Constrained, Interactive Design of Framed Curves

Interaktive Auslegung von Dreibeinkurven unter geometrischen und dynamischen Randbedingungen

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Stephan Alt

Tag der mündlichen Prüfung:	15.12.2025
1. Referent:	Prof. Dr. Hartmut Prautzsch (IVD)
2. Referent:	Prof. Dr. Tamim Asfour (H^2T)

Abstract

This work covers methods that facilitate the design of spatial paths that are to be traced by a system of rigid bodies, such as a car, train or bobsled. In these applications, the path geometry and the resulting dynamics of the rigid bodies are closely tied to each other. The path is not only seen, but also felt by the bodies or any human patron sitting inside. This imposes additional constraints on the underlying geometric object. Similarly as for spatial trajectories of rigid bodies, we need to specify both the position and the orientation of the path. We will model such paths using *framed curves*, geometric objects composed of a spatial curve and its *moving frame*.

In this thesis, we introduce framed curves as a representation for spatial tracks in the context of layout design. We show how, given a particular framed curve, we can evaluate the fairness of the track it represents, check common design constraints and perform various geometric operations, such as constructing *sweep surfaces* and *offset curves*. Furthermore, framed curves can be implemented as general joints in the context of multi body analysis, allowing us to solve the related inverse kinematics problem using established multi body systems (MBS) libraries. We propose an algorithm that approximates the equations of motion of a vehicle efficiently and with sufficient accuracy for our application, the design of roller coaster layouts. The results of this algorithm is verified against both real-world measurements and theoretical ground truth data. We also propose two different modeling techniques for the design of framed curves in an interactive CAGD setting. First, using a *curvature based curve definition*, framed curves may be generated from three input functions using numerical integration. Second, using *polynomial spline interpolation* and *non-linear optimization*, energy minimizing curves and moving frames can be generated using several human-defined interpolation constraints and an application-defined loss function.

Kurzfassung

Diese Arbeit behandelt Algorithmen zum Entwurf und zur Auswertung räumlicher Pfade, welche von Fahrzeugen wie Automobilen, Zügen oder Schlitten befahren werden. Bei diesen Anwendungen gibt es einen engen Zusammenhang zwischen der Geometrie dieser Pfade und der resultierenden Dynamik der involvierten Starrkörper, bei der nicht nur die Position, sondern auch die Ausrichtung dieser Körper eine wesentliche Rolle spielt. Im Gegensatz zu anderen Anwendungen des CAGD müssen die resultierende Pfade also nicht nur schön aussehen, sondern auch eine angenehme Durchfahrung ermöglichen. Hierdurch ergeben sich zusätzliche Anforderungen an die Pfade, die wir als *Dreibeinkurve* (engl. *framed curve*), bestehend aus einer Raumkurve und einem *begleitenden Dreibein*, modellieren werden.

Die vorliegende Arbeit nutzt Dreibeinkurven als Repräsentation für räumlich geformte Trassen (z. B. Schienen oder Straßen). Wir zeigen, wie sich in der Trassierung häufig verwendete Konzepte und Randbedingungen auf Dreibeinkurven abbilden lassen und wie Dreibeinkurven im Rahmen von Mehrkörpersimulationen als kinematisches Gelenk implementiert werden können. Dies ermöglicht ein effizientes Lösen der *inversen Kinematik* mithilfe bestehender Softwarebibliotheken. Wir entwickeln einen Algorithmus, der die Bewegungsgleichungen von Fahrzeugen entlang von Dreibeinkurven für den Entwurf von Achterbahnen effizient und ausreichend genau approximiert, und validieren diesen mithilfe von Ergebnissen exakterer Methoden sowie Messungen bestehender Achterbahnen. Anschließend stellt diese Arbeit interaktive Trassierungsmethoden für Dreibeinkurven vor. Mithilfe einer *krümmungsbasierten Kurvendefinition* können Dreibeinkurven durch numerische Integration erzeugt werden, wobei ein Benutzer drei verschiedene Funktionen als Eingabe spezifiziert. Eine Kombination von *Interpolation durch polynomielle Splines* und *nicht-linearen Optimierungsmethoden* ermöglicht die Approximation von *energieminimierenden Kurven* und begleitenden Dreibeinen.

Contents

List of Figures

List of Abbreviations

CAD	computer aided design
CAGD	computer aided geometric design
NURBS	non-uniform rational B-spline
RMF	rotation minimizing frame
IK	inverse kinematics
MEC	minimum energy curve
SI-MEC	scale invariant minimum energy curve
MVC	minimum variation curve
SI-MVC	scale invariant minimum variation curve
MJC	minimum jerk curve
ASTM	American Society for Testing and Materials
TÜV	Technischer Überwachungsverein Technical Inspection Association
DIN	Deutsches Institut für Normung German Institute for Standardisation
THC	theoretical hourly capacity
SDF	signed distance function
WYSIWYG	what you see is what you get

Notation and Symbols

The following notation is used throughout this thesis. Deviations from this notation for specific sections or equations are explicitly stated.

General Notation

a, b, c	scalar values
β, ψ, ϕ	angles
$a(u), b(t)$	functions
$\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3$	unit vectors of the standard basis
$\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$	vectors
$\mathbf{a}, \mathbf{b}, \mathbf{c}$	homogeneous/extended coordinate vectors (see Section 4.1)
$\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$	matrices
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	homogeneous/extended matrices (see Section 4.1)
$\boldsymbol{a}^\mathsf{T}, \boldsymbol{A}^\mathsf{T}$	vector transpose, matrix transpose
$\boldsymbol{A}^{-1}$	inverse matrix
$\mathcal{A}, \mathcal{B}$	rigid body, reference frame
$\mathrm{Rot}(\boldsymbol{a}, \alpha)$	transformation matrix of the rotation about $\boldsymbol{a}$ by an angle α
$\mathrm{Trans}(\boldsymbol{a})$	transformation matrix of the translation by $\boldsymbol{a}$
$\mathbb{Z}, \mathbb{R}, \mathbb{C}, \mathbb{H}$	set of integers, real/complex numbers, quaternions
$\boldsymbol{A} \cdot \boldsymbol{B}$	multiplication
$\|\cdot\|$	Euclidean norm
$\langle \boldsymbol{a}, \boldsymbol{b} \rangle = \boldsymbol{a}^\mathsf{T}\boldsymbol{b}$	Euclidean inner product, dot product
$\boldsymbol{a} \otimes \boldsymbol{b} = \boldsymbol{a}\boldsymbol{b}^\mathsf{T}$	outer product
$\boldsymbol{a} \times \boldsymbol{b}$	cross product in $\mathbb{R}^3$
$f'(s) = \frac{df}{ds}$	derivative of f with respect to its parameter
$f^{(4)}(s) = \frac{d^4 f}{ds^4}$	higher order derivative
$\frac{df(u)}{ds} = f' \cdot \frac{du}{ds}$	derivative of f with respect to some other parameter s
$\dot{a} = \frac{d\boldsymbol{a}}{dt}$	time derivative of a

In the following, we list symbols and constants that will be frequently used throughout this thesis. In case the underlying concepts are introduced, we also refer to the section in which the respective symbols are defined. In case numerical values are given in figures or tables, we assume the units listed in the rightmost column except where explicitly stated otherwise. For the constants, we specify their values in the rightmost column below.

Frequently Used Symbols

u	general curve parameter	[]
$s, \boldsymbol{p}$	arc length, position	$[m]$
θ, α, γ	yaw, pitch, roll angles (Section 4.4)	[rad]
$\boldsymbol{p}(s)$, $\boldsymbol{p}(u)$	(parametric) curve (Chapter 5)	
κ, τ	curvature and torsion of a curve (Section 5.4)	$[m^{-1}]$
$\boldsymbol{t}$, $\boldsymbol{n}$, $\boldsymbol{b}$	tangent, principal normal and binormal of the Frenet frame (Section 5.4)	
$\boldsymbol{Q}$	rotation matrix, moving frame (Sections 4.3 and 5.6)	
$\mathbf{T}(s)$, $\mathbf{T}(u)$	(parametric) framed curve (Section 5.6)	
κ_g, κ_n, τ_r	geodesic curvature, normal curvature and relative torsion of a framed curve (Section 5.6)	$[m^{-1}]$
$\boldsymbol{t}$, $\boldsymbol{u}$, $\boldsymbol{v}$	tangent and normals of a moving frame (Section 5.6)	
$\boldsymbol{d}$	(generalized) Darboux vector (Sections 5.4 and 5.6)	
t	time	$[s]$
$v, \dot{\boldsymbol{p}}$	linear velocity	$[m/s]$
$a, \boldsymbol{a}$	linear acceleration	$[m/s^2]$
$\dot{a}, \dot{\boldsymbol{a}}$	linear jerk	$[m/s^3]$
$\omega, \dot{\phi}$	angular velocity (Section 6.1)	$[\mathrm{rad}/s]$
$\dot{\omega}, \ddot{\phi}$	angular acceleration (Section 6.1)	$[\mathrm{rad}/s^2]$

Constants

π	ratio of a circle's circumference to its diameter	$3.14159\ldots$
g	gravitational acceleration on earth's surface	$9.80665 m/s^2$

Introduction

Transportation plays a central role in modern societies. Especially land based transportation relies on dedicated transportation ways, which are carefully planned and optimized with respect to wear and tear and passenger comfort during traversal. In particular, the field of railway design has a rich and long history, with continuous innovation to this day. Naturally, other fields and applications in which predefined paths are followed by physical bodies are fed from the vast knowledge that accumulated in the field of railway design. Examples of such applications are the design of sports facilities such as bobsled or luge tracks, velodromes, ski jumping hills or recreational facilities such as waterslides, summer toboggan runs or roller coasters.

The underlying problem of layout design is inherently geometric. As opposed to most applications found in geometric design however, the resulting curves are not only seen, which require them to be pleasing to the eye, but also felt. As a result, a rich interplay between the fields of *Differential Geometry* and *Kinematics* emerges.

Especially in the field of roller coasters, the design of spatial tracks is taken to the extreme. Theme parks often thrive for the next "higher, faster, further" attraction, such that the industry is continuously pushing the envelope in creating more complex maneuvers in terms of track geometry and ride dynamics. This leads to ever decreasing margin for error and thus requires ever more sophisticated methods in layout design on one hand and manufacturing on the other. This thesis focuses on the former, exploring algorithms to interactively design and evaluate the geometry and resulting dynamics of spatial tracks.

1 State of the Art

In this section, we give an overview of the established state of the art in the related fields of *layout design*. As this thesis is largely interdisciplinary, we concentrate on the most important and relevant topics in these fields and encourage the reader to follow the cited literature for more details. An introduction to the fundamentals and the state of the art in *geometric design* and *kinematics* is given in Chapter 3 since these fields are more general and less specific for our underlying application.

1.1 Track Layout Design

The study of railroad track design goes back to the first half of the 18^{th} century [34]. At that time, most of the tracks and roads were exclusively planned for slowly moving coaches and carts. Since then, the design of tracks has been subject of continued research, mainly due to the ever increasing requirements for railways to be more comfortable, to need less maintenance and to traverse tracks at higher velocities. Therefore, it is no longer sufficient to use the most basic geometric design elements *straight lines* and *circular arcs*, since tracks designed this way have instantaneous jumps of curvature leading to jerky movement of the vehicles during traversal. Therefore, *transition curves* such as *parabolic curves* [94] and *clothoids* [47] are widely used, as they allow smooth transitions between straight and curved sections of track.

The current gold standard method of track layout design consists of three distinct steps [74, Chapters 3 and 4].

1. Using various elementary curves, the planar layout $\bar{\boldsymbol{p}}(u) : I \to \mathbb{R}^2$ of the track's center is constructed in a piecewise fashion. It defines the top down orthographic projection of the track's centerline. We denote the arc length of $\bar{\boldsymbol{p}}$ with $\bar{s}$ and the radius of curvature of $\bar{\boldsymbol{p}}$ with $\bar{r}_h$.

2. The *height profile* of the track $z(\bar{s})$ is designed along the arc length of $\bar{\boldsymbol{p}}$. As with the layout, this is achieved by chaining several elementary curves in the z-$\bar{s}$-plane in a piecewise fashion, while ensuring a functional relationship between the projected arc length $\bar{s}$ and the height z. After this step, the resulting spatial curve $\boldsymbol{p} : I \to \mathbb{R}^3$ and its arc length s can be computed.

3. In order to reduce lateral forces on the track, vehicles and passengers, the *superelevation* or *cant* c of the track is defined in a similar fashion to the height profile. The superelevation defines the vertical displacement between the two rails with gauge w of a track and thus is a measure of its *roll angle*

$$\gamma = \arcsin\left(\frac{c}{w}\right), \tag{1.1}$$

the angle at which a passenger would see a perfectly level horizon.

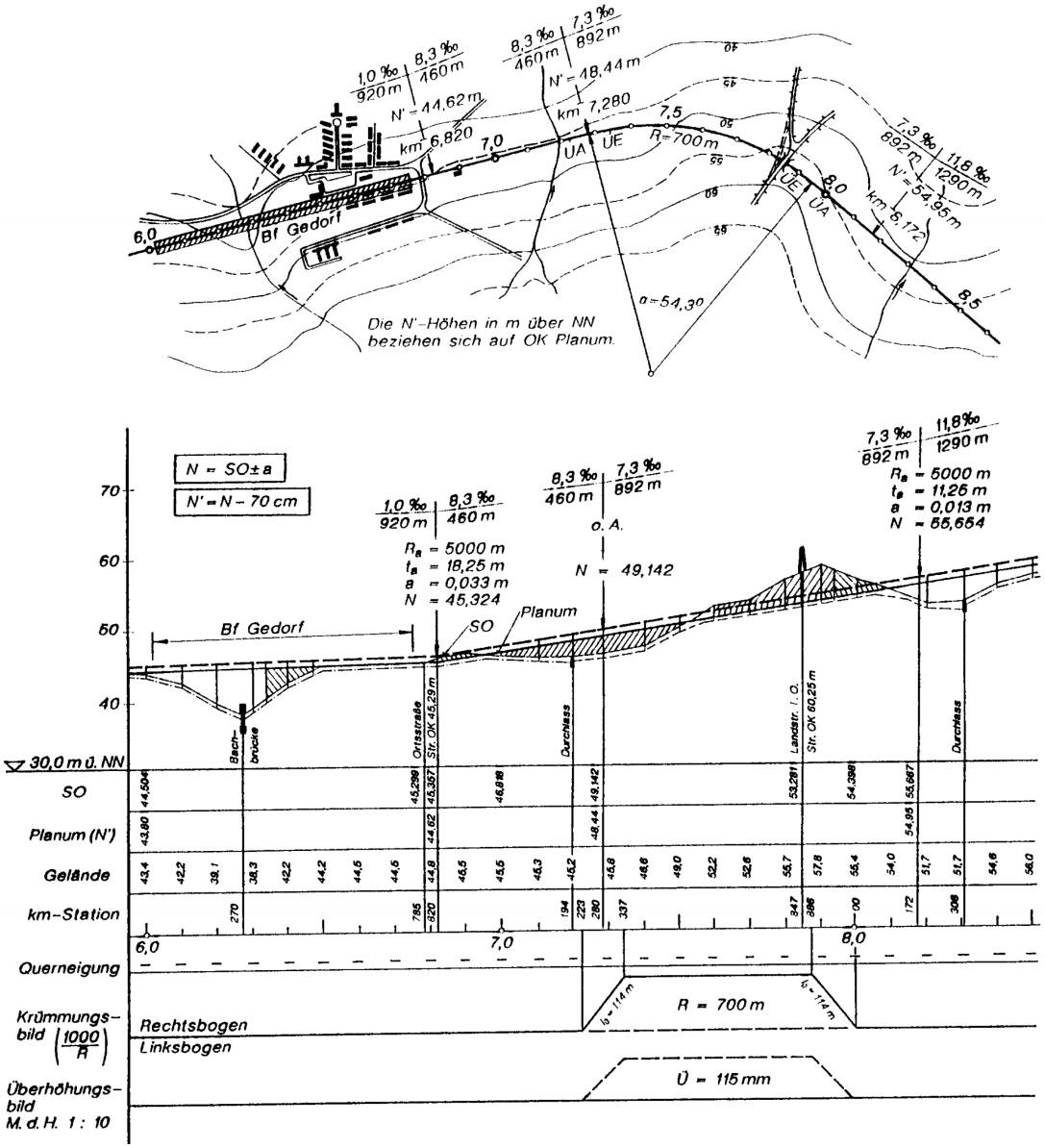


Figure 1.1: Technical drawing of a railroad track [29].

The key advantage of the above procedure is the fact that the design of a spatial track is reduced to three mostly independent planar problems. These are not only easier to solve individually, but also ensure the existence of unambiguous technical drawings of the track layout for planning, manufacturing and installation. Figure 1.1 shows an exemplary technical drawing, including – from top to bottom – the layout plan, the height profile of the track and terrain, the curvature plot and the superelevation plot.

Several properties of traditional tracks facilitate this design method. First, since the propulsion capabilities of trains relative to their weight is quite low, the inclination angle α of tracks should be limited to small absolute angles [67, Section 3.5.1]. This greatly reduces the projective distortion introduced by the top view orthographic

Figure 1.2: The roller coaster *Revolution* is one of the oldest still operational roller coasters with a vertical looping [58].

projection. The resulting spatial track is mostly determined by its characterizing planar layout curve $\bar{\boldsymbol{p}}$. Second, tracks are usually designed for a specific nominal velocity during traversal, which is kept constant for extended sections of the track. Since the track's inclination angle is sufficiently small, the superelevation of a track is mostly dependent on the radius of curvature of the planar layout. Moreover, since the roll angle is also sufficiently small, the superelevation and curvature are approximately proportional using small angle approximations, see Figure 1.1 for reference. Last, combining variations of curvature and superelevation with variations of inclination is highly discouraged [67, Section 3.5.2], further reducing the complexity of the resulting track geometry.

The above way to design spatial tracks is closely related to the *Euler angles*, more specifically to the *yaw, pitch and roll* convention also used in aerospace engineering [87, Section 1.3] and robotics [46, Chapter 2]. The *yaw angle* θ is the tangent angle of $\bar{\boldsymbol{p}}$, the *pitch angle* α the tangent angle of $z(\bar{s})$ and the *banking*

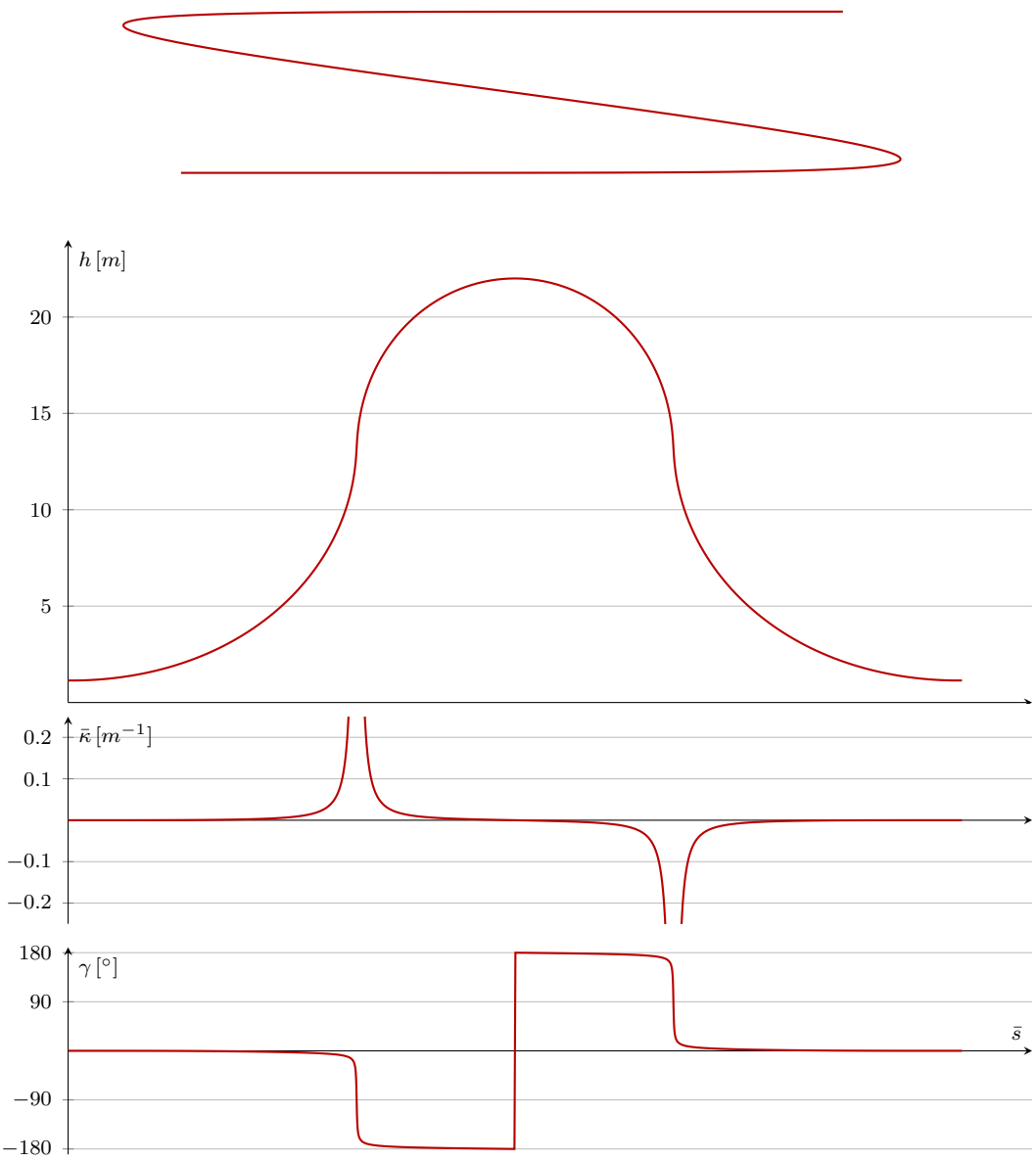


Figure 1.3: A vertical looping constructed by projection onto the x-y-plane. Sections close to the singularity at $\alpha = \pm\pi/2$ are not well behaved.

angle or *roll angle* γ of the track is defined by the superelevation plot according to Equation (1.1). The corresponding rotation matrix

$$\begin{aligned} \boldsymbol{Q} &= \begin{bmatrix} \boldsymbol{t} & \boldsymbol{u} & \boldsymbol{v} \end{bmatrix} \\ &= \begin{bmatrix} C_\theta C_\alpha & -S_\theta C_\gamma + C_\theta S_\alpha S_\gamma & S_\theta S_\gamma + C_\theta S_\alpha C_\gamma \\ S_\theta C_\alpha & C_\theta C_\gamma + S_\theta S_\alpha S_\gamma & -C_\theta S_\gamma + S_\theta S_\alpha C_\gamma \\ -S_\alpha & S_\gamma C_\alpha & C_\gamma C_\alpha \end{bmatrix} \end{aligned}$$

with

$$C_\alpha \coloneqq \cos(\alpha),$$
$$S_\alpha \coloneqq \sin(\alpha),$$

defines the tangent vector $\boldsymbol{t}$, the normal vector $\boldsymbol{u}$ and the binormal vector $\boldsymbol{v}$ of the track. Using the close relationship to the *Euler angles*, we can show that we can not describe general spatial curves using this method, as would be required in applications for which the above assumptions do not hold. The reason is the singularity of the Euler angles for $\alpha = \pm\pi/2$ also known as *gimbal lock* [42]. In places where $\alpha = \pm\pi/2$ the curve $\bar{\boldsymbol{p}}$ degenerates to a point which leads to the height profile no longer being a function with respect to $\bar{s}$. More severely, both γ and θ are not uniquely defined. In conventional applications, such as railroad design, this case can be ignored since the pitch angle α is bounded due to physical constraints of the vehicles. For roller coasters however, for which the above method has since been adapted, such extreme angles cannot be excluded or may even be desired in some track sections. One well known example is the vertical looping element, shown in Figure 1.2, featured in various roller coasters. As one can see, even coming close to the singularity leads to significant projective distortions and ill-behaved curve properties $\bar{\kappa}$ and γ.

For these elements, state of the art solutions invoke different projection planes to construct $\bar{\boldsymbol{p}}$ and $z(\bar{s})$, while taking special care in connecting these elements with the remaining track. Figures 1.3 and 1.4 demonstrate the impact of choosing a suitable projection plane for designing a vertical looping. Just like Figure 1.1, they show the layout, the height profile, the curvature plot and the superelevation plot of a vertical loop for two different projections. We can clearly see that a fitting choice of projection plane such as the one in Figure 1.4 is crucial when designing such elements using the classical design approach.

The remaining properties mentioned above may also not hold for layout design for roller coaster tracks. Many roller coasters have no onboard propulsion system, so the velocity of the trains is mostly determined by gravitational forces and thus highly varying along the track. Additionally, due to the comparatively large inertial accelerations, large roll angles are often needed in order to reduce lateral accelerations, with combinations of curvatures in the top view and curvatures in the height profile being the norm rather than the exception.

1.2 Heartline Principle

In the context of roller coaster design, the so called *heartline principle* has been adapted as a quasi standard design principle [90; 99, Section 3.1.1]. Instead of understanding the curve $\boldsymbol{p}$ as the *center of rails* of the resulting track, $\boldsymbol{p}$ becomes the *heartline*, i.e., the *center of gravity* of the vehicles or the passengers, or any other point of interest of the vehicles. Now, the roll angle γ with respect to $\boldsymbol{p}$ only affects the position of the track but not the heartline, effectively minimizing

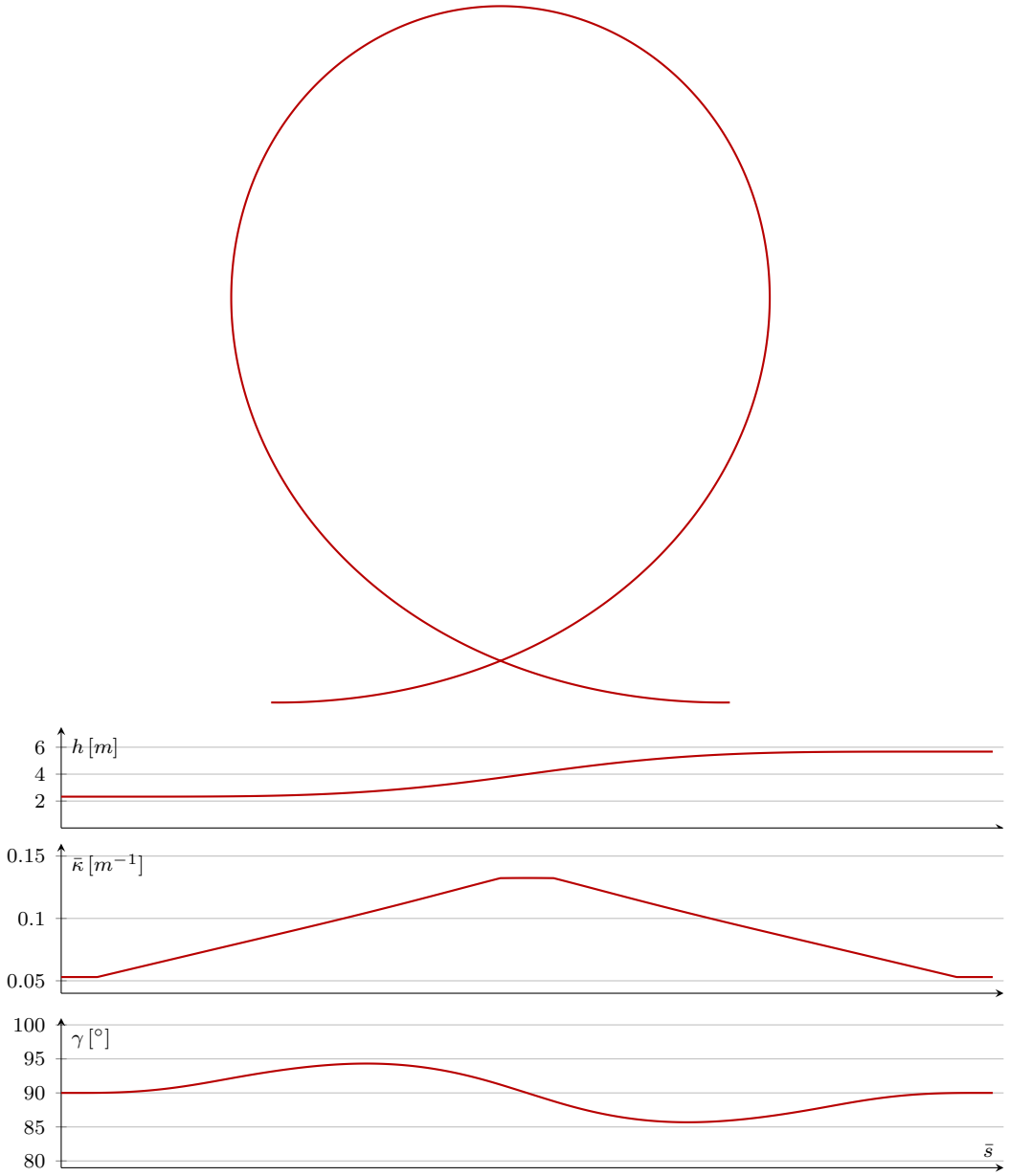


Figure 1.4: A vertical looping constructed by projection onto the x-z plane. Sections near the singularity at $\alpha = \pm\pi/2$ are avoided.

movement of the respective point of interest due to rolling motion. Due to the *parallel axis theorem* [7, Section 1.5; 75, Section 5.7.6], this effect leads to a much more natural movement of the vehicles with less stress on the rails, vehicles and passengers. A striking example of the application of the heartline principle is shown in Figure 1.5.

Since most vehicle constructions have a vertical symmetry, the heartline is in most cases located somewhere along the local up-axis of the underlying track, or in other words the track's normal $\boldsymbol{v}$. As modern railroads and maglevs have

Figure 1.5: A practical example of the heartline principle [57].

ever increasing requirements with respect to comfort and speed, the original idea of the heartline principle has been partially adapted under the German name *Schwerpunkttrassierung*, loosely translated as *layout design of the center of gravity* and patented in the field of railroads in combination with the so called *Wiener Bogen* (Vienna's Curve) [39; 40], a transition curve that allows for higher order contacts to its neighboring curve segments than the clothoid. Additionally, in some countries, e.g. Germany, railways are designed in such a way that the superelevation is applied to the outer rail of any given curve [79, Section 4.2.2]. These cases can be thought of as using the inner rail as the heartline.

1.3 Multi Body Dynamics along Tracks

Proprietary software packages such as Simpack [19] with its *Rail Module* have been specialized with respect to evaluating motion of multi body systems (MBS) traveling along spatial curves conforming to the classical representation of tracks introduced in Section 1.1. As such, they are widely used in the related industrial fields such as railway development and design [67, Section 6.3]. For more specialized applications which require more general motion along generalized spatial curves, customized implementations [71] or extensions to existing multi body dynamics libaries such as MOBILE [48; 91] have been proposed. In order to integrate spatial tracks as generalized kinematic joints in one of these general purpose libraries, the rigid body motion along such spatial curves has to be formalized. This requires a

generic mathematical representation of such tracks that is compatible with the respective multi body system library.

Especially the work of Tändl et al. [90; 91; 92; 93], which focusses on the interaction between curve properties and resulting dynamics, is an important reference work for this thesis. They regard a spatial base curve $\boldsymbol{p}$ and its Frenet frame as a kinetostatic transition element and formulate the corresponding equations of motion along such paths. They note that special care has to be taken for inflections of $\boldsymbol{p}$ since they correspond to first order singularities of the Frenet frame [91]. Additionally, the standard process of evaluating the respective motion along the path involves calculating the arc length of the underlying curve $\boldsymbol{p}$, which generally requires time consuming numerical integration. Therefore, they investigate the use of *Pythagorean hodograph* (PH) curves in their framework since their arc length can be evaluated exactly and thus in a more efficient manner [93].

Similarly, Pombo et al. [71] construct a generalized curve joint based on the Frenet frame, while using cubic splines, Akima splines [3] and shape preserving splines as base curves $\boldsymbol{p}$. As solely using the Frenet frame does not allow for sufficient degrees of freedom during the layout design, they define the superelevation of the track relative to the Frenet frame and thus allow for angular displacement of the resulting track and the base frame. As with the approach taken by Tändl et al., special care has to be taken for inflections or straight sections of the curve. They also note that spatial tracks that are traversed with significant velocity should be at least G^2 continuous.

2 Problem Statement

Although the focus of our research lies on the design of roller coaster layouts, the goal of this thesis is to approach the problem of modeling spatial paths for the purpose of traversal in a general setting with as little assumptions about the underlying application as possible. Consequently, we aim to advance the state of the art and gain new insights in the following topics.

2.1 Representation of Spatial Tracks

First, we seek a universally applicable, concise representation of spatial paths. We have shown that the state of the art representation admits singularities for certain configurations. The field of *Differential Geometry* exposes tools that help representing spatial paths in a more general way while allowing state of the art design principles to be employed in a natural manner.

2.2 Modeling Motion along Spatial Tracks

Given our newly proposed representation of spatial paths, we aim to improve existing techniques for evaluating the motion of kinematic systems along such paths. These improvements may either be achieved with respect to the quality and accuracy of the calculated motion or with respect to more efficient calculations of approximated motion.

2.3 Design and Modeling of Spatial Tracks

Last and with the most focus in this work, we seek new methods for designing and modeling spatial track layouts. Since we propose a new representation of spatial tracks, it is crucial for a designer to have access to high quality design tools that are compatible with our representation, while exposing both existing and novel modeling methods. We aim to expand the commonly used tool set, which consists of the planar geometric primitives *straight line*, *circular arc* and various transition curves such as *clothoids*, with spatial primitives such as *polynomial curves*, *energy minimizing curves* and *polynomial spirals*. As a prerequisite, we need to look for desirable properties of spatial curves to be used in layout design. Given these properties, we then seek families of curves that carry these properties and can be used in an interactive CAGD systems for use in real-world applications.

3 Outline of this Thesis

The remainder of this thesis is structured as follows. In Chapters 4 to 6, we introduce the relevant mathematical topics that will be used in this thesis. Here, we also present the commonly used notations and symbols in order to familiarize the reader. Chapter 7 covers common constraints and desirable properties of spatial tracks. On that basis we propose *framed curves* as a general representation of spatial paths in Chapter 8. We discuss several properties of framed curves and show how common operations and manipulations can be performed using this representation to demonstrate its suitability in the context of our application. Chapter 9 covers how we calculate the motion of kinematic systems that are linked to spatial tracks. In this chapter, we also compare and evaluate our methods to previously existing approaches as well as real-world measurements. In Chapters 10 to 13, we introduce several methods for the interactive design of high quality spatial tracks using framed curves. These methods not only cover the design of a track's position in space, but also the modeling of the local orientation of the track. Throughout this part of the thesis, we also seek to measure fairness of the resulting spatial track in an objective fashion. Several known measures of curve fairness are introduced and evaluated in our context. Chapters 14 and 15 conclude this thesis with a summary of our findings and an outlook into possible future work.

Fundamentals

This part introduces various theoretical concepts that are required to understand this thesis. A seasoned reader in any of the fields may skip the respective sections or parts of them, but skimming through them is recommended in order to get used to the notation used throughout this thesis. The subjects are mostly ordered logically from the most general and elemental to more specific ones.

4 Euclidean Space

In this thesis, we will describe and investigate objects and their motion in physical space, which we will model by the *Euclidean space* $\mathbb{R}^3$ [12, Chapter 16]. As it is an *affine space*, we can distinguish between points, the elements of the Euclidean point space, and vectors, which are the elements of its associated *vector space*. For brevity, we will represent both by the elements $\boldsymbol{x} \in \mathbb{R}^3$, with vectors

$$\boldsymbol{v} = \boldsymbol{q} - \boldsymbol{p}$$

being the difference of two points $\boldsymbol{p}$ and $\boldsymbol{q}$. Furthermore, we use the *Euclidean norm* $\|\boldsymbol{v}\| = \sqrt{\langle \boldsymbol{v}, \boldsymbol{v} \rangle}$ to determine the length of a vector $\boldsymbol{v}$, and thus the distance between two points $\boldsymbol{p}$ and $\boldsymbol{q}$. The Euclidean space itself has no concept of some point being the origin. In case we need this concept, as is the case in many practical applications, we can always define some point $\mathbf{0}$ as the origin of our world or global coordinate system.

4.1 Extended and Homogeneous Coordinates

The use of *extended coordinates* $\mathbb{R}^n \times \{0, 1\}$ helps us distinguish between points and vectors, with

$$\mathbf{v} = \begin{bmatrix} \boldsymbol{v} \\ 0 \end{bmatrix}$$

representing a vector and

$$\mathbf{p} = \begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix}$$

representing a point [72, Section 1.1]. When using extended coordinates, it is obvious that some operations on points and vectors are undefined in Euclidean

space, such as multiplication of a scalar a and a point $\boldsymbol{p}$ or addition of two points $\boldsymbol{p}$ and $\boldsymbol{q}$. Extended coordinates are also helpful to represent *affine maps* in a terse manner, as explained further below.

The non-zero multiples

$$\mathbf{p} = \begin{bmatrix} w\boldsymbol{p} \\ w \end{bmatrix}, w \in \mathbb{R} \setminus \{0\},$$

are the inifinitely many *homogeneous coordinate vectors* of a point $\boldsymbol{p}$ [27, Section 7.4]. Similarly, the non-zero multiples

$$\mathbf{v} = \begin{bmatrix} w\boldsymbol{v} \\ 0 \end{bmatrix}, w \in \mathbb{R} \setminus \{0\},$$

can be thought of as *points at infinity* in the direction $\pm\boldsymbol{v} \neq \mathbf{0}$, forming the *equivalence class* of all vectors with the same direction. Thus, points and directions in $\mathbb{R}^n$ are represented by lines in $\mathbb{R}^{n+1}$ through, but without, the origin, forming the projective space $\mathbb{RP}^n$. Homogeneous coordinates are therefore a powerful tool to represent *projective maps* [12, Part V].

Note: The normalized homogeneous representation with $w = 1$ leads us back to the extended coordinates of points and vectors in Euclidean space. Throughout this thesis, we denote both extended and (more generally) homogeneous coordinate vectors by straight boldface letters $\mathbf{p}$ and $\mathbf{v}$, as opposed to regular vectors, which we will denote by italic boldface letters $\boldsymbol{p}$ and $\boldsymbol{v}$. Additionally, we will use the same distinction for matrix representations, see below.

Definition 4.1 (affine transformations)**:** Maps of the form $f : \mathbb{R}^n \to \mathbb{R}^n$ in the context of points in Euclidean space are called *transformations*. Affine transformations [12, Chapter 11] are represented by a matrix $\boldsymbol{Q} \in \mathbb{R}^{n \times n}$ and a vector $\boldsymbol{r} \in \mathbb{R}^n$ such that

$$f(\boldsymbol{p}) = \boldsymbol{Q}\boldsymbol{p} + \boldsymbol{r}. \tag{4.1}$$

These include non-uniform scaling, shearing and rotation (with respect to the origin) followed by a translation by the vector $\boldsymbol{r}$. Figure 4.1 shows an example of an affine transformation in $\mathbb{R}^2$ defined by the affine map

$$f(\boldsymbol{p}) = \begin{bmatrix} 2 & 1 \\ 1 & -1 \end{bmatrix} \boldsymbol{p} + \begin{bmatrix} 3 \\ 4 \end{bmatrix}.$$

Vectors $\boldsymbol{v}$ between two points $\boldsymbol{p}$ and $\boldsymbol{q}$ are mapped to

$$f(\boldsymbol{q}) - f(\boldsymbol{p}) = \boldsymbol{Q}\,(\boldsymbol{q} - \boldsymbol{p})$$

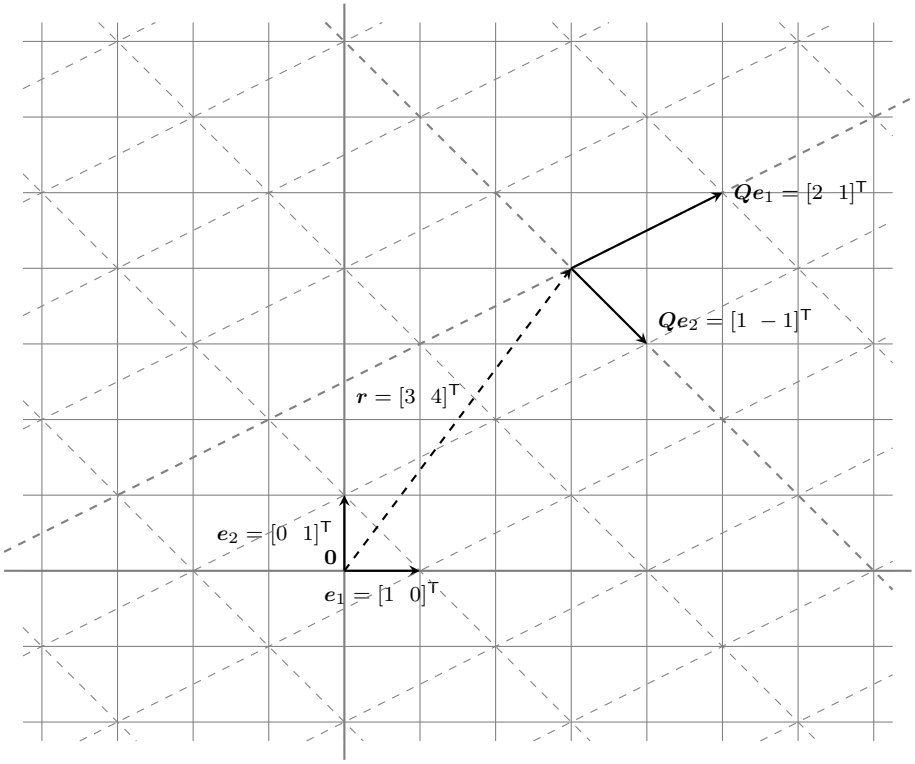


Figure 4.1: Exemplary affine transformation in $\mathbb{R}^2$ visualized.

and thus the so-called *underlying linear map* of f for vectors is

$$g(\boldsymbol{v}) = \boldsymbol{Q}\boldsymbol{v}. \tag{4.2}$$

Using extended (or homogeneous) coordinates, an affine transformation is represented by an *augmented matrix* $\mathbf{T}$ [72, Section 1.3]. Thus, the map is of the form

$$\begin{aligned} f(\mathbf{p}) &= \mathbf{T}\mathbf{p} \\ &= \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{r} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} \mathbf{p}. \end{aligned} \tag{4.3}$$

The interested reader is encouraged to verify that (4.3) is equivalent to (4.1) for points and (4.2) for vectors.

4.2 The Special Euclidean Group

Throughout this thesis, the matrix $\boldsymbol{Q}$ above will usually be a *rotation matrix* with $\det(\boldsymbol{Q}) = 1$, meaning that its column vectors are *orthonormal vectors*. In that case,

the transformation defined by $\mathbf{T}$ preserves the Euclidean distance $\|\cdot\|$ between any two points $\boldsymbol{p}$ and $\boldsymbol{q}$ in space, that is

$$\begin{aligned}\|\boldsymbol{p}-\boldsymbol{q}\| &= \|\boldsymbol{Q}\boldsymbol{p}-\boldsymbol{Q}\boldsymbol{q}\| \\ &= \|\mathbf{T}\mathbf{p}-\mathbf{T}\mathbf{q}\|\,.\end{aligned}$$

As a consequence, angles, areas and volumes are also preserved by the transformation $\mathbf{T}$ [81, Section 2.5]. We will often describe the pose of some rigid body $\mathcal{B}$ by $\mathbf{T}$. As shown in Figure 4.2, the vector $\boldsymbol{r}$ describes the position and the matrix $\boldsymbol{Q}$ the orientation of some coordinate system fixed to $\mathcal{B}$ with respect to some other reference coordinate system. Such transformations are called *rigid body transformation* or in the context of time dependent behavior *Euclidean motion* [81]. They form the *Special Euclidean Group* $SE(3)$ under multiplication since they satisfy all four *group axioms*.

1. **Closure:** For all $\mathbf{A},\mathbf{B} \in SE(3)$, $\mathbf{AB} \in SE(3)$.

2. **Associativity:** For all elements $\mathbf{A}$,$\mathbf{B}$,$\mathbf{C} \in SE(3)$, the group operation is associative, $(\mathbf{AB})\,\mathbf{C} = \mathbf{A}\,(\mathbf{BC})$.

3. **Identity:** There exists an element $\mathbf{E} \in SE(3)$, such that for all $\mathbf{A} \in SE(3)$ the equation $\mathbf{EA} = \mathbf{AE} = \mathbf{A}$ holds.

4. **Inverse:** For each $\mathbf{A} \in SE(3)$, there exists an inverse element $\mathbf{A}^{-1} \in SE(3)$, such that $\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{E}$. For

$$\mathbf{A} = \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{r} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix},$$

we get

$$\begin{aligned}\mathbf{A}^{-1} &= \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{r} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}^{-1} \\ &= \begin{bmatrix} \boldsymbol{Q}^\mathsf{T} & -\boldsymbol{Q}^\mathsf{T}\boldsymbol{r} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}\end{aligned} \tag{4.4}$$

since

$$\boldsymbol{Q}^\mathsf{T}\boldsymbol{Q} = \boldsymbol{I} \tag{4.5}$$

holds for rotation matrices.

Remark: Setting $\boldsymbol{r} = \mathbf{0}$ yields the subgroup of all rotations in 3-space called *Special Orthogonal Group* $SO(3)$, while setting $\boldsymbol{Q} = \boldsymbol{I}$ yields the subgroup of all translations called *Translational Group* $T(3)$. Furthermore, the groups $T(n)$, $SO(n)$ and $SE(n)$ are real *Lie groups*, meaning that they are continuous and that their respective group operations and their inverses are smooth [35; 41; 81]. Among other things, this allows for a differential calculus, which is crucial for this thesis.

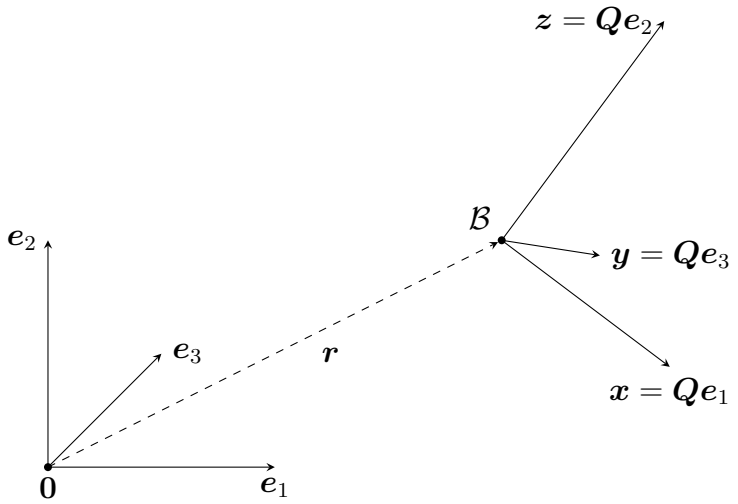


Figure 4.2: A rigid body transformation describes both the position and the orientation of a body $\mathcal{B}$ with respect to some reference coordinate system.

4.3 Rotation Matrices

In the previous section, we have already introduced rotation matrices as a way of describing the orientation of a rigid body $\mathcal{B}$ in space. As shown in Figure 4.2, the matrix

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{bmatrix} \in \mathbb{R}^{3\times 3}$$

describes the unit vectors $\boldsymbol{x}, \boldsymbol{y}$ and $\boldsymbol{z}$ of the body-fixed coordinate system of $\mathcal{B}$ relative to some reference coordinate system. The number of *rotational degrees of freedom* of $\mathcal{B}$ is directly linked to the number of degrees of freedom of $\boldsymbol{Q}$. Six of the nine parameters of $\boldsymbol{Q}$ are constrained to ensure the orthonormality of $\boldsymbol{Q}$, namely

$$\begin{aligned} \|\boldsymbol{x}\| &= 1 \\ \|\boldsymbol{y}\| &= 1 \qquad (4.6) \\ \langle \boldsymbol{x}, \boldsymbol{y} \rangle &= 0 \\ \boldsymbol{z} &= \boldsymbol{x} \times \boldsymbol{y}. \qquad (4.7) \end{aligned}$$

This leaves three degrees of freedom in 3-space, and similarly a single degree of freedom in 2-space, with the constraints given by equations (4.6). In general, the number of degrees of freedom of $SO(n)$ is given by

$$\binom{n}{2} = \frac{n\,(n-1)}{2}.$$

4.4 Euler Angles

We can understand the three rotational degrees of freedom in 3-space as the decomposition of any rotation matrix $\boldsymbol{Q}$ into three distinct rotations about different axes. The respective rotation angles are called *Euler angles* and, depending on the sequence of rotations performed, they are also sometimes referred to as *Cardan angles*. They are a second, very concise way to describe orientation. Throughout this thesis, we will often use the abbreviations $C_\phi \coloneqq \cos(\phi)$ and $S_\phi \coloneqq \sin(\phi)$ and use the notation $\mathrm{Rot}(\boldsymbol{v}, \phi)$ to describe the rotation about a vector $\boldsymbol{v}$ by the angle ϕ. In particular, we use rotations about the three basis vectors

$$\mathrm{Rot}(\boldsymbol{e}_1, \gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}$$

$$\mathrm{Rot}(\boldsymbol{e}_2, \alpha) = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}$$

$$\mathrm{Rot}(\boldsymbol{e}_3, \theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

to decompose a rotation matrix $\boldsymbol{Q}$ resulting in

$$\begin{aligned} \boldsymbol{Q}(\theta, \alpha, \gamma) &= \mathrm{Rot}(\boldsymbol{e}_1, \gamma) \cdot \mathrm{Rot}(\boldsymbol{e}_2, \alpha) \cdot \mathrm{Rot}(\boldsymbol{e}_3, \theta) \\ &= \begin{bmatrix} C_\theta C_\alpha & -S_\theta C_\gamma + C_\theta S_\alpha S_\gamma & S_\theta S_\gamma + C_\theta S_\alpha C_\gamma \\ S_\theta C_\alpha & C_\theta C_\gamma + S_\theta S_\alpha S_\gamma & -C_\theta S_\gamma + S_\theta S_\alpha C_\gamma \\ -S_\alpha & S_\gamma C_\alpha & C_\gamma C_\alpha \end{bmatrix}. \end{aligned} \tag{4.8}$$

We will refer to the angles θ, α and γ as *yaw*, *pitch* and *roll* angle, respectively. These terms, although often denoted by different symbols, are found in a wide range of applications, i.e. aerospace engineering [87, Section 1.3] and robotics [46, Chapter 2].

Note: For $\alpha = \pm\pi/2$, the matrix $\boldsymbol{Q}$ degenerates to a single rotation by $\theta \mp \gamma$ about $\boldsymbol{e}_3$ while aligning the x-axis of the image with $\mp\boldsymbol{e}_3$ since

$$\boldsymbol{Q}(\theta, \pm\pi/2, \gamma) = \begin{bmatrix} 0 & -\sin(\theta \mp \gamma) & \cos(\theta \mp \gamma) \\ 0 & \cos(\theta \mp \gamma) & \sin(\theta \mp \gamma) \\ \mp 1 & 0 & 0 \end{bmatrix}.$$

Furthermore, due to the periodicity of the trigonometric functions, we can show that

$$\begin{aligned} \boldsymbol{Q}(\theta, \alpha, \gamma) &= \boldsymbol{Q}(\theta + 2\pi, \alpha, \gamma) \\ &= \boldsymbol{Q}(\theta, \alpha, \gamma + 2\pi). \end{aligned}$$

Consequently, the map given by $\boldsymbol{Q}(\theta, \alpha, \gamma)$ is only invertible for

$$\begin{aligned} \theta &\in]-\pi, \pi] \\ \alpha &\in]-\pi/2, \pi/2[\\ \gamma &\in]-\pi, \pi] \end{aligned}$$

and using equation (4.8) it is possible to determine the inverse maps

$$\theta(\boldsymbol{Q}) = \quad \arctan\left(\frac{\boldsymbol{Q}_{21}}{\boldsymbol{Q}_{11}}\right), \tag{4.9}$$

$$\alpha(\boldsymbol{Q}) = -\arcsin\left(\boldsymbol{Q}_{31}\right), \tag{4.10}$$

$$\gamma(\boldsymbol{Q}) = \quad \arctan\left(\frac{\boldsymbol{Q}_{32}}{\boldsymbol{Q}_{33}}\right). \tag{4.11}$$

The singularity at $\alpha = \pm\pi/2$ of the Euler angles is commonly known as *gimbal lock*. It is particularly problematic in applications in which angular velocities $\boldsymbol{\omega}$ have to be mapped to the velocities $\dot{\theta}$, $\dot{\alpha}$ and $\dot{\gamma}$. Near the singularity, these become arbitrarily large even for low angular velocities [42; 46; 81].

