Towards the non-stretchable and non-elongating string with stress-strain handling

R. ĐURIKOVIČ AND E. SIEBENSTICH

Abstract

We propose an approach for real-time physically semi-realistic animation of strings which directly manipulates the string positions by position based dynamics. The main advantage of a position based dynamics is its controllability. Instability problems of explicit integration schemes can be avoided. Specifically, we offer the following three contributions. We introduce the non-elongating and non-stretchable mass spring dynamics model based on Position Based Dynamics to simulate 1D string. We introduce a method for propagating the twisting angle along the chain of segments. In addition, we solve collision constraints by regularly distributing the spheres along the chain segments followed by particle projection to validate the positions. Proposed strain limiting constraint can handle the strings fixed in multiple locations contrary to single fixed side as is common for hair models. The use of multiple constraints provides an efficient treatment for stiff twisting and non-stretchable mass spring dynamics model.

Mathematics Subject Classification 2010: 78A05, 68P01, 68U05

Keywords: non-elongating strings, twisting, tearing, flicking, string animation

1. INTRODUCTION

In this paper we present a method for real-time physically semi-realistic animation of strings with emphasis on believable behavior during twisting, tearing and flicking. The last result is the physically-based model that is visually plausible and stable, yet simple to implement.

The contribution of this paper: 1. We propose the non-elongating and non-stretchable properties of mass spring dynamics model based on Position Based Dynamics (PBD) [Müller et al. 2007] to simulate 1D string. 2. We propose a method for propagating the twisting angle increments associated with each segment, which can handle both uniform and nonuniform torsional rigidity. 3. We propose a method for tension calculation resulting from twisting, tearing and flicking.

We describe details of implementation and stumbling blocks we've encountered due to hardware limitation, design choices and other reasons. Finally, we evaluate results of our system compared to real world samples. We analyze system's performance

©IEEE CS, 2018. This is a revision of the work published in 2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW), ISBN: 978-1-5386-9184-7, Takayama, Japan

both from perspective of visual fidelity and responsiveness and give examples of our system's output.

2. RELATED WORK

The modeling nonlinear dynamics, and simulation of elastic rods is an active field from numerical and topological point of view in mathematical analysis [van der Heijden and Thompson 2000; Goyal et al. 2008]. It is impossible to survey the many works in this area, so for the state of the art in strand and hair simulation, refer to the survey by Ward et al. [Ward et al. 2007].

In graphics, Pai [K. 2002] applied a discretization of the Cosserat rod model to simulate a strand. Bertails et al. [Bertails et al. 2006] used a piecewise helical discretization to produce compelling animations of curly hair using few elements per strand.

In order to simulate a string, several traditional methods have been proposed, such as mass-spring systems [Hair 1991; Selle et al. 2008], rigid multibody serial chains [Hadap 2006], geometric approaches [Rivers and James 2007], and elastic energy-based models [Bertails et al. 2006; Spillmann and Teschner 2007; Bergou et al. 2008]. However, effects such as twisting, tearing, and flicking of a string are not considered all together in a single dynamic system.

In order to handle inextensible objects simulated by deformation models, different methods for stretch resistance have been proposed Provot [Institut and Provot 1996] and Bridson et al. [Bridson et al. 2005] proposed to constrain the length of springs to not stretch or compress beyond a given limit in the context of mass spring simulations. Some alternative ways of stabilizing stiff simulation using a global solver of a regular quad mesh were also proposed [Baraff and Witkin 1998; Müller et al. 2014; Goldenthal et al. 2007], including a constraint method based on impulse [Irving et al. 2007].

In the Finite Element Method (FEM) research area, Picinbono et al. [Picinbono et al. 2003] limit strain by adding an energy term to penalize strain in certain directions. Perez et al. [Perez et al. 2013] used Lagrange multipliers to constrain strain components isotropically. Hernandez et al. [Hernandez et al. 2013] improve this method to support anisotropic material. The linear system resulting from these methods becomes over-constrained and has to be solved as a least squares problem. In contrast, in the PBD framework, vertices are simply fixed by setting their inverse mass to zero.

These methods and many of their sequels are problematic in the case of excessive stretch or when discontinuous events such as rupture should occur.

