New tools for animation and design : a haptic-based system for stop motion camera animation and curve design by algebraic-trigonometric Pythagorean Hodograph curves

Laura Saini

Université de Valenciennes et du Hainaut-Cambrésis

Thèse de doctorat

Pour obtenir le grade de Docteur de l'Université de

VALENCIENNES ET DU HAINAUT-CAMBRESIS

Discipline, spécialité selon la liste des spécialités pour lesquelles l'Ecole Doctorale est accréditée :
Mathématiques appliquées

Présentée et soutenue par Laura, SAINI.

Le 13/06/2013, à Valenciennes

Ecole doctorale :
Sciences Pour l'Ingénieur (SPI)

Equipe de recherche, Laboratoire :
Laboratoire de Mathématiques et ses Applications de Valenciennes (LAMAV)

Nouveaux outils pour l'animation et le design : Système d'animation de caméra pour la stop motion, fondée sur une interface haptique et design de courbes par des courbes algébriques-trigonométriques à hodographe pythagorien

JURY

Président du jury
- Ali Mehmeti, Felix. Professeur des Universités - Université de Valenciennes.

Rapporteurs
- Biard, Luc. Maître de Conférences, HDR - Université Joseph Fourier de Grenoble 1.
- Daniel, Marc. Professeur des Universités - Université d'Aix-Marseille.

Examinateurs
- Ali Mehmeti, Felix. Professeur des Universités - Université de Valenciennes.
- Neveu, Marc. Professeur des Universités - Université de Bourgogne.

Directeur de thèse
- Albrecht, Gudrun. Professeure des Universités - Université de Valenciennes.
Co-directeur de thèse : Romani, Lucia. Chercheuse - Université de Milan-Bicocca.

Membres invités
- Lissarrague, Nicolas. Maître de Conférences, Laboratoire Culture, Arts, Littérature, Histoires des Sociétés et des Territoires Etrangers (CALHISTE) - Université de Valenciennes.

Université de Valenciennes et du Hainaut-Cambrésis

Thèse de doctorat

Pour obtenir le grade de Docteur de l'Université de

VALENCIENNES ET DU HAINAUT-CAMBRESIS

Spécialité : MATHÉMATIQUES APPLIQUÉES

Présentée et soutenue par Laura SAINI.

Le 13/06/2013, à Valenciennes

École doctorale : Sciences Pour l'Ingénieur (SPI)

Laboratoire : Laboratoire de Mathématiques et ses Applications de Valenciennes (LAMAV)

New tools for animation and design: a haptic-based system for stop motion camera animation and curve design by algebraic-trigonometric Pythagorean Hodograph curves.

Nouveaux outils pour l'animation et le design : système d'animation de caméra pour la stop motion, fondée sur une interface haptique et design de courbes par des courbes algébriques-trigonométriques à hodographe pythagorien.

JURY

Rapporteurs

- BIARD, Luc. Maître de Conférences, HDR - Université Joseph Fourier de Grenoble 1.
- DANIEL, Marc. Professeur des Universités - Université d'Aix-Marseille.

Examinateurs

- ALI MEHMETI, Felix. Professeur des Universités - Université de Valenciennes.
- NEVEU, Marc. Professeur des Universités - Université de Bourgogne.

Directrice de thèse : ALBRECHT, Gudrun. Professeure des Universités - Université de Valenciennes.
Co-directrice de thèse : ROMANI, Lucia. Chercheuse - Université de Milan-Bicocca.

Abstract

In Computer Aided (Geometric) Design (CA(G)D) curves are usually represented in a parameterized form. This way of expressing curves is practical as well as efficient. It is independent of the choice of the coordinate system, it lends itself well to geometric transformations, such as rotations, translations, and scaling, and the differential geometric properties of a curve, as length and curvature, are invariant under re-parameterization. The simplicity of this form is used for the design of geometric models for object shapes in many applications including automotive, shipbuilding, and aerospace industries, industrial and architectural design, and many more. In addition, this representation allows to generate points on a curve in the direction of increasing values of parameter. This property is also widely used to produce computer animation. In this context, in this thesis, we study stop motion animation, that is an animation technique that brings objects, such as puppets or clay models, alive by photographing a series of positions and then playing them as a continuous sequence. Originally, only the objects are moved in a stop-motion animation, because animating the camera is very complicated. Moreover, it is an impossible job to move the camera frame by frame along a continuous curve to produce a smooth movement. Because of these constraints, the camera was strongly fixed on the real stage for a long time. But on the other hand, camera movements subject to the influence of floor irregularities, human manipulations and mechanical imperfections are mainly recognized as part of the aesthetic cinematographic specificity, and therefore desirable to a certain extent. So far, even traditional animation methods in 3D software animation programs suffer from limitations in producing realistic camera moves. Our objective is to create a new motion control system specifically designed for stop motion that is able to simulate a realistic camera animation, be adapted to stop motion stages in terms of size and weight as well as be accessible to any-sized budget productions. In chapter 2, we describe the partial solutions proposed in the literature and we use them with the aim to overcome the existing drawbacks of the 3D animation software. We aim at simulating a 3D camera movement that can integrate constraints and imperfections of real camera devices by using a haptic interface. We focus on "Keyframing animation" and propose a system that separates position and speed of the trajectory curve. Once elaborated, the keyframes, recorded by a haptic interface, are exported, frame by frame, to the motion control software, which allows to calibrate the motion control robot, to control the camera settings and, finally, to execute the sequences. We describe the whole system and explain in detail the mathematical processing to obtain different camera movements by using a haptic interface for motion capture in chapter 3. In our system we can control a rational parametric cubic Bézier curve and manage the speed of our robot, by using the concept of the Ease Curve, which represents arc length over time-frame. In this way, we can find a new parameterization for our rational parametric cubic Bézier curve that is less noisy than the movement captured by the

haptic system but still respects the original design intent. We thus can simulate different behaviours of a real camera device to give stop motion animators total artistic freedom. In the first step we project the motion capture sequence on the rational parametric cubic Bézier curve to allow the robot to move along the given trajectory. We know the length of the trajectory curve, by approximating the integral using a Gaussian quadrature. By using Newton's method we determine curve parameters to calculate the coordinates of the corresponding curve points. In the second step we determine an "ideal" speed that maintains the haptic one. By using the least squares method, we fit a quartic polynomial to the discrete data of the Ease Curve that represents the "ideal" speed. Thus, we can calculate the ideal curve parameterization. In the third step we calculate, by blending, different Ease Curves between the "haptic" and the "ideal" one. Finally, we determine the corresponding curve parameterizations. To get a smooth trajectory curve that takes into account the constraints of a real camera move we create a new mathematical representation of the camera's trajectory based on the concept of Pythagorean-Hodograph (PH) curves, described in chapter 4. This representation allows to determine the curve's arc-length in an analytical way as well as an exact rational parameterization of the offset curves, and is usable in general curve design applications. Our new class of curves is built-upon a five-dimensional mixed algebraic-trigonometric space and is called "Algebraic-Trigonometric Pythagorean-Hodograph" or ATPH. It depends on a parameter which can be used as shape parameter and reproduces arcs of arbitrary length of planar trigonometric curves, as circles, cardioids, deltoids, limacons, lemniscates, piriforms. These properties help us to create a realistic camera movement. To this end, in particular, we solve the first order Hermite interpolation problem and construct spirals as G^2 transition elements between a line segment and a circle, as well as between a pair of external circles. Our system, specifically designed to help animators realize a realistic camera movement for stop motion, will be a benefit for all types of stop motion productions. With an optimized workflow, such a system will significantly encourage creativity while respecting the handwork aesthetic of stop motion, intensify cinematographic illusion by giving life to camera and allow as much freedom for camera moves as on a real stage.

Résumé

En C(G)AO, Conception (Géométrique) Assistée par Ordinateur, les courbes sont généralement représentées sous forme paramétrique. Cela permet une grande simplicité de la modélisation de la forme de l'objet, qui est utilisée dans de nombreuses applications, telles que l'automobile, la construction navale, l'aérospatiale et le design industriel et architectural. En outre, la possibilité de générer une séquence ordonnée de points sur la courbe trouve aussi de nombreuses applications dans l'animation 3D. Dans ce cadre, cette thèse propose l'étude de la stop motion, qui est un type particulier d'animation, inventée dans les premiers temps de la naissance du cinéma, qui permet de rendre "vivants" presque tous les objets, en donnant l'impression qu'ils bougent par eux-mêmes. L'illusion du mouvement est obtenue en manipulant ces objets dans une succession de poses figées, en photographiant chacune de ces poses et en projetant le résultat comme une séquence continue. Ce procédé long et complexe est encore plus périlleux lorsqu'il concerne un mouvement de caméra : toute erreur se traduit par une secousse de toute l'image et un mouvement d'autant plus saccadé et gênant pour le spectateur. Par conséquent, sur un tournage en stop motion, la caméra est le plus souvent fermement arrimée au sol. Cependant, l'importance des mouvements de caméra comme partie de l'esthétique spécifiquement cinématographique a été maintes fois soulignée et théorisée. L'objectif de la thèse est donc de proposer un système de contrôle de mouvement par ordinateur, capable de simuler un mouvement de caméra comme s'il avait été réalisé par un appareil de prise de vue réel et pouvant être (re)produit facilement car ne nécessitant qu'un outillage et des matériaux accessibles, pour un coût aussi modeste que possible. L'étude bibliographique, décrite dans le chapitre 2, porte sur l'utilisation des principaux logiciels d'animation et sur les détails théoriques et des solutions existantes. Elle nous a permis de créer un moyen innovant et plus efficace pour aider la génération d'animations en stop motion par le biais des dispositifs haptiques et de mettre au point un système de montage nouveau et intuitif pour l'animation d'une caméra par ordinateur. Nous proposons un système de création d'animation par images clés qui sait dissocier le comportement spatial et temporel d'un mouvement en deux courbes différentes. Les images clés, générées par un périphérique de motion capture (Novint Falcon), sont envoyées à un logiciel de gestion de motion capture, de motion control et de prise de vue, puis à un robot qui exécute la séquence vidéo. Les étapes de ce système et les détails du procédé mathématique utilisé sont décrits dans le chapitre 3. Une courbe paramétrique Bézier rationnelle cubique nous permet de contrôler la trajectoire du robot et les rotations de la caméra. Elle est parametrisée à l'aide d'une Ease Curve, qui représente l'abscisse curviligne sur le temps et qui permet de changer la vitesse de notre robot. Cela nous permet d'éliminer le bruit dans la séquence capturée par le système haptique tout en gardant le mouvement enregistré, et cela donne une plus grande liberté de choix artistique à l'animateur/trice. La première étape consiste à projeter la séquence de points cap-

turée par le système haptique sur la courbe paramétrique Bézier rationnelle cubique pour permettre au robot de suivre la trajectoire initiale. À partir de la longueur de la courbe, approximée par une quadrature de Gauss, nous trouvons les valeurs de ses paramètres, en utilisant la méthode de Newton, pour déterminer les coordonnées des points correspondants sur la courbe. La deuxième étape détermine une vitesse "idéale" qui conserve l'évolution de la vitesse haptique. En utilisant la méthode des moindres carrés, nous déterminons à partir des données discrètes de l'Ease Curve un polynôme de degré quatre qui représente la vitesse "idéale". Nous pouvons ainsi déterminer la paramétrisation idéale de la courbe. Dans la troisième étape, nous calculons différentes Ease Curves entre l'"haptique" et l'"idéale", par une combinaison barycentrique. Finalement, nous déterminons les paramétrisations "intermédiaires" correspondantes. Afin d'imiter au mieux le comportement du matériel de prise de vue cinématographique réel, nous avons développé, dans le chapitre 4, une nouvelle représentation mathématique de la trajectoire de la caméra à partir des courbes polynomiales paramétriques Pythagorean-Hodograph (PH), aussi utilisable dans des applications de design de courbes. Cette représentation permet notamment de calculer l'abscisse curviligne à l'aide de formules analytiques plutôt que numériques et d'avoir une paramétrisation rationnelle exacte des courbes offset. Notre nouvelle classe de courbes, construite sur un espace algébrique-trigonométrique est nommée "Algebraic-Trigonometric Pythagorean-Hodograph" ou ATPH. Elle permet en outre de modifier la forme de la courbe en utilisant un paramètre de forme et de reproduire des arcs de courbes trigonométriques planes, tels que cercles, cardioïdes, deltoïdes, limaçons, lemniscates, piriformes. Ces propriétés nous aident à modéliser le mouvement d'une caméra réelle. Pour cela, nous résolvons en particulier un problème d'interpolation d'Hermite et nous construisons des courbes de transition entre une droite et un cercle ou entre deux cercles externes possédant des raccords de continuité G^2. Ce système permettra d'une part de faciliter le travail de l'animateur/trice et d'autre part ouvrira un champ de possibilités supplémentaires quant à l'ajout de contraintes ciblées. Enfin, il aura des applications pratiques dans le domaine de la prise de vue en stop motion.

Remerciements

Ma thèse de doctorat a été réalisée grâce au financement de la "Région Nord-Pas de Calais", ainsi que de l'Université de Valenciennes et du Hainaut-Cambrésis.
Je leur adresse tout particulièrement mes remerciements.

J'exprime mes remerciements à Luc Biard et Marc Daniel qui ont accepté d'être les rapporteurs de cette thèse et d'avoir pris une grande partie de leur temps pour lire ce mémoire. Je remercie aussi l'ensemble des membres du jury, Felix Ali Mehmeti et Marc Neveu pour avoir accepté d'examiner mon travail.

Je souhaite remercier Nicolas Lissarrague qui m'a introduit à la partie pratique et applicative de cette thèse et qui a permit de développer la partie artistique de ce projet.

Enfin, j'exprime ma gratitude à Gudrun Albrecht et à Lucia Romani, directrice et co-directrice de thèse, pour leur encadrement et leurs conseils.

Contents

1 Introduction **1**

2 Stop motion **5**
2.1 What is 3D animation? . . . 5
2.2 Stop motion: a state of the art . . . 5
2.2.1 Camera movements in current stop motion productions . . . 7

3 Existing tools for camera animation **11**
3.1 Practice . . . 11
3.1.1 Keyframing Animation . . . 11
3.1.2 Path Constraint Animation . . . 12
3.2 Theory . . . 14
3.2.1 Techniques of interpolation . . . 14
3.2.2 Parametrisation . . . 16

4 A new system for generating stop motion camera movements **21**
4.1 Introduction to an haptic system . . . 21
4.2 Steps of the whole system . . . 22
4.2.1 Motion capture device . . . 24
4.2.2 Camera movement management . . . 24
4.2.3 Motion control software . . . 25
4.2.4 Motion control robot and digital camera . . . 26
4.3 Mathematical processing . . . 26
4.3.1 Translation . . . 33
4.3.2 Rotation . . . 34
4.4 System assessment . . . 35

5 A new class of curves: Algebraic-Trigonometric Pythagorean-Hodograph **39**
5.1 Normalized B-bases for pure trigonometric and mixed algebraic–trigonometric spaces . . . 40
5.1.1 General results on normalized B-bases of spaces of real functions 40
5.1.2 Normalized B-bases of spaces of trigonometric and mixed algebraic–trigonometric functions . . . 42
5.2 AT-Bézier curves over the mixed algebraic-trigonometric space U_5 . . . 48
5.3 Algebraic-Trigonometric Pythagorean Hodograph (ATPH) curves and their properties: real representation . . . 53
5.4 C^1 Hermite interpolation problem: first approach . . . 55
5.5 Algebraic-Trigonometric Pythagorean Hodograph (ATPH) curves and their properties: complex representation . . . 58

5.6 C^1 Hermite interpolation problem: reformulation and analysis 63
5.6.1 How to identify the "best" ATPH Hermite interpolant 65
5.7 Construction of ATPH spirals 73
5.7.1 ATPH spiral for designing a G^2 transition between a line and a circle 74
5.7.2 S-shaped ATPH spiral for designing a G^2 transition between two circles 82

6 Conclusions **89**

A The structure of the motion control system **91**
A.1 Haptic interface: Novint Falcon 92
A.2 Matlab 94
A.3 3D Studio Max 97
A.4 Motion control software 99
A.4.1 Open source Motion control software: home made 100
A.4.2 Professional Motion control software: Dragonframe 101
A.5 Motion control robot and digital camera 101
A.5.1 Future developments: towards Version 0.3.3 102

Bibliography **111**

Chapter 1

Introduction

In Computer Aided (Geometric) Design (CA(G)D) curves are usually represented in a parameterized form. In general, a parametric curve is a function of one independent parameter, usually denoted by t, and it is commonly written as:

$$P(t) = (x(t), y(t)) \quad \text{for} \quad a \leq t \leq b, \quad \text{with } a, b \in \mathbb{R}.$$

In Figure 1.1 an example of a parametric curve. This way of expressing curves is

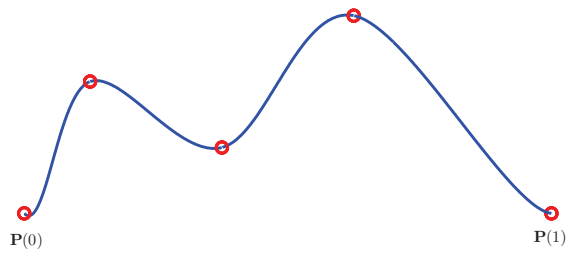


Figure 1.1: Exemple of a parametric curve $\mathbf{P}(t)$ for $0 \leq t \leq 1$.

practical as well as efficient. It is independent of the choice of the coordinate system, it lends itself well to geometric transformations, such as rotations, translations and scaling, and the differential geometric properties of a curve, as length and curvature, are invariant under reparametrisation. The simplicity of this form is used for the design of geometric models for object shapes in many applications including automotive, shipbuilding, and aerospace industries, industrial and architectural design, and many more. In addition, this representation allows to generate points on a curve in the direction of increasing values of the parameter. This property is also widely used to produce computer animation, as described in [Lasseter 1987].

In this thesis, in particular, we study stop motion animation, that is an animation technique that brings objects, such as puppets or clay models, alive by photographing a series of positions and then playing them as a continuous sequence, as we can see in Figure 1.2. A detailed description of this animation technique is

Figure 1.2: Sequence of photos to create a stop motion video.

presented in chapter 2. Originally, only the objects are moved in a stop-motion animation, because animating the camera is very complicated. Moreover, it is an impossible job to move the camera frame by frame along a continuous curve to produce a smooth movement. Because of these constraints, the camera was strongly fixed on the real stage for a long time. On the contrary, on real stages camera movements are mainly recognized as part of the aesthetic cinematographic specificity. They are subject to the influence of floor irregularities, human manipulations and mechanical imperfections and therefore desirable to a certain extent as these provide realism to the stop motion movies. So far, even traditional animation methods in 3D software animation programs suffer from limitations in producing realistic camera moves. Our objective is to create a new motion control system specifically designed for stop motion that is able to simulate a realistic camera animation, be adapted to stop motion stages in terms of size and weight as well as be accessible to any-sized budget productions. In chapter 3, we describe the 3D animation theory (see, e.g., [Sharpe 1982], [Guenter 1990], [Hongling 2002], [Parent 2004] and [Eberly 2008]) and the partial solutions proposed in the literature by [Snibbe 1995], [Steketee 1985] and [Watt 1991]. We use them with the aim to overcome the existing drawbacks of the 3D animation software, as Maya (see, e.g., [Derakhshani 2009]) and 3D Studio Max (see, e.g., [Murdock 2001]). We aim at simulating a 3D camera movement that looks as realistic as possible by integrating constraints and imperfections of real camera devices by using a haptic interface. We focus on "Keyframing animation" (see, e.g., [Verth 2004], [Govil-Pai 2004] and [Kochanek 1984]) and propose a system that separates position and speed of the trajectory curve. Once elaborated, the keyframes, recorded by a haptic interface, are exported, frame by frame, to the motion control software, which allows to calibrate the motion control robot, to control the camera settings and, finally, to execute the sequences. We describe the whole system and explain in detail the mathematical processing to obtain different camera movements by using a haptic interface for motion capture in chapter 4. In our system, that uses the graphical interface from [Fünfzig 2010], we can control a rational parametric cubic Bézier curve and manage the speed of our robot, by using the concept of the Ease Curve, which represents arc length over time-frame. In this way, we can find a new parametrisation for our rational parametric cubic Bézier curve that is less noisy than the movement captured by the haptic system but still respects the original design intent. We thus can simulate different behaviours of a real camera device to give stop motion animators total artistic freedom. In the first step we project the motion capture sequence on the rational parametric cubic Bézier curve to allow the robot to move along the given trajectory. We know

the length of the trajectory curve, by approximating the integral using a Gaussian quadrature (as described in [Saini 2010]). By using Newton's method we determine curve parameters to calculate the coordinates of the corresponding curve points. In the second step we determine an "ideal" speed that maintains the haptic one. By using the least squares method, we fit a quartic polynomial to the discrete data of the Ease Curve that represents the "ideal" speed. Thus, we can calculate the ideal curve parametrisation. In the third step we calculate, by blending, different Ease Curves between the "haptic" and the "ideal" one. Finally, we determine the corresponding curve parametrisations.

To get a smooth trajectory curve that takes into account the constraints of a real camera move we create a new mathematical representation of the camera's trajectory. In chapter 5, we present a trigonometric analogue of the Pythagorean Hodograph quintic, featured by the property of possessing exactly computable arc length and offset curves. First, the novel class of Algebraic–Trigonometric PH curves (called for short ATPH curves) is presented in a real representation. We construct C^1 ATPH curves that interpolate given end points and associated end derivatives. Next, we introduce, and we use hereinafter, a complex representation and we reformulate the previous results in this concise notation. This allows us to thoroughly analyse the obtained solutions of the Hermite interpolation problem. Then, we present G^2-continuous ATPH curves of monotone curvature joining basic elements such as line segments and circles. In both application contexts we will show that the ATPH interpolants compare favorably with their polynomial PH counterpart. In fact, concerning the C^1 Hermite interpolation problem, we will see that, although there exist Hermite data such that all possible *polynomial* PH solutions manifest undesired self-intersections, *ATPH* interpolants constructed from the same information turn out to be free of loops if the free parameter α is suitably selected. Moreover, the free parameter α can be also conveniently exploited either to improve the curvature behavior of ATPH spirals joining G^2-continuously a line and a circle or to adjust the location of the second point of contact of the spiral as well as the curvature profile and/or variation in the case of S-shaped spirals joining G^2-continuously a pair of external circles. We would like to use these properties to help create a realistic camera movement.

In chapter 6 we conclude summarising and commenting our system and describing our work in progress. It is specifically designed to help animators realize a realistic camera movement for stop motion and it will be a benefit for all types of stop motion productions and 3D animations. With an optimized workflow, such a system will significantly encourage creativity while respecting the handwork aesthetic of stop motion, intensify cinematographic illusion by giving life to camera and allow as much freedom for camera moves as on a real stage.

Chapter 2

Stop motion

2.1 What is 3D animation?

Computer animation is the art of creating moving images via the use of computers. It is a subfield of computer graphics (it studies the manipulation of visual and geometric information using mathematical and computational techniques) and animation (from Latin *animatio*, animation literally means "the act to bringing life"). In a 2D animation the objects and the images are in two dimensions, on the other hand in a 3D animation there are 3D models and figures.
In both cases the cinematographic animation is based on the 20th century traditional animation techniques, called cel animation or hand-drawn animation. The illusion of movement is created by displaying successively a sequence of slightly different draws to create smooth and continuous movements. Traditionally, animation was created by drawing each individual image, or frame, in the animated sequence. The lead animator creates the keys, or important frames, and a second animator creates the in-between frames. For these reasons the first animator is called "key-animator", the second one "tweener" and the name of this technique is *keyframing animation*. By the beginning of 21th century the most basic computer animation tools assist the process of traditional animation by automatically generating some of the frames of animation and by using 3D graphics.

2.2 Stop motion: a state of the art

In 1896, one year after the official birth of Cinema, Eugène Promio in *Vues de Venise* puts his camera on a boat making the first travelling of the cinema history. Since W.D. Griffith, camera movements are not only used to follow the action on stage, but they are also and mainly recognized as one of the expression of a cinematographic specificity, a way to construct atmosphere, to play with space and much more. J.L. Godard even certified that "the tracking shot is a moral issue". Camera moves have greatly contributed to generating the personal style and aesthetics of directors such as S. Spielberg, R. Polanski, R. Altman, A. Resnais, S. Kubrick, M. Scorsese, M. Cimino, O.Welles or F.F Coppola, and some of these camera moves are now part of the history of cinema like the opening shot of *Touch of Evil* (Orson Welles, 1958).
In 1897, two years after the official birth of Cinema, Georges Méliès did a short advertising (its title has been lost) where some wooden toy letters move "magically" to make the advertiser's name [Harryhausen and Dalton 2008]. This short film is known as the first *stop motion animation* movie, a special type of animation where

an object (model, puppet, clay) is moved slightly between two frames: when all photos are gathered to make a sequence, the illusion of movement is created. It's a (very) long and (very) tedious process. Since then, technicians and directors have explored the creative possibilities of this technique, from Dziga Vertov's *Man with a movie camera* (1929), Merian C. Cooper and Ernest B. Schoedsack's *King Kong* (1933) (Figure 2.1(a)) to Nathan Juran's *The 7th voyage of Sinbad* (1958) or Tim Burton's *Corpse bride* (2005) (Figure 2.1(b)).

Figure 2.1: Poster of *King Kong* (left) and *Corpse bride* (right).

Technical specificities

Stop-motion is an animation technique that brings objects, such as puppets or clay models, alive by photographing a series of positions and then playing these as a continuous sequence. In particular, if we also move the camera we obtain a *stop-motion camera animation*. The camera shots are made frame by frame and the camera is slightly moved between frames. Once they are assembled it produces an illusion of movement. The main difficulty of this technique is that if an animator does a mistake on set, it is not possible to go back and repeat elements of a movement as it can never be recaptured exactly in the same way. Moreover, an animator can produce 4 to 12 seconds of animation maximum per day. It is almost an impossible job to move camera frame by frame along a continuous curve to produce a smooth move. First, because stop motion frequently uses model sets, which means that the camera movements must be scaled down. Second, because the slightest imprecision

produces a shake in the final sequence (the intensity of which depends on the scale factor). A mistake with the camera affects the whole image. Third, because unlike on real stage, there is no way to make several shots and choose the best as the process is too long, as we have seen before. For a long time, these technical constraints forced directors to fix the camera on the stop motion stage, while directors on real stage added an aesthetic value by using camera movements. To overcome this frustrating status, stop motion animators first tried to find tricks that could give more freedom to the camera: the first camera movement in stop motion was done with a dolly that was moved on a tilted plane with a rope and a gear system (*The Story of the tortoise and the hare*, Ray Harryhausen, 1952). More frequently, instead of moving the camera, the animators move the set itself. The result is visually the same, but the process is easier and safer than moving the camera. The whole train pursuit in *Wallace & Gromit - A grand day out* (Nick Park, 1989) was completely made with this trick: only the walls of the apartment were moved during the shot (so that the camera also captured the motion blur), but onscreen it looks like if the two heroes were moving quickly on their model train. If these tricks and tries have allowed some more freedom on stage, they have important limits: they can work for one axis of freedom (unusually two, like the travelling and panoramic move in *The Secret adventure of Thom Thumb*, David Borthwick, 1993) but no more than two axes of freedom as the process would become too complex and hazardous; they work well with linear acceleration moves but are not precise enough to handle acceleration ease in and out, or slow moves. To be able to obtain the same camera move freedom as on a real stage, Henry Selick used for the first time a motion controlled camera for Tim Burton's *The Nightmare before Christmas* (1993), see Figure 2.2(a) for the poster. The motion controlled camera can execute any movement on up to six axes, with a great precision, and can repeat the movement at will as it is computer controlled. Henry Selick used it again for *James and the giant peach* (1996) while Aardman's Production made experimentations with the same kind of device in *Chicken Run* (Peter Lord & Nick Park, 2000), see the movie's poster in Figure 2.2(b).

2.2.1 Camera movements in current stop motion productions

Since the 1993-2000 progressive appearance of motion control on stop motion sets, the stop motion movies are divided in two categories: the ones made by major production studios using motion controlled camera movements and all the others using handmade camera movements. Advantages and disadvantages of the two ways to animate a camera frame by frame can be sumarized as follows:

- "Handmade camera movement": moving the camera (or the model stage) frame by frame has two important limits. First, the movement can only be executed up to two axes of freedom, because for more than two axes the handling becomes too complex. Second, despite noisy animation is part of the stop motion aesthetic and should'nt be removed, its amplitude and characteristics cannot be controlled with handmade animation and thus can be considered as a drawback. The advantage is its cost. Ray Harryhausen is the first in the

Figure 2.2: Poster of *The Nightmare before Christmas* (left) and *Chicken Run* (right).

history of cinema to use this method, in *The Story of the Turtoise and the Hare* (1952), but other movements can be found in *Koyaanisqatsi* (Godfrey Reggio, 1982), *The secret adventure of Tom Thumb* (David Borthwick, 1993) (Figure 2.3(a)) or *MUTO* (BLU, 2008).

- "Motion controlled camera movement": The main advantage is that any kind of movement can be executed. Until recently, their limits were their cost and size. Since a couple of years ago, new kinds of motion control devices are more accessible, smaller and cheaper (Animoko and Dito Systems, for example). Nevertheless, all existing motion control systems still have non-neglectable limits regarding stop motion animation aesthetics and creativity: the camera movements are too perfect as they are based on smooth mathematical curves, none of the existing motion control systems belongs to the open source / open hardware world and the cheaper systems offer only two axes of freedom for rotation and one for translation. Many recent movies and advertising use such systems, like *Corpse Bride* (Tim Burton & Mike Johnson, 2005), *Wallace and Gromit: A matter of loaf and death* (Steve Box & Nick Park, 2008) ((Figure 2.3(b))) or *Brother's printers clips* (2010).

In this context we cite "*Stop-frame is like live music, played on traditional instruments, compared to a studio recording using the finest instruments in the world, all the latest technology and some electronic instruments. The latter is more polished,*

more perfect, bigger, better, showier - but maybe lacks humanity. Stopframe is much less perfect, much less polished, unrepeatable, inaccurate - in a word, human.", Peter Lord [Harryhausen 2008].

Figure 2.3: Poster of *The secret adventure of Tom Thumb* (left) and *Wallace and Gromit: A matter of loaf and death* (right).

In both cases, the specific noise that real camera devices produce is missing, and the result looks either too perfect for the stop motion aesthetic, or very shaky in an unnatural manner. In fact, procedural noise generated by 3D software is far from being able to simulate the specific noise made by a camera-on-the-shoulder or a dolly for example. The main problem with motion control is that it is driven by 3D curves that are too perfect to simulate a realistic camera device. The next chapter shows how 3D software programs manage animation curves and underlines their limits in the perspective of realistic camera animation. Our objective is to create a realistic 3D camera animation and to reproduce this movement with a motion control system by a stop motion animation technique. Then, we present a new motion control system specifically designed for stop motion, that is able to simulate a realistic camera animation in order to give stop motion animators total freedom of camera movement that maintains the handwork visual aesthetics of stop motion. In particular we are aiming at simulating a 3D camera movement that can integrate constraints and imperfections (noise) of real camera devices by using a haptic interface.

Chapter 3

Existing tools for camera animation

In this chapter we present a state of the art for 3D animation of camera movements. After describing the current practice in existing commercial software programs (section 3.1) we detail the underlying theory and existing theoretical improvement proposals (section 3.2).

3.1 Practice

The main 3D animation software programs are Maya (see, e.g., [Derakhshani 2009]), 3D Studio Max (see, e.g., [Murdock 2001]) Lightwave, Blender, Cinema4D, Softimage and Houdini. They have two main different tools to animate an object: *Keyframing animation* and *Path constraints animation*. We describe the two methods by emphasizing the advantages and disadvantages of each.

3.1.1 Keyframing Animation

Keyframing Animation is based on the traditional animation technique, where the user only sets the n important frames, called *keyframes*, and, using interpolation techniques, the software program generates the r intermediate frames, called *in-betweens*. An object in 3D space is represented with respect to time: the 3D position $\mathbf{p}_i$ of the object corresponds to the time t_i for $i = 1, \ldots, r$, with $r \geq n$ and keyframes are associated with specific values of the time parameter t. The object's trajectory $\mathbf{P}(t)$ in 3D space is composed of its three coordinates $x(t)$, $y(t)$ and $z(t)$, where all these curves may be visualized in parallel by the animation software program. See Figure 3.1 for an illustration of an object and its $n = 3$ keyframes at values of the time parameter t_1, t_{12} and t_{24} and its $r = 24$ positions in 3D space. Figure 3.2 shows an editor visualizing the corresponding components $x(t)$, $y(t)$ and $z(t)$. This editor, which more generally allows to show different attributes of the curve, such as position coordinates, colors, texture, rotation etc., is called "Curve Editor" in 3D Studio Max and "Graph Editor" in Maya. The user Once $n \leq r$ keyframes, i.e., n locations in time t_i and corresponding space positions $\mathbf{p}_i$, have been specified by the user, the software program determines a piecewise parameterized curve $\mathbf{P}(t) = \bigcup_{i=1}^{n-1} \mathbf{P}_i(t)$ composed of the curves $\mathbf{P}_i(t)$ interpolating the points $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$.

The role of the Curve/Graph Editor is twofold: it determines the space position of the object, i.e., its space trajectory, as well as the way the object moves

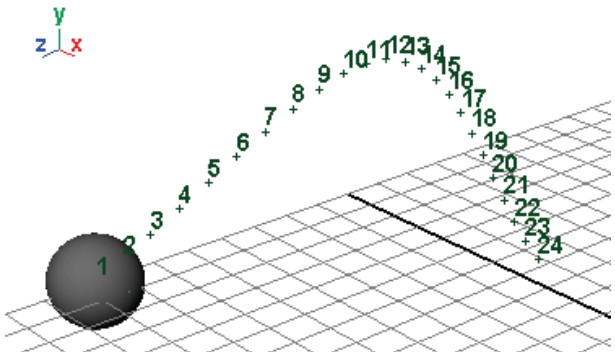


Figure 3.1: Representation of an object and its 3D trajectory in Maya.

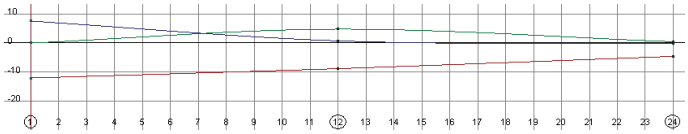


Figure 3.2: Maya's Graph Editor shows parameterized curves of the object's position attributes over time (x, y respectively z coordinate in red, green respectively blue).

along its space trajectory, i.e., its speed. Between two keyframes the interpolation curve is automatically calculated together with its parameterization which in general is not uniform. That means that for equally spaced parameter values t_i the corresponding space positions $\mathbf{p}_i$ are in general not equally spaced on the space trajectory. In order to achieve equal spacings on $\mathbf{P}(t)$ an animator modifies the curves in the Curve/Graph Editor thus changing also the space trajectory. Summarizing, "Keyframing Animation" has the following advantages(+) and disadvantages(-):

+ the local control of the space trajectory is very good and flexible;

+ the Curve/Graph Editor allows to control a large number of curve attributes, including the addition of noise in order to produce a realistic movement;

- speed and position are dependent;

- there isn't a global control of the curve, but only locally between any two keyframes.

3.1.2 Path Constraint Animation

Another motion animation method, called Path Constraint Animation, allows to move an object along a 3D curve as trajectory, called Path Constraint (Figure 3.3). The curve that we have in the corresponding Curve/Graph Editor, which in this case shows keyframes over time, explains how the object moves along it: by default this curve, called Ease Curve, is linear and can not directly be modified (Figure 3.4(a)). But we can add another curve to it (Figure 3.4(b)), and the resulting mean

curve allows to control and change the acceleration/deceleration. In Figure 3.5 we can see different types of movements, such as a linear movement in Figure 3.5(a), an accelerating movement (Ease In) in Figure 3.5(b) and a decelerating movement (Ease Out) in Figure 3.5(c). Path Constraint Animation is very efficient to control the speed, when we have a simple trajectory. But it is difficult to generate a complex movement or to add noise to the space trajectory. Path Constraint Animation separates the position curve from the speed curve but doesn't allow to add new constraints to the space trajectory. In fact we have more control over time than space-position and we remark the following advantages(+) and disadvantages(-):

+ speed and position are independent;

+ there is a global control of the space trajectory;

- it's impossible to add other constraints, such as noise, to the Path Constraint. It's only possible to achieve a curve trajectory as an average of constraints.

- there are less degrees of freedom for modifying the space trajectory than in the keyframing animation method (e.g., the x, y, z coordinates can not be accessed in the Curve/Graph Editor).

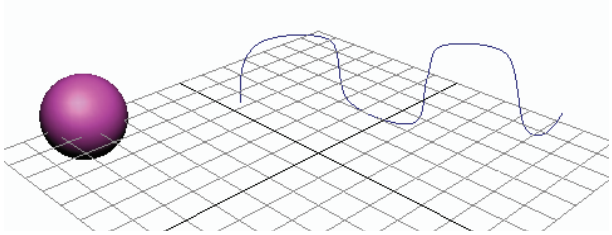
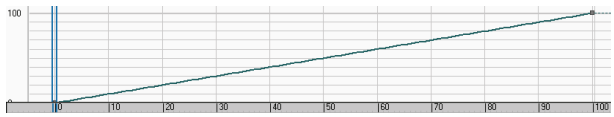
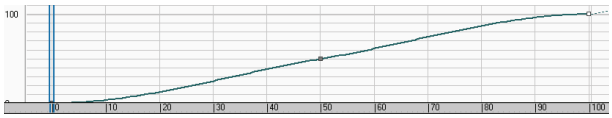


Figure 3.3: 3D Studio Max's Path Constraint Animation interface.

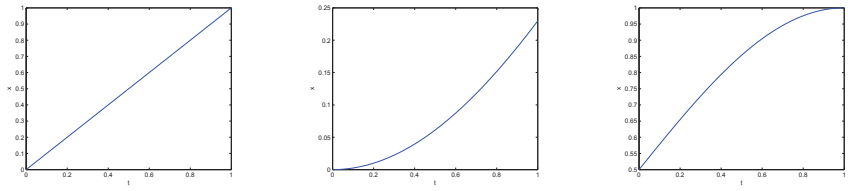


(a) Default linear Ease Curve (can't be modified)



(b) Secondary Ease Curve (can be modified)

Figure 3.4: 3D Studio Max's Curve Editor for Path Constraint Animation.



(a) Linear movement (b) Acceleration (c) Deceleration

Figure 3.5: Ease Curve that defines how the object moves.

3.2 Theory

In order to overcome the disadvantages within the existing computer animation software programs in this section we will review the partial solutions that have been proposed in literature aiming at a complete separation of the space trajectory and the speed curve of the object.

The most popular technique used in 3D computer animation is the keyframe inbetweening technique, as described in section 3.1. In-betweens are calculated by interpolation of a sequence of points. The type of interpolation used depends on the final motion desired. In subsection 3.2.1 we describe some basic interpolation methods and then in subsections 3.2.2 and 3.2.2 we explain how to use them to define an animation, in reference to existing work in literature.

3.2.1 Techniques of interpolation

We have a sequence of n points $\mathbf{p}_i$ with $i = 1, \ldots, n$. We want to find a sequence of $n-1$ curves $\mathbf{P}_i(u)$ that interpolate the points $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$ for $u \in [0,1]$. There are many different interpolation methods, but the most used in animation are the following (see, e.g., [Verth 2004] and [Govil-Pai 2004]):

- *piecewise linear interpolation*

$$\mathbf{P}_i(u) = \begin{bmatrix} u & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_i \\ \mathbf{p}_{i+1} \end{bmatrix},$$

  where the coincidence $\mathbf{P}_i(1) = \mathbf{P}_{i+1}(0)$ guarantees $\mathcal{C}^0$ curve continuity;

- *piecewise Hermite cubic curves*

$$\mathbf{P}_i(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cdot$$

$$\cdot \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_i \\ \mathbf{p}_{i+1} \\ \mathbf{p}'_i \\ \mathbf{p}'_{i+1} \end{bmatrix},$$

where $\mathbf{P}_i(1) = \mathbf{P}_{i+1}(0)$ and $\mathbf{P}'_i(1) = \mathbf{P}'_{i+1}(0)$ guarantee $\mathcal{C}^1$ curve continuity. $\mathbf{p}'_i$ and $\mathbf{p}'_{i+1}$ denote the first derivative data to be provided. In particular we have a *Catmull-Rom curve* if tangents are calculated as

$$\mathbf{p}'_i = \frac{1}{2}(\mathbf{p}_{i+1} - \mathbf{p}_{i-1}),$$

$$\mathbf{p}'_{i+1} = \frac{1}{2}(\mathbf{p}_{i+2} - \mathbf{p}_i),$$

see Figure 3.6, blue. We get a *TCB curve*, ([Kochanek 1984]), if tangents depend on three parameters: *tension* τ_i which controls how sharply the curve bends at a point $\mathbf{p}_i$, *continuity* c_i which controls the continuity or discontinuity at a point $\mathbf{p}_i$, and *bias* b_i which controls the direction of the path at point $\mathbf{p}_i$. The general form of these tangent vectors is

$$\begin{aligned}\mathbf{p}'_i &= \frac{(1-\tau_i)(1+c_i)(1+b_i)}{2} \cdot (\mathbf{p}_i - \mathbf{p}_{i-1}) + \\ &+ \frac{(1-\tau_i)(1-c_i)(1-b_i)}{2}(\mathbf{p}_{i+1} - \mathbf{p}_i), \\ \mathbf{p}'_{i+1} &= \frac{(1-\tau_{i+1})(1-c_{i+1})(1+b_{i+1})}{2} \cdot (\mathbf{p}_{i+1} - \mathbf{p}_i) + \\ &+ \frac{(1-\tau_{i+1})(1+c_{i+1})(1-b_{i+1})}{2}(\mathbf{p}_{i+2} - \mathbf{p}_{i+1}),\end{aligned}$$

see Figure 3.6, black, where we assume an equal number of in-betweens within each interval and we look only at the path of the motion. If the number of in-betweens for adjacent intervals is not equal, we have different step sizes, thus creating a discontinuity in the speed of motion. In order to maintain the speed continuity it becomes necessary to weight the tangent vectors, see [Kochanek 1984].

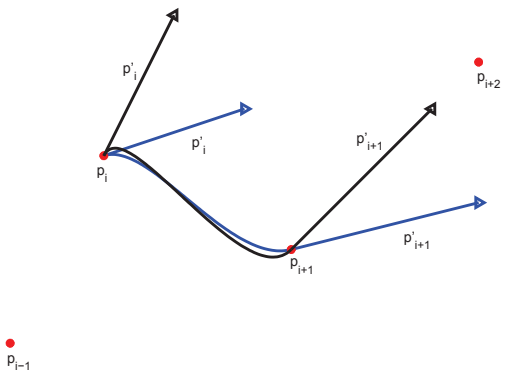


Figure 3.6: From the sequence of 4 points $p_{i-1} = (1, 1, 0)$, $p_i = (2, 3, 0)$, $p_{i+1} = (4, 2, 0)$, $p_{i+2} = (6, 4, 0)$ we have represented the corresponding Catmull-Rom curve and its tangent vectors in blue and the TCB curve, with $\tau_i = \tau_{i+1} = 0.5$, $c_i = c_{i+1} = 1$ and $b_i = b_{i+1} = 0.5$, and its tangent vectors in black.