4.5 Quaternions

Elements of the group of *quaternions* $\mathbb{H}$ are a third method of describing rotations in 3-space, much like complex numbers $z \in \mathbb{C}$ can be used to describe rotations in 2-space. A quaternion $a + bi + cj + dk \in \mathbb{H}$ can be seen as consisting of a scalar part a and an imaginary vector part

$$\boldsymbol{v} = \begin{bmatrix} b & c & d \end{bmatrix}^{\mathsf{T}}$$

and is denoted as $\boldsymbol{q} = (a, \boldsymbol{v})$. Similarly to the equation $i^2 = -1$ for the complex numbers,

$$i^2 = j^2 = k^2 = ijk = -1 \tag{4.12}$$

expresses the relationship between the real and imaginary parts of a quaternion. The product of two quaternions $\boldsymbol{q_0} = (a_0, \boldsymbol{v}_0)$ and $\boldsymbol{q_1} = (a_1, \boldsymbol{v}_1)$ can be derived from Equation (4.12) as

$$\boldsymbol{q_0} \cdot \boldsymbol{q_1} = \left(a_0 a_1 - \langle \boldsymbol{v}_0, \boldsymbol{v}_1 \rangle, a_0 \boldsymbol{v}_1 + a_1 \boldsymbol{v}_0 + \boldsymbol{v}_0 \times \boldsymbol{v}_1\right). \tag{4.13}$$

Without going into too much detail, any unit quaternion can be expressed in the form

$$\boldsymbol{q} = \left(\cos\left(\frac{\phi}{2}\right), \boldsymbol{a}\sin\left(\frac{\phi}{2}\right)\right)$$

to represent a rotation by the angle ϕ about the unit vector $\boldsymbol{a} \in \mathbb{R}^3$. This rotation applied to some vector $\boldsymbol{x} \in \mathbb{R}^3$ can then be computed as

$$(0, \boldsymbol{y}) = \boldsymbol{q}\,(0, \boldsymbol{x})\,\boldsymbol{q}^*$$

with

$$\boldsymbol{q}^* = \left(\cos\left(\frac{\phi}{2}\right), -\boldsymbol{a}\sin\left(\frac{\phi}{2}\right)\right)$$

being the *conjugate* quaternion of $\boldsymbol{q}$. The corresponding rotation matrix of a unit quaternion $\boldsymbol{q} = a + bi + cj + dk$ is

$$\mathrm{Rot}(\boldsymbol{a}, \phi) = \begin{bmatrix} 1 - 2\,(c^2 + d^2) & 2\,(bc - ad) & 2\,(bd + ac) \\ 2\,(bc + ad) & 1 - 2\,(a^2 + d^2) & 2\,(cd - ab) \\ 2\,(bd - ac) & 2\,(cd + ab) & 1 - 2\,(b^2 + c^2) \end{bmatrix}. \tag{4.14}$$

Since only unit quaternions represent rotations, one of the four elements of a quaternion is constrained by the other three to ensure $\|\boldsymbol{q}\| = 1$. However, in contrast to the Euler angles, unit quaternions do not feature any singular configurations, i.e., there is no value of $\boldsymbol{q}$ for which the velocity $\dot{\boldsymbol{q}}$ may grow arbitrarily large for a given angular velocity $\boldsymbol{\omega}$. This property greatly increases their practical usefulness and versatility to express arbitrary rotations and rotational motion in a smooth manner. The interested reader is referred to Farouki [27, Chapter 5] who gives a more in-depth introduction to quaternions and their applications.

5 Differential Geometry of Parametric Curves

In the following section, we introduce *curves* in Euclidean space and their geometric properties. The most relevant representation of curves for this thesis are *parametric curves*.

Definition 5.1 (parametric curve)**:** A point

$$\boldsymbol{p}(u) = \begin{bmatrix} p_1(u) \\ p_2(u) \\ \vdots \\ p_n(u) \end{bmatrix} \in \mathbb{R}^n$$

dependent on a scalar parameter over some interval $u \in I$ determines a *parametric curve* [72, Section 1.4].

We will refer to parametric curves in $\mathbb{R}^2$ as *plane curves* and in $\mathbb{R}^3$ as *spatial curves*. Furthermore, spatial curves are called *planar* if they lie in some plane $\mathcal{P}$, and thus may be thought of as a plane curve transformed into 3-space by some rigid body transformation $\mathbf{T}$.

Definition 5.2 (parametric speed)**:** The scalar value $\|\boldsymbol{p}'(u)\|$ of a parametric curve $\boldsymbol{p}(u)$ is called *parametric speed* [27, Section 8.1.1].

Note: Two parametric curves may represent the same geometric object. For example, the parametric curves $\boldsymbol{p}$ and $\boldsymbol{q}$ given by

$$\boldsymbol{p}(u) = \begin{bmatrix} 2\sin(u) \\ \cos(u) \end{bmatrix}, \qquad u \in [0, 2\pi[\tag{5.1}$$

$$\boldsymbol{q}(v) = \begin{bmatrix} -2\sin(v^2) \\ \cos(v^2) \end{bmatrix}, \qquad v \in \left[0, \sqrt{2\pi}\right[\tag{5.2}$$

trace out the same ellipse shown in Figure 5.1, but in different directions and with different parametric speeds. This can be shown by finding the *reparametrization* $u(v)$ for which $\boldsymbol{p}(u(v)) = \boldsymbol{q}(v)$ holds. In the example above, this reparametrization is given by $u(v) = -v^2$. Parametric curves that satisfy $\|p'\| \neq 0$ over their domain are called *regular*, with $\boldsymbol{p}$ in the example above being a regular curve.

5.1 Arc Length Parametrization

We can calculate the arc length of a curve segment

$$s(u_a, u_b) = \int_{u_a}^{u_b} \|\boldsymbol{p}'(u)\| \, du \tag{5.3}$$

by integrating its parametric speed [24, Section 1-3].

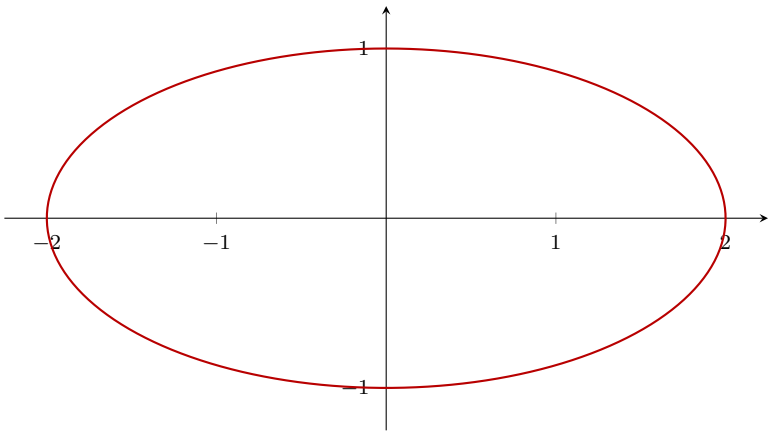


Figure 5.1: Ellipse traced out by the parametric curves given by Equation (5.1) and (5.2).

Definition 5.3 (arc length parametrization)**:** A parametric curve $\boldsymbol{p}(s), s \in [0, l]$ is called *parametrized by its arc length* if $\|\boldsymbol{p}'\| = 1 \; \forall \; s \in [0, l]$.

Unfortunately, there are only few exceptional types of freeform curves that have closed form arc length parametrizations, such as *Pythagorean hodograph* (PH) curves [27, Chapter 17]. For general curves, we have to assume that the integral (5.3) has no closed form solution and can thus can only be approximated numerically.

5.2 Geometric Continuity

The order of parametric continuity of a parametric curve does not imply smoothness of the underlying geometric object. Consider the curve

$$\boldsymbol{p}(u) = \begin{bmatrix} 2\cos(u) + \cos(2u) \\ 2\sin(u) - \sin(2u) \end{bmatrix} \tag{5.4}$$

also known as the *tricuspoid* shown in Figure 5.2. It has derivatives of all orders, and thus is in the differentiability class C^∞. However, its first derivative

$$\boldsymbol{p}'(u) = \begin{bmatrix} -2\sin(u) - 2\sin(2u) \\ 2\cos(u) - 2\cos(2u) \end{bmatrix}$$

vanishes and reverses its direction for $u = 2/3n\pi, n \in \mathbb{Z}$, resulting in *cusps* at the corresponding points. Therefore, we need a definition for smoothness in a geometric sense.

Definition 5.4 (geometric continuity)**:** A curve is G^n continuous if it can be reparametrized as a regular parametric curve with C^n continuity [22; 26, Chapter 12; 45].

With this definition, the tricuspoid is in class G^0, since all regular parametrizations can only be C^0 continuous at the cusps.

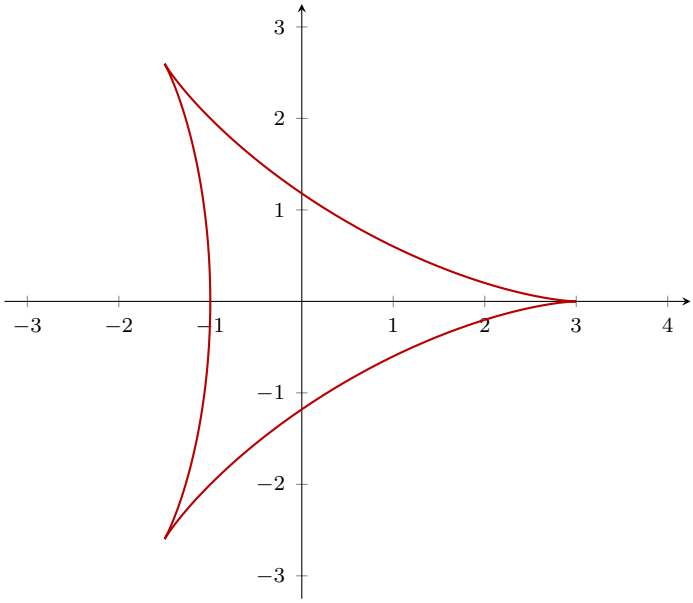


Figure 5.2: The tricuspoid has three cusps although its parameterization given by Equation (5.4) is in class C^∞.

5.3 Geometric Derivatives

Definition 5.5 (tangent)**:** The unit vector

$$\boldsymbol{t} = \frac{\boldsymbol{p}'(u)}{\|\boldsymbol{p}'(u)\|} \tag{5.5}$$

of a curve $\boldsymbol{p}$ is called its *tangent vector*. It can be seen as the geometric equivalent of the first derivative of a curve [27, Section 8.1.1; 89, Section 1-2].

With the exception of its direction, it is independent of $\boldsymbol{p}$'s parametrization. Taking its arc-length derivative yields the curvature vector

$$\boldsymbol{\kappa} = \frac{d\boldsymbol{t}}{ds} \tag{5.6}$$

whose length is the inverse radius of the *osculating circle* at the respective point on the curve [89, Section 1-4]. We can calculate its magnitude $\|\boldsymbol{\kappa}\|$, the curvature κ, in terms of the derivatives of $\boldsymbol{p}(u)$ by substituting Equation (5.6) into

$$\begin{aligned}\boldsymbol{p}'' &= \frac{d\boldsymbol{t}}{du}\|\boldsymbol{p}'\| + \boldsymbol{t}\frac{d^2s}{du^2} \\ &= \boldsymbol{\kappa}\|\boldsymbol{p}'\|^2 + \boldsymbol{t}\frac{d^2s}{du^2}.\end{aligned}$$

From there, we get

$$\begin{aligned}\boldsymbol{p}' \times \boldsymbol{p}'' &= \boldsymbol{p}' \times \left(\boldsymbol{\kappa} \left\| \boldsymbol{p}' \right\|^2 + \boldsymbol{t} \frac{d^2 s}{du^2} \right) \\ &= \left\| \boldsymbol{p}' \right\|^3 \boldsymbol{t} \times \boldsymbol{\kappa}\end{aligned}$$

and finally

$$\kappa = \frac{\left\| \boldsymbol{p}' \times \boldsymbol{p}'' \right\|}{\left\| \boldsymbol{p}' \right\|^3}.$$

We define the curve's *principal normal* $\boldsymbol{n}$ to be the unit vector pointing in the direction of the curvature vector, that is, towards the center of the curve's *osculating circle*. As a consequence, we get

$$\begin{aligned}\boldsymbol{\kappa} &= \boldsymbol{n}\kappa \\ &= \boldsymbol{n} \frac{\left\| \boldsymbol{p}' \times \boldsymbol{p}'' \right\|}{\left\| \boldsymbol{p}' \right\|^3}\end{aligned} \tag{5.7}$$

and

$$\boldsymbol{n} = \frac{\boldsymbol{p}' \times \boldsymbol{p}'' \times \boldsymbol{p}'}{\left\| \boldsymbol{p}' \times \boldsymbol{p}'' \times \boldsymbol{p}' \right\|}.$$

Note: For plane curves, $\boldsymbol{n}$ is the unit vector perpendicular to $\boldsymbol{t}$ such that κ is a signed quantity. For spatial or higher dimensional curves, the curvature κ is non-negative and the normal $\boldsymbol{n}$ is only defined if the curvature κ is non-zero.

The derivative of $\boldsymbol{\kappa}$ with respect to the curve's arc length consists of two components since the product rule applied to Equation (5.7) yields

$$\frac{d\boldsymbol{\kappa}}{ds} = \frac{d\boldsymbol{n}}{ds}\kappa + \boldsymbol{n}\frac{d\kappa}{ds}.$$

While the scalar $d\kappa/ds$ is a measure for how much the osculating circle changes in size, the vector $d\boldsymbol{n}/ds$ indicates how the orientation of the osculating circle changes. Since $\boldsymbol{n}$ is a unit vector, $d\boldsymbol{n}/ds$ has to be perpendicular to $\boldsymbol{n}$. We can show that it has to be of the form

$$\frac{d\boldsymbol{n}}{ds} = -\kappa\boldsymbol{t} + \tau\boldsymbol{b} \tag{5.8}$$

with

$$\boldsymbol{b} = \boldsymbol{t} \times \boldsymbol{n}$$

being called the *binormal vector* and

$$\tau = \left\langle \frac{d\boldsymbol{n}}{ds}, \boldsymbol{b} \right\rangle$$

being called the *torsion* of the curve [89, Section 1-5]. We can compute the torsion τ with respect to derivatives of $\boldsymbol{p}$ as

$$\tau = \frac{\langle \boldsymbol{p}' \times \boldsymbol{p}'', \boldsymbol{p}''' \rangle}{\|\boldsymbol{p}' \times \boldsymbol{p}''\|^2}$$

as long as it is regular, at least three times differentiable and its first two derivatives $\boldsymbol{p}'$and $\boldsymbol{p}''$ are linearly independent [32]. Analogously, higher order geometric derivatives can be obtained if $\boldsymbol{p}$ spans a higher dimensional space. For spatial curves, the vectors $\boldsymbol{t}$, $\boldsymbol{n}$ and $\boldsymbol{b}$ form an orthonormal basis, so any change of $\boldsymbol{b}$ has to be a linear combination of $\boldsymbol{t}$ and $\boldsymbol{n}$. In fact, to preserve orthonormality of $\boldsymbol{t}$, $\boldsymbol{n}$ and $\boldsymbol{b}$

$$\frac{d\boldsymbol{b}}{ds} = -\tau \boldsymbol{n} \tag{5.9}$$

needs to hold. The functions $\kappa(s)$ and $\tau(s)$ are referred to as the *natural equations* of a spatial curve as they uniquely describe the purely geometric properties of any sufficiently smooth spatial curve, as does $\kappa(s)$ uniquely describe the purely geometric properties of a plane curve [89, Section 1-8]. That is, two curves are congruent if their natural equations $\kappa(s)$, $\tau(s)$ are equal [24, Section 1-5; 44]. Given some starting point $\boldsymbol{p}(0)$, starting tangent $\boldsymbol{t}(0)$ and starting normal $\boldsymbol{n}(0)$, we can construct a curve given its natural equations by solving the system of differential equations (5.10) below, see [89, Section 1-8] for more details.

Remark: Some authors such as Farouki [27, Section 8.1] or Adams [4] call κ and τ *intrinsic* properties of curves. Since they are not intrinsic in the same sense as, e.g., the *Gaussian curvature* of surfaces [24, Chapter 4], we refrain from using the term *intrinsic* in favor of the term *purely geometric*.

5.4 Frenet-Serret Frame

Assuming arc length parametrization, we can rewrite Equations (5.5), (5.6), (5.8) and (5.9) more concisely as

$$\begin{bmatrix} \boldsymbol{t} \\ \boldsymbol{n} \\ \boldsymbol{b} \\ \boldsymbol{p} \end{bmatrix}' = \begin{bmatrix} 0 & \kappa & 0 & 0 \\ -\kappa & 0 & \tau & 0 \\ 0 & -\tau & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{t} \\ \boldsymbol{n} \\ \boldsymbol{b} \\ \boldsymbol{p} \end{bmatrix}. \tag{5.10}$$

They are also known as the *Frenet-Serret formulas* [32; 82]. The vectors $\boldsymbol{t}$, $\boldsymbol{n}$ and $\boldsymbol{b}$ form the so-called *Frenet frame* of $\boldsymbol{p}$. It is unique and exists as long as $\boldsymbol{t}' \neq 0$, otherwise there is no way to determine the principal normal $\boldsymbol{n}$ and binormal $\boldsymbol{b}$. If

we reverse the direction of a curve $\boldsymbol{p}$, the respective tangent and binormal vector get reversed as a result.

Definition 5.6 (Darboux vector)**:** The angular velocity $\boldsymbol{d} = \tau\boldsymbol{t} + \kappa\boldsymbol{b}$ of a curve $\boldsymbol{p}$ is called *Darboux vector* [27, Section 8.4.4]. It satisfies

$$\begin{aligned}\boldsymbol{t}' &= \boldsymbol{d} \times \boldsymbol{t}\\ \boldsymbol{n}' &= \boldsymbol{d} \times \boldsymbol{n}\\ \boldsymbol{b}' &= \boldsymbol{d} \times \boldsymbol{b}.\end{aligned}$$

5.5 Rotation Minimizing Frames

Due to its concise nature and uniqueness, the Frenet frame has become the de-facto standard way to assign a frame to a curve. The terms *normal*, *binormal*, *curvature* and *torsion* of a curve have a direct link to the Frenet-Serret formulas and find widespread use both in academia and in industrial applications. Nevertheless, other ways to frame a curve have been proposed, such as the *rotation minimizing frames* (RMF) proposed by Bishop [9]. Any frame satisfying

$$\begin{bmatrix}\boldsymbol{t}\\ \boldsymbol{u}\\ \boldsymbol{v}\\ \boldsymbol{p}\end{bmatrix}' = \begin{bmatrix}0 & \kappa_1 & -\kappa_2 & 0\\ -\kappa_1 & 0 & 0 & 0\\ \kappa_2 & 0 & 0 & 0\\ 1 & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\boldsymbol{t}\\ \boldsymbol{u}\\ \boldsymbol{v}\\ \boldsymbol{p}\end{bmatrix} \tag{5.11}$$

is considered rotation minimizing. From Equation (5.11) we can see, that an RMF does not permit any rotations about the curve's tangent. In that sense, the frame exhibits the least possible rotation to keep its orthonormality while aligning the tangent with the curve's trajectory.

In contrast to the Frenet frame, there exist a one-parameter family of RMFs since the starting normals $\boldsymbol{u}(u_0)$ and $\boldsymbol{v}(u_0)$ can be chosen freely, as long as they form an orthonormal frame with $\boldsymbol{t}(u_0)$. Furthermore, given the initial normals, there is in general no closed form solution to determine the values $\boldsymbol{u}$, $\boldsymbol{v}$, κ_1 and κ_2 for a given curve $\boldsymbol{p}$. If we define the angle $\phi(u)$ to be the angular displacement between the Frenet frame and some RMF, we need to evaluate the integral

$$\phi(u) = \phi_0 - \int_{u_0}^{u} \tau(\bar{u})\,\|\boldsymbol{p}'(\bar{u})\|\,d\bar{u} \tag{5.12}$$

to determine an RMF, see [27, Section 30.4] which, in general, assumes no closed form solution. Several numerical methods to approximate RMFs have been proposed, such as approximating a solution of Equation (5.12) [37], approximation by G^1 circular splines [101] or by Pythagorean hodograph curves [59] and discrete approximations based on projections [51], rotations [10] or double reflections [102]. For most applications, these methods yield sufficiently close approximations to an RMF.

5.6 Moving Frames and Framed Curves

Both the Frenet frame and the RMF are examples of *moving frames* [85] – sometimes also referred to as (to a curve) *adapted frames* [27, Section 30.2] – of a curve $\boldsymbol{p}$.

Definition 5.7 (moving frame)**:** Given a regular curve $\boldsymbol{p} \in \mathbb{R}^3$, a frame

$$\boldsymbol{Q} = \left[\begin{array}{ccc} \boldsymbol{t} & \boldsymbol{u} & \boldsymbol{v} \end{array} \right] \in SO(3) \tag{5.13}$$

that satisfies

$$\boldsymbol{t} = \frac{\boldsymbol{p}'}{\|\boldsymbol{p}'\|}$$

is called *moving frame* of $\boldsymbol{p}$.

Due to the orthonormality of the frame,

$$\boldsymbol{Q}^{\mathsf{T}}\boldsymbol{Q} = \boldsymbol{I}$$

holds, which after differentiation leads to

$$\boldsymbol{Q}^{\mathsf{T}}\boldsymbol{Q}' = -\,\boldsymbol{Q}'^{\mathsf{T}}\,\boldsymbol{Q}.$$

Consequently, the matrix $\boldsymbol{K} = \boldsymbol{Q}^{\mathsf{T}}\boldsymbol{Q}' = -\,\boldsymbol{Q}'^{\mathsf{T}}\,\boldsymbol{Q}$ is skew-symmetric and thus of the form

$$\boldsymbol{K} = \begin{bmatrix} 0 & -\kappa_g & \kappa_n \\ \kappa_g & 0 & -\tau_r \\ -\kappa_n & \tau_r & 0 \end{bmatrix}. \tag{5.14}$$

This leads us to the most general way to frame a curve

$$\begin{aligned} \boldsymbol{Q}' &= -\boldsymbol{Q}\,\boldsymbol{Q}'^{\mathsf{T}}\,\boldsymbol{Q} \\ &= \boldsymbol{Q}\boldsymbol{K} \end{aligned}$$

or, after matrix transposition,

$$\begin{bmatrix} \boldsymbol{t} \\ \boldsymbol{u} \\ \boldsymbol{v} \\ \boldsymbol{p} \end{bmatrix}' = \begin{bmatrix} 0 & \kappa_g & -\kappa_n & 0 \\ -\kappa_g & 0 & \tau_r & 0 \\ \kappa_n & -\tau_r & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{t} \\ \boldsymbol{u} \\ \boldsymbol{v} \\ \boldsymbol{p} \end{bmatrix}. \tag{5.15}$$

There are three general curvatures, the *geodesic curvature* κ_g, the *normal curvature* κ_n and *relative torsion* τ_r. These terms are usually used in the context of the *Darboux frame* [18], which is a moving frame of a curve $\boldsymbol{p}(u)$ embedded in some oriented surface $\mathcal{S}$. However, there is no need to invoke an oriented surface to define this most general way to frame a spatial curve. Nevertheless, we adapt these

terms in the context of this work. Setting $\kappa_g = \kappa$, $\kappa_n = 0$ and $\tau_r = \tau$ yields the Frenet frame and setting $\tau_r = 0$ yields RMFs with $\kappa_g = \kappa_1$ and $\kappa_n = \kappa_2$.

We can extend the notion of the *Darboux vector* to general moving frames. The generalized Darboux vector

$$\boldsymbol{d} = \tau_r \boldsymbol{t} + \kappa_g \boldsymbol{v} + \kappa_n \boldsymbol{u} \tag{5.16}$$

satisfies

$$\begin{aligned} \boldsymbol{t}' &= \boldsymbol{d} \times \boldsymbol{t} \\ \boldsymbol{u}' &= \boldsymbol{d} \times \boldsymbol{u} \\ \boldsymbol{v}' &= \boldsymbol{d} \times \boldsymbol{v}. \end{aligned}$$

As an example, since $\langle \boldsymbol{d}, \boldsymbol{t} \rangle = \tau_r = 0$ holds for all RMFs, their generalized Darboux vector is given by

$$\begin{aligned} \boldsymbol{d} &= \boldsymbol{t} \times (\boldsymbol{d} \times \boldsymbol{t}) \\ &= \boldsymbol{t} \times \boldsymbol{t}' \\ &= \boldsymbol{t} \times \kappa \boldsymbol{n} \\ &= \kappa \boldsymbol{b}. \end{aligned} \tag{5.17}$$

As a direct result, all generalized Darboux vectors are of the form

$$\boldsymbol{d} = \tau_r \boldsymbol{t} + \kappa \boldsymbol{b}.$$

Definition 5.8 (framed curve)**:** Maps of the form

$$\mathbf{T}(u) = \begin{bmatrix} \boldsymbol{Q}(u) & \boldsymbol{p}(u) \\ \mathbf{0}^{\mathsf{T}} & 1 \end{bmatrix} \in SE(3),$$

with $\boldsymbol{p}(u)$ being a parametric curve and $\boldsymbol{Q}(u)$ being a moving frame of $\boldsymbol{p}$ are called *framed curves*.

Note: Our representation of framed curves as an augmented matrix $\mathbf{T} \in SE(3)$ differs from the one used by Honda et al. [85; 43], who choose to leave out the tangent $\boldsymbol{t}$ of the base curve $\boldsymbol{p}$ in their representation. The underlying geometric object is the same in this work and the work of Honda et al. and thus, all of their findings also apply in our context. Additionally, the above definitions of both moving frames and framed curves could be generalized to higher dimensions. Since this generalization has no relevance to our work, we chose to define them in 3-space only.

5.7 Polynomial and Rational Curves

Definition 5.9 (Polynomial and rational curves)**:** Parametric curves that are defined only by polynomials of degree d or less are called *polynomial curves* of degree d. Similarly, curves defined by rational functions of degree d are called *rational curves* of degree d [72, Section 1.4].

Due to their flexibility, efficiency and numerical stability during evaluation and rendering these types of curves play a crucial role in *computer aided geometric design* (CAGD) systems.

Definition 5.10 (Bézier representation)**:** A polynomial curve $\boldsymbol{p}(u) \in \mathbb{R}^n$ is in its *Bézier representation* if it is of the form

$$\boldsymbol{p}(u) = \sum_{i=0}^{d} \boldsymbol{b}_i B_i^d(u)$$

with

$$B_i^d(u) = \binom{d}{i} u^i \left(1-u\right)^{d-i}, u \in [0,1]$$

being the *Bernstein polynomials* of degree d and $\boldsymbol{b}_i \in \mathbb{R}^n$ being the so-called Bézier points of $\boldsymbol{p}$ [72, Chapter 2].

The degree four polynomials B_0^4 to B_4^4 as well as an exemplary degree four polynomial with its Bézier points are shown in Figure 5.3. The *Bézier representation* has many useful properties, a detailed and extensive overview is given by Prautzsch et al. [72, Chapters 2 and 3].

5.8 Polynomial Splines

Splines are curves that consist of (e.g. polynomial) curve segments that are connected with some prescribed geometric smoothness. We can construct polynomial splines by connecting several polynomial curve segments with G^k contact. As an example, two polynomial curves $\boldsymbol{p}_1(u) = \sum \boldsymbol{a}_i B_i^n(u)$ and $\boldsymbol{p}_2(u) = \sum \boldsymbol{b}_i B_i^n(u)$ of degree d in Bézier representation have G^1 contact if

$$\boldsymbol{b}_0 = \boldsymbol{a}_d$$

and

$$\boldsymbol{b}_1 - \boldsymbol{b}_0 = \lambda \left(\boldsymbol{a}_d - \boldsymbol{a}_{d-1}\right), \lambda > 0.$$

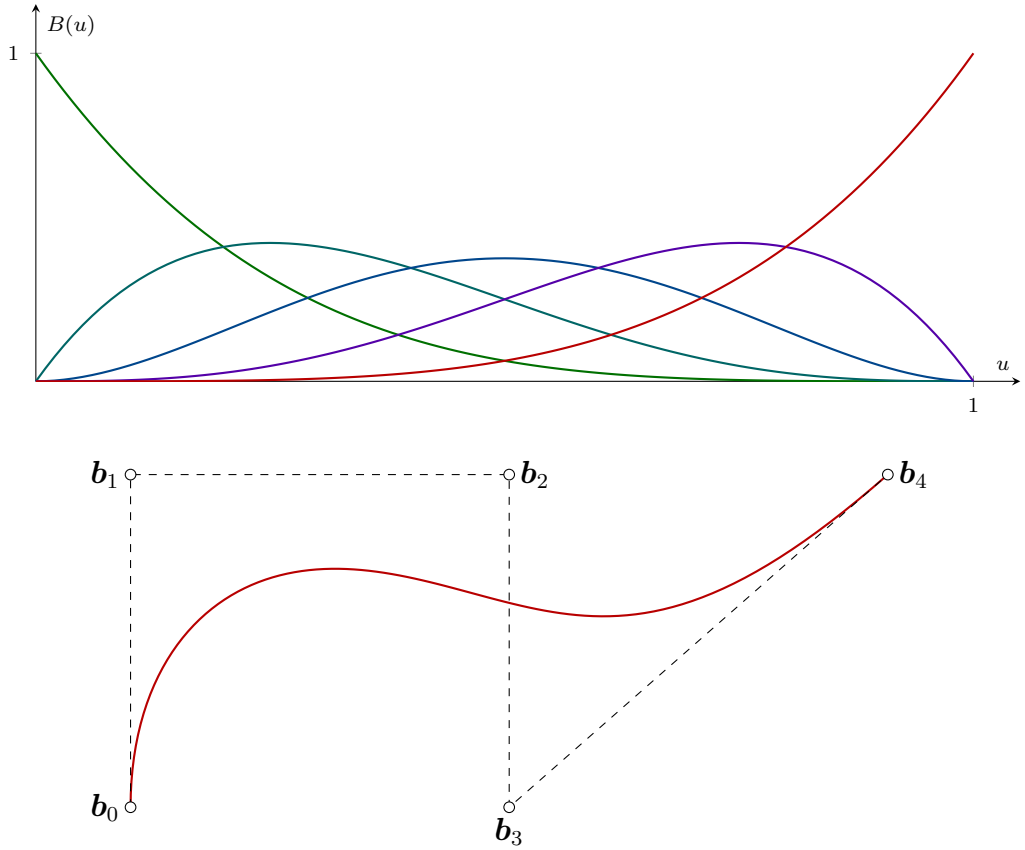


Figure 5.3: The degree four Bernstein polynomials B_0^4 to B_4^4 and a quartic curve (red) with its Bézier polygon (dashed).

The resulting spline consisting of $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ is defined as

$$\boldsymbol{p}(u) = \begin{cases} \boldsymbol{p}_1(u), & 0 \leq u < 1 \\ \boldsymbol{p}_2(u-1), & 1 \leq u \leq 2 \end{cases} .$$

Especially for polynomial splines that involve higher order contacts, constructions such as the one above quickly become tedious and impractical, especially in an interactive setting in which the Bézier points are subject to frequent modifications. Instead of using the Bernstein polynomials, we can write a C^{n-1} continuous polynomial spline

$$\boldsymbol{p}(u) = \sum_{i=0}^{n-1} \boldsymbol{c}_i N_i^d(u)$$

as an affine combination of some control points $\boldsymbol{c}_i, 0 \leq i < n$, using the degree d basis spline functions $N_i^d(u)$ [72, Section 5.2].

Definition 5.11 (B-spline)**:** Given a sequence of $m = n + d + 1$ increasing knots $a_j, 0 \leq j < m$, the basis spline functions $N_i^d(u)$ are recursively defined as

$$N_i^d(u) = \frac{u - a_i}{a_{i+d} - a_i} \cdot N_i^{d-1}(u) + \frac{a_{i+d+1} - u}{a_{i+d+1} - a_{i+1}} \cdot N_{i+1}^{d-1}(u)$$

with

$$N_i^0(u) = \begin{cases} 1, & a_i \leq u < a_{i+1} \\ 0, & \text{otherwise} \end{cases} .$$

These functions are also called *B-splines* [76].

This recursion was found by Cox [17] and de Boor and Mansfield [21] independently. An extensive overview on various algorithms for B-splines is given by Prautzsch et al. [72, Chapters 5 and 6]. In the following, we give a short introduction for the the most relevant algorithms in the context of this thesis.

Interpolation

We can interpolate a sequence of points $\boldsymbol{x}_i, 0 \leq i < n$, by a polynomial spline $\boldsymbol{p}$ of degree d by applying the following algorithm [72, Section 6.7].

1. Assign a parameter u_i, also called *node*, to each $\boldsymbol{x}_i$.

2. Construct a knot sequence $a_j, 0 \leq j < n + d + 1$.

3. Solve the system of linear equations given by

$$\begin{array}{ccccc} \boldsymbol{N} & & & & \cdot \quad \boldsymbol{C} \quad = \quad \boldsymbol{X} \end{array}$$

$$\begin{bmatrix} N_0^d(u_0) & N_1^d(u_0) & N_2^d(u_0) & & N_m^d(u_0) \\ N_0^d(u_1) & N_1^d(u_1) & N_2^d(u_1) & \cdots & N_m^d(u_1) \\ N_0^d(u_2) & N_1^d(u_2) & N_2^d(u_2) & & N_m^d(u_2) \\ & \vdots & & \ddots & \vdots \\ N_0^d(u_n) & N_1^d(u_n) & N_2^d(u_n) & \cdots & N_m^d(u_n) \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{c}_0 \\ \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \\ \vdots \\ \boldsymbol{c}_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_n \end{bmatrix} \tag{5.18}$$

This algorithm can be extended to also interpolate one or more of the first d derivatives of $\boldsymbol{p}$, for example by solving

$$\begin{array}{ccccc} \boldsymbol{N} & & & & \cdot \quad \boldsymbol{C} \quad = \quad \boldsymbol{X} \end{array}$$

$$\begin{bmatrix} N_0^d(u_0) & N_1^d(u_0) & N_2^d(u_0) & & N_{n+2}^d(u_0) \\ N'^d_0(u_0) & N'^d_1(u_0) & N'^d_2(u_0) & \cdots & N'^d_{n+2}(u_0) \\ N_0^d(u_1) & N_1^d(u_1) & N_2^d(u_1) & & N_{n+2}^d(u_1) \\ & \vdots & & \ddots & \vdots \\ N_0^d(u_n) & N_1^d(u_n) & N_2^d(u_n) & & N_{n+2}^d(u_n) \\ N'^d_0(u_n) & N'^d_1(u_n) & N'^d_2(u_n) & \cdots & N'^d_{n+2}(u_n) \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{c}_0 \\ \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \\ \vdots \\ \boldsymbol{c}_{n+2} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{x}'_0 \\ \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_n \\ \boldsymbol{x}'_n \end{bmatrix}$$

in case the endpoints $\boldsymbol{x}_0$ and $\boldsymbol{x}_n$ are to be interpolated given specific first derivatives $\boldsymbol{x}'_0$ and $\boldsymbol{x}'_n$ at u_0 and u_n, respectively. The first derivative of N_i^d is given as

$$\frac{dN_i^d(u)}{du} = \frac{d}{a_{i+d} - a_i} N_i^{d-1}(u) - \frac{d}{a_{i+d+1} - a_{i+1}} N_{i+1}^{d-1}(u).$$

Derivatives of higher order may be generated recursively.

Note: The respective interpolation problem has a unique solution if $\boldsymbol{N}$ is invertible. Thus, the respective polynomial spline $\boldsymbol{p}$ needs as many control points as there are constraints, either in form of points to be interpolated or in form of derivatives to be satisfied. Additionally, the nodes u_i and knots a_j have to be chosen such that $u_i \in]a_i, a_{i+d+1}[$, in which case $\boldsymbol{N}$ has full rank [72, Section 6.7].

Approximation

If we choose the number of control points to be less than the number of constraints, a corresponding polynomial spline may not be able to interpolate all the specified points $\boldsymbol{x}_i$ since the corresponding system of linear equations (5.18) is underdetermined [72, Section 4.6]. Nevertheless, an approximation of the $\boldsymbol{x}_i$ at the nodes u_i in the least squares sense can be achieved by solving the optimization problem

$$\underset{\boldsymbol{C}}{\text{minimize}} \quad \|\boldsymbol{N}\boldsymbol{C} - \boldsymbol{X}\|,$$

e.g., by computing

$$\boldsymbol{C} = \left(\boldsymbol{N}^\mathsf{T}\boldsymbol{N}\right)^{-1} \boldsymbol{N}^\mathsf{T}\boldsymbol{X}.$$

In our context, least-squares approximation of specified points $\vec{x}_i$ is less relevant than interpolation. However, it is a simple example of an optimization problem applied to polynomial splines, which has significant relevance in this work. In Chapters 12 and 13, we will go into much more detail on solving various interpolation and optimization problems using polynomial splines.

5.9 Polynomial Spirals

Using the natural equation $\kappa(s)$ for plane curves and $\kappa(s)$, $\tau(s)$ for spatial curves, we can define any sufficiently smooth curve $\boldsymbol{p}(s)$ by integration [4; 44]. For general functions $\kappa(s)$, $\tau(s)$ the resulting curve $\boldsymbol{p}$ has no closed-form representation. However, in practice we can approximate such a curve with sufficient accuracy by, e.g., numerical integration given some initial position $\boldsymbol{p}(0)$ and tangent vector $\boldsymbol{t}(0)$.

Definition 5.12 (polynomial spiral)**:** A plane curve is called *polynomial spiral* of degree d if its natural equation $\kappa(s)$ is a polynomial of degree $d \geq 0$ [55, Section 7.3].

Figure 5.4 shows the simplest non-trivial polynomial spiral, the *clothoid* or *Euler-Spiral* with $d = 1$, on the top and a cubic polynomial spiral on the bottom.

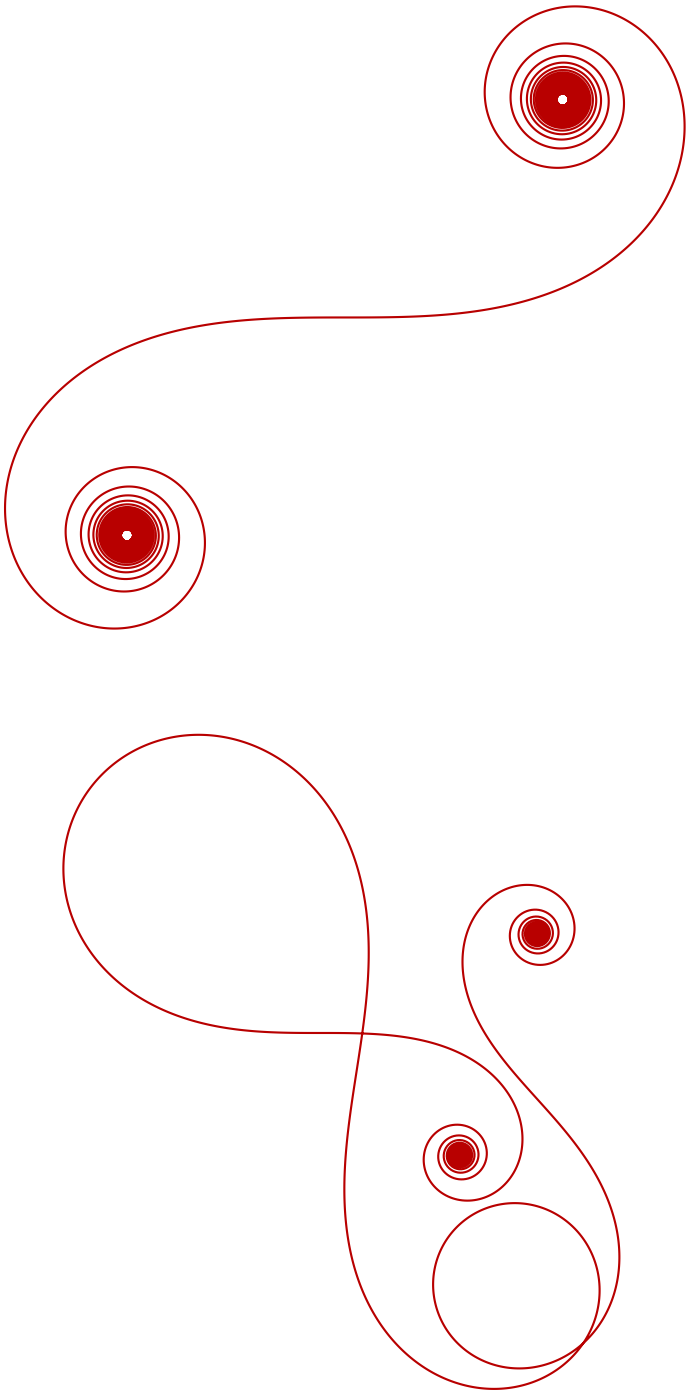


Figure 5.4: clothoid with $\kappa(s) = s$ on top and cubic polynomial spiral with $\kappa(s) = -s(s+2)(s+4.65)$ on the bottom

Due to the properties of polynomial functions, every non-trivial polynomial spiral has up to d inflections and exactly two focal points to which the spiral comes aribtrarily close as $s \to \pm\infty$. Polynomial spirals find widespread use as so-called *transition curves* in the classical layouting scheme introduced in Chapter 1. An extensive study of these transition curves is given by Schuhr [78, Section 4.2.2]. While polynomial spirals have favorable and fair curvature profiles by construction, problems like interpolation of given points in space are naturally quite difficult to solve for this class of curves [55, Chapter 8].

Definition 5.13 (spatial polynomial spiral)**:** A curve $\boldsymbol{p}(s) \in \mathbb{R}^3$ is called *polynomial spiral* of degree d if there exists a moving frame $\mathbf{Q}$ of $\boldsymbol{p}$ for which the curvatures $\kappa_g(s)$, $\kappa_n(s)$ and $\tau_r(s)$ are polynomials of degree d.

The above definition of spatial polynomial spirals is deliberately kept general to cover both Mehlum's Spiral [60], also called spherical clothoid, with $d = 1$ and its moving frame being some RMF (5.11), and 3D clothoids [31; 38] with $d = 1$ and its moving frame being the Frenet frame (5.10). Different polynomial spirals of degree d may have up to G^{d+1} contact and thus may form non-linear splines, which can be used similarly to polynomial splines for interactive curve design [55; 100].

5.10 Energy Minimizing Curves

In the context of this thesis, curves that interpolate end points with prescribed tangents while minimizing some energy E are of particular interest. The so-called *elastica* or *minimum energy curves* (MEC) are a well studied example of such curves [27, Chapter 14; 55, Chapter 3]. They are closely related to physical splines since they minimize their physical bending energy

$$E = \int \kappa^2 ds.$$

For parametric curves in $\mathbb{R}^3$, this integral can be rewritten as

$$E = \int \frac{\|\boldsymbol{p}' \times \boldsymbol{p}''\|^2}{\|\boldsymbol{p}'\|^5} du,$$

which, in general, has no closed-form representation. Using calculus of variations [53], it can be shown that an elastica interpolating two points with tangent constraints satisfies the second-order non-linear differential equation

$$\kappa'' + 0.5\kappa^3 - 0.5c\kappa = 0. \tag{5.19}$$

Several of such MEC segments may form non-linear splines which interpolate a sequence of points $\boldsymbol{x}_i$. There have been many contributions to construct plane elastica interpolants using a variety of techniques [14; 16], with few contributions that extend into 3-space, e.g., by Veltkamp et al. [95]. Lee et al. [53] give insights into Equation (5.19) and perform an extensive study about the properties of

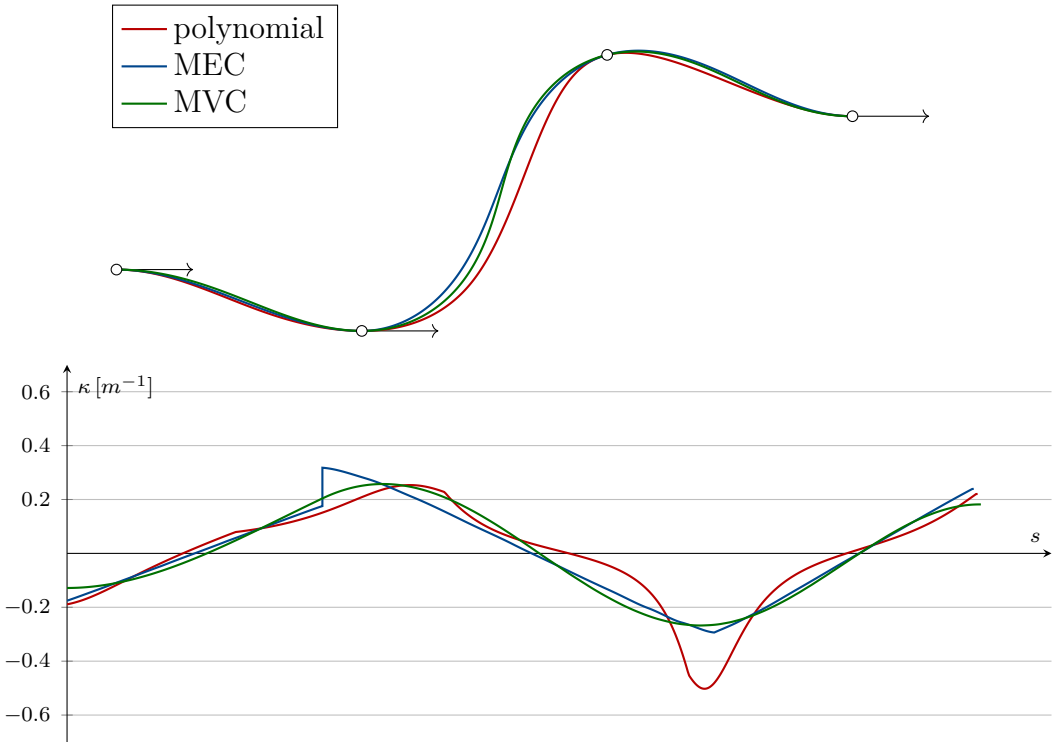


Figure 5.5: Comparison of a cubic polynomial spline, MEC and MVC interpolant.

interpolants that minimize bending energy. Among other things, this study shows that the MEC segments have G^2 contact at the interpolation points, and that this contact is reduced to G^1 continuity at points with a prescribed tangent, see Figure 5.5.

A closely related family of curves are the so-called *minimum variation curves* (MVC) [61, Chapter 4]. Compared to the elastica, their energy integral

$$E(\boldsymbol{p}) = \int \kappa'^2 \, ds$$

is of higher order as is the non-linear differential equation

$$\kappa^{(4)} + \kappa'' \kappa^2 - 0.5\kappa \, \kappa'^2 + c\kappa = 0,$$

which all MVC satisfy [13]. Similarly to MEC interpolants, MVC interpolants are comprised of segments between each pair of interpolation points. As opposed to the G^2 contact in the bending energy case, the MVC segments have G^4 contact [55, Section 7.2.2]. This continuity is reduced to G^3 in the presence of a prescribed tangent and to G^2 in the presence of prescribed curvature. Moreton [61] provides an extensive study of MVC, gives an implementation on how to approximate them using quintic polynomials and extends the idea to surfaces of minimum curvature variation.

6 Kinematics

In the field of *Kinematics* the description of motion of bodies is studied. These bodies may in the simplest case be infinitessimally small, point-like objects without a sense of orientation, but in general they are deformable or rigid bodies with all six degrees of freedom in Eulidean space. In this work, we will focus on rigid bodies that are kinematically coupled by *joints* that constrain their relative motion in some way [46, Section 1.2.2]. Such rigid bodies form so-called *kinematic systems*, which can have widely varying topologies, from serial *kinematic chains* often found in industrial robots to highly branching trees and interconnected graphs found in automotive or railway applications. In this section, we will use the continuous and smooth structure of the Special Euclidean Group to not only describe the position and orientation, but also the differential properties like velocity and acceleration of rigid bodies in space.

6.1 Rigid Body Motion

The motion of a rigid body $\mathcal{B}$ can be described by its time-dependent rigid body transformation $\mathbf{T}(t) \in SE(3)$ [46, Chapter 4]. Using a fixed reference relative to the body, the transformation $\mathbf{T}$ depicts the body's location and orientation relative to some other frame, e.g., the static global coordinate system. In this thesis, we will define this reference frame to be at some meaningful location such as the body's center of gravity, with a meaningful alignment of the axes, mainly with the local x, y and z-axes pointing foward (east), left (north) and up, respectively.

If we need to distinguish between several transformations, we will denote $\mathbf{T}_{\mathcal{A}}^{\mathcal{B}}$ as the transformation from body $\mathcal{A}$ to $\mathcal{B}$. If $\mathbf{p}_{\mathcal{B}}$ is the position of a point relative to body $\mathcal{B}$, $\mathbf{p}_{\mathcal{A}} = \mathbf{T}_{\mathcal{A}}^{\mathcal{B}} \cdot \mathbf{p}_{\mathcal{B}}$ is its position relative to $\mathcal{A}$. Furthermore, we can reverse the transformation by computing the inverse $\mathbf{T}_{\mathcal{B}}^{\mathcal{A}} = {\mathbf{T}_{\mathcal{A}}^{\mathcal{B}}}^{-1}$ as shown in Equation (4.4).

Remark: Although we commonly say *transformation from $\mathcal{A}$ to $\mathcal{B}$*, mathematically coordinates relative to $\mathcal{B}$ will be transformed to coordinates relative to $\mathcal{A}$. The transformation matrix shows the relative position and orientation of $\mathcal{B}$ in $\mathcal{A}$, thus shifting our attention from a representation in $\mathcal{A}$ to a representation relative to $\mathcal{B}$ while reading equations from left to right.

To describe the motion of a rigid body $\mathcal{B}$, we have to investigate the time-dependent properties of its transformation matrix $\mathbf{T}(t)$. The linear and rotational velocity of $\mathcal{B}$ can be computed by taking the derivative

$$\dot{\mathbf{T}} = \begin{bmatrix} \dot{\boldsymbol{Q}} & \dot{\boldsymbol{p}} \\ \mathbf{0}^{\mathsf{T}} & 0 \end{bmatrix},$$

with $\dot{\boldsymbol{p}}$ being the linear velocity and $\dot{\boldsymbol{Q}}$ being related to the body's angular velocity.

Since $\boldsymbol{Q}$ is orthonormal, we can use Equations (5.13) to (5.14) to compute the matrix

$$\boldsymbol{\Omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

such that $\dot{\boldsymbol{Q}} = \boldsymbol{Q}\boldsymbol{\Omega}$. The three parameters ω_x, ω_y and ω_z correspond to the instantaneous angular velocities about the local x, y and z-axes of $\mathcal{B}$ at any given time. The corresponding angular velocity relative to the reference frame is $\boldsymbol{\omega} = \boldsymbol{Q}[\omega_x, \omega_y, \omega_z]^\mathsf{T}$. Similarly, we can express the linear velocity as $\dot{\boldsymbol{p}} = \boldsymbol{Q}[v_x, v_y, v_z]^\mathsf{T}$ and define the general velocity matrix

$$\mathbf{V} = \begin{bmatrix} 0 & -\omega_z & \omega_y & v_x \\ \omega_z & 0 & -\omega_x & v_y \\ -\omega_y & \omega_x & 0 & v_z \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.1}$$

such that $\dot{\mathbf{T}} = \mathbf{T}\mathbf{V}$. Note the striking similarity of above equations with Equations (5.14) and (5.15) in the previous section.

Computing the general linear and rotational accelerations can be done in the same fashion by evaluating the second derivative $\ddot{\mathbf{T}} = \mathbf{T}(\mathbf{V}^2 + \dot{\mathbf{V}})$, with

$$\mathbf{V}^2 = \begin{bmatrix} -\omega_y^2 - \omega_z^2 & \omega_x\omega_y & \omega_x\omega_z & v_z\omega_y - v_y\omega_z \\ \omega_x\omega_y & -\omega_x^2 - \omega_z^2 & \omega_y\omega_z & v_x\omega_z - v_z\omega_x \\ \omega_x\omega_z & \omega_y\omega_z & -\omega_y^2 - \omega_z^2 & v_y\omega_x - v_x\omega_y \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.2}$$

contributing centrifugal and coriolis accelerations and

$$\dot{\mathbf{V}} = \begin{bmatrix} 0 & -\dot{\omega}_z & \dot{\omega}_y & \dot{v}_x \\ \dot{\omega}_z & 0 & -\dot{\omega}_x & \dot{v}_y \\ -\dot{\omega}_y & \dot{\omega}_x & 0 & \dot{v}_z \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.3}$$

contributing the linear accelerations and so-called Euler accelerations or azimuthal accelerations caused by variation of angular velocity [63, Section 10.2].