3. PROPOSED STRING MODEL

Before we describe the methodology of our model, we would like to clarify few terms we often use in reference to our string model that might seem contradictory to how our model actually behaves.

Non-elongating - we regard elongation as permanent deformation of string represented by increased length of segments. What we truly mean when we use a term non-elongating is that this deformation does not happen arbitrarily but only as an exact deformation defined by stress-strain curve. As such it is also possible to create an instance of our string that never elongates.

Non-stretchable - similarly to non-elongation, we use it to define any stretching of our string as systematic and dependent on stress-strain curve. Only exception is created by parameter used in strain limiting that can allow for some stretching to decrease amount of necessary cycles needed when string is fixed in 2 or more particles.

Proposed model based on Position Based Dynamics approach (PBD) in this paper is a modification of [Selle et al. 2008; Rungjiratananon et al. 2012; Rungjiratananon et al. 2010] method used to simulate elastic objects and proven itself to be fast, stable and simple. It was newly modified to take advantage of the fact that string as structure is ultimately 1-dimensional object which allowed for further simplification. Proposed modifications are in collision constraint module that can be handled easily, by resolving the penetrations with regularly spreading the spheres along the segment followed by projection of particles to valid locations. We also propose a modification of strain limiting constraint that can handle the strings fixed in multiple locations contrary to single fixed side as is common for hair models [Rungjiratananon et al. 2012; Rungjiratananon et al. 2010]. We also propose the use of constraints and constraint forces, providing an efficient treatment for stiff twisting and non-stretchable mass spring dynamics model.

3.1. String

Underlying structure of the string model consists of *regions*, n *segments* and $N = (n + 1)$ *particles*, where segment consists of two neighbouring particles and manages torsion, twist by an angle θ_i and non-elongating distance between the two particles. Segment is unique and every particle belongs to at most 2 segments. We can think

of it as a very rigid link. Region encompass several consecutive segments with their particles and is used for bending and stiffness simulation, whether in terms of twisting or bending. The bigger the region the faster the torsion propagates along the string and in addition to that the string is harder to bend.

To encapsulate entire functionality of our model we use a string element, see Fig. 1. It can contain multiple instances of itself to represent individual torn of pieces of string from the original. This recursive structure allows for easier handling of multiple instances and free user from micromanaging newly created strings. As such it is possible to efficiently change parameters or check for interactions with environment.

Fig. 1: String structure, Particle, Segment and Region. Region can undergo global transformations translation and rotation depended on w_{CSM} vector.

3.2. Particle

As smallest element we use particles to store information about position, velocity and forces that are applied on string. We made it as only element we allow user to directly interact with, as it is easiest to select through interface and offers way to find unique fit for other elements. User can apply twist and exert force. It is always center for sphere for collision detection.

3.3. Segment

Segments consist of two adjacent particles. They are used to handle twist, stress and strain in given part of string. Length of segment define the distance between particle and can change according to stress-strain curve. During collision detection, depending on pre-set parameter, multiple (even overlapping) spheres can be positions between its particles in regular intervals.

3.4. Region

Region is centered on the particle and several of its neighbours to both sides. It is defined by positive half-width and as such consists at least of 3 particles and 2 segments. It is mainly used to regulate stiffness of string. Wider it is, faster changes in twist spread and string bends less.

4. SIMULATION

We will now derive the new strain-based position constraints such as the traditional distance constraints or positional constraints. Let us assume we have system of N particles with positions $\mathbf{x}_i$, velocities $\mathbf{v}_i$ and inverse masses $\mathbf{w}_i$. For the sake of completeness, we briefly show the basic concepts of our simulation loop based on PBD:

```

Result: Update of  $\mathbf{x}_i, \mathbf{v}_i$ 
initialize  $\mathbf{x}_i, \mathbf{v}_i$ ;
while simulation step do
     $\mathbf{v}_i \leftarrow \mathbf{v}_i + \Delta t \mathbf{f}_i$ ;
     $\mathbf{p}_i \leftarrow \mathbf{x}_i + \Delta t \mathbf{v}_i$ ;
     $\mathbf{p}_i \leftarrow \text{solve}(C(\mathbf{p}_1, \dots, \mathbf{p}_N) = 0)$ ;
     $\mathbf{p}_i \leftarrow \text{solve}(\text{CollisionConstraint}(\mathbf{p}_1, \dots, \mathbf{p}_N))$ ;
     $\mathbf{v}_i \leftarrow (\mathbf{p}_i - \mathbf{x}_i) \mathbf{x}_i + \mathbf{v}_i / \Delta t$ ;
     $\mathbf{v}_i \leftarrow \text{Damping}(\mathbf{v}_1, \dots, \mathbf{v}_N)$ ;
     $\mathbf{x}_i \leftarrow \mathbf{p}_i$ ;
end