3.2.2 Parametrisation

In computer animation object or virtual camera motion is usually defined by means of a parametric space curve $\mathbf{P}(t)$, where t is the curve parameter. An object is moved along the curve $\mathbf{P}(t)$ by advancing the parameter by a constant amount Δt and by calculating the coordinates of the corresponding curve points. Usually the uniform displacement Δt in the parameter domain does not correspond to uniform distances on the curve. It is thus difficult to control the speed by which the object moves along the curve. Only arc length parametrization guarantees uniform distances on the curve $\mathbf{P}(t)$. The related arc length parameter is usually referred to by s and the problem thus consists in relating the parameter t to the arc length parameter s by a function $s = A(t)$. The difficulty is that in general there is no analytic expression for this function A and therefore we can't calculate the inverse of the arc length function, $t = A^{-1}(s)$. As a consequence, numerical techniques have been proposed (see, e.g., [Sharpe 1982], [Guenter 1990], [Hongling 2002], [Parent 2004], [Eberly 2008]) to compute t for each specified s. In accordance with the notation introduced in subsection 3.2.1 we describe the proposed procedure for the segment $\mathbf{P}_i(u)$ of the interpolation curve between the points $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$. The local parameter $u \in [0,1]$ is related to the time $t \in [t_i, t_{i+1}]$ by $u = \frac{t-t_i}{t_{i+1}-t_i}$. We consider the set $u_j \in [0,1]$ for $j = 0, \ldots, m$ of monotonously increasing real parameter values, where $u_0 = 0$ and $u_m = 1$. For simplicity of notation in the following description we will denote the curve piece $\mathbf{P}_i(u)$ by $\mathbf{Q}(u)$. We start calculating the length l of $\mathbf{Q}(u)$ by approximating the following integral by a Gaussian quadrature or a Gaussian adaptive method:

$$l = \int_{u_0}^{u_m} \left|\left| \frac{d\mathbf{Q}}{du} \right|\right| du = \int_{u_0}^{u_m} \sqrt{\left(\frac{dx}{du}\right)^2 + \left(\frac{dy}{du}\right)^2 + \left(\frac{dz}{du}\right)^2}\, du.$$

We want to find the parameter values $\widetilde{u}_0, \widetilde{u}_1, \ldots, \widetilde{u}_m$ corresponding to $m+1$ equally spaced points on the curve $\mathbf{Q}(u)$ where the spacing is given by the arc length $s = l/m$. We thus want to solve the following equation:

$$F(\widetilde{u}_j) = \int_{\widetilde{u}_{j-1}}^{\widetilde{u}_j} \left|\left| \frac{d\mathbf{Q}}{du} \right|\right| du - s = 0,$$

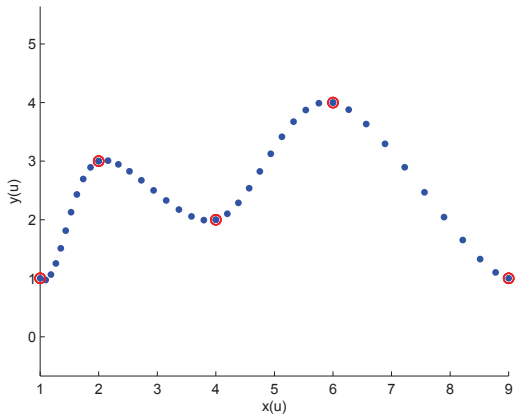
with $j = 1, \ldots, m$ and $\widetilde{u}_0 = u_0$. We can solve it by using Newton's method:

$$\widetilde{u}_j^k = \widetilde{u}_j^{k-1} - \frac{F(\widetilde{u}_j^{k-1})}{F'(\widetilde{u}_j^{k-1})}, \quad k = 1, 2, \ldots$$

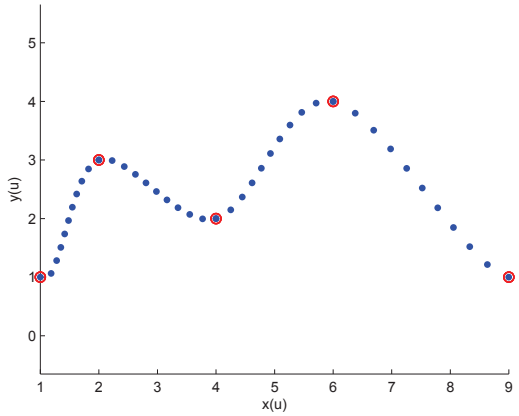
where $F(\widetilde{u}_j^{k-1})$ is approximated using standard numerical integrators and $F'(\widetilde{u}_j^{k-1})$ is straightforward because we have a formula for $\mathbf{Q}(u)$ and we can compute $d\mathbf{Q}/du$ from it. A reasonable choice for the initial iterate is

$$\widetilde{u}_j^0 = \widetilde{u}_{j-1} + \frac{s}{l}(u_m - u_0)$$

where s/l is the fraction of arc length at which the point should be located. When $F(\widetilde{u}_j^k)$ is sufficiently near to zero or a maximum number k of iterates has been computed, $\widetilde{u}_j^k$ is accepted as $\widetilde{u}_j$ and the iteration is repeated for $j = 1, \ldots, m$. Finally, the parametric curve is recalculated with the new parameter values $\widetilde{u}_j$ in terms of arc length s. In Figure 3.7 we represent a piecewise curve with two different parametriza-



(a) Equally spaced initial parameter values



(b) Equally spaced arc lengths parameter values

Figure 3.7: A parametric piecewise Catmull-Rom spline from a sequence of points (red circles) parameterized by equally spaced initial parameter values u_j (Figure 3.7(a)) and by equally spaced arc length parameter values $\widetilde{u}_j$(Figure 3.7(b)).

tions (equally spaced initial parameter values u_j (Figure 3.7(a)) and equally spaced arc length parameter values $\widetilde{u}_j$ (Figure 3.7(b))) for the same sequence of points $\mathbf{p}_i$. Please note that in this case equal spacing between every two consecutive points $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$ is assured, but the spacings differ from one piece $\mathbf{P}_i(u)$ to another $\mathbf{P}_k(u)$ $(k \neq i)$. Moreover, the arc length parametrization also permits to control the entire interpolation curve. In fact, we can fix the length s and determine $m = l/s$

equally spaced points on the entire curve $\mathbf{P}(t)$. See Figure 3.8 for an illustration. It

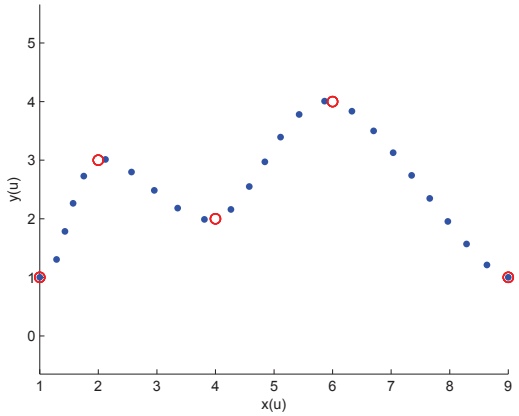


Figure 3.8: The entire parametric Catmull-Rom spline from a sequence of points (red circles) parameterized by equally spaced arc length parameters (blue points).

is now possible to control the speed at which the curve is traversed. The curve is parameterized by arc length, i.e., a unit change in the parameter value results in a unit change on the trajectory. In the next chapter, this procedure will be used to reparameterize a rational parametric cubic Bézier curve under constraints given by an Ease Curve. The curve $s(t)$, that corresponds to Maya's and 3D Studio Max's Ease Curve, controls only the speed along the space curve. Thus, to find the position along the motion curve at a given time t, arc length s is determined from the Ease curve and then s is moved along the Position Curve (Figure 3.9). On this basis,

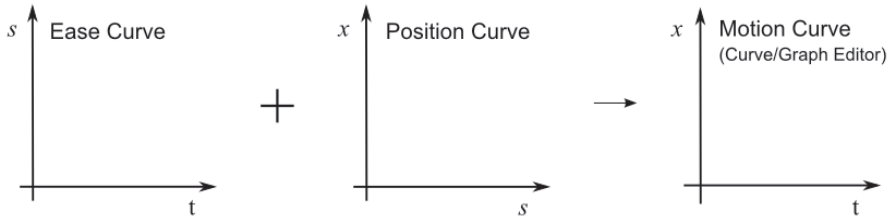


Figure 3.9: The Motion Curve in the Curve/Graph Editor is the combination of the Ease Curve and the Position Curve.

[Snibbe 1995] introduces "displacement functions" that allow to modify the space trajectory as well as the Ease curve. A function $G(t) \in [0, 1]$ is added to $\mathbf{P}(t)$ or to $s(t)$ over the interval $[t_i, t_{i+1}]$ representing the desired change for the curve. See Figure 3.10 for an illustration. We note that a change in $\mathbf{P}(t)$ changes the total arc length. To maintain the same speed it is thus necessary to scale the curve $s(t)$.

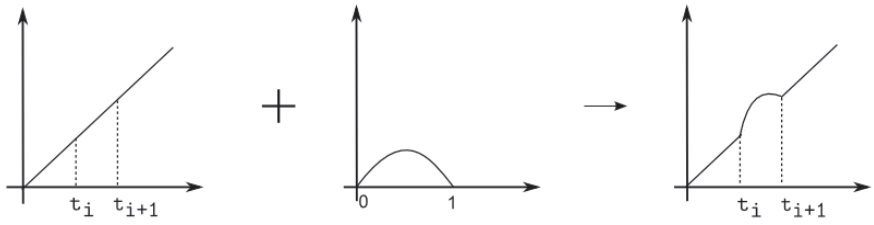


Figure 3.10: Snibbe's displacement function method.

Kinetic control

To better integrate this theoretical part, we describe the work of [Steketee 1985]. These authors generalize the Ease Curve, which they call a "kinetic spline", to drive any motion parameter, such as position, orientation, color and transparency (see also [Watt 1991]). The method is called "double interpolant" because they create the following two cubic interpolants:

1. *Kinetic spline* expresses the keyframing number as a function of time, i.e., relates keyframes to time. The n keyframe numbers $k_1, \ldots, k_n$ are interpolated at assigned times $t_1, \ldots, t_n$.

2. *Position spline* expresses the value of the motion variable as a function of the keyframe number. The n motion parameters $\mathbf{p}_1, \ldots, \mathbf{p}_n$ are interpolated at key values $k_1, \ldots, k_n$.

If the kinetic spline function is $\kappa : t \rightarrow \kappa(t)$ and the position spline function is $p : \kappa(t) \rightarrow x$, where x stands for attributes such as position coordinates, colors, texture, rotation etc., the composition of the two interpolants creates the motion function $m : t \rightarrow x$, where:

$$(p \circ \kappa)(t) = p(\kappa(t)) = m(t).$$

This function expresses the value of the motion parameter as a function of time. This method is important because when modifying the timing of the keyframes, the speed and acceleration of the motion are modified, without changing the positions defined by the keyframes. The Motion Curve can be divided into two graphs, one for the position and one for the time, so as to solve one of the problems of software animation programs, that is the dependence between position and time. In Figure 3.11 the method is shown schematically. In particular if arc length s is used instead of keyframes k, the Keyframing Animation, described in section 3.2.2, can be interpreted as the just described "double interpolant" method.

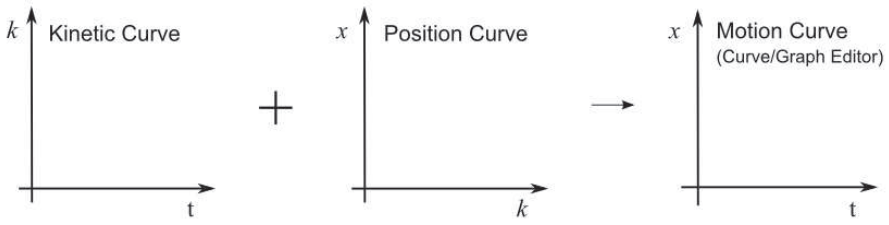


Figure 3.11: The Motion Curve is the combination of the Kinetic Curve and the Position Curve.

Chapter 4

A new system for generating stop motion camera movements

The analysis, in chapter 3, of the existing tools for camera animation allows us to develop a new system that takes into account only the advantages of these two methods. In particular, we create a new system based on the keyframing animation technique aiming at a complete separation of the space trajectory and the speed curve of the camera by using the arc-length parametrisation. In this chapter, after a brief introduction on the haptic system in section 4.1, we present in section 4.2 a new motion control system able to add constraints, by using a haptic interface, that greatly contributes to producing imperfections and the behavior of a real camera device. In section 4.4, we present an assessment of our system carried out with a class of students of the "Arts plastiques et Création numérique" Master of the University of Valenciennes. The first step of our system presented in subsection 4.2.1 has been published in [Saini 2011]. The description of the current prototype of the final system is submitted for publication.

4.1 Introduction to an haptic system

Haptics is the science of applying touch (tactile) sensation and control to interaction with computer applications. By using special input/output devices (joysticks, data gloves, or other devices), users can receive feedback from computer applications in the form of felt sensations in the hand or other parts of the body. In combination with a visual display, haptics technology can be used to train people for tasks requiring hand-eye coordination, such as surgery and space ship maneuvers. It can also be used for games in which you feel as well as see your interactions with images. For example, you might play tennis with another computer user somewhere else in the world. Both of you can see the moving ball and, using the haptic device, position and swing your tennis racket and feel the impact of the ball.

There are different haptic systems on the market. In this thesis we use one of the devices from Novint Technologies: Novint Falcon, because it is very cheap and versatile. The Novint Falcon (Figure 4.1(a)) was created primarily for the gaming industry in 2007. In fact, the optional "pistol grip" attachment (Figure 4.1(b)) makes the Novint Falcon an excellent addition to any of the supported FPS (First person Shooter) games, allowing gamers to immerse themselves even further into the gaming experience.

(a) Novint Falcon with standard grip (b) optional Pistol Grip

Figure 4.1: Novint Falcon and Pistol Grip.

The Novint Falcon is an interesting device used in many applications, as video games, medical training, as well as computer animation. In particular, in our thesis we use it to control the 3D camera movements for stop motion animation. The use of this haptic system allows the animator to determine the camera and robot speed along the trajectory.

4.2 Steps of the whole system

We describe, in this section, a new motion control system specifically designed for stop motion that is able to simulate a realistic camera animation in order to give stop motion animators total freedom of camera movement that maintains the handwork visual aesthetics of stop motion. In particular we are aiming at simulating a 3D camera movement that can integrate constraints and imperfections (noise) of real camera devices by using a haptic interface, the Novint Falcon described in section 4.1. The whole system can be summarized in the following steps, as shown in Figure 4.2, and detailed in the forthcoming subsections. We start to generate a rational parametric cubic Bézier curve by drawing the four control points and its corresponding weights. This represents the ideal camera trajectory. Once the curve is constructed, we determine the camera speed along the curve by using the haptic interface ("Motion capture device"). This peripheral allows us to feel the inertia of the real camera devices. Thus, we integrate a real noise. It represents the human manipulation imperfections. Moreover, the haptic interface feedback allows us to stay near the curve, without exactly following it. In this way, we can integrate a transversal noise corresponding to the spatial movement imperfections of the real camera device. However, the mechanical constraints of the current robot version don't allow to work on the transversal noise reproduction yet. So we add only the longitudinal noise. Next, we re-elaborate the recorded positions by the haptic interface ("Camera movement management"). Once the speed is chosen, we export the parametrisation points into a software that controls and calibrates the robot and camera movement ("Motion control software"). The sequence can be executed to create the video ("Motion control robot and digital camera").

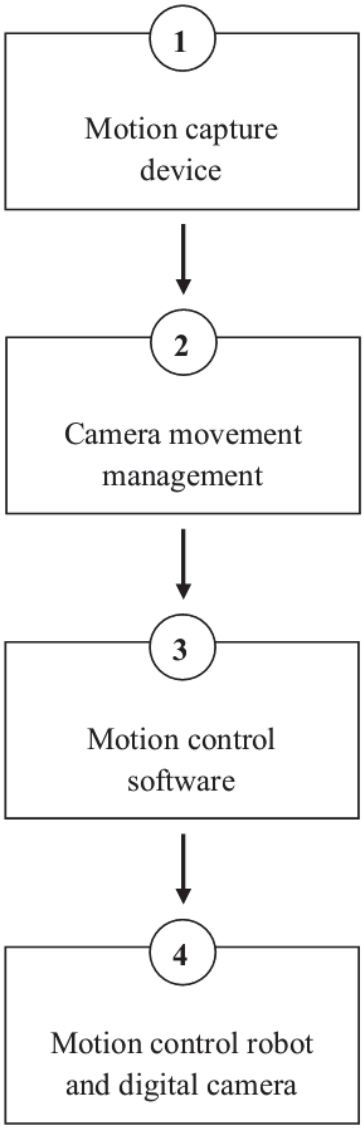


Figure 4.2: Motion control system diagram.

4.2.1 Motion capture device

We use the haptic interface to obtain the movement of the camera. Haptic devices are spatial input devices, which can themselves generate a force on the input point (force feedback). In our system we can control a rational parametric cubic Bézier curve, by using the graphical interface from [Fünfzig 2010]. The user can be guided along the curve. Firstly, the interactor point x is moved to the nearest point $p(t_x)$ on the curve. As common in haptics, a spring force in the direction $p(t_x) - x$ is used to achieve this. Secondly, the user can move forward in time along the curve (Figure 4.3). We add the possibility to determine the interactor position for every

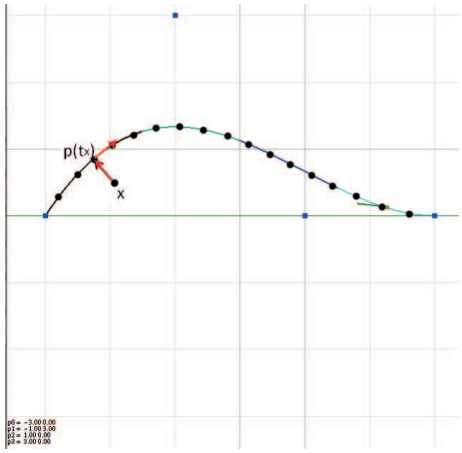


Figure 4.3: Haptic input (grey) following a curve in the graphical interface.

Δt of time. In order to respect the standard frequency, at which an imaging device produces unique consecutive images, we set $\Delta t = 0.04$ seconds, to obtain 25 "frames per second". By using the button in the back of the knob, we start recording the interactor positions along the curve and then we move along the curve with the interactor. When we release the button, we stop the recording and we obtain a sequence of n positions $\mathbf{p}_i^H$, $i = 0, \ldots, n-1$, where H indicates the haptic system's positions. The sequence $\mathbf{p}_i^H$ represents the curve's parametrisation (Figure 4.4).

4.2.2 Camera movement management

The motion capture sequence $\mathbf{p}_i^H$ is not exactly on the parametric cubic curve, as we can see in Figure 4.5. Moreover the corresponding speed is too noisy to obtain a realistic camera movement, as shown in Figure 4.13. We want the camera to exactly follow the rational parametric cubic Bézier curve $\mathbf{P}(t)$, because our robot is constrained to the given trajectory. For these reasons, to obtain a sequence of points on the curve and to simulate different behaviours of a real camera device, we have to elaborate the interactor positions $\mathbf{p}_i^H$. By a mathematical processing, described in the next section, and a 3D animation software visualization, we obtain the new positions of the virtual camera that we can export frame by frame to the

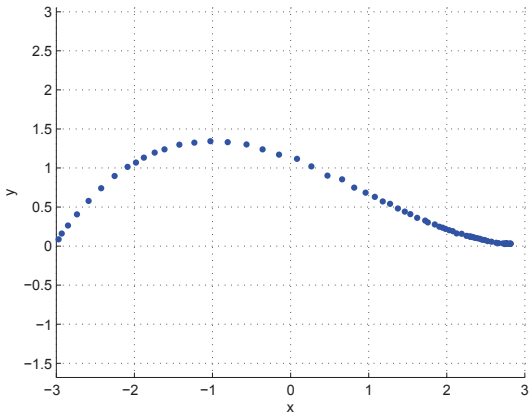


Figure 4.4: Matlab's visualisation of $\mathbf{p}_i^H$.

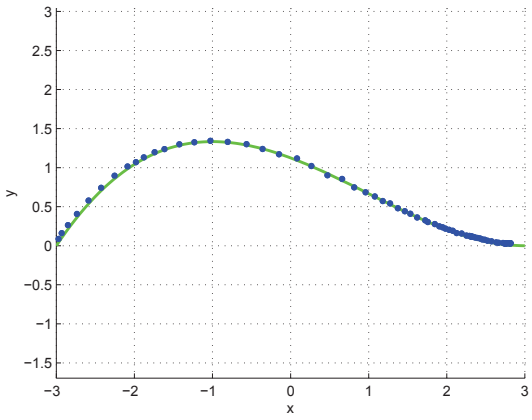


Figure 4.5: A rational parametric cubic Bézier curve parameterized by using the haptic system. In blue the interactor positions $\mathbf{p}_i^H$.

motion control system (Figure 4.17).

4.2.3 Motion control software

Using the imported data the motion control software allows to calibrate the motion control robot, to control the camera settings and, finally, to execute the sequence, as shown in Figure 4.7. It is based on an Arduino system (Figure 4.8), an open-source platform that allows, by its microcontroller, to send electronic signals to the robot and move it.

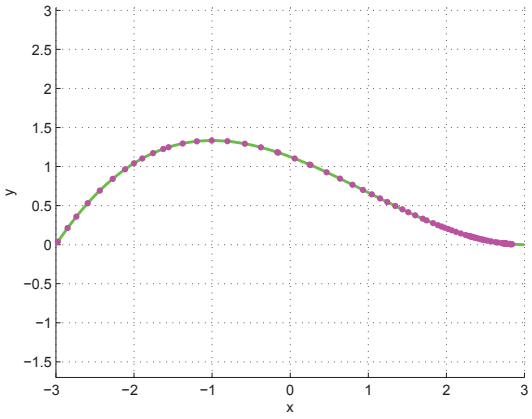


Figure 4.6: "Intermediate" curve parametrisation corresponding to the Ease Curve $S^B(f)$ in Figure 4.16.

4.2.4 Motion control robot and digital camera

A motion control robot, on which is positioned a camera and which is able to move on one translation axis and two rotation axes, is used to record the video. It is affordable for any-sized budget production, handles a 2Kg camera, and has at least 1/10th of a millimeter precision for positioning and 1/10th of a degree precision for rotating. See Figure 4.9 for an illustration.

4.3 Mathematical processing

In this section we describe in detail the mathematical processing for managing the camera movement summarized in subsection 4.2.2. In particular, in subsection 4.3.1, we describe the robot's trajectory and in subsection 4.3.2 the camera's rotations. To manage the speed of our robot, we use the concept of an Ease Curve, which represents arc length over time-frame. In the first step (*Projection: "haptic" curve parametrisation*) we project the motion capture sequence $\mathbf{p}_i^H$ on the rational parametric cubic Bézier curve. In the second step (*Fitting: "ideal" Ease Curve*) we determine an "ideal" speed that maintains the haptic speed, but is less noisy, and on this basis we calculate the ideal curve parametrisation (*"Ideal" curve parametrisation*). In the fourth step (*Blending: "intermediate" Ease Curves*) we calculate different Ease Curves between the "haptic" and the "ideal" one. Finally, we determine the corresponding curve parametrisations (*"Intermediate" curve parametrisations*). Now, we describe in detail every step.

1. *Projection: "haptic" curve parametrisation*
   The sequence of points $\mathbf{p}_i^H(x_i, y_i)$ that we obtain from the haptic system is not exactly on the rational parametric cubic Bézier curve $\mathbf{P}(t)$, as we can see in Figure 4.10. To allow the robot to move along the given trajectory, we

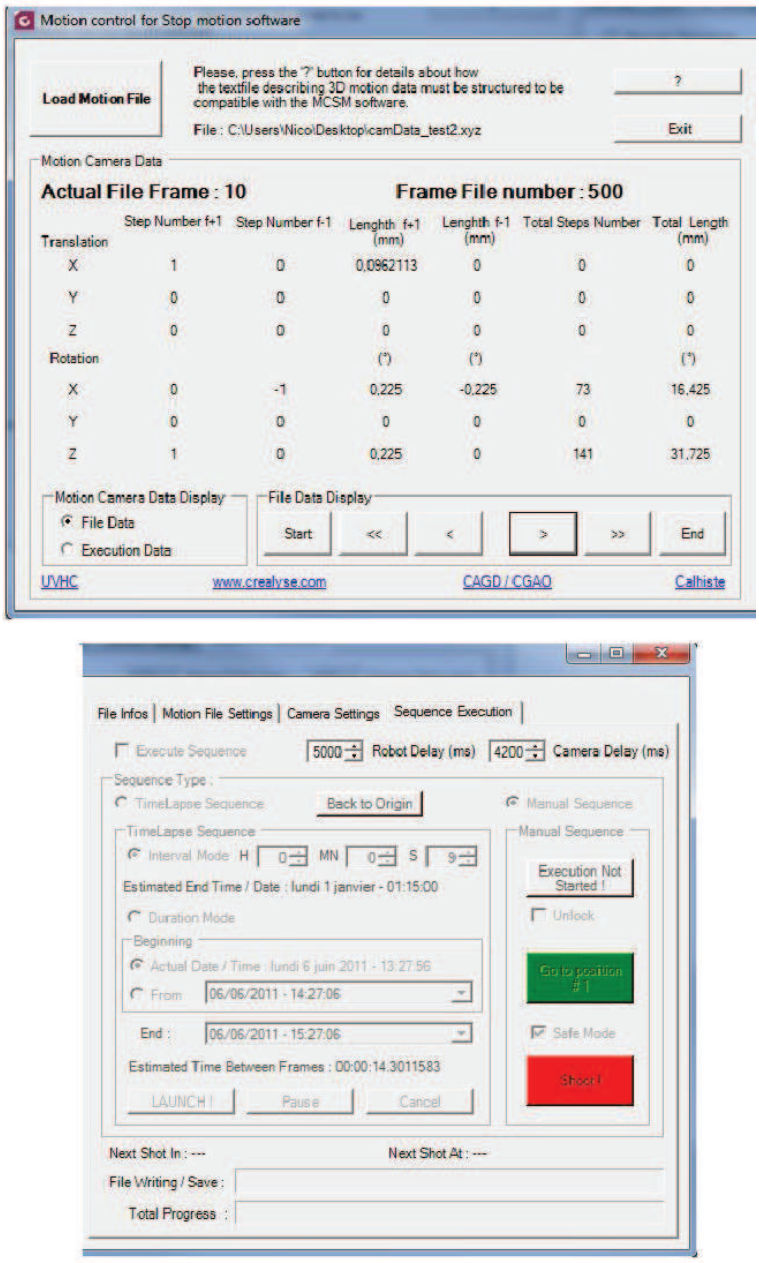


Figure 4.7: Robot and camera software interface.

have to project the sequence on the curve $\mathbf{P}(t)$, as shown in Figure 4.11. We compute the length l of $\mathbf{P}(t)$, where $t \in [0,1]$, by approximating the integral using a Gaussian quadrature or a Gaussian adaptive method, as described in [Saini 2010]. We then calculate, for $i = 1, \ldots, n-1$, $sp_i^H = ||\mathbf{p}_i^H - \mathbf{p}_{i-1}^H||$. We need curve parameters $\widetilde{t}_i \in [0,1]$ with $i = 0, \ldots, n-1$ to calculate the coordinates of the corresponding curve points $\widetilde{\mathbf{p}}_i^H$. A reasonable choice for the

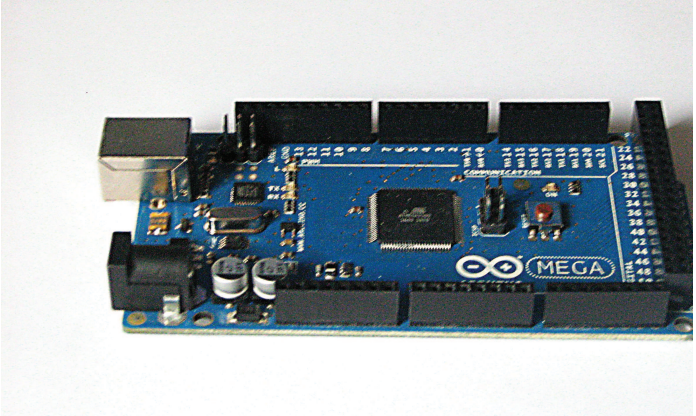


Figure 4.8: Arduino Mega 2560.

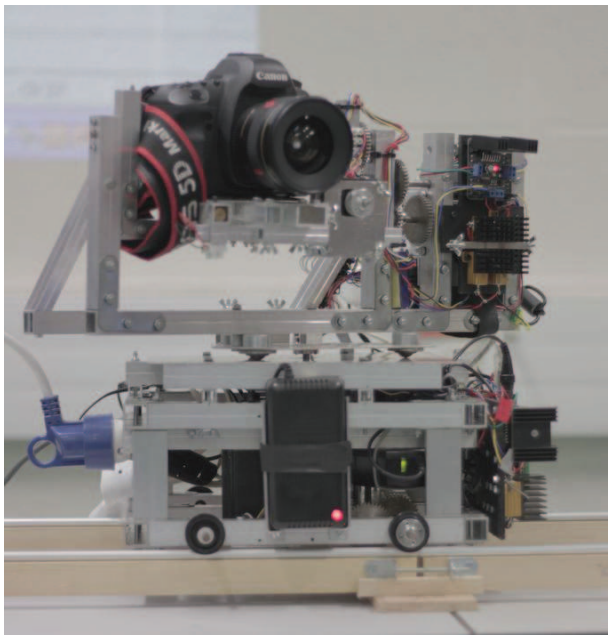


Figure 4.9: Robot with 3 axes of freedom and its camera (Canon EOS 5D mark ii).

initial parameter is $\widetilde{t}_0 = \frac{||\mathbf{p}_0^H - \mathbf{P}(0)||}{l}$, because it represents the fraction of arc length at which the first point should be located. We thus want to solve the following equation:

$$F(\widetilde{t}_i) = \int_{\widetilde{t}_{i-1}}^{\widetilde{t}_i} \left|\left| \frac{d\mathbf{P}}{dt} \right|\right| dt - sp_i^H = 0,$$

with $i = 1, \ldots, n-1$. We can solve it by using Newton's method :

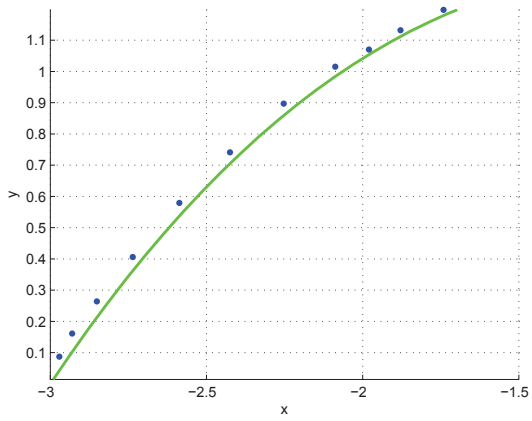


Figure 4.10: Visualization of the points $\mathbf{p}_i^H$ (blue) and the rational parametric cubic Bézier curve $\mathbf{P}(t)$ (green).

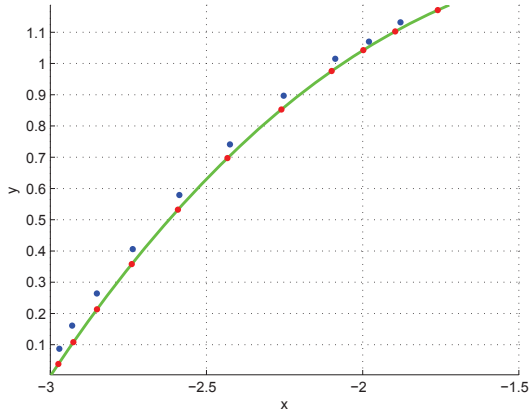


Figure 4.11: Visualization of the points $\widetilde{\mathbf{p}}_i^H$ (red), the points $\mathbf{p}_i^H$ (blue) and the rational parametric cubic Bézier curve $\mathbf{P}(t)$ (green).

$$\widetilde{t}_i^k = \widetilde{t}_i^{k-1} - \frac{F(\widetilde{t}_i^{k-1})}{F'(\widetilde{t}_i^{k-1})}, \quad k = 1, 2, \dots$$

where $\widetilde{t}_i^0 = \widetilde{t}_{i-1} + \frac{sp_i^H}{l}$ and $F(\widetilde{t}_i^{k-1})$ is approximated using standard numerical integrators and $F'(\widetilde{t}_i^{k-1})$ is straightforward because we have a formula for the rational parametric cubic Bézier curve $\mathbf{P}(t)$ and we can compute $d\mathbf{P}/dt$ from it. When F is sufficiently near zero or a maximum number k of iterates has been computed, $\widetilde{t}_i^k$ is accepted as $\widetilde{t}_i$ and the iteration is repeated for $i = 1, \dots, n-1$. If the calculated $\widetilde{t}_i > 1$ we set $\widetilde{t}_i = 1$. Finally, the parametric curve $\mathbf{P}(\widetilde{t}_i)$ is recalculated with the new parameter values $\widetilde{t}_i$, as represented in Figure 4.12. This procedure is described in detail in section 3.2.2. This allowed us to

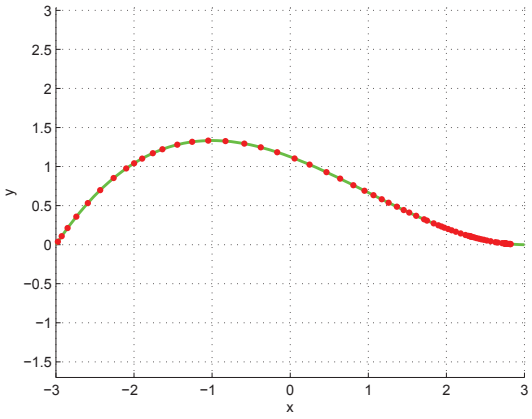


Figure 4.12: Projection of the haptic positions $\mathbf{p}_i^H$ on the curve $\mathbf{P}(t)$ resulting in the red points $\widetilde{\mathbf{p}}_i^H$.

publish the article [Saini 2010].

2. *Fitting: "ideal" Ease Curve*
   In Figure 4.13 we visualize the Ease Curve $S^H(f)$, which represents arc length s over time-frame f. We calculate

$$s_i^H = \begin{cases} s_{i-1}^H + sp_i^H, & \text{if } x_i > x_{i-1} \\ s_{i-1}^H - sp_i^H, & \text{if } x_i < x_{i-1} \end{cases}$$

   with $s_0^H = 0$ and x_i the abscissa of $\mathbf{p}_i^H$ with $x_i \neq x_{i-1}\ \forall i$. On the $y-$axis we have the $s_i^H = S^H(f_i)$ and on the $x-$axis equally spaced frame parameter values f_i with $f_i \in [0, 24]$.

   By analyzing the Ease Curve, in Figure 4.13 we observe that the speed at which the curve is traversed and thus the resulting movement obtained by the haptic system is too noisy for our purpose. So, we want to find a new parametrisation for our rational parametric cubic Bézier curve that is less noisy but still respects the haptic movement. By using the least squares method we fit a quartic polynomial to the discrete data of the Ease Curve $S^H(f)$, as represented in Figure 4.14. Thus, we obtain an "ideal" speed $S^I(f_i) = s_i^I$.

3. *"Ideal" curve parametrisation*
   By applying Newton's method as described above and by replacing sp_i^H by $sp_i^I = |s_i^I - s_{i-1}^I|$ for $i = 1, \ldots, n-1$, we find the new sequence of parameter values $\widetilde{t}_i^I$. Thus, we can calculate the corresponding points along the curve, to obtain the "ideal" curve parametrisation $\mathbf{P}(\widetilde{t}_i^I)$ (Figure 4.15). It respects the haptic movement but eliminates the superfluous noise.

4. *Blending: "intermediate" Ease Curves*
   We want to propose several parametrisations such that an animator can choose

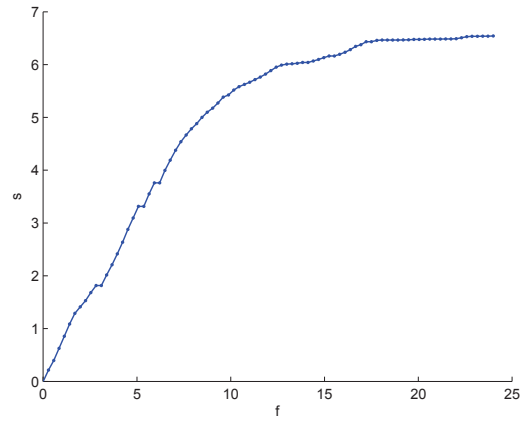


Figure 4.13: $S^H(f)$ Ease Curve of $\mathbf{p}_i^H$.

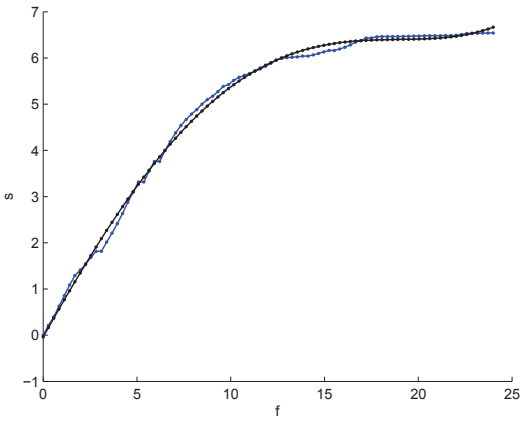


Figure 4.14: Least squares Ease Curve $S^I(f)$ (black) of the discrete data of the Ease Curve $S^H(f)$ (blue) of Figure 4.13.

the one that best fits his/her design intention. We thus determine an intermediate Ease Curve between the "ideal" $S^I(f)$ and the haptic one $S^H(f)$. We define for $i = 0, \ldots, n-1$

$$\lambda_i := \alpha_i \cdot \left(\frac{d_i}{||\mathbf{d}||} \right) \tag{4.1}$$

where $d_i = |s_i^H - s_i^I|$, $\mathbf{d} = (d_0, d_1, d_2, \ldots, d_{n-1})$ and with $\alpha_i \in [0, \frac{||\mathbf{d}||}{d_i}]$. We then can calculate an "intermediate" Ease Curve $S^B(f)$ such that $s_i^B = S^B(f_i)$ by blending:

$$s_i^B = (1 - \lambda_i) s_i^H + \lambda_i s_i^I.$$

If $\alpha_i = 0 \ \forall i$ we have the "haptic" Ease Curve $S^H(f)$ and if $\alpha_i = \frac{||\mathbf{d}||}{d_i} \ \forall i$ we obtain the "ideal" one $S^I(f)$. The definition (4.1) assures that we respect the

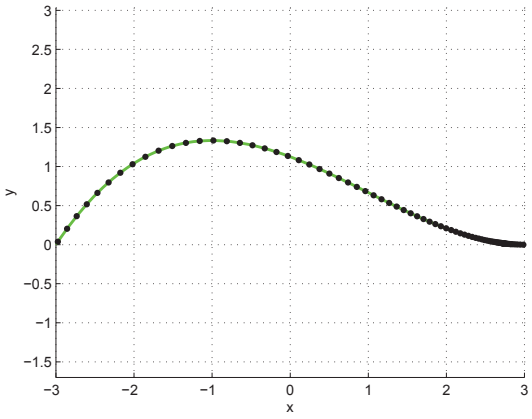


Figure 4.15: "Ideal" curve parametrisation corresponding to the Ease Curve $S^I(f)$ of Figure 4.14.

haptic speed $S^H(f)$ because if s_i^H is close to s_i^I, the influence of s_i^I on s_i^B is neglectable. The curve $S^I(f)$ modifies $S^H(f)$ only if s_i^H is far from s_i^I. In figure 4.16 we can see the "intermediate" Ease Curve $S^B(f)$ with $\alpha_i = 0.5\ \forall i$.

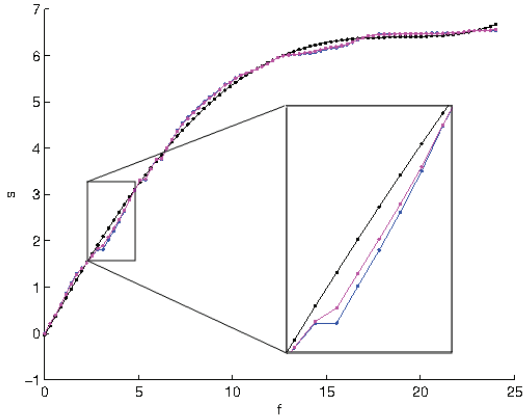


Figure 4.16: In magenta the "intermediate" Ease Curve $S^B(f)$ with $\alpha_i = 0.5\ \forall i$ between the "haptic" Ease Curve $S^H(f)$ (blue) and the "ideal" Ease Curve $S^I(f)$ (black).

5. *"Intermediate" curve parametrisations*
   By applying Newton's method to the "blended" Ease Curve $S^B(f)$ and by using $sp_i^B = |s_i^B - s_{i-1}^B|$ for $i = 1, \ldots, n-1$, we find the new sequence of parameter values $\widetilde{t}_i^B$. We can now calculate the corresponding points along the curve, to obtain the "intermediate" curve parametrisation $\mathbf{P}(\widetilde{t}_i^B)$, shown

in Figure 4.17. It should be noted, that the use of the α_i parameter also allows

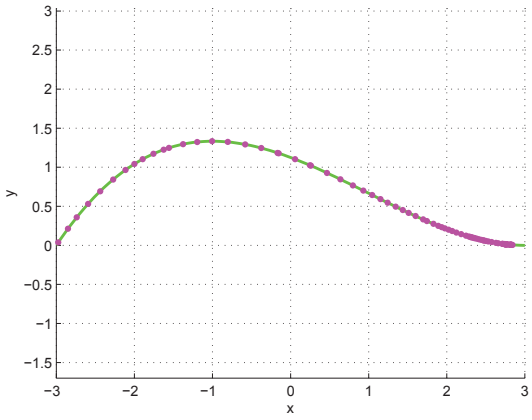


Figure 4.17: "Intermediate" curve parametrisation corresponding to the Ease Curve $S^B(f)$ in Figure 4.16.

to modify only a zone of the speed.

This processing, implemented in $Matlab^{®}$, allows a great flexibility in controlling the speed along the curve, without modifying the space trajectory. Since our system is able to move on one translation axis and two rotation axes, we need to control two particular curves: a straight line and a semicircle and the corresponding procedure is described in the following sections.

4.3.1 Translation

So far our robot is able to move along a straight line. We thus want to control this linear translational robot movement.

To have an intuitive correspondence with the real robot trajectory, we set the Bézier cubic curve control points as:

$$cp_1 = (-3, 0), \quad cp_2 = (-1, 0), \quad cp_3 = (1, 0), \quad cp_4 = (3, 0)$$

and all the weights equal to 1, as shown in Figure 4.18. We record the haptic positions $\mathbf{p}_i^{H_t}$ concerning the translation and we export them in $Matlab^{®}$. Now, we follow the procedure described in the previous section to obtain different speed choices for our robot movement. We import these new curve parametrisations (haptic, ideal, intermediate) in an external script in 3D Studio Max. We visualize the trajectory and we simulate the camera's movement. An example of haptic curve and its corresponding intermediate parametrisation with $\alpha_i = \frac{||\mathbf{d}||}{d_i} \cdot 0.9 \ \forall i$ are visualized in Figure 4.19.

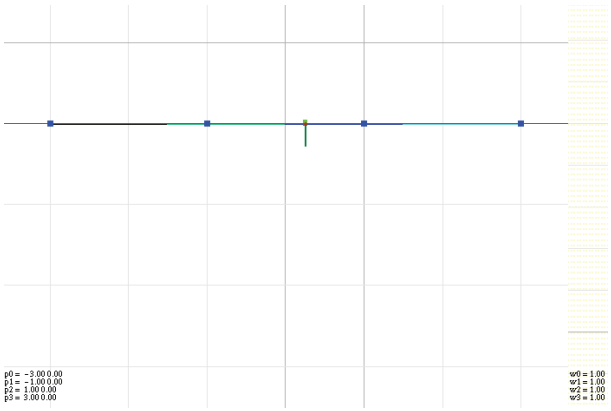


Figure 4.18: Robot trajectory in the graphical interface.