6.2 Kinematic Joints

In real-world applications, the motion of a body $\mathcal{B}$ is often constrained by its environment. Especially in mechanical applications, we regularly find some kinematic link between a body $\mathcal{B}$ and some other body $\mathcal{A}$, constraining their relative motion while allowing selected degrees of freedom.

Definition 6.1 (joint)**:** A *joint* is a continuously differentiable map of the form

$$\mathbf{T}(\boldsymbol{q}) \in SE(3)$$

Its parameter $\boldsymbol{q} \in Q$ is called *joint vector* [83]. We call the dimension of the *joint space* Q the *number of mobilities* of the joint.

The number of degrees of freedom of a joint $\mathbf{T}$ is closely linked to its number of mobilities. Using only a scalar value q the two most basic joint types are translational and rotational joints [46, Section 1.2.2]. Their rigid body transformations $\mathbf{T}(q)$ are

$$\begin{aligned}\mathbf{T}_{\text{trans}}(q) &= \begin{bmatrix} \boldsymbol{I} & q\boldsymbol{e}_3 \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} \\ &=: \operatorname{Trans}(q\boldsymbol{e}_3)\end{aligned}$$

and

$$\mathbf{T}_{\text{rot}}(q) = \operatorname{Rot}(\boldsymbol{e}_3, q)\,.$$

Remark: Without loss of generality, we use the local z-axis as the movement axis while we can define constant transformations $\mathbf{T}_{\mathcal{A}}^{\mathcal{J}}$ and $\mathbf{T}_{\mathcal{J}}^{\mathcal{B}}$ to denote the transformation from some body $\mathcal{A}$ to the joint $\mathcal{J}$ and from the joint $\mathcal{J}$ to some other body $\mathcal{B}$. Using this convention, the transformation from $\mathcal{A}$ to $\mathcal{B}$ mobilized by some joint $\mathcal{J}$ is given by

$$\mathbf{T}_{\mathcal{A}}^{\mathcal{B}}(q) = \mathbf{T}_{\mathcal{A}}^{\mathcal{J}} \cdot \mathbf{T}(q) \cdot \mathbf{T}_{\mathcal{J}}^{\mathcal{B}}.$$

We can construct composite joint types by concatenating basic joints. For example the *Cardan joint* or *universal joint*

$$\mathbf{T}_{\mathcal{A}\,cardan}^{\mathcal{B}}(p, q) = \mathbf{T}_{\mathcal{A}}^{\mathcal{J}} \begin{bmatrix} C_pC_q & -C_pS_q & -S_p & 0 \\ S_pC_q & -S_pS_q & C_p & 0 \\ -S_q & -C_q & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{T}_{\mathcal{J}}^{\mathcal{B}}$$

combines two rotational joints for a total of two degrees of freedom. Ball joints allow arbitrary rotational motion and may be represented by using Euler angles

$$\boldsymbol{q} = \begin{bmatrix} \theta & \alpha & \gamma \end{bmatrix}^\mathsf{T}$$

as its joint vector. However, due to the singularity at $\alpha = \pm\pi/2$ this may be unfeasible in practice. Therefore, ball joints are often represented by a unit quaternion $\boldsymbol{q} \in \mathbb{H}, \|\boldsymbol{q}\| = 1$, with the joint matrix being of the form (4.14). More complex joint types can be constructed in a similar fashion, see Seth et al. [83] for more examples.

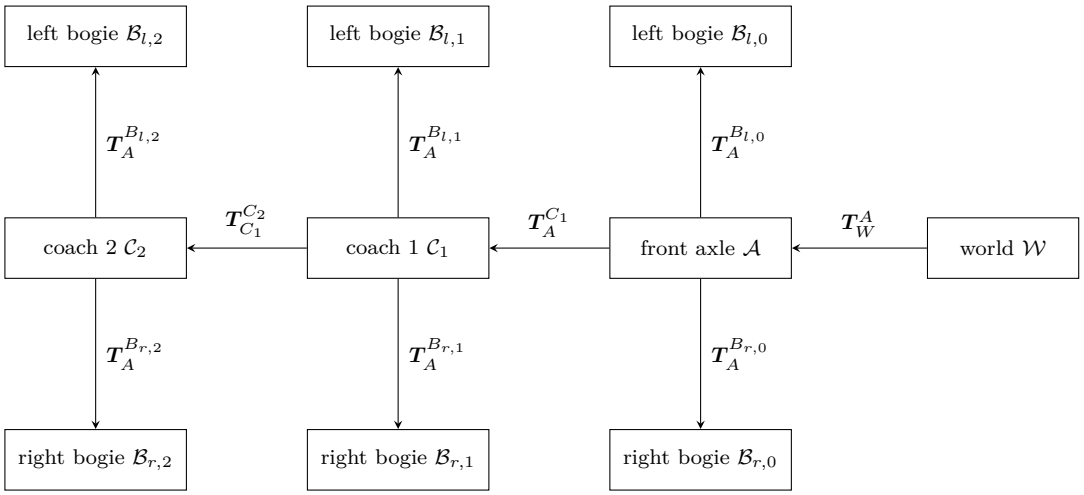


Figure 6.1: Simplified kinematic model of a two coach train.

6.3 Kinematic Systems

Kinematic systems are often multi-level hierarchical structures consisting of several bodies connected by various different types of joints. A simplified kinematic model of the a roller coaster train is depicted in Figure 6.1. Each sub-body is represented by a node in the shown directed graph while joints and the corresponding transformation matrices that define the relationship between two bodies are depicted by the edges of the graph. To determine the position of any body within this graph with respect to another body, we can traverse the graph and evaluate the transformations that form the respective *kinematic chain*. If we traverse an edge in reverse direction, its corresponding transformation needs to be inverted. The kinematic chains from world $\mathcal{W}$ to the second left bogie $\mathcal{B}_{l,1}$ and from the second coach $\mathcal{C}_2$ to the front axle $\mathcal{A}$ are given by

$$\mathbf{T}_{\mathcal{W}}^{\mathcal{K}_{l,1}} = \mathbf{T}_{\mathcal{W}}^{\mathcal{A}}\, \mathbf{T}_{\mathcal{A}}^{\mathcal{C}_1}\, \mathbf{T}_{\mathcal{C}_1}^{\mathcal{K}_{l,1}}$$

and

$$\mathbf{T}_{\mathcal{C}_2}^{\mathcal{A}} = \left(\mathbf{T}_{\mathcal{C}_1}^{\mathcal{C}_2}\right)^{-1} \left(\mathbf{T}_{\mathcal{A}}^{\mathcal{C}_1}\right)^{-1}.$$

Remark: The directions of the edges indicate parent child relationships between two bodies. These relationships may be chosen arbitrarily, as reversing the direction of an edge is achieved by inverting the corresponding joint transformation. However, the chosen parent child relationship of two bodies usually has some contextual meaning.

In general, the more joints are involved in a kinematic chain, the more mobilities are used to represent the final transformation, allowing for more complex motion. Therefore, the total number of mobilities, that is the sum of the mobilities of all joints in a kinematic system, is a good first indicator for the system's complexity.

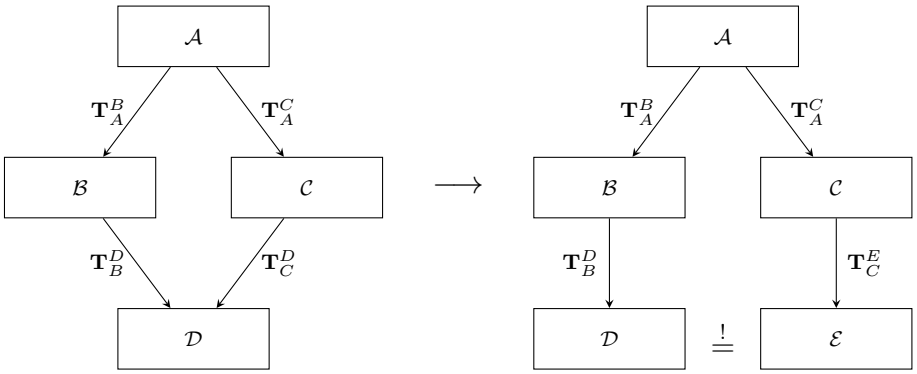


Figure 6.2: Simple example for a closed-loop kinematic system.

6.4 Kinematic Constraints

Our first example shown in Figure 6.1 shows a kinematic system with a tree-like topology, with the body $\mathcal{W}$ being the root of the tree and the bodies $\mathcal{B}_{l,i}$, $\mathcal{B}_{r,i}$, $i = 0 \dots 2$ being the leaf nodes. We can also consider kinematic systems with loops like the one shown in Figure 6.2. Due to the circular dependency between the four bodies $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ and $\mathcal{D}$, one of the joints is constrained by the other joints in the loop

$$\begin{aligned}\mathbf{T}_{\mathcal{A}}^{\mathcal{D}} &= \mathbf{T}_{\mathcal{A}}^{\mathcal{B}}\,\mathbf{T}_{\mathcal{B}}^{\mathcal{D}} \\ &= \mathbf{T}_{\mathcal{A}}^{\mathcal{C}}\,\mathbf{T}_{\mathcal{C}}^{\mathcal{D}}.\end{aligned}$$

We can convert these *parallel mechanisms* to a tree-like topology by introducing a new body $\mathcal{E}$ and constraining the bodies $\mathcal{D}$ and $\mathcal{E}$ to be coincident by setting $\mathbf{T}_{\mathcal{A}}^{\mathcal{D}} = \mathbf{T}_{\mathcal{A}}^{\mathcal{E}}$. Other types of kinematic constraints may limit the allowable joint vector $\boldsymbol{q}$ of a kinematic system to some bounded volume in joint space, limit the pose of certain bodies in Euclidean space or model collisions between bodies. Some of the mobilities of the kinematic system may be needed to satisfy such constraints, effectively reducing the number of degrees of freedom of the system. Sherman et al. [84] provide an extensive overview on how joints and constraints are modeled in current state of the art multi body system (MBS) libraries, with Mueller [64] giving insights into how kinematic topologies may be represented for kinematic analysis.

6.5 Forward Kinematics

Given a kinematic system and its trajectory $\boldsymbol{q}(t) \in Q$ in joint space, determining the time-dependent motion of the respective bodies in Euclidean space is called *forward kinematics* [46, Chapter 5]. While determining the position of each body $\mathcal{B}$ at each point in time t is a straightforward process by evaluating $\mathbf{T}_{\mathcal{W}}^{\mathcal{B}}(\boldsymbol{q}(t))$,

computing the differential quantities like velocity and acceleration requires more effort. With

$$\mathbf{T}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{x}(\boldsymbol{q}) & \boldsymbol{y}(\boldsymbol{q}) & \boldsymbol{z}(\boldsymbol{q}) & \boldsymbol{p}(\boldsymbol{q}) \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{6.4}$$

we can compute the linear velocity

$$\begin{aligned} \dot{\boldsymbol{p}}(\boldsymbol{q}) &= \frac{d\boldsymbol{p}}{d\boldsymbol{q}} \cdot \dot{\boldsymbol{q}} \\ &= \boldsymbol{J}_p(\boldsymbol{q}) \cdot \dot{\boldsymbol{q}} \end{aligned} \tag{6.5}$$

and the angular velocity

$$\begin{aligned} \boldsymbol{\omega}(\boldsymbol{q}) &= \langle \dot{\boldsymbol{y}}, \boldsymbol{z} \rangle \cdot \boldsymbol{x} + \langle \dot{\boldsymbol{z}}, \boldsymbol{x} \rangle \cdot \boldsymbol{y} + \langle \dot{\boldsymbol{x}}, \boldsymbol{y} \rangle \cdot \boldsymbol{z} \\ &= \begin{bmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{bmatrix} \begin{bmatrix} \boldsymbol{z}^\mathsf{T} \cdot \boldsymbol{J}_y(\boldsymbol{q}) \\ \boldsymbol{x}^\mathsf{T} \cdot \boldsymbol{J}_z(\boldsymbol{q}) \\ \boldsymbol{y}^\mathsf{T} \cdot \boldsymbol{J}_x(\boldsymbol{q}) \end{bmatrix} \dot{\boldsymbol{q}} \\ &= \boldsymbol{J}_r(\boldsymbol{q}) \cdot \dot{\boldsymbol{q}} \end{aligned} \tag{6.6}$$

with the help of the *Jacobian matrices* $\boldsymbol{J_p}$ and $\boldsymbol{J}_r$ of $\mathbf{T}$ with respect to the joint vector $\boldsymbol{q}$. We can determine the linear and angular accelerations in a similar fashion by calculating the second order derivative. Note again, that so far, we are not concerned about the forces and torque that induce this motion. This would require the knowledge about masses and inertia of the respective bodies and thus the field of *Rigid Body Dynamics*, which is only tangentially relevant for this thesis and therefore omitted in this chapter. The interested reader is referred to the literature about this subject, such as by Featherstone [28] or Jazar [46, Part II].

6.6 Inverse Kinematics

The inverse problem of *forward kinematics*, that is, determining a trajectory $\boldsymbol{q}(t)$ in joint space from a given trajectory $\mathbf{M}(t)$ in Euclidean space, is called *inverse kinematics* or *backwards kinematics* [46, Chapter 6]. While forward kinematics always has a unique solution given by $\mathbf{T}(\boldsymbol{q}(t))$, the map $\mathbf{T}(\boldsymbol{q})$ is in general neither injective nor surjective, that is for any given $\mathbf{M} \in SE(3)$, there may be an arbitrary number of distinct vectors $\boldsymbol{q}$ (including none and infinitely many) that satisfy $\mathbf{T}(\boldsymbol{q}) = \mathbf{M}$ subject to all constraints. For some kinematic systems, the inverse kinematics problem can be solved algebraically, e.g., by using geometric observations [81, Chapter 5], but in general, solutions to the inverse kinematics problem may only be approximated using numerical methods, such as the one outlined below [46, Section 6.3].

Given an estimation of the joint vector $\boldsymbol{q}_i$ for a given pose $\mathbf{M}$, we can solve the non-linear optimization problem

$$\underset{\Delta \boldsymbol{q}}{\text{minimize}} \quad \|\mathbf{T}(\boldsymbol{q}_i + \Delta \boldsymbol{q}) - \mathbf{M}\|$$

iteratively, e.g., by Newton's method [80, Section 4.3.2], in which case the discrete change of the joint vector in each iterations is computed as

$$\Delta \boldsymbol{q} = \left(\boldsymbol{J}^\mathsf{T} \boldsymbol{J}\right)^{-1} \boldsymbol{J}^\mathsf{T} \left(\mathbf{T}(\boldsymbol{q}_i) - \mathbf{M}\right)$$

and determines a next estimation

$$\boldsymbol{q}_{i+1} = \boldsymbol{q}_i + \Delta \boldsymbol{q}$$

of the joint vector. The algorithm terminates either if $\|\mathbf{T}(\boldsymbol{q}_i) - \mathbf{M}\| < \epsilon$, in which case a sufficiently accurate solution has been found or if $\|\Delta \boldsymbol{q}\| < \delta$ in the case that the target transformation $\mathbf{M}$ cannot be reached by the kinematic chain. Depending on the choice of the initial joint vector $\boldsymbol{q}_0$, different solutions $\boldsymbol{q}$ satisfying $\|\mathbf{T}(\boldsymbol{q}) - \mathbf{M}\| < \epsilon$ may be found.

Layout Design for Track-Bound Vehicles

This thesis covers algorithms that facilitate interactive design and modeling of spatial tracks for various applications. These applications may range from transportation like the layout of roads, railroad tracks or maglev tracks to the design of sports facilities like luge or bobsleigh tracks and recreational facilities like roller coasters and (water) slides. Since roller coasters exhibit in some sense the most varied, versatile and sophisticated layouts of the applications above, we believe that designing algorithms for this application is particularly challenging and may give new insights also for related fields. However, before we look at interactive design techniques for layouts, we need to establish a framework of how to model and evaluate spatial tracks, which is the main purpose of the following chapters.

This part of the thesis is structured as follows. We introduce common design constraints in the realm of roller coasters and desirable properties of roller coaster tracks in Chapter 7. In Chapter 8, we formalize how tracks can be modeled using framed curves as introduced in Section 5.6. In this chapter, we also introduce various concepts such as *offset curves*, *continuity* and *sweeps* in the context of framed curves. Furthermore, we show how these concepts may be applied to our setting. In Chapter 9, we introduce how to evaluate the kinematics of systems of rigid bodies traveling along tracks, and we use the techniques and methods from Chapter 8 extensively to transition from the realm of differential geometry to kinematics.

7 Design Constraints and Track Fairness

Several factors determine the feasibility of a track layout in the context of a specific application. Formally speaking, the problem of finding the optimal track layout is a constrained optimization problem of the form

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & L(\boldsymbol{x}) \\ \text{subject to} \quad & C_i(\boldsymbol{x}) \geq 0 \quad , i = 0,...,m, \end{aligned} \tag{7.1}$$

where $\boldsymbol{x} \in F$ represents the features of the track in some *feature space* F, $L(\boldsymbol{x})$ is a loss function that penalizes undesired track features and $C_i(\boldsymbol{x})$ are constraint functions that discard unfeasible tracks altogether. We can think of the C_i as hard constraints and of L as a function that represents soft constraints, rewarding desirable track features like reduced cost, wear and tear, ease of manufacturing or increased comfort during traversal.

In order to get a sense of the types of constraints we may encounter in real-world applications, we introduce several common constraints and important concepts that affect the design of roller coaster tracks. A more complete and general overview of roller coaster design is given by Väisänen [99]. As we will see, the constraints range from well-defined, easy to model, hard constraints to fuzzy, experience-based human factors with little to no underlying mathematical model.

7.1 Basic Geometric Constraints

The most basic type of constraint comes from the fact that some geometric properties of a spatial track are prohibited for different reasons. For most applications, a maximum absolute pitch or inclination angle $\alpha_{\max}$ is imposed on the track to ensure that motorized vehicles are able to overcome the resulting downhill force [67, Section 3.5.1]. Similarly, the roll angle γ or superelevation may be constrained to ensure that all vehicles are prevented from tipping over even when stationary [67, Section 3.2.2].

Note: Since virtually all classical applications constrain the pitch angle α to small absolute values, the singularity of the Euler angles is implicitly avoided and thus of no concern. This is why the classical approach of designing a layout introduced in Section 1.1 remains the de-facto standard to this day. In the context of roller coasters however, both α and γ may in general be completely unconstrained.

The kinematics of the vehicles may further constrain the differential properties of the underlying track. Excessive curvatures naturally lead to collisions either of parts of the vehicles with themselves or with the track. In general, only an inverse kinematics solver using a full kinematic model of the train can definitively predict such collisions. That being said, an effective mitigation against collisions is to limit the curvatures of the underlying track to values that are well within the

motion envelope of the train. Nottbeck [67, Chapter 3] lists and motivates various curvature constraints that apply to the design of railroad tracks.

7.2 Dynamic constraints

The curvature limits may vary along the track if we consider the limits on dynamic quantities such as forces, accelerations or angular velocities imposed on the coaches and passengers. These dynamic quantities may only be accurately determined by a sophisticated multi-body simulation, but again, simple approximations may aid in efficiently determining sections of high accelerations in existing tracks.

In the context of roller coasters, the maximum accelerations that are allowed to be experienced by passengers are defined by the standards that are in effect in the respective parts of the world. For example, both the American ASTM F2291 [5] and the European DIN EN 13814-1 [23] not only constrain the gravitoinertial acceleration vector – the superposition of gravitational and inertial accelerations –

$$\boldsymbol{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

relative to the passenger coordinate system to some bounded volume in $\mathbb{R}^3$, but also adapt this volume based on the duration Δt in case of sustained accelerations. A section through the volume defined by ASTM F2291 for $a_x = 0$ and $\Delta t = 0$ as well as the time-dependent limits $a_{z,\max}(\Delta t)$ and $a_{y,\max}(\Delta t)$ are shown in Figure 7.1.

7.3 Block Sections and Capacity

In applications involving unmotorized vehicles, the vehicles' path velocity along extended track sections is purely dependent on gravity and friction. Due to both internal and external factors like bearing quality, vehicle mass, temperature and wind speed, the vehicles' path velocity may vary significantly during operation [36, Section 5.2]. All track layouts, not only those of roller coasters, should be designed in a way that a vehicle is able to traverse the track safely under all nominal operating conditions. In the case of gravity driven roller coasters, this means that a vehicle shall safely reach the next section that is able to control the vehicle velocity by brakes or external drives. Since exceptional circumstances may always lead to vehicles being stuck somewhere along the track without immediate means of recovery, collision avoidance between two vehicles is mandatory. To achieve this, roller coaster tracks are logically divided into several disjoint *block sections*. If we can guarantee that each block section is occupied by at most one vehicle at all times, the possibility for collisions between two vehicles is eliminated. Brakes or other means of controlling the velocity of a vehicle at the end of each block section are used and dimensioned to ensure that no vehicle can enter an already occupied

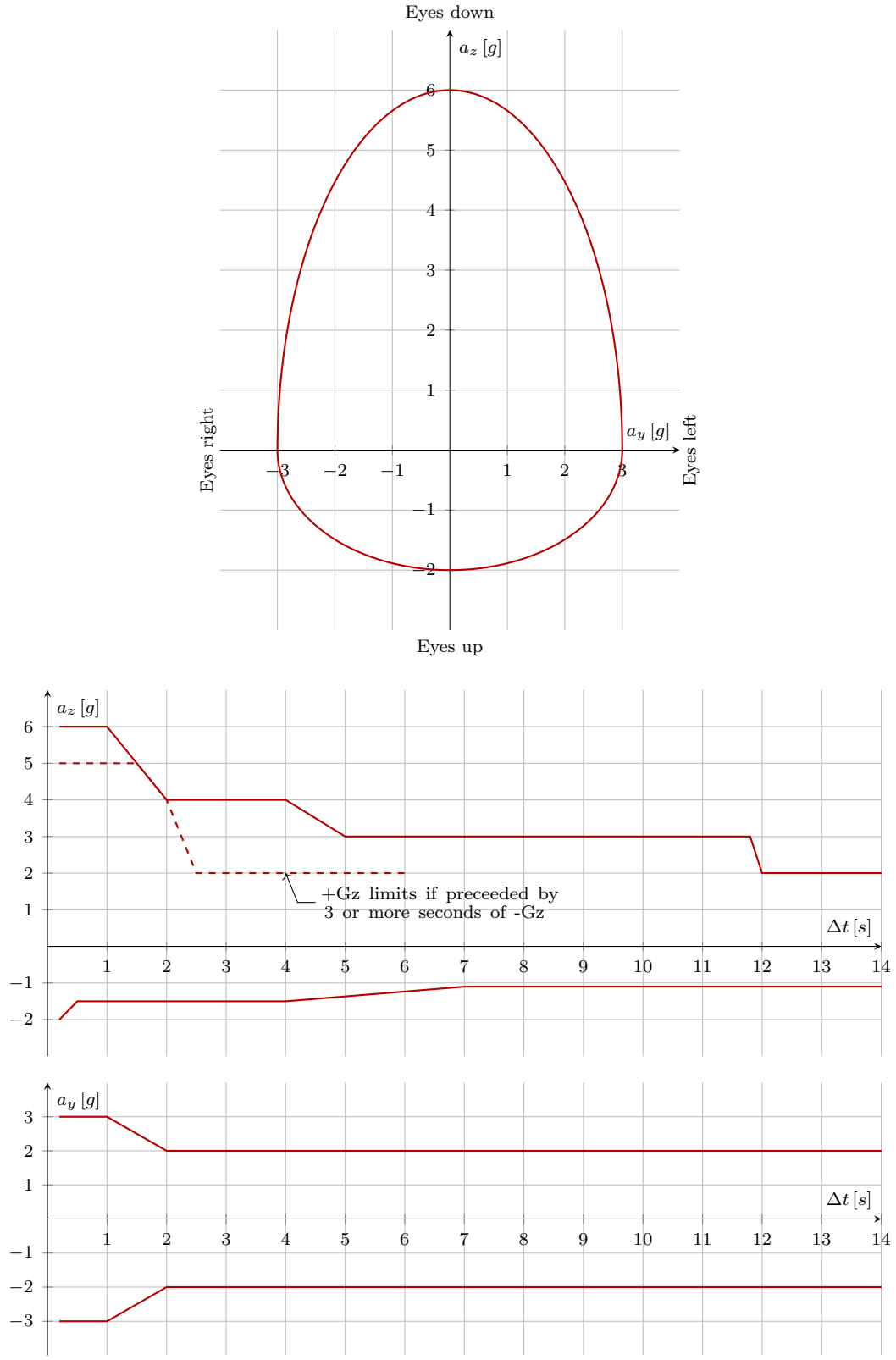


Figure 7.1: Combined acceleration limits (top) and acceleration duration limits for a_z (middle) and a_y (bottom) according to the American standard ASTM F2291 [5].

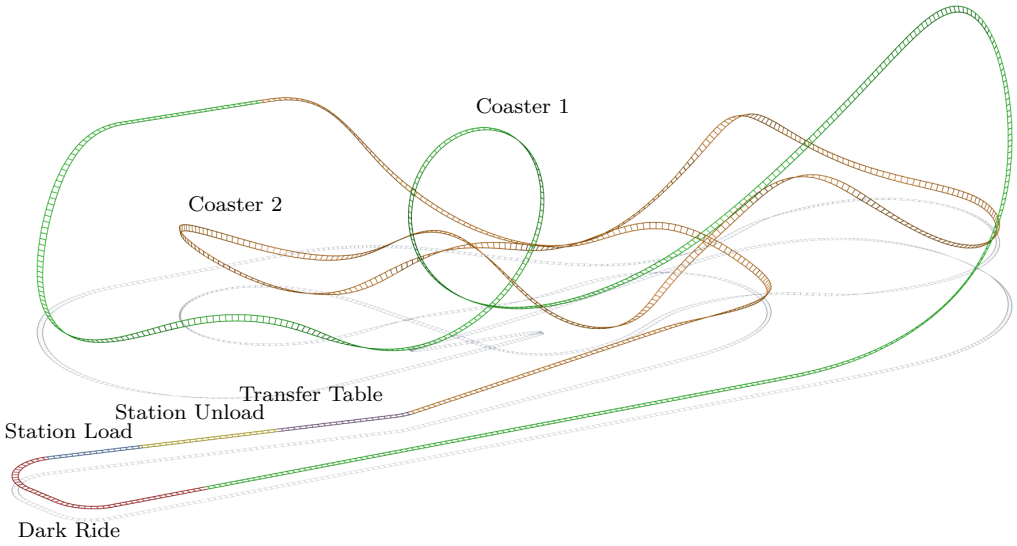


Figure 7.2: Exemplary layout and its logical subdivision into several block sections.

block [99, Section 2.4]. See Figure 7.2 on how a closed loop layout may be divided into several block sections.

Given a layout consisting of n block sections, we can calculate the block occupation time t_i for each block i as the longest expected time a vehicle remains within the respective block section. The longest block occupation time of a layout gives a strict upper bound for the *throughput* c measured in vehicles per unit of time of the layout. The corresponding block sections are therefore also often called *critical blocks*. Given a closed loop layout consisting of n blocks, a maximum of $n-1$ vehicles may be deployed on the track without creating a deadlock. Given the expected time t_{cycle} needed for a single vehicle to complete a full cycle of the complete layout, we can calculate the theoretical throughput c with $x < n$ vehicles in the loop as

$$c = \min\left(\frac{x}{t_{\text{cycle}}}, \frac{1}{t_i}\right), 1 \leq i \leq n.$$

Note: If there exists a critical block i such that

$$t_i > \frac{t_{\text{cycle}}}{x},$$

this block becomes the bottleneck of the layout.

In case the occupation times of all blocks are sufficiently close to each other, we

can identify the theoretical maximum throughput of the layout using $n-1$ vehicles as

$$c_{\max} = \frac{n-1}{t_{\text{cycle}}}.$$

Increasing the throughput further can only be achieved by adding block sections or reducing the cycle time. Additionally, we can express the throughput in terms of passengers per unit of time given the information how many people can be transported by each vehicle. The so-called *Theoretical Hourly Capacity* (THC) is an important characteristic for roller coasters and other transportation devices, and therefore needs to be considered early in the design process. Since the velocity of roller coaster vehicles varies widely along the track, from a standstill in the station blocks to large velocities in the gravity driven part of a coaster, the physical length of the different block sections has to be adapted accordingly. We refer again to Figure 7.2 that shows this effect.

During operation, the actual hourly capacity is influenced by a number of external factors such as the fraction of seats left empty, the loading and unloading time of the vehicles and the dynamic properties of the vehicles in the free-running sections of the layout. All these factors can be included in a stochastic model to simulate the actual operation of a roller coaster and give an estimate of the real throughput. Such simulations may identify flaws in the design, such as block sections that are designed without wide enough margin and that may consequently lead to cascading block stops in case of slower than average vehicles.

7.4 Clearance Envelope

Designers have to ensure that all vehicles can traverse a track layout safely. In particular, they have to avoid collisions of the vehicles and the occupants of the vehicles with surrounding obstacles or with the track structure itself. The so-called *clearance envelope* of a layout is the surface that marks the boundary of the volume that needs to stay clear of obstacles. Consequently, a track layout has to be designed in such a way that it is free of clashes between any obstacles and its clearance envelope. In the case of roller coasters, the various international standards describe how the clearance envelope relative to any given coach has to be determined, with Figure 7.3 showing the requirements of the already mentioned American standard ASTM F2291 [5].

A track's clearance envelope needs to be determined from the coach trajectory along said track. A first approximation is given by the *sweep surface* of some closed plane curve $\boldsymbol{g}$ called *clearance profile* along the track, see Section 8.6. However, the actual cross section of the clearance envelope may vary along the track since the coach trajectories may differ significantly from the track. In this case, more sophisticated methods need to be employed. It can be sufficient to generate a

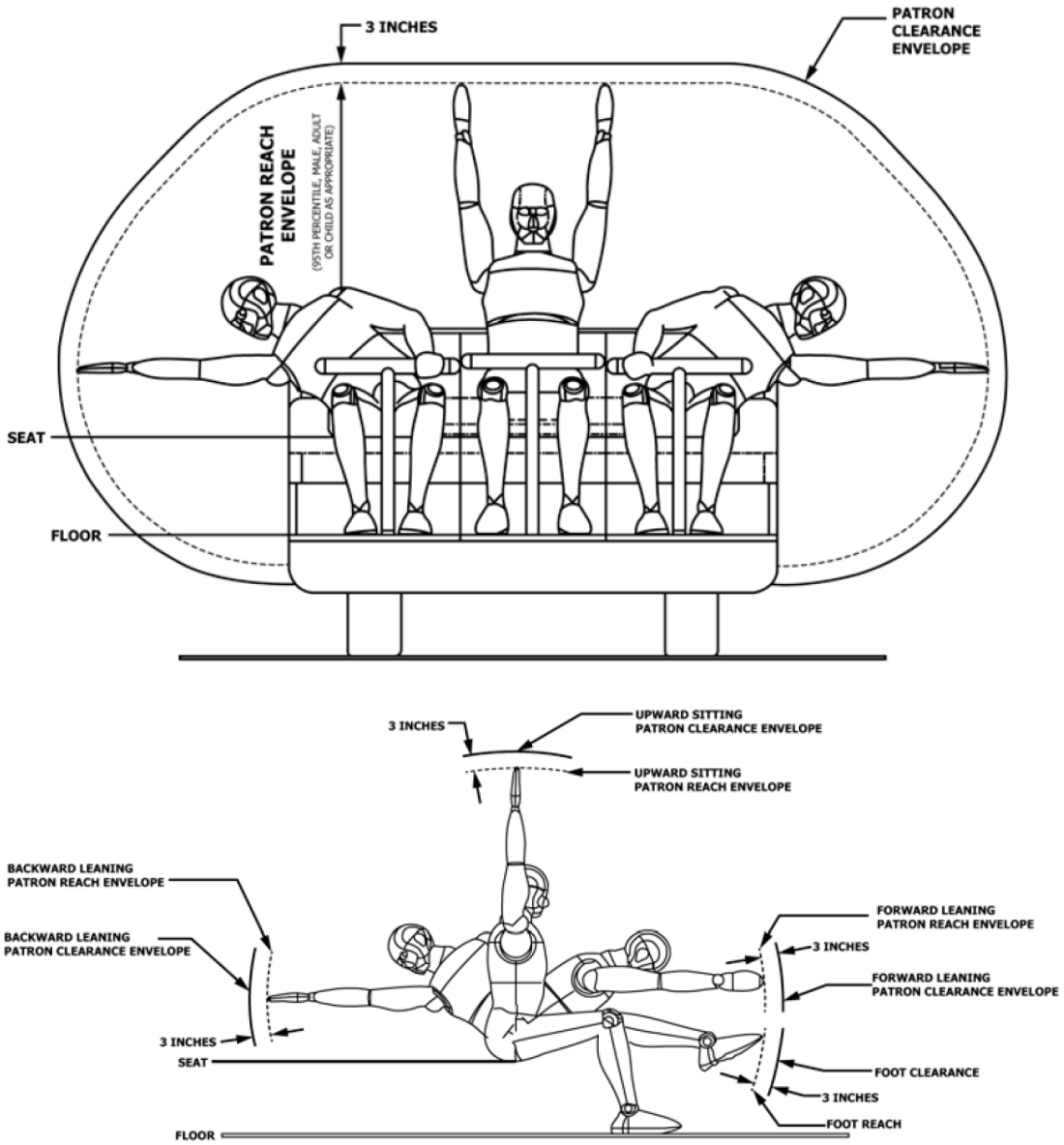


Figure 7.3: The clearance profile defines how far away obstacles have to be from a track [5].

conservative approximation by sweeping a larger profile $\boldsymbol{h}$ that includes all possible poses of the involved coaches relative to the track. Alternatively, the clearance profile $\boldsymbol{g}$ may be adapted along the track. In the most general case, the clearance envelope is constructed as a *swept volume* of the clearance volume with respect to the vehicles along their trajectory, see Section 8.9.

7.5 Other Extrinsic Constraints

All the previously mentioned constraints mainly focus on the feasibility of the layout itself, disregarding any external factors. In practice, a layout often has to be adapted to the local conditions of the targeted environment. Factors such as

required facilities, surrounding buildings, terrain, vegetation and overall project budget play an important role in the development of new layouts, not only in the context of roller coasters [67; 99]. In contrast to the constraints introduced above, most of these constraints can hardly be formalized, such that human intervention in the design process is crucial. Therefore, we focus on algorithms that can be deployed in an interactive setting. These kinds of algorithms need to give the designer enough control to satisfy all extrinsic requirements and constraints while producing feasible and fair tracks.

7.6 Acceleration Profile

Among the set of tracks that satisfy all constraints of a given application, the optimal track is to be identified. In the context of railway design and – to some extent – roller coaster design, tracks which induce a smooth gravitoinertial acceleration profile during traversal are preferred over ones that admit a jerky or bumpy acceleration profile. Assuming constant path velocity and gravitational accelerations acting only perpendicular to the track, the inertial accelerations generated by traversing a track are proportional to its curvature. Under these conditions, the optimal elementary curves used to design tracks are *straight lines*, which induce no accelerations, *circular arcs*, which induce constant accelerations, and *clothoids*, which induce constant jerk, as the connecting element between segments straight lines and circular arcs. Therefore, it is not surprising that these three elementary curves have been well established in classical railway design to this day.

In the context of roller coasters, a track is generally traversed with highly changing velocities and frequently admits significant slopes, with inertial and gravitational accelerations not necessarily acting perpendicular to each other. In the case of many roller coasters, the vehicles do not have propulsion systems, in which case the vehicle velocity is mainly determined by gravity, friction and the height profile of the track. Given some initial velocity vector $\boldsymbol{v}(0) = v_0 \boldsymbol{e}_1, v_0 > 0$, the gravitational acceleration vector $\boldsymbol{g} = g\boldsymbol{e}_2$ and a target gravitointerial acceleration a, the system of differential equations

$$\begin{aligned} \frac{d\boldsymbol{p}}{dt} &= \boldsymbol{v}(t) \\ \frac{d\boldsymbol{v}}{dt} &= \begin{bmatrix} -a\frac{\langle \boldsymbol{v}, \boldsymbol{e}_2 \rangle}{\|\boldsymbol{v}\|} \\ a\frac{\langle \boldsymbol{v}, \boldsymbol{e}_1 \rangle}{\|\boldsymbol{v}\|} - g \end{bmatrix} \end{aligned} \tag{7.2}$$

defines the shape of the corresponding *curve of constant acceleration* in $\mathbb{R}^2$. This initial value problem has a few solutions that admit a closed form representation of $\boldsymbol{p}(t)$, such as the circle

$$\boldsymbol{p}(t) = \begin{bmatrix} \frac{v_0^2}{a} \sin\left(\frac{a}{v}t\right) \\ \frac{v_0^2}{a}\left(1 - \cos\left(\frac{a}{v}t\right)\right) \end{bmatrix}$$

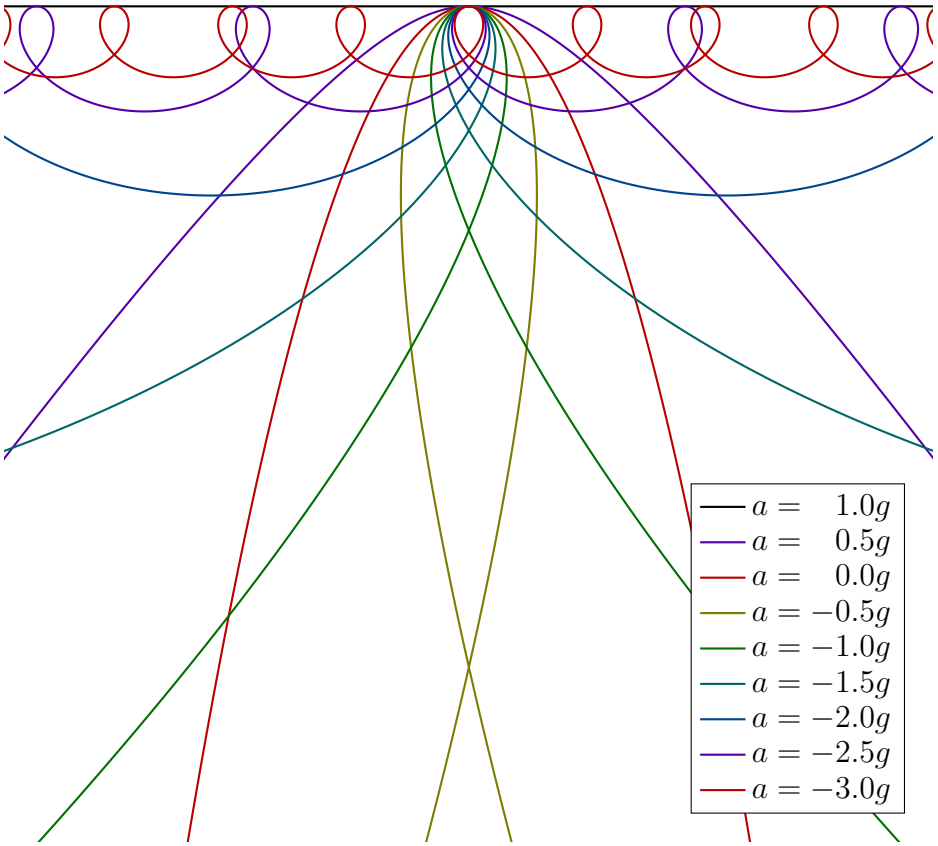


Figure 7.4: Curves of constant acceleration within a gravitational field.

for $g = 0$ and $a \neq 0$, the parabola

$$\boldsymbol{p}(t) = \begin{bmatrix} v_0 t \\ -\frac{g}{2} t^2 \end{bmatrix}$$

for $a = 0$ and $g \neq 0$, and the straight line

$$\boldsymbol{p}(t) = \begin{bmatrix} v_0 t \\ 0 \end{bmatrix}$$

for $a = g$. For all other cases, there is to our knowledge no closed form solution for the curve of constant acceleration. Nevertheless, we can compute approximations for these cases using numerical integration of the system of differential equations (7.2). Some of these approximations are shown in Figure 7.4. We can see, that for $\|a\| > g$ the curves resemble *prolate trochoids* with periodic behavior

$$\boldsymbol{p}(t) = \boldsymbol{p}(t + \Delta t) + \boldsymbol{q}.$$

For $0 < |a| < g$, we can show that the curves admit asymptotic behaviour and thus resemble *hyperbola*. For a velocity vector of the form

$$\boldsymbol{v} = v \begin{bmatrix} \cos(\alpha) \\ \sin(\alpha) \end{bmatrix},$$

we can calculate the rate of change of the slope angle α as

$$\dot{\alpha} = \frac{a - g\cos\alpha}{v}$$

after a few calculation steps. Thus, the slope angle α converges towards

$$\alpha = \arccos\left(\frac{a}{g}\right)$$

as $t \to \infty$.

In practice, curves of constant acceleration or approximations of them are widely used, and most predominantly they can be observed in the case of so-called *Airtime Hills*. These are long sections of unbanked track with $a < 0.2g$ in order to convey a feeling of weightlessness or even being ejected from one's seat. Figure 7.5 shows two examples of such sections and the corresponding curve of constant acceleration overlaid in black as a visual guide for the respective heartline trajectory. The first example uses a parabola ($a = 0g$) to generate a sustained sensation of weightlessness while the second example uses a curve with $a \approx -0.75g$ to convey a feeling of being ejected from the seat. Similarly, unbanked valleys of modern roller coasters closely follow the curves of constant acceleration with $2.5g < |a| < 4.5g$.

Note: When using Differential Equation (7.2), we model a traveling point mass inside a gravitational field and frictionless environment with both the inertial acceleration and the gravitational acceleration being applied to the same point mass. All of these assumptions are simplifications that can lead to significant inaccuracies when designing real-world tracks since we are interested in the gravitoinertial acceleration experienced by the patrons inside a coach with the center of gravity of the coach having a significant offset to that of the patron. Furthermore, most roller coaster vehicles are long trains consisting of several coaches, such that the resulting path velocity of the vehicle is determined by the locus of the combined center of gravity of all coaches. As a rule of thumb, the longer a vehicle in relation to the track's features and the larger the offset between a patron and the center of gravity of a single coach, the more deviation between the resulting acceleration and the target acceleration a we will encounter when using solutions of Equation (7.2). This, and the fact that it is easier to find unobstructed reference pictures of such coasters, is why we chose two relatively large roller coasters as examples in Figure 7.5.

Figure 7.5: Real-world examples of curves of constant acceleration [49; 105].

7.7 Lateral Accelerations

Designing a track with a smooth acceleration profile is important but by far not sufficient. As can be seen in Figure 7.1, how much humans are able to cope with accelerations depends on their angle of attack. In the sitting position, accelerations in the positive directions a_z and a_x are much more manageable for the human body [20, Chapter 4; 106, Section 6.5], which is why lateral accelerations a_y imposed on the human body should be reduced to reasonable levels. In the context of roller coasters, the layout designer is able to directly influence the angle of attack of the gravitoinertial accelerations in the y-z-plane by varying the roll angle γ of the track. In addition to designing a smooth acceleration profile, the designer's task is to find a banking profile resulting in smooth and relatively low lateral accelerations $|a_y|$. As a rule of thumb, trajectories resulting in lateral accelerations below $|a_y| < 1g$ while keeping the jerk below $|\dot{a}_y| < 10g/s$ have shown to be comfortable.

7.8 Angular Motion

As with linear accelerations, there are human limits for enduring angular motion. Unfortunately, most work on human tolerance of angular motion focusses on so-called *transient* angular accelerations, since they are a common cause for traumatic brain injury. Some guidance on sustained rotational motion is given by Whitmore et al. [106, Section 6.5.2]. Since current standards give no guidance on their limits, a designer should be careful not to expose a human patron to excessive angular motion. For traditional roller coasters, this is especially true for rotations about the local x-axis, since they are decoupled from the linear accelerations induced by the track and therefore usually admit the highest angular rates and accelerations. In Chapter 9, we describe how angular motion along spatial tracks can be calculated and techniques for limiting said motion by means of modifying the underlying track will be described in Chapter 13.

8 Representing Spatial Tracks

We can use framed curves $\mathbf{T}$ as introduced in Section 5 to represent spatial tracks in a straightforward fashion. As shown in Figure 8.1, the moving frame

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{t} & \boldsymbol{u} & \boldsymbol{v} \end{bmatrix}$$

of a framed curve $\mathbf{T}$ can be used to describe the local orientation of a track. Throughout this thesis, we will use the normal $\boldsymbol{u}$ as the local left vector and the normal $\boldsymbol{v}$ as the local up vector. In order to model a track with gauge w similarly to the one shown in Figure 8.1, the trajectory of the two corresponding rails can be determined by evaluating the *offset curves*

$$\begin{aligned} \boldsymbol{p}_{\text{left}} &= \boldsymbol{p} + 0.5w\boldsymbol{u} \\ \boldsymbol{p}_{\text{right}} &= \boldsymbol{p} - 0.5w\boldsymbol{u}. \end{aligned}$$

Since $\boldsymbol{u}$ is guaranteed to be perpendicular to the curve's tangent $\boldsymbol{t}$, the gauge w is maintained throughout the track.

The curvatures κ_g, κ_n of $\mathbf{T}$ are the reciprocals of the horizontal and vertical track radii

$$r_h = \kappa_g^{-1}$$

and

$$r_v = -\kappa_n^{-1},$$

where left curves and valleys have a positive radius r_h or r_v, respectively. The relative torsion τ_r of $\mathbf{T}$ indicates how much the track twists about its tangent vector and therefore is a representation of the physical torsion of the track. Such a twisted track section can be seen in Figure 8.3 below.

Remark: Since the curvatures κ_g, κ_n and τ_r are purely geometric properties of a framed curve, they are independent of the reference coordinate system. In particular, the radius r_h is not the radius $\bar{r}_h$ of the projection of $\boldsymbol{p}$ into the global x-y-plane. Similarly, the radius r_v is not the radius $\bar{r}_v$ of the height profile $z(\bar{s})$ of $\boldsymbol{p}$. These two radii $\bar{r}_h$ and $\bar{r}_v$ are commonly used in classical layout design as introduced in Section 1.1 and can be computed from the framed curve $\mathbf{T}$ as

$$\begin{aligned} \bar{r}_h^{-1} &= -\frac{\sin(\gamma)}{\cos^2(\alpha)}\kappa_n + \frac{\cos(\gamma)}{\cos^2(\alpha)}\kappa_g \\ \bar{r}_v^{-1} &= -\cos(\gamma)\,\kappa_n - \sin(\gamma)\,\kappa_g \end{aligned}$$

with γ and α being the roll and pitch angles of the moving frame $\boldsymbol{Q}$.

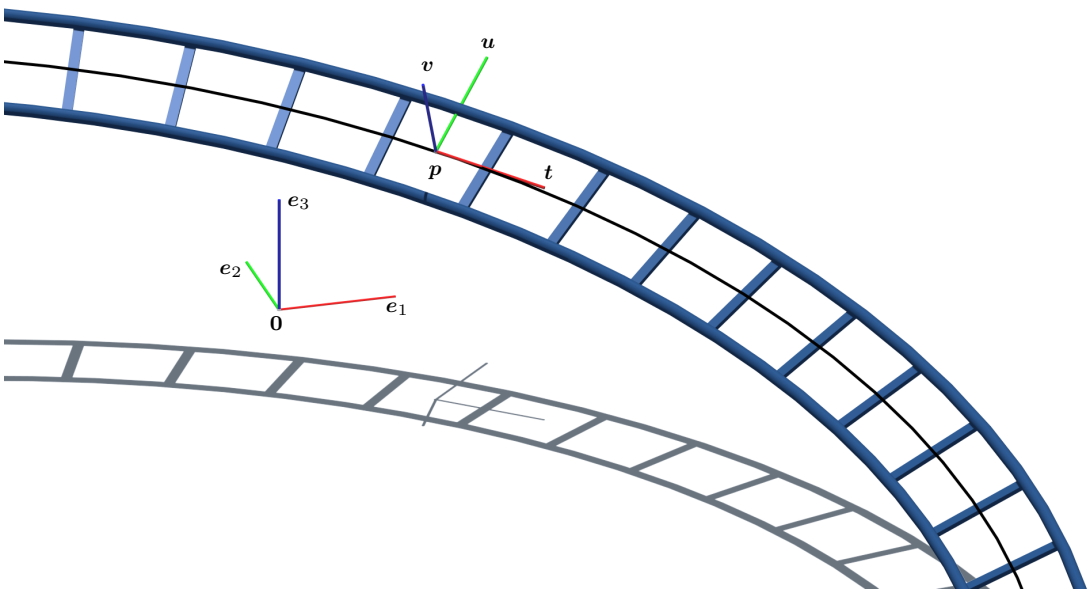


Figure 8.1: Framed curves carry all information needed to model spatial tracks.

8.1 Approximating the Arc Length Parameterization

In real-world applications, a common task is to determine the arc length of a section of track or to evaluate the track $\mathbf{T}$ with respect to its arc length. In Section 5.1 we already discussed that there is in general no exact representation of $\mathbf{T}(s)$ for most types of curves. However, we have found that determining

$$s(u) = \int_{u_0}^{u} \|\boldsymbol{p}'(v)\| \, dv$$

numerically with sufficient accuracy, e.g., to well below the manufacturing tolerance or range of expected thermal expansion of the track, is sufficient in practice. Hence, there is no need to limit the types of base curves to those with closed form arc length representations in these applications. Given an arc length s, it may be expensive to compute its corresponding parameter $u(s)$ and consequently $\mathbf{T}(u(s))$ with sufficient accuracy. Thus, we precompute $\mathbf{T}(u_i)$ for sufficiently dense samples $u_i = u(s_i)$ of the arc length s. In any follow-up computation, we can then approximate $\mathbf{T}(u(s))$ by looking up and interpolating the precomputed samples $\mathbf{T}(u_i)$. The samples s_i and the interpolation scheme need to be chosen such that this interpolation will yield results that meet the precision requirements of the underlying application.

8.2 Offset Curves

The small example of evaluating the individual rails in the opening of this chapter already shows that *offset curves* of a framed curve $\mathbf{T}$ play an important role in layout design, evaluation and especially track manufacturing.

Definition 8.1 (offset curve)**:** Given a framed curve $\mathbf{T}(u) \in SE(3)$ and a constant vector $\boldsymbol{r} \in \mathbb{R}^3$, the parametric curve

$$\mathbf{p}_{\boldsymbol{r}}(u) = \mathbf{T}(u) \begin{bmatrix} \boldsymbol{r} \\ 1 \end{bmatrix}$$

is called *offset curve* of $\mathbf{T}$ with respect to $\boldsymbol{r}$.

The first derivative of $\mathbf{p}_{\boldsymbol{r}}$ with respect to the arc length s of $\boldsymbol{p}$ can be determined as

$$\begin{aligned} \frac{d\mathbf{p}_{\boldsymbol{r}}}{ds} &= \mathbf{T} \cdot \begin{bmatrix} 1 - r_y\kappa_g + r_z\kappa_n \\ r_x\kappa_g - r_z\tau_r \\ -r_x\kappa_n + r_y\tau_r \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{t} + \boldsymbol{d} \times \boldsymbol{Q}\boldsymbol{r} \\ 0 \end{bmatrix} \end{aligned} \tag{8.1}$$

using Equations (5.15) and (5.16). With

$$\begin{aligned} n_x &= 1 - r_y\kappa_g + r_z\kappa_n, \\ n_y &= \quad r_x\kappa_g - r_z\tau_r, \\ n_z &= -r_x\kappa_n + r_y\tau_r, \end{aligned}$$

and

$$\begin{aligned} \left\| \frac{d\boldsymbol{p}_{\boldsymbol{r}}}{ds} \right\| &= \frac{ds_{\boldsymbol{r}}}{ds} \\ &= \sqrt{n_x^2 + n_y^2 + n_z^2} \end{aligned} \tag{8.2}$$

being the parametric speed of the offset curve $\boldsymbol{p}_{\boldsymbol{r}}(s)$ with respect to $\boldsymbol{p}$'s arc length, we can compute the tangent vector of $\boldsymbol{p}_{\boldsymbol{r}}$ as

$$\boldsymbol{t}_{\boldsymbol{r}} = \frac{\boldsymbol{t} + \boldsymbol{d} \times \boldsymbol{Q}\boldsymbol{r}}{\sqrt{n_x^2 + n_y^2 + n_z^2}}. \tag{8.3}$$

Remark: The offset curve $\boldsymbol{p}_{\boldsymbol{r}}$ is not regular if $\boldsymbol{t} + \boldsymbol{d} \times \boldsymbol{Q}\boldsymbol{r} = \boldsymbol{0}$. In practice during layout design, we usually have $\|\boldsymbol{r}\| \cdot \|\boldsymbol{d}\| < 1$ since the curvatures of the track are limited and the offsets of interest usually are close to the track center. In the following, we therefore assume that $\boldsymbol{p}_{\boldsymbol{r}}$ is regular, except where explicitly stated otherwise.