```

To achieve plausible behaviour of string we calculate forces, $\mathbf{f}_i$, affecting individual particles. Firstly, external forces are taken into account, such as gravity applied directly on string or any interacting object. Secondly, any direct effect caused by user through interface and lastly, inner forces caused by shape matching, twisting and flicking.

The positions are modified by a solver to meet a set of constraints $C(\mathbf{p}_1, \dots, \mathbf{p}_N) = 0$ that is zero when the constraint is satisfied. The solver iterate multiple times over all constraints solving the system of non-linear equations. Local linearization of a single constraint function C by Taylor series results in the positional corrections $\Delta \mathbf{p}_i$ for point $\mathbf{p}_i$ computed as

$$\Delta \mathbf{p}_i = -skw_i \frac{\partial}{\partial \mathbf{p}_i} C(\mathbf{p}_1, \dots, \mathbf{p}_N), \quad (1)$$

where k is the Lagrange multiplier

$$k = \frac{C(\mathbf{p}_1, \dots, \mathbf{p}_N)}{\sum_j w_j \left| \frac{\partial}{\partial \mathbf{p}_i} C(\mathbf{p}_1, \dots, \mathbf{p}_N) \right|^2} \quad (2)$$

and $k \in [0, 1]$ is the stiffness parameter.

4.1. Damping

For damping, we use a general approach that specifically damps the relative velocities with respect to a constraint C as

$$\mathbf{v}_i \leftarrow \mathbf{v}_i - k \left(\sum_{j=1}^N \mathbf{v}_j^T \mathbf{n}_j \right) \mathbf{n}_i, \quad (3)$$

where $\mathbf{n}_i = \Delta \mathbf{p}_i / |\Delta \mathbf{p}_i|$. Note that the summation term cancels out when the mode itself does not change.

Fig. 2: Region translation and rotation. x_i^0 is the original position, x_i is the position updated by external and constraint forces and g_i is the goal position of particle i

4.2. Positional Constraint within Region

The particles are independently moved by external forces and constraints and then an optimal rigid transformation (i.e., rotation and translation) of each region is computed, refer Fig. 2. The rigid transformation of a particle position is called a goal position. Position of each particle is calculated as multiple goal positions weighted in the overlapping regions by particle per-region mass $\tilde{m}_i = \frac{m_i}{N_r}$, where m_i is the mass of particle i and N_r is the total number of regions that the particle belongs to.

For the given set of original points x_i^0 and the set of deformed points x_i , at every step of simulation, for every region, rotational and translational matrix is calculated, which is then used to decide the goal positions, g_i , of particles of that region. Final goal position of particle is a weighted sum of goal positions in regions it belongs to, as

described in Eq. 4. Finally, each particle position is updated toward the goal position.

$$g_i = \sum_{r \in R_i} [R_r(x_i^0 - x_{cm,r}^0) + x_{cm,r}]. \quad (4)$$

Change of region is defined by three variables: $x_{cm,r}^0$ is center of mass for region at the beginning of animation step from original points, $x_{cm,r}$ is center of mass for region after particles were moved, R_r is rotational matrix describing by what degree has region rotated.