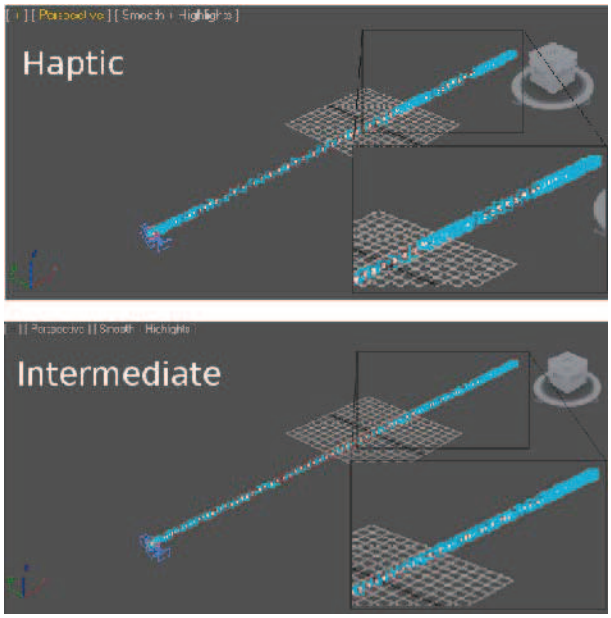


Figure 4.19: Projected haptic trajectory parametrisation $\widetilde{\mathbf{p}}_i^{H_t}$ and Intermediate trajectory parametrisation $\widetilde{\mathbf{p}}_i^{B_t}$ with $\alpha_i = \frac{||\mathbf{d}||}{d_i} \cdot 0.9 \; \forall i$ visualized in 3D Studio Max.

4.3.2 Rotation

The camera on our robot can move on two rotation axes. Thus, we want to control the rotation in the $yz-$plane and in the $xy-$plane (Figure 4.20). For both planes, the camera can rotate by an angle $\omega \in [0, \pi]$. We represent and visualize this angle by a semicircle (Figure 4.21). The conditions needed to reproduce a circle with a rational parametric cubic Bézier curve are given in [Piegl 1995]. In particular, the degree three case has the advantage that it can represent an arc of 180 degrees. We set the Bézier cubic curve control points as:

$$cp_1 = (1, 0), \quad cp_2 = (1, 2), \quad cp_3 = (-1, 2), \quad cp_4 = (-1, 0)$$

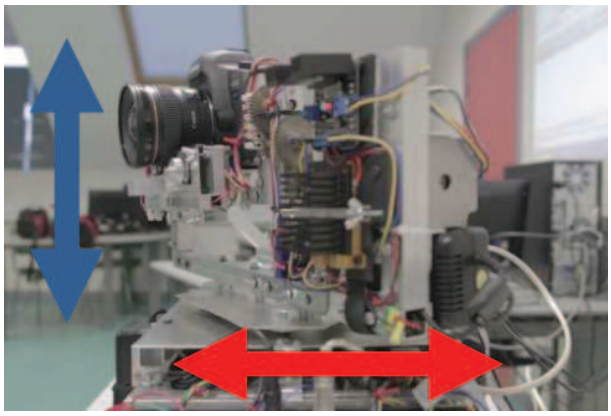


Figure 4.20: Visualization of the camera rotation in the yz (vertical panoramic) (blue arrow) and xy (horizontal panoramic) plane (red arrow).

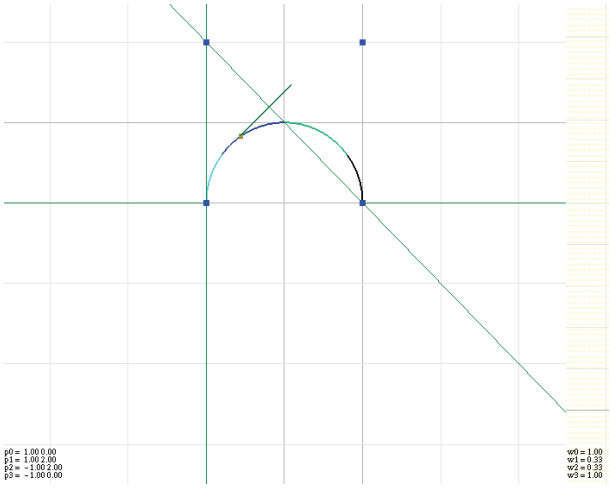


Figure 4.21: Semicircle in the graphical interface.

and the weights as:

$$w_1 = 1, \quad w_2 = \frac{1}{3}, \quad w_3 = \frac{1}{3}, \quad w_4 = 1.$$

The same procedure is applied to the two rotations. We determine two haptic sequences of points $\widetilde{\mathbf{p}}_i^{H_{r1}}$ and $\widetilde{\mathbf{p}}_i^{H_{r2}}$. We export them into Matlab to obtain our speed elaboration. We visualize an example of the camera's rotation in the $yz-$plane in Figure 4.22 ("haptic", "ideal" and "intermediate" parametrisation).

4.4 System assessment

The evaluation of the system is very important to see if it does what it is supposed to do and if it produces visually convincing camera movements, as well as to identify problems and possible improvements. So, it is essential to get feedback from users

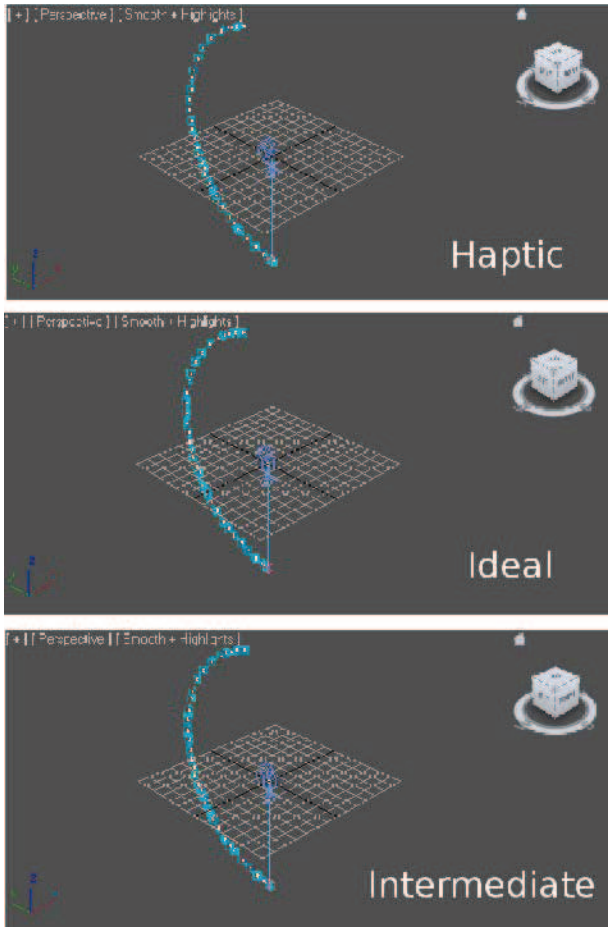


Figure 4.22: Haptic ($\widetilde{\mathbf{p}}_i^{Hr2}$), ideal ($\widetilde{\mathbf{p}}_i^{Ir2}$) and intermediate ($\widetilde{\mathbf{p}}_i^{Br2}$) $yz-$rotation parametrisation visualized in 3D Studio Max.

of the system.
As part of a project with the Art department of the University of Valenciennes, we asked a class of 14 students of the "Arts plastiques et Création numérique" Master to test our system. In particular they tested the different steps, from motion capture through the haptic device to importing the data in the motion control software. They didn't directly move the robot and the camera, but they visualized in 3D Studio Max the different movements ("haptic", "ideal" and "intermediate"). The recorded movements are currently used in the realisation of a stop motion video clip. Before having the system tested, we gave a general presentation of the whole system, to explain why we decided to create it and how it works. After that, every student had the possibility to use the haptic interface to determine both the robot's translation and rotations and visualize his/her results in 3D Studio Max.
We gathered as much information as possible about the system by having them fill out a questionnaire. We collected information about efficiency, ease of use, appropriateness of the whole system for translation and rotation movements. In particular,

four principal parts composed the questionnaire: "system's presentation", "motion capture movement (for translation and rotation)", "whole system" and "general questions". In Figure 4.23 we summarize the results of the questionnaire. In the

QUESTIONS \ LEVEL		1 (-)	2	3	4	5 (+)
System's presentation (*)		-	-	-	7	7
Motion capture movement	Translation (**)	-	-	3	8	3
	Rotation (**)	-	-	8	4	2
Whole system (***)		-	-	2	10	2
General questions (***)		-	-	-	7	7

(*) CLEAR
(**)INTUITIVE
(***)EFFICIENT

Figure 4.23: A summary of the questionnaire filled out by a class of 14 students of the "Arts plastiques et Création numérique" Master of the University of Valenciennes.

first part we asked if the system presentation was clear and if they needed more details. For all students the presentation was clear (for 7 students it was very clear and for the others clear), even if mathematics is not their domain. They only remarked that it is necessary to say that translation and rotation are separated.
In the second part we asked if both motion capture trajectory and rotation are intuitive (very intuitive, intuitive, moderately intuitive, not very intuitive, not intuitive). For all students the robot's translation is easy to do (for 3 students it is very intuitive, for 8 students intuitive and for 3 students moderately intuitive). They only suggested to improve the graphical interface visualization by adding a camera icon or a hand drawing, to be more immersed in the system. Regarding the camera rotations, they were found to be a bit less intuitive (for 8 students it is moderately intuitive, for 4 students it is intuitive and for 2 students very intuitive). The students' feedback is that rotating camera movements are not so simple to execute correctly. Anyway, the rotation in the $yz-$plane (up/down camera rotation) is better than the rotation in the $xy-$plane (left/right camera rotation), because they have the sensation of holding in their hands a camera tripod head rather than the camera itself. As a consequence, the resulting movement is inverted, because if you lift up the tripod, the "head" on which the camera is fixed goes down and vice versa. Indeed, the mathematical processing, in both cases, is very appropriate

(for 7 students the process is very efficient and for the others it is efficient). They noticed that the resulting movements match the realistic haptic ones.
The third part is about the evaluation of the whole system. We asked if the system is efficient and what are the weaknesses of the system and what they would like to improve. The outcome of the evaluation shows that the whole system is efficient. For 10 students it is efficient, for 2 moderately efficient and for the remaining 2 very efficient. It works well and allows to easily reproduce realistic camera movements. The result doesn't look like a computer generated move, because the haptic device allows to add noise in a natural manner. On the other hand, half of the students performed the desired movement with some difficulties. The main reason for that is that they lack practical experience with the haptic system. They said that with more practice and several tests on the use of the haptic device, they would be able to improve their results. The main technical and ergonomic problem is the scale representation of the trajectory (the real is 6 meters and the virtual 6 centimeters). Thus, controlling and managing the speed along the curve is complicated. They suggested to amplify the translation length and the rotation radius. They remarked that it's not too much of a problem to control translation and rotation separately, even though it is not very intuitive. They also noted that there are too many steps before visualizing the movement in 3D Studio Max.
Another question was about the resulting movements from the mathematical processing. They found this process efficient (for 7 students it is very efficient and for the remaining 7 efficient). They remarked that the haptic movement needs to be modified, but that the elaborated movements respect what they wanted to represent. In particular, they said that the most correct movement is the one created with $\alpha_i = \frac{||\mathbf{d}||}{d_i} \cdot 0.9 \ \forall i$, that is a movement near to the ideal one, as shown in Figure 4.19. Finally, we asked what they liked or not of the whole system and to add some remarks. Overall, all students found the new motion control system intuitive and innovative. It allows to simulate a realistic camera movement as in the world of cinematography and, with respect to existing 3D animation software (cf. section 3.1) it provides the user in an intuitive way with the important features of position-speed separation and addition of realistic noise. Moreover, it suits a wide range of users, because it is simple to use and has an affordable price.

Chapter 5

A new class of curves: Algebraic-Trigonometric Pythagorean-Hodograph

The purpose of this chapter is to investigate on the existence of planar, algebraic–trigonometric Pythagorean Hodograph curves defined over a mixed algebraic–trigonometric space possessing a normalized B–basis. These curves are shown to be the analogue of the polynomial Pythagorean–Hodograph (PH) quintics in the considered non-polynomial space - due to the fact that they enjoy an analogous property on the hodograph - and are thus called Algebraic-Trigonometric Pythagorean-Hodograph (ATPH) curves.
Their planar polynomial counterpart was originally introduced by Farouki and Sakkalis in 1990 in the article [Farouki 1990]. These curves, commonly called PH curves since the Euclidean norm of their hodograph is also a polynomial, have the useful properties of admitting a closed–form polynomial representation of their arc–length as well as exact rational parameterization of their offset curves. Since their introduction they have widely been investigated mainly for solving practical problems from applications that particularly benefit from the PH curves' particular properties, such as CNC machining. Rational and spatial counterparts of polynomial PH curves have as well been proposed, but we are not aware of any attempt of defining Pythagorean Hodograph curves over a mixed algebraic–trigonometric space. This kind of spaces, deeply investigated by J.M. Carnicer, E. Mainar, J.M. Peña and J. Sánchez-Reyes [Mainar 2001, Carnicer 2004, Mainar 2007, Mainar 2010] in the last decade, offer the advantage of providing an exact description of both linear segments and circular arcs. This additional feature makes them even more attractive in applications like CNC machining since the path of the cutting tool is usually described in terms of line segments and circular arcs. Moreover, these properties can help us to get a smooth trajectory curve that takes into account the constraints of a real camera move, as described in chapter 4. Thus, we believe the extension of the PH property from parametric polynomial curves to parametric curves defined over a mixed linear–trigonometric space, beneficial to the development of more and more advanced algorithms for the guide of CNC milling machines and to help animators realize a realistic camera movement. For this purpose, in this chapter we revisit several important publications such as [Farouki 1995, Moon 2001, Farouki 1997, Habib 2008], dealing with the solution of different Hermite interpolation problems by polynomial PH curves, and generalize them to our non-polynomial context. In particular,

while [Farouki 1995] and [Moon 2001] solve the C^1 Hermite interpolation problem by polynomial PH quintics and analyze the shape of the obtained four solutions, [Farouki 1997] and [Habib 2008] consider the problem of joining basic elements such as line segments and circles by G^2–continuous polynomial PH quintics of monotone curvature, also referred to as PH spirals. More precisely, the remainder of the chapter is organized as follows.

In Section 5.1 we first recall general results on normalized B–bases of spaces of real functions and then specialize them for some pure trigonometric and mixed algebraic–trigonometric spaces. Section 5.2 considers generalized Bézier curves defined over a five–dimensional algebraic–trigonometric space, shows their reproduction capabilities of well–known trigonometric curves and proposes a fast subdivision–based evaluation algorithm for them. Section 5.3 is dedicated to the real definition and construction of algebraic–trigonometric PH curves generalizing the well-known polynomial PH quintics. These curves have the property that the Euclidean norm of their hodograph is a trigonometric function, their arc-length is a mixed algebraic–trigonometric function and their unit tangent, unit normal as well as signed curvature are described by rational trigonometric functions. Then, in Section 5.4 the class of ATPH curves in the real representation is employed to solve the C^1 Hermite interpolation problem. To facilitate the solution of the above Hermite problems a complex representation of the novel class of algebraic–trigonometric PH curves is introduced in section 5.6, and a general constructive approach exploiting the key properties in the new representation is proposed. For completeness, in section 5.7 we reformulate the Hermite problem in this concise complex notation and the obtained four solutions are analyzed and the best one is identified. Advantages of ATPH interpolants over the classical polynomial PH solutions are highlighted with the help of illustrative examples. Section 5.7 deals with the construction of ATPH curves with monotone curvature, also called ATPH spirals, for joining G^2-continuously a line segment and a circle as well as two external circles. In both cases the obtained solutions turn out to be more flexible than their polynomial PH counterparts, thanks to the additional shape parameter offered by the ATPH representation. The work presented in this chapter is submitted for publication.

5.1 Normalized B-bases for pure trigonometric and mixed algebraic–trigonometric spaces

5.1.1 General results on normalized B-bases of spaces of real functions

Let $\Phi(I)$ be a finite-dimensional vector space of real functions defined on $I \subseteq \mathbb{R}$ and let $\{\varphi_0, ..., \varphi_n\}$ be a basis of $\Phi(I)$. Given a sequence of points $\mathbf{p}_i$, $i = 0, ..., n$, we define a parametric curve over Φ as $\mathbf{x}(t) = \sum_{i=0}^{n} \mathbf{p}_i \, \varphi_i(t)$, $t \in I$, and we call the polyline connecting $\mathbf{p}_0, ..., \mathbf{p}_n$ the control polygon of $\mathbf{x}(t)$.

Definition 1. *If the system of functions* $(\varphi_0, ..., \varphi_n)$ *satisfies the condition of par-*

tition of unity on I, that is $\sum_{i=0}^{n}\varphi_i(t)=1$ for all $t\in I$, then $(\varphi_0,...,\varphi_n)$ is termed normalized *on I.*

If $(\varphi_0,...,\varphi_n)$ is a normalized system of non-negative functions, then the curve $\mathbf{x}(t)$ possesses the convex hull property and is affine invariant [Carnicer 1994]. These geometric properties correspond to some properties concerning the collocation matrices of the system of functions $(\varphi_0,...,\varphi_n)$. Taking a system of functions $(\varphi_0,...,\varphi_n)$ defined on $I\subseteq\mathbb{R}$, the collocation matrix of $(\varphi_0,...,\varphi_n)$ at $t_0<...<t_m$ in I is given by

$$M\begin{pmatrix}\varphi_0,...,\varphi_n\\ t_0,...,t_m\end{pmatrix}:=(\varphi_j(t_i))_{i=0,...,m;\,j=0,...,n}.$$

Clearly $(\varphi_0,...,\varphi_n)$ is a normalized system of non-negative functions if and only if all its collocation matrices are stochastic (i.e. non-negative and such that the sum of each row is 1).

Definition 2. *If all the collocation matrices of the system $(\varphi_0,...,\varphi_n)$ are totally positive (which means that all their minors are non-negative), then $(\varphi_0,...,\varphi_n)$ is termed* totally positive *(TP).*

If $(\varphi_0,...,\varphi_n)$ is a normalized TP system (NTP system for short), then the curve $\mathbf{x}(t)$ inherits many shape properties of the control polygon, such as the endpoint interpolation property and the variation-diminishing property (see [Carnicer 1994]). In [Carnicer 1994] it was also proved that a space Φ with a normalized TP basis always has a *unique* normalized B-basis $(B_0,...,B_n)$, which is the basis with optimal shape-preserving properties. More precisely, we recall

Definition 3. *A TP basis $(B_0,...,B_n)$ of a space of functions Φ is called a* B-basis *if any other TP basis $(\varphi_0,...,\varphi_n)$ of Φ is of the form $(\varphi_0,...,\varphi_n)=(B_0,...,B_n)K$, where the matrix K of change of basis is a non-singular TP matrix.*

According to the results in [Carnicer 1994], a totally positive basis $\{\varphi_0,...,\varphi_n\}$ defined over an interval I is a B-basis if and only if the following condition holds:

$$\inf\left\{\frac{\varphi_i(t)}{\varphi_j(t)}\ :\ t\in I,\ \varphi_j(t)\neq 0\right\}=0\qquad\forall i\neq j.$$

For instance, for the space of polynomials of degree less than or equal to n on a compact interval of the real line, the Bernstein basis is the normalized B-basis, and, for the corresponding space of polynomial splines, the B-spline basis is the normalized B-basis.

Let us proceed by recalling that a space of functions Φ defined on I is $C^j(I)$ if every $\varphi\in\Phi$ is j times continuously differentiable. This notion allows us to formulate the following two definitions.

Definition 4. *A space of functions Φ defined on I is termed an* extended Chebyshev space *if it is an $(n+1)$-dimensional subspace of $C^n(I)$ such that each of its non-zero functions has at most n zeroes, counting multiplicities.*

Definition 5. *A basis* $(\varphi_0, ..., \varphi_n)$ *of an* $(n+1)$*-dimensional space* Φ *in* $C^n(I)$ *is termed* canonical *at* $t \in I$ *if the Wronskian matrix* $W(\varphi_0, ..., \varphi_n)(t) := (\varphi_j^{(i)}(t))_{0 \le i,j \le n}$ *is lower triangular with non-zero diagonal entries.*

The following result, proved in [Carnicer 2004], provides an equivalence between the existence of a normalized B-basis for the extended Chebyshev space Φ and the property of its space of derivatives $\Phi' := \{\varphi' \,|\, \varphi \in \Phi\}$ of being an extended Chebyshev space.

Proposition 1. *Let* Φ *be an* $(n+1)$*-dimensional subspace of* $C^n(I)$ *such that* $1 \in \Phi$*. Then* Φ *is an extended Chebyshev space with a normalized B-basis on* I *if and only if the space* Φ' *is an extended Chebyshev space.*

In the next subsection we use these notions to present normalized B-bases for certain spaces of trigonometric and mixed algebraic–trigonometric functions.

5.1.2 Normalized B-bases of spaces of trigonometric and mixed algebraic–trigonometric functions

Let $t \in I_\alpha$ with $I_\alpha = [0, \alpha]$ and $0 < \alpha < \pi$. For an arbitrary $m \in \mathbb{N}$ we consider the space of order m trigonometric polynomials

$$\tilde{U}_{2m} = \langle 1, \{\sin(\ell t), \cos(\ell t)\}_{\ell=1}^{m} \rangle,$$

and we recall the formula proposed in [Sánchez-Reyes 1998] to define the normalized B-basis for such a space. After introducing the change of variable

$$\tau(t) := \frac{1}{2} + \frac{\tan\left(\frac{t}{2} - \frac{\alpha}{4}\right)}{2\tan\left(\frac{\alpha}{4}\right)}, \qquad \tau \in [0,1], \quad t \in I_\alpha,$$

and the $2m+1$ Bernstein polynomials

$$b_i^{2m}(\tau) = \binom{2m}{i} \tau^i (1-\tau)^{2m-i}, \qquad i = 0, ..., 2m,$$

we define the normalized B-basis for $\tilde{U}_{2m}$ in terms of the Bernstein basis $b_i^{2m}(\tau)$ as

$$\tilde{B}_i^{2m}(t) = \left(\frac{\cos\left(\frac{t}{2} - \frac{\alpha}{4}\right)}{\cos\left(\frac{\alpha}{4}\right)} \right)^{2m} \mu_i \, b_i^{2m}(\tau), \qquad i = 0, ..., 2m,$$

where

$$\mu_i \equiv \mu_{2m-i} := \binom{2m}{i}^{-1} \sum_{r=0}^{(i/2)} \binom{m}{i-r} \binom{i-r}{r} \left(2\cos\left(\frac{\alpha}{2}\right)\right)^{i-2r}, \qquad i = 0, ..., m,$$

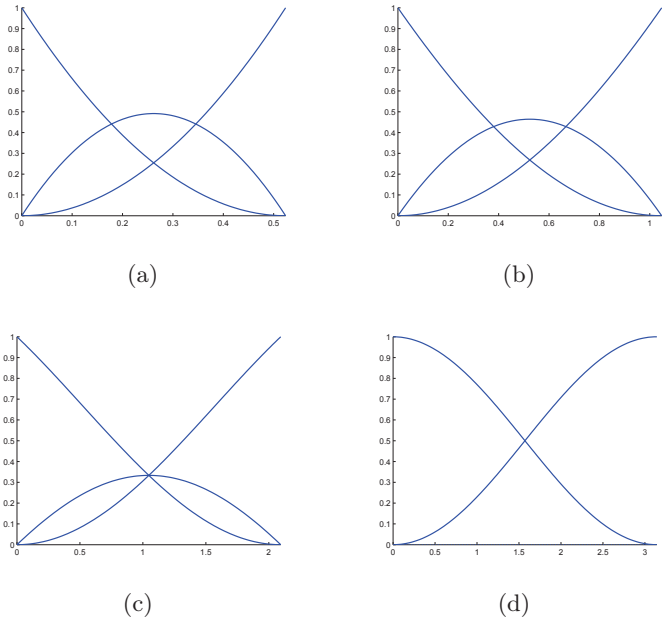
and $(i/2) := \max\{j \in \mathbb{Z} : j \le \frac{i}{2}\}$ denotes the highest integer value lower than or equal to $i/2$. Note that, being $0 < \alpha < \pi$, then for any choice of α in that range we have that μ_i are strictly positive values. Moreover, as pointed out in [Sánchez-Reyes 1998], the basis functions $\tilde{B}_i^{2m}$, $i = 0, ..., 2m$ possess the following properties:

(i) Symmetry: $\tilde{B}_i^{2m}(t) = \tilde{B}_{2m-i}^{2m}(\alpha - t),\ t \in I_\alpha$;

(ii) Positivity: $\tilde{B}_i^{2m}(t) \geq 0,\ t \in I_\alpha$;

(iii) Partition of unity: $\sum_{i=0}^{2m} \tilde{B}_i^{2m}(t) = 1,\ t \in I_\alpha$;

(iv) Recursion: $\tilde{B}_i^{2m} = \tilde{B}_0^2\, \tilde{B}_i^{2(m-1)} + \tilde{B}_1^2\, \tilde{B}_{i-1}^{2(m-1)} + \tilde{B}_2^2\, \tilde{B}_{i-2}^{2(m-1)},\quad m \geq 2\,.$



(a) (b) (c) (d)

Figure 5.1: Basis functions $\tilde{B}_i^2(t)$, $i = 0, 1, 2$ for $t \in I_\alpha$ and $\alpha = \frac{\pi}{6}$ (a), $\alpha = \frac{\pi}{3}$ (b), $\alpha = \frac{2}{3}\pi$ (c), $\alpha \to \pi$ (d).

The functions $\tilde{B}_i^{2m}(t)$, $i = 0, ..., 2m$ can be regarded as the true equivalent of the Bernstein polynomials in $\tilde{U}_{2m}$, and they tend to the ordinary Bernstein polynomials of degree $2m$ whenever $\alpha \to 0$. On the other hand, note that the parameter α, if progressively increased in its range of definition, offers an interesting tension-like effect (see Figure 5.1 (a),(b),(c)) and, when approaching π, only the first and the last functions of the normalized B-basis $\tilde{B}_i^{2m}$, $i = 0, ..., 2m$ are non-vanishing. Therefore the associated curve $\mathbf{x}(t) = \sum_{i=0}^{2m} \mathbf{p}_i \tilde{B}_i^{2m}(t)$, $t \in I_\alpha$ degenerates to the segment $\mathbf{p}_0\mathbf{p}_{2m}$. For instance, when $\alpha \to \pi$, the normalized B-basis $\tilde{B}_i^2$, $i = 0, 1, 2$ assumes the following form

$$\lim_{\alpha\to\pi} \tilde{B}_0^2(t) = \frac{1}{2}(1 + \cos(t)), \qquad \lim_{\alpha\to\pi} \tilde{B}_1^2(t) = 0, \qquad \lim_{\alpha\to\pi} \tilde{B}_2^2(t) = \frac{1}{2}(1 - \cos(t)).$$

This limit case is illustrated in Figure 5.1(d).

For later use we give the explicit expressions of the normalized B-bases for the spaces $\tilde{U}_2$, $\tilde{U}_4$ and $\tilde{U}_6$, hereinafter denoted by $\{\tilde{B}_i^2\}_{i=0,1,2}$, $\{\tilde{B}_i^4\}_{i=0,\cdots,4}$, $\{\tilde{B}_i^6\}_{i=0,\cdots,6}$, respectively:

$$
\begin{aligned}
\tilde{B}_0^2(t) &= \tfrac{\cos(\alpha-t)-1}{\cos(\alpha)-1},\\
\tilde{B}_1^2(t) &= \tfrac{\cos(\alpha)-\cos(t)-\cos(\alpha-t)+1}{\cos(\alpha)-1},\\
\tilde{B}_2^2(t) &= \tfrac{\cos(t)-1}{\cos(\alpha)-1},\\
\tilde{B}_0^4(t) &= \tfrac{(\cos(\alpha-t)-1)^2}{(\cos(\alpha)-1)^2},\\
\tilde{B}_1^4(t) &= \tfrac{2(\cos(\alpha-t)-1)(\cos(\alpha)-\cos(t)-\cos(\alpha-t)+1)}{(\cos(\alpha)-1)^2},\\
\tilde{B}_2^4(t) &= \tfrac{2(\cos(\alpha-t)-1)(\cos(t)-1)+(\cos(\alpha)-\cos(t)-\cos(\alpha-t)+1)^2}{(\cos(\alpha)-1)^2},\\
\tilde{B}_3^4(t) &= \tfrac{2(\cos(t)-1)(\cos(\alpha)-\cos(t)-\cos(\alpha-t)+1)}{(\cos(\alpha)-1)^2},\\
\tilde{B}_4^4(t) &= \tfrac{(\cos(t)-1)^2}{(\cos(\alpha)-1)^2},\\
\tilde{B}_0^6(t) &= \tfrac{(\cos(\alpha-t)-1)^3}{(\cos(\alpha)-1)^3},\\
\tilde{B}_1^6(t) &= \tfrac{-48\cos(\alpha/2)\sin(t/2)\sin(\alpha/2-t/2)^5}{(\cos(\alpha)-1)^3},\\
\tilde{B}_2^6(t) &= \tfrac{3\,(2\cos(\alpha)+3)(\cos(t)-1)(\cos(\alpha-t)-1)^2}{(\cos(\alpha)-1)^3},\\
\tilde{B}_3^6(t) &= \tfrac{-32\sin(t/2)^3\sin(\alpha/2-t/2)^3\cos(\alpha/2)(\cos(\alpha)+4)}{(\cos(\alpha)-1)^3},\\
\tilde{B}_4^6(t) &= \tfrac{3\,(2\cos(\alpha)+3)(\cos(t)-1)^2(\cos(\alpha-t)-1)}{(\cos(\alpha)-1)^3},\\
\tilde{B}_5^6(t) &= \tfrac{-48\cos(\alpha/2)\sin(t/2)^5\sin(\alpha/2-t/2)}{(\cos(\alpha)-1)^3},\\
\tilde{B}_6^6(t) &= \tfrac{(\cos(t)-1)^3}{(\cos(\alpha)-1)^3}.
\end{aligned}
\tag{5.1}
$$

Moreover, we also recall that in [Mainar 2007] it is shown how to derive a *de Casteljau-like algorithm* for the evaluation of trigonometric Bézier curves over the space $\tilde{U}_2$, which can be developed as follows. After rewriting the basis $\{\tilde{B}_i^2(t)\}_{i=0,1,2}$, $t\in[0,\alpha]$ in terms of the trigonometric function $C(t):=\cos(t)-1$, we obtain

$$\left(\tilde{B}_0^2(t),\,\tilde{B}_1^2(t),\,\tilde{B}_2^2(t)\right)=\left(\frac{C(\alpha-t)}{C(\alpha)},\,1-\frac{C(\alpha-t)}{C(\alpha)}-\frac{C(t)}{C(\alpha)},\,\frac{C(t)}{C(\alpha)}\right),\qquad t\in[0,\alpha].$$

Then let us observe that $S(t):=\sin(t)$, $t\in[0,\alpha]$, can be written in terms of the basis functions $\tilde{B}_i^2$, $i=0,1,2$ on $[0,\alpha]$ as $S(t)=\beta_0\tilde{B}_0^2(t)+\beta_1\tilde{B}_1^2(t)+\beta_2\tilde{B}_2^2(t)$, $t\in[0,\alpha]$ with $\beta_0=0$, $\beta_1=-\frac{C(\alpha)}{S(\alpha)}$, $\beta_2=S(\alpha)$. Moreover, since the normalized B-basis of $\tilde{U}_2$ on the intervals $[0,t]$ and $[t,\alpha]$ can be written as

$$\left(\tilde{B}_0^2(u),\,\tilde{B}_1^2(u),\,\tilde{B}_2^2(u)\right)=\left(\frac{C(t-u)}{C(t)},\,1-\frac{C(t-u)}{C(t)}-\frac{C(u)}{C(t)},\,\frac{C(u)}{C(t)}\right),\qquad u\in[0,t],$$

and

$$\left(\tilde{B}_0^2(u),\,\tilde{B}_1^2(u),\,\tilde{B}_2^2(u)\right)=\left(\frac{C(\alpha-u)}{C(\alpha-t)},\,1-\frac{C(\alpha-u)}{C(\alpha-t)}-\frac{C(u-t)}{C(\alpha-t)},\,\frac{C(u-t)}{C(\alpha-t)}\right),\qquad u\in[t,\alpha],$$

respectively, it follows that the function $S(u)$, on the subintervals $[0,t]$ and $[t,\alpha]$ can be respectively written in terms of the basis functions $\tilde{B}_i^2$, $i=0,1,2$ as

$$S(u)=\begin{cases}\nu_0\tilde{B}_0^2(u)+\nu_1\tilde{B}_1^2(u)+\nu_2\tilde{B}_2^2(u), & u\in[0,t],\\ \eta_0\tilde{B}_0^2(u)+\eta_1\tilde{B}_1^2(u)+\eta_2\tilde{B}_2^2(u), & u\in[t,\alpha],\end{cases}$$

with $\nu_0 = 0,\ \nu_1 = -\frac{C(t)}{S(t)},\ \nu_2 = S(t)$ and $\eta_0 = S(t),\ \eta_1 = S(\alpha) + S'(\alpha)\,\frac{C(\alpha-t)}{S(\alpha-t)},\ \eta_2 = S(\alpha)$. Thus, given a trigonometric Bézier curve $\mathbf{x}(t) = \sum_{i=0}^{2} \mathbf{p}_i \tilde{B}_i^2(t),\ t \in [0, \alpha]$ its evaluation at an arbitrary parameter value $t^* \in [0, \alpha]$ provides the following intermediate control points

$$\begin{array}{rcll} \mathbf{p}_0^1 & = & (1 - \lambda_0^0(t^*))\mathbf{p}_0^0 + \lambda_0^0(t^*)\mathbf{p}_1^0, & \\ \mathbf{p}_1^1 & = & (1 - \lambda_1^0(t^*))\mathbf{p}_1^0 + \lambda_1^0(t^*)\mathbf{p}_2^0, & \text{where} \quad \mathbf{p}_i^0 = \mathbf{p}_i,\ i = 0, 1, 2 \\ \mathbf{p}_0^2 & = & (1 - \lambda_0^1(t^*))\mathbf{p}_0^1 + \lambda_0^1(t^*)\mathbf{p}_1^1, & \end{array} \tag{5.2}$$

and

$$\begin{array}{rcll} \lambda_0^0(t) & = & \frac{\nu_1-\beta_0}{\beta_1-\beta_0} = \frac{\sin(\alpha)\,(\cos(t)-1)}{\sin(t)\,(\cos(\alpha)-1)}\,, & \\ \lambda_1^0(t) & = & \frac{\eta_1-\beta_1}{\beta_2-\beta_1} = \frac{\cos(t)-\cos(\alpha)+\cos(\alpha-t)-1}{\sin(\alpha-t)\,\sin(\alpha)}\,, & \text{such that} \quad \mathbf{x}(t^*) = \mathbf{p}_0^2. \\ \lambda_0^1(t) & = & \frac{\eta_0-\nu_1}{\eta_1-\nu_1} = \frac{\sin(\alpha-t)(\cos(t)-1)}{\sin(\alpha)-\sin(t)-\sin(\alpha-t)}\,, & \end{array}$$

In Figure 5.2 we illustrate an application example of the de Casteljau-like algorithm

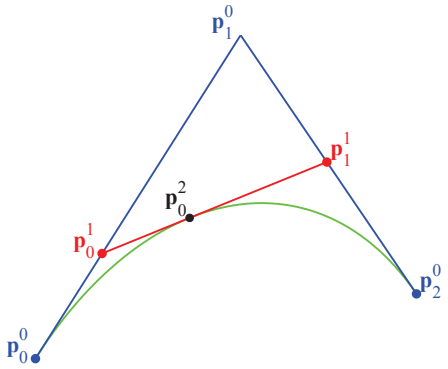


Figure 5.2: Subdivision of a trigonometric Bézier curve in $\tilde{U}_2$ at the parameter $t^* = \frac{\pi}{5} \in I_\alpha = [0, \frac{\pi}{2}]$.

to subdivide the trigonometric Bézier curve $\mathbf{x}(t)$ at a given parameter t^*.

So far, we have focussed our attention on normalized B-bases of pure trigonometric spaces. We conclude this section by recalling some results from the articles [Mainar 2001] and [Mainar 2010] about normalized B-bases of algebraic-trigonometric spaces. In particular, in [Mainar 2001] a normalized B-basis of the mixed linear-trigonometric functional space

$$U_5 = \{1,\ t,\ \sin(t),\ \cos(t),\ \sin(2t),\ \cos(2t)\},$$

hereinafter denoted by $\{B_i^5(t)\}_{i=0,\dots,5}$, is presented for $t \in I_\alpha$ and $0 < \alpha < 2\pi$. Introducing the notations

$$
\begin{aligned}
c_{00} &= 72\alpha n_1 n_2, & c_{10} &= -18 s_1 n_2 (16 s_1^4 \alpha - n_0),\\
c_{01} &= -72 n_1 n_2, & c_{11} &= 288 s_1^5 n_2,\\
c_{02} &= 96 c_2 n_1 n_2, & c_{12} &= -24 n_2 s_1 (16 s_1^4 c_2 + s_2 n_0),\\
c_{03} &= -96 s_2 n_1 n_2, & c_{13} &= 24 n_2 s_1 (16 s_1^4 s_2 - c_2 n_0),\\
c_{04} &= -12 c_3 n_1 n_2, & c_{14} &= 6 s_1 n_2 (s_3 n_0 + 8 s_1^4 c_3),\\
c_{05} &= 12 s_3 n_1 n_2, & c_{15} &= 6 s_1 n_2 (c_3 n_0 - 8 s_1^4 s_3),
\end{aligned}
$$

$$
\begin{aligned}
c_{20} &= 6 n_0 (-3 s_2 s_1 (c_2 + 2) + \alpha (5 s_1 + c_1 s_3)), & c_{30} &= -3 s_1 n_0 (n_0 + 8 c_1 n_1),\\
c_{21} &= -48 s_1^5 n_0, & c_{31} &= 48 s_1^5 n_0,\\
c_{22} &= 24 s_2^2 n_0 (2 s_1 - \alpha c_1), & c_{32} &= 24 s_1 s_2 n_0 (s_2 - \alpha),\\
c_{23} &= 12 n_0 (c_1 - c_1 c_3 + s_1 s_3 - \alpha (2 s_1 + c_1 s_3)), & c_{33} &= 4 s_1 n_0 (n_0 + 8 c_1 n_1),\\
c_{24} &= 6 n_0 (c_2 - 1)(s_1 (5 c_2 + 3) - 4 \alpha c_1^3), & c_{34} &= 12 s_1^2 n_0 (2 c_1 \alpha - s_1 (c_2 + 3)),\\
c_{25} &= 6 n_0 (c_1 (c_2 - 1)(5 c_2 - 2) + \alpha s_1 (c_3 + 2 c_2)), & c_{35} &= -s_1 n_0 (n_0 + 8 c_1 n_1),
\end{aligned}
$$

$$
\begin{aligned}
c_{40} &= 18 s_1 n_0 n_2, & c_{50} &= 0,\\
c_{41} &= -288 s_1^5 n_2, & c_{51} &= 72 n_1 n_2,\\
c_{42} &= 384 s_1^5 n_2, & c_{52} &= -96 n_1 n_2,\\
c_{43} &= -24 s_1 n_0 n_2, & c_{53} &= 0,\\
c_{44} &= -48 s_1^5 n_2, & c_{54} &= 12 n_1 n_2,\\
c_{45} &= 6 s_1 n_0 n_2, & c_{55} &= 0,
\end{aligned}
$$

where

$$s_1 = \sin(\tfrac{\alpha}{2}), \quad c_1 = \cos(\tfrac{\alpha}{2}), \quad s_2 = \sin(\alpha), \quad c_2 = \cos(\alpha), \quad s_3 = \sin(2\alpha), \quad c_3 = \cos(2\alpha), \tag{5.3}$$

and

$$n_0 = 6\alpha - 8 s_2 + s_3, \quad n_1 = c_1 (s_2 - 3\alpha) + 4 s_1, \quad n_2 = (2 + c_2)\alpha - 3 s_2, \tag{5.4}$$

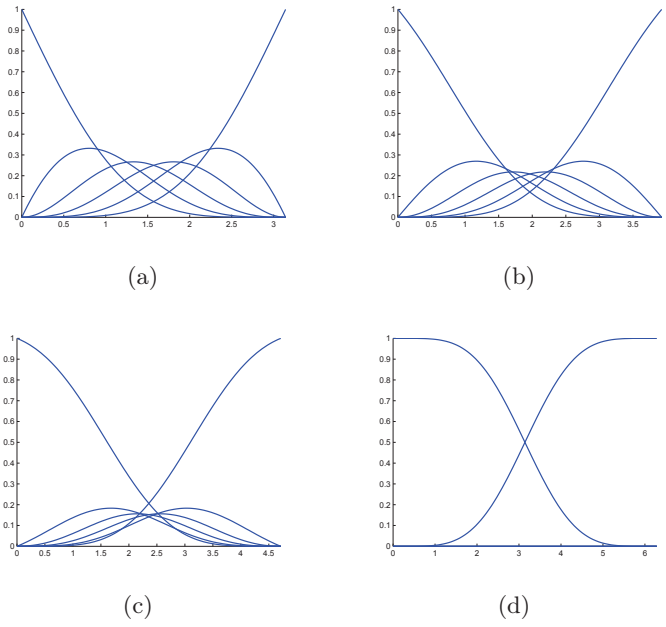
the normalized B-basis of U_5 can be written using the explicit expressions

$$B_i^5(t) = \frac{1}{12 n_0 n_1 n_2} (c_{i0} + c_{i1} t + c_{i2} \sin(t) + c_{i3} \cos(t) + c_{i4} \sin(2t) + c_{i5} \cos(2t)), \quad i = 0, \dots, 5. \tag{5.5}$$

As in the previous case, the parameter α plays a tension-like effect which is illustrated in Figure 5.3 (a),(b),(c). Moreover, when α tends to 2π the normalized B-basis in (5.5) assumes the following form

$$
\begin{aligned}
&\lim_{\alpha \to 2\pi} B_0^5(t) = \frac{1}{12\pi} (12\pi - 6t + 8 \sin(t) - \sin(2t)),\\
&\lim_{\alpha \to 2\pi} B_1^5(t) = \lim_{\alpha \to 2\pi} B_2^5(t) = \lim_{\alpha \to 2\pi} B_3^5(t) = \lim_{\alpha \to 2\pi} B_4^5(t) = 0,\\
&\lim_{\alpha \to 2\pi} B_5^5(t) = \frac{1}{12\pi} (6t - 8 \sin(t) + \sin(2t)).
\end{aligned}
$$

This limit case is illustrated in Figure 5.3(d).



(a) (b)

(c) (d)

Figure 5.3: Basis functions $B_i^5(t)$, $i = 0, ..., 5$ for $t \in I_\alpha$ and $\alpha = \pi$ (a), $\alpha = \frac{5}{4}\pi$ (b), $\alpha = \frac{3}{2}\pi$ (c), $\alpha \to 2\pi$ (d).

