Arclength Approximation for Reference-Point Generation

by 5

Mohsen M. Madi

A thesis
presented to the University of Manitoba
in partial fulfilment of the
requirements for the degree of
Master of Science
in
Computer Science

Winnipeg, Manitoba, Canada, 1996

ISBN 0-612-13333-8

Name ______

Dissertation Abstracts International and *Masters Abstracts International* are arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation or thesis. Enter the corresponding four-digit code in the spaces provided.

Computer Science

SUBJECT TERM

0	9	8	4

SUBJECT CODE

UMI

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS

Architecture	0729
Art History	0377
Cinema	0900
Dance	0378
Fine Arts	0357
Information Science	0723
Journalism	0391
Library Science	0399
Mass Communications	0708
Music	0413
Speech Communication	0459
Theater	0465

EDUCATION

General	0515
Administration	0514
Adult and Continuing	0516
Agricultural	0517
Art	0273
Bilingual and Multicultural	0282
Business	0688
Community College	0275
Curriculum and Instruction	0727
Early Childhood	0518
Elementary	0524
Finance	0277
Guidance and Counseling	0519
Health	0680
Higher	0745
History of	0520
Home Economics	0278
Industrial	0521
Language and Literature	0279
Mathematics	0280
Music	0522
Philosophy of	0998
Physical	0523
Psychology	0525
Reading	0535
Religious	0527
Sciences	0714
Secondary	0533
Social Sciences	0534
Sociology of	0340
Special	0529
Teacher Training	0530
Technology	0710
Tests and Measurements	0288
Vocational	0747

LANGUAGE, LITERATURE AND LINGUISTICS

Language	
General	0679
Ancient	0289
Linguistics	0290
Modern	0291
Literature	
General	0401
Classical	0294
Comparative	0295
Medieval	0297
Modern	0298
African	0316
American	0591
Asian	0305
Canadian (English)	0352
Canadian (French)	0355
English	0593
Germanic	0311
Latin American	0312
Middle Eastern	0315
Romance	0313
Slavic and East European	0314

PHILOSOPHY, RELIGION AND THEOLOGY

Philosophy	0422
Religion	
General	0318
Biblical Studies	0321
Clergy	0319
History of	0320
Philosophy of	0322
Theology	0469

SOCIAL SCIENCES

American Studies	0323
Anthropology	
Archaeology	0324
Cultural	0326
Physical	0327
Business Administration	
General	0310
Accounting	0272
Banking	0770
Management	0454
Marketing	0338
Canadian Studies	0385
Economics	
General	0501
Agricultural	0503
Commerce-Business	0505
Finance	0508
History	0509
Labor	0510
Theory	0511
Folklore	0358
Geography	0366
Gerontology	0351
History	
General	0578
Ancient	0579
Medieval	0581
Modern	0582
Black	0328
African	0331
Asia, Australia and Oceania	0332
Canadian	0334
European	0335
Latin American	0336
Middle Eastern	0333
United States	0337
History of Science	0585
Law	0398
Political Science	
General	0615
International Law and Relations	0616
Public Administration	0617
Recreation	0814
Social Work	0452
Sociology	
General	0626
Criminology and Penology	0627
Demography	0938
Ethnic and Racial Studies	0631
Individual and Family Studies	0628
Industrial and Labor Relations	0629
Public and Social Welfare	0630
Social Structure and Development	0700
Theory and Methods	0344
Transportation	0709
Urban and Regional Planning	0999
Women's Studies	0453

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES

Agriculture	
General	0473
Agronomy	0285
Animal Culture and Nutrition	0475
Animal Pathology	0476
Food Science and Technology	0359
Forestry and Wildlife	0478
Plant Culture	0479
Plant Pathology	0480
Plant Physiology	0817
Range Management	0777
Wood Technology	0746
Biology	
General	0306
Anatomy	0287
Biostatistics	0308
Botany	0309
Cell	0379
Ecology	0329
Entomology	0353
Genetics	0369
Limnology	0793
Microbiology	0410
Molecular	0307
Neuroscience	0317
Oceanography	0416
Physiology	0433
Radiation	0821
Veterinary Science	0778
Zoology	0472
Biophysics	
General	0786
Medical	0760

EARTH SCIENCES

Biogeochemistry	0425
Geochemistry	0996
Geodesy	0370
Geology	0372
Geophysics	0373
Hydrology	0388
Mineralogy	0411
Paleobotany	0345
Paleoecology	0426
Paleontology	0418
Paleozoology	0985
Palynology	0427
Physical Geography	0368
Physical Oceanography	0415

HEALTH AND ENVIRONMENTAL SCIENCES

Environmental Sciences	0768
Health Sciences	
General	0566
Audiology	0300
Chemotherapy	0992
Dentistry	0567
Education	0350
Hospital Management	0769
Human Development	0758
Immunology	0982
Medicine and Surgery	0564
Mental Health	0347
Nursing	0569
Nutrition	0570
Obstetrics and Gynecology	0380
Occupational Health and Therapy	0354
Ophthalmology	0381
Pathology	0571
Pharmacology	0419
Pharmacy	0572
Physical Therapy	0382
Public Health	0573
Radiology	0574
Recreation	0575
Speech Pathology	0460
Toxicology	0383
Home Economics	0386

PHYSICAL SCIENCES

Pure Sciences

Chemistry	
General	0485
Agricultural	0749
Analytical	0486
Biochemistry	0487
Inorganic	0488
Nuclear	0738
Organic	0490
Pharmaceutical	0491
Physical	0494
Polymer	0495
Radiation	0754
Mathematics	0405
Physics	
General	0605
Acoustics	0986
Astronomy and Astrophysics	0606
Atmospheric Science	0608
Atomic	0748
Electronics and Electricity	0607
Elementary Particles and High Energy	0798
Fluid and Plasma	0759
Molecular	0609
Nuclear	0610
Optics	0752
Radiation	0756
Solid State	0611
Statistics	0463

Applied Sciences

Applied Mechanics	0346
Computer Science	0984
Engineering	
General	0537
Aerospace	0538
Agricultural	0539
Automotive	0540
Biomedical	0541
Chemical	0542
Civil	0543
Electronics and Electrical	0544
Heat and Thermodynamics	0348
Hydraulic	0545
Industrial	0546
Marine	0547
Materials Science	0794
Mechanical	0548
Metallurgy	0743
Mining	0551
Nuclear	0552
Packaging	0549
Petroleum	0765
Sanitary and Municipal	0554
System Science	0790
Geotechnology	0428
Operations Research	0796
Plastics Technology	0795
Textile Technology	0994

PSYCHOLOGY

General	0621
Behavioral	0384
Clinical	0622
Developmental	0620
Experimental	0623
Industrial	0624
Personality	0625
Physiological	0989
Psychobiology	0349
Psychometrics	0632
Social	0451

Nom ______________________________

Dissertation Abstracts International est organisé en catégories de sujets. Veuillez s.v.p. choisir le sujet qui décrit le mieux votre thèse et inscrivez le code numérique approprié dans l'espace réservé ci-dessous.

______________________________ □□□□ U·M·I

SUJET CODE DE SUJET

Catégories par sujets

HUMANITÉS ET SCIENCES SOCIALES

COMMUNICATIONS ET LES ARTS

Architecture	0729
Beaux-arts	0357
Bibliothéconomie	0399
Cinéma	0900
Communication verbale	0459
Communications	0708
Danse	0378
Histoire de l'art	0377
Journalisme	0391
Musique	0413
Sciences de l'information	0723
Théâtre	0465

ÉDUCATION

Généralités	515
Administration	0514
Art	0273
Collèges communautaires	0275
Commerce	0688
Économie domestique	0278
Éducation permanente	0516
Éducation préscolaire	0518
Éducation sanitaire	0680
Enseignement agricole	0517
Enseignement bilingue et multiculturel	0282
Enseignement industriel	0521
Enseignement primaire.	0524
Enseignement professionnel	0747
Enseignement religieux	0527
Enseignement secondaire	0533
Enseignement spécial	0529
Enseignement supérieur	0745
Évaluation	0288
Finances	0277
Formation des enseignants	0530
Histoire de l'éducation	0520
Langues et littérature	0279
Lecture	0535
Mathématiques	0280
Musique	0522
Orientation et consultation	0519
Philosophie de l'éducation	0998
Physique	0523
Programmes d'études et enseignement	0727
Psychologie	0525
Sciences	0714
Sciences sociales	0534
Sociologie de l'éducation	0340
Technologie	0710

LANGUE, LITTÉRATURE ET LINGUISTIQUE

Langues	
Généralités	0679
Anciennes	0289
Linguistique	0290
Modernes	0291
Littérature	
Généralités	0401
Anciennes	0294
Comparée	0295
Mediévale	0297
Moderne	0298
Africaine	0316
Américaine	0591
Anglaise	0593
Asiatique	0305
Canadienne (Anglaise)	0352
Canadienne (Française)	0355
Germanique	0311
Latino-américaine	0312
Moyen-orientale	0315
Romane	0313
Slave et est-européenne	0314

PHILOSOPHIE, RELIGION ET THÉOLOGIE

Philosophie	0422
Religion	
Généralités	0318
Clergé	0319
Études bibliques	0321
Histoire des religions	0320
Philosophie de la religion	0322
Théologie	0469

SCIENCES SOCIALES

Anthropologie	
Archéologie	0324
Culturelle	0326
Physique	0327
Droit	0398
Économie	
Généralités	0501
Commerce-Affaires	0505
Économie agricole	0503
Économie du travail	0510
Finances	0508
Histoire	0509
Théorie	0511
Études américaines	0323
Études canadiennes	0385
Études féministes	0453
Folklore	0358
Géographie	0366
Gérontologie	0351
Gestion des affaires	
Généralités	0310
Administration	0454
Banques	0770
Comptabilité	0272
Marketing	0338
Histoire	
Histoire générale	0578
Ancienne	0579
Médiévale	0581
Moderne	0582
Histoire des noirs	0328
Africaine	0331
Canadienne	0334
États-Unis	0337
Européenne	0335
Moyen-orientale	0333
Latino-américaine	0336
Asie, Australie et Océanie	0332
Histoire des sciences	0585
Loisirs	0814
Planification urbaine et régionale	0999
Science politique	
Généralités	0615
Administration publique	0617
Droit et relations internationales	0616
Sociologie	
Généralités	0626
Aide et bien-àtre social	0630
Criminologie et établissements pénitentiaires	0627
Démographie	0938
Études de l' individu et de la famille	0628
Études des relations interethniques et des relations raciales	0631
Structure et développement social	0700
Théorie et méthodes.	0344
Travail et relations industrielles	0629
Transports	0709
Travail social	0452

SCIENCES ET INGÉNIERIE

SCIENCES BIOLOGIQUES

Agriculture	
Généralités	0473
Agronomie.	0285
Alimentation et technologie alimentaire	0359
Culture	0479
Élevage et alimentation	0475
Exploitation des péturages	0777
Pathologie animale	0476
Pathologie végétale	0480
Physiologie végétale	0817
Sylviculture et faune	0478
Technologie du bois	0746
Biologie	
Généralités	0306
Anatomie	0287
Biologie (Statistiques)	0308
Biologie moléculaire	0307
Botanique	0309
Cellule	0379
Écologie	0329
Entomologie	0353
Génétique	0369
Limnologie	0793
Microbiologie	0410
Neurologie	0317
Océanographie	0416
Physiologie	0433
Radiation	0821
Science vétérinaire	0778
Zoologie	0472
Biophysique	
Généralités	0786
Medicale	0760

SCIENCES DE LA TERRE

Biogéochimie	0425
Géochimie	0996
Géodésie	0370
Géographie physique	0368
Géologie	0372
Géophysique	0373
Hydrologie	0388
Minéralogie	0411
Océanographie physique	0415
Paléobotanique	0345
Paléoécologie	0426
Paléontologie	0418
Paléozoologie	0985
Palynologie	0427

SCIENCES DE LA SANTÉ ET DE L'ENVIRONNEMENT

Économie domestique	0386
Sciences de l'environnement	0768
Sciences de la santé	
Généralités	0566
Administration des hipitaux	0769
Alimentation et nutrition	0570
Audiologie	0300
Chimiothérapie	0992
Dentisterie	0567
Développement humain	0758
Enseignement	0350
Immunologie	0982
Loisirs	0575
Médecine du travail et thérapie	0354
Médecine et chirurgie	0564
Obstétrique et gynécologie	0380
Ophtalmologie	0381
Orthophonie	0460
Pathologie	0571
Pharmacie	0572
Pharmacologie	0419
Physiothérapie	0382
Radiologie	0574
Santé mentale	0347
Santé publique	0573
Soins infirmiers	0569
Toxicologie	0383

SCIENCES PHYSIQUES

Sciences Pures

Chimie	
Genéralités	0485
Biochimie	487
Chimie agricole	0749
Chimie analytique	0486
Chimie minérale	0488
Chimie nucléaire	0738
Chimie organique	0490
Chimie pharmaceutique	0491
Physique	0494
PolymÇres	0495
Radiation	0754
Mathématiques	0405
Physique	
Généralités	0605
Acoustique	0986
Astronomie et astrophysique	0606
Electronique et électricité	0607
Fluides et plasma	0759
Météorologie	0608
Optique	0752
Particules (Physique nucléaire)	0798
Physique atomique	0748
Physique de l'état solide	0611
Physique moléculaire	0609
Physique nucléoire	0610
Radiation	0756
Statistiques	0463

Sciences Appliqués Et Technologie

Informatique	0984
Ingénierie	
Généralités	0537
Agricole	0539
Automobile	0540
Biomédicale	0541
Chaleur et ther modynamique	0348
Conditionnement (Emballage)	0549
Génie aérospatial	0538
Génie chimique	0542
Génie civil	0543
Génie électronique et électrique	0544
Génie industriel	0546
Génie mécanique	0548
Génie nucléaire	0552
Ingénierie des systèmes	0790
Mécanique navale	0547
Métallurgie	0743
Science des matériaux	0794
Technique du pétrole	0765
Technique minière	0551
Techniques sanitaires et municipales	0554
Technologie hydraulique	0545
Mécanique appliquée	0346
Géotechnologie	0428
Matières plastiques (Technologie)	0795
Recherche opérationnelle	0796
Textiles et tissus (Technologie)	0794

PSYCHOLOGIE

Généralités	0621
Personnalité	0625
Psychobiologie	0349
Psychologie clinique	0622
Psychologie du comportement	0384
Psychologie du développement	0620
Psychologie expérimentale	0623
Psychologie industrielle	0624
Psychologie physiologique	0989
Psychologie sociale	0451
Psychométrie	0632

ARCLENGTH APPROXIMATION FOR REFERENCE-POINT GENERATION

BY

MOHSEN M. MADI

A Thesis/Practicum submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

I hereby declare that I am the sole author of this thesis.

I authorize the University of Manitoba to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize the University of Manitoba to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

The University of Manitoba requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

Acknowledgment

In the name of Allah, the Merciful, the Compassionate. Praise be to Him that He made me Submissive to Him. Thanks are to Him that he sent us the prophets to guide us to the straight path, especially for those of determination: Noah, Ibraham, Moses, Jesus, and Mohammad, may peace and bless of Allah be upon them and the rest of the prophets.

All the love to my mother, my father, who are always in my heart and mind, and whose my accomplishment in life is non but the result of Allah's answering to their prayers and supplication to Him for my success.

To my dear wife, whom whatever words of thanks I say, they would not be enough to do justice to her, and whom I ask Allah, that he makes her my wife in the Hereafter, as he made her be my wife in the Herein. And to our children, Ahmed & Einas, whom just remembering them, makes me know what I want to do in life.

To my supervisor, Dr. Desmond Walton, whose *amazing* patience, *infinite* help in all aspects, and *brotherly* support made me achieve what I did not know I had in me. I can never thank him enough.

I thank Dr. Dereck Meek for all the quality time and work he provided me with, and whose points of inspiration here and there helped shape up my work extensively. Special thanks to Dr. Rida Farouki, whose guiding responses and quality work helped my know the dimensions of our field. And to Dr. Robert Thomas, for providing the final touches to the work accomplished here.

Abstract

In applications such as CNC machining, highway and railway design, manufacturing industry, and animation, there is an ongoing need to systematically generate sets of reference points with prescribed arclengths along parametric curves, with considerable accuracy and real-time performance. In the process, mechanisms to result in a parameter set that yield the coordinates of the reference points along the curve $\mathbf{Q}(t) = \{x(t), y(t)\}$ are sought.

The presence of arclength parameterizable expressions usually yields the parameter set that is necessary to generate the reference points, however, for typical design curves, such an expression is often not available in closed form. It is desirable to find efficient ways to compensate for lack of arclength parameterization.

In this dissertation, several analytical and numerical methods for approximating arclength parameterization are presented. These methods are examined for both accuracy and real-time processing requirements. The application of generating uniformly spaced reference points along the paths of several curves is chosen for the purposes of illustration, and bench marking among the presented methods.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In computer-aided design (CAD), computer-aided manufacturing (CAM), and computer-aided geometric design (CAGD) applications, there is a need to accurately and efficiently generate a set of *reference points* distributed according to prescribed dimensions along curves. Reference points are points on a curve such that the arclength between neighboring points matches prescribed values. In addition, technical applications often require that reference points be equally spaced in terms of the arclength. Following is a representative sample of these applications.

- *CNC machining.* In computer numerical control (CNC) machining, computers with CAD systems are instructed to produce a set of reference points along the path of machine parts according to specific pre-fed dimensions, the final product may be expected to be affixed to other parts by bolts through adjacent holes in locations marked by reference points [23][40].

- *Highway and railway design.* In these areas of civil and mechanical engineering, engineers may need to determine the locations of $N+1$ reference stakes, where any two lie k meters apart along a designed curved roadway to facilitate highway construction and cost measurements.

- *Manufacturing industry.* In the airplane manufacturing industry for example, thousands of uniformly spaced reference points that are to lie along smooth paths in a fuselage have to be computed; these reference points serve as the locations for holes to be made to affix complementing parts by bolts. Uniform distribution is desired here to distribute tension evenly across the fuselage.

- *Graphical simulation and animation.* In order for animated objects to appear realistic during simulation, it is necessary to pre-determine intermediate positions the object is to appear at. It is desirable that these positions be uniformly spaced [30].

A first step towards the solution is to abstract the physical paths along objects of interest by parametric curves.

In order to efficiently calculate reference points along parametric curves, several problems have to be solved, usually in the following order:

(a) Obtain an expression for the arclength (in closed form if possible).

(b) Compute the total arclength.

(c) Re-parameterize the curve equation using the inverse of the arclength expression obtained in (a).

If a closed-form solution is available to the first item of the above, an *ideal* solution for reference points generation/location along the curve of interest is at hand.

Arclength re-parameterization is useful because the arclength is an intrinsic quantity of the curve and arclength parameterization is an intrinsic property of the curve [6][27]. Existence of the arclength expression in an analytical algebraic form greatly facilitates the application of several tasks that aid in curve analysis and design [47]. In order to generate a logically prescribed set of reference points along the *abstracted* curve, for example, the parametric equation obtained in (c) above is parameterized accordingly, e.g., uniform increments of the parameter value yield uniformly spaced reference points. In [23], this is known as *constant feedrate.* The example in the next subsection clarifies these points.

1.2 An Example

Consider the following parametric circle equation:

$$\mathbf{Q}(t) = r\{\sin(t), \cos(t)\}, \quad \text{where } \; 0 \leq t \leq 2\pi. \tag{1.1}$$

An arclength expression $s(t)$ is obtained as follows:

$$s(t) = \int_0^t \sqrt{r^2\left(\cos^2(\tau) + \sin^2(\tau)\right)}\, \mathrm{d}\tau = rt. \tag{1.2}$$

The total arclength $\mathcal{L}$ is computed by setting t to the upper limit in the above equation. This yields

$$\mathcal{L} = 2\pi r. \tag{1.3}$$

From (1.2), $t(s) = s/r$. Equation (1.1) is re-parameterized by the arclength to yield the following arclength re-parameterized equation:

$$\mathbf{Q}(s) = r\{\sin\left(\frac{s}{r}\right), \cos\left(\frac{s}{r}\right)\}, \quad \text{where } \; 0 \leq s \leq \mathcal{L}. \tag{1.4}$$

To generate a set of $N+1$ uniformly spaced reference points

$$\{\mathbf{p}_i \,|\, i = 0, 1, \ldots, N\}$$

along the circle path, (1.4) is evaluated with the set

$$\{s_i = i\Delta s \,|\, \Delta s = \mathcal{L}/N \text{ and } i = 0, 1, \ldots, N\}. \tag{1.5}$$

The parameterization of (1.4) is thus such that uniform increments of the parameter (as obtained in (1.5)) are actual uniform steps taken along the curve.

1.3 The Problem

The above example deliberately illustrates a case where all steps to be taken towards the ideal solution ((1.2)–(1.4)) were easy to compute and express in elementary functions. Equation (1.2) in particular, is analytically reducible into a manageable linear function that allowed easy extraction of the arclength s in terms of t and vice versa. In addition, it is important to note the relationship existing between t and s in the above example: It is a linear relationship where t is a *normalization* of s. This type of relationship is always present if the *curvature* is constant (such as in straight lines, circles, and helices) and allows an almost direct form of arclength parameterization. This simple relationship is unusual. Most curves modeling practical objects have variable curvature, and other than the fact that both s and t are monotone and increasing, no other guiding relationship may be assumed.

In general, the problem may be described as follows. Consider the following more general form of (1.2) to obtain the arclength:

$$s(t) = \int_0^t \|\mathbf{Q}'(\tau)\| \,\mathrm{d}\tau = \int_0^t \sqrt{\mathbf{Q}'(\tau) \cdot \mathbf{Q}'(\tau)} \,\mathrm{d}\tau. \tag{1.6}$$

For most curves $\mathbf{Q}(t)$, it is impossible to integrate (1.6) in an analytic fashion, and even when this is possible, deriving an expression for $t(s)$ from (1.6) may be impossible. As a result, (1.6) does not usually lend itself to a practical representation [40][47]. The ideal solution mentioned in Section 1.1 is therefore rarely possible; so it is necessary to search for numerical methods to calculate the arclength and to approximate an arclength re-parameterization.

1.4 Existing Methods

Existing methods that are available for generating reference points are mentioned next. Some of them are examined further in the succeeding chapters.

1.4.1 Direct Generation with "t"

In this method, the *dimensionless* parameter t, at which $\mathbf{Q}(t)$ is evaluated, is used to generate reference points along the curve. A set of $N+1$ uniformly spaced reference points may be approximated by evaluating $\mathbf{Q}(t)$ at points at the following parameter set:

$$\{t_i = i\Delta t \,|\, \Delta t = \frac{i}{N}, i = 0, 1, \ldots, N\}. \tag{1.7}$$

This method is simple and is sufficient for curves that possess constant curvature, but is unsuitable otherwise as it appears that reference points generated by $\{t_i\}$ cluster in regions of high curvature, and are sparse in regions of low curvature. Figure 1.1 illustrates these observations.

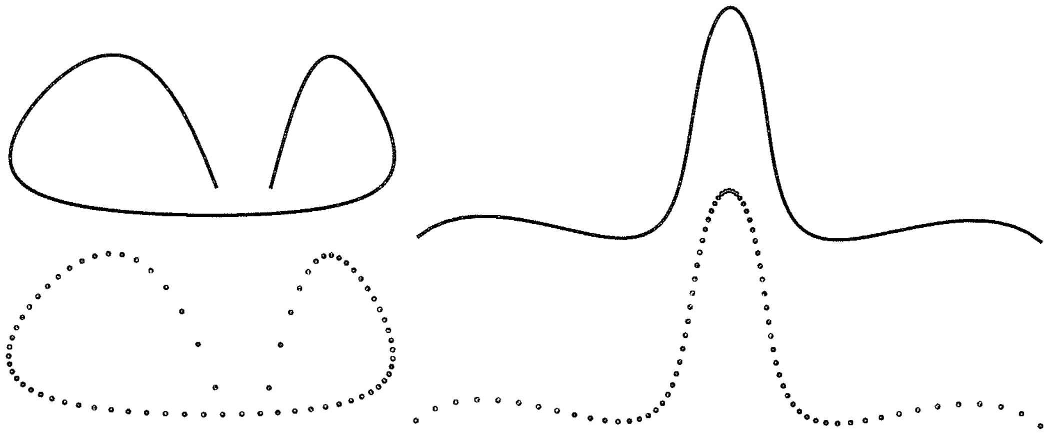

Figure 1.1: Two eighth-degree Bézier curves with reference points generated by a uniformly spaced set $\{t_i\}$, as defined in (1.7).

1.4.2 Sharpe & Thorne's Method

The method described by Sharpe and Thorne in [40] is a numerical technique that can accurately produce reference points at the prescribed offsets. However, it has a high computational cost associated with "extracting" the corresponding parametric value for each reference-point to be generated. Consider the following non-linear equation used to find the $N+1$ parameter values needed to generate $N+1$ reference points:

$$M(t) = \int_{t_{i-1}}^{t} \sqrt{\mathbf{Q}'(\tau) \cdot \mathbf{Q}'(\tau)}\, d\tau - \ell_i = 0, \quad i = 1..N, \tag{1.8}$$

where t_{i-1} is the parameter value corresponding to the last computed reference-point, the solution $t = t_i$ is the value corresponding to the next reference-point, and ℓ_i is the desired length, at which the next reference-point is to be generated.

In order to obtain t_i, a few Newton-Raphson iterations are applied:

$$\tau_j = \tau_{j-1} - \frac{M(\tau_{j-1})}{M'(\tau_{j-1})}, \quad \tau_0 = t_{i-1}, \quad j = 1, 2, \ldots, k, \tag{1.9}$$

where $M'(\tau_j)$ is the derivative of $M(\tau_j)$. The value of t_i is given by τ_k, where k is the number of iterations required for convergence to an acceptable accuracy.

For applications requiring real-time processing, or those requiring only an approximation of uniform spacing of reference points, this method is may be impractical.

1.4.3 Farouki's Method

Recently, Farouki developed a method that approximates arclength parameterization of Bézier curves [18]. The algorithm starts by first transforming the given polynomial $\mathbf{Q}(t)$ into a rational function $\tilde{\mathbf{Q}}(t)$ so that a set of weights $w_{i=0..n}$, n being the curve degree, is available for manipulation. The values of w_i are determined in such a way that the parameter t is the best approximation to the arclength parameterization. The algorithm is mathematically involved and is not as accurate as the Sharpe and Thorne method, nevertheless, it is fast enough for real-time processing requirements and should therefore be suitable for many speed-critical applications.

1.5 Special Curves

Although most curves in practical use do not have a closed-form expression available for (1.6), there are some exceptions, two of which are the Pythagorean-hodograph curves, and the clothoid (or Cornu spiral). They are mentioned next.

1.5.1 Pythagorean-hodograph Curves

Pythagorean-hodograph curves were introduced into the CAD and CAGD literature by Farouki & Shah [22]. Consider the following curve:

$$\mathbf{Q}(t) = \{x(t), y(t)\}. \tag{1.10}$$

$\mathbf{Q}(t)$ is said to have a Pythagorean-hodograph if its derivative $\mathbf{Q}'(t) = \{x'(t), y'(t)\}$ is such that there exists a parametric polynomial $\sigma(t)$ satisfying:

$$\sigma^2(t) \equiv x'^2(t) + y'^2(t). \tag{1.11}$$

These curves are convenient because their arclength may be obtained in closed form (polynomial) at any point on the curve. They also have the advantage of having rational offsets.

1.5.2 Clothoids

Clothoids are another family of curves which have a closed-form expression for the arclength, in addition to other properties described in [33][34][39][43]. However, clothoids also have limitations that make them impractical for general curve design purposes. For example, the position of any point on the clothoid is given in terms of integrals, whereas in CAD and CAGD applications, one usually prefers to work with polynomial curves.

1.6 Objectives

It is the objective of this dissertation to examine the problem of arclength calculation and arclength re-parameterization in detail, and to survey and offer alternate solutions for the steps in the ideal solution (steps (a), (b), and (c) from Section 1.1), that is:

- *Survey and propose methods for computing the arclengths of parametric curves.* The methods are generally numerical in nature. Some are highlighted as reliable arclength calculating methods.

- *Develop methods that approximate arclength parameterization.* The relationship between the arclength s and the parameter t is exploited, and an analytical interpolation method that approximates the intrinsic function $t(s)$ is developed, with application of generating uniformly spaced reference points.

The primary objective is to examine the feasibility of using a single functional entity (rather than a piecewise function such as a spline) to approximate $t(s)$. Not only is such an approximation convenient for reference-point generation, but it would also be valuable in analyzing properties of curves, e.g., the curvature is usually known as a function of t, but actually depends on s.

1.7 Organization

This thesis is divided into six chapters. Chapter 2 presents an overview of arclengths of some curves used in CAGD/CAD; it serves as a literature review and also provides the foundation necessary for the development of methods to be surveyed and proposed in later chapters. In Chapter 3, several methods for calculating the arclength are presented for different curves, and methods for approximating arclength parameterization are introduced. Chapter 4 serves as the application part of this thesis: reference-point generator algorithms are presented; experiments are carried out to illustrate capabilities of the proposed methods through examples and comparisons with other existing methods. Chapter 5 gives an insight into the cost of running the algorithms discussed in the Chapter 4, and shows how the method proposed in this thesis may be modified to produce accurate results. Finally, Chapter 6 concludes the thesis with a general summary of the work accomplished, and points out interesting related areas for future work.

This thesis focuses on planar curves because they are more convenient to visualize. The results extend easily to space curves as indicated in [22][26].

1.8 Notation

Following is a list of symbols and abbreviations used in this thesis. The lists are arranged in alphabetical order, and serve as a quick index for abbreviations particular to this thesis.

$\mathbb{E}^n$	n dimensional Euclidean space
I	Number of Simpson intervals (always even)
$\mathcal{L}$	The total arclength
$\Im$	Number of chords used to approximate a curve
$N+1$	Number of reference points
n	Curve degree
$\mathbf{Q}(t)$	A parametric Bézier curve
$\mathbb{R}$	The set of real numbers
$\mathbf{r}_i$	The *ith* reference-point
$\mathbb{Z}^+$	The set of positive integers

BPF	Basic parametric flow Algorithm
CAD	Computer-aided design
CAGD	Computer-aided geometric design
CAM	Computer-aided manufacturing
CC	Cumulative Chordlength Algorithm
CNC	Computer numerical control
F-OP	Farouki's optimal parameterization method
MPM	Modified proposed method
PM	Proposed method
PH	Pythagorean-hodograph
PHC	Pythagorean-hodograph Curve Algorithm
RP	Reference-point
RPG	Reference-point generator
ST	Sharpe & Thorne Algorithm

Chapter 2

Practical Curves

This chapter surveys some common curves used in CAGD/CAD with respect to the problems discussed in Chapter 1. The dissertation is focused on *parametric* curves that are represented in *Bézier* format. These representations provide powerful tools that have been proven to be practical in design and analysis of curves and surfaces [15][25][28].

