UNIVERSITÀ DEGLI STUDI
DELL'INSUBRIA

Dipartimento di Scienza e Alta Tecnologia
Doctoral course in Computer Science and Computational Mathematics

PhD Thesis

SMOOTH PATH PLANNING WITH PYTHAGOREAN–HODOGRAPH SPLINE CURVES: GEOMETRIC DESIGN AND MOTION CONTROL

DUCCIO MUGNAINI

Supervisor: *Prof. Alessandra Sestini*
Co-supervisor: *PhD. Carlotta Giannelli*

Cycle XXX
2014-2017

Duccio Mugnaini
Smooth path planning with Pythagorean–hodograph spline curves: geometric design and motion control,
Doctoral course in Computer Science and Computational Mathematics, © Cycle XXX 2014-2017

CONTENTS

ABSTRACT

This thesis addresses two significative problems regarding autonomous systems, namely path and trajectory planning. Path planning deals with finding a suitable path from a start to a goal position by exploiting a given representation of the environment. Trajectory planning schemes govern the motion along the path by generating appropriate reference (path) points. In such context, the development of path planning methodologies generating feasible and desirable paths is of fundamental importance, as well as the design of feedrate scheduling algorithms capable to completely describe the motion on the desired path under given kinematics restrictions.

We propose a two-step approach for the construction of planar smooth collision-free navigation paths. Obstacle avoidance techniques that rely on classical data structures are initially considered for the identification of piecewise linear paths that do not intersect with the obstacles of a given scenario. The environment can be described by specifying the boundaries of the obstacles (*path planning*) or by an adjacency matrix associated with uniform cells that cover the whole scene (*path finding*). Since the shape of the intermediate piecewise linear path remarkably influences the smoothness of the final path, different approaches are proposed to solve the path planning/finding problem in the first step of the method by exploiting certain angle-based criteria.

In the second step of the scheme we rely on spline interpolation algorithms with tension parameters to provide a smooth planar control strategy. In particular, we consider Pythagorean–hodograph (PH) curves, since they provide an exact computation of fundamental geometric quantities. The vertices of the previously produced piecewise linear paths are interpolated by using a G^1 or G^2 interpolation scheme with tension based on PH splines. In both cases, a strategy based on the asymptotic analysis of the interpolation scheme is developed in order to get an automatic selection of the tension parameters.

To completely describe the motion along the path we present a configurable trajectory planning strategy for the offline definition of time-dependent C^2 piecewise quintic feedrates. Since the less restrictive formulations of our strategy can usually still ensure all the desired bounds while simultaneously producing faster motions, the configurability feature is useful not only when reduced motion control is desired but also when full kinematic control has to be guaranteed. The newly introduced approach can be applied to any planar path with a piecewise sufficiently smooth parametric representation. When PH spline curves are considered, the corresponding accurate and efficient CNC interpolator algorithms can be exploited.

LIST OF PUBBLICATIONS

- C. Giannelli, D. Mugnaini, A. Sestini. *Path planning with obstacle avoidance by* G^1 *PH quintic splines*, Computer-Aided Design, 75:47-60, 2016.

- M. Donatelli, C. Giannelli, D. Mugnaini, A. Sestini. *Curvature continuous path planning and path finding based on PH splines with tension*, Computer-Aided Design, 88:14-30, 2017.

- R.T. Farouki, C. Giannelli, D. Mugnaini, A. Sestini. *Path planning with Pythagorean–hodograph curves for unmanned or autonomous vehicles* , Journal of Aerospace Engineering, to appear.

- C. Giannelli, D. Mugnaini, A.Sestini. C^2 *continuous time–dependent feedrate scheduling with configurable kinematic constraints*, in preparation.

1

INTRODUCTION

The increasing computing capacity and the decreasing cost of electronic components have brought autonomous systems to come in our everyday lives. In the last decade we have assisted to a wide, fast and growing use of autonomous systems in many areas. Nowadays, for instance, robots are no longer just used in factories to help and relieve humans from heavy and repetitive tasks, but they are also employed in a wide set of contexts, including for example washing floors, agricultural machinery management, war air missions and explorations of areas inaccessible to humans. Consequently, the definition of *advanced* and *reliable* route planning techniques for such advanced systems is becoming of primary importance.

The main focus of this thesis is the smooth path and trajectory planning problem for a point robot representation in a planar environment characterized by the presence of stationary obstacles. Such problems have been intensively studied in recent years, see e.g., [41, 14, 52] and references therein. The aim of path planning is to define a *collision-free path* from an initial to a target position in a certain scenario that includes a set of obstacles. Trajectory planning instead allows us to completely describe the motion along the resulting path by assigning a suitable time-law which prescribes the velocity, acceleration and jerk along the path while (possibly) satisfying given kinematic bounds. In order to avoid forbidden configurations[1] related to a given scenario, several graph–like structures may be considered, see for example [41] for a recent survey related to possible collision–free piecewise linear solutions. In this work the obstacle configuration is obtained either by the direct geometrical description of planar convex polygons or by an image of the environment. In the second case an established image segmentation[2] algorithm [11] to detect the pixels representing an obstacle is suitably exploited. A uniform grid[3] is then usually considered, consisting of free and obstructed cells related to the image pixels to describe the navigability within the environment. This facilitates the description of complex and general environments.

Given the description of the environment, to obtain a feasible path we rely on roadmap techniques which exploit an environment navigability structure

---

[1] According to the terminology in [14], a configuration is defined by a set of parameters which completely describe the position of the object within the environment. Hence a forbidden configuration is a set of parameters that brings the object to collide with an obstacle.

[2] This method became useful when only an image of the environment is available.

[3] When uniform grids are used to represent the adjacent relationships between cells (in our case corresponding to each pixel), the term *path finding*, in place of path planning, is more appropriate.

described by a graph, see e.g., [6, 51]). This kind of approaches only produces a collision-free piecewise linear path, an unsatisfactory solution for many real world applications. Hence, to provide an optimal trade-off between the accuracy of the prescribed trajectory and the flexibility of interactive navigations, the information concerning the collision-free piecewise linear path may be subsequently combined with spline interpolation techniques that provide a smooth planar control strategy, see e.g., [48]. By considering interpolation schemes[4] with tension control — see e.g., [4, 32] and the references therein — as a control tool on the shape of the interpolating curve, we present the following two–step approach for *smooth* path planning with obstacle avoidance depicted in Figure 1:

- Step 1: we focus on the identification of the most suitable piecewise linear path P_1 free from collision with obstacles. For this reason, we have developed several algorithms for the modification of the shortest piecewise linear path associated to the trapezoidal map, visibility graph, adjacent-grid and a newly introduced dual graph according to certain angle–based criteria.
- Step 2: P_1 is replaced by a feasible curvilinear counterpart P_2 described by a *Pythagorean–hodograph* (PH) quintic spline curve that interpolates the P_1 vertices. We consider this class of curves since they usually provide paths with fair shape and always guarantee exact computation of fundamental geometric quantities, like curvature and arc length [22]. This can also facilitate the physical part of the motion which requires accurate arc length and curvature computations [78].

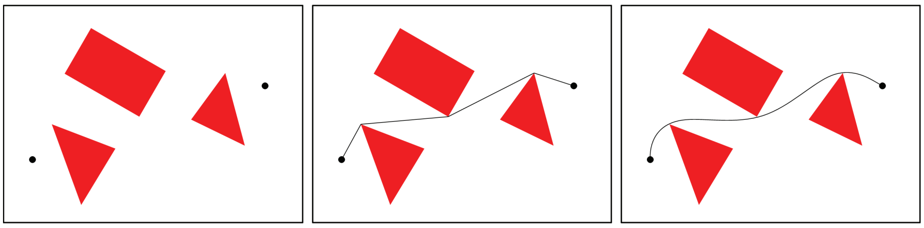


Figure 1. From left to right: a bounded polygonal environment with two assigned positions, the resulting collision-free piecewise linear path (Step 1) and its curvilinear counterpart (Step 2).

---

[4] Note that a standard way to construct PH curves relies on the solution of interpolation problems, since their non–linear nature usually precludes a direct definition in terms of Bézier/B–spline control polygons, see e.g., [34, 1, 46, 28] and [26, 33, 70] for the planar and the spatial case, respectively. Hence interpolation methods seem to be the most suitable way to solve the path planning problem with this class of curves.

The resulting path from the path planning (or path finding) phase consists in a G^1 or G^2 Pythagorean-Hodograph (PH) quintic spline curve[5] generated by an automatic selection of tension parameters to guarantee the absence of collision with the obstacles along the entire path. In literature path smoothing strategies have been proposed using different classes of parametric curves, such as B-spline, see e.g., [54, 59, 60]. These methods are usually characterized by an iterative procedure and do not guarantee an automatic identification of a curve free from collision. Consequently, after a collision detection, the curve is *adjusted* (e.g. by control points modification with some fixed criteria) and a new curve evaluation is required to check its feasibility, see e.g., [58].

In the recent years PH curves have been also applied in the Unmanned Autonomous Vehicles (UAV) context to represent collision-free paths which must satisfy various constraints, such as bounds on the path curvature or climb angle, avoidance of environmental obstacles and maintenance of safe separations in vehicle swarms, see e.g. [25, 5, 16, 56, 65, 67, 71, 76]. In this work we also introduce some novel utilities based on PH curves which are of particular interest in the path planning problem for single and multiple autonomous vehicles [27].

Once the collision-free curvilinear path P_2 is identified, the design of a suitable time-law is crucial to achieve high-speed control and completely describe the object motion along the path. For this reason a trajectory planning algorithm has been developed to generate a suitable time-law, called *feedrate* function, under given kinematics bounds. The determination of fast and reliable feedrate schedulers is fundamental to achieve trajectory planning characterized by high-speed and high-accuracy, necessary features in many applications such as robotics, digital animation and computer aided manufacturing. To obtain a smooth feedrate profile with chord error constraint, many interpolation algorithms were developed, see e.g., [3, 53, 81, 8, 50, 55, 82] as well as [77] and reference therein. Many of them do not guarantee a priori any desired kinematic constraints. Thus in the final part of the thesis we present an accurate, efficient and versatile off-line feedrate scheduler that produces a trapezoidal time-dependant globally C^2 piecewise polynomial quintic feedrate profile constrained by chord tolerance, velocity, acceleration and jerk limitations. The proposed scheduler is configurable since it presents two classes of parameters which are represented by the kinematics bounds and the value of the curvature and exploited to set the feedrate values along the curve.

At running time, thanks to the PH intrinsic structure of the considered collision-free curvilinear path, an accurate and fast PH interpolator [37] can be used to determine a sequence of *reference points*, which represent the positions along the path associated to a uniform time spacing. The results obtained with Pythagorean-hodograh spline curves for trajectory planning (especially

---

[5] G^1 and G^2 both require that two adjacent spline curve segments share a common unit tangent. Besides that G^2 continuity requires that they share also a common center of curvature at the join point, respectively.

for CNC applications) have confirmed the potential of this approach, see e.g., [78, 29, 30, 80, 75, 35] as well as [22] and references therein.

The structure of the thesis is as follows.

Chapter 2 provides the preliminary material that introduces the problem setting and the graph structures considered in the subsequent algorithms, namely the trapezoidal map, the visibility graph, the adjacent grid and the dual graph. The design of a piecewise linear collision–free path is also addressed through the introduction of different algorithms that rely on the information provided by the above mentioned data structures. In particular, a symmetric solution, which minimizes as much as possible the total cumulative angle between consecutive segments of the path, is obtained by exploiting the novel dual graph structure.

Chapter 3 provides an overview of Pythagorean–hodograph curves by focusing on G^1 and G^2 PH quintic spline interpolants with tension parameters. An asymptotic analysis that can be exploited for choosing the free parameters involved in the interpolation scheme is developed in Section 3.2.2 and 3.3.1 for the G^1 and G^2 schemes, respectively. Illustrative examples in a non-trivial obstructed scenarios are presented in Section 3.4.1 and 3.4.2.

Chapter 4 addresses representative applications of PH curves in UAV context: the maintenance of safe separations within swarms of autonomous vehicles in Section 4.3; the construction of families of curved paths with identical arc lengths (ensuring the simultaneous arrival of multiple autonomous vehicles) in Section 4.4; the imposition of path curvatures consistent with vehicle steering rates in Section 4.5; and curvature–continuous rounding of sharp corners in piecewise–linear paths through a field of polygon obstacles in Section 4.6.

Chapter 5 introduces an off-line feedrate scheduling method to control the motion of an object constrained by chord tolerance, acceleration and jerk limitations. In Section 5.2 some preliminary concepts and kinematic constraints are introduced. The configurable time-dependent feedrate scheduling algorithm is described in Section 5.3 while the CNC interpolator algorithm for PH curves is reviewed in Section 5.4. Finally, Section 5.5 illustrates the performance of our approach on a selection of significative examples.

Chapter 6 concludes the thesis by summarizing the main contributions and possible future developments.

2

PATH PLANNING AND RELATED DATA STRUCTURES

2.1 PATH PLANNING

The aim of *path planning* is to define a suitable collision–free path from an initial to a target position in a certain scenario that includes a set of obstacles, as shown in Figure 2. To formally define the path planning problem we need to introduce some concepts to describe the physical position or state that a robot can assume during the motion. In robotics, the position of every point of the robot is usually defined as its *configuration* $\mathbf{q}$ and depends on the specific robot model. The parameters needed to specify the robot configuration identify the *degrees of freedom* associated to the problem. All possible configurations define the *configuration space* $\mathcal{Q}$ (or *C–space*) that can be divided in two subsets: $\mathcal{Q}_{\mathtt{free}}$, the set of robot configurations free from collision with obstacles, and $\mathcal{Q}_{\mathtt{obst}}$, the obstacle configuration space given by the set of intersection configurations between the robot and any obstacle. Hence, a configuration $\mathbf{q}$ is just a point in the configuration space and the number of the degrees of freedom represents the dimension of $\mathcal{Q}$ [14]. Therefore the path planning problem can be formally stated as follows:

Definition 1 *Given a two– or three–dimensional rigid body and a certain scenario with a prescribed set of obstacles, the path planning problem looks for a smooth vector function* $\boldsymbol{r}(t) : [0, 1] \to \mathcal{Q}$ *so that*

$$\boldsymbol{r}(0) = \boldsymbol{q}_s \ , \ \boldsymbol{r}(1) = \boldsymbol{q}_g \qquad \text{and} \qquad \boldsymbol{r}(t) \in \mathcal{Q}_{\mathtt{free}} \quad \forall t \in [0, 1],$$

where $\boldsymbol{q}_s$ *and* $\boldsymbol{q}_g$ *correspond to the start and goal configuration, respectively. The desired path is the image of* $\boldsymbol{r}$ *in* $\mathcal{Q}$.

A path planning algorithm is *complete* if and only if it finds a feasible path when $\mathbf{q}_g$ is reachable from $\mathbf{q}_s$, otherwise it correctly reports that no solution exists. Furthermore a path planner is defined *optimal* if it returns the optimal path with respect to some criterion.

The path planning problem can be considered in a wide set of scenarios by taking into account different problem settings. In particular, we consider the simplest planar *point robot problem* where a circular robot of radius r is moving within a certain scenario characterized by convex obstacles of polygonal shapes. Consequently, the configuration space is just obtained by augmenting the obstacle boundaries by their polygonal offset at distance r, as shown in Figure 3.

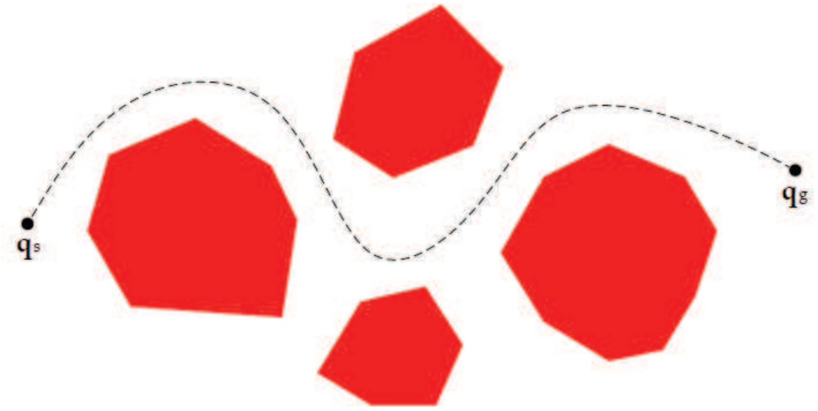
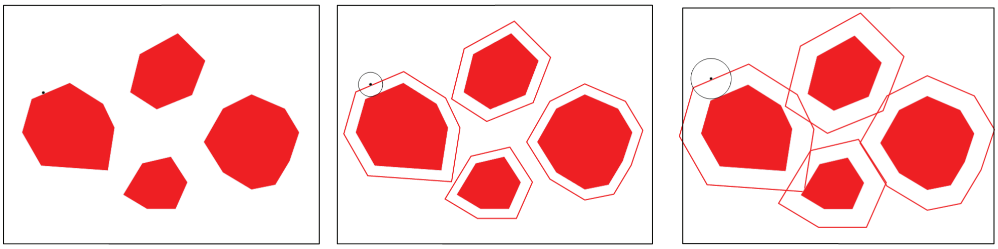


Figure 2. An example of collision free path that connects a start ($\mathbf{q}_s$) to a goal ($\mathbf{q}_g$) position in an planar environment with polygonal obstacles (red).



(a) (b) (c)

Figure 3. A polygonal obstacle configuration (a) and the related scenarios obtained by increasing the robot radius (b-c).

Then, after this preliminary offset computation, we can consider the robot as a moving point within a two–dimensional environment characterized by a known set of static obstacles with a fixed position. In this context, the robot follows an a priori identified path, constructed according to some *off–line* controller. In order to identify a feasible path, certain *navigability structures* that contain information about the free configuration space are usually exploited. Several techniques to obtain different navigability structures are available as described in Section 2.2.

2.2 ROADMAP

In order to plan a path we need to represent the environment by exploiting a suitable structure called *map*. *Mapping* instead is the task of generating the environment model of robot workspace from sensor data. Since in this work we assume the robot to be a priori equipped with a map of the environment, we do not discuss the mapping step. The right choice of a suitable structure for the environment representation is crucial to obtain a efficient and reliable path planner.

There are three main types of map:

- *topological*: represent the environment with a graph-like structure, where nodes correspond to some distinguished features and edges represent an adjacency relationship between nodes;
- *geometric*: the environment is represented by geometric primitives, usually 2D polygonal shape in the planar case and triangular meshes in 3D case;
- *grid structure*: the environment is subdivided in small cells and each cell is marked free if the corresponding portion of the environment is free from obstacle collision, otherwise it is marked occupied.

Environment representation techniques can be also classified in *exact* and *approximate* cell decomposition. Exact cell decomposition techniques subdivide the configuration space into smaller convex polygons, which are connected by the arcs of a graph that is exploited to find a feasible path. Approximate cell decomposition subdivide the entire configuration space into rectangular region and labels each rectanguloid as being completely filled or completely empty. In this work we exploit topological maps called *roadmaps* [14] which allow to reduce the information concerning the navigability of the considered environment. A roadmap is essentially a graph (directed or undirected) embedded in $\mathcal{Q}_{\mathtt{free}}$ and, consequently, the corresponding sets of edges and arcs may be used to guide the robot between the different obstacles. A node of the graph corresponds to a specific admissible configuration, while every arc identifies a feasible path to reach another admissible configuration, as shown in Figure 4. Each configuration along each arc is an admissible configuration. Formally we can define a roadmap as follows:

Definition 2 *A roadmap* $\mathcal{R}$ *is the union of curves that, for any* $\boldsymbol{q}_s, \boldsymbol{q}_g \in \mathcal{Q}_{\mathtt{free}}$*, satisfy the following properties.*

(i) Accessibility: there exists a path from $\boldsymbol{q}_s \in \mathcal{Q}_{\mathtt{free}}$ *to some* $\boldsymbol{q}_s^* \in \mathcal{R}$*.*

(ii) Departibility: there exists a path from some $\boldsymbol{q}_g^* \in \mathcal{R}$ *to* $\boldsymbol{q}_g \in \mathcal{Q}_{\mathtt{free}}$*.*

(iii) Connectivity: there exists a path in $\mathcal{R}$ *from* $\boldsymbol{q}_s^* \in \mathcal{R}$ *to* $\boldsymbol{q}_g^* \in \mathcal{R}$*.*

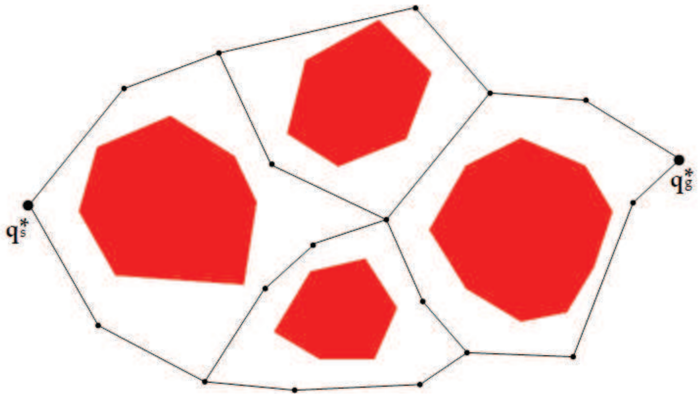


Figure 4. An example of *roadmap* for a given polygonal obstacle configuration.

Different kinds of roadmaps (and related graph representations) may be considered, like the visibility maps (visibility graphs), deformation retracts (generalized Voronoi diagrams), retract-like structures (generalized Voronoi graphs), piecewise retracts (rod–hierarchical generalized Voronoi graph), and silhouettes.

2.2.1 *Trapezoidal Map*

Trapezoidal map is an example of exact cell decomposition [14, 41]. This kind of methods describe the environment by subdividing the free space into small convex polygons which are then connected by a graph. Consequently a graph search algorithm can then be used to find a suitable path. Exact cell decomposition strategies divide $\mathcal{Q}_{\mathtt{free}}$ in a set of non–overlapping cells, each of which is characterized by a simple geometry not only to facilitate the computation of a path between any two configurations inside a cell, but also to simplify the adjacency test of two cells and the construction of a path crossing the common boundaries between two adjacent cells. After the initial cell decomposition, a connectivity graph is built to represent the adjacency relationship between cells.

The *trapezoidal map* (or trapezoidal/vertical decomposition) is a well–known example of exact cell decomposition algorithm. Given a polygonal environment, the first step is to define a bounding box R that includes all obstacles. A trapezoidal map is conventionally obtained by drawing two vertical extensions

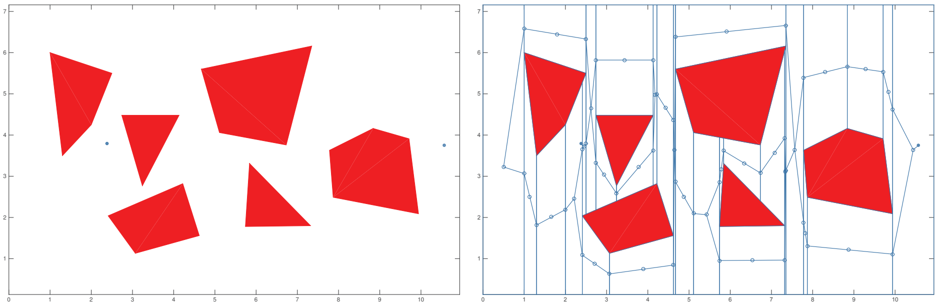


Figure 5. A simple scenario (S1) together with a start and goal configuration (left). The corresponding trapezoidal decomposition is also shown (right).

(one going upwards and the other going downwards) from every vertex of the obstacles to the first intersection with an obstacle edge or to the bounding box. Once $Q_{\mathtt{free}}$ is partitioned in a set of trapezoidal cells, then the adjacency graph is defined by placing a node inside each cell, e.g. its geometric centroid, and additional ones in the middle of the vertical extensions. An arc is then defined between two vertices of these two kinds of nodes associated to the same trapezoid. In order to construct the target path, $\mathbf{q}_s$ and $\mathbf{q}_g$ are added to the graph by simply connecting them to the central vertices $\mathbf{v}_s$ and $\mathbf{v}_g$ of the two corresponding trapezoids. Different graph search algorithms on the trapezoidal map identify admissible collision–free paths between $\mathbf{v}_s$ and $\mathbf{v}_g$. To determine the vertical extension we need to iterate on polygonal vertex $v_i, i = 1 \dots n$. For each vertex v_i the naive algorithm checks the intersection between the line through v_i with each polygonal edge $e_j, j = 1 \dots m$. This require $\mathcal{O}(n)$ time, hence $\mathcal{O}(n^2)$ time to construct the trapezoidal decomposition. By considering a suitable sweep approach with balanced search tree structures the complexity can be reduced to $\mathcal{O}(n \log n)$ [6]. An example of trapezoidal map is shown in Figure 5. As we can observe the resulting path is placed in the middle between the obstacles that is a important property from the collision avoidance point of view. However our experiments reveled that this kind of path is not a good candidate to be replaced with a curve since it often presents high cumulative angle between adjacent segments.

2.2.2 *Visibility Graph*

The *visibility graph* is a widely known roadmap method in computational geometry [6, 14]. Given a set of polygonal obstacles in the plane, the nodes of the visibility graph correspond to the vertices of each polygon. An arc e between two

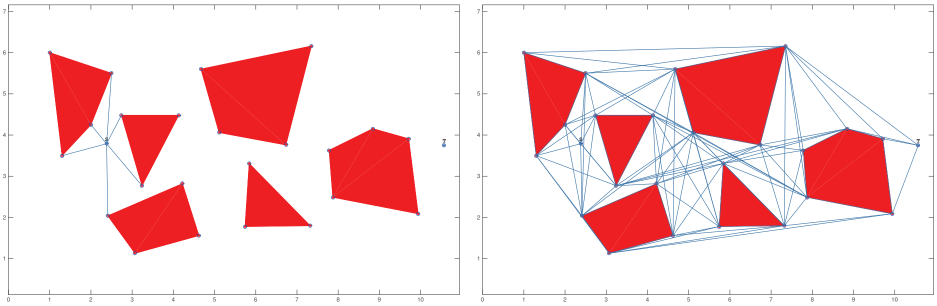


Figure 6. A simple scenario (S1) together with the corresponding visibility graph (VG).

vertices v_i, v_j belongs to the graph $G(\mathcal{V}, \mathcal{E})$ if the linear segment that connects these vertices does not intersect any obstacle,

$$e_{i,j} \in \mathcal{E} \Leftrightarrow (1-t)s_i + ts_j \in \mathcal{Q}_{free} \ \forall t \in [0,1].$$

To complete the roadmap, start $\mathbf{q}_s$ and goal $\mathbf{q}_g$ positions are added to the set of graph nodes together with the corresponding arcs needed to connect $\mathbf{q}_s$ and $\mathbf{q}_g$ to G. The naive algorithm to compute the visibility graph has complexity $\mathcal{O}(n^3)$, where n is the number of nodes in the graph. By considering a suitable sweep approach with balanced search tree structures the complexity can be reduced to $\mathcal{O}(n^2 \log n)$ [6].

Figure 6 (right) shows the visibility graph associated to a simple scenario. In case of planar environments, by using the visibility graph, together with classical graph search algorithms, an optimal and complete solution to the point vehicle problem associated to the shortest path can be obtained [6].

2.2.3 *Approximate cell decomposition*

Approximate cell decomposition techniques subdivide the whole environment by a uniform grid consisting of *free* and *obstructed* cells, identifying the free space and the obstacles, respectively. Note that in general this assumption implies that the obstacles, which in principle can have any shape, are necessarily approximated by polygons (2D) or polyhedra (3D). The accuracy of such approximation clearly increases with the grid dimension. This kind of description is adopted e.g. for the design of visual games [9], logistics applications or crowd simulations. The information related to the navigability of the free configuration space is encapsulated in an adjacency matrix that stores the graph edges connecting free cells. This kind of environment description is also suited when the scene is known only through an image. In this case the cells are squared and can be identified with pixels. A suitable segmentation method [43] is necessary for their labelling. Near cells are connected along the 4 cardinal directions (4–connected grid maps) or even along the four diagonal ones (8–connected maps) as shown in Figure 7.

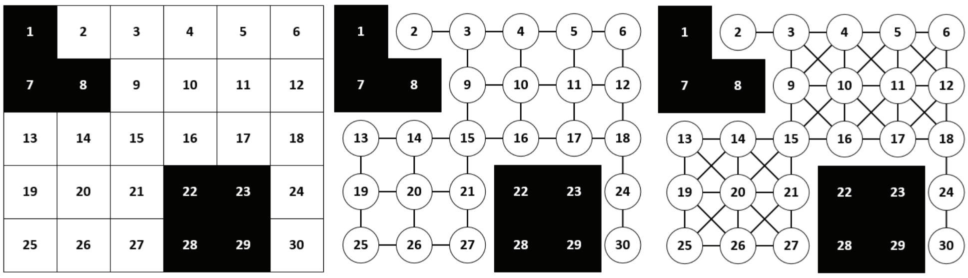


Figure 7. A simple scenario subdivided by a uniform grid with two obstacle in black (left), the related roadmaps respectively obtained using 4 (center) and 8–connections (right).

2.3 ANGLE BASED ALGORITHMS FOR COLLISION–FREE PIECEWISE LINEAR PATHS

Classical graph search algorithms can be used either on the trapezoidal map or the visibility graph to identify admissible piecewise linear paths that do not intersect any obstacles. A standard choice relies on the Dijkstra's algorithm that computes the *shortest path* between two vertices of a given graph [17]. Note that the shortest path associated to the visibility graph is the *absolute* shortest path from the start to the goal position with respect to the considered scenario, see e.g., [6].

In order to obtain a final path defined by a curve without significant curvature peaks, the identification of suitable piecewise linear paths with a small angle between two adjacent segments is required in the first step of the method.[1] For this reason, we introduce four graph search algorithms tailored for trapezoidal map (see Section 2.3.1), visibility graph (see Section 2.3.2, 2.3.3) and the novel dual graph algorithm (see Section 2.3.4). In the description of these algorithms, the routine for the shortest path computation is indicated as follows:

SP(`nodes, edges, start, goal`)

and returns the list of nodes of the shortest path between the node `start` and `goal` of the graph represented in terms of the input variables `nodes` and `edges`.

2.3.1 *Modified trapezoidal map shortest path (MTMSP)*

We consider the trapezoidal decomposition defined on a given polygonal environment. As shown in Figure 8 (left), the shortest path related to the graph defining the trapezoidal map may easily be characterized by several angle variations, even for relatively simple configurations. In order to identify a more suitable path, we delete a series of unnecessary vertices by applying several refinement steps to the initial shortest path. The procedure to define the *modified trapezoidal map shortest path* (MTMSP) is detailed in ALGORITHM 1 (see Appendix A.2).

Together with the procedure for the shortest path computation (SP) introduced in the previous section, three additional subroutines are considered in ALGORITHM 1 (line 4–6) and detailed below.

- VISIBILITYCHECK(`shortest, obstacles`): performs an iterative visibility check on `shortest`. If two non–consecutive nodes can be connected with a segment that does intersect any obstacle in `obstacles`, the nodes between them are deleted from the resulting modified shortest path.

---

[1] Not oriented convex angles measured in degrees are used in the first step of our method.

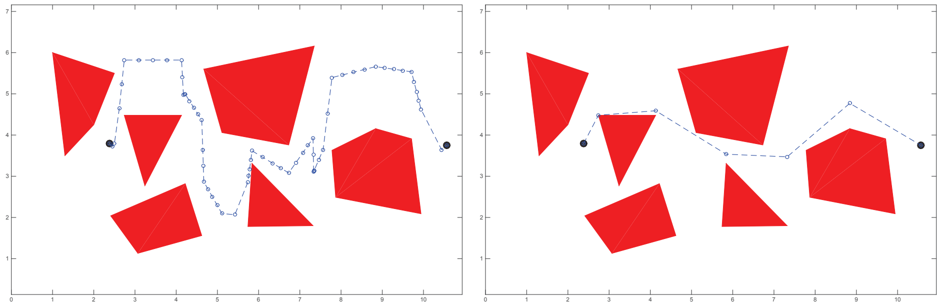


Figure 8. Shortest path (TMSP, left) and modified trapezoidal map shortest path (MTMSP, right) obtained with Algorithm 1 by considering the trapezoidal map related to scenario S1 of Figure 5.

- CollinearCheck(`modTMSP, obstacles, maxDist`): performs an iterative check of collinearity on `modTMSP`. Given three nodes n_1, n_2, n_3 along the path, if n_2 lies on the segment s that connects n_1 to n_3 or the distance between n_2 and s is less than `maxDist` and $s \in \mathcal{Q}_{free}$, then n_2 can be removed from `modTMSP`.

