2-3-2014

Numerical Analysis and Computational Topology for Scientific Visualization

Hugh Cassidy
University of Connecticut, hugh.p.cassidy@gmail.com

Follow this and additional works at: https://opencommons.uconn.edu/dissertations

Recommended Citation
https://opencommons.uconn.edu/dissertations/314
In Computer Aided Geometric Design (CAGD) B-splines are frequently used to model complex geometric objects. The spline models are smooth structures but piecewise linear (PL) approximations are typically used to render the spline. Aeronautical, automotive and chemical simulations rely on topological algorithms to provide mathematically correct visualization. Topological changes are of significant interest to domain scientists, where self-intersection is a critical event that is often difficult to detect. This research focuses on algorithms that guarantee the topological integrity of the spline models frequently used in geometric design systems. In particular the focus is on utilizing the properties of subdivision together with ambient isotopy as a measure of topological equivalence.
Numerical Analysis and Computational Topology for Scientific Visualization

Hugh P. Cassidy
B.Sc.(hons) University College Dublin 2001
H.Dip. National University of Ireland, Maynooth 2002
M.Sc. National University of Ireland, Maynooth 2003
M.S. Syracuse University 2007

A Dissertation
Submitted in Partial Fulfilment of the Requirements for the Degree of Doctor of Philosophy at the University of Connecticut 2014
APPROVAL PAGE

Doctor of Philosophy Dissertation

Numerical Analysis and Computational Topology for Scientific Visualization

presented by

Hugh P. Cassidy,

Major Advisor: ________________________________

Thomas J. Peters

Associate Advisor: ________________________________

Alexander Russell

Associate Advisor: ________________________________

Robert McCartney

University of Connecticut

2014

ii
Acknowledgements

First and foremost, I would like to express my deepest gratitude to my advisor, Professor Thomas J. Peters. Throughout my studies Professor Peters has been a terrific teacher and mentor with an infectious enthusiasm for practical mathematics. He went beyond the call of duty on countless occasions and without his guidance this dissertation would not have been possible.

I would also like to thank my associate advisors, Professor Alexander Russell and Professor Robert McCartney for agreeing to be on my dissertation committee and for their thoughtful comments and suggestions.

My thanks also to Dr. Kirk Jordan for his helpful and stimulating collaboration.

Last, but not least, I would like to thank my parents for their unwavering support and encouragement throughout my studies.
## CONTENTS

1. **Introduction** ...................................................... 1
   1.1 Mathematically Precise Visualization ..................... 2
   1.2 Efficient and Correct Representation of B-Splines ....... 4

2. **Background** ................................................... 5
   2.1 Bézier curves ................................................. 5
   2.2 Topological Equivalence .................................... 6

3. **Approximation of Hodograph Curves** .......................... 10
   3.1 Introduction ................................................ 10
   3.2 Subdivision and derivative curves ......................... 12
   3.3 Discrete and continuous derivatives ...................... 21
   3.4 Conclusion .................................................. 25
4. Topology for Dynamic Graphics ..................................... 26

4.1 Introduction ....................................................... 27

4.2 Background, Motivation and Notation .......................... 28

4.2.1 Curves and control polygons ................................. 29

4.2.2 Related Work .................................................. 30

4.3 Graphics Efficiency Experiment ................................. 33

4.4 Notation for perturbation analysis ............................... 36

4.5 Non-differentiable Perturbations ................................. 38

4.5.1 Perturbing a single point ..................................... 39

4.5.2 Perturbing multiple points ................................... 40

4.6 Differentiable Perturbations ..................................... 42

4.6.1 Perturbation strategy .......................................... 42

4.6.2 Perturbing a single point ..................................... 45

4.6.3 Perturbing multiple points ................................... 47

4.7 An Example Predictive Analysis .................................. 48

4.8 Conclusions ....................................................... 51
5. Numerical Techniques for Isotopic Approximation ........................................... 53

5.1 MSD and Tubular Neighborhoods .............................................................. 54

5.2 Newton’s Method ....................................................................................... 56
  5.2.1 Newton Iterations .............................................................................. 56
  5.2.2 Algorithmic Difficulties ..................................................................... 58

5.3 Discrete Approximation Method ............................................................... 60
  5.3.1 Convergence of PL Structures ........................................................... 61
  5.3.2 Vector Perturbations .......................................................................... 63
  5.3.3 Notation ............................................................................................ 67
  5.3.4 Quadratic Approximation .................................................................. 68
  5.3.5 Estimating MSD ............................................................................... 70
  5.3.6 Numerical Approximation .................................................................. 72

6. Conclusion .................................................................................................... 73
LIST OF FIGURES

1.1 Data flow of a molecular dynamics simulation.\textsuperscript{1} .................. 2
1.2 Backbone of a protein molecule. ................................................. 3

2.1 Subdivision of a cubic curve. ....................................................... 6
2.2 Frames of a closed curve animation in $\mathbb{R}^3$. ......................... 7

4.1 Spurious self-intersection in PL structure. ................................. 36
4.2 Zoomed in view of perturbation. .................................................. 36
4.3 Perturbation over a single time step. .......................................... 38
4.4 Type 1 perturbation. ................................................................. 43
4.5 Type 2 perturbation. ................................................................. 44

5.1 Pipe radius depends on curvature and MSD. ............................. 56
5.2 Control polygon and corresponding curve. .................................. 60
5.3 Newton’s method returns a bad estimate of $\sigma$. ....................... 61
5.4 Subsequent subdivisions for approximation. ............................... 62
5.5 Use PL structure to approximate MSD.  

5.6 Triangle formed by $(0,0), \tilde{u}$ and $R(-\phi)\tilde{u}$.
1. INTRODUCTION

Consider the problem posed by the massive quantity (sometimes petabytes) of floating point data generated by a molecular dynamics simulation. The flow of information (depicted in Figure 1.1), starts with simulation input for the molecules and their governing equations. The simulation then runs in a high performance computing (HPC) environment which generates floating point output. Visualization of the data allows domain scientists to quickly view the otherwise unintelligible simulation output.

The floating point output of the simulation is typically dumped to disk and individual time steps are visualized at a later time. These HPC simulations can be costly and time consuming. If inappropriate simulation parameters have been chosen initially, then by the time the domain scientists view the visual output it is too late to alter the parameters. However if the dynamic visualization is provided in real time as the simulation runs then the expert user can guide the simulation and avoid costly mistakes.

For practical use the visualization needs to be mathematically precise, topologically correct and synchronized with the ongoing simulation. The display
should preserve crucial topological characteristics such as the embedding of the molecule and self-intersections.

Traditional graphics techniques can introduce spurious topological changes \cite{3, 24} that exist only in the graphical model that could mislead domain scientists.

\textbf{Fig. 1.1:} Data flow of a molecular dynamics simulation.\textsuperscript{1}

\textbf{1.1 Mathematically Precise Visualization}

The three dimensional structure of large biological molecules is recorded in text files. These files are publicly available from the Protein Data Bank (PDB) \cite{2}. The data in these files are typically obtained by X-ray crystallography or NMR spectroscopy and are submitted by experts from around the world. Geometric and connectivity information is extracted from the given PDB file to form the backbone of the molecule (Figure 1.2). Points of the

\textsuperscript{1} Blue Gene image credit: www.our-picks.com
Molecule backbone image credit: Kevin Marinelli
backbone are then interpolated using a spline curve for a dynamic model of the writhing molecule. The simulation code describes how points on the curve move under changes in critical variables such as temperature, pressure and acidity. The topological integrity of the model can be maintained by constraining the movement of the spline to within particular neighborhoods [15]. These neighborhoods need to be recomputed as the simulation runs. This requires the development of rigorous, robust and tractable methods to compute the neighborhoods. In this thesis such techniques are investigated.

Fig. 1.2: Backbone of a protein molecule.
1.2 Efficient and Correct Representation of B-Splines

During simulation, the molecular movement is implemented by successive perturbations of the spline model. A typical display method for spline curves is to render a PL approximation for each frame that lies within a prescribed tolerance of the curve. Perturbing the graphics implementation can introduce subtle, but significant, topological inconsistencies with the underlying spline model.

This thesis provides efficient perturbation strategies together with specific numerical analyses to guarantee that topological fidelity is preserved.
2. BACKGROUND

We present the curve and topological definitions that are central to this work.

2.1 Bézier curves

Definition 2.1.1. A degree $n$ Bézier curve is defined by $n + 1$ control points, $X = \{p_0, \cdots, p_n\}$. The curve is given by

$$c(t) = \sum_{i=0}^{n} \binom{n}{i} (i - t)^{n-i} p_i, \quad t \in [0, 1],$$

where

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}.$$

Remark 2.1.1. We assume $X \subset \mathbb{R}^3$. The PL structure formed by $p_0, \cdots, p_n$ is called the control polygon of the curve [25].

A subdivision algorithm operates on $X$ to generate two PL curves, each having $n + 1$ vertices, denoted, respectively as $X_L$ and $X_R$, as shown in Figure 2.1. The union $X_L \cup X_R$ is also a control polygon for $c$ but lies closer
to $c$ than the original control polygon. This process can be repeated to obtain a PL graphical approximation that is within a prescribed distance [23] of the curve $c$. This process will be described in more detail in section 5.3.1.

\[ X \]

Fig. 2.1: Subdivision of a cubic curve.

### 2.2 Topological Equivalence

Perhaps the most fundamental measure of topological equivalence is homeomorphism. Loosely, speaking two objects are homeomorphic if one is the continuous image of the other – where bending, stretching and shrinking are allowed but tearing or changes in intersections are not. The precise definition is as follows.
Definition 2.2.1. A homeomorphism is a mapping

\[ f : X \rightarrow Y \]

where \( X \) and \( Y \) are subsets of \( \mathbb{R}^n \) such that

1. \( f \) is bijective, and
2. \( f \) and \( f^{-1} \) are continuous.

If such an \( f \) exists \( X \) and \( Y \) are said to be homeomorphic.

Homeomorphism, however, does not capture the embedding of an object in space. This is a desirable property for the purposes of visualization. Consider the three frames of a simple animation of a closed curve in \( \mathbb{R}^3 \) depicted in Figure 2.2.

![Fig. 2.2: Frames of a closed curve animation in \( \mathbb{R}^3 \).](image-url)
The animation progresses from left to right. Note that there are no self-intersections in the first and last frames, but the second frame has an obvious self-intersection. The first and last frames are homeomorphic to each other but are not homeomorphic to the second frame due to the self-intersection. Observe that we can easily untangle the curve in the first frame to form a loop by pulling the region indicated by the solid triangle out of the plane and then untwisting the curve. However we cannot do this with the last frame without ripping the curve or forcing the curve to pass through itself, i.e. these frames have a different embedding in three dimensional space.

Typically in animation we want to avoid introducing self-intersections between frames. Many detection methods [5, 19] rely upon temporal coherence and operate on a per frame basis in the sense that they perform a geometric test for self-intersection in each frame. Clearly this can lead to problems if time is discretized inappropriately. For example, if we miss the middle frame of our simple animation we would not detect that a self-intersection occurred. In common practice the computer graphics community uses the ‘eyeball metric’ to detect obvious self-intersections missed by automated detection algorithm. In the case of this simple example such a self-intersection is easily observed, however, if we have a model with the level of sophistication illustrated in Figure 1.2 this visual detection is virtually impossible. Often in computer graphics, if we cannot see the artifact this is considered good enough. However, in scientific visualization this is not the case as self-intersections are of critical interest.
Also different embeddings of the same molecular structure can have different chemical properties. We can use topology to discriminate different embeddings, but homeomorphism is clearly not a sufficient. The notion of ambient isotopy does capture the embedding of objects in space and is defined as follows.