8.3 Framed Offset Curves

To be able to iteratively apply offsets, we need to assign a moving frame $\boldsymbol{Q_r} \in SO(3)$ to an offset curve $\boldsymbol{p_r}$ based on its parent curve $\mathbf{T}$. Since $\boldsymbol{Q_r}$ is only a valid moving frame of $\boldsymbol{p_r}$ if

$$\boldsymbol{Q_r e}_1 = \boldsymbol{t_r}, \tag{8.4}$$

there exist a one-parameter family of possible frames due to the choice of $\boldsymbol{u_r}$ and $\boldsymbol{v_r}$. We prefer a moving frame $\boldsymbol{Q_r}$ that satisfies

$$\boldsymbol{Q}^\mathsf{T}\boldsymbol{Q_r r} = \boldsymbol{r}, \tag{8.5}$$

because we can then reconstruct the original curve $\boldsymbol{p}$ as the offset

$$\begin{aligned}\boldsymbol{p} &= \mathbf{T}_r \begin{bmatrix} -\boldsymbol{r} \\ 1 \end{bmatrix} \\ &= -\boldsymbol{Q_r r} + \boldsymbol{Q r} + \boldsymbol{p}\end{aligned}$$

of the framed offset curve

$$\mathbf{T}_r = \begin{bmatrix} \boldsymbol{Q_r} & \boldsymbol{p_r} \\ 0 & 1 \end{bmatrix}.$$

For Equation (8.5) to hold, $\boldsymbol{r}$ has to be an eigenvector of the rotation matrix $\boldsymbol{Q}^\mathsf{T}\boldsymbol{Q_r}$. Since the only (real) eigenvector of a non-trivial rotation matrix is its rotation axis vector, $\boldsymbol{Q}^\mathsf{T}\boldsymbol{Q_r} = \mathrm{Rot}(\boldsymbol{r}, \phi)$ follows directly from Equation (8.5).

Proposition 8.1: The rotation $\boldsymbol{Q}\mathrm{Rot}(\boldsymbol{r}, \phi)$ represents a valid moving frame, i.e., it satisfies (8.4), if $\langle \boldsymbol{r}, \boldsymbol{e}_1 \rangle = 0$.

Proof. Without loss of generality, we can assume $\boldsymbol{Q} = \boldsymbol{I}$ for the proof of this proposition. We can evaluate the first column vector of $\mathrm{Rot}(\boldsymbol{r}, \phi)$ by applying Equation (4.14). Using Equation (8.1),

$$\hat{\boldsymbol{r}} = \frac{\boldsymbol{r}}{\|\boldsymbol{r}\|}$$

and several trigonometric identities, Equation (8.4) becomes

$$\begin{bmatrix} 1 - \left(\hat{r}_y^2 + \hat{r}_z^2\right)(1 - \cos(\phi)) \\ \hat{r}_x\hat{r}_y(1 - \cos(\phi)) - \hat{r}_z \sin(\phi) \\ \hat{r}_x\hat{r}_z(1 - \cos(\phi)) + \hat{r}_y \sin(\phi) \end{bmatrix} = \frac{ds}{ds_{\boldsymbol{r}}} \begin{bmatrix} 1 - r_y\kappa_g + r_z\kappa_n \\ r_x\kappa_g - r_z\tau_r \\ -r_x\kappa_n + r_y\tau_r \end{bmatrix},$$

with the rotation angle ϕ being the only unknown. For $r_x \neq 0$ this has no solution ϕ

for general framed curves $\mathbf{T}$. For $r_x = 0$, we can further simplify the equation above and obtain

$$\sin(\phi) = \|\boldsymbol{r}\| \frac{ds}{ds_{\boldsymbol{r}}} \tau_r$$

as the solution ϕ satisfying Equation (8.4). □

In fact, for $r_x = 0$, the rotation $\mathrm{Rot}(\boldsymbol{r}, \phi)$ is the minimal rotation, i.e., the rotation with the smallest rotation angle, that maps $\boldsymbol{e}_1$ to $\boldsymbol{Q}^\mathsf{T}\boldsymbol{t}_{\boldsymbol{r}}$. Its rotation axis vector is

$$\begin{aligned}
\boldsymbol{a} &= \boldsymbol{e}_1 \times \boldsymbol{Q}^\mathsf{T}\boldsymbol{t}_{\boldsymbol{r}} \\
&= \frac{ds}{ds_{\boldsymbol{r}}} \begin{bmatrix} 0 \\ -n_z \\ n_y \end{bmatrix} \\
&= \frac{ds}{ds_{\boldsymbol{r}}} \begin{bmatrix} 0 \\ r_x\kappa_n - r_y\tau_r \\ r_x\kappa_g - r_z\tau_r \end{bmatrix}
\end{aligned}$$

and the corresponding rotation angle is

$$\begin{aligned}
\phi &= \arccos(\langle \boldsymbol{t}, \boldsymbol{t}_{\boldsymbol{r}} \rangle) \\
&= \arctan\left(\frac{\sqrt{n_y^2 + n_z^2}}{n_x} \right).
\end{aligned}$$

From here, we can verify that

$$\boldsymbol{a} = -\tau_r \frac{ds}{ds_{\boldsymbol{r}}} \boldsymbol{r}$$

for $r_x = 0$. After a rather lengthy computation involving Equation (4.14) and several trigonometric identities, we get the corresponding rotation matrix

$$\mathrm{Rot}(\boldsymbol{a}, \phi) = \begin{bmatrix} n_x \frac{ds}{ds_{\boldsymbol{r}}} & -n_y \frac{ds}{ds_{\boldsymbol{r}}} & -n_z \frac{ds}{ds_{\boldsymbol{r}}} \\ n_y \frac{ds}{ds_{\boldsymbol{r}}} & \frac{n_z^2}{n_z^2+n_y^2} + \frac{n_y^2 n_x}{n_z^2+n_y^2} \frac{ds}{ds_{\boldsymbol{r}}} & \frac{n_z n_y}{n_z^2+n_y^2} \left(n_x \frac{ds}{ds_{\boldsymbol{r}}} - 1 \right) \\ n_z \frac{ds}{ds_{\boldsymbol{r}}} & \frac{n_z n_y}{n_z^2+n_y^2} \left(n_x \frac{ds}{ds_{\boldsymbol{r}}} - 1 \right) & \frac{n_y^2}{n_z^2+n_y^2} + \frac{n_z^2 n_x}{n_z^2+n_y^2} \frac{ds}{ds_{\boldsymbol{r}}} \end{bmatrix}. \tag{8.6}$$

Remark: For $n_y = 0, n_z \neq 0$, the matrix simply is

$$\mathrm{Rot}(\boldsymbol{a}, \phi) = \begin{bmatrix} \frac{n_x}{\sqrt{n_x^2+n_z^2}} & 0 & -\frac{n_z}{\sqrt{n_x^2+n_z^2}} \\ 0 & 1 & 0 \\ \frac{n_z}{\sqrt{n_x^2+n_z^2}} & 0 & \frac{n_x}{\sqrt{n_x^2+n_z^2}} \end{bmatrix}$$

and in case $n_z = 0, n_y \neq 0$ we get

$$\mathrm{Rot}(\boldsymbol{a}, \phi) = \begin{bmatrix} \frac{n_x}{\sqrt{n_x^2+n_y^2}} & -\frac{n_y}{\sqrt{n_x^2+n_y^2}} & 0 \\ \frac{n_y}{\sqrt{n_x^2+n_y^2}} & \frac{n_x}{\sqrt{n_x^2+n_y^2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Obviously, in both cases $\mathrm{Rot}(\boldsymbol{a}, \phi)$ converges to the identity matrix $\boldsymbol{I}$ for $n_y \to 0$ and $n_z \to 0$.

The rotation given by Equation (8.6) always satisfies Equation (8.4) and thus always assigns a valid moving frame to $\boldsymbol{p_r}$. Since it also satisfies (8.5) for $r_x = 0$, which due to Proposition 8.1 is the best we can do, it is a natural choice to assign a moving frame to an offset curve. The resulting framed offset curve $\mathbf{T}_{\boldsymbol{r}}$ can be expressed in terms of the base curve $\mathbf{T}$ as

$$\mathbf{T}_{\boldsymbol{r}} = \mathbf{T} \begin{bmatrix} \mathrm{Rot}(\boldsymbol{a}, \phi) & \boldsymbol{r} \\ \mathbf{0}^{\mathsf{T}} & 0 \end{bmatrix}. \tag{8.7}$$

For example, in case $\boldsymbol{r} = a \cdot \boldsymbol{e}_3$

$$\mathbf{T}_{\boldsymbol{r}} = \mathbf{T} \begin{bmatrix} \frac{1+a\kappa_n}{\sqrt{(1+a\kappa_n)^2+(a\tau_r)^2}} & \frac{a\tau_r}{\sqrt{(1+a\kappa_n)^2+(a\tau_r)^2}} & 0 & 0 \\ \frac{-a\tau_r}{\sqrt{(1+a\kappa_n)^2+(a\tau_r)^2}} & \frac{1+a\kappa_n}{\sqrt{(1+a\kappa_n)^2+(a\tau_r)^2}} & 0 & 0 \\ 0 & 0 & 1 & a \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

The curvatures of $\mathbf{T}_{\boldsymbol{r}}$ can be derived from Equation (5.15) as

$$\begin{aligned} \kappa_g &= \langle \boldsymbol{t}'_{\boldsymbol{r}}, \boldsymbol{u}_{\boldsymbol{r}} \rangle \\ \kappa_n &= \langle \boldsymbol{t}'_{\boldsymbol{r}}, \boldsymbol{v}_{\boldsymbol{r}} \rangle \\ \tau_r &= \langle \boldsymbol{v}'_{\boldsymbol{r}}, \boldsymbol{u}_{\boldsymbol{r}} \rangle \end{aligned}. \tag{8.8}$$

Expressing them with respect to the base curve $\mathbf{T}$ and its curvatures is possible, albeit rather involved for general offsets $\boldsymbol{r}$. As we will see later, even for more specific offsets $\boldsymbol{r}$ the corresponding expressions quickly become quite lengthy.

Remark: Given two offset vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, the offset curves

$$\begin{aligned} \mathbf{p}_{a,b} &= \mathbf{T}_{\boldsymbol{a}}\mathbf{b} \\ \mathbf{p}_{b,a} &= \mathbf{T}_{\boldsymbol{b}}\mathbf{a} \\ \mathbf{p}_{a+b} &= \mathbf{T} \begin{bmatrix} \boldsymbol{a}+\boldsymbol{b} \\ 1 \end{bmatrix} \end{aligned}$$

are in general not the same. This is due to the construction of the moving frame for offset curves and the fact that matrix multiplication is not commutative.

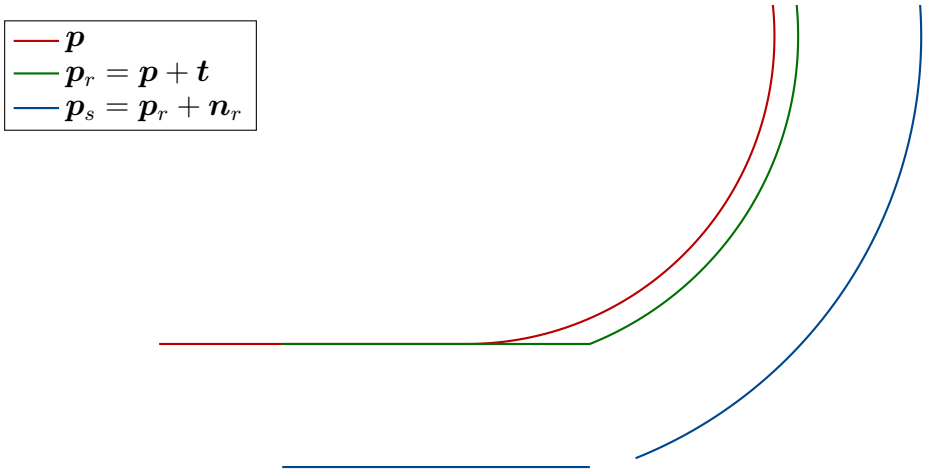


Figure 8.2: Offset curves have lower degree of continuity than their parent curve.

8.4 Continuity of Framed Curves

We can think of track layouts as framed splines consisting of several curve segments $\mathbf{T}_i$ which have some G^n contact. While the curves $\mathbf{T}_i$ themselves have some continuity, special care has to be taken to ensure continuity between the individual curves that make up the complete track. In order to classify the smoothness of framed curves, we first need to extend Definition 5.4 of geometric continuity of curves to framed curves.

Definition 8.2: A framed curve

$$\mathbf{T}(u) = \begin{bmatrix} \boldsymbol{t} & \boldsymbol{u} & \boldsymbol{v} & \boldsymbol{p} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

is in class G^n if $\boldsymbol{p}$ is a G^n curve and if $\boldsymbol{t}$, $\boldsymbol{u}$ and $\boldsymbol{v}$ of the arc length parameterization of $\boldsymbol{p}$ are in class C^{n-1}.

Using this definition, we can derive a few propositions.

Proposition 8.2: All RMFs $\boldsymbol{Q}$ of a G^n continuous curve $\boldsymbol{p}$ yield a G^n continuous framed curve $\mathbf{T}$.

Proof. By the definition of RMFs, their Darboux vector must satisfy $\langle \boldsymbol{d}, \boldsymbol{t} \rangle = 0$ and $\boldsymbol{t}' = \boldsymbol{d} \times \boldsymbol{t}$. If $\boldsymbol{p}$ is parametrized by its arc length, $\boldsymbol{t} = \boldsymbol{p}'$ is in C^{n-1}, $\boldsymbol{t}' = \boldsymbol{p}''$ is in C^{n-2} and

$$\begin{aligned} \boldsymbol{t} \times \boldsymbol{t}' &= \boldsymbol{t} \times (\boldsymbol{d} \times \boldsymbol{t}) \\ &= \boldsymbol{d} \end{aligned}$$

is in C^{n-2}. Starting with an initial moving frame

$$\boldsymbol{Q}(0) = \begin{bmatrix} \boldsymbol{t}(0) & \boldsymbol{u}(0) & \boldsymbol{v}(0) \end{bmatrix},$$

we can solve the differential equations

$$\begin{aligned} \boldsymbol{u}' &= \boldsymbol{d} \times \boldsymbol{u}, \\ \boldsymbol{v}' &= \boldsymbol{d} \times \boldsymbol{v}. \end{aligned}$$

This results in C^{n-1} continuous functions $\boldsymbol{u}(s)$ and $\boldsymbol{v}(s)$ by an argument similar to the one described by Struik in his proof of the *Fundamental Theorem for space curves*, see [89, Section 1-8], which completes the proof of Proposition 8.2. □

Remark: As opposed to RMFs, the Frenet frame of a G^n continuous curve $\boldsymbol{p}$ does in general not yield a G^n continuous framed curve since at inflections of $\boldsymbol{p}$ the normals $\boldsymbol{n}$ and $\boldsymbol{b}$ of the Frenet frame are discontinuous.

Proposition 8.3: All regular offset curves $\boldsymbol{p}_r$ of a G^n continuous framed curve $\mathbf{T}$ are G^{n-1} continuous.

Proof. Due to the construction of $\boldsymbol{p}_r$ as a linear combination of $\boldsymbol{t}$, $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{p}$ and with Definition 8.2, we know that $\boldsymbol{p}_r$ is in C^{n-1} if $\boldsymbol{p}$ has been reparametrized to its arc length. Since it is also regular it follows that $\boldsymbol{p}_r$ is in G^{n-1}. □

Proposition 8.4: All regular framed offset curves $\mathbf{T}_r$ of a G^n continuous framed curve $\mathbf{T}$ are G^{n-1} continuous.

Proof. As $\mathbf{T}$ is in G^n, its curvatures κ_n, κ_g and τ_r are in C^{n-2}. Since $\boldsymbol{r}$ is constant, $\mathrm{Trans}(\boldsymbol{r})$ is in C^∞ and all elements of $\mathrm{Rot}(\boldsymbol{a}, \phi)$ are in C^{n-2} as long as $\mathbf{T}_r$ is regular, see Equation (8.6). Thus, $\boldsymbol{t}_r$, $\boldsymbol{u}_r$ and $\boldsymbol{v}_r$ are in C^{n-2}. Using Proposition 8.3, we conclude that $\mathbf{T}_r$ is in G^{n-1}. □

Figure 8.2 shows a G^1 continuous plane curve $\boldsymbol{p}$ and two levels of offset curves with decreasing order of continuity. As we have shown, the same holds for spatial curves.

In the context of roller coasters, it is common to work with G^1, G^2 and G^3 curves, depending on the expected path velocity of the vehicles during traversal of the respective contacts. In some cases, it suffices to work with G^0 curves, depending on the direction and the magnitude of the discontinuity of the moving frame $\boldsymbol{Q}$. We will discuss and motivate these best-practices in terms of geometric continuity of tracks later in this chapter.

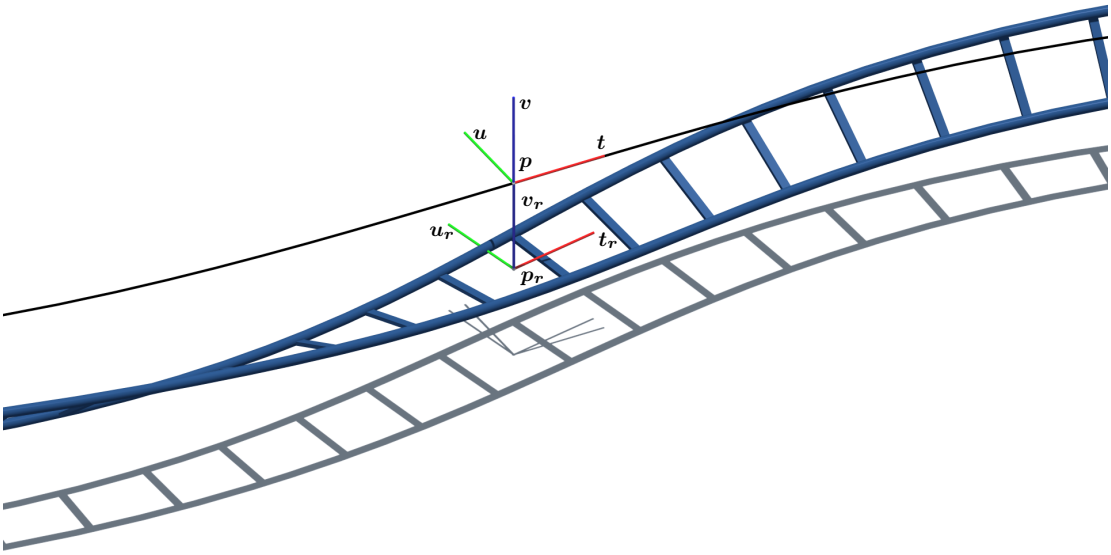


Figure 8.3: Using the heartline principle, the track center can be seen as the offset curve of the heartline and vice versa.

Remark: We can informally extend the classes of continuous functions to the class C^{-1} of all *Riemann integrable* functions (such as the *Heaviside step function*) since their antiderivatives are by the *Second Fundamental Theorem of Calculus* continuous and thus in C^0. Similarly, the *Dirac delta function* may be thought of as being in class C^{-2} since its antiderivative is in class C^{-1}.

8.5 Heartline Principle

Being able to compute offset curves of given framed curves allows us to apply the heartline principle introduced in Section 1.2 in a straightforward fashion. If we interpret the base curve $\mathbf{T}(u)$ as the heartline of a layout, we can compute the track's centerline $\mathbf{T}_{\boldsymbol{r}}$ as the framed offset curve for any value of $\boldsymbol{r}$. In practice, vectors of the form

$$\boldsymbol{r} = -h\boldsymbol{e}_3 \tag{8.9}$$

are almost exclusively used due to the symmetry of most track cross sections. Figure 8.3 shows a track constructed with $h = 1$. For any value of h, the resulting framed offset curve of the track's centerline is

$$\mathbf{T}_{\boldsymbol{r}} = \mathbf{T} \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 & 0 \\ \sin(\phi) & \cos(\phi) & 0 & 0 \\ 0 & 0 & 1 & -h \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

with

$$\phi = \arctan\left(\frac{h\tau_r}{1 - h\kappa_n}\right). \tag{8.10}$$

We can identify the curvatures of the track given the curvatures of the heartline as

$$\begin{aligned}
\kappa_{g,\boldsymbol{r}} &= \frac{\kappa_g}{\sqrt{(1-h\kappa_n)^2 + \tau_r^2 h^2}} + \frac{h^2\left(\kappa_n\tau_r' - \kappa_n'\tau_r\right) - h\tau_r'}{\sqrt{(1-\kappa_n h)^2 + \tau_r^2 h^2}^3} \\
\kappa_{n,\boldsymbol{r}} &= \frac{\kappa_n\left(1-\kappa_n h\right) - \tau_r^2 h}{(1-\kappa_n h)^2 + \tau_r^2 h^2} \\
\tau_{r,\boldsymbol{r}} &= \frac{\tau_r}{(1-\kappa_n h)^2 + \tau_r^2 h^2}
\end{aligned}$$

after a lengthy, but rather straightforward calculation using Equations (8.8) and (5.15). As we can see, only $\kappa_{g,\boldsymbol{r}}$ has a lower order of continuity than its respective counterpart κ_g after applying the offset $\boldsymbol{r}$ and assuming all curvatures κ_g, κ_n and τ_r have the same order of continuity. This is due the construction of the track's centerline $\boldsymbol{p_r}$ as a linear combination of $\boldsymbol{p}$ and $\boldsymbol{v}$, but neither $\boldsymbol{t}$ nor $\boldsymbol{u}$. Similarly, if we now construct the corresponding rails as the offset curves

$$\mathbf{p}_{\text{left}} = \mathbf{T}_{\boldsymbol{r}} \begin{bmatrix} 0 & 0.5w & 0 & 1 \end{bmatrix}^{\mathsf{T}}$$

and

$$\mathbf{p}_{\text{right}} = \mathbf{T}_{\boldsymbol{r}} \begin{bmatrix} 0 & -0.5w & 0 & 1 \end{bmatrix}^{\mathsf{T}},$$

only the respective normal curvature κ_n loses order of continuity by the same argument. We conclude that, when using offset vectors of the form (8.9), the final track geometry determined by the track's individual rails is in class G^{n-1} if we model the heartline as a G^n continuous framed curve, although the contact geometry has been constructed using two levels of offset curves.

8.6 Sweep Surfaces

In order to correctly and accurately render cylindrical rails as volumetric objects as shown in Figure 8.3, we need to evaluate them as a sweep surface.

Definition 8.3 (sweep surface)**:** Given a curve $\boldsymbol{g}(v)$, the *sweep* of $\boldsymbol{g}$ with respect to a framed curve $\mathbf{T}(u)$ is the parametric surface

$$\mathbf{p}(u, v) = \mathbf{T}(u)\mathbf{g}(v).$$

In practice, the curve $\boldsymbol{g}$ usually lies in the y-z-plane.

Example: Given the framed offset curve

$$\mathbf{T}_{\text{left}} = \mathbf{T} \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 & 0 \\ \sin(\phi) & \cos(\phi) & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\psi) & 0 & \sin(\psi) & 0 \\ 0 & 1 & 0 & 0.5w \\ -\sin(\psi) & 0 & \cos(\psi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

of the left rail with ϕ as in Equation (8.10) and

$$\psi = \arctan\left(\frac{-0.5w\tau_{r,\boldsymbol{r}}}{1 + 0.5w\kappa_{g,\boldsymbol{r}}}\right),$$

the parametric surface

$$\mathbf{p}(u,v) = \mathbf{T}_{\text{left}}(u) \begin{bmatrix} 0 \\ 0.5d\cos(v) \\ 0.5d\sin(v) \\ 1 \end{bmatrix}, v \in [0, 2\pi[\,,$$

defines the pipe surface with diameter d. Similarly, profiles other than circles can be swept in the same fashion. For example, the clearance profile of a roller coaster such as the one shown in the top half of Figure 7.3 can be swept along the track to approximate the clearance envelope, see Section 7.4.

8.7 Ruled Surfaces along Framed Curves

The sweep of a straight line $\boldsymbol{g}(v) = \boldsymbol{a}v + \boldsymbol{b}$ along a framed curve $\mathbf{T}$ is the *ruled surface*

$$\mathbf{p}(u,v) = \mathbf{T}(u) \begin{bmatrix} \boldsymbol{b} \\ 1 \end{bmatrix} + v\mathbf{T}(u) \begin{bmatrix} \boldsymbol{a} \\ 0 \end{bmatrix}. \tag{8.11}$$

For simplicity's sake, we will now only consider ruled surfaces where $\boldsymbol{b} = \mathbf{0}$.

Proposition 8.5: Given a framed curve $\mathbf{T}(u)$, the ruled surface $\mathbf{p}(u,v) = \boldsymbol{p}(u) + v\mathbf{T}(u)\mathbf{a}$ is developable if

1. the moving frame of $\mathbf{T}$ is an RMF and $\langle \boldsymbol{a}, \boldsymbol{e}_1 \rangle = 0$ or

2. $\boldsymbol{a} = a_x \boldsymbol{e}_1$, a case more commonly known as *tangent developable*.

Proof. As with the proof of Proposition 8.1, we can assume $\mathbf{T} = \mathbf{I}$ without loss of generality. A ruled surface $\boldsymbol{p}(u,v) = \boldsymbol{d}(u) + v\boldsymbol{c}(u)$ is developable, if and only if the

Figure 8.4: Real-world examples of a track constructed using ruled surfaces [6].

vectors $\boldsymbol{d}'$, $\boldsymbol{c}$ and $\boldsymbol{c}'$ are linearly dependent [12, Section 32.4]. Using Equations (5.15) and (8.11) we get

$$\begin{aligned}
\boldsymbol{d}' &= \mathbf{T}(u)\,\mathbf{e}_1, \\
\boldsymbol{c} &= \mathbf{T}(u) \begin{bmatrix} \boldsymbol{a} \\ 0 \end{bmatrix}, \\
\boldsymbol{c}' &= \mathbf{T}(u) \begin{bmatrix} 1 - \kappa_g a_y + \kappa_n a_z \\ \kappa_g a_x - \tau_r a_z \\ -\kappa_n a_x + \tau_r a_y \\ 0 \end{bmatrix}.
\end{aligned}$$

These are linearly dependent if

$$\begin{aligned}
\det \begin{bmatrix} \boldsymbol{d}' & \boldsymbol{c} & \boldsymbol{c}' \end{bmatrix} &= \det \begin{bmatrix} 1 & a_x & 1 - \kappa_g a_y + \kappa_n a_z \\ 0 & a_y & \kappa_g a_x - \tau_r a_z \\ 0 & a_z & -\kappa_n a_x + \tau_r a_y \end{bmatrix} \\
&= -a_y \left(\kappa_n a_x - \tau_r a_y\right) - a_z \left(\kappa_g a_x - \tau_r a_z\right)
\end{aligned}$$

vanishes. In both cases of Proposition 8.5 at least one factor of each term above evaluates to zero. Other cases exist, but either rely on specific ratios of the curvatures κ_g, κ_n and τ_r based on the vector $\boldsymbol{a}$ or $\boldsymbol{a}$ to vary with u. □

Figure 8.5: Triangle strips may be used to approximate ruled surfaces along a track [52].

In our context, ruled surfaces play an important role if any components of a bent track are constructed using metal sheets such as the track shown in Figure 8.4. These metal sheets are first cut, then deformed into their final shape before being welded together. Assuming infinitely thin and non-plastic sheets, the deformation of the initially flat sheets is only possible if the ruled surfaces are also developable, since their Gaussian curvature can not be changed. In practice, metal sheets are to an extend deformable, which allows us to use moving frames other than an RMF to construct spatial tracks this way. However, the more we deviate from an RMF, in other words the larger $\|\tau_r\|$, the more internal stress is applied on the metal sheet during the deformation process. Alternatively, the respective surfaces may be discretized and approximated by triangle strips, which are developable by construction, see Figure 8.5

8.8 Rectification

Given a framed curve $\mathbf{T}$, consider the map

$$f : \mathbb{R}^3 \to \mathbb{R}^3 \times \{0, 1\}, \boldsymbol{x} \mapsto \mathbf{T}(x_1) \begin{bmatrix} 0 \\ x_2 \\ x_3 \\ 1 \end{bmatrix}. \tag{8.12}$$

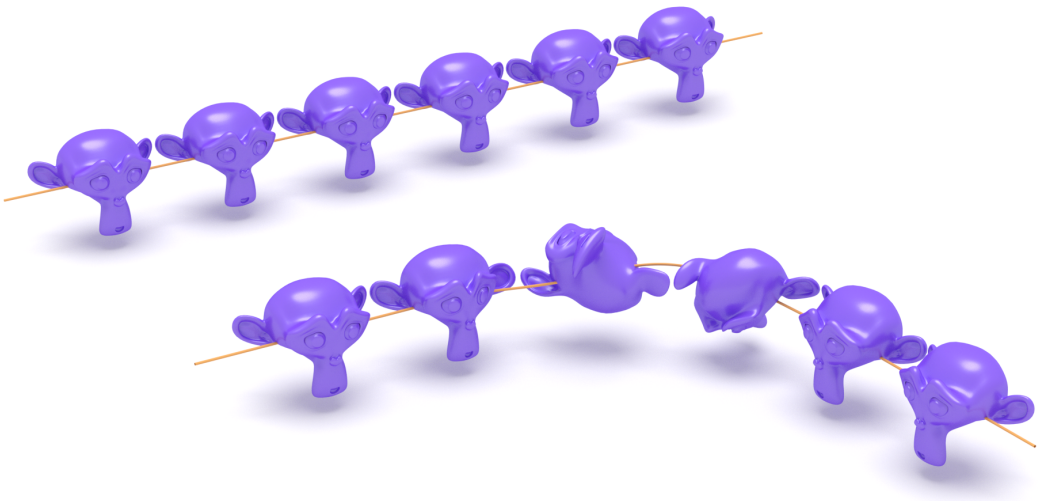


Figure 8.6: The object on the upper left is being deformed according to a framed curve, resulting in the object on the lower right.

If we apply f to all points $\boldsymbol{x}$ of a volumetric object, the resulting object will be deformed according to and along the framed curve $\mathbf{T}$ as shown in Figure 8.6 with the object being a chain of the commonly used 3D model *Suzanne*.

Proposition 8.6: The map f given by Equation (8.12) is conformal if $\mathbf{T}$ is framed using an RMF.

Proof. The Jacobian matrix of f is

$$\begin{aligned}
\boldsymbol{J} &= \begin{bmatrix} \frac{df}{dx_1} & \frac{df}{dx_2} & \frac{df}{dx_3} \end{bmatrix} \\
&= \begin{bmatrix} (\tau_r \boldsymbol{v}(x_1) - \kappa_g \boldsymbol{t}(x_1))\, x_2 + (\kappa_n \boldsymbol{t}(x_1) - \tau_r \boldsymbol{u}(x_1))\, x_3 + \boldsymbol{t}(x_1) & \boldsymbol{u}(x_1) & \boldsymbol{v}(x_1) \end{bmatrix} \\
&= \begin{bmatrix} x_2 \tau_r \boldsymbol{v}(x_1) - x_3 \tau_r \boldsymbol{u}(x_1) + \boldsymbol{t}(x_1)\,(1 - x_2 \kappa_g + x_3 \kappa_n) & \boldsymbol{u}(x_1) & \boldsymbol{v}(x_1) \end{bmatrix},
\end{aligned}$$

whose columns are orthogonal (but not necessarily orthonormal) for $\tau_r = 0$. This is a sufficient condition for f to be conformal. □

The *inverse map* f^{-1} determines for which parameters u a particular point $\boldsymbol{x}$ lies in the normal plane of $\mathbf{T}$. As can be seen in Figure 8.7, sweeps along the curve $\mathbf{T}$ are straightened out by f^{-1}, resulting in the sweep of the same profile along a straight line. This property is particularly useful in the context of verifying the clearance of a spatial track, as long as the clearance envelope itself is constructed as a sweep surface. Without using f^{-1} we need to search for volume intersections of the clearance envelope with the surrounding objects. Using f^{-1} on the surroundings

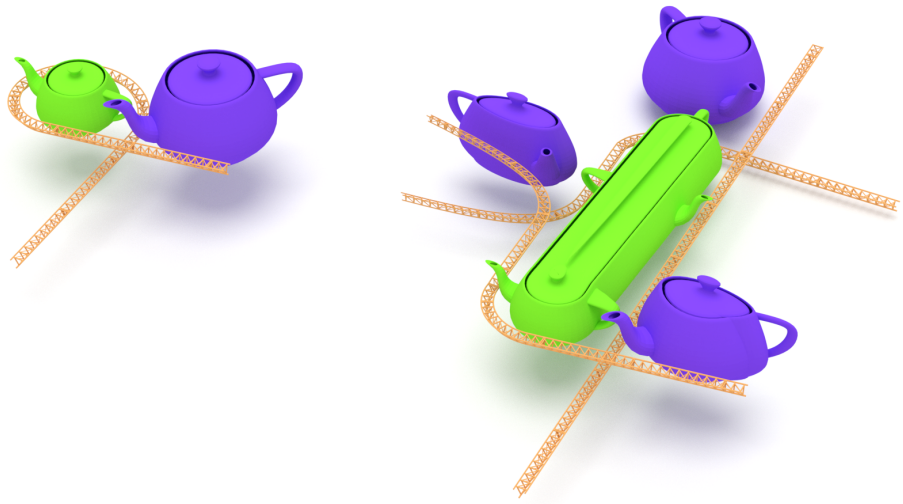


Figure 8.7: Recitification mapping along the orange track from left to right. The mapping is obviously not injective.

of a track reduces the problem to searching for planar intersections of the mapped surrounding objects with the clearance profile.

When implementing an algorithm that computes the image of an object with respect to the *inverse map* f^{-1}, special care has to be taken for regions of space near the rotation axis of the curve's normal plane [25]. In this region the topology of the corresponding object changes since the normal plane passes through the object in different directions, as is the case for the green teapot in Figure 8.7. Additionally, since the normal plane of the curve may pass any point in space several times, we can get duplicated objects by computing f^{-1} of an object, as is the case for the orange track and the blue teapot in Figure 8.7.

8.9 Swept Volumes

As already mentioned, the most accurate way of determining the clearance envelope of a track is to consider the reach volume $V \subset \mathbb{R}^3$ of patrons sitting inside a coach $\mathcal{C}$ relative to the coach coordinate system and then generating the *swept volume* of V along the coach trajectory $\mathbf{T}_\mathcal{C}$, which may be some general rigid body motion. An example of a swept volume is shown on the right side of Figure 8.8, with the input volume V in a few characteristic poses along its trajectory shown on the left side of the Figure.

Figure 8.8: Swept volume of a sample object along a self-intersecting trajectory.

Definition 8.4: Given a volume $V \subset \mathbb{R}^3$ and a trajectory $\mathbf{T}, [u_0, u_1] \to \mathrm{SE}(3)$, the volume

$$U : \bigcup \mathbf{T}(u)\mathbf{q}, u \in [u_0, u_1], \mathbf{q} \in V,$$

is called *swept volume*.

Several contributions have been made to approximate U or its boundary δU given V or δV and $\mathbf{T}$. An overview of the extensive literature on this subject including its applications is given by Abdel-Malek et al. [1]. We will now shortly discuss two more recent contributions. Peternell et al. [69] calculate the *characteristic curve*, that is the curve on δV for which every point moves tangentially to δV, at each time step and connect these characteristic curves to approximate δU. Von Dziegielewski et al. approximate U using six depth buffer images [97] and later using an octree [98]. They reconstruct an approximation of δU by using the so-called *Delaunay refinement*. Furthermore, they achieve a conservative approximation of δU by first expanding their volume representation before applying the Delaunay refinement.

An important observation with respect to swept volumes is the following. If V is given by it's *signed distance function (SDF)* [68, Chapter 2] $\phi_V(\mathbf{x})$, we can calculate the SDF of U by evaluating

$$\phi_U(\mathbf{x}) = \min_u \left(\phi_V\left(\mathbf{T}^{-1}(u)\mathbf{x}\right)\right).$$

Generating a discretized approximation of δU can be achieved, e.g., by applying the *Marching Cubes Algorithm* [56] on ϕ_U [77]. In fact, this method has been used to generate the swept volume shown in Figure 8.8.

9 Kinematics along Framed Curves

Given a track represented by its framed curve $\mathbf{T}(s)$, we can treat it or any of its offset curves as a one parameter joint in the context of kinematic analysis. It is possible to describe the motion of a body $\mathcal{B}$ directly connected to the track (inheriting the track's frame) by introducing the reparameterization $s(t)$, which defines the position s of $\mathcal{B}$ along the curve at every given time t. The linear velocity

$$\begin{aligned} \dot{\boldsymbol{p}} &= \frac{ds}{dt}\frac{d\boldsymbol{p}}{ds} \\ &= v\boldsymbol{t} \end{aligned} \tag{9.1}$$

and acceleration

$$\begin{aligned} \boldsymbol{a} &= \ddot{\boldsymbol{p}} \\ &= \dot{v}\boldsymbol{t} + v^2\boldsymbol{t}' \\ &= \dot{v}\boldsymbol{t} + v^2\left(\kappa_g\boldsymbol{u} - \kappa_n\boldsymbol{v}\right), \end{aligned} \tag{9.2}$$

of $\mathcal{B}$ are dependent on the differential properties of $\mathbf{T}$ and the path velocity v and acceleration $\dot{v}$ of $\mathcal{B}$ along $\mathbf{T}$. We can see that the well-known formulas for the centripetal accelerations emerge naturally from the curve's differential properties. Similarly, we can compute the angular velocities and accelerations of $\mathcal{B}$ using the Darboux vector $\boldsymbol{d}$ as

$$\boldsymbol{\omega} = v\boldsymbol{d}. \tag{9.3}$$

Determining the angular accelerations from the above equation as

$$\dot{\boldsymbol{\omega}} = \boldsymbol{d}'v^2 + \boldsymbol{d}\dot{v} \tag{9.4}$$

is then straightforward.

For most real-world applications, considering single bodies directly connected to some framed curve does not yield sufficiently accurate results, since the connection of the relevant bodies to the track is more indirect. For roller coasters, the patrons sitting in a coach are connected via some kinematic chain to the wheel bogies, which in turn consist of several wheels that may have some dampened and suspended contact to the rails, while other wheels may have no contact at all. In order to accurately model the resulting motion, we need to solve the corresponding *inverse dynamics* problem, e.g., using *Featherstone's Algorithm* [28, Chapter 5]. Since roller coaster trains usually consist of dozens of rigid bodies and thus have hundreds of mobilities, with several non-linear force elements connecting the wheels with the rails, such a method would certainly not be feasible for real time or even faster than real time analysis, which is to be preferred in interactive CAD applications, such as the one we are proposing.

Instead, we seek a compromise between computation time and accuracy when it

comes to calculating the motion of the individual bodies. Our first simplification arises from the assumption that we can express the Euclidean motion $\mathbf{T}_i(s)$ of each of the train's bodies $\mathcal{B}_i, 0 \leq i < n$, at any point along the track given only the train's kinematic structure and the underlying framed curve $\mathbf{T}(s)$, but crucially not given the train's motion and dynamic properties. This separates the concerns between computing train kinematics and train dynamics, allowing us to compute the trajectories $\mathbf{T}_i(s)$ in parallel by splitting up the track into several sections which can be evaluated concurrently. If we know the trajectories $\mathbf{T}_i(s)$ with respect to the curves's arc length, we only need to compute the reparametrization $s(t)$ to evaluate the approximated velocities and accelerations of all bodies. While computing the $\mathbf{T}_i(s)$ is a variant of inverse kinematics, we can determine the function $s(t)$ by numerical integration of the train's path velocity $v_t(t) = \dot{s}(t)$. On most roller coasters, the train runs freely on the track for extensive sections such that the *law of conservation of energy* can be used to determine v_t efficiently [36, Section 3.1.2]. Given the masses m_i and moments of inertia $\boldsymbol{I}_i$ of the train's n individual bodies, the most important contributions to the total energy of the train are the potential energy

$$E_{\text{pot}} = g \sum_{i=0}^{n-1} m_i \left\langle \boldsymbol{p}_i, \boldsymbol{e}_3 \right\rangle$$

assuming that the origin of the local reference frame $\mathbf{T}_i(s)$ of each body is located at its respective center of gravity, the kinetic energy

$$E_{\text{kin}} = \frac{1}{2} \sum_{i=0}^{n-1} m_i \dot{\boldsymbol{p}}_i^{\,2} + \boldsymbol{\omega}_i^{\mathrm{T}} \boldsymbol{I}_i \boldsymbol{\omega}_i \tag{9.5}$$

and energies transfered between the train and its environment

$$E_{\text{env}} = \int_0^s F_{\text{env}} \, ds',$$

such as friction, internal or external propulsion systems or braking devices. The only unknown of the resulting energy equation

$$E = E_{\text{kin}} + E_{\text{pot}} + E_{\text{env}} \tag{9.6}$$

is the path velocity v_t. Since solving these equations is simple compared to solving the inverse kinematics problem, it is plausible to compute several different friction and loading configurations resulting in various different reparameterizations $s(t)$ without much overhead.

Note: Due to the location of the centers of gravity of the involved rigid bodies, there is a subtle difference between the train's velocity $\|\dot{\boldsymbol{p}}\| = v$ in Equation (9.5)

and its path velocity $v_t(t) = \dot{s}(t)$. From our experience, using the factor given by Equation (8.2) to map

$$v = v_t \frac{ds_{\boldsymbol{r}}}{ds}$$

works reasonably well in practice. In the equation above we use the resulting center of gravity of the vehicle in its neutral pose as the offset $\boldsymbol{r}$.

9.1 Inverse Kinematics

We will now consider a typical roller coaster train such as the one shown in Figure 9.1. The train consists of five coaches and a front axle, each connected to a set of wheel bogies by a series of rotational joints, resulting in a high-level kinematic structure similar to the one shown in Figure 6.1. The wheel bogies themselves consist of two running wheels, two side wheels and two upstop wheels, whereas all but the running wheels have a suspended connection to the respective bogie. The center of gravity of the individual coaches and the front axle are depicted in Figure 9.1, whereas we neglect both the inertia and the masses of all other rigid bodies of the train. Since in the context of roller coasters, we are also interested in evaluating the motion of the patrons sitting inside the train, we also depicted the standardized location for patron accelerometer measurements by small frames.

In order to accurately determine the motion of such a train, we set up the respective kinematic system and use standard inverse kinematics algorithms to determine the pose of each rigid body when the train is located at any given arc length s. Similarly to evaluating the track with respect to its arc length, we can store these poses as a dense and equidistant sampling for cheap lookups during further calculations. When implementing and evaluating this procedure, a few subtleties arise.

First, we neglect the dynamics of the wheel suspensions since we want to rely on kinematic analysis only. As a result, we evaluate the pose of the train in its steady state, with every suspension of each axle being equally engaged. Consider for example the configurations shown in Figure 9.2. In this example, the axles of the coach have a rigid connection to its main structure, such that the track gauge w cannot be observed by the axles when traversing curved sections of track. Without suspension of the side wheels, we would expect significant constraint forces on the wheel bogies. Considering the dynamics of the train, the amount of displacement of the suspensions is heavily dependent on the suspension characteristics, mass properties of the involved rigid bodies and the path velocity of the coach. In the case of purely kinematic analysis, we set up the pose of the coach in such a way that the sum of distances between the side wheels and the rails is minimized in the least squares sense.

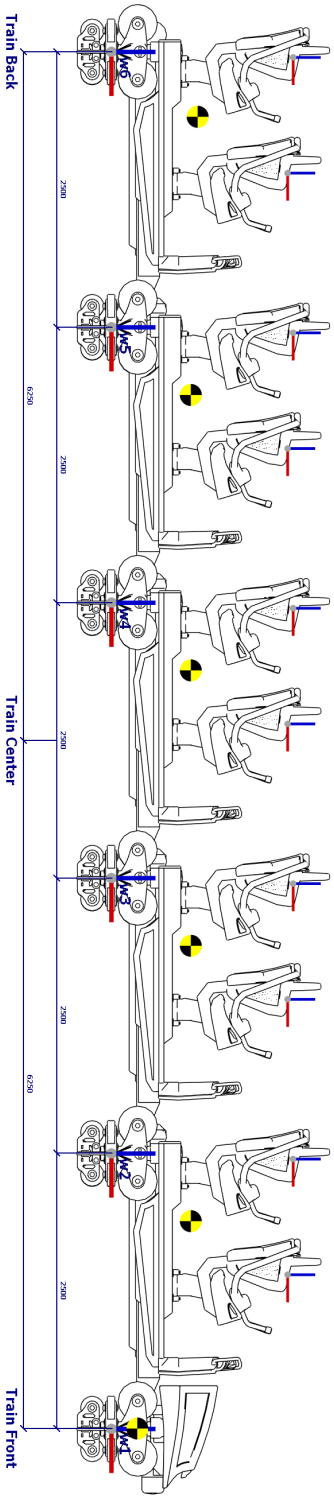


Figure 9.1: Side view of an exemplary train.

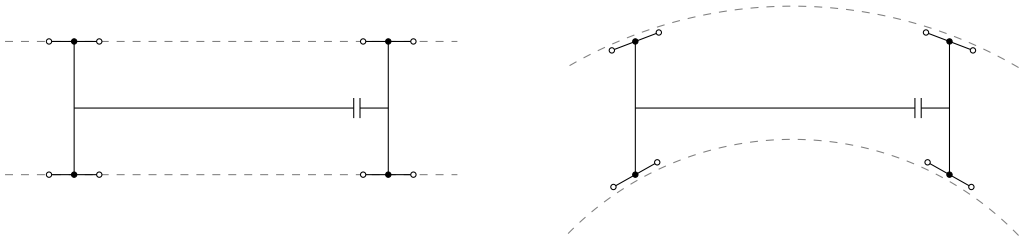


Figure 9.2: Coach pose in a straight section and a heavily curved section with constant track gauge w (top view).

Second, we need to clearly define the term „location s of the train along the track $\mathbf{T}$“. While this may be trivially easy to define for straight sections of track, a rigorous definition for general sections is needed in order to employ the algorithm above. Since our goal is to employ some numerical integration scheme to evaluate the motion of the train, it is crucial that we define the train's position s in a smooth manner, with an intuitive solution for the trivial case. In our current implementation, we use the following method.

1. For each wheel $\mathcal{W}_j, 0 \leq j < m$, calculate the tangential wheel displacement d_j in the neutral pose of the train, such that $\min(d_j) = -\max(d_j)$. This ensures that the train center will be the reference point for the position s for straight sections of track, even in the case of irregular wheel configurations.

2. The train position along the track is then

$$s = \sum \frac{s_j - d_j}{m},$$

   with s_j being the track's arc length of the contact point between $\mathcal{W}_j$ and the corresponding rail.

We can determine the poses $\mathbf{T}_i(s)$ and their derivatives by solving the respective inverse kinematics problem. Consequently, after computing the reparametrization $s(t)$ and using Equations (6.1) to (6.3), we can evaluate the velocities and accelerations of all bodies $\mathcal{B}_i$.

9.2 Simplified Kinematics

In order to further reduce the complexity of the calculations involved, we develop a simplified kinematic model. Its goal is to facilitate faster than real time evaluation of train motion, while still being as accurate as is necessary with respect to evaluating the pose and kinematics of the most important bodies, which in our application

are the train's coaches $\mathcal{C}_i, i = 0 \dots n-1$. In our simplified kinematic model, these are connected to the axles $\mathcal{A}_i$ and $\mathcal{A}_{i+1}$ to the front and back, respectively. Instead of evaluating the bogie kinematics and the contact of each wheel with the rails, we assume that the axles inherit their frame from the underlying track

$$\begin{aligned}\mathbf{T}_{\mathcal{A},i}(s) &= \mathbf{T}(s_i) \\ &= \begin{bmatrix} \boldsymbol{x}_{\mathcal{A},i} & \boldsymbol{y}_{\mathcal{A},i} & \boldsymbol{z}_{\mathcal{A},i} & \boldsymbol{p}_{\mathcal{A},i} \\ 0 & 0 & 0 & 1 \end{bmatrix}.\end{aligned}$$

This effectively prunes the kinematic tree of a train to the first level below each coach while merging the left and right bogies together to form the respective axle. The coaches $\mathcal{C}_i$ form the connection between two axles $\mathcal{A}_i$ and $\mathcal{A}_{i+1}$

$$\begin{aligned}\mathbf{T}_{\mathcal{C},i}(s) &= \begin{bmatrix} \boldsymbol{x}_{\mathcal{C},i} & \boldsymbol{y}_{\mathcal{C},i} & \boldsymbol{z}_{\mathcal{C},i} & \boldsymbol{p}_{\mathcal{C},i} \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \frac{\boldsymbol{p}_{\mathcal{A},i} - \boldsymbol{p}_{\mathcal{A},i+1}}{\left\|\boldsymbol{p}_{\mathcal{A},i} - \boldsymbol{p}_{\mathcal{A},i+1}\right\|} & \frac{\boldsymbol{z}_{\mathcal{A},i+1} \times \boldsymbol{x}_{\mathcal{C},i}}{\left\|\boldsymbol{z}_{\mathcal{A},i+1} \times \boldsymbol{x}_{\mathcal{C},i}\right\|} & \frac{\boldsymbol{x}_{\mathcal{C},i} \times \boldsymbol{y}_{\mathcal{C},i}}{\left\|\boldsymbol{x}_{\mathcal{C},i} \times \boldsymbol{y}_{\mathcal{C},i}\right\|} & \boldsymbol{p}_{\mathcal{A},i+1} \\ 0, & 0 & 0 & 1 \end{bmatrix},\end{aligned}$$

whereas we define $\boldsymbol{y}_{\mathcal{C},i}$ to lie in the x-y-plane of $\boldsymbol{T}_{\mathcal{A},i+1}$. Therefore, we may regard $\mathcal{A}_{i+1}$ as the main axle of $\mathcal{C}_i$ and $\mathcal{A}_i$ as its support axle. Similarly to above, we can denote the axle displacements d_j for the neutral train pose to define the train's position along the track. The axle positions s_i can be computed by iteratively solving

$$\sum_{i=0}^{n} \frac{s_i - d_i}{n+1} = s$$

subject to

$$s_i - s_{i+1} = \|\boldsymbol{p}_{\mathcal{A},i} - \boldsymbol{p}_{\mathcal{A},i+1}\|$$

starting at $s_i = s + d_i$, until all values converge.

