Cords: Interactive Modeling of 3D Curves with Physics-Like Properties

Patrick Coleman Karan Singh*

University of Toronto

Figure 1: Use of Cords in the animated film *Ryan*

Abstract

A large body of research addresses the representation, construction and interactive control of 3D curves for various applications. Interactively controlling curves with empirical physical attributes of stiffness and elasticity that bend and wrap around 3D scene geometry is a common and challenging problem. In this paper we address this problem with a novel 3D curve called a *cord*. A cord is created and subsequently controlled in real-time by defining a secondary curve called the guide curve. The guide curve represents the general 3D path that the user wishes the cord to follow while bending and wrapping around scene geometry. Cords have attributes of length, stiffness, and elasticity that allow them to empirically capture a range of behavior from wire to string to rubber bands. Cords that make contact with geometry have a representation in the 2D parameter space of scene objects and can be used not only as geometry for modeling and animation but as 2D parametric strokes for rendering. We present techniques for the creation and interactive control of cords and show examples illustrating their creative use within the animation system *Maya*.

1 Introduction

Curves are a quintessential Computer Graphics primitive. They define features for object modeling, motion paths for animation, and artistic strokes for rendering.

Mathematical representations of 3D curves with various geometric properties have been a subject of study for many decades [Farin 2001]. Research in this area has largely focused on the design of parametric polynomial curves that are defined using a set of geometric constraints. Parametric curves represented by a set of control points, such as Bezier curves and NURBS, are used to construct continuous, piecewise polynomial curves that provide good control over geometric attributes of tangency and curvature. More recently, piecewise linear curves [Barzel 1997; Pai 2002; Balakrishnan et al. 1999; Grossman et al. 2003] controlled using higher level interaction techniques and continuous curves defined by subdivision rules [Chaiken 1974; DeRose et al. 1998] are being used to define 3D curve shapes. Conversion among different curve representations is possible in a robust and efficient manner as a result of research on curve fitting [Farin 2001].

While general 3D curve creation and interactive control is a mature science with widespread commercial use, there is a class of common problems involving 3D curves that are difficult to solve with these representations and techniques. These problems, described below, motivate *cords*, which are a new 3D curve primitive that models curves with physics-like properties that bend and wrap around 3D geometry.

1.1 Motivation

Curves are often used to represent physical objects that are tubular in shape (see Figure 2a). They are also basic construction primitives for most surface-modeling applications. Current interactive curve-modeling techniques, barring a few approaches [Fiume 1995; Barzel 1997; Pai 2002], are not well-suited to the modeling curves like cords. In animation, 3D Curves model the temporal trajectories of scene elements. Motion paths that bend around scene elements are important to goal directed motion and path planning [Featherstone 1987; Latomb 1991]. Motion paths also require the ability to define not just the position but the orientation of scene elements animated along the curve. Curves also define brushstrokes for artistic expression on 2D surface manifolds. While there is no physical equivalent to the notion of a 3D stroke, a strong comparison can be made to the use of wires in 3D sculpting (see Figure 2b). 3D curves that conform to geometry also manifest themselves implicitly in a large body of art as evidenced by Eschers use of 3D curves as negative space in Figure 2c. 3D strokes as applied to rendering often require a 2D parameterization of space around the curve beyond the typical Frenet frame [Bloomenthal 1990] formulation.

Our work has the same motivations as Barzel [Barzel 1997] and was driven by the demands of an animated film. As seen in Figure 1 and the video accompanying this paper, the need arose to control

*e-mail: {patrick | karan}@dgp.toronto.edu

(a) 3D curve construction of shoe-lace.

(b) Wire-sculpture (J. Peterson), ©2002 Dover Abrams.

(c) Bond of union (M.C. Escher), ©2002 Cordon Art B.V.-Baarn-Holland.

Figure 2: Real world primitives analogous to cords

with keyframe precision, the growth and motion of a number of cords in a highly dynamic scene, often in an exaggerated manner that would be hard to achieve with physical simulation. Compared to complex objects like clothing that are intractable to animate using any technique but physical simulation [Baraff et al. 2003], curves are sparse geometric primitives. On one hand this makes them easier for an animator to specify but they require greater precision and control since there is less visible geometry to critque. This contrast motivates the design of the user interface to cords.