The chapter is organized as follows: Section 2.1 gives a general introduction to Bézier curves of several degrees. It is emphasized how the problems of arclength calculation and arclength re-parameterization are more difficult as non-linearity is introduced. Section 2.2 discusses Pythagorean-hodograph (PH) curves. Section 2.3 gives a general definition of B-spline curves, with particular emphasis on uniform B-splines because of their simple relationship to Bézier curves. Finally, Section 2.4 concludes with a general summary, and discusses the problems of arclength calculation and locating reference points (by using/approximating arclength parameterization) with respect to the curves discussed.

2.1 Bézier curves

Bézier curves of any degree can be represented by special polynomials known as the *Bernstein* polynomials. Bézier curves are defined as:

$$\mathbf{Q}(t) = \sum_{i=0}^{n} \mathbf{p}_i B_{i,n}(t), \qquad 0 \leq t \leq 1, \tag{2.1}$$

where $B_{i,n}(t)$ are Bernstein basis functions:

$$B_{i,n}(t) = \begin{pmatrix} n \\ i \end{pmatrix} (1-t)^{n-i} t^i, \tag{2.2}$$

n denotes curve degree, and $\mathbf{p}_i \in \mathbb{E}^2$ are the Bézier points that constitute the guiding polygon of the curve.

Geometric properties of Bézier curves follow from the properties of Bernstein polynomials and can be found in [15][41]. Some of the properties of interest here are:

- Affine invariance: Transformation operations such as translation, rotation, or uniform scaling that are applied to the curve leave the ratios of the lengths of the control polygon legs, and the angles in the control polygon unchanged.
- Endpoint interpolation: That is: $\mathbf{Q}(0) = \mathbf{p}_0$ and $\mathbf{Q}(1) = \mathbf{p}_n$. The design of curves becomes more intuitive and is simplified because of this property.
- Convex-hull property. A Bézier curve always lies inside the convex-hull of its control polygon. This facilitates the study of the curve's behavior, such as knowledge of how the curve behaves away from the end points.

It is first shown how the problems of arclength calculation and arclength reparameterization are trivially solved for linear Bézier curves. These problems are more challenging for higher degree Bézier curves.

2.1.1 Linear Bézier Curves

Linear Béziers are straight lines in $\in \mathbb{E}^2$. Letting $n = 1$ in (2.1), the following is obtained:

$$\mathbf{p} = \mathbf{Q}(t) = \mathbf{p}_0(1 - t) + \mathbf{p}_1 t. \tag{2.3}$$

The points $\mathbf{p}_0$ and $\mathbf{p}_1$ are the end points of the line segment defined by $\mathbf{Q}(t)$, and $\mathbf{p}$ is a point on the line. Setting the parameter $t = 0$ causes $\mathbf{p}$ to equal $\mathbf{p}_0$, setting $t = 1$ causes $\mathbf{p}$ to equal $\mathbf{p}_1$, any other value for $t \in (0, 1)$ is an internal point on the line segment $\mathbf{Q}(t)$. This is illustrated in Figure 2.1.

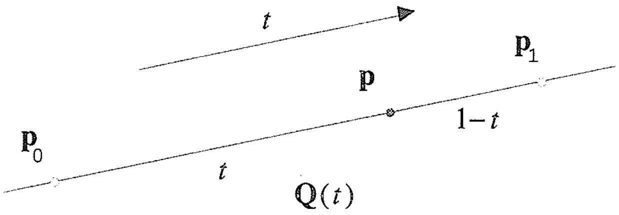

Figure 2.1: The two points $\mathbf{p}_0$ and $\mathbf{p}_1$ delimit the line segment defined by $\mathbf{Q}(t)$. The point $\mathbf{p}$ divides $\mathbf{Q}(t)$ in the ratio $t : 1{-}t$.

The point $\mathbf{p}$ divides the straight line segment defined by $\mathbf{Q}(t)$ in the ratio $t : 1{-}t$; such a point is referred to as a barycentric combination of the end points $\mathbf{p}_0$ and $\mathbf{p}_1$. In this regard, t is considered to be a *normalization* of the arclength. The following derivation shows the linear relationship between t and the arclength. Rearrangement of (2.3) gives:

$$\mathbf{p} = \mathbf{p}_0 - t(\mathbf{p}_1 - \mathbf{p}_0).$$

Solving for t gives:

$$t = \frac{||\mathbf{p} - \mathbf{p}_0||}{||\mathbf{p}_1 - \mathbf{p}_0||}. \tag{2.4}$$

The relationship between t and any point $\mathbf{p} = \mathbf{Q}(\mathrm{t})$ can be readily seen from (2.3) and (2.4): Any value for t in (2.3) defines a unique point $\mathbf{p}$, and substituting $\mathbf{p}$ in (2.4) gives t back.

Furthermore, by letting the arclength be $s = ||\mathbf{p} - \mathbf{p}_0||$, and the total arclength be $\mathcal{L} = ||\mathbf{p}_1 - \mathbf{p}_0||$ in (2.4), the following fundamental relationship between t and s may be written as follows:

$$t = \quad t(s) \quad = s/\mathcal{L}, \quad 0 \leq s \leq \mathcal{L}, \quad \text{and} \tag{2.5}$$

$$s = \quad s(t) \quad = \mathcal{L}t. \tag{2.6}$$

This straightforward relationship between t and s is of great practical importance to architects, draftsmen, or any designers whose designs and drawings are made up of straight lines: Equation (2.5) gives a t corresponding to a desired arclength s, which in turn can be used in (2.3) to get the coordinates of the corresponding reference point $\mathbf{p}$.

Unfortunately, as it will be shown in the next subsections for non-linear Bézier curves, this straightforward relationship is lost, so other techniques to obtain the same benefits (resulting from the relationship between t and s) are sought.

2.1.2 Quadratic Bézier Curves

Quadratic Bézier curves are parabolas represented by Bernstein polynomials [24]. Letting $n = 2$ in (2.1), the following quadratic Bézier equation is obtained:

$$\mathbf{p} = \mathbf{Q}(t) = \mathbf{p}_0(1-t)^2 + 2\mathbf{p}_1(1-t)t + \mathbf{p}_2 t^2. \tag{2.7}$$

Figure 2.2 shows a quadratic Bézier with its control polygon shown by the thin lines. Quadratic Bézier curves are usually used in applications such as interpolation of functional data, joining circular arcs and straight lines, and other applications mentioned in [45]. The arclength of a quadratic Bézier curve can be obtained in closed form. A derivation for obtaining the arclength s as a function of the dimensionless parameter t is shown next.

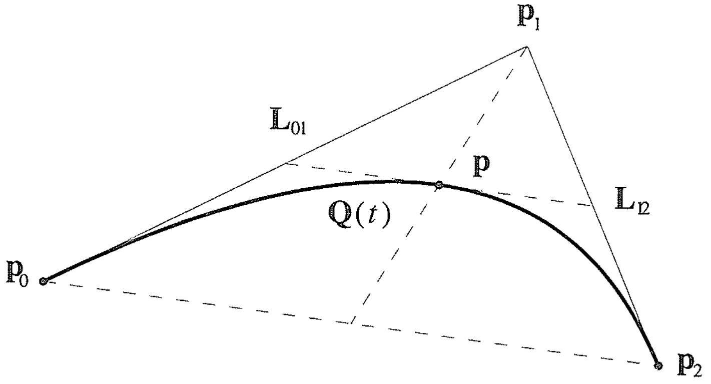

Figure 2.2: $\mathbf{Q}(t)$ is a quadratic Bézier curve with points $\mathbf{p}_{i=0,1,2}$. $\mathbf{p}$ is the point $\mathbf{Q}(\frac{1}{2})$. $\mathbf{L}_{01} = \mathbf{p}_1 - \mathbf{p}_0$ and $\mathbf{L}_{12} = \mathbf{p}_2 - \mathbf{p}_1$. The tangent of $\mathbf{Q}(t)$ at $\mathbf{p}$ is parallel to the line $\mathbf{p}_0\mathbf{p}_2$ and bisects $\mathbf{L}_{01}$ and $\mathbf{L}_{12}$. The line from $\mathbf{p}_1$ passing through $\mathbf{p}$ bisects segment $\mathbf{p}_0\mathbf{p}_2$.

Consider the quadratic Bézier curve shown in Figure 2.3. The arclength $s(t)$ is determined by the integral:

$$\begin{aligned} s(t) &= \int_0^t \|\mathbf{Q}'(\tau)\| \, \mathrm{d}\tau, \\ &= \int_0^t \sqrt{\mathbf{Q}'(\tau) \cdot \mathbf{Q}'(\tau)} \, \mathrm{d}\tau, \end{aligned} \tag{2.8}$$

where

$$\mathbf{Q}'(t) = 2(\mathbf{p}_1 - \mathbf{p}_0)(1-t) + 2(\mathbf{p}_2 - \mathbf{p}_1)t,$$

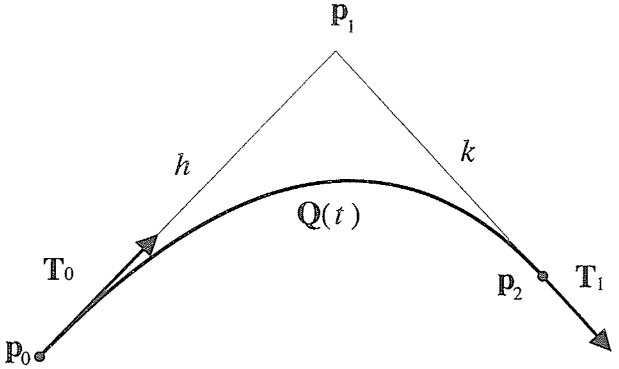

Figure 2.3: $\mathbf{Q}(t)$ is a quadratic Bézier curve. $\mathbf{T}_0$ and $\mathbf{T}_1$ are unit tangent vectors with the arrows showing the tangential directions at $\mathbf{p}_0$ and $\mathbf{p}_2$, respectively; h and k are magnitudes of the vectors $\mathbf{p}_1 - \mathbf{p}_0$ and $\mathbf{p}_2 - \mathbf{p}_1$, respectively.

$$= 2h\mathbf{T}_0(1-t) + 2k\mathbf{T}_1 t. \tag{2.9}$$

Taking the dot product of (2.9) with itself gives the following:

$$\begin{aligned} \mathbf{Q}'(t)\cdot\mathbf{Q}'(t) &= 4h^2(1-t)^2 + 8hk(\mathbf{T}_0\cdot\mathbf{T}_1)(1-t)t + 4k^2t^2, \\ &= 4\Big\{(h^2 - 2hk(\mathbf{T}_0\cdot\mathbf{T}_1) + k^2)t^2 + \\ &\qquad (2hk(\mathbf{T}_0\cdot\mathbf{T}_1) - 2h^2)t + h^2\Big\}. \end{aligned} \tag{2.10}$$

Letting $a = (h^2 - 2hk(\mathbf{T}_0\cdot\mathbf{T}_1) + k^2)$, $b = 2hk(\mathbf{T}_0\cdot\mathbf{T}_1) - 2h^2$, and $c = h^2$, (2.8) is rewritten as follows:

$$s(t) = 2\int_0^t \sqrt{a\tau^2 + b\tau + c}\,\mathrm{d}\tau. \tag{2.11}$$

Using tables of integrals [4, p. 57], (2.11) is evaluated to give the following:

$$s(t) = 2\left[\frac{2a\tau + b}{4a}\sqrt{\Phi} + \frac{4ac - b^2}{8a\sqrt{a}}\log(2a\tau + b + 2\sqrt{a\Phi})\right]_0^t, \tag{2.12}$$

where $\Phi = a\tau^2 + b\tau + c$. The total arclength $\mathcal{L}$ can be determined by evaluating (2.12) at $t = 1$.

Equation (2.12) shows how the arclength s can be determined as a function of t for a quadratic Bézier curve, just as was the case with (2.6) for linear Bézier curves. It can be seen how the cost and complexity of obtaining an expression for the arclength of a quadratic curve is significantly more than the corresponding cost for a linear curve.

Furthermore, whereas an expression for obtaining t as a function of s for linear Bézier curves is simple to derive as shown by (2.5), an expression for t as a function of s is not available for quadratic Bézier curves. The symbolic processor XMaple, for example, was unable to express t as a function of s in (2.12). This problem will be dealt with in the next chapter where a numerical method that approximates t as a function of s is presented.

2.1.3 Cubic Bézier Curves

Many real-world objects are inherently both complex and smooth, and much of computer graphics involves modeling either already existing real-world objects (such as faces, mountains, maps) or designing objects "from scratch" (such as fuselages, highways, ...) [25, p. 471].

Because the classes of curves that are generated by an *nth*-degree polynomial are a superset of those generated by a lower degree polynomial, additional flexibility can be attained by using higher-degree curves. Figure 2.4 illustrates this flexibility concept with some cubic Béziers. It should be noted that it will take *at least* two quadratic Bézier curves in order to obtain any of the shapes in Figure 2.4.

The shape of the Bézier curve can be controlled by adjusting the vertices of its control polygon; certain positions of these vertices will exhibit curves with inflection points, loops, or cusps such as those shown in Figure 2.4. Expressions on the

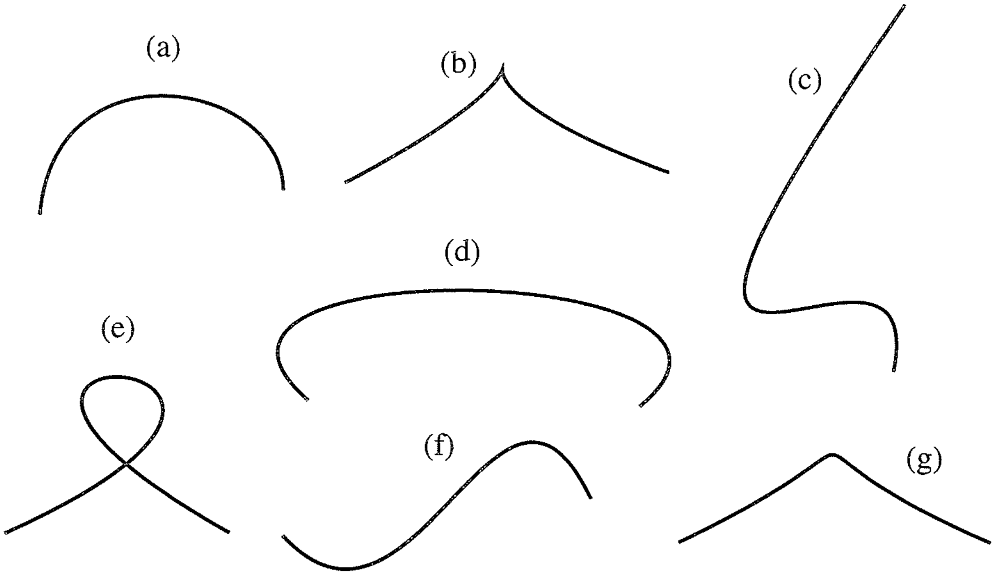

Figure 2.4: All the curves in this figure are one-segment cubic Bézier curves. These curves reflect the flexibility with which objects can be modeled. Figures (a) & (d) show different C-curves; (b) & (e) exhibit a cusp and a loop, respectively; (c) shows how an object like a nose may be modeled; both (c) & (f) show forms of S-curves and therefore each has an inflection point; and finally, (g) shows a mountain-shaped object that has two inflection points.

number of inflections points, loops, or cusps that may result from a parametric curve of degree n are given in [41, pp. 103-112].

Substituting $n = 3$ in (2.1) results in the following cubic Bézier expression:

$$\mathbf{p} = \mathbf{Q}(t) = \mathbf{p}_0(1-t)^3 + 3\mathbf{p}_1(1-t)^2 t + 3\mathbf{p}_2(1-t)t^2 + \mathbf{p}_3 t^3. \tag{2.13}$$

Figure 2.5 shows a cubic Bézier curve with its control polygon shown as thin lines. Generally, for low degree Bézier curves, the curve mimics the control polygon rather

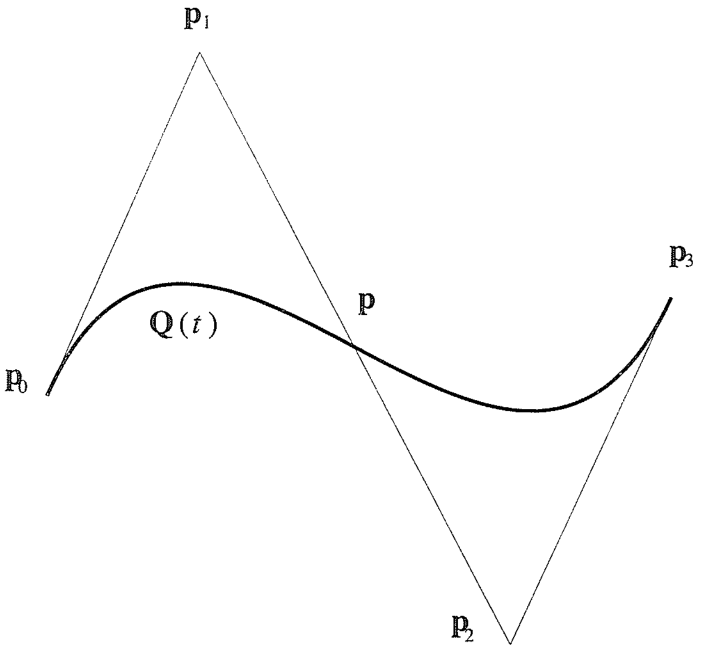

Figure 2.5: A cubic Bézier S-curve.

closely. End points are interpolated since $\mathbf{Q}(0) = \mathbf{p}_0$ and $\mathbf{Q}(1) = \mathbf{p}_n$. Tangent vectors at end points are $\mathbf{Q}'(0) = n(\mathbf{p}_1 - \mathbf{p}_0)$ and $\mathbf{Q}'(1) = n(\mathbf{p}_n - \mathbf{p}_{n-1})$, respectively. A Bézier curve always lies inside the convex hull of its control polygon (see Section 2.1). The convex hull of $\mathbf{Q}(t)$ in Figure 2.5 is the quadrilateral with vertices $\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$.

An advantage of the control polygon is that it indicates a range that $\mathcal{L}$ is bounded by; for cubic Béziers for instance, the value of $\mathcal{L}$ will always satisfy the following inequality[1] relationship [38]:

$$||\mathbf{p}_3 - \mathbf{p}_0|| \leq \mathcal{L} \leq ||\mathbf{p}_1 - \mathbf{p}_0|| + ||\mathbf{p}_2 - \mathbf{p}_1|| + ||\mathbf{p}_3 - \mathbf{p}_2||, \tag{2.14}$$

[1] Inequalities (2.14) and (2.15) become equalities only when all points $\mathbf{p}_0$ through $\mathbf{p}_n$ are collinear.

or more generally for any Bézier curve:

$$\|\mathbf{p}_n - \mathbf{p}_0\| \leq \mathcal{L} \leq \|\mathbf{p}_1 - \mathbf{p}_0\| + \|\mathbf{p}_2 - \mathbf{p}_1\| + \cdots + \|\mathbf{p}_n - \mathbf{p}_{n-1}\|. \tag{2.15}$$

In critical applications, however, where the exact (or very close to the exact) $\mathcal{L}$ is needed, usage of (2.15) is impractical since it only gives an upper and a lower bound on $\mathcal{L}$. A closed-form expression for the arclength of general cubic Bézier curves is not available. Numerical methods for calculating the arclength appear to be the only option. Some methods are described in Chapter 3.

2.2 Pythagorean-Hodograph Curves

Recently, Farouki and Sakkalis in [22] identified a special class of Bézier polynomials; closed-form expressions for their arclength as a function of the parameter are available. These are the Pythagorean-hodograph (PH) Bézier curves. In the next two subsections, their properties and their algebraic construction are described.

2.2.1 Properties & Related Issues

PH curves possess the following important properties (see [17][22]):

- Their arclength is expressible in closed-form as polynomial functions.
- A method for approximating arclength parameterization exists. The method combines both analytical and numerical algorithms and is rather lengthy. However, precision of up to 12 decimal digits is obtainable in a few Newton-Raphson iterations.

- Their offsets are representable by rational *independent* curves[2]. In CAD applications such as highway design and railway route location, CNC machining, or in the textile or shoe industry, the generation of offset curves is desirable. An offset curve to $\mathbf{Q}(t)$ that is $\pm d$ units away from it, in the direction of its unit normal $\mathbf{N}(t)$ is defined by $\hat{\mathbf{Q}}(t) = \mathbf{Q}(t) \pm d\mathbf{N}(t)$.

  When rendering offset curves, loops or cusps may result [29]. It is desirable to represent the curve independently to facilitate direct manipulation of the resultant curve, rather than having its representation be dependent on the original curve. Further material on offset curves is found in [14][34].

  Generally, independent representations of offset curves can not be obtained in polynomial/rational form; they are usually irreducible algebraic formulas of high degree [17].

There is a price, however, to pay for the aforementioned attractive properties. Besides the complexity of their algebraic and geometric construction, PH curves have fewer degrees of freedom than the general Bézier curves of the same degree. A lemma in [22] states that:

> *PH curves of degree* n *have (at most)* $n+3$ *degrees of freedom, i.e.,* $n-1$ *fewer than the* $2(n+1)$ *degrees of freedom associated with general polynomial curves of the same degree.*

To see this, consider the cubic Bézier curves shown in Figure 2.6. In order for these curves to qualify as PH curves, the following two conditions must be met (see Theorem 1 in [22]):

$$L_2 = \sqrt{L_1 L_3} \qquad \text{and} \qquad \theta_1 = \theta_2. \tag{2.16}$$

[2] In [34], Meek and Walton define offset curves as "the locus of points which are a fixed perpendicular distance away from the curve".

Hence, for PH cubics, only 6 degrees of freedom are available for design purposes,

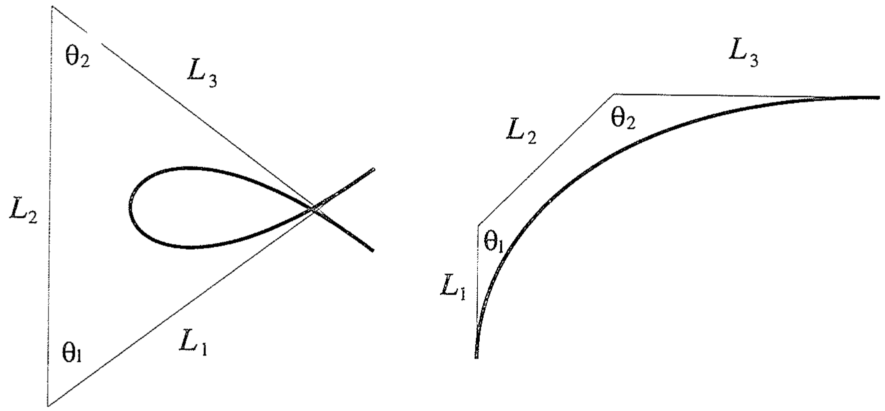

Figure 2.6: $L_i = \|\mathbf{p}_i - \mathbf{p}_{i-1}\|$. Two PH curves satisfying the conditions of (2.16).

as opposed to 8 degrees of freedom available to general cubics. This makes PH cubics inconvenient for interpolation and modeling purposes.

Other constraints on PH cubics stem from the fact that they are segments of a single curve known as *Tschirnhausen*'s cubic which dates back to 1690 [17]. It is defined as follows:

$$\mathbf{T}(t) = \{r\,(t^2-1),\ \frac{\pm r}{\sqrt{3}}\,t(t^2-1)\}, \tag{2.17}$$

where $-r$ is the x-axis intercept. The curve on the left in Figure 2.6 may be considered as an instance of Tschirnhausen's cubic (with end-points at infinity since $-\infty < t < \infty$). In this regard, the curve on the right in Figure 2.6 may be considered to be a small segment of that on the left (after applying suitable transformations such as rotations, uniform scalings, and bounded parameterizations). PH cubics cannot exhibit cusps or inflection points. These constraints and rigidities enforced on the shape of the curve may be alleviated by using PH quintic curves, but the complexity and the cost of using them is more involved.

Further references to PH curves can be found in [19][21][37] and the references therein.

2.2.2 Algebraic Construction

Recall from Subsection 1.5.1 that a curve $\mathbf{Q}(t) = \{x(t), y(t)\}$ has a PH if the expression $x'^2(t) + y'^2(t)$ is the square of a polynomial $\sigma(t)$ [17]. This triple

$$\{x'(t), y'(t), \sigma(t)\}$$

satisfies:

$$\begin{aligned} x'(t) &= u^2(t) - v^2(t), \\ y'(t) &= 2u(t)v(t), \\ \sigma(t) &= u^2(t) + v^2(t), \end{aligned} \tag{2.18}$$

for a PH cubic, where

$$\left.\begin{aligned} u(t) = u_0(1-t) + u_1 t \\ v(t) = v_0(1-t) + v_1 t \end{aligned}\right\}, \qquad u_i, v_i, \in I\!R, i = 0, 1. \tag{2.19}$$

The control points of a cubic PH are constructed as follows:

$$\begin{aligned} \mathbf{p}_1 &= \mathbf{p}_0 + \tfrac{1}{3}(u_0^2 - v_0^2, 2u_0v_0), \\ \mathbf{p}_2 &= \mathbf{p}_1 + \tfrac{1}{3}(u_0u_1 - v_0v_1, u_0v_1 + u_1v_0), \\ \mathbf{p}_3 &= \mathbf{p}_2 + \tfrac{1}{3}(u_1^2 - v_1^2, 2u_1v_1), \end{aligned} \tag{2.20}$$

where $\mathbf{p}_0$ is arbitrary.

Similarly, a quintic PH may be constructed by defining $u(t)$ and $v(t)$ to be the following two quadratic polynomials:

$$\left.\begin{aligned} u(t) = u_0(1-t)^2 + 2u_1(1-t)t + u_2t^2 \\ v(t) = v_0(1-t)^2 + 2v_1(1-t)t + v_2t^2 \end{aligned}\right\}, \qquad u_i, v_i \in I\!R, i = 0, 1, 2, \tag{2.21}$$

with control points defined as:

$$\begin{aligned}
\mathbf{p}_1 &= \mathbf{p}_0 + \tfrac{1}{5}(u_0^2 - v_0^2, 2u_0v_0),\\
\mathbf{p}_2 &= \mathbf{p}_1 + \tfrac{1}{5}(u_0u_1 - v_0v_1, u_0v_1 + u_1v_0),\\
\mathbf{p}_3 &= \mathbf{p}_2 + \tfrac{2}{15}(u_1^2 - v_1^2, 2u_1v_1) + \tfrac{1}{15}(u_0u_2 - v_0v_2, u_0v_2 + u_2v_0),\\
\mathbf{p}_4 &= \mathbf{p}_3 + \tfrac{1}{5}(u_1u_2 - v_1v_2, u_1v_2 + u_2v_1),\\
\mathbf{p}_5 &= \mathbf{p}_4 + \tfrac{1}{5}(u_2^2 - v_2^2, 2u_2v_2),
\end{aligned} \tag{2.22}$$

where $\mathbf{p}_0$ is arbitrary. The following constraints on the values of $\{u_0, u_1, u_2\}$ and $\{v_0, v_1, v_2\}$ must hold [20]:

1. Neither $\{u_0, u_1, u_2\}$ nor $\{v_0, v_1, v_2\}$ may be set to $\{0, 0, 0\}$.

2. Either $\{u_0, u_1, u_2\}$ or $\{v_0, v_1, v_2\}$ may be set to $\{k, k, k\}$, but not both, where $k \in I\!R$ and $k \neq 0$.

3. The inequality $(u_2v_0 - u_0v_2)^2 \neq 4(u_0v_1 - u_1v_0)(u_1v_2 - u_2v_1)$ must hold.

2.3 B-spline curves

B-splines are widely used for approximating curves and surfaces. They are very practical for design purposes as well as for modeling and interpolation purposes [2][28].

A B-splines basis of order d (degree $d-1$), with $m+1$ control vertices $\mathbf{v}_{i=0,1,\ldots,m}$, is defined by a recursive expression that was independently reached by de Boor [13], and Cox [10]. Given a partition $[\ldots, t_{-1}, t_0, t_1, t_2, \ldots]$ on the infinite real-line, the basis may be written as follows.

$$\begin{aligned} B_{i,1}(t) &= \begin{cases} 1, & \text{if } t_i \leq t < t_{i+1} \\ 0, & \text{otherwise} \end{cases} \\ B_{i,d}(t) &= \frac{t - t_i}{t_{i+d-1} - t_i} B_{i,d-1}(t) + \frac{t_{i+d} - t}{t_{i+d} - t_{i+1}} B_{i+1,d-1}(t), \end{aligned} \tag{2.23}$$

where $2 \leq d \leq m+1$. A B-spline curve is defined by the following expression:

$$\mathbf{Q}(t) = \sum_{i=1}^{m+1} \mathrm{v}_{i-1} B_{i,d}(t), \qquad t_{d-1} \leq t < t_{m+1}. \tag{2.24}$$

Some properties and advantages of B-splines are mentioned next.

- The sum of the B-spline blending functions on the interval $[t_i, t_{i+1}]$ is unity (assuming that all basis functions are ≥ 0). That is:

  $$\sum_{j=i-d+2}^{i+1} B_{j,d}(t) = 1, \quad d - 1 \leq i \leq m. \tag{2.25}$$

  implying that the convex hull property holds for B-splines.

- Compact support, i.e., $B_{i,d} = 0$ for $t < t_i,\ t \geq t_{i+d}$.

- Each segment of a B-spline curve is influenced by d vertices.

- Any single vertex influences the shape of at most d curve segments.

- Automatic C^{d-2} continuity over the internal $m - d + 1$ knots, resulting in smooth curve and surface designs.

- Complex curve shapes may be attained by using a large number of control points, rather than using a high degree curve. This is a major advantage over Bézier curves.

The relationship between a B-spline curve and its control polygon is not as intuitive as it is for Bézier curves; the end points of the control polygon are not interpolated and some geometry is needed for manual construction of the curve, as shown in Figure 2.7. Mechanisms for understanding the relationship between a B-spline curve and its control polygon are found in [35]. In addition, B-spline basis functions are more complex than the Bernstein's. Depending on the knot vector format, several types of B-splines may result [2][3][25]. Only uniform B-splines are considered in this thesis.