Translational matrix can be calculated as simply as a difference between centers of mass, which are average of masses of particles. Approach for calculating rotation matrix can be found in [Müller et al. 2005]. Unlike in [Müller et al. 2005] our shape is 1-D string, therefore we can use Singular Value Decomposition (SVD) of weighted covariance matrix

$$A = \left(\sum_i \tilde{m}_i \mathbf{p}_i \mathbf{q}_i^T \right) \left(\sum_i \tilde{m}_i \mathbf{q}_i \mathbf{q}_i^T \right)^{-1}, \quad (5)$$

where

$$\begin{aligned} \mathbf{q}_i &= x_i^0 - x_{cm,r}^0, \\ \mathbf{p}_i &= x_i - x_{cm,r}, \end{aligned} \quad (6)$$

to obtain the rotation matrix R_r . That computes optimal rigid transformation between two sets of 3D points.

$$[U, S, V] = SVD(A), \quad (7)$$

two of the output matrices are used to calculate the rotation matrix

$$R = VU^T. \quad (8)$$

4.2.1. Goal position calculation. In order to implement the positional changes of particles, regions play the primary role. For every region we compute optimal rigid transformation (i.e. rotation and translation). These are represented by rotation and translation matrix Eq. 8. We keep two sets of particles:

- First set are particles in stable state that are result of last iteration of model and has been most likely displayed to user.
- Second set consists of particles affected by external forces and moved to new position without any constraint.

```

for all particles  $i$  do
  initialize original position  $x_i^0, x_i \leftarrow x_i^0$ 
end for
loop
  for all particles  $i$  do
     $f_{i,ext} \leftarrow \text{ComputeCollisionForce}() + f_{gravity}$ 
  end for
  for all particles  $i$  do
     $v_i \leftarrow v_i + dt \frac{f_i}{m_i}$ 
     $x_i \leftarrow x_i + dt v_i$ 
  end for
  for all chain region  $R_i$  do
     $x_{cm} \leftarrow \text{ComputeOptimalTranslation}()$ 
     $R \leftarrow \text{ComputeOptimalRotation}()$ 
  end for
  for all particles  $i$  do
     $g_i \leftarrow \text{ComputeGoalPosition}()$ 
  end for
  for all particles  $i$  do
     $g_i \leftarrow \text{StrainLimiting}()$ 
     $x_i \leftarrow g_i$ 
     $v_i \leftarrow v_i + \frac{g_i - x_i}{dt}$ 
  end for
end loop

```

Algorithm 1: Chain shape matching algorithm.

After we compute new goal positions for every region, we calculate goal position for each particle which is an average of per region goal positions of given particle, see Eq. 4. At this point we can calculate force necessary to move particles of second set to goal position:

$$f_i(t + dt) = \frac{g_i(t) - x_i(t)}{dt}. \quad (9)$$

Because of possible collision it is might be necessary to run adjust position several times, so any changes are stored in second set. Only when everything finishes without collision, particles of first set are updated. It is also good practice of not discarding last stable state, in case a new frame of animation would need to be produced before we are finished. Following Alg. 1 should clarify the process and the order of calculations.

4.3. Twisting Constrains

According to the shape of the string, in the initial state, an initial twisting angle, θ_i^0 , is assigned to each segment, $\mathbf{e}_i$, see Fig. 3. However, the point of twisting is always a particle and as such, the incurred twist is transferred to one or two adjacent segment without change in magnitude.

Fig. 3: Material frame the curvature binormal $\mathbf{b}_i$, the heading direction $\mathbf{e}_i$ and the normal vector $\mathbf{n}_i$. Twist propagation along the string depends on the size of region k while the amount of propagated twist depends on twisting stiffness β .

Model allows for a segment to be locked in, to simulate it being fixed and maintaining constant degree of twist. In such case, twisting effect can propagate to its neighbouring segments but the locked segments twist would remain constant until released and twist can't propagate past it.

Segments twisted by an external force, attempt to propagate twist towards the neighbouring segments in effort to even out. Size of neighbourhood influenced in every iteration depends on the size of region while the amount of propagated twist depends on predetermined twisting stiffness β .