- ClusterCheck(`nodeList, obstacles, cluDist`): performs an iterative proximity check on `modTMSP`. Given three consecutive nodes n_1, n_2, n_3, if the distance between n_1, n_2 is less then `cluDist` and the segment connecting n_1 to n_3 is free from obstacle collisions with respect to `obstacles`, then n_2 is deleted from `modTMSP`.

The result obtained with Algorithm 1 by considering the trapezoidal map related to scenario S1 of Figure 5 is shown in Figure 8 (right) together with the initial shortest path (left).

2.3.2 *Modified visibility graph shortest path (MVGSP)*

We consider the visibility graph defined on a given polygonal environment. The path is initialized with the shortest path connecting the start to the goal positions, $\mathbf{q}_s$ and $\mathbf{q}_g$. In order to control the angles on the path without increasing too much its length, we perform an iterative check on the angles between successive edges e_1 and e_2 of the current path, see Figure 9. By moving in the direction from $\mathbf{q}_s$ to $\mathbf{q}_g$, the vertices of the current path are processed. When an angle greater than a given threshold is found on a vertex C, the edges connecting $\mathbf{q}_s$ to C, as well as e_2, are temporary removed from the list of edges of the visibility graph. The shortest path connecting C to $\mathbf{q}_g$ on the modified visibility graph is then computed, if possible. The current path is updated only if the angle with vertex in C between e_1 and the first edge of the new path is smaller than the given threshold. Otherwise the path is not updated and the next vertex along the path is processed. ALGORITHM 2 describes in detail this procedure which defines the *modified visibility graph shortest path* (MVGSP), see Appendix A.2.

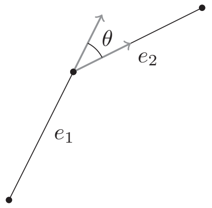


Figure 9. The angle θ between the consecutive edges e_1, e_2.

The result obtained with ALGORITHM 2 by considering the visibility graph related to scenario S1 is shown in Figure 10 together with the initial shortest path.

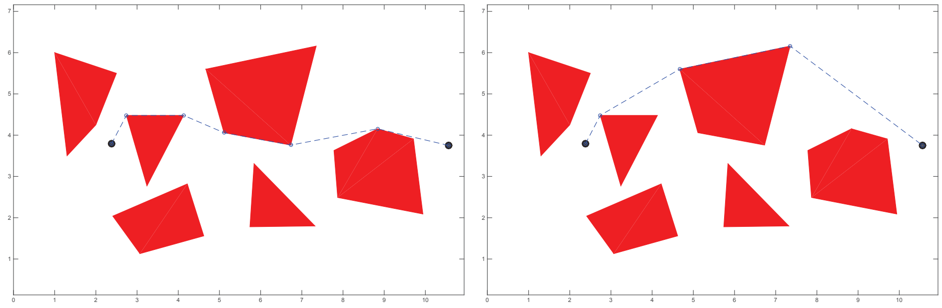


Figure 10. Shortest path (VGSP) associated to the visibility graph on scenario S1 shown in Figure 6 (left). The paths obtained with Algorithm 2 and 3 are coincident in this case and are shown on the right (MVGSP and GVGP).

2.3.3 *Greedy visibility graph path (GVGP)*

By considering again the visibility graph, we may also apply a *greedy* approach to compute a path from $\mathbf{q}_s$ to $\mathbf{q}_g$ that does not present angles of significative amplitude between adjacent edges. By starting from an initial segment, that can be for example the first edge along the shortest path, the path can be iteratively constructed by subsequently choosing the next edge in the visibility graph with the minimum angle that satisfies a certain threshold. If at a certain step, there are no edges adjacent to the current one satisfying the chosen threshold, the algorithm goes backward by looking for alternative solutions, if possible (this clearly depends on the given angle threshold).

The procedure to define the *greedy visibility graph path* (GVGP) is detailed in ALGORITHM 3, see Appendix A.2. Note that the input variable `visited` has to be initialized with two nodes: $\mathbf{q}_s$ and one of its adjacent node in the visibility graph.

The result obtained with Algorithm 3 by considering the visibility graph related to scenario S1 of Figure 6 is shown on the right of Figure 10. In this case, `visited` is initialized with the first two nodes of the shortest path shown on the left of Figure 10. Note that the paths obtained with ALGORITHM 2 and ALGORITHM 3 are coincident in this case.

	TMSP	MTMSP	VGSP	MVGSP/GVGP
# points	35	7	7	5
lenght	18.9537	9.6710	8.8054	9.7696
α_{min}	0	29.1960	12.7071	18.3246
α_{max}	151.6063	71.0412	62.4472	48.6153
α_{avg}	44.1021	47.4704	28.6454	33.0867

Table 1. Comparison of the four P_1 paths for S1 shown in Figures 8 and 10 (all angles are expressed in degrees).

The comparison of shortest paths related to the trapezoidal map and to the visibility graph (TMSP and VGSP) for scenario S1, together with the modified version computed with Algorithm 1, 2, and 3 (MTMSP, MVGSP/GVGP) is presented in Table 1. The number of points (# points) along the different paths as well as their lenghts (lenght) are quite similar for the different solutions. The only exception is the shortest path on the trapezoidal map, which necessarily requires suitable modifications, see also Figure 8. By also looking at the minimum, maximum and average angle ($\alpha_{min}, \alpha_{max}, \alpha_{avg}$) along the paths, we can see how the modified versions are able to produce P_1 paths with better shapes.

2.3.4 *Dual Graph algorithm (DGA)*

In Section 2.3.2 and 2.3.3 two algorithms were already introduced in order to reduce the amplitude of the angles between successive edges of the path selected from the visibility graph. Even if the numerical experiments confirmed that both methods are capable to ensure angle reduction with respect to the shortest path, both solutions are not completely satisfactory since they are not symmetric. This

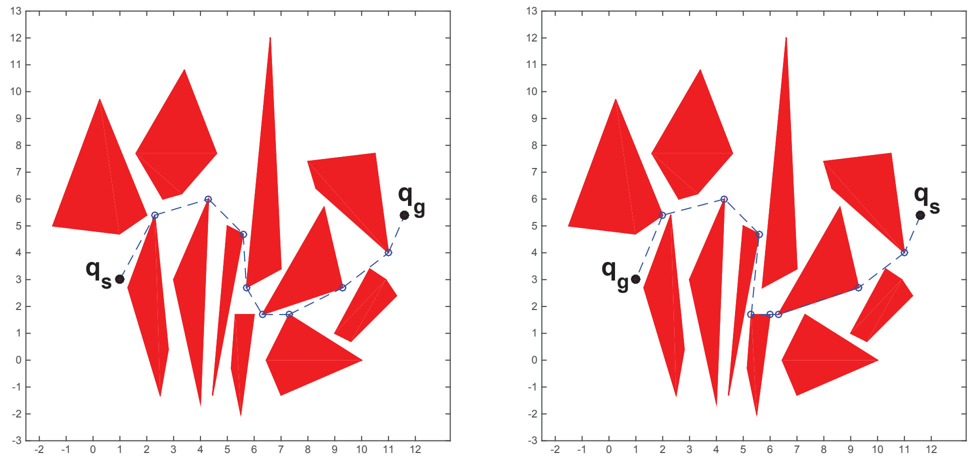


Figure 11. Two different paths produced on the visibility graph by Algorithm 2 when the initial and final positions $\mathbf{q}_s$ and $\mathbf{q}_g$ are swapped.

means that these two algorithms can produce different paths if $\mathbf{q}_s$ and $\mathbf{q}_g$ are swapped, see for example Figure 11.

In order to overcome this drawback, an alternative solution is here proposed which will be denoted as *dual graph algorithm (DGA)*. This latter exploit the dual graph structure which represent an intermediate reliable tool built from a roadmap (in our case the visibility graph) to extract a suitable path different from the shortest one.

Let $G(V, E)$ be a connected undirected graph defined in terms of a set of vertices (V) and edges (E) that includes $\mathbf{q}_s$ and $\mathbf{q}_g$ in V. The DGA is capable to select the path connecting $\mathbf{q}_s$ to $\mathbf{q}_g$ in G with minimal cumulative angle. This goal is achieved in two phases: the construction of the dual graph and the application of a graph search algorithm.

In the **first phase** an additional weighted directed graph $G'(V', E')$, called *Dual Graph* (DG), is associated with G and with a couple of edges $e_s = \mathbf{q}_s\mathbf{c}$ and $e_g = \mathbf{i}\mathbf{q}_g$ in E, that join $\mathbf{q}_s$ and $\mathbf{q}_g$ to another vertex in V, see Figure 12. Note that the assignment of these two edges is necessary to define on G' the start and goal positions, $\mathbf{q}'_s$ and $\mathbf{q}'_g$, and corresponds to prescribe the initial and final edges of the path in G produced by the algorithm. The procedure for constructing G' can be summarized as follows, see also Figure 12 (left and center).

Procedure

Input: navigability structure specified through a connected undirected graph $G(V, E)$ with $\mathbf{q}_s, \mathbf{q}_g \in V$, and a couple of edges $\mathbf{q}'_s, \mathbf{q}'_g \in E$, with $\mathbf{q}'_s = \mathbf{q}_s\mathbf{c}$ and $\mathbf{q}'_g = \mathbf{i}\mathbf{q}_g$.

1) vertices: for each arc in E connecting two vertices v and w in V, two vertices are inserted in V' and labelled as vw and wv, respectively. Then

the following vertices in V' are excluded: vertices whose label ends with $\mathbf{q}_s$; vertices whose label begins with $\mathbf{q}_s$ but is different from $\mathbf{q}_s\mathbf{c}$; vertices whose label begins with $\mathbf{q}_g$; vertices whose label ends with $\mathbf{q}_g$ but is different from $\mathbf{iq}_g$;

2) initial and final positions: $\mathbf{q}'_s = \mathbf{q}_s\mathbf{c}$ and $\mathbf{q}'_g = \mathbf{iq}_g$;

3) directed edges[2]: a vertex in V' labelled as vw is connected by a directed arc to all the other vertices in V' whose label begins with the letter w;

4) weights: each edge in E' connecting two vertices labelled as vw and wz is associated with a weight equal to the amplitude of the angle between the extension of the edge in E connecting v and w and the edge joining w and z.

Output: weighted directed graph $G'(V', E')$. In the **second phase** the Dijkstra

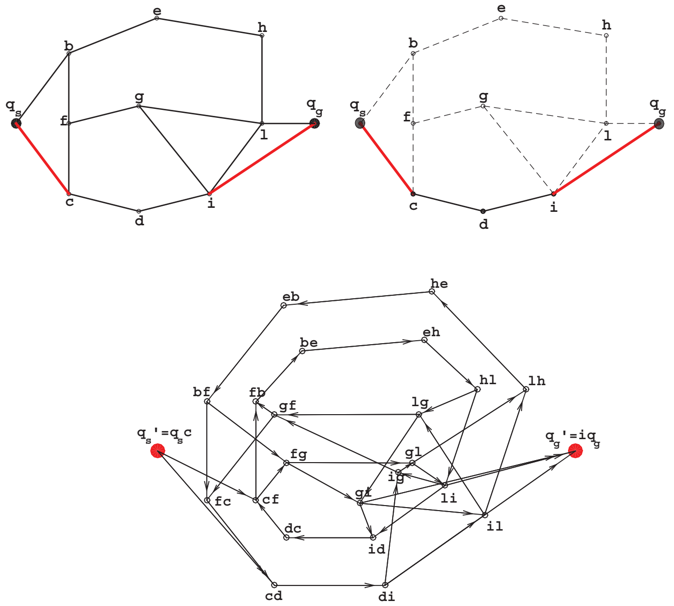


Figure 12. DGA algorithm: an undirected graph G (top left) with initial and final edges given by $\mathbf{q}_s\mathbf{c}$ and $\mathbf{iq}_g$ (thick red line); the related weighted directed dual graph G' (bottom) with initial and final positions given by $\mathbf{q}'_s = \mathbf{q}_s\mathbf{c}$ and $\mathbf{q}'_g = \mathbf{iq}_g$ (red points); the path with minimal cumulative angle in the primal graph G (top right, thick red/black line) which connects $\mathbf{q}_s$ to $\mathbf{q}_g$ with highlighted first and last edges (thick red line).

---

[2] Step 3) means that each edge in E' corresponds to an oriented sequence of three vertices in G while step 4) makes possible the angle control in the second phase. Note that, for any admissible selection, $\mathbf{q}'_s$ and $\mathbf{q}'_g$ are necessarily connected in G', since G is connected.

algorithm is used on G' to extract the cheapest path connecting $\mathbf{q}'_s$ to $\mathbf{q}'_g$. This latter path on G' is translated in a path on G connecting $\mathbf{q}_s$ to $\mathbf{q}_g$ with prescribed initial and final edges, see Figure 12 (right). The path with minimal cumulative angle can be obtained by comparing the paths produced by the algorithm for all the possible choices of the initial and final edges $\mathbf{q}'_s$ and $\mathbf{q}'_g$. In order to obtain the polyline with minimal cumulative angle shown for example in Figure 12 (right), it is necessary to compare $N_{out} \cdot N_{in}$ different polylines, where N_{out} and N_{in} respectively denote the number of edges outgoing from $\mathbf{q}_s$ and incoming in $\mathbf{q}_g$.

Remark 1 *The DGA algorithm is fully symmetric. Indeed, if $\mathbf{q}_s$ and $\mathbf{q}_g$ are swapped, the Dijkstra algorithm applied on G' produces a reversed path associated with the same path on G.*

Remark 2 *The DGA algorithm minimizes the cumulative angle but it can not guarantee the absence of large angles in the selected path. A variant of the algorithm can be defined to ensure that each angle is less than a given threshold `tol`. After G' has been generated and before moving to the second phase, the method should remove from E' all the edges whose weights are greater than `tol`. Clearly, this variant could fail if `tol` is chosen too small.*

The results of two experiments (PP1 and PP2) for two different environments represented in PP form are here reported and discussed, see Figure 13. We rely on the variant of the algorithm mentioned in Remark 2 by setting the tolerance `tol` equal to 75°.

Table 2 summarizes the main details of all the solutions obtained corresponding to all possible selections of the initial and final edges e_s and e_g.[3]

For each solution the related row of the table reports the number of vertices (# vertices), the length (L), the cumulative angle (α_{cum}) and the maximum angle (α_{max}) of the piecewise linear path. Observe that, for some solutions related to Example PP2, no details are reported in the table, since for the considered tolerance no path exists in the dual graph G' connecting $\mathbf{q}'_s$ to $\mathbf{q}'_g$ (see Remark 2).

The paths selected as input data for path smoothing (see Chapter 3) are shown in Figure 13 and they correspond to sol. 12 and sol. 3 for examples PP1 and PP2, respectively (see Table 2).

Note that sol. 12 for example PP1 has the minimum cumulative angle while sol. 3 for example PP2 minimizes α_{max}. Even if longer and with greater value of α_{cum}, the second one has been selected since it produces a final smoother path with also a more interesting shape than sol. 5 which minimizes α_{cum}.

---

[3] In Example PP1 (PP2) there are 4 (3) possible choices of e_s and 4 (3) of e_g.

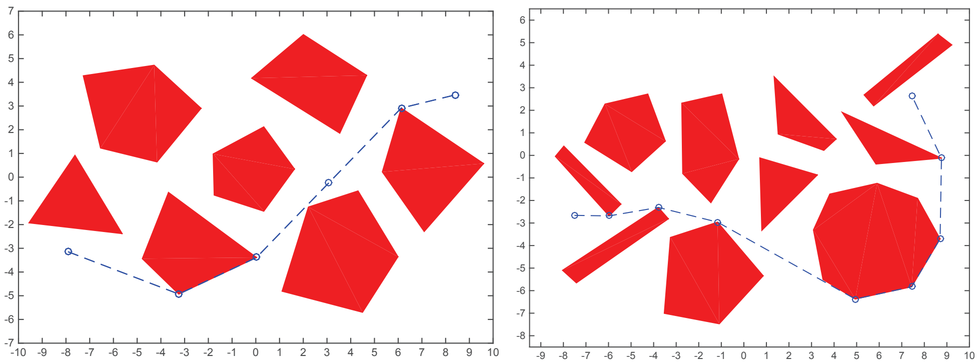


Figure 13. Examples PP1 (left) and PP2 (right): the environments and the related polylines from step 1 (sol. 12 for PP1 and sol. 3 for PP2, see Table 2).

2.3.5 *Pathfinding*

A number of different graph search algorithms can be subsequently applied on the adjacency matrix to address the problem of path finding. They may range from standard breadth–first or depth–first search, to Dijkstra's algorithm for the shortest path computation, and the more common choice of A* algorithm [44] or related extensions and modifications, see e.g. [9, 69] and references therein. The A* approach is characterized by an heuristic function that drives the search by estimating the distance from a start to a goal position (both identified by certain cells). This function is admissible if it does not overestimate the true distance between two cells. The Manhattan and Octile distances are two common choices for 4– and 8–connected maps, respectively. Given the start and goal positions $\mathbf{p}_s = (x_0, y_0)$, $\mathbf{p}_t = (x_1, y_1)$, they are defined as

$$h_M := a + b\,, \qquad h_Q := \sqrt{2}\,c + (d - c)\,,$$

where $a = |x_0 - x_1|$, $b = |y_0 - y_1|$, $c = \min(a, b)$, and $d = \max(a, b)$. The heuristic guides the path search along the grid map in order to reduce the number of graph nodes (cells) to be visited. In this way, the A* algorithm iteratively looks for the next good candidate to be visited. The algorithm keeps track of adjacent free cells of visited nodes and finishes when the current node is the goal configuration.

In particular we apply A* algorithm with Octile distance on the 8–connected grid map obtained by the image segmentation method summarized in the Appendix A.1.

Since we are working with an adjacency grid, the initial collision–free piecewise linear path obtained with the algorithm consists of a high number of segments that cross all the grid cells along the path, see e.g. Figure 15. For this

Dual Graph Results: example PP1				
sol.	# vertices	L	α_{cum}	α_{max}
1	7	20.768	134.829	72.896
2	6	20.660	103.444	61.794
3	8	24.899	233.322	69.694
4	9	18.268	104.683	32.901
5	7	25.075	196.474	73.672
6	6	24.967	165.089	73.672
7	12	37.734	418.644	73.672
8	8	25.135	214.325	73.672
9	10	26.820	325.983	72.896
10	9	26.712	294.598	63.760
11	6	23.991	133.442	61.649
12	6	19.722	99.048	46.131
13	8	21.941	195.329	73.799
14	7	21.833	163.943	73.799
15	7	24.281	195.548	61.649
16	7	20.011	161.155	69.390

Dual Graph Results: example PP2				
sol.	# vertices	L	α_{cum}	α_{max}
1	-	-	-	-
2	9	18.139	181.238	50.590
3	10	25.119	194.158	47.073
4	-	-	-	-
5	9	19.902	155.161	57.565
6	10	29.633	282.708	70.896
7	-	-	-	-
8	11	24.372	247.887	65.519
9	11	27.725	217.620	60.045

Table 2. Details of all the possible piecewise linear paths produced by step 1 with `tol` = 75° for the environments and the starting and goal positions considered in Examples PP1–PP2.

reason, we iteratively apply string–pulling and collinear vertices elimination to achieve a path whose vertices could be suitably interpolated. The two functions CollinearElimination and StringPulling perform these tasks. The first one simply removes collinear points along the path, while the second one executes an iterative visibility check along the path (if two non–consecutive nodes can be connected with a segment that does not intersect any obstacle, the nodes between them are deleted). At each iteration the number of nodes that compose the path is reduced, and the refinement loop ends when this is no longer true. At the end of this iterative loop, the JoinVertices procedure is called to join each two consecutive vertices at a distance less than a certain threshold (`maxDist`) along the path, whenever possible. If s is the segment whose end–points are two near vertices with respect to `maxDist`, a new node is placed at the intersection point between the two segments adjacent to s and these original adjacent points are eliminated. Finally, when an edge $\Delta\mathbf{p}_k$ of the polyline lies on the boundary of an obstacle positioned in the half–plane generated by $\Delta\mathbf{p}_k$ and not containing $\Delta\mathbf{p}_{k-1}$ and $\Delta\mathbf{p}_{k+1}$, the vertices $\mathbf{p}_k$ and $\mathbf{p}_{k-1}$ are suitably moved as shown in Figure 14.

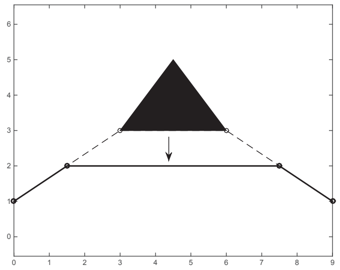


Figure 14. Sweeping edge.

The overall refinement scheme is presented in ALGORITHM 4. Figure 15 (top left) shows the path computed with A* algorithm together with the refined path obtained with ALGORITHM 4 (top right) for a test environment described in the path finding form. Obviously, the number of vertices along the path is significantly reduced after the refinement procedure, see the zoom of two selected regions on the bottom of Figure 15.

Remark 3 *In both the path planning and finding context, the polyline produced by the algorithms is post processed if it has Z–shaped sequences of three successive edges. In this case the inner edge of any Z sequence is divided into two edges joining at its middle point. This corresponds to insert an additional interpolation point whenever the polyline suggests an inflection point along the smooth path. These additional points simplify the automatic choice of the tension parameters described in the next chapter.*

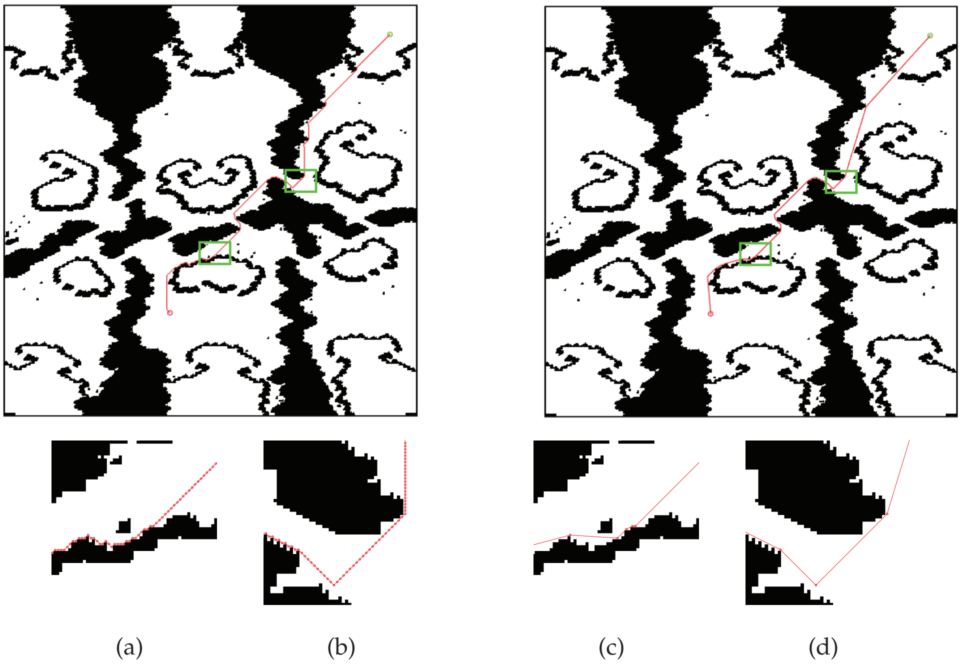

(a) (b) (c) (d)

Figure 15. Top: the path computed with A* algorithm with 636 nodes (left) and the refined path obtained with ALGORITHM 4 with 23 nodes (right). Also two selected regions are shown (green boxes). Bottom: the zoom of the piecewise linear paths in the two selected regions for the solution obtained with A* algorithm (a-b) and ALGORITHM 4 (c-d).

3

PATH SMOOTHING

3.1 PYTHAGOREAN–HODOGRAPH CURVES

The piecewise linear admissible path P_1, produced exploiting a roadmap method described in Chapter 2, is used in the second step of the method to define an admissible smooth path P_2 that also connects $\mathbf{q}_s$ to $\mathbf{q}_g$. In order to guarantee a strong relation between P_2 and P_1, we rely on interpolation schemes with tension control, see e.g., [4]. The smooth P_2 path is then the image of a parametric curve $\mathbf{r}(t), t \in [0\,,\,1]$, interpolating the vertices $\mathbf{p}_k$, $k = 0, \ldots, N$, that define P_1 and suitable tension parameters are used for ensuring that $\mathbf{r}(t)$ is sufficiently "close" to P_1[1]. This guarantees that the smooth interpolant also defines a collision–free path. In particular, we require that $\mathbf{r}(t)$ is a quintic spline curve such that

$$\mathbf{r}(t_k) = \mathbf{p}_k, \qquad k = 0, \ldots, N,$$

where the interpolation parameters $t_k, k = 0, \ldots, N$, with $0 = t_0 < \cdots < t_N = 1$, also define the spline knots (for the experiments they have always been chosen using the standard chord length parameterization [45]). Furthermore, we require that, for all $k = 1, \ldots, N$, the restriction $\mathbf{r}_k$ of $\mathbf{r}$ to $[t_{k-1}\,,\,t_k]$ has a Pythagorean–hodograph (PH) structure [22]. PH curves are polynomial parametric curves whose *hodograph* (first derivative) verify the Pythagorean condition, that is the sum of the squares of its components is the perfect square of a polynomial. Such curves were firstly introduced in [36] and later on a broad research activity has been devoted to the development of both, theory and application to the solution of several geometric design problems. We adopt PH splines in our path planning approach because interpolants with PH structure are not only usually fair curves, but they can also make easier and more accurate the successive application of related time laws to define a motion on the given path (see Chapter 5).

The structure of this chapter is as follows. Subsection 3.1.1 briefly introduces planar PH curves while Subsection 3.2 describes the exploited local representation in Bézier form. Section 3.2 and 3.3 respectively summarize the local G^1 and $C^1 \cap G^2$ interpolation scheme and automatic tension strategy based on PH quintic splines with tension which was introduced in [32] and is here adopted for our second step. Note however that in [32] the tension parameters were used to define a shape–preserving scheme, while in the present context they

---

[1] We consider P_1 valid to be replaced with his smooth counterpart only if each segment that is conincident with an edge of an obstacle have the two adjacent segments beloging to the half plane of the obstacle.

are exploited to guarantee that each spline segment belongs to an admissible polygonal region, as described in Subsection 3.2.1. As it will be detailed in Subsection 3.3.1 and 3.3.2, using such regions, the asymptotic analysis of the scheme developed in terms of the tension parameters allows us to identify *a priori* suitable values for them.

3.1.1 *Planar PH curves*

Planar polynomial curves $\mathbf{r}(t) = (x(t), y(t))$, $t \in [0, 1]$, that satisfy

$$|\mathbf{r}'(t)|^2 = x'^2(t) + y'^2(t) = \lambda^2(t)\,, \tag{3.1}$$

for some polynomial $\lambda(t)$ are known as Pythagorean–hodograph curves. Although PH curves of degree n have just $n+3$ degrees of freedom (compared to $2n+2$ for general degree–n polynomial curves), the Pythagorean condition (3.1) ensures that PH plane curves have the following distinctive properties:

- polynomial arc–length functions $s(t) = \int_0^t |\mathbf{r}'(\tau)|\, d\tau$;
- unit tangent $\mathbf{t}$, normal unit vector $\mathbf{n}$, and signed curvature κ all rational;
- rational offset curves $\mathbf{r}_d(t) = \mathbf{r}(t) + d\,\mathbf{n}(t)$, $d \in \mathbb{R}$;
- closed–form expression for the bending energy [21], given by the integral of the square of the curvature with respect to arc length and commonly used as “fairness” measure.

In order to satisfy condition (3.1), the three polynomials $x'(t)$, $y'(t)$ and $\lambda(t)$ must comprise a Pythagorean triple. Hence, according to the result of Kubota [49], these polynomials must be expressible in terms of other real polynomials $u(t)$, $v(t)$ and $h(t)$ in the form

$$x'(t) = h(t)\left[u^2(t) - v^2(t)\right], \qquad y'(t) = 2\,h(t)\,u(t)\,v(t)\,, \tag{3.2}$$

with corresponding polynomial parametric speed

$$\lambda(t) = h(t)\left[u^2(t) + v^2(t)\right]. \tag{3.3}$$

If the polynomials u and v have a common factor, this can be included into h, so we may assume without loss of generality that u and v are relatively prime. Moreover, if u and v are of degree m at most and $p = \deg h$, the PH curve $\mathbf{r}(t)$ obtained by integrating the hodograph components (3.2) is of degree $n = p + 2m + 1$. In the case of *primitive* hodograph, characterized by the fact $\gcd(x', y') =$ constant, which necessarily implies $\deg h = 0$, PH curves are always of *odd* degree $n = 2m + 1$. In order to avoid irregular curve points where the hodograph vanishes, primitive hodographs are usually considered for the development of related application algorithms.

Focusing on primitive curves with $h(t) = 1$ and $\gcd(u, v) =$ constant, the parametric speed (3.3) reduces to $\lambda(t) = u^2(t) + v^2(t)$ and, since it has no real roots, it does not change its sign. By expressing $\lambda(t)$ in the Bernstein basis,

$$b_k^n(t) = \binom{n}{k} t^k(1-t)^{n-k}, \qquad k = 0, \ldots, n, \tag{3.4}$$

the arc length function

$$s(t) = \int_0^t \lambda(t)\,dt \quad \text{with} \quad \lambda(t) = |\mathbf{r}'(t)| = \sum_{k=0}^{n-1} \lambda_k b_k^{n-1}(t)$$

reduces to the polynomial form

$$s(t) = \sum_{k=0}^{n} s_k b_k^n(t), \quad \text{where} \quad s_k = \frac{1}{n}\sum_{j=0}^{k-1} \lambda_j, \quad k, = 1, \ldots, n,$$

and $s_0 = 0$, while the total arc–length $L = s(1)$ can be expressed simply as

$$L = \frac{1}{n}\sum_{i=0}^{n-1} \lambda_i .$$

The PH structure is embodied in the complex representation, wherein a PH curve of degree $n = 2m + 1$ is generated [20] from a degree–m complex "pre–image" polynomial

$$\mathbf{w}(t) = u(t) + i\,v(t) = \sum_{k=0}^{m} \mathbf{w}_k \binom{m}{k} (1-t)^{m-k} t^k \tag{3.5}$$

with complex Bernstein coefficients $\mathbf{w}_k = u_k + i\,v_k$ by integration of the expression

$$\mathbf{r}'(t) = \mathbf{w}^2(t) . \tag{3.6}$$

The parametric speed, unit tangent, and curvature of $\mathbf{r}(t)$ may be formulated [20] in terms of $\mathbf{w}(t)$ as

$$\lambda(t) = |\mathbf{w}(t)|^2 , \quad \mathbf{t}(t) = \frac{\mathbf{w}^2(t)}{\sigma(t)} , \quad \kappa(t) = 2\,\frac{\mathrm{Im}(\overline{\mathbf{w}}(t)\mathbf{w}'(t))}{\sigma^2(t)} . \tag{3.7}$$

3.1.2 *PH quintics splines*

To define the smooth path P_2 interpolating the vertices $\mathbf{p}_k, k = 0, \ldots N$, of P_1 as described in Section 3.1, we adopt the following local representation in Bézier form of the k–th quintic spline segment $\mathbf{r}_k$ of $\mathbf{r}$,

$$\mathbf{r}_k(t) = \sum_{j=0}^{5} \mathbf{q}_{k,j}\, b_j^5 \left(\frac{t - t_{k-1}}{\ell_k} \right), \quad t \in [t_{k-1}\, , \, t_k]\, , \tag{3.8}$$

for $k = 1, \ldots, N$, where $\ell_k = t_k - t_{k-1}$ and b_j^5, $j = 0, \ldots, 5$, are the Bernstein polynomials of degree five. Since $\mathbf{r}_k$ is required to have a PH structure, the Bézier control points $\mathbf{q}_{k,j}, j = 0, \ldots, 5$, can be defined in terms of three complex numbers, $\mathbf{u}_k, \mathbf{v}_k$ and $\mathbf{z}_k$ and of the two assigned points $\mathbf{p}_{k-1}, \mathbf{p}_k$ (also interpreted as complex numbers),

$$\begin{aligned} \mathbf{q}_{k,0} &= \mathbf{p}_{k-1}\, , & \mathbf{q}_{k,1} &= \mathbf{q}_{k,0} + \frac{\ell_k}{5}\mathbf{u}_k^2\, , \\ \mathbf{q}_{k,2} &= \mathbf{q}_{k,1} + \frac{\ell_k}{5}\mathbf{u}_k\mathbf{z}_k\, , & \mathbf{q}_{k,3} &= \mathbf{q}_{k,4} - \frac{\ell_k}{5}\mathbf{z}_k\mathbf{v}_k\, , \\ \mathbf{q}_{k,4} &= \mathbf{q}_{k,5} - \frac{\ell_k}{5}\mathbf{v}_k^2\, , & \mathbf{q}_{k,5} &= \mathbf{p}_k\, . \end{aligned} \tag{3.9}$$

This definition of the control points immediately ensures the required interpolation of $\mathbf{p}_{k-1}$ and $\mathbf{p}_k$ at $t = t_{k-1}$ and $t = t_k$, respectively. It also ensures[2] that $\mathbf{r}_k'(\tau) = \mathbf{w}_k^2(\tau)$, where $\tau = (t - t_{k-1})/\ell_k$ is the local parameter in $[0\, , \, 1]$, and $\mathbf{w}_k(\tau) = \mathbf{u}_k b_0^2(\tau) + \mathbf{z}_k b_1^2(\tau) + \mathbf{v}_k b_2^2(\tau)$, if the following quadratic equation is fulfilled,

$$2\mathbf{z}_k^2 + 3(\mathbf{u}_k + \mathbf{v}_k)\,\mathbf{z}_k + \left(3(\mathbf{u}_k^2 + \mathbf{v}_k^2) + \mathbf{u}_k\mathbf{v}_k - \frac{15}{\ell_k}\Delta\mathbf{p}_k \right) = \mathbf{0}, \tag{3.10}$$

where $\Delta\mathbf{p}_k = \mathbf{p}_k - \mathbf{p}_{k-1}$ for $i = 1, \ldots, N$. In the next section suitable Hermite conditions will be imposed to determinate the coefficients $\mathbf{u}_k$ and $\mathbf{v}_k$.