In [Mainar 2001] it was also proven that the normalized B-basis $\{B_i^5(t)\}_{i=0,...,5}$ of the space U_5 tends to the ordinary Bernstein polynomials of degree 5, as α approaches 0. We further observe that, if the free parameter α is restricted to the interval $(0, \frac{2}{3}\pi)$, then the space U_5 is an extended Chebyshev space (see [Mainar 2010]) and the normalized B-basis B_i^5, $i = 0, ..., 5$ for such a space can be obtained using an iterative integral procedure starting from the (not normalized) B-basis

$$\begin{aligned} B_0^3(t) &= \frac{\sin(2\alpha-2t)-2\sin(\alpha-t)}{\sin(2\alpha)-2\sin(\alpha)}, \\ B_1^3(t) &= \frac{2\sin(\alpha-2t)-2\sin(\alpha-t)-\sin(2\alpha-t)+\sin(2\alpha-2t)+3\sin(t)}{\sin(2\alpha)-\sin(3\alpha)+\sin(\alpha)}, \\ B_2^3(t) &= \frac{\sin(2t)+3\sin(\alpha-t)-2\sin(\alpha-2t)-\sin(\alpha+t)-2\sin(t)}{\sin(\alpha)+\sin(2\alpha)-\sin(3\alpha)}, \\ B_3^3(t) &= \frac{\sin(2t)-2\sin(t)}{\sin(2\alpha)-2\sin(\alpha)}, \end{aligned} \tag{5.6}$$

for the extended Chebyshev space $U_3 := \langle \cos(t), \sin(t), \cos(2t), \sin(2t) \rangle$ where $t \in [0, \alpha]$ and $0 < \alpha < \frac{2}{3}\pi$. In fact, starting from the B-basis in (5.6), for all $n \geq 4$ we can define a normalized B-basis of order $n+1$ of the space $U_n := \langle 1, t, ..., t^{n-4}, U_3 \rangle$ via the iterative integral construction

$$\begin{aligned} B_0^n(t) &:= 1 - \textstyle\int_0^t \delta_0^{n-1} B_0^{n-1}(s)\, ds, \\ B_i^n(t) &:= \textstyle\int_0^t \left(\delta_{i-1}^{n-1} B_{i-1}^{n-1}(s) - \delta_i^{n-1} B_i^{n-1}(s)\right) ds, \quad i = 1, \cdots, n-1, \\ B_n^n(t) &:= \textstyle\int_0^t \delta_{n-1}^{n-1} B_{n-1}^{n-1}(s)\, ds, \end{aligned} \tag{5.7}$$

where $\delta_i^{n-1} := 1/\int_0^\alpha B_i^{n-1}(s)\,ds,\ \ i = 0, \cdots, n-1$ (see [Mainar 2010]).

Thus, applying once the recursion (5.7) we can construct a normalized B-basis of the space $U_4 = \langle 1, U_3\rangle = \tilde{U}_4$, which, being unique, coincides with the expression for $\tilde{B}_i^4$, $i = 0, ..., 4$ given in (5.1). Moreover, applying twice the recursion (5.7) we can construct a normalized B-basis of the space $U_5 = \langle 1, t, U_3\rangle$, which, being unique, coincides with the expression given in (5.5).

5.2 AT-Bézier curves over the mixed algebraic-trigonometric space U_5

In the following we are particularly interested in the parametric curves defined over the mixed algebraic-trigonometric space U_5. We will refer to these curves as *algebraic-trigonometric Bézier curves* or *AT-Bézier curves*. From the results in [Mainar 2001] it is well known that, since the space U_5 has a normalized B-basis, then we can define parametric curves over U_5 through a control polygon in a similar way to our familiar polynomial Bézier case. More precisely, an AT-Bézier curve defined over the space U_5 can be described by the Bézier-like form

$$\mathbf{x}(t) = \sum_{i=0}^{5} \mathbf{p}_i\, B_i^5(t)\,, \qquad t \in [0, \alpha], \quad 0 < \alpha < 2\pi, \tag{5.8}$$

where B_i^5, $i = 0, \cdots, 5$ are the basis functions given in (5.5).

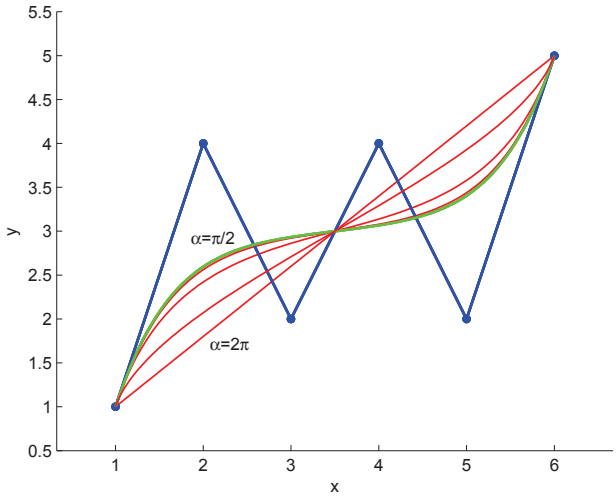


Figure 5.4: Comparison of an ordinary degree 5 Bézier curve (green) and AT–Bézier curves (red) for $\alpha = \frac{\pi}{2}, \pi, \frac{3}{2}\pi, 2\pi$ obtained from the same control polygon (blue).

These curves possess all the good properties of the polynomial Bézier curves such as containment in the convex hull, affine invariance, variation diminishing property, interpolation of end points and tangency to the control polygon at the end points.

Furthermore they depend on the parameter α which can be used as shape parameter. Figure 5.4 shows the ordinary degree 5 Bézier curve compared to AT–Bézier curves obtained for different values of α, starting from the same control polygon. Another advantage of AT–Bézier curves lies in the fact that they are capable of reproducing arcs of arbitrary length (depending on the choice of $\alpha \in (0, 2\pi)$) of planar trigonometric curves such as the ones displayed in Figure 5.5. Taking into account that $\mathbf{x}(t)$ is expressed in terms of the normalized B-basis $B_i^5(t)$, $i = 0, ..., 5$ whose basis functions are linear combinations of the functions $\{1, t, \sin(t), \cos(t), \sin(2t), \cos(2t)\}$ by which these classical trigonometric curves are composed, the explicit formulae for abscissae and ordinatae of the control points can be carried out by solving 6×6 systems of linear equations.

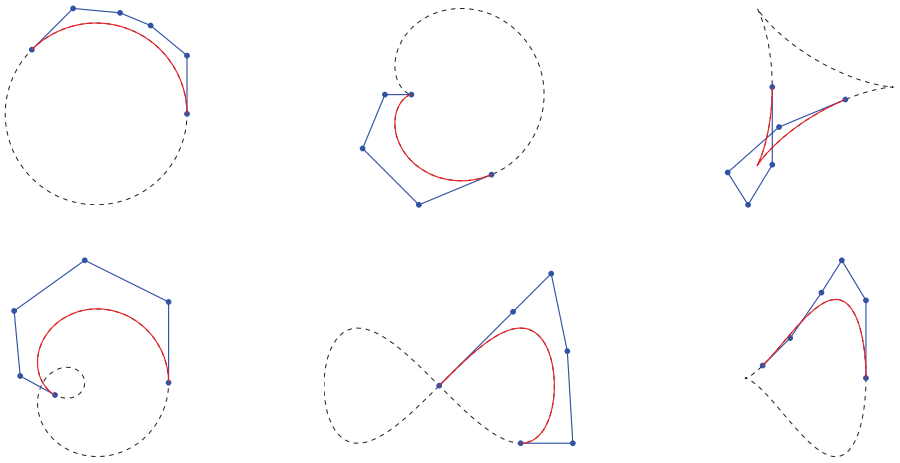


Figure 5.5: Reproduction of arcs of different trigonometric curves by means of AT–Bézier curves over U_5 with $\alpha = \frac{3}{4}\pi$. From left to right, circle, cardioid, deltoid (top), limacon, lemniscate, piriform (bottom).

In the absence of a de Casteljau-like algorithm for the evaluation of these AT-Bézier curves, we propose a fast visualization algorithm for them via a non–stationary subdivision scheme [Romani 2009]. To this end, following the reasoning in [Mainar 2010], for $t \in [0, \alpha]$, $0 < \alpha < \frac{2}{3}\pi$, we can construct a generalized (not normalized) non-negative uniform B-spline basis for the space U_3, which turns out to be totally positive under the stronger requirement $0 < \alpha < \frac{\pi}{2}$. More precisely, the restriction of these generalized B-spline basis functions to the interval $[0, \alpha]$ yields

the following four functions N_i^3, $i = 0, 1, 2, 3$, having the expressions

$$N_0^3(t) = \frac{\sin(2t)-2\sin(t)}{\sin(2\alpha)-2\sin(\alpha)},$$
$$N_1^3(t) = \frac{4(\cos(\alpha)+2)\left(\sin(2\alpha+t)-\sin(2\alpha-t)-\sin(\alpha-t)-\frac{1}{2}\left(\sin(\alpha+2t)-\sin(\alpha-2t)-\sin(2(\alpha-t))\right)\right)}{2\sin(2\alpha)+\sin(3\alpha)-7\sin(\alpha)},$$
$$N_2^3(t) = \frac{4(\cos(\alpha)+2)\left(\sin(3\alpha-t)-\sin(\alpha+t)-\sin(t)-\frac{1}{2}(\sin(\alpha-2t)-\sin(2t)+\sin(3\alpha-2t))\right)}{2\sin(2\alpha)+\sin(3\alpha)-7\sin(\alpha)},$$
$$N_3^3(t) = \frac{\sin(2(\alpha-t))-2\sin(\alpha-t)}{\sin(2\alpha)-2\sin(\alpha)}.$$

Using the iterative procedure described in [Mainar 2010], for all $n \geq 4$ we can define a normalized, generalized uniform B-spline basis of order $n+1$ of the space $U_n := \langle 1, t, ..., t^{n-4}, U_3\rangle$. Then, extending the results in [Romani 2004] about the matrix conversion between Bézier and B-spline polynomial representations, for U_5 we work out the 6×6 matrix transforming the six functions N_i^5, $i = 0, \ldots, 5$ obtained as restrictions of the normalized, generalized, uniform B-spline basis functions of order 6 to the interval $[0, \alpha]$, into the six basis functions B_i^5, $i = 0, ..., 5$ providing the (Bézier-like) normalized B-basis of the same space on the interval $[0, \alpha]$. This matrix, denoted by $M^{(5)}$ has the following entries

$$M^{(5)} = \frac{1}{D}\begin{bmatrix} 0 & m_1^{(5)} & m_6^{(5)} & m_{10}^{(5)} & m_6^{(5)} & m_1^{(5)} \\ 0 & m_2^{(5)} & m_7^{(5)} & m_{10}^{(5)} & m_5^{(5)} & 0 \\ 0 & m_3^{(5)} & m_8^{(5)} & m_9^{(5)} & m_4^{(5)} & 0 \\ 0 & m_4^{(5)} & m_9^{(5)} & m_8^{(5)} & m_3^{(5)} & 0 \\ 0 & m_5^{(5)} & m_{10}^{(5)} & m_7^{(5)} & m_2^{(5)} & 0 \\ m_1^{(5)} & m_6^{(5)} & m_{10}^{(5)} & m_6^{(5)} & m_1^{(5)} & 0 \end{bmatrix} \qquad \text{with} \quad D = 384\,\alpha\, c_1^2\, s_1^4,$$

$$m_1^{(5)} = 6\alpha - 8s_2 + s_3, \qquad m_2^{(5)} = 2m_1^{(5)},$$
$$m_3^{(5)} = -12\Big((2\alpha - s_2)c_2 + \alpha - 2s_2\Big), \qquad m_4^{(5)} = 6\Big(8(c_2+1)(\alpha - s_2) + 2c_3\alpha - s_3\Big),$$
$$m_5^{(5)} = 2\Big(8s_2(1+4c_1^4) + 6\alpha(1-8c_1^4) + s_3\Big), \qquad m_6^{(5)} = -4\Big(6c_1^2(2c_2-1)\alpha - 7s_2c_2^2 + s_2c_2\Big),$$
$$m_7^{(5)} = 2m_6^{(5)} - m_5^{(5)}, \qquad m_8^{(5)} = -12\Big((10c_2^2 + 10c_2 + 1)\alpha - s_2(11c_2 + 6c_2^2 + 4)\Big),$$
$$m_9^{(5)} = D - m_3^{(5)} - m_4^{(5)} - m_8^{(5)}, \qquad m_{10}^{(5)} = D - 2(m_1^{(5)} + m_6^{(5)}),$$

and $s_1, c_1, s_2, c_2, s_3, c_3$ as in (5.3). Since for all $\alpha \in (0, \frac{\pi}{2})$ the matrix $M^{(5)}$ is invertible, we can compute its inverse which has the form

$$R^{(5)} := (M^{(5)})^{-1} = \frac{1}{m_1^{(5)}\rho_6}\begin{bmatrix} 0 & r_2^{(5)} & r_8^{(5)} & r_{13}^{(5)} & r_7^{(5)} & r_1^{(5)} \\ 0 & r_3^{(5)} & r_9^{(5)} & r_{12}^{(5)} & r_6^{(5)} & 0 \\ 0 & r_4^{(5)} & r_{10}^{(5)} & r_{11}^{(5)} & r_5^{(5)} & 0 \\ 0 & r_5^{(5)} & r_{11}^{(5)} & r_{10}^{(5)} & r_4^{(5)} & 0 \\ 0 & r_6^{(5)} & r_{12}^{(5)} & r_9^{(5)} & r_3^{(5)} & 0 \\ r_1^{(5)} & r_7^{(5)} & r_{13}^{(5)} & r_8^{(5)} & r_2^{(5)} & 0 \end{bmatrix},$$

where

$$
\begin{aligned}
r_1^{(5)} &= D\rho_6, \\
r_4^{(5)} &= -m_1^{(5)}\rho_5, \\
r_3^{(5)} &= -m_1^{(5)}\Big(\rho_4 - (m_3^{(5)} + m_8^{(5)})\rho_1\Big)D + r_4^{(5)}, \\
r_2^{(5)} &= -(\rho_7 + (m_4^{(5)} + m_8^{(5)})\rho_3)D + r_4^{(5)}, \\
r_{10}^{(5)} &= m_1^{(5)}(\rho_5 + \rho_2\rho_1), \\
r_9^{(5)} &= m_1^{(5)}\Big(\rho_4 - (2m_1^{(5)} - m_5^{(5)} + 2m_6^{(5)})\rho_1\Big)D + r_{10}^{(5)}, \\
r_8^{(5)} &= (\rho_7 + 2m_6^{(5)}\rho_3)D + r_{10}^{(5)},
\end{aligned}
\qquad
\begin{aligned}
r_{11}^{(5)} &= -m_1^{(5)}(2m_1^{(5)} - m_5^{(5)})\rho_1 D + r_{10}^{(5)}, \\
r_{12}^{(5)} &= m_1^{(5)}(\rho_4 + (2m_6^{(5)} - D)\rho_1)D + r_{10}^{(5)}, \\
r_{13}^{(5)} &= \Big(\rho_7 - (2m_6^{(5)} - D)\rho_3\Big)D + r_{11}^{(5)}, \\
r_5^{(5)} &= m_1^{(5)}\rho_6 - (r_4^{(5)} + r_{10}^{(5)} + r_{11}^{(5)}), \\
r_6^{(5)} &= m_1^{(5)}\rho_6 - (r_3^{(5)} + r_9^{(5)} + r_{12}^{(5)}), \\
r_7^{(5)} &= m_1^{(5)}\rho_6 - (r_2^{(5)} + r_8^{(5)} + r_{13}^{(5)} + r_1^{(5)}),
\end{aligned}
$$

and

$$
\begin{aligned}
\rho_1 &= (2m_1^{(5)} + m_5^{(5)}) - (m_3^{(5)} + m_4^{(5)}), \\
\rho_2 &= (2m_1^{(5)} - m_5^{(5)})(m_4^{(5)} + m_8^{(5)}) - 2m_6^{(5)}(m_3^{(5)} - m_4^{(5)}), \\
\rho_3 &= (m_6^{(5)} - m_5^{(5)} + m_1^{(5)})\rho_1, \\
\rho_4 &= (2m_1^{(5)} - m_3^{(5)})D - \rho_2,
\end{aligned}
\qquad
\begin{aligned}
\rho_5 &= (m_3^{(5)}m_5^{(5)} - 2m_1^{(5)}m_4^{(5)})D + (m_3^{(5)} + m_4^{(5)})\rho_2, \\
\rho_6 &= \Big((m_5^{(5)} - m_4^{(5)})D + \rho_2 - \rho_4\Big)\rho_1, \\
\rho_7 &= -(m_1^{(5)} + m_6^{(5)})\rho_4 + (m_1^{(5)}D - \rho_2)\rho_1 - \rho_5.
\end{aligned}
$$

Taking into account that, for $t \in [0, \alpha]$ and $0 < \alpha < \frac{\pi}{2}$,

$$[B_0^5\ B_1^5\ B_2^5\ B_3^5\ B_4^5\ B_5^5] = [N_0^5\ N_1^5\ N_2^5\ N_3^5\ N_4^5\ N_5^5]\, R^{(5)},$$

the B–spline representation

$$\mathbf{x}(t) = \sum_{i=0}^{5} \mathbf{q}_i N_i^5(t)\,, \qquad t \in [0, \alpha],$$

of the curve (5.8) is thus obtained by $[\mathbf{q}_0\ \mathbf{q}_1\ \mathbf{q}_2\ \mathbf{q}_3\ \mathbf{q}_4\ \mathbf{q}_5]^T = R^{(5)}\ [\mathbf{p}_0\ \mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_4\ \mathbf{p}_5]^T$. See Figure 5.6 for the illustration of both sets of control points.

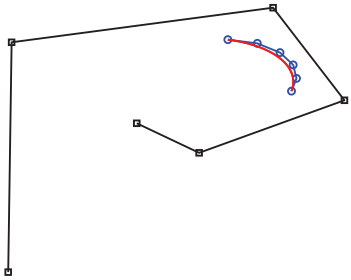


Figure 5.6: B-spline control points (black squares) and Bézier control points (blue circles) for the AT-Bézier curve (red) defined on $t \in [0, \alpha]$ with $\alpha = \frac{\pi}{5}$.

This transformation allows us to apply the non-stationary approximating subdivision scheme proposed in [Romani 2009, Section 4.2] to achieve a fast evaluation algorithm of the algebraic-trigonometric Bézier curves over the space U_5.

In fact this approximating subdivision scheme reproducing functions in the space $\{1, x, e^{tx}, e^{-tx}, e^{2tx}, e^{-2tx}\}$, is a specific member of the family of approximating subdivision schemes proposed in [Romani 2009, Section 2.2], which is shown to generate in the limit exponential B-splines. Given an initial polygon $\mathbf{q}_i^0 := \mathbf{q}_i$, $i = 0, \ldots, 5$, for all successive refinement levels $k \geq 0$ the subdivision rules are given by

$$\begin{aligned} \mathbf{q}_{2j}^{k+1} &= a_1^k\ (\mathbf{q}_{j-1}^k + \mathbf{q}_{j+1}^k) + a_3^k\ \mathbf{q}_j^k, \\ \mathbf{q}_{2j+1}^{k+1} &= a_0^k\ (\mathbf{q}_{j-1}^k + \mathbf{q}_{j+2}^k) + a_2^k\ (\mathbf{q}_j^k + \mathbf{q}_{j+1}^k), \end{aligned} \tag{5.9}$$

with

$$a_0^k = \frac{1}{16(v^k)^2(v^k+1)},\quad a_1^k = \frac{2v^k+1}{8v^k(v^k+1)},\quad a_2^k = \frac{8(v^k)^3+8(v^k)^2-1}{16(v^k)^2(v^k+1)},\quad a_3^k = \frac{4(v^k)^2+2v^k-1}{4v^k(v^k+1)},$$

and v^k updated through $v^k = \sqrt{\frac{1+v^{k-1}}{2}}$ starting from the initial value $v^{-1} = \cos(\alpha)$. As shown in [Romani 2009], this non-stationary subdivision algorithm converges in the limit to the AT–Bézier curve defined by the control polygon $\mathbf{p}_i$, $i = 0, ..., 5$ and the angle $\alpha \in \left(0, \frac{\pi}{2}\right)$ (see Figure 5.7 for an application example of this evaluation algorithm).

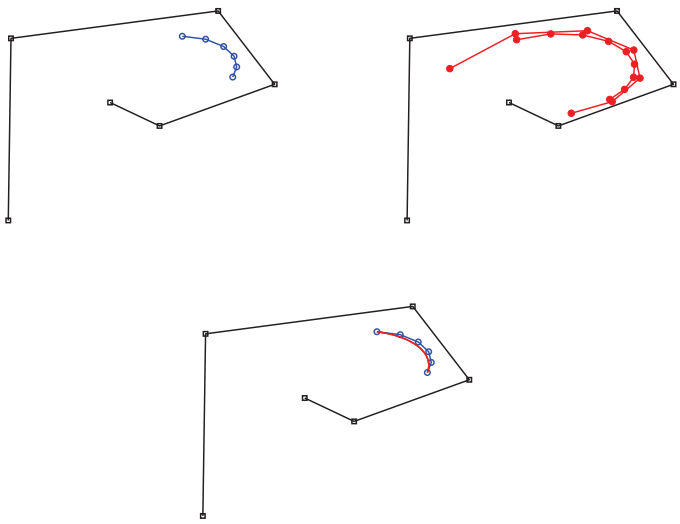


Figure 5.7: Left: initial B-spline control polygon (black) and corresponding Bézier control polygon (blue). Middle: first two refinements of the B-spline control polygon. Right: corresponding limit curve (red) for $\alpha = \frac{\pi}{5}$ and $k = 6$.

Since the initial control polygon $Q^0 = \{\mathbf{q}_j^0\}_{j=0,\ldots,\mathcal{N}_0-1}$ has $\mathcal{N}_0 = 6$ vertices, the polygon $Q^k = \{\mathbf{q}_j^k\}_{j=0,\ldots,\mathcal{N}_k-1}$ obtained at step k through the refinement rules (5.9), contains $\mathcal{N}_k = 2^k + 5$ vertices, of which $2^{k-1} + 3$ are even and $2^{k-1} + 2$ are odd. The computational cost of computing a point of Q^k from those of Q^{k-1} is equal to 2 multiplications and 2 additions for q_{2j}^{k+1} and 2 multiplications and 3 additions for

q_{2j+1}^{k+1}. Moreover, we have 17 multiplications and 9 additions to calculate a_0, a_1, a_2 and a_3 (in fact we have 3 multiplications and 1 addition for a_0, 4 multiplications and 2 additions for a_1, 5 multiplications and 3 additions for a_2 and 5 multiplications and 3 additions for a_3). We take into account that to compute v^k from v^{k-1} we have 1 multiplication, 1 addition and 1 square root and we have 2 multiplications to determine v^2 and v^3. By observing that the coefficients v^k and a_i^k don't change for every level k, we conclude that the number of operations required to compute all the vertices of Q^k applying the refinement rules (5.9) to the vertices of Q^0 is equal to $\sum_{i=1}^{k} 9\,2^{i-1} + 31\,i + 22 = 9(2^k - 1) + \frac{k}{2}(31k + 75)$.

5.3 Algebraic-Trigonometric Pythagorean Hodograph (ATPH) curves and their properties: real representation

Exploiting the fact that if $f \in \tilde{U}_2$ then $f^2 \in \tilde{U}_4$ and $\int f^2 \in U_5$, we now extend the well–known definition of polynomial Pythagorean–Hodograph (PH) curves to the algebraic-trigonometric case, replacing the space of quadratic polynomials $\langle 1, t, t^2\rangle$ by the space $\tilde{U}_2 = \langle 1, \sin(t), \cos(t)\rangle$. Since $f \in \tilde{U}_2$ exists for $\alpha \in (0, \pi)$, the construction of the new class of Pythagorean-Hodograph curves is restricted to $t \in [0, \alpha]$ with $\alpha \in (0, \pi)$.

Definition 6. *Let $u(t)$, $v(t)$ and $\zeta(t)$ be non-zero real functions in the space $\tilde{U}_2$ such that $u(t)$ and $v(t)$ are relatively prime (namely $\gcd(u(t), v(t)) = 1$) and both non–constant. Then, a planar parametric curve $\mathbf{x}(t) = (x(t), y(t))$ whose first derivative is of the form*

$$x'(t) = \zeta(t)\Big(u^2(t) - v^2(t)\Big) \qquad \text{and} \qquad y'(t) = 2\zeta(t)u(t)v(t) \tag{5.10}$$

is called Algebraic–Trigonometric PH curve *or* ATPH curve.

As in the case of polynomial PH curves, the curve's parametric speed is given by

$$\sigma(t) := \sqrt{(x'(t))^2 + (y'(t))^2} = \zeta(t)\,(u^2(t) + v^2(t)) \tag{5.11}$$

and its unit tangent, unit normal and (signed) curvature are given respectively by

$$\mathbf{t} = \frac{(u^2 - v^2, 2uv)}{u^2 + v^2}, \quad \mathbf{n} = \frac{(2uv, v^2 - u^2)}{u^2 + v^2}, \quad \kappa = \frac{2(uv' - u'v)}{\zeta(u^2 + v^2)^2}, \tag{5.12}$$

where, for conciseness, in (5.12) the parameter t is omitted.

In the following we will restrict our attention to the regular case $\zeta(t) = 1$. By integrating (5.10) for $\zeta(t) = 1$ we obtain a parametric curve in the mixed algebraic–trigonometric space U_5 which is expressable in the basis (5.5) as formulated in the following Proposition 2 which uses the notation

$$a_{10} = n_0 - 6n_2,\ a_{20} = n_2(c_2+2),\ a_{21} = 3n_2 s_2,\ a_{22} = 3n_2(c_2+1),\ a_{23} = n_2(2c_2+1), \tag{5.13}$$

where n_i, s_i, c_i from (5.3) and (5.4), and the new variables

$$u_3 = u_0 + u_2\,,\ v_3 = v_0 + v_2, \tag{5.14}$$

as well as the abbreviations:

$$\gamma_{ij}^{\pm} = u_i u_j \pm v_i v_j\,,\ \delta_{ij} = u_i v_j + u_j v_i\,,\ i,j \in \{0,1,2,3\}. \tag{5.15}$$

Proposition 2. *A planar, parametric curve over the mixed algebraic–trigonometric space U_5 expressed in terms of the normalized B-basis (5.5) as*

$$\mathbf{x}(t) = \sum_{i=0}^{5} \mathbf{p}_i B_i^5(t), \qquad t \in [0,\alpha], \tag{5.16}$$

is a (non-cuspidal) Algebraic–Trigonometric PH curve in the sense of Definition 6 if and only if its control points can be expressed in the form

$$\mathbf{p}_1 = \mathbf{p}_0 + \frac{n_0}{16 s_1^4}(\gamma_{33}^-, \delta_{33}) \tag{5.17}$$

$$\mathbf{p}_2 = \mathbf{p}_1 + \frac{a_{10}}{8 s_1^4}((\gamma_{33}^-, \delta_{33}) + \tan\left(\frac{\alpha}{2}\right)(\gamma_{13}^-, \delta_{13})) \tag{5.18}$$

$$\begin{aligned}\mathbf{p}_3 = \mathbf{p}_2 + \frac{1}{4 s_1^4}(&a_{20}(\gamma_{00}^-, \delta_{00}) + a_{21}(\gamma_{13}^-, \delta_{13}) + a_{22}(\gamma_{02}^-, \delta_{02}) + \frac{4 s_1^4}{1-c_2} n_2(\gamma_{11}^-, \delta_{11}) \\ &+ a_{23}(\gamma_{22}^-, \delta_{22}))\end{aligned} \tag{5.19}$$

$$\begin{aligned}\mathbf{p}_4 = \mathbf{p}_3 + \frac{a_{10}}{8 s_1^4}((&\gamma_{00}^-, \delta_{00}) + (s_2 + \tan\left(\frac{\alpha}{2}\right))(\gamma_{01}^-, \delta_{01}) + (c_2+1)(\gamma_{02}^-, \delta_{02}) + \\ &+ (1-c_2)(\gamma_{11}^-, \delta_{11}) + \frac{s_4}{c_1}(\gamma_{12}^-, \delta_{12}) + c_2(\gamma_{22}^-, \delta_{22}))\end{aligned} \tag{5.20}$$

$$\begin{aligned}\mathbf{p}_5 = \mathbf{p}_4 + \frac{n_0}{16 s_1^4}((&\gamma_{00}^-, \delta_{00}) + 2 s_2(\gamma_{01}^-, \delta_{01}) + 2 c_2(\gamma_{02}^-, \delta_{02}) + s_2^2(\gamma_{11}^-, \delta_{11}) + s_3(\gamma_{12}^-, \delta_{12}) + \\ &+ c_2^2(\gamma_{22}^-, \delta_{22}))\end{aligned} \tag{5.21}$$

in terms of the real values (u_0, u_1, u_2), (v_0, v_1, v_2) and the abbreviations (5.3), (5.4), (5.13), (5.14) and (5.15).

Proof. We define the following two functions in the space $\tilde{U}_2$:

$$\begin{aligned} u(t) &= u_0 + u_1 \sin(t) + u_2 \cos(t), \\ v(t) &= v_0 + v_1 \sin(t) + v_2 \cos(t), \end{aligned} \tag{5.22}$$

with $t \in [0,\alpha]$. We substitue these functions into (5.10) and by integrating for $\zeta(t) = 1$ we obtain

$$\begin{aligned} x(t) &= \int x'(\tau)d\tau = k_1 + \widetilde{a}_0 + \widetilde{a}_1 t + \widetilde{a}_2 \sin(t) + \widetilde{a}_3 \cos(t) + \widetilde{a}_4 \sin(2t) + \widetilde{a}_0 \cos(2t) \\ y(t) &= \int y'(\tau)d\tau = k_2 + \widetilde{b}_0 + \widetilde{b}_1 t + \widetilde{b}_2 \sin(t) + \widetilde{b}_3 \cos(t) + \widetilde{b}_4 \sin(2t) + \widetilde{b}_0 \cos(2t) \end{aligned} \tag{5.23}$$

where

$$\begin{aligned} \widetilde{a}_0 &= \tfrac{1}{2}(v_1v_2 - u_1u_2), & \widetilde{b}_0 &= -\tfrac{1}{2}(u_2v_1 + u_1v_2), \\ \widetilde{a}_1 &= \tfrac{1}{2}(u_1^2 - v_1^2 + u_2^2 - v_2^2) + u_0^2 - v_0^2, & \widetilde{b}_1 &= 2u_0v_0 + u_1v_1 + u_2v_2, \\ \widetilde{a}_2 &= 2(u_0u_2 - v_0v_2), & \widetilde{b}_2 &= 2(u_0v_2 + u_2v_0), \\ \widetilde{a}_3 &= 2(v_0v_1 - u_0u_1), & \widetilde{b}_3 &= -2(u_0v_1 + u_1v_0), \\ \widetilde{a}_4 &= \tfrac{1}{4}(u_2^2 - u_1^2 + v_1^2 - v_2^2), & \widetilde{b}_4 &= \tfrac{1}{2}(u_2v_2 - u_1v_1), \end{aligned}$$

and k_1 and k_2 the integration constants. Expressing the basis functions of the functional space $U_5 = \{1, t, \sin(t), \cos(t), \sin(2t), \cos(2t)\}$ in terms of the normalized B-basis (5.5) and substituting these basis functions in (5.23) yields the expressions (5.17) - (5.21), where $\mathbf{p}_0 = (k_1, k_2) - (\gamma_{01}^- + \gamma_{13}^-, \delta_{01} + \delta_{13})$. □

By (5.10) we have

$$\sigma(t) = u^2(t) + v^2(t)$$

and thus the arc-length is calculated to be

$$\begin{aligned} \int \sigma(t)dt &= -\tfrac{1}{2}\gamma_{12}^+ + \tfrac{1}{2}\left((2\gamma_{00}^+ + \gamma_{11}^+ + \gamma_{22}^+)t + 4\gamma_{02}^+ \sin(t)\right. \\ &- \left. 4\gamma_{01}^+ \cos(t) + \tfrac{1}{2}(\gamma_{22}^+ - \gamma_{11}^+)\sin(2t) - \gamma_{12}^+ \cos(2t)\right). \end{aligned} \tag{5.24}$$

5.4 C^1 Hermite interpolation problem: first approach

In this section we consider the following problem.

Problem 1. Given arbitrary control points $\mathbf{p}_0 \neq \mathbf{p}_1$ and $\mathbf{p}_4 \neq \mathbf{p}_5$ of an AT-Bézier curve $\mathbf{x}(t) = \sum_{i=0}^{5} \mathbf{p}_i B_i^5(t)$, $t \in [0, \alpha]$, defined over the space U_5, we look for the two remaining inner control points $\mathbf{p}_2$ and $\mathbf{p}_3$ such that all six are expressible in the form given by equations (5.17)-(5.21) for some real values of $(u_0,\ u_1,\ u_2)$ and $(v_0,\ v_1,\ v_2)$.

In order to solve *Problem* 1 we need the following Lemma, recalled from [Farouki 1995].

Lemma 1. *For all real values a and b, the real solutions to the equations*

$$u^2 - v^2 = a \ \text{ and } \ 2uv = b \tag{5.25}$$

may be expressed in the form

$$(u, v) = \pm\left(\sqrt{\frac{1}{2}(c + a)}, \operatorname{sign}(b)\sqrt{\frac{1}{2}(c - a)}\right), \tag{5.26}$$

where $c = \sqrt{a^2 + b^2}$ and we take $\operatorname{sign}(b) = \pm 1$ when $b = 0$.

By defining $\Delta\mathbf{p}_i = (\Delta x_i, \Delta y_i) = \mathbf{p}_{i+1} - \mathbf{p}_i$ for $i = 0, \ldots, n-1$ with $(\Delta x_i, \Delta y_i) = (x_{i+1} - x_i, y_{i+1} - y_i)$ and $|\Delta\mathbf{p}_i| = \sqrt{(\Delta x_i)^2 + (\Delta y_i)^2}$ for $i = 0, \ldots, n-1$, the solutions of *Problem 1* are given by the following Proposition

Proposition 3. *The solutions of the Hermite interpolation problem are given by the following values for the coefficients* (u_0, u_1, u_2) *and* (v_0, v_1, v_2)*:*

$$\begin{aligned}
(u_1, v_1) &= \frac{1}{\sqrt{\gamma_1}}\Big(\frac{1}{\sqrt{\gamma_1}}(s_1(u_3\,\widetilde{g} + u_4\,\widetilde{f}), s_1(v_3\,\widetilde{g} + v_4\,\widetilde{f})) \pm \sqrt{\frac{1}{2}}(\sqrt{c+a}, \mathrm{sign}(b)\sqrt{c-a})\Big) \\
(u_2, v_2) &= \frac{1}{c_2 - 1}\Big(2\sqrt{\frac{2s_1^4}{n_0}}(\pm\sqrt{|\Delta\mathbf{p}_4| + \Delta x_4} - (\pm\sqrt{|\Delta\mathbf{p}_0| + \Delta x_0}), \qquad (5.27) \\
&\quad \pm\mathrm{sign}(\Delta y_4)\sqrt{|\Delta\mathbf{p}_4| - \Delta x_4} - (\pm\mathrm{sign}(\Delta y_0)\sqrt{|\Delta\mathbf{p}_0| - \Delta x_0})) - (s_2 u_1, s_2 v_1)\Big) \\
(u_0, v_0) &= \frac{1}{c_2 - 1}\Big(2\sqrt{\frac{2s_1^4}{n_0}}(\pm c_2\sqrt{|\Delta\mathbf{p}_0| + \Delta x_0} - (\pm\sqrt{|\Delta\mathbf{p}_4| + \Delta x_4}), \\
&\quad \pm c_2\mathrm{sign}(\Delta y_0)\sqrt{|\Delta\mathbf{p}_0| - \Delta x_0} - (\pm\mathrm{sign}(\Delta y_4)\sqrt{|\Delta\mathbf{p}_4| - \Delta x_4}) + (s_2 u_1, s_2 v_1)\Big)
\end{aligned}$$

where the quantities $\widetilde{f}$*,* $\widetilde{g}$*,* a*,* b*, and* $c = \sqrt{a^2 + b^2}$ *are defined by*

$$\widetilde{f} = 3c_1\alpha - 2s_1c_1^2 - 4s_1, \quad \widetilde{g} = c_1\alpha - 4c_1^3\alpha + 10s_1c_1^2 - 4s_1, \qquad (5.28)$$

$$a = 4s_1^4((x_4 - x_1) - \gamma), \quad b = 4s_1^4((y_4 - y_1) - \delta) \qquad (5.29)$$

with

$$\gamma = \frac{d}{16n_2s_1^4}(u_3^2 - v_3^2 + u_4^2 - v_4^2) + e\frac{c_2 - 1}{8n_2s_1^4}(u_3u_4 - v_3v_4), \qquad (5.30)$$

$$\delta = \frac{d}{2n_2(c_2 - 1)^2}(u_3v_3 + u_4v_4) + \frac{e}{2n_2(c_2 - 1)}(u_3v_4 + v_3u_4), \qquad (5.31)$$

where

$$d = -9(c_2 + 1)\alpha^2 + (c_2 + 5)(6s_2\alpha + (c_2 - 1)(c_2 + 5)),$$

$$e = (2c_2 + 1)\alpha^2 - 6s_2\alpha + (c_2 - 1)(c_2 - 7),$$

and

$$(u_3, v_3) = \pm 2\sqrt{\frac{2s_1^4}{n_0}}(\sqrt{|\Delta\mathbf{p}_0| + \Delta x_0}, \mathrm{sign}(\Delta y_0)\sqrt{|\Delta\mathbf{p}_0| - \Delta x_0}), \qquad (5.32)$$

$$(u_4, v_4) = \pm 2\sqrt{\frac{2s_1^4}{n_0}}(\sqrt{|\Delta\mathbf{p}_4| + \Delta x_4}, \mathrm{sign}(\Delta y_4)\sqrt{|\Delta\mathbf{p}_4| - \Delta x_4}). \qquad (5.33)$$

Proof. Given arbitrary points $\mathbf{p}_0 \neq \mathbf{p}_1$ and $\mathbf{p}_4 \neq \mathbf{p}_5$, we can find the two additional control points $\mathbf{p}_2$ and $\mathbf{p}_3$. To this end, we introduce the new variables

$$\begin{aligned} u_4 &= u_3 + 2s_1\,(u_1c_1 - u_2\,s_1)\,, \\ v_4 &= v_3 + 2s_1\,(v_1c_1 - v_2\,s_1)\,, \end{aligned} \tag{5.34}$$

and then, the equations (5.21) become

$$\mathbf{p}_5 = \mathbf{p}_4 + \frac{n_0}{16s_1^4}\,(u_4^2 - v_4^2, 2\,u_4\,v_4)\,. \tag{5.35}$$

By applying Lemma 1 to the equations (5.17) and (5.35), we obtain the expressions (5.32) and (5.33). By solving the four linear equations (5.14) and (5.34) for the variables u_0, u_2, v_0, v_2 we obtain the solutions

$$u_0 = \frac{1}{c_2 - 1}(c_2u_3 - u_4 + s_2u_1) \tag{5.36}$$

$$u_2 = \frac{1}{c_2 - 1}(-u_3 + u_4 - s_2u_1) \tag{5.37}$$

$$v_0 = \frac{1}{c_2 - 1}(c_2v_3 - v_4 + s_2v_1) \tag{5.38}$$

$$v_2 = \frac{1}{c_2 - 1}(-v_3 + v_4 - s_2v_1) \tag{5.39}$$

which we insert in

$$\mathbf{p}_4 - \mathbf{p}_1 = (\mathbf{p}_4 - \mathbf{p}_3) + (\mathbf{p}_3 - \mathbf{p}_2) + (\mathbf{p}_2 - \mathbf{p}_1)\,. \tag{5.40}$$

By introducing the variables

$$\begin{aligned} \tilde{u}_1 &= \sqrt{\gamma_1}\,u_1 - \frac{\delta_1}{\sqrt{\gamma_1}}\,, \\ \tilde{v}_1 &= \sqrt{\gamma_1}\,v_1 - \frac{\delta_2}{\sqrt{\gamma_1}}\,, \end{aligned} \tag{5.41}$$

where

$$\gamma_1 = -2\,s_1^2\,(6s_1c_1 - 2c_1^2\alpha - \alpha), \tag{5.42}$$

$$\delta_1 = s_1(u_3\,\widetilde{g} + u_4\,\widetilde{f}) \tag{5.43}$$

$$\delta_2 = s_1(v_3\,\widetilde{g} + v_4\,\widetilde{f}) \tag{5.44}$$

with $\widetilde{f}$ and $\widetilde{g}$ from (5.28), the equation (5.40) becomes

$$\mathbf{p}_4 - \mathbf{p}_1 = \frac{1}{4s_1^4}\,(\tilde{u}_1^2 - \tilde{v}_1^2, 2\,\tilde{u}_1\,\tilde{v}_1) + (\gamma, \delta), \tag{5.45}$$

where γ respectively δ from (5.30) respectively (5.31).

By applying Lemma 1 to the equation (5.45), we obtain

$$(\widetilde{u}_1, \widetilde{v}_1) = \pm\sqrt{\frac{1}{2}}(\sqrt{c+a}, \text{sign}(b)\sqrt{c-a}) \tag{5.46}$$

where the quantities a, b, and $c = \sqrt{a^2+b^2}$ are defined by (5.29).

From (5.41) we have

$$u_1 = \frac{1}{\sqrt{\gamma_1}}(\frac{\delta_1}{\sqrt{\gamma_1}} + \widetilde{u}_1), \qquad v_1 = \frac{1}{\sqrt{\gamma_1}}(\frac{\delta_2}{\sqrt{\gamma_1}} + \widetilde{v}_1).$$

in which we substitue (5.46) and we obtain the expressions for (u_1, v_1).

Into (5.36)-(5.39) we replace (5.32) and (5.33) and we find the expressions (5.27). □

5.5 Algebraic-Trigonometric Pythagorean Hodograph (ATPH) curves and their properties: complex representation

By observing the control point expressions in Proposition 2, we note that we can simplify their form. By introducing the complex notation $\mathbf{u}(t) = u(t) + \mathrm{i}v(t)$ we can write (5.22) as

$$\mathbf{u}(t) = \mathbf{u}_0 + \mathbf{u}_1 \sin(t) + \mathbf{u}_2 \cos(t)\,, \tag{5.47}$$

where $\mathbf{u}_i = u_i + \mathrm{i}v_i$ with $u_i, v_i \in \mathbb{R}$.

With this complex representation, Proposition 2 becomes:

Proposition 4. *A planar, parametric curve over the mixed algebraic–trigonometric space U_5 expressed in terms of the normalized B-basis (5.5) as*

$$\mathbf{x}(t) = \sum_{i=0}^{5} \mathbf{p}_i B_i^5(t), \qquad t \in [0, \alpha], \tag{5.48}$$

is a (non-cuspidal) Algebraic–Trigonometric PH curve in the sense of Definition 6 if and only if its control points can be expressed in the form

$$\mathbf{p}_1 = \mathbf{p}_0 + \frac{n_0}{16 s_1^4}\mathbf{u}_3^2 \tag{5.49}$$

$$\mathbf{p}_2 = \mathbf{p}_1 + \frac{a_{10}}{8 s_1^4}(\mathbf{u}_3^2 + \tan\left(\frac{\alpha}{2}\right)\mathbf{u}_1\mathbf{u}_3) \tag{5.50}$$

$$\mathbf{p}_3 = \mathbf{p}_2 + \frac{1}{4 s_1^4}(a_{20}\mathbf{u}_0^2 + a_{21}\mathbf{u}_1\mathbf{u}_3 + a_{22}\mathbf{u}_0\mathbf{u}_2 + \frac{4 s_1^4}{1-c_2} n_2 \mathbf{u}_1^2 + a_{23}\mathbf{u}_2^2) \tag{5.51}$$

$$\begin{aligned}\mathbf{p}_4 = \mathbf{p}_3 + \frac{a_{10}}{8 s_1^4}(\mathbf{u}_0^2 + (s_2 + \tan\left(\frac{\alpha}{2}\right))\mathbf{u}_0\mathbf{u}_1 + (c_2+1)\mathbf{u}_0\mathbf{u}_2 \\ +(1-c_2)\mathbf{u}_1^2 + \frac{s_4}{c_1}\mathbf{u}_1\mathbf{u}_2 + c_2\mathbf{u}_2^2)\end{aligned} \tag{5.52}$$

$$\mathbf{p}_5 = \mathbf{p}_4 + \frac{n_0}{16 s_1^4}(\mathbf{u}_0^2 + 2s_2\mathbf{u}_0\mathbf{u}_1 + 2c_2\mathbf{u}_0\mathbf{u}_2 + s_2^2\mathbf{u}_1^2 + s_3\mathbf{u}_1\mathbf{u}_2 + c_2^2\mathbf{u}_2^2) \tag{5.53}$$

in terms of the complex values $\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2$ *and* $\mathbf{u}_3 = \mathbf{u}_0 + \mathbf{u}_2$ *and the abbreviations (5.13).*

For solving the Hermite interpolation *Problem 1* the complex values $\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2$ have to be determined in order to obtain the control points in the form (5.49)-(5.53).