**Definition 2.2.2.** Two subspaces, \(X\) and \(Y\), of \(\mathbb{R}^n\) are said to be ambient isotopic if there exists a continuous function \(H : \mathbb{R}^n \times [0, 1] \rightarrow \mathbb{R}^n\) such that

1. \(H(\cdot, 0)\) is the identity on \(\mathbb{R}^n\),
2. \(H(X, 1) = Y\), and
3. \(\forall t \in [0, 1], H(\cdot, t)\) is a homeomorphism.

\(H\) is called an ambient isotopy.

Since the first and last frames of the animation in Figure 2.2 have different embeddings in \(\mathbb{R}^3\) they are not ambient isotopic. Furthermore the parameter \(t\) can be thought of as time, making ambient isotopy particularly suited to animation. It has been shown [15] that neighborhoods can be defined around parametric curves such that a rich class of perturbations of the curve consists of ambient isotopies. While the concept of these neighborhoods is relevant throughout this thesis the precise details will not be required until Chapter 5, where a formal definition is provided.
3. APPROXIMATION OF HODOGRAPH CURVES

The discrete derivative of a Bézier curve is known to converge to the continuous derivative under subdivision [20, 22]. After a sufficient number of subdivisions one can obtain a ‘good’ approximation of the derivative. However, no practical error bound is known, nor is the number of subdivisions required for a satisfactory approximation. Here we derive a quantifiable error bound for the discrete derivative. Also the number of subdivisions required for a prescribed error tolerance is calculated. The cases of uniform and non-uniform subdivision are treated.

3.1 Introduction

Bézier curves are widely used in computer graphics, scientific visualization and finite element approximation. Frequently, a Bézier curve is approximated by the piecewise linear (PL) structure of the control polygon. Knowing how accurately the piecewise linear geometry approximates the smooth geometry can be crucial [4].

It is well known that the discrete derivative of a Bézier curve generated
by repeated subdivision converges to the continuous derivative of the curve [20, 22]. It is also known that the discrete derivative converges at a rate of $O\left(\frac{1}{2^m}\right)$, where $m$ is the number of uniform subdivisions [20]. In other words after numerous subdivisions the discrete derivative can give a ‘good’ approximation of the continuous derivative. This raises the questions of how many subdivisions are required and how accurate is the approximation. Since uniform subdivision does not always provide optimal convergence [23] we examine uniform and non-uniform cases. We establish a relationship between the hodograph of a subdivided curve and the subdivision of a hodograph curve. Using this relationship and the established bound on the distance between a Bézier curve and its control polygon [23] we can determine error bounds for the discrete derivative.

Theorem 3.3.2 expresses how the traditional derivative for the Bézier curve $c$ can be evaluated at the parameter value $t$ by appropriate subdivision. Previous work [20] emphasized commutativity of the operators for discrete differentiation and subdivision, whereas we prove the commutativity of the forward difference operator and uniform subdivision as a means to equate the traditional and discrete derivatives at any given parametric value. Those authors [20] have a similar evaluation formula to the one presented here, but our method of proof appears to be more direct, providing possibly different theoretical insights. In particular, our approach easily includes the generality of non-uniform subdivision which appears to be only implicit otherwise [20].
3.2 Subdivision and derivative curves

It is known that many properties of Bézier curves can be derived without referring to the Bernstein polynomials by use of certain operators that act on the control points of a given curve [9]. We proceed in the spirit of this approach.

Let

\[ X = \{p_0, \ldots, p_n\} \]

denote the set of control points of a degree \( n \) Bézier curve. Define the derivative operator \( \Delta \) by

\[ \Delta X = \{p_1 - p_0, \ldots, p_n - p_{n-1}\}. \]

Note that the control points of the hodograph of a curve with control points \( X \) are given by \( \{n(p_1 - p_0), \ldots, n(p_n - p_{n-1})\} \). For the sake of simplicity, and without consequence, we ignore the constant factor \( n \).

Define \( E^k_i(X) \) to be the \( i^{th} \) point generated in the \( k^{th} \) step of the de Casteljau algorithm applied to the set of control points \( X \). Using this notation the de Casteljau algorithm with subdivision parameter \( \alpha \) can be described as follows.
where $E_i^{k+1}(X) = (1 - \alpha)E_i^k(X) + \alpha E_{i+1}^k(X)$ and $i \in \{1, 2, \ldots, n + 1 - k\}$.

The above triangular matrix can be understood as follows. Two vertically successive entries of the left-most (or first) column are linearly combined to produce the $n$-entries of the second column. Successive columns are produced similarly. The top row and the diagonal contain the control points of the two resultant curves from the subdivision. Subdividing a curve with control points $X$ results in two curves, namely $X_L$ and $X_R$ (see Figure 2.1), with control points

\[ X_L = \{E_1^0(X), E_1^1(X), \ldots, E_1^n(X)\} \quad \text{(i.e. the top row)}, \quad \text{and} \]

\[ X_R = \{E_1^n(X), E_2^{n-1}(X), \ldots, E_{n+1}^0(X)\} \quad \text{(i.e. the main diagonal)}. \]

For each curve resulting from the subdivision, the control points of its hodo-
graph can be found by applying $\Delta$ to the control points of the subdivided curve. So we have

\[
\{E^1_1(X) - E^0_1(X), \ldots, E^n_1(X) - E^{n-1}_1(X)\} \quad \text{and} \\
\{E^{n-1}_2(X) - E^n_1(X), \ldots, E^0_{n+1}(X) - E^1_n(X)\}.
\]

Subdividing the derivative curve is done by applying DeCasteljau’s algorithm on the points of $\Delta X$ which yields

\[
\{E^0_1(\Delta X), \ldots, E^{n-1}_1(\Delta X)\} \quad \text{and} \\
\{E^{n-1}_1(\Delta X), \ldots, E^0_n(\Delta X)\}.
\]

Define

\[
\Delta^k_{E^k_i}(X) = E^{k+1}_i(X) - E^k_i(X).
\]

Notice that $\{\Delta^k_{E^k_i}(X)\}_{k=0}^{n-1}$ is the set of control points for the derivative of the first curve in the subdivision mentioned above.

**Lemma 3.2.1.** If $k \in \{0, 1, \ldots, n - 1\}$ and $i \in \{1, 2, \ldots, n + 1 - k\}$ then

\[
\Delta^k_{E^k_i}(X) = \alpha E^k_i(\Delta X).
\]
Proof:

Proceed by induction of \( k \), showing that the relation holds for all values of \( i \).

Letting \( k = 0 \) we have

\[
\Delta^0_{E_i}(X) = E^1_i(X) - E^0_i(X)
\]

\[
= ((1 - \alpha)E^0_i(X) + \alpha E^0_{i+1}(X)) - E^0_i(X)
\]

\[
= ((1 - \alpha)p_{i-1} + \alpha p_i) - p_{i-1}
\]

\[
= \alpha(p_i - p_{i-1}).
\]

Now by definition

\[
\alpha E^0_i(\Delta X) = \alpha(p_i - p_{i-1}).
\]

So the hypothesis is true for the case \( k = 0 \) for all values of \( i \).

Assume that for the \( k^{th} \) case the hypothesis holds for all \( i \). So we have

\[
\Delta^k_{E_i}(X) = \alpha E^k_i(\Delta X).
\]

By definition

\[
E^{k+1}_i(\Delta X) = (1 - \alpha)E^k_i(\Delta X) + \alpha E^k_{i+1}(\Delta X).
\]
Also, by definition, we have

\[ \Delta_{E_i}^{k+1}(X) = E_{i}^{k+2}(X) - E_{i}^{k+1}(X) \]

\[ = (1 - \alpha)E_i^{k+1}(X) + \alpha E_{i+1}^{k+1}(X) - [(1 - \alpha)E_i^k(X) + \alpha E_{i+1}^k(X)] \]

\[ = (1 - \alpha)\Delta_{E_i}^k(X) + \alpha \Delta_{E_{i+1}}^k(X) \]

\[ = (1 - \alpha)\alpha E_i^k(\Delta X) + \alpha^2 E_{i+1}^k(\Delta X) \quad \text{(by induction hypothesis)} \]

\[ = \alpha E_{i}^{k+1}(\Delta X). \quad \Box \]

**Lemma 3.2.2.** If \( k \in \{0, 1, \ldots, n\} \) and \( i \in \{1, 2, \ldots, n + 1 - k\} \) then

\[ E_i^k(\Delta X) = E_{i+1}^k(X) - E_i^k(X). \]

**Proof:**

For the case \( k = 0 \) and for all \( i \) we have

\[ E_i^0(\Delta X) = p_i - p_{i-1} \]

\[ = E_{i+1}^0(X) - E_i^0(X). \]

Now assume the hypothesis holds for the \( k^{th} \) case for all \( i \).

\[ E_i^k(\Delta X) = E_{i+1}^k(X) - E_i^k(X). \]
Then

\[ E^{k+1}_{i}(\Delta X) = (1 - \alpha)E_{i}^{k}(\Delta X) + \alpha E_{i+1}^{k}(\Delta X) \]
\[ = (1 - \alpha)(E_{i+1}^{k}(X) - E_{i}^{k}(X)) + \alpha(E_{i+2}^{k}(X) - E_{i+1}^{k}(X)) \]
\[ = [(1 - \alpha)E_{i+1}^{k}(X) + \alpha E_{i+2}^{k}(X)] - [(1 - \alpha)E_{i}^{k}(X) + \alpha E_{i+1}^{k}(X)] \]
\[ = E^{k+1}_{i+1}(X) - E^{k+1}_{i}(X). \] □

Now define the operator

\[ \Delta_{i}^{i}(X) = E^{n-i}_{i+1}(X) - E^{n-i+1}_{i}(X). \]

Notice that \( \{\Delta_{i}^{i}(X)\}_{i=1}^{n} \) is the set of control points for the derivative of the second curve in the subdivision mentioned above.

**Lemma 3.2.3.** If \( i \in \{1, 2, \cdots, n\} \) then

\[ \Delta_{i}^{i}(X) = (1 - \alpha)E^{n-i}_{i}(\Delta X). \]

**Proof:** For the case \( i = 1 \) we have

\[ \Delta_{1}^{1}(X) = E^{n-1}_{2}(X) - E^{n}_{1}(X) \]
\[ = E^{n-1}_{2}(X) - (1 - \alpha)E^{n-1}_{1}(X) - \alpha E^{n-1}_{2}(X) \]
\[ = (1 - \alpha)(E^{n-1}_{2}(X) - E^{n-1}_{1}(X)) \]
\[ = (1 - \alpha)E^{n-1}_{1}(\Delta X) \quad \text{(by Lemma 2)}. \]
Assume that the \( i \)th case holds, i.e.

\[
\Delta^i_f(X) = (1 - \alpha)E^{n-i}_i(\Delta X).
\]

Now

\[
\Delta^{i+1}_f(X) = E^{n-i-1}_{i+2}(X) - E^{n-i-1}_{i+1}(X)
= (1 - \alpha)(E^{n-i-1}_{i+1}(X) - E^{n-i-1}_i(X))
= (1 - \alpha)E^{n-i-1}_{i+1}(\Delta X) \quad \text{(by Lemma 2)}.
\]

Let \( c(t) \) be a Bézier curve. When subdivision parameter \( \alpha = 1/2 \) we call this uniform subdivision. Define

\[
[X]^{i,\alpha}_j
\]

to be the \( j \)-th sub-control polygon after applying \( i \) subdivisions with parameter \( \alpha \in (0, 1) \), where \( j \in \{1, \cdots, 2^i\} \). Note that we can restate Lemmas 1 and 3 in terms of this notation as follows: For control points \( X = \{p_0, \cdots, p_n\} \) the following hold

\[
\Delta[X]^{1,\alpha}_1 = \alpha[\Delta X]^{1,\alpha}_1
\]

and

\[
\Delta[X]^{1,\alpha}_2 = (1 - \alpha)[\Delta X]^{1,\alpha}_2
\]
for $\alpha \in (0, 1)$. We can use these results to establish the relationship between the order of subdivision and differentiation.

