Computers & Graphics (2023)

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

ESBD: Exponential Strain-based Dynamics using XPBD Algorithm

Ozan Cetinaslan[a]

[a]*Instituto de Telecomunicações, Faculdade de Ciências da Universidade do Porto, Rua Campo Alegre 1055, 4169-007 Porto, Portugal*

ARTICLE INFO

Article history:
Received August 18, 2023

Keywords: Physics-based Simulation, Position-based Dynamics, Green's Strain Tensor, Deformable Models, Mesh Deformation

ABSTRACT

Strain-based Dynamics (SBD) [1] is a well-known method to simulate the deformable models by using Green's strain tensor within the Position-based Dynamics (PBD) algorithm. Similar to the other position-based constraints, SBD performs stable and robust. However, when we transform the base algorithm from PBD to Extended Position-based Dynamics (XPBD), SBD suffers from the cumbersome parameter adjustments and uncontrollable numerical dissipation, which generates highly stiff results. To overcome these limitations, we propose Exponential Strain-based Dynamics (ESBD) which reformulates the Green's strain tensor with an exponential treatment. Our proposed method is less sensitive to the increased numerical dissipation, so it provides vivid simulation of deformable objects with fewer parameter tunings. In order to present the advantages, we compare our method with SBD by using the XPBD method and other well-known techniques. We observe that ESBD improves the hyperelastic behavior of the Green's strain tensor as a projected constraint without altering the underlying XPBD algorithm. Therefore, it is possible to adapt ESBD to the other existing numerical integration and iteration methods. Our proposed method is stable, straightforward to follow and produces high quality dynamic simulations.

1. Introduction

Physics-based simulation of deformable objects is an important and longstanding research subject of computer graphics. Over the years, researchers have been working on different numerical integration techniques and iteration methods to improve the stability, performance and visual quality of the simulations. Common techniques include Finite Element Method (FEM) [2], Position-based Dynamics (PBD) [3], Projective Dynamics (PD) [4], Material Point Method (MPM) [5]. These methods utilize numerical integration methods (such as Euler, Verlet, Newmark, etc.) combined with iterative solvers (such as Jacobi, Gauss-Seidel, Newton-Raphson, etc.). Within those notable methods, PBD is a quite popular technique to simulate physics-based phenomena due to its stability, performance and simplicity. But most importantly, PBD is versatile since it formulates inner-resistance of the models as projected constraints. Therefore, PBD provides the simulation of many different concepts such as constitutive models, fluids, rigid-body simulations, etc.

Strain-based Dynamics (SBD) [1] is a PBD based technique to simulate the deformable objects by using Green's strain tensor. SBD intrinsically contains all the typical properties of PBD and performs quite stable. However, since SBD utilizes Green's strain tensor, principal stretches and shear forces are computed simultaneously for each mesh primitive. Therefore, simulating SBD is more costly than simulating a mass-spring constraint in terms of performance.

Extended Position-based Dynamics (XPBD) [6] improves the Gauss-Seidel solver of PBD with a total Lagrange multiplier and a compliance parameter which is associated with the material stiffness. Furthermore, XPBD method implicitly utilizes Rayleigh dissipation function within the Gauss-Seidel solver. Although this improvement provides a notable performance gain, the numerical damping of the system directly de-

e-mail: ozan.cetinaslan@gmail.com (Ozan Cetinaslan)

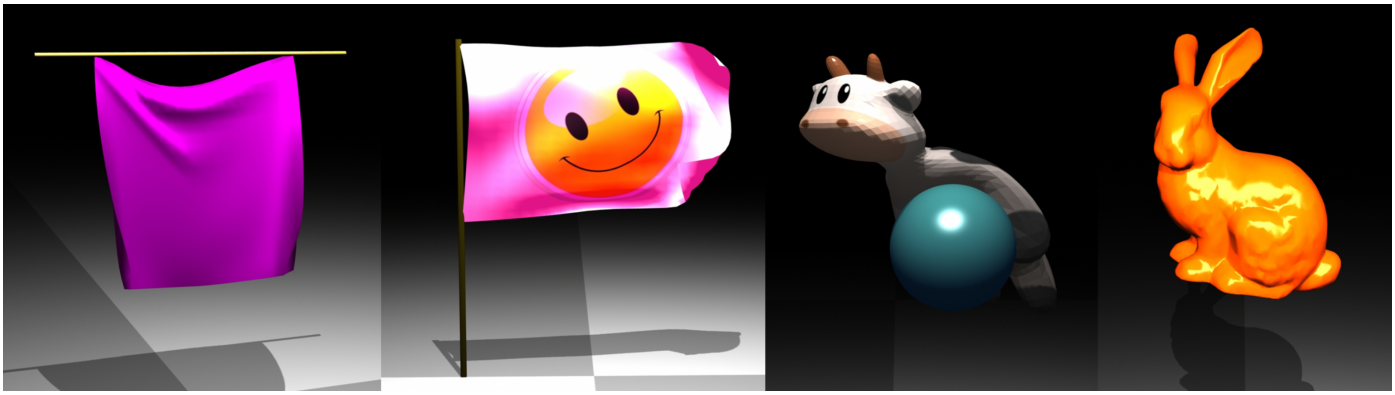


Fig. 1: Exponential Strain-based Dynamics (ESBD) is presented for the XPBD simulation of deformable objects. ESBD enhances the hyperelastic behavior of the Green's strain tensor and is successfully capable of simulating triangle and volumetric models under the constant forces, such as gravity and wind. Our method also performs well during the collision and contact capturing.

pends on the value of the compliance parameter regardless of the damping coefficient. For example, when the compliance parameter is tuned to higher values, the motion is damped unexpectedly. On the other hand, tuning the compliance to lower values reduces the desired damping effect which may lead to unintended results. Besides that, iteration count remains as a critical element for the stability and stiffness of the models which means that higher or lower iteration count has a direct impact on the stiffness of the models (there are also other material specific parameters for stiffness such as Young's Modulus, Poisson's Ratio, spring stiffness, etc.). One solution to this shortcoming can be changing the extended Gauss-Seidel solver of XPBD. However, tailing the Gauss-Seidel solver is a tedious task due to maintaining the generality and it may lead to losing the efficient dissipation of XPBD unintentionally or possible stability issues during extreme stretching and compression as in [7].

When we adapt the Green's strain tensor of SBD to the XPBD algorithm, we observe that the projected constraints of Green's strain tensor are heavily affected by the aforementioned dependencies. This is due to the fact that SBD requires the computation of many positional constraints in each iteration. Therefore, adjusting and tuning all the mentioned parameters become more challenging. For example, when we apply higher values to the compliance parameter and iteration count, the numerical dissipation completely dominates the simulation and the model behaves too stiff, similar to a rigid-body. As a result, obtaining vivid and dynamic simulation results can be a time-consuming and cumbersome process due to many parameter adjustments with trial and error tests. Recently, this shortcoming has been reported in Exponential Spring Potential Energy Functions (ESPEFs) method [8] for the mass-spring systems (*ESPEF refers to ESPEFs [8] in the rest of the paper*). ESPEF has proposed a generic exponential-based formulation to soften the extreme stiffness of the model with reduced parameter tunings and has the ability to operate with many different spring potential functions. Since ESPEF operates on the edges of the mesh, it is straightforward to apply ESPEF for the triangle mesh models.

In this paper, we propose the Exponential Strain-based Dynamics (ESBD) by reformulating the projected constraints of the Green's strain tensor on an exponential basis (Figure 1). We take advantage of the generic formulation of ESPEF and apply it on the principal stretch and shear constraints that we obtain from the Green's strain tensor. Unlike ESPEF, ESBD performs iterative computations on the face primitives (or elements) of the mesh model. Furthermore, ESPEF computes only one projected constraint at each iteration. On the other hand, ESBD computes three projected constraints for the triangle meshes (two principal stretches and one shear), six projected constraints for the volumetric meshes (three principal stretches and three shears) in each iteration. Therefore, ESBD is more demanding in terms of implementation and performs slightly slower than ESPEF.

When we compare ESBD with SBD, our method provides more vivid and dynamic visual results with a minimum artifact that is caused by higher compliance and iteration count. Besides that, ESBD generates more wrinkles and foldings than SBD with higher shear potentials and less dissipated motion. Moreover, ESBD provides more stretchy surface deformations and jiggling during the simulation of volumetric models with reduced parameter tuning compared to SBD. These advantages are due to the fact that ESBD implicitly converts the linear behavior of SBD to nonlinear elastic response with the exponential factor by retaining the same computational complexity that XPBD provides. The presented quantitative comparisons also support the visual results. When ESBD is compared with ESPEF, other constitutive models such as StVK and Neo-Hookean materials and Projective Dynamics, ESBD continues to provide more pleasing results under the same simulation conditions.

The major goal of ESBD is to come up with a simple and generic solution that enhances the hyperelastic behavior of SBD with less parameter adjustments and less dissipation while maintaining the base XPBD algorithm. Therefore, ESBD can be utilized as a novel position-based constraint alongside with the existing constraints. Our method is straightforward in terms of application since it is based on SBD and ESPEF formulations, it is practical and can be applicable to the existing frameworks.

Our major contributions are listed below:

- We introduce a novel exponential-based formulation of

Green's strain tensor for 2D/3D deformable models.

- We additionally provide the exponential version of the modified Green's strain tensor from [1].
- We adapt the Strain-based Dynamics method [1] to the XPBD algorithm [6].
- We validate our method against the XPBD version of SBD and other methods, such as ESPEF [8], XPBD version of [9] and local/global methods ([10] and [4]) by applying several different visual and quantitative comparisons.

2. Related Work

This section reviews the literature by focusing on Position-based Dynamics solvers and the simulation of deformable objects.