3.2 SMOOTHING WITH G^1 PH QUINTICS WITH TENSION PARAMETERS

We summarize the local G^1 interpolation scheme based on PH quintic splines with tension, originally introduced in [32] and here adopted as first strategy for our second step. This scheme is a variant of the C^1 PH quintic Hermite interpolation scheme introduced in [34], whose approximation power was studied in [79]. In Subsection 3.2.2 we introduce an asymptotic analysis of the expression of the local control polygon $\mathbf{q}_{k,j}, j = 0, \ldots, 5$, associated with the k–th PH spline segment $\mathbf{r}_k$ defined in Subsection 3.1.2 for G^1 case. The analysis, developed

---

[2] The prime symbol denote the derivative with respect to the time.

through a preliminary proposition followed by a main theorem, shows that, when the tension parameter, denoted by σ_k, is small enough, $\mathbf{r}_k$ belongs to the corresponding convex admissible region R_k, and consequently to the free configuration space. Subsequently, in Subsection 3.2.3 we introduce an automatic strategy based on these analysis for a suitable selection of the tension parameters.

In order to obtain a G^1 Hermite PH spline interpolation scheme, the coefficients $\mathbf{u}_k$ and $\mathbf{v}_k$ in (3.9) are determined by imposing two additional end conditions on the hodograph, namely

$$\frac{d\mathbf{r}_k}{dt}(t_{k-1}) = c_k^2\,\sigma_k^2\,\mathbf{T}_{k-1}\,, \quad \frac{d\mathbf{r}_k}{dt}(t_k) = d_k^2\,\sigma_k^2\,\mathbf{T}_k\,. \tag{3.11}$$

Here $\mathbf{T}_{k-1}$ and $\mathbf{T}_k$ are given unit vectors specifying the direction of the tangent to the curve at $t = t_{k-1}$ and at $t = t_k$, respectively, while c_k and d_k together with σ_k are free positive parameters which are used to modify the shape of the k–th PH quintic segment. In particular, the coefficients c_k and d_k are selected following the strategy introduced in [4, Section 8], while $\sigma_k \leqslant 1$ is used as a tension parameter.

By using a local reference system with the x axis aligned with the displacement $\Delta\mathbf{p}_k$, the unit tangents can be expressed in complex form as

$$\mathbf{T}_{k-1} = \exp(i\,\theta_{k-1}^{(k)})\,, \quad \mathbf{T}_k = \exp(i\,\theta_k^{(k)})\,, \tag{3.12}$$

with $\theta_{k-1}^{(k)}, \theta_k^{(k)} \in (-\pi, \pi]$. Note that several approaches can be used for the definition of these unit tangents (see Remark 4 below) but most formulas ensure that for reasonable data $\theta_{k-1}^{(k)}, \theta_k^{(k)} \in (-\pi/2, \pi/2]$ for every k–th spline segment. In view of the derivative formula of Bézier curves, the two conditions in (3.11) correspond to $\mathbf{u}_k^2 = c_k^2\,\sigma_k^2\,\mathbf{T}_{k-1}$ and $\mathbf{v}_k^2 = d_k^2\,\sigma_k^2\,\mathbf{T}_k$, and consequently

$$\mathbf{u}_k = c_k\,\sigma_k\,\exp(i\,\alpha_k)\,, \quad \mathbf{v}_k = d_k\,\sigma_k\,\exp(i\,\beta_k)\,, \tag{3.13}$$

where the angles α_k and β_k can assume two different values, $\alpha_k \in \{\theta_{k-1}^{(k)}/2, \pi + \theta_{k-1}^{(k)}/2\}$ and $\beta_k \in \{\theta_k^{(k)}/2, \pi + \theta_k^{(k)}/2\}$.

If $(\mathbf{u}_k, \mathbf{v}_k, \mathbf{z}_k)$ is a solution of (3.13) and (3.10), $(-\mathbf{u}_k, -\mathbf{v}_k, -\mathbf{z}_k)$ is also another solution that corresponds to the same control points in (3.9). This implies that there are only four distinct triples $(\mathbf{u}_k, \mathbf{v}_k, \mathbf{z}_k)$ solving both (3.13) and (3.10) which correspond to as many different $\mathbf{r}_k$ obtainable with this scheme [34]. Thus, considering that the shapes of these four distinct alternatives can be very different and that the problem concerns each spline segment, an automatic strategy to select a reasonable $\mathbf{r}_k$ is necessary. In particular, we choose

$$\alpha_k = \frac{\theta_{k-1}^{(k)}}{2}\,, \quad \beta_k = \frac{\theta_k^{(k)}}{2}\,, \tag{3.14}$$

because this selection ensures that when $\mathbf{T}_{k-1}$ and $\mathbf{T}_k$ have a positive projection on $\Delta\mathbf{p}_k$, this is also true for $\mathbf{u}_k$ and $\mathbf{v}_k$. In addition, the solution $\mathbf{z}_k$ of (3.10) with maximal real part is chosen.

Remark 4 *For specifying the unit tangents* $\boldsymbol{T}_k, k = 0, \dots, N$, *we have considered two different approaches. The first one (T1) corresponds to the definition introduced in [4], namely*

$$\boldsymbol{T}_k = \frac{|\kappa_{k+1}|\Delta\boldsymbol{p}_k + |\kappa_{k-1}|\Delta\boldsymbol{p}_{k+1}}{\big|\,|\kappa_{k+1}|\,\Delta\boldsymbol{p}_k + |\kappa_{k-1}|\,\Delta\boldsymbol{p}_{k+1}\,\big|},$$

for $k = 1, \dots, N-1$ *and*[3]

$$\boldsymbol{T}_0 = \frac{|\kappa_1|\Delta\boldsymbol{p}_0 + |\kappa_0|\Delta\boldsymbol{p}_1}{|\,|\kappa_1|\,\Delta\boldsymbol{p}_0 + |\kappa_0|\,\Delta\boldsymbol{p}_1\,|},$$

$$\boldsymbol{T}_N = \frac{|\kappa_N|\Delta\boldsymbol{p}_N + |\kappa_{N-1}|\Delta\boldsymbol{p}_{N+1}}{\big|\,|\kappa_n|\,\Delta\boldsymbol{p}_N + |\kappa_{N-1}|\,\Delta\boldsymbol{p}_{N+1}\,\big|},$$

where $\kappa_k = \Delta\boldsymbol{p}_k \times \Delta\boldsymbol{p}_{k+1}$ *The second one (T2) is obtained by firstly using the algorithm described in [28] to compute the* C^2 *PH spline interpolant to the set of points* $\boldsymbol{p}_0, \dots, \boldsymbol{p}_N$ *and then defining the* $\boldsymbol{T}_k, k = 0, \dots, N$ *as its unit tangents at the knots. Note that, even if this method is more expensive, it often allows us to obtain paths of higher fair.*

3.2.1 *Admissible regions*

To ensure the property of obstacle avoidance to the final P_2 path, we require that each spline segment belongs to a certain convex *admissible region* associated with the corresponding displacement. The admissible region is either a triangle, a trapezoid or a parallelogram included in Ω_{free} with an edge coincident with the displacement. In order to identify a region of the free configuration space where the smooth P_2 path can lie without intersecting any obstacle, the midpoint of the segment connecting $\mathbf{p}_k$ and $\mathbf{p}_{k+1}$ is inserted as additional interpolation point with knot $(t_k + t_{k+1})/2$, whenever the polyline P_1 suggests an inflection point along P_2.[4] Consequently, the number of spline segments can increase from N to $N_e \geqslant N$.

For any edge e_k between $\mathbf{p}_k$ and $\mathbf{p}_{k+1}$, $k = 0, \dots, N_e - 1$, we consider the extensions of e_{k-1} and e_{k+1}, starting from $\mathbf{p}_k$ and $\mathbf{p}_{k+1}$, respectively.[5] If $\mathbf{p}_k$, $\mathbf{p}_{k-1}$, and $\mathbf{p}_{k+1}$ are aligned, the left extension for e_k and the right one for e_{k-1} are both defined as the straight line with origin in $\mathbf{p}_k$ orthogonal to e_k, as shown in Figure 16. If the two extensions related to e_k intersect and generate with e_k a

---

[3] Two additional "phantom" points are considered to properly define $\mathbf{T}_0, k_0, \mathbf{T}_N, k_N$.

[4] If $\mathbf{p}_k$ is one of these additional points, the corresponding unit tangent required by the G^1 interpolation scheme is defined by applying a small rotation to e_k to obtain the desired change of sign in the curvature.

[5] For the first and last edge the left and right extensions are defined as the half line orthogonal to the edge with origin in the first and last point of P_1 respectively, see Figure 16 (top left).

triangle included in Ω_{free}, this triangle defines the admissible region associated with e_k. Otherwise, the admissible region R_k related to e_k is defined as the highest trapezoid or parallelogram included in Ω_{free} with a basis coincident with e_k and the other two opposite edges on the extensions. The complete set of possible cases related to a certain segment is shown in Figure 16. The three plots in the first row of the figure show the triangular cases. However, the shape of the admissible region may be determined by the intersection (if any) of the lateral extensions with either the bounding box of the scenario or one (or more) obstacles (see the three plots in the second row of Figure 16). In this case, one edge of the convex polygon that determines the region is the segment parallel to e_k through the aforementioned intersection points with extrema at the intersections with the left and right extension.

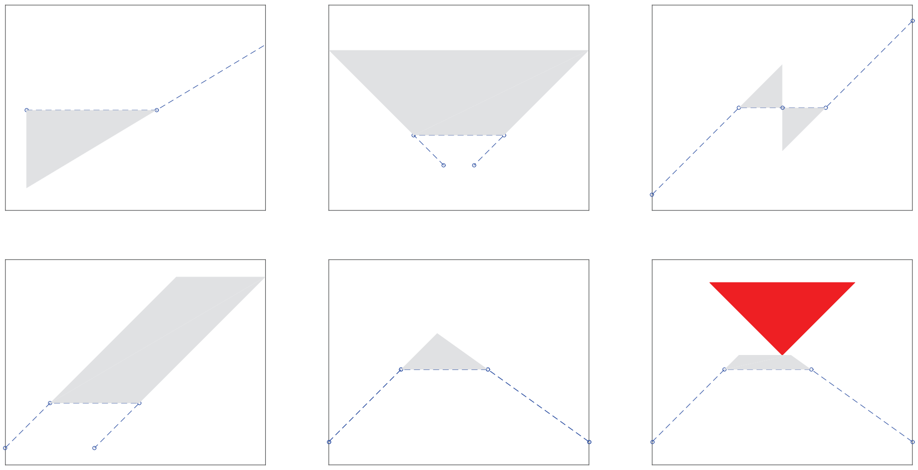


Figure 16. Admissible regions: possible cases.

The pictures on the left in Figures 17, 18, and 19 show in grey the admissible regions associated with the three polylines MTMP, VGSP, MVGSP/GVGP for the simple scenario S1 introduced in Figure 5. The height H_k of the admissible region related to e_k is then suitably taken into account to determine the value of the corresponding tension parameter σ_k, see Section 3.3.1.

	MTMSP	VGSP	MVGSP /GVGP
T1	9.8132	8.8506	9.8329
T2	9.8606	8.8599	9.8860

Table 3. Total arc lengths of the P_2 paths for scenario S1 shown in Figure 17-18-19 respectively in the first, second and third column.

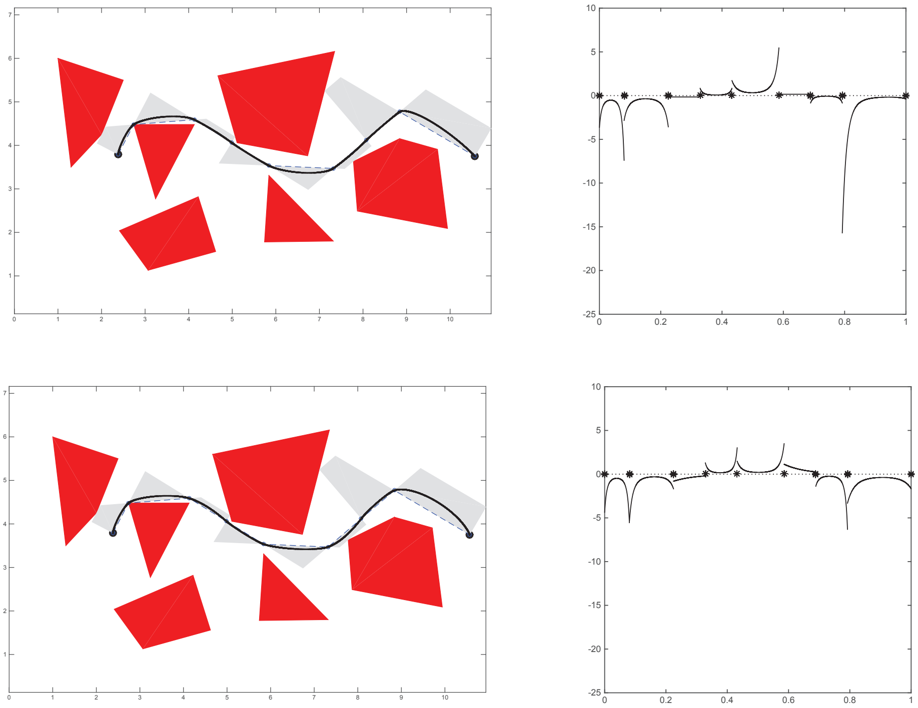


Figure 17. G^1 PH quintic spline paths (solid line, left) obtained from the MTMSP path (dashed line, left) for scenario S1 by using T1 (top) and T2 (bottom) tangents are shown with the corresponding admissible regions (grey). The curvature plots of the two spline paths are also shown (right).

The results related to the final P_2 paths produced by our approach for the simple scenario S1 are presented in Figures 17–19 and Table 3. Figure 17 shows the P_2 paths obtained when P_1 is the modified trapezoidal map shortest path (MTMSP) shown in Figure 8 (bottom). The paths corresponding to the T1 and T2 choice for the unit tangents are shown, see Remark 4. The corresponding total arc lengths are given in the first column of Table 3. The related curvature plots are also shown in Figure. 17. Note how the T2 choice is able to reduce the curvature extrema of the corresponding interpolant. Analogously, Figure 18 and 19 shows the P_2 paths obtained from the P_1 paths generated with the VGSP and MVGSP/GVGP algorithms, previously shown in Figure 10, with the T1 and T2 choice for the unit tangents, respectively. The corresponding total arc lengths are reported in the last two columns of Table 3. As in the previous case, the curvature plots of the paths obtained using T2 exhibit smaller variations with respect to those constructed using the T1 alternative — see the plots on the right of these two figures.

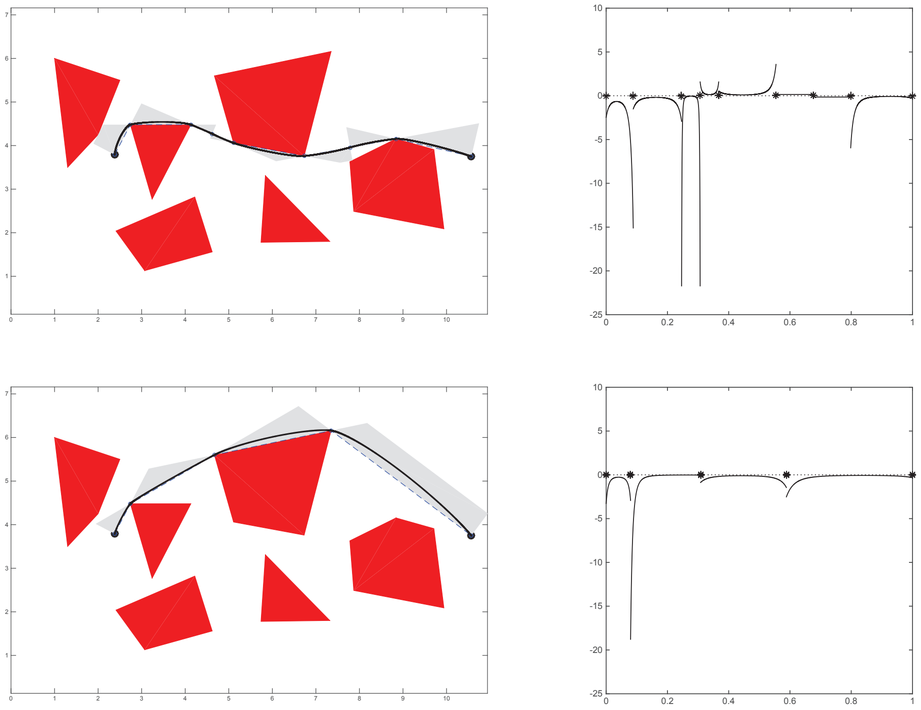


Figure 18. G^1 PH quintic spline paths (solid lines, left) obtained from VGSP (dashed lines, top left) and MVGSP/GVGP (dashed line, bottom left) for scenario S1 by using T1 tangents are shown with the corresponding admissible regions (grey). The curvature plots of the two spline paths are also shown (right).

For comparison, Figure 20, 21 and 22 show the results obtained by considering the different piecewise linear solutions together with cubic G^1 splines according to the choice of T1 and T2 unit tangents. The values of the tension parameters are chosen as in the corresponding PH quintic solution related to the same piecewise linear path and shown in Figures 17, 18 and 19. We may note that PH splines exhibit a smaller curvature range, a benefit from the fairness point of view, in all test cases.

3.2.2 G^1 *Asymptotic analysis*

Since we are focusing on a generic PH spline segment, we omit in the following the k subscript from the notation. Furthermore, by choosing the local coordinate system already used in (3.12) to represent the unit tangents, we can also assume without loss of generality

$$\Delta \mathbf{p} = m \exp(i 0)\,, \tag{3.15}$$

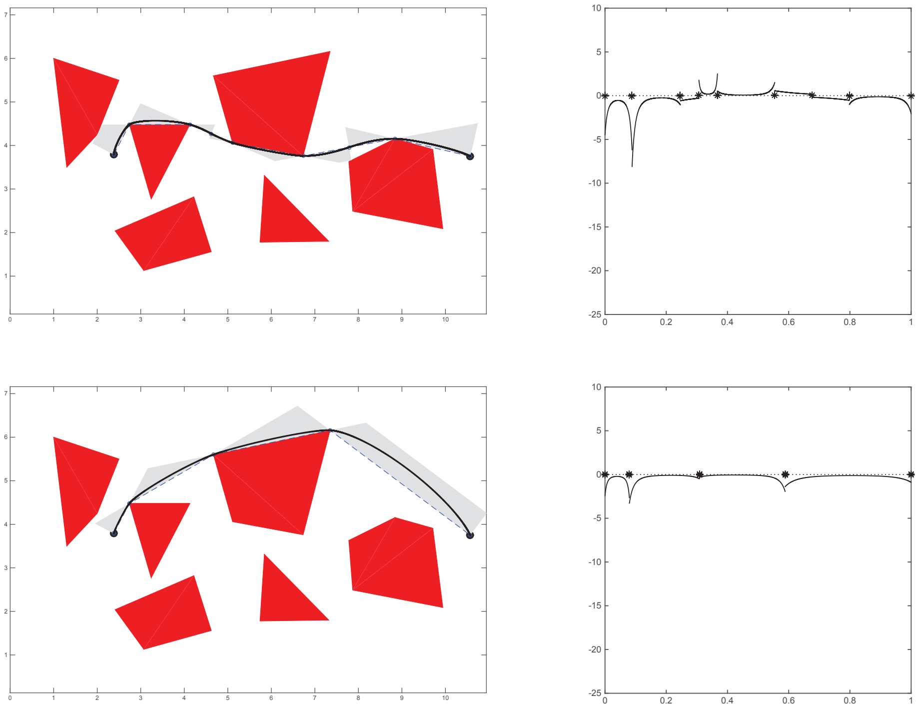


Figure 19. G^1 PH quintic spline paths (solid lines, left) obtained from VGSP (dashed lines, top left) and MVGSP/GVGP (dashed line, bottom left) for scenario S1 by using T2 tangents are shown with the corresponding admissible regions (grey). The curvature plots of the two spline paths are also shown (right).

where $m > 0$ denotes the modulus of the displacement. Now from (3.13) we have that

$$\mathbf{u} = \sigma\, c\, \exp(i\, \alpha)\,, \quad \mathbf{v} = \sigma\, d\, \exp(i\, \beta)\,, \tag{3.16}$$

where c, d are fixed positive quantities and α and β are fixed angles in $(-\pi/2\,,\, \pi/2]$, all not depending on σ (see Subsection 3.2). The other complex quantity which is necessary to define the control polygon is $\mathbf{z}$ and it is defined as the solution with maximal real part of the quadratic equation given in (3.10). The following proposition gives the Taylor expansion of $\gamma = \gamma(\sigma)$ and $\phi = \phi(\sigma)$, where

$$\mathbf{z} = \gamma \exp(i\, \phi)\,, \tag{3.17}$$

with $\gamma \in \mathbb{R}$ and $\phi \in (-\pi/2\,,\, \pi/2]$.

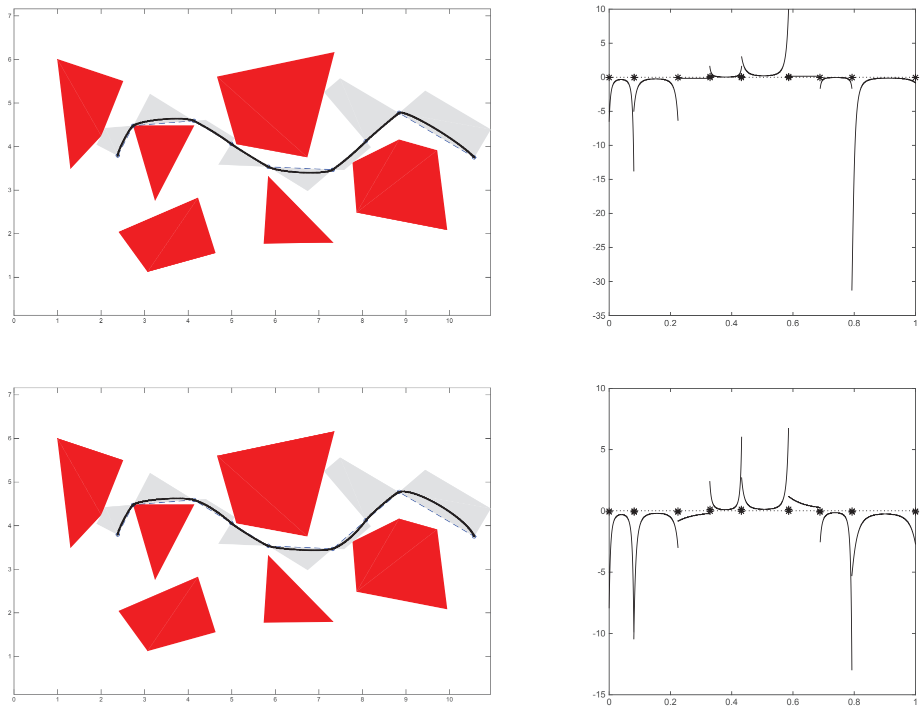


Figure 20. G^1 cubic spline paths (solid lines, left) obtained from MTMSP (dashed lines, left) for scenario S1 by using T1 (top) and T2 (bottom) tangents are shown with the corresponding admissible regions (grey). The curvature plots of the two spline paths are also shown (right).

Proposition 1 *The angle* $\phi = \phi(\sigma)$ *and the scalar* $\gamma = \gamma(\sigma)$ *defining the polar form of* z *as in (3.17) have the following Taylor expansions*

$$\phi(\sigma) = f_0 + f_1\sigma + \frac{1}{2}f_2\sigma^2 + \mathcal{O}(\sigma^3) \tag{3.18}$$

and

$$\gamma(\sigma) = g_0 + g_1\sigma + \mathcal{O}(\sigma^2), \tag{3.19}$$

where

$$\begin{aligned} f_0 &= 0, \quad f_1 = -\frac{3b}{\sqrt{8q}}, \quad f_2 = -\frac{p}{q}f_1, \quad g_0 = \sqrt{\frac{q}{2}}, \\ g_1 &= -\frac{p}{\sqrt{8q}} - \frac{3}{4}a - \frac{p}{3b}, \quad \textit{if } b \neq 0, \end{aligned} \tag{3.20}$$

with

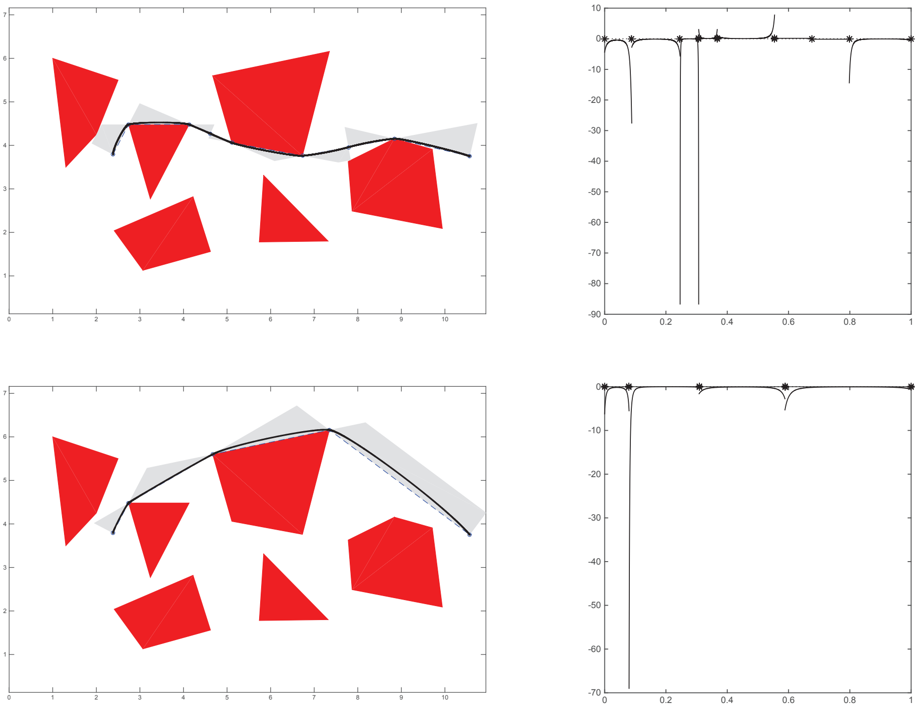


Figure 21. G^1 cubic spline paths (solid lines, left) obtained from VGSP (dashed line, top left) and MVGSP/GVGP (dashed line, bottom left) for scenario S1 by using T1 tangents are shown with the corresponding admissible regions (grey). The curvature plots of the two spline paths are also shown (right).

$$\begin{cases} a = c\cos\alpha + d\cos\beta\,, \\ b = c\sin\alpha + d\sin\beta\,, \\ p = 3(c^2\sin 2\alpha + d^2\sin 2\beta) + cd\sin(\alpha+\beta)\,. \\ q = \frac{15}{\ell}m\,. \end{cases} \tag{3.21}$$

Proof : The complex equation (3.10) can be rewritten as the following (nonlinear) system in the real unknowns γ and ϕ, defined by two scalar equations that correspond to the real and the imaginary part of (3.10),

$$\begin{cases} x_2\gamma^2 + x_1\gamma + x_0 & = 0\,, \\ y_2\gamma^2 + y_1\gamma + y_0 & = 0\,, \end{cases} \tag{3.22}$$

where

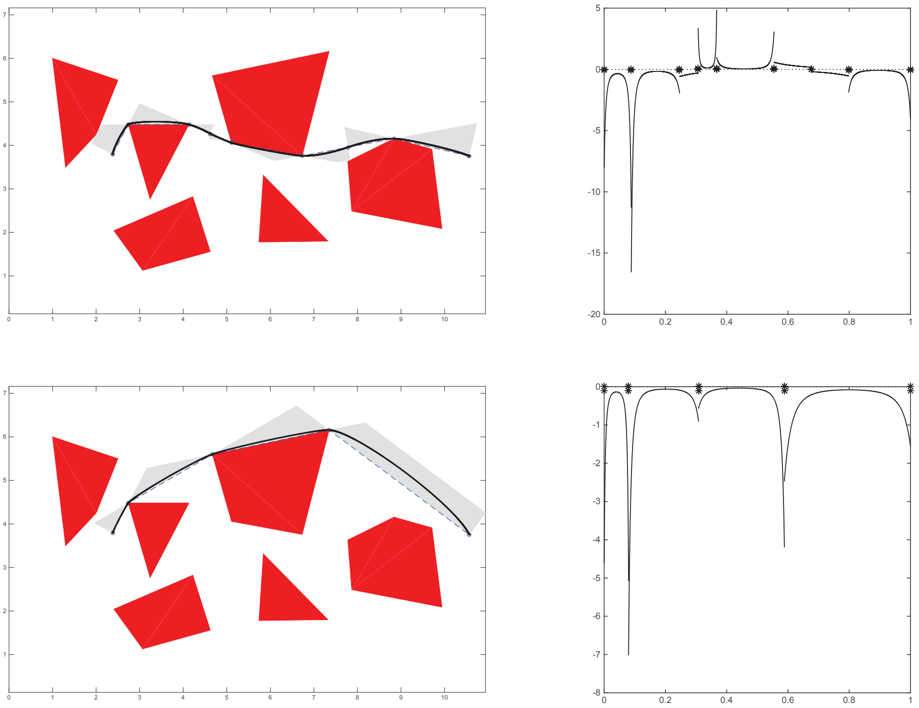


Figure 22. G^1 cubic spline paths (solid lines, left) obtained from VGSP (dashed line, top left) and MVGSP/GVGP (dashed line, bottom left) for scenario S1 by using T2 tangents are shown with the corresponding admissible regions (grey). The curvature plots of the two spline paths are also shown (right).

$$\begin{aligned} x_2 &= 2\cos 2\phi\,, \\ x_1 &= 3\,(\mathrm{Re}\,(\mathbf{u}+\mathbf{v})\cos\phi - \mathrm{Im}\,(\mathbf{u}+\mathbf{v})\sin\phi) \\ &= 3\,\sigma[a\cos\phi - b\sin\phi]\,, \\ x_0 &= \left(3\mathrm{Re}\,(\mathbf{u}^2+\mathbf{v}^2) + \mathrm{Re}\,(\mathbf{u}\mathbf{v}) - \tfrac{15}{\ell}m\right) \;=\; \sigma^2\,e \,-\, q\,, \end{aligned} \tag{3.23}$$

with $e = 3(c^2\cos 2\alpha + d^2\cos 2\beta) + cd\cos(\alpha+\beta)$, and

$$\begin{aligned} y_2 &= 2\sin 2\phi\,, \\ y_1 &= 3\,(\mathrm{Im}\,(\mathbf{u}+\mathbf{v})\cos\phi + \mathrm{Re}\,(\mathbf{u}+\mathbf{v})\sin\phi) \\ &= 3\,\sigma[b\cos\phi + a\sin\phi]\,, \\ y_0 &= \left(3\mathrm{Im}\,(\mathbf{u}^2+\mathbf{v}^2) + \mathrm{Im}\,(\mathbf{u}\mathbf{v})\right) \;=\; \sigma^2\,p\,. \end{aligned} \tag{3.24}$$