**Example 3.2.1.** Given control points $X = \{(0, -1), (2, 1), (2 - 2), (0, -2)\}$, subdividing uniformly once gives

$$[X]^{1\frac{1}{2}}_1 = \{(0, -1), (1, 0), \left(\frac{3}{2}, -\frac{1}{4}\right), \left(\frac{3}{2}, -\frac{3}{4}\right)\}$$

and

$$[X]^{1\frac{1}{2}}_2 = \left\{\left(\frac{3}{2}, -\frac{3}{4}\right), \left(\frac{3}{2}, -\frac{5}{4}\right), (1, -2), (0, -2)\right\}.$$

Applying the derivative operator yields

$$\Delta[X]^{1\frac{1}{2}}_1 = \{(3, 3), \left(\frac{3}{2}, -\frac{3}{4}\right), \left(0, -\frac{3}{2}\right)\}$$

and

$$\Delta[X]^{1\frac{1}{2}}_2 \left\{\left(0, -\frac{3}{2}\right), \left(-\frac{3}{2}, -\frac{9}{4}\right), (-3, 0)\right\}.$$

Now applying the derivative operator to $P$ gives

$$\Delta X = \{(6, 6), (0, -9), (-6, 0)\}.$$

Subdividing uniformly yields

$$[\Delta X]^{1\frac{1}{2}}_1 = \{(6, 6), \left(\frac{3}{2}, -\frac{3}{2}\right), (0, -3)\}.$$
and

\[
[\Delta X] \frac{1}{2} _2 = \left\{ (0, -3), \left(-3, -\frac{9}{2}\right), (-6, 0) \right\}.
\]

Notice

\[
\Delta \left[ X \right] ^{1/2} _1 = \frac{1}{2} \left[ \Delta X \right] ^{1/2} _1
\]

and

\[
\Delta \left[ X \right] ^{1/2} _2 = \left( 1 - \frac{1}{2} \right) \left[ \Delta X \right] ^{1/2} _2.
\]

Applying two uniform subdivisions to \( X \) gives

\[
\left\{ \left[ X \right] ^{2 \frac{1}{2}} _j \right\} ^4 _{j=1} = \left\{ \left\{ (0, -1), \left(\frac{1}{2}, -\frac{1}{2}\right), \left(\frac{7}{8}, -\frac{5}{16}\right), \left(\frac{9}{8}, -\frac{5}{16}\right) \right\}, \left\{ \left(\frac{9}{8}, -\frac{5}{16}\right), \left(\frac{11}{8}, -\frac{5}{16}\right), \left(\frac{3}{2}, -\frac{1}{2}\right), \left(\frac{3}{2}, -\frac{3}{4}\right) \right\}, \left\{ \left(\frac{3}{2}, -\frac{3}{4}\right), \left(\frac{3}{2}, -1\right), \left(\frac{11}{8}, -\frac{21}{16}\right), \left(\frac{9}{8}, -\frac{25}{16}\right) \right\}, \left\{ \left(\frac{9}{8}, -\frac{25}{16}\right), \left(\frac{7}{8}, -\frac{29}{16}\right), \left(\frac{1}{2}, -2\right), (0, -2) \right\} \right\}.
\]

Applying the derivative operator yields

\[
\Delta \left\{ \left[ X \right] ^{2 \frac{1}{2}} _j \right\} ^4 _{j=1} = \left\{ \left\{ \left(\frac{3}{2}, -\frac{3}{4}\right), \left(\frac{9}{8}, -\frac{9}{16}\right), \left(\frac{3}{4}, 0\right) \right\}, \left\{ \left(\frac{3}{4}, 0\right), \left(\frac{3}{2}, -\frac{9}{16}\right), \left(0, -\frac{3}{4}\right) \right\}, \left\{ (0, -\frac{3}{4}), \left(-\frac{3}{8}, -\frac{15}{16}\right), \left(-\frac{3}{4}, -\frac{3}{4}\right) \right\}, \left\{ \left(-\frac{3}{4}, -\frac{3}{4}\right), \left(-\frac{9}{8}, -\frac{9}{16}\right), \left(-\frac{3}{2}, 0\right) \right\} \right\}.
\]
Subdividing uniformly $[\Delta X]^{1/2}_1$ and $[\Delta X]^{1/2}_2$ yields

$$\left\{ [\Delta X]^{2/3}_j \right\}_{j=1}^4$$

$$= \left\{ \left\{ (3, 3), \left( \frac{9}{4}, \frac{9}{8} \right), \left( \frac{3}{2}, 0 \right) \right\}, \left\{ (3, 0), \left( \frac{3}{4}, -\frac{9}{8} \right), \left( 0, -\frac{3}{2} \right) \right\}, \left\{ (0, -\frac{3}{2}), \left( -\frac{3}{4}, -\frac{15}{8} \right), \left( -\frac{3}{2}, -\frac{3}{2} \right) \right\}, \left\{ \left( -\frac{3}{2}, -\frac{3}{2} \right), \left( -\frac{9}{4}, -\frac{9}{8} \right), (-3, 0) \right\} \right\}$$

Observe that

$$\Delta [X]^{2/3}_j = \frac{1}{2} [\Delta X]^{2/3}_j$$

for $j = 1, 2, 3, 4$ as expected.

### 3.3 Discrete and continuous derivatives

Define $\mathcal{L}(X, [a, b])$ to be the uniform parametrization of the control polygon $X = \{p_0, p_1, \ldots, p_n\}$. So

$$\mathcal{L}(X, [a, b]) \left( a + j \frac{b-a}{n} \right) = p_j$$

and $\mathcal{L}(X, [a, b])$ is linear on the intervals

$$\left[ j \frac{b-a}{n}, (j+1) \frac{b-a}{n} \right] \quad \text{for} \quad j = 0, 1, \ldots n - 1.$$
For a Bézier curve $c(t)$ defined on $[a, b]$ with control points $X = \{ p_0, \cdots, p_n \}$ the discrete derivative is defined [20, 22] as
\[
D[c(t)] = \frac{1}{(b - a)} \mathcal{L}(n \Delta X, [a, b]).
\]

Applying $m$ subdivisions with subdivision parameter $\alpha$ yields $2^m$ subcontrol polygons. Each subcontrol polygon has control points $[X]_{i}^{m, \alpha}$ and an associated interval $I_{i}^{(m, \alpha)}$, where $i = 1, \cdots, 2^m$. The refined discrete derivative is given by
\[
D_{m, \alpha}[c(t)] = \begin{cases}
\frac{1}{|I_{1}^{(m, \alpha)}|} \mathcal{L} \left( n \Delta \left[ X \right]_{1}^{m, \alpha}, I_{1}^{(m, \alpha)} \right) & t \in I_{1}^{(m, \alpha)} \\
\vdots & \\
\frac{1}{|I_{2^m}^{(m, \alpha)}|} \mathcal{L} \left( n \Delta \left[ X \right]_{2^m}^{m, \alpha}, I_{2^m}^{(m, \alpha)} \right) & t \in I_{2^m}^{(m, \alpha)}.
\end{cases}
\]

**Theorem 3.3.1.** For a Bézier curve with control points $X = \{ p_0, \cdots, p_n \}$ the discrete derivative can be written as follows
\[
D_{m, \alpha}[c(t)] = \begin{cases}
\mathcal{L} \left( n \Delta \left[ X \right]_{1}^{m, \alpha}, I_{1}^{(m, \alpha)} \right) & t \in I_{1}^{(m, \alpha)} \\
\vdots & \\
\mathcal{L} \left( n \Delta \left[ X \right]_{2^m}^{m, \alpha}, I_{2^m}^{(m, \alpha)} \right) & t \in I_{2^m}^{(m, \alpha)}
\end{cases}
\]

**Proof:**
This follows directly from the definition of $D_{m, \alpha}[c(t)]$ and Lemmas 1 and 3.
It is known that discrete derivatives from successive subdivisions converge to the continuous derivative at a rate of $O(\frac{1}{2^m})$ after $m$ subdivisions [20]. Also there is an existing result that gives a sharp upper bound for the distance between a curve and its control polygon after $m$ subdivisions [23]. Combining the latter with Theorem 4 we can calculate an upper bound for the error of the discrete derivative.

The second centered difference is defined as $\Delta_2 p_i = p_{i-1} - 2p_i + p_{i+1}$ and $\Delta_2(X) = \{\Delta_2 p_i\}_{i=0}^n$. Then

$$\|\Delta_2(X)\|_\infty = \max \|\Delta_2 p_i\|.$$

The following constant has been defined [23] that is dependent on the degree of the curve, $n$,

$$N_\infty(n) = \frac{\left\lfloor \frac{n}{2} \right\rfloor \left\lceil \frac{n}{2} \right\rceil}{2n}.$$  

After $m$ subdivisions with subdivision parameter $\alpha$ the maximum distance between a curve and its control polygon is bounded above by [23]

$$x^{2m} N_\infty(n) \|\Delta_2(X)\|_\infty \text{ where } x = \max\{\alpha, 1 - \alpha\}$$

Note that under favorable conditions this bound is sharp [23]. The optimal subdivision parameter depends on the degree of the curve and can be com-
puted for any degree [23]. For example for \( n = 3 \) the optimal subdivision parameter is given by \( \alpha \approx 0.43 \) [23].

**Theorem 3.3.2.** Given a Bézier curve with control points \( X = \{p_0, \cdots, p_n\} \), the maximum distance between the discrete derivative and the continuous derivative after \( m \) subdivisions with subdivision parameter \( \alpha \) is bounded by

\[
x^{2m} n N_\infty (n - 1) \| \Delta_2 (\Delta X) \|_\infty \text{ where } x = \max\{\alpha, 1 - \alpha\}.
\]

**Proof:**

The curve \( c(t) \) has control points \( X = \{p_0, \cdots, p_n\} \). The hodograph of \( c(t) \) has control points \( n \Delta X = n \{p_1 - p_0, \cdots, p_n - p_{n-1}\} \). After \( m \) subdivisions with subdivision parameter \( \alpha \) the maximum distance between the hodograph curve and its control polygon is given by

\[
x^{2m} N_\infty (n - 1) n \| \Delta_2 (\Delta X) \|_\infty \text{ where } x = \max\{\alpha, 1 - \alpha\}. \quad \Box
\]

**Corollary 3.3.3.** To have error tolerance \( \epsilon \) for approximating \( \frac{d}{dt} c(t) \) with \( D_{m,\alpha}[c(t)] \) then at least \( m \) subdivisions are required where

\[
m = \left\lceil \frac{\ln(\epsilon/K)}{2 \ln(x)} \right\rceil,
\]

where \( K = n N_\infty (n - 1) \| \Delta_2 (\Delta X) \|_\infty \) and \( x = \max\{\alpha, 1 - \alpha\} \).
3.4 Conclusion

In this chapter we have established error bounds for the discrete derivative as an approximation of the hodograph of a Bézier curve. Also for a given error tolerance we have provided a method to calculate the minimum number of subdivisions required for the approximation to be within the specified tolerance.
4. TOPOLOGY FOR DYNAMIC GRAPHICS

In computer graphics and scientific visualization, Bézier curves are common geometric representations. In this chapter we investigate techniques to ensure ambient isotopic equivalence during dynamic visualization of writhing Bézier curves.