Thus, in order to rewrite Proposition 3 in this complex notation, we recall a well-known result from the algebra of complex numbers, which directly follows from De Moivre's theorem; it is the complex equivalent of Lemma 1.

Lemma 2. *Let* $\mathbf{a} = a + \mathrm{i}b = |\mathbf{a}| \exp(\mathrm{i}\omega) = |\mathbf{a}|(\cos(\omega) + \mathrm{i}\sin(\omega))$, $\omega = \arg(\mathbf{a}) \in [-\pi, \pi]$. *Then, the solution of the equation* $\mathbf{u}^2 = \mathbf{a}$ *over* $\mathbb{C}$ *is given by*

$$\mathbf{u} = \pm|\mathbf{a}|^{\frac{1}{2}} \exp\left(\mathrm{i}\frac{\omega}{2}\right) = \pm|\mathbf{a}|^{\frac{1}{2}} \left(\cos\left(\frac{\omega}{2}\right) + \mathrm{i}\sin\left(\frac{\omega}{2}\right)\right) .$$

As it is well-known, the two solutions of Lemma 2 are positioned on the circle of radius $|\mathbf{a}|^{\frac{1}{2}}$. See Figure 5.8 for an illustration. By Lemma 2 we obtain the following complex equivalent of Proposition 3.

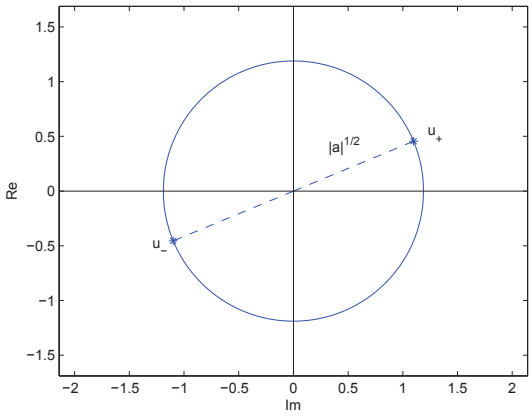


Figure 5.8: Visualization of the two solutions ($\mathbf{u}_+$ and $\mathbf{u}_-$) of the complex equation $\mathbf{u}^2 = 1 + i$.

Proposition 5. *The solutions of the Hermite interpolation Problem* 1 *are given by the following values for the coefficients* $\mathbf{u}_0$, $\mathbf{u}_1$, $\mathbf{u}_2$*:*

$$\begin{aligned} \mathbf{u}_1 &= \frac{1}{\sqrt{\gamma_1}}\mathbf{u}_5 + \frac{s_1}{\gamma_1}(g\mathbf{u}_3 + f\mathbf{u}_4)\,, \\ \mathbf{u}_0 &= \frac{1}{c_2 - 1}(c_2\mathbf{u}_3 - \mathbf{u}_4 + s_2\mathbf{u}_1)\,, \\ \mathbf{u}_2 &= \frac{1}{c_2 - 1}(-\mathbf{u}_3 + \mathbf{u}_4 - s_2\mathbf{u}_1)\,, \end{aligned} \tag{5.54}$$

where

$$\mathbf{u}_k = \pm|\mathbf{b}_k|^{\frac{1}{2}} \exp(i\frac{\omega_k}{2}) = \pm|\mathbf{b}_k|^{\frac{1}{2}}(\cos(\frac{\omega_k}{2}) + i\sin(\frac{\omega_k}{2}))\,,\ k = 3, 4, 5 \tag{5.55}$$

with

$$\mathbf{b}_3 = \frac{16s_1^4}{n_0}(\mathbf{p}_1-\mathbf{p}_0)\,,\ \mathbf{b}_4 = \frac{16s_1^4}{n_0}(\mathbf{p}_5-\mathbf{p}_4)\,,\ \mathbf{b}_5 = 4s_1^4(\mathbf{p}_4-\mathbf{p}_1)-\frac{d}{4n_2}(\mathbf{u}_3^2+\mathbf{u}_4^2)-\frac{e(c_2-1)}{2n_2}\mathbf{u}_3\mathbf{u}_4\,, \tag{5.56}$$

and $\omega_k = \arg(\mathbf{b}_k)$.

This notation helps us to introduce a concise complex representation for ATPH curves. We observe that we can write (5.47) as

$$\mathbf{w}(t) = \mathbf{w}_0\tilde{B}_0^2(t) + \mathbf{w}_1\tilde{B}_1^2(t) + \mathbf{w}_2\tilde{B}_2^2(t)\,, \tag{5.57}$$

where

$$\mathbf{w}_0 = \mathbf{u}(0) = \mathbf{u}_0+\mathbf{u}_2\,,\ \mathbf{w}_1 = \mathbf{u}_0+\tan\left(\frac{\alpha}{2}\right)\mathbf{u}_1+\mathbf{u}_2\,,\ \mathbf{w}_2 = \mathbf{u}(\alpha) = \mathbf{u}_0+\mathbf{u}_1\sin(\alpha)+\mathbf{u}_2\cos(\alpha)\,. \tag{5.58}$$

We thus write

$$\mathbf{x}'(t) = x'(t) + \mathrm{i}y'(t) = u^2(t) - v^2(t) + \mathrm{i}2u(t)v(t) = \mathbf{w}^2(t)\,. \tag{5.59}$$

We can thus reformulate Proposition 2 as follows

Proposition 6. *A planar, parametric curve over the mixed algebraic–trigonometric space U_5 expressed in terms of the normalized B-basis (5.5) as*

$$\mathbf{x}(t) = \sum_{i=0}^{5}\mathbf{p}_i B_i^5(t), \qquad t\in[0,\alpha], \tag{5.60}$$

is a (non-cuspidal) Algebraic–Trigonometric PH curve in the sense of Definition 6 if and only if its control points can be expressed in the form

$$\mathbf{p}_1 = \mathbf{p}_0 + \frac{n_0}{16s_1^4}\mathbf{w}_0^2, \tag{5.61}$$

$$\mathbf{p}_2 = \mathbf{p}_1 + \frac{n_0-6n_2}{8s_1^4}\mathbf{w}_0\mathbf{w}_1, \tag{5.62}$$

$$\mathbf{p}_3 = \mathbf{p}_2 + \frac{n_2}{4s_1^4}\Big((1+c_2)\mathbf{w}_1^2 + \mathbf{w}_0\mathbf{w}_2\Big), \tag{5.63}$$

$$\mathbf{p}_4 = \mathbf{p}_3 + \frac{n_0-6n_2}{8s_1^4}\mathbf{w}_1\mathbf{w}_2, \tag{5.64}$$

$$\mathbf{p}_5 = \mathbf{p}_4 + \frac{n_0}{16s_1^4}\mathbf{w}_2^2\,. \tag{5.65}$$

where $\mathbf{w}_0,\mathbf{w}_1,\mathbf{w}_2$ are complex values and s_1, c_2, n_0, n_2 denote the abbreviations in (5.3)-(5.4).

Proof. We substitute the function (5.57) into (5.59) and by integrating we obtain

$$\mathbf{x}(t) = \int \mathbf{x}'(s)\,ds = \mathbf{k}+\mathbf{a}_0+\mathbf{a}_1 t+\mathbf{a}_2\sin(t)+\mathbf{a}_3\cos(t)+\mathbf{a}_4\sin(2t)+\mathbf{a}_0\cos(2t) \tag{5.66}$$

where

$$\mathbf{a}_0 = -\frac{1}{2}\mathbf{u}_1\mathbf{u}_2\,, \quad \mathbf{a}_1 = \frac{1}{2}(\mathbf{u}_1^2+\mathbf{u}_2^2)+\mathbf{u}_0^2\,, \quad \mathbf{a}_2 = 2\mathbf{u}_0\mathbf{u}_2\,, \quad \mathbf{a}_3 = -2\mathbf{u}_0\mathbf{u}_1\,, \quad \mathbf{a}_4 = \frac{1}{4}(-\mathbf{u}_1^2+\mathbf{u}_2^2),$$

with

$$\mathbf{u}_0 = (1+c_2)\mathbf{w}_1 - \mathbf{w}_0 - \mathbf{w}_2\,, \quad \mathbf{u}_1 = s_2(\mathbf{w}_0 - \mathbf{w}_1)\,, \quad \mathbf{u}_2 = c_2(\mathbf{w}_0 - \mathbf{w}_1) + \mathbf{w}_2 - \mathbf{w}_1 \tag{5.67}$$

and $\mathbf{k}$ the integration constant. Expressing the basis functions of the functional space $U_5 = \{1,\, t,\, \sin(t),\, \cos(t),\, \sin(2t),\, \cos(2t)\}$ in terms of the normalized B–basis (5.5) and substituting these basis functions in (5.66) yields the expressions (5.61)-(5.65), where $\mathbf{p}_0 = \mathbf{k} + \mathbf{a}_0$.

□

We can remark that with this complex representation we obtain a more concise formulation of the control point expressions in (5.61)-(5.65). For this reason, in the following we will exclusively use this complex notation to describe ATPH curves properties and their applications. We first reformulate in the complex notation the arc-length of an ATPH curve and its key properties.

By (5.11) we have

$$\sigma(t) := \sqrt{(x'(t))^2 + (y'(t))^2} = |\mathbf{w}^2(t)|$$

and thus γ_{ij}^+, with $i, j \in \{0, 1, 2\}$ of the arc-length of an ATPH curve in (5.24) are given by

$$\gamma_{ij}^+ = \mathrm{Re}\left(\frac{\mathbf{u}_i}{\mathbf{u}_j}\right)|\mathbf{u}_j|^2, \qquad i, j \in \{0, 1, 2\}, \tag{5.68}$$

with $\mathbf{u}_i$, $i = 0, 1, 2$ in (5.67).

We continue by showing that, like polynomial PH curves, ATPH curves admit not only an exact representation of the arc length, but also of their offset curves. In the following, the offset curve of the ATPH curve $\mathbf{x}(t) = \sum_{i=0}^{5} \mathbf{p}_i B_i^5(t)$, $t \in [0, \alpha]$ at oriented distance d along the unit normal vector $\mathbf{n}(t)$ is denoted by $\mathbf{x}_d(t)$ and given by $\mathbf{x}_d(t) = \mathbf{x}(t) + d\,\mathbf{n}(t)$, $t \in [0, \alpha]$. The normal vector $\mathbf{n}$ has a rational ATPH representation over the extended Chebyshev space $U_4 := \tilde{U}_4 = \langle 1, \cos(t), \sin(t), \cos(2t), \sin(2t)\rangle$ since

$$\mathbf{n}(t) = \frac{-\mathrm{i}\mathbf{w}^2(t)}{\mathbf{w}(t)\overline{\mathbf{w}}(t)}\,,$$

where

$$\mathbf{w}^2(t) = \mathbf{w}_0^2 B_0^4(t) + \mathbf{w}_0\mathbf{w}_1 B_1^4(t) + \frac{\mathbf{w}_0\mathbf{w}_2 + (1+\cos(\alpha))\mathbf{w}_1^2}{2+\cos(\alpha)} B_2^4(t) + \mathbf{w}_1\mathbf{w}_2 B_3^4(t) + \mathbf{w}_2^2 B_4^4(t), \tag{5.69}$$

and

$$\begin{aligned}\mathbf{w}(t)\overline{\mathbf{w}}(t) \;=\; & \mathbf{w}_0\overline{\mathbf{w}}_0 B_0^4(t) + \tfrac{1}{2}(\mathbf{w}_0\overline{\mathbf{w}}_1 + \mathbf{w}_1\overline{\mathbf{w}}_0) B_1^4(t) + \tfrac{\mathbf{w}_0\overline{\mathbf{w}}_2 + 2(1+\cos(\alpha))\mathbf{w}_1\overline{\mathbf{w}}_1 + \mathbf{w}_2\overline{\mathbf{w}}_0}{2(2+\cos(\alpha))} B_2^4(t) \\ & + \tfrac{1}{2}(\mathbf{w}_1\overline{\mathbf{w}}_2 + \mathbf{w}_2\overline{\mathbf{w}}_1) B_3^4(t) + \mathbf{w}_2\overline{\mathbf{w}}_2 B_4^4(t)\,,\end{aligned}$$

with $B_i^4(t) = \tilde{B}_i^4(t)$, $i = 0, \ldots, 4$ from (5.1). We thus obtain

$$\mathbf{n}(t) = \frac{\sum_{i=0}^{4} \tilde{v}_i \tilde{\mathbf{p}}_i \, B_i^4(t)}{\sum_{j=0}^{4} \tilde{v}_j \, B_j^4(t)}, \qquad t \in [0, \alpha],$$

where

$$\begin{array}{lcllcl}
\tilde{v}_0 & = & \mathbf{w}_0\overline{\mathbf{w}}_0 = |\mathbf{w}_0|^2, & \tilde{v}_0\tilde{\mathbf{p}}_0 & = & -\mathrm{i}\mathbf{w}_0^2, \\
\tilde{v}_1 & = & \frac{1}{2}(\mathbf{w}_0\overline{\mathbf{w}}_1 + \mathbf{w}_1\overline{\mathbf{w}}_0), & \tilde{v}_1\tilde{\mathbf{p}}_1 & = & -\mathrm{i}\mathbf{w}_0\mathbf{w}_1, \\
\tilde{v}_2 & = & \frac{\mathbf{w}_0\overline{\mathbf{w}}_2 + 2(1+\cos(\alpha))\mathbf{w}_1\overline{\mathbf{w}}_1 + \mathbf{w}_2\overline{\mathbf{w}}_0}{2(2+\cos(\alpha))}, & \tilde{v}_2\tilde{\mathbf{p}}_2 & = & -\mathrm{i}\,\frac{\mathbf{w}_0\mathbf{w}_2 + (1+\cos(\alpha))\mathbf{w}_1^2}{2+\cos(\alpha)}, \\
\tilde{v}_3 & = & \frac{1}{2}(\mathbf{w}_1\overline{\mathbf{w}}_2 + \mathbf{w}_2\overline{\mathbf{w}}_1), & \tilde{v}_3\tilde{\mathbf{p}}_3 & = & -\mathrm{i}\mathbf{w}_1\mathbf{w}_2, \\
\tilde{v}_4 & = & \mathbf{w}_2\overline{\mathbf{w}}_2 = |\mathbf{w}_2|^2, & \tilde{v}_4\tilde{\mathbf{p}}_4 & = & -\mathrm{i}\mathbf{w}_2^2.
\end{array}$$

Now, since $\mathbf{x}(t) = \sum_{i=0}^{5} \mathbf{p}_i B_i^5(t)$, we can define the offset curve $\mathbf{x}_d(t)$ as a rational algebraic-trigonometric curve in terms of the normalized B-basis of the extended Chebyshev space obtained from the multiplication of U_4 and U_5. Recall that being

$$\tilde{U}_2 = \langle 1, \sin(t), \cos(t) \rangle, \qquad \tilde{U}_4 = \langle 1, \sin(t), \cos(t), \sin(2t), \cos(2t) \rangle$$

and

$$\tilde{U}_8 = \langle 1, \sin(t), \cos(t), \sin(2t), \cos(2t), \sin(3t), \cos(3t)\ \sin(4t), \cos(4t) \rangle,$$

then the following relationships are satisfied:

$$\tilde{U}_2 * \tilde{U}_2 = \tilde{U}_4, \qquad \tilde{U}_4 * \tilde{U}_4 = \tilde{U}_8.$$

Therefore, being $U_5 = \langle 1, t, \sin(t), \cos(t), \sin(2t), \cos(2t) \rangle$ we have

$$\hat{U} := \tilde{U}_4 * U_5 = \langle \tilde{U}_8, t, t\sin(t), t\cos(t), t\sin(2t), t\cos(2t) \rangle.$$

The offset curve $\mathbf{x}_d(t)$ of the ATPH curve $\mathbf{x}(t)$ is thus a rational algebraic-trigonometric curve of the form

$$\mathbf{x}_d(t) = \mathbf{x}(t) + d\,\mathbf{n}(t) = \sum_{i=0}^{5} \mathbf{p}_i B_i^5(t) + d\,\frac{\sum_{i=0}^{4} \tilde{v}_i \tilde{\mathbf{p}}_i \, B_i^4(t)}{\sum_{j=0}^{4} \tilde{v}_j \, B_j^4(t)} = \frac{\sum_{i=0}^{13} \hat{v}_i \, \hat{\mathbf{p}}_i \, \hat{B}_i^{13}(t)}{\sum_{j=0}^{13} \hat{v}_j \hat{B}_j^{13}(t)}, \qquad t \in I_\alpha,$$

where $\hat{B}_i^{13}$, $i = 0, ..., 13$ are the basis functions of the space $\hat{U}$. This normalized B–basis of $\hat{U}$ will be presented in a forthcoming article.

5.6 C^1 Hermite interpolation problem: reformulation and analysis

By the notation of Proposition 6, for solving the Hermite interpolation *Problem 1*, the complex values $\mathbf{w}_0$,$\mathbf{w}_1$,$\mathbf{w}_2$ have to be determined in order to obtain the control points in the form (5.61)-(5.65).

Due to the following properties of AT-Bézier curves built-upon the normalized B-basis B_i^5, $i = 0, \cdots, 5$,

$$\mathbf{x}(0) = \mathbf{p}_0, \quad \mathbf{x}(\alpha) = \mathbf{p}_5, \quad \mathbf{x}'(0) = \frac{16s_1^4}{n_0}(\mathbf{p}_1 - \mathbf{p}_0), \quad \mathbf{x}'(\alpha) = \frac{16s_1^4}{n_0}(\mathbf{p}_5 - \mathbf{p}_4),$$

with s_1 in (5.3) and n_0 in (5.4), this problem can be obviously regarded as a C^1 Hermite interpolation problem to prescribed end points $\mathbf{p}_0$, $\mathbf{p}_5$ and tangent vectors at these end points. Hereinafter the tangent vectors at $\mathbf{p}_0$, $\mathbf{p}_5$ will be denoted by $\mathbf{d}_0, \mathbf{d}_2$, respectively, since (as we will see later) they are directly related to the values of $\mathbf{w}_0$,$\mathbf{w}_2$. In analogy to Propositions 3 and 5 we obtain the following simpler version.

Proposition 7. *The solutions of the Hermite interpolation Problem* 1 *in terms of the complex values* $\mathbf{w}_0, \mathbf{w}_1, \mathbf{w}_2$ *are given by*

$$\begin{aligned}
\mathbf{w}_0 &= \pm|\mathbf{d}_0|^{\frac{1}{2}} \exp\left(\mathrm{i}\frac{\omega_0}{2}\right) = \pm|\mathbf{d}_0|^{\frac{1}{2}} \left(\cos\left(\frac{\omega_0}{2}\right) + \mathrm{i}\sin\left(\frac{\omega_0}{2}\right)\right), \\
\mathbf{w}_2 &= \pm|\mathbf{d}_2|^{\frac{1}{2}} \exp\left(\mathrm{i}\frac{\omega_2}{2}\right) = \pm|\mathbf{d}_2|^{\frac{1}{2}} \left(\cos\left(\frac{\omega_2}{2}\right) + \mathrm{i}\sin\left(\frac{\omega_2}{2}\right)\right), \\
\mathbf{w}_1 &= \pm|\mathbf{d}_1|^{\frac{1}{2}} \left(\cos\left(\frac{\omega_1}{2}\right) + \mathrm{i}\sin\left(\frac{\omega_1}{2}\right)\right) - \frac{n_0 - 6n_2}{4n_2(1+c_2)}(\mathbf{w}_0 + \mathbf{w}_2),
\end{aligned} \tag{5.70}$$

where

$$\begin{aligned}
\mathbf{d}_0 &= \tfrac{16s_1^4}{n_0}(\mathbf{p}_1 - \mathbf{p}_0), \\
\mathbf{d}_2 &= \tfrac{16s_1^4}{n_0}(\mathbf{p}_5 - \mathbf{p}_4), \\
\mathbf{d}_1 &= \tfrac{1}{1+c_2}\left(\tfrac{4s_1^4}{n_2}(\mathbf{p}_4 - \mathbf{p}_1) + \tfrac{(n_0-6n_2)^2}{16n_2^2(1+c_2)}(\mathbf{w}_0 + \mathbf{w}_2)^2 - \mathbf{w}_0\mathbf{w}_2\right),
\end{aligned}$$

$\omega_k = \arg(\mathbf{d}_k)$, $k = 0, 1, 2$ *and* s_1, c_2, n_0, n_2 *are the abbreviations in (5.3)-(5.4).*

Proof. For completeness we provide the proof which follows the same line of reasoning as the one of Proposition 3. By applying Lemma 2 to the equations (5.61) and (5.65) we obtain the expressions of $\mathbf{w}_0$ and $\mathbf{w}_2$ in (5.70). Then, writing

$$\mathbf{p}_4 - \mathbf{p}_1 = (\mathbf{p}_4 - \mathbf{p}_3) + (\mathbf{p}_3 - \mathbf{p}_2) + (\mathbf{p}_2 - \mathbf{p}_1),$$

and substituting from (5.61)-(5.65) we obtain

$$\frac{8s_1^4}{n_0 - 6n_2}(\mathbf{p}_4 - \mathbf{p}_1) = \mathbf{w}_1\mathbf{w}_2 + \frac{2n_2}{n_0 - 6n_2}((1+c_2)\mathbf{w}_1^2 + \mathbf{w}_0\mathbf{w}_2) + \mathbf{w}_0\mathbf{w}_1. \tag{5.71}$$

By the change of variable

$$\widetilde{\mathbf{w}}_1 = \mathbf{w}_1 + \frac{n_0 - 6n_2}{4n_2(1+c_2)}(\mathbf{w}_0 + \mathbf{w}_2), \tag{5.72}$$

equation (5.71) becomes

$$\widetilde{\mathbf{w}}_1^2 = \frac{4s_1^4}{n_2(1+c_2)}(\mathbf{p}_4 - \mathbf{p}_1) + \frac{(n_0 - 6n_2)^2}{16n_2^2(1+c_2)^2}(\mathbf{w}_0 + \mathbf{w}_2)^2 - \frac{1}{1+c_2}\mathbf{w}_0\mathbf{w}_2. \tag{5.73}$$

Applying Lemma 2 to (5.73) we find

$$\widetilde{\mathbf{w}}_1 = \pm|\mathbf{d}_1|^{\frac{1}{2}} \exp\left(\mathrm{i}\frac{\omega_1}{2}\right) = \pm|\mathbf{d}_1|^{\frac{1}{2}} \left(\cos\left(\frac{\omega_1}{2}\right) + \mathrm{i}\sin\left(\frac{\omega_1}{2}\right)\right) \tag{5.74}$$

with

$$\mathbf{d}_1 = \frac{4s_1^4}{n_2(1+c_2)}(\mathbf{p}_4 - \mathbf{p}_1) + \frac{(n_0 - 6n_2)^2}{16n_2^2(1+c_2)^2}(\mathbf{w}_0 + \mathbf{w}_2)^2 - \frac{1}{1+c_2}\mathbf{w}_0\mathbf{w}_2.$$

Finally, by substituting (5.74) in (5.72) the expression of $\mathbf{w}_1$ in (5.70) is obtained. □

Remark 1. *In the expressions (5.70) we have three independent signs. Thus, we can construct eight ATPH interpolants. We remark that if we take (-$\mathbf{w}_0$, -$\mathbf{w}_1$, -$\mathbf{w}_2$) or ($\mathbf{w}_0$, $\mathbf{w}_1$, $\mathbf{w}_2$) we obtain the same interpolant. Moreover, we observe that in the expressions (5.61) - (5.65) we have homogeneous quadratic forms in the coefficients of $\mathbf{w}(t)$. We can thus fix the sign in any one of the three expressions (5.70) and obtain only four distinct interpolants.*

Remark 2. *We further observe that the free parameter α acts as a shape parameter for the ATPH interpolants. This can be clearly seen in Figure 5.9 where different ATPH interpolants to the same end points and associated end derivatives (all corresponding to a positive choice of the signs of $\mathbf{w}_2$ and $\mathbf{w}_1$) are displayed together with the standard polynomial PH quintic solving the same C^1 Hermite problem. We note that for increasing values of α the curves become longer and longer. This seems to be in contradiction to the behaviour of the AT-Bézier curves which for $\alpha \to 2\pi$ become shorter and shorter. The reason for this fact simply lies in the dependency on α of the control points of the ATPH curve according to equations (5.61)-(5.65).*

Figure 5.10 shows the behavior of all four possible families of ATPH curves interpolating given end points and end derivatives, for different choices of $\alpha \in (0, \pi)$. We obtain the four families of ATPH interpolants from the sign choices $++$, $+-$, $-+$, $--$ in the expressions of $\mathbf{w}_2$ and $\mathbf{w}_1$. In the following we will always refer to the four families of ATPH interpolants by pointing out these sign combinations.

We conclude by observing that the arc-length of the ATPH curve (corresponding to the evaluation of the function in (5.24) between 0 and α) has the expression

$$\begin{aligned} S_\alpha = \int_0^\alpha \sigma(t)dt \;&=\; 2\gamma_{01} + \tfrac{1}{2}\gamma_{12} + (\gamma_{00} + \tfrac{1}{2}(\gamma_{11} + \gamma_{22}))\,\alpha + 2\gamma_{02}\sin(\alpha) + \\ &-\; 2\gamma_{01}\cos(\alpha) + \tfrac{1}{4}(\gamma_{22} - \gamma_{11})\sin(2\alpha) - \tfrac{1}{2}\gamma_{12}\cos(2\alpha) \end{aligned} \tag{5.75}$$

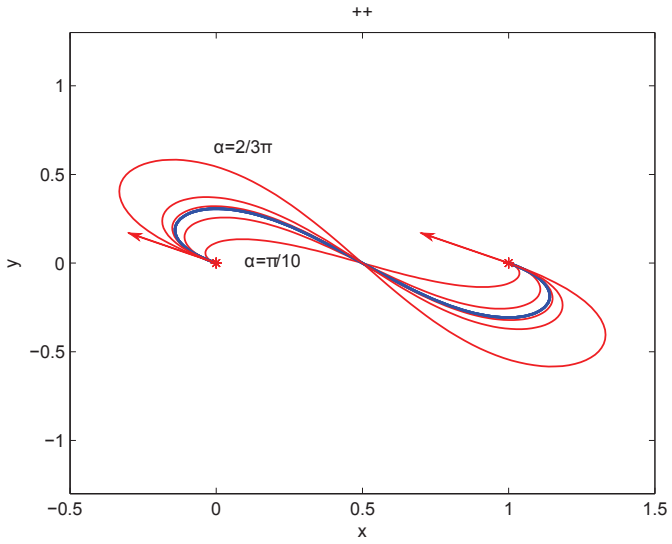


Figure 5.9: Comparison of ATPH interpolants (red curves) to given end points $\mathbf{p}_0 = 0$, $\mathbf{p}_5 = 1$ and associated end derivatives $\mathbf{d}_0 = \mathbf{d}_2 = -3+\mathrm{i}\frac{3(4+\sqrt{5})}{11}$ (here plotted with a scale factor of $\frac{1}{5}$ to fit into the picture), obtained for a positive choice of $\mathbf{w}_2$ and $\mathbf{w}_1$ $(++)$ and increasing values of $\alpha \in (0, \pi)$ (respectively $\alpha = \frac{\pi}{10}, \frac{\pi}{4}, \frac{\pi}{3}, \frac{2}{5}\pi, \frac{2}{3}\pi$), with the $(++)$ PH quintic interpolant (blue curve) solving the same C^1 Hermite interpolation problem.

with γ_{ij} in (5.68), and thus turns out to be monotonically increasing for increasing values of α, as clearly shown in Figure 5.11.

By observing numerous examples, all exhibiting the same qualitative behaviour as the ones in Figures 5.9 and 5.10, we can conclude that the most interesting ATPH interpolants for practical use are obtained for $\alpha \in (0, \frac{\pi}{2})$. Since, as already shown in Section 5.2, the fast visualization algorithm of ATPH curves exists only for values of α in the same range, hereinafter we will restrict our attention to the case $\alpha \in (0, \frac{\pi}{2})$.

5.6.1 How to identify the "best" ATPH Hermite interpolant

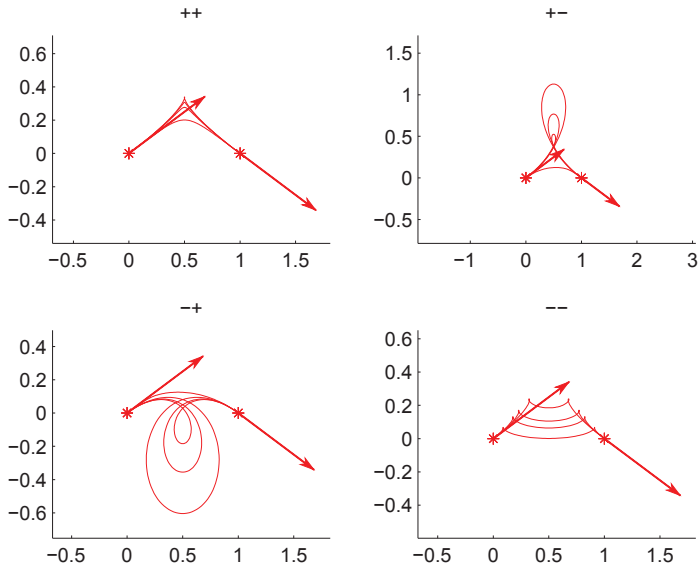
As in [Farouki 1995] we introduce the rotation index

$$\mathcal{R} = \frac{1}{2\pi}\int_0^\alpha \kappa(t)|\mathbf{x}'(t)|dt\,,$$

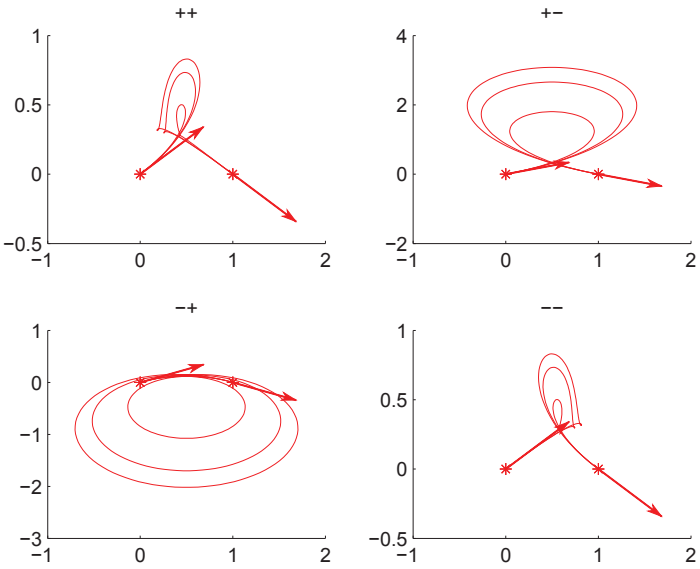
and the absolute rotation index

$$\mathcal{R}_{abs} = \frac{1}{2\pi}\int_0^\alpha |\kappa(t)||\mathbf{x}'(t)|dt\,,$$

as these are the quantities whose minimization allows us to identify the "best" ATPH Hermite interpolant. Since $\kappa(t) = \frac{\mathrm{Im}(\bar{\mathbf{x}}'(t)\mathbf{x}''(t))}{|\mathbf{x}'(t)|^3}$ we have $\kappa(t)|\mathbf{x}'(t)| = \frac{\mathrm{Im}(\bar{\mathbf{x}}'(t)\mathbf{x}''(t))}{|\mathbf{x}'(t)|^2}$, and considering (5.59) we get $\kappa(t)|\mathbf{x}'(t)| = 2\,\mathrm{Im}\left(\frac{\mathbf{w}'(t)}{\mathbf{w}(t)}\right)$. The rotation index and the absolute rotation index can thus be rewritten in the equivalent form



(a) $\alpha = \frac{\pi}{5}, \frac{\pi}{3}, \frac{2}{5}\pi, \frac{\pi}{2}$



(b) $\alpha = \frac{2}{3}\pi, \frac{5}{6}\pi, \frac{9}{10}\pi$

Figure 5.10: The four families of ATPH interpolants to given end points $\mathbf{p}_0 = 0$, $\mathbf{p}_5 = 1$ and end derivatives $\mathbf{d}_0 = 6a + \mathrm{i}3a$, $\mathbf{d}_2 = 6a - \mathrm{i}3a$ with $a = \frac{4+\sqrt{5}}{11}$ (here plotted with a scale factor of $\frac{1}{5}$ to fit into the picture), obtained for different values of $\alpha \in (0, \pi)$.

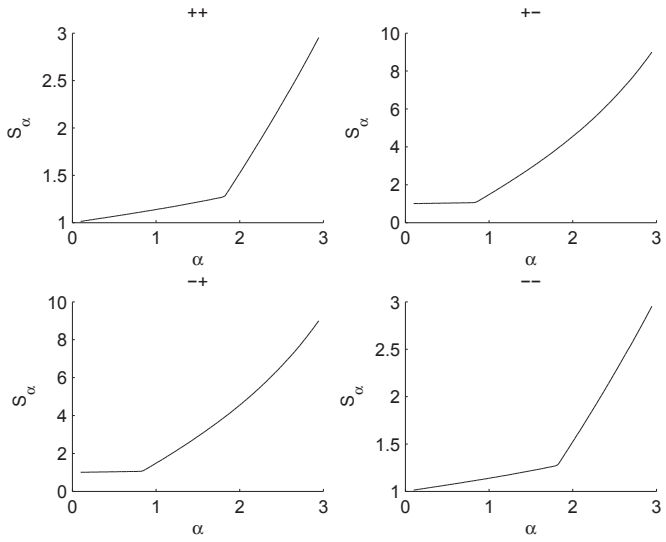


Figure 5.11: Behaviour of the arc-length of the ATPH curves in Figure 5.10 for $\alpha \in (0, \pi)$.

$$\mathcal{R} = \frac{1}{2\pi \mathrm{i}} \int_0^\alpha \frac{\mathbf{w}'(t)\bar{\mathbf{w}}(t) - \mathbf{w}(t)\bar{\mathbf{w}}'(t)}{\mathbf{w}(t)\bar{\mathbf{w}}(t)} dt\,, \quad \mathcal{R}_{abs} = \frac{1}{2\pi} \int_0^\alpha \frac{|\mathbf{w}'(t)\bar{\mathbf{w}}(t) - \mathbf{w}(t)\bar{\mathbf{w}}'(t)|}{\mathbf{w}(t)\bar{\mathbf{w}}(t)} dt. \tag{5.76}$$

Recalling that $\mathbf{w}(t) = u(t) + \mathrm{i}v(t)$, we can also rewrite (5.76) in the real formulation as follows

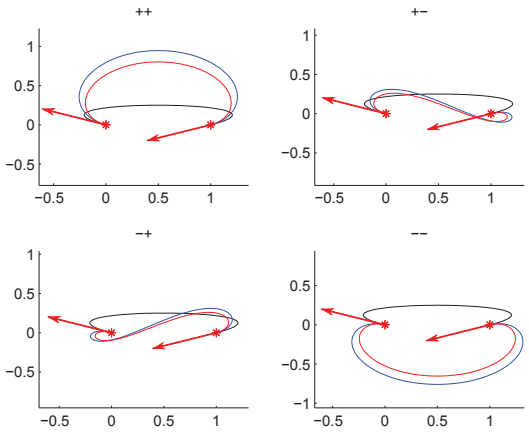
$$\mathcal{R} = \frac{1}{\pi} \int_0^\alpha \frac{(u(t)v'(t) - u'(t)v(t))}{u(t)^2 + v(t)^2} dt\,, \quad \mathcal{R}_{abs} = \frac{1}{\pi} \int_0^\alpha \frac{|(u(t)v'(t) - u'(t)v(t))|}{u(t)^2 + v(t)^2} dt.$$

Now, if we define $z(t) := \frac{v(t)}{u(t)} \in \mathbb{R}$, we can apply [Farouki 1995, Lemma 3] thus obtaining

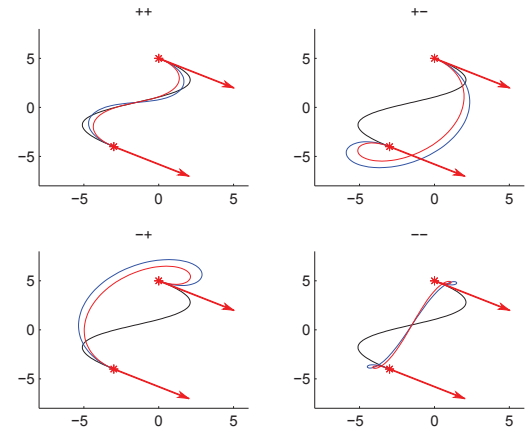
$$\begin{aligned} \mathcal{R}^{ATPH} &= \tfrac{1}{\pi}\Big(\arctan(z(\alpha)) - \arctan(z(0))\Big) - \mathcal{I}_0^\alpha(z(t)), \\ \mathcal{R}^{ATPH}_{abs} &= \tfrac{1}{\pi}\left(\sum_{j=0}^{n-1} \mathrm{sign}_{I_j}(\xi(t)) \Big(\arctan(z(t_{j+1})) - \arctan(z(t_j))\Big)\right) + \mathcal{S}(u(t))\,, \end{aligned}$$

where

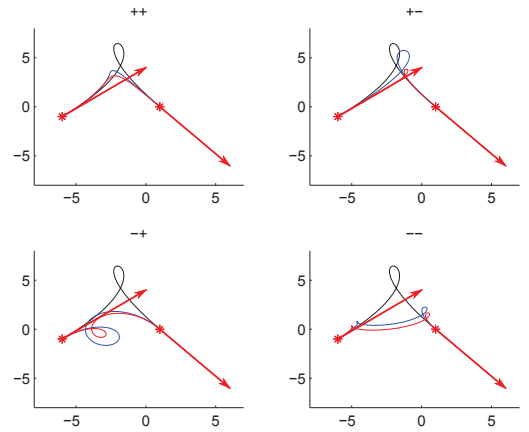
- $\mathcal{I}_0^\alpha(z(t))$ is the Cauchy index of $z(t)$;
- $\xi(t) = u(t)v'(t) - u'(t)v(t)$;
- t_j, $j = 0, \ldots, n$ are the zeros of $\xi(t)$ in the interval $[0, \alpha]$ with $t_0 = 0$ and $t_n = \alpha$;
- $\mathrm{sign}_{I_j}(\xi(t))$ denotes the sign of $\xi(t)$ on the interval $I_j = [t_j, t_{j+1}]$, $j = 0, \ldots, n-1$;
- $\mathcal{S}(u(t))$ is the number of zeros of $u(t)$ in the interval $[0, \alpha]$.



(a) $\mathbf{p}_0 = 0$, $\mathbf{p}_5 = 1$, $\mathbf{d}_0 = -3 + \mathrm{i}$, $\mathbf{d}_2 = -3 - \mathrm{i}$



(b) $\mathbf{p}_0 = 5\mathrm{i}$, $\mathbf{p}_5 = -3 - 4\mathrm{i}$, $\mathbf{d}_0 = \mathbf{d}_2 = 25 - 15\mathrm{i}$



(c) $\mathbf{p}_0 = -6 - \mathrm{i}$, $\mathbf{p}_5 = 1$, $\mathbf{d}_0 = 30 + 25\mathrm{i}$, $\mathbf{d}_2 = 25 - 30\mathrm{i}$

Figure 5.12: Comparison of the ATPH interpolant obtained for $\alpha = \frac{\pi}{4}$ (red curve), with the corresponding PH quintic interpolant (blue curve) and the unique cubic interpolant (black curve) to the given end points $\mathbf{p}_0$,$\mathbf{p}_5$ and associated derivatives $\mathbf{d}_0$,$\mathbf{d}_2$ (here plotted with a scale factor of $\frac{1}{5}$ to fit into the picture).

The examples in Figure 5.12 confirm how the "best" solution to the Hermite interpolation problem can be identified as the one with the smallest absolute rotation index (see Table 5.1).

R_{abs}^{ATPH}	(a)	(b)	(c)
$(++)$	0.8976	0.7280	0.3589
$(+-)$	1.1515	1	0.7542
$(-+)$	1.1515	1	1.25
$(--)$	1.024	1.5566	1.75

Table 5.1: ATPH absolute rotation indices for the examples in Figure 5.12.

The purpose of Table 5.2 and Figure 5.13 is to show that ATPH interpolants compare favorably with their polynomial counterpart represented by the well-known PH quintics. In fact, for the chosen set of end points and associated end derivatives, we can see that all the four possible PH curves exhibit undesired self-intersections, although the standard cubic interpolant does not. On the other hand, with the $(++)$ ATPH interpolant obtained, for instance, with the choice $\alpha = \frac{\pi}{5}$, we can eliminate this unwanted self-intersection as in the cubic curve case.

We conclude by observing that, for certain Hermite data, the concept of the "best" interpolant as the interpolant with the smallest absolute rotation index is ambiguous. An example of this phenomenon is represented in Figure 5.14 where we have two ATPH curves (the ones having signs $(+-)$ and $(-+)$) with the same smallest absolute rotation index (see Table 5.3), none of them being optimal. In fact, the most similar to the standard cubic interpolant is still the one having signs $(++)$. The criterion based on the absolute rotation index requires us to construct all four interpolants, and then compare a quantitative shape measure. In practice, it is preferable to have the ability to construct directly the best interpolant only, as the one being free of loops. In the following we show that this construction is possible if we restrict our attention to cases with "reasonable" Hermite data. Given arbitrary end points $\mathbf{p}_0$ and $\mathbf{p}_5$, we can define "reasonable" Hermite data requiring the associated end derivatives, $\mathbf{d}_0$ and $\mathbf{d}_2$, to vary in a prescribed domain. In particular, the

	R_{abs}^{PH}	R_{abs}^{ATPH}
$(++)$	1.395	0.7270
$(+-)$	1	1
$(-+)$	1	1
$(--)$	1.5504	1.8959

Table 5.2: Comparison of PH absolute rotation indices and ATPH absolute rotation indices for the example in Figure 5.13.

	R_{abs}^{PH}	R_{abs}^{ATPH}
$(++)$	1.1009	1.0831
$(+-)$	1	1
$(-+)$	1	1
$(--)$	1.2266	1.2199

Table 5.3: Comparison of PH absolute rotation indices and ATPH absolute rotation indices for the example in Figure 5.14.

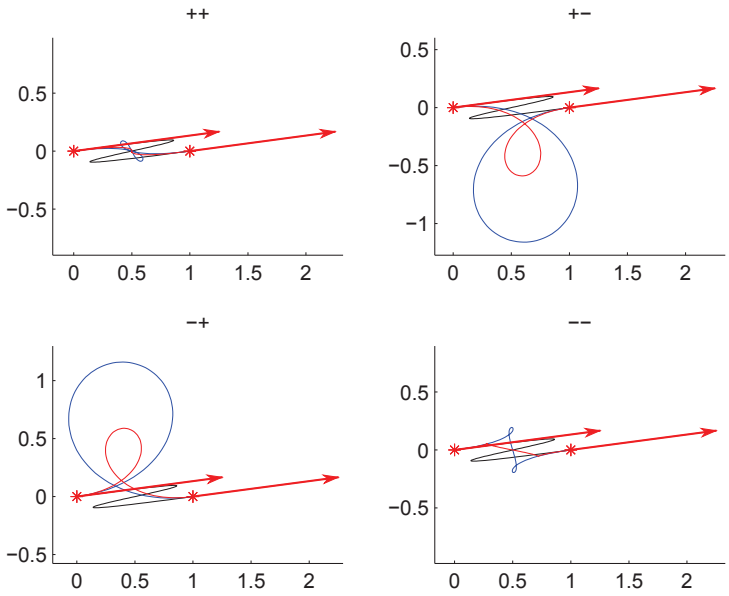


Figure 5.13: The four ATPH interpolants (red curves) to the end points $\mathbf{p}_0 = 0$, $\mathbf{p}_5 = 1$ and associated end derivatives $\mathbf{d}_0 = \mathbf{d}_2 = 7.5 + \mathrm{i}$ (here plotted with a scale factor of $\frac{1}{6}$ to fit into the picture), obtained for $\alpha = \frac{\pi}{5}$. The corresponding PH quintic interpolant and the unique cubic interpolant to the same data are depicted in blue and black, respectively.

domain for "reasonable" end-derivatives $\mathbf{d}_0$, $\mathbf{d}_2$ is defined as the open half-disk

$$\mathcal{D}_\alpha = \left\{ \mathbf{d} \in \mathbb{C} \,:\, \mathrm{Re}(\mathbf{d}(\overline{\mathbf{p}_5 - \mathbf{p}_0})) > 0 \text{ and } |\mathbf{d}| < \frac{3}{\alpha} |\mathbf{p}_5 - \mathbf{p}_0| \right\} \quad \text{with} \quad \alpha \in \left(0, \frac{\pi}{2}\right). \tag{5.77}$$

This choice of the domain $\mathcal{D}_\alpha$ (see Figure 5.15 for an illustration) does not constitute a limitation for the practical applications we have in mind. Moreover, for such "reasonable" Hermite data, we can guarantee that the cubic interpolant always has a well-behaved tangent variation, as shown in [Moon 2001, Proposition 1]. In the following proposition we show that also the (++) ATPH interpolant enjoys this property whenever $\mathbf{d}_0$, $\mathbf{d}_2$ are chosen in $\mathcal{D}_\alpha$.