The increment of the twisting angle of the segment is propagated to the next segments to minimize the elastic energy in the string. In other words, the string tries to minimize the twisting angles between each connected segment. Similarly, to position calculation, twist of a region, $\Delta\Theta_k^{region}$, is an weighted average of the twisting angle increment [Rungjiratananon et al. 2012], $\Delta\theta_i = \theta_i - \theta_i^0$, of the segments in the region k weighted by mass m_i , see Eq. 10. Where S_k is a set of segments within region k . Finally, the twist of segment, $\theta_i = \Delta\Theta_i^{segment}$, is a weighted sum of regional twists, containing the segment, see Eq. 11. Where R is the set of regions to which segment i belongs to.

$$\Delta\Theta_k^{region} = \frac{\sum_{j \in S_k} [m_j \Delta\theta_j]}{\sum_{j \in S_k} [m_j]}, \quad (10)$$

$$\Delta\Theta_i^{segment} = \sum_{k \in R} [\Delta\Theta_k^{region}]. \quad (11)$$

Relative twist of the segment exerts a force on a particle that is treated as external force and scales with twisting stiffness and relative twist of segment compared to its neighbours derived from elastic energy equation [Bergou et al. 2008] as follows:

$$\begin{aligned} \mathbf{f}_i^{twist} &= \frac{\beta}{L} (\theta_{i+1} - 2\theta_i + \theta_{i-1}) \left(\frac{-\kappa\mathbf{b}_{i+1} - \kappa\mathbf{b}_{i-1}}{2l} \right), \\ \kappa\mathbf{b}_i &= 2 \frac{\mathbf{e}_{i-1} \times \mathbf{e}_i}{|\mathbf{e}_{i-1}| |\mathbf{e}_i| + \mathbf{e}_{i-1} \cdot \mathbf{e}_i}, \end{aligned} \quad (12)$$

where $\kappa\mathbf{b}_i$ is the curvature binormal, $\mathbf{e}_i$ is the segment vector, l is the length of the segment, β is the twisting stiffness of the string, and L is the total length of the string.

Heading of segments is used to find the curvature binormal, which describes the correct heading of twisting force. Its heading does change with direction of twisting. Twisting force, $\mathbf{f}_i^{twist}$, is created when there is a difference in twist between 3 neighbouring segments. Afterwards, the calculated force is distributed among two particles of central segment. As the entire equation is build around the difference of twist and the heading of neighbouring segments, in case of reaching the end of string or fixed point, we create temporary segment with identical heading and twist as the segment with fixed point. User control of constant β allows a user define how much should the string knot.

4.4. Strain Limiting Constrain

To make sure the simulated string remains non-elongating, strain limiting algorithm [Selle et al. 2008] needs to be used. Instead of using the large Young's modulus, which can lead to numerical instability, position based constraints are often imposed so that the length of each segment does not exceed a certain threshold L_{max} [Baraff and Witkin 1998; Ďurikovič 2002; Goldenthal et al. 2007]. There are several heuristics approaches possible: Shrink each segment after shape matching; Adjust near tip particle in the propagation direction.

The basic idea is to change the particle position based on last moved direction to

keep the maximal length of segment, see Fig. 4. Although it works for a single fixed side and as such can finish iteratively in single pass of all particles within the string, this can be extended to 2 fixed ends or multiple fixed ends by subdividing the string and simulating the parts independently. Our strain limiting with 2 fixed ends can reach stable state in several iterations while switching between starting and ending points. Unfortunately, string with 2 fixed points can iterate without stable result, when string is too short to avoid the obstacle. In this case the only option is to allow the string to tear itself at point of deepest conflict.

Fig. 4: Vectors of strain limiting. Blue dots are original particle positions and the green dots are new positions after strain limiting.

There can be a special case of string having no fixed particle. It could occur during tearing when free part of string would simply fall off. In such a situation, strain limiting automatically behaves as if the central particle is fixed so in such case strain limiting would be individually enacted on left and right part of string to limit movement of string from its center of mass.

We believe, that by adding collision detection directly into strain limiting we can significantly reduce number of iterations necessary. Important advantage is the cascading approach of strain limiting that moves onto next particle only after the previous was stabilized, see Fig. 5.

4.5. Collision Handling Constrain

To simulate collisions of structure with 1-D string, sphere-sphere collisions are ideal because of simplicity of calculation and easy implementation on GPU. Every particle is a center for collision sphere. Multiple spheres can be regularly spread out over the length of segment, their number and size is adjustable while spheres on the segment can overlap.

Fig. 5: Vectors of strain limiting with obstacle. Blue dots are original particle positions and the green dots are new positions after collision detection followed by strain limiting .