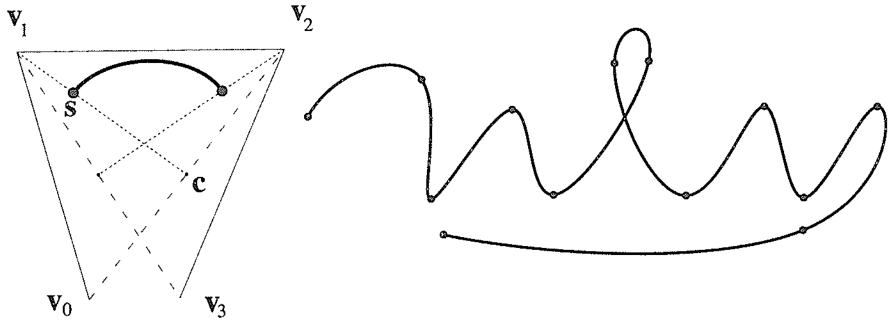

Figure 2.7: Both curves in this figure are cubic B-spline curves. The one on the left shows a one-segment curve with its control polygon represented by the thin lines. To determine the starting point of such a curve, a line is drawn from $\mathbf{v}_0$ to $\mathbf{v}_2$, $\mathbf{v}_0\mathbf{v}_2$ is then intersected by $\mathbf{v}_1\mathbf{c}$ at $\mathbf{c} = \frac{1}{2}(\mathbf{v}_0 + \mathbf{v}_2)$, the starting point is $\mathbf{s} = \mathbf{v}_1 + \frac{1}{3}(\mathbf{c} - \mathbf{v}_1)$. The end point is obtained in a similar manner. The curve on the right shows multiple segments joined together with C^2 continuity preserved at the joints.

Uniform B-splines are so named because of the knot vector used to generate them; it has the following format:

$$\{t_0,\ t_1,\ \cdots,\ t_{m+d-1},\ t_{m+d}\},\quad t_i<t_{i+1},\quad t_{i+1}-t_i=t_i-t_{i-1}. \tag{2.26}$$

Any segment in Figure 2.7 for example may be generated by the following knot vector:

$$\{0,\ 1,\ 2,\ 3.\ 4,\ 5,\ 6,\ 7\},\quad d=4,\quad m+1=4. \tag{2.27}$$

Substituting (2.26) into (2.23) and (2.24) for $n = 3$ yields the following matrix expression for a uniform cubic B-spline.

$$\mathbf{Q}(t)=\frac{1}{6}\,[t^3\ \ t^2\ \ t\ \ 1]\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}\begin{bmatrix} \mathbf{v}_0 \\ \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{bmatrix}. \tag{2.28}$$

In addition to the previous mentioned properties, one of the main advantages in using uniform B-splines is that the Bézier control points defining the same curve segment are easily locatable. This is illustrated in Figure 2.8. To convert the representation of the middle B-spline segment defined by $\mathbf{v}_{i=1,2,3,4}$ in Figure 2.8 into a Bézier format, for example, the Bézier control points $\mathbf{p}_{i=0,1,2,3}$ defined in (2.13) are set to the following:

$$\begin{aligned} \mathbf{p}_0 &= \tfrac{1}{6}\,(\mathbf{v}_1+4\mathbf{v}_2+\mathbf{v}_3), \\ \mathbf{p}_1 &= \tfrac{1}{3}\,(2\mathbf{v}_2+\mathbf{v}_3), \\ \mathbf{p}_2 &= \tfrac{1}{3}\,(\mathbf{v}_2+2\mathbf{v}_3), \\ \mathbf{p}_3 &= \tfrac{1}{6}\,(\mathbf{v}_2+4\mathbf{v}_3+\mathbf{v}_4). \end{aligned} \tag{2.29}$$

The advantages of this dual-representation is that while a curve can be *flexibly* designed using B-spline basis functions, tasks such as finding the arclength [17][21]

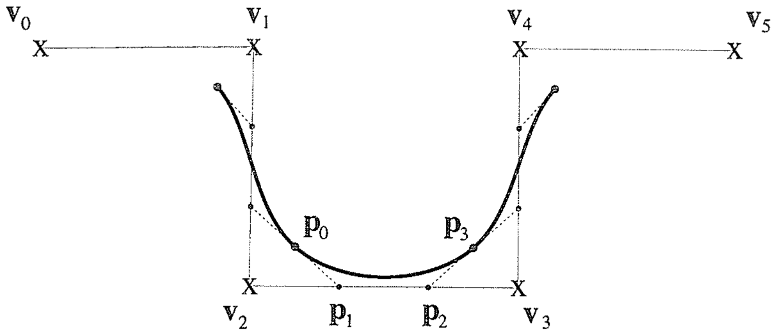

Figure 2.8: In this figure, the thin lines connect the control points of the cubic B-spline represented by the x's. The larger dots on the curve are the places where cubic B-spline segments join with C^2 continuity, they also denote the start and end vertices of the corresponding cubic Bézier. The smaller dots on the polygon denote the inner control points of that Bézier segment.

[22][23], repeated subdivision [15][25], curvature analysis [42], and other tasks that have been investigated for cubic Bézier curves can be applied to cubic B-splines.

2.4 Summary

Bézier curves of several degrees were introduced. It was shown how the problem of calculating the arclength increases in complexity as the order of the polynomial increases.

PH curves were introduced. It was shown how they avoid some of the key problems (see Chapter 1) inherent in other parametric curves, but that they have their own shortcomings that affects the freedom of design.

Uniform B-splines were included because the results of this thesis are applicable to designs employing them. It was shown how any segment defined as a uniform B-spline curve may be represented in Bézier format; so the results of this dissertation are directly applicable to uniform B-spline curves.

As for the problem of arclength parameterization, it is shown how the introduction of non-linearity in polynomials of degree ≥ 2 makes it difficult or impossible to solve for the parameter t as a function of the arclength s.

In the next chapter, several methods for calculating the arclength are presented. In addition, some existing methods for generating reference points along parametric curves will be listed, and a new approach with results and performance comparable to existing methods is proposed.

Chapter 3

Arclengths and Point Locating

In Chapter 2, an overview of the problem of evaluating arclengths of different curves was presented. Only a few curves possess the property that their arclength can be expressed in closed form. It is therefore inherently difficult to search for arclength-parameterizable expressions for parametric curves in general.

The problem of solving for the arclength of a curve as a function of the curve parameter, or solving for the parameter as a function of the arclength of a curve was also shown to be trivial for simple curves (as discussed in Subsection 2.1.1), but that finding such solutions becomes difficult as the curve's degree increases.

In this Chapter, several methods for computing the arclength of general parametric curves are presented, and methods that are designed to extract an arclength parameterization for certain parametric curves are discussed. The chapter concludes with a proposed method that approximates the arclength parameterization for any parametric curve.

Chapter organization is as follows: Section 3.1 presents several methods for computing the arclength of various curves. Arclength calculation is needed because many reference-point generating algorithms depend on the arclength *a priori* as input, for normalization purposes and other purposes discussed in Chapter 4. In Section 3.2, the problem of solving for the parameter t in terms of the arclength s is illustrated for some special curves. It is assumed from Section 3.2 onwards, that a parametric curve $\mathbf{Q}(t)$, with total arclength $\mathcal{L}$ is given. In Section 3.3, a method to approximate arclength parameterization is presented. The method approximates the function $t(s)$ for any regular, parametric curve of any degree. This enables the systematic generation of reference points along the path of a curve. Section 3.4 concludes this chapter with an illustration of how a simple reference-point-generator algorithm is used to generate a set of reference points along the path of a linear curve. This problem is more deeply investigated in Chapter 4 for higher-degree curves.

3.1 Arclength Calculation

To calculate the arclength $s(t)$ of a parametric expression $\mathbf{Q}(t)$, one first attempts to obtain a closed-form expression for

$$s(t) = \int_{t_0}^{t} \sqrt{\mathbf{Q}'(\tau) \cdot \mathbf{Q}'(\tau)}\, \mathrm{d}\tau, \tag{3.1}$$

because (3.1) provides optimal precision with minimum computation. However, this type of integral is not usually expressible as an algebraic function [17][40][43][47]. It is therefore necessary to resort to numerical techniques to compensate for the lack of a closed-form expressions. It is the intent of this section to list closed-form expressions for curves which possess them, and present several numerical methods to obtain the arclength of any curve presented in a parametric form.

In the following four subsections 3.1.1–3.1.4, curves that have closed-form expression for the arclength are discussed. The last two subsections 3.1.5–3.1.6 present numerical techniques that may be employed when a closed-form solution does not exist.

3.1.1 Linear Curves

Linear curves are the simplest case to consider. They are used extensively by architects or draftsmen whose models consist mainly of straight, intersecting lines (e.g., for designing buildings, sharp-edge devices, etc.).

Given two points in a plane, $\mathbf{p}_0 = \{x_0, y_0\}$ and $\mathbf{p}_1 = \{x_1, y_1\}$, the arclength can be obtained by measuring the length of the line that passes through them. Letting $n = 1$ in (2.1) gives:

$$\mathbf{Q}(t) = \mathbf{p}_0(1-t) + \mathbf{p}_1 t. \tag{3.2}$$

The arclength of (3.2) may be determined by the following expression:

$$\begin{aligned} s(t) &= \int_0^t \|\mathbf{Q}'(\tau)\| \, \mathrm{d}\tau, \\ &= \int_0^t \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} \, \mathrm{d}\tau. \end{aligned} \tag{3.3}$$

The square root term of (3.3) is simply a number $\ell = \|\mathbf{p}_1 - \mathbf{p}_0\|$, so

$$s(t) = \int_0^t \ell \, \mathrm{d}\tau = [\ell\,\tau]_0^t = \ell\, t. \tag{3.4}$$

The total arclength $\mathcal{L}$ may be evaluated by setting $t = 1$ in the above expression:

$$\mathcal{L} = s(1) = \ell. \tag{3.5}$$

3.1.2 Quadratic Curves

Quadratic curves are represented by second-degree polynomials. Letting $n = 2$ in (2.1) results in the following expression (2.7):

$$\mathbf{Q}(t) = \mathbf{p}_0(1-t)^2 + 2\mathbf{p}_1(1-t)t + \mathbf{p}_2 t^2. \tag{3.6}$$

The arclength of any quadratic curve may be obtained by the following expression (2.12):

$$s(t) = 2\left[\frac{2a\tau + b}{4a}\sqrt{\Phi} + \frac{4ac - b^2}{8a\sqrt{a}}\log(2a\tau + b + 2\sqrt{a\Phi})\right]_0^t, \tag{3.7}$$

where $\Phi = a\tau^2 + b\tau + c$, and $a, b,$ and c are as derived in Section 2.1.2. Again, the total arclength $\mathcal{L}$ is obtained by evaluating (3.7) at $t = 1$.

3.1.3 PH Curves

PH curves were presented in Section 2.2. It was mentioned that they are a subclass of general Bézier curves and they have fewer degrees of freedom. PH cubics, for example, must satisfy the constraints outlined by (2.16), and PH quintics must satisfy the constraints outlined in Section 2.2.2, in addition to a geometrical constraint on the polygon legs lengths as mentioned in [22]:

$$L_2/L_4 = \sqrt{L_1/L_5}, \qquad L_i = \|\mathbf{p}_i - \mathbf{p}_{i-1}\|, \quad i = 1, 2, \ldots, 5. \tag{3.8}$$

The arclength of a PH curve of odd degree n is obtained by the following closed-form Bernstein basis functions,

$$s(t) = \sum_{k=0}^{n} s_k \begin{pmatrix} n \\ k \end{pmatrix} (1-t)^{n-k} t^k, \tag{3.9}$$

where

$$s_0 = 0, \qquad s_k = \frac{1}{n}\sum_{j=0}^{k-1}\sigma_j, \qquad \text{for } k = 1..n, \tag{3.10}$$

and σ_j is defined by

$$\sigma_j = \sum_{i=\max(0,j-m)}^{\min(m,j)} \frac{\begin{pmatrix} m \\ i \end{pmatrix}\begin{pmatrix} m \\ j-i \end{pmatrix}}{\begin{pmatrix} n-1 \\ j \end{pmatrix}}(u_i u_{j-i} + v_i v_{j-i}), \tag{3.11}$$

for $j = 0, 1, \ldots, n-1$, $m = \frac{1}{2}(n-1)$, and $u_i, v_i \in I\!R$, with constraints described in Section 2.2.2.

Cubic PH Curves

The arclength of a cubic PH curve is obtained by setting $n = 3$ in (3.9). The resulting $\sigma_{i=0,1,2}$ from (3.11) are as follows:

$$\begin{aligned} \sigma_0 &= u_0^2 + v_0^2, \\ \sigma_1 &= u_0 u_1 + v_0 v_1, \\ \sigma_2 &= u_1^2 + v_1^2. \end{aligned} \tag{3.12}$$

The total arclength $\mathcal{L}$ is obtained when (3.9) is evaluated at $t = 1$, which is:

$$\mathcal{L} = s(1) = \frac{\sigma_0 + \sigma_1 + \sigma_2}{3}. \tag{3.13}$$

In addition to (3.13), the following expression given in terms of the geometry of the polygon legs also determines the total arclength of a cubic PH curve (see Figure 2.6):

$$\mathcal{L} = L_1 - L_2 \cos\theta + L_3, \qquad \theta = \theta_1 = \theta_2. \tag{3.14}$$

See [21] for more details on (3.14).

Quintic PH Curves

For a quintic PH curve ($n = 5$), (3.11) gives $\sigma_{i=0,1,\ldots,4}$ in the following manner:

$$\begin{aligned}\sigma_0 &= u_0^2 + v_0^2,\\ \sigma_1 &= u_0u_1 + v_0v_1,\\ \sigma_2 &= \tfrac{2}{3}(u_1^2 + v_1^2) + \tfrac{1}{3}(u_0u_2 + v_0v_2),\\ \sigma_3 &= u_1u_2 + v_1v_2,\\ \sigma_4 &= u_2^2 + v_2^2.\end{aligned} \tag{3.15}$$

The total arclength given when $t = 1$ in (3.9) is:

$$\mathcal{L} = s(1) = \frac{\sigma_0 + \sigma_1 + \sigma_2 + \sigma_3 + \sigma_4}{5}. \tag{3.16}$$

Generally, for a PH curve of degree n, the total arclength is given by the following expression:

$$\mathcal{L} = \frac{\sigma_0 + \sigma_1 + \cdots + \sigma_{n-1}}{n}. \tag{3.17}$$

3.1.4 The Helix

The helix is a parametric curve in $\mathbb{E}^3$ with a closed-form expression for the arclength. It is defined parametrically by

$$\mathbf{Q}(t) = \{a\cos t, a\sin t, bt\}, \qquad a, b \in \mathbb{R}, \tag{3.18}$$

with arclength

$$s(t) = \sqrt{a^2 + b^2}\, t. \tag{3.19}$$

For theory and applications on helices, see [5][9][47].

3.1.5 Cumulative Chord-Length Approximation

Cumulative chord-length approximation, or piecewise linear approximation, is the process of determining the length of a curve by summing the approximating polygonal segments to that curve. The error that results from this method can be made arbitrarily small by using smaller polygonal arcs to approximate the given curve [25, p. 472][31].

In order to calculate the arclength of $\mathbf{Q}(t)$, let $\mathbf{v}_i = \mathbf{Q}(t_i), i = 0, 1, \ldots, \Im$, the number of polygonal segments used to approximate the curve, so that the points $\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_{\Im}$ form $\mathcal{P}$, the piecewise approximating polygon to $\mathbf{Q}(t)$. The arclength of $\mathbf{Q}(t)$ can be approximated by summing all the segments $\|\mathbf{v}_i - \mathbf{v}_{i-1}\|$ of $\mathcal{P}$ for $i = 1, 2, \cdots, \Im$. That is, the arclength $s(t)$ at $t = t_k = k/\Im$ is approximated by

$$s(t) \approx \sum_{i=1}^{k} \|\mathbf{v}_i - \mathbf{v}_{i-1}\|, \quad \text{where} \ \ k \in [1, \Im]. \tag{3.20}$$

The total arclength $\mathcal{L}$ is obtained from (3.20) when $k = \Im$.

3.1.6 Calculating Arclengths with Simpson's Rule

Simpson's rule is a much more accurate method than the chordlength method described in the previous subsection for finding the arclength. It is chosen for the work accomplished in this thesis for the following reasons:

- It is a very popular and frequently used rule in approximating integrals because of its high accuracy and relative simplicity [1][12][36].
- Error analysis may be initiated by using Simpson's error term on the integral expression (3.1). The significance of this is that it allows "fine tuning" to select the number of intervals that reduces the error to an acceptable tolerance [12].

Detailed description of Simpson's method may be found in many calculus or numerical-methods books [32]. A brief definition is given here.

Let I (even, preferably a power of two) be the number of Simpson intervals used for Simpson's rule, implying that $\mathbf{Q}(t)$ is approximated by $I/2$ segments. The function $s(t)$ may then be approximated as follows:

$$s(t) \approx \frac{t - t_0}{3I} \left[\alpha_0 + 4\alpha_1 + 2\alpha_2 + 4\alpha_3 \cdots + 2\alpha_{I-2} + 4\alpha_{I-1} + \alpha_I\right], \qquad (3.21)$$

where $\alpha_{i=0,1,\ldots,I} = \|\mathbf{Q}'(t_i)\|$, and $t, t_0 \in [0,1]$, with $t > t_0$.

3.2 The Parameter t as a Function of the Arclength

In Section 1.3, the problem of solving for the parameter t as a function of the arclength s was introduced. It was emphasized that the non-linearity inherent in (1.6) made it, in most cases, impossible to extract t on one side, and the rest of (1.6) on the other side. Such an expression would result in an arclength parameterization.

In this section, the process of solving for t is briefly reviewed for curves that possess closed-form expressions for $t(s)$. In the next section, a proposed approximation method for the function $t(s)$ is introduced.

3.2.1 Linear Curves

In Section 3.1.1, it was shown that for linear curves, $s(t)$ was a linear expression. This linearity makes it straightforward to find the inverse function $t(s)$, which is sought here. The resulting expression is:

$$t(s) = s/\mathcal{L}, \qquad 0 \leq s \leq \mathcal{L}. \tag{3.22}$$

See also Section 2.1.1 for further details of this derivation. Notice how simple a process it is to generate reference points that are uniformly spaced along $\mathbf{Q}(t)$. Section 3.4 discusses reference-point generation in more detail for linear curves.

3.2.2 PH Curves

PH curves have a closed-form expression for $s(t)$, which have already been introduced in Section 3.1.3 with (3.9). Unlike expression (3.22) however, $t(s)$ here must be approximated.

Like Sharpe & Thorne's method in [40] (or Section 1.4.2 in this thesis), Farouki considers each $t_i = t(s_i)$ to be a real root such that $s(t_i) - s_i = 0$. It is therefore necessary to develop a method that will obtain and refine each t_i. A detailed description of the mathematical derivation of this method is found in [17]. The following are the main points.

Recall from Section 2.2.2 that the magnitude of the derivative of a PH curve of odd degree n is given by

$$\sigma(t) = \|\mathbf{Q}'(t)\| = \sqrt{x'^2(t) + y'^2(t)} \quad \Longrightarrow \quad \sigma(t) = u^2(t) + v^2(t). \tag{3.23}$$

Since $u(t)$ and $v(t)$ of degree $m = (n-1)/2$ are functions that may be written in Bernstein form (see (2.19) and (2.21)), so can $\sigma(t)$:

$$\sigma(t) = \sum_{k=0}^{n-1} \sigma_k \begin{pmatrix} n-1 \\ k \end{pmatrix} (1-t)^{n-1-k} t^k, \tag{3.24}$$

where σ_k is given by (3.11). The parametric value t_d that generates a point $\mathbf{Q}(t_d)$ along the curve, with given distance s_d from a known point $\mathbf{Q}(t_p)$, where $t_p < t_d$, may now first be approximated by

$$\tau_0 = t_p + \frac{s_d}{\sigma(t_p)}, \qquad s(t_p) \leq s_d + s(t_p) \leq \mathcal{L}, \tag{3.25}$$

where $s(t)$ here is defined by (3.9) and then refined by a few Newton-Raphson iterations as follows:

$$\tau_j = \tau_{j-1} - \frac{s(\tau_{j-1}) - s_d}{\sigma(\tau_{j-1})}, \qquad j = 1, 2, \ldots, k. \tag{3.26}$$

The value assigned to t_d is τ_k, where k is the number of iterations required to converge to an acceptable accuracy.

3.3 The Proposed Approximation Method

An abstract statement of the problem to be solved may be put as follows:

> Given a point $\mathbf{Q}(t_p)$ and a distance s_d, find the parameter t_d such that $\mathbf{Q}(t_d)$ is a distance s_d along the curve $\mathbf{Q}(t)$ from $\mathbf{Q}(t_p)$.

It is the intent of this section, to develop the theory behind a method, based on Hermite interpolation, that aims to solve the above problem by approximating the function $t(s)$ for any parametric curve. It is left to Chapter 4 to present the practical results of this proposed approximation method.

In the following subsection, the steps taken to approximate the function $t(s)$ by a cubic polynomial are presented. The succeeding subsection examines the approximation of $t(s)$ using higher-degree polynomials. Again, it is assumed that for a given curve $\mathbf{Q}(t)$, the total arclength $\mathcal{L}$ has already been calculated.

3.3.1 Using a Cubic Interpolation-Function

In all the expressions $s(t)$ and $t(s)$ seen previously, s is always a monotone increasing function of the parameter t, and t is always a monotone increasing function of the arclength s. To address the problem put forward in the opening of this section, however, knowledge of the rate of their variability with respect to each other is also needed; the proposed approximation method attempts to capture this relationship, and provides a meaningful accurate correspondence between the flow of t and s.

The idea is to first derive the function $t(s)$ from $s(t)$ for a general parametric curve $\mathbf{Q}(t)$, and then approximate it by another polynomial function, using interpolation techniques such as Hermite interpolation [11][15][25]. Attention is first focused on a cubic interpolator function $f(s)$. Namely,

$$f(s) = as^3 + bs^2 + cs + d. \tag{3.27}$$

A higher-degree interpolator function $h(s)$ is considered afterwards.

The inverse function of $s(t) = \int_0^t \|\mathbf{Q}'(\tau)\|\,\mathrm{d}\tau$ may be derived as follows. First, $s(t)$ is differentiated to result in

$$s'(t) = \frac{\mathrm{d}s}{\mathrm{d}t} = \|\mathbf{Q}'(t)\|. \tag{3.28}$$

From (3.28), $t'(s)$ may be written as follows:

$$t'(s) = \frac{\mathrm{d}t}{\mathrm{d}s} = \frac{1}{\|\mathbf{Q}'(t)\|}. \tag{3.29}$$

Upon integration of (3.29), the following results:

$$t(s) = \int_0^s \frac{1}{\|\mathbf{Q}'(\tau)\|}\,\mathrm{d}\tau. \tag{3.30}$$

The complexity of $t(s)$ is evident from the above expression, and an attempt to simplify it may prove impractical in most cases. However, the value of $t(s)$ at two crucial parametric values is already known, namely, at $s = 0$ and $s = \mathcal{L}$. Further, for a cubic interpolator, two more pieces of data are needed to extract values for the four coefficients of $f(s)$ in (3.27): a, b, c and d. Equation (3.29) may be used to obtain some of this data, the rest is inferred. The data from $t(s)$ and $t'(s)$ are used as follows.

Since $s = 0 \implies t = 0$, it follows that

$$t(0) = 0, \tag{3.31}$$

and

$$t'(0) = \frac{1}{\|\mathbf{Q}'(0)\|}. \tag{3.32}$$

Since $s = \mathcal{L} \implies t = 1$, it follows that

$$t(\mathcal{L}) = 1, \tag{3.33}$$

and

$$t'(\mathcal{L}) = \frac{1}{\|\mathbf{Q}'(1)\|}. \tag{3.34}$$

Differentiation of (3.27) gives

$$f'(s) = 3as^2 + 2bs + c. \tag{3.35}$$

The four coefficients of $f(s)$ may thus be determined in the following manner:

1. d is obtained as follows:

$$f(0) = t(0) \quad \Longrightarrow \quad d = 0. \tag{3.36}$$

2. c is obtained as follows:

$$f'(0) = t'(0) \quad \Longrightarrow \quad c = \frac{1}{||\mathbf{Q}'(0)||}. \tag{3.37}$$

Now (3.33) and (3.34), in conjunction with (3.27) and (3.35) yield the following two equations in two unknowns, namely:

$$a\mathcal{L}^3 + b\mathcal{L}^2 = 1 - c\mathcal{L},$$

and

$$3a\mathcal{L}^2 + 2b\mathcal{L} = \frac{1}{||\mathbf{Q}'(1)||} - c,$$

from which a and b may be solved (note that c is known from (3.37)). Hence

3. a is obtained as follows:

$$f'(\mathcal{L}) = t'(\mathcal{L}) \quad \Longrightarrow \quad a = \frac{1}{\mathcal{L}^2}\left(c + \frac{1}{||\mathbf{Q}'(1)||}\right) - \frac{2}{\mathcal{L}^3}. \tag{3.38}$$

4. And b is obtained as follows:

$$f(\mathcal{L}) = t(\mathcal{L}) \quad \Longrightarrow \quad b = \frac{1}{\mathcal{L}^2} - \frac{c}{\mathcal{L}} - a\mathcal{L}. \tag{3.39}$$

Now that the four coefficients have been determined, an approximate solution to the problem statement put in the opening of this section may be proposed as follows:

> To determine the point on $\mathbf{Q}(t)$ which is a distance s_d away from $\mathbf{Q}(t_p)$, evaluate $\mathbf{Q}(t)$ at $t_d = f(s_p + s_d)$, $0 < s_p + s_d \leq \mathcal{L}$, where s_p is the arclength from $\mathbf{Q}(0)$ to $\mathbf{Q}(t_p)$.

3.3.2 Using Higher Degree Interpolation-Functions

For design and modeling purposes, cubic polynomials are widely used in practice for their relatively low degree and high accuracy [15][17][41]; their flexibility will be evident from the examples shown in Chapter 4. However, at the expense of a few more coefficients to be determined in a higher-degree interpolation function, a quintic polynomial is put to the test.

Following the same method used for determining the required coefficients for $f(s)$ in (3.27), we start by defining a quintic polynomial $h(s)$ to approximate $t(s)$ as

$$h(s) = as^5 + bs^4 + cs^3 + ds^2 + es + g. \tag{3.40}$$

Six equations are required to determine the coefficient in (3.40). They are chosen to correspond to $h(0)$, $h(\mathcal{L})$, $h'(0)$, $h'(\mathcal{L})$, $h''(0)$, and $h''(\mathcal{L})$, where

$$h'(s) = 5as^4 + 4bs^3 + 3cs^2 + 2ds + e, \tag{3.41}$$

and

$$h''(s) = 20as^3 + 12bs^2 + 6cs + 2d. \tag{3.42}$$

As for the cubic case, $t(s)$ and its derivatives are used to provide the necessary data to determine the coefficients in the above equations. That is: $t(0)$, $t(\mathcal{L})$, $t'(0)$, $t'(\mathcal{L})$, $t''(0)$, and $t''(\mathcal{L})$, where $t(s)$ and $t'(s)$ are described by (3.30) and (3.29), respectively, and $t''(s)$ is obtained by differentiating (3.29) to give

$$t''(s) = \frac{\mathrm{d}}{\mathrm{d}s}t'(s) = \frac{\mathrm{d}}{\mathrm{d}t}\left\{\frac{1}{\|\mathbf{Q}'(t)\|}\right\}\frac{\mathrm{d}t}{\mathrm{d}s}. \tag{3.43}$$

Hence

$$t''(s) = -\frac{\{\mathbf{Q}'(t) \cdot \mathbf{Q}''(t)\}}{\{\mathbf{Q}'(t) \cdot \mathbf{Q}'(t)\}^2}. \tag{3.44}$$

Evaluating $t''(s)$ at $s = 0$ and $s = \mathcal{L}$ gives:

$$t''(0) = -\frac{\{\mathbf{Q}'(0) \cdot \mathbf{Q}''(0)\}}{\{\mathbf{Q}'(0) \cdot \mathbf{Q}'(0)\}^2}, \tag{3.45}$$

and

$$t''(\mathcal{L}) = -\frac{\{\mathbf{Q}'(1) \cdot \mathbf{Q}''(1)\}}{\{\mathbf{Q}'(1) \cdot \mathbf{Q}'(1)\}^2}, \tag{3.46}$$

respectively. Based on the following equalities:

$$\begin{aligned} h(0) &= t(0), & h(\mathcal{L}) &= t(\mathcal{L}), \\ h'(0) &= t'(0), & h'(\mathcal{L}) &= t'(\mathcal{L}), \\ h''(0) &= t''(0), & h''(\mathcal{L}) &= t''(\mathcal{L}), \end{aligned} \tag{3.47}$$

a system of equations is formulated in the following manner:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ \mathcal{L}^5 & \mathcal{L}^4 & \mathcal{L}^3 & \mathcal{L}^2 & \mathcal{L} & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 5\mathcal{L}^4 & 4\mathcal{L}^3 & 3\mathcal{L}^2 & 2\mathcal{L} & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 20\mathcal{L}^3 & 12\mathcal{L}^2 & 6\mathcal{L} & 2 & 0 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ g \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ t'(0) \\ t'(\mathcal{L}) \\ t''(0) \\ t''(\mathcal{L}) \end{bmatrix}. \tag{3.48}$$

Using simple linear algebra techniques, the six coefficients are solved for to give:

$$\begin{aligned} g &= 0, \\ e &= t'(0), \\ d &= t''(0)/2, \\ c &= \left\{\mathcal{L}^2 t''(\mathcal{L}) - 6d\mathcal{L}^2 - 12e\mathcal{L} + 20 - 8\mathcal{L}t'(\mathcal{L})\right\} / 2\mathcal{L}^3, \\ b &= \left\{3d\mathcal{L}^2 - \mathcal{L}^2 t''(\mathcal{L}) + 8e\mathcal{L} - 15 + 7\mathcal{L}t'(\mathcal{L})\right\} / \mathcal{L}^4, \\ a &= \left\{1 - b\mathcal{L}^4 - c\mathcal{L}^3 - d\mathcal{L}^2 - e\mathcal{L}\right\} / \mathcal{L}^5. \end{aligned} \tag{3.49}$$

Another way of solving this type of Hermite interpolation problem is by using the two point Taylor interpolation in [11, p. 37]. See Appendix A for details on this.

Note that the coefficients of these interpolating polynomials are only computed once for the curve to be approximated.

It is left to Chapter 4 to state observations about using the quintic, or higher-degree interpolating polynomials, rather than using $f(s)$, the cubic interpolating polynomial defined in Section 3.3.1.

3.4 Reference-Point Calculation

In generating reference points along the path of a curve $\mathbf{Q}(t)$, technical applications may require the spacing between adjacent reference points on the curve to be constant (uniform), linear, or even as a function of the curvature of the curve [23]. To compare the various algorithms for generating reference points, as discussed in Chapter 4, uniform spacing between adjacent reference points will be used in this thesis.

In this section, reference-point generation is considered by generating a uniformly spaced set of reference points along a linear curve. Chapter 4 contains a detailed discussion and application of several reference-point-generator algorithms as applied to general polynomial curves.

Consider the linear curve $\mathbf{Q}(t)$ shown in Figure 3.1. To generate a set of $N+1$ reference points $\mathbf{r}_i, i = 0, 1, \ldots, N$, uniformly spaced as shown in Figure 3.1, it is necessary to determine the value of each t_i, such that evaluating $\mathbf{Q}(t)$ at $t = t_i$, results in the correct location of the corresponding reference point $\mathbf{r}_i$.