1.2 Approach

Most stroke based approaches for computer graphics have adapted 2D techniques established in physical media. Comparing our 3D curves to cords of wire, ribbon, or string allows an alternate perspective to curve creation [Balakrishnan et al. 1999]: one of unrolling prefabricated 3D curve segments into a scene and then manipulating them among geometric primitives as one would manipulate wire or ribbon in the real world. We decouple the curve an animator controls, from the curve that represents the 3D cord. We call the control curve a *guide* curve that defines the general path that a 3D cord should follow. The cord itself is then constructed using the guide curve, stiffness and elasticity parameters and constraints based on scene geometry. The guide curve being only a general specification of the path of the cord is now amenable to high-level curve manipulation. The actual algorithm that generates the cord involves a combination of geometric stepping in the spirit of tape drawing [Balakrishnan et al. 1999] and an intersection test with 3D scene geometry. The length of unrolled tape empirically captures stiffness and the length the curve is grown to relative to the length of the guide curve captures elasticity. Intersection testing ensures that the cord bends around scene elements without penetration. When the cord is on the surface of 3D scene geometry, the surface normal and curve tangent define a reference frame that is interpolated in regions of the curve that are unconstrained.

Our stepping approach generates a poly-line approximation of a continuous curve but we prove in section 5 that a cord is indeed a mathematically defined 3D curve primitive that has a continuous definition as the step size tends to zero.

This paper thus contributes a novel 3D curve primitive called a cord, that is defined by a parametric guide curve and 3D geometry, with explicit parameters that control, stiffness, elasticity and length of the curve. The formulation and implementation of cords is elegant (less than 10 lines of code), robust and very efficient for use in complex animation scenes.

1.3 Overview

The rest of this paper is organized as follows: Section 2 presents related work. Section 3, describes our framework for the creation and editing of 3D cords. Section 4 presents the cord generation algorithm in detail. Section 5 provides an analysis of the mathematical properties from the cord algorithm. We conclude in Section 6 with a number of example applications of cords.

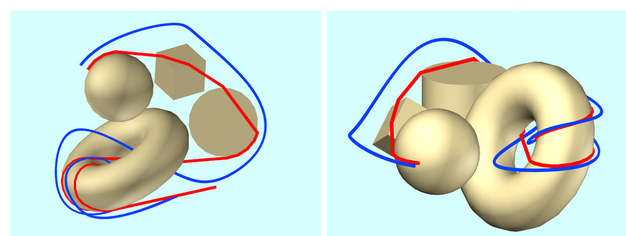


Figure 3: Example guide curve (blue) and cord (red) among 3D geometry

2 Related work

The theory of mathematical representation of curves has existed for centuries [Farin 2001]. Problems related to finding the convex hull of geometry and path planning for collision avoidance studied in Computational Geometry [Preparata and Shamos 1993] and Robotics [Latomb 1991] while not directly addressing the problem in this paper have a similar geometric flavour. Related work to this paper can be broadly classified into two categories. Creation and control of curves for use in physics or physics-like applications and high-level manipulation of 3D curves for modeling and animation.

2.1 Curves with physical properties

The two most relevant peices of work to this are Barzel's work on faking dynamics [Barzel 1997] and Strands [Pai 2002].

Barzel [Barzel 1997] argues, as we do, for the need for curve primitives that possess dynamic properties but are under keyframe control. The paper presents a collection of simple shape primitives that capture the visual behaviour of ropes and springs by overlaying detailed wave-like deformations on a gross overall shape which is along the lines of work on multiresolution curves [Finkelstein and Salesin 1994] and curve alalogies [Hertzmann et al. 2002]. The techniques described in the paper complement our work well and can be used in two ways: using a cord to define the rest shape in Barzel's work or using his resulting curve to defined a fairly inelastic guide curve.

Strands as presented by Pai [Pai 2002] are phyiscally simulated curve primitives. Strands have the benefit of being part of an overall physical simulation and address scenarios where a precisely simulated output curve is desired with minimal user control from the user. The work is tailored to surgical sewing applications, and, is controlled by moving its two end points. Cloth modeling [Baraff et al. 2003], hair simulation [Magnenat-Thalmann et al. 2000] are other practical approaches to physical simulation have produced impressive results[Terzopoulos and Qin 1994; James and Pai 1999; Popović et al. 2000; Chenney and Forsyth 2000; Popović et al. 2003], the objects cords are intended to model would requre interactive manipulation of deformable material simulations, which remains an open problem.