2.1. Position-based Dynamics Solvers

Position-based Dynamics (PBD) is a well-known method in academia and industry. This is due to its unconditional stability, performance and simplicity. The roots of PBD can be found in [11] and [12] as constraint-based dynamics. After, official PBD algorithm was introduced by Müller et al. [3] just after the Shape Matching method [13], which can be considered as the prior method of PBD. Nucleus [14] was also based on the same fundamentals of PBD. After, Macklin et al. [15] presented a high performance unified framework based on PBD with a Jacobian solver. More recently, Extended Position-based Dynamics (XPBD) [6] improved the Gauss-Seidel solver of PBD with a compliance parameter in order to address the iteration count dependency of the PBD method. On the other hand, Tournier et al. [16] applied the second-order derivatives to the projected constraints in order to avoid the possible instabilities of the simulations. Shortly after, Macklin et al. [17] proposed the Small Steps to XPBD method in order to improve the convergence and energy conservation of the XPBD method. Cetinaslan proposed an alteration to the Gauss-Seidel part of the XPBD algorithm based on energy differences in order to improve the stability of the solver in [18] and [7]. Primal/dual decent methods [19] is one of the latest improvement over position-based solvers which provides local and local/global solution to the physics-based simulations including rigid-body dynamics, deformable objects and collision handling. In this paper, we employ the original XPBD algorithm as our base solver for our simulations. The tutorial from Bender et al. [20] is an excellent documentation that only focus on PBD oriented algorithms and methods.

2.2. Position-based Deformable Objects

PBD is a very versatile method that successfully simulates many different physical phenomena such as deformable objects, rigid-bodies, cloth, flesh and fluids. But in the scope of this paper, we just focus on the related work of deformable objects. Due to the constraint-based nature of the PBD method, novel techniques and applications have been mostly presented as projected constraints. For example, Müller and Chentanez proposed wrinkle meshes and oriented particles respectively in [21] and [22] as projected constraints to simulate cloth and deformable objects. Diziol et al. [23] presented a Shape Matching based deformation technique for surface meshes with a novel volume conservation constraint. After, inextensible cloths [24] and inextensible hair/fur [25] methods were proposed to simulate strand-based deformable models. Müller et al. presented the Strain-based Dynamics [1] to simulate the elastic models by using Green's strain tensor as projected constraints and Bender et al. [9] proposed the simulation of constitutive models by using PBD algorithm. Furthermore, it is possible to see PBD to simulate elastic rods [26, 27, 28], local deformations to simulate the subspace of the model [29, 30] and local collisions and contacts [31]. Recently, ESPEF method [8] has been proposed by Cetinaslan to simulate deformable objects more vivid by reformulating existing spring potential energy functions on an exponential basis. Besides that, Macklin and Müller [32] has reformulated the stable Neo-hookean method of [33] and adapted to XPBD algorithm. Furthermore, there have been some advances on the performance side of the position-based simulations. Novel techniques have been proposed to speed up PBD simulations of deformable objects, such as Multigrid method [34], Chebyshev method [35], Graph-Coloring [36, 37], Operator Splitting method [38] and Red-Black Coloring [39].

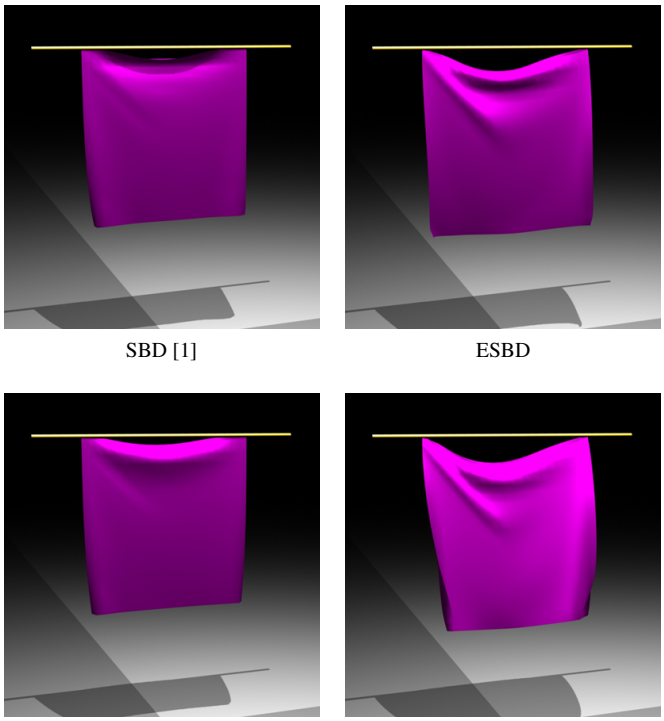
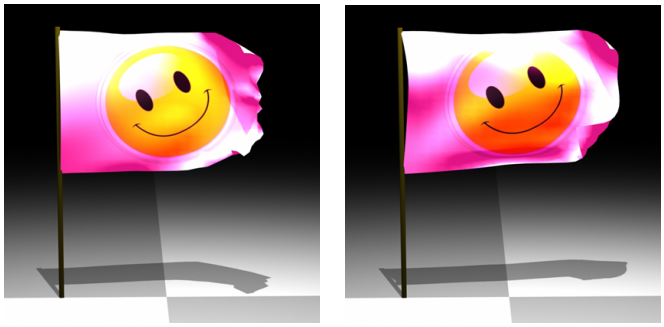


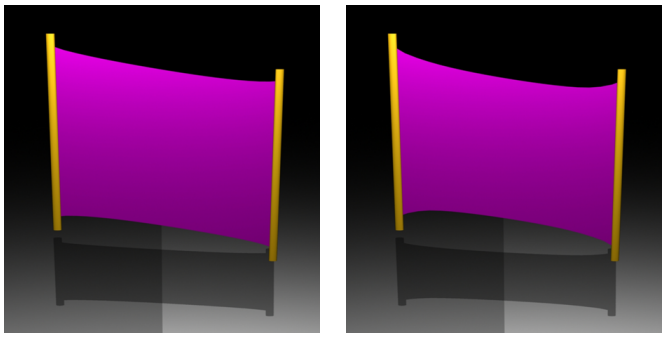
Fig. 2: A piece of cloth (with 800 triangles) falls and hangs under gravity. All simulations are executed under the same conditions, such as compliance parameter α = 0.00001, iteration count = 200, damping coef. = 0.3, step-size = 1/24. Since the compliance parameter and the iteration count have a direct impact to the damping of the XPBD method, the higher values affect SBD negatively. The motion is damped earlier than expected with less wrinkle generation. In ESBD, these artifacts are reduced.



SBD [1] ESBD

SBD Mod. [1] ESBD Mod.

Fig. 3: A flag (with 800 triangles) blows under the wind force. All simulations are executed under the same conditions, such as compliance parameter α = 0.00001, iteration count = 200, damping coef. = 0.3, step-size = 1/24. It is clear that our ESBD method blows the flag more dynamic while SBD resists to the constant wind due to unexpected numerical dissipation.



SBD [1] ESBD

SBD Mod. [1] ESBD Mod.

Fig. 4: A piece of cloth (with 800 triangles) is stretched from the fixed sides. All simulations are executed under the same conditions, such as compliance parameter α = 0.00001, iteration count = 200, damping coef. = 0.3, step-size = 1/24. In this simulation test case, area preservation constraint is also attached to SBD and ESBD. Although the simulation conditions are the same, the visual results are rather different. This difference is due to the stiff behaviour of SBD under the stretching test.

2.3. Deformable Object Simulations

There are other notable algorithms to simulate deformable objects effectively, such as Local/Global methods, FEM-based solvers and novel integration techniques. Fast Mass/Spring method [10], Projective Dynamics [4], Quasi-Newton method [40] and ADMM solver [41] are popular techniques based on the local/global solution to simulate deformable objects. Local/Global methods approach to the problem of simulating deformable objects as an energy minimization problem. If the geometric structure of the model remains consistent, the results of the local/global methods are fast and stable. Furthermore, FEM-based methods are known with their slow but stable behavior such as: Cloth simulation [42, 43] and FEM-based material design [44, 45, 46, 47, 33] are notable methods to simulate deformable objects with high quality visual results. Moreover, novel exponential integrators [48, 49, 50, 51] are utilized to simulate deformable objects with less numerical damping. However, these techniques offer a set of challenging algorithms based on exponential integration technique with Krylov method, so the results can be expensive in terms of performance. A recent tutorial by Kim and Eberle [52] provides the fundamental information on the deformable object simulations, deformation gradients and constitutive material models.

In our proposed ESBD method, we are knowledgeable about the prior works which have been mostly dominated by the Green's strain tensor. In contrast to the previous research, we propose a novel method that reformulates the Green's strain tensor on an exponential basis by taking advantage of the generic formulation of ESPEF method [8]. Different from the Strain-based Dynamics [1], we adapted the XPBD algorithm to our method instead of the PBD method. ESBD can handle the uncontrollable damping more smoothly with a vivid motion pattern. Since our solution is generic and does not require any alteration to the numerical integration method or the dissipation function of the physics solver, the novel formulation of ESBD can be easily utilized by the other physics-based solvers or existing simulation frameworks.

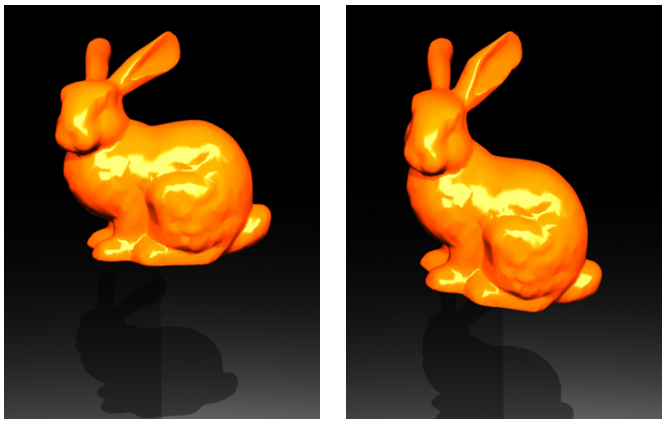
3. Background

In this section, we review two subjects that are the prerequisites of our method. First, we present a brief information about the XPBD algorithm [6]. Second, we explain the fundamentals of exponential-based potential energy formulation [8].