Because the relationship between t and s in the above figure is linear (which

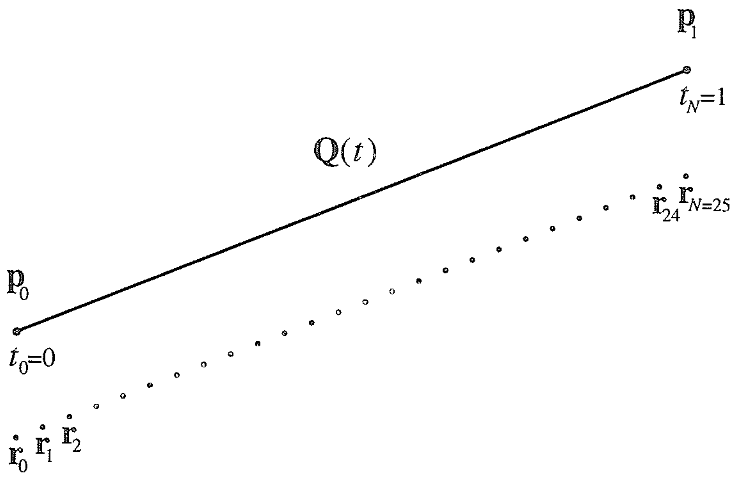

Figure 3.1: A linear Bézier curve $\mathbf{Q}(t)$ shown in the top of the figure. The lower dotted line represents a set of points $\mathbf{r}_i$ (translated down for visibility).

is true whenever the curvature is constant), each t_i is calculated in the following simple manner:

$$t_i = i\Delta s, \tag{3.50}$$

where $\Delta s = \frac{1}{N}$, and $i = 0, 1, \ldots, N$.

In the next chapter, we present methods to generate reference points for non-linear curves. That is, to determine each t_i that generates the corresponding $\mathbf{r}_i$ on the path of $\mathbf{Q}(t)$.

Chapter 4

Numerical Experiments with Reference-Point Generators

In this chapter, the various approximators of the function $t(s)$ are listed, compared and contrasted by formulating them into algorithms designed specifically to generate uniformly spaced reference points along curves.

These algorithms are called reference-point generators (RPGs), it is shown how the different RPGs discussed in this thesis generate points along these curves; the intention is to show how close the reference points generated by each RPG are to corresponding reference points generated by an actual arclength parameterization. Uniform spacing between adjacent reference points (RPs) is assumed throughout this chapter.

Chapter organization is as follows. Section 4.1 introduces the sample test curves. Section 4.2 lists and discusses the various RPG methods presented in this thesis in an algorithmic fashion. In Section 4.3, the result of applying the algorithms from Section 4.2 on the sample curves from Section 4.1 is graphically illustrated. Note that in sections 4.2 and 4.3, the cubic interpolator version of the proposed method (Section 3.3.1) is used. Section 4.4 discusses the result of applying some of the algorithms to curves of high degree, such as ninth-degree curves. The chapter is concluded in Section 4.5 with a general discussion on the reliability of the proposed method presented in Section 4.2.2.

4.1 Sample Test Curves

In each of the following figures, a number of curves is shown. Most curves shown are typical of curves used for general design purposes, others are included for purposes of illustration.

The dot shown on each curve indicates the starting control point. The control points for these curves are listed in Appendix B in normalized format (i.e., the total arclength of each curve is unity); note that this does not change the shapes of the curves shown since uniform scaling is an affine transformation. Affine transformations leave angles and line ratios unchanged [15] [25].

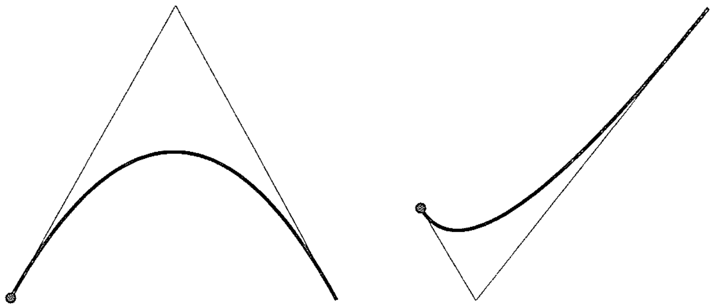

Figure 4.1: Quadratic curves. The one on the right was reproduced from [18].

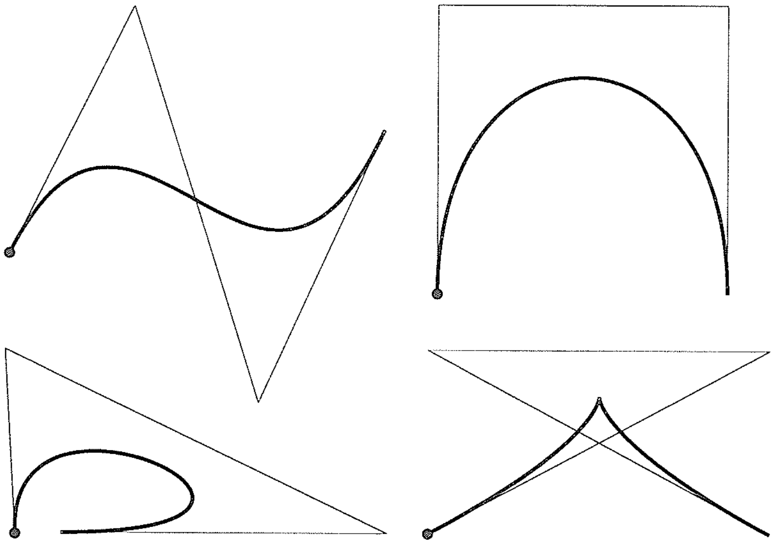

Figure 4.2: Cubic curves. The upper-right curve is a PH curve; the lower-right curve exhibits a cusp.

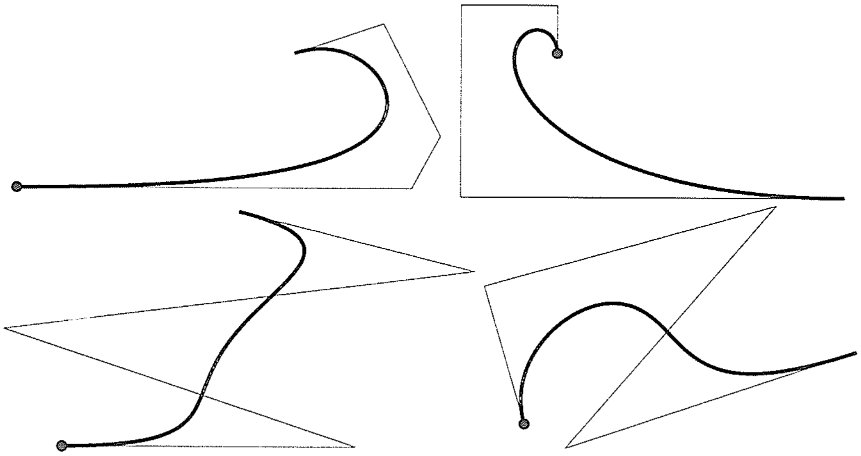

Figure 4.3: Quartic curves. The upper-left curve was reproduced from [18].

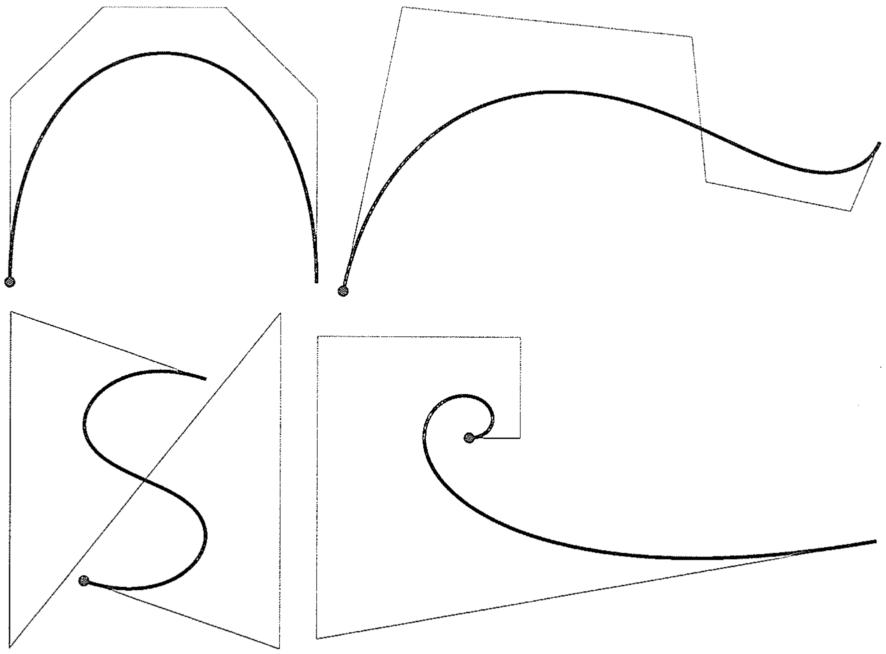

Figure 4.4: Quintic curves. The upper-right curve is a PH curves.

4.2 The Reference-Point-Generator Algorithms

This section lists the various RPG algorithms discussed in this thesis. The following definitions and identifiers apply to all algorithms, unless stated differently.

$N+1$: *the number of reference points (RPs) to be generated*

$\mathbf{Q}(t)$: *a parametric Bézier curve (see Eq. (2.1)),* $t \in [0,1]$

n: *curve degree*

$\mathcal{L}$: *the total arclength of the given curve*

ΔL: $\mathcal{L}/N$

$\mathbf{p}_i$: *Bézier control point,* $i = 0,1,2,\ldots,n$

$\mathbf{r}_i$: *the set of resulting RPs,* $i = 0,1,\ldots,N$, *where* $\mathbf{r}_0$ *and* $\mathbf{r}_N$ *are the* *points* $\mathbf{p}_0$ *and* $\mathbf{p}_n$*, respectively*

t_i*: the parameter values corresponding to the reference points,* $\mathbf{r}_i = \mathbf{Q}(t_i)$

It is assumed that $\mathcal{L}$ has been calculated to a satisfactory precision for algorithms that need it. Note that the parameterization of $\mathbf{Q}(t)$ is such that $t_0 = 0$ and $t_N = 1$ generate the RPs that lie on $\mathbf{p}_0$ and $\mathbf{p}_n$, respectively; so these two points are not regenerated. For convenience, the following abbreviations are used for the algorithms discussed herein.

BPF: basic parametric flow;

PM: proposed method (cubic interpolating function);

PM-5: proposed method (quintic interpolating function);

F-OP: Farouki's optimal parameterization;

ST: Sharpe & Thorne's algorithm;

CC: cumulative chordlength approximation;

MPM: modified proposed method (spline interpolating function); and

PHC: Pythagorean-hodograph curve algorithm.

The last algorithm in the above list is specific for PH curves. PH curves are included here for comparison, as an alternative to curves for which the arclength is not known in closed form.

It is left to Chapter 5 to examine the cost of running the following algorithms.

4.2.1 Basic Parametric Flow Algorithm

The BPF algorithm is the simplest of all the algorithms. Uniform increments of Δt of the parameter t are used to generate RPs along $\mathbf{Q}(t)$. The algorithm is written as follows.

```
RPG: BPF()
    $\Delta t \leftarrow 1/N, \;\; t_0 \leftarrow 0$
    for $i \leftarrow 1$ to $N-1$
        $t_i \leftarrow t_{i-1} + \Delta t$
        $\mathbf{r}_i \leftarrow \mathbf{Q}(t_i)$
END
```

Remarks on BPF

The BPF algorithm is the fastest method for RPs generation. However, because the distribution of RPs is highly affected by the shape of the curve, the BPF algorithm is not always useful in practice. It has been observed that uniform spacing of RPs is lost in variable curvature curves. RPs tend to have a high concentration in high curvature regions and a low concentration in low curvature regions, in spite of the uniform increments of Δt (see also Figure 1.1).

4.2.2 Proposed Method Algorithm

The PM algorithm is an implementation of the method proposed in this thesis (see Section 3.3.1). The idea is to generate a set $\{f_i \mid i = 1, 2, \ldots, N-1\}$ by the approximating interpolating function, such that evaluation of $\{\mathbf{Q}(f_i)\}$ renders reference points that are approximately uniformly spaced.

The PM algorithm starts by calculating the coefficients a, b, and c (derived in Section 3.3.1) of the cubic interpolating function. The function $f(s)$ in (3.27) is evaluated at $\{i\Delta L\}$ to yield $\{f_i\}$.

```
RPG: PM()
    compute $\mathcal{L}$, scale $||\mathbf{Q}'(0)||$ and $||\mathbf{Q}'(1)||$    {this scales $\mathbf{Q}(t)$}
    $c \leftarrow 1/||\mathbf{Q}'(0)||$
    $a \leftarrow c + 1/||\mathbf{Q}'(1)|| - 2$
    $b \leftarrow 1 - c - a$
    $\Delta L \leftarrow 1/N,\ \ell_0 \leftarrow 0$
    for $i \leftarrow 1$ to $N-1$
        $\ell_i \leftarrow \ell_{i-1} + \Delta L$
        $f_i \leftarrow ((a\ell_i + b)\ell_i + c)\ell_i$    {Eq. (3.27) using Horner's method}
        $\mathbf{r}_i \leftarrow \mathbf{Q}(f_i)$
END
```

where scaling $||\mathbf{Q}'(0)||$ and $||\mathbf{Q}'(1)||$ implies dividing them by $\mathcal{L}$, this scales the whole curve such that $\mathcal{L} = 1$.

Remarks on PM

For curves whose curvature is strongly variable, PM clusters RPs towards high curvature regions. However, it is a considerable improvement over BPF since for most practical curves, its spacing of RPs is close to being uniform.

Furthermore, as shown later in the figures in Section 4.3, the closeness of PM's RPs spacing to that produced by arclength parameterization is comparable to that produced by F-OP (discussed next). In some cases, especially when curves exhibit some symmetry, or when the lengths of polygon legs are not too disparate, PM's spacings were much closer to being uniform than those of the F-OP algorithm.

In practice, it is found that PM provides reliable results when $f(s)$ is a monotone, increasing function, because it is approximating a monotone, increasing function

$t(s)$. If, however, $f(s)$ is not monotone, then it cannot provide a good approximation of $t(s)$, and should therefore not be used. This is discussed further in Section 4.5.

4.2.3 Farouki's Optimal Parameterization Algorithm

Farouki's optimal parameterization (F-OP) is mathematically a rather intricate process [18]. The given polynomial curve $\mathbf{Q}(t)$ is first transformed into an equivalent rational form by transforming the parameter t in (2.1) (by applying a Möbius transformation) as follows:

$$t = \frac{(1-\alpha)u}{\alpha(1-u)+(1-\alpha)u}, \qquad 0<\alpha<1, \quad 0 \le u \le 1. \tag{4.1}$$

Substituting (4.1) into (2.1) results in the following rational form, equivalent to (2.1):

$$\tilde{\mathbf{Q}}(u) = \frac{\sum_{i=0}^{n} w_i \mathbf{p}_i B_{i,n}(u)}{\sum_{i=0}^{n} w_i B_{i,n}(u)}, \qquad w_i = (1-\alpha)^i \alpha^{n-i}, \tag{4.2}$$

where $B_{i,n}(u)$ is defined by (2.2). The objective is to find the set of weights $\{w_i\}$ so that u approximates an arclength parameter. The problem is thus to find the "best" value for α in (4.2).

```
RPG: F-OP()
  compute $\mathcal{L}$, normalize $\mathbf{p}_{i \leftarrow 0,1,\ldots,n}$
  for $i \leftarrow 0$ to $n-1$
      $\Delta\mathbf{p}_i \leftarrow \mathbf{p}_{i+1} - \mathbf{p}_i$
  for $i \leftarrow 0$ to $n$
```

```
        $\mathbf{c}_i \leftarrow i(n-i+1)\Delta\mathbf{p}_{i-1} - (i+1)(n-i)\Delta\mathbf{p}_i$
    for $i \leftarrow 0$ to $2n-1$
        $b_i \leftarrow \sum_{j=\max(0,i-n+1)}^{\min(n,i)} \frac{\binom{n}{j}\binom{n-1}{i-j}}{\binom{2n-1}{i}} (\mathbf{c}_j \cdot \Delta\mathbf{p}_{i-j})$
    for $i \leftarrow 4$ to $2n+3$
        $r \leftarrow 2n+3-i$
        $C_0^r \leftarrow \sum_{k=0}^{2n-1-r} (-1)^k \binom{2n-1-r}{k} b_{k+r}$
        $C_r^r \leftarrow \sum_{k=0}^{2n-1-r} (-1)^{r+k} \binom{2n-1-r}{k} b_k$
    $q_0 \leftarrow \sum_{i=4}^{2n+3} \frac{1}{i-1} \binom{2n-1}{i-4} C_0^{2n+3-i}$
    $q_{2n+2} \leftarrow -\sum_{i=4}^{2n+3} \frac{1}{i-1} \binom{2n-1}{i-4} C_{2n+3-i}^{2n+3-i}$
    $\alpha \leftarrow \left(1 + \sqrt{-q_{2n+2}/q_0}\right)^{-1}$
    for $i \leftarrow 0$ to $n$
        $\mathbf{p}_i \leftarrow \mathbf{p}_i \mathcal{L}$
    $\Delta u \leftarrow 1/N, \quad u_0 \leftarrow 0$
    for $i \leftarrow 1$ to $N-1$
        $u_i \leftarrow u_{i-1} + \Delta u$
        $\mathbf{r}_i \leftarrow \tilde{\mathbf{Q}}(u_i)$        {using Eq. (4.2)}
END
```

Remarks on F-OP

As indicated in [18], F-OP produces its best results when it is applied to a Bézier curve whose control polygon legs are of disparate lengths; the more disparate the lengths of the legs, the closer is the uniformity of spacing of RPs to that produced by arclength parameterization.

On the other hand, it has been observed that when F-OP is used on a curve whose lengths of polygon legs are not much different, the distribution of RPs resembles that which is produced by BPF; the distribution is the same as that of BPF when applied to a symmetric curve.

The above observations may be explained as follows. For symmetric curves, the value of α is $1/2$. This implies that the set of weights $\{w_i\}$ will have no contribution since $w_i = (1/2)^n \, \forall \, \mathbf{r}_i$; equal w_i's imply that the rational function $\tilde{\mathbf{Q}}(u)$ becomes a Bézier polynomial of degree n, and the result is similar to that produced by BPF algorithm.

4.2.4 Sharpe & Thorne's Algorithm

The idea behind the (ST) algorithm is simple and produces desired results with high accuracy [40]. However, it is a computationally expensive method and may not be practical for real-time processing requirements.

The idea is to find t_i's such that the point $\mathbf{Q}(t_i)$ is exactly ΔL along the curve from $\mathbf{Q}(t_{i-1})$. In this regard, t_i is a root of a non-linear equation that is to be solved by applying Newton-Raphson iterations until convergence to a satisfactory tolerance is achieved. In addition, a suitable systematic way of providing an initial value to initiate the search for each t_i is usually required.

The following definitions are needed. Define $M(t)$ as follows:

$$M(t) = \int_{t_{i-1}}^{t} \|\mathbf{Q}'(\tau)\| \, \mathrm{d}\tau - i\Delta L, \tag{4.3}$$

where t_{i-1} is the parametric value that generated the last reference point, and $i\Delta L$ is the distance sought. The derivative of $M(t)$ is

$$M'(t) = \|\mathbf{Q}'(t)\|. \tag{4.4}$$

The task is to find t_i such that $M(t_i) = 0$. The following means of determining an initial approximation, ε_i, to initiate the search for t_i seems to work well in practice.

$$\varepsilon_i = \max(i\Delta t,\, t_{i-1}). \tag{4.5}$$

The algorithm may now be written as follows.

```
RPG: ST()
    compute $\mathcal{L}$,   $t_0 \leftarrow 0$
    $\Delta t \leftarrow 1/N$,   $\Delta L \leftarrow \mathcal{L}/N$
    for $i \leftarrow 1$ to $N-1$
        $t_i \leftarrow \varepsilon_i$
        repeat
            $t_i \leftarrow t_i - \frac{M(t_i)}{M'(t_i)}$
        until convergence to a given tolerance
        $\mathbf{r}_i \leftarrow \mathbf{Q}(t_i)$
END
```

Remarks on ST

Of all the algorithms discussed, the ST algorithm is closest to producing RPs with accurate uniform spacing. It has been observed that six or fewer Newton-Raphson iterations usually yield 7–9 digits of accuracy. However, it is also the most computationally intensive method (illustrated later in Chapter 5). This is due to the number of Newton-Raphson iterations needed for convergence, in which the integral (4.3) is computed many times. In addition, for every evaluation of (4.3), the function $\|\mathbf{Q}'(t)\|$ has to be evaluated many times. In practice, it was found that four Simpson intervals[1] for every segment provided sufficient accuracy.

4.2.5 Cumulative Chordlength Algorithm

Cumulative chordlength is a straightforward method to approximate the arclength of a curve. This method can be exploited to generate RPs that visually seem to be uniformly spaced.

The algorithm is as follows: while computing the arclength, the set $\{s_k \,|\, k = 0, 1, \ldots, \Im\}$ ($\Im$ being the number of chords used to approximate the curve) keeps track of the cumulative chordlength thus far. Reference points at distances

$$\{i\Delta d \,|\, i = 1, 2, \ldots, N-1 \text{ and } \Delta d = s_\Im/N\},$$

where $s_\Im$ is the total chordlength, may then be located by searching for their closest values in $\{s_k\}$, and then further refining those values by means of linear interpolation. That is, t_i, the parameter value for the ith reference-point $\mathbf{r}_i$, at distance

[1] In [40], Sharpe & Thorne suggest using the Romberg algorithm, but this does not change the number of integral evaluations in the algorithm.

$i\Delta d$, is approximated by the function $A(i,k)$ as follows:

$$t_i = A(i,k) = \Delta u \left(k - 1 + \frac{i\Delta d - s_{k-1}}{s_k - s_{k-1}} \right), \tag{4.6}$$

where $s_{k-1} \leq i\Delta d < s_k$, and $\Delta u = 1/\Im$. An algorithm that implements this follows.

```
RPG: CC()
    $\mathbf{v}_0 \leftarrow \mathbf{p}_0, \quad s_0 \leftarrow 0$
    $\Delta u \leftarrow 1/\Im, \quad u_0 \leftarrow 0$
    for $i \leftarrow 1$ to $\Im$
        $u_i \leftarrow u_{i-1} + \Delta u$
        $\mathbf{v}_i \leftarrow \mathbf{Q}(u_i)$
        $s_i \leftarrow s_{i-1} + \|\mathbf{v}_i - \mathbf{v}_{i-1}\|$
    $\Delta d \leftarrow s_\Im / N, \quad k \leftarrow 0$
    for $i \leftarrow 1$ to $N$–1
        while $i\Delta d \geq s_k$
            $k \leftarrow k + 1$
        $t_i \leftarrow A(i,k)$        {*using Eq. (4.6)*}
        $\mathbf{r}_i = \mathbf{Q}(t_i)$
END
```

Remarks on CC

In order for the CC algorithm to produce uniform RPs, the number of chords $\Im$ used to approximate the curve may have to be quite high. On average, in the examples used in practice, setting $\Im$ to 64 usually produced a parameter set $\{t_i\}$ that agreed with that of Sharpe & Thorne to the second-least significant decimal digit (this implies that results are visually indistinguishable). Nevertheless, this algorithm had a much faster running time than the ST algorithm, due to the simplicity of the computations carried out.

4.2.6 Pythagorean-Hodograph Curve Algorithm

For PH curves of odd degree n, a polynomial expression is available for $\sigma(t) = s'(t) = \|\mathbf{Q}'(t)\|$, which is used in the denominator in the Newton-Raphson iteration. It is a special case of the ST algorithm since it depends on Newton-Raphson iterations; however, up to 12 digits of precision are often attained in no more than two or three iterations [17].

Starting with the following definitions (see (3.9)):

$$s(t_i) = i\Delta L = \int_{t_0}^{t_i} \sigma(\tau)\mathrm{d}\tau, \qquad i = 1, 2, \dots, N-1, \tag{4.7}$$

where

$$\sigma(t) = \sum_{i=0}^{n-1} \sigma_i \begin{pmatrix} n-1 \\ i \end{pmatrix} (1-t)^{n-1-i} t^i, \tag{4.8}$$

and $\sigma_{i=0,1,\dots,n-1}$ is given by (3.11). The complete algorithm follows.

```
RPG: PHC()
    compute $\sigma_{i \leftarrow 0,1,\dots,n-1}$        {using Eq. (3.11)}
    for $i \leftarrow 1$ to $N-1$
        $t_i \leftarrow t_{i-1} + \Delta L / \sigma(t_{i-1})$
        repeat
            $t_i \leftarrow t_i - \{s(t_i) - i\Delta L\}/\sigma(t_i)$
        until convergence to a given tolerance
        $\mathbf{r}_i = \mathbf{Q}(t_i)$
END
```

Remarks on PHC

The algorithm produces RPs with accurate uniform spacing, with less computational requirements than needed by the ST algorithm. However, it does not serve as a general purpose algorithm, since the curve in which it is locating RPs has to be a PH. It is therefore omitted from further discussions in Chapter 5.

4.3 Visual Results

In this section, the algorithms discussed thus far are applied to the curves presented in Section 4.1. The objective is to show how close the reference points generated by the various methods are to the exact reference points, and how the results of the PM (proposed method) developed in this dissertation compare to those of other methods.

Each of the following figures is organized as follows. The actual curve is shown first, followed by plots showing only RPs (reference points) along their translated paths. In each case, four RP plots are shown: those resulting from BPF (basic parametric flow), ST (Sharpe & Thorne), F-OP (Farouki's optimal parameterization), and from PM.

The ST algorithm produces RPs with very accurate results (the parameter set $\{t_i\}$ that generated RPs using ST was tested against that produced by the PHC algorithm on PH curves, and 6–8 digits of accuracy were yielded in four or five iterations). The ST reference points are considered to be exact and thus are used to compare other results with.

RP plots resulting from the PHC (PH curve) algorithm are not shown since they are only applicable to PH curves.

RP plots resulting from the CC algorithm with different numbers of chords are shown at the end of this section in Figure 4.19.

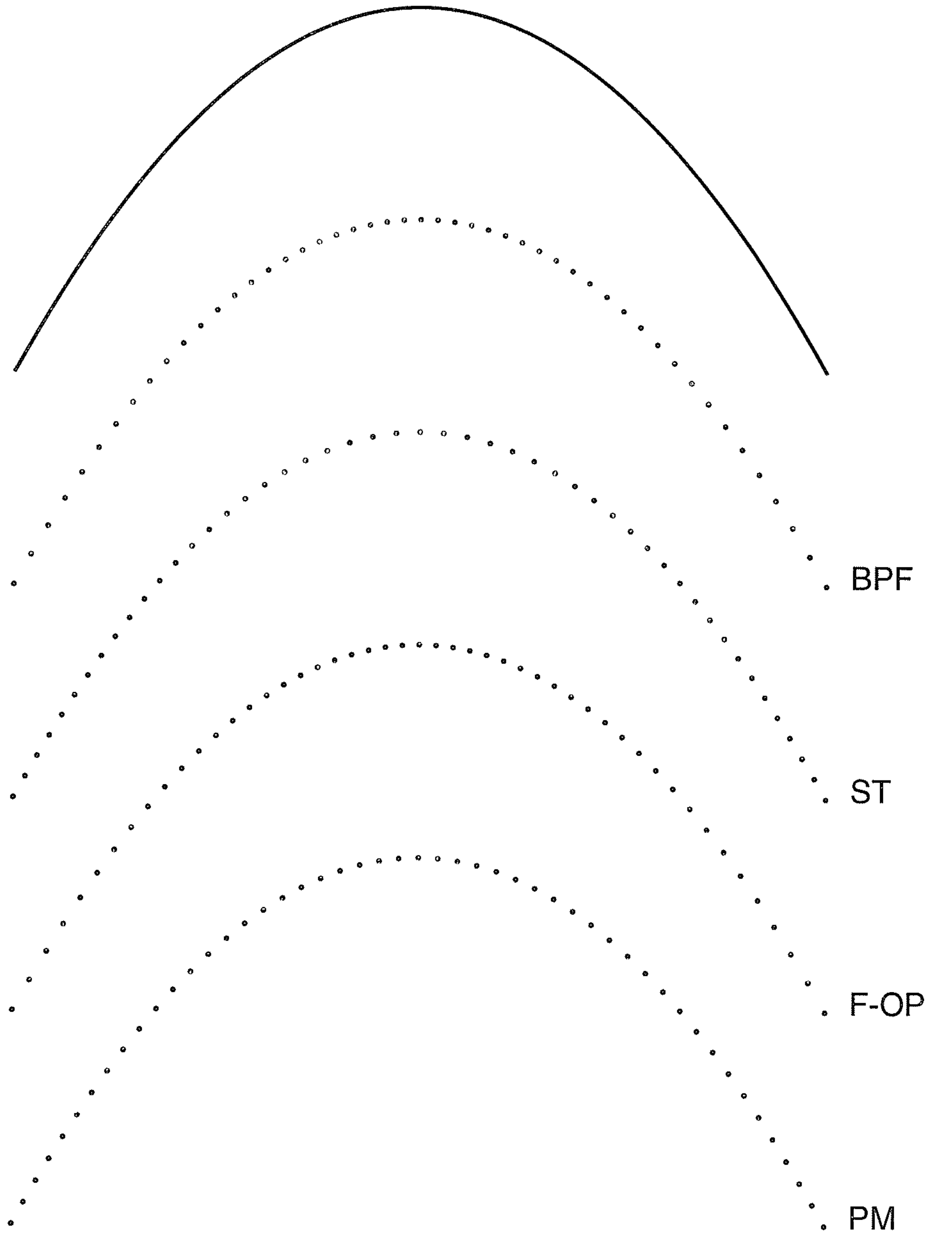

Figure 4.5: The quadratic curve shown here is from Figure 4.1.

