry: An Applied Introduction*. ACM Press, 2008, pp. 95–100. 2

[SDGK18] Smith B., De Goes F., Kim T.: Stable Neo-Hookean Flesh Simulation. *ACM Transactions on Graphics 37*, 2 (July 2018), 1–15. 7

[SG09] Stern A., Grinspun E.: Implicit-Explicit Variational Integration of Highly Oscillatory Problems. *Multiscale Model. Simul. 7*, 4 (2009), 1779–1794. 2

[SKZF11] Schroeder C., Kwatra N., Zheng W., Fedkiw R.: Asynchronous Evolution for Fully-Implicit and Semi-Implicit Time Integration. *Computer Graphics Forum 30*, 7 (Sept. 2011), 1983–1992. 2

[SSF13] Su J., Sheth R., Fedkiw R.: Energy Conservation for the Simulation of Deformable Bodies. *IEEE Transactions on Visualization and Computer Graphics 19*, 2 (2013), 189–200. 2

[TNGF15] Tournier M., Nesme M., Gilles B., Faure F.: Stable Constrained Dynamics. *ACM Transactions on Graphics* (2015), 1–10. 3, 5

[TPBF87] Terzopoulos D., Platt J., Barr A., Fleischer K.: Elastically Deformable Models. In *ACM Conference on Computer Graphics and Interactive Techniques* (Aug. 1987), ACM, pp. 205–214. 2

[TPS08] Thomaszewski B., Pabst S., Strasser W.: Asynchronous Cloth Simulation. In *Computer Graphics International* (2008). 2

[TSIF05] Teran J., Sifakis E., Irving G., Fedkiw R.: Robust quasistatic finite elements and flesh simulation. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2005), pp. 181–190. 3

[VCM95] Volino P., Courchesne M., Magnenat Thalmann N.: Versatile and Efficient Techniques for Simulating Cloth and Other Deformable Objects. In *ACM Conference on Computer Graphics and Interactive Techniques* (1995), ACM, pp. 137–144. 2

[VJ19] Verschoor M., Jalba A. C.: Efficient and accurate collision response for elastically deformable models. *ACM Transactions on Graphics 38*, 2 (2019), 1–20. 5

[Wan15] Wang H.: A Chebyshev Semi-iterative Approach for Accelerating Projective and Position-based Dynamics. *ACM Transactions on Graphics 34*, 6 (2015), 246:1–246:9. 3

[XB17] Xu H., Barbič J.: Example-Based Damping Design. *ACM Transactions on Graphics 36*, 4 (2017). 2, 3, 7

[ZLB16] Zhao D., Li Y., Barbic J.: Asynchronous implicit backward Euler integration. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2016), Eurographics Association / ACM, pp. 1–9. 2