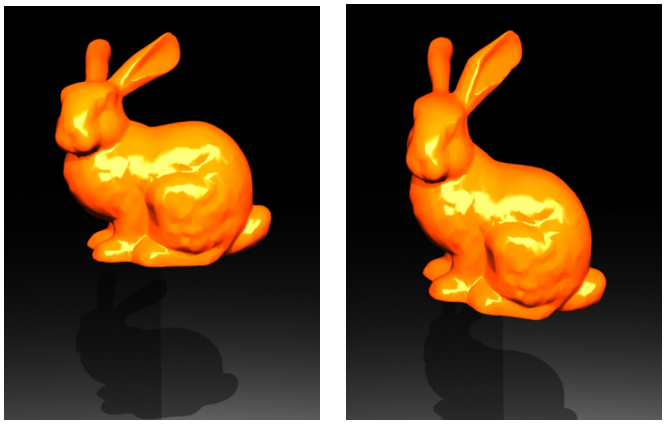
3.1. XPBD Algorithm

XPBD algorithm [6] is not only the extended version of the PBD method [3], but also it can be considered as the superior algorithm of PBD. This inherently means that we can obtain the PBD method by just altering the compliance parameter of XPBD.

The algorithm of XPBD works in three steps:



SBD [1] ESBD



SBD Mod. [1] ESBD Mod.

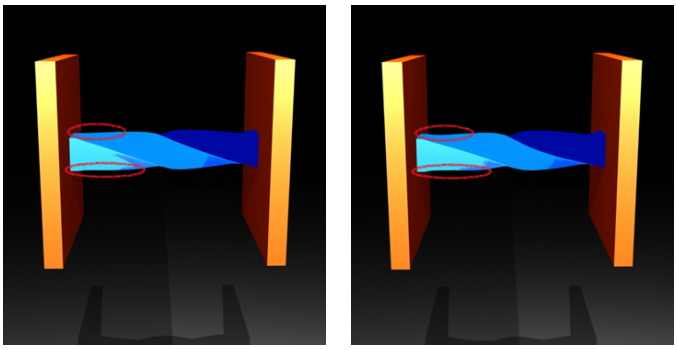
Fig. 5: Bunny model (with 8192 triangles) falls and hangs under gravity. All simulations are executed under the same conditions, such as $\alpha = 0.0001$, iteration count = 200, damping coef. = 0.3, step-size = 1/24. Volume preservation constraint is attached to SBD and ESBD. When the iteration count is higher, 3D volumetric models behave similar to a rigid-body with SBD. On the other hand, ESBD provides more stretchy behavior under exact same conditions. Furthermore, the SBD is extremely sensitive to damping, so the bunny model barely moves under the gravitational force.

- First, an integration scheme is computed to predict the position of each particle ($x \in \mathbb{R}^3$). Verlet or Euler integration schemes are mostly employed.
- Second, a non-linear Gauss-Seidel method is performed sequentially to project the final positions of the particles. This part of the algorithm requires many iteration cycles.
- Third, the resulting velocities of the particles are computed based on the new positions.

Computing the time integration scheme for XPBD is a straightforward process which provides the predictions of the particle positions within the simulation cycle. The common form of the integration scheme, that is used by XPBD, is defined as in Equation (1):

$$x_{n+1} = x_n + hv_n + h^2 w f_{ext} \tag{1}$$

where x_{n+1} and x_n are the predicted and current particle positions respectively, h is the simulation step-size, v_n is the current velocity, w is the inverse mass of the particle and f_{ext} is the applied external forces. Although Verlet or Euler methods are the most popular methods that are utilized by XPBD, it is also possible to see the second-order time integration scheme with more conserved energy features in [20]. However, in this paper we employ the integration scheme which is defined in Equation (1) due to maintaining more stability within our simulations.



SBD [1] ESBD

Fig. 6: Beam model (with 412 triangles) is fixed from both sides and twisted 180 degrees. All simulations are executed under the same conditions, such as compliance parameter $\alpha = 0.00001$, iteration count = 200, damping coef. = 0.3, step-size = 1/24. Volume preservation constraint is attached to SBD and ESBD. Since the twisted sides are fixed, the results seem visually similar but not identical. Especially, ESBD behaves slightly more hyperelastic than SBD. The differences are marked on the left side of the models.

After predicting the positions, XPBD performs a non-linear Gauss-Seidel process to obtain the final positions of the particles with an assigned iteration count. This is the most challenging part of the XPBD algorithm. Since XPBD defines the model as an interconnected particles with projected bilateral constraints ($C(x)$), the main idea is to conserve the bilateral behavior ($C(x+\Delta x) = 0$). Linear and angular momentums are also conserved implicitly in the direction of the constraint gradients ($\nabla_x C(x)$). Therefore, the particle displacements are defined as in Equation (2):

$$\Delta x_i = w_i \nabla_{x_i} C(x) \Delta\lambda_i \tag{2}$$

where $\Delta\lambda_i$ is the instant Lagrange multiplier which is obtained by substituting Equation (2) in a Taylor series by satisfying $C(x + \Delta x) = 0$ as in Equation (3):

$$C(x + \Delta x) \approx C(x) + \nabla_x C(x) \cdot \Delta x = 0 \tag{3}$$

After, the resulting Lagrange multiplier is extended with a compliance parameter (α) that is associated with the inverse material stiffness. At the end, the Lagrange multiplier is defined as in Equation (4):

$$\Delta\lambda_i = -\frac{C(x) + \tilde{\alpha}\lambda_i}{(\sum_i w_i |\nabla_{x_i} C(x)|^2) + \tilde{\alpha}} \tag{4}$$

where $\tilde{\alpha} = \alpha/h^2$, λ_i is the total Lagrange multiplier which is computed with $\lambda_i = \lambda_i + \Delta\lambda_i$ formula in each iteration. It should be noted that if α is assigned to zero, XPBD algorithm transforms to PBD.

ALGORITHM 1: XPBD Algorithm

repeat
  $x_{n+1} \leftarrow x_n + hv_n + h^2 w f_{ext}$;
  initialize total Lagrange multiplier $\lambda_0 \leftarrow 0$;
  while $k <$ *iterationCount* **do**
    for *each Constraint* **do**
      compute $\Delta\lambda$ (Eq. (4) or Eq. (5)) ;
      compute Δx (Eq. (2)) ;
      update $\lambda_{k+1} \leftarrow \lambda_k + \Delta\lambda$;
      update $x_{k+1} \leftarrow x_k + \Delta x$;
    end
    $k \leftarrow k + 1$;
  end
  update positions $x_{n+1} \leftarrow x_k$;
  update velocities $v_{n+1} \leftarrow (x_{n+1} - x_n)/h$;
until *currentFrame* $\leq$ *lastFrame*;

Furthermore, XPBD provides another extension to the Lagrange multiplier ($\Delta\lambda_i$) from Equation (4) with the Rayleigh dissipation function ($R = \frac{1}{2}\dot{C}(x)^T\beta\dot{C}(x)$ where β is the damping coefficient). This is a significant improvement in terms of performance since XPBD algorithm provides the damping implicitly within $\Delta\lambda_i$. Therefore, the new Lagrange multiplier with damping is defined as in Equation (5):

$$\Delta\lambda_i = -\frac{C(x) + \tilde{\alpha}\lambda_i + \gamma(\nabla C(x) \cdot v_n)}{(1+\gamma)(\sum_i w_i |\nabla_{x_i} C(x)|^2) + \tilde{\alpha}} \tag{5}$$

where $\gamma = \tilde{\alpha}\beta h$ and $v_n = (x_n - x_{n-1})$. However, it should be noted that redefining the Lagrange multiplier ($\Delta\lambda_i$) with the damping function makes the compliance parameter (α) significantly critical since its value has a direct impact to the overall motion of the simulation. For example, assigning higher values to α may overdamp the motion or assigning lower values may dissolve the desired dissipation and lead to unexpected results. Therefore, it should be noted that the value of α is as important as the value of β for the numerical dissipation of the overall motion.

In the final step, the velocities are corrected as shown in Equation (6):

$$v_{n+1} = (x_{n+1} - x_n)/h \tag{6}$$

After the Gauss-Seidel process computes the final positions of the particles (x_{n+1}) within the assigned iteration count, the velocities can be easily obtained based on the positions of the particles. This straightforward velocity computation makes XPBD algorithm unconditionally stable compared to the other methods that utilize the force-based velocity computations. The XPBD process is summarized in Algorithm 1.

3.2. Exponential Spring Potential Energy Function (ESPEF)

ESPEF [8] reformulates the existing spring potential energy functions on an exponential basis. ESPEF is a generic formulation that is based on the diatomic potential energy function from [53] and it can accept many different spring potential energy functions such as PBD's stretching constraint or classical Hookean spring [10]. The generic form of ESPEF is defined as in Equation (7):

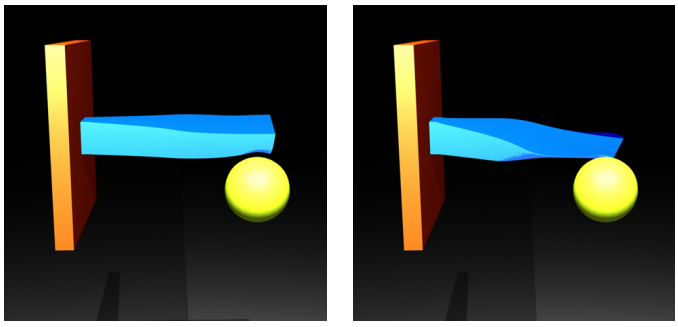
$$C_{ESPEF}(x) = k\Big(1 - e^{C_{Spring}(x)}\Big)^2 \tag{7}$$

where $x \in \mathbb{R}^3$ are the spring endpoints that consist of x_1 and x_2, $k \geq 0$ is the spring coefficient and $C_{Spring}(x)$ is the spring potential energy function.

Furthermore, the gradient of ESPEF is derived by employing the chain rule and defined as in Equation (8):

$$\nabla C_{ESPEF} = 2k(1 - e^{C_{Spring}})(-e^{C_{Spring}})\nabla C_{Spring} \tag{8}$$

Since the mass-spring concept is a very well-studied subject, it is quite straightforward to reformulate the existing spring potential functions by following ESPEF procedure which provides rapid and plausible results.