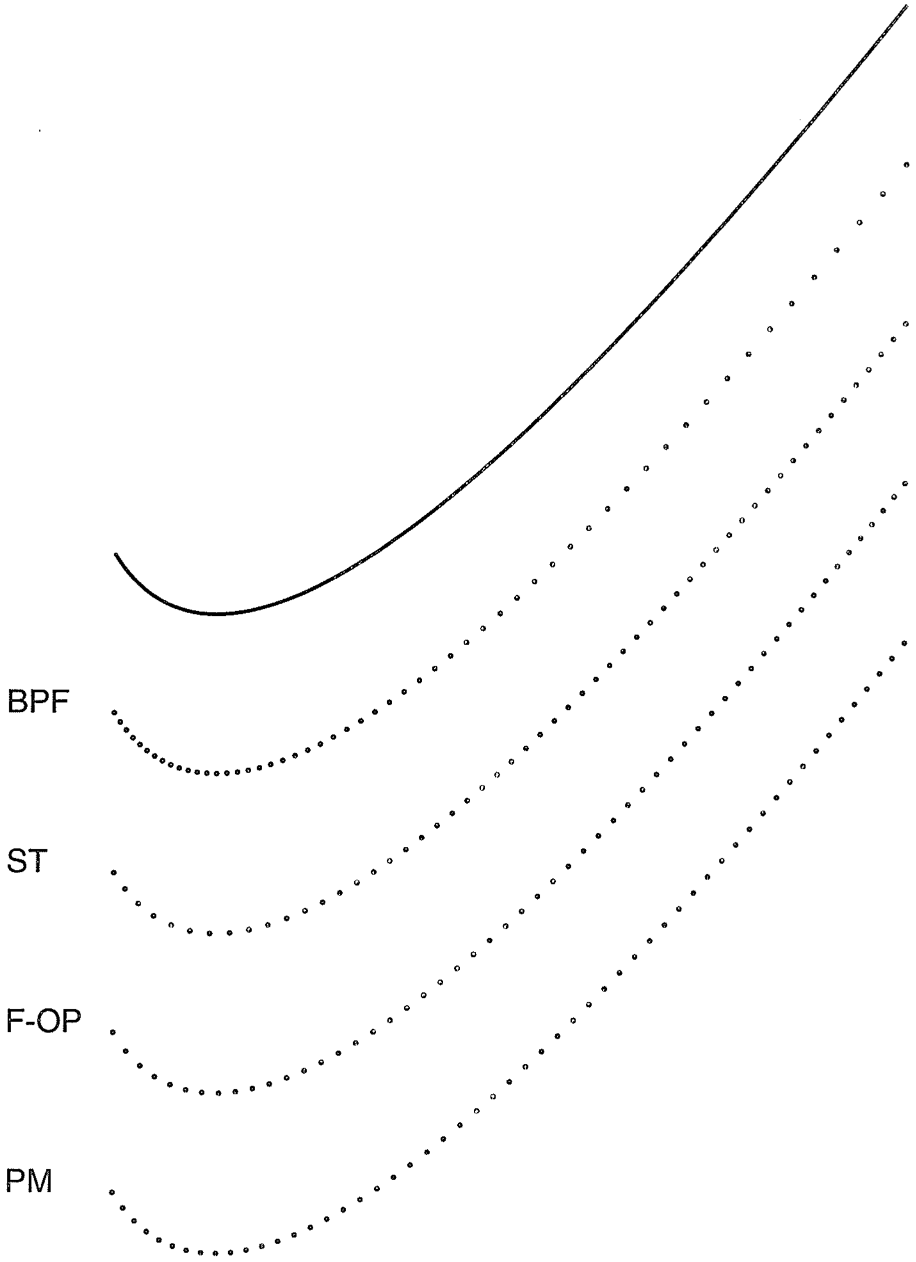

Figure 4.6: The other quadratic curve shown from Figure 4.1. Also shown in [18].

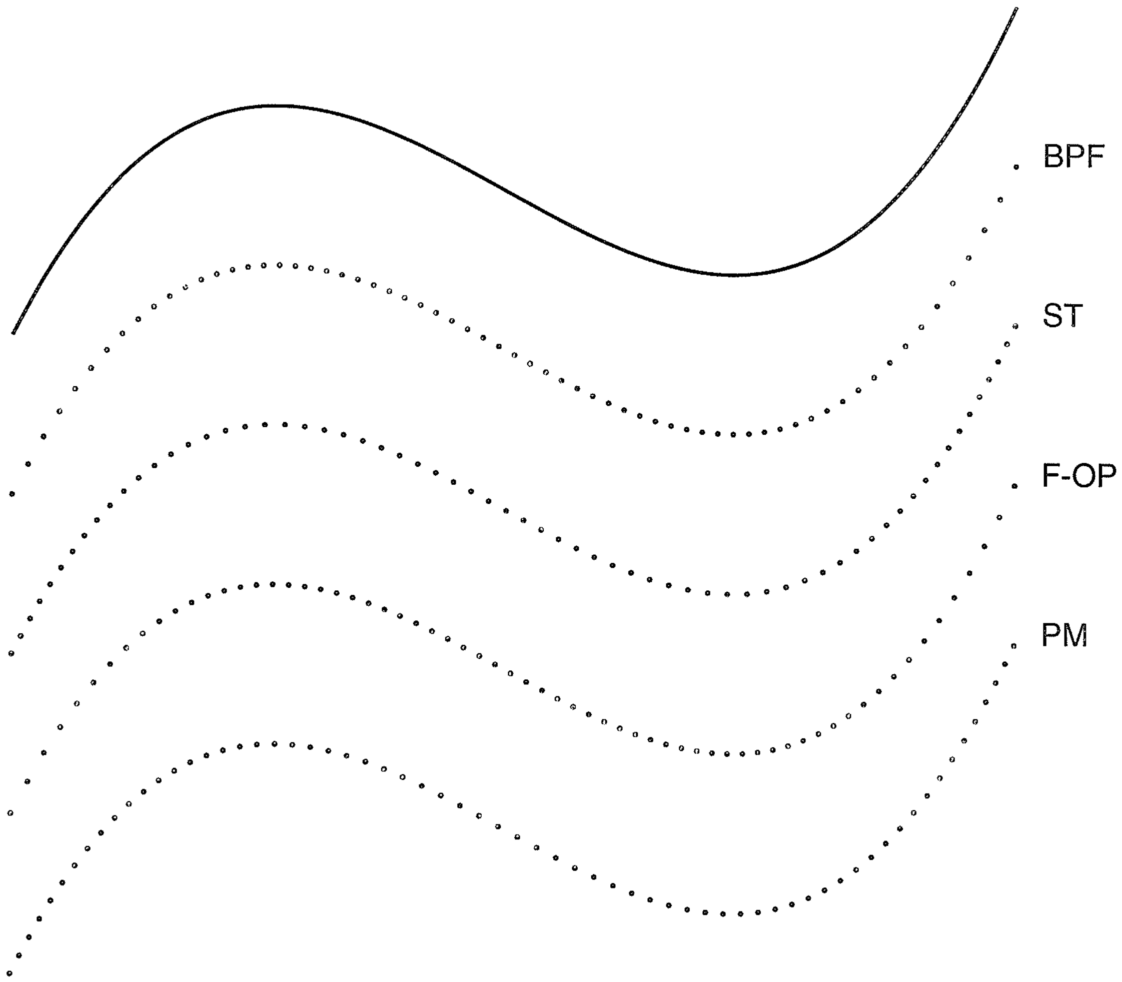

Figure 4.7: Cubic S-curve from Figure 4.2.

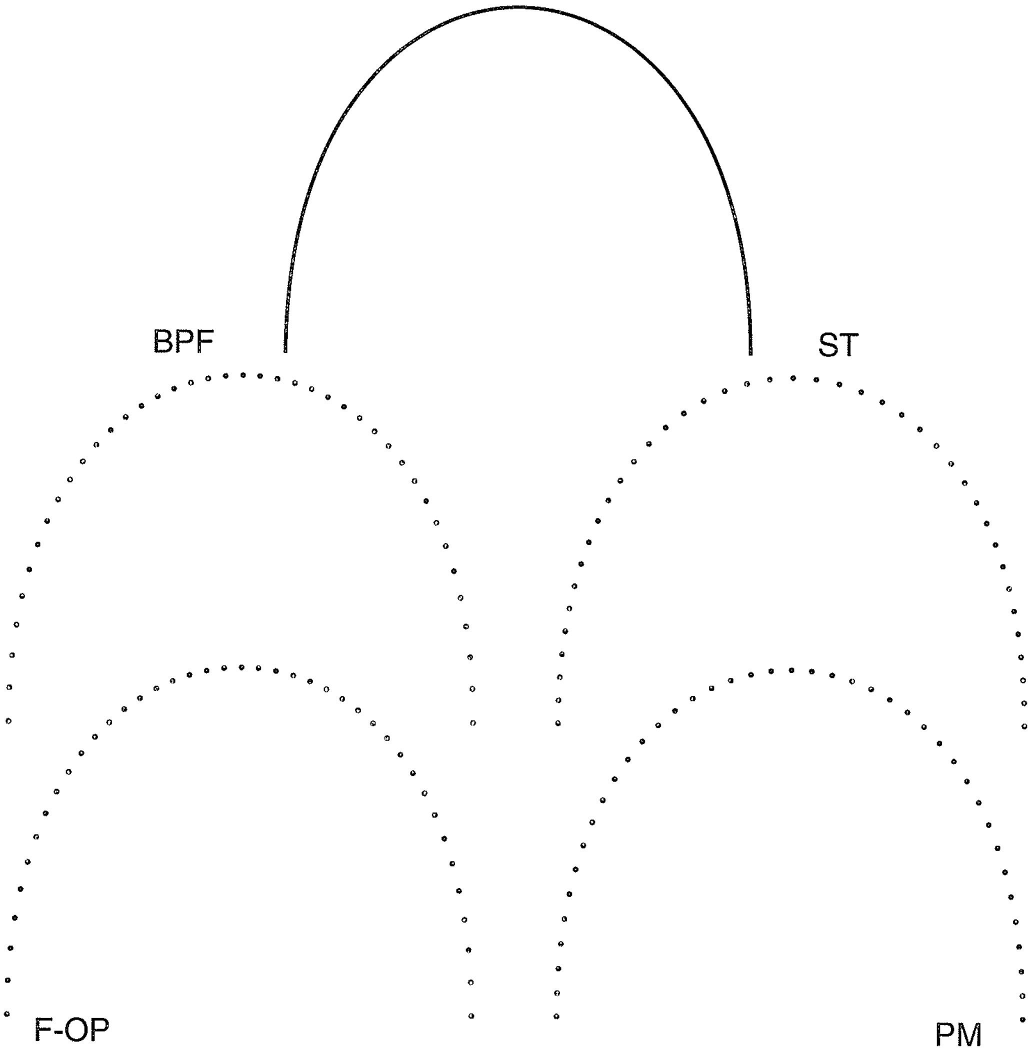

Figure 4.8: The PH curve from Figure 4.2; notice similarity in RPs spacing of BPF and F-OP. This similarity exists for all symmetric curves.

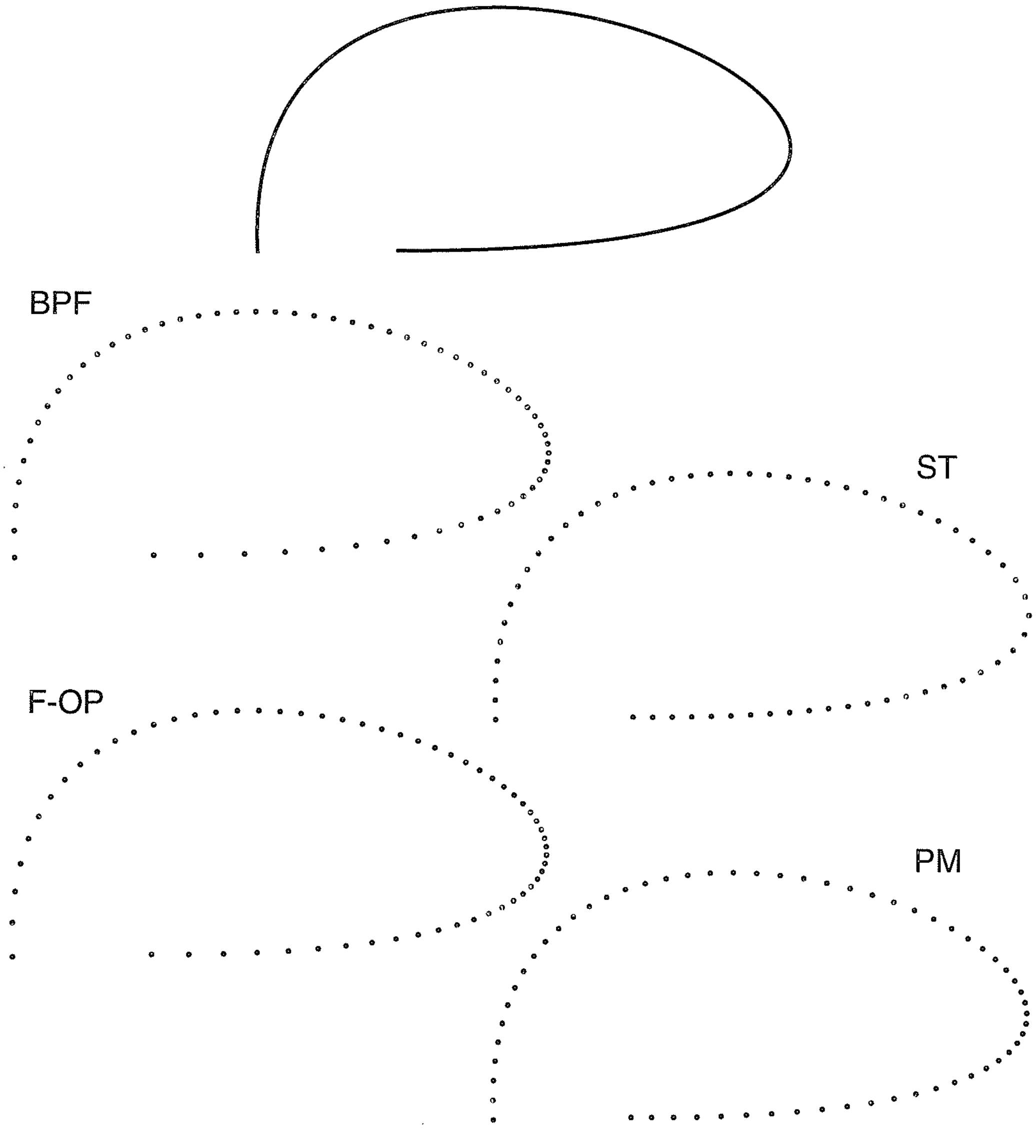

Figure 4.9: The third cubic curve from Figure 4.2 exhibiting high curvature regions.

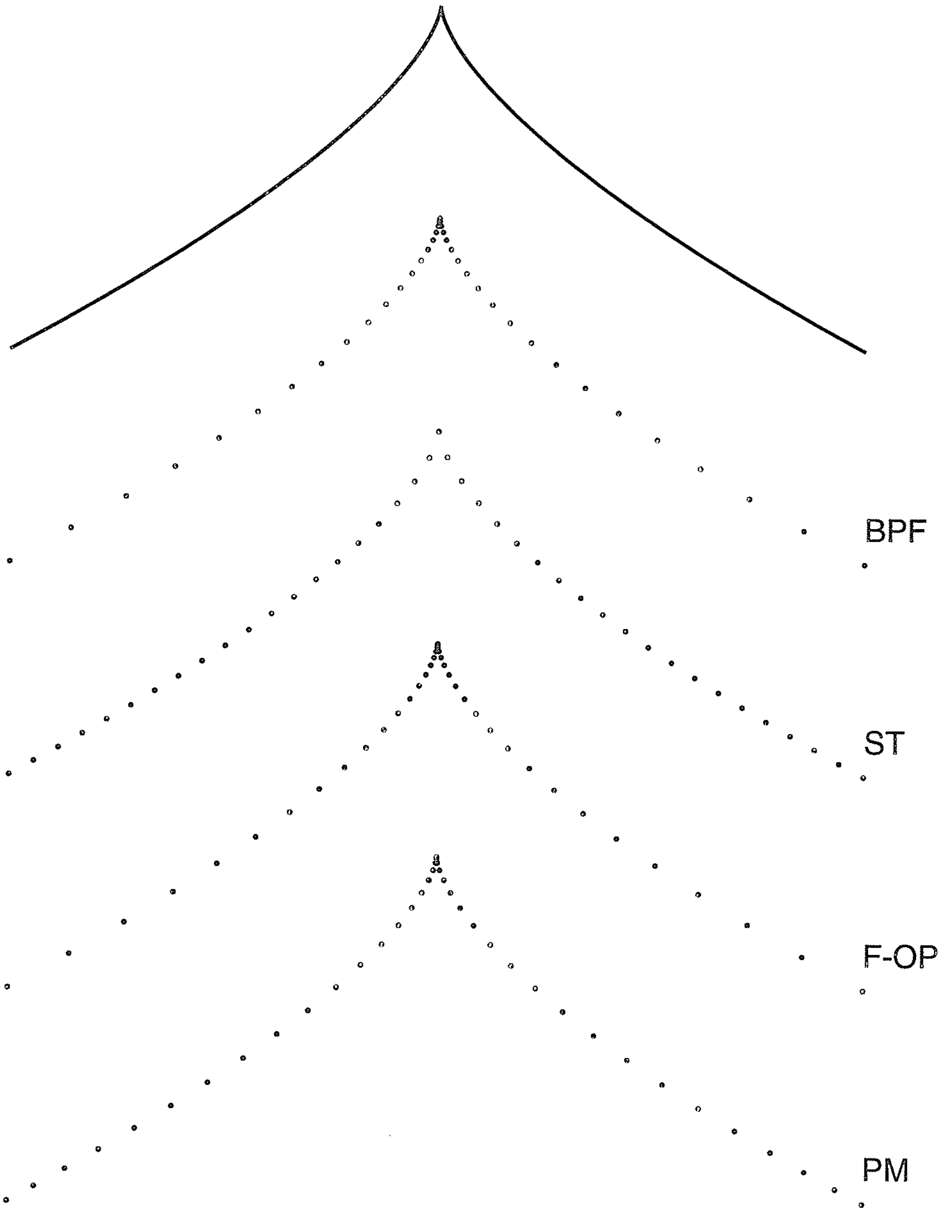

Figure 4.10: The cubic curve exhibiting a cusp from Figure 4.2.

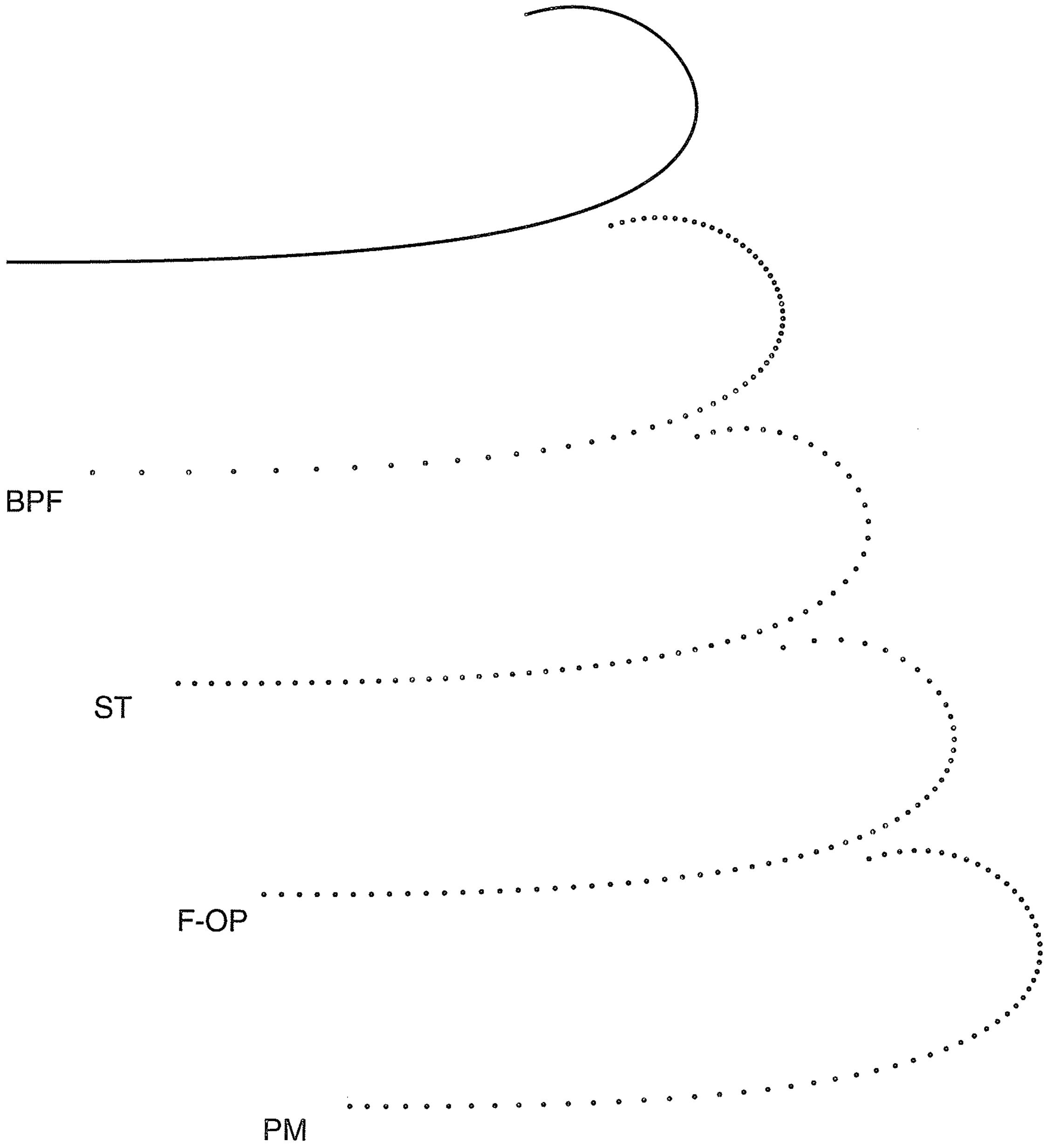

Figure 4.11: The first quartic curve from Figure 4.3; also shown in [18].

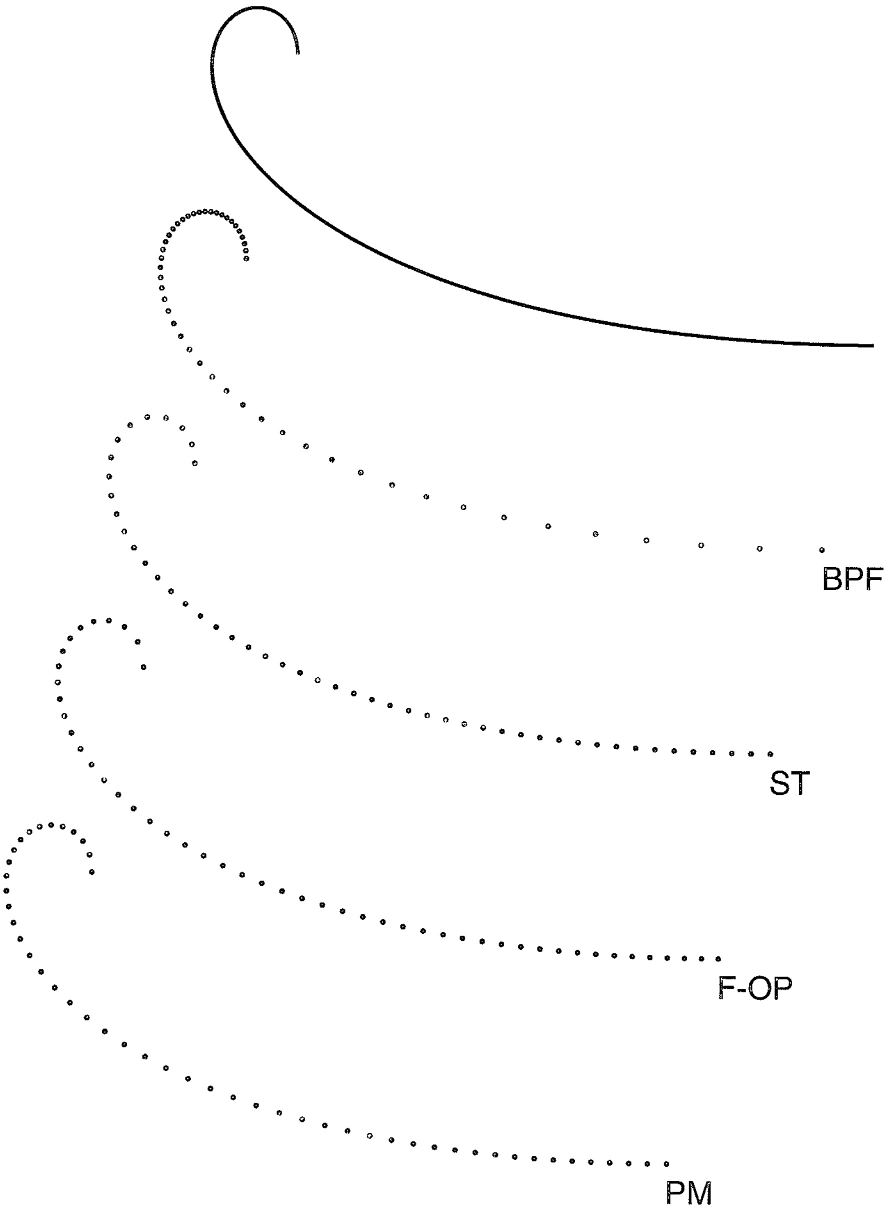

Figure 4.12: A quartic curve with a high curvature region. From Figure 4.3.

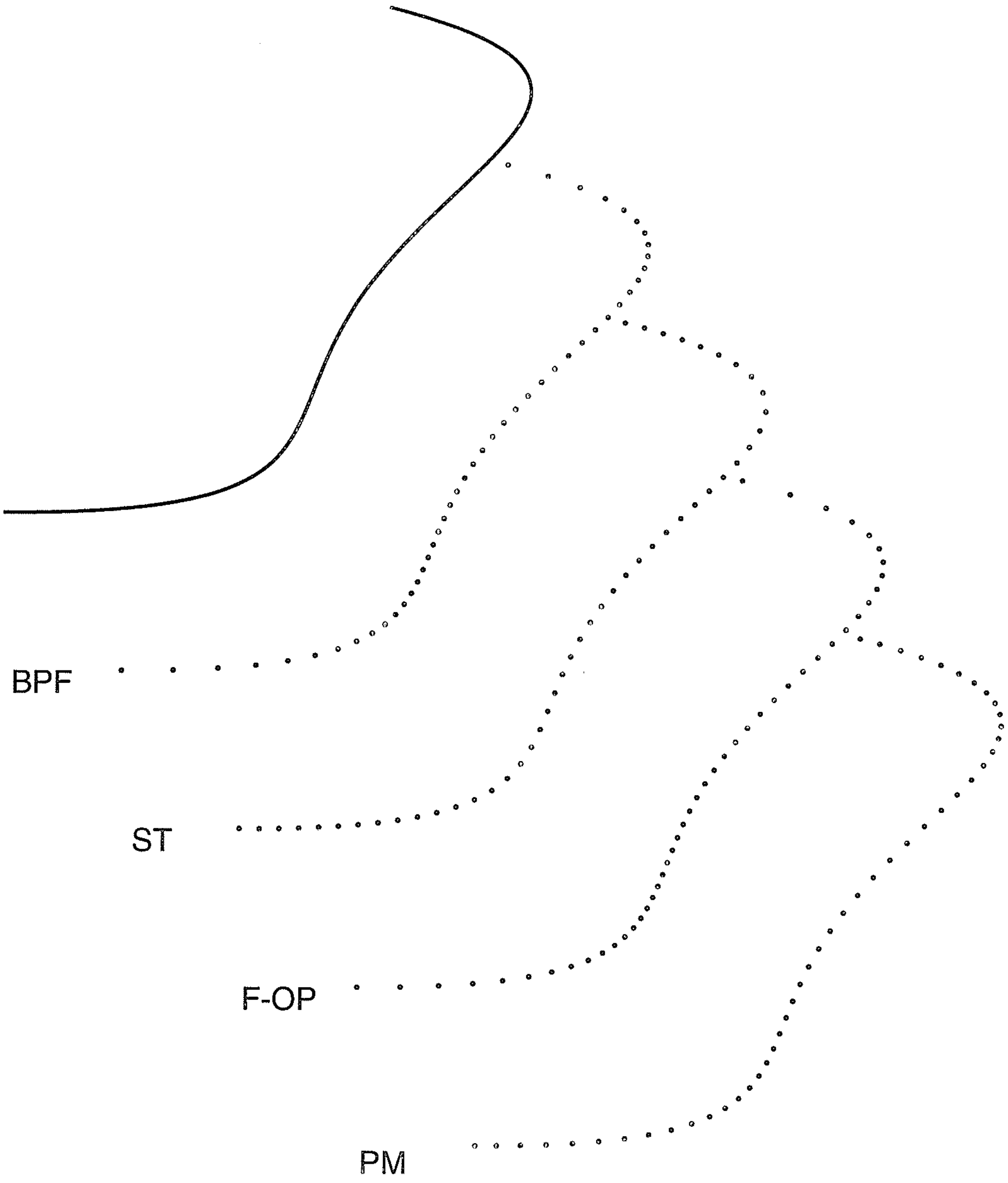

Figure 4.13: A quartic M-curve exhibiting 2 inflection points. From Figure 4.3.

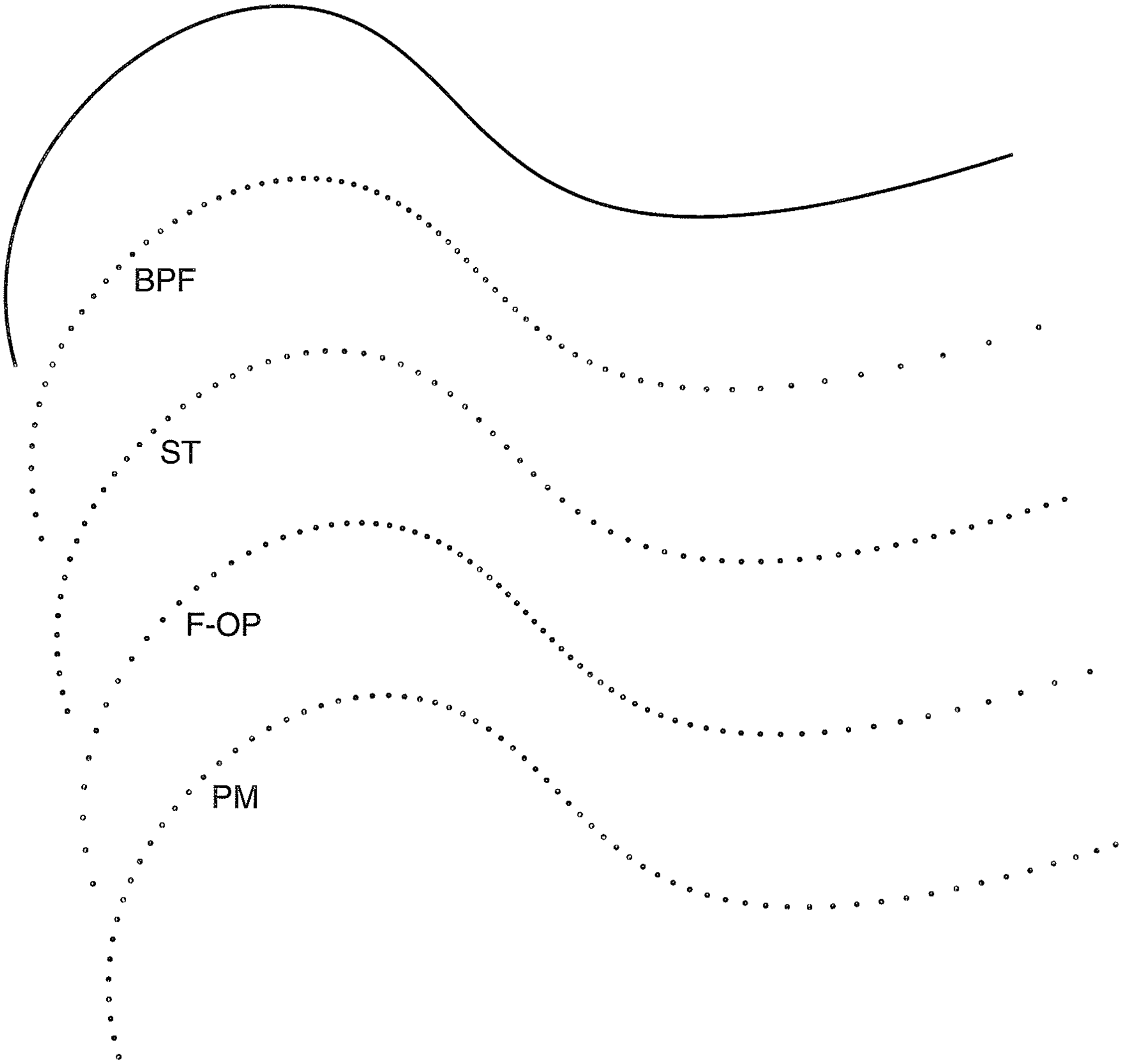

Figure 4.14: The last quartic curve from Figure 4.3.