5. STRESS AND STRAIN HANDLING

In term of managing stress and strain of string we approach it as follows:

- (1) After the position of string is calculated, we estimate stress applied on each segment between affected particle and nearest neighbor particles.
- (2) Secondly, when every segment has calculated its strain value, we use stress-strain curves to calculate the stretching and elongation that given segment can be exposed to. It is also possible that strain would reach too high value (also defined by stress-strain curve) and it would tear itself into two parts.
- (3) When new length of segments is finally calculated, Eq. 13, strain limiting algorithm is used to shift all particles that aren't affixed in such way to keep up the reasonable shape and correct heading of string.

$$L_{segment} = L_0 + L_{elongation} + L_{stretch}. \quad (13)$$

5.1. Stress-Strain curve

To get a proper idea of what stress-strain curves actually are, it is important to understand our perception of terms stress and strain.

—*Stress* is representative of amount of force exerted on given segment. Whether it is stretching force along the main axis of string or torsional forces from change in twist. Regardless of attributes of strings, if the same force is applied or relative twist of neighbouring segments is the same, stress will also be the same.

—*Strain* is the physical representation, or effect, of strain. It describes the stretching of segment and is dependant on specific attributes of string, mainly material.

Stress-strain curve is than our way of transforming abstract value of stress, independent of actual properties of string, into a specific effect. It describes how

much of the stretching is only temporary and will disappear the moment strain lessens and how big stretching will result in permanent elongation. The point between them is referred to as *yield point* and reaching it means segment can no longer stretch and any additional changes are part of permanent elongation.

Perhaps, for us the most important point of stress-strain curve is *rupture point* signalling greatest stress and strain the segment can sustain without breaking. Tensile stress-strain curve is measured by linear elongation and the torsional stress-strain curve is measured by torsion move. Data can be freely downloaded for common materials.

5.2. Stress estimation

Estimation of stress is processed after all forces, both external and internal are calculated but before strain limiting is applied. To avoid accidental tearing a delay is used and the excessive external force is distributed as stress only after it is verified that particle can no longer move in the direction of force. We estimate stress as proportional to distance between segment particles beyond the original length of segment. Because direction of the flicking force is calculated from vector of supplementary particle describing applied force and selected particle of string that is affected. As such we only need to store the magnitude of flicking force.

Strain is transformed into force and applied to particles only if:

- string tears,
- string is released.

In both cases, direction of force is determined by last particle in string (or particle that was held by user) is *head* or *tail*. Function initializing this force iterate over all particles until it reaches a *fixed* particle and always spreads from particle (or 2 in case of tearing).

5.3. Tearing and Flicking

As the model simulates strings that are unstretchable, at most times some of the force that is being applied to model should not be visually present. This excessive force is being distributed among the segments and if at any point exceeds the yield point, it permanently elongates the segments and decreases rupture point. If strain reaches rupture point, segment ceases to exist and strings separates. Because after every iteration the strain is applied as inverted force to particles, if the external force no longer influences the string (or it ruptures), it causes a flicking effect, see Fig. 6.

Tensile stress and strain values can be computed by estimating the tensions in the string. To derive the tensions, we also consider the particle positions computed without strain limiting.

We can handle the behaviour of flicking by using the tension. When an inextensible string is pulled and released or torn apart, the applied stress vanishes, but the tensile strain of the segment from the elongated length remains. The bouncing back force could be computed from an internal tensile stress translated from the tensile strain by referencing the stress-strain curve. However, with our tension estimation technique, we can directly use the tension as the bouncing back force.

Fig. 6: Flicking of string with both ends fixed.

6. RESULTS

We tested our application on Alienware desktop X51 equipped with Intel Core4 i7-4770 with processor clock speed at 3,40GHz. It is fitted with NVidia GeForce GTX 960Ti OEM graphics card with 32 unified stream-processors (run both vertex and fragment shaders) and runs Windows Vista Service Pack 2 operating system. Any timing measurements performed relate to this setup.

6.1. Speed performance

Table I shows results of speed experiments with different number of particles and region widths. As a control sample, we use frames 20-50, because time of step also depends on size of the portion of string in relative motion. As at the begging sizable part of string is falling without restriction and after 50th frame some parts of string with narrow regions are already static.