A major benefit of this model is that we can use the known differential properties of $\mathbf{T}$ to compute the velocities and accelerations of all bodies. Computing the axle motion is straightforward using Equations (9.1) to (9.4) due to its assumed direct connection to the underlying track. Assuming we know the path velocity $v_{\mathcal{A},j} = \|\dot{\boldsymbol{p}}_{\mathcal{A},j}\|$ of all axles $\mathcal{A}_j$, we can further compute the linear and angular accelerations $\boldsymbol{a}_{\mathcal{C},i}$, $\dot{\boldsymbol{\omega}}_{\mathcal{C},i}$ and the angular velocities $\boldsymbol{\omega}_{\mathcal{C},i}$ of the coaches $\mathcal{C}_i$ as

$$\begin{aligned}\boldsymbol{a}_{\mathcal{C},i} &= \boldsymbol{a}_{\mathcal{A},i+1} \\ \boldsymbol{\omega}_{\mathcal{C},i} &= \dot{\boldsymbol{x}}_{\mathcal{C},i} \times \boldsymbol{x}_{\mathcal{C},i} + \langle \boldsymbol{\omega}_{\mathcal{A},i+1}, \boldsymbol{x}_{\mathcal{C},i} \rangle \, \boldsymbol{x}_{\mathcal{C},i} \\ \dot{\boldsymbol{\omega}}_{\mathcal{C},i} &= \ddot{\boldsymbol{x}}_{\mathcal{C},i} \times \boldsymbol{x}_{\mathcal{C},i} + \langle \dot{\boldsymbol{\omega}}_{\mathcal{A},i+1}, \boldsymbol{x}_{\mathcal{C},i} \rangle \, \boldsymbol{x}_{\mathcal{C},i} + \langle \boldsymbol{\omega}_{\mathcal{A},i+1}, \dot{\boldsymbol{x}}_{\mathcal{C},i} \rangle \, \boldsymbol{x}_{\mathcal{C},i} + \langle \boldsymbol{\omega}_{\mathcal{A},i+1}, \boldsymbol{x}_{\mathcal{C},i} \rangle \, \dot{\boldsymbol{x}}_{\mathcal{C},i}\end{aligned}$$

with

$$\dot{\boldsymbol{x}}_{\mathcal{C},i} = \frac{\dot{\boldsymbol{p}}_{\mathcal{A},i} - \dot{\boldsymbol{p}}_{\mathcal{A},i+1}}{\|\boldsymbol{p}_{\mathcal{A},i} - \boldsymbol{p}_{\mathcal{A},i+1}\|},$$
$$\ddot{\boldsymbol{x}}_{\mathcal{C},i} = \frac{\boldsymbol{a}_{\mathcal{A},i} - \boldsymbol{a}_{\mathcal{A},i+1}}{\|\boldsymbol{p}_{\mathcal{A},i} - \boldsymbol{p}_{\mathcal{A},i+1}\|}$$

and the velocities and accelerations of the axles being given by Equations (9.1) to (9.4). To evaluate the linear velocity and acceleration of a point with constant offset $\boldsymbol{r}$ relative to a coach $\mathcal{C}_i$, we can use the commonly known formulas for rigid bodies

$$\begin{aligned}
\boldsymbol{p}_r &= \boldsymbol{Q}\boldsymbol{r} + \boldsymbol{p} \\
\dot{\boldsymbol{p}}_r &= \dot{\boldsymbol{Q}}\boldsymbol{r} + \dot{\boldsymbol{p}} \\
&= \boldsymbol{\omega} \times \boldsymbol{Q}\boldsymbol{r} + \dot{\boldsymbol{p}} \\
\ddot{\boldsymbol{p}}_r &= \ddot{\boldsymbol{Q}}\boldsymbol{r} + \ddot{\boldsymbol{p}} \\
&= \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \boldsymbol{Q}\boldsymbol{r}) + \dot{\boldsymbol{\omega}} \times \boldsymbol{Q}\boldsymbol{r} + \ddot{\boldsymbol{p}},
\end{aligned}$$

where $\boldsymbol{Q}$ is the upper 3×3 matrix of $\mathbf{T}$. These formulas result from the Equations (6.1) to (6.3) in Section 6.1, and a more in-depth elaboration is given by, e.g., Jazar [46, Section 10.2]. Since the coaches are considered rigid bodies, their angular velocities and accelerations are independent of the reference point $\boldsymbol{r}$ on the body.

9.3 Evaluation

In the following, we compare the different methods to evaluate train motion introduced above with respect to both accuracy and computation time. In order to examine the sometimes subtle differences between the methods introduced in the previous section, we discuss an academic example first. The track shown in Figure 9.3 consists of four distinct sections of interest. It starts with a downhill slope terminated by constant curvature with G^1 contact on both ends, followed by two curves of constant curvature, the first of which having G^1 contact to the neighboring straight sections and the second using clothoids to ensure G^2 contact. The final section is a so-called *heartline roll* with a heartline offset of $h = 1$ and constant τ_r. To ensure sufficiently smooth contact, the relative torsion τ_r starts and finishes using a linear transition with respect to the heartline. The respective curvatures κ_g, κ_n and τ_r of the track center are shown in the lower half of Figure 9.3.

Note: Since τ_r is in C^0, the resulting framed curve with respect to the heartline is in G^2. Thus, both the track center and the rails as shown in Figure 9.3 are G^1 continuous, see Section 8.5.

Figures 9.4 to 9.7 show the calculated linear gravitoinertial accelerations of a virtual accelerometer which is positioned in the center between the two axles of

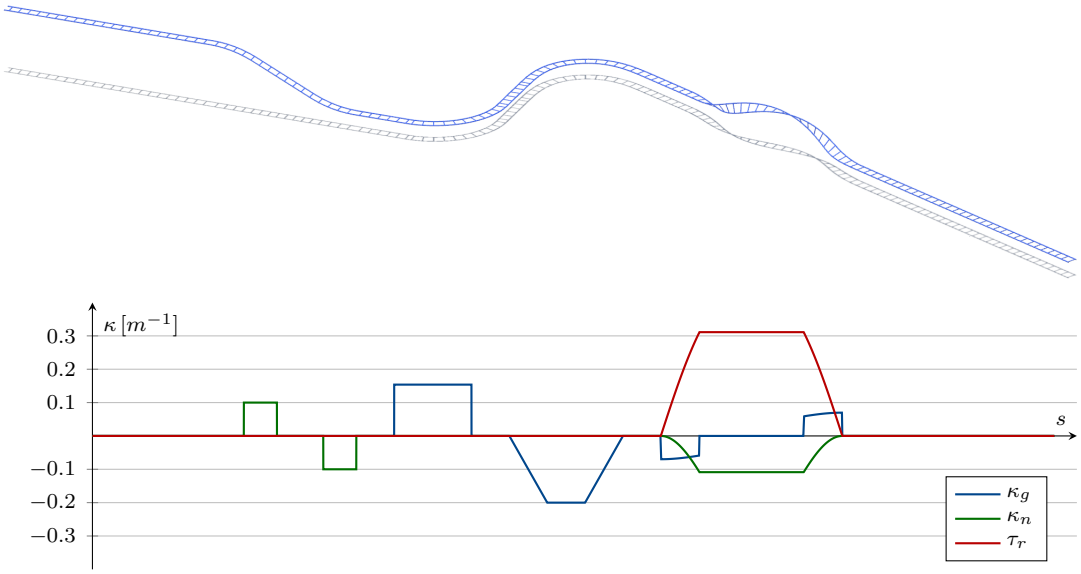


Figure 9.3: Track used in the first example comparing different kinetics algorithms.

the frontmost coach at height $z = 1.6$ above the track's center. We compare a naive approach with the virtual accelerometer being mounted to a body $\mathcal{B}$ with immediate connection to the track, the simplified kinematic model as introduced in Section 9.2 and a full inverse kinematics model as introduced in Section 9.1. The number of mobilities of the train alone, without the contact elements to the respective track, for each model is zero, ten and 109 respectively. The effect of the increasing level of detail regarding the train kinematics can already be seen in the acceleration profiles for the G^1 continuous sections *downhill slope* (Figure 9.4) and *left curve* (Figure 9.5). Since all information about the train kinematics is stripped away, the resulting accelerations of the naive model closely resemble the underlying curvature profile of the track, taking into account the path velocity v_t and acceleration $\dot{v}_t$. Looking at the green graphs of the simplified model, we can identify the sections during which each axle of the respective coach passes the individual curvature discontinuities. Since the virtual sensor is located at the center between two axles, the effect of the individual axles are effectively averaged out. This effect is even more predominant when looking at the blue graphs that show the results of the full inverse kinematics approach. Here, we can even distinguish between each individual wheel traversing the G^1 contact of the track, resulting in four distinct steps for a_z in Figure 9.4 and even six distinct steps for a_y in Figure 9.5. The two additional steps result from the fact that the left and right wheels of an axle traverse the curvature discontinuity at different points in time, an effect the simplified model cannot simulate.

Since we examine the motion of the frontmost coach of the train, we can observe that the resulting tangential accelerations during and after the downward slope in the lower half of Figure 9.4 are non-zero. This is due to the fact that the rest of the train prevents the coach from accelerating down the slope and pushes the coach into the following straight section and well into the left curve shown in Figure 9.5.

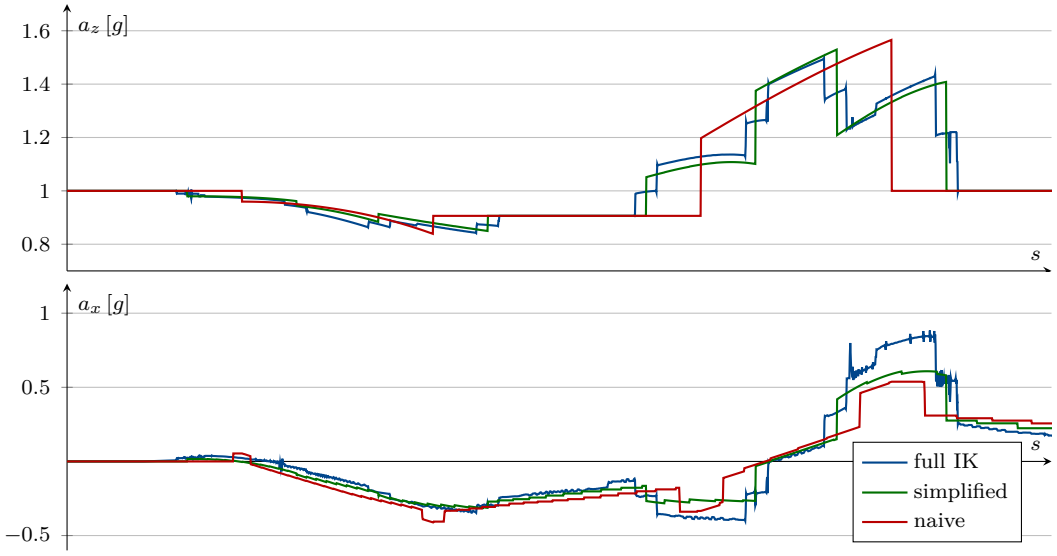


Figure 9.4: Calculated accelerations experienced in the downward slope section.

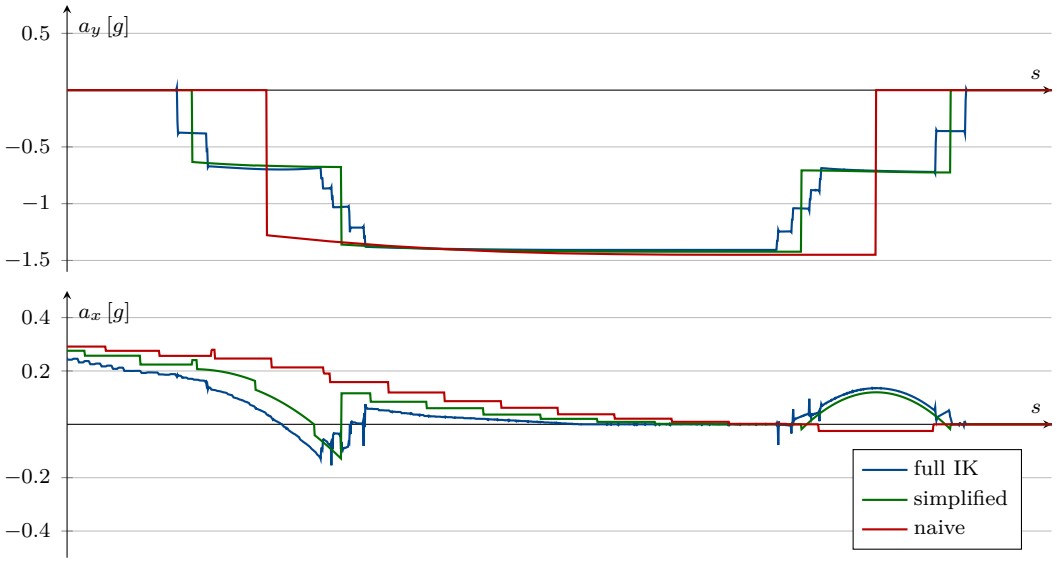


Figure 9.5: Calculated accelerations experienced in the left curve section.

This effect is noticable when riding roller coasters with particularly long trains in the far back or front, and it is the main reason for roller coaster enthusiasts to prefer these extreme seating positions.

Another effect with regards to tangential accelerations can be observed in the transitions to and from constant curvature in the lower half of Figure 9.5 and 9.6. The shown deviations from the naive acceleration profiles for the simplified and full IK model can be traced back to the centrifugal accelerations of the curve and the tilted pose of the coach with respect to its direction of travel. Since for the naive approach, the local x-axis of the coach coincides with the curve's tangent $\boldsymbol{t}$,

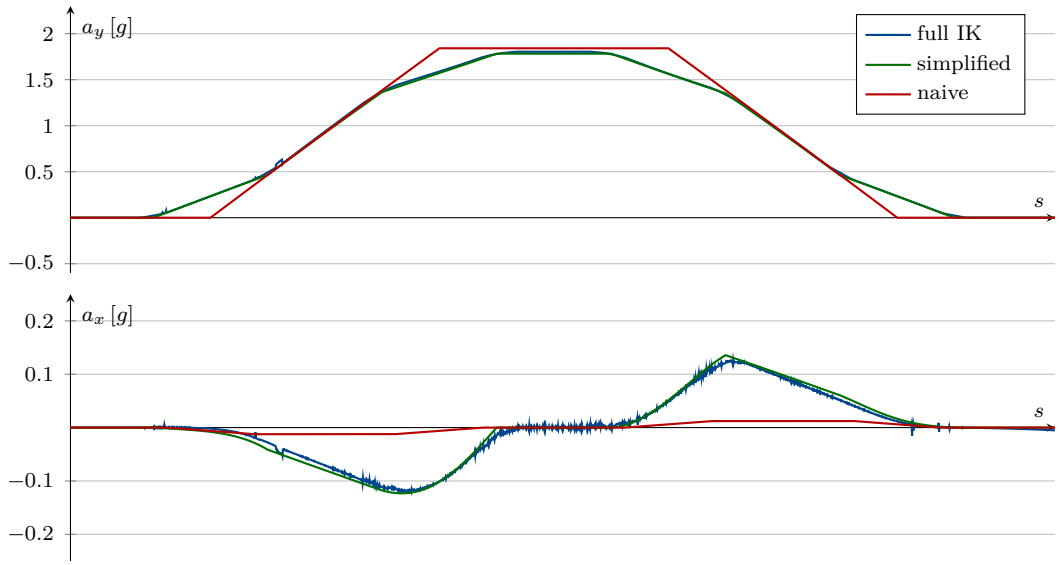


Figure 9.6: Calculated accelerations experienced in the right curve section.

no such effect can be seen there. As soon as we evaluate the motion in more detail, the direction of travel is decoupled from the local x-axis of the frame of reference.

Similar effects to the ones described for the G^1 left curve (Figure 9.5) can be observed for the G^2 right curve (Figure 9.6). Again, we can see the averaging of axle motion, as the simplified and full IK graphs resemble the result of a convolution of the naive graph with two off-center Dirac delta functions. However, for the G^2 continuous case these effects due to the underlying train kinematics are diminished. In fact, differences between the full IK and simplified model are now barely visible, with the simplified model yielding less noisy data since it does not rely as much on numerical differentiation as the full IK approach. As a rule of thumb, as the continuity of the underlying curve increases and the length of the respective curvature transitions becomes significantly longer than the features of the train kinematics, e.g., the distances between axles or wheels, dynamic effects due to train kinematics become smaller. On the other hand, train kinematics become the predominant source of dynamic effects for less continuous curves.

As soon as we look at the data of more complex maneuvers such as the *heartline roll* shown in Figure 9.7, the interpretation becomes increasingly difficult. Due to the heartline effect, the track's characteristics and thus the train dynamics are a result of the superposition of angular motion about all three axes and the direction of travel relative to the coach frame is constantly changing. Furthermore, since the curvatures, in particular those of the heartline itself, are relatively low, the accelerations depicted in Figure 9.7 are mostly due to gravity. Hence, the accuracy of the calculated coach pose mostly determines the accuracy of the calculated accelerations. This is why the difference between the results of all three models is largest in this case. As in the previous track sections, the simplified and full IK model show significant dynamic effects due to the individual axes or bogies

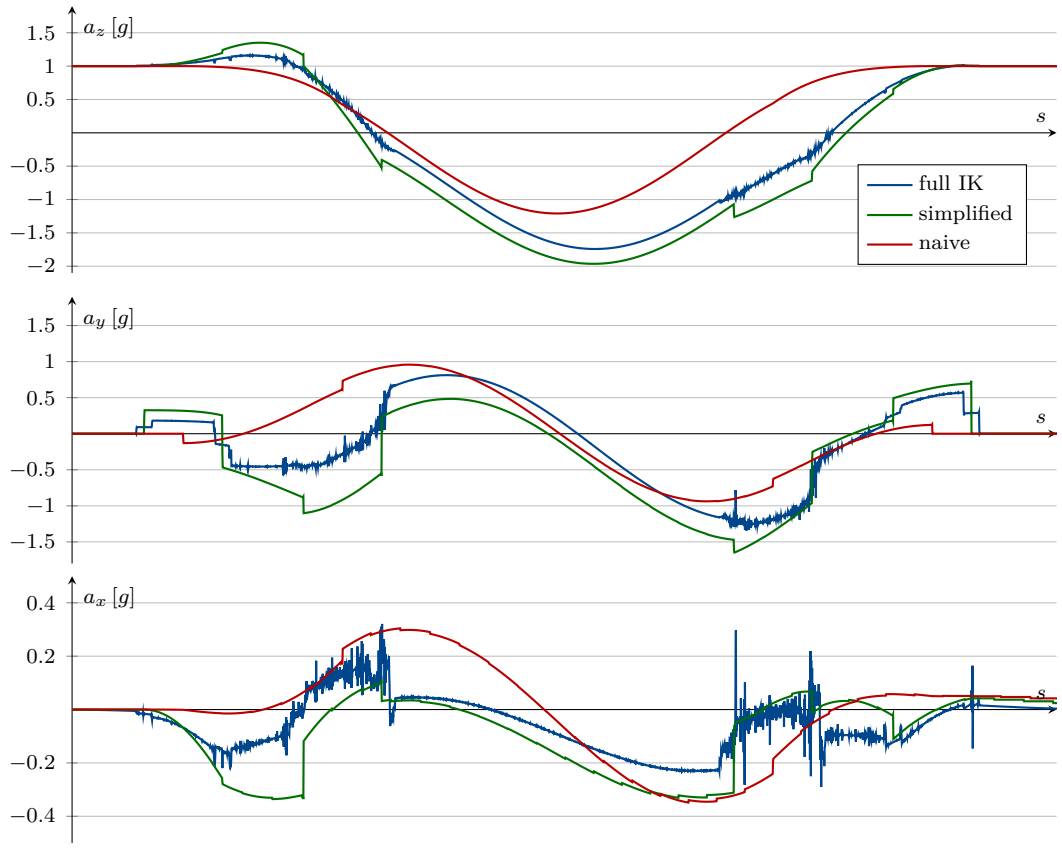


Figure 9.7: Calculated accelerations experienced in the heartline roll section.

entering and leaving the higher order discontinuities at different points in time. Since the simplified model assumes that the joints connecting axles and coaches are located at the track's centerline and additionally cannot account for the bogies of an axle being at different arc lengths of the track, sections of large relative torsion τ_r show the largest difference between the full IK model and the simplified model. We found this shortcoming of the simplified model to be neglegible in practice since τ_r is limited to avoid collisions between the involved bodies. That being said, adaptions to the simplified model could be made if necessary to account for tilting axes in the $\boldsymbol{t}$-$\boldsymbol{u}$-plane of the track and/or to account for the location of joints between axles and coaches.

Computing the full inverse kinematics of the train along the track in eight parallel threads using a custom implementation of a rail joint in the general purpose MBS library Simbody [84] with sufficient accuracy and resolution to yield the results shown in Figures 9.4 to 9.7 took 133.5s on an Intel Core i7-7700k which is clocked at 4.5 Ghz. On the same system, evaluating the simplified and naive model on a single thread took 16 ms and 12ms respectively.

As a second example, we consider the operational roller coaster shown in Figure 9.8. Due to its total height of 60 m and maximum velocity of 34 m/s, the resulting layout geometry is mostly constrained by the underlying vehicle dynamics, resulting

Figure 9.8: Operational roller coaster used as second example [33].

in comparably low curvatures and relative torsion. Consequently, we expect little influence of the underlying train kinematics on the resulting accelerations. Since our current implementation of the full IK approach relies on numerical differentiation to evaluate the gravitoinertial accelerations, the resulting data for such a layout is very noisy. Therefore in Figure 9.9, instead of using the results of the full IK approach as ground truth, we compare the naive and simplified model with accelerometer measurements that were performed during commissioning of the roller coaster. Additionally, we have plotted the accelerations computed by the method previously employed at the company Mack Rides, which solely takes the geometry of the heartline into account, in red. We can confirm, that the different methods yield quite similar results since the kinematic structure of the train has little effect on the resulting gravitoinertial accelerations. Even if we completely disregard the position of the virtual accelerometer and just use the geometric properties of the underlying heartline, the accelerations are predicted reasonably well throughout most sections. The largest difference between the different methods can be seen at $t = 58\,s$, the location of maximum relative torsion τ_r throughout the track. There, the angular velocity of the coaches induces inertial accelerations resulting in significantly different values of a_z, while the train kinematics has some additional impact on the measured a_y which can only be replicated when using the simplified method.

As opposed to the large roller coaster in the previous example, we chose a relatively small and tight roller coaster layout as our final example. It is shown in Figure 9.10 and features the same train kinematics as the previous example. However, since the

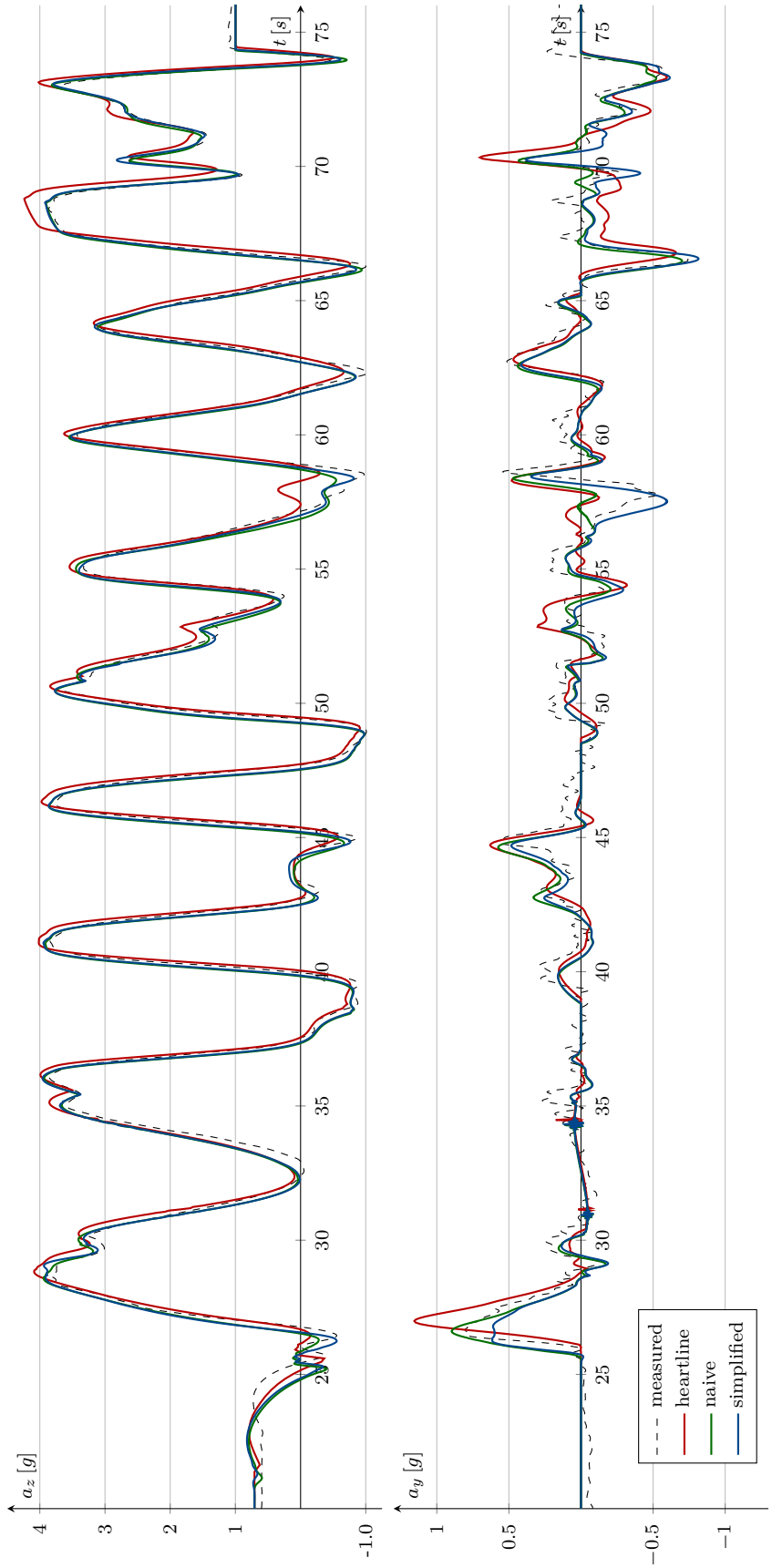


Figure 9.9: Measured and calculated accelerations for the roller coaster shown in Figure 9.8.

Figure 9.10: Operational roller coaster used in the final example [103].

geometric features of the underlying track are much more pronounced, we expect a larger influence of train kinematics and choice of reference point on the computed gravitoinertial accelerations. In Figure 9.11, we compare real-world measurements in dashed black with the three different simulated acceleration profiles, first using the method previously used at the company Mack Rides in red, second using offset curves but no kinematic model of the train in green, and finally using our simplified kinematic model of the train in blue as we did in the previous example. Here however, various sections along the layout show significant differences between the three models used with the most sophisticated and detailed model yielding the most accurate results.

9.4 Conclusions

The observations in the examples above lead to several conclusions with regard to how track layouts and their underlying framed curve should be designed. Since humans are particularly sensitive to accelerations and their respective jerks, and since the resulting dynamics of a track layout are mostly determined by the geometry of the underlying framed curve, we are interested in creating smooth and fair curvature profiles, especially in sections of large expected path velocity v_t. In order to limit the jerk experienced during traversal, variation in curvature needs to be kept in mind during the design process. That is why the baseline continuity for any track traversed with significant dynamic effects should be G^2. However, G^3 continuity is preferable in which case the jerk is not only limited but also continuous, which further improves passenger comfort. When using the heartline principle, the continuity of the heartline needs to be adapted accordingly.

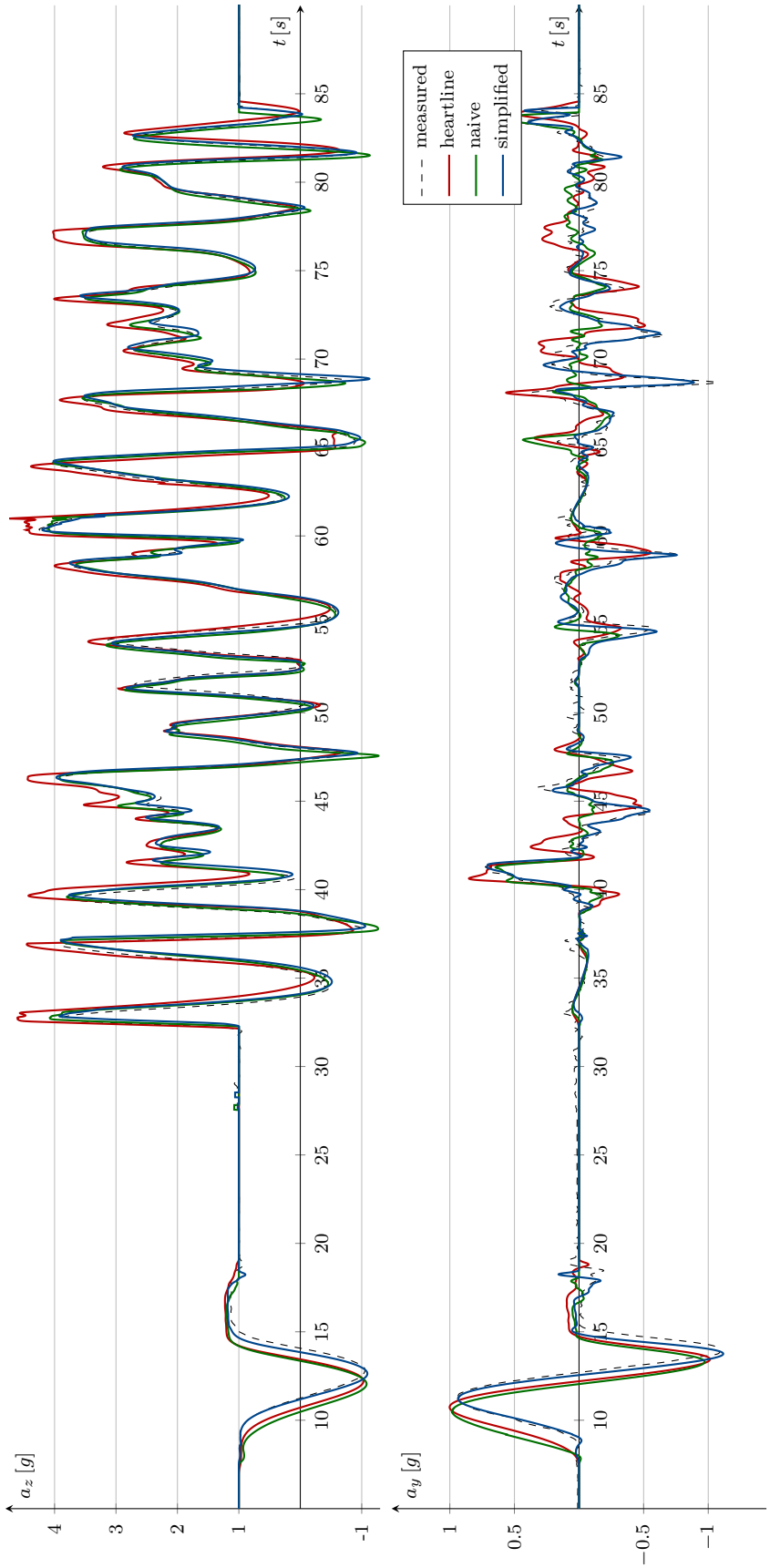


Figure 9.11: Measured and calculated accelerations for the roller coaster shown in Figure 9.10.

On the other hand, for sections that are traversed with low velocity, e.g., in and around the station and maintenance area, it is sufficient for a vehicle to be able to traverse the track without collisions since the dynamic effects due to the underlying track are negligible compared to the dynamic effects due to the respective coach kinematics. To save space or facilitate manufacturing of the track pieces we may even allow certain kinks in the track in these sections as long as their magnitude is low enough. This can be done because the coaches themselves will not strictly follow the curve $\mathbf{T}$ but rather smooth out and dampen the kink in the track due to their kinematic linkage all the way down to the dampened contact elements to the track. As we have seen, the lower the framed curve's order of continuity is, the more these features that depend on the train kinematics will come into effect.

Modeling Techniques for Framed Curves

Designing framed curves for track layout design is a classical example of a CAGD (computer aided geometric design) application. Although using existing standard CAGD software for the purpose of layout design is possible, we claim that developing and using special purpose software, which considers the specific requirements with respect to the track geometry and resulting dynamics, leads to a significant improvement on both track quality and efficiency in the design process. Due to the fuzzy nature of the design constraints, we believe that at the time of this writing, developing fully automated algorithms without human interaction is unfeasible. This is why, in the following chapters, we introduce and discuss various algorithms and techniques for interactive CAGD of framed curves, with acceptable response times to keep the designer engaged and productive. Additionally, we seek to reduce the mental load on the designer as much as possible. Instead of focussing on fine tuning of parameters to improve curve fairness, the designers should spend most of their time and effort expressing their design intent in an intuitive and terse manner. This can be achieved by keeping the dimension of the *design space*, i.e., the space of possible modifications a user can perform, as low as possible, mostly by exposing only the most meaningful design parameters to the designer and by making sure that the curve generation algorithms construct fair curves.

We start this part of the thesis by introducing a wholistic approach that we call *curvature based curve definition* in Chapter 10. Using this approach a designer can model non-linear framed splines by specifying the underlying curvature profile. As a consequence, the designer has full authority over the shape of the curve at the cost of limited positional control. In contrast to this, we explore interpolating polynomial splines in the context of interactive design of spatial tracks in Chapter 11. These splines give the designer full positional control over the generated curve, whereas curvature control is limited. In this chapter, we show how the basic polynomial spline interpolation as introduced in Section 5.8 can be adapted to accept purely geometric input. In Chapter 12, we further adapt our spline interpolation scheme to approximate *energy minimizing curves*, which explicitly optimize the fairness of their curvature profiles, with polynomial splines. Since some of the presented algorithms only generate spatial curves $\boldsymbol{p}$ as opposed to framed curves $\mathbf{T}$, we cover in Chapter 13 how fair moving frames $\boldsymbol{Q}$ can be generated interactively given some existing base curve $\boldsymbol{p}$.

10 Curvature Based Curve Definition

The curvature profile of a framed curve plays a central role in its fairness in the context of track layout design. In Section 5.9, we introduce (spatial) polynomial spirals as a type of curve with a by definition smooth curvature profile. Single polynomial spirals lack adaptability for use in our application due to the limited number of parameters that control their shape. However, splines consisting of several polynomial spiral segments are a promising candidate for use in interactive layout design. Similar to single polynomial spirals, we can approximate these splines by solving an initial value problem using numerical integration.

10.1 Curve Generation

As mentioned in Section 5.3, the curvature $\kappa(s)$ and torsion $\tau(s)$ uniquely define the purely geometric properties of a spatial curve $\boldsymbol{p}$ and its Frenet frame given by $\boldsymbol{t}$, $\boldsymbol{n}$ and $\boldsymbol{b}$ [24, Section 1-5; 44; 89, Section 1-8]. Moreover, they also uniquely define the framed curve

$$\mathbf{T} = \begin{bmatrix} \boldsymbol{t} & \boldsymbol{n} & \boldsymbol{b} & \boldsymbol{p} \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

For a given initial frame $\mathbf{T}(0)$, we can formulate the corresponding initial value problem

$$\mathbf{T}'(s) = \mathbf{T}(s) \begin{bmatrix} 0 & -\kappa(s) & 0 & 1 \\ \kappa(s) & 0 & -\tau(s) & 0 \\ 0 & \tau(s) & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{10.1}$$

that defines the framed curve. Since for general $\kappa(s)$ and $\tau(s)$ the above problem has no closed-form solution, we need to approximate the curve $\mathbf{T}(s)$, e.g., by computing a sufficiently dense sampling using some numerical integration scheme. During numerical integration, special care has to be taken to keep the moving frame orthonormal at all times. A naive implementation evaluates

$$\mathbf{T}(s) + \mathbf{T}'(s) \cdot \Delta s \tag{10.2}$$

in each integration step, failing to preserve orthonormality of the frame. This would not be feasible without additionally applying some orthogonalization scheme in each step. Alternatively, we can use the Darboux vector $\boldsymbol{d}$ and calculate

$$\begin{aligned} \boldsymbol{Q}(s + \Delta s) &= \boldsymbol{Q}(s) \cdot \operatorname{Rot}\left(\frac{\boldsymbol{Q}(s)^{\mathsf{T}} \boldsymbol{d}(s)}{\|\boldsymbol{d}(s)\|}, \|\boldsymbol{d}(s)\| \, \Delta s \right), \\ \boldsymbol{p}(s + \Delta s) &= \boldsymbol{p}(s) + \Delta s \frac{\boldsymbol{t}(s) + \boldsymbol{t}(s + \Delta s)}{2}, \end{aligned} \tag{10.3}$$

with

$$\boldsymbol{Q}(s)^{\mathsf{T}}\boldsymbol{d}(s) = \begin{bmatrix} \tau(s) \\ 0 \\ \kappa(s) \end{bmatrix},$$

which preserves the orthonormality of $\boldsymbol{Q}(s)$.

Remark: The choice of step size Δs affects the result of the numerical integration method similarly to the choice of the orthogonalization scheme when using Equation (10.2). However, as the step size shrinks, so will the error introduced by the numerical integration. In order to further reduce the integration error, more sophisticated numerical integration methods such as the *Runge Kutta Method* may be used [80, Section 8.2; 88, Section 2.4].

Solving the initial value problem (10.1) to construct $\boldsymbol{p}$ requires the evaluation of the associated moving frame $\boldsymbol{Q}$. Since our goal is to construct framed curves $\mathbf{T}$, as opposed to unframed spatial curves $\boldsymbol{p}$, it seems natural to just use the generated curve including its moving frame as opposed to disregarding $\boldsymbol{Q}$ after computation. However, although every sufficiently smooth curve $\boldsymbol{p}$ can be constructed using the functions $\kappa(s)$ and $\tau(s)$, we cannot construct general framed curves $\mathbf{T}$ in this manner, which would require us to use all three generalized curvatures. Using the curvatures $\kappa_g(s)$, $\kappa_n(s)$ and $\tau_r(s)$, we obtain the initial value problem

$$\mathbf{T}(s)' = \mathbf{T}(s) \begin{bmatrix} 0 & -\kappa_g(s) & \kappa_n(s) & 1 \\ \kappa_g(s) & 0 & -\tau_r(s) & 0 \\ -\kappa_n(s) & \tau_r(s) & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{10.4}$$

which can be solved analogously to above. Using this modeling technique of a framed curve turns out to be very powerful in our context since the respective curvatures have immediate geometric meaning and play a major role in the resulting dynamics when traversing a track.

A CAGD system using the curvature based curve definition needs to allow the user to model the input functions in some meaningful manner while supplying timely visual feedback on the resulting curve. Since this type of modeling is indirect, designers cannot verify whether the resulting curve satisfies any positional constraints just based on their inputs and without visual feedback. Additionally, since we rely on integration, any changes to the input functions at some parameter s_0 affects the whole output curve for all $s \geq s_0$. This non-localness is a key disadvantage of this modeling technique and the reason why we encourage designers to use it for limited sections of track only.

10.2 Extensions

Using the curve generation method above, we present two possible extensions that can help a designer in specific settings. First, we can replicate the classical layout design scheme introduced in Section 1.1. In this scheme, the curvatures $\hat{\kappa}_h$ of the layout plan, $\hat{\kappa}_v$ of the height profile and the roll angle γ are of main interest to the designer and shall determine the final shape of the track. We can compute the curvatures of the resulting framed curve based on these input functions using the relationship

$$\begin{bmatrix} \kappa_g \\ \kappa_n \\ \tau_r \end{bmatrix} = \begin{bmatrix} \cos(\gamma)\cos^2(\alpha) & \sin(\gamma) & 0 \\ \sin(\gamma)\cos^2(\alpha) & -\cos(\gamma) & 0 \\ -\sin(\alpha)\cos(\alpha) & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{\kappa}_h \\ \hat{\kappa}_v \\ \gamma' \end{bmatrix} \tag{10.5}$$

where

$$\alpha = -\arcsin(\langle \boldsymbol{t}, \boldsymbol{e}_3 \rangle)$$

is the pitch angle of the track. These curvatures can then be used as input for the curve generation algorithm from Section 10.1. For the classical layouting scheme, curves are commonly parameterized with respect to their projected arc length $\bar{s}$ in the layout plan. Thus letting the designers use the input functions $\hat{\kappa}_h(\bar{s})$, $\hat{\kappa}_v(\bar{s})$ and $\gamma(\bar{s})$ with subsequent evaluation of (10.5) and (10.4) is preferred over the spatial arc length counterparts $\hat{\kappa}_h(s)$, $\hat{\kappa}_v(s)$ and $\gamma(s)$.

Second, we combine the idea of *curves of constant acceleration* introduced in Section 7.6 with the modeling techniques presented in Section 10.1. For sections of roller coaster track that are constrained by the gravitoinertial accelerations acting on the patrons while traversing the track $\mathbf{T}(s)$ with varying path velocities $v_t(s)$, we can determine the shape of the track by specifying the intended acceleration profile. Assuming a point mass m with some total energy E_{tot} and no friction, the corresponding path velocity of said point mass is

$$v = \sqrt{2\left(\frac{E_{tot}}{m} - g\,\langle \boldsymbol{p}, \boldsymbol{e}_3 \rangle\right)}.$$

Given some intended acceleration a_z along the curve's normal $\boldsymbol{v}$ and acceleration a_y along the curve's normal $\boldsymbol{u}$ the resulting gravitoinertial acceleration vector

$$\begin{aligned} \boldsymbol{a} &= a_z\boldsymbol{v} + a_y\boldsymbol{u} - g\boldsymbol{e}_3 \\ &= v^2\left(\kappa_n\boldsymbol{v} + \kappa_g\boldsymbol{u}\right) - g\boldsymbol{e}_3 \end{aligned}$$

determines the curvatures of the underlying framed curve

$$\begin{aligned} \kappa_g &= \frac{a_y - g\,\langle \boldsymbol{e}_3, \boldsymbol{u} \rangle}{v^2} \\ \kappa_n &= \frac{a_z - g\,\langle \boldsymbol{e}_3, \boldsymbol{v} \rangle}{v^2}. \end{aligned}$$

	function	transition curve
linear	$g(s) = s$	clothoid, Euler-Spiral [47]
cubic	$g(s) = s^2(3 - 2s)$	Bloss curve [11]
quintic	$g(s) = s^3\,(10 + s(6s - 15))$	suggested by Watorek [104]
cosine	$g(s) = 0.5 - 0.5\cos(\pi s)$	suggested by Vojacek [96]
sine	$g(s) = s - \sin(2\pi s)\,/\,(2\pi)$	suggested by Klein [50]

Figure 10.1: Several transition curves and their curvature functions $g(s)$, loosely adapted from [79, Figure 4.5].

To our knowledge, Rill [73] was the first to consider constructing spatial tracks this way, albeit his method did not account for rolling motion induced by relative torsion τ_r. This motion can be modeled, e.g., by additionally specifying the angular velocity ω_x about the curve's tangent $\boldsymbol{t}$, resulting in

$$\tau_r = \frac{\omega_x}{v}.$$

Again, we can evaluate the respective framed curve by using these curvatures as input for the curve generation algorithm from Section 10.1.

Modeling spatial tracks for roller coasters in this way is very powerful and expressive since the designer can directly model what a human patron sitting inside the roller coaster train feels when traversing the respective track. In fact, several tools in the hobbyist community surrounding roller coaster design have been developed using this exact modeling technique, including the open source software FVD^{++} [86], which the author of this thesis developed well before starting a professional career and research in this particular topic.

10.3 Design Techniques

Implementing a curve generation algorithm for the curve definition schemes above is a rather straightforward process. Due to the advances in computing hardware, numerical integration of the corresponding systems of differential equations with sufficient accuracy is possible with interactive response times [86]. A CAGD system that implements the curvature based curve definition lets the designer specify the respective input functions. We choose

$$\kappa_j(s) = \sum_{i=0}^{n} f_{j,i}(s), j = 1, \ldots, 3,$$

where

$$f_{j,i}(s) := \begin{cases} 0, & s < m_{j,i} \\ k_{j,i}\, g(\frac{s - m_{j,i}}{l_{j,i}}), & m_{j,i} \le s \le m_{j,i} + l_{j,i} \\ k_{j,i}\, g(1), & m_{j,i} + l_{j,i} < s \end{cases}.$$

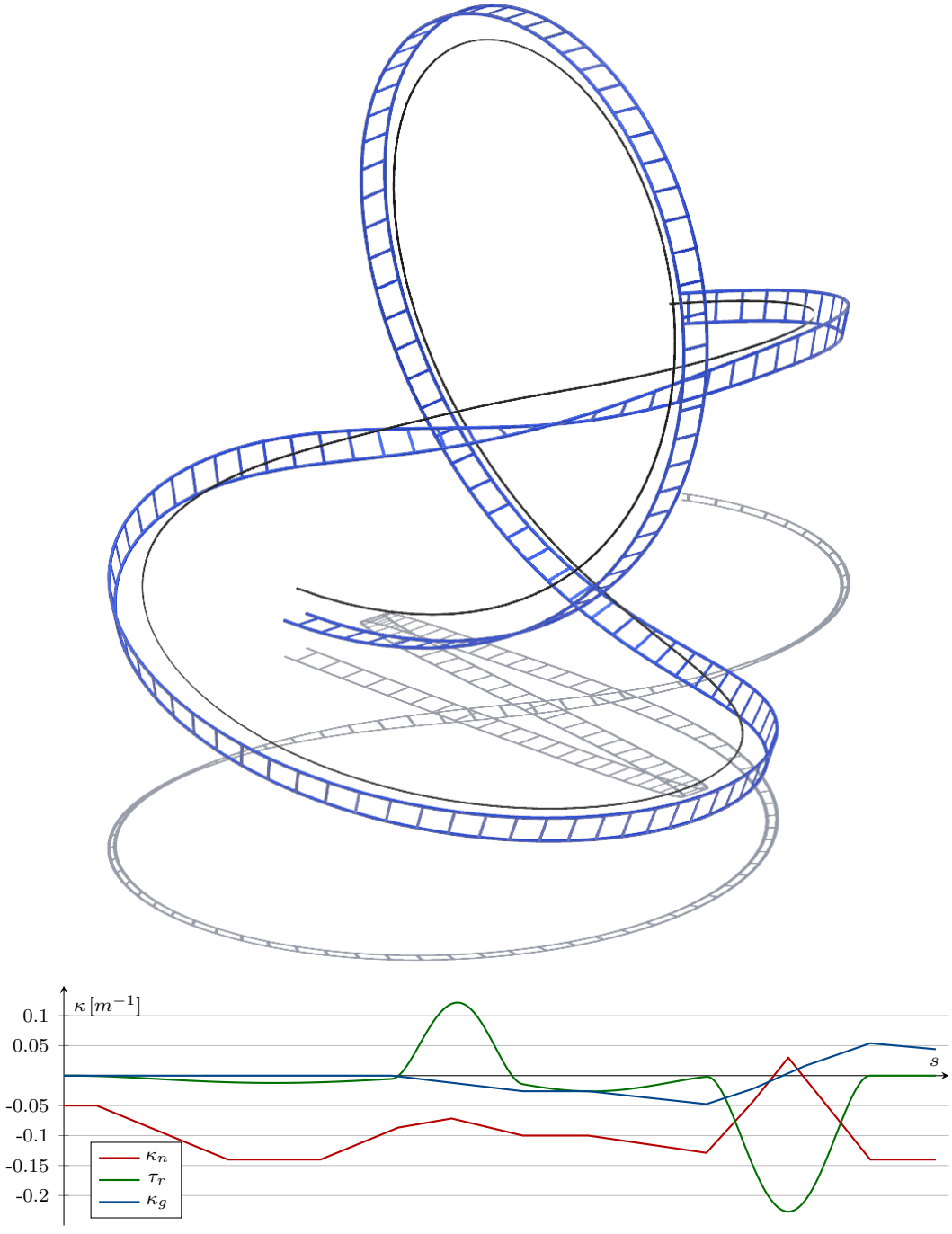


Figure 10.2: The vertical looping threaded by an S-curve shown on top is defined as a piecewise quartic polynomial spiral spline with the curvature profile shown on the bottom.

Several functions g that correspond to so-called *transition curves* [79, Chapter 4] in classical layout design are given in Figure 10.1. A designer may shift and scale these functions as deemed appropriate using the parameters $k_{j,i}$, $l_{j,i}$ and $m_{j,i}$. Our CAGD system lets the designer choose these transitions and their parameters for each input function alongside the initial frame $\mathbf{T}(0)$, giving designers enough descriptive power to achieve their creative intent.

A framed curve that is constructed using linear transitions for κ_n and κ_g and quartic transitions for τ_r can be seen in Figure 10.2. By limiting the input parameters and possible input functions, framed curves designed this way are in some sense *fair by construction*. This comes at the cost of manual fine tuning of the input parameters to achieve positional constraints such as collision avoidance between sections of track. Our implementation is able to provide visual feedback on the resulting framed curve within interactive response times after any modification to the designer's input, allowing early detection of arising conflicts.

The exemplary framed curve shown in Figure 10.3 demonstrates the advantage of using gravitoinertial accelerations as opposed to curvatures as input functions. The figure shows a roller coaster element that traces out a zero-g parabola superposed with a 360° rolling maneuver. The underlying framed curve was created using two cubic transitions for a_z to and from the zero-g parabola and a single quartic transition for ω_x to perform the rolling maneuver. The superposition of both motions results in a non-trivial curvature profile which would require significantly more parameters to construct using curvatures as inputs. Allowing designers to specify precisely their intent, in this case to create a moment of weightlessness during a rolling motion, results in both a terse curve definition and high quality resulting framed curve.

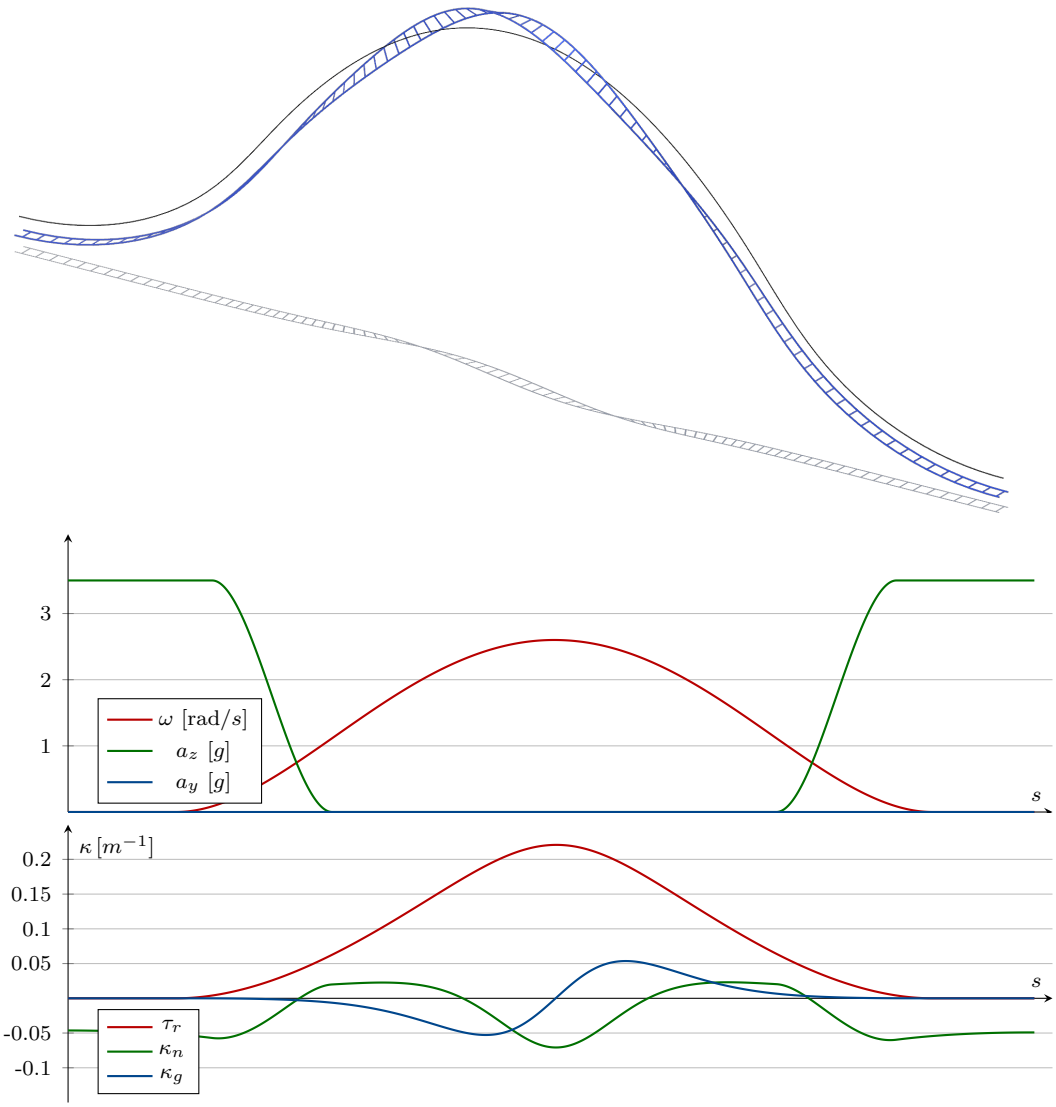


Figure 10.3: A zero-g roll (top) defined by its gravitoinertial acceleration profile (center). The resulting curvatures of the underlying framed curve are shown in the bottommost plots.

11 Polynomial Spline Interpolation

In Chapter 5, we introduced polynomial splines as a type of curve commonly used by CAGD systems. Polynomial splines have a multitude of degrees of freedom, with both the knots a_j and the control points $\boldsymbol{c}_i$ being free to be chosen. A common approach in CAGD systems is to hide the choice of the underlying knot vector since it is unintuitive and since desired shapes can usually be achieved by solely choosing a suitable control polygon [70, Section 9.2].

By choosing control points, designers give a recipe on how to construct a polynomial spline. As opposed to expressing *what* they want to achieve, they are required to specify *how* to generate the intended result. Additionally, it is rather difficult for designers to determine whether or not a polynomial spline will satisfy some positional constraint purely based on its control polygon and without seeing the resulting curve. Consider the polynomial spline shown in Figure 11.1, which has been constructed using two different schemes. In the top of Figure 11.1, the user defined the control polygon of the spline consisting of five control points whereas on the bottom points and derivatives are interpolated. When solving the corresponding polynomial spline interpolation problem, see Section 5.8, these constraints result in the control polygon shown on the top, but they carry the information of which particular constraints were defined by a user, i.e., what the design intent of said user is. In this example, the user specified three points to be interpolated, of which two needed to be traversed in some specific direction, as opposed to, e.g., interpolating five different points with no particular direction.