Note that, when σ vanishes, we have $x_1 = y_1 = y_0 = 0$ and $x_0 = -q$. From the second equation in (3.22), we deduce that y_2 must vanish, namely $\phi = 0$ or $\phi = \pi/2$. However, in order to get a real value for γ from the first equation in

(3.22), only $\phi = 0$ is acceptable which implies that $f_0 = 0$ in (3.18). In addition, since we select the solution of (3.10) with maximal real part, we have

$$g_0 = \gamma(0) = \sqrt{\frac{q}{2}}\,, \tag{3.25}$$

with q defined as in (3.21). In order to compute the other Taylor coefficients in (3.18), we use the resultant method [47] from which we derive that γ is a real solution of the system in (3.22) if and only if the Sylvester matrix

$$A = \begin{pmatrix} x_2 & x_1 & x_0 & 0 \\ 0 & x_2 & x_1 & x_0 \\ y_2 & y_1 & y_0 & 0 \\ 0 & y_2 & y_1 & y_0 \end{pmatrix}$$

is singular and in such case it can be expressed as follows,

$$\gamma(\sigma) \;=\; -\frac{x_2 y_0 - x_0 y_2}{x_2 y_1 - x_1 y_2}\,. \tag{3.26}$$

Hence, we have to require that $\det(A)$ vanishes, that is

$$\begin{aligned} 0 =\, & x_2 \left[x_0 \left(y_1^2 - y_0 y_2\right) + y_0 \left(x_2 y_0 - x_1 y_1\right)\right] \\ & + y_2 \left[x_0 \left(x_0 y_2 - x_1 y_1\right) + y_0 \left(x_1^2 - x_0 x_2\right)\right]. \end{aligned} \tag{3.27}$$

Now, from the Taylor expansion of $\phi(\sigma)$ in (3.18) we have

$$\begin{aligned} \sin\phi &= f_1\,\sigma + \tfrac{1}{2} f_2 \sigma^2 + \mathcal{O}(\sigma^3)\,, \\ \cos\phi &= 1 - \tfrac{1}{2} f_1^2 \sigma^2 + \mathcal{O}(\sigma^3)\,, \\ \sin 2\phi &= 2 f_1\,\sigma + f_2 \sigma^2 + \mathcal{O}(\sigma^3)\,, \\ \cos 2\phi &= 1 - 2 f_1^2 \sigma^2 + \mathcal{O}(\sigma^3)\,. \end{aligned} \tag{3.28}$$

From (3.23) and (3.24), we can then obtain the following expansions for $x_i, i = 0, 1, 2$,

$$\begin{aligned} x_2 &= 2 - 4 f_1^2 \sigma^2 + \mathcal{O}(\sigma^3)\,, \\ x_1 &= 3a\,\sigma - 3b f_1\,\sigma^2 + \mathcal{O}(\sigma^3)\,, \\ x_0 &= \sigma^2\, e - \, q\,, \end{aligned} \tag{3.29}$$

and $y_i, i = 0, 1, 2$,

$$\begin{aligned} y_2 &= 4 f_1\,\sigma + 2 f_2 \sigma^2 + \mathcal{O}(\sigma^3)\,, \\ y_1 &= 3b\,\sigma + 3a f_1\,\sigma^2 + \mathcal{O}(\sigma^3)\,, \\ y_0 &= \sigma^2\, p\,, \end{aligned} \tag{3.30}$$

respectively. Thus, with some computation, we can rewrite the equation in (3.27) as follows,

$$2q(-9b^2 + 8qf_1^2)\,\sigma^2 + 16f_1 q\,(f_2 q + f_1 p)\,\sigma^3 + \mathcal{O}(\sigma^4) \;=\; 0\,,$$

which implies that f_1 and f_2 in (3.18) are defined as in (3.20), unless for the sign uncertainty in the f_1 formula. By using (3.30) to compute the first order expansion for $\gamma = \gamma(\sigma)$ defined in (3.26), when $b \neq 0$ we get

$$\gamma(\sigma) = -\frac{2q}{3b}\,f_1 - \left(\frac{2aqf_1^2}{3b^2} + \frac{f_2 q + p}{3b}\right)\sigma + \mathcal{O}(\sigma^2)\,. \tag{3.31}$$

Note that this confirms that the sign of f_1 has to be fixed as in (3.20), since we already know that $\gamma(0) = g_0 = \sqrt{\frac{q}{2}}$. Furthermore, by replacing the selected expressions of f_1 and f_2 in (3.31), we get the expression of g_1 reported in (3.20) when $b \neq 0$. ■

Theorem 1 *If both unit tangents $\boldsymbol{T}_j$, $j = k-1, k$ can be expressed as a linear combination of $\Delta \boldsymbol{p}_j$ and $\Delta \boldsymbol{p}_{j+1}$ with nonnegative coefficients,*[6] *when the tension parameter σ_k is small enough, the PH spline segment $\boldsymbol{r}_k$ belongs to the corresponding convex admissible region R_k and, consequently, to the free configuration space.*

Proof : Since $\mathbf{q}_{k,0} = \mathbf{p}_{k-1}$ and $\mathbf{q}_{k,5} = \mathbf{p}_k$, being R_k the convex admissible region defined as specified in the previous section, we need to check that for a sufficiently small value of σ_k all the other control points $\mathbf{q}_{k,j}, j = 1, \dots, 4$ belong to R_k (in view of the convex hull property of Bézier curves, this also implies $\mathbf{r}_k \in R_k$). We again omit the k subscript from the notation and we consider the local coordinate system already used in (3.12) to represent the unit tangents. This also implies, without loss of generality, that $\Delta\mathbf{p}$ can be assumed as in (3.15).

By using (3.16), (3.17) and the expansions in (3.18)–(3.20) and (3.28), from (3.9) the following expressions of the control polygon legs can be derived,

$$\begin{aligned}
(\mathbf{q}_1 - \mathbf{q}_0) &= \frac{\ell}{5} c^2\,\sigma^2\,\exp(i\,2\alpha)\,,\\
(\mathbf{q}_2 - \mathbf{q}_1) &= \frac{\ell}{5} c\,\sigma\,[g_0 \cos\alpha + (g_1 \cos\alpha - g_0 f_1 \sin\alpha)\,\sigma)\\
&\quad + i\,(g_0 \sin\alpha + (g_1 \sin\alpha + g_0 f_1 \cos\alpha)\,\sigma)\,] + \mathcal{O}(\sigma^3),\\
(\mathbf{q}_4 - \mathbf{q}_3) &= \frac{\ell}{5} d\,\sigma[\,(g_0 \cos\beta + (g_1 \cos\beta - g_0 f_1 \sin\beta)\,\sigma)\\
&\quad - i\,(g_0 \sin\beta + (g_1 \sin\beta + g_0 f_1 \cos\beta)\,\sigma)\,] + \mathcal{O}(\sigma^3),\\
(\mathbf{q}_5 - \mathbf{q}_4) &= \frac{\ell}{5} d^2\,\sigma^2\,\exp(i\,2\beta),
\end{aligned}$$

---

[6] For the first and last segments we assume that two suitable phantom points are fixed to define $\Delta\mathbf{p}_0$ and $\Delta\mathbf{p}_{N+1}$.

which immediately imply that $(\mathbf{q}_3 - \mathbf{q}_2) = \Delta\mathbf{p} + \mathcal{O}(\sigma)$. Considering that $\exp(i\,2\alpha)$ and $\exp(i\,2\beta)$ are aligned with the left and right unit tangents, respectively, and taking into account the definition of the convex admissible region R, thanks to the hypothesis on the unit tangents orientation, from the first and last of these relations we can derive that for σ sufficiently small both $\mathbf{q}_1$ and $\mathbf{q}_4$ belong to R. On the other hand, the leading term of $(\mathbf{q}_2 - \mathbf{q}_1)$ and of $(\mathbf{q}_4 - \mathbf{q}_3)$ are oriented as $\exp(i\,\alpha)$ and $\exp(i\,\beta)$, respectively . From the choice for α and β fixed in (3.14), since $(\mathbf{q}_3 - \mathbf{q}_2) = \Delta\mathbf{p} + \mathcal{O}(\sigma)$, we can obtain that also $\mathbf{q}_2$ and $\mathbf{q}_3$ belong to R when σ is sufficiently small. ■

3.2.3 G^1 *Automatic selection of the tension parameter*

Let us now derive a reasonable automatic selection for σ_k using the previous analysis. As in the previous subsection, for brevity we omit the subscript k. According to (3.9), $\mathbf{q}_2 - \mathbf{q}_1$ and $\mathbf{q}_4 - \mathbf{q}_3$, for a fixed value of σ are aligned with $\exp(i\,(\alpha + \phi))$ and with $\exp(i\,(\beta + \phi))$, respectively. Thus, in order to control these orientations, when b is not zero, we first require that

$$\sigma \leqslant \min\{\sigma_L\,,\,\sigma_R\}$$

where $\sigma_L = |\alpha|/|f_1|$ and $\sigma_R = |\beta|/|f_1|$. This requires that the absolute value of the first order expansion of $\phi(\sigma)$ is less or equal to both $|\alpha|$ and $|\beta|$. In this way $\mathbf{q}_2 - \mathbf{q}_1$ and $\mathbf{q}_4 - \mathbf{q}_3$ have intermediate orientations between the left unit tangent and $\Delta\mathbf{p}$ and between $\Delta\mathbf{p}$ and the right one, respectively.

Furthermore, in order to keep the control polygon inside the convex admissible region R, we determine another upper bound for σ aimed to control the distances D_j of all the control points $\mathbf{q}_j, j = 1, \ldots, 4$ from $\Delta\mathbf{p}$. In particular, we determine a bound for σ which guarantees that the second order Taylor expansions of such distances from $\Delta\mathbf{p}$ are less or equal than the height H of R (see the previous section). Considering that in the adopted local reference system the distances of the inner control points from the displacement are simply given by the absolute values of their imaginary parts, we have that

$$\begin{aligned}
D_1 &= \frac{\ell}{5}\, c^2\, \sigma^2\, |\sin 2\alpha|\,,\\
D_2 &= \sigma\, \frac{\ell}{5}\, c\, |g_0 \sin\alpha + (c\,\sin 2\alpha \\
&\quad + g_1 \sin\alpha + g_0\, f_1\,\cos\alpha)\,\sigma + \mathcal{O}(\sigma^2)|\,,\\
D_3 &= \sigma\, \frac{\ell}{5}\, d\, |g_0 \sin\beta + (d\,\sin 2\beta \\
&\quad + g_1 \sin\beta + g_0\, f_1\,\cos\beta)\,\sigma + \mathcal{O}(\sigma^2)|\,,\\
D_4 &= \frac{\ell}{5}\, d^2\, \sigma^2\, |\sin 2\beta|\,.
\end{aligned}$$

In order to further simplify the analysis, by denoting with $E(D_i)$, $i = 2, 3$, the expressions of $D_i, i = 2, 3$, without considering the rest of the expansion, the following inequalities can easily be obtained,

$$E(D_2) \leqslant \sigma(Q_{21} + \sigma(Q_{22} + Q_{23}))\,,$$
$$E(D_3) \leqslant \sigma(Q_{31} + \sigma\,(Q_{32} + Q_{33}))\,,$$

where,

$$\begin{aligned} Q_{21} &= \frac{\ell}{5}\, c\, g_0 |\sin\alpha|\,, \quad Q_{22} = \frac{\ell}{5}\, c^2 |\sin 2\alpha|\,, \\ Q_{23} &= \frac{\ell}{5}\, c\, |g_1 \sin\alpha + g_0\, f_1\, \cos\beta|\,, \quad Q_{31} = \frac{\ell}{5}\, d\, g_0 |\sin\beta|\,, \\ Q_{32} &= \frac{\ell}{5}\, d^2 |\sin 2\beta|\,, \quad Q_{33} = \frac{\ell}{5}\, d\, |g_1 \sin\beta + g_0\, f_1\, \cos\beta|\,. \end{aligned}$$

Relating for brevity just to $E(D_2)$, when b is not zero, we first require that $\sigma\, Q_{22} \leqslant Q_{21}/2$ and $\sigma\, Q_{23} \leqslant Q_{21}/2$ and then that $\sigma\, Q_{21} \leqslant H/2$, that is $\sigma \leqslant \min\{\sigma_{L1}\,, \sigma_{L2}\,, \sigma_{L3}\}$, with

$$\sigma_{L1} = \frac{g_0}{4c|\cos\alpha|}\,, \quad \sigma_{L2} = \frac{g_0 |\sin\alpha|}{|g_1 \sin\alpha + g_0\, f_1\, \cos\alpha|}\,,$$
$$\sigma_{L3} = \frac{5}{\ell}\, \frac{H}{2c g_0 |\sin\alpha|}\,.$$

Note that any $\sigma_{Lj}, j = 1, 2, 3$, has to be set to one if the corresponding denominator vanishes. Summarizing, when b is not zero, we require

$$\sigma \leqslant \min\{1\,, \sigma_L\,, \sigma_R\,, \sigma_{L1}\,, \sigma_{L2}\,, \sigma_{L3}\,, \sigma_{R1}\,, \sigma_{R2}\,, \sigma_{R3}\}\,, \tag{3.32}$$

where the thresholds $\sigma_{Rj}, j = 1, 2, 3$, are defined analogously to $\sigma_{Lj}, j = 1, 2, 3$, just replacing β and d with α and c, respectively. In the special case $b = 0$ we have to consider only the first order term of the distances because we are not able to derive g_1. In this case, we just consider $\sigma \leqslant \min\{1\,, \sigma_{L3}\,, \sigma_{R3}\}$.

In our numerical experiments, the use of an upper bound according to the previous analysis for selecting σ in each spline segment has always ensured the identification of a control polygon of reasonable shape included in the corresponding admissible region.

3.3 SMOOTHING WITH G^2 PH QUINTICS WITH TENSION PARAMETERS

The second strategy adopted for our second step relies on the G^2 PH quintic spline with tension parameter, hence the related scheme introduced in [32] is here briefly summarized and its limit behaviour is analyzed in Subsection 3.3.1. Then in Subsection 3.3.2 an automatic strategy for a suitable selection of the tension parameters for the G^2 case is introduced.

Considering the representation in Bézier form introduced in Section 3.1.2, the G^2 smoothness is ensured as in [68] by requiring that the curve is $C^1 \cap G^2$. This leads us to the following assumptions,

$$\begin{aligned} \mathbf{u}_k &= 4\,(\ell_k \mathbf{z}_{k-1} + \ell_{k-1}\mathbf{z}_k)/a_{k-1}\,,\ k = 2,\dots,N, \\ \mathbf{v}_k &= 4\,(\ell_{k+1}\mathbf{z}_k + \ell_k \mathbf{z}_{k+1})/a_k\,,\ k = 1,\dots,N-1, \end{aligned} \tag{3.33}$$

where a_k is written in terms of the tension parameters ν_k as $a_k = 4(\ell_k + \ell_{k+1}) + \nu_k\sqrt{\ell_k \ell_{k+1}}$. The complex coefficients $\mathbf{u}_1$ and $\mathbf{v}_N$ are fixed in order to satisfy suitable boundary conditions. As here we are interested just in open curves, we relate to the first kind of boundary conditions proposed in [32], which guarantee cubic end spans. Such conditions are imposed by requiring that,

$$\begin{aligned} \mathbf{u}_1 &= 4\,(a_0 \mathbf{z}_1 - \ell_1 \mathbf{z}_2)/a_1\,, \\ \mathbf{v}_N &= 4\,(-\ell_N \mathbf{z}_{N-1} + a_N \mathbf{z}_N)/a_{N-1}\,, \end{aligned} \tag{3.34}$$

where $a_0 = 2\ell_1 + \ell_2 + \nu_0\sqrt{\ell_1\ell_2}$, $a_N = 2\ell_N + \ell_{N-1} + \nu_N\sqrt{\ell_{N-1}\ell_N}$ and

$$\nu_0 = \frac{\nu_1}{2}, \qquad \nu_N = \frac{\nu_{N-1}}{2}. \tag{3.35}$$

By combining (3.10) with (3.33), a homogeneous quadratic equation in the complex unknowns $\mathbf{z}_{k-1}, \mathbf{z}_k$ and $\mathbf{z}_{k+1}$ can be derived, namely

$$\alpha_k \mathbf{z}_{k-1}^2 + (2+\beta_k)\mathbf{z}_k^2 + \gamma_k \mathbf{z}_{k+1}^2 + \delta_k \mathbf{z}_{k-1}\mathbf{z}_{k+1} + \epsilon_k \mathbf{z}_{k-1}\mathbf{z}_k + \zeta_k \mathbf{z}_k \mathbf{z}_{k+1} = \frac{15}{\ell_k}\Delta\mathbf{p}_k\,, \tag{3.36}$$

for $k = 2,\dots,N-1$, where

$$\begin{aligned} \alpha_k &= 48\frac{\ell_k^2}{a_{k-1}^2}\,, \qquad \gamma_k = 48\frac{\ell_k^2}{a_k^2}\,, \qquad \delta_k = 16\frac{\ell_k^2}{a_{k-1}a_k}\,, \\ \beta_k &= 12\left(\frac{\ell_{k-1}}{a_{k-1}} + \frac{\ell_{k+1}}{a_k}\right) + 16\left(\frac{\ell_{k-1}\ell_{k+1}}{a_{k-1}a_k} + 3\frac{\ell_{k-1}^2}{a_{k-1}^2} + 3\frac{\ell_{k+1}^2}{a_k^2}\right), \\ \epsilon_k &= 12\frac{\ell_k}{a_{k-1}} + 16\left(\frac{\ell_k\ell_{k+1}}{a_{k-1}a_k} + 6\frac{\ell_{k-1}\ell_k}{a_{k-1}^2}\right), \\ \zeta_k &= 12\frac{\ell_k}{a_k} + 16\left(\frac{\ell_{k-1}\ell_k}{a_{k-1}a_k} + 6\frac{\ell_k\ell_{k+1}}{a_k^2}\right). \end{aligned}$$

Finally, two additional equations can be obtained by combining (3.10) for $k = 1$ and $k = N$ with (3.33) and (3.34),

$$\begin{aligned} (20+\beta_1)\mathbf{z}_1^2 + \gamma_1\mathbf{z}_2^2 + \zeta_1\mathbf{z}_1\mathbf{z}_2 &= \tfrac{15}{\ell_1}\Delta\mathbf{p}_1\,, \\ \alpha_N\mathbf{z}_{N-1}^2 + (20+\beta_N)\mathbf{z}_N^2 + \epsilon_N\mathbf{z}_{N-1}\mathbf{z}_N &= \tfrac{15}{\ell_N}\Delta\mathbf{p}_N, \end{aligned} \tag{3.37}$$

where

$$\begin{aligned} \beta_1 &= 4\left(12\frac{a_0^2}{a_1^2} - 3 + 4\frac{\ell_2 a_0}{a_1^2} + 12\frac{\ell_2^2}{a_1^2}\right), \quad \alpha_N = 80\frac{\ell_N^2}{a_{N-1}^2}, \\ \gamma_1 &= 80\frac{\ell_1^2}{a_1^2}, \quad \zeta_1 = \frac{\ell_1(\ell_2 - a_0)}{a_1^2}, \quad \epsilon_N = \frac{\ell_N(\ell_{N-1} - a_N)}{a_{N-1}^2}, \\ \beta_N &= 4\left(12\frac{a_N^2}{a_{N-1}^2} - 3 + 4\frac{\ell_{N-1} a_N}{a_{N-1}^2} + 12\frac{\ell_{N-1}^2}{a_{N-1}^2}\right). \end{aligned}$$

As already stated in [1], the set of solutions of the quadratic non–linear system in (3.36) and (3.37) generates 2^{N-1} different PH spline interpolants with cubic end span but a suitable one can be obtained by using the Newton–Raphson method with a careful initialization strategy, a key issue to ensure the convergence of the method to the "good" solution [32].

3.3.1 G^2 *Asymptotic analysis*

The procedure discussed in 3.2.1 is used to associate to $\Delta\mathbf{p}_k$ a certain area R_k that identifies an *admissible region* for the k–th segment of the PH spline. Clearly, thanks to the convex hull property of Bézier curves, if all the inner control points $\mathbf{q}_{k,j}, j = 1, \ldots, 4$, are in convex region R_k, so is $\mathbf{r}_k$. Thus, if the distance of all $\mathbf{q}_{k,j}, j = 1, \ldots, 4$, from $\Delta\mathbf{p}_k$ is less then the positive height H_k of T_k and the control polygon has a suitable shape — its edges are suitably orientated with respect to $\Delta\mathbf{p}_j, j = k-1, k, k+1$ — we can conclude that the k–th segment of the curve belongs to the corresponding admissible region. We then need to analyze in detail the limit configuration of the Bézier control points $\mathbf{q}_{k,j}, j = 0, \ldots, 5$, $k = 1, \ldots, N$, when the values $\sigma_k = 1/\nu_k$, $k = 0, \ldots, N$, tend to zero. Even if this analysis was already described in [32], we present it here in a different and more detailed form tailored to the application problem of interest.

By taking into account the expressions of the coefficients in equations (3.36) and (3.37), it is not difficult to prove that, for any solution $\mathbf{z}_k$, we have

$$\lim_{\sigma_{k-1},\sigma_k \to 0} \mathbf{z}_k^2 = c_k^2\, \Delta\mathbf{p}_k, \qquad k = 1, \ldots, N,$$

where $c_k = \sqrt{15/(2\ell_k)}, k = 2, \ldots, N-1$, and $c_1 = \sqrt{3/(4\ell_1)},\ c_N = \sqrt{3/(4\ell_N)}$. In particular, we are interested in the solution $\mathbf{z}_k, k = 1, \ldots, N$, of the quadratic non–linear system that satisfies

$$\lim_{\sigma_{k-1},\sigma_k \to 0} \mathbf{z}_k = c_k \sqrt{|\Delta\mathbf{p}_k|}\, \exp\left(i\frac{\theta_k}{2}\right), \; k = 1, \ldots, N, \tag{3.38}$$

where

$$\exp(i\,\theta_k) = \Delta\mathbf{p}_k/|\Delta\mathbf{p}_k|, \quad k = 1, \ldots, N, \quad -\pi < \theta_k \leqslant \pi.$$

In view of (3.9), the following lemma, corollary and proposition allow us to understand which is the limit shape of the first two edges of the control polygon associated with the k–th segment. As the definition in (3.9) is fully symmetric, similar considerations can be made for the last two edges of the control polygon.

Lemma 1 *If* $2 \leqslant k \leqslant N$ *and* $\theta_{k-1} \neq \theta_k$*, we have*

$$\frac{1}{2}\min\{\theta_{k-1}\,,\,\theta_k\} < \arg(\boldsymbol{U}_k^{(0)}) < \frac{1}{2}\max\{\theta_{k-1}\,,\,\theta_k\}\,, \tag{3.39}$$

where the function $\arg : \mathbb{C} \to \mathbb{R}$ *defines the argument of a complex number in* $(-\pi\,,\,\pi]$ *and*

$$\boldsymbol{U}_k^{(0)} = \left(\lim_{\sigma_{k-2},\sigma_{k-1},\sigma_k \to 0} \frac{\boldsymbol{u}_k}{\sigma_{k-1}}\right). \tag{3.40}$$

Furthermore, $\arg(\boldsymbol{U}_k^{(0)}) = \theta_k$ *when* $\theta_{k-1} = \theta_k$*.*

Proof : From (3.33), we obtain:

$$\lim_{\sigma_{k-2},\sigma_{k-1},\sigma_k \to 0} \frac{\mathbf{u}_k}{\sigma_{k-1}} = 4\left(d_k^{(l)}\exp\left(i\frac{\theta_{k-1}}{2}\right) + d_k^{(r)}\exp\left(i\frac{\theta_k}{2}\right)\right),$$

with $d_k^{(l)} = c_{k-1}\sqrt{\frac{\ell_k}{\ell_{k-1}}}$ and $d_k^{(r)} = c_k\sqrt{\frac{\ell_{k-1}}{\ell_k}}$. This immediately implies the statement of the lemma. ∎

Corollary 1 *If* $2 \leqslant k \leqslant N$ *and* $\theta_{k-1} > \theta_k$*, we have*

$$\theta_{k-1} > \arg(\boldsymbol{U}_k^{(0)}\,\boldsymbol{U}_k^{(0)}) > \arg(\boldsymbol{U}_k^{(0)}\boldsymbol{z}_k) > \theta_k, \tag{3.41}$$

with $\boldsymbol{U}_k^{(0)}$ *defined as in (3.40). Conversely, if* $\theta_{k-1} < \theta_k$*, we have*

$$\theta_{k-1} < \arg(\boldsymbol{U}_k^{(0)}\,\boldsymbol{U}_k^{(0)}) < \arg(\boldsymbol{U}_k^{(0)}\boldsymbol{z}_k) < \theta_k.$$

Proof : We restrict the proof to the case $\theta_{k-1} > \theta_k$ (the other case may be derived in a similar way). The previous lemma implies

$$\frac{\theta_{k-1}}{2} > \arg(\mathbf{U}_k^{(0)}) > \frac{\theta_k}{2}.$$

Since

$$\begin{aligned} \arg(\mathbf{U}_k^{(0)}\,\mathbf{U}_k^{(0)}) &= 2\arg(\mathbf{U}_k^{(0)})\,, \\ \arg(\mathbf{U}_k^{(0)}\mathbf{z}_k) &= \arg(\mathbf{U}_k^{(0)}) + \frac{\theta_k}{2}\,, \end{aligned}$$

the statement is easily verified. ∎ The definition of the Bézier control points in

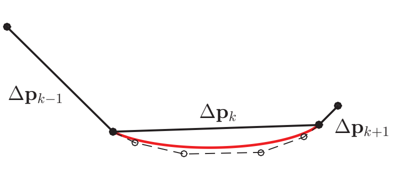


Figure 23. The shape of the control polygon for suitable selection of the tension parameters. The standard case is shown on the left (one spline segment), while the case related to an inflection point along the curve is shown on the right (two consecutive spline segments).

(3.9) together with the previous corollary imply that $(\mathbf{q}_{k,1} - \mathbf{q}_{k,0})$ has a limit orientation between $\Delta\mathbf{p}_{k-1}$ and $\Delta\mathbf{p}_k$. Analogously, $(\mathbf{q}_{k,2} - \mathbf{q}_{k,1})$ has a limit orientation between $(\mathbf{q}_{k,1} - \mathbf{q}_{k,0})$ and $\Delta\mathbf{p}_k$. Moreover, as either $\mathbf{u}_k$ or $\mathbf{v}_k$ tend to vanish when all the $\sigma_j, j = k-2, \dots, k+1$, tend to zero, the lengths of all the edges of the k–th control polygon tend to vanish, except for the one of $(\mathbf{q}_{k,3} - \mathbf{q}_{k,2})$ — since this edge tends to $\Delta\mathbf{p}_k$. This implies that the control polygon tends to flatten on $\Delta\mathbf{p}_k$. When $\theta_{k-1} = \theta_k$, from Lemma 3.3.1 we can only deduce that the limit orientation of $(\mathbf{q}_{k,j} - \mathbf{q}_{k,j-1}), j = 1, 2$ is the same of $\Delta\mathbf{p}_k$. In this case, in order to understand the corresponding orientations for sufficiently big values of the tension parameters ν_k, $k = 0, \dots, N$, we then need to consider the first order expansion of $\mathbf{z}_j, j = k-1, k$ and $\mathbf{u}_k$ in terms of σ (where we are assuming that $\sigma_j = s_j\,\sigma$ for some positive constants s_j). Since in equation (3.36) the coefficients α_k, δ_k and γ_k are of second order, while β_k, ϵ_k, and ζ_k are of first order, by using their first order approximations, $\beta_k \approx \hat{\beta}_k\sigma$, $\epsilon_k \approx \hat{\epsilon}_k\sigma$, and $\zeta_k \approx \hat{\zeta}_k\sigma$, where $\hat{\beta}, \hat{\epsilon}$ and $\hat{\zeta}$ are all positive, and

$$\mathbf{z}_j \;\approx\; \mathbf{z}_j^{(0)} + \sigma\,\mathbf{z}_j^{(1)}\,,$$

we obtain

$$\mathbf{z}_j^{(1)} \;=\; -\frac{\hat{\epsilon}_j}{2\hat{\beta}_j}\,\mathbf{z}_{j-1}^{(0)} - \frac{\hat{\zeta}_j}{2\hat{\beta}_j}\,\mathbf{z}_{j+1}^{(0)},$$

where $\mathbf{z}_k^{(0)} := \lim_{\sigma_{k-1},\sigma_k\to 0} \mathbf{z}_k$ as defined in (3.38). Thus, when $\theta_{k-1} = \theta_k$, we have

$$\begin{aligned}
\mathbf{z}_{k-1}^{(1)} &= -e_{k-1}\exp\left(i\,\frac{\theta_{k-2}}{2}\right) - f_{k-1}\exp\left(i\,\frac{\theta_k}{2}\right),\\
\mathbf{z}_{k}^{(1)} &= -e_{k}\exp\left(i\,\frac{\theta_{k}}{2}\right) - f_{k}\exp\left(i\,\frac{\theta_{k+1}}{2}\right),
\end{aligned}$$

with positive coefficients e_j and f_j, $j = k-1, k$. The first order expansion of $\mathbf{u}_k/\sigma_{k-1}$ can be then derived from (3.33) as follows

$$\frac{\mathbf{u}_k}{\sigma_{k-1}} \approx \mathbf{U}_k^{(0)} + \sigma\mathbf{U}_k^{(1)},$$

where

$$\begin{aligned}
\mathbf{U}_k^{(1)} = \; & -p_k\exp\left(i\,\tfrac{\theta_{k-2}}{2}\right) + g_k\exp\left(i\,\tfrac{\theta_k}{2}\right) +\\
& -r_k\exp\left(i\,\tfrac{\theta_{k+1}}{2}\right),
\end{aligned}$$

with both p_k and r_k positive. Hence, by considering the expressions of the products $\mathbf{U}_k^{(0)}\mathbf{U}_k^{(1)}$ and $\mathbf{U}_k^{(1)}\mathbf{z}_k^{(0)}$, we can formulate the following proposition.

Proposition 2 *When* $\theta_{k-1} = \theta_k$ *and* $(\theta_{k-2}-\theta_k)(\theta_{k+1}-\theta_k) > 0$, *for* σ *sufficiently small the component orthogonal to* $\Delta\boldsymbol{p}_k$ *of both* $(\boldsymbol{q}_{k,j}-\boldsymbol{q}_{k,j-1})$, *for* $j = 1, 2$, *is reversed with respect to the component orthogonal to* $\Delta\boldsymbol{p}_k$ *of* $\Delta\boldsymbol{p}_j$, *for* $j = k-2, k+1$.

Note that $\theta_{k-1} = \theta_k$ means that $\Delta\mathbf{p}_{k-1}$ and $\Delta\mathbf{p}_k$ are aligned while the assumption $(\theta_{k-2}-\theta_k)(\theta_{k+1}-\theta_k) > 0$ means that the successive four edges $\Delta\mathbf{p}_j$, for $j = k-2, \dots, k+1$, are Z–shaped (remind that, in view of the assumptions on our polyline, when $\theta_{k-1} = \theta_k$, this is the only possibility). Figure 23 shows the shape of the control polygon for tension parameters conformal to the above analysis.

Let us now complete the analysis for the first spline segment (again, the situation in the last spline segment can be derived accordingly). Due to (3.34) and (3.35), we have

$$\lim_{\sigma_1\to 0} \mathbf{u}_1 = 2c_1\sqrt{|\Delta\mathbf{p}_1|}\,\exp\left(i\frac{\alpha_1}{2}\right). \tag{3.42}$$

In view of (3.9), this immediately implies that $(\mathbf{q}_{1,j}-\mathbf{q}_{1,j-1})$, for $j = 1, 2$, tend to align with $\Delta\mathbf{p}_1$. More precisely, considering the first order expansion of the first equation in (3.37) and deriving from (3.34) a corresponding expansion for $\mathbf{u}_1$, we can also state that, when a suitable solution of the quadratic system is considered, in the limit the components of $(\mathbf{q}_{1,j}-\mathbf{q}_{1,j-1})$, $j = 1, 2$, orthogonal to $\Delta\mathbf{p}_1$ belong to the same half–plane to which the extension of $\Delta\mathbf{p}_2$ starting from $\mathbf{p}_1$ belongs to. We can then conclude that also the first and last control polygons have a suitable limit shape.

We can summarize the overall analysis in the following theorem.