SBD [1] ESBD

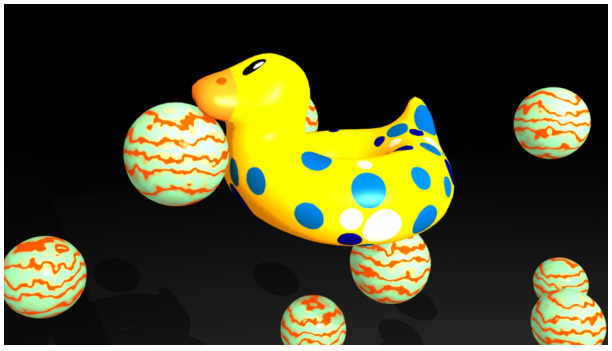
Fig. 7: Beam model (with 412 triangles) is collided with a falling sphere. All simulations are executed under the same conditions, such as $\alpha = 0.00001$, iteration count = 200, damping coef. = 0.3, step-size = 1/24. Volume preservation constraint is attached to SBD and ESBD. Since ESBD provides more hyperelasticity to the model, it generates more elastic motion after the collision. The image shows the frame number 104 of the overall sequences for both test cases and the difference is notable.

4. Method

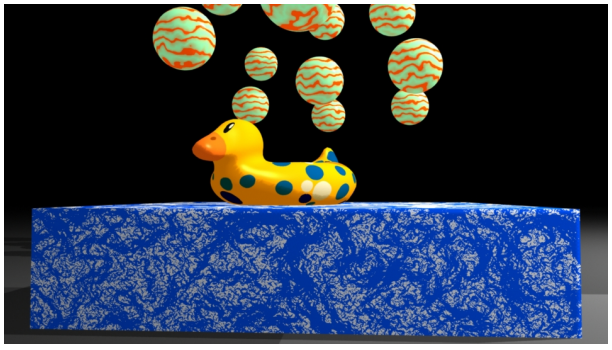
Exponential Strain-based Dynamics (ESBD) follows the same footsteps of ESPEF; nevertheless ESBD utilizes the Green's strain tensor from Strain-based Dynamics (SBD) [1] instead of the spring potential energy functions. SBD defines the Green's strain tensor as a projected constraint of PBD on triangle and volumetric meshes. The Green's strain tensor is defined as in Equation (9):

$$G = \frac{1}{2}(F^T F - I) \tag{9}$$

where I is an identity matrix, F is the deformation gradient and defined as $F = D_s D_m^{-1}$ where D_s is the deformed shape matrix, D_m is the initial material matrix. In case the model is a triangle mesh, $F \in \mathbb{R}^{2x2}$ and $D_s = (x_2 - x_1, x_3 - x_1)$ with $D_m = (X_2 - X_1, X_3 - X_1)$. If the model is a volumetric mesh, $F \in \mathbb{R}^{3x3}$ and $D_s = (x_1 - x_0, x_2 - x_0, x_3 - x_0)$ with $D_m = (X_1 - X_0, X_2 - X_0, X_3 - X_0)$. The diagonal entries of G from Equation (9) determines the principal stretches and non-diagonal entries are employed for shear. Therefore, strain-based



Sphere Collision Constraint



Arbitrary Object Collision Constraint

Fig. 8: Duck Lifebuoy model (with 6174 triangles) falls and collides with many spheres under gravity and lands on a box. The precision of the collision is tested. Top: Sphere collision constraint. Bottom: Arbitrary objects collision constraint. These tests are performed under high compliance value (0.0001) and low iteration count (50). The model with our ESBD method performs pleasing gentle contact with the sphere and box obstacles. Since the compliance value is high, the motion of the SBD simulation is damped unexpectedly. Therefore, model with SBD moves much slower than expected. The overall sequence can be found in the accompanying video vol. 1.

constraints are defined respectively as in Equations (10) and (11):

$$C_{StrainStretch}(x) = S_{ii} - 1 \tag{10}$$

$$C_{StrainShear}(x) = S_{ij}, i < j \tag{11}$$

where $S = F^T F$. Since D_s and D_m are not square for triangle meshes, it is not possible to apply an inversion operation on D_m. To overcome that problem, we follow the orthonormalization procedure explained in [1] and use the texture coordinates to define D_m in 2D space.

Furthermore, SBD also introduced the modified versions of the strain-based projected constraints. The first modification focuses on linearizing the principal stretches from Equation (10). It is achieved by taking the square root of the stretching values. The modified principal stretch is defined as in Equation (12):

$$C_{StrainStretchModified}(x) = \sqrt{S_{ii}} - 1 \tag{12}$$

Moreover, SBD modifies the shear constraint in order to decouple the shear from the principal stretches. The modified shear function is defined as in Equation (13):

$$C_{StrainShearModified}(x) = \frac{S_{ij}}{|f_i||f_j|} \tag{13}$$

where f_i and f_j represents the ith and jth column vectors of the deformation gradient ($F = [f_1, f_2]$ for 2D, $F = [f_1, f_2, f_3]$ for 3D).

Since there is no particular indication that states that ESPEF can only be applied to the spring potential functions, we transform ESPEF into ESBD by utilizing the principal stretch and shear constraints of Green's strain tensor. Therefore, we reformulated the regular and modified SBD constraints from Equation (10) to Equation (13) based on Equation (7).

According to our new definition, exponential principal stretch and exponential modified principal stretch constraints are defined respectively as in Equation (14) and (15):

$$C_{Exp.Prin.Str.}(x) = k\left(1 - e^{S_{ii}-1}\right)^2 \tag{14}$$

$$C_{Exp.Prin.Str.Mod.}(x) = k\left(1 - e^{\sqrt{S_{ii}}-1}\right)^2 \tag{15}$$

where $k \geq 0$ is an optional stiffness parameter ($k = 1$ during our simulations). Gradient derivation of ESBD is more challenging than ESPEF. In order to obtain the gradient of ESBD, we have to take the initial material matrix (D_m) into account. D_m^{-1} matrix contains two columns ($[c_1, c_2]$) for triangle meshes and three columns ($[c_1, c_2, c_3]$) for volumetric meshes. As mentioned before, the same case applies for the deformation gradient (F). By following the procedure in Equation (8), the corresponding gradients are obtained as in Equation (16) and (17):

$$\nabla_x C_{Exp.Prin.Str.}(x) = -2k\left(1 - e^{S_{ii}-1}\right)\left(-e^{S_{ii}-1}\right)\left(f_i c_i^T + f_i c_i^T\right) \tag{16}$$

SBD Mod. [1]

ESBD Mod.

Fig. 9: The cow model (with 5856 triangles) is collided with a sphere aggressively. All simulations are executed under the same conditions, such as $\alpha = 0.0001$, iteration count = 200, damping coef. = 0.3, step-size = 1/24. Volume preservation constraint is attached to SBD and ESBD. Both SBD and ESBD handle the collision well. Since ESBD has more stretchy behavior, it generates more vivid visual result than SBD.

$$\nabla_x C_{Exp.Prin.Str.Mod}(x) = -2k\Big(1 - e^{\sqrt{S_{ii}}-1}\Big)\Big(-e^{\sqrt{S_{ii}}-1}\Big)\Big(\frac{f_i c_i^T + f_i c_i^T}{2\sqrt{S_{ii}}}\Big) \tag{17}$$

Furthermore, our new definition of exponential shear and exponential modified shear constraints are shown respectively as in Equation (18) and (19):

$$C_{Exp.Shear}(x) = k\Big(1 - e^{S_{ij}}\Big)^2 \tag{18}$$

$$C_{Exp.ShearMod.}(x) = k\Big(1 - e^{\frac{S_{ij}}{|f_i||f_j|}}\Big)^2 \tag{19}$$

Gradient derivation of the exponential shear forces are essential for the Gauss-Seidel process of XPBD algortihm. Therefore, similar to the exponential principal stretches, the gradients of the exponential shear forces are obtained as in Equation (20) and (21):

$$\nabla_x C_{Exp.Shear}(x) = -2k\Big(1 - e^{S_{ij}}\Big)\Big(-e^{S_{ij}}\Big)\Big(f_j c_i^T + f_i c_j^T\Big) \tag{20}$$

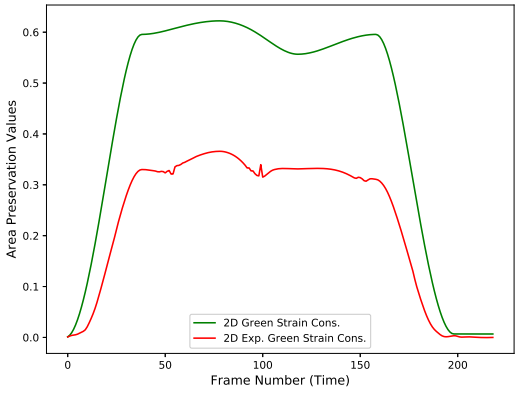
$$\nabla_x C_{Exp.ShearMod.}(x) = -2k\Big(1 - e^{\frac{S_{ij}}{|f_i||f_j|}}\Big)\Big(-e^{\frac{S_{ij}}{|f_i||f_j|}}\Big)\Big(\frac{1}{|f_i||f_j|}\nabla S_{ij}\Big) - \Big(\frac{(|f_j|^2 f_i c_i^T) + (|f_i|^2 f_j c_j^T)}{|f_i|^3|f_j|^3} S_{ij}\Big) \tag{21}$$

Since computing ESBD is more demanding than ESPEF, we provide an example function for the exponential principal stretch and shear constraints for the triangle meshes in the supplemental material. By following the same fashion, it is straightforward to implement the volumetric mesh version and the modified versions.

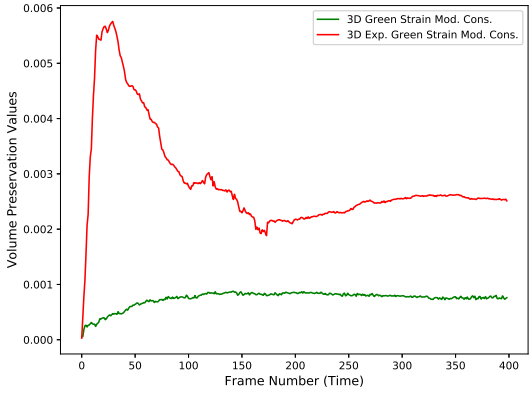
5. Volume/Area Preservation and Collision

Green's strain tensor does not provide implicit volume or area preservation unlike constitutive models such as StVK or Neo-Hookean materials. During the simulations, it is necessary to preserve the volume of solids or the area of cloth models for visually pleasing results. Therefore, we attach the volume preservation constraint ($C_{volume} = det(D_s) - det(D_m)$) for the volumetric meshes and the area preservation constraint ($C_{area} = |x_2 \times x_3|^2 - |X_2 \times X_3|^2$) for the triangle meshes. After, these constraints are attached to the exponential Green's strain tensor during our simulations. Although overconstraining the simulation causes a slight performance cost, these constraints are easy to implement and provide an effective visual outcome for the volume and area preservation.