Proposition 8. *If the Hermite end derivatives* $\mathbf{d}_0$ *and* $\mathbf{d}_2$ *lie within the domain* $\mathcal{D}_\alpha$*, the hodograph* $\mathbf{x}'(t)$ *of the (++) ATPH interpolant* $\mathbf{x}(t)$ *remains inside* $\mathcal{D}_\alpha$ *for all* $t \in [0, \alpha]$ *and* $\alpha \in (0, \frac{\pi}{2})$.

Proof. Without loss of generality, we choose the end points $\mathbf{p}_0 = 0$ and $\mathbf{p}_5 = 1$. We then recall that, for a general ATPH interpolant $\mathbf{x}(t)$, the hodograph is given by (5.69), *i.e.*

$$\mathbf{x}'(t) = \mathbf{w}^2(t) = \Big(\mathbf{w}_0 B_0^2(t) + \mathbf{w}_1 B_1^2(t) + \mathbf{w}_2 B_2^2(t) \Big)^2 .$$

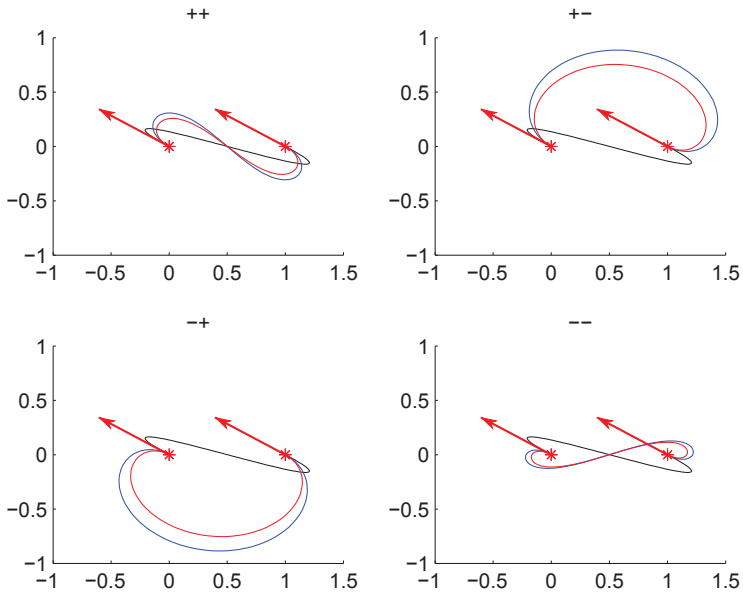


Figure 5.14: The four ATPH interpolants (red curves) to the end points $\mathbf{p}_0 = 0$, $\mathbf{p}_5 = 1$ and associated end derivatives $\mathbf{d}_0 = \mathbf{d}_2 = -3 + \mathrm{i}$ (here plotted with a scale factor of $\frac{1}{5}$ to fit into the picture), obtained for $\alpha = \frac{\pi}{4}$. The corresponding PH quintic interpolant and the unique cubic interpolant to the same data are depicted in blue and black, respectively.

If we consider the (++) ATPH interpolant from (5.70), due to the fact that $\mathbf{x}'(0) = \mathbf{w}^2(0) = \mathbf{w}_0^2 = \mathbf{d}_0$ and $\mathbf{x}'(\alpha) = \mathbf{w}^2(\alpha) = \mathbf{w}_2^2 = \mathbf{d}_2$ with $\mathbf{d}_0, \mathbf{d}_2 \in \mathcal{D}_\alpha$, we have

$$\begin{aligned} \mathbf{w}_0 &= \sqrt{\mathbf{d}_0}, \\ \mathbf{w}_2 &= \sqrt{\mathbf{d}_2}, \\ \mathbf{w}_1 &= |\mathbf{d}_1|^{\frac{1}{2}}(\cos(\tfrac{\omega_1}{2}) + \mathrm{i}\sin(\tfrac{\omega_1}{2})) - \tfrac{n_0 - 6n_2}{4n_2(1+c_2)}(\mathbf{w}_0 + \mathbf{w}_2) = g(\mathbf{w}_0, \mathbf{w}_2), \end{aligned} \tag{5.78}$$

where $\sqrt{\mathbf{d}_i}$ for $i = 0, 2$ denotes the principal value of the complex square root, *i.e.*, with positive real part, and

$$\mathbf{d}_1 = \frac{1}{1+c_2}\left(\frac{4s_1^4}{n_2}(\mathbf{p}_4 - \mathbf{p}_1) + \frac{(n_0 - 6n_2)^2}{16n_2^2(1+c_2)}(\mathbf{w}_0 + \mathbf{w}_2)^2 - \mathbf{w}_0\mathbf{w}_2\right) \quad \text{with} \quad \omega_1 = \arg(\mathbf{d}_1).$$

To show that $\mathbf{x}'(t)$ is contained within $\mathcal{D}_\alpha$ for all $t \in [0, \alpha]$ and $\alpha \in (0, \frac{\pi}{2})$, we can verify that for all $t \in [0, \alpha]$ $\mathbf{w}(t)$ lies within the open set

$$\sqrt{\mathcal{D}_\alpha} = \left\{ \mathbf{z} \in \mathbb{C} \,:\, \mathrm{Re}(\mathbf{z}) > |\mathrm{Im}(\mathbf{z})| \;\text{ and }\; |\mathbf{z}| < \sqrt{\frac{3}{\alpha}} \right\}, \qquad \text{with} \quad \alpha \in \left(0, \frac{\pi}{2}\right).$$

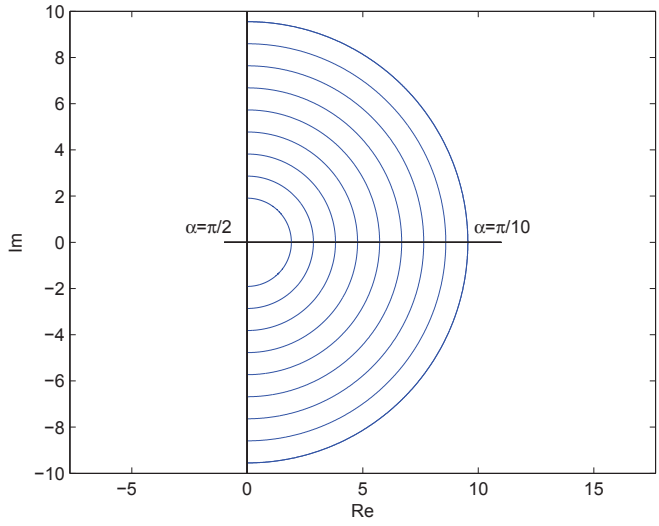


Figure 5.15: Visualization of the domain $\mathcal{D}_\alpha$ from (5.77) for the fixed end points $\mathbf{p}_0 = 0$, $\mathbf{p}_5 = 1$.

Following the proof in [Moon 2001, Proposition 2], we apply the de Casteljau-like algorithm previously described in Subsection 5.1.2 for the evaluation of trigonometric Bézier curves over the space $\tilde{U}_2$, to split $\mathbf{w}(t)$ in correspondence of the parameter $t = \frac{\alpha}{2}$. From equation (5.2) we get that the control points of the two subcurves $\mathbf{w}_l(t)$, $\mathbf{w}_r(t)$ joining at such location are

$$\mathbf{w}_l(t) = \sqrt{\mathbf{x}_l'(t)} = \mathbf{w}_0\tilde{B}_0^2(t) + \mathbf{m}_{01}\tilde{B}_1^2(t) + \frac{1}{2}(\mathbf{m}_{01} + \mathbf{m}_{12})\tilde{B}_2^2(t),$$

$$\mathbf{w}_r(t) = \sqrt{\mathbf{x}_r'(t)} = \frac{1}{2}(\mathbf{m}_{01} + \mathbf{m}_{12})\tilde{B}_0^2(t) + \mathbf{m}_{12}\tilde{B}_1^2(t) + \mathbf{w}_2\tilde{B}_2^2(t),$$

where

$$\mathbf{m}_{01} = \frac{\cos(\frac{\alpha}{2})\mathbf{w}_1 + \mathbf{w}_0}{\cos(\frac{\alpha}{2}) + 1} \qquad \text{and} \qquad \mathbf{m}_{12} = \frac{\cos(\frac{\alpha}{2})\mathbf{w}_1 + \mathbf{w}_2}{\cos(\frac{\alpha}{2}) + 1}. \tag{5.79}$$

By the convexity of $\sqrt{\mathcal{D}_\alpha}$ and the convex hull property of the generalized Bézier curves built-upon the normalized B-basis $\tilde{B}_i^2(t)$, $i = 0, 1, 2$, it suffices to show that for arbitrary values of $\mathbf{w}_0$ and $\mathbf{w}_2$ in $\sqrt{\mathcal{D}_\alpha}$, $\mathbf{m}_{01}$ and $\mathbf{m}_{12}$ are also contained in $\sqrt{\mathcal{D}_\alpha}$. To this purpose, we use the *Minkowski geometric algebra* of complex sets [Farouki 2000b, Farouki 2000a] and a generalization of Minkowski sums and products by the concept of an *implicitly-defined set* [Farouki 2000a]

$$\mathcal{A} \odot_f \mathcal{B} = \{f(\mathbf{a}, \mathbf{b}) \,:\, \mathbf{a} \in \mathcal{A}, \mathbf{b} \in \mathcal{B}\},$$

corresponding to a given bivariate function $f(\mathbf{a}, \mathbf{b})$. From (5.78) and (5.79) we have that the Bézier coefficients $\mathbf{m}_{01}$ and $\mathbf{m}_{12}$ can be regarded as complex-valued

functions of values $\mathbf{w}_0$, $\mathbf{w}_2$ chosen from $\sqrt{\mathcal{D}_\alpha}$, *i.e.*

$$\mathbf{m}_{01} = f_0(\mathbf{w}_0, \mathbf{w}_2) = \frac{\cos(\frac{\alpha}{2})g(\mathbf{w}_0, \mathbf{w}_2) + \mathbf{w}_0}{\cos(\frac{\alpha}{2}) + 1}, \quad \mathbf{m}_{12} = f_1(\mathbf{w}_0, \mathbf{w}_2) = \frac{\cos(\frac{\alpha}{2})g(\mathbf{w}_0, \mathbf{w}_2) + \mathbf{w}_2}{\cos(\frac{\alpha}{2}) + 1}.$$

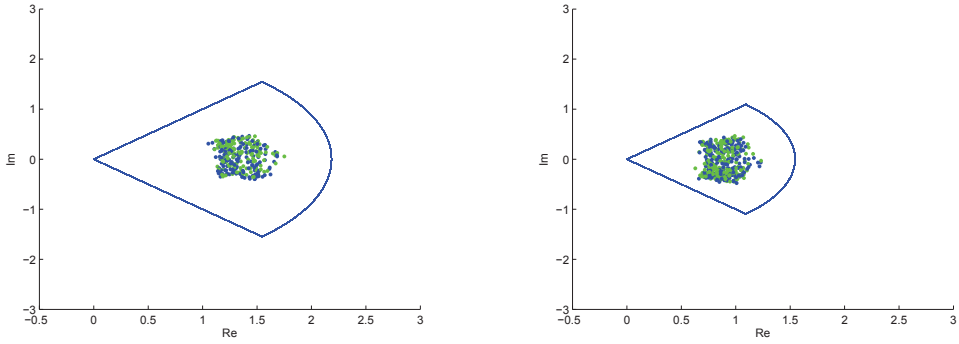
Therefore, the control points $\mathbf{m}_{01}$ and $\mathbf{m}_{12}$ lie in the region defined by

$$\sqrt{\mathcal{D}_\alpha} \odot_{f_i} \sqrt{\mathcal{D}_\alpha} = \{f_i(\mathbf{z}_1, \mathbf{z}_2) \,:\, \mathbf{z}_1 \in \sqrt{\mathcal{D}_\alpha},\, \mathbf{z}_2 \in \sqrt{\mathcal{D}_\alpha}\}, \quad i = 0, 1.$$

Although this implicitly-defined set does not allow a closed–form evaluation, we can evaluate it numerically, thus obtaining $\sqrt{\mathcal{D}_\alpha} \odot_{f_i} \sqrt{\mathcal{D}_\alpha} \subset \sqrt{\mathcal{D}_\alpha}$, as shown in Figure 5.16. As a consequence, $\mathbf{w}(t)$ is entirely contained within $\sqrt{\mathcal{D}_\alpha}$ for all $\alpha \in (0, \frac{\pi}{2})$, and thus the hodograph $\mathbf{x}'(t) = \mathbf{w}^2(t)$ of the ATPH (++) interpolant $\mathbf{x}(t)$ lies within $\mathcal{D}_\alpha$ for all the same choices of α, which concludes the proof.

□

By Proposition 8 we can thus select the (++) ATPH interpolant directly, without explicitly constructing all four solutions and comparing them.



(a) $\alpha = \frac{\pi}{5}$ (b) $\alpha = \frac{2}{5}\pi$

Figure 5.16: Visualization of the hodograph control points $\mathbf{m}_{01}$ (green) and $\mathbf{m}_{12}$ (blue) in the domain $\sqrt{\mathcal{D}_\alpha}$ for $\alpha = \frac{\pi}{5}$ (a) and $\alpha = \frac{2}{5}\pi$ (b).

5.7 Construction of ATPH spirals

In this section we define *ATPH spirals* as Algebraic-Trigonometric Pythagorean Hodograph curves with monotone curvature, i.e., with no interior curvature extrema. Spiral segments are widely used in several practical applications such as, e.g., highway and railway design, or robot path planning. The control of the curvature behavior is also desirable in many CAD and CAGD applications.
While quintic PH curves with monotone curvature can be also obtained [Farouki 1997], the advantage of an ATPH spiral lies in the fact that it allows for higher flexibility and better curvature variation. In the next sections we will describe

a general construction of ATPH spirals to be used as G^2 transition elements either between a line and a circle that do not intersect or between a pair of external circles, and we will highlight their advantages with respect to polynomial PH spirals.

5.7.1 ATPH spiral for designing a G^2 transition between a line and a circle

Let $\mathbf{x}(t) = \sum_{i=0}^{5} \mathbf{p}_i B_i^5(t)$, $t \in [0, \alpha]$ be an ATPH curve with control points $\mathbf{p}_i$, $i = 0, ..., 5$ as in (5.61)-(5.65), that satisfies the equation

$$\mathbf{x}(t) = \mathbf{p}_0 + \int_0^t \mathbf{w}^2(s)\, ds,$$

where $\mathbf{p}_0$ is a complex arbitrary integration constant, and

$$\mathbf{w}(t) = \mathbf{w}_0 \tilde{B}_0^2(t) + \mathbf{w}_1 \tilde{B}_1^2(t) + \mathbf{w}_2 \tilde{B}_2^2(t),$$

with $\mathbf{w}_j = u_j + \mathrm{i}v_j = w_j e^{\mathrm{i}\theta_j}$, $j = 0, 1, 2$. Given two vectors $\mathbf{t}_0$ and $\mathbf{t}_\alpha$, in the following we define an ATPH spiral $\mathbf{x}(t)$ such that

(a) $\mathbf{x}(0) = \mathbf{p}_0$, $\mathbf{t}_0$ is the tangent at $t = 0$, $\kappa(0) = 0$;

(b) $\mathbf{t}_\alpha$ is the tangent at $t = \alpha$, $\kappa(\alpha) = \frac{1}{R}$ $(R > 0)$, $\kappa'(\alpha) = 0$;

(c) $\mathbf{x}(t)$ has monotonically increasing curvature, i.e., $\kappa'(t) > 0$ for all $t \in [0, \alpha]$.

Without loss of generality we assume $\mathbf{p}_0 = 0$. In this way $\mathbf{x}(0) = 0$. Moreover, we set $\mathbf{t}_0 = 1$ and $\mathbf{t}_\alpha = e^{\mathrm{i}\theta}$, with $0 < \theta < \pi$. Thus, being $\mathbf{x}'(t) = \mathbf{w}^2(t)$, from the second condition in (a) we get $\mathbf{x}'(0) = \mathbf{w}_0^2 = w_0^2$ and from the first condition in (b) we obtain $\mathbf{x}'(\alpha) = \mathbf{w}_2^2 = w_2^2 e^{\mathrm{i}\theta}$. From these two equations we immediately have

$$\mathbf{w}_0 = w_0 \in \mathbb{R}\backslash\{0\} \qquad \text{and} \qquad \mathbf{w}_2 = w_2 e^{\mathrm{i}\frac{\theta}{2}}, \quad 0 < \theta < \pi, \;\; w_2 \in \mathbb{R}\backslash\{0\}. \tag{5.80}$$

Now, recalling that the (signed) curvature of the ATPH curve has the expression

$$\kappa(t) = 2\, \frac{\mathrm{Im}\Big(\overline{\mathbf{w}}(t)\, \mathbf{w}'(t)\Big)}{|\mathbf{w}(t)|^4}, \tag{5.81}$$

and

$$\mathbf{w}'(0) = \cot\left(\frac{\alpha}{2}\right)(\mathbf{w}_1 - \mathbf{w}_0),$$

the value of $\kappa(0)$ can be simply written in the form

$$\kappa(0) = 2\, \cot\left(\frac{\alpha}{2}\right)\, \frac{\mathrm{Im}(\mathbf{w}_1)}{w_0^3}.$$

By requiring $\kappa(0) = 0$ (third condition in (a)) we thus get $\mathrm{Im}(\mathbf{w}_1) = 0$, i.e.,

$$\mathbf{w}_1 = w_1 \in \mathbb{R}\backslash\{0\}. \tag{5.82}$$

On the other hand, for the endpoint $t = \alpha$, being

$$\mathbf{w}'(\alpha) = \cot\left(\frac{\alpha}{2}\right)(\mathbf{w}_2 - \mathbf{w}_1),$$

and substituting $\mathbf{w}_1 = w_1$, from (5.81) we obtain

$$\kappa(\alpha) = 2\,\cot\left(\frac{\alpha}{2}\right)\,\frac{w_1}{w_2^3}\,\sin\left(\frac{\theta}{2}\right).$$

Therefore, requiring that $\kappa(\alpha) = \frac{1}{R}$, with $R > 0$, (second condition in (b)) we find

$$w_1 = \frac{w_2^3\,\tan\left(\frac{\alpha}{2}\right)}{2R\sin\left(\frac{\theta}{2}\right)}.$$

We now proceed by imposing the condition $\kappa'(\alpha) = 0$ (the last of (b)). Since

$$\kappa'(t) = 2\frac{\mathrm{Im}\left(\overline{\mathbf{w}}^2(t)\,(\mathbf{w}(t)\mathbf{w}''(t) - 2\mathbf{w}'^2(t))\right)}{|\mathbf{w}(t)|^6},$$

and

$$\mathbf{w}''(\alpha) = \frac{1}{1-\cos(\alpha)}\Big(\mathbf{w}_0 - (1+\cos(\alpha))\mathbf{w}_1 + \cos(\alpha)\mathbf{w}_2\Big),$$

then setting $\kappa'(\alpha) = 0$ gives

$$\mathrm{Im}\Big(3(1+\cos(\alpha))\,|\mathbf{w}_2|^2\mathbf{w}_1\overline{\mathbf{w}}_2 + |\mathbf{w}_2|^2\mathbf{w}_0\overline{\mathbf{w}}_2 - 2(1+\cos(\alpha))\,\mathbf{w}_1^2\overline{\mathbf{w}}_2^2\Big) = 0,$$

and inserting the values of $\mathbf{w}_j$, $j = 0, 1, 2$, from (5.80), (5.82) the last equation provides

$$w_0 = \frac{w_2^3\Big(2(1-\cos(\alpha))w_2^2\cos\left(\frac{\theta}{2}\right) - 3R\sin(\alpha)\sin\left(\frac{\theta}{2}\right)\Big)}{2R^2\sin^2\left(\frac{\theta}{2}\right)}.$$

Now, introducing the notation $M = \frac{w_2^2}{R}$, we can rewrite the expressions for the coefficients $\mathbf{w}_j$, $j = 0, 1, 2$ in terms of the parameter M as follows:

$$\begin{aligned}
\mathbf{w}_0 &= w_0 = \sqrt{RM^3}\,\frac{2(1-\cos(\alpha))M\cos\left(\frac{\theta}{2}\right) - 3\sin(\alpha)\sin\left(\frac{\theta}{2}\right)}{2\sin^2\left(\frac{\theta}{2}\right)},\\
\mathbf{w}_1 &= w_1 = \sqrt{RM^3}\,\frac{1-\cos(\alpha)}{2\sin(\alpha)\sin\left(\frac{\theta}{2}\right)},\\
\mathbf{w}_2 &= \sqrt{RM}\,\Big(\cos\left(\tfrac{\theta}{2}\right) + \mathrm{i}\sin\left(\tfrac{\theta}{2}\right)\Big).
\end{aligned}$$

We now still have to satisfy the additional requirement $\kappa'(t) > 0$ for all $t \in [0, \alpha]$ (condition (c)). This will provide a lower bound on the admissible values of w_2. Looking at the formula of $\kappa'(t)$, it is evident that the problem reduces to study a sufficient condition on w_2 that guarantees

$$\mathrm{Im}\left(\overline{\mathbf{w}}^2(t)\,(\mathbf{w}(t)\mathbf{w}''(t) - 2\mathbf{w}'^2(t))\right) > 0 \qquad \forall t \in [0, \alpha]. \tag{5.83}$$

Let us start by observing that we can write

$$\mathrm{Im}\left(\overline{\mathbf{w}}^2(t)\left(\mathbf{w}(t)\mathbf{w}''(t)-2\mathbf{w}'^2(t)\right)\right)=\frac{M^3R^2}{24\tilde{s}^5s_1c_1}\sum_{i=0}^{6}b_i\tilde{B}_i^6(t),$$

where

$$\begin{aligned}
b_0 &= 12M^2s_1^2c_1\left(2Ms_1\tilde{c}-3\tilde{s}c_1\right)^3,\\
b_1 &= 2M^2s_1^2\left(2Ms_1\tilde{c}-3\tilde{s}c_1\right)^2\left(16M\tilde{c}s_1c_1-\tilde{s}(24c_1^2+1)\right),\\
b_2 &= \tfrac{4Ms_1c_1}{4c_1^2+1}\Big(56M^4s_1^4\tilde{c}^3-200M^3s_1^3c_1\tilde{s}\tilde{c}^2+6M^2s_1^2\tilde{c}\tilde{s}^2(37c_1^2-4\tilde{c}^2-1)\\
&\qquad +9Ms_1c_1\tilde{s}^3(8s_1^2-8\tilde{s}^2+1)-54c_1^2\tilde{c}\tilde{s}^4\Big),\\
b_3 &= \tfrac{3Ms_1\tilde{s}}{3+2c_1^2}\Big(56M^3s_1^3\tilde{c}^2-156M^2c_1s_1^2\tilde{c}\tilde{s}+Ms_1\tilde{s}^2(108c_1^2-24\tilde{c}^2-1)+36c_1\tilde{c}\tilde{s}^3\Big),\\
b_4 &= \tfrac{4c_1\tilde{s}^2}{4c_1^2+1}\Big(6M^3s_1^3\tilde{c}(4\tilde{c}^2+3)-2M^2s_1^2c_1\tilde{s}(42\tilde{c}^2+13)+18Ms_1(4c_1^2-1)\tilde{s}^2\tilde{c}+21c_1\tilde{s}^3\Big),\\
b_5 &= 2\tilde{s}^3\Big(2M^2s_1^2(6\tilde{c}^2+1)-36Ms_1c_1\tilde{s}\tilde{c}+3(8c_1^2-1)\tilde{s}^2\Big),\\
b_6 &= 0,
\end{aligned}$$

with $\tilde{s}:=\sin\left(\frac{\theta}{2}\right)$, $\tilde{c}:=\cos\left(\frac{\theta}{2}\right)$ and s_1, c_1 from (5.3).

Remark 3. *Note that the Bézier coefficient $b_6=0$ agrees with the requirement $\kappa'(\alpha)=0$.*

Thus, if we assume $0<\alpha<\pi/2$ and $0<\theta<\pi$, it turns out that any arbitrary choice of w_2 satisfying the inequality

$$w_2>\sqrt{k^*\,R\,\tan\left(\frac{\theta}{2}\right)},$$

with

$$k^*=\frac{5}{2}\cot\left(\frac{\alpha}{2}\right)-\cot(\alpha),$$

guarantees that all the coefficients b_i, $i=0,...,5$ are strictly positive and therefore (5.83) is satisfied. As a consequence, for any arbitrary $k>k^*$ we can equivalently set

$$w_2=\sqrt{k\,R\,\tan\left(\frac{\theta}{2}\right)},$$

and rewrite the coefficients $\mathbf{w}_j$, $j=0,1,2$ in terms of the parameter k as follows:

$$\begin{aligned}
\mathbf{w}_0 &= \sqrt{kR\tan\left(\tfrac{\theta}{2}\right)}\,\tfrac{k}{2}\Big(2(1-c_2)k-3s_2\Big)\,\sec\left(\tfrac{\theta}{2}\right),\\
\mathbf{w}_1 &= \sqrt{kR\tan\left(\tfrac{\theta}{2}\right)}\,\tfrac{k}{2}\,\tfrac{s_1}{c_1}\,\sec\left(\tfrac{\theta}{2}\right),\\
\mathbf{w}_2 &= \sqrt{kR\tan\left(\tfrac{\theta}{2}\right)}\,\left(\cos\left(\tfrac{\theta}{2}\right)+\mathrm{i}\sin\left(\tfrac{\theta}{2}\right)\right),
\end{aligned}\tag{5.84}$$

with s_i, c_i, $i=1,2$ from (5.3).

Thus, taking into account formulae (5.84), the displacement between the end points of the ATPH spiral

$$\mathbf{p}_5-\mathbf{p}_0 = \frac{1}{16s_1^4}\left(n_0\mathbf{w}_0^2 + 4n_2(1+c_2)\mathbf{w}_1^2 + n_0\mathbf{w}_2^2 + 2(n_0-6n_2)\mathbf{w}_1(\mathbf{w}_0+\mathbf{w}_2) + 4n_2\mathbf{w}_0\mathbf{w}_2\right),$$

can be expressed as

$$\mathbf{p}_5 - \mathbf{p}_0 = \frac{kR\tan\left(\frac{\theta}{2}\right)}{16c_1s_1^4}\left(\mathcal{F}+\mathrm{i}\mathcal{G}\right),$$

with

$$\begin{aligned}\mathcal{F} &= \frac{1}{4c_1\cos^2\left(\frac{\theta}{2}\right)}\Big[4n_0s_1^2s_2^2k^4 + 2s_2(c_2-1)(6n_2+(3c_2+2)n_0)k^3\\ &+ s_2^2\left(3(3c_1^2-1)n_0 + 4\left(2\cos^2\left(\frac{\theta}{2}\right)+5\right)n_2\right)k^2\\ &+ 2s_2\cos^2\left(\frac{\theta}{2}\right)(n_0-6n_2(c_2+2))k + 4c_1^2n_0\cos^2\left(\frac{\theta}{2}\right)\cos(\theta)\Big],\end{aligned}$$

and

$$\mathcal{G} = \tan\left(\frac{\theta}{2}\right)\left(4c_1n_2(1-c_2)k^2 + s_1(n_0-6n_2(c_2+2))k + 2c_1n_0\cos^2\left(\frac{\theta}{2}\right)\right).$$

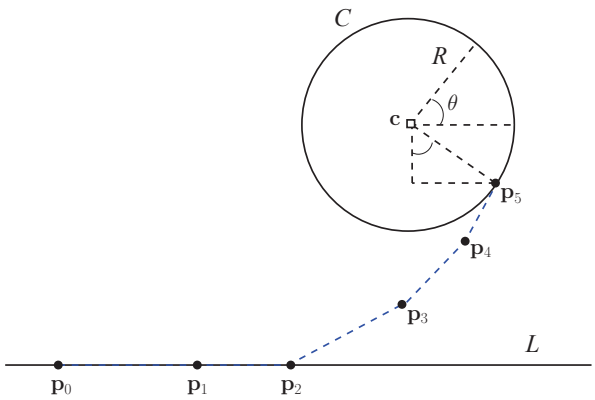


Figure 5.17: Geometrical parameters of an ATPH spiral used as G^2 connection between a line L and a circle C.

Now, if we want to exploit the ATPH spiral satisfying the above conditions (a),(b),(c) to connect G^2 continuously a line L and a circle C that do not intersect, without loss of generality we can assume L to be the x-axis with the transition curve starting at $\mathbf{p}_0 = x_0$ and take C of radius R and center $\mathbf{c} = l + \mathrm{i}h$ with $h > R$. Measuring the angular position θ on the circle in the anticlockwise sense from a perpendicular dropped to L from $\mathbf{c}$ (see Figure 5.17), the transition curve is identified by the requirement

$$\mathrm{Im}(\mathbf{p}_5 - \mathbf{p}_0) = h - R\cos(\theta). \tag{5.85}$$

Using standard trigonometric relations, equation (5.85) leads to

$$Q(\cos(\theta)) := a_2 \cos^2(\theta) + a_1 \cos(\theta) + a_0 = 0, \tag{5.86}$$

where

$$\begin{aligned} a_0 &= 4Rc_1n_2(1-c_2)k^3 + Rs_1(n_0 - 6n_2(c_2+2))k^2 + Rc_1n_0k - 16c_1s_1^4h, \\ a_1 &= 4Rc_1n_2(c_2-1)k^3 + Rs_1(6n_2(c_2+2) - n_0)k^2 + 16c_1s_1^4(R-h), \\ a_2 &= -Rc_1(kn_0 - 16s_1^4), \end{aligned} \tag{5.87}$$

Since

$$Q(0) = a_0 = Rk\left(4c_1n_2(1-c_2)k^2 + s_1(n_0 - 6n_2(c_2+2))k + c_1n_0\right) - 16c_1s_1^4h,$$

and

$$Q(1) = a_2 + a_1 + a_0 = 32c_1s_1^4(R-h),$$

being $0 < \alpha < \pi/2$ and $0 < \theta < \pi$, when h satisfies

$$R < h < A(k,\alpha)\, R, \tag{5.88}$$

with

$$A(k,\alpha) = k\,\frac{(4c_1n_2(1-c_2)k^2 + s_1(n_0 - 6n_2(c_2+2))k + c_1n_0)}{16c_1s_1^4},$$

we have $Q(0) > 0$ and $Q(1) < 0$, so that the equation (5.86) has a unique root satisfying $\cos(\theta) \in (0,1)$. Note that, when $k = k^*\left(= \frac{5}{2}\cot\left(\frac{\alpha}{2}\right) - \cot(\alpha)\right)$, then

$$A(k,\alpha) > 1 \qquad \forall \alpha \in \left(0, \tfrac{\pi}{2}\right),$$

and $A(k,\alpha)$ is monotonically increasing for $k > k^*$ (see Figure 5.18). In this way, for any value of $\alpha \in (0,\pi/2)$, we can make the upper bound on h as large as we want by taking k sufficiently large. Remember also that, if we denote by $\overline{k}$ the value such that $A(\overline{k},\alpha) = \frac{h}{R}$, we need to certainly choose $k > \max(k^*, \overline{k})$ in order to satisfy (5.88). See Figure 5.19 for an illustrative example of this situation. For the sake of clarity, we conclude by summarizing the steps of the algorithm to draw an ATPH spiral to connect G^2 continuously a line L corresponding to the x-axis and a circle C of radius $R > 0$ and center (l,h) with $h > R$.

Remark 4. *Note that the requirement $\kappa'(\alpha) = 0$ indeed implies a G^3 contact of the ATPH spiral with the circle C, while the contact with the line L is only G^2.*

Algorithm
Input: l, $h > 0$, $R \in (0,h)$, $\alpha \in (0,\pi/2)$.

1. Choose k such that inequality (5.88) is satisfied;
2. Compute a_0, a_1, a_2 in (5.87);

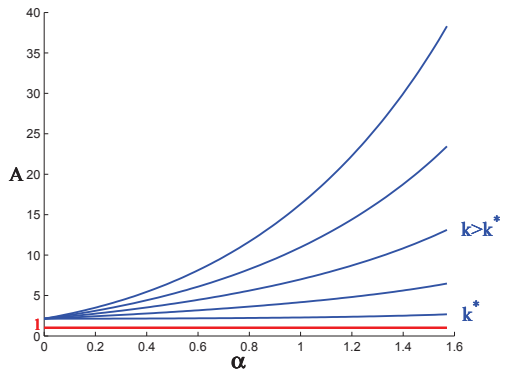


Figure 5.18: Behavior of $A(k, \alpha)$ for $\alpha \in (0, \pi/2)$.

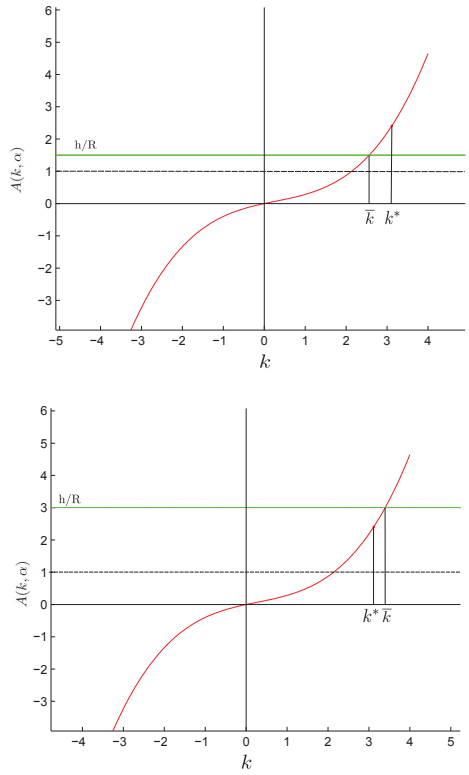


Figure 5.19: Behaviour of $A(k, \alpha)$ for $\alpha = \frac{2}{5}\pi$ and $R = 2$, $h = 3$ (left), $R = 1$, $h = 3$ (right).

3. Determine $\theta \in (0, \pi)$ by solving (5.86);

4. Work out the values of $\mathbf{w}_0, \mathbf{w}_1, \mathbf{w}_2$ given in (5.84);

5. Determine x_0 by solving the equation $\mathrm{Re}(\mathbf{p}_5 - \mathbf{p}_0) = l + R\sin(\theta) - x_0$.

Output: Control points $\mathbf{p}_0 = x_0$, $\mathbf{p}_i$, $i = 1, ..., 4$, $\mathbf{p}_5 = l + R\sin(\theta) + \mathrm{i}(h - R\cos(\theta))$ of the ATPH spiral connecting L and C. As an application example of the previously developed constructive strategy we consider the following choices: $R = 2$, $l = 4$ and $h = 3$. In Figure 5.20 we display the obtained ATPH spiral for $\alpha = \frac{\pi}{5}$, $\alpha = \frac{\pi}{3}$ and $\alpha = \frac{2}{5}\pi$ when $k = k^*$, and in Figure 5.21 the corresponding quintic PH spiral for which k satisfies $R < h < \frac{2k^3 - 3k^2 + 12k}{60} R$ (see [Farouki 1997]).

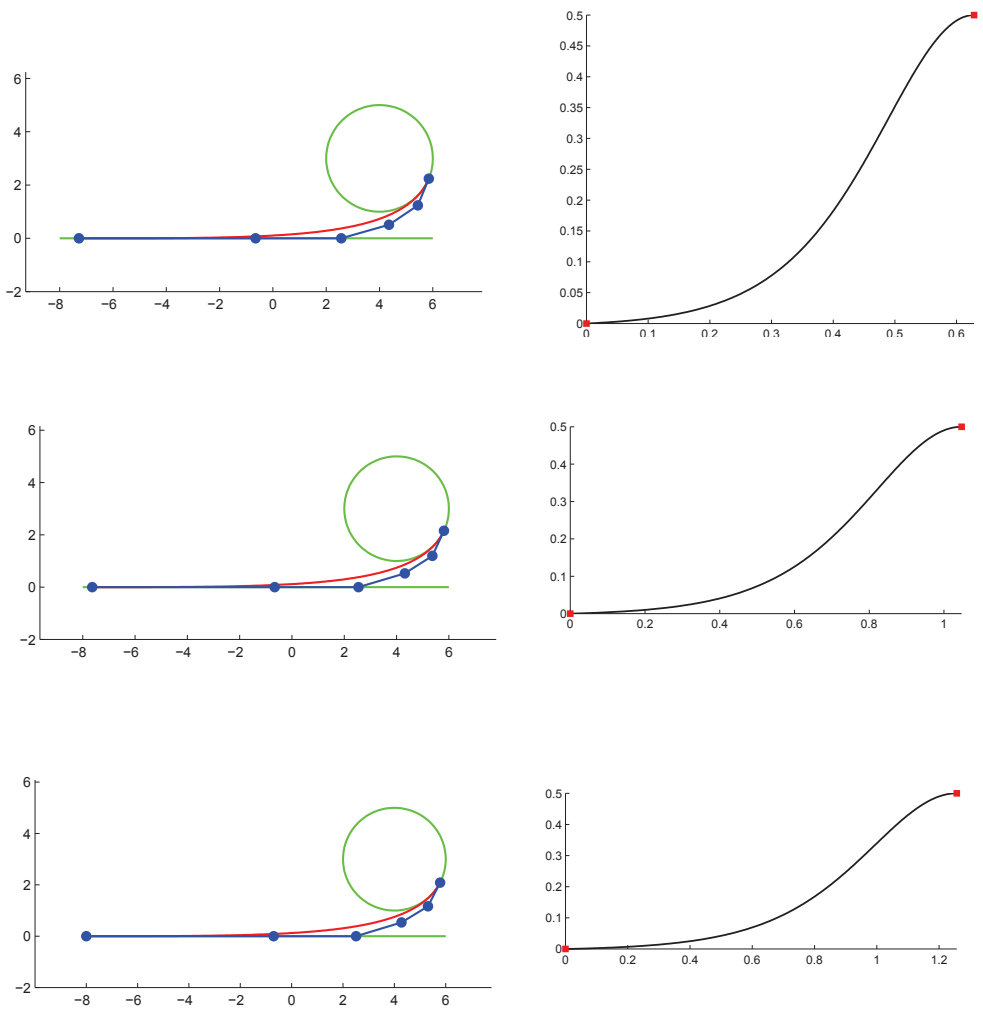


Figure 5.20: First column: ATPH spirals for $\alpha = \frac{\pi}{5}$, $\alpha = \frac{\pi}{3}$ and $\alpha = \frac{2}{5}\pi$, where $k = k^*$. Second column: corresponding curvature plots.

In Figure 5.26 we plot the behavior of $\kappa'(t)$ for the ATPH spirals in Figure 5.20 and the quintic PH spiral in Figure 5.21. As we can see, for an ATPH spiral (black), the maximum of the function $\kappa'(t)$ is decreasing for increasing values of α, while for a PH spiral (blue) it is obviously a fixed value. As a consequence ATPH spirals compare favorably with their polynomial PH counterparts since the parameter α can

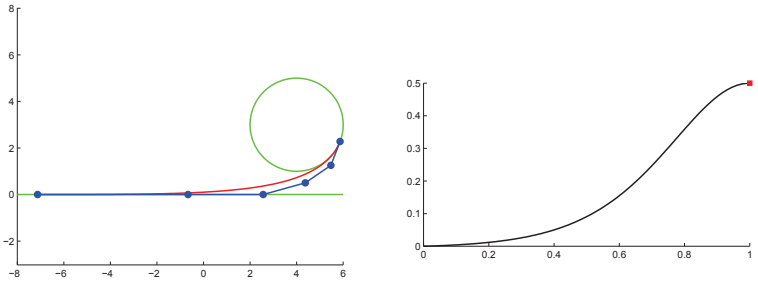


Figure 5.21: Quintic PH spiral for $k = 4$ (left) and its curvature plot (right).

be suitably selected to control the curvature variation and ensure the best possible behavior.

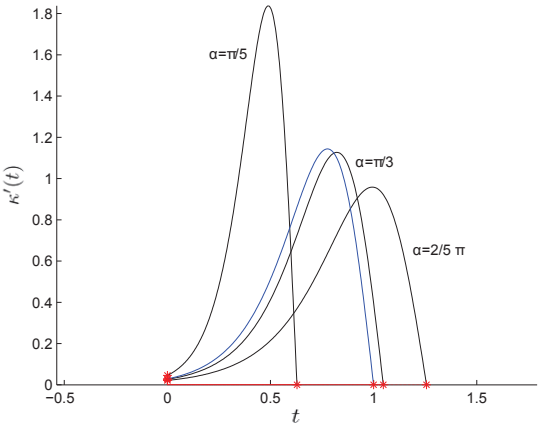


Figure 5.22: Behavior of $\kappa'(t)$ for the ATPH spirals in Figure 5.20 with $\alpha = \frac{\pi}{5}, \frac{\pi}{3}, \frac{2}{5}\pi$ (black) and for the quintic PH spiral in Figure 5.21 (blue).

Furthermore, by evaluating the arc length of the ATPH spiral over the interval $[0, \alpha]$ from formula (5.75) with $\mathbf{w}_0, \mathbf{w}_1, \mathbf{w}_2$ in (5.84), we can obtain the length of the ATPH spiral

$$S_\alpha = \frac{R}{8} k \tan\left(\frac{\theta}{2}\right) \Big(\left(8\alpha s_2^2 k^2 + 16 s_2 (s_1^2 - 3\alpha c_1^2) k + \alpha(72 c_1^4 + 1) - s_2(c_2 + 12 s_2)\right) \frac{s_1^2}{c_1^2} k^2 \sec^2\left(\frac{\theta}{2}\right) + 4(8 s_1^2 s_2 k^2 - 2 s_1 s_2 (6 c_1 - s_1) k + s_2 c_2 + \alpha) \Big),$$

with s_i, c_i, $i = 1, 2$ from (5.3).

Observe that, for given R and θ, the last expression is positive and monotonically increasing with k for $k \geq k^*$; as a consequence, the length of the ATPH spiral increases with k. It is clear that the parameter k can be selected such that the total arc length S_α assumes a specified value. This requires to solve a polynomial equation of degree 5. Although this is possible also in the polynomial case presented

in [Farouki 1997], if using an ATPH spiral, once the value of k has been fixed, we still have another free parameter α that can be used for other purposes.