2.2 3D Curve manipulation

Creating and manipulating curve in 3D is far less straightforward than working with planar curves. Placement of multiple 3D curves that often overlap spatially, while maintaining curve length [Fiume 1995] only exacerbates this problem. Arbitrary 3D strokes in space are typically created awkwardly in two steps, by first drawing out the projection of the curve on a plane and then editing it in other views to manipulate it outside the plane. This has been alleviated somewhat by interaction devices such as ShapeTape [Grossman et al. 2003] that allow the direct control of curves in 3D and higher-level curve manipulation techniques [Balakrishnan et al. 1999]. We simplify our problem of creating and editing 3D cords by the introduction of a guide-curve which is a rough indicator of the path taken in the space by the actual cord, leaveing animators free to manipulate and control a simpler 3D curve using existing 3D curve editing techniques.

3 Cord Generation

Cords have been developed with the goal of providing intuitive artistic control within a key-frame animation environment without the need for simulation. The guide curve, an arrbitrary parametric curve, acts as both an interface to specifying the shape of a cord and an abstraction of the desired shape. The intrinsic property of stiffness is comparable to a resistance to bending sharply around geometry in the scene, while the guide curve bounds the space in which a stiff cord can exist. Elasticity in combination with rest length describes the characteristic stretching of rubberband like materials. Stiffness and elasticity are mutually exclusive; the stretching of an elastic cord with stiffness will not increase its rate of bending. While such manipulations are coupled in real-world materials, the independent parameterization produces easily controlled, predictable behavior. Mathematically, elasticity models tendency to change in arc length, and stiffness models tendency to change in curvature and torsion. Figure 4 demonstrates how a cord cord changes shape as its length is increased, its stiffnes increased, and its ability to stretch when elasticity is non-zero.

Cord generation is a deterministic process dependent on the current configuration of the guide curve and geometric primitives with which the cord will interact. The cord properties of length, elasticity, and stiffness affect the generation of the cord, as does a step size which is indicative of the sampling dicretization of the guide curve. The generation of fully elastic cords of zero stiffness is described first, as the incorporation of length and elasticity properties can be considered as refinements to the basic algorithm. These refinements are then presented, as the general cord generation algorithm is developed. The technique is independent of dimension, although it is illustrated in 2D for clarity.

3.1 Fully Elastic, Non-Stiff Cords

Fully elastic cords with no stiffness are analagous to rubberbands that can be stretched to infinite length, but with zero rest length. Such cords will stretch to meet the end points of the guide curve with no bending resistance, similar to how string wraps around sharp corners, but unlike a geodesic in that the shortest path is in general not taken. In relation to the guide curve, the cord appears tied to the two endpoints. The shape of the cord approximately follows the path of the guide curve around complex geometry.

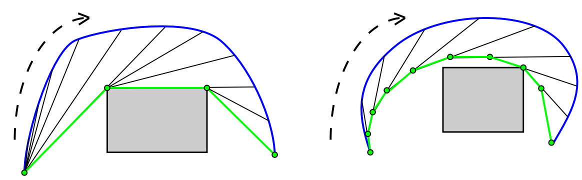
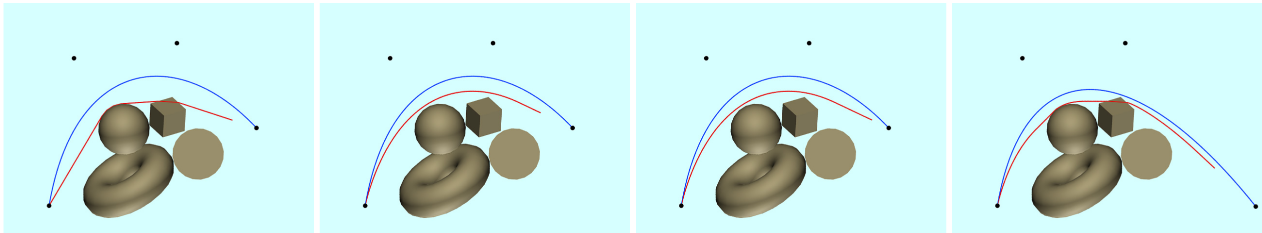