During our simulations, we have taken advantage of the sphere and arbitrary object collision functions as the unilateral collision constraints. Sphere collision function is one of the most common methods to demonstrate the fundamental collision cases. It is straightforward to implement and performs effectively in terms of contact response speed and stability. It is defined as: $C_{sphere}(x) = R - |x - S_{cen}| \geq 0$ where x is the colliding particle, R is the sphere radius and S_{cen} is the center point of the sphere. For the collision of the objects that have arbitrary shapes other than sphere, we have implemented the arbitrary object collision function from [4]. Although the concept of collision detection of the arbitrary objects is slightly more challenging than the simple sphere collisions, they share similar principles. The main goal is to search for the closest collision point p with a surface normal n_s according to the colliding particle x. It is satisfied as: $C_{arbitrary}(x) = (x - p) \cdot n_s \geq 0$. Both sphere and arbitrary object collision functions are implemented with the other constraints including ESBD and we have not observed any significant performance leak during the collision capturing.

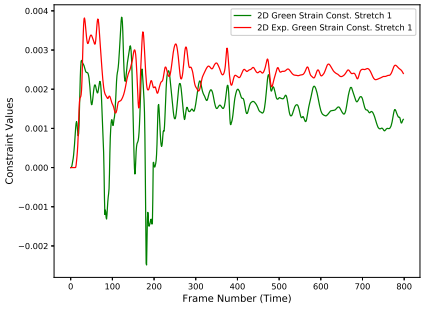


Area Preservation

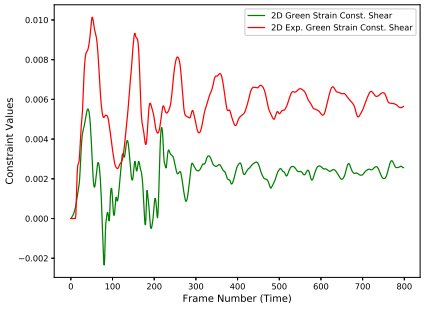


Volume Preservation

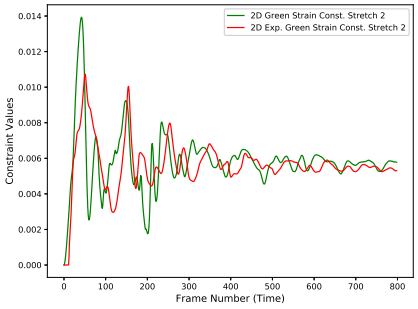
Fig. 10: Top row shows the comparison of the area preservation of the cloth stretching test case (from Figure 4). ESBD's enhanced elastic behavior provides better area preservation than SBD. Since SBD stretches the cloth with a rigid behavior, it homogeneously extends the areas of the triangle primitives of the model. Bottom row shows the comparison of the volume preservation of the bunny model test case (from Figure 5). It is not a surprising fact that the Green's strain tensor of SBD preserves the volume better than ESBD, since SBD barely moves due to extreme rigid behavior. However, this is not a serious concern. The volume differences between SBD and ESBD is within the tolerable limits which can be seen in Figure 5 and the accompanying video vol. 1.



SBD vs. ESBD (Principal Stretch 1)



SBD vs. ESBD (Shear)



SBD vs. ESBD (Principal Stretch 2)

Fig. 11: The projected constraint value differences of SBD and ESBD are compared for the triangle mesh case from the hanging cloth example in Figure 2. Although the differences in principal stretches are not notable, principal stretch 1 of ESBD provides higher results than SBD. On the other hand, shear values of ESBD completely outperforms SBD, therefore ESBD provides more plausible wrinkles and folding with a dynamic oscillation pattern.

6. Results

We have implemented our method and our own version of Strain-based Dynamics (SBD) [1] within the XPBD algorithm [6] as a plugin for Autodesk Maya by using C++. Furthermore, ESPEF [8], XPBD simulation of constitutive models [9] and local/global methods ([10] and [4]) have been implemented in the same fashion. In the scope of this paper, our plugin performs under a single-threaded CPU implementation without any third-party library. 2D and 3D triangle meshes have been used for the visual experiments and the example scenes have been generated by using Python. The resulting animation sequences can be seen in the accompanying videos volume 1 and volume 2.

6.1. Performance

Although the content of this paper does not cover any performance optimizations, both ESBD and SBD perform with similar frame rates. In our example test cases, we have observed only 2 FPS (frame per second) difference in each sequence due to the overhead of the exponential operations that are provided in our solution (in Table 1). Furthermore, it should be noted that the projected constraints that are based on the Green's strain tensor have an intrinsic disadvantage in terms of performance compared to the other projected constraints. During each iteration, SBD and ESBD compute three projected constraints (two principal stretches and one shear) for the triangle meshes and six projected constraints (three principal stretches and three shears) for the volumetric models with their gradients. However, other methods such as classical mass-spring methods (Hookean spring) or constitutive materials (Neo-Hookean or StVK materials) provide only one potential energy function to be utilized as the projected constraint within the XPBD method. For example, simulating the Bunny model with a constitutive material requires to compute only one projected constraint for the XPBD iteration cycle. On the other hand, same simulation requires to compute six projected constraints using the Green's strain tensor. Additionally, a volume preservation constraint has to be appended since Green's strain tensor does not have an implicit volume preservation unlike constitutive materials. Therefore, the performance of the other methods, such as ESPEF [8] and PBD simulation of continuous materials [9], is more advantageous than the XPBD simulation of Green's strain tensor. As a result, both SBD and ESBD methods can be considered as computation-intensive methods. Since our ESBD method targets to optimize the SBD within XPBD algorithm, comparing the performance of ESBD with the other techniques does not draw a concrete conclusion. However, our ESBD method has many other advantages especially on usability, parameter adjustments and visual aesthetics which are discussed in the next sections. All the examples presented within this paper have been performed on a 4-core Intel i7-4700 2.4 GHz machine with 8 GB of RAM and an nVidia GeForce GT 755M GPU. The details of the models and performance rates are listed in Table 1.

6.2. Triangle Meshes

We have tested the hanging cloth, flag and cloth stretching simulations with triangle meshes in Figures 2, 3 and 4. During the simulations, we have compared visual and numerical results of our method with SBD [1]. Since these comparisons are performed on triangle meshes, both formulations are computed under $F \in \mathbb{R}^{2x2}$. In order to keep the system stability and visual quality, we have applied a high number of iterations such as 200. Besides that, we have applied a small damping value for overall aesthetics of the system. ESPEF method [8] already showed that the exponential formulations provide vivid motion under high number of iterations and compliance value. We observe the same fact even more clear while comparing ESBD with SBD. The cloth, that falls and hangs under the constant gravitational force (in Figure 2), damps the motion earlier in SBD than our ESBD method. Furthermore, cloth with ESBD formulation moves more dynamic than SBD. Numerical results also support the visual simulations. In Figure 11, we plot the principal stretches and shear differences that are computed as position-based constraints. We observe a slight difference in principal stretches. Principal stretch 1 produces slightly higher results in ESBD, but principal stretch 2 surprisingly produces the same results. On the other hand, shear differences are notably higher in our ESBD method than SBD which explain the

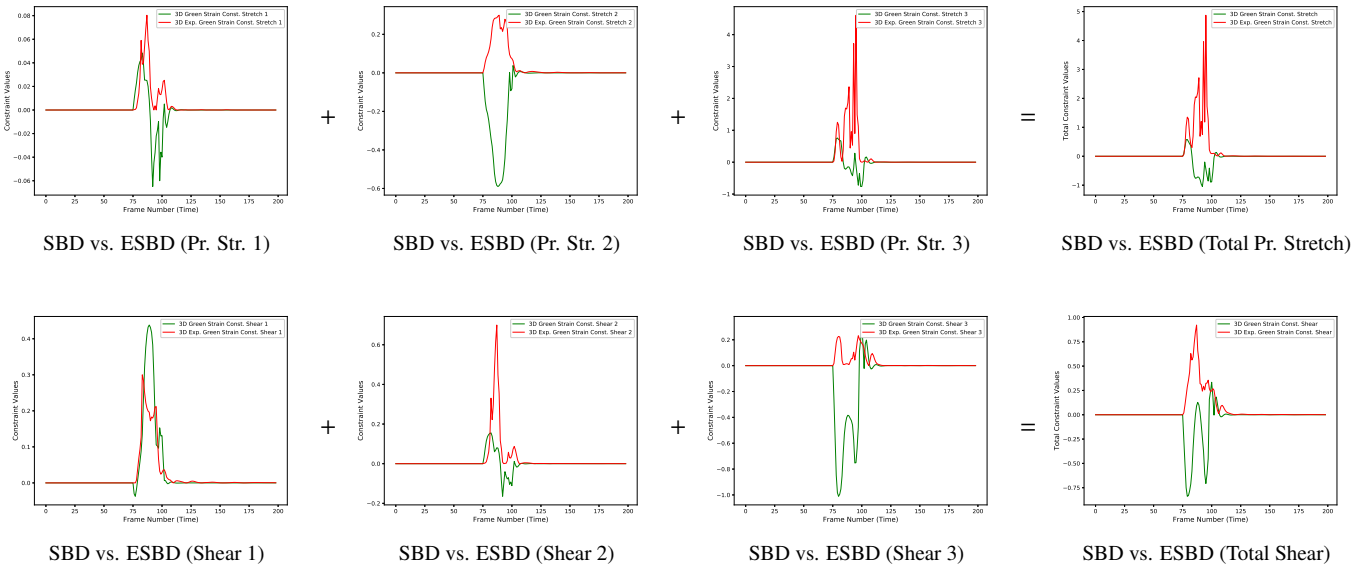


Fig. 12: The projected constraint value differences of SBD and ESBD are compared for the beam-sphere collision test case (from Figure 7). In each constraint (for each principal stretch and shear constraints), there are interesting and distinct differences in each frame. Although some cases are in favor of ESBD and some cases are in favor of SBD, the total principal stretch and shear values clearly demonstrate the higher results for ESBD which cause the stretchy and vivid oscillation pattern to the beam model during and after the collision.

vivid oscillations and pleasing wrinkles under the gravitational force. During the flag simulations (in Figure 3), we have applied a constant wind force to the model. Since the wind force is already a random noisy type of external force, it is not possible to plot a graph. However, the visual differences are obvious that ESBD formulation produces more dynamic blows than SBD. The method of SBD suffers from the impact of high compliance value on the damping function. Therefore, SBD always tends to damp the motion which can not be observed with ESBD. In another simulation test case with the triangle meshes, we have fixed the left and right sides of the cloth and stretched simultaneously to both sides. Side by side comparison (in Figure 4) shows that the visual results are rather different. Since ESBD provides more stretchy behavior with the enhanced shear term, cloth stretches more softer with ESBD method. Besides that, we have attached the area preservation constraint to SBD and ESBD in this test case. The results (in Figure 10, top) demonstrate that the area is conserved better with ESBD than SBD. This is due to the stiff behavior of SBD which causes homogenous elongation to the triangle primitives of the cloth model. This issue is not observed in ESBD.