Theorem 2 *If the tension parameters are sufficiently big, then the* k*–th segment of the* G^2 *PH quintic spline belongs to* T_k, $\forall k = 1, \dots, N$.

3.3.2 G^2 *Automatic selection of the tension parameters*

We are now interested in using the analysis described in Subsection 3.3.1 to reduce the choice of all the tension parameters $\nu_k = 1/\sigma_k, k = 0, \dots, N$, to the selection of just one shape positive parameter σ which can be initially set to 1 and later modified if needed or desired. Note that this strategy is designed just to bound the distance of the control points defining each spline segment from the corresponding displacement.

Let us express the limit configuration of the k–th control polygon in terms of a local reference system with the x axis aligned with $\Delta\mathbf{p}_k$ so that

$$\begin{aligned} \Delta\mathbf{p}_{k-1} &= |\Delta\mathbf{p}_{k-1}| \exp{(i\,\omega_L)}, \\ \Delta\mathbf{p}_{k} &= |\Delta\mathbf{p}_{k}| \exp{(i\,0)}, \\ \Delta\mathbf{p}_{k+1} &= |\Delta\mathbf{p}_{k+1}| \exp{(i\,\omega_R)}, \end{aligned}$$

with $\omega_L = [\theta_{k-1} - \theta_k]$, $\omega_R = [\theta_{k+1} - \theta_k]$, with $[\theta] = \arg{(\exp(i\,\theta))}$. By setting

$$\begin{aligned} b_k \exp{(i\phi_k^L)} &= \lim_{\sigma_{k-2},\sigma_{k-1},\sigma_k \to 0} \frac{\mathbf{u}_k}{\sigma_{k-1}}, \\ d_k \exp{(i\phi_k^R)} &= \lim_{\sigma_{k},\sigma_{k+1},\sigma_{k+2} \to 0} \frac{\mathbf{v}_k}{\sigma_{k}}, \end{aligned}$$

with ϕ_k^L and ϕ_k^R both belonging to $(-\frac{\pi}{2}, \frac{\pi}{2}]$, we get

$$\begin{aligned} \mathrm{dist}(\mathbf{q}_{k,1}, \Delta\mathbf{p}_k) &\approx \sigma_{k-1}^2 \frac{\ell_k}{5} b_k^2 \,|\sin(2\phi_k^L)|, \\ \mathrm{dist}(\mathbf{q}_{k,2}, \Delta\mathbf{p}_k) &\approx \sigma_{k-1} \frac{\ell_k}{5} b_k \,|\sigma_{k-1} b \sin(2\phi_k^L) + c_k \sin(\phi_k^L)|, \end{aligned}$$

for $k = 2, \dots, N$, where $\mathrm{dist}(\mathbf{q}, \Delta\mathbf{p}_k)$ denotes the distance of a point $\mathbf{q}$ from the straight line connecting $\mathbf{p}_{k-1}$ to $\mathbf{p}_k$. Analogously, we also get

$$\begin{aligned} \mathrm{dist}(\mathbf{q}_{k,3}, \Delta\mathbf{p}_k) &\approx \sigma_k \frac{\ell_k}{5} d_k \,|\sigma_k d_k \sin(2\phi_k^R) + c_k \sin(\phi_k^R)|, \\ \mathrm{dist}(\mathbf{q}_{k,4}, \Delta\mathbf{p}_k) &\approx \sigma_k^2 \frac{\ell_k}{5} d_k^2 \,|\sin(2\phi_k^R)|, \end{aligned}$$

for $k = 1, \dots, N-1$. In order to derive not too strict bounds, we consider only the first order approximation of these distances. Thus we require the following inequalities

$$\begin{aligned} \sigma_{k-1} b_k c_k \frac{\ell_k}{5} |\sin(\phi_k^L)| &\leqslant H_k, \\ \sigma_k c_k d_k \frac{\ell_k}{5} |\sin(\phi_k^R)| &\leqslant H_k. \end{aligned}$$

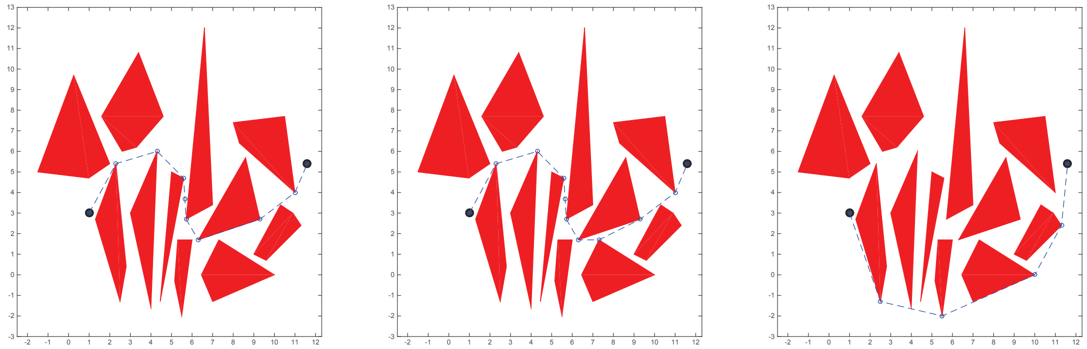


Figure 24. Three different piecewise linear paths for scenario S2 associated with the visibility graph: shortest path (VGSP, top left), modified shortest path (MVGSP, top right) and greedy path (GVGP, bottom).

When ϕ_k^L and ϕ_k^R do not vanish, this corresponds to

$$\begin{aligned} \sigma_{k-1} &\leqslant \frac{5H_k}{2\ell_k b_k c_k}\,\frac{1}{|\sin\phi_k^L|}\,, && k = 2,\dots,N\,, \\ \sigma_k &\leqslant \frac{5H_k}{2\ell_k c_k d_k}\,\frac{1}{|\sin\phi_k^R|}\,, && k = 1,\dots,N-1\,. \end{aligned} \tag{3.43}$$

Clearly, for each $\sigma_k, k = 1,\dots,N-1$, we have to take into account two inequalities coming from the two adjacent segments joining in $\mathbf{p}_k$. Then each σ_k is fixed equal to σ times the minimum of the two related upper bounds.

3.4 NUMERICAL RESULTS

3.4.1 G^1 *case: complex scenario*

In this subsection the results obtained with our approach for the complex scenario S2 shown in Figure 24 are discussed. For brevity, we report only the results obtained using the visibility graph. Note, however, that in this case they are always better than those based on the trapezoidal map decomposition. The comparison of the P_1 visibility graph shortest path (VGSP) with the modified versions computed with Algorithms 2 (MVGSP) and 3 (GVGP) is presented in Figure 24 and Table 4. Note that the GVGP algorithm has been applied with an “ad hoc” choice of the starting edge to emphasize the possibility of alternative admissible solutions. The number of points (# points) along the different P_1 paths as well as their lengths are quite similar for the three solutions. On the other hand, we may note that the two piecewise linear paths MVGSP and GVGP exhibit smaller maximum, and average angle variations ($\alpha_{\min}, \alpha_{\max}, \alpha_{\text{avg}}$) with respect to the shortest path (VGSP).

	VGSP	MVGSP	GVGP
# points	9	10	6
length	16.6391	16.7129	18.3036
α_{min}	18.9704	10.8403	22.7323
α_{max}	77.8943	61.9205	57.6353
α_{avg}	43.0702	37.6864	38.7647

Table 4. Comparison of the three piecewise linear paths for scenario S2 shown in Figure 24.

	VGSP	MVGSP	GVGP
T1	16.8280	16.8774	18.4677
T2	16.8180	16.8648	18.5471

Table 5. Total arc length of the PH spline paths shown in Figure 25 and 26 for scenario S2.

Figures 25 and 26 shows the corresponding P_2 paths and curvature plots obtained by considering T1 and T2 tangents, respectively, while Table 5 gives the associated total arc lengths. As expected, the shortest P_2 path is the one related to the VGSP. On the other hand, by introducing an additional interpolation point along the path and, consequently, slightly increasing the total arc length, the solution based on MVGSP is able to reduce the curvature pick in the last by one spline segment. The P_2 path related to the GVGP offers a smooth alternative solution which is remarkably different.

For comparison, Figures 27 and 28 show the result obtained by considering the different piecewise linear solutions together with cubic G^1 splines according to the choice of T1 and T2 unit tangents, respectively. The values of the tension parameters are chosen as in the corresponding PH quintic solution related to same piecewise linear path and shown in Figures 25 and 26. We note that also for this second scenario PH splines exhibit a smaller curvature range in all test cases.

3.4.2 G^2 *case: path planning and path finding*

In this section we present the results related to: two environments represented in the Path Planning (PP) form, and two pairs of target/goal positions in two environments represented in the Path Finding (PF) form, a house map and a geographic area. Note that for all these experiments we have always assumed the value of the shape parameter σ equal to 1. Starting from the same polyline produced in the first step of the method, we always compare the results of the

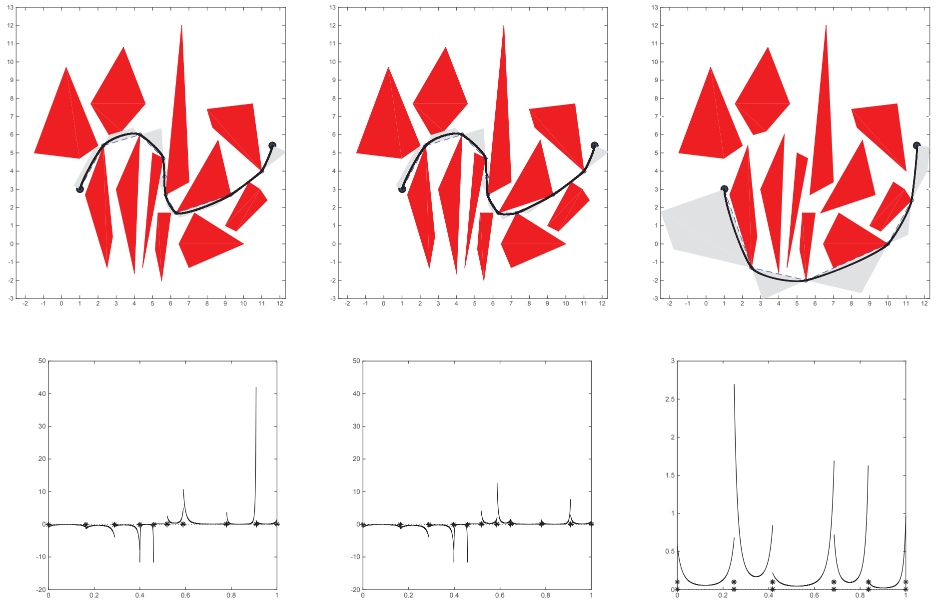


Figure 25. G^1 PH quintic spline paths (solid lines, top) for scenario S2 obtained starting from the piecewise linear paths (dashed lines, top): VGSP (left), MVGSP (center) and GVGP (right) by using T1 tangents are shown with the corresponding admissible regions (grey). The curvature plots of the three spline paths are also shown (bottom).

$C^1 \cap G^2$ PH approach with those of the G^1 PH scheme considered in [40], which produces curvature discontinuous paths. For a better smoothness comparison, the curvature plot of any path obtained in the different examples is also shown. Finally, two comparisons with different spline approaches are reported in Figures 31 and 51.

Path planning

Considering the examples PP1 and PP2 introduced in Section 2.3.4, Figures 29 and 30 show that in both the experiments the $C^1 \cap G^2$ approach combined with the proposed automatic choice of the tension parameters is capable to produce admissible paths which are significantly smoother than their G^1 counterpart. The increase in the path length is 0.9 percent for PP1 and 0.3 for PP2.

In order to compare the PH spline approach with more standard G^2 spline solutions, Figure 31 shows a comparison of the results obtained by our approach and the cubic ν–spline method introduced in [68] which is an alternative interpolation scheme with tension. In both cases reduced degree end spans and zero tension parameters are used, since in this case tension is not necessary for obstacle avoidance. The figure clearly shows that the PH spline solution is better

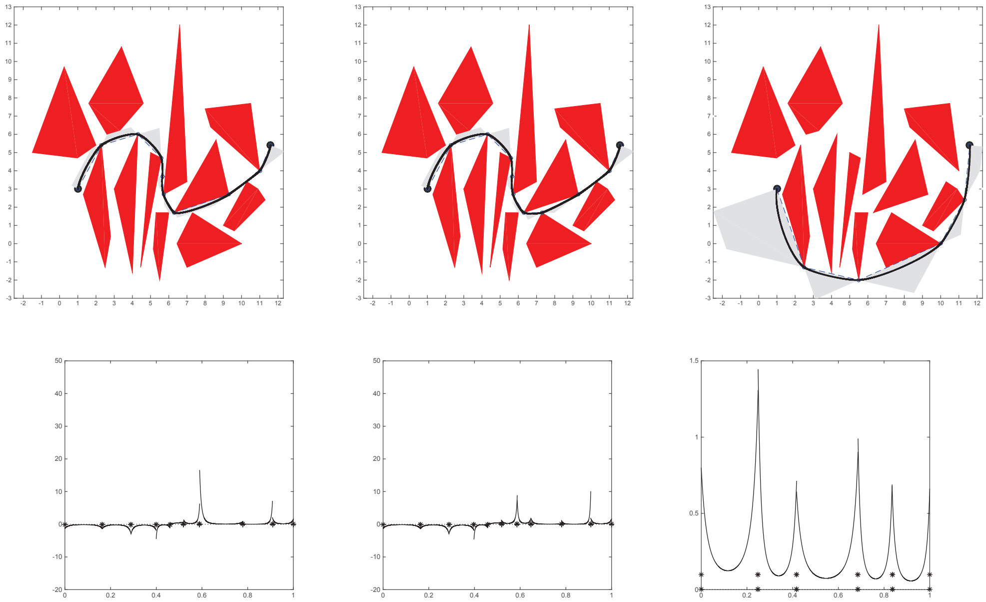


Figure 26. G^1 PH quintic spline paths (solid lines, top) for scenario S2 obtained starting from the piecewise linear paths (dashed lines, top): VGSP (left), MVGSP (center) and GVGP (right) by using T2 tangents are shown with the corresponding admissible regions (grey). The curvature plots of the three spline paths are also shown (bottom).

from the curvature point of view. This behaviour has been observed also for other configurations, even if the difference can be less prominent. Note that the fairness of PH representatons was already stated in the literature [22].

Path finding

Even in this context, two different scenarios are considered. They are obtained by applying the two–stage image segmentation method (described in Appendix A.1) to the images shown in Figures 32 and 33.[7][8]

The following results are obtained considering as scale of magnitude the dimension (in pixels) of the original segmented image. The resolution of the two images shown on the top of Figures 32 and 33 are 730×600 and 700×540 pixels, respectively.

---

[7] The reported image of Manhattan is available at `http://freegeographytools.com`

[8] The parameters λ, μ, ρ necessary as input to the considered segmentation method are set to $3, 1, 2$, respectively. Note however that the results are not highly sensitive to this choice. In both cases, we use $p = 2$ to define two different clusters of pixels: one representing the free cells (free configuration space) and the other associated to the obstructed cells (obstacles).

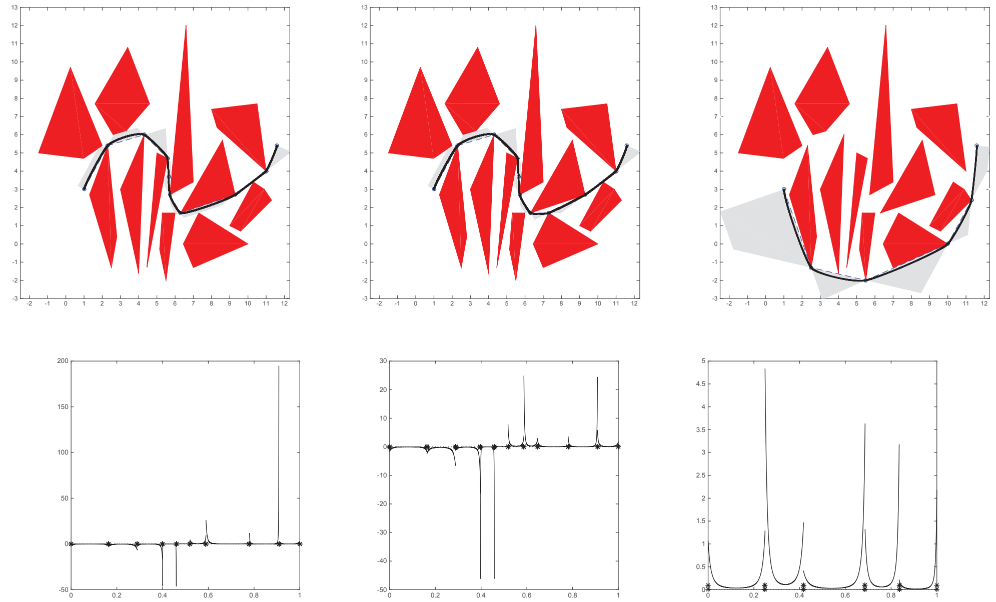


Figure 27. G^1 cubic spline paths (solid lines, top) for scenario S2 obtained starting from the piecewise linear paths (dashed lines, top) — VGSP (left), MVGSP (center) and GVGP (right) — by using T1 tangents are shown with the corresponding admissible regions (grey). The curvature plots of the three spline paths are also shown (bottom).

Figures 34 and 35 show the piecewise linear paths obtained by applying Algorithm 4 to the initial paths computed by A* algorithm for four configurations, two related to the House (PF1-PF2) and the others (PF3-PF4) to the Manhattan scenario. Note that the polyline in Example PF4 has three close successive vertices that cannot be removed in view of the small obstacle close to the last edge on the right.

Figures 36 and 37 show that even in this case the new scheme is capable to produce remarkably smoother final paths with a modest increase in the path length (0.7 percent for PF1 and 0.7 percent for PF2). For the Manhattan environment still Figures 38 and 39 confirm a relevant improvement in the path smoothness but at the price of a more significant increase of the path length (0.7 percent for PF3 and 2.1 for PF4). Intermediate solutions can be obtained by selecting for the shape parameter σ values less than 1.

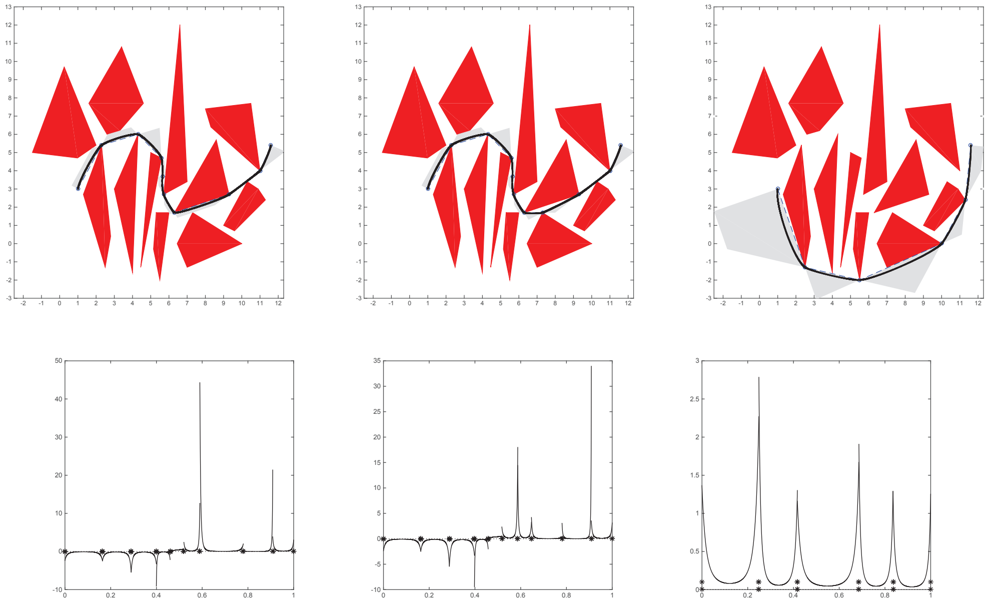


Figure 28. G^1 cubic spline paths (solid lines, top) for scenario S2 obtained starting from the piecewise linear paths (dashed lines, top) — VGSP (left), MVGSP (center) and GVGP (right) — by using T2 tangents are shown with the corresponding

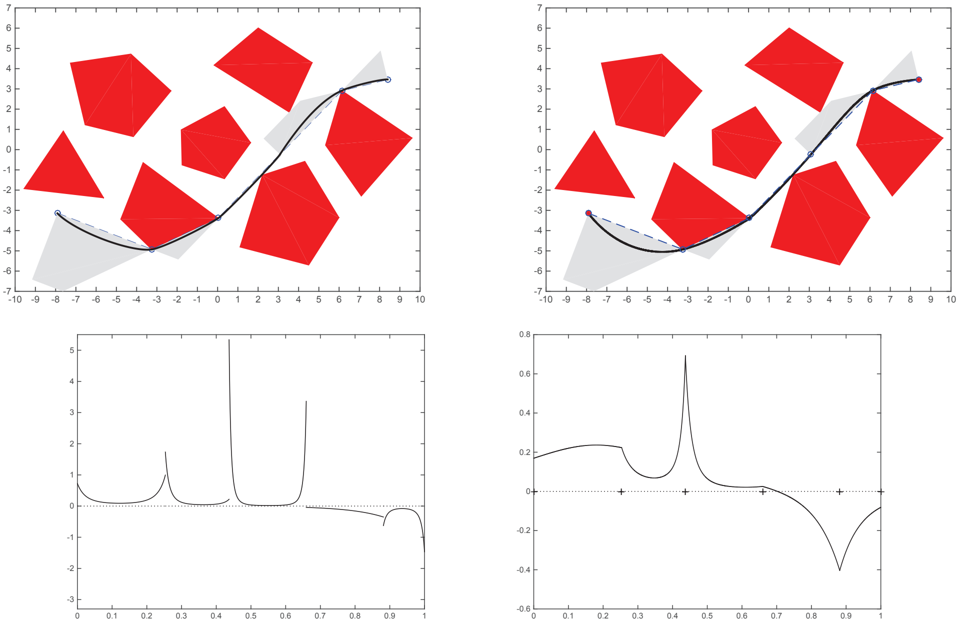


Figure 29. Example PP1. G^1 PH quintic spline (top left) and $C^1 \cap G^2$ PH quintic spline (top right). The curvature plots of the two PH spline interpolants are also shown (bottom).

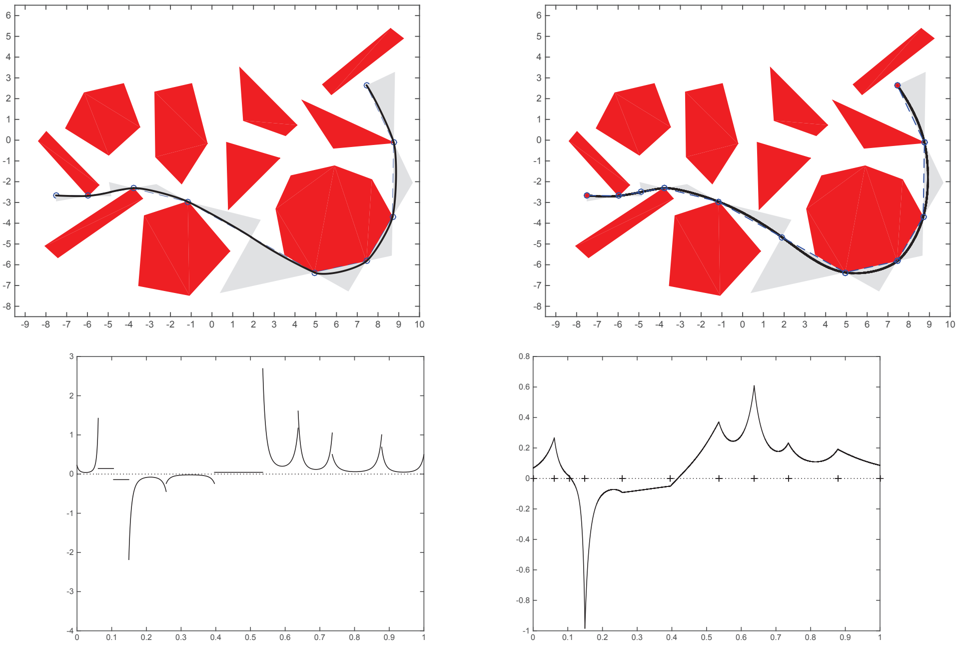


Figure 30. Example PP2. G^1 PH quintic spline (top left) and $C^1 \cap G^2$ PH quintic spline (top right). The curvature plots of the two PH spline interpolants are also shown (bottom).

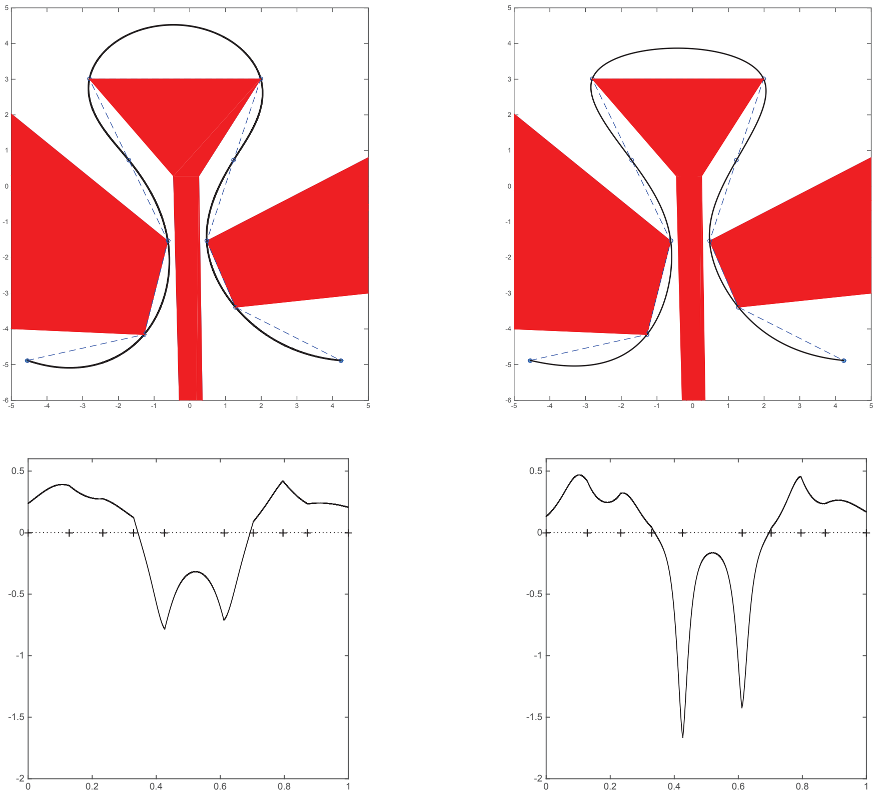


Figure 31. Comparison between two admissible curvature continuous paths: PH spline (top left) and cubic ν–spline (top right) schemes. The curvature plots of the two paths are also shown (bottom).

Figure 32. A house map (top) and the segmentation result (bottom).

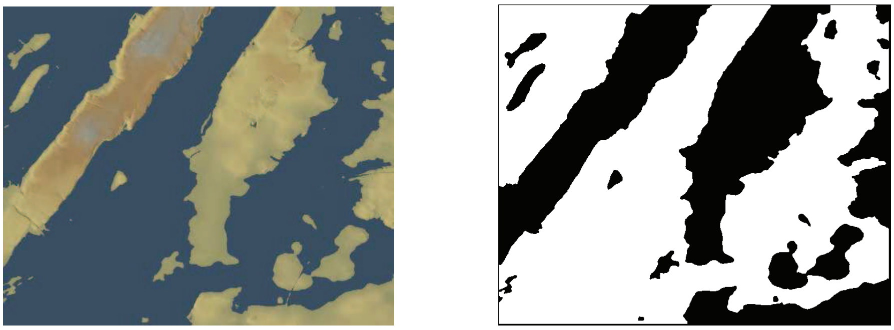
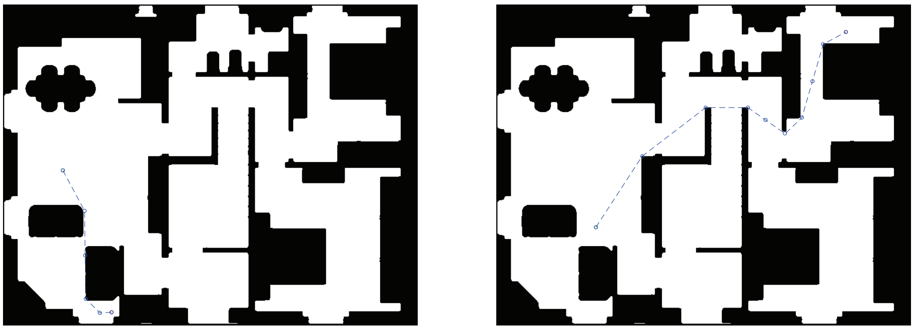


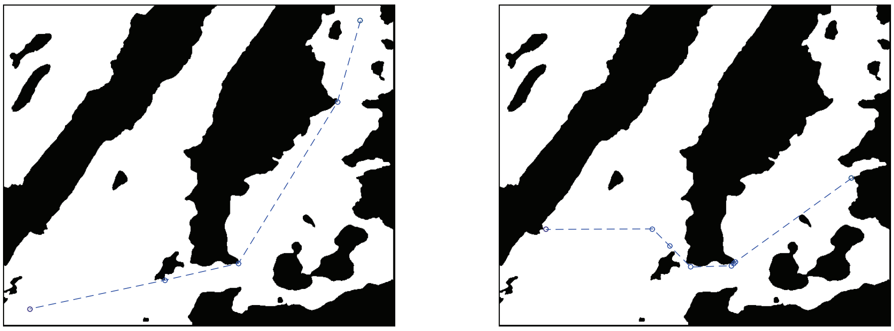
Figure 33. A terrain image of southern Manhattan Island as it would look like with eight meters of flooding (top) and the segmentation result (bottom).



(a) Example PF1: # of vertices = 6

(b) Example PF2: # of vertices = 9

Figure 34. The polylines computed with ALGORITHM 4 in the house environment shown in Figure 32 for two different target/goal positions.



(a) Example PF3: # of vertices = 5

(b) Example PF4: # of vertices = 8

Figure 35. The polylines computed with ALGORITHM 4 in the Manhattan environment shown in Figure 32 for two different target/goal positions.

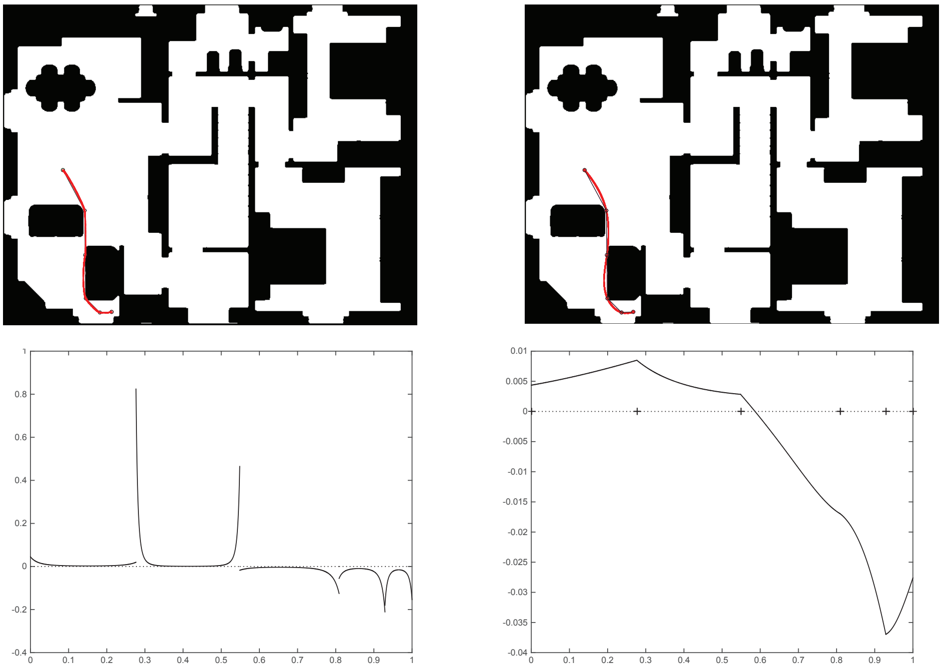


Figure 36. Example PF1: G^1 PH quintic spline with automatic strategy (left), G^2 PH quintic spline with automatic strategy (right). The signed curvature plots of the two PH spline interpolants are also shown (below).