Figure 5: Geometric construction of nonstiff and stiff strands

The algorithm used to generate such a cord is conceptually similar to the computation of a two-dimensional convex hull [Preparata and Shamos 1993]. A regular sampling of the guide curve parameterization is precomputed, ideally a regular sampling along an arc-length parametrization. The cord is generated along a particular direction relative to the guide curve, having an initial point coincident with an endpoint of the guide curve. Rays are repeatedly cast toward samples of the guide curve from the most recently added cord point (Figure 5a). When no geometry is intersected by the ray in the region convex relative to the guide curve, the algorithm moves on to the next guide curve sample using the same initial point for ray casts. When a ray intersection is detected, the current and previous sample points are used in conjunction with bisection along the curve to isolate a grazing intersection with the geometry. This intersection point is appended to the cord, and sampling of the guide curve resumes using the previous sample point. This process continues until the guide curve samples are exhausted. Pseudo-code is presented below.

```
i = 1;
cord.append(guide.start());
while (not done) {
    tTest = guideSamples[i];
  ray.direction = guide.pointAt(tTest) - cord.last();
  ray.origin = cord.last();
  if (ray.intersectGeometry()) {
      cord.append(refine( tLast, tTest, guide));
  } else {
      i = i + 1;
      if (i == guideSamples.size()) {
          cord.append(
              guide.pointAt(guideSamples[i - 1]));
          done = true;
      }
  }
}
```



(a) Fixed length cord (b) Increased stiffness (c) Increasing elasticity (d) Elastic stretching

Figure 4: Effect of editing cord properties

The only requirement of the algorithm is the ability to perform ray intersection tests with the geometric primitives. In addition, it is assumed that the guide curve does not intersect the geometric primitives, as it conceptually models how the cord should wrap around such primitives. This behavior is not well defined for intersecting guide curves.

3.2 Incorporating Stiffness

Intuitively, stiffness models a cord's resistence to bending. Cord stiffness is represented as scaler value in the range [0, 1], such that a cord with no stiffness is equivalent to the cord generated with algorithm as presented above, and a cord with full stiffness is equivalent to the guide curve. Within this model the guide curve acts acts as an outer bound on where in space the guide curve can exist. This property can be especially valuable to a user, as stiff cord behavior can be constrained without incorporating expensive global computations into the algorithm.

To generate a stiff cord, the algorithm casts rays using the same sampling scheme as described above. However, the algorithm is modified such that each non-intersecting ray contributes a segment to the cord equal to the proportional length of the ray to the guide curve sample, where the proportion is equal to the stiffness value (Figure 5b). When the stiffness is zero, no extra segments are added, and the result is equivalent to the previous algorithm. When the stiffness is one, each point of the cord lies along the guide curve. Modifications to the algorithm are as follows:

```
...
if (i == guideSamples.size()) {
    ...
} else {
    next = ray.origin + ray.direction * stiffness;
    cord.append(next);
}
...
```

Within this framework, a user also has the ability to model sharp turns in a stiff cord as would result from the application of external forces. Introducing a region of high curvature in the guide curve will produce a corresponding region of increased curvature in the cord at a predictable location. Stiffness can also vary along the curve for greater local control.

3.3 Length and General Elasticity

The ability to define a constant length allows cords to model real-world materials which do not stretch. Cord length is modeled relative to the first point; this allows it to be incorporated into the algorithm as an early exit point. As each segment is appended, the total length is tracked. If the addition of a segment causes the cord to be longer than the defined length, it is truncated. If the end of the guide curve is reached before the desired length is accumulated, the final segment is extended such the resulting cord has the desired length.