As can be seen our from Tab. I, our expectation of region half-width on speed of our algorithm is significantly smaller than the effect of number of particles and could be limited even further with more efficient calls and data storage and retrieval.

Table I: Performance of our implementation.

Half-width of Regions	Number of Particles		
	<i>1I</i>	<i>2I</i>	<i>4I</i>
1	0.68 ^a	0.184	0.615
4	0.042 ^a	0.219	0.686
8	0.001 ^a	0.238	0.872

^aMeasured as [ms] per frame.

From our test we can infer that the only parameter definitely affecting speed of computation is number of particles and segment spheres used in collision detection. From practical perspective, because particles also act as spheres for collision detection, decreasing number of particles and increasing number of segment spheres would positively increase speed as segments spheres are used solely for collision detection. And therefore number of particles is main identifier of algorithmic speed.

6.2. Evaluation

We have created a model that simulates behavior of string with adjustable levels of stretching and elongation. It is capable of propagating twist and computing forces resulting from twisting and can lead to knotting.

When force is applied on particle to which particle can't respond because of constraints, stress-strain curve is used to calculate stretching and elongation. In case stress surpasses given threshold string ruptures. Afterwards flicking comes into effect where stress is converted to force and applied. Flicking effect may be also initiated if affecting force disappears.

Behaviour and movement of string is adjustable through several parameters. Chief of all are:

- number of particles,
- half-width of regions,
- length of segments,
- stress-strain curve,
- number of segment spheres.

With exception of *number of particles* which influences general precision of model, each of these parameters influences specific behaviour of string.

Half-width of regions is used to define stiffness of string. It should however be always set with respect to *number of particles* as too narrow or wide regions are harmful to stable model.

Fig. 7: Flicking of string with one end fixed. String stiffness is determined by half-width of regions as 12, 8, 4, and 1 from left in columns. The columns show the simulation frames at time 20, 40, and 60.

Segment length is fairly self-explanatory and determines total length of string. It is linked with *stress-strain curve* because stretch and elongation are represented as fraction of original *segment length*.

Finally, *number of segment spheres* is used to fine-tune precision of collision detection. Fewer spheres result in bumpier shape used for collision detection but increase speed of calculations.

7. CONCLUSION

This paper described a method for animation of string-like objects, being able to reproduce realistically effects like twisting, tearing and flicking. Ultimately, as our string is semi-realistic model, we will need to tune the attributes and we hope to be as closely with reality as we can. We have chosen several criteria to validate correctness of our model. Correctness of our model is measured by: Fidelity of motion, Reaction to outside forces, Twisting and forces generated by it, Tearing from twisting or straining, Flicking effect cause by release of strain.