A typical display method is to render a piecewise linear (PL) approximation that lies within a prescribed tolerance of the curve. During dynamic visualization, the control points of a Bézier curve are perturbed to change the shape of the curve. The distance between the perturbed PL structure and the perturbed curve it represents can change significantly, possibly changing the underlying topology and introducing unwanted artifacts to the display. We give a strategy to perturb the curve smoothly and keep track of the error introduced by perturbations. This allows us to refine the PL curve when appropriate and avoid spurious topological changes. This work is motivated by applications to visualization of Big Data from simulations on high performance computing architectures.
4.1 Introduction

In geometric modeling Bézier curves are frequently used to model complex geometric objects [7]. The curve models are smooth structures but PL approximations are typically used to render the spline. Recent work [4, 18] has presented methods for topologically correct visualization of an original static model, as well as some special cases of perturbing models. New methods are presented here for the topologically correct dynamic visualization, as motivated by molecular simulations. As points on the PL model are perturbed over the course of the simulation, the PL model could diverge significantly from the smooth model that it represents. This may introduce topological artifacts to the display, resulting in a flawed image that could mislead domain scientists.

Our formal analysis is motivated by graphics experiments, which are summarized in Experiment 4.3.1. We observed that the PL approximation used for graphics could be perturbed for more time steps, while still preserving ambient isotopic equivalence than might be expected from previously published bounds [11]. This data-specific a posteriori analysis led us to question whether we could develop rigorous, predictive methods for the permissible number of time steps. A method based upon second centered differences has been developed for that predictive capability to support efficient frame generation, where this new method is motivated by Experiment 4.3.1, with a formal analysis in Example 4.7.1.
Many perturbation strategies are possible, but in dynamic visualization, re-
taining differentiability over time is often desirable, so our predictive method
is presented in the context of a representative differentiable perturbation
strategy. However, the formal analysis is quite general, and other pertur-
bation strategies could easily be integrated by a user interested in other
applications. Our exposition first uses a non-differentiable strategy to intro-
duce some central concepts within this simplified context, but the ensuing
differentiable strategy is then used in the rest of the development. Our
distinctive contributions are analyses of the amount of error introduced by
each perturbation. This error can be monitored and the PL model can be
refined as necessary to avoid unwanted topological changes. For ease of no-
tation the investigation below is performed on Bézier curves, however the
analysis is identical for general B-spline curves [7]. The motivating graphics
experiments are summarized in Section 4.3 and a representative analysis is
presented as Example 4.7.1.

4.2 Background, Motivation and Notation

In this section we introduce some fundamental definitions and notation.
4.2.1 Curves and control polygons

We remind the reader that a degree \( d \) Bézier curve with control points \( X = \{q_0, \ldots, q_d\} \) is denoted by

\[
c(t) = \sum_{i=0}^{d} \binom{d}{i} (i - t)^{d-i} q_i
\]

where the PL curve connecting \( q_0, \ldots, q_d \) is called the control polygon of \( c \).

Recall that a subdivision algorithm operates on \( X \) to generate two PL curves, each having \( d + 1 \) vertices, denoted, respectively as \( X_L \) and \( X_R \), as shown in Figure 2.1.

**Definition 4.2.1.** Given the polygon generated by \( X = \{q_0, \ldots, q_d\} \), the second centered difference of a given control point \( q_i \) is defined as

\[
\Delta_2 q_i = q_{i-1} - 2q_i + q_{i+1}.
\]

We define \( \Delta_2 q_0 = \Delta_2 q_d = 0 \). The maximal second centered difference of the polygon generated by \( X \) is given by

\[
\|\Delta_2 X\|_\infty = \max_{0 \leq i \leq d} \|\Delta_2 q_i\|.
\]

Given a degree \( d \) curve with control points \( X = \{q_0, \ldots, q_d\} \), after \( \alpha \) uniform subdivisions the maximal distance between the control polygon and the curve...
has been shown [23] to be

\[
\left( \frac{1}{2} \right)^{2\alpha} \| \Delta_2 X \|_\infty N_\infty(d).
\]

Here \( N_\infty(d) = \frac{\lfloor d/2 \rfloor \lceil d/2 \rceil}{2d} \). Note that this distance is actually attained [23]. So subdividing \( \alpha \) times guarantees that the PL structure is within the specified tolerance for display, \( \epsilon_d \), where

\[
\alpha = \left\lceil -\frac{1}{2} \log_2 \left( \frac{\epsilon_d}{\| \Delta_2 X \|_\infty N_\infty(d)} \right) \right\rceil.
\]

4.2.2 Related Work

Molecular simulations are run on high performance computing (HPC) architectures, often generating petabytes of data, initiating a typical ‘Big Data’ problem. This data output is too voluminous for standard numerical analytic techniques and dynamic visualization has become a common zero-th order analysis. The supportive dynamic visualization techniques are well-established [16, 17] and will not be addressed further. The vitally important and novel support from this work is to provide rigorously proven numerical assurances that the frames being viewed have appropriate approximation in order to avoid topological artifacts in the images that could prove misleading to the domain scientists [12, 6]. To establish context for this work, a brief overview will be given of the three primary facets of supportive mathematics, geometric models and molecular simulations. The emphasis here is upon
the new mathematics to meet the new Big Data challenges posed by the recent prevalence of these petabytes of simulation output, where this emerging mathematics is being developed from a blend of theory and experimentation. There are so many tools available for molecular visualization, that it suffices to provide two broad summary portals [16, 17]. Often protein data is of interest, which appears publicly in an international resource [2]. The indicated resources do not directly provide geometric models of the molecules visualized – only images are produced.

The molecular simulation research [26, 28, 29, 30, 31] closely aligns with the work presented here, with [31] being of particular interest because of its use of splines to model molecules, as also assumed here. Alternate geometric representations have been considered [13, 14, 27] for molecules, but the choice of splines here is offered as a very broad, fundamental representation, which could be examined for adaptation to these alternate representations. The more contemporary Big Data issues had not yet appeared when this earlier work had already been completed.

The emphasis here upon geometric representations echoes much work in computer-aided geometric design [7]. In particular, this dynamic molecular visualization has been synergistically pursued with an emerging virtual reality (VR) engineering design laboratory [10]. A fascinating common use is of 1-dimensional geometry to model the molecule writhing proteins and design features [8], where the latter application is integrated with a constraint solver.
Motivating Applications  The mathematics proven here was motivated by design of dynamic visualization for molecular simulations in HPC. As an initial analysis, a dynamic visualization is synchronized with the ongoing simulation. The graphics at each frame are displayed by PL approximations, raising the possibility that an image could show an intersection on a writhing molecule where none occurs on the more accurate spline model. The isotopic analysis presented is designed to integrate the necessary numerical accuracy with sufficient performance for dynamic visualization. Subdivision is chosen for the PL approximation, but the analysis presented here could easily be adapted to other PL approximation techniques, such as PL interpolation through selected points on the curve. Proteins are typical objects of interest, modeled as spline curves. Public data bases [2] provide spatial co-ordinates for interpolation to create a spline model. However, there can easily be hundreds of thousands of such co-ordinates, so that interpolation by a single segment spline would be also have degree on the order of hundreds of thousands — typically prohibitive for interactive graphics, where much lower degree is preferred (often as low as degree 3, but rarely higher than degree 8). Sufficiently accurate, low degree models can be created by the composite curves [7] used here. Since these geometric molecular models are not readily available in the public resources [2, 16, 17] prototype software is also being developed to provide those models, but reports on those tools will appear elsewhere.
The presented mathematical analysis has guided our algorithmic design so that we are now confident that we can use splines of sufficiently low degree, while maintaining desired topological characteristics. It remains to integrate these topology preserving techniques into the supporting dynamic visualizations discussed. That full breadth of experimental work is beyond the scope of the present paper and remains as subject for future publications. The crucial novel concepts are presented here.

### 4.3 Graphics Efficiency Experiment

The efficient use of PL approximations in dynamic visualizations has previously appeared [11], as a way to ensure correct graphics topology during animation, as previously presented relative to isotopic equivalence. That previous strategy [11] will now be briefly summarized, where this work adds the additional perspective of practical limits on the number of frames where this aggressive strategy can be invoked. As perspective on the extreme data and performance demands of this environment, it is instructive to note the order of 30 - 60 frames per second to synchronize dynamic visualization with a simulation producing petabytes of output.

Throughout the course of a molecular dynamics simulation, the molecule moves as reflected by movement of a spline. Each frame will use PL approximation. Here are two graphics display options to consider:
Option 1: At each time step, perturb the spline and create a new PL approximation for display.

Option 2: Create a PL approximation of the spline at some initial time step. Continue to perturb this PL approximation until it is no longer sufficiently accurate for graphics display.

Clearly, Option 2 can eliminate the approximation algorithm at some time steps. The previous work [11] provided existence theorems for maintaining isotopic equivalence during continued perturbation of these PL approximations. This work refines [11] by now providing specific numerical analyses to show exactly how many subsequent frames can invoke this aggressive strategy, before it becomes necessary to create a new PL approximation to ensure ongoing topological fidelity between the spline and its graphics approximation.

A representative graphics experiment will be summarized to show implications of Option 2. A sufficient\(^2\) perturbation bound [11, Proposition 5.2] to preserve ambient isotopy is \((1/2)\nu\), with \(\nu\) defined as the minimal distance between points and edges of a PL curve [1]. With the control points here, we note that \((1/2)\nu = 1/2\). We will show, later, that this upper bound, while sufficient to preserve ambient isotopy, leaves open the possibility of more aggressive perturbation strategies.

\(^2\) There is an obvious typographical error [11, Proposition 5.2], but the value used here is correct.
Experiment 4.3.1. Consider the non-self intersecting $C^1$ composite cubic Bézier curve in $\mathbb{R}^2$, as depicted on the left hand side of Figure 4.1. The control points are given by the following points together with their reflections through the line $y = 3$:

$$(0, 6), (1, 5), (2, 4.5), (3, 5.25), (4, 6), (5, 7), (6, 8), (7, 9), (9, 10), (11, 11), (13, 12), (15, 13), (17, 13.35), (19, 13.7), (21, 13), (22, 12), (23, 11), (24, 10), (24.5, 8), (25, 6), (25, 4), (25, 3).$$

The control polygon is green with red control points, the underlying curve is black. Note that the differences are most easily noted near the top of both images. Perturbing $p_u$ and $p_v$ over ten time steps introduces a self intersection to the PL structure that is not present in the underlying spline curve, as illustrated on the right hand side of Figure 4.1 and Figure 4.2. For brevity of presentation, the example of Figure 4.1 presents the graphics of the original and perturbed Bézier curves to show that both are non-self-intersecting, which can be rigorously verified [1]. We return to this example for a detailed analysis in Section 4.6.3.