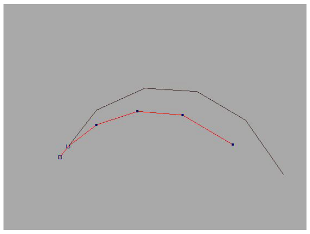
Having defined a fully elastic cord and an inelastic cord (constant length), it is straightforward to model general elasticity as the variation between these two lengths. The length, as described above, is now consired as *rest length*. Elasticity, like stiffness, is a scalar in the range [0, 1]. The fully elastic curve is first computed as above; we refer to its length as the *elastic length*. The elasticity value then defines *actual length* as a linear interpolation between the elastic length and the rest length. If the elastic length is greater than the rest length, and hence the actual length, the computed fully elastic curve is truncated at the actual length. If the rest length is greater, the final segment of the curve is extended beyond the last point of the guide curve along the end tangent of the cord such that the cord has length equal to the interpolated actual length. While this final case has no analogue to a real world manipulation, it maintains cord continuity relative to changes in all parameters. The following is appended to algorithm:

```
...
elasticLength = cord.length();
actualLength = elasticity * elasticLength +
    (1 - elasticity) * restLength;
if (elasticLength > restLength)
    cord.truncate(actualLength);
else
    cord.last() = cord.last() +
        cord.endTangent.normalize() *
        (actualLength - elasticLength)
```

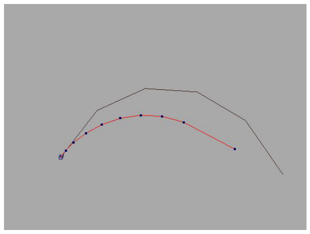
4 Analysis

In this section we mathematically consider geometric cord properties.

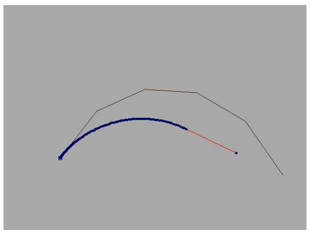
Position: By construction, a cord (if long enough) clearly interpolates the two end points of the guide curve. While attempting to follow the guide curve, the cord wraps around scene geometry. Considered in a 2D plane, the cord traces a convex hull around geometry in the direction of the guide curve. In 3D, the cord similarly wraps around the 3D convex hull while traversing a curve that can be thought of as a geodesic whose locality is controlled by the guide curve. In general, however, the entire cord between the two end points will not mathematically be a geodesic. As the stiffness of a curve increases it lifts off the geometry while maintaining contact at various points until lifting off entirely.



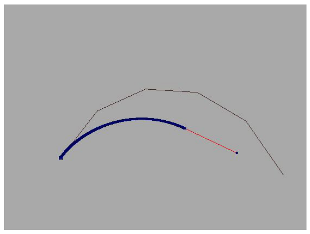
(a) 5 steps



(b) 10 steps



(c) 100 steps



(d) 1000 steps

Figure 6: Varying the number of steps

Tangent: The curve, by construction, is tangent continuous everywhere except at extreme stiffness values. At $stiffness = 0$ the cord has the same continuity as the geometry it conforms to and and is identical to the guide curve at $stiffness = 1$. Elasticity only affects the parameterization and length of the cord and not its shape.

Curvature: The cord attempts to be be a continuous curve with a minimum overall curvature along the curve that satisfies the geometric constraints, while having a maximum allowed curvature at any point along the curve (defined by $stiffness$). We only empirically capture this behavior.

Continuity: The geometric steps of size that models bending energy has been used to inspire the creation of many continuous parametric curves [Farin 2001; Balakrishnan et al. 1999]. While cords of non-zero stiffness are intuitively continuous, we wished to invesitgate their limit behaviour as the step size approaches zero. For stiffness defined as a fixed ratio r of the line segments from the current point to the sample on the guide curve, it is easy to see that as the step size approaches zero, the cord degenerates to the guide curve. We thus considered an adaptive value for r, setting stiffness to be the factor that related r and step size s such that $r = stiffness * s$. This results in curves that are stable with respect to a changing step s and converge to a continuous curve as $s \rightarrow 0$. The proof of convergence requires us to consider point samples p_i along the cord for a guide curve $f(t)$, parametric in t. The cord construction defines a recurrence relation for p_i:
$p_i = p_{i-1} + stiffness * s * (f(i * s) - p_{i-1})$ and $p_0 = f(0)$
This recurrence relation can be obtained as a solution to a differential equation of the guide curve using Euler's method [Braun 1983]. The theorem that Euler's method converges to a solution of the differential equation guarantees that our recursion converges to a continuous curve. The analytic solution of the differential equation will provide an analytic formula of the resulting cord. This analysis of continuity does not account for the cord making contact and bends around geometry.