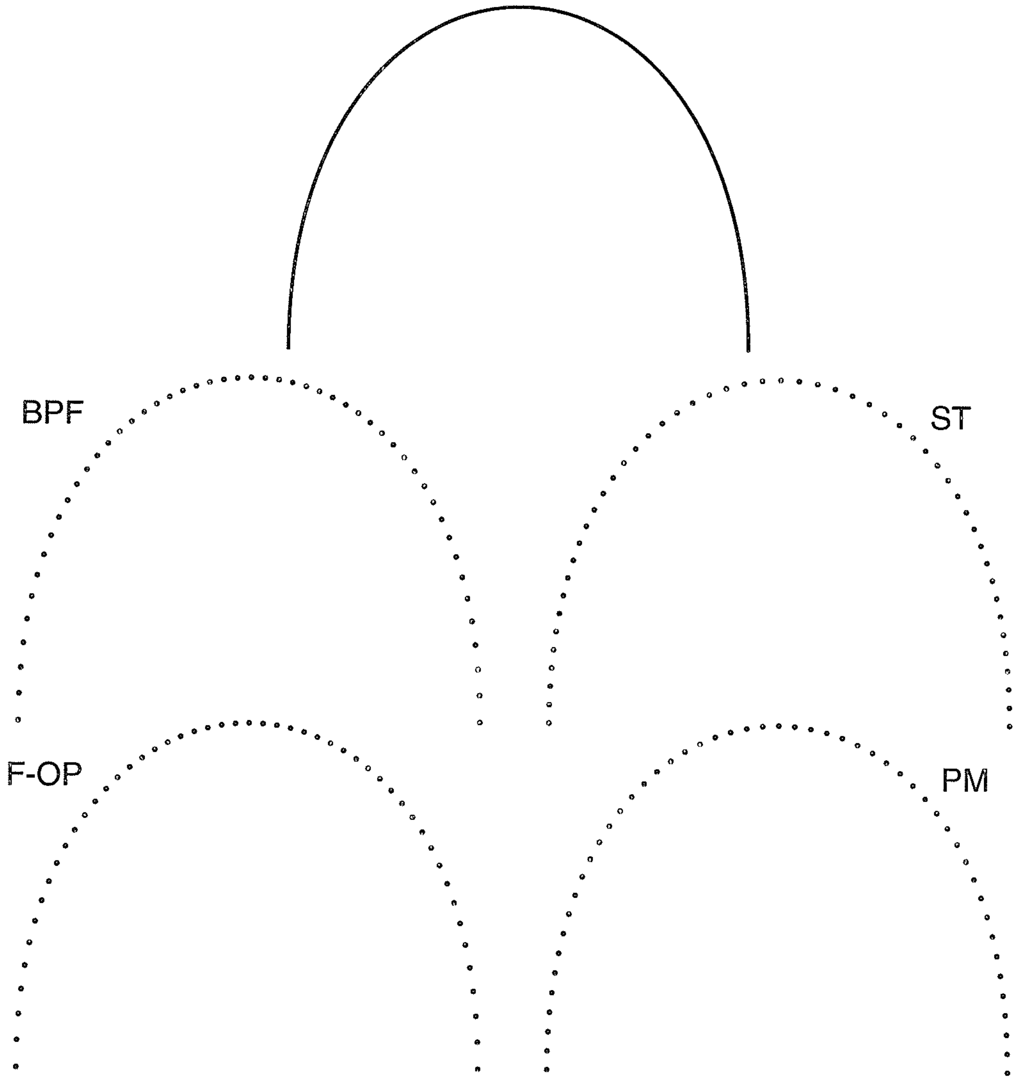

Figure 4.15: A symmetric quintic C-curve from Figure 4.4.

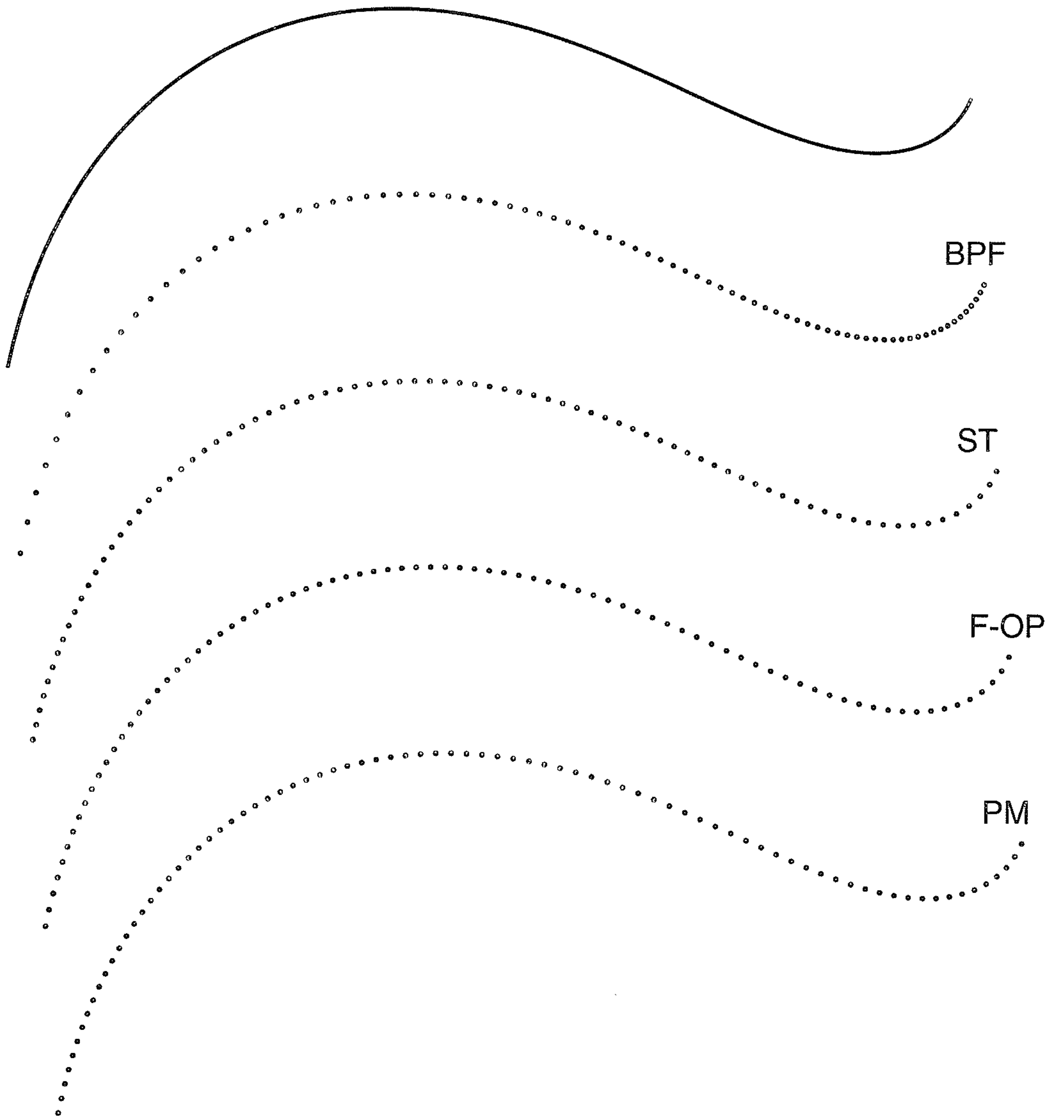

Figure 4.16: The quintic PH curve from Figure 4.4.

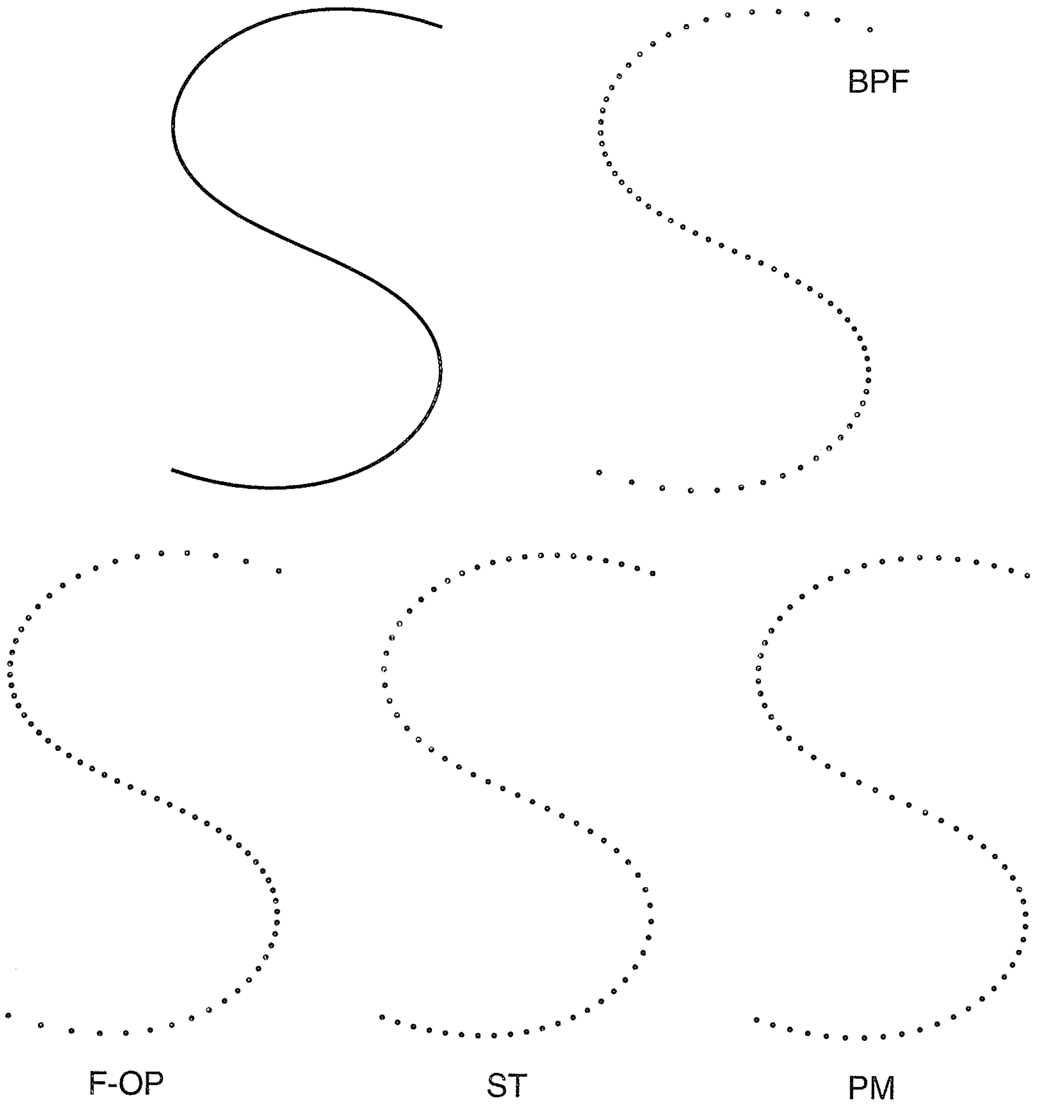

Figure 4.17: The quintic S-curve from Figure 4.4.

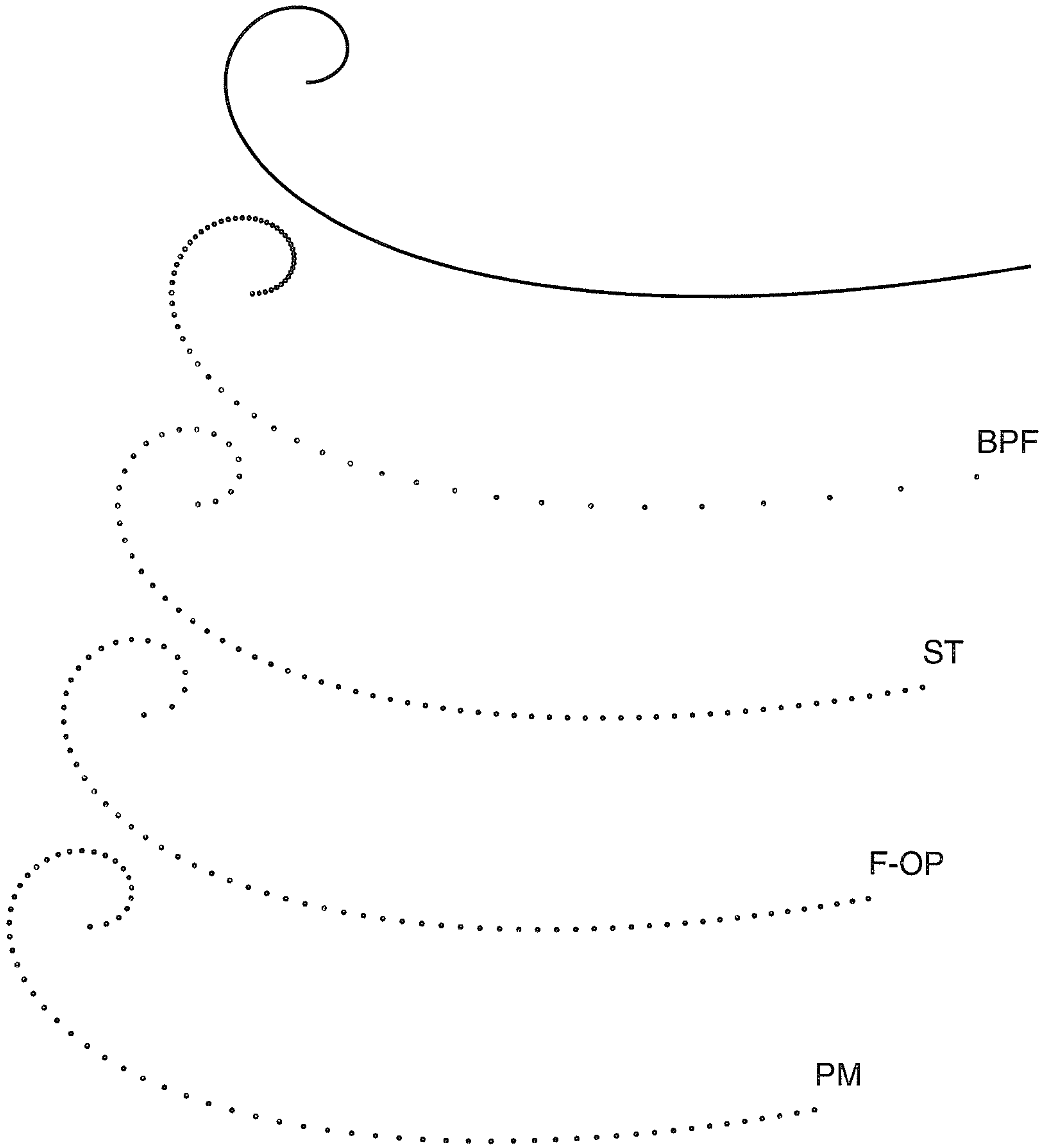

Figure 4.18: The quintic curve from Figure 4.4.

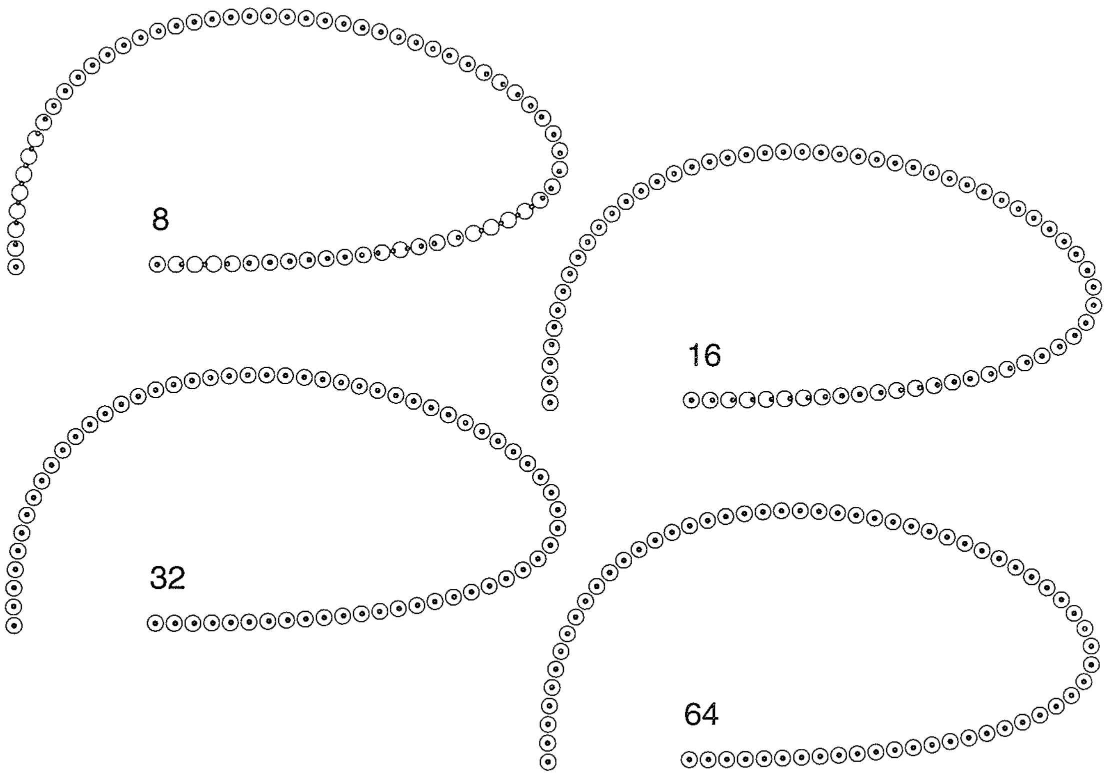

Figure 4.19: In this figure, the circles denote RPs generated by the ST algorithm, while the dots denote RPs generated by the CC algorithm. The number shown on each plot indicates the number of chords used by the CC algorithm. Careful inspection of these plots shows that the 64 chords plot is the only one where both sets of RPs visually agree.

4.4 High-Degree Curves

Since the PM depends largely on the tangent vectors at the end points to perform its approximation of arclength parameterization, it is expected to produce good visual results for quadratic and cubic curves. However, it also seems to perform reasonably well for quartic and quintic curves (as shown in the previous figures), perhaps because the behavior of the curve away from the end tangent vectors is still sufficiently influenced by the information at the end points.

Figures 4.20 and 4.21 illustrate the above observations for yet higher degree curves. It it emphasized however, that typical design curves do not usually exceed a quintic degree.

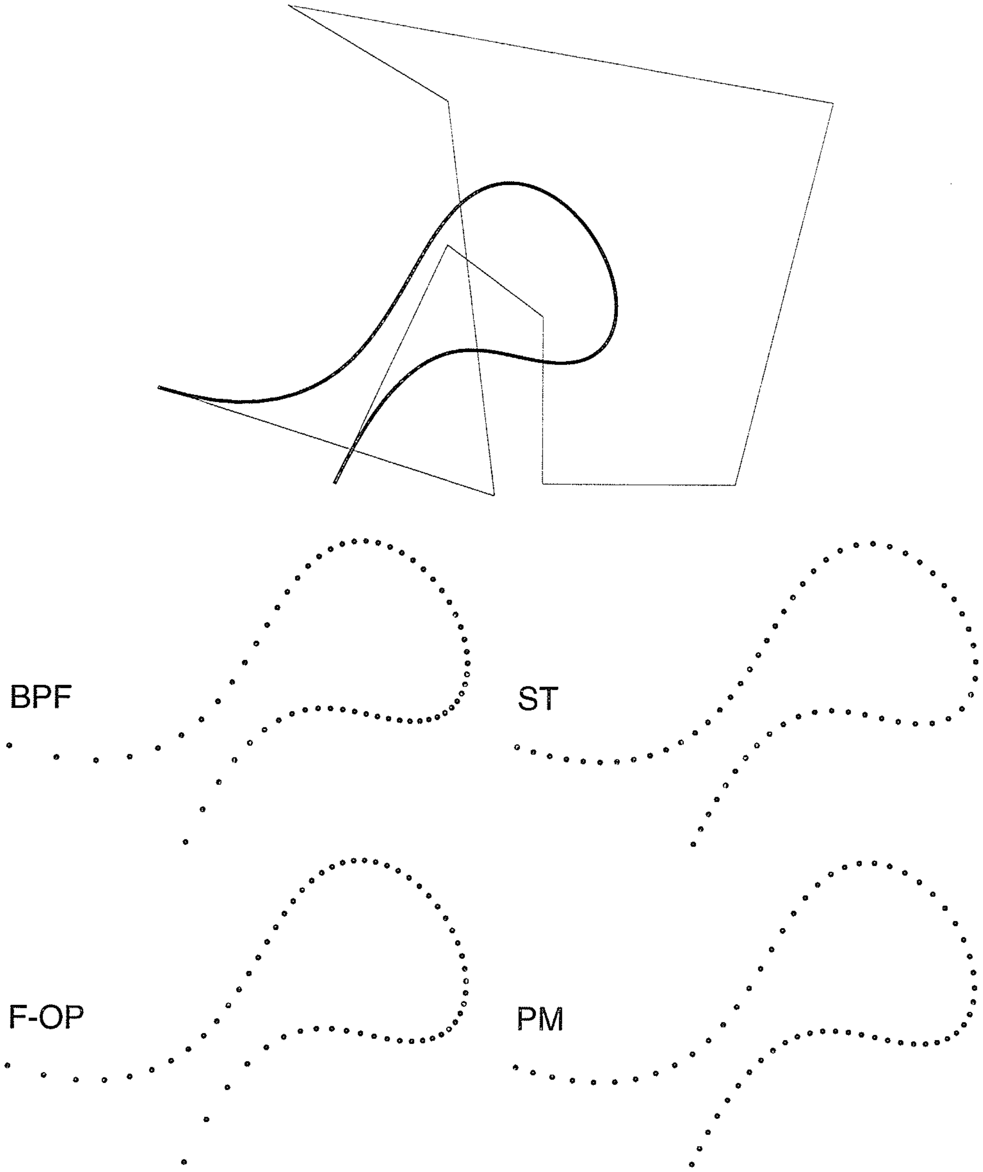

Figure 4.20: A ninth-degree curve.

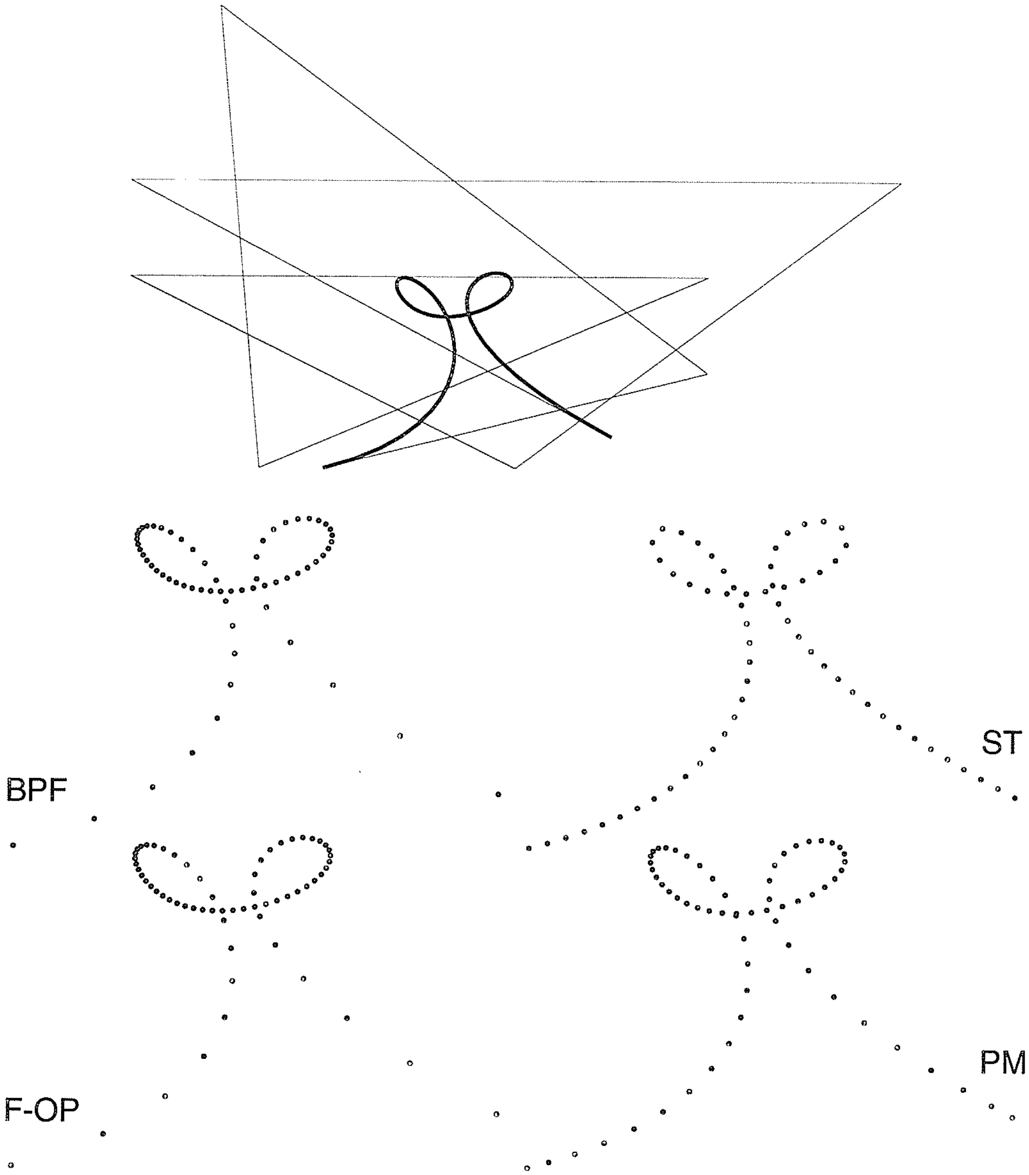

Figure 4.21: Another ninth-degree curve exhibiting two loops.

4.5 Higher-Degree Approximation

In Section 3.3.2, an interpolating function of fifth-degree was presented. This interpolating method is called PM-5. The belief was that more information on the curve's behavior at the end points would offer a better approximation. However, the representative sample of curves on which tests were performed (shown in figures 4.22[2], 4.23, and 4.24) indicates that, even though it may produce a better approximation of $t(s)$ for some curves, it is more error prone for others.

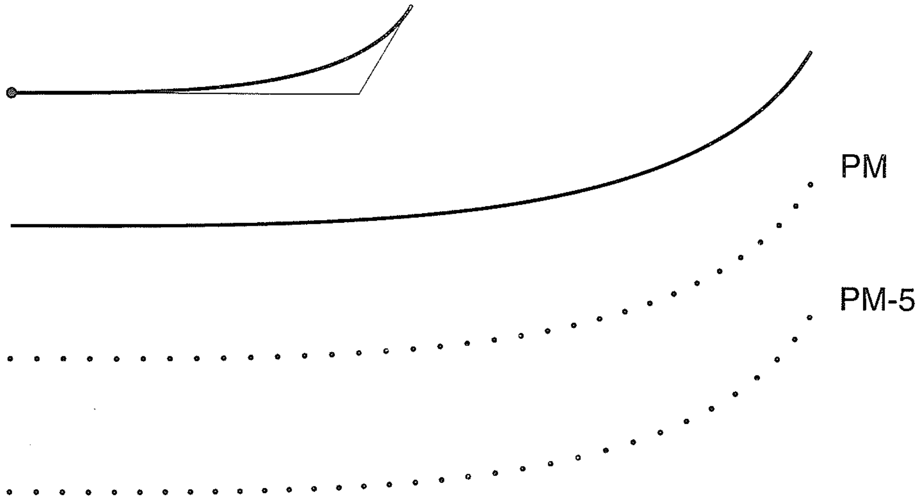

Figure 4.22: A cubic Bézier spiral curve, on which both PM and PM-5 produce spacing that is visually uniform.