We note that the previous bound with of $(1/2)\nu = 1/2$ would have guaranteed that the first 5 time steps were permissible. When these visual experiments showed that topological fidelity could be preserved until the 10th step, we pursued a deeper analysis to explicate identification of this longer preservation of topology, as presented next.
4.4 Notation for perturbation analysis

We now define the notation required for the perturbation analysis. We shall examine $n$ time steps denoted $\{t_1, \cdots, t_n\}$, $t_0$ denotes the time at initializa-
We assume that we are given a refined control polygon so that it is within $\epsilon_d$ (as introduced in Section 4.2) of the represented curve. Note if $\alpha$ subdivisions are required then, from the original set of control points $\{q_0, \ldots, q_d\}$, there are generated $w$ control points where $w = 2^\alpha d + 1$. Denote the subdivided, but unperturbed, control polygon by

$$X_0 = \{p_0, p_1, \cdots, p_w\}.$$  

Let $X_i$ denote the perturbed control polygon at time $t_i$. Assume we are supplied with a $(w + 1) \times n$ perturbation matrix, $\Gamma$, where each row contains perturbation vectors for a corresponding control point and each column contains the perturbation vectors for all control points at the corresponding time step, i.e.

$$\Gamma =
\begin{pmatrix}
  p_0 & \gamma_{0,1} & \gamma_{0,2} & \cdots & \gamma_{0,n} \\
  p_1 & \gamma_{1,1} & \gamma_{1,2} & \cdots & \gamma_{1,n} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  p_w & \gamma_{w,1} & \gamma_{w,2} & \cdots & \gamma_{w,n}
\end{pmatrix}
$$

where $\gamma_{i,j}$ denotes the perturbation vector applied to $p_i$ at time $t_j$ (may be the zero vector).
Fig. 4.3: Perturbation over a single time step.

Let \( \delta_j p_i \) denote the coordinates of the point that originated at \( p_i \) at \( t_j \), i.e.

\[
\delta_j p_i = p_i + \sum_{k=1}^{j} \gamma_{i,k}.
\]

4.5 Non-differentiable Perturbations

In cases where maintaining differentiability of the curve is not required, we may simply perturb each point by the prescribed vector. At \( t_0 \) we are given \( X_0 \) and \( \Gamma \) as described above. At each \( t_i \) we can calculate \( X_i \) from \( \Gamma \) and \( X_{i-1} \).

Given the points \( X_0 = \{ p_0, p_1, p_2, p_3 \} \) and the perturbation matrix,

\[
\Gamma = \begin{bmatrix}
0 & 0 \\
\gamma_{1,1} & \gamma_{1,2} \\
0 & 0 \\
\gamma_{3,1} & \gamma_{3,2}
\end{bmatrix}
\]
as depicted in Figure 4.3. We can calculate

\[ X_1 = \{ \delta_1 p_0, \delta_1 p_1, \delta_1 p_2, \delta_1 p_3 \} = \{ p_0, p_1 + \gamma_{1,1}, p_2, p_3 + \gamma_{3,1} \}, \text{ and} \]

\[ X_2 = \{ \delta_2 p_0, \delta_2 p_1, \delta_2 p_2, \delta_2 p_3 \} = \{ p_0, p_1 + \gamma_{1,1} + \gamma_{1,2}, p_2, p_3 + \gamma_{3,1} + \gamma_{3,2} \}. \]

4.5.1 Perturbing a single point

First we consider perturbing a single point over a single time step. At initialization we have

\[ X_0 = \{ p_0, \ldots, p_j, \ldots, p_w \}. \]

Note that

\[ \| \Delta_2 X_0 \|_{\infty} N_{\infty}(d) \leq \epsilon_d. \]

Let \( p_j \) be the point being perturbed. At time \( t_1 \) the point \( p_j \) is perturbed to \( p_j + \gamma_{j,1} \) and all other points remain in their original positions. So we have,

\[ X_1 = \{ p_0, \ldots, \delta p_j, \ldots, p_w \} = \{ p_0, \ldots, p_j + \gamma_{j,1}, \ldots, p_w \}. \]

The only second differences effected are \( \Delta_2(p_{j-1}), \Delta_2(p_j) \) and \( \Delta_2(p_{j+1}) \).
Therefore,

$$\| \Delta_2 X_1 \|_\infty = \max \{ \| \Delta_2 X_0 \|_\infty, \| \Delta_2 (\delta_1 p_{j-1}) \|, \| \Delta_2 (\delta_1 p_j) \|, \| \Delta_2 (\delta_1 p_{j+1}) \| \} ,$$

where

$$\Delta_2 (\delta_1 p_{j-1}) = p_{j-2} - p_{j-1} + p_j + \gamma_{j,1} = \Delta_2 (p_{j-1}) + \gamma_{j,1} ,$$
$$\Delta_2 (\delta_1 p_j) = \Delta_2 (p_j) - 2 \gamma_{j,1}, \text{ and}$$
$$\Delta_2 (\delta_1 p_{j+1}) = \Delta_2 (p_{j+1}) + \gamma_{j,1}. $$

This approach extends easily to $n$ time steps

$$\| \Delta_2 X_n \|_\infty = \max \{ \| \Delta_2 X_0 \|_\infty, \| \Delta_2 (\delta_n p_{j-1}) \|, \| \Delta_2 (\delta_n p_j) \|, \| \Delta_2 (\delta_n p_{j+1}) \| \}$$

where $\Delta_2 (\delta_n p_{j-1}) = \Delta_2 (p_{j-1}) + \sum_{i=1}^n \gamma_{j,i}$, $\Delta_2 (\delta_n p_j) = \Delta_2 (p_j) - 2 \sum_{i=1}^n \gamma_{j,i}$
and $\Delta_2 (\delta_n p_{j+1}) = \Delta_2 (p_{j+1}) + \sum_{i=1}^n \gamma_{j,i}$.

\textbf{4.5.2 Perturbing multiple points}

To perturb multiple points over multiple time steps, using the information supplied by $\Gamma$, sort the points being perturbed into adjacency chains, i.e. sets of adjacent control points denoted $Q_0, \cdots, Q_s$ where each $Q_i$ contains either a single point or a list of adjacent points to be perturbed. This is necessary as chains of different length have different effects on the second differences.
that involve points in that chain. Let $|Q_i| = u$. If $u = 1$ then this is treated as in the single point case above. If $u = 2$ then we write $Q_i = \{p_k, p_{k+1}\}$, and we compute the effected centered differences as follows:

$$\Delta_2(\delta np_{k-1}) = \Delta_2(p_{k-1}) + \sum_{j=1}^{n} \gamma_{k,j},$$

$$\Delta_2(\delta np_k) = \Delta_2(p_k) + \sum_{j=1}^{n} (\gamma_{k+1,j} - 2\gamma_{k,j}),$$

$$\Delta_2(\delta np_{k+1}) = \Delta_2(p_{k+1}) + \sum_{j=1}^{n} (\gamma_{k,j} - 2\gamma_{k+1,j}),$$

$$\Delta_2(\delta np_{k+2}) = \Delta_2(p_{k+2}) + \sum_{j=1}^{n} \gamma_{k+1,j}.$$ 

If $u \geq 3$ then $Q_i = \{p_k, \ldots, p_{k+v}\}$ for some $v \geq 2$. The effected centered differences are computed:

$$\Delta_2(\delta np_{k-1}) = \Delta_2(p_{k-1}) + \sum_{j=1}^{n} \gamma_{k,j},$$

$$\Delta_2(\delta np_k) = \Delta_2(p_k) + \sum_{j=1}^{n} (\gamma_{k+1,j} - 2\gamma_{k,j}),$$

$$\vdots$$

$$\Delta_2(\delta np_s) = \Delta_2(p_s) + \sum_{j=1}^{n} (\gamma_{s-1,j} - 2\gamma_{s,j} + \gamma_{s+1,j}),$$

$$\vdots$$

$$\Delta_2(\delta np_{k+v}) = \Delta_2(p_{k+v}) + \sum_{j=1}^{n} (\gamma_{k+v-1,j} - 2\gamma_{k+v,j}),$$

$$\Delta_2(\delta np_{k+v+1}) = \Delta_2(p_{k+v+1}) + \sum_{j=1}^{n} \gamma_{k+v,j}.$$
4.6 Differentiable Perturbations

When it is also desirable to maintain a degree of differentiability either for appearances, analysis or both, we define a perturbation strategy that guarantees $C^1$ continuity (assuming the original curve is at least $C^1$).

4.6.1 Perturbation strategy

We are given a composite Bézier curve to perturb. Recall that a junction point is a point where curve segments meet. We identify three types of point:

- **Type 1:** A point adjacent to a junction point (two points are considered adjacent if they are consecutive control points).

- **Type 2:** A junction point.

- **Type 3:** A point that is neither a junction point nor adjacent to a junction point.

To maintain $C^1$ continuity we must require that the tangent edges with a shared junction point be collinear and have the same length [7].

**Type 1**

If we perturb a type 1 point in order to satisfy the $C^1$ criteria we perturb the junction point to the midpoint of the line segment joining its adjacent
points. This approach is illustrated in the following example, where \( p_2 \) is being perturbed, relative to the junction point of \( p_3 \).

![Figure 4.4: Type 1 perturbation.](image)

**Example 4.6.1.** Given a composite cubic control polygon with sub polygons \( \{p_0, p_1, p_2, p_3\} \) and \( \{p_3, q_1, q_2, q_3\} \) as depicted in Figure 4.4. If we perturb \( p_2 \) by a vector \( \gamma \):

\[
p_2 \rightarrow \delta p_2 = p_2 + \gamma,
\]

then to maintain \( C^1 \) continuity we perturb \( p_3 \) as follows:

\[
p_3 \rightarrow \bar{p}_3 = \frac{\delta p_2 + q_1}{2}.
\]

**Type 2**

To maintain \( C^1 \) differentiability when perturbing a type 2 point we must also perturb its adjacent points by the same vector so the tangent edges are collinear and have the same length.
Example 4.6.2. Here we have a composite cubic control polygon with sub polygons \{p_0, p_1, p_2, p_3\} and \{p_3, q_1, q_2, q_3\} as shown in Figure 4.5. Perturbing \(p_3\) by \(\gamma\) has the following effect:

\[
\begin{align*}
  p_2 & \rightarrow \delta p_2 = p_2 + \gamma, \\
p_3 & \rightarrow \delta p_3 = p_3 + \gamma, \\
q_1 & \rightarrow \delta q_1 = q_1 + \gamma.
\end{align*}
\]

Type 3

Since Type 3 points do not effect tangent edges we can just perturb them as normal without perturbing neighboring points.
4.6.2  Perturbing a single point

We can now examine the effect of perturbing a single point using the strategy outlined above. Given

\[ X_0 = \{ p_0, \cdots, p_j, \cdots, p_w \} . \]

Note that

\[ \| \Delta_2 X_0 \|_\infty N_\infty(d) \leq \epsilon_d \]

Let \( Y = \{ p_j \} \) for some \( j \in \{ 0, 1, \cdots, w \} \). At time \( t_1 \) the point \( p_j \) is perturbed to \( p_j + \gamma_{j,1} \), note that adjacent points may be perturbed depending on the type.