5 Example Applications and Results

Cords have been implemented as a plugin to the *Maya* modeling and animation system. We now look at two applications of cords, for modeling and rendering, respectively.

5.1 Wide and Thick Cords

Wide cords are a variation of cords for modeling primitives that can be represented as long two-dimensional manifolds, such as flat ribbons or straps. Thick cords can represent objects such as thick tubes or rope. In both cases, the Cord model is extended to have width or thickness, which may vary along the length of the curve, and a local orientation radially about the cord. Orientation is expressed as vector in the normal plane of the curve, which is used to define the surface of a flat cord or the radial orientation of a thick cord. For flat surfaces, this normal vector should be defined such that the cord surface lies flat on the geometric surface. The common definition of a curve normal is ill-suited for this task, as it would generally result in geometric occlusion. Furthermore, it does not have a defined direction for straight sections of a curve. To define this vector, the geometry normal is recorded for all points of contact. Additional points along the cord have this vector defined as the smaller angular interpolation within the local curve normal plane between the two surrounding well defined points, resulting in a continuous surface normal along the cord. This technique is demonstrated in Figure 7a.

In addition, constraining wide chords against locally convex geometry or thick cords against any geometry requires that the intersection test be replaced with a proximity test in the cord generation algorithm. The geometry that these cords produce it fed back into the system to allow such cords to appropriately cross over themselevs. An example of a thick cord with self-overlap is shown in Figure 7b.

5.2 Artistic Primitives

Artistic primitives such as paint strokes or procedural geometric curves can be defined about an underlying curve primitive [Hertzmann et al. 2002]. Existing software packages allow such artistic primitives to be defined on two-dimensional canvases or embedded within parametric 3D surfaces. These technigues can be extremely limiting, as general 3D curves are difficult to manipulate by hand. Cords can be used to easily specify the locations of such curves in space or about arbitrary geometry. Figure 7c demonstrates the use of a chord wrapped around a head to define a CSG modeling primitive to create an image in the spirit of Escher's *Bond of Union* (Figure 2c). From this standpoint, the cord represents an implicit 3D stoke.

Cords have also found extensive use in the production of the animated film *Ryan* (Figure 1). Without the ability to interactively specify a curve that maintains continuity in time with a simple interface, such a sequence would have never been possible given the time constraints of animated production.

6 Conclusion

Cords provide an intuitive, interactive technique for specifying curves with physics-like properties. They can be interactively manipulated in the presence of geometric primitives with behavior akin to string or wire. This approach is favorable to editing traditional 3D curve models by hand, as the detailed interaction with geometry would be intractable to specify. The approach is contiuous in space with respect to manipulations of the guide curve and cord properties, allowing them to be used in a typical animation environment. Finally, cords capture intuitive, physically-based qualities without