We have also tested ESBD against other well-known methods, such as ESPEF [8] and local/global methods ([10] and [4]). For the simulation of hanging cloth, we have compared our ESBD method with ESPEF - stretching and ESPEF - Hookean spring methods with $k_{stiff} = 100$ where k_{stiff} is the spring stiffness. We know that both methods share the same exponential foundations, so ESBD and ESPEF share the similar energy pattern. This fact can be clearly seen in the accompanying video vol. 1. However, since the details of the formulations and computations are quite different, ESPEF - Hookean spring behaves more loose than ESBD under the same conditions. This can be seen in Figure 13 (top). For the flag simulation, we have compared ESBD with the local/global methods, such as Fast Simulation of Mass-Spring Sys. [10] and Projective Dynamics [4]. Both XPBD and the local/global methods utilize the implicit numerical integration method and are well-known as fast methods for interactive applications. However, as the beneath solvers of both methods are clearly distinct, the resulting flag simulations perform with different motion patterns. Fast Simulation of Mass-Spring Sys. [10] and Projective Dynamics [4] employ an alternating optimization approach to solve the potential energy functions of the models whereby the iteration count is not a limiting factor. Therefore, 10 iterations are sufficient for the local/global methods with 0.8 stiffness to dynamically blow the flag under the constant wind force. On the other hand, ESBD blows the flag sharper with the default parameter set and provides a more vivid sequence under the same wind force. The differences are presented in Figure 16 and the accompanying video vol. 1.

6.3. *Volumetric Meshes*

In other test cases, we have tested the volumetric meshes with the bunny model in Figure 5, the beam model in Figure 6 and the torus model (that can be seen in the accompanying video vol. 1). During our tests with volumetric meshes, SBD and ESBD formulations are computed under $F \in \mathbb{R}^{3x3}$ and the volume preservation constraint is attached. Besides that, it should be noted that our volumetric meshes are also triangle meshes that are used as the surface mesh of the volumetric model. This is achieved by tracking the center of mass of the model in each simulation cycle. In our first test case with the volumetric models, we have hung the bunny model from the ears under the gravitational force (in Figure 5). SBD simulation suffers from

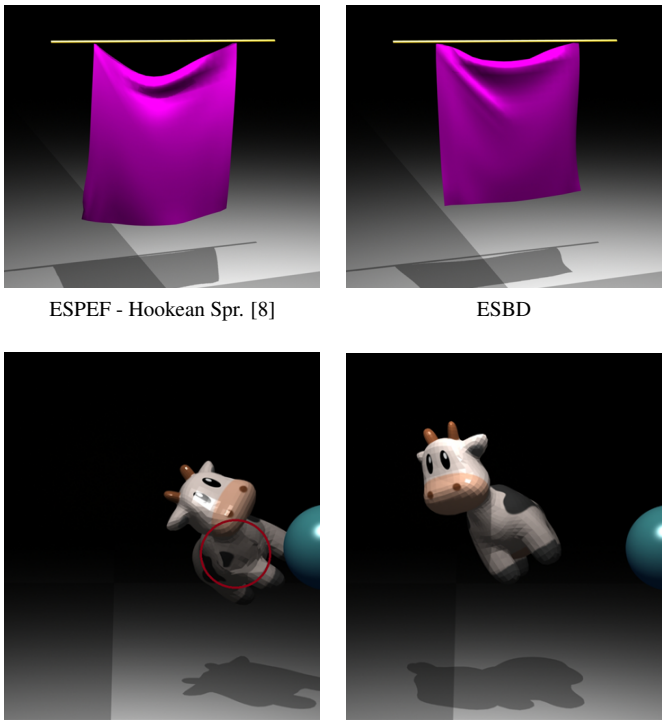


Fig. 13: At the top row, a piece of cloth (with 800 triangles) falls and hangs under gravity. At the bottom row, a cow model (with 5856 triangles) is collided with a sphere aggressively. All simulations are executed under the same conditions, such as iteration count = 200, damping coef. = 0.3, step-size = 1/24, compliance parameter $\alpha = 0.00001$ for cloth and $\alpha = 0.0001$ for cow. Although ESPEF and ESBD share the same exponential foundations, ESPEF - Hookean spring constraint behaves more loose than ESBD for the cloth test case. At the bottom row, StVK material could not recover the cow model successfully after an aggressive collision. Under the same simulation conditions, ESBD instantly recovers the model from the collision impact.

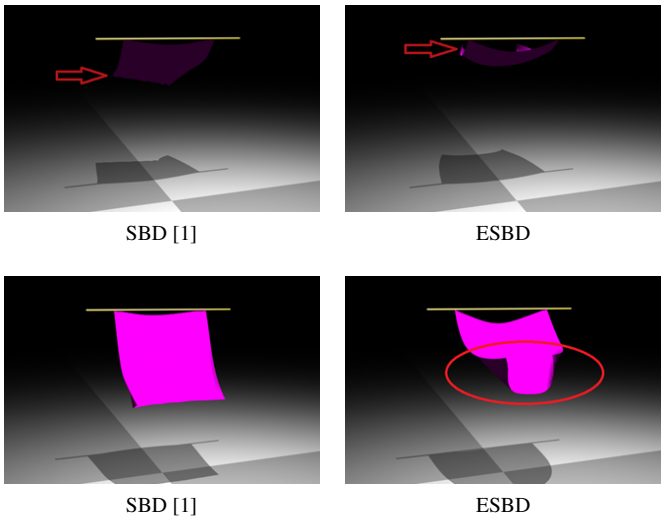
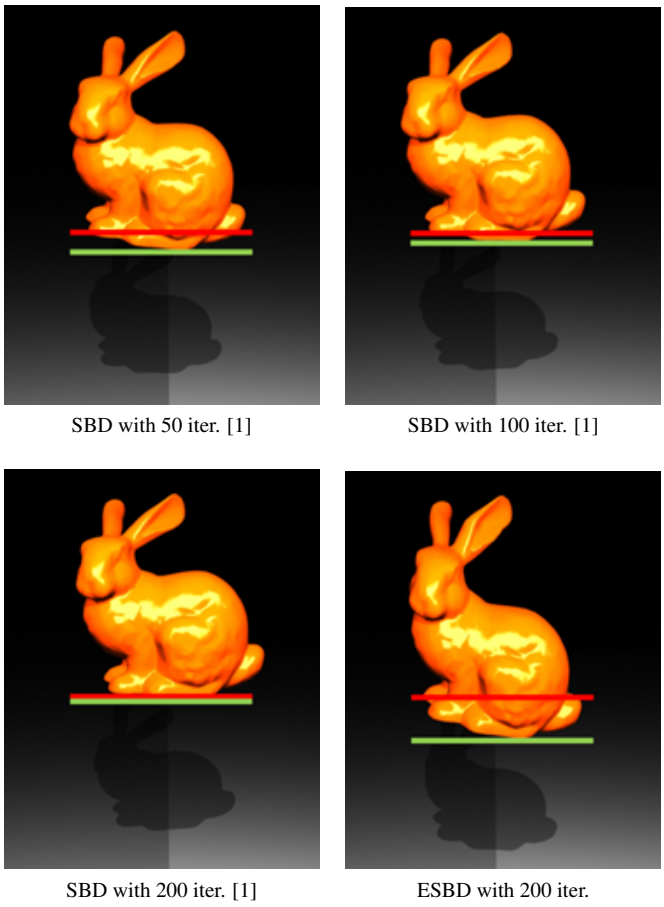


Fig. 14: A piece of cloth (with 800 triangles) falls and hangs under gravity. SBD by using XPBD algorithm may require many parameter adjustments to achieve vivid simulation results. On the other hand, our ESBD method provides vivid visual results with the default parameter set. In the cloth simulation test case, we have altered the compliance parameter and iteration count of SBD and maintained the default parameter set of ESBD. Under different conditions, our ESBD method has provided more oscillations, wrinkles and buckling than SBD. The highlighted differences are obtained for the same frames and the detailed comparisons are provided in the accompanying video vol. 2.

the high compliance value and iteration count tremendously and the bunny model has barely moved which is not possible to observe in the supplemental video vol. 1. On the other hand, our ESBD method has provided a plausible stretchy effect to the bunny model that allowed the model to move under the gravitational force. The bunny model has completely behaved like a static rigid-body under the SBD formulation. Therefore, the volume is better preserved with the SBD method than ESBD. This is not a surprising fact, since the bunny model has been barely deformed with SBD. While the volume differences are quite small that are presented in Figure 10 (bottom), we do not necessarily consider this fact as a disadvantage, especially considering the notable volume changes in other methods, such as ESPEF [8] and [37]. In another test case, we gently stretch and compress the torus model manually. The model is fixed from the bottom vertices. During our interactive simulations with ESBD, stretching the model has been snappy and ESBD produces plausible jiggles on the surface of the model. On the other hand, it is possible to produce jiggles on the surface of the model by using the SBD formulations, but again surface jiggling has been damped due to the impact of the high compliance value on the damping function. In the beam twisting test case (in Figure 6), we have fixed the right and left sides of the beam and rotated 180 degrees from the right side. Although the visual results of both ESBD and SBD appear similar, there exist slight differences in the details which are caused due to higher elastic energy output of ESBD. The differences are marked in Figure 6 and the accompanying video vol. 1.