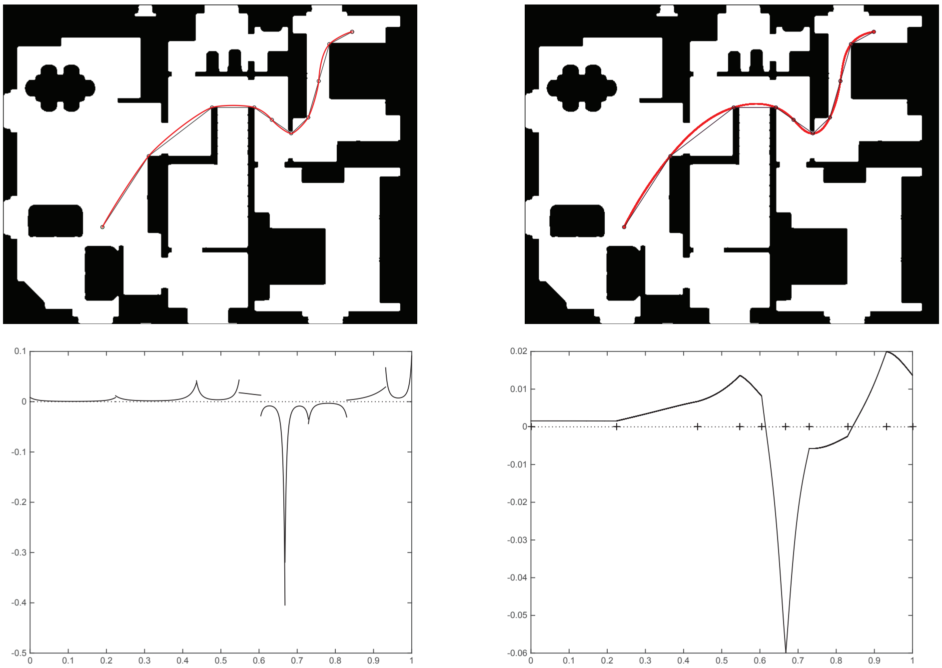


Figure 37. Example PF2: G^1 PH quintic spline with automatic strategy (top left), $C^1 \cap G^2$ PH quintic spline with automatic strategy (top right). The curvature plots of the two PH spline interpolants are also shown (bottom).

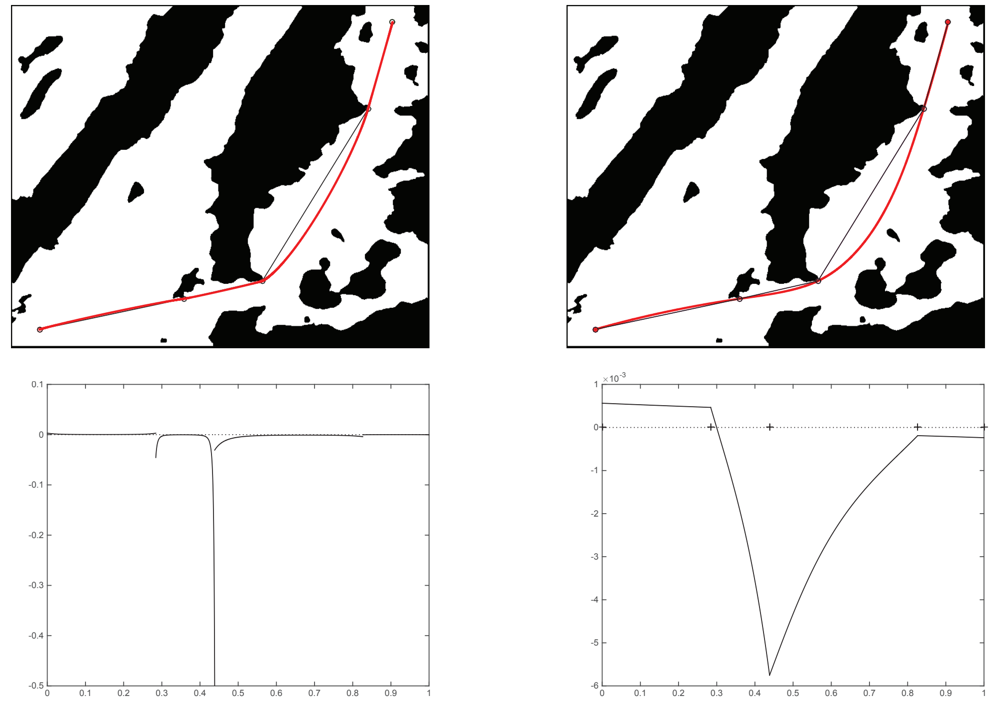


Figure 38. Example PF3. G^1 PH quintic spline with automatic strategy (top left) and $C^1 \cap G^2$ PH quintic spline with automatic strategy (top right). The curvature plots of the two PH spline interpolants are also shown (bottom).

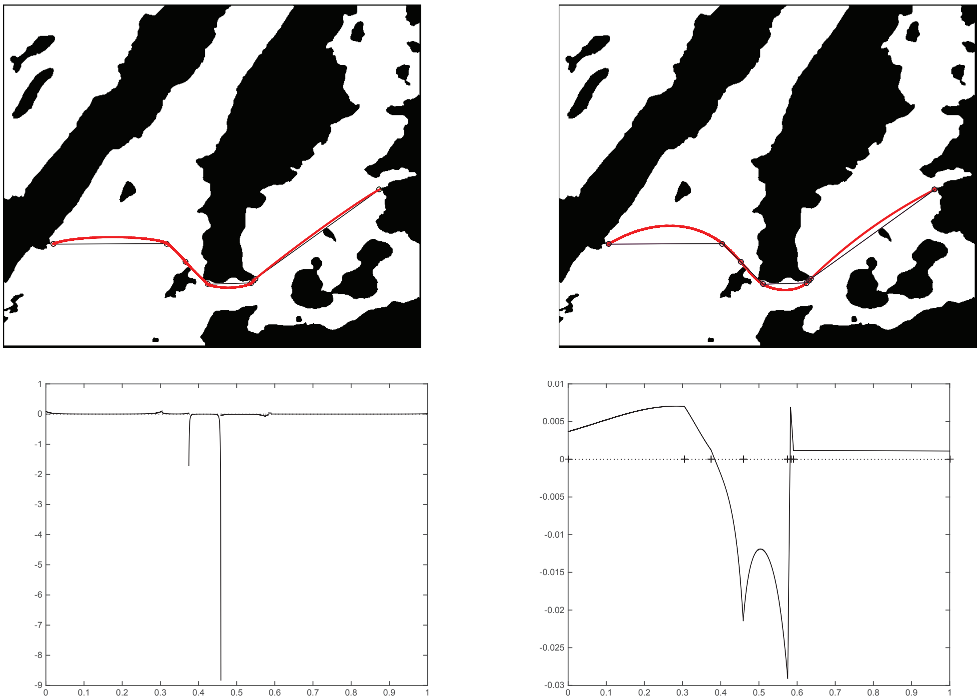


Figure 39. Example PF4: G^1 PH quintic spline with automatic strategy (left), G^2 PH quintic spline with automatic strategy (right). The signed curvature plots of the two PH spline interpolants are also shown (below).

4

PATH PLANNING WITH PYTHAGOREAN–HODOGRAPH CURVES FOR UAV

4.1 PATH PLANNING AND UAV

Many authors have recently proposed the use of *Pythagorean–hodograph* curves in the context of path planning for autonomous or remotely–operated aerial, land, or submarine vehicles, such as unmanned aerial vehicles (UAVs) or autonomous underwater vehicles (AUVs): see [5, 10, 13, 15, 16, 56, 62, 63, 64, 65, 66, 67, 74, 72, 71, 76]. Feasible paths must satisfy various constraints, such as bounds on the path curvature or climb angle, avoidance of environmental obstacles, and maintenance of safe separations in vehicle swarms.

PH curves were introduced in [36] to facilitate *exact* computations of basic geometrical properties that require numerical approximations in the context of general polynomial curves (see [22] for a comprehensive treatment). These properties have been extensively and systematically exploited in the context of computer numerical control (CNC) manufacturing systems [19, 29, 30, 37].

This chapter introduces some novel PH curve utilities that are of particular interest in the path planning problem for multiple autonomous vehicles. Some representative applications of PH curves are then addressed: the maintenance of safe separations within swarms of autonomous vehicles in Section 4.3; the construction of families of curved paths with identical arc lengths, ensuring the simultaneous arrival of multiple autonomous vehicles in Section 4.4; finally the imposition of path curvatures consistent with vehicle steering rates in Section 4.5.

4.2 PLANAR PYTHAGOREAN–HODOGRAPH QUINTICS CURVES

The simplest planar PH curves that can inflect, and have shape freedoms similar to those of "ordinary" cubics, are the quintics. A planar PH quintic is defined by choosing a quadratic polynomial $\mathbf{w}(\xi)$, with Bernstein coefficients $\mathbf{w}_0, \mathbf{w}_1, \mathbf{w}_2$. On integrating (3.6), the Bézier control points $\mathbf{p}_k = x_k + i\, y_k$ of the resulting PH quintic

$$\mathbf{r}(\xi) = \sum_{k=0}^{5} \mathbf{p}_k \binom{5}{k} (1-\xi)^{5-k} \xi^k$$

may be expressed [20] as

$$\begin{aligned}
\mathbf{p}_1 &= \mathbf{p}_0 + \frac{1}{5}\,\mathbf{w}_0^2\,, \\
\mathbf{p}_2 &= \mathbf{p}_1 + \frac{1}{5}\,\mathbf{w}_0\mathbf{w}_1\,, \\
\mathbf{p}_3 &= \mathbf{p}_2 + \frac{1}{5}\,\frac{2\mathbf{w}_1^2 + \mathbf{w}_0\mathbf{w}_2}{3}\,, \\
\mathbf{p}_4 &= \mathbf{p}_3 + \frac{1}{5}\,\mathbf{w}_1\mathbf{w}_2\,, \\
\mathbf{p}_5 &= \mathbf{p}_4 + \frac{1}{5}\,\mathbf{w}_2^2\,,
\end{aligned} \tag{4.1}$$

where $\mathbf{p}_0$ is a free integration constant. The parametric speed polynomial

$$\sigma(\xi) = \sum_{k=0}^{4} \sigma_k \binom{4}{k} (1-\xi)^{4-k}\xi^k \tag{4.2}$$

has the Bernstein coefficients

$$\begin{aligned}
\sigma_0 &= |\mathbf{w}_0|^2\,, \quad \sigma_1 = \mathrm{Re}(\overline{\mathbf{w}}_0\mathbf{w}_1)\,, \\
\sigma_2 &= \frac{2|\mathbf{w}_1|^2 + \mathrm{Re}(\overline{\mathbf{w}}_0\mathbf{w}_2)}{3}\,, \\
\sigma_3 &= \mathrm{Re}(\overline{\mathbf{w}}_1\mathbf{w}_2)\,, \quad \sigma_4 = |\mathbf{w}_2|^2\,.
\end{aligned}$$

Correspondingly, the Bernstein form

$$s(\xi) = \sum_{k=0}^{5} s_k \binom{5}{k} (1-\xi)^{5-k}\xi^k \tag{4.3}$$

of the arc length polynomial is specified by the coefficients

$$s_0 = 0 \qquad \text{and} \qquad s_k = \frac{1}{5}\sum_{j=0}^{k-1} \sigma_j\,, \quad k = 1,\ldots,5 \tag{4.4}$$

and the total arc length is

$$S = s(1) = s_5 = \frac{\sigma_0 + \sigma_1 + \sigma_2 + \sigma_3 + \sigma_4}{5}\,. \tag{4.5}$$

Based on the complex representation, algorithms for the construction and analysis of planar PH curves have reached a mature state of development — for example, see [24, 28, 34, 23].

4.3 MAINTENANCE OF MINIMUM SEPARATION

Consider the problem of determining whether the traversal of two planar PH quintic paths $\mathbf{r}(\mu)$, $\mu \in [0,1]$ and $\mathbf{s}(\nu)$, $\nu \in [0,1]$ at a constant speed V will incur a close approach with separation less than a safe minimum distance δ. If $p(\mu)$ and $q(\nu)$ are the polynomial arc length functions for $\mathbf{r}(\mu)$ and $\mathbf{s}(\nu)$, and we assume the motions start at $\mathbf{r}(0)$ and $\mathbf{s}(0)$ at time $t = 0$, this amounts to identifying if parameter values $(\mu, \nu) \in [0,1]^2$ exist, such that

$$p(\mu) = q(\nu) \qquad \text{and} \qquad |\,\mathbf{r}(\mu) - \mathbf{s}(\nu)\,| < \delta\,.$$

Geometrically, this corresponds to asking whether the value of the bivariate function $d(\mu,\nu) = |\,\mathbf{r}(\mu) - \mathbf{s}(\nu)\,|$, restricted to the degree 5 algebraic curve defined by the equation $f(\mu,\nu) = p(\mu) - q(\nu) = 0$, dips below δ.

In general, this problem has no closed–form solution, but the polynomial nature of $p(\mu)$ and $q(\nu)$ allows an essentially exact test for a maintenance of the safe closest–approach threshold δ for any specified sampling distance Δs (corresponding to a sampling time $\Delta t = \Delta s/V$) along the paths. Namely, if μ_k, ν_k $(\leqslant 1)$ are the unique real roots of the equations

$$p(\mu_k) \;=\; q(\nu_k) \;=\; k\Delta s\,, \quad k = 0, 1, 2, \ldots$$

the distance at time $t_k = k\Delta t$ is $\delta_k = |\,\mathbf{r}(\mu_k) - \mathbf{s}(\nu_k)\,|$. For modest Δs values μ_k, ν_k are good estimates of μ_{k+1}, ν_{k+1} so the latter can be accurately and efficiently computed from the former by a few Netwon–Raphson iterations.

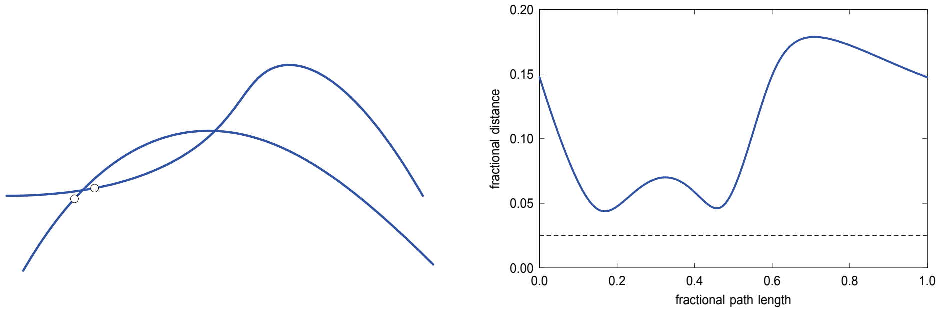


Figure 40. Left: two planar PH quintic paths of equal arc length traversed at constant speed, with the points of closest approach indicated as dots. Right: fractional distance along the paths as a function of fractional path length.

Figure 40 illustrates an application of this methodology[1]. Two planar PH quintic paths, of equal arc length S, are to be traversed at the same constant speed and a minimum safe separation of $0.025\,S$ is desired. Although these

---

[1] The fractional distance between two given curves $\mathbf{r}(\mu)$ and $\mathbf{s}(\nu)$ with same length L at time t_k is given by $\frac{\delta_k}{L}$. The fractional path length is the normalized arc length.

paths intersect twice, the computation with sampling distance $\Delta s = 0.005\,S$ indicates that the minimum separation has nevertheless been achieved. The method can be easily extended to spatial PH curves, and can also be adapted to accommodate variable speeds along the paths [30, 37].

The use of *offset curves* to define "safety zones" of a specified width about the paths of multiple autonomous vehicles has also been proposed [73, 74, 72]. This is a conservative approach, that considers only the path geometry (not path timing) in maintaining safe vehicle separations: see Figure 41. PH curves have the advantage, in this context, that their offsets $\mathbf{r}_d(\xi) = \mathbf{r}(\xi) \pm d\,\mathbf{n}(\xi)$, where $\mathbf{n}(\xi)$ is the unit normal to $\mathbf{r}(\xi)$ and d is the offset distance, are *rational curves* [36]: they admit parameterizations $\mathbf{r}_d(\xi) = (X(\xi)/W(\xi), Y(\xi)/W(\xi))$ where $W(\xi), X(\xi), Y(\xi)$ are relatively prime polynomials specifying the offset curve homogeneous coordinates. However, it should be noted that the offset curves illustrated in [73, 74, 72] are clearly incorrect.

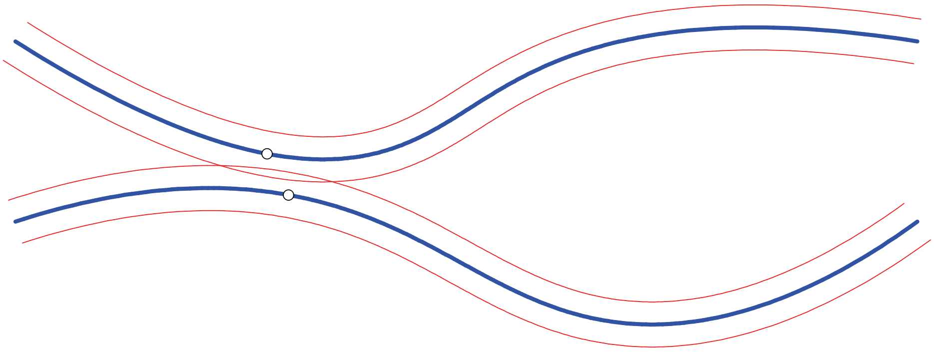


Figure 41. Two paths (blue) of identical arc length S with their offsets (red) at distance $\pm 0.025\,S$. Although the intersection of the offset curves indicates a potential violation of the prescribed minimum safe separation $\delta = 0.05\,S$, the actual points of closest approach (indicated by dots) are at distance $0.0515\,S$.

4.4 SIMULTANEOUS ARRIVALS

Several authors have discussed the problem of ensuring simultaneous arrival of autonomous vehicles that follow different paths to a specified destination [13, 74, 72]. If the vehicles maintain the same constant speed, this amounts to designing smooth paths with different shapes but identical arc lengths. It has recently been demonstrated [24] that, for the case of planar PH quintic paths with different initial/final conditions, the requirement of equal arc lengths can be *exactly* achieved using a simple algorithm. The construction of planar PH quintics with prescribed initial/final points $\mathbf{q}_0, \mathbf{q}_1$, tangents $\mathbf{t}_0, \mathbf{t}_1$, and arc length S may be summarized as follows (see [24] for further details).

Using the complex representation with $\mathbf{q}_0 = x_0 + i\,y_0$, $\mathbf{q}_1 = x_1 + i\,y_1$ and $\mathbf{t}_0 = \exp(i\,\theta_0)$, $\mathbf{t}_1 = \exp(i\,\theta_1)$ and setting $\Delta\mathbf{q} = \mathbf{q}_1 - \mathbf{q}_0 := |\Delta\mathbf{q}|\exp(i\,\alpha)$, the specified data is first reduced to canonical form by: (i) subtracting $\mathbf{q}_0$ from $\mathbf{q}_0$ and $\mathbf{q}_1$; (ii) dividing $\mathbf{q}_1$ by $\Delta\mathbf{q}$; (iii) multiplying $\mathbf{t}_0$ and $\mathbf{t}_1$ by $\exp(-\,i\,\alpha)$; and (iv) dividing S by $|\Delta\mathbf{q}|$. This amounts to a translation/rotation/scaling transformation, that maps $\mathbf{q}_0$ and $\mathbf{q}_1$ to the points 0 and 1 on the real axis. Once the canonical form solution has been computed, it can be restored to the original coordinates by taking $\mathbf{p}_0 = \mathbf{q}_0$ in (4.1) and multiplying $\mathbf{w}_0, \mathbf{w}_1, \mathbf{w}_2$ by $\sqrt{|\Delta\mathbf{q}|}\exp(i\,\frac{1}{2}\alpha)$ before substituting in (4.1). For brevity, we consider only the generic case $\theta_1 \neq \pm\,\theta_0$ (see [24] for the special cases $\theta_1 = \pm\,\theta_0$). For canonical form data, the coefficients of (3.5) are expressed as

$$\mathbf{w}_0 = w\exp(i\tfrac{1}{2}\theta_0)\,, \quad \mathbf{w}_1 = u + i\,v\,, \quad \mathbf{w}_2 = w\exp(i\tfrac{1}{2}\theta_1)\,,$$

where the real unknowns u, v, w are determined as follows. First, setting

$$\begin{aligned}
a_2 &= 2(c_0 s_1 - c_1 s_0)^2\,,\\
a_1 &= 3\,[\,2(c_0 c_1 + s_0 s_1 - 3)S + 3(c_0^2 - s_0^2 + c_1^2 - s_1^2) - 2(c_0 c_1 - s_0 s_1)\,]\,,\\
a_0 &= 36(S^2 - 1)\,,
\end{aligned}$$

where $(c_0, s_0) := (\cos\frac{1}{2}\theta_0, \sin\frac{1}{2}\theta_0)$ and $(c_1, s_1) := (\cos\frac{1}{2}\theta_1, \sin\frac{1}{2}\theta_1)$, let

$$z = \frac{-\,a_1 - \sqrt{a_1^2 - 4a_2 a_0}}{2a_2} \tag{4.6}$$

be the smaller root[2] of the equation $a_2 z^2 + a_1 z + a_0 = 0$, and set $w = \sqrt{z}$. Then, let $\mu, \nu = \pm 1$ be such that $\mu\nu$ and $(c_0 s_1 + c_1 s_0 - 3c_0 s_0 - 3c_1 s_1)z$ have the same sign. For each of the two μ, ν pairs thus identified, set

$$u = \frac{-3(c_0 + c_1)w + \mu\sqrt{p(z)}}{4} \quad \text{and} \quad v = \frac{-3(s_0 + s_1)w + \nu\sqrt{q(z)}}{4}\,, \tag{4.7}$$

where

$$p(z) = 60(S + 1) - (15c_0^2 + 15c_1^2 - 10c_0 c_1)z\,, \tag{4.8}$$

$$q(z) = 60(S - 1) - (15s_0^2 + 15s_1^2 - 10s_0 s_1)z\,. \tag{4.9}$$

In general, this yields two formal solutions to the problem of matching the given data, of which one has an attractive shape and the other exhibits an undesirable looping behavior — the latter can be identified and discarded by comparing the absolute rotation index[3] for the two solutions. Figure 42 shows a family

---

[2] It has been shown [24] that this equation always has two positive real roots, and the expressions (4.8) and (4.9) are both non–negative at the particular root (4.6).

[3] The rotation index R of a C^2 planar curve $\mathbf{r}(t), t \in [0,1]$ with curvature $\kappa(t)$ is defined by $R = \frac{1}{2\pi}\int_0^1 \kappa(t)|\mathbf{r}'(t)|dt$ and is a classical means of characterizing global shape properties of curves (see e.g., [22]).

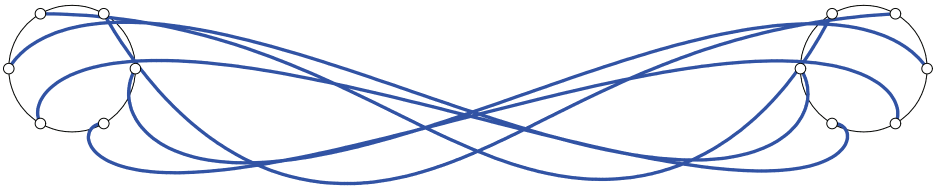


Figure 42. A family of simultaneous–arrival paths for a swarm of six unmanned constant speed vehicles, departing and arriving in different directions from a set of corresponding equidistant points on an initial and final target circle.

of simultaneous–arrival paths for a swarm of six unmanned vehicles that start from equidistant points on an initial circle, and end at corresponding points on a target circle. The path length S is chosen to be 12.5% greater than the linear distance between the start and end points, and each vehicle has different departure/arrival directions. It is evident from Figure 43, which plots the 15 pair–wise separations (computed by the method described in Section 4.3), that a minimum safe separation equal to 1% of S is achieved despite the criss–cross nature of the different paths. Although the main focus of this paper is on

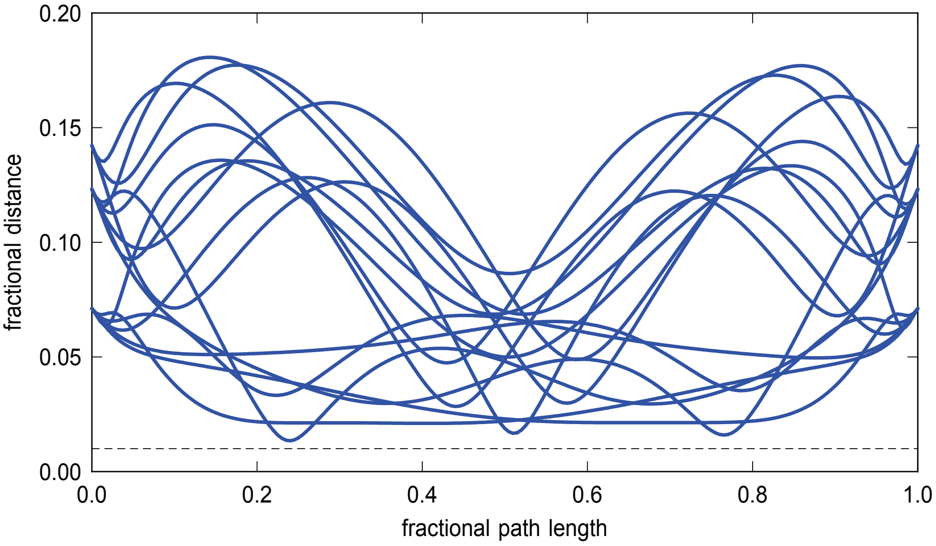


Figure 43. The 15 pair–wise separation plots for the six paths in Figure 42.

planar PH paths, one possible means of constructing spatial PH paths of equal arc length deserves mention. The interpolation of initial and final points and derivatives $\mathbf{p}_i, \mathbf{d}_i$ and $\mathbf{p}_f, \mathbf{d}_f$ by spatial PH quintics yields coefficients $\mathcal{A}_0(\phi_0)$ and $\mathcal{A}_2(\phi_2)$ for the quadratic quaternion polynomial $\mathcal{A}(\xi)$ dependent on $\mathbf{d}_i$ and $\mathbf{d}_f$ and angular parameters ϕ_0 and ϕ_2, while $\mathcal{A}_1$ is determined from the displacement $\mathbf{p}_f - \mathbf{p}_i$. Setting $\alpha = \frac{1}{2}(\phi_0 + \phi_2)$ and $\beta = \phi_2 - \phi_0$, it can be shown

[26] that the arc length of the resulting spatial PH curves depends only on β. Thus, varying α with β fixed yields spatial PH paths of different shapes but identical arc lengths.

4.5 CURVATURE EXTREMA OF PH QUINTICS

An unmanned vehicle can exert only a finite steering torque, which imposes an upper bound on the magnitude of feasible path curvatures [2, 10, 65, 73, 72]. Thus, the ability to identify the curvature extrema of PH curves is required, in order to ensure that prescribed curvature bounds are observed. From (3.7), the curvature of a planar PH quintic has the form $\kappa(\xi) = h(\xi)/\sigma^2(\xi)$, where the quadratic polynomial $h(\xi)$ has the Bernstein coefficients

$$h_0 = 4\,\mathrm{Im}(\overline{\mathbf{w}}_0\mathbf{w}_1)\,, \quad h_1 = 2\,\mathrm{Im}(\overline{\mathbf{w}}_0\mathbf{w}_2)\,, \quad h_2 = 4\,\mathrm{Im}(\overline{\mathbf{w}}_1\mathbf{w}_2)\,. \tag{4.10}$$

Extrema of the curvature occur at the roots of its derivative,

$$\kappa'(\xi) = \frac{f(\xi)}{\sigma^3(\xi)}\,, \quad f(\xi) := \sigma(\xi)h'(\xi) - 2\,h(\xi)\sigma'(\xi)\,.$$

The quintic polynomial $q(\xi)$ has the Bernstein coefficients

$$\begin{aligned}
f_0 &= 2\sigma_0h_1 + 6\sigma_0h_0 - 8\sigma_1h_0\,,\\
f_1 &= (2\sigma_0h_2 + 14\sigma_0h_1 - 8\sigma_1h_1 + 16\sigma_1h_0 - 24\sigma_2h_0)/5\,,\\
f_2 &= (4\,\sigma_0h_2 + 20\sigma_1h_1 - 18\sigma_2h_1 + 6\sigma_2h_0 - 12\sigma_3h_0)/5\,,\\
f_3 &= (12\,\sigma_1h_2 - 6\sigma_2h_2 + 18\sigma_2h_1 - 20\sigma_3h_1 - 4\sigma_4h_0)/5\,,\\
f_4 &= (24\sigma_2h_2 - 16\sigma_3h_2 + 8\sigma_3h_1 - 14\sigma_4h_1 - 2\sigma_4h_0)/5\,,\\
f_5 &= 8\sigma_3h_2 - 6\sigma_4h_2 - 2\sigma_4h_1\,,
\end{aligned}$$

and its real roots on the interval $\xi \in [\,0,1\,]$ may be efficiently computed using the subdivision and variation–diminishing properties of the Bernstein form.

Prior studies [74] have suggested modifying the PH curve end derivative magnitudes as a means of subduing the extremum path curvature magnitude below a prescribed bound. However, no information is provided on how the extremal curvature is identified, and whether the prescribed curvature bound can always be satisfied by modifying the derivative magnitudes. Moreover, the plots in Figure 3 of [74] appear to be incorrect, since they indicate that curvature is non–differentiable at certain points along the paths.

Two methods for minimizing the maximum curvature magnitude κ_{max} of paths with prescribed end points and tangent directions are considered here. In the first, the magnitudes $\ell_0 = |\mathbf{r}'(0)|$ and $\ell_1 = |\mathbf{r}'(1)|$ of the end derivatives are varied to obtain the (ℓ_0, ℓ_1) combination that gives the smallest κ_{max}. The second employs the procedure described in Section 4.4 in constructing paths of increasing arc length S, to identify the value that yields the smallest κ_{max}. Both methods use exact κ_{max} values, determined by comparing the values of $\kappa(\xi)$ at the roots $\xi \in (0, 1)$ of the polynomial $f(\xi)$, and the interval end points $\xi = 0$ and 1. Both also impose an upper bound S_{max} on the allowed path arc length, since S should be reasonably commensurate with the linear distance between the path end points, and in some cases it may be possible to make κ_{max} very small by allowing S to become unreasonably large [7]. Figure 44 shows interpolants to the end points $\mathbf{q}_0 = (0, 0)$ and $\mathbf{q}_1 = (1, 0)$ and end tangent directions $\theta_0 = \pi/3$, $\theta_1 = -3\pi/4$ for prescribed arc lengths $S = 1.1, \ldots, 1.6$, together with their curvature plots. For smaller values of S the extremal curvature occurs at an end point, but for larger values it is at an interior point. As seen in Figure 45, no

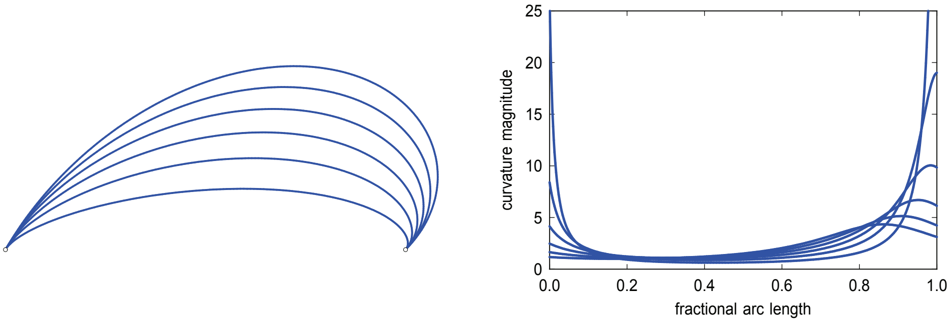


Figure 44. Left: A family of planar PH quintic interpolants to the end points $\mathbf{q}_0 = (0, 0)$ and $\mathbf{q}_1 = (1, 0)$ with tangent angles $\theta_0 = \pi/3$ and $\theta_1 = -3\pi/4$, for arc lengths $S = 1.1, \ldots, 1.6$. Right: curvature profiles for these interpolants.