11.1 Applying Geometric Constraints

In our setting, the designer should not be forced to consider the curve as some parametric function, but rather as a geometric object with purely geometric properties. The user should think of points, tangent vectors, curvatures and curvature variation to be interpolated by some curve. The fact that the curve is represented as a parametric function with respect to some opaque parameter should be of no concern to the user. The goal of the underlying curve generation algorithm is to find the best curve – including its parametrization – that satisfies the given constraints. Keeping this in mind, specifying purely geometric interpolation constraints already reduces the dimension of the design space for the example in Figure 11.1, since the user only specifies the tangent vector, but not the parametric speed, as in the length of the first parametric derivative, at the respective points.

Suppose there is a way of determining the nodes u_i of all interpolation points $\boldsymbol{x}_i, 0 \leq i \leq n$. Apart from these points $\boldsymbol{x}_i$, the user may optionally define the corresponding tangent vectors $\boldsymbol{t}_i$, curvature vectors $\boldsymbol{\kappa}_i$ and curvature variation $\boldsymbol{\kappa}_i'$. In the following, we will think of

$$\boldsymbol{f} = \begin{bmatrix} \boldsymbol{x}_0 & \boldsymbol{t}_0 & \boldsymbol{\kappa}_0 & \dots & \boldsymbol{x}_n & \boldsymbol{t}_n & \boldsymbol{\kappa}_n \end{bmatrix} \in \mathbb{R}^{3\times m}$$

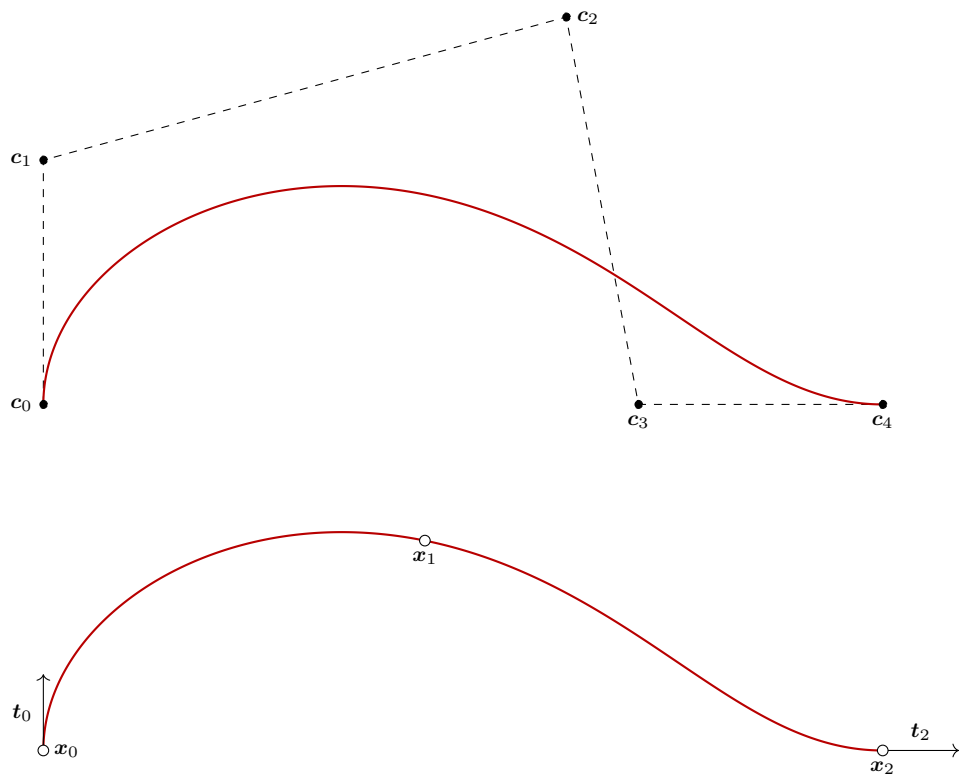


Figure 11.1: Polynomial spline with its control polygon (top) and the same polynomial spline and its interpolation constraints (bottom).

as an element of the *design space*. We want the user to only specify these values and be presented with the most fair curve $\boldsymbol{p}(u)$ satisfying these geometric constraints. As a first step, the geometric derivatives need to be transformed to derivatives with respect to the parameter u. This is achieved by introducing scaling factors σ_i, σ_i', σ_i'' such that

$$\begin{aligned}
\frac{d\boldsymbol{p}(u)}{du_i} &= \sigma_i \boldsymbol{t}_i, \\
\frac{d^2\boldsymbol{p}(u)}{du_i^2} &= \sigma_i' \boldsymbol{t}_i + \sigma_i^2 \boldsymbol{\kappa}_i, \\
\frac{d^3\boldsymbol{p}(u)}{du_i^3} &= \sigma_i'' \boldsymbol{t}_i + 3\sigma_i \sigma_i' \boldsymbol{\kappa}_i + \sigma_i^3 \boldsymbol{\kappa}_i'.
\end{aligned}$$

Note: Since higher order derivatives depend on the ones of lower order, we need to require that for each of the interpolation points $\boldsymbol{x}_i$, all geometric derivatives up to the considered order are defined. Furthermore, we can think of σ_i, σ_i' and σ_i'' as additional shape parameters of $\boldsymbol{p}$, which may be adapted by the curve generation algorithm to improve the curve's fairness.

In case the parametrization of the spline $\boldsymbol{p}$ approaches arc length parametrization, the values for σ_i approach 1 and σ_i', σ_i'' vanish. Furthermore, since of all curves interpolating a sequence of points $\boldsymbol{x}_i$ at given parameters u_i, the degree d polynomial spline interpolant minimizes the integral

$$\int \left(\frac{d^{d-1}\boldsymbol{p}(u)}{du^{d-1}} \right)^2 du, \tag{11.1}$$

and since for arc length parametrized curves, derivatives with respect to s and u are equal, a cubic spline approaching arc length parametrization would approach an elastica, as would a quartic spline approach an MVC, which are both introduced in Section 5.10. A proof of the above statement is given, e.g., by Ahlberg [2, Theorem 3.1.1] or more recently Farouki [27, Theorem 14.1] for $d = 3$, with the extension to $d > 3$ being straightforward.

11.2 Parametrization

Over the last decades, there have been several contributions to find suitable parametrization schemes for polynomial spline interpolation in the context of various applications. In general, this problem is about how to choose

1. the nodes u_i for which the spline satisfies $\boldsymbol{p}(u_i) = \boldsymbol{x}_i$, $\boldsymbol{p}'(u_i) = \boldsymbol{x}_i'$, ..., and

2. the knots a_j which determine the boundaries between the polynomial curve segments of the spline.

Both subproblems are highly interdependent but of different nature. Given $m = 5$ points, three first order derivatives and one second order derivative that are to be interpolated by a cubic spline as shown in Figure 11.5, a total of five nodes $u_0, \ldots, u_4$ need to be determined. However, to get a unique solution that satisfies all interpolation constraints, the polynomial spline interpolant of degree $d = 3$ requires $n = 9$ control points $\boldsymbol{c}_0, \ldots, \boldsymbol{c}_8$ and thus $n + d + 1 = 13$ knots $v_0, \ldots, v_{12}$. If we change the interpolation problem at hand by either removing or adding a derivative constraint, or by changing the degree d, a new knot vector of different size has to be generated while the number of nodes stays unchanged.

Piegl and Tiller [70, Section 9.2.1] propose a quite flexible and powerful method for knot selection. They determine the n scalars w_j as the nodes u_i used to calculate the j^{th} row of the matrix $\boldsymbol{N}$, see Section 5.8. Then, they average the values w_j to obtain the knots

$$a_j = \begin{cases} u_0 & , \quad 0 \leq j \leq d \\ \frac{1}{d} \sum_{i=j-d}^{j-1} w_i & , \quad d+1 \leq j \leq n \\ u_m & , \quad n+1 \leq j \leq n+d+1. \end{cases}$$

Since we can only specify derivative constraints up to order $d-1$ resulting in up to d consecutive w_j being equal, this averaging avoids knots with multiplicity and as a consequence, reduced degree of continuity of the resulting interpolant. Therefore, this so-called *averaging method* can be applied to interpolation problems that include derivative constraints, using splines of arbitrary degree d for any choice of the nodes u_i. We will use this knot selection scheme for the remainder of this chapter.

With the knot selection scheme fixed, the choice of nodes u_i given the interpolation constraints $\boldsymbol{f}$ remains to control the parametrization of an interpolating polynomial spline. The simplest parametrization scheme is the *equidistant parametrization*

$$u_i = c \cdot i$$

for some constant c. This scheme is completely independent of the underlying interpolation points $\boldsymbol{x}_i$. As a result, the interpolating spline is invariant to affine transformations $\mathbf{A}$. In other words, determining the interpolation spline $\hat{\mathbf{p}}$ of the transformed interpolation constraints $\hat{\mathbf{x}}_i = \mathbf{A}\mathbf{x}_i$ results in the transformed spline $\mathbf{A}\mathbf{p}$. However, the equidistant parametrization usually performs poorly when the points $\boldsymbol{x}_i$ are spaced unevenly. In these cases, the resulting spline often exhibits unwanted loops and thus lacks fairness [54].

As a simple extension to the *equidistant parametrization*, we can increment the parameter proportionally to the Euclidean distance between two points $\boldsymbol{x}_i$, which leads to the *chord length parametrization* [26, Section 7.7]

$$\begin{aligned} u_0 &= 0, \\ u_{i+1} &= u_i + \|\boldsymbol{x}_{i+1} - \boldsymbol{x}_i\| . \end{aligned} \tag{11.2}$$

Since the resulting parameters depend on the shape of the underlying data, this scheme does not produce affine invariant splines, since non-uniform scaling or shear transformations affect the relative magnitude of the Euclidean distance. However, interpolating splines resulting from this method interpolate the points $\boldsymbol{x}_i$ much more naturally, while staying invariant with respect to rigid body transformations and uniform scaling.

Several more involved methods have been proposed. In our context, the work of Foley and Nielson [30; 65; 66] is of particular interest. Their work focusses on affinely invariant parametrization methods. Instead of using the Euclidean distance, Nielson constructs a matrix $\boldsymbol{Q}$ from the underlying data set $\boldsymbol{x}_i$ such that

$$M(\boldsymbol{y}, \boldsymbol{z}) = \sqrt{(\boldsymbol{y} - \boldsymbol{z})\, \boldsymbol{Q}\, (\boldsymbol{y} - \boldsymbol{z})}$$

is affinely invariant for any two points $\boldsymbol{y}$ and $\boldsymbol{z}$ in the data set. Applying the *chord length parametrization* with respect to this *Nielson metric* M leads to affine invariance of the resulting spline [65]. Foley and Nielson also introduce the so-called

angle method [30], which takes the turning angles θ_i between three consecutive interpolation points $\boldsymbol{x}_{i-1}, \boldsymbol{x}_i, \boldsymbol{x}_{i+1}$ into account. They choose the nodes as

$$\begin{aligned} u_0 &= 0, \\ u_{i+1} &= u_i + d_i \left(1 + c\frac{\Theta_i d_{i-1}}{d_{i-1} + d_i} + c\frac{\Theta_{i+1} d_{i+1}}{d_i + d_{i+1}}\right) \end{aligned}$$

with

$$\begin{aligned} d_i &= \|\Delta \boldsymbol{x_i}\| \\ &= \|\boldsymbol{x}_{i+1} - \boldsymbol{x}_i\|, \\ \Theta_i &= \min(\theta_i, \theta_{\max}), \\ \theta_i &= \arccos\left(\left\langle \frac{\Delta \boldsymbol{x_i}}{\|\Delta \boldsymbol{x_i}\|}, \frac{\Delta \boldsymbol{x_{i-1}}}{\|\Delta \boldsymbol{x_{i-1}}\|} \right\rangle\right) \end{aligned} \tag{11.3}$$

and empirically chosen constants $c = 1.5$ and $\theta_{\max} = \pi/2$. Furthermore, when calculating the angles θ_i and distances d_i with respect to the *Nielson metric M* above, the *angle method* can also be made affinely invariant [30].

The *angle method* was specifically designed to solve problems with the *chord length* method when dealing with poorly scaled and scattered data by keeping a tighter fit at the cost of highly varying curvatures. Due to the relatively large constant factor c, the node differences no longer approximate the expected arc length of $\boldsymbol{p}$ between two interpolation points, leading to poor approximations when choosing derivative constraints with the natural choice of the shape parameter $\sigma_i \approx 1$. We found a much lower constant of $c \approx 0.2$ to work reasonably well, increasing the node difference for chords with expected high curvature and thus finding a better approximation of the arc length between successive interpolation points $\boldsymbol{x}_i$.

11.3 Curvature Parametrization Method

We adapt the idea of the *angle method* to derive a novel parametrization scheme, which we call *curvature method*. Like the *angle method*, it is an extension of the *chord length parametrization*, which can be seen as a first order approximation of the curve's arc length given the interpolation constraints. A second order approximation does not only consider the chord $\boldsymbol{x}_i, \boldsymbol{x}_{i+1}$ to determine the node u_{i+1}, but also takes the turning angles θ_i of three consecutive interpolation points into account. Consider the two circles $\boldsymbol{x}_{i-1}, \boldsymbol{x}_i, \boldsymbol{x}_{i+1}$ and $\boldsymbol{x}_i, \boldsymbol{x}_{i+1}, \boldsymbol{x}_{i+2}$ depicted in Figure 11.2. We expect an interpolant $\boldsymbol{p}$ between $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$ to have a shape similar to these two circles. In other words, the expected curvature of $\boldsymbol{p}$ in that span is the average curvature of the two circles. Thus, given the turning angles θ_i from Equation (11.3) above, we can calculate the expected curvatures

$$\kappa_i = 2\frac{\sin(\theta_i)}{\|\boldsymbol{x}_{i+1} - \boldsymbol{x}_{i-1}\|} \tag{11.4}$$

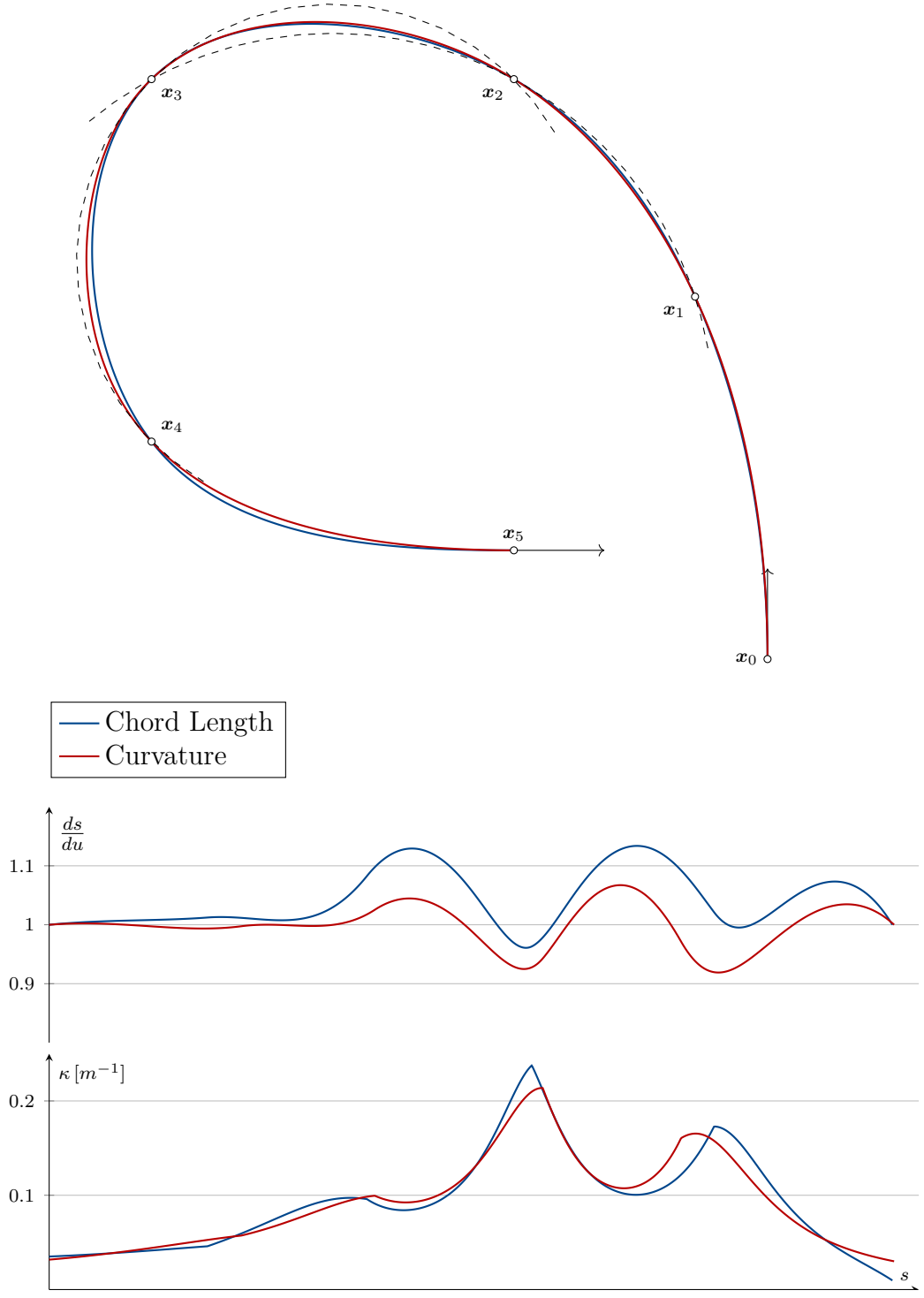


Figure 11.2: The curvature parametrization estimates the distance between two points by fitting circular arcs.

of the two circles. We then use their average

$$\hat{\kappa_i} = \frac{\kappa_{i+1} + \kappa_i}{2}$$

to estimate the arc length of $\boldsymbol{p}$ between $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$ as the arc length

$$l_i = \frac{2 \arcsin\left(0.5 d_i \hat{\kappa_i}\right)}{\hat{\kappa_i}} \tag{11.5}$$

of a circular arc through the two points with curvature $\hat{\kappa_i}$. The corresponding nodes are

$$\begin{aligned} u_0 &= 0, \\ u_{i+1} &= u_i + l_i. \end{aligned}$$

Consider the chord between $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ in Figure 11.2. The arc length of the dashed arcs is a much better approximation of the arc length of the interpolating spline than the Euclidean distance $\|\boldsymbol{x}_3 - \boldsymbol{x}_2\|$. As a result, the *curvature method* approximates the arc length of the total curve more closely. Consequently, the parametric speed $\|\boldsymbol{p}'(u)\|$ deviates less from unit speed, resulting in a more homogeneous curvature profile and a fairer curve.

The curvature method has a natural extension to spatial curves. However, since the curvatures κ_i are non-negative quantities for spatial curves as opposed to signed quantities for planar curves, the above computation overestimates the arc length between interpolation points in the vicinity of implied inflections. To get a better approximation of the arc length for spatial curves, we can take the average

$$\hat{\kappa_i} = \frac{\|\boldsymbol{d}_{i+1} + \boldsymbol{d}_i\|}{2}$$

of the RMF Darboux vector

$$\boldsymbol{d} = \kappa \boldsymbol{b}$$

of the circles, see Equation (5.17). It can be computed from (11.4) as

$$\boldsymbol{d}_i = 2 \frac{(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i) \times (\boldsymbol{x}_i - \boldsymbol{x}_{i-1})}{\|\boldsymbol{x}_{i+1} - \boldsymbol{x}_i\| \cdot \|\boldsymbol{x}_i - \boldsymbol{x}_{i-1}\| \cdot \|\boldsymbol{x}_{i+1} - \boldsymbol{x}_{i-1}\|}$$

using the identity

$$\|\boldsymbol{a} \times \boldsymbol{b}\| = \sin(\phi) \|\boldsymbol{a}\| \|\boldsymbol{b}\|.$$

Another improvement to the *curvature method* can be made for chords in which at least one of the interpolation points has an associated tangent constraint. Instead of using the circle through consecutive points, we can construct a circle from the

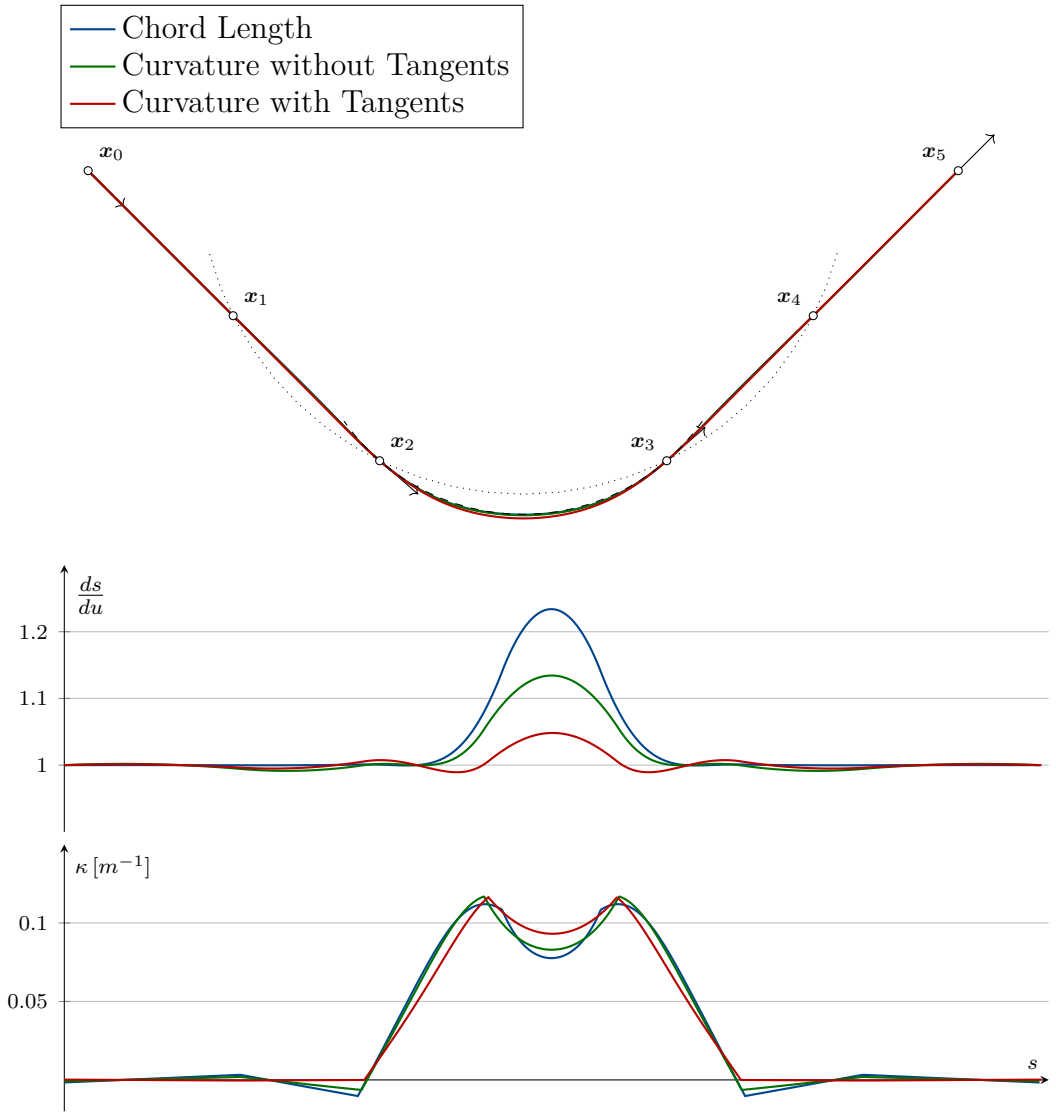


Figure 11.3: The curvature parametrization can be improved by taking the interpolation tangents into account.

two points of the chord and one tangent, as shown in Figure 11.3. Again, we can conclude that polynomial splines of more uniform parametric speed have a more favorable curvature profile.

11.4 Iteratively Refined Parametrization

Further improvement towards approximating the arc length parametrization can be achieved by iteratively adapting the nodes u_i to equal the measured arc length of the spline. Starting with some initial nodes $u_{i,0}$, e.g., determined by the chord

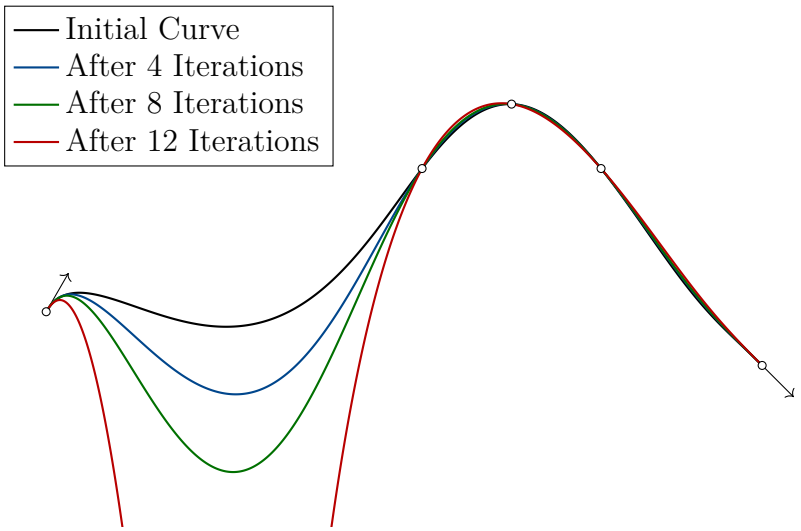


Figure 11.4: The iterative parametrization may fail to converge for specific inputs.

length method, we can use the arc length of the interpolating spline $\boldsymbol{p}_j$ of iteration j up to each interpolation point $\boldsymbol{x}_i$ to compute

$$u_{i+1,j+1} = u_{i,j+1} + \int_{u_{i,j}}^{u_{i+1,j}} \left\| \frac{d\boldsymbol{p}_j}{du} \right\| du.$$

This iteratively improved parametrization does in general not yield curves of finite arc length, as can be seen in Figure 11.4. So far, we could not find a way of predicting for a set of inputs $\boldsymbol{x}_i$, $\boldsymbol{t}_i$, $\sigma_i, \ldots$, whether or not the iterative method converges. Our current implementation uses a simple detection heuristic during each iteration step to decide if the arc length of the curve is likely converging or diverging and aborts in the latter case. Similarly, we could not find an example in which two different initial parametrizations converge to different fixed points when iteratively refining the nodes. Thus, it seems that in case of convergence, the resulting curve $\boldsymbol{p}$ is unique, and its arc length up to each of its control points $\boldsymbol{x}_i$ is

$$u_i \approx \int_0^{u_i} \left\| \frac{d\boldsymbol{p}}{du} \right\| du$$

by construction. In particular, the total arc length s of the curve is simply u_m. In Figure 11.5 we compare polynomial spline interpolants using the *chord length*, *curvature* and *iterative* method. We can see, that the iterative parametrization yields the best results, both in terms of least variation of parametric speed and in terms of smoothest curvature profile.

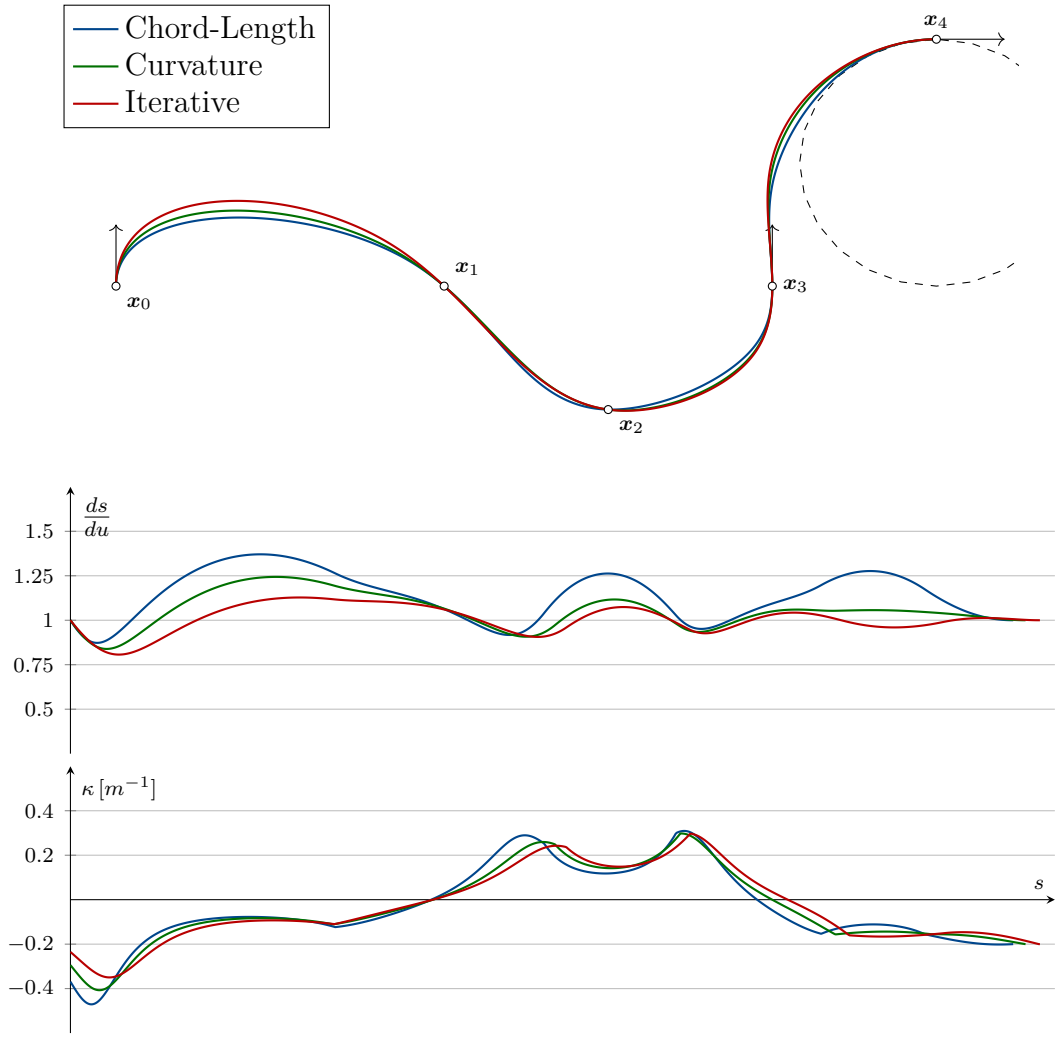


Figure 11.5: Comparison of the curvature and iterative parametrization schemes with the chord length method.

12 Optimization Techniques

In the previous chapter we evaluate various polynomial splines, their parametric speed and curvature profiles, and decide visually which of these curves are in some sense fairer than the others. The further we optimize curve generation algorithms, the more an objective measure of *curve fairness* is needed since it becomes increasingly difficult to intuitively classify the resulting curves. Additionally, if we find an objective measure of fairness that suits our application, we can set up and solve an optimization problem in order to find the most fair curve subject to given interpolation constraints.

12.1 Measuring Curve Fairness

Depending on the context, several factors contribute to what humans consider a fair curve, with Levien [55] and Moreton [61] giving extensive lists of desirable curve properties in the context of CAGD. They also include survey results on what kind of curves humans consider visually most fair. In the following, we introduce and discuss several already established objective measures of curve fairness. We also discuss their extensions to spatial curves and their applicability in the context of our work.

One of the most predominant measures for curve fairness in the literature is the *bending energy*

$$E = \int \kappa^2 ds. \tag{12.1}$$

This energy is analogous to the potential energy of a bent elastic strip. Such strips were used to construct *physical splines* long before the advent of CAD. Minimizing the bending energy is seen as a desired goal in various CAD contexts, such that different approaches of approximating these so-called *elastica* have been developed [8; 15; 53; 55]. Most of these approaches focus on efficiently generating planar elastica. Nevertheless, the integral (12.1) has a trivial extension to spatial curves, and e.g. Veltkamp et al. [95] approximated spatial curves of minimum energy.

Moreton noted, that a spatial bending energy modeling a thin elastic strip – as opposed to a thin elastic wire – also needs to account for twisting [61]. Therefore, he proposed the extended energy integral

$$E = \int \kappa^2 + \tau^2 ds \tag{12.2}$$

without further evaluation. In fact, to account for any ratio of bending and twisting stiffness of the idealized elastic strip, we need to consider the integral

$$E = \int \sigma_1 \kappa^2 + \sigma_2 \tau^2 ds \tag{12.3}$$

with any non-negative weights σ_1 and σ_2.

In some applications, the physical analogy of bending an elastic strip into a given shape does not make much sense. Consider a road that follows a given planar curve $\boldsymbol{p}$. Suppose, we design the road in such a way that a driver needs minimal adjustment (in the least squares sense) of the steering wheel during traversal. Whatever curvature the road has at any given position, the optimal way of continuing said road is to not change its curvature at all. As opposed to penalizing the curvature itself, we penalize its variation

$$E = \int \kappa'^2 \, ds. \tag{12.4}$$

Given this energy, the *reference shape*, the shape for which $E = 0$, is a circular arc, as opposed to a straight line for the bending energy (12.1). Since the corresponding energy integral considers terms of higher order, curves that minimize curvature variation have a higher order of continuity than curves that minimize their bending energy [61; 62].

There are several ways to extend the curvature variation energy (12.4) to spatial curves. Penalizing just the variation of the scalar curvature κ results in curves of constant curvature but arbitrary torsion. Therefore, Moreton [61] proposes the energy integral

$$E = \int \boldsymbol{d}'^2 \, ds \tag{12.5}$$

with

$$\boldsymbol{d} = \boldsymbol{b}\kappa$$

being the RMF Darboux vector (5.17). Using the Frenet-Serret formulas (5.10), we can rewrite the integral above as

$$E = \int \kappa'^2 + \tau^2\kappa^2 ds. \tag{12.6}$$

Intuitively, this energy penalizes curves that change their osculating circle, either in magnitude or in orientation. The corresponding reference shape for this energy is again a circular arc, just as for planar minimum variation curves.

Extending the road analogy above to spatial curves leads to a different energy. In a heavily simplified mental model and disregarding gravity, airplanes are able to fly arbitrary helical shapes without adjustment of the flight stick, we penalize variation of torsion as well as curvature and get

$$E = \int \kappa'^2 + \tau'^2 \, ds. \tag{12.7}$$

Although the energy (12.6) is more widely used for spatial curves, the energy (12.7) has applications, e.g., in 3D curve completion [38].

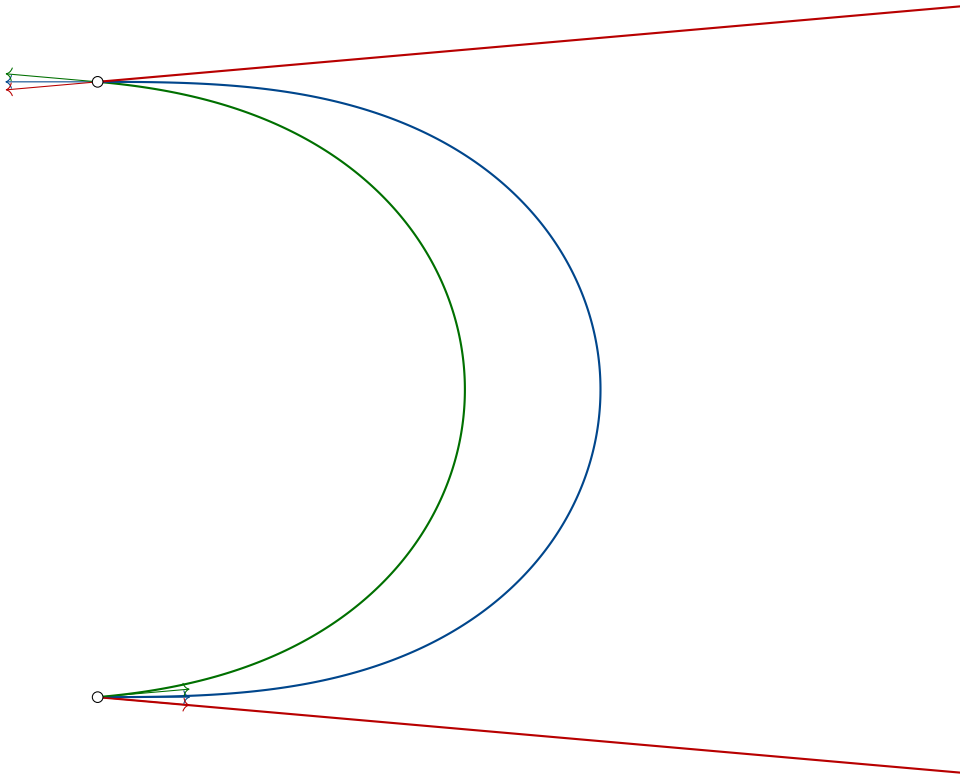


Figure 12.1: In some cases, no curve of finite length minimizes the bending energy.

For all energies above, it is possible to find constraints for which there is no curve of finite length minimizing that energy. Intuitively, this is due to the fact that we can always decrease the total energy of a curve by expanding it in such a way that the respective integrand decreases faster than the curve's total arc length increases. A well known example of such constraints in the case of minimizing bending energy (12.1) is shown in Figure 12.1. One possible solution to this problem is to penalize long curves, e.g., by adding a constant term c to the energy integral

$$E = \int \kappa^2 + c \, ds.$$

Obviously, a curve minimizing this integral is of finite length. However, note that E and thus the resulting curve heavily depend on the choice of c and that E is not scale invariant.

Moreton et al. [62] propose a much more elegant solution to obtain scale invariant energies. The scale invariant bending energy is

$$E = \int 1 ds \cdot \int \kappa^2 ds \tag{12.8}$$

and the scale invariant curvature variation energy is

$$E = \left(\int 1 ds \right)^3 \cdot \int {\kappa'}^2 \, ds. \tag{12.9}$$

These energies are more stable than their non scale invariant counterparts. The work of Moreton et al. also includes several examples demonstrating the behavior of curves that minimize bending energy, curvature variation and their scale invariant counterparts. In the following, we adapt the naming convention of Moreton et al. and refer to these curves as minimum energy curves (MEC), scale invariant minimum energy curves (SI-MEC), minimum variation curves (MVC) and scale invariant minimum variation curve (SI-MVC), respectively. We refer to this class of curves as *energy minimizing curves*.

12.2 Approximating Energy Minimizing Curves

Polynomial spline interpolants can be used to approximate energy minimizing curves. The energies E of a curve $\boldsymbol{p}$ introduced in the previous section can be expressed as an integral

$$E = \int f^2(s) ds.$$

We can approximate this integral by the finite sum

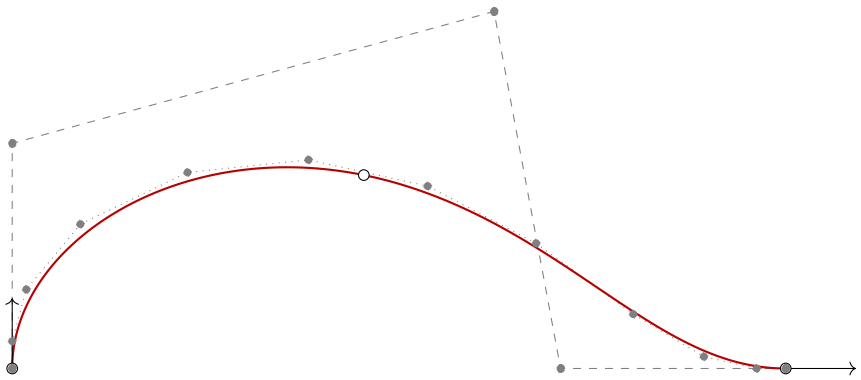
$$\begin{aligned} E &\approx \sum_{j=0}^{k} f^2\left(u_j\right) \left\| \frac{d\boldsymbol{p}}{du}(u_j) \right\| \Delta u \\ &= \sum_{j=0}^{k} \left(f\left(u_j\right) \sqrt{\left\| \frac{d\boldsymbol{p}}{du}(u_j) \right\| \Delta u} \right)^2 \\ &=: \sum_{j=0}^{k} \hat{f}^2\left(u_j\right), \\ \Delta u &:= \Delta u_0 = \ldots = \Delta u_{k-1}. \end{aligned} \tag{12.10}$$

Given this approximation, we can formulate the non-linear equality constrained least squares problem

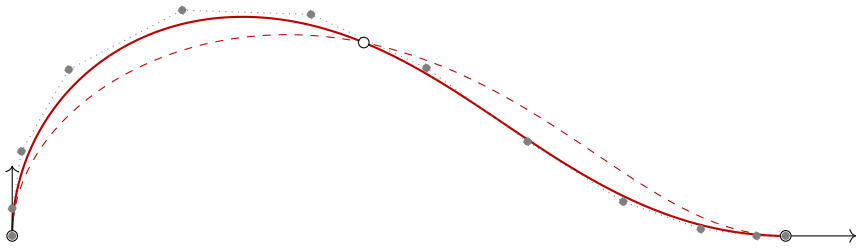
$$\begin{aligned} \underset{\boldsymbol{p}}{\text{minimize}} \quad & \sum_{j=0}^{k} \hat{f}^2\left(u_j\right) \\ \text{subject to} \quad & \boldsymbol{NC} = \boldsymbol{X} \end{aligned} \tag{12.11}$$

which needs to be solved in order to yield a polynomial spline that minimizes E. The above problem can be solved by standard optimization algorithms such as *gradient descent*, the *Gauss-Newton algorithm* or by the *Levenberg-Marquardt algorithm*. We have implemented the latter due to its stability and adaptability.

Our curve generation algorithm is visualized in Figure 12.2. Given n interpolation



a) Initial spline interpolating three points and two tangents. The dotted control polygon $\boldsymbol{C}_0$ is obtained from the dashed polygon after several knot insertions to increase the degrees of freedom of the spline.



b) Starting with $\boldsymbol{C}_0$, constrained non-linear optimization yields an approximating curve of minimum energy. The initial curve is shown as a dashed line for reference.

Figure 12.2: Generation of a spline that approximates a curve of minimum energy.

constraints, we first compute the interpolating spline with n control points. We then refine the interpolating spline by inserting m knots [72, Section 7.11], resulting in $n+m$ control points, which we will refer to as the initial control vector $\boldsymbol{C}_0$. Next, we use the *Levenberg-Marquardt algorithm* to calculate an update $\Delta\boldsymbol{C} = \boldsymbol{C}_{i+1} - \boldsymbol{C}_i$ that leads the curve towards a local minimum of the energy function E, while still satisfying all constraints. Since $\boldsymbol{N}\boldsymbol{C}_0 = \boldsymbol{X}$ holds, the linearized problem

$$\begin{aligned} \underset{\Delta\boldsymbol{C}}{\text{minimize}} \quad & \|\boldsymbol{J}\Delta\boldsymbol{C} + \boldsymbol{E}\|^2 \\ \text{subject to} \quad & \boldsymbol{N}\Delta\boldsymbol{C} = \boldsymbol{0} \end{aligned}$$

determines the update $\Delta \boldsymbol{C}$ while preserving the interpolation constraints at all nodes, where

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,0}}(u_0) & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,1}}(u_0) & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,2}}(u_0) & & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,j}}(u_0) \\ \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,0}}(u_1) & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,1}}(u_1) & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,2}}(u_1) & \cdots & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,j}}(u_1) \\ \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,0}}(u_2) & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,1}}(u_2) & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,2}}(u_2) & & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,j}}(u_2) \\ & \vdots & & \ddots & \vdots \\ \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,0}}(u_{k-1}) & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,1}}(u_{k-1}) & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,2}}(u_{k-1}) & \cdots & \frac{\partial \hat{f}}{\partial \boldsymbol{C}_{i,j}}(u_{k-1}) \end{bmatrix}$$

denotes the *Jacobian matrix* of $\hat{f}$ with respect to the elements of $\boldsymbol{C}_i$ and

$$\boldsymbol{E} = \begin{bmatrix} \hat{f}(u_0) & \hat{f}(u_1) & \hat{f}(u_2) & \dots & \hat{f}(u_{k-1}) \end{bmatrix}^{\mathsf{T}}$$

contains the energy contributions of the curve at the sampling points.

Using *Lagrangian multipliers* $\boldsymbol{\lambda}$ and a damping factor μ, the solution to the system of linear equations

$$\begin{bmatrix} \boldsymbol{J}^{\mathsf{T}}\boldsymbol{J} + \mu \boldsymbol{I} & \boldsymbol{N}^{\mathsf{T}} \\ \boldsymbol{N} & 0 \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{C} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{J}^{\mathsf{T}}\boldsymbol{E} \\ \boldsymbol{0} \end{bmatrix}$$

yields the update $\Delta \boldsymbol{C}$ [80, Section 6.4]. The optimization step is repeated until $\|\Delta \boldsymbol{C}\|$ is sufficiently small. Figure 12.3 shows a specific constraint setup for which our method converges to different curves depending on the choice of initial curve, e.g., due to a different parametrization scheme. In our context, we accept these ambiguities since the designer can resolve them by revising the interpolation constraints.

Remark: There is a close relationship between the number n of constraints, the number m of knot insertions and the number k of samples to approximate the energy integral. Obviously, if we choose $m = 0$, the result of the optimization problem above is simply the initial interpolating spline since the constraints fully determine the curve. If we choose $m \geq k$, the resulting spline can satisfy all n constraints as well as independently minimize the $\hat{f}(u_j)$ for all $u_j, j = 0, \dots, k-1$. In practice, we want the spline to have enough degrees of freedom to minimize the energy E sufficiently, but at the same time to not show signs of *overfitting*. This is achieved by requiring $k > m$ at least, where larger values of k relative to m lead to closer approximations of the energy integral and higher values of m lead to closer approximations of the energy minimizing curve. Figure 12.4 shows the respective MVC energy values given the simple input of two end point and tangent constraints shown in Figure 12.5.

In case of the scale invariant energies (12.8) and (12.9), the energy is a product of

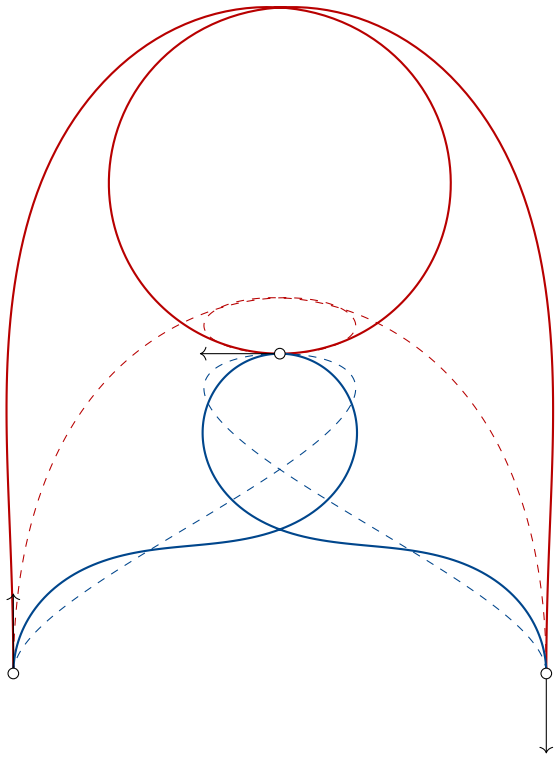


Figure 12.3: Depending on the initial curve (dashed lines), different local energy minima (solid lines) may be found.

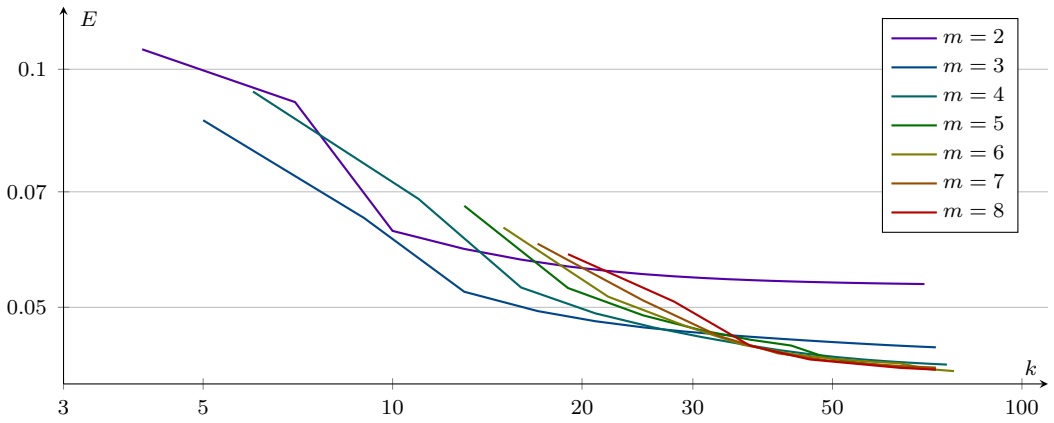


Figure 12.4: MVC energies for various values of m and k for the curve shown in Figure 12.5.

integrals. Using the approximation (12.10), we need to minimize expressions of the form

$$E \approx \left(\sum_{j=0}^{k} \hat{f}^2 \left(u_j \right) \right)^a \cdot \sum_{j=0}^{k} \hat{g}^2 \left(u_j \right),$$

with $a = 1$ for the SI-MEC and $a = 3$ for the SI-MVC case. This results in Jacobian matrices $\boldsymbol{J}$ with $O(k^{a+1})$ rows, which dramatically increases the computational complexity of each iteration of the *Levenberg-Marquardt algorithm*, especially in case of the SI-MVC. In our implementation, we approximate this energy by

$$E \approx aF^{a-1}G \cdot \sum_{j=0}^{k} \hat{f}^2 \left(u_j \right) + F \sum_{j=0}^{k} \hat{g}^2 \left(u_j \right)$$

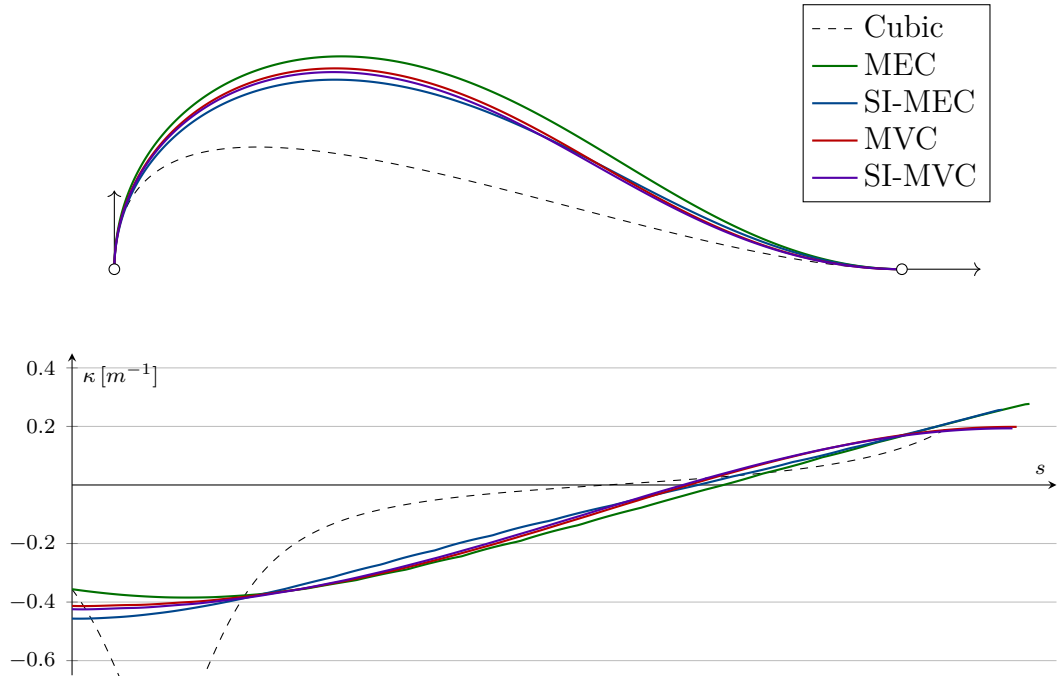
where $F = \sum \hat{f}^2$ and $G = \sum \hat{g}^2$ of the previous iteration. The respective Jacobian matrix $\boldsymbol{J}$ consists of only $O(k)$ rows, which increases performance in each iteration at the cost of convergence speed.

Calculating the Jacobian matrices for the various energies presented above is rather lengthy and involved, which is why we have moved the respective calculations to Appendix A.