6.4. *Collisions*

Collision and contact capture are one of the most critical aspects of the simulation frameworks. In the scope of this paper, we have utilized sphere and arbitrary object collision functions. In our collision test cases, we have tested both the triangle and the volumetric meshes in order to observe the differences between SBD and ESBD under the collision. In our first test case, the cloth falls on the static sphere model under both SBD and ESBD simulations (that can be seen in the accompanying video vol. 1). Since ESBD provides more dynamic motion to the cloth model, we observe more wrinkle generation under ESBD simulations. On the other hand, due to the high compliance value and iteration count, SBD simulates the cloth model similar to a “thin aluminum sheet”. Furthermore, we have tested another case where a duck lifebuoy (volumetric) model falls under gravity. The model gently contacts with many spheres and lands on a box (in Figure 8). In this test case, an arbitrary object collision function has been employed additionally for the box obstacle. Our ESBD method performs the collisions smoothly as expected. It should be noted that high compliance value and low iteration count have been assigned for this collision test case. Since the high compliance value has a direct impact on the damping of the simulation, the motion of SBD is damped unexpectedly. The model could not reach to the box obstacle.

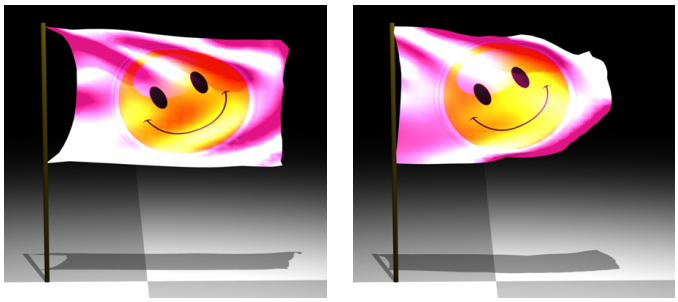


SBD with 50 iter. [1] SBD with 100 iter. [1]

SBD with 200 iter. [1] ESBD with 200 iter.

Fig. 15: Bunny model (with 8192 triangles) falls and hangs under gravity. Volume preservation constraint is attached to SBD and ESBD. The default parameter set has been maintained for the ESBD simulations. On the other hand, SBD simulations have been executed with $\alpha = 0.0001$ and different iteration counts, such as 50, 100 and 200. When the iteration count gets higher, the stiff behavior of the 3D volumetric model increases. SBD provides rather limited stretching to the model with 50 and 100 iteration counts. Bunny model barely moves with 200 iteration count. This artifact is not observed with ESBD simulations using the default parameter set. Red and green lines show the initial and final positions of the model from the bottom level respectively. More comparisons are provided in the accompanying video vol. 2.

In ESBD, this issue has not been observed. After, we have simulated two reverse cases with volumetric models. In these test cases, we throw a sphere to a beam model which is fixed from the left end (in Figure 7) and a cow model (in Figure 9). Both SBD and ESBD methods perform well. However, ESBD provides more stretchy behavior to the models, so the beam and cow models handle the contact with a more plausible hyperelastic feature. The difference is more notable during the simulation with the beam model. The quantitative differences in Figure 12 show that ESBD notably improves the hyperelasticity of SBD especially in total principal stretches under the high compliance and iteration count values. We have highlighted these comparisons in the accompanying video vol. 1. In our last test case, we have compared ESBD with constitutive models, such as StVK and Neo-Hookean materials (XPBD version of [9]). After the sphere aggressively contacts with the cow model, constitutive models are not able to recover the model successfully. Under



Projective Dynamics [4] ESBD

Fig. 16: A flag (with 800 triangles) blows under the wind force. ESBD is simulated with the compliance parameter $\alpha = 0.00001$, iteration count = 200, damping coef. = 0.3, step-size = 1/24. Projective Dynamics is simulated with iteration count = 10, stiffness coef. = 0.8. Since the beneath solvers of both methods are distinct, the motion patterns are not similar. Our ESBD method blows the flag sharper. The image shows the frame number 415 of the overall sequences for both test cases and the difference is notable.

the same simulation conditions, this fact is not observed with ESBD (in accompanying video vol. 1 and Figure 13, bottom).

6.5. Comparison With Different Parameters

In the accompanying video vol. 2, we have compared ESBD with the Green's strain tensor of SBD under different parameter values. During these comparisons, our ESBD method maintains its default parameter set in order to highlight the fact that ESBD provides visually attractive results without tuning the parameters of XPBD algorithm. We already know that XPBD has several parameters to tune for obtaining vivid visual results such as iteration count, compliance parameter, step-size, stiffness, damping coefficient, etc. Therefore, the number of combinations to alter these parameters can be excessively high. In our comparison tests, we pick only the iteration count and compliance parameters which have notable impact on XPBD method and assign different values to them one by one for the simulation of Green's strain tensor of SBD.

Compliance Parameter: The numerical value of compliance parameter (α) carries a critical importance as it has a direct impact to the damping function. This fact can be easily seen in Eq. (5) where γ is composed of compliance parameter and damping coefficient. Therefore, when we assign lower values to compliance parameter such as 1.0×10^{-6} or 1.0×10^{-5} for the simulation of the Green's strain tensor of SBD, it degrades the effect of the damping function on the system. Furthermore, the model loses its deformable behavior significantly, since compliance parameter defines the inverse stiffness, and acts as if it is a rigid body. These facts can be clearly seen in the test cases of bunny and cow models. On the other hand, when the compliance parameter increases to higher values such as 1.0×10^{-4} or 1.0×10^{-3}, it provides an unintended increase to the numerical dissipation. However, the model slightly maintains its deformable behavior. During the test cases of volumetric models, these artifacts can be clearly seen in the accompanying video vol. 2. The cloth simulations are also affected from these shortcomings, especially we have not observed vivid oscillations and pleasing wrinkle generations during the simulations (in Figure 14).

Table 1: Details of the models and performance rates of our example cases. Frame rates are obtained directly from the default interface of Maya. SBD performs slightly better than ESBD. Performance differences (2 FPS in each sequence) are due to the exponential operations that we utilize in ESBD computations.

Details of the models with iteration counts and frame rate (FPS) values					
Model	# vertices	# edges	# faces (tri.)	# iters	FPS
Cloth (with SBD [1])	441	1240	800	200	20
Cloth (with ESBD)	441	1240	800	200	18
Torus (with SBD [1])	900	2700	1800	50	34
Torus (with ESBD)	900	2700	1800	50	32
Pirate Flag (with SBD [1])	441	1240	800	200	20
Pirate Flag (with ESBD)	441	1240	800	200	18
Bunny (with SBD [1])	4098	12288	8192	200	7
Bunny (with ESBD)	4098	12288	8192	200	5
Beam (with SBD [1])	208	618	412	200	36
Beam (with ESBD)	208	618	412	200	34
Cow (with SBD [1])	2930	8784	5856	200	9
Cow (with ESBD)	2930	8784	5856	200	7
Duck Lifebuoy (with SBD [1])	3087	9261	6174	50	13
Duck Lifebuoy (with ESBD)	3087	9261	6174	50	11

Iteration Count: The stability, convergence and the stiffness of the models are directly dependent to the iteration count of the XPBD algorithm. In our tests, we set the iteration counts to 50, 100 and 200 for the XPBD solver (in Figure 15). Therefore, stability is not a concern in our test cases. However, we clearly observe that numerical dissipation is still an issue with the Green's strain tensor of SBD. For example, the cloth model resists to oscillate during the simulations. Moreover, the volumetric models maintains their deformable behavior without any vivid jiggling effect which is the outcome of the uncontrollable dissipation during the simulation of the Green's strain tensor of SBD. We provide a clear demonstration in the accompanying video vol. 2.

The aforementioned shortcomings are not observed with our ESBD method with its default parameter set. ESBD solves the linearity of the Green's strain tensor of SBD with its exponential form and provides a straightforward solution to the uncontrollable numerical dissipation. After all the demonstrated test cases, we conclude that our ESBD method can be suited for the applications that require enhanced hyperelastic effects with collision and contact capturing. However, that does not necessarily mean that the presented simulation results can only be obtained by using our ESBD method. The other methods can obtain the similar (or closer) results but they require a cumbersome parameter adjustments and many trial and error tests. ESBD method rather simplifies this process, as shown previously, especially for the complex scenes where stiff and soft materials are employed simultaneously.

7. Conclusions

In this paper, we have proposed Exponential Strain-based Dynamics (ESBD) as a solution to the extreme stiff behavior of Strain-based Dynamics (SBD) [1] for the XPBD [6] simulation of deformable objects. ESBD takes advantage of the generic formulation of ESPEF [8] and does not require any modification to the underlying XPBD algorithm. Since ESBD reformulates the regular and modified Green's strain tensor of SBD on an exponential basis, the linear response of SBD is optimized to nonlinear elastic response in our method. Therefore, ESBD improves the hyperelastic behavior of the Green's strain tensor without any additional fine-tuning operations. Moreover, ESBD provides more dynamic motions while SBD suffers from the unexpected numerical dissipation due to the higher parameter values of the compliance term and iteration count. We have presented visual and quantitative comparisons of ESBD and SBD under the exact same conditions. Moreover, we have compared ESBD with the other advanced methods, such as ESPEF [8], Projective Dynamics [4] and XPBD version of constitutive materials [9]. The results clearly show the advantages of our method over the XPBD simulation of Strain-based Dynamics and the other well-known methods. ESBD has the capability to operate on any triangular and volumetric models similar to SBD. Our method is straightforward, easy to implement and adaptable to the existing simulation frameworks.

7.1. Limitations and Future Work

Our ESBD method and SBD [1] run within the XPBD algorithm in the scope of this paper; hence, we inherit the same advantages and disadvantages of XPBD in terms of convergence regime, scalability and stability. There is still enough room to extend our method in several ways. While our ESBD method generates visually pleasing motion and plausible deformations, our method may behave sensitive under the extreme conditions, such as extreme stretches and extreme compression. This is due to the fact that the exponential-based formulation of the Green's strain tensor may produce too high or too low results that are beyond the expected limits of the model under the extreme conditions. This can be resolved by applying higher-order numerical integration methods that provide more accurate solutions to the Gauss-Seidel solver. Furthermore, employing a more accurate numerical integration technique can improve the compliance dependency in both SBD and ESBD methods and yield more vivid results with better energy conservation regardless of the compliance parameter value.