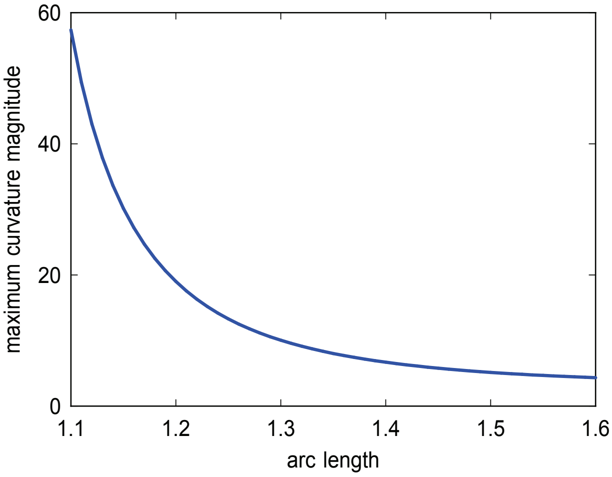


Figure 45. Maximum curvature magnitude versus arc length for paths with end points $\mathbf{q}_0 = (0, 0)$, $\mathbf{q}_1 = (1, 0)$ and tangent angles $\theta_0 = \pi/3$, $\theta_1 = -3\pi/4$.

minimum of κ_{max} exists for $S \leqslant 1.6$, and increasing S beyond ~ 1.3 yields only marginal reduction of κ_{max}.

As an alternative to varying S, Figure 46 illustrates the effect of changing the end derivative magnitudes (for brevity, assumed equal) for interpolants to the end points $\mathbf{q}_0 = (0,0)$ and $\mathbf{q}_1 = (1,0)$ and tangent directions $\theta_0 = \pi/6$ and $\theta_1 = \pi/4$, using $\ell_0 = \ell_1 = 1.0, 1.5, \ldots, 4.0$. In this case, it is seen that the value of $\ell_0 = \ell_1$ has a strong influence on the nature of the curvature variation, and Figure 47 shows that the extremum curvature magnitude κ_{max} has a distinct minimum with respect to the magnitude of the end derivatives.

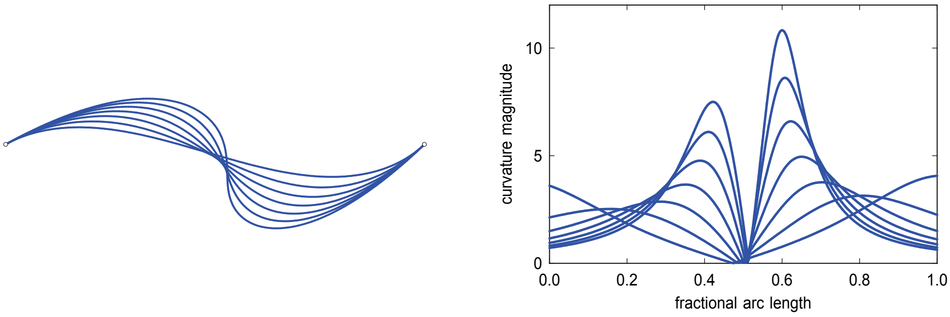


Figure 46. Left: Planar PH quintic interpolants to end points $\mathbf{q}_0 = (0,0)$ and $\mathbf{q}_1 = (1,0)$ with tangent angles $\theta_0 = \pi/6$, $\theta_1 = \pi/4$ and derivative magnitudes $\ell_0 = \ell_1 = 1.0, 1.5, \ldots, 4.0$. Right: the curvature plots for these interpolants.

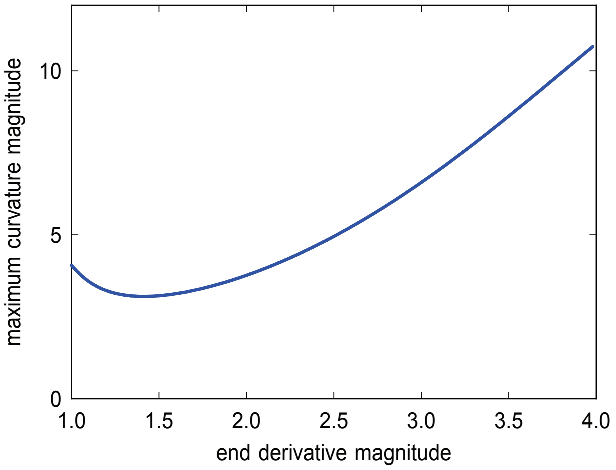


Figure 47. Largest curvature magnitude versus end derivative magnitude for paths from $\mathbf{q}_0 = (0,0)$ to $\mathbf{q}_1 = (1,0)$ and tangent angles $\theta_0 = \pi/6$, $\theta_1 = \pi/4$.

4.6 G^2 PATH CORNER ROUNDING AMONG OBSTACLES

Another topic that has attracted considerable interest [56, 65, 73, 71] is path planning for autonomous vehicles that must navigate collision–free paths of bounded curvature through a field of obstacles. We consider here the classical problem of constructing a planar path of minimum length through a field of polygonal obstacles (the vehicle is assumed to be of negligible size compared to the dimensions of the obstacles). Since the theoretical minimum–length path is piecewise–linear, it is incompatible with path curvature constraints imposed by vehicle steering mechanisms. It is shown that planar PH quintic segments can be employed to "round" the sharp corners of optimal piecewise–linear paths in a curvature–continuous manner, that guarantees satisfaction of both the obstacle avoidance and curvature bound constraints.

The solution to the basic path planning problem is based on constructing a visibility graph, and using standard algorithms to identify the shortest path between two graph vertices. In [40] this method was modified to replace the linear path segments by a G^1 sequence of PH quintics. The aim of the present approach is to retain the piecewise–linear solution, as far as possible, but to "round" its sharp corners using PH quintics meeting the linear segments with G^2 continuity. The obstacles are first offset[4] by a distance d, to be determined so as to ensure satisfaction of the obstacle avoidance and extremum curvature constraints. The optimal piecewise–linear path through these offset obstacles is then computed by means of the standard algorithms.

Consider the rounding of two line segments meeting at a corner point $\mathbf{p}_c$. The sharp corner is to be replaced by a PH quintic $\mathbf{r}(\xi)$, $\xi \in [\,0,1\,]$ starting at a point $\mathbf{r}(0) = \mathbf{p}_i$ on the incoming segment, and ending at a point $\mathbf{r}(1) = \mathbf{p}_o$ on the outgoing segment, where $|\mathbf{p}_c - \mathbf{p}_i| = |\mathbf{p}_o - \mathbf{p}_c|$ (= L, say). To ensure a G^2 rounded corner, the end tangents $\mathbf{t}(0)$ and $\mathbf{t}(1)$ should be parallel to $\mathbf{p}_c - \mathbf{p}_i$ and $\mathbf{p}_o - \mathbf{p}_c$ respectively and the end curvatures must satisfy $\kappa(0) = \kappa(1) = 0$. It is convenient consider data in the canonical form[5]

$$\mathbf{p}_i = (0,0)\,, \quad \mathbf{p}_c = (L,0)\,, \quad \mathbf{p}_o = ((1+\cos\theta)L, \sin\theta\, L)\,,$$

the "turning angle" θ being measured positive anti–clockwise (see Figure 48). The solution for general data $\mathbf{p}_i, \mathbf{p}_c, \mathbf{p}_o$ is then obtained [35] by taking $\mathbf{p}_0 = \mathbf{p}_i$ in (4.1), and multiplying the coefficients $\mathbf{w}_0, \mathbf{w}_1, \mathbf{w}_2$ of (3.5) by $\exp(\mathrm{i}\frac{1}{2}\phi)$ where, writing $\mathbf{p}_i = (x_i, y_i)$ and $\mathbf{p}_c = (x_c, y_c)$, the angle ϕ is determined from

$$\cos\phi = \frac{x_c - x_i}{|\mathbf{p}_c - \mathbf{p}_i|}\,, \quad \sin\phi = \frac{y_c - y_i}{|\mathbf{p}_c - \mathbf{p}_i|}\,.$$

---

[4] For simplicity, we use a "polygonal" offset, without circular fillets at the sharp corners: the obstacle avoidance constraint is nevertheless exactly observed. Note that the obstacle offset is not required by the method in [40] if the vehicle is of negligible size.

[5] The canonical form employed here is somewhat different from that used in Section 4.4, in the context of paths with equal arc lengths.

The solution for canonical–form data is specified [35] in terms of L and θ by

$$\mathbf{w}_0 = \lambda\sqrt{L}\,, \quad \mathbf{w}_1 = 0\,, \quad \mathbf{w}_2 = \lambda\sqrt{L}\exp(\mathrm{i}\tfrac{1}{2}\theta)\,, \tag{4.11}$$

where

$$\lambda = \sqrt{30\cos\tfrac{1}{2}\theta/(6\cos\tfrac{1}{2}\theta+1)}\,.$$

The control points of $\mathbf{r}(\xi)$ are obtained from (4.1). Note that, since $\mathbf{w}_1 = 0$, we have $\mathbf{p}_2 = \mathbf{p}_1$ and $\mathbf{p}_4 = \mathbf{p}_3$, so the G^2 PH quintic corner curves have just four distinct control points [20] — examples are illustrated in Figure 48. The parametric speed (4.2) of the corner curve has the Bernstein coefficients

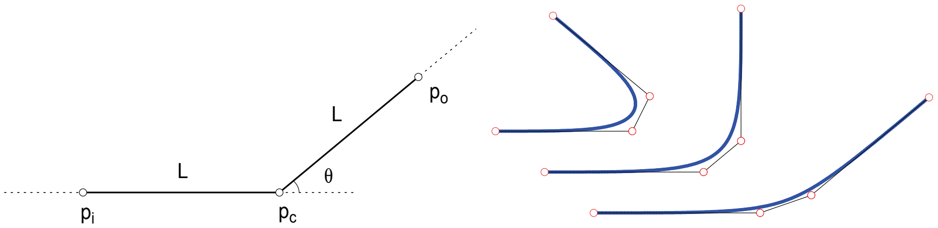


Figure 48. Left: canonical data defining a G^2 PH quintic corner curve. Right: examples of PH quintic corner curves for turning angles θ = 45°, 90°, 135°.

$$\sigma_0 = \lambda^2 L\,, \quad \sigma_1 = 0\,, \quad \sigma_2 = \frac{\lambda^2 L\cos\frac{1}{2}\theta}{3}\,, \quad \sigma_3 = 0\,, \quad \sigma_4 = \lambda^2 L\,,$$

and the curvature can be expressed as

$$\kappa(\xi) = 4\lambda^2 L\sin\tfrac{1}{2}\theta\,\frac{(1-\xi)\xi}{\sigma^2(\xi)}\,. \tag{4.12}$$

The extremum curvature $\kappa_e = \kappa(\frac{1}{2})$ and total arc length S are given [35] by

$$\kappa_e = \frac{32(6\cos\frac{1}{2}\theta+1)\tan\frac{1}{2}\theta}{15L(\cos\frac{1}{2}\theta+1)^2} \quad\text{and}\quad S = \frac{2L(6+\cos\frac{1}{2}\theta)\cos\frac{1}{2}\theta}{6\cos\frac{1}{2}\theta+1}\,. \tag{4.13}$$

Finally, the deviation of the mid–point $\mathbf{r}(\frac{1}{2})$ from the corner point $\mathbf{p}_c$ is

$$\delta = \frac{(3\cos\frac{1}{2}\theta+8)\,|\sin\frac{1}{2}\theta\,|\,L}{8(6\cos\frac{1}{2}\theta+1)}\,. \tag{4.14}$$

Since θ is fixed by the corner geometry, the parameter L should be used to adjust the extremum curvature κ_e and deviation δ.

For the (non–rounded) offset at distance d to a convex polygonal obstacle, the offset point to a vertex with a turning angle θ is distance $d\sec\frac{1}{2}\theta$ from that vertex, and to prevent intrusion of the corner rounding curve within the obstacle, this must exceed the value (4.14) — i.e. we must have

$$d \sec \tfrac{1}{2}\theta \; > \; \frac{(3\cos\frac{1}{2}\theta + 8)\,|\sin\frac{1}{2}\theta\,|\,L}{8(6\cos\frac{1}{2}\theta + 1)} \,,$$

or equivalently

$$L \; < \; L_{max} \; := \; \frac{8(6 + \sec\frac{1}{2}\theta)\,d}{(3\cos\frac{1}{2}\theta + 8)\,|\sin\frac{1}{2}\theta\,|} \,. \tag{4.15}$$

Now if κ_{max} is the maximum allowed magnitude of path curvature, and $r_{min} = 1/\kappa_{max}$ is the minimum radius of curvature, from (4.13) the condition $|\kappa_e| < \kappa_{max}$ becomes

$$\frac{32(6\cos\frac{1}{2}\theta + 1)|\tan\frac{1}{2}\theta|}{15L(\cos\frac{1}{2}\theta + 1)^2} \; < \; \frac{1}{r_{min}} \,,$$

or equivalently

$$L \; > \; L_{min} \; := \; \frac{32(6\cos\frac{1}{2}\theta + 1)|\tan\frac{1}{2}\theta|\, r_{min}}{15(\cos\frac{1}{2}\theta + 1)^2} \,. \tag{4.16}$$

If the bounds (4.15) and (4.16) on L are to be compatible, we must have

$$\frac{8(6 + \sec\frac{1}{2}\theta)\,d}{(3\cos\frac{1}{2}\theta + 8)\,|\sin\frac{1}{2}\theta\,|} \; > \; \frac{32(6\cos\frac{1}{2}\theta + 1)|\tan\frac{1}{2}\theta|\, r_{min}}{15(\cos\frac{1}{2}\theta + 1)^2} \,,$$

which can be reduced to

$$d \; > \; \frac{4(3\cos\frac{1}{2}\theta + 8)\sin^2\frac{1}{2}\theta}{15(\cos\frac{1}{2}\theta + 1)^2}\, r_{min} \,. \tag{4.17}$$

The factor multiplying r_{min} on the right is monotonically increasing with $|\theta|$, from 0 when $\theta = 0$ to $32/15$ when $\theta = \pm\pi$. If r_{min} is specified *a* priori, and the largest turning angle magnitude θ_{max} is identified, substituting it in (4.17) determines the minimum offset distance d_{min} ensuring satisfaction of both the maximum path curvature and obstacle non–intrusion conditions. With $d = d_{min}$, we have $L_{min} = L_{max}$ at the corner(s) for which $|\theta| = \theta_{max}$, and the particular value $L = L_{min} = L_{max}$ must be used. At the other corners, it may be possible to use larger values of L, but the sum of the L values for two consecutive corners cannot exceed the linear path length between them. Thus, for small and closely–spaced obstacles, it may not be possible to satisfy both the collision avoidance and curvature bound constraints.

Figure 49 shows the nominal piecewise–linear path among a field of offset obstacles, and this path with G^2 PH quintic rounded corners using $d = 0.2136$

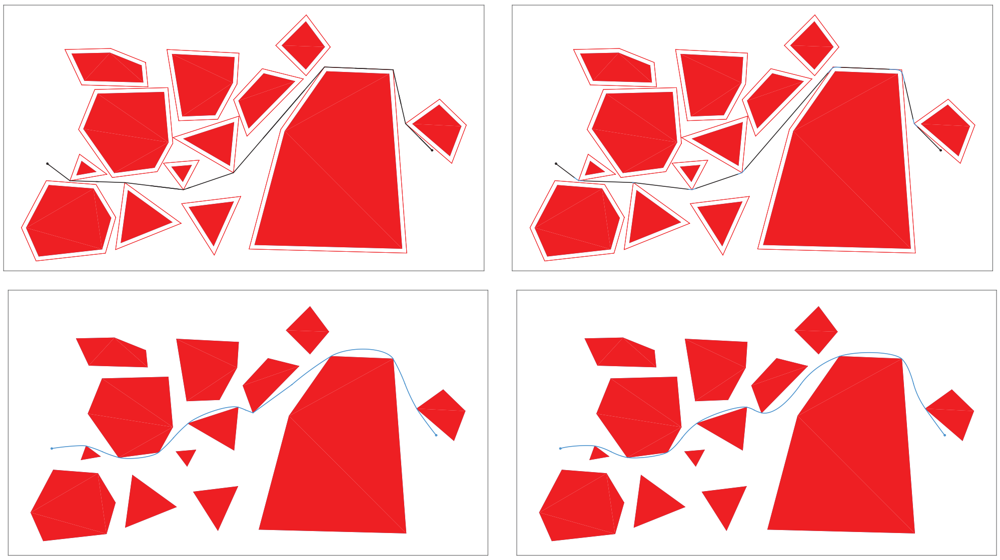


Figure 49. Top: piecewise–linear path among a field of offset obstacles (left) and this path with G^2 PH quintic rounded corners (right). Bottom: G^1 PH quintic paths with tangents defined by the T1 (left) and T2 (right) method.

and the curvature bound $|\kappa| \leqslant 5$. Also shown are two paths specified by G^1 PH quintic segments, that do not require obstacles offsets and use different strategies for assigning the nodal tangents [40] — the T1 method employs a weighted average of successive linear displacement vectors and the T2 method uses tangents obtained from a C^2 PH quintic spline fit [28]. The piecewise–linear path with G^2 rounded corners has an overall length 23.8254, somewhat less than the value 23.9394 for the sharp–corner path. Note that the two G^1 piecewise PH quintic paths take a different route through the obstacle field, and have larger overall lengths — namely, 25.6534 using the T1 method for assigning nodal tangents, and 25.3819 using the T2 method.

Figure 50 compares curvature plots for the G^2 rounded–corner path and the two G^1 paths. The former evidently attains the curvature bound $\kappa_{max} = 5$ around each corner. Using the T1 tangent assignment method, the G^1 path clearly exceeds this bound, and exhibits large curvature discontinuities. With the T2 tangent method, the curvature extrema are substantially suppressed, although significant path curvature discontinuites are still evident. Figure 51 compares the proposed G^2 corner rounding scheme with the G^2 PH quintic spline obtained using the strategy described in Section 3.3.2 (see also [18]). The curvature bound is set to 5, since higher values lead to overlapping offsets. The curvature plots reported on the right of Figure 51 show that the curvature is remarkably lower than 5 for both methods and that the curvature variation is reduced for the tension scheme. A more intricate example is shown in Figure 52, involving a path through a single polygonal obstacle with many convex and concave

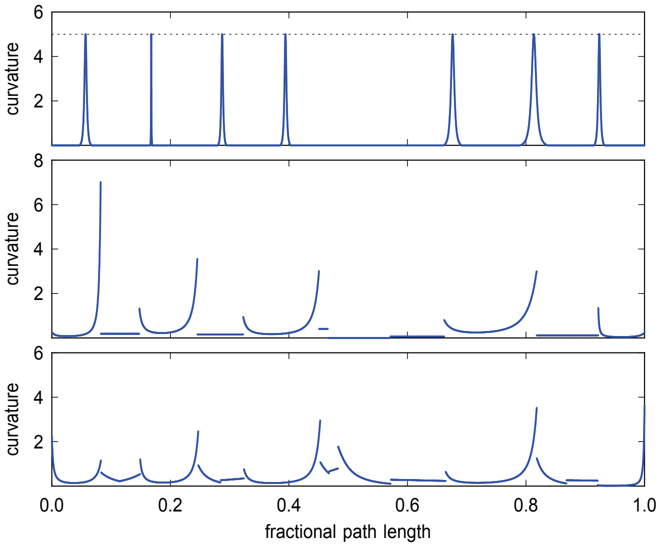


Figure 50. Upper: curvature magnitude for the piecewise–linear path with G^2 PH quintic rounded corners, showing the curvature bound $|\kappa| \leqslant 5$. Also shown are curvature magnitudes for the G^1 piecewise PH quintic paths with nodal tangents defined by the T1 method (center) and T2 method (lower).

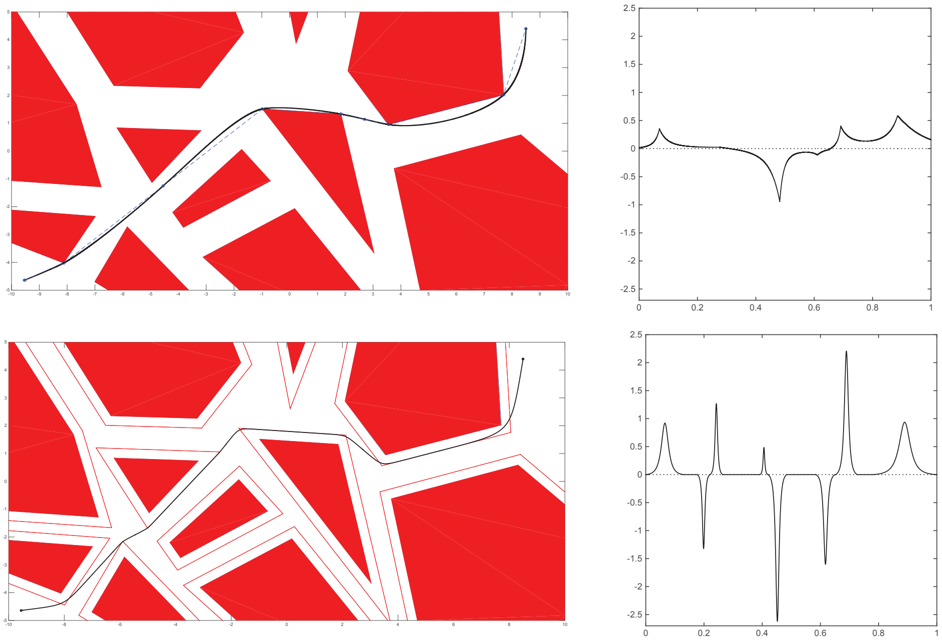


Figure 51. Top left: the admissible path produced by our $C^1 \cup G^2$ interpolation scheme with tension. Bottom left: the enlarged obstacles and the admissible G^2 path produced by the PH corner rounding approach with curvature constraint. The curvature plots of the two paths are also shown on the right.

corners, using the values $d = 0.1736$ and $\kappa_{max} = 5$. The piecewise–linear path involves 14 sharp corners, and in every case the G^2 PH quintic corner rounding curves avoid obstacle encroachment and exactly attain the specified curvature bound. Obviously the curvature bounds have to be properly chosen by also taking into account the separation distance of the obstacles. The examples shown in Figures 49 and 52 employ the value L_{min} defined by (4.16) at each corner, so the corner curves are of minimun size and attain the prescribed curvature magnitude bound κ_{max}. Using larger L values yields lower corner curvatures, and somewhat smaller total path lengths. However, L values smaller than L_{max} defined by (4.15) are generally necessary, to avoid any overlap of consecutive corner curves. Figure 53 compares the path and curvature distribution for the $L = L_{min}$ case in Figure 49 with a case in which L is as large as possible for each corner, without overlap of the corner curves (the path still avoid encroachment on any of the obstacles). The use of larger L values evidently yields smaller peak curvature magnitudes, and the overall arc length is slightly reduced, from $S = 23.8254$ to $S = 23.5836$.

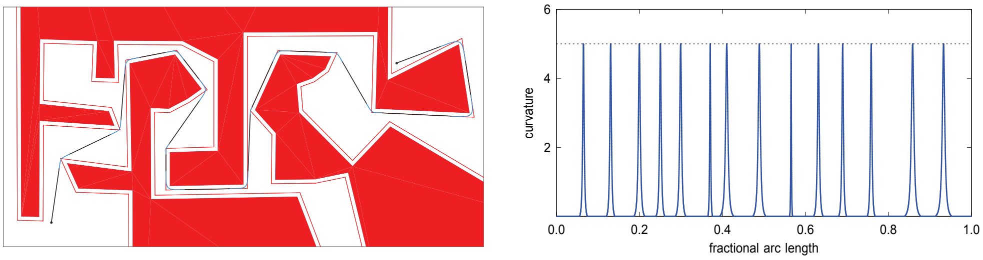


Figure 52. Left: piecewise–linear path through a single intricate obstacle with sharp corners rounded by G^2 PH quintic segments that satisfy the curvature bound $|\kappa| \leqslant 5$. Right: variation of the curvature magnitude along this path.

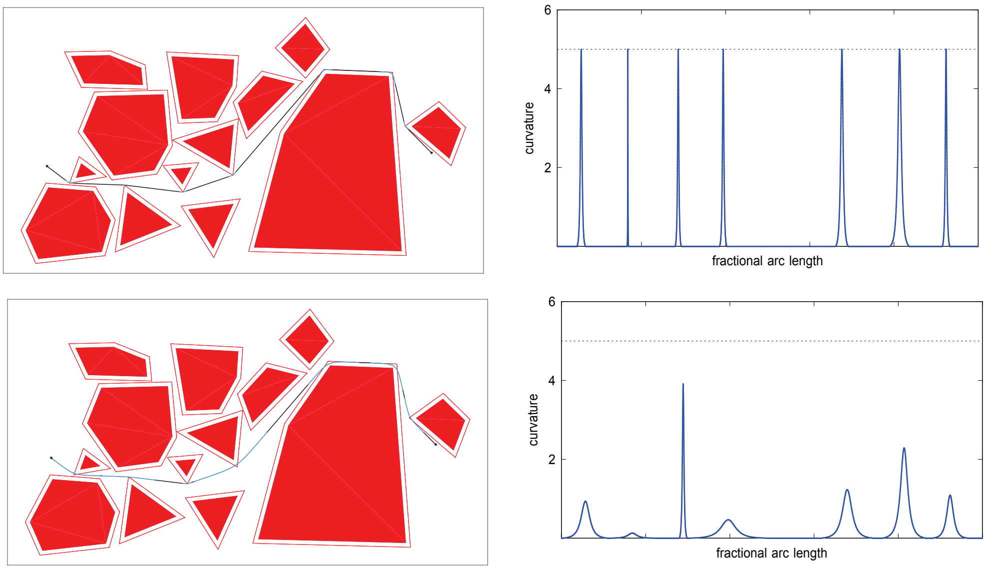


Figure 53. Paths and curvature plots for the obstacle field in Figure 49 using PH corner curves of minimum size L_{min} in each case (upper), and the largest size L in each case that does not incur overlap of the corner curves (lower).

5

TRAJECTORY PLANNING WITH C^2 CONTINUOUS TIME–DEPENDENT FEEDRATE SCHEDULING

5.1 INTRODUCTION

Trajectory planning is a fundamental issue in several fields related to automation, such as robotics, digital animation and computer aided manufacturing. It requires the identification of a feasible path and the definition of an admissible time law. Even if conventional controllers were restricted to piecewise linear or circular paths, modern software systems enable the design of smooth curvilinear paths where zero velocity values can be avoided. Once the path is identified, the design of a suitable time law is crucial to achieve high-speed control. This task corresponds to specify the object speed along the path, generally called *feedrate* function.

The final output of a trajectory planning scheme are the *reference points* for the control system. They indicate the prescribed machine positions with respect to uniform time spacing and constitute the input values for the controller to guide the movement of the machine. The feedrate scheduler provides the velocity profile for the planned path by taking into account the given geometric path and kinematic constraints. A characterizing feature of any feedrate scheduler consists in the level of kinematic control it ensures or tries to ensure which is highly variable in different approaches. For example, full jerk control is rarely ensured, since the contribution of its centripetal component is often ignored. Smooth motions are strongly preferable since the time law should avoid unnecessary vibrations that may also cause mechanical damages and only schedulers providing C^2 feedrate functions allow the possibility of ensuring acceleration and jerk continuity[1]. This latter is an important feature of the scheduler since sudden changes or reversal in the kinematic profiles are avoided in the resulting motion, see e.g., [38].

In view of their compatibility with standard spline representation of Computer Aided Design systems, different kind of Non-Uniform Rational B-splines (NURBS) interpolators were recently proposed. They can follow an offline or online approach, whether the scheduling computation are executed before or during the interpolation. While the offline interpolator avoids real-time interaction at the price of using additional storage of computed data, the online approach requires more computational power at running time. Examples of

---

[1] Acceleration and jerk continuity are ensured if the path is sufficiently smooth.

offline and online NURBS interpolators with chord error, acceleration and jerk limitations were presented in [53, 81] and [3], respectively.

In the literature the feedrate function is often expressed directly in time terms but also in terms of different variables, i.e. the curve parameter or the curvature. For example, expressing it with respect to the curvature highly facilitates the chord error control, an important feature in CNC machining and processing, see e.g., [37, 82]. Real–time interpolator algorithms for different feedrate variations (parameter–dependent, curvature–dependent and hybrid formulations) were also recently considered in connection with a corner rounding approach for high–speed machining [35]. A time-dependent feedrate instead simplifies the development of controllers that take into account different kinematic constraints. Among the wide set of references in this setting, trigonometric feedrate profiles have been considered by several authors in connection with different continuity requirements, see e.g., [53] for the C^1 case and [81] for the case of higher continuity. In this dissertation we present offline time-dependent feedrate scheduler on planar paths providing a C^2 quintic spline feedrate profile. The scheduler first scans the path to detect a set of special points which are used to subdivide the curve into successive blocks where the feedrate is defined with a typically trapezoidal profile as depicted in Figure 54.

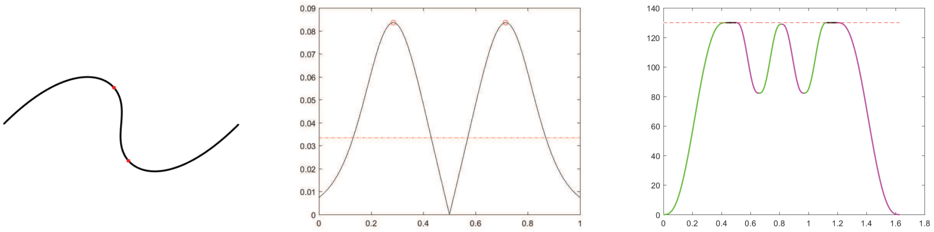


Figure 54. An example of PH curve subdivision for the feedrate generation: a PH curve with special point detected depicted as red dots (left), the related absolute value of the curvature (center) exploited to subdivide the curve with a given threshold (dashed line) and the curvature peaks highlighted (red circles) and the resultant trapezoidal profile (right).

Preliminarily, the blocks maximal feedrate values are suitably assigned, as well as the feedrate values at special points, taking into account not only the kinematic constraints but also the constraints connected to the blocks length. Then in each block the feedrate is defined as a 3–pieces C^2 piecewise quintic polynomial function of the time variable, where each block traversal time is minimized taking into account suitably derived bounds on the feedrate first and second derivatives.

The proposed scheduler is configurable since it presents two kinds of parameters represented by the kinematics bounds and the value of the curvature related to each curve block. Setting the kinematic bounds desired acceleration, tangential or full jerk control can be obtained meanwhile considering the mean or the maximum curvature, the aforementioned kinematic bounds can be imposed respectively in a weak and strong way. In order to exploit accurate and reliable real-time interpolators, the class of Pythagorean-hodograh (PH) curves is here considered [22] for specifying the considered planar paths. The polynomial nature of the parametric speed of a PH curve enables an exact computation of its arc-length and leads to robust processing algorithms. By avoiding repeated numerical approximations, the precision of the method is strongly enforced. The results obtained with Pythagorean-hodograh spline curves in CNC applications have confirmed the potential of this approach, see e.g., [30, 80, 75], as well as [22] and references therein. The complete trajectory planning scheme should properly integrate the feedrate scheduling with the geometric part of the motion. For this reason we have used PH spline scheme introduced in [28, 32] with a suitable choice of tension parameters as described in Section 3.2 and 3.3. These path planning schemes can be suitably combined with the feedrate scheduler here proposed to properly define an optimal trajectory for a given path. The structure of this chapter is as follows. Section 5.2 introduces some preliminary concepts and the kinematic constraints here considered. The two developed configurable feedrate schedulers are presented in Section 5.3 while the CNC interpolator algorithm for PH curves is reviewed in Section 5.4. Finally, Section 5.5 illustrates the performance of our approach on two significant example (see [39] for some others).