**Type 1.** After all time steps are completed we have

\[ X_n = \left\{ p_0, \cdots, p_j + \sum_{k=1}^{n} \gamma_{j,k}, \frac{p_j + \sum_{k=1}^{n} \gamma_{j,k} + p_{j+2}}{2}, \cdots, p_w \right\} . \]

The modified second centered differences are

\[ \Delta_2(\delta_n p_{j-1}) = \Delta_2 p_{j-1} + \sum_{k=1}^{n} \gamma_{j,k} , \]
\[
\Delta_2(\delta_p) = \left(p_j - \frac{3}{2}p_j + \frac{1}{2}p_j + \frac{1}{2}p_j + \frac{1}{2}p_j + \frac{1}{2}p_j\right) - \frac{3}{2} \sum_{k=1}^{n} \gamma_{j,k},
\]

\[
\Delta_2(\delta_{p+1}) = 0,
\]

\[
\Delta_2(\delta_{p+2}) = \left(\frac{1}{2}p_j - \frac{3}{2}p_j + p_j + p_j + p_j + p_j\right) + \frac{1}{2} \sum_{k=1}^{n} \gamma_{j,k}.
\]

**Type 2.** After the first time step

\[
X_1 = \{p_0, \cdots, p_{j-1} + \gamma_{j,1}, p_j + \gamma_{j,1}, p_{j+1} + \gamma_{j,1} \cdots, p_w\},
\]

\(p_{j-1}, p_j\) and \(p_{j+1}\) are each perturbed by \(\gamma_{j,i}\) at time \(t_i, i \in \{1, \cdots, n\}\).

At time \(t_n\) we have

\[
\|\Delta_2 X_n\|_\infty = \max\{\|\Delta_2 X_0\|_\infty, \max\{\|\Delta_2 (\delta_n p_k)\|_{1:k=2}\}\}.
\]

The changes to the second centered differences are

\[
\Delta_2 (\delta_{p-2}) = \Delta_2 (p_{j-2}) + \sum_{k=1}^{n} \gamma_{j,k},
\]

\[
\Delta_2 (\delta_{p-1}) = \Delta_2 (p_{j-1}) - \sum_{k=1}^{n} \gamma_{j,k},
\]

\[
\Delta_2 (\delta_{p}) = 0
\]

\[
\Delta_2 (\delta_{p+1}) = \Delta_2 (p_{j-1}) - \sum_{k=1}^{n} \gamma_{j,k},
\]

\[
\Delta_2 (\delta_{p+2}) = \Delta_2 (p_{j+2}) + \sum_{k=1}^{n} \gamma_{j,k}.
\]
Type 3. If we are perturbing a type 3 point then at time $t_n$ we have

$$\|\Delta_2 X_n\|_\infty = \max\{\|\Delta_2 X_0\|_\infty, \|\Delta_2 (\delta_n p_{j-1})\|, \|\Delta_2 (\delta_n p_j)\|, \|\Delta_2 (\delta_n p_{j+1})\|\},$$

with the changes in second centered differences:

$$\|\Delta_2 (\delta_n p_{j-1})\|_\infty = \|\Delta_2 (p_{j-1}) + \sum_{i=1}^{n} \gamma_{j,i}\|_\infty,$$

$$\|\Delta_2 (\delta_n p_j)\|_\infty = \|\Delta_2 (p_j) - 2 \sum_{i=1}^{n} \gamma_{j,i}\|_\infty,$$

$$\|\Delta_2 (\delta_n p_{j+1})\|_\infty = \|\Delta_2 (p_{j+1}) + \sum_{i=1}^{n} \gamma_{j,i}\|_\infty.$$

4.6.3 Perturbing multiple points

Let $p_j$ and $p_{j+2}$ be type 1 point, so $p_j$ is a type 2. We consider the illustrative case where $p_j, p_{j+1}$ and $p_{j+2}$ are each being perturbed over $n$ time steps:

$$p_j \rightarrow p_j + \sum_{k=1}^{n} (\gamma_{j,k} + \gamma_{j+1,k}),$$

$$p_{j+1} \rightarrow \frac{1}{2} \left( p_j + p_{j+2} + \sum_{k=1}^{n} (\gamma_{j,k} + 2\gamma_{j+1,k} + \gamma_{j+2,k}) \right),$$

$$p_{j+2} \rightarrow p_{j+2} + \sum_{k=1}^{n} (\gamma_{j+1,k} + \gamma_{j+2,k}).$$
The effect on the second differences is as follows:

\[ \Delta_2(\delta_{n,p_j}) = \Delta_2 p_{j-1} + \sum_{k=1}^{n} (\gamma_{j,k} + \gamma_{j+1,k}), \]
\[ \Delta_2(\delta_{n,p_j}) = p_{j-1} - \frac{3}{2} p_j + \frac{1}{2} p_{j+2} + \sum_{k=0}^{n} \left( -\frac{3}{2} \gamma_{j,k} - \gamma_{j+1,k} + \frac{1}{2} \gamma_{j+2,k} \right), \]
\[ \Delta_2(\delta_{n,p_{j+1}}) = 0, \]
\[ \Delta_2(\delta_{n,p_{j+2}}) = \frac{1}{2} p_j - \frac{3}{2} p_{j+2} + p_{j+3} + \sum_{k=1}^{n} \left( \frac{1}{2} \gamma_{j,k} - \gamma_{j+1,k} - \frac{3}{2} \gamma_{j+2,k} \right), \]
\[ \Delta_2(\delta_{n,p_{j+3}}) = \Delta_2 p_{j+3} + \sum_{k=1}^{n} (\gamma_{j+1,k} + \gamma_{j+2,k}). \]

4.7 An Example Predictive Analysis

Our predictive method is now applied to formalize the empirical observations of Experiment 4.3.1, explicating extensions beyond previous bounds [11].

Example 4.7.1. The cubic Bézier curve of Experiment 4.3.1 was specifically synthesized to permit more aggressive PL graphics perturbations than previously known [11]. Given control points

\[ X_0 = \{(0,6),(1,5),(2,4.5),\ldots,(2,1.5),(1,1),(0,0)\}. \]

Denote \{(0,6),(1,5),(2,4.5)\} by \( U = \{p_{u-2},p_{u-1},p_u\} \) and \{(2,1.5),(1,1),(0,0)\} by \( V = \{p_v,p_{v+1},p_{v+2}\} \). Let the display tolerance, \( \epsilon_d = 1.9167 \). The maximal distance between the curve and the control polygon is 5/12 which we trivially note is less than the given \( \epsilon_d \). Say we wish to perturb the points in \( U \) and \( V \)
over 10 time steps with perturbation vectors

\[
\{\gamma_{u-2,k}\}_{k=1}^{10} = \{\gamma_{u-1,k}\}_{k=1}^{10} = \{\gamma_{u,k}\}_{k=1}^{10}
\]

and

\[
\{\gamma_{v,k}\}_{k=1}^{10} = \{\gamma_{v+1,k}\}_{k=1}^{10} = \{\gamma_{v+2,k}\}_{k=1}^{10}
\]

where

\[
\{\gamma_{u,k}\}_{k=1}^{10} = \{(0, 5/20), (0, 4/20), (0, 4/20), (0, 4/20), (0, 3/20),
(0, 2/20), (0, 1/20), (0, 1/20), (0, 1/20), (0, 5/20)\}, \text{ and}
\]

\[
\{\gamma_{v,k}\}_{k=1}^{10} = \{(0, -5/20), (0, -4/20), (0, -4/20), (0, -4/20), (0, -3/20),
(0, -2/20), (0, -1/20), (0, -1/20), (0, -1/20), (0, -5/20)\}.
\]

For this curve, \(1/2\nu = 1/2\), a value which is clearly exceeded after 5 steps of this strategy. Since previous criteria [11] were only sufficient, the rest of this example demonstrates that greater perturbation is possible to support efficiency in Strategy 2. Since the analysis for points in \(U\) and \(V\) is identical we shall focus on \(V\). Notice that

\[
\sum_{k=1}^{10} \gamma_{v,k} = \left(0, \frac{3}{2}\right).
\]
Since $p_v$ is a type 2 point, the junction point $p_j = (3, 3/4)$ will also be perturbed as described above. Denote the control point following $p_j$ by $p_{j+1}$. After ten perturbations the effect is as follows,

$$p_v = (2, 3/2) \rightarrow \delta_{10}p_v = (2, 3)$$

and

$$p_j = (3, 3/4) \rightarrow \delta_{10}p_j = (3, 3/2).$$

We require that for each $i$,

$$\|\Delta_2X_i\|_\infty + \| \sum_{j=1}^{i} \gamma_{X,j} \| < \epsilon_d.$$

Here $\sum \gamma_{X,j}$ is the sum of the perturbation vectors applied to the control point that yields $\|\Delta_2X_i\|_\infty$. These quantities are easily calculated using the analysis above. It is easy to see that $\|\Delta_2X_i\|_\infty + \| \sum_{j=1}^{i} \gamma_{X,j} \| < \epsilon_d$ for $i = 1, \cdots, 9$.

At the ninth time step we have

$$\|\Delta_2X_9\|_\infty + \| \sum_{j=1}^{9} \gamma_{X,j} \| = 0.65 + 1.25 = 1.9 < \epsilon_d$$

At the tenth time step

$$\|\Delta_2X_{10}\|_\infty + \| \sum_{j=1}^{10} \gamma_{X,j} \| = 0.667 + 1.5 = 2.167 > \epsilon_d$$

Observing this we are now aware of the need to refine the control polygon.
by subdivision. Note that [4] and [18] allow us to determine the amount of subdivision required so that an ambient isotopic approximation is guaranteed.

4.8 Conclusions

For dynamic visualization of molecular simulations it is important to ensure that the rendered curve and the underlying spline are ambient isotopic at each time step. That global bounds on these perturbations can be exceeded if only local perturbations are executed is obvious, the performance imperatives for dynamic visualization make such data-specific refinements relevant, as is explored here. This can be achieved by keeping track of changes to the second centered differences and applying further subdivision as required. The above analysis was performed for B-spline curves, the surface case was not pursued but we expect that the results can be extended to B-spline surfaces easily.

The molecules modeled certainly have 3-dimensional structure that is not captured by the 1-dimensional spline models. The reduction in dimension was chosen to support the performance demands of dynamic visualization of an ongoing simulation producing petabytes of output, while still being able to capture essential topological characteristics.

A similar reduction of dimension was undertaken to simplify engineering design studies [8]. The user identifies boundaries, that are modeled as 1-dimensional curves, as abstractions to convey design intent. This low-order geometry affords interactive manipulation and constraint satisfaction.
Emerging VR techniques rely upon hand and finger gestures to express design variations. It would be desirable to adapt such gestures to interactive steering of these molecular simulations, providing further opportunities to share these research perspectives. Indeed, some of the required emphasis on graphics manipulation is being pursued, concurrently, under collaborative research [10] for gesture based editing during production of computer animations in engineering design.
5. NUMERICAL TECHNIQUES FOR ISOTOPIC APPROXIMATION

We have seen in previous chapters that subdivision is commonly used as an approximation technique for Bézier splines. This chapter investigates numerical techniques to complement subdivision to preserve ambient isotopy. We compute a neighborhood where the curve and its PL approximant under subdivision have the same embedding. Furthermore, many perturbations within this neighborhood maintain the initial embedding. The static approximation guarantees are important for graphics and the dynamic assurances are crucial for animation. The popular approach of using Newton’s method to solve the associated system of equations of degree $2n - 1$ provides no guarantee of convergence [32]. We present an approximating numerical technique that reduces these equations to quadratics which can be solved in closed form.
5.1 MSD and Tubular Neighborhoods

In this section we define the concepts of pipe surfaces and minimum separation distance and relate these concepts to topologically faithful approximations of splines.