REFERENCES

- BARAFF, D. AND WITKIN, A. 1998. Large steps in cloth simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '98. ACM, New York, NY, USA, 43–54.
- BERGOU, M., WARDETSKY, M., ROBINSON, S., AUDOLY, B., AND GRINSUN, E. 2008. Discrete elastic rods. *ACM Trans. Graph.* 27, 3 (aug), 63:1–63:12.
- BERTAILES, F., AUDOLY, B., CANI, M.-P., QUERLEUX, B., LEROY, F., AND LÉVÊQUE, J.-L. 2006. Super-helices for predicting the dynamics of natural hair. In *ACM SIGGRAPH 2006 Papers*. SIGGRAPH '06. ACM, New York, NY, USA, 1180–1187.
- BRIDSON, R., MARINO, S., AND FEDKIW, R. 2005. Simulation of clothing with folds and wrinkles. In *ACM SIGGRAPH 2005 Courses*. SIGGRAPH '05. ACM, New York, NY, USA.
- GOLDENTHAL, R., HARMON, D., FATTAL, R., BERCOVIER, M., AND GRINSUN, E. 2007. Efficient simulation of inextensible cloth. *ACM Trans. Graph.* 26, 3 (jul), 49.
- GOYAL, S., PERKINS, N., AND LEE, C. L. 2008. Non-linear dynamic intertwining of rods with self-contact. *International Journal of Non-Linear Mechanics* 43, 1, 65 – 73.
- HADAP, S. 2006. Oriented strands: Dynamics of stiff multi-body system. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '06. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 91–100.
- HAIR, D. C. 1991. Legalese: A legal argumentation tool. *SIGCHI Bull.* 23, 1 (Jan.), 71–74.
- HERNANDEZ, F., CIRIO, G., PEREZ, A. G., AND OTADUY, M. A. 2013. Anisotropic strain limiting. In *Proc. of Congreso Espanol de Informatica Grafica*.
- INSTITUT, X. P. AND PROVOT, X. 1996. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *In Graphics Interface*. 147–154.
- IRVING, G., SCHROEDER, C., AND FEDKIW, R. 2007. Volume conserving finite element simulations of deformable models. *ACM Trans. Graph.* 26, 3 (jul).
- K., P. D. 2002. Strands: Interactive simulation of thin solids using cosserat models. *Computer Graphics Forum* 21, 3, 347–352.
- MÜLLER, M., CHENTANEZ, N., KIM, T.-Y., AND MACKLIN, M. 2014. Strain based dynamics. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '14. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 149–157.
- MÜLLER, M., HEIDELBERGER, B., HENNIX, M., AND RATCLIFF, J. 2007. Position based dynamics. *J. Vis. Comun. Image Represent.* 18, 2 (apr), 109–118.
- MÜLLER, M., HEIDELBERGER, B., TESCHNER, M., AND GROSS, M. 2005. Meshless deformations based on shape matching. In *ACM SIGGRAPH 2005 Papers*. SIGGRAPH '05. ACM, New York, NY, USA, 471–478.
- PEREZ, A. G., CIRIO, G., HERNANDEZ, F., GARRE, C., AND OTADUY, M. A. 2013. Strain limiting for soft finger contact simulation. In *2013 World Haptics Conference (WHC)*. 79–84.
- PICINBONO, G., DELINGETTE, H., AND AYACHE, N. 2003. Non-linear anisotropic elasticity for real-time surgery simulation. *Graph. Models* 65, 5 (sep), 305–321.

- RIVERS, A. R. AND JAMES, D. L. 2007. Fastlsm: Fast lattice shape matching for robust real-time deformation. In *ACM SIGGRAPH 2007 Papers*. SIGGRAPH '07. ACM, New York, NY, USA.
- RUNGJIRATANANON, W., KANAMORI, Y., METAAPHANON, N., BANDO, Y., CHEN, B.-Y., AND NISHITA, T. 2012. Animating strings with twisting, tearing and flicking effects. *COMPUTER ANIMATION AND VIRTUAL WORLDS* 18, 2 (mar), 113–124.
- RUNGJIRATANANON, W., KANAMORI, Y., AND NISHITA, T. 2010. Elastic rod simulation by chain shape matching with twisting effect. In *ACM SIGGRAPH ASIA 2010 Sketches*. SA '10. ACM, New York, NY, USA, 27:1–27:2.
- SELLE, A., LENTINE, M., AND FEDKIW, R. 2008. A mass spring model for hair simulation. *ACM Transactions on Graphics* 27, 3 (aug), 64:1–64:11.
- SPILLMANN, J. AND TESCHNER, M. 2007. Corde: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '07. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 63–72.
- VAN DER HEIJDEN, G. AND THOMPSON, J. 2000. Helical and localised buckling in twisted rods: A unified analysis of the symmetric case. *Nonlinear Dynamics* 21, 1 (Jan), 71–99.
- ĎURIKOVIČ, R. 2002. Animation of soap bubble dynamics, cluster formation and collision. *Computer Graphics Forum* 20, 3, 67–76.
- WARD, K., BERTAILS, F., Y. KIM, T., MARSCHNER, S. R., P. CANI, M., AND LIN, M. C. 2007. A survey on hair modeling: Styling, simulation, and rendering. *IEEE Transactions on Visualization and Computer Graphics* 13, 2 (March), 213–234.

Roman Ďurikovič
Faculty of Mathematics, Physics and Informatics,
Comenius University,
842 48 Bratislava, Slovak Republic
email: roman.durikovic@fmph.uniba.sk

Erich Siebenstich
Mathematics, Physics and Informatics,
Comenius University,
842 48 Bratislava, Slovak Republic