The above may be explained as follows. The function $t(s)$ to be approximated is a monotone increasing function, and any function that attempts to approximate it

[2] In [44], it is shown that a cubic Bézier curve possessing polygon legs such that (a) $L_1 = L_2, L_i = ||\mathbf{p}_i - \mathbf{p}_{i-1}||$, (b) $\mathbf{p}_0, \mathbf{p}_1$, and $\mathbf{p}_2$ are collinear, and (c) $L_3 = \frac{6}{5}L_1 \cos\theta$, is a spiral, with θ being the angle between the lines $\mathbf{p}_1\mathbf{p}_2$ and $\mathbf{p}_2\mathbf{p}_3$

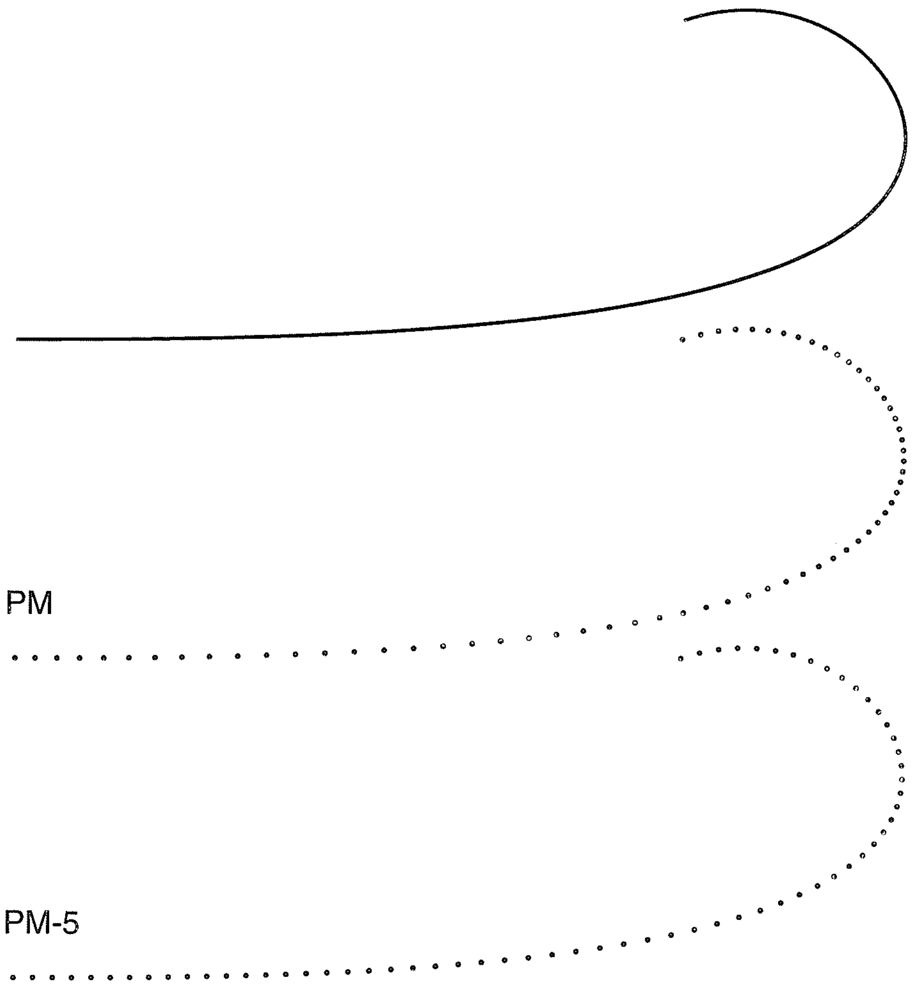

Figure 4.23: The quartic curve from Figure 4.11 (also in [18]). An illustration of when PM-5 performs better than PM.

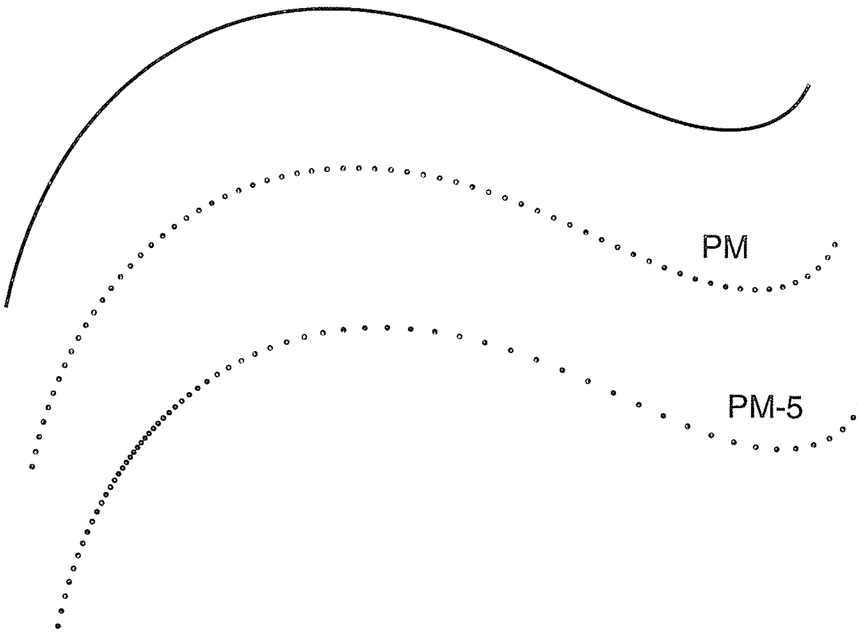

Figure 4.24: The quintic PH curve from Figures 4.4 & 4.16 . An illustration of when a higher-order interpolation polynomial does not perform well.

should be monotone and increasing. Because the tangent vectors at the end points may not always serve as reliable source of information about the rest of the curve (such as when an end polygon leg is significantly small, in proportion to the rest of the legs), the quality of approximation may accordingly degrade.

Any approximating function $A(s)$ that attempts to approximate $t(s)$, regardless of its degree, may have one or more inflection point. An inflection point along the approximating function implies that the rate of change with which the parameter t varies with respect to the arclength s, has changed from faster to slower, or vice versa. It is therefore desirable that when $t(s)$ has one or more inflection points, the approximating function $A(s)$ should also have inflection points. This is illustrated in Figure 4.25. A problem arises, however, when $A(s)$ has one or more critical values in the range $[0, \mathcal{L}]$, i.e., when $A(s)$ attains a maximum/minimum value in $[0, \mathcal{L}]$ such that $A'(s) = 0$. When this happens, $A(s)$ is not a monotone increasing function, and hence, it is not suitable to approximate $t(s)$. In practice, anomalies in the approximation were mainly due to the presence of a significantly short end-polygon leg, in proportion to the rest of the polygon legs. Figures 4.25, 4.26, and 4.27 illustrate the above observations in a "step-by-step" fashion, in which the arclength $\mathcal{L}$ has been normalized to unity. It is noted, however, that typical design curves do not usually have polygon-leg lengths that are out of proportion.

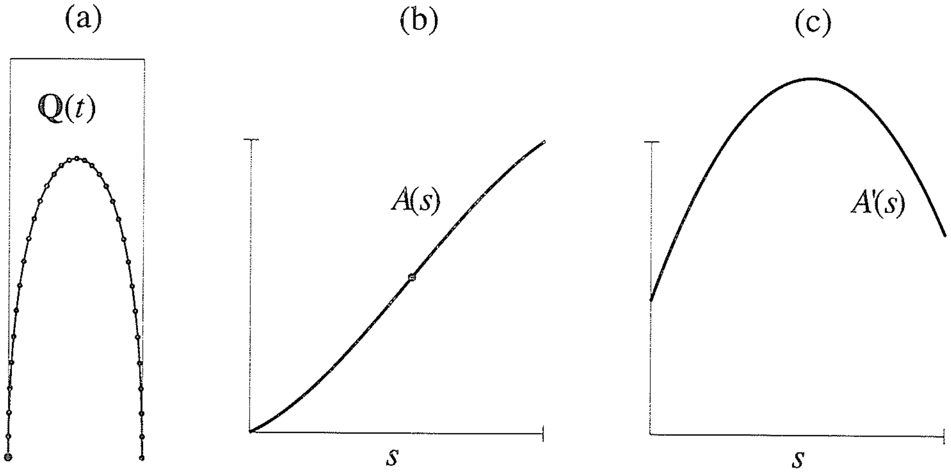

Figure 4.25: (a) The curve whose $t(s)$ is to be approximated. The dots compose a set of reference points generated on $\mathbf{Q}(t)$. (b) The approximating function $A(s)$. The dot on the curve denotes an inflection point. Before the inflection point, t progresses with a faster rate than s, the opposite is true after the IP. (c) $A'(s)$ does not have zeros in the range $[0, \mathcal{L}]$, since $A(s)$ is monotone and increasing.

From Figure 4.27, it is observed that when an approximating function $A(s)$ is such that $A'(s) = 0, s \in [0, \mathcal{L}]$, then $A(s)$ is not reliable.

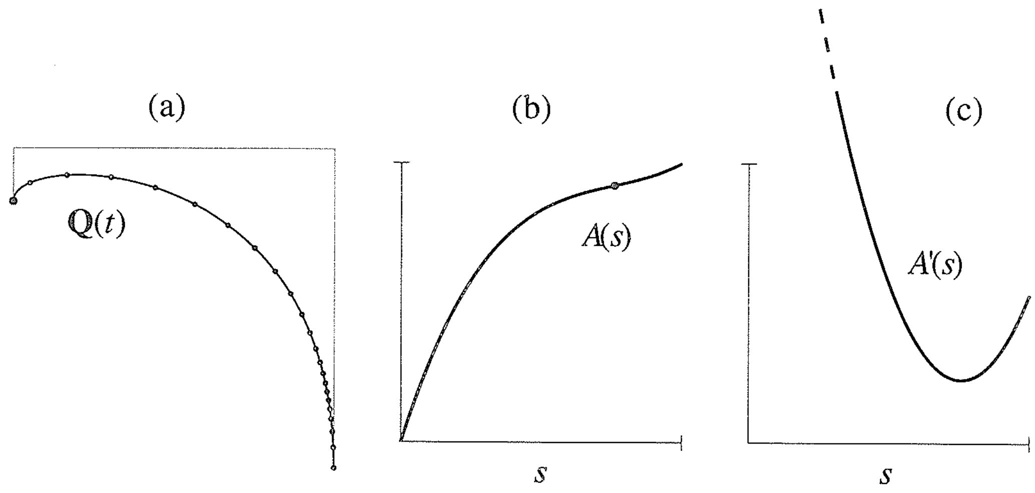

Figure 4.26: (a) $\mathbf{Q}(t)$ is a curve whose end-polygon leg is relatively small. As a result, $A(s)$ in (b) appears to be almost non-monotone and increasing. Accordingly, $A'(s)$ in (c) almost has roots in the range $[0, \mathcal{L}]$, a clear indication of non-reliability.

Furthermore, whereas the derivative of a cubic approximating function such as $f(s)$ from Section 3.3.1 may have at most two real roots in $[0, \mathcal{L}]$, derivatives of higher-order approximating functions such as $h(s)$ from Section 3.3.2 may have more than two roots in $[0, \mathcal{L}]$, and are therefore more unlikely to be reliable. This may be seen from approximating $t(s)$ by $h(s)$ for the quintic of Figure 4.24, one of whose end polygon legs is relatively short (shown in Figure 4.4, upper right).

A quadratic approximating function is less likely to have roots in the critical range $[0, \mathcal{L}]$. However, it does not allow for inflection points and may therefore prove to be a rigid, inflexible approximator for the purposes of this research.

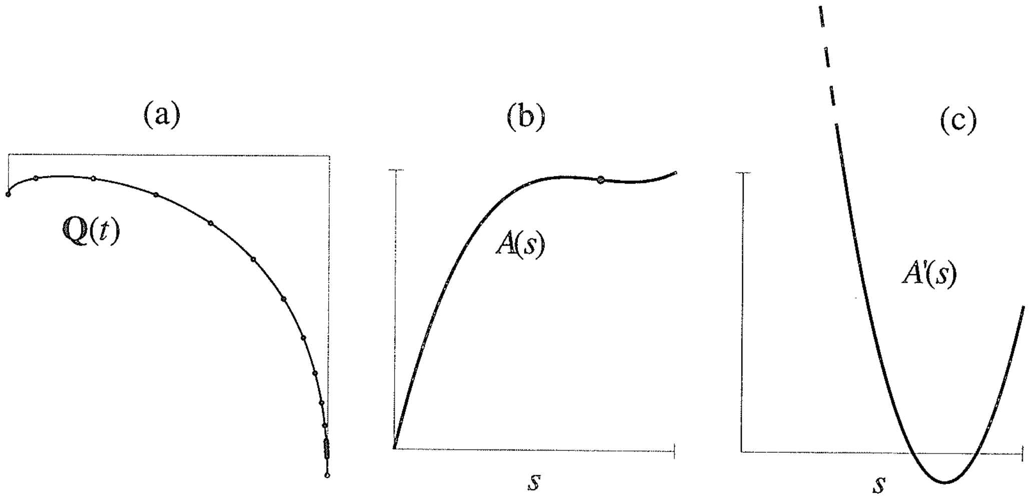

Figure 4.27: (a) The initial polygon leg is taken to be small enough to cause unacceptable results. It is shown that 55% of the RPs are clustered in a small region of low curvature! (b) $A(s)$ has an inflection point. A maximum and a minimum value occur before and after the inflection point, respectively. The presence violates the monotony required for such approximating functions. Accordingly, $A'(s)$ is shown to have zeros in the range $[0, \mathcal{L}]$ where critical values occur.

Chapter 5

Comparative Discussion on Reference-Point Generators

In this chapter, the cost of each reference-point generator (RPG) when applied to an *nth* degree Bézier curve is discussed. A generalization of the the proposed method (PM) discussed in the previous chapter is introduced. Although it departs from the basic objective of finding an approximation of the function $t(s)$ as a single entity rather than a spline, it is of interest because it is also a generalization of the cumulative chordlength method.

Chapter organization is as follows. Section 5.1 discusses the preliminaries behind developing mechanisms to compare costs of operations, and then compares the expected running times of some of the algorithms, based on the count of operations executed by each algorithm. The generalization of the proposed method is discussed in Section 5.2.

5.1 Operation Costs of Reference-Point Generators

One way to compare the performance of the reference-point-generator algorithms discussed thus far is to compare the cumulative cost of operations carried out by each algorithm.

In this section, the process of evaluating the cost of each algorithm is introduced. In considering the cost of these algorithms, the process of generating an RP, as opposed to the setup time required by them, is emphasized. For the following subsections, the following rules are assumed for convenience:

(a) Curves and vector points are assumed to be in $\mathbb{E}^2$.

(b) In deriving cost expressions, the following notation is used:

$$C(\Psi, \beta) = (\# \text{ of multiplications}, \# \text{ of function calls}),$$

where $C(\Psi, \beta)$ reads: *the cost of evaluating β times the expression Ψ.* Divisions required in the evaluation of Ψ are counted as multiplications; the number of function calls is a count of calls made to functions such as the square-root function, logarithmic functions, etc. For example, if

$$\Psi(x) = ax^3 + b\sqrt{\sin(c)} + d, \quad x = 5, 6, 7,$$

then

$$C(\Psi, \beta) = C(ax^3 + b\sqrt{\sin(c)} + d, 3) = (10, 2),$$

since for each x, the first term requires three multiplications (giving a total of nine multiplications), whereas the second term needs to be computed only once (a multiplication and two function calls are needed), thereby resulting in a total cost of 10 multiplications and two function calls.

Note that operations of additions and subtractions are not considered.

The calculations assume a reasonable implementation and are not necessarily optimal.

5.1.1 Cost of Common Operations

In this subsection, the cost of evaluating expressions that are common to most of the algorithms of interest in this thesis is carried out. These operations usually involve the evaluation of quantities such as $\mathbf{Q}(t), \mathbf{Q}'(t), s(t)$.

Cost of $\mathbf{Q}(t)$

Consider the following expansion of $\mathbf{Q}(t)$ of degree n, defined in (2.1):

$$\begin{aligned}\mathbf{Q}(t) &= \mathbf{p}_0(1-t)^n + \mathbf{p}_1 n(1-t)^{n-1}t + \mathbf{p}_2\begin{pmatrix} n \\ 2 \end{pmatrix}(1-t)^{n-2}t^2 + \\ &\quad \cdots + \mathbf{p}_{\lfloor n/2 \rfloor}\begin{pmatrix} n \\ \lfloor n/2 \rfloor \end{pmatrix}(1-t)^{n-\lfloor n/2 \rfloor}t^{\lfloor n/2 \rfloor} + \cdots + \\ &\quad \mathbf{p}_{n-2}\begin{pmatrix} n \\ n-2 \end{pmatrix}(1-t)^2 t^{n-2} + \mathbf{p}_{n-1}n(1-t)t^{n-1} + \mathbf{p}_n t^n.\end{aligned} \tag{5.1}$$

To evaluate the cost of (5.1) at $t = t_k$, the following points are noted:

(a) There is a total of $n+1$ terms, denoted $T_0, T_1, \cdots, T_n$.

(b) To evaluate $(1-t_k)^{i+1}$ from $(1-t_k)^i$ (or, t_k^{i+1} from t_k^i), only one multiplication is required. Thus, to compute all the functions $(1-t_k)^2, \cdots, (1-t_k)^n$, and $t_k^2, \cdots, t_k^n$,

$$2(n-1) \tag{5.2}$$

multiplications are required.

(c) There are

$$(n-1) \tag{5.3}$$

products of the form $(1-t_k)^i t_k^{n-i}$, $i \neq 0, n$.

(d) Because of the symmetry of binomial coefficients, only coefficients from T_2 to $T_{\lfloor n/2 \rfloor}, n \geq 4$, need to be computed. Binomial coefficients for T_0, T_1, T_{n-1}, and T_n have the values unity, n, n, and unity, respectively. Thus, if the following recursive definition

$$\left(\begin{array}{c} n \\ i \end{array}\right) = \frac{n-i+1}{i} \left(\begin{array}{c} n \\ i-1 \end{array}\right), \quad 2 \leq i \leq \lfloor n/2 \rfloor$$

for computing the binomial coefficients is used, only a total of

$$2(\lfloor n/2 \rfloor - 1) \tag{5.4}$$

multiplications are required.

(e) A total of

$$2(n-1) \tag{5.5}$$

further multiplications is required to obtain the values

$$\begin{pmatrix} n \\ i \end{pmatrix} \mathbf{p}_i, \quad i = 1, 2, \ldots, n-1.$$

(f) Finally, an additional 2 multiplications are required per T_i, to multiply the above expression in (e) by the rest of the term T_i, giving a total of

$$2(n+1) \tag{5.6}$$

multiplications.

Thus, the total cost of $\mathbf{Q}(t_k)$ may be obtained by summing the expressions from (5.2) to (5.6):

$$\begin{aligned} C(\mathbf{Q}(t_k), 1) &= \left(2(n-1) + (n-1) + 2(\left\lfloor \frac{n}{2} \right\rfloor - 1) + 2(n-1) + 2(n+1),\ 0\right) \\ &= \left(7n + 2\left\lfloor \frac{n}{2} \right\rfloor - 5, 0\right), \quad n \geq 2. \end{aligned} \tag{5.7}$$

Note that the second parameter in the above expression is zero, since no function calls are involved.

The cost of computing N (for clarity, N is used to represent the number of RPs instead of $N+1$) points along a Bézier curve $\mathbf{Q}(t)$ of degree n is derived assuming that after the first computation, the binomial coefficients (which are also multiplied by the Bézier control points) are not re-computed (i.e., steps (d) & (e) are omitted from the second evaluation onwards). Thus, for $n \geq 2, N \geq 1$, the cost of $\mathbf{Q}(t)$ is:

$$\begin{aligned} C(\mathbf{Q}(t), N) &= \left(N\left\{3(n-1) + 2(n+1)\right\} + 2\left\{\left\lfloor \frac{n}{2} \right\rfloor - 1\right\} + 2(n-1), 0\right), \\ &= \left(N(5n-1) + 2\left\{n + \left\lfloor \frac{n}{2} \right\rfloor - 2\right\}, 0\right). \end{aligned} \tag{5.8}$$

Cost of $\mathbf{Q}'(t)$

The derivative of a Bézier curve is another Bézier curve [15]. Thus, $\mathbf{Q}'(t)$ may be expressed as:

$$\mathbf{Q}'(t) = n\left\{\sum_{i=0}^{n-1}\binom{n-1}{i}(\mathbf{p}_{i+1}-\mathbf{p}_i)(1-t)^{n-1-i}t^i\right\}. \tag{5.9}$$

An expression cost may be easily obtained for $\mathbf{Q}'(t)$ by replacing n by $n-1$ in (5.7), and then adding two multiplications for the external n, that is:

$$\begin{aligned} C(\mathbf{Q}'(t),1) &= \left(7(n-1)+2\left\lfloor\frac{n-1}{2}\right\rfloor-3,0\right), \\ &= \left(7n+2\left\lfloor\frac{n-1}{2}\right\rfloor-10,0\right), \quad \text{for } n \geq 3. \end{aligned} \tag{5.10}$$

In addition, the cost of computing $\mathbf{Q}'(t)$ N times is derived by replacing n by $n-1$ in (5.8), and then adding 2 multiplications each time. This results in

$$\begin{aligned} C(\mathbf{Q}'(t),N) &= \left(N(5(n-1)-1)+2\left\{(n-1)+\left\lfloor\frac{n-1}{2}\right\rfloor-2\right\}+2N,0\right), \\ &= \left((5N+2)n+2\left\lfloor\frac{n-1}{2}\right\rfloor-4N-6,0\right). \end{aligned} \tag{5.11}$$

Cost of Computing the arclength $s(t)$

Consider the following representation of Simpson's rule for obtaining the arclength (see Section 3.1.6):

$$s(t) \approx \frac{t-t_0}{3I}\left[\alpha_0+4\sum_{i=0}^{(I-2)/2}\alpha_{2i+1}+2\sum_{i=1}^{(I-2)/2}\alpha_{2i}+\alpha_I\right]. \tag{5.12}$$

In the above expression, for any value of $I \geq 4$, I being an even number of intervals, there are four more multiplications to compute $s(t)$. There is a total of $I+1$

functions of the form $\alpha_i = \sqrt{\mathbf{Q}'^2_x(t_i) + \mathbf{Q}'^2_y(t_i)}$, their cost may be expressed as follows:

$$\begin{aligned} C(\alpha_i, I+1) &= C(\mathbf{Q}'(t), I+1) + (I+1)(2,1), \\ &= \left((5I+7)n + 2\left\lfloor \frac{n-1}{2} \right\rfloor - 2(I+2), I+1\right). \end{aligned} \tag{5.13}$$

The total cost of $s(t)$ is therefore

$$\begin{aligned} C(s(t), 1) &= C(\alpha, I+1) + (4,0), \\ &= \left((5I+7)n + 2\left\lfloor \frac{n-1}{2} \right\rfloor - 2I, I+1\right). \end{aligned} \tag{5.14}$$

Attention is now turned to the cost evaluation of the algorithms discussed in the previous chapter in subsections 4.2.1–4.2.5. Because of the non-generality of the PHC algorithm (see Section 4.2.6), it is not considered in the following discussion.

5.1.2 Cost of BPF

The BPF algorithm was introduced in Section 4.2.1. Its setup time is negligible.

When generating reference points, two operations are involved as indicated by the algorithm portion shown below (extracted from the algorithm shown in Section 4.2.1):

```
RPG: BPF()
   ⋮
       $t_i \leftarrow t_{i-1} + \Delta t$
       $\mathbf{r}_i \leftarrow \mathbf{Q}(t_i)$
END
```

The first operation to obtain t_i is of little significance to the algorithm cost. The second is simply an evaluation of $\mathbf{Q}(t)$ N times, whose cost has been determined by (5.8). Thus, the cost of generating N RPs using the BPF algorithm is

$$C(\mathsf{BPF}_{gen.}, 1) = C(\mathbf{Q}(t), N). \tag{5.15}$$

5.1.3 Cost of PM

The PM algorithm requires little setup time. Consider the following algorithm setup portion extracted from the algorithm presented in Section 4.2.2:

```
RPG: PM()
    compute $\mathcal{L}$, scale $||\mathbf{Q}'(0)||$ and $||\mathbf{Q}'(1)||$
    $c \leftarrow 1/||\mathbf{Q}'(0)||$
    $a \leftarrow c + 1/||\mathbf{Q}'(1)|| - 2$
    $b \leftarrow 1 - c - a$
    ⋮
END
```

Cost of computing $\mathcal{L}$ is given by (5.14). Scaling the magnitude at the end points (i.e., dividing $||\mathbf{Q}'(0)||$ and $||\mathbf{Q}'(1)||$ by $\mathcal{L}$) requires two multiplications (note that $||\mathbf{Q}'(0)||$ and $||\mathbf{Q}'(1)||$ have already been computed when calculating $\mathcal{L}$). Computing the coefficients of $f(s)$ in (3.27) requires two additional multiplications. This results in a total setup time of

$$C(\mathsf{PM}_{setup}, 1) = (4, 0) + C(s(t), 1). \tag{5.16}$$

To evaluate the cost of generating N RPs using the PM algorithm, consider the following extraction from Section 4.2.2:

```
RPG: PM()
  ⋮
      $f_i \leftarrow ((a\ell_i + b)\ell_i + c)\ell_i$        {Eq. (3.27) using Horner's method}
      $\mathbf{r}_i \leftarrow \mathbf{Q}(f_i)$
END
```

shows that no more than three multiplications added to the cost of $\mathbf{Q}(t)$ (given in (5.8)) are required per RP, giving a total cost of

$$\begin{aligned} C(\mathsf{PM}_{gen.}, 1) &= C(\mathbf{Q}(t), N) + 3(N, 0), \\ &= \left(N(5n+2) + 2\left\{n + \left\lfloor\frac{n}{2}\right\rfloor - 2\right\}, 0\right). \end{aligned} \tag{5.17}$$

5.1.4 Cost of F-OP

The F-OP algorithm was introduced in Section 4.2.3. Consider the following algorithm setup portion (extracted from Section 4.2.3, with steps labelled for later reference):

```
RPG: F-OP()
  compute $\mathcal{L}$, scale $\mathbf{p}_{i \leftarrow 0,1,\ldots,n}$                    (s1)
  for $i \leftarrow 0$ to $n-1$                                      (s2)
      $\Delta\mathbf{p}_i \leftarrow \mathbf{p}_{i+1} - \mathbf{p}_i$
  for $i \leftarrow 0$ to $n$                                        (s3)
      $\mathbf{c}_i \leftarrow i(n-i+1)\Delta\mathbf{p}_{i-1} - (i+1)(n-i)\Delta\mathbf{p}_i$
  for $i \leftarrow 0$ to $2n-1$                                     (s4)
      $b_i \leftarrow \left\{\sum_{j=\max(0,i-n+1)}^{\min(n,i)} \binom{n}{j}\binom{n-1}{i-j}(\mathbf{c}_j \cdot \Delta\mathbf{p}_{i-j})\right\} \Big/ \binom{2n-1}{i}$
  for $i \leftarrow 4$ to $2n+3$                                     (s5)
```

```
        $r \leftarrow 2n + 3 - i$
        $C_0^r \leftarrow \sum_{k=0}^{2n-1-r} (-1)^k \binom{2n-1-r}{k} b_{k+r}$
        $C_r^r \leftarrow \sum_{k=0}^{2n-1-r} (-1)^{r+k} \binom{2n-1-r}{k} b_k$
    $q_0 \leftarrow \sum_{i=4}^{2n+3} \frac{1}{i-1} \binom{2n-1}{i-4} C_0^{2n+3-i}$                    (s6)
    $q_{2n+2} \leftarrow -\sum_{i=4}^{2n+3} \frac{1}{i-1} \binom{2n-1}{i-4} C_{2n+3-i}^{2n+3-i}$        (s7)
    $\alpha \leftarrow \left(1 + \sqrt{-q_{2n+2}/q_0}\right)^{-1}$                                      (s8)
    ⋮
END
```

The cost expression of each of the above steps follows.

(s1) In addition to computing the arclength, there are $2(n+1)$ divisions required to scale $\mathbf{p}_{i=0,1,\ldots,n}$, resulting in a cost expression $C(s(t),n) + (2(n+1),0)$.

(s2) $(0,0)$.

(s3) The for loop is executed $n+1$ times. Each time six multiplications are carried out. This results in the cost expression $(6(n+1),0)$.

(s4) There are $2n$ b_i's to be computed. Each b_i has $(n+1)/2$ terms on average. Assuming that two multiplications are required to evaluate each binomial coefficient, two for the dot product, the summation term requires eight multiplications, and three multiplications for the denominator, giving a total $8(2n)(n+1)/2 + 3(2n)$ multiplications, the cost expression is therefore $(2n(4n+7),0)$.

(s5) The for loop is executed $2n$ times. Each of C_0^r and C_r^r has $(2n+1)/2$ terms on average. Assuming that two multiplications are required to evaluate the binomial coefficient, plus another two to multiply b_{k+r}, b_k, there are $4(2n)(2n+1)/2$ multiplications. Hence, the cost expression is $(4n(2n+1), 0)$.

(s6) There are $2n$ terms, each requiring 4 multiplications, resulting in cost of $(8n, 0)$.

(s7) There are $2n$ terms; using the results of (s6), only one multiplication is necessary for each term. The cost expression is therefore $(2n, 0)$.

(s8) One division, one reciprocal, and one function call are required. The cost is therefore $(2, 1)$.

The total setup time for the F-OP algorithm may now be determined as follows:

$$\begin{aligned} C(\text{F-OP}_{setup}, 1) &= C((\text{s1}), 1) + C((\text{s2}), 1) + \cdots + C((\text{s8}), 1), \\ &= (16n^2 + 36n + 10, 1) + C(s(t), 1). \end{aligned} \tag{5.18}$$

Turning the attention to the cost of generating an RP using this algorithm, consider the following portion from the same algorithm:

```
RPG: F-OP()
  ⋮
      $t_i \leftarrow t_{i-1} + \Delta t$
      $\mathbf{r}_i \leftarrow \tilde{\mathbf{Q}}(t_i)$      $\{$using Eq. (4.2)$\}$
END
```

Re-writing (4.2) in the following manner:

$$\tilde{\mathbf{Q}}(t_i) = \frac{\sum_{k=0}^{n} \mathbf{p}_k \begin{pmatrix} n \\ k \end{pmatrix} \{(1-\alpha_i)t_i\}^k \{\alpha_i(1-t_i)\}^{n-k}}{\sum_{k=0}^{n} \begin{pmatrix} n \\ k \end{pmatrix} \{(1-\alpha_i)t_i\}^k \{\alpha_i(1-t_i)\}^{n-k}}, \tag{5.19}$$

results in the following cost expression:

$$\begin{aligned} C(\mathsf{F\text{-}OP}_{gen.}, 1) &= C(\tilde{\mathbf{Q}}(t), N) = C(\mathbf{Q}(t), N) + (3N, 0) \\ &= \left(N(5n+2) + 2\left\{n + \left\lfloor \frac{n}{2} \right\rfloor - 2\right\}, 0\right), \end{aligned} \tag{5.20}$$

where the three extra multiplications per RP are the result of: (a) multiplying $(1-\alpha_i)$ by t_i, (b) multiplying α_i by $(1-t_i)$, and (c) the division. This is similar to the cost of generating an RP using the PM algorithm.

5.1.5 Cost of ST

The ST algorithm requires the total arclength $\mathcal{L}$ to be computed in the setup, whose cost expression is given by (5.14).