5.7.2 S-shaped ATPH spiral for designing a G^2 transition between two circles

In this section we consider the problem of designing an S-shaped ATPH spiral to join two given circles Ω_0, Ω_1 with centers $\boldsymbol{C}_0$, $\boldsymbol{C}_1$ and radii r_0, r_1, such that at both points of contact G^2 continuity is ensured. We denote by r the distance between the centers of the two circles

$$r = |\boldsymbol{C}_1 - \boldsymbol{C}_0|, \tag{5.89}$$

and we define $r_1 = \frac{r_0}{\mu^3}$, where $\mu \geq 1$. Moreover, without loss of generality (after suitable translation, rotation and reflection), for the starting point of the ATPH curve we can assume $\mathbf{p}_0 = 0$, as well as the curvature at the starting point being 1 (i.e. $r_0 = 1$), $\mathbf{p}_1$ lying on the positive x-axis ($\mathrm{Re}(\mathbf{p}_1) > 0$), the larger circle having the center $\boldsymbol{C}_0 = \mathrm{i}$, and the end point $\mathbf{p}_5$ lying above the x-axis ($\mathrm{Im}(\mathbf{p}_5) > 0$). We then denote by θ and $2n\theta$ the angles from $\mathbf{p}_1 - \mathbf{p}_0$ to $\mathbf{p}_2 - \mathbf{p}_1$ and from $\mathbf{p}_1 - \mathbf{p}_0$ to $\mathbf{p}_5 - \mathbf{p}_4$, respectively (see Figure 5.23). By construction, it turns out that θ and $2n\theta$ may vary at most between 0 and $\frac{\pi}{2}$.

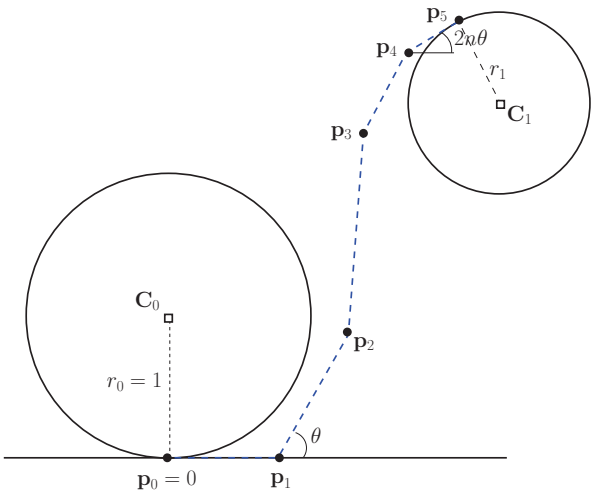


Figure 5.23: Geometrical parameters of an S-shaped ATPH spiral used as G^2 connection between two external circles.

Summarizing, our goal is to define an S-shaped ATPH spiral that satisfies all the following requirements:

(i) $\mathbf{x}(0) = \mathbf{p}_0 = 0$

(ii) $\mathbf{t}_0 = 1$ is the tangent at $t = 0$ (i.e., $\mathbf{x}'(0) \parallel 1$)

(iii) $\mathbf{t}_\alpha = e^{2\mathrm{i}n\theta}$ is the tangent at $t = \alpha$ (i.e., $\mathbf{x}'(\alpha) \parallel (\cos(2n\theta) + \mathrm{i}\sin(2n\theta))$)

(iv) $(\mathbf{p}_2 - \mathbf{p}_1) \parallel (\cos(\theta) + \mathrm{i}\sin(\theta))$

(v) $(\kappa(0), \kappa(\alpha)) = (1, -\mu^3)$

(vi) $\mathbf{x}(t)$ has monotonously decreasing curvature, i.e. $\kappa'(t) < 0$ for all $t \in [0, \alpha]$.

By Kneser's theorem [Guggenheimer 1963], it follows that a single S-shaped segment cannot have monotone curvature both in case of tangent and intersecting circles, as well as when one is contained into the other. Thus, we can draw S-shaped ATPH transitions satisfying the above requirements, only under the condition $r_0 + r_1 < r$. As in the previous section, in order to determine the ATPH spiral solving the above problem, we need to work out its complex coefficients $\mathbf{w}_j$, $j = 0, 1, 2$. First of all, recalling that $\mathbf{x}'(t) = \mathbf{w}^2(t)$, from (ii) and (iii) we obtain respectively $\mathbf{w}_0 = w_0 \in \mathbb{R}\backslash\{0\}$ and $\mathbf{w}_2 = w_2\, e^{in\theta}$, $w_2 \in \mathbb{R}\backslash\{0\}$. Then, being $\mathbf{p}_2 - \mathbf{p}_1 = \frac{n_0 - 6n_2}{8s_1^4}\, \mathbf{w}_0\mathbf{w}_1$ with $\mathbf{w}_0 = w_0$, $\mathbf{w}_1 = w_1 e^{\mathrm{i}\theta_1}$ and $w_0, w_1 \in \mathbb{R}\backslash\{0\}$, from (iv) we immediately get $\theta_1 = \theta$. Before imposing condition (v), we recall once again that the (signed) curvature of the ATPH curve has the expression in (5.81). Thus, by requiring $\kappa(0) = 1$, we obtain $w_1 = \frac{w_0^3}{2\sin(\theta)} \tan\left(\frac{\alpha}{2}\right)$ and then $\mathbf{w}_1 = \frac{w_0^3}{2} \tan\left(\frac{\alpha}{2}\right)(\cot(\theta) + \mathrm{i})$. On the other hand, for the endpoint $t = \alpha$, condition $\kappa(\alpha) = -\mu^3$ leads to $w_2 = \frac{w_0}{\mu} \sqrt[3]{\frac{\sin((1-n)\theta)}{\sin\theta}}$. In conclusion, after introducing the notation

$$h = w_0^2, \qquad \varsigma = \frac{\cos(n\theta)}{\mu} \sqrt[3]{\frac{\sin((1-n)\theta)}{\sin\theta}}, \tag{5.90}$$

we can rewrite the coefficients $\mathbf{w}_j$, $j = 0, 1, 2$ in terms of h and ς as follows:

$$\begin{aligned} \mathbf{w}_0 &= \sqrt{h} \\ \mathbf{w}_1 &= \sqrt{h}\, \tfrac{h}{2} \tan\left(\tfrac{\alpha}{2}\right) \cot(\theta) + \mathrm{i}\sqrt{h}\, \tfrac{h}{2} \tan\left(\tfrac{\alpha}{2}\right) \\ \mathbf{w}_2 &= \sqrt{h}\,\varsigma + \mathrm{i}\sqrt{h}\,\varsigma \tan(n\theta) \end{aligned} \tag{5.91}$$

Thus, since $h, n, \mu, \theta, \alpha$ are free parameters, it turns out that there exists a whole family of ATPH curves $\mathbf{x}(t)$ satisfying the conditions (i)-(v). From (5.61)-(5.65) with $\mathbf{p}_0 = 0$ we obtain

$$\mathbf{p}_5 = \frac{n_0}{16s_1^4}(\mathbf{w}_0^2 + \mathbf{w}_2^2) + \frac{n_0 - 6n_2}{8s_1^4}\mathbf{w}_1(\mathbf{w}_0 + \mathbf{w}_2) + \frac{n_2}{4s_1^4}\left((1 + c_2)\mathbf{w}_1^2 + \mathbf{w}_0\mathbf{w}_2\right),$$

where $\mathbf{w}_0, \mathbf{w}_1, \mathbf{w}_2$ are given by (5.91). Also taking into account that the coordinates of the centers of the two circles are

$$\boldsymbol{C}_0 = \mathbf{p}_0 + r_0\,(0, 1) = (0, 1), \qquad \boldsymbol{C}_1 = \mathbf{p}_5 + r_1\,(\sin(2n\theta), -\cos(2n\theta)) =: (p, q),$$

having denoted by r the distance between $\boldsymbol{C}_0$ and $\boldsymbol{C}_1$, condition (5.89) determines a polynomial equation $f(h) = 0$ given by $f(h) = p^2 + (q-1)^2 - r^2$, where $p = Ah^3 + Bh^2 + Ch + D$, $q = Eh^3 + Fh^2 + Gh + H$, and

$$\begin{aligned} A &= \frac{n_2 \tan^2(\alpha/2)\,(c_2+1)\,(1-\tan^2(\theta))}{16s_1^4 \tan^2(\theta)}, & E &= \frac{n_2 \tan^2(\alpha/2)(c_2+1)\cot(\theta)}{8s_1^4}, \\ B &= \frac{\tan(\alpha/2)(n_0-6n_2)(1+\varsigma(1-\tan(n\theta)\tan(\theta)))}{16s_1^4 \tan(\theta)} & F &= \frac{\tan(\alpha/2)(n_0-6n_2)(1+\varsigma(1+\tan(n\theta)\cot(\theta)))}{16s_1^4}, \\ C &= \frac{n_0+4\varsigma n_2+\varsigma^2 n_0(1-\tan^2(n\theta))}{16s_1^4}, & G &= \frac{\varsigma\tan(n\theta)(2n_2+\varsigma n_0)}{8s_1^4}, \\ D &= \frac{\sin(2n\theta)}{\mu^3}, & H &= -\frac{\cos(2n\theta)}{\mu^3}. \end{aligned}$$

with s_1, c_2, n_0, n_2 from (5.3), (5.4) and ς from (5.90). Therefore, fixing r, it turns out that h is not at all a positive free parameter, but it needs to be determined such that $f(h) = 0$. Let us start by observing that, if $\theta \in (0, \frac{\pi}{4}]$ and $n \in (0, \frac{1}{2})$, then $A, B, C, D, E, F, G > 0$ and $H < 0$. Observe also that the polynomial equation $f(h) = 0$ is of degree six and it can be conveniently rewritten as $\sum_{\ell=0}^{6} g_\ell(n, \mu, \theta, \alpha)\, h^\ell = 0$ where the coefficients $g_\ell(n, \mu, \theta, \alpha)$, $\ell = 0, ..., 6$ have expressions

$$\begin{aligned}
g_0(n, \mu, \theta, \alpha) &= D^2 + H^2 + 1 - r^2 - 2H, \\
g_1(n, \mu, \theta, \alpha) &= 2(CD + GH - G), \\
g_2(n, \mu, \theta, \alpha) &= C^2 + 2BD + G^2 + 2FH - 2F, \\
g_3(n, \mu, \theta, \alpha) &= 2(AD + BC + EH + FG - E), \\
g_4(n, \mu, \theta, \alpha) &= B^2 + 2AC + F^2 + 2EG, \\
g_5(n, \mu, \theta, \alpha) &= 2(AB + EF), \\
g_6(n, \mu, \theta, \alpha) &= A^2 + E^2.
\end{aligned}$$

Looking at the expressions of the coefficients g_ℓ, $\ell = 0, ..., 6$ it trivially follows that $g_6 > 0$ and $g_0 < 0$ (since $r > r_0 + r_1$). Moreover, from the previous observation on the signs of A, B, C, D, E, F, G, H it turns out that $g_2 < 0$ and $g_4, g_5 > 0$. Therefore, independently of the sign of g_1 and g_3, using the Descartes rule of signs and the intermediate value theorem, the existence of a positive solution of $f(h) = 0$ immediately follows, which guarantees the existence of the ATPH spiral. This solution is certainly unique if we further assume $\mu \geq 1.5$ because in this case the condition $g_1 < 0$ is also satisfied.
Now we still need to satisfy the last requirement $\kappa'(t) < 0$ for all $t \in [0, \alpha]$ (condition (vi)). This will provide stronger restrictions on the admissible values of the free parameters. Looking at the formula of $\kappa'(t)$, it is evident that the problem reduces to study sufficient conditions on the free parameters that ensure

$$\mathrm{Im}\left(\overline{\mathbf{w}}^2(t)\left(\mathbf{w}(t)\mathbf{w}''(t) - 2\mathbf{w}'^2(t)\right)\right) < 0 \qquad \forall t \in [0, \alpha].$$

Let us start by observing that we can write

$$\mathrm{Im}\left(\overline{\mathbf{w}}^2(t)\left(\mathbf{w}(t)\mathbf{w}''(t) - 2\mathbf{w}'^2(t)\right)\right) = \frac{h^2}{4s_1^2} \sum_{i=0}^{6} b_i \tilde{B}_i^6(t)$$

with

$$
\begin{aligned}
b_0 &= 2\cot(\theta)(c_2-1)h^2 + 3s_2h + 2\varsigma\tan(n\theta),\\
b_1 &= -\tfrac{1}{3c_1}\left(\tfrac{2s_1^3}{\sin^2(\theta)}h^3 + \cot(\theta)c_1(c_2-1)h^2 + (5\varsigma+\varsigma\tan(n\theta)\cot(\theta)-3)s_1h - 8\varsigma c_1\tan(n\theta)\right),\\
b_2 &= -\tfrac{1}{3(2c_2+3)\cos(n\theta)\sin^2(\theta)}\Big(\cos(n\theta)s_1^2s_2h^3 + 2s_1^2((1-3\varsigma)\sin(\theta(n-2)) - \sin(\theta(n+2))\\
&\qquad +6\varsigma\sin(n\theta))h^2 + \varsigma s_1c_1(3\cos(\theta(n+2)) - 10\cos(\theta(n-2))\\
&\qquad +7\cos(n\theta))h + 2\varsigma(6\varsigma-7)\sin(n\theta)\sin^2(\theta)\Big),\\
b_3 &= -\tfrac{hs_1}{2c_1(c_2+4)\cos^2(n\theta)\sin^3(\theta)}\Big(s_1^2\cos(n\theta)(\varsigma\sin(\theta(n-1)) - \cos(n\theta)\sin(\theta))h^2\\
&\qquad +3\varsigma c_1s_1\cos(n\theta)(\cos(\theta(n-1)) - \cos(\theta(n-3)))h\\
&\qquad +\varsigma\sin^2(\theta)((1+8\varsigma)\sin(\theta) - 3\sin(\theta(2n+1))\\
&\qquad +2(3\varsigma-2)\sin(\theta(2n-1)))\Big),\\
b_4 &= -\tfrac{2\varsigma}{3(2c_2+3)\cos^3(n\theta)\sin^3(\theta)}\Big(-c_1s_1^3\sin(\theta(n-1))\cos^2(n\theta)h^3\\
&\qquad -\tfrac{s_1^2}{4}\cos(n\theta)((3+4\varsigma)\cos(\theta(2n-1)) + (3-4\varsigma)\cos(\theta(2n-3))\\
&\qquad -6\cos(\theta(2n+1)) + 3(\cos(3\theta) - \cos(\theta)))h^2\\
&\qquad +\varsigma s_1c_1\sin^2(\theta)\cos(n\theta)(3\sin(\theta(2n-1)) + 10\sin(\theta))h\\
&\qquad +\varsigma\sin(n\theta)\sin^3(\theta)(7\varsigma - 6\cos^2(n\theta))\Big),\\
b_5 &= -\tfrac{\varsigma}{6c_1\cos^3(n\theta)\sin^3(\theta)}\Big(-4s_1^3\sin(\theta(n-1))\cos^2(n\theta)h^3 - \varsigma s_1^2c_1\cos(n\theta)(\cos(\theta(2n-1))\\
&\qquad -\cos(\theta(2n-3)))h^2 + \varsigma s_1\sin^2(\theta)(2\sin(\theta(n+1))\\
&\qquad +(6\varsigma-5)\sin(\theta(n-1)) - 3\sin(\theta(3n-1)))h + 16\varsigma^2c_1\sin(n\theta)\sin^3(\theta)\Big),\\
b_6 &= -\tfrac{\varsigma^2}{\cos^3(n\theta)\sin^3(\theta)}\Big(s_1^2\cos(n\theta)(\cos(\theta(2n-1)) - \cos(\theta(2n-3)))h^2\\
&\qquad -3\varsigma s_2\sin((1-n)\theta)\sin^2(\theta)h + 2\varsigma\sin(n\theta)\sin^3(\theta)\Big)
\end{aligned}
$$

and ς from (5.90). Thus, we need to determine under which conditions on the free parameters θ, n, μ, the Bézier curve $\sum_{i=0}^{6} b_i\tilde{B}_i^6(t)$ is strictly negative. Under the assumption $\theta \in (0, \pi/4]$ and $n \in (0, 1/2)$ it turns out that $b_i < 0$ for $i = 0, 1, 2, 4, 5, 6$, while the sign of b_3 is not fixed. Anyway, assuming $\mu \in [1, 5]$, it is easy to verify numerically that $\sum_{i=0}^{6} b_i\tilde{B}_i^6(t) < 0$ for all $t \in [0, \alpha]$ and $\alpha \in (0, \frac{\pi}{2})$.

As an application example of the constructive strategy of S-shaped ATPH spirals we consider the following choices: $r_0 = 1$, $\mu = \sqrt[3]{2}$, $r_1 = \frac{1}{2}$, $r = 2$, $n = \frac{1}{3}$, $\theta = \frac{\pi}{5}$. In Figure 5.24, from top to bottom, we display the quintic PH spiral in [Habib 2008] and the ATPH spirals with $\alpha = \frac{7}{24}\pi$ and $\alpha = \frac{5}{12}\pi$, obtained from the above data.

Note that, although the quintic PH curve in Figure 5.25 (left) has monotone curvature, in general the S-shaped transition elements constructed in [Habib 2008] do not guarantee the fulfillment of condition (vi). Moreover, from Figure 5.24 we can see that the additional free parameter α included in the ATPH approach can be used either to modify the location of the second point of contact of the spiral or to adjust the curvature profile and/or curvature variation of the spiral (see Figure 5.26). Therefore ATPH spirals again compare favorably with their polynomial PH counterpart.

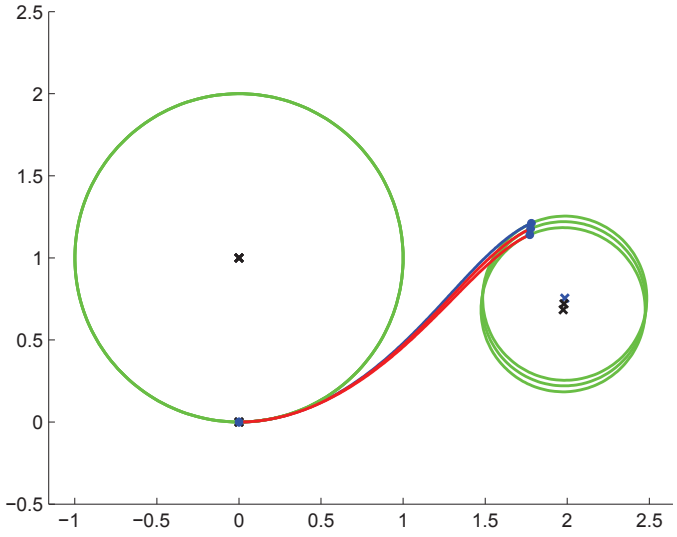


Figure 5.24: From top to bottom: S-shaped quintic PH spiral in [Habib 2008] (blue) and S-shaped ATPH spirals with $\alpha = \frac{7}{24}\pi$ and $\alpha = \frac{5}{12}\pi$ (red).

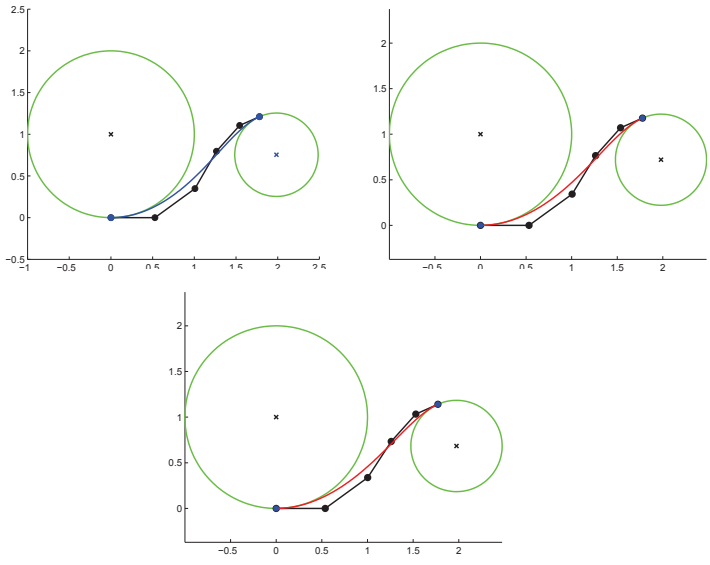


Figure 5.25: Control polygon for the S-shaped quintic PH spiral (left) and the S-shaped ATPH spirals with $\alpha = \frac{7}{24}\pi$ (center), $\alpha = \frac{5}{12}\pi$ (right) displayed in Figure 5.24.

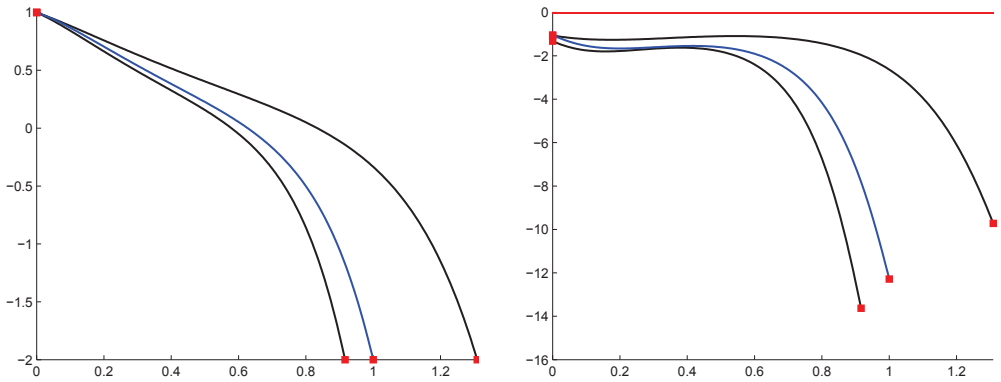


Figure 5.26: Behavior of $\kappa(t)$ (left) and $\kappa'(t)$ (right) for the S-shaped spirals in Figures 5.24-5.25. From top to bottom the plot of the functions concerning the ATPH spiral with $\alpha = \frac{5}{12}\pi$, the quintic PH spiral in [Habib 2008] (blue) and the ATPH spiral with $\alpha = \frac{7}{24}\pi$.

Chapter 6

Conclusions

The research of this thesis started with the study of the area of stop motion camera animation. We outlined what is stop motion animation and what happens when an animator also moves the camera. We observed that a limited amount and an aesthetic quality of noise is desirable to a certain extent since it contributes to realism and to the particular aesthetics of stop motion.

Next, we made a bibliographic research on the most important 3D animation software programs that can control camera moves. We remarked that so far the traditional animation methods in these programs suffer from limitations in producing realistic camera moves. We summarized the main advantages and disadvantages of the software's tools and the mathematical background of these techniques. We described several interpolation techniques to fit a piecewise curve to a sequence of given points (keyframes), depending on the final motion desired. In order to overcome the major disadvantage (dependence of position and speed) of the most popular animation technique, the "Keyframing Animation", several approaches aim at reparameterising the curve by arc length and thus controlling the movement along the curve by an Ease Curve.

The analysis of the advantages and disadvantages of the ways to animate a camera frame by frame allowed us to define our objective. We want to create, by a stop motion animation technique, a 3D animation that looks as realistic as possible. We used the partial solutions proposed in the literature with the aim of overcoming the existing drawbacks of 3D animation software and motion control systems. We presented a new motion control system specifically designed for stop motion that is able to simulate a realistic camera animation with a low budget in order to give stop motion animators total freedom of camera movement that maintains the handwork visual aesthetics of stop motion. In particular we aimed at simulating a 3D realistic camera movement that can integrate constraints and imperfections (noise) of real camera devices by using a haptic interface. We described the whole system, we explained in detail the mathematical processing to obtain different camera movements by using a haptic interface for motion capture. We presented also an assessment of our system carried out with a class of students of the "Art plastiques et Création numérique" Master of the University of Valenciennes. These tests encouraged us to improve our system and to present the future developments.

We observed that the animators need a smooth trajectory curve that takes into account the constraints of a real camera move. Thus, we created a new class of algebraic-trigonometric curves based on the concept of Pythagorean-Hodograph (PH) curves. This representation allows to determine the curve's arc-length in an

analytical way as well as an exact rational parameterization of the offset curves, and is usable in general curve design applications. Our new class of curves is built-upon a five-dimensional mixed algebraic-trigonometric space and is called "Algebraic-Trigonometric Pythagorean-Hodograph" or ATPH. It depends on a parameter which can be used as shape parameter and reproduces arcs of arbitrary length of planar trigonometric curves, as circles, cardioids, deltoids, limacons, lemniscates, piriforms. These properties help us to create a realistic camera movement. To this end, in particular, we solved the first order Hermite interpolation problem and constructed spirals as G^2 transition elements between a line segment and a circle, as well as between a pair of external circles.

We can conclude that our system, specifically designed to help animators realize a realistic camera movement for stop motion, will be a benefit for all types of stop motion productions and 3D animations. With an optimized workflow, such a system will significantly encourage creativity while respecting the handwork aesthetic of stop motion, intensify cinematographic illusion by giving life to camera and allow as much freedom for camera moves as on a real stage.

For our future developments we would like to use this class of curves in our motion control system and we would like to improve the latter by creating an open source system in which a central part (Motion control software) synchronizes the three stages of the process (Camera movement management, Motion capture device and Motion control robot and digital camera).

Appendix A

The structure of the motion control system

We present the technical details of the motion control system described in chapter 4. The diagram in Figure A.1 summarizes the structure of the motion control system.

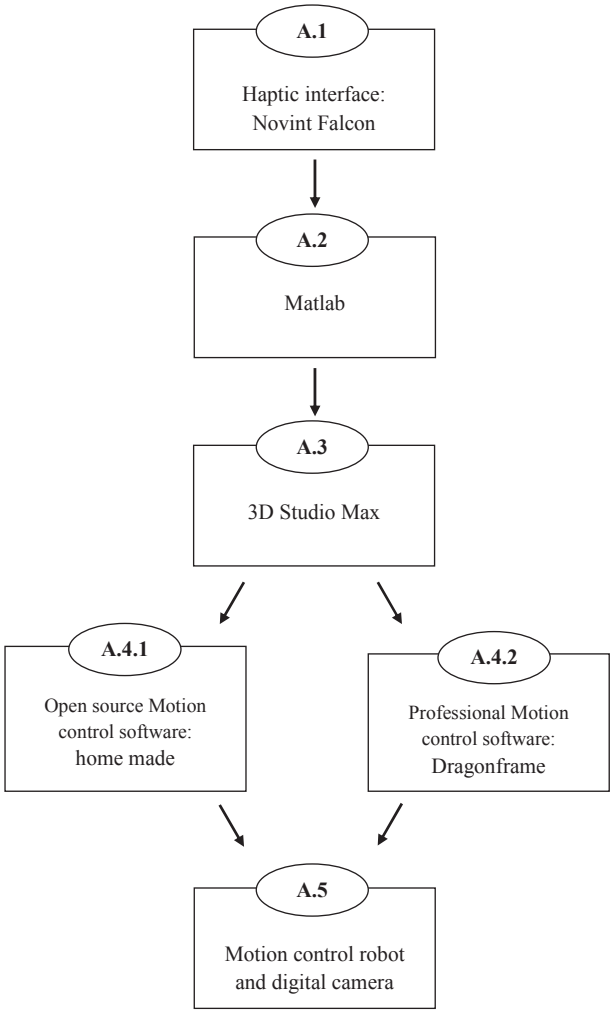


Figure A.1: The structure of the motion control system.

In the following sections, we describe, in detail, the steps of the motion control system.

A.1 Haptic interface: Novint Falcon

The first step involves the haptic system (Novint Falcon, http://www.novint.com/) to create the animation, as explained in section 4.1. We use the Visual Studio's implementation from [Fünfzig 2010] for haptic interaction with a rational parametric cubic Bézier curve. We added the possibility to record haptic positions for every Δt time by pressing the back button on the Novint Falcon (see Figure A.2).

Figure A.2: In the red circle the back button on the Novint Falcon to record robot's positions.

The Visual Studio's code for this step is:

```
void HapticsLogic::getPosition(Point& point){
SYSTEMTIME a;
    GetSystemTime(&a);
    int b;
    hdlToolButtons(&b);
    point.x = m_positionApp[0];
    point.y = m_positionApp[1];
struct tm* newtime;
time_t t = time(NULL);
    tm *time;
    time = localtime(&t);

    //To start haptic positions recording

    if(b==4){
if(var==1){
int sec1 =a.wSecond;
int mill_sec1 = a.wMilliseconds;
printf("Time stop = %i,%i \n",sec1,mill_sec1);
var=2;
FILE * pFileX;
pFileX = fopen ("tempoSTART.txt","w");
fprintf (pFileX, "%i, ",sec1);
```

```
fclose (pFileX);
}
int sec = a.wMilliseconds;
        if (i==1){
sec_prec=sec;
cout << "msec=" << sec_prec <<  endl;
i=2;
    }
        int sec_passed;
    sec_passed = abs(sec-sec_prec);

        if (sec_passed >=960){
sec_passed=41;
        }
        int delta_t;
delta_t = 40;
if(sec_passed>=delta_t && sec_passed<=45){
printf("delta_sec_passed = %i \n",sec_passed);
        c=c+1;
            point.x=((float)((int)(point.x*1000.0f)))/1000.0f;
        point.y=((float)((int)(point.y*1000.0f)))/1000.0f;

        printf("number of points = %i \n",c);
        cout << "positions=" << point.x << ", " << point.y  <<  endl;
            i=1;
        var=3;
}
}
}
```

We export these positions in Matlab's files:

```
//File with the x coordinates
FILE * pFileX;
pFileX = fopen ("../haptic_x.m","w");
fprintf (pFileX, "x_t=[%f ",point.x);
fclose (pFileX);

//File with the y coordinates
FILE * pFileY;
pFileY = fopen ("../haptic_y.m","w");
fprintf (pFileY, "y_t=[%f ",point.y);
fclose (pFileY);
```

A.2 Matlab

In the second step we use Matlab to elaborate the haptic system's positions (see Figure A.3) and to find new parametrization curves, as explained in chapter 4. We

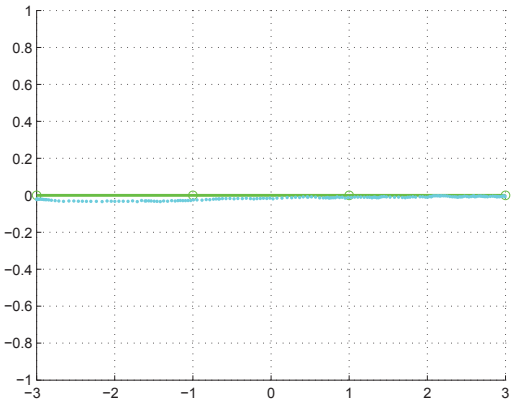


Figure A.3: Visualization of haptic system's positions (light blue) on the robot trajectory (green line).

summarize all the steps explained in section 4.3 by the Matlab code:

```
%Haptic system's positions from Visual studio

haptic_x;
haptic_y;
plot(x_t(:),y_t(:),'c.','Markersize',7);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%1. Projection: "haptic" curve parametrisation

for i=1:length(x_t)-1
    spH(i)=sqrt((x_t(i+1)-x_t(i))^2+(y_t(i+1)-y_t(i))^2);
end

for i=2:length(x_t)-1
    sH(1)=0;
    sH(i)=sH(i-1)+spH(i-1);
    if sH(i)>L
        sH(i)=L;
    end
end
```

```
sH(length(x_t))=L;

for i=2:length(sH)-1
    y_e_haptic(i)=sH(i+1);
end

% Ease curve plot for the projection haptic positions

figure(2)
hold on
plot(x_e_haptic,y_e_haptic,'.-r','MarkerSize',7,'LineWidth',1)

% Parametrisation curve for the projection haptic positions

[t_param]=ease_param(sH,t,w,xv,yv,x_t,L);
[x_haptic, y_haptic]=bezier_rat(t_param,w,xv,yv);

figure(1)
hold on
plot(x_haptic,y_haptic,'.r','MarkerSize',7);
axis equal

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%2. Fitting: "ideal" Ease Curve

x_e_haptic=0:24/(length(spH)-1):24;
pcoeff=polyfit(x_e_haptic,y_e_haptic,4);
xp=0:0.1:24;
yp=polyval(pcoeff,xp);

x_e_ideal=0:24/(length(spH)):24;
y_e_ideal=polyval(pcoeff,x_e_ideal);

% Ease curve plot for the ideal positions
figure(2)
hold on
plot(x_e_ideal,y_e_ideal,'.-g','MarkerSize',7,'LineWidth',1)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 3."Ideal" curve parametrisation

[t_param]=ease_param(y_e_ideal,t,w,xv,yv,x_t,L);
[x_ideal, y_ideal]=bezier_rat(t_param,w,xv,yv);
```

```
% Parametrisation curve for the projection haptic positions

figure(1)
hold on
plot(x_medi,y_medi,'.k','MarkerSize',7);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%4. Blending: "intermediate" Ease Curves

for i=1:length(x_e_haptic)
   d(i)=abs(y_e_haptic(i)-y_e_ideal(i));
end

for i=1:length(x_e_haptic)
   lambda(i)=(d(i)/norm(d));
end

for i=1:length(x_e_haptic)
    y_e_middle(i)=((1-lambda(i)) * y_e_haptic(i)+(lambda(i))*y_e_ideal(i));
% Ease curve plot for the intermediate positions
    figure(2)
    hold on
    plot(x_e_haptic(i),y_e_middle(i),'.k')
end

figure(2)
hold on
plot(x_e_haptic,y_e_middle,'-k')


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%5."Intermediate" curve parametrisations

[t_param]=ease_param(y_e_middle,t,w,xv,yv,x_t,L);
[x_middle, y_middle]=bezier_rat(t_param,w,xv,yv);

figure(1)
hold on
plot(x_middle,y_middle,'.k','MarkerSize',7);
```

We export these new positions in a 3D Studio Max'script as follows:

```
maxscript(x_haptic,y_haptic);
maxscript(x_ideal,y_ideal);
maxscript(x_middle,y_middle);
```

```
//File with the x coordinates
FILE * pFileX;
pFileX = fopen ("../haptic_x.m","w");
fprintf (pFileX, "x_t=[%f ",point.x);
fclose (pFileX);

//File with the y coordinates
FILE * pFileY;
pFileY = fopen ("../haptic_y.m","w");
fprintf (pFileY, "y_t=[%f ",point.y);
fclose (pFileY);
```

A.3 3D Studio Max

In the third step we use 3D Studio Max to visualize (see Figure A.4) and to adjust the robot and camera trajectories from Matlab's data. In the following we present

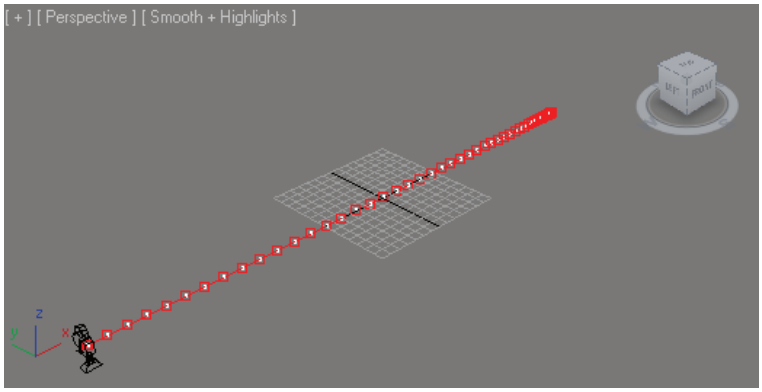


Figure A.4: Visualization of camera's trajectory from Matlab's data.

the 3D Studio Max script to obtain the camera's trajectory in Figure A.4:

```
px=#(-2.971, -2.81582, -2.65421, -2.48705, -2.31515, -2.13934, -1.96036, -1.77899,
-1.59591, -1.41183, -1.2274, -1.04323, -0.859931, -0.67806, -0.498155, -0.320721,
-0.146235, 0.024858, 0.192141, 0.355227, 0.51376, 0.667412, 0.815885, 0.958913,
1.09626, 1.22771, 1.35309, 1.47225, 1.58508, 1.69148, 1.79139, 1.88478, 1.97167,
 2.05206, 2.12603, 2.19366, 2.25507, 2.31042, 2.35987, 2.40365, 2.44198, 2.47513,
  2.50341, 2.52714, 2.54667, 2.5624, 2.57474, 2.58414, 2.59107, 2.59604, 2.59959,
   2.60227, 2.6047, 2.60748, 2.61128, 2.61678, 2.6247, 2.63577, 2.65078, 2.67053,
   2.69584, 2.72759, 2.76667, 2.81399)
py=#(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
```

```
pz=#(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)

pathObj=$Camera001

for i = 1 to 64 do(
px[i]=px[i]*100
py[i]=py[i]*0
pz[i]=pz[i]*0
)
animate on
(
    for t = 1 to 64 do at time t(
        pathObj.position = [px[t],0,0]
    )
)
```

Now, we re-scale (see Figure A.5) and re-time (see Figure A.6) data to adjust them to the model stage and to the storyboard.

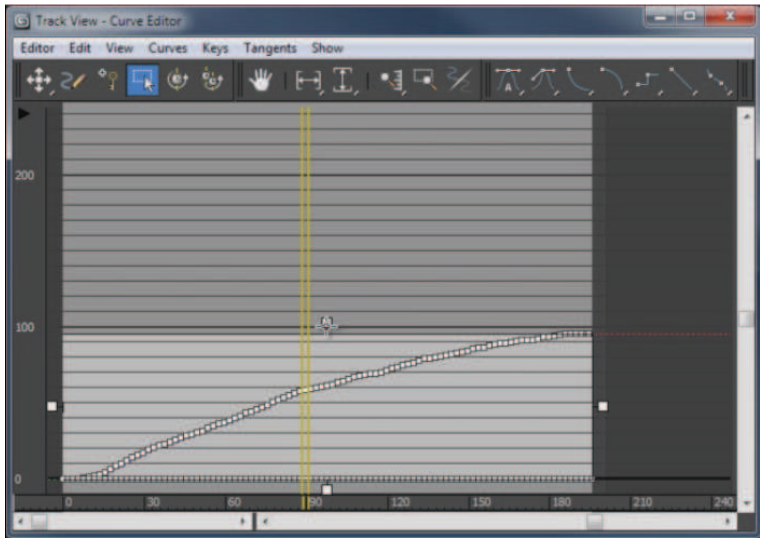


Figure A.5: 3D Studio max trajectory's re-scaling.

When positions are arranged with our storyboard, we export the camera animation to the stop motion software, as shown in Figure A.7.

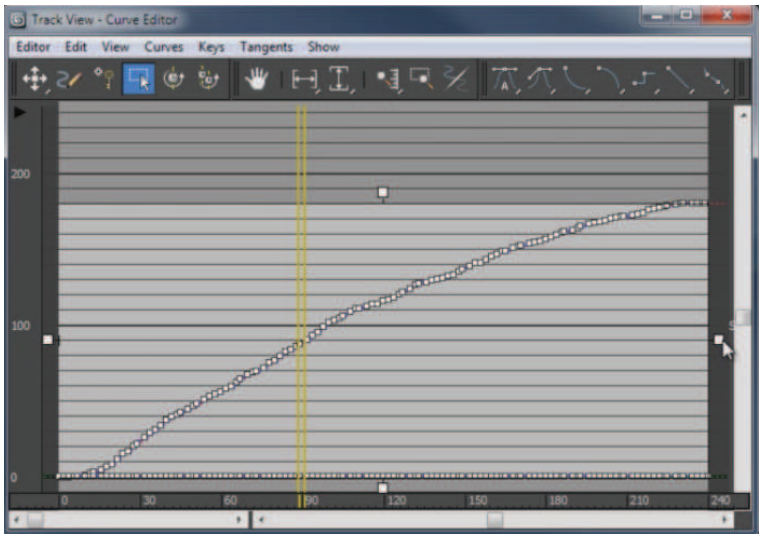


Figure A.6: 3D Studio max trajectory's Re-timing.

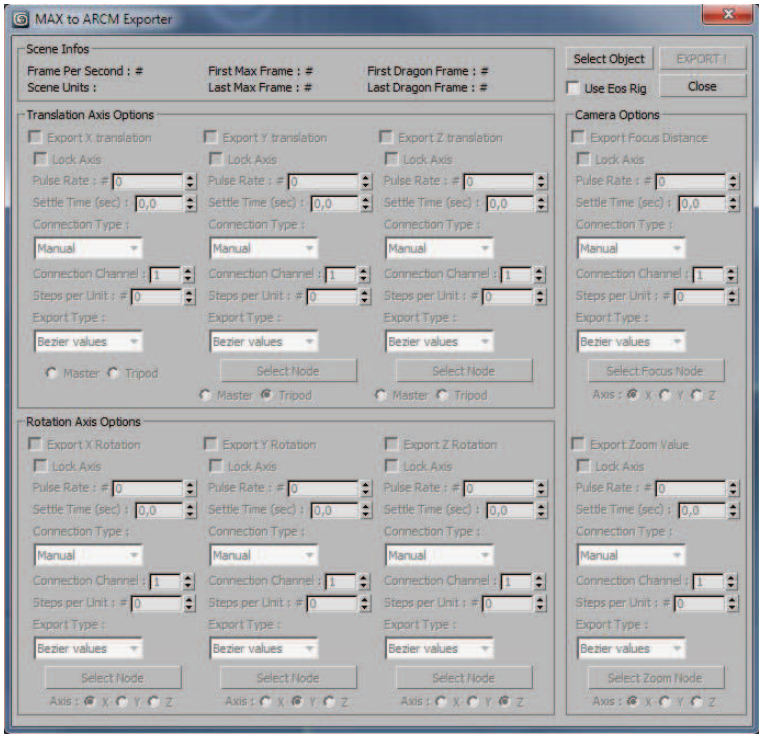


Figure A.7: Screen shot of 3D Studio Max to export the camera animation to the stop motion software.

A.4 Motion control software

The 3D Studio Max data are elaborated with a software which include motion control options. We can choose between a specific software we developped or Dragonframe

software. Both are based on an Arduino system, as explained in section 4.2.3. This choice allows us to diffuse our system to a largest audience. In subsections A.4.1 and A.4.2 we describe how, with one or the other, we can control and calibrate the robot and the camera.

A.4.1 Open source Motion control software: home made

We describe the subsequent versions of our home made motion control software that allows to import the 3D Studio Max's data.

- Version 0.1

  In Figure A.8 we can see a screen shot of the first version. It is a very simple version related to the first prototype of our system. It can not control the camera, but only the robot.

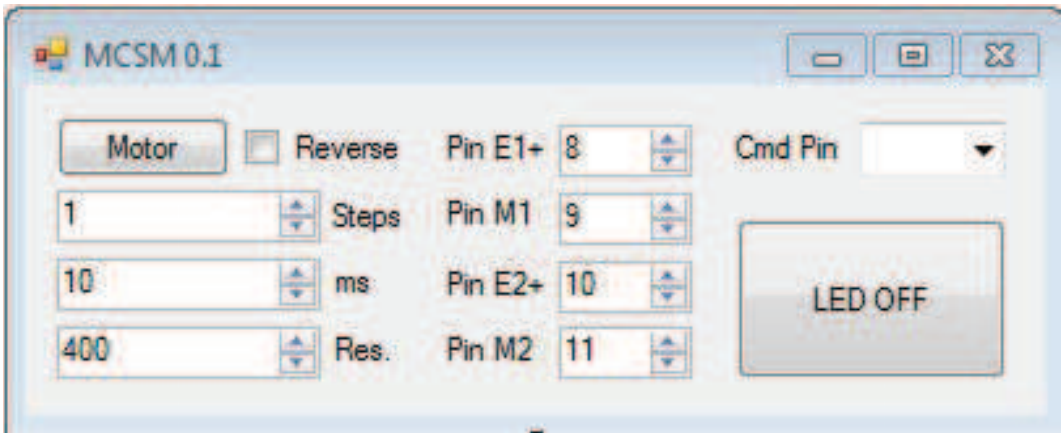


Figure A.8: Version 0.1 of the home made motion control software.

- Version 0.2

  In Figure A.9 we can see a screen shot of the second version. We have three motors that allow us to control robot translation and camera rotations.