12.3 Evaluation of Different Energy Measures

Using the scheme above, we can compute polynomial spline representations that approximate various interpolating energy minimizing curves. Figure 12.5 shows a simple example of planar curves interpolating two points with specified tangents. The curves are calculated with $m = 15$ and $k = 4096$, which leads to close approximations of the energy minimizing curves. From the table, we observe that the (SI-)MEC and (SI-)MVC curves indeed minimize the respective energies. Furthermore, we see that all energy minimizing curves are much more visually pleasing and have a much more favorable curvature profile than the interpolating cubic. Furthermore, the MEC and MVC admit a longer arc length than their respective scale invariant counterpart.

Since we use polynomial splines, we can construct energy minimizing interpolants that satisfy an arbitrary number of geometric constraints and thus consist of several MEC/MVC segments, see Section 5.10. However, since higher order geometric constraints reduce the continuity of the spline at the respective interpolating points [53], a designer should be incentivized to add interpolation points instead of specifying higher order geometric constraints to existing points where possible. The reduction in continuity of energy minimizing interpolants at interpolation points with added constraints should be replicated in the underlying polynomial spline by adding knots of respective multiplicity at the corresponding nodes. In our implementation, we observe that otherwise the optimization becomes less stable,

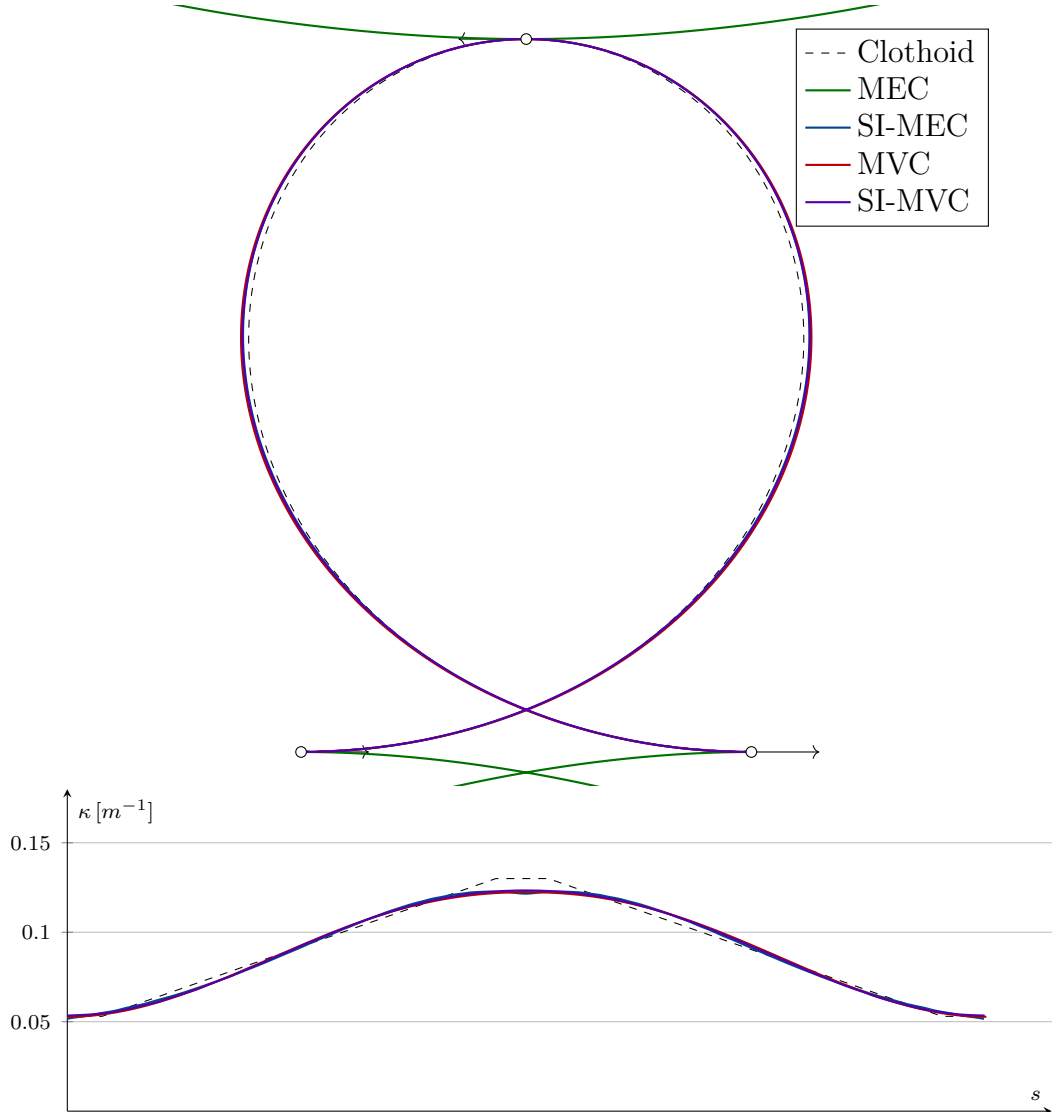


	s	$\int \kappa^2 ds$	$s \int \kappa^2 ds$	$\int \kappa'^2\, ds$	$s^3 \int \kappa'^2\, ds$
Cubic	11.17	1.078	12.044	0.6194	864.04
MEC	12.21	**0.775**	9.465	0.0459	83.46
SI-MEC	11.86	0.786	**9.329**	0.0463	77.35
MVC	12.06	0.782	9.436	**0.0398**	69.83
SI-MVC	12.01	0.784	9.420	0.0399	**69.07**

Figure 12.5: Comparison of the MEC, SI-MEC, MVC and SI-MVC curve.

since creating these higher order discontinuities plays a central role in minimizing the respective energy.

In Section 8, we mention that in the context of roller coasters, curves that are traversed with a sufficiently high velocity need to be at least G^2 continuous to ensure a smooth acceleration profile during traversal. We therefore consider (SI-)MEC splines to be less feasible than curves of higher order continuity, although they create favorable trajectories between control points in case a solution of finite length exists. Furthermore, we found that MEC splines are difficult to work with in practice, since many common inputs have no MEC solution of finite length. Figure 12.6 shows an example of such a case. Referring back to Section 1.1, vertical loopings have traditionally been designed using circular arcs connected with clothoids. The resulting teardrop shape results in a favorable acceleration profile during traversal since gravity reduces the train velocity at the top of the looping and gravitational and inertial accelerations partly cancel each other out at the top. When designing this shape using an interpolating spline, a natural and intuitive choice for the control points are its extrema, since these are its most



	s	$\int \kappa^2 ds$	$s \int \kappa^2 ds$	$\int \kappa'^2\, ds$	$s^3 \int \kappa'^2\, ds$
Clothoid	69.38	0.682	47.34	0.395	132.0
MEC	∞	**0**	∞	—	—
SI-MEC	69.91	0.609	**42.55**	0.358	122.3
MVC	70.09	0.607	42.56	**0.348**	119.9
SI-MVC	69.99	0.609	42.60	0.350	**119.9**

Figure 12.6: Using only the extrema as control points, splines of minimum energy closely resemble the traditional clothoid looping.

striking features. Using just these points and their respective tangent vectors, the resulting SI-MEC, MVC and SI-MVC splines are visually indistinguishable in their appearance while there is no finite length MEC solution. Furthermore, all well behaved splines of minimum energy result in shapes and curvature profiles that closely resemble the traditional clothoid shape, with the (SI-)MVC splines additionally being of higher order continuity than the G^2 continuous trajectory of the clothoid loop. This finding greatly strenghens our confidence that (SI-)MVC splines are a promising candidate for potential CAGD tools in the context of layout design, not only for roller coasters, but also for other types of tracks.

12.4 Spatial Energy Minimizing Curves

Our spline interpolation algorithm poses no restrictions on the dimension of the curve. Thus, there is no additional challenge involved in approximating spatial energy minimizing interpolants, except for finding a suitable energy to minimize. As we already discussed in Section 12.1, two major variants of the (scale invariant) curvature variation energy have been proposed, minimizing the variation of the RMF Darboux vector (12.6) and using the first derivative of the torsion (12.7), which we call *space MVC* in the following.

Figures 12.7 and 12.8 show two interpolation conditions and their respective MVC and space MVC splines. We can see that both variants behave quite similarly as long as the underlying data does not imply any inflections, such as in Figure 12.7. On the other hand and as shown in Figure 12.8 with implied inflections, there are radical differences between the MVC and space MVC spline, which can intuitively be explained by the reference shapes of both variants.

We found that MVC splines minimizing (12.6) behave more predictably in an interactive setting. In the example shown in Figure 12.8, the space MVC is sensitive to small changes of the interpolation constraints, which may confuse designers. Moreover, the MVC spline follows the implied shape of the user input more closely without generating new features that may or may not be intended.

12.5 Minimum Jerk Curves

Curves that minimize curvature variation also minimize the jerk experienced when being traversed with constant velocity. As stated in Section 7.6, this is true for many common applications such as railway design. If we cannot assume constant path velocity, a curve that minimizes jerk during traversal may differ significantly from the MVC. The reference shape of such curves is no longer the circular arc, but rather a curve of constant inertial acceleration, see Section 7.6. Fortunately, extending the concept of MVC to curves of minimum jerk (MJC) is rather straightforward in our scheme. First, we extend the energy integral from

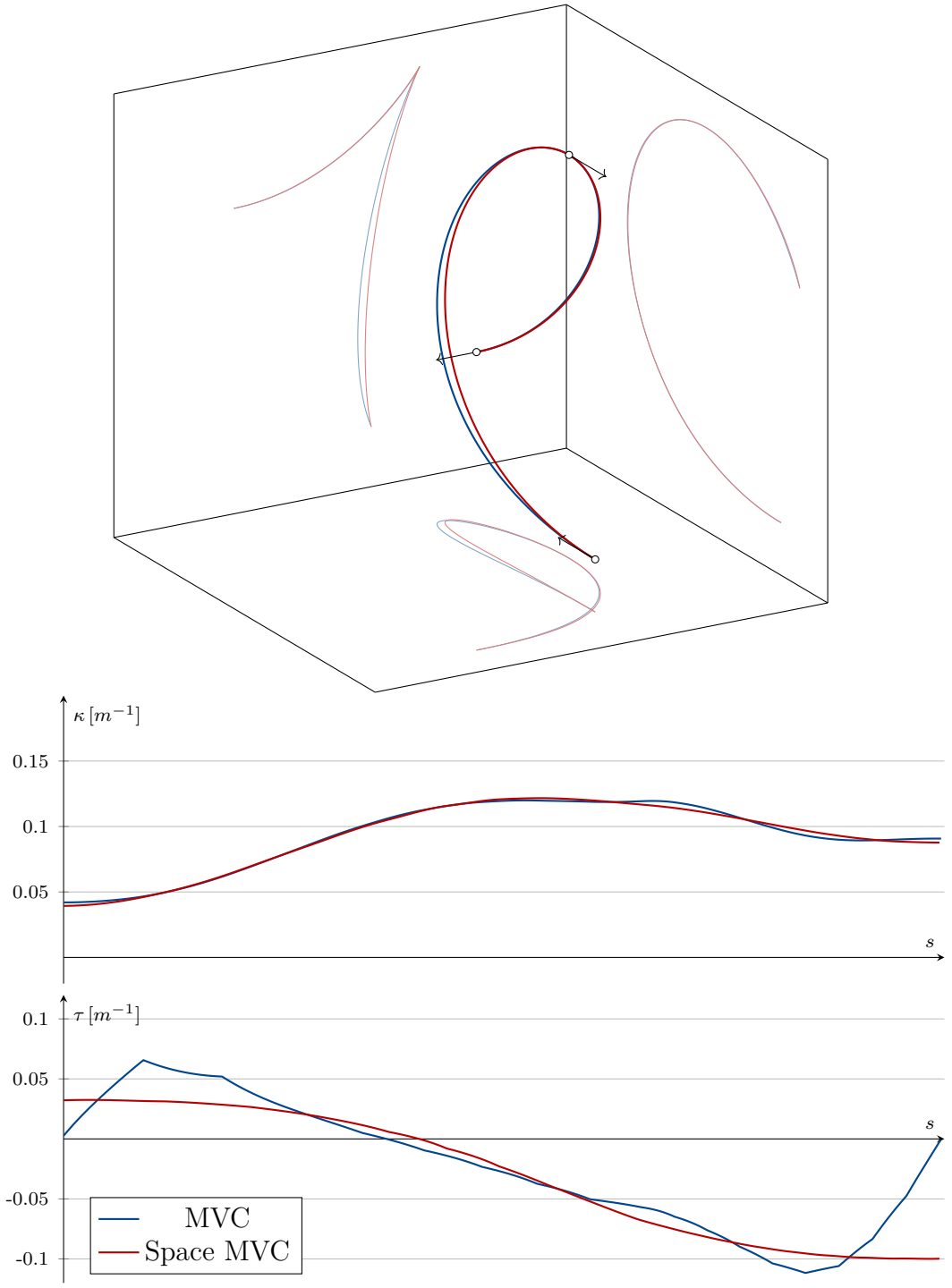


Figure 12.7: For certain input constraints, MVC and space MVC follow similar paths.

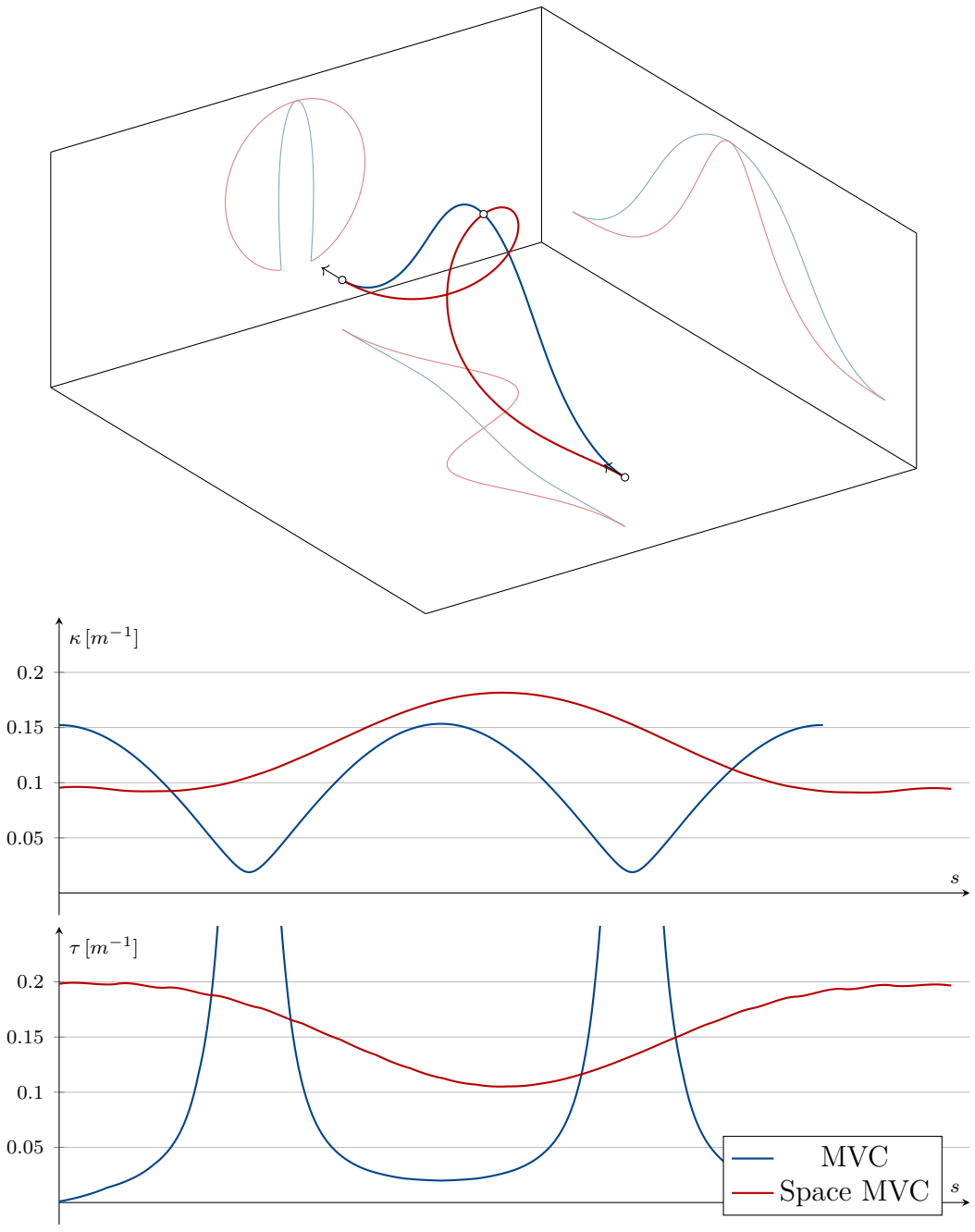


Figure 12.8: For other inputs, the MVC and space MVC greatly differ due to the different reference shapes of the two.

Equation (12.5) to also consider the path velocity v. Since we can express the inertial acceleration during traversal as

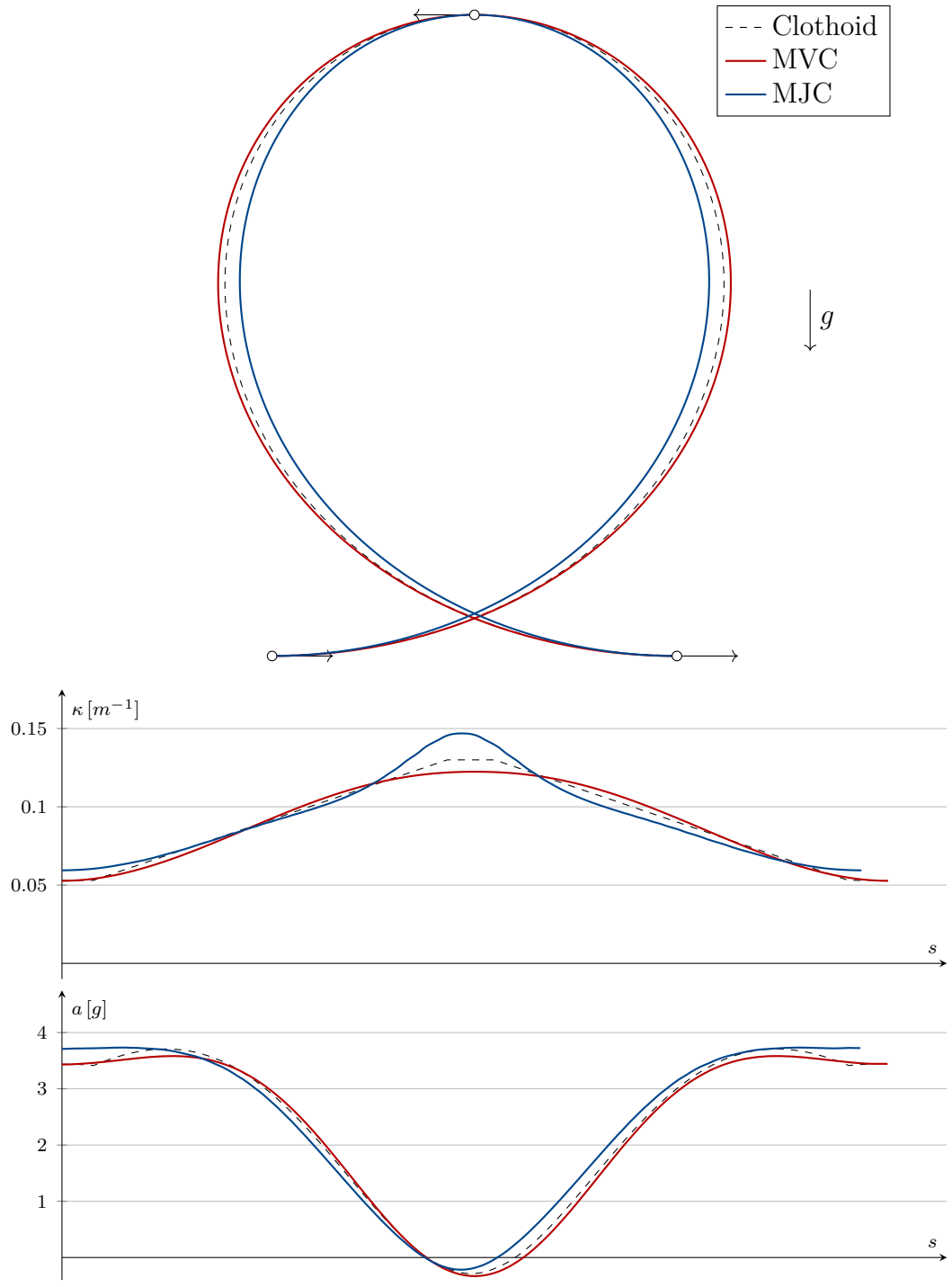
$$\boldsymbol{a} = v^2 \boldsymbol{b} \kappa,$$

the resulting energy for curves of minimum jerk is

$$\begin{aligned} E &= \int \dot{\boldsymbol{a}}^2 \, dt \\ &= \int \left(2\dot{v}v\kappa + v^3\kappa'\right)^2 + \left(v^3\tau\kappa\right)^2 dt \\ &= \int v \left(\left(2\dot{v}\kappa + v^2\kappa'\right)^2 + \left(v^2\tau\kappa\right)^2\right) ds. \end{aligned} \tag{12.12}$$

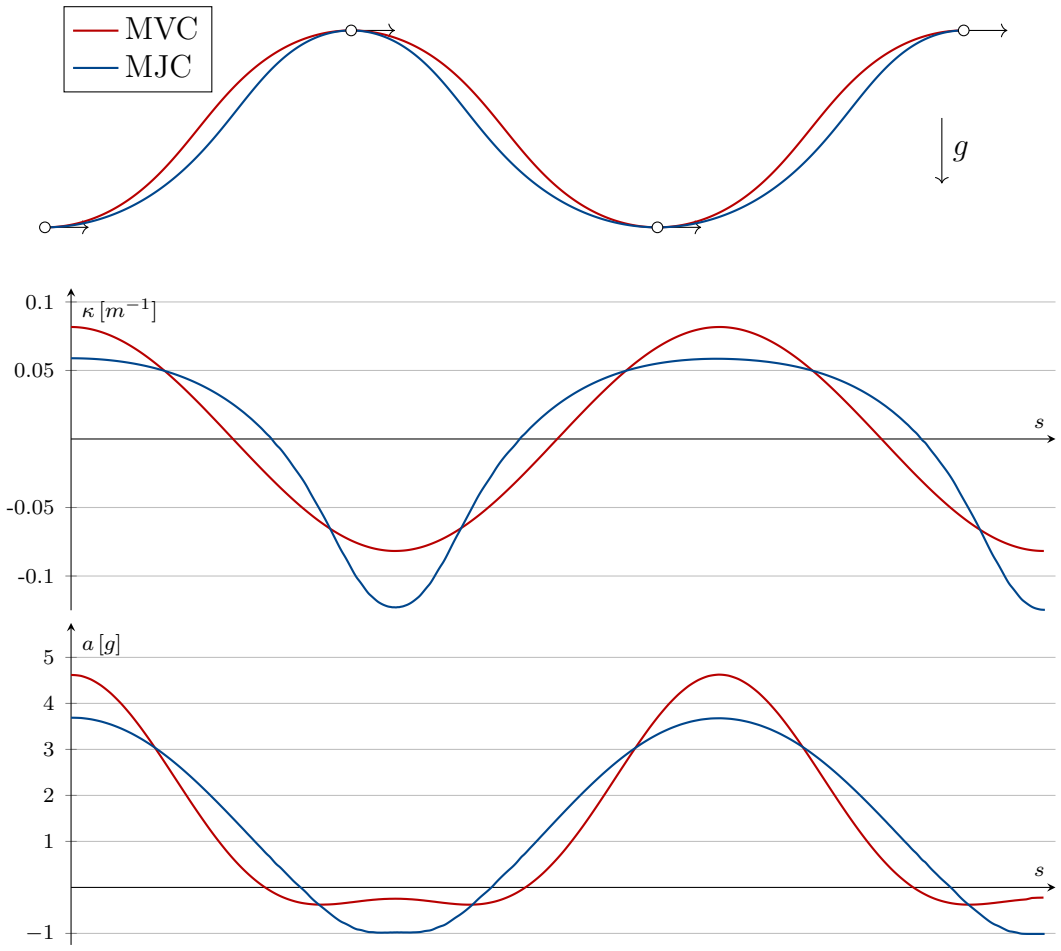
The calculation of the respective Jacobian matrix used in our spline interpolation scheme can be found in Appendix A.

In Figure 12.9, we pick up the example from Figure 12.6 and compare the acceleration profile of the MVC, MJC and the classical clothoid loop. We can see that, although the curvature profile of the MJC is much less favorable than both of the clothoid and MVC, the acceleration profile admits less variation, resulting in the least jerk during traversal. The more a coach approaches the apex of the vertical looping, the less velocity it has and the higher the curvature needs to be in order to maintain the same inertial acceleration. We demonstrate a more pronounced example of this effect in Figure 12.10. Since the interpolation constraints follow a regular pattern in this example, the MVC results in a point symmetric shape. Due to the coach being slower on the maxima of the curve, the curvature profile of the MJC is skewed with larger curvatures at the maxima and lower curvatures at the minima. As a result, the acceleration profile is much fairer as indicated by the respective energy.



	s	$\int \kappa'^2\,ds$	$\int \dot{a}^2\,dt$
Clothoid	69.38	0.395	44.62
MVC	70.09	**0.348**	41.85
MJC	67.92	0.56	**40.20**

Figure 12.9: Curves of minimum jerk adapt the curvature profile based on path velocity, resulting in favorable acceleration profiles.



	s	$\int \kappa'^2\,ds$	$\int \dot{a}^2\,dt$
MVC	103.80	**2.83**	140.01
MJC	103.88	4.77	**85.80**

Figure 12.10: Airtime hill elements constructed using MJC and MVC splines.

13 Framing A Curve

Throughout the previous chapter, we exclusively talk about generating fair parametric curves $\boldsymbol{p}$. To complete a fair framed curve $\mathbf{T}$ given $\boldsymbol{p}$, we need to construct a fair moving frame $\boldsymbol{Q}$. In Chapter 5, we introduc two state of the art approaches to generate such moving frames, namely the *Frenet frame* and *rotation minimizing frames* (RMF). For layout design, these moving frames can hardly be used effectively, mostly due to their lack of flexibility and descriptive power in an interactive CAGD setting. As with constructing the curves $\boldsymbol{p}$, we seek methods which allow users to express their design intent while generating fair moving frames. As a consequence, we need to establish of what we consider *fair* first, before it is possible to evaluate framed curves and develop optimization techniques for moving frames.

13.1 Fairness of Moving Frames

Given a curve $\boldsymbol{p}(s)$ with a curvature profile $\kappa(s)$, the choice of an associated moving frame $\boldsymbol{Q}$ determines

1. how the curve's curvature κ is split into the curvatures κ_n and κ_g and

2. the relative torsion τ_r.

If the framed curve $\mathbf{T}$ composed of $\boldsymbol{p}$ and $\boldsymbol{Q}$ represents a spatial track, the former determines along which normal the inertial accelerations induced by $\boldsymbol{p}$ act while the latter induces angular motion about the local x-axis. Any measure of fairness of moving frames in the context of track layout design needs to account for both effects. Gravitoinertial accelerations acting along the normal $\boldsymbol{v}$ are greatly preferred to those acting along $\boldsymbol{u}$ while angular motion should be kept within acceptable limits, see Chapter 7. Nevertheless, we will investigate both goals in isolation first, starting with minimizing rotational motion, before proposing a measure of fairness for moving frames.

Proposition 13.1: Of all moving frames of a curve $\boldsymbol{p}$, the RMF minimizes the length of the generalized Darboux vector.

Proof. Let ϕ be the angular displacement between the Frenet frame and a moving frame $\boldsymbol{Q}$ of $\boldsymbol{p}$ such that

$$\boldsymbol{b} = \cos(\phi)\boldsymbol{v} + \sin(\phi)\boldsymbol{u}$$

holds. The generalized Darboux vector of $\boldsymbol{Q}$ given by Equation (5.16) can be rewritten as

$$\boldsymbol{d} = \tau_r \boldsymbol{t} + \kappa \boldsymbol{b}$$

since

$$\begin{aligned}\kappa \boldsymbol{b} &= \kappa \cos(\phi)\boldsymbol{v} + \kappa \sin(\phi)\boldsymbol{u} \\ &= \kappa_g \boldsymbol{v} + \kappa_n \boldsymbol{u}.\end{aligned}$$

Whereas $\boldsymbol{t}$, $\boldsymbol{b}$ and κ are properties of $\boldsymbol{p}$, only τ_r is determined by the choice of moving frame $\boldsymbol{Q}$. For the RMF $\tau_r = 0$ holds, which minimizes

$$\|\boldsymbol{d}\| = \sqrt{\kappa^2 + \tau_r^2}.$$

□

Since every curve $\boldsymbol{p}$ has a one parameter family of RMFs due to the choice of initial normals $\boldsymbol{u}(0)$ and $\boldsymbol{v}(0)$, we can determine the RMF that minimizes

$$\int \langle \boldsymbol{a}, \boldsymbol{u} \rangle \, dt$$

where $\boldsymbol{a}$ is the gravitoinertial acceleration experienced during traversal of $\mathbf{T}$. However, this may in general yield unacceptable acceleration profiles in practice. Only considering rotational motion for fairness is insufficient since it disregards the resulting acceleration profile while traversing $\mathbf{T}$.

Consider a rigid body $\mathcal{B}$ that inherits the frame of a framed curve $\mathbf{T}$ and traverses it with path velocity v and path acceleration $\dot{v}$ in a gravitational field. The gravitoinertial acceleration of $\mathcal{B}$ is given by

$$\boldsymbol{a} = v^2 \kappa \boldsymbol{n} + \dot{v}\boldsymbol{t} + g\boldsymbol{e}_3,$$

where $\boldsymbol{n}$ is the principal normal of $\boldsymbol{p}$ according to Equation (5.7) and g is the earth's gravitational acceleration. In order to construct a frame of minimal lateral acceleration, i.e., accelerations along the normal $\boldsymbol{u}$, we set

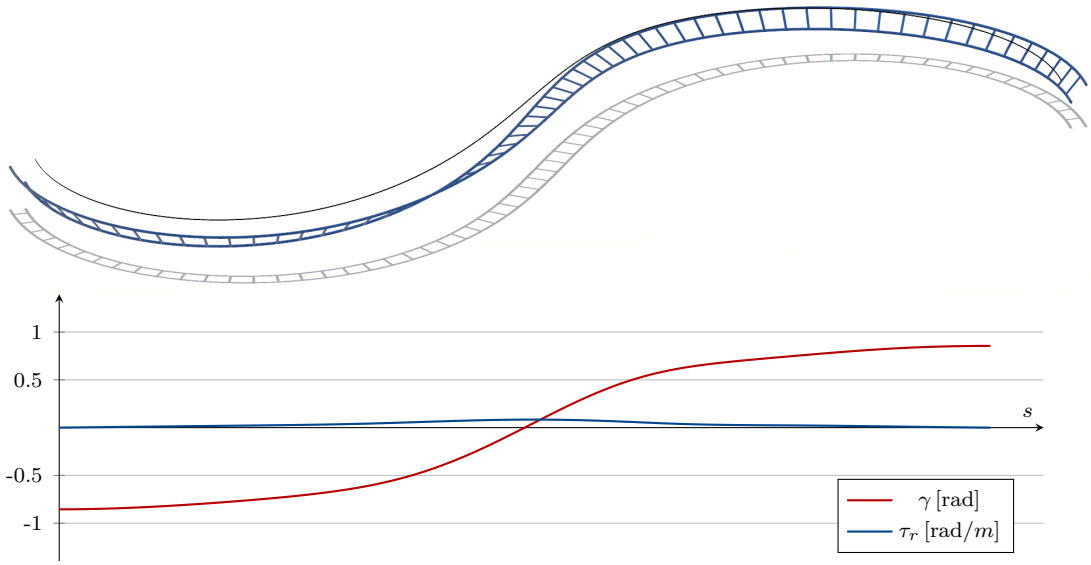
$$\boldsymbol{a} = \begin{bmatrix} \boldsymbol{t} & \boldsymbol{u} & \boldsymbol{v} \end{bmatrix} \begin{bmatrix} a_x \\ 0 \\ a_z \end{bmatrix}.$$

Since $\langle \boldsymbol{n}, \boldsymbol{t} \rangle = 0$, we get

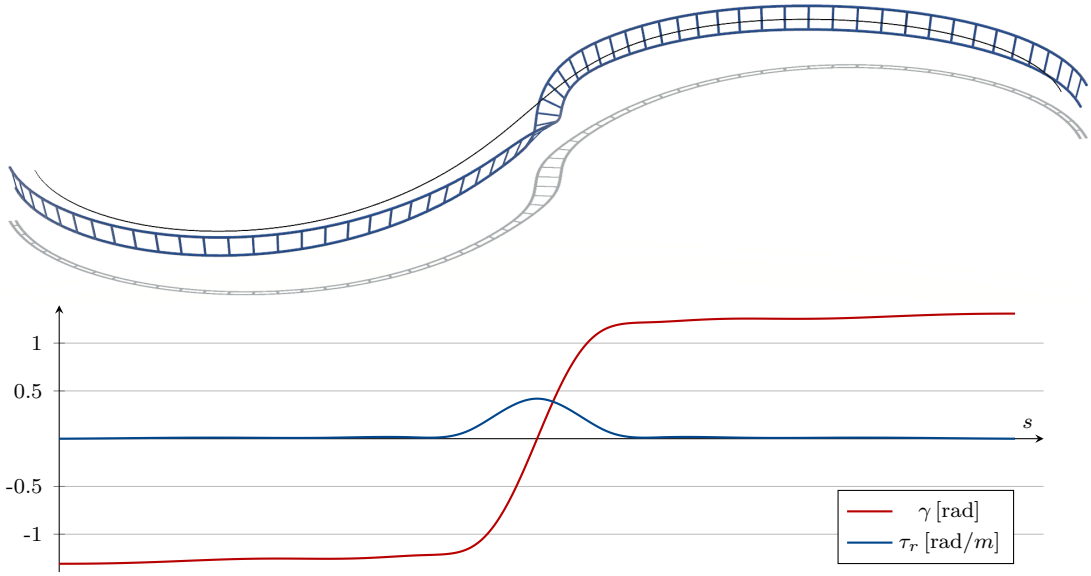
$$\begin{aligned}a_x &= \left\langle v^2 \kappa \boldsymbol{n} + \dot{v}\boldsymbol{t} + g\boldsymbol{e}_3, \boldsymbol{t} \right\rangle \\ &= \dot{v} + g t_3\end{aligned}$$

and consequently the normal $\boldsymbol{v}$ needs to be

$$\boldsymbol{v} = \frac{v^2 \kappa \boldsymbol{n} + g\,(1 - t_3)\,\boldsymbol{e}_3}{\|v^2 \kappa \boldsymbol{n} + g\,(1 - t_3)\,\boldsymbol{e}_3\|}.$$



a) For lower velocities, both the roll angle γ and track twist τ_r for which the lateral accelerations are minimized stay within reasonable limits.



b) At higher velocities, the relative torsion τ_r may grow to unreasonably high values, resulting in excessive rotational accelerations and velocities during traversal.

Figure 13.1: S-curve using a roll function that minimizes lateral accelerations for various velocities.

For $g/(v^2\kappa) \to 0$, this frame converges to the Frenet frame since $\boldsymbol{v} \to \boldsymbol{n}$. Due to this convergence, the resulting relative torsion τ_r and consequently the angular motion of $\mathcal{B}$ may be unbounded if we were to use this frame in practice. The sharper the change in curvature of the underlying curve $\boldsymbol{p}$ becomes or the faster the curve is traversed, the larger τ_r has to be in order to minimize the lateral accelerations in direction of $\boldsymbol{u}$, see Figure 13.1.

A compromise between minimizing just angular motion or just lateral accelerations is to minimize their weighted sum

$$\int \sigma_0 \, {a_y}^2 + \sigma_1 \left(v\tau_r\right)^2 dt \tag{13.1}$$

with non-negative weights σ_0 and σ_1 whose values need to be determined empirically based on the underlying application. From our experience in the field of roller coasters, a more sophisticated measure of fairness shall also account for angular accelerations and penalize twisted track sections for small path velocities. These requirements are captured by the integral

$$\int \sigma_0 \, {a_y}^2 + \sigma_1 \left(\left(v + \sigma_2\right)\tau_r\right)^2 + \sigma_3 \left(\dot{v}\tau_r + v\tau_r'\right)^2 dt, \tag{13.2}$$

which reduces to Equation (13.1) for $\sigma_2 = \sigma_3 = 0$. There are cases however, for which no moving frame may achieve acceptable results since either the rotational motion or the lateral accelerations become excessive, i.e., not suitable for human patrons. In those cases, the designer needs to adapt the underlying curve $\boldsymbol{p}$ accordingly. In fact, from our experience this case is rather common, which is why we try to keep our proposed CAGD system as responsive as possible.

13.2 Moving Frames in Classical Layout Design

As already mentioned in Section 1.1, the classical way to design railroad tracks is closely related to the Euler angles. The tangent $\boldsymbol{t} \neq \pm\boldsymbol{e}_3$ of a curve $\boldsymbol{p}$ determines the yaw and pitch angles as

$$\begin{aligned} \theta &= \arctan\left(\frac{\langle \boldsymbol{t}, \boldsymbol{e}_2 \rangle}{\langle \boldsymbol{t}, \boldsymbol{e}_1 \rangle}\right), \\ \alpha &= -\arcsin\left(\langle \boldsymbol{t}, \boldsymbol{e}_3 \rangle\right), \end{aligned} \tag{13.3}$$

see Equations (4.9) and (4.10). In contrast, the roll angle γ solely determines the orientation of the normals $\boldsymbol{u}$ and $\boldsymbol{v}$ and therefore is free to be chosen. Consequently, we can specify the roll function $\gamma(s)$ of the track either graphically, e.g., by a

hand drawing the so-called *superelevation plot*, or algebraically, which yields the corresponding framed curve

$$\mathbf{T} = \begin{bmatrix} C_\theta C_\alpha & -S_\theta C_\gamma + C_\theta S_\alpha S_\gamma & S_\theta S_\gamma + C_\theta S_\alpha C_\gamma & \\ S_\theta C_\alpha & C_\theta C_\gamma + S_\theta S_\alpha S_\gamma & -C_\theta S_\gamma + S_\theta S_\alpha C_\gamma & \boldsymbol{p} \\ -S_\alpha & S_\gamma C_\alpha & C_\gamma C_\alpha & \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{13.4}$$

After computing the derivative

$$\mathbf{T}' = \mathbf{T} \begin{bmatrix} 0 & -\theta' C_\alpha C_\gamma + \alpha' S_\gamma & \alpha' C_\gamma + \theta' C_\alpha S_\gamma & 1 \\ \theta' C_\alpha C_\gamma - \alpha' S_\gamma & 0 & -\gamma' + \theta' S_\alpha & 0 \\ -\alpha' C_\gamma - \theta' C_\alpha S_\gamma & \gamma' - \theta' S_\alpha & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

and equaling the terms in Equation (5.15) we can derive its curvatures as

$$\begin{bmatrix} \kappa_g \\ \kappa_n \\ \tau_r \end{bmatrix} = \begin{bmatrix} \cos(\alpha)\cos(\gamma) & -\sin(\gamma) & 0 \\ \cos(\alpha)\sin(\gamma) & \cos(\gamma) & 0 \\ -\sin(\alpha) & 0 & 1 \end{bmatrix} \begin{bmatrix} \theta \\ \alpha \\ \gamma \end{bmatrix}'. \tag{13.5}$$

Since Equation (13.5) is very similar to Equation (10.5) in Section 10.2, we can see that the differential quantities θ' and α' have a close relationship to the curvatures of the layout plan $\bar{\boldsymbol{p}}$ and height profile $z(\bar{s})$ as introduced in Section 1.1. Since in the classical layouting scheme the function $\gamma(s)$ is entirely user defined, it is the designer's task to ensure sufficient continuity for it as well as to make sure that it creates a fair moving frame.

13.3 Modeling Techniques

Using Equation (13.5) above, we can identify the function

$$\gamma(s) = \arctan\left(-\frac{\alpha'}{\cos(\alpha)\theta'}\right)$$

for which the moving frame (13.4) is the Frenet frame as $\kappa_n = 0$, and

$$\gamma(s) = \gamma_0 - \int \sin(\alpha)\theta' ds$$

for which the moving frame is rotation minimizing. An important observation from these constructions is the following. We can transform any moving frame $\boldsymbol{Q}(s)$ into another moving frame

$$\hat{\boldsymbol{Q}}(s) = \boldsymbol{Q}(s)\mathrm{Rot}\left(\boldsymbol{e}_1, \phi(s)\right) \tag{13.6}$$

of the same base curve $\boldsymbol{p}$. Intuitively, the function $\phi(s)$ determines the angular displacement between a base moving frame $\boldsymbol{Q}$ and a resulting moving frame $\hat{\boldsymbol{Q}}$.

Similarly to Equation (13.5), the curvatures of the framed curve $\hat{\mathbf{T}}$ can be expressed in terms of the curvatures of $\mathbf{T}$ as

$$\begin{bmatrix} \hat{\kappa_g} \\ \hat{\kappa_n} \\ \hat{\tau_r} \end{bmatrix} = \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 & 0 \\ \sin(\phi) & \cos(\phi) & 0 & 0 \\ 0 & 0 & 1 & \phi' \end{bmatrix} \begin{bmatrix} \kappa_g \\ \kappa_n \\ \tau_r \\ 1 \end{bmatrix}. \tag{13.7}$$

A popular approach to approximate the RMF uses this construction by applying the function

$$\phi(s) = -\int \tau ds$$

to the Frenet frame by numerical integration [37].

If $\mathbf{T}$ is at least G^n continuous and ϕ is at least C^n continuous, the resulting framed curve $\hat{\mathbf{T}}$ is in G^n, see Equation (13.6). If only a curve $\boldsymbol{p}$ is given, a suitable base moving frame $\boldsymbol{Q}$ needs to be used to construct $\mathbf{T}$ first. Due to Proposition 8.2, any RMF can be used as such a base moving frame. For curves $\boldsymbol{p}$ which admit sufficiently small pitch angles α, the Euler angle frame given by Equations (13.4) and (13.3) with $\gamma = 0$ can also be used. In this case $\phi(s) = \gamma(s)$ holds and the classical layouting scheme is replicated. In their work, Pombo et al. [71] use the Frenet frame as a base, although they do not mention how inflections are handeled in their framework. Since sections with $\kappa = 0$ and inflections are rather common in track layout design, we discourage the use of the Frenet frame as a base frame for the above scheme.

Choosing other framed curves $\mathbf{T}$ as input is also possible, including framed curves constructed by the same method. In fact, if the function ϕ is of the form

$$\phi(s) = \sum_i \phi_i(s)$$

we can apply each individual function ϕ_i separately and regardless of order. Due to the commutativity of rotations about the same axis this becomes obvious when looking at how the framed curve $\hat{\mathbf{T}}$ is constructed.

13.4 Polynomial Moving Frames

In an interactive setting, the function $\phi(s)$ may be entirely user defined. We can use polynomial splines to give a designer enough descriptive power to interactively construct moving frames using the above method. In our case, the user is already familiar with interpolating polynomial splines from Chapter 11 and thus we adapt the same principles for constructing moving frames. As with constructing the curve $\boldsymbol{p}$, the user is only interested in manipulating the geometric properties of the resulting moving frame independently of its parametrization. Hence, we will first approximate the arc length parametrization of $\boldsymbol{p}$ and define ϕ with respect

to the arc length. Besides the target orientation given by the normals $\boldsymbol{u}$ and $\boldsymbol{v}$, the user is able to additionally specify a desired relative torsion τ_r or change in relative torsion τ_r' at specific points s_i along $\boldsymbol{p}$.

Given a curve $\boldsymbol{p}(s)$ and a base moving frame $\boldsymbol{Q}(s)$, e.g., the RMF, the algorithm that computes the polynomial spline $\phi(s)$ and $\hat{\boldsymbol{Q}}(s) = \boldsymbol{Q}(s)\mathrm{Rot}\left(\boldsymbol{e}_1, \phi(s)\right)$ works as follows.

1. For all $n+1$ nodes $0 = s_0 < s_i < s_n = s, 0 < i < n$, compute the rotation angle ϕ_i needed to observe the target orientation given by the user. If the user additionally specifies a desired relative torsion $\tau_{r,i}$ or change in relative torsion $\tau_{r,i}'$, also compute

   $$\phi_i' = \tau_{r,i} - \tau_r(s_i)$$

   and

   $$\phi_i'' = \tau_{r,i}' - \tau_r'(s_i).$$

2. Given the parameters s_i and the target values ϕ_i, ϕ_i', ϕ_i'', choose a knot selection scheme as introduced in Section 11.2 and set up the system of linear equations $\boldsymbol{NC} = \boldsymbol{X}$ as introduced in Section 5.8.

3. The solution $\boldsymbol{C}$ consists of the control points of the resulting polynomial spline $\phi(s)$ which satisfies all user defined constraints by construction.

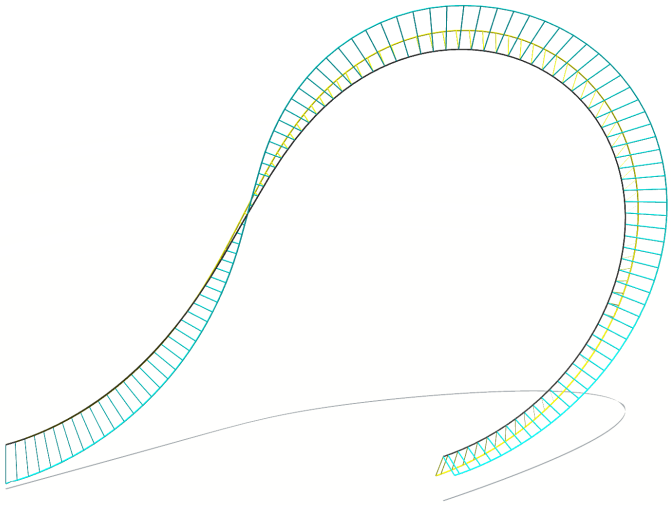
A step-by-step visualization of the above algorithm is shown in both Figures 13.2 and 13.3, once using the RMF as a base and once using the Euler angle frame with $\gamma = 0$ as a base. Several things can be noted.

- Since we use the RMF as a base in Figure 13.2, the derivative of the plot shown in Figure 13.2 is $\hat{\tau_r}$.

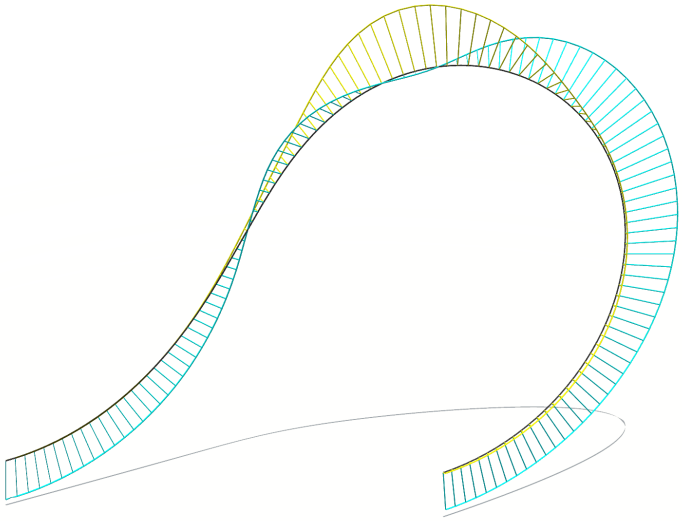
- Since parts of the given curve have a large pitch angle α combined with a sharp change in yaw angle θ, the Euler angle frame admits significant relative torsion $\tau_r = -\theta\sin(\alpha)$. The resulting relative torsion $\hat{\tau_r}$ has unnatural bumps.

- The resulting framed curves $\hat{\mathbf{T}}$ of Figure 13.2 and 13.3 are only equal at the nodes s_i. Due to the different base moving frames, the resulting polynomial splines $\phi(s)$ and thus the framed curves $\hat{\mathbf{T}}$ are *not* equal.

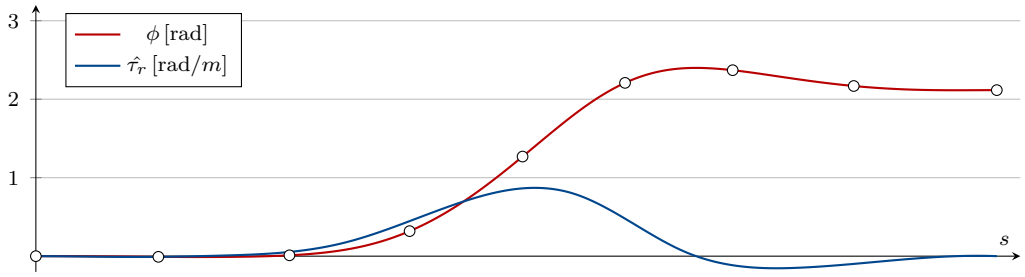
Remark: We noted earlier, that users should only be concerned about at which point along the curve the final moving frame observes a specific orientation. However, they should not need to think about in which direction the moving frame twists or



a) Input curve $\boldsymbol{p}$ and its RMF visualized by showing the curvature combs of κ_n in cyan and κ_g in yellow.

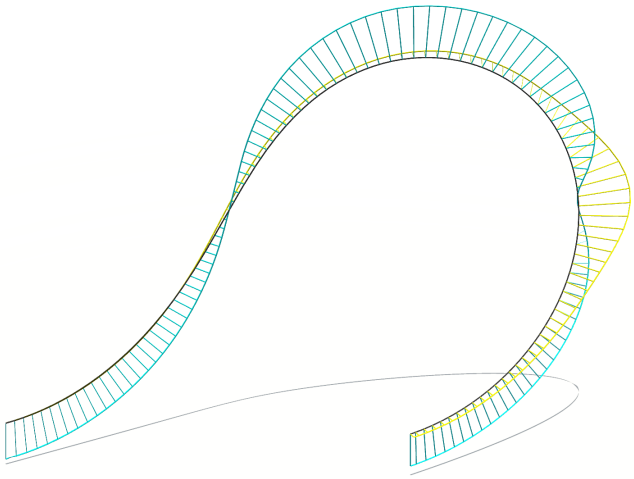


b) Output framed curve $\hat{\mathbf{T}}$. the relative torsion $\hat{\tau}_r$ is visualized as the twist of the curvature combs about $\boldsymbol{t}$.

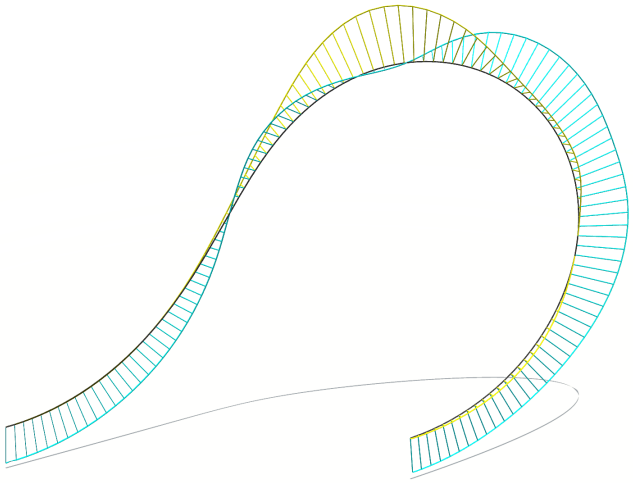


c) Function $\phi(s)$ and its interpolation constraints $\phi_i(s_i)$ of the transformation from $\mathbf{T}$ to $\hat{\mathbf{T}}$ above.

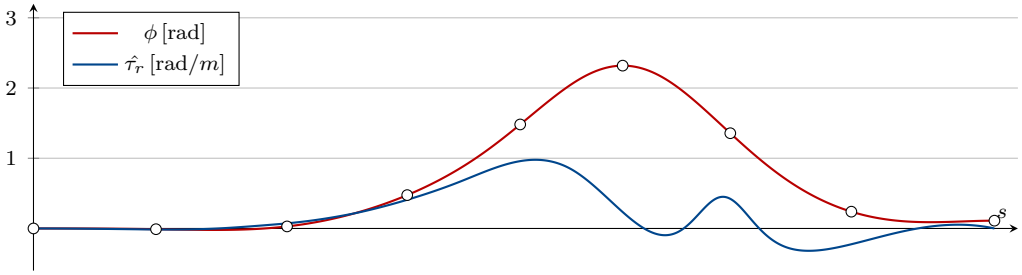
Figure 13.2: Moving frame generation using the RMF as a base.



a) Input curve $\boldsymbol{p}$ and its Euler angle based frame. Note, that κ_n lies in the plane spanned by $\boldsymbol{t}$ and $\boldsymbol{e}_3$, and κ_g is always horizontal.



b) Output framed curve $\hat{\mathbf{T}}$ given the same interpolation conditions as in Figure 13.2.



c) Function $\phi(s)$ and its interpolation constraints $\phi_i(s_i)$ and the resulting relative torsion $\hat{\tau}_r$.