**Definition 5.1.1.** A pipe surface associated with a given curve \( c \) is defined as the envelope of the one parameter set of congruent spheres centered at the points of \( c \).

A pipe surface is said to be nonsingular if it does not self intersect. Recall that in a metric space, a set \( V \) is a neighborhood of a point \( p \) if there exists an open ball with center \( p \) and radius \( r > 0 \) that is contained in \( V \). Define a tubular neighborhood of \( c \) to consist of the interior points of a nonsingular pipe surface. Then specified perturbations of \( c \) within this neighborhood are ambient isotopic to \( c \) \([15]\). The non-singularity of a pipe surface with a sufficiently smooth spine curve depends only on its radius \([15]\). The radius of a non-singular pipe surface depends on the maximal curvature (a well treated calculus problem) and the separation distance between points on different parts of its spine curve – a concept we shall discuss presently.

**Definition 5.1.2.** For a non-self-intersecting parametric curve, \( c \), where
\[
c : [0, 1] \rightarrow \mathbb{R}^3
\]
for distinct values \( s, t \in [0, 1] \) the line segment \([c(s), c(t)]\) is said to be a double normal if it is normal to \( c \) at both of the end points \( c(s) \) and \( c(t) \).
Remark 5.1.1 follows similar previous work [15].

**Remark 5.1.1.** If $c$ is a closed curve $s, t$ should be distinct values in $[0, 1)$.

The minimum distance between a point $p$ and a curve $c(t)$ is determined by the shortest vector $c(t) - p$ for some $t$ so that the vector is perpendicular to the curve at $c(t)$, i.e. $(c(t) - p) \cdot c'(t) = 0$. Generalizing this view we can consider a double normal of a non-self-intersecting curve $c$ to be any line segment $[c(s), c(t)]$ of nonzero length that satisfies the equations

$$[c(s) - c(t)] \cdot c'(s) = 0 \quad (5.1)$$

$$[c(s) - c(t)] \cdot c'(t) = 0 \quad (5.2)$$

**Definition 5.1.3.** The minimum separation distance (MSD) for a given curve $c$ is defined to be the minimum of all lengths of all double normals on $c$ and is denoted by $\sigma$.

**Remark 5.1.2.** Note that MSD is also referred to a global separation distance.

To compute the MSD for a given curve we first compute the set $M$ which consists of all $(s, t) \in [0, 1] \times [0, 1]$ that satisfy (5.1) and (5.2). Then

$$\sigma = \min_{(s, t) \in M} \|c(s) - c(t)\|.$$ 

Since (5.1) and (5.2) are bivariate polynomials of degree $2n - 1$, theoretically,
we could use algebraic techniques to solve the system [25]. However this approach has known algorithmic difficulties [25]. Hence alternative numerical techniques need to be considered.

5.2 Newton’s Method

Since computing MSD involves minimizing a functional one tends to think of Newton’s method as a viable strategy. This is the popular approach [18, 21].

5.2.1 Newton Iterations

Define $F_i : [0, 1]^2 \rightarrow \mathbb{R}$ for $i = 1, 2$ as follows

$$F_1(s, t) = [c(s) - c(t)] \cdot c'(s)$$

$$F_2(s, t) = [c(s) - c(t)] \cdot c'(t)$$
We want to find all \((s, t) \in [0, 1] \times [0, 1]\) such that \(F_i(s, t) = 0, \ i = 1, 2\).

Define \(F : [0, 1]^2 \to \mathbb{R}^2\) by

\[
F(s, t) = \begin{bmatrix} F_1(s, t) \\ F_2(s, t) \end{bmatrix}
\]

A Newton iteration for this system is given by

\[
\begin{bmatrix} s_{i+1} \\ t_{i+1} \end{bmatrix} = \begin{bmatrix} s_i \\ t_i \end{bmatrix} - J^{-1}(s_i, t_i) \begin{bmatrix} F_1(s_i, t_i) \\ F_2(s_i, t_i) \end{bmatrix}
\]

where \(J^{-1}(s_i, t_i)\) is the inverse Jacobian matrix. The Jacobian matrix is given by

\[
J(s, t) = \begin{bmatrix} \frac{\partial F_1}{\partial s} & \frac{\partial F_1}{\partial t} \\ \frac{\partial F_2}{\partial s} & \frac{\partial F_2}{\partial t} \end{bmatrix}
\]

where

\[
\frac{\partial F_1}{\partial s} = [c(s) - c(t)] \cdot c''(s) + c'(s) \cdot c'(s),
\]

\[
\frac{\partial F_1}{\partial t} = (-c'(t) \cdot c'(s)),
\]

\[
\frac{\partial F_2}{\partial t} = [c(s) - c(t)] \cdot c''(t) + c'(t) \cdot c'(t),
\]

\[
\frac{\partial F_2}{\partial s} = (c'(s) \cdot c'(t)).
\]
The inverse Jacobian matrix is given by

\[ J(s, t)^{-1} = \frac{1}{W} \begin{bmatrix} \frac{\partial F_2}{\partial t} & -\frac{\partial F_1}{\partial t} \\ -\frac{\partial F_2}{\partial s} & \frac{\partial F_1}{\partial s} \end{bmatrix}, \]

where

\[ W = \frac{\partial F_1}{\partial s} \frac{\partial F_2}{\partial t} - \frac{\partial F_1}{\partial t} \frac{\partial F_2}{\partial s}. \]

Note that \( W \) is a degree \( 4n - 4 \) polynomial function. Observe that if

\[ c''(s) = c''(t) = 0 \]

then \( W = 0. \)

### 5.2.2 Algorithmic Difficulties

For this approach we need to select a number of seeds and retain the results of all convergent sequences as approximate double normal line segments. Then the estimate of MSD, denoted \( \hat{\sigma} \), is the shortest of these. There are two concerns with this method:

1. Can we guarantee convergence?

2. Even if we can assume convergence, can we guarantee that we can approximate all double normals for a given curve? Omitting the shortest double normal would obviously be problematic.
In theory, the Kantorovich theorem could be employed to identify the existence of roots and guarantee convergence [32]. However, such an approach requires that the Jacobian matrix is non-singular. To test for this involves finding the roots of the polynomial \( W \), which is possibly more difficult than the original problem.

The following example illustrates that even a relatively unsophisticated curve can be pathological for Newton’s method.

**Example 5.2.1.** Consider the composite cubic Bézier curve, denoted \( c(t) \), consisting of eight subcurves with the following control points:

- \((0, 1), (1, 1), (2, 1), (3, 1)\)
- \((3, 1), (3, 6), (8, 6), (8, 1)\)
- \((8, 1), (8, .5), (8, .25), (8, 0)\)
- \((8, 0), (8, −5), (3, −5), (3, 0)\)
- \((3, 0), (2, 0), (1, 0), (0, 0)\)
- \((0, 0), (0, −5), (−5, −5), (−5, 0)\)
- \((−5, 0), (−5, .25), (−5, .5), (−5, 1)\)
- \((−5, 1), (−5, 6), (0, 6), (0, 1)\).

This control polygon and curve is depicted in Figure 5.2. Notice that

\[
c''(t) = 0
\]

on the first, third, fifth and seventh subcurves. Newton iterations cannot be
performed on these subcurves due to the singularity of the inverse Jacobian. An implementation of Newton’s method could either return an error or skip regions where the inverse Jacobian is singular. The latter option in this case returns \( \hat{\sigma} = 5 \), when in fact \( \sigma = 1 \) (see the bottom of Figure 5.3).

![Fig. 5.2: Control polygon and corresponding curve.](image)

5.3 Discrete Approximation Method

We shall now examine how we can make use of the convergence properties of the PL control polygon to the curved geometry to approximate MSD.
5.3.1 Convergence of PL Structures

The process of subdivision applied to a curve \(c\) produces a refined control polygon that can be used as a PL graphics representation of \(c\) (Figure 5.4). As we have seen in previous chapters, there is an existing result that allows us to compute the number of subdivisions required so that we are within a specified tolerance of the curve [23]. For the sake of convenience we restate the result here.

Given a degree \(n\) Bézier curve \(c(t)\) with control points \(X = \{p_0, \ldots, p_n\}\), then after \(m\) uniform subdivisions the maximal Hausdorff distance between the control polygon and the curve is given by

\[
\left(\frac{1}{2}\right)^{2m} \|\Delta_2 X\|_\infty N(\infty)(n).
\]
Fig. 5.4: Subsequent subdivisions for approximation.

Here \( N_\infty(n) = \frac{\lceil n/2 \rceil \lfloor n/2 \rfloor}{2n} \). Subdividing \( m \) times guarantees that the PL structure is within a specified tolerance \( \epsilon \) where

\[
m = \left\lfloor -\frac{1}{2} \log_2 \left( \frac{\epsilon}{\|\Delta_2 X\|_\infty N_\infty(n)} \right) \right\rfloor
\]

We have seen, in Chapter 3, that analogous results hold for the derivative curve.
These results suggest that it is possible to attain an approximation for MSD within specified error bounds by using a refined PL approximation of the curve (Figure 5.5).

Fig. 5.5: Use PL structure to approximate MSD.

5.3.2 Vector Perturbations

Since the MSD equations (5.1 and 5.2) involve dot products it is crucial for this approach to be able to quantify errors introduced by perturbing the vectors involved. In this section we present some technical lemmas that
address this.

We will first bound the dot product of vectors in $\mathbb{R}^2$. Then, using the fact that any two vectors in $\mathbb{R}^3$ can be rotated into the $xy$-plane without altering the magnitude of their dot product, we provide a bound on dot products in $\mathbb{R}^3$.

**Lemma 5.3.1.** Given $u, v \in \mathbb{R}^2$ such that $u \cdot v = 0$. If $\tilde{u} = u + \delta_u$ and $\tilde{v} = u + \delta_v$, where $\delta_u, \delta_v \in \mathbb{R}^2$ with $\|\delta_u\| \leq \epsilon_u$ and $\|\delta_v\| \leq \epsilon_v$ where

$$\epsilon_u + \epsilon_v \leq \|\tilde{u} - \tilde{v}\|.$$

Then

$$\|R(-\phi)\tilde{v} - v\| \leq \epsilon_v + \frac{\epsilon_u}{\cos(\phi/2)}$$

where $R(-\phi)$ is the rotation taking $\tilde{u}$ to the $x$-axis and is given by

$$R(-\phi) = \begin{bmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{bmatrix} \quad \text{where} \quad \phi = \sin^{-1}(\epsilon_u/\|\tilde{u}\|).$$

**Proof.** Without loss of generality we consider the case where

$$u = (u, 0) \quad \text{and} \quad v = (0, v),$$

and the perturbed points $\tilde{u}$ and $\tilde{v}$ lie in the first quadrant of the $xy$-plane.
Assume the worst case i.e.,

$$\mathbf{\tilde{u}} = (u, \epsilon_u) \text{ and } \mathbf{\tilde{v}} = (\epsilon_v, v).$$

Let $\phi$ denote the angle between $\mathbf{\tilde{u}}$ and the positive $x$-axis, i.e. $\phi = \sin^{-1}(\epsilon_u/\|\mathbf{\tilde{u}}\|).$

Let $\alpha$ denote the straight line distance travelled by $\mathbf{\tilde{u}}$ being rotated to the $x$-axis via $R(-\phi).$ Observe that

$$\|\mathbf{v} - R(-\phi)\mathbf{\tilde{v}}\| = \epsilon_v + \alpha.$$

To compute $\alpha,$ consider the isosceles triangle formed by $(0, 0), \mathbf{\tilde{u}} \text{ and } R(-\phi)\mathbf{\tilde{u}},$ the internal angles are given by $\phi \text{ and } \rho$ where $\rho = \frac{\pi - \phi}{2}$ (See Figure 5.6).