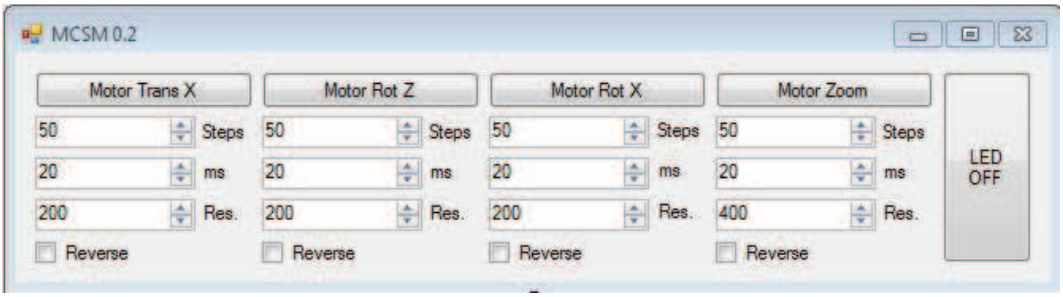


Figure A.9: Version 0.2 of the home made motion control software.

- Version 0.3

  The version 0.3 is shown in Figure A.10. We can control three translation axes and three rotation axes as well as the zoom and the focus of the camera. We use a general electronic system, as shown is Figure A.11.

  - Version 0.3.1

    The current version is improved by the use of an Arduino system. The stop motion system is calibrated (Figure A.12), camera and sequence settings are adjusted (Figure A.13) and finally the shooting sequence is launched (Figures A.14(a) and A.14(b)).

A.4.2 Professional Motion control software: Dragonframe

Our system addresses amateurs as well as professionals in this field. The first category can't afford a motion control software. For this reason, we created a home made motion control software. On the other hand, for their productions many professionals use the commercial software for stop motion, called Dragonframe (http://www.dragonframe.com/). It is a proprietary motion editor software which is not based on common licences featuring numerous tools that help an animator to capture frames and to produce a stop motion animation. Therefore, to trade our system to professionals as well, it also allows to use the Dragonframe software. In Figures A.15 and A.16 we can see two examples of the Dragonframe's motion control interface.

A.5 Motion control robot and digital camera

To execute the sequence we move the robot and the camera to take photos. We detail now the mechanical and electronical system's part. The construction of the robot required different steps. We describe different versions of our robot from the first until the last one and we present also our future developments.

- Version 0.1

  The very first robot (see Figure A.17) was made with the pieces of a scanner. The robot never worked well enough to make a video but its limits gave the first practical basis for the conception of the next version.

- Version 0.2

  The main purpose of the second version (see Figure A.18) was to find how powerful the steppers motors must be to move a professional camera. Tests showed that with an optimized gear train, the motors we found could move more than 20 Kg. It could move only along one translation axis, without controlling rotation axes. With this prototype, we realized the first complete shooting test. This allowed us to approve our project.

- Version 0.3

  This is the first version with three degrees of freedom (see Figure A.19) and allowing to move like a dolly. Much more elaborated, its precision is 0,225° for rotations and 0,1 mm for translation. It is the first prototype that moves along rails, with a metal armature.

  - Version 0.3.1

    This version of the robot constitues an important evolution that replaces the old electronic motor control with a full Arduino interface. It is the first version to be fully based on Creative Commons Licenses, that is one of several public copyright licenses that allow the distribution of copyrighted works (http://creativecommons.fr/). We give people the right to share, use, and even build upon our work. We can see a 3D version in Figure A.20 and a real one in Figure A.21. With this version some tests with the Paris-based stop motion production company *Two Left Hands* (http://www.twolefthands.fr/) have been made. These tests allowed us to improve the rotation axes precision.

  - Version 0.3.2

    We are working on this version. Its features are being listed, based on the tests made with version 0.3.1. We added control focus and zoom movements, but they are not combined with two rotation axes. Main improvements will be: precision, degrees of freedom and position sensors. It is extremely precise (1/8000° of degrees) to be able to make smooth focus change even at macro distance.

A.5.1 Future developments: towards Version 0.3.3

The analysis in the previous section as well as a short positive test of the system by the Paris-based stop motion production company *Two Left Hands* (http://www.twolefthands.fr/) brings us to the conclusion that our system can be proposed as a new method to help animators realize a realistic camera movement for stop motion animation. For our future work we would like to improve the motion capture rotation, e.g., by combining it with the translation movement, and we would like to create an open source system in which a central part (Motion control software) synchronizes the three stages of the process (Camera movement management, Motion capture device and Motion control robot and digital camera), as summarized in Figure A.22.

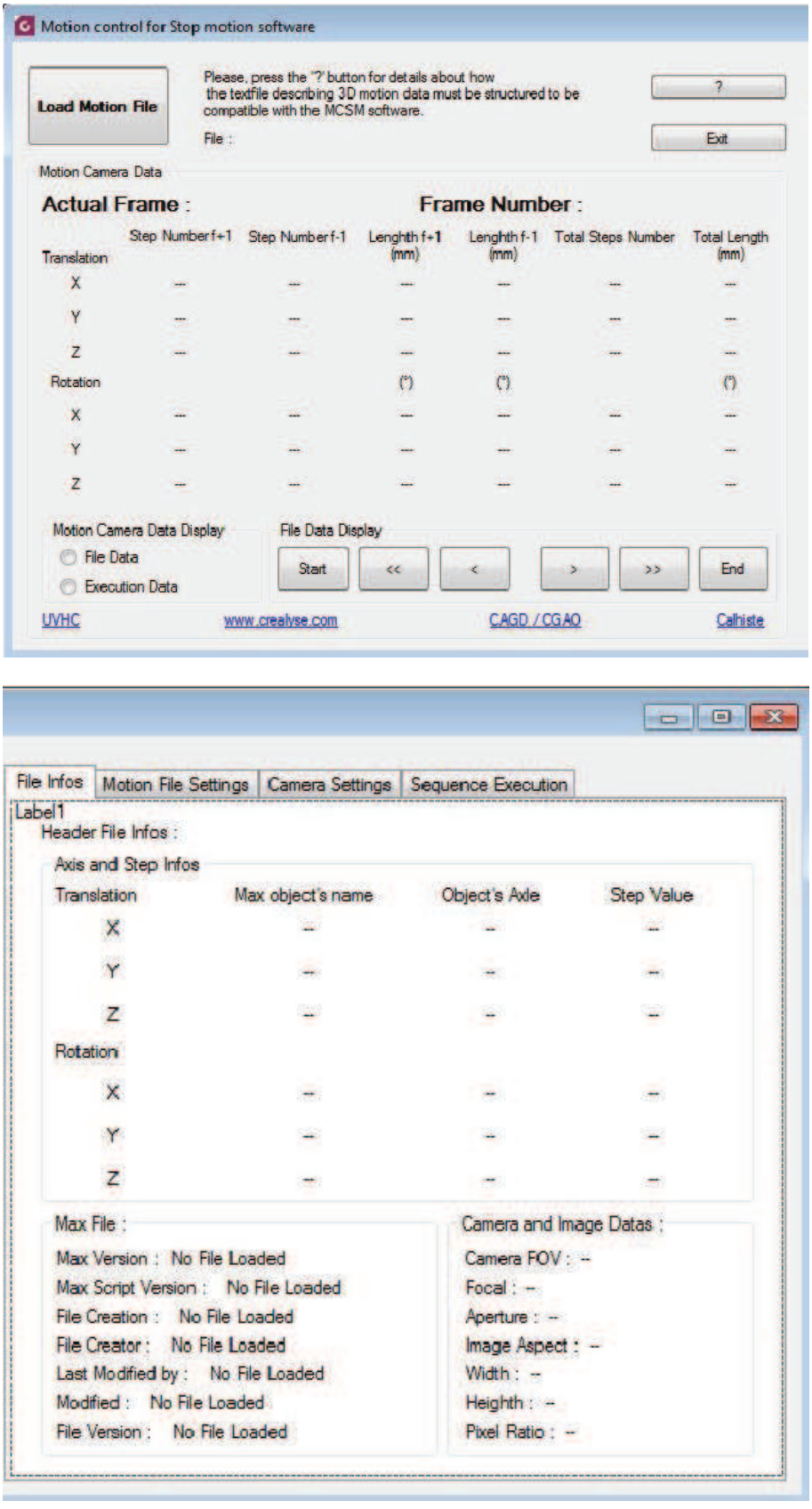
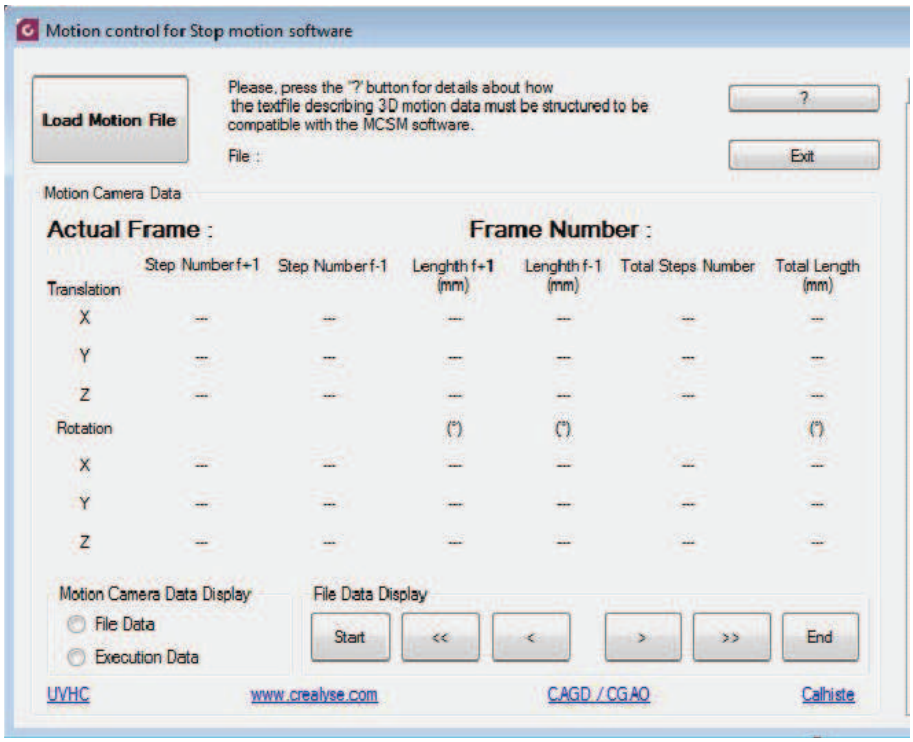


Figure A.10: Screen shot of the interface (version 0.3) to import 3D Studio Max's data.



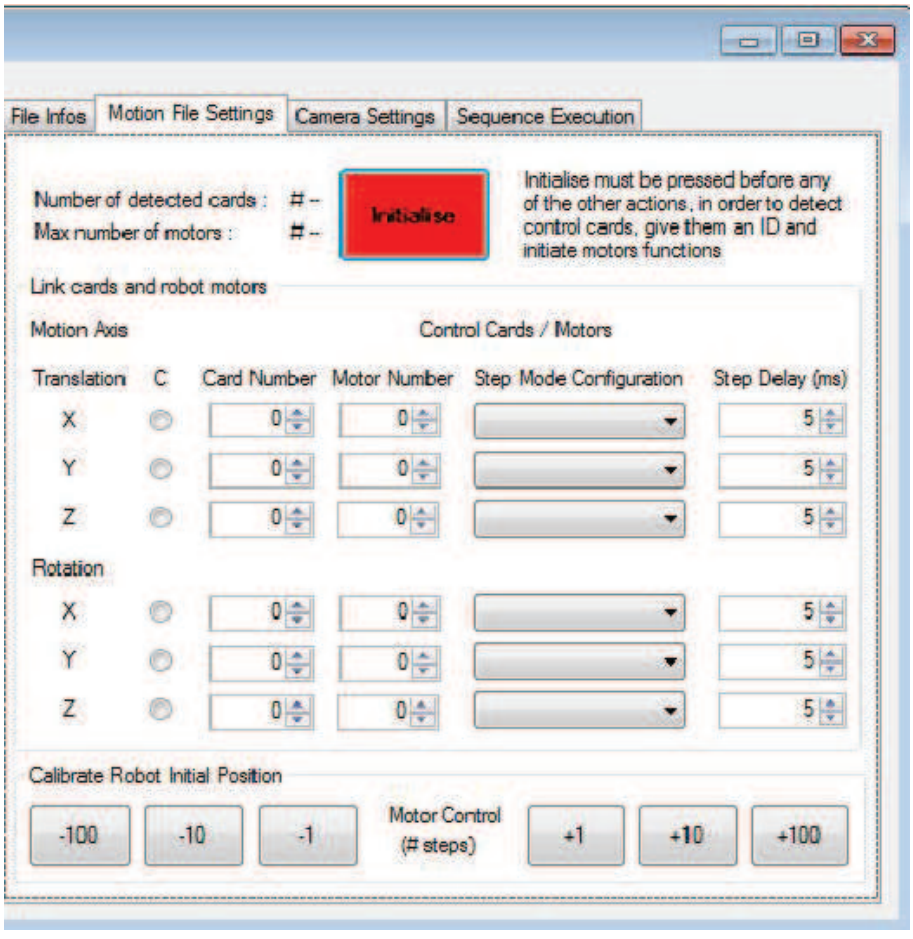


Figure A.11: Screen shot of the interface (version 0.3) to calibrate the system.

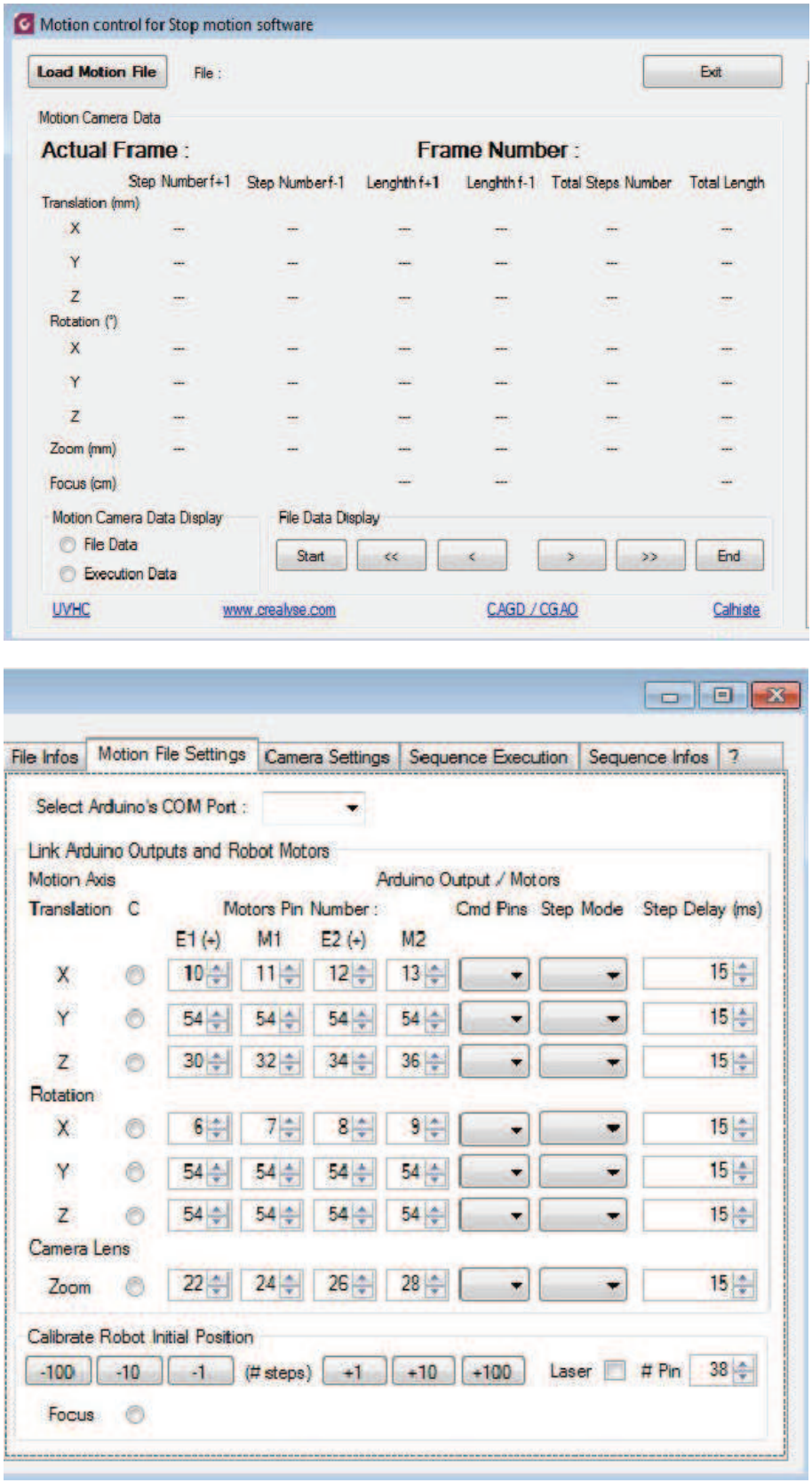


Figure A.12: Screen shot of the interface (version 0.3.1) to calibrate the system with an Arduino interface.

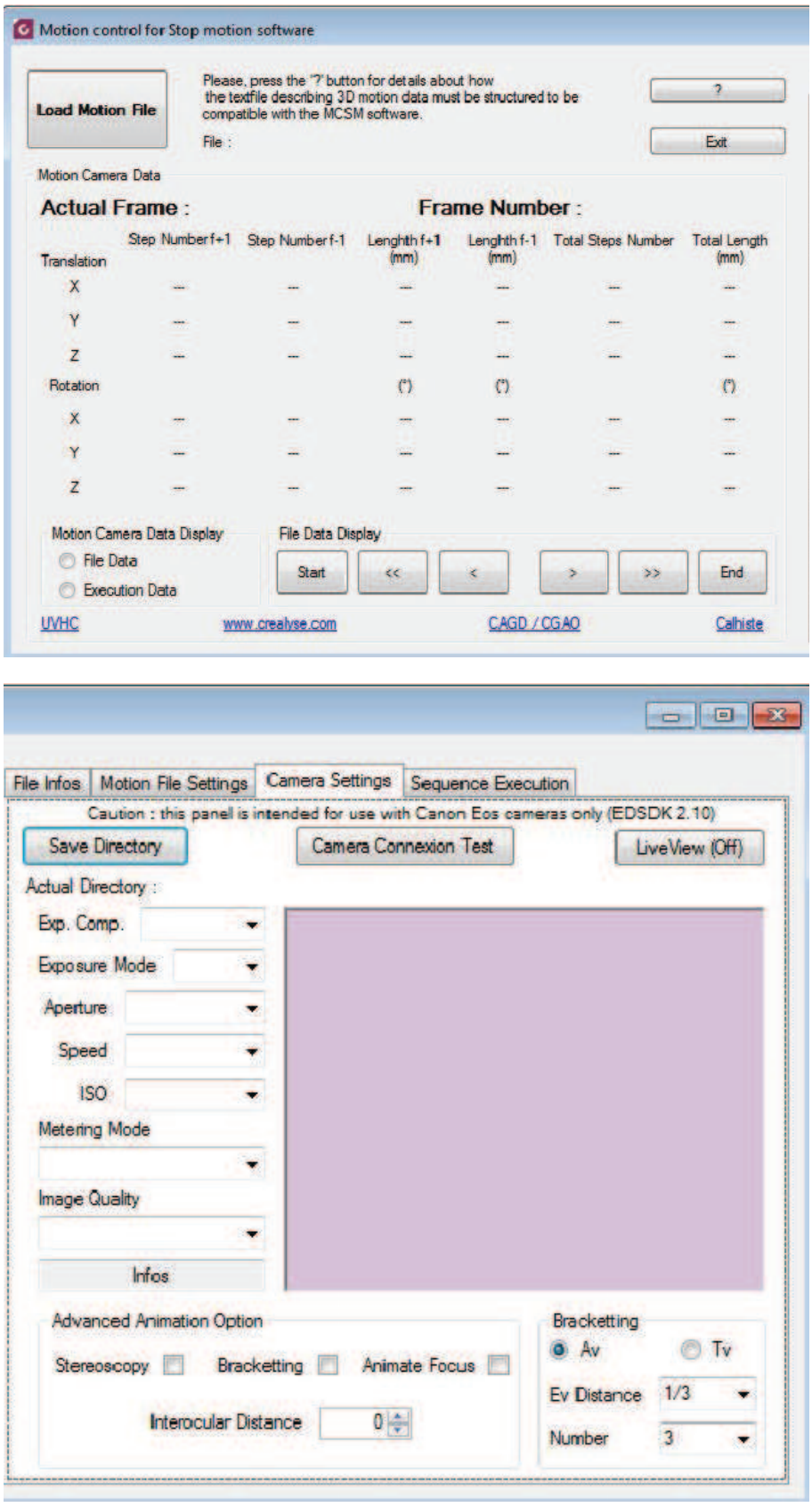
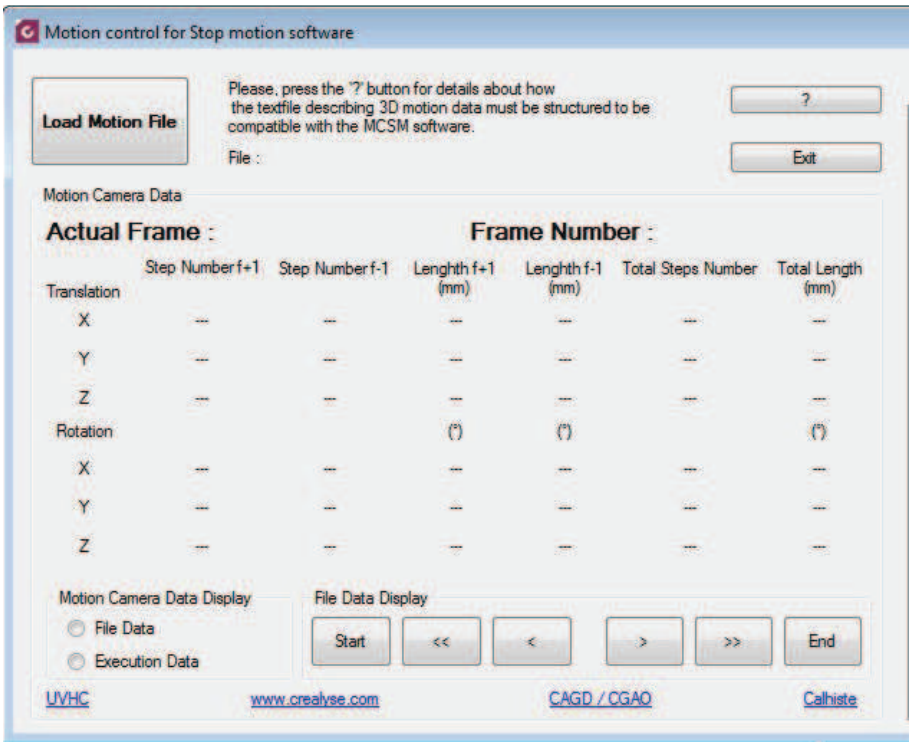


Figure A.13: Screen shot of the interface (version 0.3.1) to adjust camera and sequence settings.



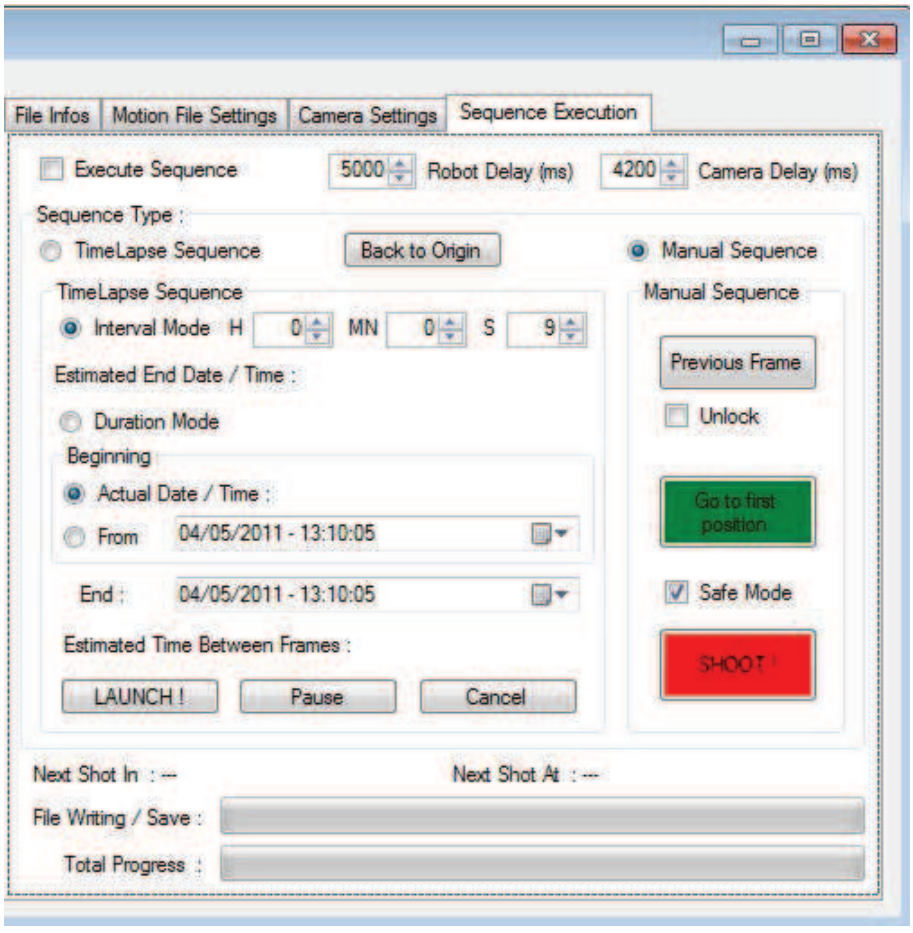


Figure A.14: Screen shot of the interface (version 0.3.1) to launch the shooting sequence.

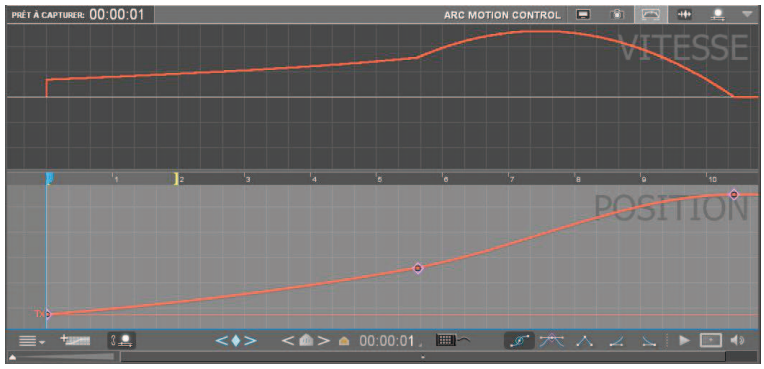


Figure A.15: Screen shot of the Dragonframe interface for a 3D camera animation.

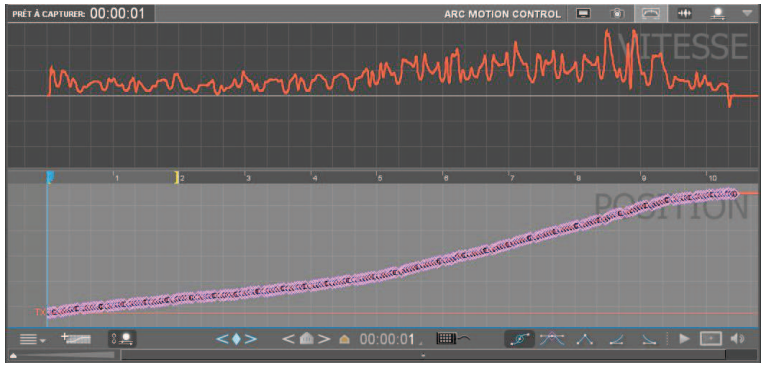
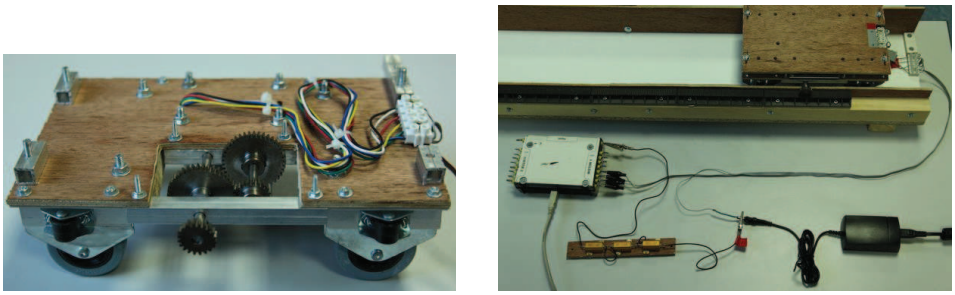


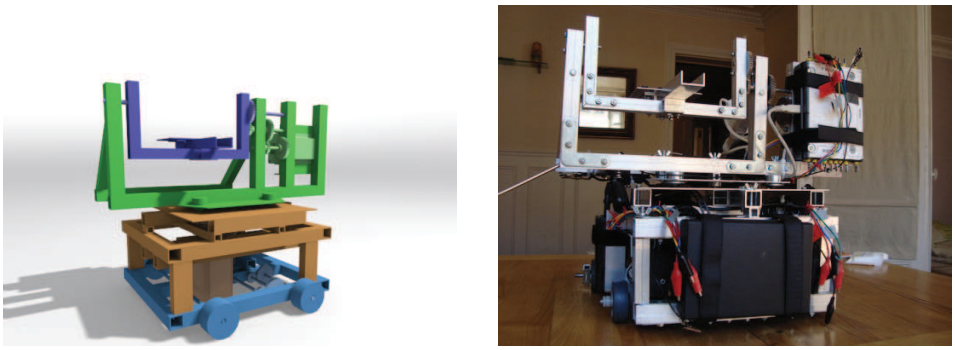
Figure A.16: Screen shot of the Dragonframe interface to import a camera animation from Matlab's data.

Figure A.17: Robot Version 0.1



(a) (b)

Figure A.18: Robot Version 0.2



(a) (b)

Figure A.19: Robot Version 0.3 in 3D (Figure A.19(a)) and real (Figure A.19(b)).

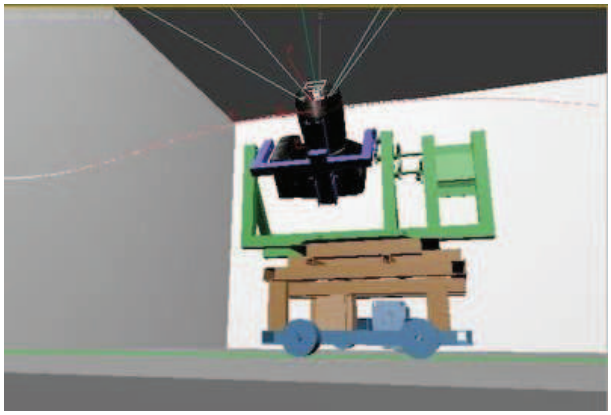
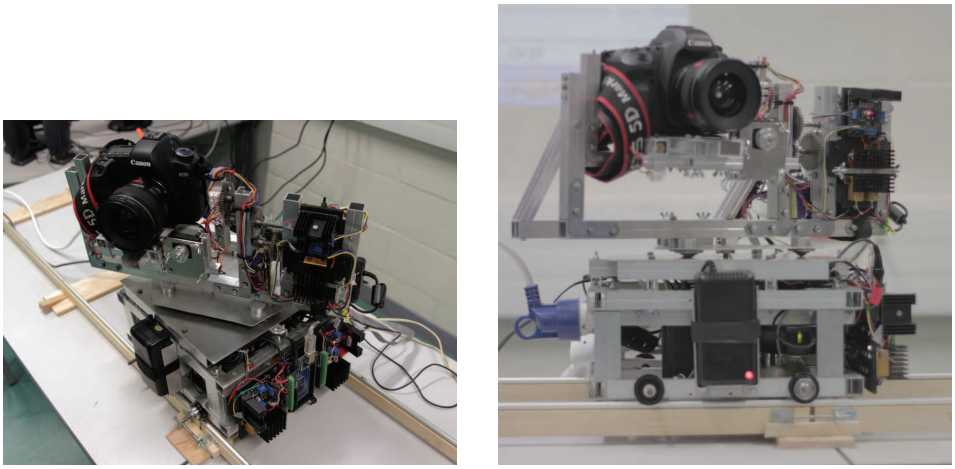


Figure A.20: Robot 3D Version 0.2.1



(a) (b)

Figure A.21: Robot Version 0.2.1

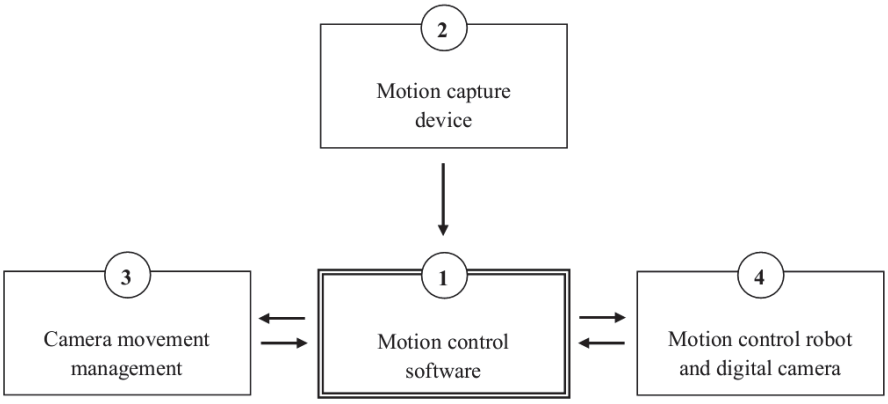


Figure A.22: Future Motion control system diagram.

Bibliography

[Carnicer 1994] J.M. Carnicer and J.M. Pena. *Totally positive bases for shape preserving curve design and optimality of B-splines.* Comput. Aided Geom. Design 11, pages 633–654, 1994. (Cited on page 41.)

[Carnicer 2004] J.M. Carnicer, E. Mainar and J.M. Pena. *Critical length for design purposes and extended Chebyshev spaces.* Constr. Approx., vol. 20, pages 55–71, 2004. (Cited on pages 39 and 42.)

[Derakhshani 2009] D. Derakhshani. Introducing Maya 2009. SYBEX Inc., Alameda, CA, USA, 2009. (Cited on pages 2 and 11.)

[Eberly 2008] D. Eberly. *Moving along a curve with specified speed.* Preprint, see http://www.geometrictools.com, 2008. (Cited on pages 2 and 16.)

[Farouki 1990] R.T. Farouki and T. Sakkalis. *Pythagorean hodographs.* IBM J. Res. Develop., vol. 34, pages 736–752, 1990. (Cited on page 39.)

[Farouki 1995] R.T. Farouki and C.A. Neff. *Hermite interpolation by pythagorean hodograph quintics.* Math. Comp., vol. 64, no. 212, pages 1589–1609, 1995. (Cited on pages 39, 40, 55, 65 and 67.)

[Farouki 1997] R.T. Farouki. *Pythagorean-hodograph quintic transition curves of monotone curvature.* Computer-Aided Design, vol. 29, no. 9, pages 601–606, 1997. (Cited on pages 39, 40, 73, 80 and 82.)

[Farouki 2000a] R.T. Farouki, H.P. Moon and B. Ravani. *Algorithms for Minkowski products and implicitly-defined complex sets.* Adv. Comput. Math., vol. 13, pages 199–229, 2000. (Cited on page 72.)

[Farouki 2000b] R.T. Farouki, H.P. Moon and B. Ravani. *Minkowski geometric algebra of complex sets.* Geometriae Dedicata, 2000. (Cited on page 72.)

[Fünfzig 2010] C. Fünfzig, P. Thomin and G. Albrecht. *Haptic manipulation of rational parametric planar cubic using shape constraints.* SAC '10 Proceedings of the 2010 ACM Symposium on Applied Computing, pages 1253–1257, 2010. (Cited on pages 2, 24 and 92.)

[Govil-Pai 2004] S. Govil-Pai. Principles of computer graphics: theory and practice using OpenGL and Maya. Springer, 2004. (Cited on pages 2 and 14.)

[Guenter 1990] B. Guenter and R. Parent. *Computing the arc length of parametric curves.* IEEE Computer Graphichs and Applications, vol. 10, no. 3, pages 72–78, May 1990. (Cited on pages 2 and 16.)

[Guggenheimer 1963] H. Guggenheimer. *Pythagorean-hodograph quintic transition curves of monotone curvature.* Differential Geometry., 1963. (Cited on page 83.)

[Habib 2008] Z. Habib and M. Sakai. *Transition between concentric or tangent circles with a single segment of G^2 PH quintic curve.* Comput. Aided Geom. Design., vol. 25, pages 247–257, 2008. (Cited on pages 39, 40, 85, 86 and 87.)

[Harryhausen 2008] R. Harryhausen and T. Dalton. A century of model animation. Aurum Editions, 2008. (Cited on page 9.)

[Hongling 2002] W. Hongling, J. Kearney and K. Atkinson. *Arc-length parameterized spline curves for real-time simulation.* In in Proc. 5th International Conference on Curves and Surfaces, pages 387–396, 2002. (Cited on pages 2 and 16.)

[Kochanek 1984] D.H.U. Kochanek and R.H. Bartels. *Interpolating splines with local tension, continuity, and bias control.* SIGGRAPH Comput. Graph., vol. 18, no. 3, pages 33–41, 1984. (Cited on pages 2 and 15.)

[Lasseter 1987] J. Lasseter. *Principles of traditional animation appplied to 3D computer animation.* Computer Graphics., vol. 21, no. 4, pages 35–44, 1987. (Cited on page 1.)

[Mainar 2001] E. Mainar, J.M. Peña and J. Sánchez-Reyes. *Shape preserving alternatives to the rational Bézier model.* Comput. Aided Geom. Design., vol. 18, pages 37–60, 2001. (Cited on pages 39, 45, 47 and 48.)

[Mainar 2007] E. Mainar and J.M. Peña. *A general class of Bernstein-like bases.* Computers and Mathematics with Applications, vol. 53, pages 1686–1703, 2007. (Cited on pages 39 and 44.)

[Mainar 2010] E. Mainar and J.M. Peña. *Optimal bases for a class of mixed spaces and their associated spline spaces.* Computers and Mathematics with Applications, vol. 59, pages 1509–1523, 2010. (Cited on pages 39, 45, 47, 48, 49 and 50.)

[Moon 2001] H.P. Moon, R.T. Farouki and H.I. Choi. *Construction and shape analysis of PH quintic Hermite interpolants.* Computer-Aided Design, vol. 18, pages 93–115, 2001. (Cited on pages 39, 40, 70 and 72.)

[Murdock 2001] K.L. Murdock. 3d Studio Max 4 Bible. Wiley Publishing, 2001. (Cited on pages 2 and 11.)

[Parent 2004] R. Parent. Computer Animation: Algorithms and techniques. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. (Cited on pages 2 and 16.)

[Piegl 1995] L. Piegl and W. Tiller. The Nurbs Book. Springer-Verlag, London, UK, 1995. (Cited on page 34.)

[Romani 2004] L. Romani and M.A. Sabin. *The conversion matrix between uniform B-spline and Bézier representations.* Comput. Aided Geom. Design., vol. 21, no. 6, pages 549–560, 2004. (Cited on page 50.)

[Romani 2009] L. Romani. *From approximating subdivision schemes for exponential splines to high-performance interpolating algorithms.* J. Comput. Appl. Math., vol. 224, no. 1, pages 383–396, 2009. (Cited on pages 49, 51 and 52.)

[Saini 2010] L. Saini, G. Albrecht, N. Lissarrague and L. Romani. *Animation 3D: le problème de mouvement de caméra.* Proceedings of the Journées du Groupe de Travail en Modélisation Géométrique (GTMG), pages 69–76, Dijon, 2010. (Cited on pages 3, 27 and 30.)

[Saini 2011] L. Saini, G. Albrecht, N. Lissarrague, L. Romani, C. Fünfzig and J.P. Bécar. *Animation 3D: mouvements de caméra réalistes pour la stop motion.* Proceedings of the 11th international conference H2PTM (Hypertextes and Hypermédias, Produits, Outils and Méthodes), pages 137–148, Hypermédias et pratiques numériques, University of Metz, Hermes Science/Lavoisier, October, 2011. (Cited on page 21.)

[Sánchez-Reyes 1998] J. Sánchez-Reyes. *Harmonic rational Bézier curves, p-Bézier curves and trigonometric polynomials.* Comput. Aided Geom. Design., vol. 15, pages 909–923, 1998. (Cited on page 42.)

[Sharpe 1982] R.J. Sharpe and R.W. Thorne. *Numerical method for extracting an arc length parameterization from parametric curves.* Computer Aided Design, vol. 14, no. 2, pages 79–81, March 1982. (Cited on pages 2 and 16.)

[Snibbe 1995] S.S. Snibbe. *A Direct Manipulation Interface for 3D Computer Animation.* Computer Graphics Forum, vol. 14, no. 3, pages 271–284, 1995. (Cited on pages 2 and 18.)

[Steketee 1985] S.N. Steketee and N.I. Badler. *Parametric keyframe interpolation incorporating kinetic adjustment and phrasing control.* SIGGRAPH Comput. Graph., vol. 19, no. 3, pages 255–262, 1985. (Cited on pages 2 and 19.)

[Verth 2004] J.V. Verth and L. Bishop. Essential mathematics for games and interactive applications: A programmer's guide. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. (Cited on pages 2 and 14.)

[Watt 1991] A. Watt and M. Watt. Advanced animation and rendering techniques. ACM, New York, NY, USA, 1991. (Cited on pages 2 and 19.)

Abstract

In the first part, we present a new system that allows to create realistic camera movements for a stop motion animation. The system improves traditional 3D software animation programs (for example Maya and 3D Studio Max) for creating stop motion camera movements by using an haptic interface. After describing the whole system, we explain in detail the mathematical processing to obtain different camera movements by using an haptic interface for motion capture. The recorded haptic positions, once elaborated, are exported, frame by frame, to the motion control software, which allows to calibrate the motion control robot, to control the camera settings and, finally, to execute the sequences. A class of students of the "Art plastiques et Création numérique" Master of the University of Valenciennes evaluated the system.
In the second part, we define a new class of Pythagorean Hodograph curves built upon a five dimensional mixed algebraic trigonometric space, and show their fundamental properties and important advantages over their well known polynomial counterpart. A complex representation for these curves is introduced and constructive approaches are provided to solve the first order Hermite interpolation problem.

Keywords: stop motion, motion control, 3D animation, camera movement, realistic simulation, pythagorean hodograph, trigonometric functions, generalized Bézier curves, Hermite interpolation, spirals.

Résumé

Dans la première partie de la thèse, nous présentons un nouveau système permettant de produire des mouvements de caméra réalistes pour l'animation stop motion. Le système permettra d'enrichir les logiciels d'animation 3D classiques (comme par exemple Maya et 3D Studio Max) afin de leur faire contrôler des mouvements de caméra pour la stop motion, grâce à l'utilisation d'une interface haptique. Nous décrivons le fonctionnement global du système. La première étape consiste à récupérer et enregistrer les données envoyées par le périphérique haptique de motion capture. Dans la seconde étape, nous réélaborons ces données par un procédé mathématique, puis les exportons vers un logiciel de 3D pour prévisualiser les mouvements de la caméra. Finalement la séquence est exécutée avec un robot de contrôle de mouvement et un appareil photo. Le système est évalué par un groupe d'étudiants du Master "Art plastiques et Création numérique" de l'Université de Valenciennes.
Dans la deuxième partie, nous définissons une nouvelle classe de courbes à partir des courbes polynomiales paramétriques à hodographe pythagorien (PH) construite sur un espace algébrique-trigonométrique. Nous montrons leurs propriétés fondamentales et leurs avantages importants par rapport à leur équivalent polynomial, grâce à l'utilisation d'un paramètre de forme. Nous introduisons une formulation complexe et nous résolvons le problème d'interpolation de Hermite.

Mots clés : stop motion, mouvement de contrôle de caméra, animation 3D, simulation réaliste, 3D hodographe pythagorien, fonctions trigonométriques, courbes de Bézier généralisées, problème d'interpolation de Hermite, spirals.