Figure 13.3: Moving frame generation using the Euler angle frame as a base.

how many full twists the frame performs to reach said orientation. We resolve this ambiguity by limiting the accumulated rotation angle between any two nodes to

$$-\pi < \int_{s_i}^{s_{i+1}} \hat{\tau}_r ds \leq \pi.$$

This policy of choosing the minimal rotation required to arrive at the next target orientation is in line with our goal to keep the relative torsion of the final frame within acceptable limits. If users are not satisfied with the resulting moving frame, they may specify additional interpolation constraints to arrive at their intended shape.

13.5 Optimization Techniques

So far, it is entirely the designer's task to ensure that the moving frame $\hat{\boldsymbol{Q}}$ is fair. Apart from the choice of polynomial and base moving frame the frame generation algorithm does not explicitly assist the designer in maximizing the fairness of $\hat{\boldsymbol{Q}}$. Similarly to how the use of curves of minimum energy requires less inputs by a user to model curves of similar fairness, employing constrained optimization techniques to construct fair moving frames could drastically reduce the number of interpolation constraints a designer has to specify. We can convert the interpolation problem above to a constrained optimzation problem by specifying an objective function that is to be minimized by ϕ. Interestingly, due to our chosen parametrization of the polynomial spline of degree d according to the arc length of the base curve $\boldsymbol{p}$, the integral

$$\int \frac{d^{d-1}\phi(s)}{ds^{d-1}}^2 ds$$

is naturally minimized along the curve, see Equation (11.1). Consequently, if we choose the RMF as a base moving frame, the moving frame $\hat{\boldsymbol{Q}}$ generated by above interpolation algorithm naturally minimizes

$$\int \frac{d^{d-2}\hat{\tau}_r(s)}{ds^{d-2}}^2 ds$$

subject to the user defined interpolation constraints. However, in the context of layout design, we have already established that minimizing these geometric quantities is not sufficient. As already mentioned, we seek to also reduce the lateral accelerations induced when traversing the framed curve $\mathbf{T}$. This is achieved by orienting the track in such a way that the normal $\boldsymbol{v}$ points in the direction of the inertial acceleration vector $\boldsymbol{a}$. The normal acceleration

$$a_z = \langle \boldsymbol{a}, \boldsymbol{v} \rangle$$

and lateral acceleration

$$a_y = \langle \boldsymbol{a}, \boldsymbol{u} \rangle$$

can also be expressed in terms of the optimal angle ϕ_{opt} and $a_{yz} := \sqrt{{a_y}^2 + {a_z}^2}$ as

$$\begin{aligned} a_y &= a_{yz} \sin(\phi - \phi_{\text{opt}}), \\ a_z &= a_{yz} \cos(\phi - \phi_{\text{opt}}). \end{aligned}$$

If we assume that the optimal angular displacement ϕ_{opt} is small, we can use small angle approximations and formulate the weighted linear least squares problem

$$\begin{array}{ll} \underset{\boldsymbol{C}}{\text{minimize}} & \left\| \boldsymbol{W}\boldsymbol{N}'\boldsymbol{C} - \boldsymbol{W}\boldsymbol{\phi}_{\text{opt}} \right\|^2 \\ \text{subject to} & \boldsymbol{N}\boldsymbol{C} = \boldsymbol{X} \end{array} \tag{13.8}$$

with $\boldsymbol{N}$ and $\boldsymbol{C}$ being as above,

$$\begin{aligned} \boldsymbol{W} &= \begin{bmatrix} a_{yz}(s_0) & a_{yz}(s_1) & \dots & a_{yz}(s_k) \end{bmatrix}, \\ \boldsymbol{\phi}_{\text{opt}} &= \begin{bmatrix} \phi_{\text{opt}}(s_0) & \phi_{\text{opt}}(s_1) & \dots & \phi_{\text{opt}}(s_k) \end{bmatrix}^{\mathsf{T}} \end{aligned}$$

and $\boldsymbol{N}'$ being the matrix of the B-splines evaluated at s_j, $0 \leq j \leq k$. Similarly to our algorithm for approximating curves of minimum energy introduced in Chapter 12, for the optimization to have any effect, we need to ensure that the polynomial spline has more degrees of freedom than interpolation conditions. However, since the objective function of (13.8) only considers lateral accelerations, the resulting polynomial spline approaches the frame of minimal lateral accelerations if given too many degrees of freedom, and thus may have unbounded relative torsion τ_r. Additionally, the small angle approximation does not hold in general, in particular it heavily penalized angles $\phi = \phi_{opt} \pm k\pi$ for any integer $k \neq 0$, although the resulting lateral accelerations are also minimized for these angles. Especially in the case of airtime hills, such as the ones shown in Figure 7.5, the angle ϕ_{opt} may flip suddenly by $\pm\pi$ due to the zero-crossing of the gravitoinertial accelerations.

Using constrained non-linear optimization methods such as those introduced in Chapter 12 resolves the issues above. By choosing a sophisticated objective function closely related to common design goals such as the one given by Equation (13.2), a human designer only needs to guide the frame generation algorithm in sections of track where the design intent is for some reason not fully in line with the common design goals. In our current implementation, we further adapted Equation (13.2) to penalize lateral accelerations a_y depending on the magnitude and sign of the resulting normal accelerations a_z. From Figure 13.4 we can see that the resulting loss function given $\tau_r = \tau_r' = 0$ roughly approximates the distance from the allowable limits according to the standard ASTM F2291 [5].

An application of this non-linear objective function is depicted in Figure 13.5, along with a chosen function $\phi(s)$ of a human expert in red and the resulting

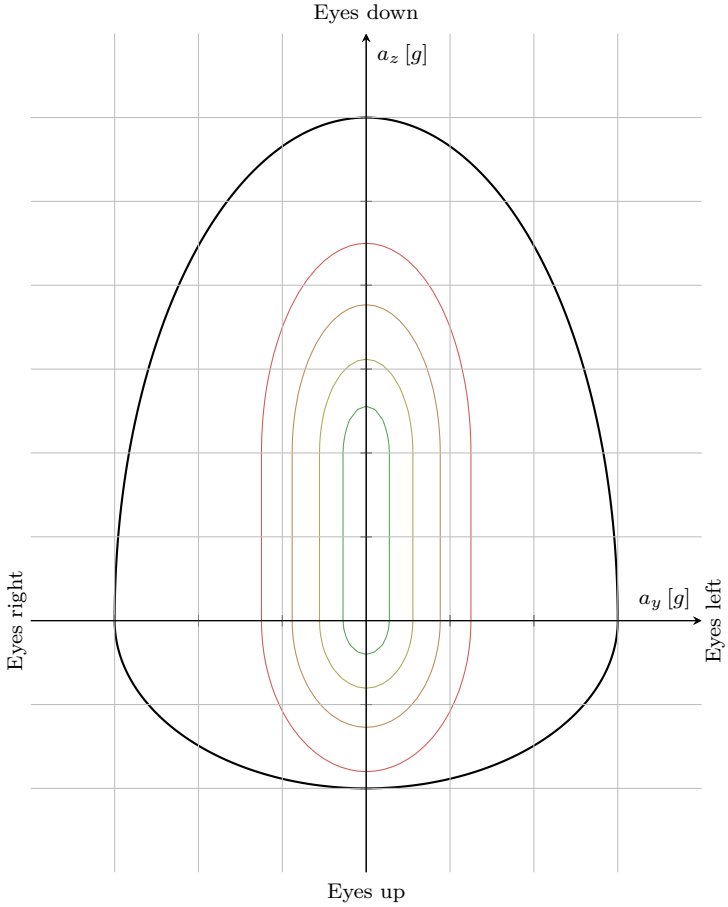


Figure 13.4: Heat map of combined accelerations in the y-z-plane relative to the allowed combined accelerations according to ASTM F2291[5].

function of an unconstrained optimization algorithm that minimizes the objective function in Figure 13.4 in blue. We can immediately identify sections of track, such as $s = 180$ to $s = 280$, in which even small deviations from ϕ_{opt} are heavily penalized as the acceleration a_{yz} is large and even small angular displacements would induce large lateral accelerations. For other sections of track however, such as at around $s = 300$, the magnitude of a_{yz} is very low and thus the angle ϕ may be chosen almost arbitrarily. Since the function $\phi(s)$ may only change the direction of the gravitoinertial accelerations during traversal, but not their magnitude, some portions along the track are naturally more restrictive than others in the choice of an appropriate angle ϕ. This leaves some creative room for human designers to add full revolutions of 2π, such as shown at around $s = 300$, or rotations in the opposite direction of the one indicated by the underlying curve, as can be seen at around $s = 725$. For these situations a human designer needs the ability to specify interpolation constraints to guide the optimization algorithm. These interpolation constraints can be seen as a representation of the design intent, showing sections of track in which the designer accepts a less fair moving frame in order to create exciting experiences for the human patrons.

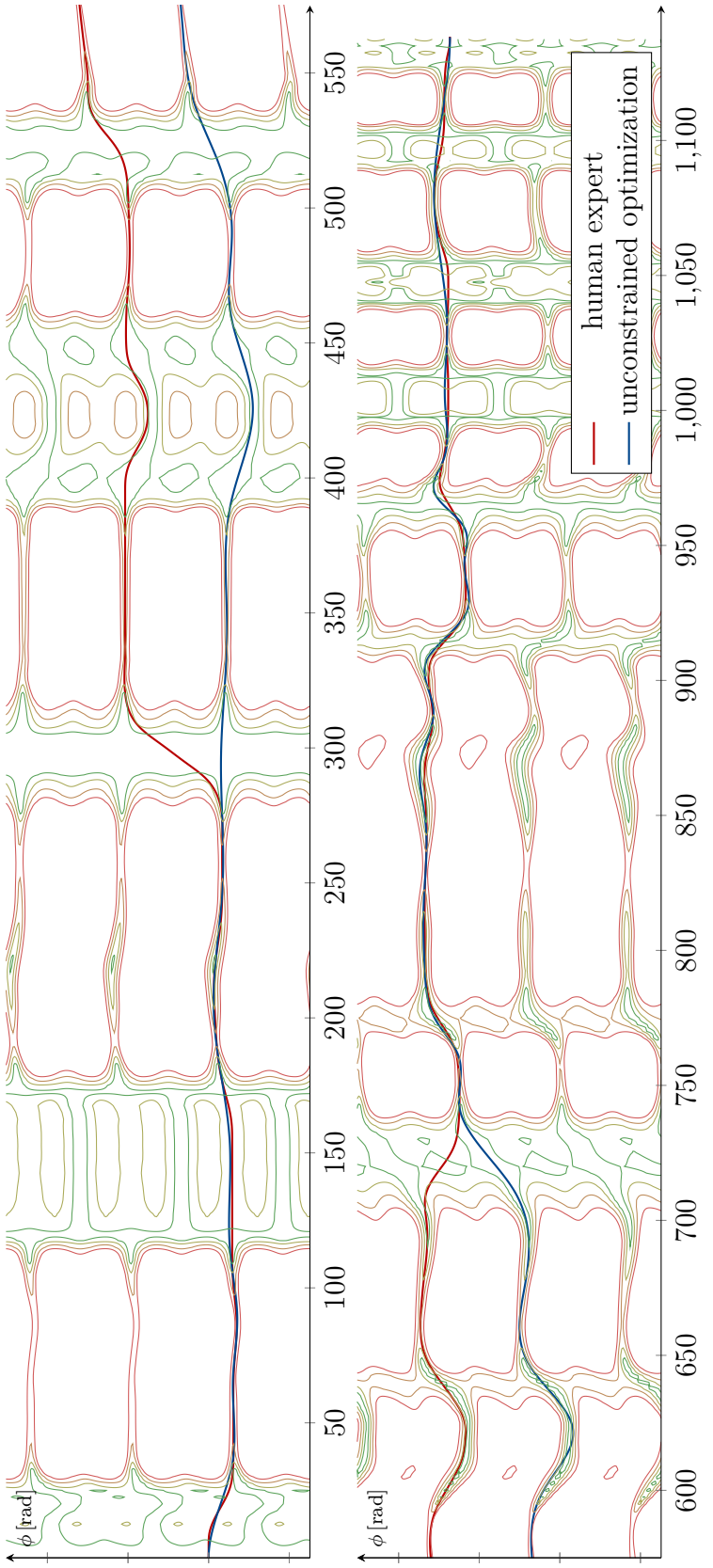


Figure 13.5: Exemplary track and its heat map (from Figure 13.4) and two different functions $\phi(s)$.

At around $s = 125$, $s = 175$ or several times between $s = 1000$ and $s = 1100$ sudden sharp rotations of magnitude $\pm\pi$ would lead to less accumulated error according to the shown heat map. These sections of track correspond to airtime hills with accelerations in the *eyes-up* direction. As shown in Figure 13.4, our chosen objective function penalizes negative a_z more than positive a_z of the same magnitude to reflect human tolerance for accelerations in these directions. However, since our loss function also penalizes high relative torsion τ_r and its derivative, it is a globally better strategy to accept the loss generated by the induced gravitoinertial accelerations in the eyes up direction in these cases. In other cases in which the negative a_z become unreasonably large, accepting the loss from high relative torsion τ_r eventually becomes a better strategy.

For most sections in this example, the output of the unconstrained optimization algorithm and the human designer are very similar, indicating that for significant parts of the curve the designers main objective is to generate a fair moving frame according to the specified objective function. This indicates that such an optimizing frame generation algorithm can greatly reduce the designer's mental load, especially during initial concept design of layouts, during which the curve $\boldsymbol{p}$ is not yet fully developed and therefore subject to a lot of significant modifications. During this design phase, the designer can fully concentrate on editing the base curve $\boldsymbol{p}$ while receiving visual feedback on a reasonably high quality framed curve $\mathbf{T}$. Later in design both the curve $\boldsymbol{p}$ and its moving frame $\boldsymbol{Q}$ are subject to more granular modifications, at which point a designer may specify constraints in order to fine tune the resulting framed curve $\mathbf{T}$.

Conclusion and Future Work

14 Conclusion

In this thesis, we provide a general framework for modeling and evaluating spatial paths which are to be traversed by kinematic systems, such as railroad or roller coaster tracks. Although this field has historically been a field of research in civil engineering, we view the related problems through the lens of differential geometry, geometric design as well as kinematics, which itself provides a lot of new insights. As with most interdisciplinary work, the main contributions of this thesis are therefore the result of linking already established topics and methods of various fields in novel and unique ways.

14.1 Representation of Spatial Tracks

In Chapter 8, we show that *framed curves* are a powerful and general representation for any kind of spatial track. The representation used by the classical layouting scheme relies heavily on several assumptions, such as the existence of a principal orthographic projection and independence of plan view and height profile. Consequently, the singularities created by this layouting scheme need to be circumvented if the underlying application does not satisfy the assumptions. We show that framed curves can be universally used as a representation for spatial tracks, since they have no singularities and do not rely on any assumptions that may limit their applicability. Additionally, we show how framed curves may be used to solve several common geometric problems in the context of layout design, such as evaluating *sweep surfaces* or *offset curves*, which allows us to apply the *heartline principle* in a natural manner.

14.2 Modeling Motion along Spatial Tracks

During kinematic analysis, framed curves can be used as a generalized joint. In Chapter 9, we derive the equations of motion along framed curves, which can be implemented for use in existing general multi body dynamics libraries. This work focusses on solving the *inverse kinematics* problem along spatial tracks. That is, given a system of rigid bodies, which may have multiple connections to the track, and, given the position with respect to arc length of said system s, determine the rigid body motion $\mathbf{T}_i(s)$ of all relevant bodies. We show that separating kinematic and kinetic analysis has several advantages in the context of layout design, since

a path may be traversed with different velocities, while the macroscopic spatial trajectories of the relevant bodies remains the same.

Given theoretical ground truth data from a full inverse kinematic model as well as real-world measurements of operational roller coasters, we additionally derive a simplified kinematic model that yields sufficiently accurate results with significant speed-up relative to the full inverse kinematics approach. This demonstrates that approximating train kinematics and kinetics in an online fashion for timely feedback in an interactive CAGD environment with sufficient accuracy is viable, facilitating the design process of the underlying spatial tracks.

14.3 Design and Modeling of Framed Curves

Since layout design for roller coasters is a highly creative, yet also very technical endeavour, it is important to develop interactive modeling algorithms that generate high quality framed curves. In this thesis, we introduce two different approaches. Using the *curvature based curve definition* as introduced in Chapter 10 a designer may specify framed curves by their differential properties, such as curvatures or gravitoinertial accelerations during traversal. In this work, we focus on generating framed curves by numerical integration of input functions that consist of several user defined transition elements. While it is trivially easy to generate framed curves with highly desirable curvature profiles in this manner, satisfying positional constraints using this method is difficult. Additionally, since our method relies on integration, local changes to the input functions potentially affect the whole curve.

A second approach introduced in Chapters 11 to 13 splits the problem of generating a framed curve up into generating a fair base curve $\boldsymbol{p}$ and a fair moving frame $\boldsymbol{Q}$ along the base curve. In Chapter 11 we build upon classical polynomial spline interpolation techniques by introducing two novel parametrization schemes that result in fairer curvature profiles than state of the art parametrizations. In Chapter 12 we propose an algorithm that approximates energy minimizing curves using polynomial splines. Of the existing and well researched energies that we investigate, we identify the *curvature variation* energy as being the best fitting for our application. Last, we expand the notion of *curves of minimum curvature variation* to also consider path velocity, resulting in *curves of minimum jerk*, and show that these curves admit highly desirable properties in our context.

In Chapter 13, we show how moving frames may be modeled in an interactive CAGD environment. Here, we make use of the fact that any moving frame may be transformed into another moving frame by applying a sufficiently continuous function $\phi(s)$. By letting a human designer define $\phi(s)$ given established modeling methods such as polynomial spline interpolation, we set up a natural and highly efficient modeling environment. We show that the *rotation minimizing frame* (RMF) can be used most universally as a base moving frame, since it guarantees maximal order of continuity while minimizing the relative torsion of the resulting framed curve. Finally, we show that (non-)linear optimization techniques can be used to

assist a designer in defining $\phi(s)$ by reducing the amount of inputs the human designer has to perform.

14.4 Implementation

As part of our research, we develop a CAGD software at the company Mack Rides, which uses a WYSIWYG approach and an intuitive modeling interface, while implementing the algorithms and methods that are introduced throughout this thesis. Using this software, we achieve response times for single modifications of the track layout of less than a second, as opposed to several dozens of seconds using the previously used traditional approach on conventional CAD systems. At the time of this writing, a total of 36 roller coaster projects with novel layouts and 6 retrofits or major modfications of of existing roller coaster layouts have already realized using this software, demonstrating the applicability of the algorithms presented in this thesis across a wide range of roller coaster types, from smaller family rides to record braking thrill rides.

In contrast to the classical layouting scheme used in conventional layout design tools, our CAGD software uses a wholistic approach in which the designer is able to manipulate spatial framed curves directly and receive timely feedback on the geometric and dynamic properties of the resulting track. In combination with the WYSIWYG interface, this facilitates the creativity of the designers as they are more likely to explore the respective design space. Despite this exploration of the design space, the average time for a designer to create a feasible conceptual layout at the company Mack Rides has significantly reduced by the introduction of our specialized software and continuously decreased during its further development, as the respective curve generation algorithms as introduced in this thesis have continuously helped to reduce manual optimization of the underlying curve geometry.

In addition to calculating and evaluating the resulting track geometry, our software computes the expected ride dynamics in an online fashion, presenting them along with the track geometry to the designer. This allows the designer to immediately check the constraints imposed by the dynamics. Since traditional CAGD software packages do not have such inbuilt capabilities, designers at the company Mack Rides previously had to use external tools to evaluate the resulting dynamics of a layout, resulting in a less integrated and therefore slower and more error prone workflow.

15 Future Work

This thesis achieves the goals set in Chapter 2. Still, further research in any of the related sub fields may give new insights and improve upon our findings. Firstly, a more rigorous treatise of the differential geometry of framed curves that is not restricted to its applications in track layout design could not only provide new insights, but also spread awareness about these rarely discussed geometric objects. Secondly, further work could use the base idea of the *simplified kinematics* approach introduced in Section 9.2 to model more complex kinematic systems. Further integration of framed curves into existing multi body dynamics libraries could both decrease computation time and reduce the noise due to numerical differentiation when simulating motion along such paths. Last, extending Levien's approach of finding non linear interpolants given specific curvature transitions in the plane [55, Chapter 8] to spatial curves or even framed curves may prove to be a powerful alternative to our approach of solving a global constrained optimization problem for polynomial splines. By specifying frames to be interpolated by segments of n-parameter non linear framed curves, such a method could potentially fuse the two modeling methods we introduced in this thesis together. Since the algorithms proposed by Levien for planar approximations of non linear splines are already quite sophisticated, we suspect that their extensions to spatial curves and – eventually – framed curves would be quite involved and thus could potentially be unfeasible in an interactive setting without extensive optimization effort.

Although many of the algorithms introduced in this work have already been used in the design of roller coasters, conclusive evaluation and quantification of their isolated impact on the final product proves difficult due to the overall complexity of roller coaster projects. We plan to continuously refine the methods and respective algorithms introduced in this thesis in the future based on the results of acceleration measurements on existing roller coasters. Unfortunately, the time between completing a layout design and the finished and operational product is rather long, such that this kind of feedback has some inevitable delay. Therefore, theoretical considerations such as the ones found throughout this thesis remain an important tool in refining the design process and associated CAGD software.

Appendix

A Approximating Energy Minimizing Curves

The MEC approximation of Equation (12.1) is given by

$$\begin{aligned}
\int \kappa^2 ds &\approx \sum \kappa^2 \Delta s \\
&\approx \sum \left(\frac{\|\boldsymbol{p}' \times \boldsymbol{p}''\|}{\|\boldsymbol{p}'\|^3} \right)^2 \cdot \boldsymbol{p}' \Delta u \\
&= \sum \left(\frac{\|\boldsymbol{p}' \times \boldsymbol{p}''\|}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{1.25}} \right)^2 \Delta u \\
&= \sum \left(\frac{\boldsymbol{p}' \times \boldsymbol{p}''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{1.25}} \sqrt{\Delta u} \right)^2 .
\end{aligned}$$

The MVC approximation of Equation (12.6) is given by

$$\begin{aligned}
\boldsymbol{b}\kappa &= \frac{\boldsymbol{p}' \times \boldsymbol{p}''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{1.5}}, \\
\frac{d}{ds}(\boldsymbol{b}\kappa) &= \boldsymbol{b}\kappa' - \boldsymbol{n}\tau\kappa \\
&= \left(\frac{\boldsymbol{p}' \times \boldsymbol{p}'''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{1.5}} - \frac{3(\boldsymbol{p}' \times \boldsymbol{p}'') \langle \boldsymbol{p}', \boldsymbol{p}'' \rangle}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{2.5}} \right) \frac{du}{ds} \\
&= \frac{\boldsymbol{p}' \times \boldsymbol{p}'''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{2}} - \frac{3(\boldsymbol{p}' \times \boldsymbol{p}'') \langle \boldsymbol{p}', \boldsymbol{p}'' \rangle}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{3}}, \\
\int \left(\frac{d}{ds}(\boldsymbol{b}\kappa) \right)^2 ds &= \int {\kappa'}^2 + \tau^2 \kappa^2 ds \\
&\approx \sum \left(\left(\frac{\boldsymbol{p}' \times \boldsymbol{p}'''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{1.75}} - \frac{3(\boldsymbol{p}' \times \boldsymbol{p}'') \langle \boldsymbol{p}', \boldsymbol{p}'' \rangle}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{2.75}} \right) \sqrt{\Delta u} \right)^2 .
\end{aligned}$$

The space MVC approximation of Equation (12.7) is given by

$$\begin{aligned}
\kappa' &= \left\langle \frac{d}{ds}(\boldsymbol{b}\kappa), \boldsymbol{b} \right\rangle \\
&= \frac{\langle \boldsymbol{p}' \times \boldsymbol{p}''', \boldsymbol{p}' \times \boldsymbol{p}'' \rangle}{\|\boldsymbol{p}' \times \boldsymbol{p}''\| \cdot \langle \boldsymbol{p}', \boldsymbol{p}' \rangle^2} - \frac{3\|\boldsymbol{p}' \times \boldsymbol{p}''\| \langle \boldsymbol{p}', \boldsymbol{p}'' \rangle}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^3},
\end{aligned}$$

$$\tau = \frac{\langle \boldsymbol{p}' \times \boldsymbol{p}'', \boldsymbol{p}''' \rangle}{\langle \boldsymbol{p}' \times \boldsymbol{p}'', \boldsymbol{p}' \times \boldsymbol{p}'' \rangle},$$

$$\begin{aligned}\tau' = &\frac{\langle \boldsymbol{p}' \times \boldsymbol{p}'', \boldsymbol{p}'''' \rangle}{\langle \boldsymbol{p}' \times \boldsymbol{p}'', \boldsymbol{p}' \times \boldsymbol{p}'' \rangle \langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{0.5}} \\ &- \frac{\langle \boldsymbol{p}' \times \boldsymbol{p}'', \boldsymbol{p}''' \rangle 2 (\boldsymbol{p}' \times \boldsymbol{p}'') (\boldsymbol{p}' \times \boldsymbol{p}''')}{\langle \boldsymbol{p}' \times \boldsymbol{p}'', \boldsymbol{p}' \times \boldsymbol{p}'' \rangle^2 \langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{0.5}},\end{aligned}$$

$$\int {\kappa'}^2 + {\tau'}^2 \, ds \approx \sum \left({\kappa'}^2 \sqrt{\Delta u} \right)^2 + \sum \left({\tau'}^2 \sqrt{\Delta u} \right)^2$$

For the calculation of the scale-invariant metrics, we additionally need to approximate the arc length by

$$\begin{aligned}\int ds &\approx \sum \|\boldsymbol{p}'\| \, \Delta u \\ &= \sum \left(\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{0.25} \sqrt{\Delta u} \right)^2.\end{aligned}$$

The energies are of the form (12.11). In order to apply the *Levenberg-Marquardt algorithm*, we need to linearize them with respect to the spline control points. In the following, we derive the respective Jacobian matrices.

In order to calculate the respective Jacobian matrices below, we make extensive use of the product, quotient and chain rules, as well as use the identities

$$\begin{aligned}\frac{\partial \boldsymbol{p}^{(n)}}{\partial^{(n)} \boldsymbol{c}_i} &= \frac{\partial \boldsymbol{N}^{(n)} \boldsymbol{C}}{\partial^{(n)} \boldsymbol{c}_i} \\ &= N_i^{(n)}, \\ \boldsymbol{a} \times \boldsymbol{b} &= -\boldsymbol{b} \times \boldsymbol{a}\end{aligned}$$

and

$$\boldsymbol{a} \times \boldsymbol{b} = \boldsymbol{A}_\times \boldsymbol{b}$$

where

$$\boldsymbol{A}_\times = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}.$$

In the MEC case, the Jacobian matrix $\boldsymbol{J}_{\text{MEC}}$ consists of 3×3 blocks

$$\boldsymbol{B}_i = N_i' \boldsymbol{D}_i + \boldsymbol{W}_{\times,i},$$

containing a skew symmetric part

$$\boldsymbol{w}_i = \frac{\boldsymbol{p}'' N_i' - \boldsymbol{p}' N_i''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{1.25}}$$

and a part resulting from an outer product

$$\boldsymbol{D}_i = -2.5\frac{(\boldsymbol{p}' \otimes (\boldsymbol{p}' \times \boldsymbol{p}''))}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{2.25}},$$

resulting in

$$\boldsymbol{J}_{\text{MEC}} = \begin{bmatrix} \boldsymbol{B}_1(u_1) & \boldsymbol{B}_2(u_1) & \boldsymbol{B}_3(u_1) & & \boldsymbol{B}_n(u_1) \\ \boldsymbol{B}_1(u_2) & \boldsymbol{B}_2(u_2) & \boldsymbol{B}_3(u_2) & \cdots & \boldsymbol{B}_n(u_2) \\ \boldsymbol{B}_1(u_3) & \boldsymbol{B}_2(u_3) & \boldsymbol{B}_3(u_3) & & \boldsymbol{B}_n(u_3) \\ & \vdots & & \ddots & \vdots \\ \boldsymbol{B}_1(u_k) & \boldsymbol{B}_2(u_k) & \boldsymbol{B}_3(u_k) & \cdots & \boldsymbol{B}_n(u_k) \end{bmatrix}.$$

The Jacobian matrix $\boldsymbol{J}_{\text{MVC}}$ for the curvature variation energy has a similar structure, although its individual components are more complex due to the higher derivatives that are involved. The blocks

$$\boldsymbol{B}_i = N'\boldsymbol{D}_{1,i} + N''\boldsymbol{D}_{2,i} + \boldsymbol{V}_{\times,i}$$

consist of two outer product parts

$$\begin{aligned} \boldsymbol{D}_{1,i} = 16.5&\frac{\langle \boldsymbol{p}', \boldsymbol{p}'' \rangle \, (\boldsymbol{p}' \otimes (\boldsymbol{p}' \times \boldsymbol{p}''))}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{3.75}} \\ - 3.5&\frac{(\boldsymbol{p}' \otimes (\boldsymbol{p}' \times \boldsymbol{p}'''))}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{2.75}} \\ - 3&\frac{(\boldsymbol{p}'' \otimes (\boldsymbol{p}' \times \boldsymbol{p}''))}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{2.75}}, \\ \boldsymbol{D}_{2,i} = -3&\frac{(\boldsymbol{p}' \otimes (\boldsymbol{p}' \times \boldsymbol{p}''))}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{2.75}} \end{aligned}$$

and the skew symmetric part $\boldsymbol{V}_{\times,i}$

$$\boldsymbol{v}_i = \frac{\boldsymbol{p}''' N_i' - \boldsymbol{p}' N_i'''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{1.75}} - 3 \langle \boldsymbol{p}', \boldsymbol{p}'' \rangle \frac{\boldsymbol{p}'' N_i' - \boldsymbol{p}' N_i''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{2.75}}$$

resulting in

$$\boldsymbol{J}_{\text{MVC}} = \begin{bmatrix} \boldsymbol{B}_1(u_1) & \boldsymbol{B}_2(u_1) & \boldsymbol{B}_3(u_1) & & \boldsymbol{B}_n(u_1) \\ \boldsymbol{B}_1(u_2) & \boldsymbol{B}_2(u_2) & \boldsymbol{B}_3(u_2) & \cdots & \boldsymbol{B}_n(u_2) \\ \boldsymbol{B}_1(u_3) & \boldsymbol{B}_2(u_3) & \boldsymbol{B}_3(u_3) & & \boldsymbol{B}_n(u_3) \\ & \vdots & & \ddots & \vdots \\ \boldsymbol{B}_1(u_k) & \boldsymbol{B}_2(u_k) & \boldsymbol{B}_3(u_k) & \cdots & \boldsymbol{B}_n(u_k) \end{bmatrix}$$

The Jacobian matrix of the space MVC can be derived analaogously. Since it is of least relevance in this thesis and additionally its explicit representation is the most unwieldy, we chose to omit it here.

The Jacobian matrix $\boldsymbol{J}_s$ for the arc length is comparatively simpler, with blocks

$$\boldsymbol{B}_i = \left(0.5 N_i' \frac{\boldsymbol{p}'}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{0.75}}\right)^\mathsf{T}$$

of size 1×3, resulting in

$$\boldsymbol{J}_s = \begin{bmatrix} \boldsymbol{B}_1(u_1) & \boldsymbol{B}_2(u_1) & \boldsymbol{B}_3(u_1) & & \boldsymbol{B}_n(u_1) \\ \boldsymbol{B}_1(u_2) & \boldsymbol{B}_2(u_2) & \boldsymbol{B}_3(u_2) & \cdots & \boldsymbol{B}_n(u_2) \\ \boldsymbol{B}_1(u_3) & \boldsymbol{B}_2(u_3) & \boldsymbol{B}_3(u_3) & & \boldsymbol{B}_n(u_3) \\ & \vdots & & \ddots & \vdots \\ \boldsymbol{B}_1(u_k) & \boldsymbol{B}_2(u_k) & \boldsymbol{B}_3(u_k) & \cdots & \boldsymbol{B}_n(u_k) \end{bmatrix}.$$

MJC approximation

In Section 12.5, we introduce curves of minimum jerk (MJC). The respective approximation of Equation (12.12) is given by

$$\begin{aligned} v^2 \boldsymbol{b} \kappa &= v^2 \frac{\boldsymbol{p}' \times \boldsymbol{p}''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{1.5}}, \\ \frac{d}{dt}\left(v^2 \boldsymbol{b} \kappa\right) &= \boldsymbol{b}\left(2\dot{v} v \kappa + v^3 \kappa'\right) - \boldsymbol{n} v^3 \tau \kappa \\ &= 2\dot{v} v \frac{\boldsymbol{p}' \times \boldsymbol{p}''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{1.5}} + v^3 \left(\frac{\boldsymbol{p}' \times \boldsymbol{p}'''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{2}} - \frac{3\left(\boldsymbol{p}' \times \boldsymbol{p}''\right)\langle \boldsymbol{p}', \boldsymbol{p}'' \rangle}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{3}}\right), \\ \int \left(\frac{d}{dt}\left(v \boldsymbol{b} \kappa\right)\right)^2 dt &= \int \left(2\dot{v} v \kappa + v^3 \kappa'\right)^2 + \left(v^3 \tau \kappa\right)^2 dt \\ &\approx \sum \left(2\dot{v}\sqrt{v \Delta u} \frac{\boldsymbol{p}' \times \boldsymbol{p}''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{1.25}} \right. \\ &\quad \left. + v^{2.5}\sqrt{\Delta u}\left(\frac{\boldsymbol{p}' \times \boldsymbol{p}'''}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{1.75}} - \frac{3\left(\boldsymbol{p}' \times \boldsymbol{p}''\right)\langle \boldsymbol{p}', \boldsymbol{p}'' \rangle}{\langle \boldsymbol{p}', \boldsymbol{p}' \rangle^{2.75}}\right)\right)^2 \end{aligned}$$

since

$$\Delta t \approx \frac{\boldsymbol{p}' \Delta u}{v}.$$

To construct the respective Jacobian matrix, we essentially have to combine the matrices of the MEC and MVC with the factors $2\dot{v}\sqrt{v}$ and $v^{2.5}$ respectively. The resulting Blocks $\boldsymbol{B}_i$ are of the form

$$\boldsymbol{B}_i = 2\dot{v}\sqrt{v}\left(N_i' \boldsymbol{D}_{\boldsymbol{i}} + \boldsymbol{W}_{\times,i}\right) + v^{2.5}\left(N' \boldsymbol{D}_{1,i} + N'' \boldsymbol{D}_{2,i} + \boldsymbol{V}_{\times,i}\right),$$

with the individual components being as above.

Bibliography

[1] Karim Abdel-Malek, Jingzhou Yang, Denis Blackmore, and Kenneth Joy. Swept volumes: Fundation, perspectives, and applications. *International Journal of Shape Modeling*, 12:87–127, June 2006.

[2] J.H. Ahlberg, E.N. Nilson, J.L. Walsh, and R. Bellman. *The Theory of Splines and Their Applications: Mathematics in Science and Engineering: A Series of Monographs and Textbooks, Vol. 38*. Number Bd. 38. Elsevier Science, 2016.

[3] Hiroshi Akima. A new method of interpolation and smooth curve fitting based on local procedures. *J. ACM*, 17(4):589602, October 1970.

[4] J. Alan Adams. The intrinsic method for curve definition. *Computer-Aided Design*, 7(4):243–249, 1975.

[5] ASTM F2291. *Standard Practice for Design of Amusement Rides and Devices*. ASTM International, 2019.

[6] Tim Baldwin. Roller Coaster Database. `https://rcdb.com/15368.htm`, 2018. [Online; accessed 01-February-2022].

[7] Beggs, J.S. *Kinematics*. Hemisphere Publishing Corporation, 1983.

[8] Garrett Birkhoff and Carl R. De Boor. Piecewise polynomial interpolation and approximation. *Approximation of functions*, pages 164–190, 1965.

[9] Richard L. Bishop. There is more than one way to frame a curve. *The American Mathematical Monthly*, 82(3), March 1975.

[10] Jules Bloomenthal. Calculation of reference frames along a space curve. December 1990.

[11] A. E. Bloss. Der Übergangsbogen mit geschwungener Überhöhungsrampe. *Organ für Fortschritte des Eisenbahnwesens, Neue Folge*, 73:15, 1936.

[12] Wolfgang Boehm and Hartmut Prautzsch. *Geometric Concepts for Geometric Design*. Peters, 1994.

[13] Alexander Brook, Alfred Bruckstein, and Ron Kimmel. On similarity-invariant fairness measures. In *Scale Space and PDE Methods in Computer Vision*, pages 456–467, April 2005.

[14] Alfred M. Bruckstein and Arun N. Netravali. On minimal energy trajectories. *Computer Vision, Graphics, and Image Processing*, 49(3):283–296, 1990.

[15] Guido H. Brunnett. Properties of minimal-energy splines. *Curve and Surface Design*, pages 3–22, January 1992.

[16] Guido H. Brunnett and Johannes Kiefer. Interpolation with minimal-energy splines. *Computer-Aided Design*, 26(2):137 – 144, 1994.

[17] Maurice G. Cox. The numerical evaluation of B-splines. *IMA Journal of Applied mathematics*, 10(2):134–149, October 1972.

[18] G. Darboux. *Leçons sur la théorie générale des surfaces et les applications géométriques du calcul infinitésimal: ptie. Lignes géodésoqies et courbure géodésique*. Cours de géometrie de la faculté des sciences. Gauthier-Villars, 1894.

[19] Dassault Systemes Deutschland GmbH. SIMULIA Simpack. `http://www.simpack.com`, 2019.

[20] J.R. Davis, R. Johnson, and J. Stepanek. *Fundamentals of Aerospace Medicine*. Lippincott Williams & Wilkins, 2008.

[21] Carl de Boor. On calculating with B-splines. *Journal of Approximation Theory*, 6(1):50 – 62, 1972.

[22] T. D. DeRose and B. A. Barsky. Geometric continuity of parametric curves: Three equivalent characterizations. *IEEE Computer Graphics and Applications*, 9(06):60–68, November 1989.

[23] DIN EN 13814-1. *Safety of amusement rides and amusement devices - Part 1: Design and manufacture; German version EN 13814-1:2019*. Deutsches Institut für Normung, 2019.

[24] Manfredo P. do Carmo. *Differentialgeometrie von Kurven und Flächen*. Springer, 1983.

[25] Nicklas Dohrn. *Transformation von Dreiecksnetzen entlang von Raumkurven*. Bachelor's thesis, Karlsruher Institut für Technologie, Karlsruhe, 2019.

[26] Gerald Farin. *Curves and Surfaces for CAGD: A Practical Guide*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 5th edition, 2002.

[27] Rida T. Farouki. *Pythagorean-Hodograph Curves: Algebra and Geometry Inseparable*. Springer Publishing Company, Incorporated, 1st edition, 2007.

[28] Roy Featherstone. *Rigid Body Dynamics Algorithms*. Springer-Verlag, Berlin, Heidelberg, 2007.

[29] Wolfgang Fengler. Einführung in die Trassierung. `https://tu-dresden.de/bu/verkehr/ibv/gvb/ressourcen/dateien/download_gvb/lehrmaterialien/ebb_1/ebb1_06f.pdf`, 2018. [Online; accessed 20-November-2019].

[30] Thomas A. Foley and Gregory M. Nielson. Knot selection for parametric spline interpolation. *Mathematical Methods in Computer Aided Geometric Design*, pages 261–272, 1989.

[31] Marco Frego. Closed form parametrisation of 3d clothoids by arclength with both linear varying curvature and torsion. *Applied Mathematics and Computation*, 421:126907, 2022.

[32] F. Frenet. Sur les courbes à double courbure. *Journal de Mathématiques Pures et Appliquées*, pages 437–447, 1852.

[33] Justin Garanovic. Roller Coaster Database. `https://rcdb.com/14440.htm`, 2018. [Online; accessed 18-March-2022].

[34] Henri Gautier. *Traité de la construction des chemins*. 1721.

[35] R. Gilmore. *Lie Groups, Lie Algebras, and Some of Their Applications*. Dover Books on Mathematics. Dover Publications, 2012.

[36] Dennis Gordt. *Untersuchung, Verifizierung und Optimierung von Berechnungsverfahren/Berechnungsmethodik zur Auslegung von Achterbahnen*. Hochschule Mannheim, August 2007.

[37] H. Guggenheimer. Computing frames along a trajectory. *Comput. Aided Geom. Des.*, 6:77–78, 1989.

[38] Gur Harary and Ayellet Tal. 3D Euler spirals for 3D curve completion. *Computational Geometry*, 45(3):115 – 126, 2012.

[39] Herbert L. Hasslinger. Das Konzept moderner Gleislinienführung. *ETR. Eisenbahntechnische Rundschau*, 2005.

[40] Herbert L. Hasslinger. Gleis mit Übergangsbogen und kräfteminimaler Überhöhungsrampe, 2005.

[41] S. Helgason. *Differential Geometry, Lie Groups, and Symmetric Spaces*. ISSN. Elsevier Science, 1979.

[42] D. Hoag. Considerations of Apollo IMU Gimbal Lock. *MIT Instrumentation Laboratory*, April 1963.

[43] Shunichi Honda and Masatomo Takahashi. Bertrand and Mannheim curves of framed curves in the 3-dimensional Euclidean space. *Turkish Journal of Mathematics*, 44:883–899, 05 2020.

[44] R. Hoppe. Ueber die Darstellung der Curven durch Krümmung und Torsion. *Journal für die reine und angewandte Mathematik*, 60:182–187, 1862.

[45] Ravi Janardan, Michiel Smid, and Debasish Dutta. Geometric and algorithmic aspects of computer-aided design and manufacturing. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 67, 2005.

[46] R.N. Jazar. *Theory of Applied Robotics: Kinematics, Dynamics, and Control (2nd Edition)*. Springer US, 2010.

[47] H. Kasper, W. Schürba, and H. Lorenz. *Die Klotoide als Trassierungselement*. Dümmler, 1968.

[48] Andrés Kecskeméthy and Manfred Hiller. An object-oriented approach for an effective formulation of multibody dynamics. *Computer Methods in Applied Mechanics and Engineering*, 115(3):287–314, 1994.

[49] The Coaster Kings. https://thecoasterkings.com. https://thecoasterkings.com/wp-content/uploads/2018/11/P6190113-e1491513412142.jpg, 2018. [Online: accessed 20-November-2019].

[50] R. Klein. Beitrag zur Gestaltung der Übergangsbögen. *Gleistechnik und Fahrbahnbau*, 13(9/10):97–102, 1937.

[51] Fopke Klok. Two moving coordinate frames for sweeping along a 3D trajectory. *Computer Aided Geometric Design*, 3:217–229, November 1986.

[52] Andreas Korb. https://coasterfriends.de/. https://coasterfriends.de/joomla/magazin/sonstige/2195-steckbrief-achterbahnhersteller-vol2-bolliger-mabillard, 2012. [Online: accessed 14-June-2025].

[53] E. H. Lee and G. E. Forsythe. Variational study of nonlinear spline curves. *SIAM Review*, 15(1):120–133, 1973.

[54] E.T.Y. Lee. Choosing nodes in parametric curve interpolation. *Computer-Aided Design*, 21:363–370, July 1989.

[55] Raphael Linus Levien. *From Spiral to Spline: Optimal Techniques in Interactive Curve Design*. PhD thesis, EECS Department, University of California, Berkeley, Dec 2009.

[56] William Lorensen and Harvey Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *ACM SIGGRAPH Computer Graphics*, 21:163–169, August 1987.

[57] Duane Marden. Roller Coaster Database. https://rcdb.com/3475.htm, 2009. [Online; accessed 14-June-2025].

[58] Duane Marden. Roller Coaster Database. https://rcdb.com/130.htm, 2019. [Online; accessed 20-November-2019].

[59] Christoph Mäurer and Bert Jüttler. Rational approximation of rotation minimizing frames using Pythagorean-hodograph cubics. *J. Geom. Graph*, 3(2):141–159, 1999.

[60] Even Mehlum. Appell and the apple: Nonlinear splines in space. *Mathematical Methods for Curves and Surfaces*, 1981.

[61] Henry P. Moreton. *Minimum Curvature Variation Curves, Networks, and Surfaces for Fair Free-form Shape Design*. PhD thesis, Berkeley, CA, USA, 1992. UMI Order No. GAX93-30652.

[62] Henry P. Moreton and Carlo H. Séquin. Scale-Invariant Minimum-Cost Curves: Fair and Robust Design Implements. In *Computer Graphics Forum*, volume 12, pages 473–484. Wiley Online Library, 1993.

[63] D. Morin. *Introduction to Classical Mechanics: With Problems and Solutions*. Cambridge University Press, 2008.

[64] Andreas Müller. Representation of the kinematic topology of mechanisms for kinematic analysis. *Mechanical Sciences*, 6:137–146, August 2015.

[65] Gregory M. Nielson. Coordinate free scattered data interpolation. *Topics in Multivariate Approximation*, pages 175 – 184, 1987.

[66] Gregory M. Nielson and Thomas A. Foley. A survey of applications of an affine invariant norm. *Mathematical Methods in Computer Aided Geometric Design*, pages 445 – 467, 1989.

[67] Alexander Bernhard Nottbeck. *Untersuchungen zu Auswirkungen von Geschwindigkeitserhöhungen auf Bahnstrecken im Bestand*. Dissertation, Technische Universität München, München, 2016.

[68] Stanley Osher and R. Fedkiw. *The Level Set Methods and Dynamic Implicit Surfaces*, volume 57. May 2004.

[69] Martin Peternell, Helmut Pottmann, Tibor Steiner, and Hongkai Zhao. Swept volumes. *Computer-Aided Design & Applications*, 2, January 2005.

[70] Les Piegl and Wayne Tiller. *The NURBS Book (2Nd Ed.)*. Springer-Verlag New York, Inc., New York, NY, USA, 1997.

[71] João Pombo and Jorge Ambrósio. Modelling tracks for roller coaster dynamics. *Int. J. of Vehicle Design*, 45:470 – 500, January 2007.

[72] Hartmut Prautzsch, Wolfgang Boehm, and Marco Paluszny. *Bézier and B-Spline Techniques*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2002.

[73] Georg Rill. Dynamic Design of Roller Coaster Figures (Dynamischer Entwurf von Achterbahnfiguren), January 1999.

[74] W. Schiemann. *Schienenverkehrstechnik: Grundlagen der Gleistrassierung*. Vieweg+Teubner Verlag, 2013.

[75] Walter Schlemmer et al. *Technische Mechanik: Statik, Dynamik, Fluidmechanik, Festigkeitslehre*. Springer-Verlag, 2009.

[76] I.J. Schoenberg. On spline functions. *Inequalities*, pages 255 – 291, 1967.

[77] W.J. Schroeder, W.E. Lorensen, and S. Linthicum. Implicit modeling of swept surfaces and volumes. In *Proceedings Visualization '94*, pages 40–45, 1994.

[78] Peter Schuhr. Reihenentwicklungen zur Berechnung des Wiener Bogens der ÖBB. *VGI - die Zeitschrift*, 94(4):213 – 217, 2006.

[79] Peter Schuhr. *Übergangsbogen für Bahnen mit hohen Fahrgeschwindigkeiten*. Deutsche Geodätische Kommission bei der Bayerischen Akademie der Wissenschaften Reihe B, Angewandte Geodäsie H. Nr. 314. Verl. der Bayerischen Akademie der Wissenschaften, München, 2007.

[80] Hans-Rudolf Schwarz and Norbert Köckler. *Numerische Mathematik*. Springer-Verlag, 2013.

[81] Selig, J.M. *Geometric Fundamentals of Robotics*. Monographs in Computer Science. Springer New York, 2004.

[82] J.-A. Serret. Sur quelques formules relatives à la théorie des courbes à double courbure. *Journal de Mathématiques Pures et Appliquées*, pages 193–207, 1851.

[83] Ajay Seth, Michael Sherman, Peter Eastman, and Scott Delp. Minimal formulation of joint motion for biomechanisms. *Nonlinear dynamics*, 62:291–303, October 2010.

[84] Michael A. Sherman, Ajay Seth, and Scott L. Delp. Simbody: multibody dynamics for biomedical research. *Procedia IUTAM*, 2:241–261, 2011.

[85] Shunichi Honda and Masatomo Takahashi. Framed curves in the Euclidean space. *Advances in Geometry*, 16(3):265–276, 2016.

[86] Stephan Alt. FVD^{++}. `https://github.com/altlenny/openFVD`, 2016.

[87] Brian L Stevens, Frank L Lewis, and Eric N Johnson. *Aircraft control and simulation: dynamics, controls design, and autonomous systems*. John Wiley & Sons, 2015.

[88] K. Strehmel, R. Weiner, and H. Podhaisky. *Numerik gewöhnlicher Differentialgleichungen: Nichtsteife, steife und differential-algebraische Gleichungen*. Vieweg+Teubner Verlag, 2012.

[89] D.J. Struik. *Lectures on Classical Differential Geometry*. Addison-Wesley series in mathematics. Addison-Wesley Publishing Company, 1961.

[90] Martin Tändl. *Dynamic Simulation and Design of Roller Coaster Motion*. Fortschritt-Berichte VDI / 20. VDI-Verlag, 2009.

[91] Martin Tändl and Andres Kecskemethy. An objectoriented framework for incorporating spline curves as generalized joints in multibody systems. *PAMM*, 3:164 – 165, December 2003.

[92] Martin Tändl and Andres Kecskemethy. Optimization of spatial tracks using a morphing approach. *Pamm*, 6:115–116, 12 2006.

[93] Martin Tändl and Andres Kecskemethy. A Comparison of B-Spline Curves and Pythagorean Hodograph Curves for Multibody Dynamics Simulation. In *12th IFToMM World Congress*, Besanǫn, France, June 2007.

[94] August van Kaven. *Vorträge über Ingenieur-Wissenschaften an der polytechnischen Schule zu Aachen: Einleitung zum Wege- und Eisenbahnbau*. 1870.

[95] Remco C Veltkamp and Wieger Wesselink. Modeling 3D curves of minimal

energy. In *Computer Graphics Forum*, volume 14, pages 97–110. Wiley Online Library, 1995.

[96] L. Vojacek. Überhöhung der Geleise in Curven. *Zeitschrift des Vereins Deutscher Ingenieure*, 12(10):617–624, 1868.

[97] A. von Dziegielewski, R. Erbes, and E. Schömer. Conservative swept volume boundary approximation. In *Proceedings of the 14th ACM Symposium on Solid and Physical Modeling*, SPM '10, page 171176, New York, NY, USA, 2010. Association for Computing Machinery.

[98] Andreas von Dziegielewski, Michael Hemmer, and Elmar Schömer. High quality conservative surface mesh generation for swept volumes. In *2012 IEEE International Conference on Robotics and Automation*, pages 764–769, 2012.

[99] Antti Väisänen. *Design of Roller Coasters*. Aalto University, July 2018.

[100] D.J. Walton and D.S. Meek. A controlled clothoid spline. *Computers & Graphics*, 29(3):353–363, 2005.

[101] Wenping Wang and Barry Joe. Robust computation of the rotation minimizing frame for sweep surface modeling. *Computer-Aided Design*, 29(5):379 – 391, 1997.

[102] Wenping Wang, Bert Jüttler, Dayue Zheng, and Yang Liu. Computation of rotation minimizing frames. *ACM Trans. Graph.*, 27(1):2:1–2:18, March 2008.

[103] Shawn Waters. Roller Coaster Database. `https://rcdb.com/16349.htm`, 2018. [Online; accessed 18-March-2022].

[104] K. Watorek. Übergangsbogen. *Organ für die Fortschritte des Eisenbahnwesens*, 44(9):186–189, 1907.

[105] Parkz Website. https://www.parkz.com.au. `https://www.parkz.com.au/cache/photo/individual_photo/construction/2017/07/247ad1e91ef25f52bfe8fa2f89c5c0eb.jpg`, 2017. [Online: accessed 20-November-2019].

[106] Mihriban Whitmore, Jennifer Boyer, and Keith Holubec. *NASA-STD-3001, Space Flight Human-System Standard and the Human Integration Design Handbook*. 2012.