To help evaluate the cost of generating an RP using this algorithm, the following algorithm portion is extracted from Section 4.2.4.

```
RPG: ST()
  ⋮
      repeat
          $t_i \leftarrow t_i - \frac{M(t_i)}{M'(t_i)}$
      until convergence to a given tolerance
      $\mathbf{r}_i \leftarrow \mathbf{Q}(t_i)$
END
```

where $M(t)$ and $M'(t)$ are given in (4.3) and (4.4), respectively. The cost of $M(t)$ is similar to that of $s(t)$ given in (5.14). The cost of $M'(t)$ is similar to that of computing one function α_i, however, its value is obtained during the arclength calculation of the $i/2th$ segment (it is the last function usually computed in the calculation of $M(t)$), and need not be re-computed. These observations result in the following initial cost expression for t_i in the repeat ··· until loop:

$$\begin{aligned} C(t_i, \gamma) &= \gamma\{C(s(t), 1) + (1, 0)\}, \\ &= C(s(t), \gamma) + (\gamma, 0). \end{aligned} \tag{5.21}$$

where γ denotes the number of iterations required to reach the desired tolerance. Once a satisfactory t_i is obtained, (2.1) is evaluated to generate the required RP. Thus, the total cost of this algorithm is:

$$\begin{aligned} C(\mathsf{ST}, 1) &= N\{C(t_i, \gamma) + C(\mathbf{Q}(t), 1)\}, \\ &= C(t_i, \gamma N) + C(\mathbf{Q}(t), N). \end{aligned} \tag{5.22}$$

5.1.6 Cost of CC

Consider the following algorithm portion extracted from Section 4.2.5:

```
RPG: CC()
  ⋮
  for $i \leftarrow 1$ to $\Im$
      $u_i \leftarrow u_{i-1} + \Delta u$
      $\mathbf{v}_i \leftarrow \mathbf{Q}(u_i)$
      $s_i \leftarrow s_{i-1} + ||\mathbf{v}_i - \mathbf{v}_{i-1}||$
  ⋮
END
```

The cost of obtaining $v_{i=0,1,\ldots,\Im}$, $\Im$ being the number of chords used to approximate the curve, is

$$C(\mathrm{v}_i, \Im) \;=\; C(\mathbf{Q}(t), \Im). \tag{5.23}$$

The cost of computing the cumulative arclengths $s_{i=0,1,\ldots,\Im}$ is two multiplications and a function call per s_i, giving

$$C(s_i, \Im) = \Im(2,1) = (2\Im, \Im). \tag{5.24}$$

Thus, the setup time of the CC algorithm is

$$\begin{aligned} C(\mathsf{CC}_{setup}, 1) &= C(\mathbf{v}, \Im) + C(s, \Im), \\ &= \left(\Im(5n+1) + 2\left\{n + \left\lfloor \frac{n}{2} \right\rfloor - 2\right\}, \Im\right). \end{aligned} \tag{5.25}$$

To obtain an expression for the cost of generating an RP, consider the following portion from the algorithm presented in Section 4.2.5:

```
RPG: CC()
  ⋮
      while $i\Delta d \geq s_k$
          $k \leftarrow k+1$
      $t_i \leftarrow A(i,k)$        {using Eq. (4.6)}
      $r_i = \mathbf{Q}(t_i)$
END
```

The cost of obtaining t_i is two multiplications per RP, at which $\mathbf{Q}(t)$ is evaluated once. The total cost to generate N RPs may therefore be estimated by the following expression:

$$C(\mathsf{CC}_{gen.}, 1) \;=\; (2N, 0) + C(\mathbf{Q}(t), N). \tag{5.26}$$

5.1.7 Tables of Costs

In this subsection, the costs of the different RPG algorithms are presented in a tabular format to facilitate comparison. There are five tables, the first one (Table 5.1) lists the cost expressions of all the algorithms, for both setup time, and RP generation time. The second and third tables consider the setup-time cost, whereas the fourth and fifth tables consider the cost of generating RPs.

Because of the wide variation in the techniques used by the various methods, a straightforward comparison is difficult. The following conventions are introduced to obtain meaningful comparisons.

Table 5.1: List of cost expressions.

$C(\mathbf{Q}(t), N)$	$\left(N(5n-1)+2\left\{n+\left\lfloor\frac{n}{2}\right\rfloor-2\right\},0\right)$
$C(s(t), I)$	$\left((5I+7)n+2\left\lfloor\frac{n-1}{2}\right\rfloor-2I, I+1\right)$

$C(\text{Alg.}, 1)$	Setup	RP Gen.
BPF	–	$C(\mathbf{Q}(t), N)$
PM	$C(s(t), 1)+(4, 0)$	$C(\mathbf{Q}(t), N)+(3N, 0)$
F-OP	$C(s(t), 1)+(16n^2+36n+10, 1)$	$C(\mathbf{Q}(t), N)+(3N, 0)$
ST	$C(s(t), 1)$	$C(s(t), \gamma N)+(\gamma N, 0)+$ $C(\mathbf{Q}(t), N)$
CC	$C(\mathbf{Q}(t), \Im)+(2\Im, \Im)$	$C(\mathbf{Q}(t), N)+(2N, 0)$

(a) The degree of the curve, n, assumes values of three, five, and nine.

(b) N, the number of RPs to be generated, is fixed at 100.

(c) The number of chords that are used in the CC algorithm, $\Im$, (only applicable to the CC algorithm), is set to 48 (note that this may be an underestimate for the number of chords required in practice).

(d) The number of Simpson intervals, I, required to compute arclengths is set as follows; to compute $\mathcal{L}$, the total arclength of $\mathbf{Q}(t)$, I is set to eight; to compute the arclength of a segment of $\mathbf{Q}(t)$ (as required by ST, for example), I is set to four.

(e) Algorithms with entries containing zero in every column are omitted from tables.

(f) Entries are in the form $m_1 + m_2 + \cdots = \mathcal{S}$, where each term corresponds to the equivalent term in the cost expressions listed in Table 5.1, and $\mathcal{S}$ is their sum.

The following pair of tables show setup operation count. Table 5.2 shows the number of multiplications executed by the different RPG algorithms. Table 5.3 shows the number of function calls executed by each RPG algorithm.

Table 5.2: Multiplication count during setup.

	$n = 3$	$n = 5$	$n = 9$
PM	$4 + 127 = 131$	$4 + 223 = 227$	$4 + 415 = 419$
F-OP	$262 + 127 = 389$	$590 + 223 = 813$	$1630 + 415 = 2045$
ST	127	223	415
CC	$676 + 96 = 772$	$1162 + 96 = 1258$	$2134 + 96 = 2230$

Table 5.3: Function-call count during setup.

	$n = 3$	$n = 5$	$n = 9$
PM	$9 + 0 = 9$	$9 + 0 = 9$	$9 + 0 = 9$
F-OP	$9 + 1 = 10$	$9 + 1 = 10$	$9 + 1 = 10$
CC	9	9	9
CC	$0 + 48$	$0 + 48$	$0 + 48$

The attention is now turned to the cost of generating RPs. Table 5.4 shows the number of multiplications executed per algorithm. Table 5.5 shows the number of function calls executed per algorithm.

Table 5.4: Multiplication count during RP generations.

	$n = 3$	$n = 5$	$n = 9$
BPF	1404	2410	4422
PM	$1404 + 300 = 1704$	$2410 + 300 = 2710$	$4422 + 300 = 4722$
F-OP	$1404 + 300 = 1704$	$2410 + 300 = 2710$	$4422 + 300 = 4722$
ST	$30000 + 400 + 1404 = 31804$	$52400 + 400 + 2410 = 55210$	$97200 + 400 + 4422 = 102022$
CC	$1404 + 200 = 1604$	$2410 + 200 = 2610$	$4422 + 200 = 4622$

Table 5.5: Function-call count during RP generation.

	$n = 3$	$n = 5$	$n = 9$
ST	2000	2000	2000

From the above tables, the following is concluded:

(a) The PM algorithm has a small setup time when compared to the others. In cases where the arclength has been pre-computed, the setup time is negligible, whereas the F-OP and the CC have relatively larger setup times, depending on the value of n, and $\Im$, respectively, as indicated in Table 5.2.

(b) From Tables 5.4 and 5.5, it is evident that the ST algorithm is the most computationally intensive algorithm to be used for locating points, both for the number of multiplications, and the number of function calls that it makes. By comparison, the other algorithms have low and similar operation counts, especially as n increases.

5.2 Using Cubic-Spline Interpolation Function

In developing the proposed method, the emphasis was on obtaining an expression for $t(s)$ (that is, t, the parameter of the curve $\mathbf{Q}(t)$, as a function of s, the arclength of the curve) in an analytical fashion, such that, when evaluated at $s_k \in [0, \mathcal{L}]$, the corresponding $t_k \in [0, 1]$ results. It is desirable, for manipulation in mathematical experiments, that it be a single function rather than a piecewise function.

The objective of the proposed method is to make available an approximation to $t(s)$ (see Chapter 1). This is similar to some of the objectives sought in [17][18][26][40]. However, it is noted that some of these methods depend on analytical expressions, such as the PM or the F-OP algorithms, whereas other methods, such as the ST algorithm, depend on numerical methods to approximate $t(s)$. The first kind of approximators have the advantage of not having to compute and maintain arrays of numbers, or use quadrature techniques to reach a satisfactory result; it is not equitable to compare their accuracy to those that depend on numerical methods. The latter have the advantage that any prescribed accuracy can be obtained by increasing the number of approximating segments to refine results in accordance with some prescribed tolerance, or increasing the number of Newton-Raphson iterations.

In this section, it is shown how the proposed method may be modified so that it also uses numerical techniques to generate reference points (RPs) along parametric curves, thereby making use of the advantages that numerical methods have. The modified version of the proposed method is called MPM, or, the modified PM. Its objective is to generate RPs with accuracy and performance comparable to methods which depend on numerical techniques.

5.2.1 The MPM Algorithm

The idea behind developing the MPM is similar to that behind the CC method: the curve $\mathbf{Q}(t)$ is first approximated by $\Im$ segments. The cumulative arclength is calculated at the end of the kth segment, $k = 1, 2, \ldots, \Im$, along with the coefficients a_k, b_k, c_k, and d_k. Because each segment is approximated over two Simpson intervals (using Simpson's rule), $\Im$ is always $I/2$, where I is an even number of Simpson intervals used to approximate the $\mathbf{Q}(t)$. This is feasible because Simpson's rule may give the intermediate arclengths at every second function. That is, the arclength of the kth segment may be determined by

$$s_k = \frac{1}{3I}(\alpha_{2k-2} + 4\alpha_{2k-1} + \alpha_{2k}) + s_{k-1}, \quad k = 1, 1, \ldots, \Im, \tag{5.27}$$

where $s_0 = 0$, and $\alpha_j = \|\mathbf{Q}'(j/I)\|$. Note that $\mathcal{L} = s_\Im$. To determine the coefficients for each segment, Equation (3.27) is rewritten in the following manner:

$$f_k(s) = a_k s^3 + b_k s^2 + c_k s + d_k, \quad s_{k-1} < s \leq s_k, \tag{5.28}$$

with derivative

$$f_k'(s) = 3a_k s^2 + 2b_k s + c_k. \tag{5.29}$$

For each segment, four equations are required to solve for the four unknowns. Following the same methodology used in Section 3.3.1 (i.e., to formulate four equations by equating each of $f_k(s)$ and $f_k'(s)$ at $s = s_{k-1}$, and $s = s_k$, with the appropriate values of $t(s)$, and $t'(s)$, respectively). Let

$$\begin{aligned} f_k(s_{k-1}) &= \frac{k-1}{\Im}, \tag{5.30}\\ f_k(s_k) &= \frac{k}{\Im}, \tag{5.31}\end{aligned}$$

and

$$\begin{aligned} f'_k(s_{k-1}) &= \frac{1}{\alpha_{2k-2}}, \tag{5.32}\\ f'_k(s_k) &= \frac{1}{\alpha_{2k}}, \tag{5.33}\end{aligned}$$

the four equations are formulated as follows:

$$\begin{aligned} a_k s_{k-1}^3 + b_k s_{k-1}^2 + c_k s_{k-1} + d_k &= (k-1)/\Im, \tag{5.34}\\ a_k s_k^3 + b_k s_k^2 + c_k s_k + d_k &= k/\Im, \tag{5.35}\\ 3a_k s_{k-1}^2 + 2b_k s_{k-1} + c_k &= 1/\alpha_{2k-2}, \tag{5.36}\\ 3a_k s_k^2 + 2b_k s_k + c_k &= 1/\alpha_{2k}. \tag{5.37}\end{aligned}$$

Again, note that $k = 1, 2, \ldots, \Im$, and that $\Im = I/2$. The coefficients a_k, b_k, c_k, and d_k may now be solved for, and are as follows:

$$\begin{aligned} a_k &= \frac{1}{(s_k - s_{k-1})^2}\left(\frac{\alpha_{2k-2} + \alpha_{2k}}{\alpha_{2k}\alpha_{2k-2}} - \frac{2}{\Im(s_k - s_{k-1})}\right), \tag{5.38}\\ b_k &= \frac{\alpha_{2k-2} - \alpha_{2k}}{2\alpha_{2k}\alpha_{2k-2}(s_k - s_{k-1})} - \frac{3}{2}a_k(s_k + s_{k-1}), \tag{5.39}\\ c_k &= \frac{1}{\alpha_{2k}} - 3a_k s_k^2 - 2b_k s_k, \tag{5.40}\\ d_k &= \frac{k}{\Im} - (a_k s_k^3 + b_k s_k^2 + c_k s_k). \tag{5.41}\end{aligned}$$

The complete algorithm follows.

```
RPG: MPM()
    compute $\alpha_{i\leftarrow 0,1,\ldots,I}$, $s_{k=0,1,\ldots,\Im}$
    scale $\alpha_i$, $s_k$ so that $s_\Im$ is unity
    compute $a_k, b_k, c_k$, and $d_k, k = 0, 1, \ldots, \Im$
    $\Delta L \leftarrow 1/N$, $k \leftarrow 0$, $\ell_0 \leftarrow 0$
    for $i \leftarrow 1$ to $N - 1$
        $\ell_i \leftarrow \ell_{i-1} + \Delta L$
        while $\ell_i > s_k$
            $k \leftarrow k + 1$
        $f_i \leftarrow ((a_k\ell_i + b_k)\ell_i + c_k)\ell_i + d_k$
        $\mathbf{r}_i$ <- $\mathbf{Q}(f_i)$
END
```

5.2.2 Visual Results

In this subsection, some RP plots are shown for both the MPM and the CC method, where both will be compared against RPs generated by the ST method. The purpose is to show how the MPM produces better results with fewer segments than is usually required for the CC, and does less computation than is required by the CC, or the ST method.

Figure 5.1[1] shows how the MPM produces better results with fewer segments than the CC algorithm. The difference in computational costs increases with the degree of the curve.

[1] The figures in Figure 5.1 are also shown in figures 4.9, 4.11, and 4.19

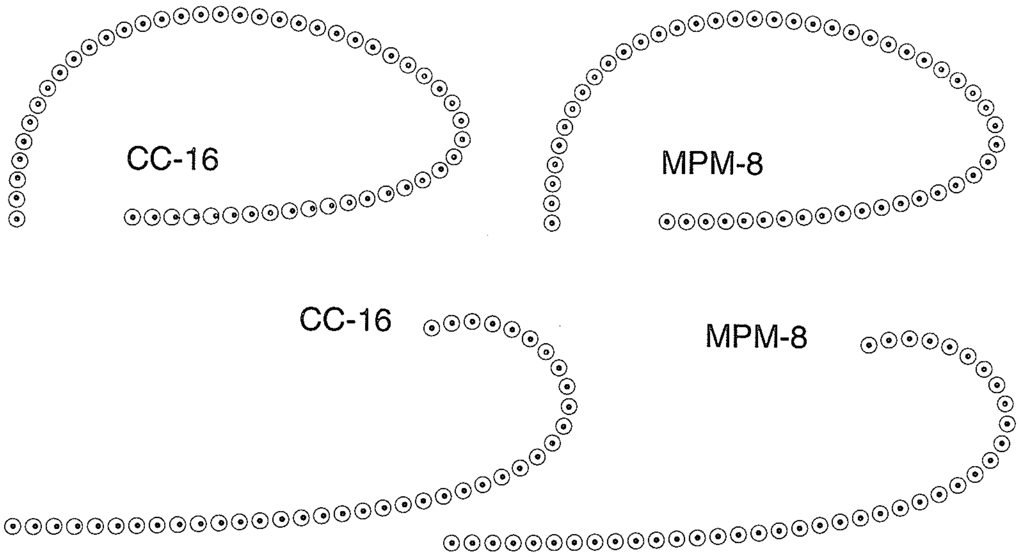

Figure 5.1: In this figure, the circles denote RPs generated by the ST algorithm, while the dots denote RPs generated by the CC and the MPM algorithms. The number following the algorithm name on each plot indicates the number of chords, segments used by the CC, and the MPM algorithms, respectively.

Chapter 6

Conclusion

A method for approximating the intrinsic arclength parameterization for parametric curves has been proposed. The main property of the proposed method is that it depends on analytical expressions, as opposed to methods which depend on numerical techniques.

Accuracy and performance of the proposed method for approximating arclength parameterization, along with that of several other methods, has been tested on the application of reference points generation. The task was to generate reference points that are uniformly spaced, along the path of parametric curves. It was shown with the aid of numerous figures, that the proposed method produces results practical enough for most design curves, with computational performance that is acceptable for real-time processing requirements.

A generalization of the proposed method to one that makes more use numerical techniques to approximate arclength parameterization, has also been developed. It was shown that the results and the performance of such a method produced accurate results, with relatively small computational requirements.

Several methods for calculating the arclength of parametric curves have been discussed. Some of them have been adapted for the work accomplished in this thesis and exploited to aid in the process of reference points generation.

In what follows, several problems that are currently under investigation are mentioned.

Problems under Investigation

In order for the proposed method (PM) to be reliably used in precision-critical applications, two areas that are of a theoretical nature should be investigated. They are as follows:

1. How many Simpson intervals are needed to achieve acceptable accuracy in obtaining the arclength of a curve segment? This is important because too few intervals may not provide sufficient accuracy, and too many intervals may by unnecessarily expensive, and may also affect accuracy.

   An expression for the error of Simpson's rule is found in books on numerical methods or analysis [7][8][12][32]. It may be written as follows:

   $$|\mathcal{E}(h)| = \frac{h^4}{180}\,|\alpha^{\mathrm{iv}}(t)|, \tag{6.1}$$

   where $h = 1/I$ is the size of the interval, I is the number of intervals (see Section 3.1.5), $\alpha(t) = \sqrt{\mathbf{Q}'(t)\cdot\mathbf{Q}'(t)}, t \in [0,1]$. Further, by prescribing a specific precision ϵ such that:

   $$|\mathcal{E}(h) \le \epsilon, \tag{6.2}$$

it might be possible to determine the appropriate size of h in (6.1). However, an expression for $\alpha^{\text{iv}}(t)$ seems to be mathematically involved, and even if it is computable, the accurate value of t is unknown [8].

2. How close is the resulting reference-point distribution to the distribution that would result from exact arclength parameterization? Precision-non-critical applications may allow a distribution that is visually satisfactory, however, it is desirable to have a numerical or visual expression of how the resulting distribution compares to that resulting from the intrinsic arclength parameterization.

   Consider, for example, the quintic PH curve from Figure 4.4 whose RP distributions from the different algorithms are shown in Figure 4.16. A visual expression for the deviation of the approximating function $f(s)$ (discussed in Section 3.3.1) from the true function $t(s)$ is depicted in Figure 6.2. It is desirable to develop mechanisms to convey more information such as:

   (a) By how much has $f(s)$ deviated from $t(s)$?

   (b) When is the error so large that $f(s)$ will not yield a satisfactory approximation?

   (c) How feasible is it to try to find a general expression for the error resulting from such deviations?

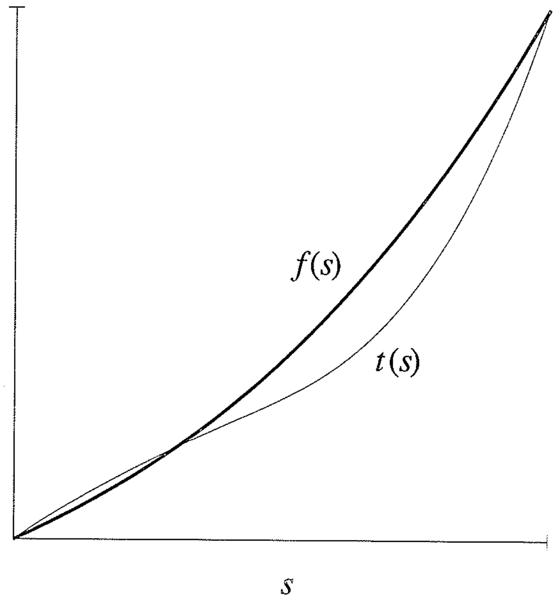

Figure 6.1: Error in approximation of $t(s)$ by $f(s)$.

Appendix A

Two-Point Taylor Interpolation

In this appendix, the two-point Taylor series [11, p. 37] is used to approximate $t(s)$ in (3.30). let s_0 and s_1 be two distinct points in $I\!R$. The polynomial $p(s) = p_{2n-1}(s)$ of degree $2n-1$ is defined as follows:

$$p(s) = (s-s_0)^n \sum_{k=0}^{n-1} \frac{B_k (s-s_1)^k}{k!} + (s-s_1)^n \sum_{k=0}^{n-1} \frac{A_k (s-s_0)^k}{k!}, \tag{A.1}$$

where

$$\begin{aligned} A_k &= \frac{\mathrm{d}^k}{\mathrm{d}s^k} \left[\frac{t(s)}{(s-s_1)^n} \right]_{s=s_0}, \\ B_k &= \frac{\mathrm{d}^k}{\mathrm{d}s^k} \left[\frac{t(s)}{(s-s_0)^n} \right]_{s=s_1}. \end{aligned} \tag{A.2}$$

Letting $n=2$, (A.1) becomes the following.

$$\begin{aligned} p(s) \quad = \quad & (s-s_0)^2 \left\{ t(s_1) + \frac{t'(s_1)}{2}(s-s_1) \right\} + \\ & (s-s_1)^2 \left\{ t(s_0) + \frac{t'(s_0)}{2}(s-s_0) \right\}, \end{aligned} \tag{A.3}$$

with derivative

$$p'(s) = (s-s_0)t'(s_1) - (s-s_1)t'(s_0). \tag{A.4}$$

Substitution in (A.4) and (A.5) for $s_0 = 0$ and $s_1 = \mathcal{L}$ shows that $p(s)$ is equivalent to $f(s)$ in (3.27), where a set of equations similar to those in (3.36–3.39) is obtained. That is:

$$p(0) = t(0),\ p(\mathcal{L}) = t(\mathcal{L}),\ \text{ and}$$

$$p'(0) = t'(0),\ p'(\mathcal{L}) = t'(\mathcal{L}).$$

See Section 3.3.1 for a complete solution.

Appendix B

Control points

Following are the data for the curves shown in Chapter 4. The data are supplied in a normalized format; the total arclength of each curve is therefore unity.

Figure 4.1

$\{[0,0]\ [.355,.643]\ [.71,0]\}$

$\{[0,.238]\ [.144,0]\ [.741,.759]\}$

Figure 4.2

$\{[0,.315]\ [.262,.839]\ [.524,0]\ [.786,.577]\}$

PH curve: $\{[0,0]\ [0,.5]\ [.5,.5]\ [.5,0]\}$

$\{[.026,0]\ [0,.474]\ [.975,.003]\ [.144,.003]\}$

$\{[0,0]\ [.763,.412]\ [0,.412]\ [.763,0]\}$

Figure 4.3

{[0, 0] [.758, 0] [.812, .101] [.703, .317] [.533, .259]}

{[.209, .313] [.209, .417] [0, .417] [0, 0] [.835, 0]}

{[.14, 0] [.839, 0] [0, .28] [1.119, .42] [.56, .56]}

{[092, .055] [0, .369] [.663, .553] [.184, 0] [.848, .221]}

Figure 4.4

{[0, 0] [0, .3] [.15, .45] [.35, .45] [.5, .3] [.5, 0]}

PH curve: {[0, 0] [.087, .429] [.52, .386] [.542, .168] [.759, .124] [.802, .228]}

{[.157, 0.143] [.571, 0] [.571, 0.714] [0, 0] [0, .714] [.414, .571]}

{[.238, .317] [.317, .317] [.317, .476] [0, .476] [0, 0] [.872, .159]}

Figure 4.22

{[.818, 3.021] [1.227, 3.021] [1.636, 3.021] [1.759, 3.232]}

Bibliography

[1] Adams, R.A., *Single-Variable Calculus*, Second edition, Addison-Wesley, 1990, 313–317.

[2] Bartels, R.H., J.C. Beatty and B.A. Barsky, *An introduction to Splines for use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann Publishers, Los Altos, California, 1987.

[3] Beach, R.C., *An Introduction to the Curves and Surfaces of Computer-Aided Design*, Van Nostrand Reinhold, New York, 1991.

[4] Bois, G.P., *Tables of Indefinite Integrals*, Dover Publications, Inc., New York, 1961.

[5] Bressoud, D.M., *Readings in Mathematics: Second Year Calculus*, Springer-Verlag, Inc, New York, 1991.

[6] Burchard, H.G., J.A. Ayers, W.H. Frey and N.S. Sapidis, *Approximating with Aesthetic Constraints*, in *Designing Fair Curves and Surfaces: Shape Quality in Geometric Modeling and Computer-Aided Design*, N.S. Sapidis, ed., SIAM, Philadelphia, 1994, 3–28.

[7] Cheney, W. and D. Kincaid, *Numerical Mathematics and Computing*, Second edition, Brooks/Cole Publishing Company, Pacific Grove, California, 1985, 45–71.

[8] Conte, S.D. and C. de Boor, *Elementary Numerical Analysis*, Second edition, McGraw-Hill Book Company, Inc., New York, 1972, 236–237.

[9] Coquillart, S., "A Control-Point-Based Sweeping Technique", *IEEE Computer Graphics and Applications*, Vol. 7, 1987, 36–45.

[10] Cox, M.G., "The Numerical Evaluation of B-splines", *J. Inst. Math. Applics.*, Vol. 10, 1972, 134–149.

[11] Davis, P.J., *Interpolation and Approximation*, Blaisdell Publishing Company, New York, 1963.

[12] Davis, P.J. and P. Rabinowitz, *Methods of Numerical Integration*, Academic Press, New York, 1975, 45–48.

[13] de Boor, C., "On Calculating with B-splines", *J. Approx. Theory*, Vol. 6, 1972, 50–62.

[14] Elbert, G. and E. Cohen, "Error Bounded Variable Distance Offset Operator for Free Form Curves and Surfaces", *Int. J. Comput. Geom. Appl.*, Singapore, Vol. 1, No 1, March 1991, 67–78.

[15] Farin, G., *Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide*, Academic Press, Boston, 1993.

[16] Farin, G., *NURB Curves and Surfaces : From Projective Geometry to Practical Use*, Wellesley, Mass., 1995.

[17] Farouki, R.T., "Pythagorean-hodograph Curves in Practical Use", in *Geometry Processing for Design and Manufacturing*, (R.E. Barnhill, ed.), SIAM, Philadelphia, 1992, 3–33.

[18] Farouki, R.T., "Optimal Parameterizations", to appear in *Computer Aided Geometric Design*, 1996.

[19] Farouki, R.T., "The Elastic Bending Energy of Pythagorean-hodograph Curves", *Computer Aided Geometric Design* (to appear), 1996.

[20] Farouki, R.T. and C.A. Neff, "Hermite Interpolation by Pythagorean-hodograph Quintics", *Mathematics of Computation*, Vol. 64, 1995, 1589–1609.

[21] Farouki, R.T. and J. Peters, "Smooth Curve Design with Double-Tschirnhausen cubics", *Annals of Numerical Mathematics* (to appear), 1996.

[22] Farouki, R.T. and T. Sakkalis, "Pythagorean Hodographs", *IBM Journal of Research and Development*, 34, 1990.

[23] Farouki, R.T. and S. Shah, "Real-time CNC interpolators for Pythagorean-hodograph curves", submitted to *Computer Aided Geometric Design*, 1995.

[24] Faux, I.D. and M.J. Pratt, *Computational Geometry for Design and Manufacture*, Ellis Horwood Limited, Chichester, England, 1981.

[25] Foley, J.D., A. Van Dam, S.K. Feiner and J.F. Hughes, *Computer Graphics: Principles and Practice*, Addison Wesley, Reading, Massachusetts, 1992.

[26] Guenter, B. and R. Parent, "Computing the Arc Length of Parametric Curves", *IEEE Computer Graphics and Applications*, Vol. 10, No 3, 1990.

[27] Guggenheimer, H.W., *Differential Geometry*, McGraw-Hill Book Company, Inc., New York, 1963, 15–17.

[28] Hearn, D. and M. Baker, *Computer Graphics*, Prentice Hall, Englewood Cliffs, New Jersey, 1994, 334–344.

[29] Hoschek, J., "Offset Curves in the Plane", *Computer-Aided Design*, Vol. 17, No 2, 1985, 77–82.

[30] Kim, M. and K. Nam, "Interpolation Solid Orientation with Circular Blending Quaternion Curves", *Computer-Aided Design*, Vol. 27, May 1995, 385–398.

[31] Lipschutz, M.M., *Theory and Problems of Differential Geometry*, Schaum's Outline Series, McGraw-Hill, New York, 1969, 51*ff.*

[32] Mathews, J.H., *Numerical Methods for Computer Science, Engineering, and Mathematics*, Prentice Hall, London, 1987, 322*ff.*

[33] Meek, D.S. and D.J. Walton, "The use of Cornu Spirals in Drawing Planar Curves of Controlled Curvature", *J. Comput. Applied Math.*, Vol. 25, 1989, 69–78.

[34] Meek, D.S. and D.J. Walton, "Offset Curves of Clothoidal Splines", *Computer-Aided Design*, Vol. 22, No 4, 1990, 199–201.

[35] Meek, D.S. and D.J. Walton, "Shape Determination of Planar Uniform Cubic B-spline Segments", *Computer-Aided Design*, Vol. 22, No 7, 1990, 434–441.

[36] Pall, G.A., *Introduction to Scientific Computing*, Appelton-Century-Crofts, New York, 1971, 316-317.

[37] Pottman, H., "Curve Design with Rational Pythagorean-hodograph curves", *Advances in Computational Mathematics 3*, 147–170.

[38] Roulier, J., "Specifying the Arclength of Bézier Curves", *Computer-Aided Geometric Design*, Vol. 10, 1993, 25–56.

[39] Schmitt, P.R., *Reactive Path Shaping: Local Path Planning for Autonomous Mobile Robots in Aisles*, Master's thesis, Dept. of Mechanical Engineering, Georgia Institute of Technology, March 1996.

[40] Sharpe, R.J. and Richard W. Thorne, "Numerical Method for Extracting an Arclength Parameterization from Parametric Curves", *Computer-Aided Design*, Vol. 14, No 2, March 1982, 79–81.

[41] Su, B.Q. and D.Y. Liu, *Computational Geometry: Curve and Surface Modeling*, Academic Press, Boston, 1989.

[42] Walton, D.J. and D.S. Meek, "Curvature Bounds for Planar B-spline Curve Segments", *Computer-Aided Design*, Vol. 20, No 3, 1988, 146–150.

[43] Walton, D.J. and D.S. Meek, "Clothoidal Splines", *Comput. & Graphics*, Vol. 14, No 1, 1990, 95–100.

[44] Walton, D.J. and D.S. Meek, "A Planar Cubic Bézier Spiral", *J. Computational & App. Math.*, (In press), 1996.

[45] Walton, D.J. and R. Xu, "Turning Point Preserving Planar Interpolation", *ACM Transaction on Graphics*, Vol. 10, No 3, July 1991, 297–311.

[46] Weil, A., *Number Theory*, Birkhäuser, Boston, 1984.

[47] Young, E.C., *Vector and Tensor Analysis*, Marcel Dekker, New York, 1993.