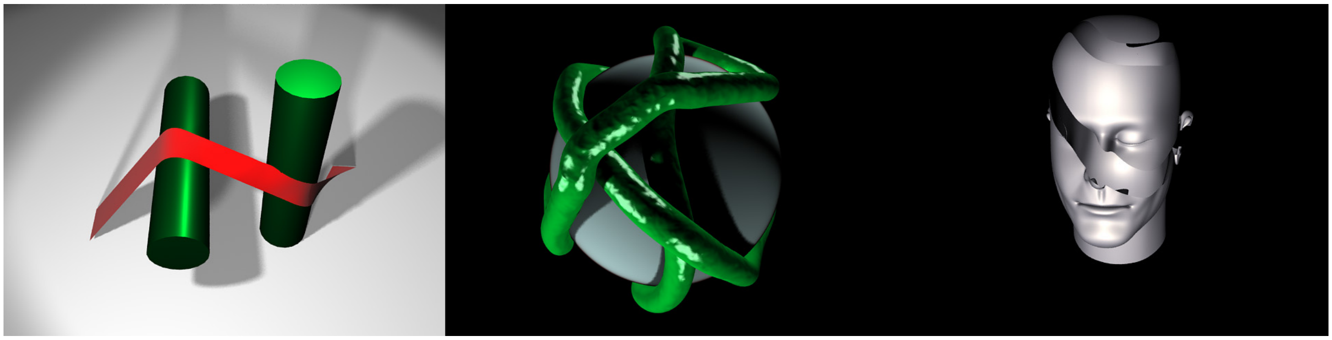


Figure 7: Example Applications of cords

the need for simulation. The use of animated cords for a number of shots in the animated film *Ryan* has further demonstarted their usefulness and viability as a modeling primitive that captures the qualities of real-world curve-like objects. We show that in the limit of infinitely many stpes,the curves generated by the cords algorithm are continuous. Deriving the analytic form of these curves remains as ongoing work.

References

BALAKRISHNAN, R., FITZMAURICE, G., KURTENBACH, G., AND BUXTON, W. 1999. Digital tape drawing. In *Proceedings of the 12th annual ACM symposium on User interface software and technology*, ACM Press, 161–169.

BARAFF, D., WITKIN, A., AND KASS, M. 2003. Untangling cloth. *ACM Trans. Graph. 22*, 3, 862–870.

BARZEL, R. 1997. Faking dynamics of ropes and springs. *IEEE Computer Graphics and Applications 17*, 3, 31–39.

BLOOMENTHAL, J. 1990. Calculation of reference frames along a space curve. *Graphics Gems*, 567–571.

BRAUN, M. 1983. *Differential Equations and their Applications (3rd ed.)*. Springer-Verlag.

CHAIKEN, G. 1974. An algorithm for high speed curve generation. In *Computer Graphics and Image Processing 3*, 346–349.

CHENNEY, S., AND FORSYTH, D. A. 2000. Sampling plausible solutions to multi-body constraint problems. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 219–228.

DEROSE, T., KASS, M., AND TRUONG, T. 1998. Subdivision surfaces in character animation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, ACM Press, 85–94.

FARIN, G. 2001. *Curves and Surfaces in Computer Aided Geometric Design: A Practical Guide*. Morgan Kaufmann.

FEATHERSTONE, R. 1987. *Robot Dynamics Algorithms*. Kluwer.

FINKELSTEIN, A., AND SALESIN, D. H. 1994. Multiresolution curves. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, ACM Press, 261–268.

FIUME, E. 1995. Isometric piecewise polynomial curves. *Computer Graphics Forum*, 47–58.

GROSSMAN, T., BALAKRISHNAN, R., AND SINGH, K. 2003. An interface for creating and manipulating curves using a high degree-of-freedom curve input device. In *Proceedings of the conference on Human factors in computing systems*, ACM Press, 185–192.

HERTZMANN, A., OLIVER, N., CURLESS, B., AND SEITZ, S. M. 2002. Curve analogies. In *Proceedings of the 13th Eurographics Workshop on Rendering*, Eurographics Association, 233–246.

JAMES, D. L., AND PAI, D. K. 1999. Artdefo: accurate real time deformable objects. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 65–72.

LATOMB, J. 1991. *Robot Motion Planning*. Kluwer Academic Publishers.

MAGNENAT-THALMANN, N., HADAP, S., AND KALRA, P. 2000. State of the art in hair simulation. In *International Workshop on Human Modeling and Animation*, Korea Computer Graphics Society, 3–9.

PAI, D. K. 2002. Strands: Interactive simulation of thin solids using cosserat models. In *Proceedings of Eurographics 2002*, Eurographics Association.

POPOVIĆ, J., SEITZ, S. M., ERDMANN, M., POPOVIĆ, Z., AND WITKIN, A. 2000. Interactive manipulation of rigid body simulations. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 209–217.

POPOVIĆ, J., SEITZ, S. M., AND ERDMANN, M. 2003. Motion sketching for control of rigid-body simulations. *ACM Trans. Graph. 22*, 4, 1034–1054.

PREPARATA, F., AND SHAMOS, M. 1993. *Computational Geometry: An Introduction*. Springer.

TERZOPOULOS, D., AND QIN, H. 1994. Dynamic nurbs with geometric constraints for interactive sculpting. *ACM Trans. Graph. 13*, 2, 103–136.