Considering the internal right triangle (see Figure 5.6) it is easy to see that
\[ \sin \rho = \frac{\epsilon_u}{\alpha}, \text{ so,} \]
\[ \alpha = \frac{\epsilon_u}{\sin \rho} = \frac{\epsilon_u}{\sin \left( \frac{\pi - \phi}{2} \right)} = \frac{\epsilon_u}{\cos (\phi/2)}. \]

Define \( R_{u,v} : \mathbb{R}^3 \rightarrow \mathbb{R}^2 \) to be the composite rotation of the vectors \( u \) and \( v \) to the \( xy \)-plane taking \( u \) to \((\|u\|, 0)\). For any \( w \in \mathbb{R}^2 \), let \([w]_y\) denote the \( y \)-coordinate of \( w \).

**Lemma 5.3.2.** Given \( u, v \in \mathbb{R}^3 \) such that \( u \cdot v = 0 \). If \( \tilde{u} = u + \delta_u \) and \( \tilde{v} = u + \delta_v \), where \( \delta_u, \delta_v \in \mathbb{R}^3 \) with \( \|\delta_u\| \leq \epsilon_u \) and \( \|\delta_v\| \leq \epsilon_v \) where

\[ \epsilon_u + \epsilon_v \leq \|\tilde{u} - \tilde{v}\|, \|u\| \geq 1 \text{ and } \|v\| \geq 1. \]

Then

\[ \frac{|\tilde{u} \cdot \tilde{v}|}{\|\tilde{u}\| \|\tilde{v}\|} \leq \frac{r}{\sqrt{r^2 + v^2}}, \]

where

\[ v = [R_{\tilde{u}v}(\tilde{v})]_y, \quad r = \epsilon_v + \frac{\epsilon_u}{\cos(\phi/2)} \text{ and } \phi = \sin^{-1}(\epsilon_u/\|\tilde{u}\|). \]
Proof. It is clear that,
\[
\tilde{u} \cdot \tilde{v} = R_{\tilde{u}\tilde{v}}(\tilde{u}) \cdot R_{\tilde{u}\tilde{v}}(\tilde{v}) \\
= (\|\tilde{u}\|, 0) \cdot R_{\tilde{u}\tilde{v}}(\tilde{v}).
\]

Applying Lemma 5.3.1 we see that
\[
\frac{(\|\tilde{u}\|, 0) \cdot R_{\tilde{u}\tilde{v}}(\tilde{v})}{\|(\|\tilde{u}\|, 0)\| \cdot \|R_{\tilde{u}\tilde{v}}(\tilde{v})\|} \leq \frac{(r, v) \cdot (\|u\|, 0)}{\|(r, v)\| \cdot (\|u\|, 0)\|} \\
= \frac{r}{\sqrt{r^2 + v^2}}.
\]

\[\square\]

5.3.3 Notation

We now introduce some notation that will be used for the remainder of this chapter.

After \( m \) subdivisions we have \( 2^m \) sub-control polygons each associated with a unique interval of the form \( I_i = [i/2^m, (i + 1)/2^m] \) for \( i = 0, 1, \cdots, 2^m - 1 \). Denote the sub-control polygon associated with \( I_i \) by

\[ C_i = \{ \ell_m(s) | s \in I_i \}. \]
Similarly denote each sub-control polygon of the derivative curve by

\[ C'_i = \{ D_m(s) | s \in I_i \}. \]

Let \( d_H(c, C_i) \) denote the maximal Hausdorff distance between the subcurve \( c \) defined on \( I_i \) and \( C_i \) and \( d_H(c', C'_i) \) denote the maximal Hausdorff distance between the derivative subcurve \( c' \) defined on \( I_i \) and \( C'_i \).

### 5.3.4 Quadratic Approximation

Using the established bounds on dot products and curve approximation, it is possible to approximate the MSD equations within arbitrary precision with quadratic forms. This is desirable due to the numerical stability of low degree polynomials. Also, these quadratic equations can be easily solved using elementary calculus.

Define the function \( F : [0, 1]^2 \to \mathbb{R} \) by

\[
F(s, t) = \frac{\| c(s) - c(t) \|}{\| c(s) - c(t) \|} \cdot \frac{c'(s)}{\| c'(s) \|}.
\]

At a double normal \( F(s, t) = F(t, s) = 0 \). Define a quadratic approximation of \( F(s, t) \) by the function \( Q_m : [0, 1]^2 \to \mathbb{R} \) by

\[
Q_m(s, t) = \frac{\ell_m(s) - \ell_m(t)}{\| \ell_m(s) - \ell_m(t) \|} \cdot \frac{D_m(s)}{\| D_m(s) \|}.
\]
Say \((s_\sigma, t_\sigma)\) yields a double normal i.e. \(F(s_\sigma, t_\sigma) = F(t_\sigma, s_\sigma) = 0\). Note that

\[c(s_\sigma) - c(t_\sigma) = \ell_m(s_\sigma) - \ell_m(t_\sigma) + \delta_1,\]

where \(\delta_1 \in \mathbb{R}^3\) and

\[\|\delta_1\| \leq \mathcal{H}(c, C_i) + \mathcal{H}(c, C_j),\]

here \(s_\sigma \in I_i\) and \(t_\sigma \in I_j\).

Also,

\[c'(s_\sigma) = D_m(s_\sigma) + \delta_2,\]

where \(\delta_2 \in \mathbb{R}^3\) and

\[\|\delta_2\| \leq \mathcal{H}(c', C'_i).\]

Let

\[\tilde{u} = \ell_m(s_\sigma) - \ell_m(t_\sigma),\]

\[\tilde{v} = D_m(s_\sigma),\]

\[\epsilon_u = \mathcal{H}(c, C_i) + \mathcal{H}(c, C_j)\]

and

\[\epsilon_v = \mathcal{H}(c', C'_i).\]

We define the composite rotation \(\mathcal{R}_{\tilde{u}\tilde{v}}\) in three steps:

1. Rotate \(\tilde{u}\) into the \(xy\)-plane.
2. Rotate \(\tilde{u}\) to \((\|\tilde{u}\|, 0)\).
3. Rotate \(\tilde{v}\) to the \(xy\)-plane.
Note at each step the rotations are applied to both vectors preserving the acute angle between them. Applying Lemma 5.3.2 gives

\[ |Q_m(s_\sigma, t_\sigma)| \leq \frac{r}{\sqrt{r^2 + v^2}} \]

It is clear that we can make an analogous statement for \(|Q_m(t_\sigma, s_\sigma)|\).

### 5.3.5 Estimating MSD

Consider a Bézier curve parametrized on \([0, 1]\). Let \(L_1\) and \(L_2\) be distinct lines segments of the control polygon after \(m\) subdivisions. Let \(I_a\) be the subinterval corresponding to \(L_1\) and \(I_b\) the subinterval corresponding to \(L_2\).

If wish to determine whether or not there is a double normal line segment in \(I_a \times I_b\) we can use the following approach. Rotate the line segment

\[ \{\ell_m(s) - \ell_m(t)|s \in I_a, t \in I_b\} \]

to the \(x\)-axis, namely to the points

\[ \{(0, \|\ell_m(s) - \ell_m(t)\|)|s \in I_a, t \in I_b\}. \]

Define \(\epsilon_{ab}\) as

\[ \epsilon_{ab} = \frac{r_{ab}}{\sqrt{r_{ab}^2 + v_{ab}^2}}, \]

70
where

\[ r_{ab} = d_H(c', C_a) + \frac{d_H(c', C_a)}{\phi_{\text{max}}}, \]

\[ \phi_{\text{max}} = \sin^{-1} \frac{d_H(c', C_a)}{\min_{(s,t) \in I_a \times I_b} \| \ell_m(s) - \ell_m(t) \|}, \]

and

\[ v_{ab} = \min_{s \in I_s} \| D_m(s) \|. \]

Define

\[ N_{ab} = \{(s,t) \in I_a \times I_b | Q_m(s,t) \leq \epsilon_{ab} \text{ and } |Q_m(s,t)| \leq \epsilon_{ab}\}. \]

So \( N_{ab} \) is the set of approximate double normals on \( I_a \times I_b \).

Let

\[ \mathcal{L}_{\text{min}}^{ab} = \min_{(s,t) \in N_{ab}} \| \ell(s) - \ell(t) \| \]

and

\[ \mathcal{L}_{\text{max}}^{ab} = \max_{(s,t) \in N_{ab}} \| \ell(s) - \ell(t) \| \]

Note that if the minimum separation distance, \( \sigma \), is realized at

\[ (s_\sigma, t_\sigma) \in I_a \times I_b, \]

then

\[ \mathcal{L}_{\text{min}}^{ab} - (d_H(c, C_a) + d_H(c, C_b)) \leq \sigma. \]
So $\hat{\sigma} = L^{ab}_{\min} - (d_H(c, C_a) + d_H(c, C_b))$ serves as a lower bound for $\sigma$. It is easy to see that

$$|\sigma - \hat{\sigma}| \leq L^{ab}_{\min} - L^{ab}_{\max} - 2(d_H(c, C_a) + d_H(c, C_b)).$$

5.3.6 Numerical Approximation

We have established that the degree $2n - 1$ equations 5.1 and 5.1 can be approximated by quadratic polynomials, the accuracy of which depends on the quality of the approximating control polygon. A brute force approach to estimating MSD would be to use the approach described in Section 5.3.5 for all pairs of line segments. This would require $\binom{M}{2}$, where $M$ is the number of line segments in the refined control polygon and returns an estimate of MSD with known error bounds.

Clearly such an approach involves many unnecessary comparisons. A culling technique to eliminate many comparisons remains the subject of future work.
6. CONCLUSION

The main theoretical contributions of this thesis are as follows:

• Rigorously proved the proportionality relationship between the order of subdivision (both uniform and non-uniform cases) and discrete differentiation of Bézier curves.

• Provided rigorous bounds for the approximation of Bézier hodograph curves.

• Defined an *apriori* expression to compute the number of subdivisions required to approximate a Bézier hodograph to within a specified tolerance.

• Provided an efficient strategy for dynamic graphics representations of spline curves that preserve crucial topological characteristics of the underlying curve.

• Derived a numerical technique to approximate the degree $2n - 1$ MSD equations with a quadratic expression within arbitrary precision.

• Outlined a technique to approximate MSD of a Bézier curve to within a specified tolerance.