In the future, we plan to adapt ESBD to the recent version of XPBD with Small Steps [17]. Moreover, we would like to apply our exponential framework to the constitutive models such as StVK and Neo-Hookean materials. Furthermore, we plan to improve our collision and contact handling implementation with more advanced methods such as [54] and [31].

Acknowledgments

Author would like to thank José Serra for proofreading, Keenan Crane for sharing "Duck Lifebuoy" and "Cow" models. Bunny model is courtesy of Stanford University. This work is funded by FCT/MCTES through national funds and when applicable co-funded EU funds under the project UIDB/50008/2020. This work is also funded by FCT Position (CEECIND/03766/2018/CP1579/CT0002).

References

[1] Müller, M, Chentanez, N, Kim, TY, Macklin, M. Strain based dynamics. In: Eurographics/ ACM SIGGRAPH Symposium on Computer Animation. 2014,.

[2] Sifakis, E, Barbic, J. Fem simulation of 3d deformable solids: A practitioner's guide to theory, discretization and model reduction. In: ACM SIGGRAPH Courses. 2012, p. 20:1–20:50.

[3] Müller, M, Heidelberger, B, Hennix, M, Ratcliff, J. Position based dynamics. Journal of Visual Communication and Image Representation 2007;18(2):109–118.

[4] Bouaziz, S, Martin, S, Liu, T, Kavan, L, Pauly, M. Projective dynamics: Fusing constraint projections for fast simulation. ACM Trans Graph 2014;33(4):154:1–154:11.

[5] Jiang, C, Schroeder, C, Teran, J, Stomakhin, A, Selle, A. The material point method for simulating continuum materials. In: ACM SIGGRAPH Courses. ACM; 2016,.

[6] Macklin, M, Müller, M, Chentanez, N. Xpbd: Position-based simulation of compliant constrained dynamics. In: Proceedings of the 9th International Conference on Motion in Games. MIG '16; ACM; 2016, p. 49–54.

[7] Cetinaslan, O, Chaves, RO. Energy embedded gauss-seidel iteration for soft body simulations. In: 32nd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI). 2019, p. 147–154.

[8] Cetinaslan, O. Espefs: Exponential spring potential energy functions for simulating deformable objects. In: Proceedings of the 14th ACM SIGGRAPH Conference on Motion, Interaction and Games. MIG '21; ACM; 2021,.

[9] Bender, J, Koschier, D, Charrier, P, Weber, D. Position-based simulation of continuous materials. Computers & Graphics 2014;44(0):1 – 10.

[10] Liu, T, Bargteil, AW, O'Brien, JF, Kavan, L. Fast simulation of mass-spring systems. ACM Trans Graph 2013;32(6):214:1–214:7.

[11] Faure, F. Interactive solid animation using linearized displacement constraints. In: Computer Animation and Simulation. Springer Vienna; 1999, p. 61–72.

[12] Jakobsen, T. Advanced character physics. In: In Proceedings of the Game Developers Conference. 2001, p. 383–401.

[13] Müller, M, Heidelberger, B, Teschner, M, Gross, M. Meshless deformations based on shape matching. ACM Trans Graph 2005;24(3):471–478.

[14] Stam, J. Nucleus: Towards a unified dynamics solver for computer graphics. In: 11th IEEE International Conference on Computer-Aided Design and Computer Graphics, CAD/Graphics '09. 2009, p. 1–11.

[15] Macklin, M, Müller, M, Chentanez, N, Kim, TY. Unified particle physics for real-time applications. ACM Trans Graph 2014;33(4).

[16] Tournier, M, Nesme, M, Gilles, B, Faure, F. Stable constrained dynamics. ACM Trans Graph 2015;34(4):132:1–132:10.

[17] Macklin, M, Storey, K, Lu, M, Terdiman, P, Chentanez, N, Jeschke, S, et al. Small steps in physics simulation. In: Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2019, p. 2:1–2:7.

[18] Cetinaslan, O. Position-based simulation of elastic models on the gpu with energy aware gauss-seidel algorithm. Computer Graphics Forum 2019;38(8):41–52.

[19] Macklin, M, Erleben, K, Müller, M, Chentanez, N, Jeschke, S, Kim, TY. Primal/dual descent methods for dynamics. Computer Graphics Forum 2020;39(8):89–100.

[20] Bender, J, Müller, M, Macklin, M. A survey on position based dynamics, 2017. In: EUROGRAPHICS Tutorials. 2017,.

[21] Müller, M, Chentanez, N. Wrinkle meshes. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2010, p. 85–92.

[22] Müller, M, Chentanez, N. Solid simulation with oriented particles. ACM Trans Graph 2011;30(4):92:1–92:10.

[23] Diziol, R, Bender, J, Bayer, D. Robust real-time deformation of incompressible surface meshes. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2011, p. 237–246.

[24] Kim, TY, Chentanez, N, Müller-Fischer, M. Long range attachments - a method to simulate inextensible clothing in computer games. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2012, p. 305–310.

[25] Müller, M, Kim, TY, Chentanez, N. Fast Simulation of Inextensible Hair and Fur. In: Workshop on Virtual Reality Interaction and Physical Simulation. The Eurographics Association; 2012,.

[26] Umetani, N, Schmidt, R, Stam, J. Position-based elastic rods. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2014, p. 21–30.

[27] Deul, C, Kugelstadt, T, Weiler, M, Bender, J. Direct position-based solver for stiff rods. Computer Graphics Forum 2018;37(6):313–324.

[28] Angles, B, Rebain, D, Macklin, M, Wyvill, B, Barthe, L, Lewis, J, et al. Viper: Volume invariant position-based elastic rods. Proc ACM Comput Graph Interact Tech 2019;2(2).

[29] Cetinaslan, O, Orvalho, V. Localized verlet integration framework for facial models. In: Articulated Motion and Deformable Objects - 9th International Conference, AMDO, Proceedings; vol. 9756 of *Lecture Notes in Computer Science*. Springer; 2016, p. 1–15.

[30] Cetinaslan, O. Localized constraint based deformation framework for triangle meshes. Entertainment Computing 2018;26:78–87.

[31] Macklin, M, Erleben, K, Müller, M, Chentanez, N, Jeschke, S, Corse, Z. Local optimization for robust signed distance field collision. Proc ACM Comput Graph Interact Tech 2020;3(1).

[32] Macklin, M, Muller, M. A constraint-based formulation of stable neo-hookean materials. In: Motion, Interaction and Games. MIG '21; ACM; 2021,.

[33] Smith, B, Goes, FD, Kim, T. Stable neo-hookean flesh simulation. ACM Trans Graph 2018;37(2).

[34] Müller, M. Hierarchical position based dynamics. In: Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS". 2008,.

[35] Wang, H. A chebyshev semi-iterative approach for accelerating projective and position-based dynamics. ACM Trans Graph 2015;34(6):246:1–246:9.

[36] Fratarcangeli, M, Tibaldo, V, Pellacini, F. Vivace: A practical gauss-seidel method for stable soft body dynamics. ACM Trans Graph 2016;35(6):214:1–214:9.

[37] Cetinaslan, O. Parallel XPBD Simulation of Modified Morse Potential - an Alternative Spring Model. In: Eurographics Symposium on Parallel Graphics and Visualization. The Eurographics Association; 2019,.

[38] Kugelstadt, T, Koschier, D, Bender, J. Fast corotated fem using operator splitting. Computer Graphics Forum 2018;37(8).

[39] Pall, P, Nylén, O, Fratarcangeli, M. Fast Quadrangular Mass-Spring Systems using Red-Black Ordering. In: Workshop on Virtual Reality Interaction and Physical Simulation. The Eurographics Association; 2018,.

[40] Liu, T, Bouaziz, S, Kavan, L. Quasi-newton methods for real-time simulation of hyperelastic materials. ACM Trans Graph 2017;36(3):23:1–23:16.

[41] Narain, R, Overby, M, Brown, GE. ADMM ⊇ projective dynamics: Fast simulation of general constitutive models. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2016, p. 21–28.

[42] Baraff, D, Witkin, A. Large steps in cloth simulation. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques. 1998,.

[43] Kim, T. A finite element formulation of baraff-witkin cloth. Computer Graphics Forum 2020;39(8):171–179.

[44] Xu, H, Sin, F, Zhu, Y, Barbič, J. Nonlinear material design using principal stretches. ACM Trans on Graphics 2015;34(4).

[45] Xu, H, Li, Y, Chen, Y, Barbič, J. Interactive material design using model reduction. ACM Trans on Graphics 2015;34(2).

[46] Li, Y, Barbič, J. Stable anisotropic materials. IEEE Trans on Visualization and Computer Graphics 2015;21(10):1129–1137.

[47] Barbič, J, Sin, F, Grinspun, E. Interactive editing of deformable simulations. ACM Trans on Graphics 2012;31(4).

[48] Michels, DL, Sobottka, GA, Weber, AG. Exponential integrators for stiff elastodynamic problems. ACM Trans Graph 2014;33(1).

[49] Michels, DL, Luan, VT, Tokman, M. A stiffly accurate integrator for elastodynamic problems. ACM Trans Graph 2017;36(4).

[50] Chen, YJ, Ascher, UM, Pai, DK. Exponential rosenbrock-euler integrators for elastodynamic simulation. IEEE Transactions on Visualization and Computer Graphics 2018;24(10):2702–2713.

[51] Chen, YJE, Sheen, SH, Ascher, UM, Pai, DK. Siere: A hybrid semi-implicit exponential integrator for efficiently simulating stiff deformable objects. ACM Trans Graph 2020;40(1).

[52] Kim, T, Eberle, D. Dynamic deformables: Implementation and production practicalities. In: ACM SIGGRAPH Courses. ACM; 2020,.

[53] Dahl, JP, Springborg, M. The morse oscillator in position space, momentum space, and phase space. The Journal of Chemical Physics 1988;88(7):4535–4547.

[54] Müller, M, Chentanez, N, Kim, TY, Macklin, M. Air meshes for robust

collision handling. ACM Trans Graph 2015;34(4):133:1–133:9.