5.2 PRELIMINARIES AND KINEMATIC CONSTRAINTS

We assume that a planar path is given in parametric form, $\mathbf{r}(\xi), \xi \in [\xi_L\,, \xi_R]$ with $\mathbf{r}$ sufficiently smooth. Then we are interested in determining a suitable time law for traversing the given path. If t denotes the time and s denotes the *a*rc length, this means that we are interested in defining the *f*eedrate $\nu(t)$, where

$$\nu(t) := \frac{ds}{dt}(t)$$

whose prescription allows also us to get the following equation implicitly characterizing the corresponding global traversal time T,

$$\int_0^T \nu(t)\,dt = S\,, \tag{5.1}$$

where S is the total length of the path. Note that in the literature the feedrate is also sometimes given as a function of ξ and in such case T can be explicitly obtained from the following formula,

$$T = \int_{\xi_L}^{\xi_R} \frac{\sigma}{\nu}\,d\xi\,,$$

where $\sigma := \frac{ds}{d\xi} = |\mathbf{r}'(\xi)|$ denotes the curve *p*arametric speed. In this paper we are interested in prescribing ν directly as a function of the time variable, so we use the relation in (5.1). Note that the absolute value $|\nu|$ of the feedrate is the velocity module, since the velocity vector $\mathbf{v} := \dot{\mathbf{r}}$ (where the dot denotes a derivative with respect to time) along the trajectory is defined at each regular point (i.e. at points where σ does not vanish) as,

$$\mathbf{v} = \nu\,\mathbf{t}\,, \tag{5.2}$$

where $\mathbf{t}$ is the unit tangent vector to the curve, that is $\mathbf{t} := \mathbf{r}'/|\mathbf{r}'|$, where the prime symbol denotes derivative with respect to the parameter ξ. Thus the absolute value of both the Cartesian components of the velocity is upper bounded from V_m if we require that

$$0 \leqslant \nu \leqslant V_m\,, \tag{5.3}$$

where V_m is called the *c*ommanded feedrate. The required positivity for ν means that we are interested only in motions avoiding direction inversions. Furthermore we have that the acceleration vector $\mathbf{a} := \ddot{\mathbf{r}}$ is well defined at any regular point and its expression is the following,

$$\mathbf{a} = \dot{\nu}\mathbf{t} - \nu^2\kappa\,\mathbf{n}\,, \tag{5.4}$$

since $\frac{dt}{ds} = -\kappa\mathbf{n}$, where

$$\mathbf{n} := \mathbf{t}\times\mathbf{z}, \text{(unit normal)} \quad \text{and} \quad \kappa := \frac{(\mathbf{r}'\times\mathbf{r}'')\cdot\mathbf{z}}{\sigma^3}\,\text{(curvature)}\,,$$

with $\mathbf{z}$ denoting a unit vector orthogonal to the path's plane. By a further differentiation of the expression of $\mathbf{a}$ given in (5.4), we obtain the expression at any regular point of the *j*erk vector $\mathbf{j} := \dddot{\mathbf{r}}$ in terms of the feedrate, of its first and second derivative and of geometric quantities connected to the given path,

$$\mathbf{j} \;=\; (\ddot{\nu} - \nu^3\kappa^2)\mathbf{t} \;-\; (3\nu\dot{\nu}\kappa + \nu^3\kappa'/\sigma)\mathbf{n}\,. \tag{5.5}$$

Now, since in CNC the motion is separately prescribed on each axis, the general interest is in bounding the absolute value of the Cartesian components of $\mathbf{a}$ and, often also those of $\mathbf{j}$. Now (5.4) can ensure for any path that both $|a_x| \leqslant A_m$ and $|a_y| \leqslant A_m$ only if we require that

$$|\dot{\nu}| \leqslant A_t := p_a\,A_m\,, \qquad \nu^2|\kappa| \leqslant A_c := \sqrt{1-p_a^2}\,A_m, \tag{5.6}$$

with $p_a \in (0\,,\,1)$, typically $p_a = 1/\sqrt{2}$. Analogously, from the expression in (5.5) we obtain that $|J_x| \leqslant J_m$ and $|J_y| \leqslant J_m$ is surely ensured if

$$|\ddot{\nu}| \leqslant J_{t1} \qquad \nu^3\kappa^2 \leqslant J_{t2} \tag{5.7}$$

and

$$3|\kappa|A_t v + |\kappa'|/\sigma v^3 \leqslant J_c, \tag{5.8}$$

where

$$J_{t1} := q_j p_j, \quad J_{t2} := (1-q_j)\, p_j J_m\,, \quad J_c := \sqrt{1-p_j^2}J_m\,,$$

with $p_j, q_j \in (0\,,\,1)$, typically $p_j = 1/\sqrt{2}$ and $q_j = 1/2$. Note that the inequality in (5.8) related to the jerk centripetal component control, is rarely considered in the literature, see [3]. Since (5.8) is an increasing polynomial in v, there exists exactly one positive value v_{jc} for v such that (5.8) holds if and only if $v \leqslant v_{jc}$.

When $k' = 0$ then $v_{j_j} = J_c/(3|k|A_t)$, otherwise when $\kappa' \neq 0$, the expression of v_{jc} is given by Cardano's rule,

$$v_{jc} = v_{jc}(\kappa, w) := \sqrt[3]{\lambda_+} + \sqrt[3]{\lambda_-}\,, \quad \text{with } \lambda_\pm = \frac{J_c}{|w|} \pm \sqrt{\left(\frac{J_c}{2|w|}\right)^2 + \left(\frac{\kappa A_t}{|w|}\right)^3}\,, \tag{5.9}$$

where $w := \kappa'/\sigma$ is the curvature derivative with respect to the arc length. Also chord error E_c is ensured during the whole path traversal. As shown in Figure 55, at a point with non vanishing curvature $E_c(j)$ is defined as the distance

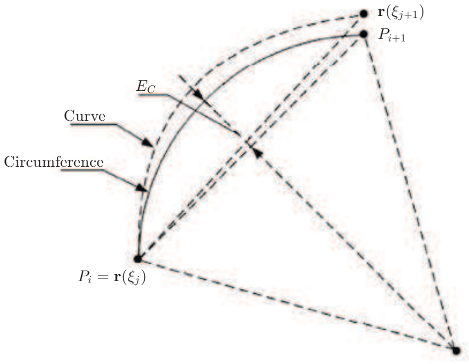


Figure 55. Chordal error $E_c(j)$ at the interpolated point $\mathbf{r}(\xi_j)$.

between the osculating circumference through the corresponding *i*nterpolated point $\mathbf{P}_j = \mathbf{r}(\xi_j)$ and a related circumference chord,

$$E_c(j) := \begin{cases} \frac{1}{|\kappa_j|} - \sqrt{\frac{1}{\kappa_j^2} - \frac{L_j^2}{4}}\,, & \text{if } \kappa_j \neq 0\,, \\ 0\,, & \text{otherwise.} \end{cases} \tag{5.10}$$

with $L_j := v(t_j)\Delta t$ which means that $L_j = \|\mathbf{P}_{j+1} - \mathbf{P}_j\|_2$, where $\mathbf{P}_{j+1}$ is the circumference next interpolated point, assuming to use a constant feedrate equal to $\nu(t_j)$ and considering that the tool moves on a straight line between any pair of successively interpolated points.

We can observe that chord error depends on the sampling time Δt since in (5.10) $L_j = \nu(t_j) * \Delta t$. Furthermore, if $\kappa_j \neq 0$, $E_c(j)$ is well defined only if $L_j \leqslant 2/|\kappa_j|$, that is only if $\nu(t_j)|\kappa_j|\Delta t \leqslant 2$.

The chordal error control is obtained by requiring that

$$E_c(j) \leqslant D\,, \tag{5.11}$$

where D is an assigned suitable positive threshold. Clearly, if $D \geqslant 1/|\kappa_j|$ then (5.11) holds. On the other hand, when $D < 1/|\kappa_j|$, with some computation it can be verified that the inequality in (5.11) corresponds to require the following condition on the feedrate

$$\nu(t_j) \leqslant \frac{2}{\Delta t}\sqrt{\frac{1}{\kappa_j^2} - \left(\frac{1}{|\kappa_j|} - D\right)^2}\,. \tag{5.12}$$

It is interesting to note also that, if at least $\nu(t_j) < \frac{2}{\Delta t|\kappa_j|}$, it is reasonable to use a first order Taylor expansion for the square root in (5.10) with respect to Δt^2, obtaining

$$E_c(j) \approx \frac{1}{|\kappa_j|}[1 - \frac{1}{2}(2 - \frac{\kappa_j^2\nu(t_j)^2\Delta t^2}{4})] = \frac{\Delta t^2}{8}\ |\kappa_j|\nu(t_j)^2\,.$$

This means that in such case the chord error is approximately proportional to the absolute value of the normal component of the acceleration. As mentioned in the 5.1, our strategy is configurable since we can choose among six alternative formulations with different level of imposed kinematic constraints. These alternatives are labelled as R0, R1, R2 and S0, S1, S2 and they can be collected into two groups, the *relaxed* formulations (R0, R1, R2) and the *strict* ones (S0, S1, S2). In any case the path is preliminarily divided into segments connected one to each other at curvature discontinuous points where the motion is stopped if such points exist on the given path.. Each segment is then further subdivided into blocks detecting on each segment a set of special points. In all the R formulations on each block the mean value of $|\kappa|$ is determined, in the R2 formulation together with the mean value of $|w|$. These values are used to impose the selected kinematic constraints in the block only in a mean sense, in order to obtain a reduced block traversal time. On the contrary in all the S formulations on each block the maximal values of the same geometric quantities are determined, in order to ensure a priori the desired kinematic constraints which always include the bound in (5.12) for chord error control. Besides that, the R0 and S0 formulations take into account only the bounds in (5.6) which concern acceleration, while R2 and S2 adds to (5.12) and (5.6) also all the bounds in (5.7) and (5.8) for full jerk control. R1 and S1 are hybrid formulations since they add to (5.12) and (5.6) only the bounds in (5.7) which are related only to the tangential component of the jerk.

5.3 C^2 TIME–DEPENDENT FEEDRATE SCHEDULING

For all the developed formulations and all the curve segments, the scheduler defines a feedrate function v such that

$$0 \leqslant v \leqslant V_m\,, \quad |\dot{v}| \leqslant A_t\,, \quad |\ddot{v}| \leqslant J_{t1}\,. \tag{5.13}$$

These inequalities however are not sufficient to ensure the desired chord, acceleration and jerk control. Besides them, considering the expressions in (5.4), (5.5) and (5.10), we can easily derive that , to ensure full kinematic control, the feedrate v should satisfy the upper bound $v \leqslant V_b(\kappa, w)$, where we recall that $w = \kappa'/\sigma$ and

$$V_b(\kappa, w) := \min\left\{ \frac{2}{\Delta t}\sqrt{\frac{1}{\kappa^2} - \left(\frac{1}{|\kappa|} - D\right)^2},\ \sqrt{\frac{A_c}{|\kappa|}},\ \sqrt[3]{\frac{J_{t2}}{\kappa^2}},\ v_{jc}(\kappa, w) \right\}, \tag{5.14}$$

with $v_{jc}(\kappa, w)$ defined as in (5.9). Clearly if for example we are not interested in guaranteeing jerk bounds, the minimum on the right hand side of (5.14) is just between the first two terms. Hence, the curve is preliminarily decomposed into *segments* whose extremes $\mathbf{B}_j, j = 1, \ldots, N-1$ called *breakpoints* are assigned points where the curve is just G^0, i.e. corner points, augmented by the initial and final positions $\mathbf{B}_0 = \mathbf{r}(\xi_L)$ and $\mathbf{B}_N = \mathbf{r}(\xi_R)$. At each breakpoint in any case we set to zero $v, \dot{v}$ and also $\ddot{v}$. We will focus below on the generic j–th curve segment, for brevity removing the subscript (j) and also assuming that time is reported to zero whenever a new segment is encountered.

As mentioned in the previous sections, on each curve segment the scheduler first of all detects a set of special points. These points allow us to subdivide the segment into blocks where a feedrate trapezoidal profile is defined. The special points are of different type for the R and S formulations and they are respectively called *critical* or *crossing* points. In the first case they are points where the absolute curvature has a local maximum exceeding a critical curvature value κ_{cr} while in the second case they are points where $|\kappa| - \kappa_{cr}$ has a sign change. The critical curvature is defined as follows for R1,S1,R2 and S2,

$$\kappa_{cr} := \min\left\{ \frac{8D}{V_m^2 \Delta t^2 + 4D^2},\ \frac{A_c}{V_m^2},\ \sqrt{\frac{J_{t2}}{V_m^3}} \right\}. \tag{5.15}$$

Note that this is the curvature value such that, assuming (5.13), if a point of the path is encountered with a feedrate greater than V_m, one among the chord error, the normal component of acceleration and the tangential component of the jerk exceed the related bounds. On the other hand, for the formulations R0 and S0, κ_{cr} reduces to the minimum between the first two terms on the right hand side of (5.15), since they do not care of jerk bounds. The special points are denoted in the following as $\mathbf{P}_1, \dots, \mathbf{P}_{M-1}$ and for notational convenience we use also the points $\mathbf{P}_0$ and $\mathbf{P}_M$ to denote the considered segment endpoints.

5.3.1 *The Relaxed (R) formulations of the scheduler*

These formulations are called Relaxed (R) because on each curve block they use just mean values for κ and w to try to ensure the desired kinematic control. As mentioned in the previous section, R0 cares only of acceleration bounds, while R1 adds also the two bounds in (5.7) and R2 requires both (5.7) and (5.8).

Feedrate initialization at critical points

Then we start by initializing the feedrate values at any critical point $\mathbf{P}_i, i = 0, \dots, M$. At $\mathbf{P}_0$ and $\mathbf{P}_M$ we prescribe zero values for v and also for $\dot{v}$ and $\ddot{v}$. Conversely, at each $\mathbf{P}_i, i = 1, \dots, M-1$, a sufficiently low positive value $V_i < V_m$ of the feedrate is assigned for v, while $\dot{v}$ and $\ddot{v}$ are again set to zero. In particular the following value $\hat{V}_i < V_m$ is chosen with R1 and R2 to initialize V_i is selected in order to ensure the desired kinematic control at these points,

$$\hat{V}_i := V_{r1}(\kappa_i), \tag{5.16}$$

where κ_i is the value of κ at $\mathbf{P}_i$ and

$$V_{r1}(\kappa) := \min \left\{ \frac{2}{\Delta t} \sqrt{\frac{1}{\kappa^2} - \left(\frac{1}{|\kappa|} - D\right)^2}, \sqrt{\frac{A_c}{|\kappa|}}, \sqrt[3]{\frac{J_{t2}}{\kappa^2}}, \right\}. \tag{5.17}$$

Not being interested in jerk control, the R0 formulation fixes $\hat{V}_i = V_{r0}(\kappa_i)$, where

$$V_{r0}(\kappa) := \min \left\{ \frac{2}{\Delta t} \sqrt{\frac{1}{\kappa_i^2} - \left(\frac{1}{|\kappa|} - D\right)^2}, \sqrt{\frac{A_c}{|\kappa|}} \right\}. \tag{5.18}$$

Note that, considering that κ' vanishes at critical points and that $\dot{v}$ is set to zero, the centripetal jerk vanishes at these points. Furthermore, being $\ddot{v}$ also set to zero, even the tangential acceleration vanishes. This implies that in (5.17) we could replace A_c and J_{t2} respectively with A_m and J_m, see for example [82], even if we have not considered this fact for defining our scheduler since our experiments have shown that this change often is not profitable [2].

Observe however that the V_i values can be further reduced for compatibility reasons connected to the lengths $S^{(i)}, i = 1, \dots, M$, see Remark 5.

---

[2] Actually in our tests we have worked with paths just admitting a C^2 parametric spline representation and often some critical points are fixed at points where w is discontinuous and can remain high in the left and/or right neighborhood of these points.

Initialization of maximal feedrate value of the curve blocks

After feedrate values at critical points has been prescribed (see Section 5.3), the motion between $\mathbf{P}_{i-1}$ and $\mathbf{P}_i$ has to be specified accordingly. Following [53] we require a *trapezoidal* feedrate profile, i.e. $\dot{v} > 0$ (acceleration phase) then $\dot{v} = 0$ (constant phase) and finally $\dot{v} < 0$ (deceleration phase). However we use a different analytical definition of the feedrate which produces a C^2 profile. In order to reduce the traversal time, we plan the feedrate so that its central constant phase at maximal feedrate $v = V_c^{(i)}$ is as long as possible. The maximal block feedrate value $V_c^{(i)}$ is initialized with $\hat{V}_c^{(i)}$ defined as follows,

$$\hat{V}_c^{(i)} :=, \begin{cases} V_{r0}(\kappa_c^{(i)}) & \text{with R0}, \\ V_{r1}(\kappa_c^{(i)}) & \text{with R1}, \\ V_b(\kappa_c^{(i)}, w_c^{(i)}) & \text{with R2}, \end{cases} \tag{5.19}$$

where $\kappa_c^{(i)}, w_c^{(i)}$ are the mean absolute values of $|\kappa|$ and $|w|$ between $\mathbf{P}_{i-1}$ and $\mathbf{P}_i$. Observe that with the R2 formulation it can happen that $V_c^{(i)} < \hat{V}_{i-1}$ or $V_c^{(i)} < \hat{V}_i$. Thus, in such case, after initializing all the block maximal feedrates, we modify the initialization of the feedrate values at critical points as follows,

$$\hat{V}_i = \min\{\hat{V}_c^{(i)}, \hat{V}_c^{(i+1)}, \hat{V}_i\}.$$

Observe also that, as well as the feedrate values at the critical points, also $V_c^{(i)}$ is possibly later further reduced in the algorithm for compatibility reasons connected to the length $S^{(i)}$ available to arrive at this maximum feedrate value under the considered bounds (see the details below).

In any case, the feedrate in the i–th block, $i = 1, \dots, M$, will have a central constant profile $v = V_c^{(i)}$ as long as possible with $V_c^{(i)}$ as near as possible to $\hat{V}_c^{(i)}$. Note that in some cases, depending on $S^{(i)}$ and on V_{i-1} and V_i, it can also happen that the three phases collapse to just two phases or to a unique phase.

Time–dependent feedrate scheduling on a curve block

In order to design a feedrate profile with the described shape and capable to ensure also the continuity of the velocity, of the acceleration and of the jerk vectors at all points of $\mathbf{r}$ where κ' is continuous, we require v to be a quintic spline function with C^2 continuity which can be easily represented in piecewise Bézier form. We denote the time interval of the i–th block as follows

$$[t_s^{(i)}, t_s^{(i)} + T^{(i)}]$$

with

$$t_s^{(1)} := 0, \quad t_s^{(i+1)} := t_s^{(i)} + T^{(i)}, \quad T^{(i)} := T_{acc}^{(i)} + T_{con}^{(i)} + T_{dec}^{(i)}.$$

where $t_s^{(i)}$ and $T^{(i)}$ denote respectively the starting and the total time spent in the acceleration ($T_{acc}^{(i)}$), constant ($T_{con}^{(i)}$) and deceleration ($T_{dec}^{(i)}$) phase related to the the traversal of the i–th block.

Considering the i–th block we can observe that the associated traversal time $T^{(i)}$ clearly can not be a priori computed, since it depends on the feedrate function $v^{(i)}(t)$ itself which is defined as follows,

$$v^{(i)}(t) := \begin{cases} v_{acc}^{(i)}(t) & \text{if } t_s^{(i)} \leqslant t < t_s^{(i)} + T_{acc}^{(i)}\,, \\ v_{con}^{(i)}(t) & \text{if } t_s^{(i)} + T_{acc}^{(i)} \leqslant t < t_s^{(i)} + T_{acc}^{(i)} + T_{con}^{(i)}\,, \\ v_{dec}^{(i)}(t) & \text{if } t_s^{(i)} + T_{acc}^{(i)} + T_{con}^{(i)} \leqslant t < t_s^{(i)} + T^{(i)}\,, \end{cases}$$

where in the constant phase we fix $v_{con}^{(i)}(t) \equiv V_c^{(i)}$. In particular we assume the following definitions for the feedrate in the acceleration phase,

$$v_{acc}^{(i)}(\tau) := V_{i-1} \sum_{j=0}^{2} b_j^5(\tau) + V_c^{(i)} \sum_{j=3}^{5} b_j^5(\tau)\,, \tag{5.20}$$

with

$$\tau := (t - t_s^{(i)})/T_{acc}^{(i)}\,,$$

and with $b_j^n, j = 0, \ldots, n$, denoting the j–th Bernstein polynomial of degree n,

$$b_j^n(\tau) := \binom{n}{j} \tau^j \, (1-\tau)^{n-j}\,.$$

Analogously, we define

$$v_{dec}^{(i)}(\tau) := V_c^{(i)} \sum_{j=0}^{2} b_j^5(\tau) + V_i \sum_{j=3}^{5} b_j^5(\tau)\,, \tag{5.21}$$

with

$$\tau := (t - t_s^{(i)} - T_{acc}^{(i)} - T_{con}^{(i)})/T_{dec}^{(i)}\,.$$

Now let us consider the constraints needed to impose in order to ensure the fulfillment of the inequalities in 5.13 and of the following compatibility condition,

$$S_{acc}^{(i)} + S_{dec}^{(i)} \leqslant S^{(i)}\,, \tag{5.22}$$

where $S_{acc}^{(i)}$ ($S_{dec}^{(i)}$) denotes the length of the portion of the i–th block of $\mathbf{r}$ done in the acceleration (deceleration) phase. Concerning the requirements in 5.13, we observe that, since $0 \leqslant V_j < V_c^{(i)} \leqslant V_m\,, j = i-1, i$, considering the properties of the Bernstein basis we can deduce that the first inequality is surely satisfied on both sides. On the other hand the other two inequalities need to be imposed. Now, considering that

$$\dot{v}_{acc}^{(i)}(t) = \frac{1}{T_{acc}^{(i)}} \frac{dv_{acc}^{(i)}}{d\tau} = 5\,\frac{V_c^{(i)} - V_{i-1}}{T_{acc}^{(i)}}\, b_2^4(\tau)$$

we obtain that $\dot{v}_{acc}^{(i)} \geqslant 0$. On the other hand we have to require the second and third inequalities in (5.13), that is

$$5\,\frac{V_c^{(i)} - V_{i-1}}{T_{acc}^{(i)}} \max_{\tau\in[0,1]} b_2^4(\tau) \leqslant p_a A_m\,, \qquad 5\,\frac{V_c^{(i)} - V_{i-1}}{(T_{acc}^{(i)})^2} \max_{\tau\in[0,1]} |(b_2^4)'(\tau)| \leqslant J_m\,.$$

Considering the expression of $b_2^4(\tau)$ these inequalities become,

$$\frac{V_c^{(i)} - V_{i-1}}{T_{acc}^{(i)}} \leqslant \frac{8}{15}\, p_a A_m\,, \qquad \frac{V_c^{(i)} - V_{i-1}}{(T_{acc}^{(i)})^2} \leqslant \frac{\sqrt{3}}{10}\, J_m\,.$$

Then, if $V_c^{(i)}$ is set equal to $x^{(i)}$, with $\hat{V}_c^{(i)} \geqslant x^{(i)} \geqslant \max\{V_{i-1}\,,\,V_i\}$, we obtain that $T_{acc}^{(i)}$ has to be fixed as follows in order to minimize the time necessary to arrive at the constant phase,

$$T_{acc}^{(i)} = T_{acc}^{(i)}(x^{(i)}) := \max\left\{\frac{15}{8p_a A_m}\,(x^{(i)} - V_{i-1})\,,\ \sqrt{\frac{10}{\sqrt{3}J_m}\,(x^{(i)} - V_{i-1})}\right\}\,. \tag{5.23}$$

With analogous computations we can verify that $\dot{v}_{dec}^{(i)} \leqslant 0$ and that we need to fix $T_{dec}^{(i)}$ as follows in order to ensure that the second and the third inequality in (5.13) holds also in the deceleration phase with minimal time for the deceleration,

$$T_{dec}^{(i)} = T_{dec}^{(i)}(x^{(i)}) := \max\left\{\frac{15}{8p_a A_m}\,(x^{(i)} - V_i)\,,\ \sqrt{\frac{10}{\sqrt{3}J_m}\,(x^{(i)} - V_i)}\right\}\,. \tag{5.24}$$

In the two formulas (5.23) and (5.24) we have emphasized the dependency of $T_{acc}^{(i)}$ and $T_{dec}^{(i)}$ on the value $x^{(i)} \in [\max\{V_{i-1}\,,\,V_i\},\,\hat{V}_c^{(i)}]$ to be selected for $V_c^{(i)}$. In particular we will fix $V_c^{(i)}$ equal to the maximum value of $x^{(i)}$ in such interval which guarantees the fulfillment of (5.22). Now, since (5.1) implies that

$$S_{acc}^{(i)} \;=\; T_{acc}^{(i)} \int_0^1 v_{acc}^{(i)}(\tau)\,d\tau\,, \quad S_{dec}^{(i)} \;=\; T_{dec}^{(i)} \int_0^1 v_{dec}^{(i)}(\tau)\,d\tau\,,$$

we need to impose the following constraint,

$$T_{acc}^{(i)} \int_0^1 v_{acc}^{(i)}(\tau)\,d\tau \;+\; T_{dec}^{(i)} \int_0^1 v_{dec}^{(i)}(\tau)\,d\tau \leqslant S^{(i)}\,,$$

which, considering (5.20) and (5.21), in view of the integration formulas for Bernstein polynomials is equivalent to the following inequality,

$$g^{(i)}(x^{(i)}) \;\leqslant\; 2S^{(i)}\,, \tag{5.25}$$

with,

$$g^{(i)}(x^{(i)}) := T^{(i)}_{acc}(x^{(i)})[V_{i-1} + x^{(i)}] + T_{dec}(x^{(i)})[V_i + x^{(i)}]\,. \qquad (5.26)$$

Thus, we choose $V_c^{(i)} = \hat{V}_c^{(i)}$ with $\hat{V}_c^{(i)}$ defined in (5.19), if $x^{(i)} = \hat{V}_c^{(i)}$ fulfills (5.25), since we want to keep the maximal admissible feedrate in the constant phase. Otherwise, considering that $g^{(i)}$ is an increasing function with respect to $x^{(i)}$, we preliminarily check if the following inequality holds,

$$g^{(i)}(\max\{V_{i-1}, V_i\}) \leqslant 2S^{(i)}\,, \qquad (5.27)$$

and, if this is the case, we fix $V_c^{(i)}$ equal to the unique solution of the equation $g^{(i)}(x^{(i)}) = 2S^{(i)}$ in the interval $[\max\{V_{i-1}, V_i\},\ \hat{V}_c^{(i)}]$.

Remark 5 *Observe that the fulfillment of the inequality in (5.27) for any i–th path block, $i = 1, \ldots, M$, has to be preliminarily checked. If this is not the case, we modify the initial feedrate values $\hat{V}_i, i = 1, \ldots, M$ to lower values $V_i, i = 1, \ldots, M$, determined by solving the following constrained minimization problem,*

$$\min \sum_{i=1}^{M-1} (V_i - \hat{V}_i)^2 \textit{ with } \hat{V}_i \geqslant V_i > 0,\ g^{(i)}(\max\{V_{i-1}, V_i\}) - 2S^{(i)} \leqslant 0\,,$$

which can be numerically solved by an iterative minimization procedure, initializing each V_i with $\min\limits_{i=1,\ldots,M} \hat{V}_i$.

5.3.2 *The Strict (S) formulations of the scheduler*

The following formulation have been called Strict (S) since kinematics bounds are imposed along the entire curve causing a consequently increase in terms of time to spend to traverse completely the assigned path. On each segment a set of inner *crossing points* $\mathbf{P}_1, \ldots, \mathbf{P}_{M-1}$ are determinated corresponding to the sign change of the quantity $|\kappa| - \kappa_{cr}$. For notational convenience we use also the points $\mathbf{P}_0$ and $\mathbf{P}_M$ to denote the considered segment endpoints. The curve portion between any pair of successive crossing points of the considered segment is again referred to as a *curve block* and $S^{(i)}$ denotes the length of the i–th curve block, $i = 1, \ldots, M$. For any segment $S^{(i)}$ has to be preliminarily computed[3]. First a safe maximal feedrate is computed for each curve block and then a suitable feedrate values at the crossing points are selected. The maximal feedrate $V_c^{(i)}$ chosen on the i–th block, $i = 1, \ldots M$, is initialized as follows

$$\hat{V}_c^{(i)} := \begin{cases} V_{r0}(\kappa_{max}^{(i)}) & \text{with S0}\,, \\ V_{r1}(\kappa_{max}^{(i)}) & \text{with S1}\,, \\ V_b(\kappa_{max}^{(i)}, w_{max}^{(i)}) & \text{with S2}\,, \end{cases}$$

---

[3] It can be observed that the philosophy for the feedrate determination is somehow reversed with respect to the R strategy.

where $\kappa_{max}^{(i)}$ and $w_{max}^{(i)}$ are the maximum values assumed by $|\kappa|$ and $|w|$ in the considered block. Then, at the crossing point $\mathbf{P}_i, i = 1, \dots, M-1$ we initialize the V_i with $\hat{V}_i$ fixed as follows,

$$\hat{V}_i := \min\{\hat{V}_c^{(i)}, \hat{V}_c^{(i+1)}\}.$$

Since the inequality in (5.27) still has to be fulfilled, Remark 5 again applies. The feedrate inside each curve block is then selected with a trapezoidal profile as described for the R formulations. However in this case in many blocks at least one between the acceleration and deceleration phases disappears.

5.4 APPLICATION TO INTERPOLATORS FOR PH CURVES

As mentioned in the introduction, in this paper we are interested in applying the introduced feedrate scheduler on paths described by PH curves since, among the different parametric curves commonly used in computer aided geometric design, the class of PHs strongly facilitates the design of efficient CNC interpolator algorithms. Thus, to avoid loss of generality, we assume that a given planar curve is preliminarily approximated by one or more than one connected C^2 quintic PH-spline curves interpolating it at suitably chosen points. The algorithm used for this aim is that introduced in [28]. Referring for brevity just to the R formulations, the corresponding whole trajectory planning approach is summarized in Figure 73. In the following subsections we briefly introduce PH interpolator, see [22] for the details.

5.4.1 *PH interpolator*

PH interpolators based on time-dependent feedrate function $v(t)$ are exploited to generate smooth motion trajectories for a PH spline curve $\mathbf{r}(\xi)$. In order to be efficient, they require $v(t)$ for $t \in [0, T]$ to be non-negative and to have a simple expression $F(t)$ for its indefinite integral such that $v(t) = \dot{F}(t)$ and this is actually the case for the feedrate defined by our scheduler. The PH curve interpolator implementation is based on the following relation:

$$s(\xi) = F(t) \tag{5.28}$$

Hence, considering a sampling time Δt and a finite sequence of time values $t_0 = \Delta t$, $t_1 = 2\Delta t, \dots, t_N = N\Delta t$, we can obtain from (5.28) the related sequence of parameter values, $0 = \xi_0 < \xi_1 < \dots, < \xi_N = 1$ as the root the following equation:

$$s(\xi_k) - F(k\Delta t) = 0 \quad \text{for } k = 1 \dots N.$$

Each ξ_k will correspond to a reference point $\mathbf{r}(\xi_k)$ on the PH curve and it will be travelled at the corresponding t_k. Newton-Raphson can be exploited to find the unique real root of the above equation, using as initial guess $\xi_k^{(0)} = \xi_{k-1}$:

$$\xi_k^{r+1} = \xi_k^r - \frac{s(\xi_k^{(r)}) - F(k\Delta t)}{\sigma(\xi_k^{(r)})}, \quad \text{for } r = 0 \ldots \tag{5.29}$$

In particular for the experiments a PH interpolator has been implemented for PH quintic splines accordingly to the class of curves which describe the planar paths. Particular care in the implementation has been necessary whenever the r-th approximation of ξ_k in (5.29) results greater than 1 (that is whenever a change of polynomial segment in the PH spline is predicted for ξ_k). Few iteration of the Newton-Raphson method have been always sufficient to obtain accurate enough approximations of the reference points. In particular in the considered examples only a dozen of iterations are usually enough.

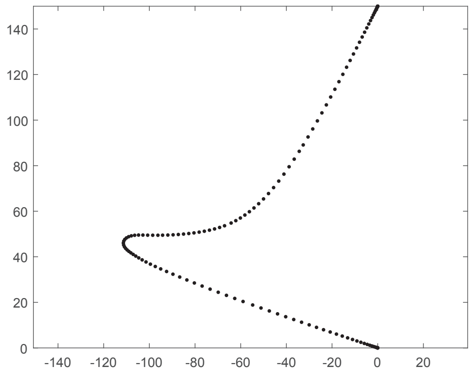


Figure 56. A set of reference points generated on a PH spline by the PH interpolator. Sampling parameter $\Delta t = 0.02$ and feedrate ν generated by the R strategy.

For sake of completeness Figure 56 shows a sequence of reference points generated by the PH-interpolator along a portion of one of the PH spline curves considered for the experiments in the next section. The feedrate is that generated by the R strategy with the parameter setting specified in the first column of Table 6, except the sampling parameter Δt which has been set to 0.02. As expected, the point density increases in correspondence of higher curvature zones.

5.5 NUMERICAL RESULTS

In this section three paths are considered, respectively named PP_2, PF_4 and *gear* to test the feasibility of the proposed feedrate scheduling algorithm. PP_2 and PF_4 are the G^2 PH quintic spline curves resulting from the examples illustrated in Section 3.4.2 (see Figure 30 and 39). On the other hand, the gear curve is directly defined as the composition of 8 C^2 PH splines, each one interpolating an assigned set of points, some of which specified as breakpoints, see Figure 57. For this last example we show the obtained results just on a quarter of the curve, since it is composed by four quarters with symmetric shape. The feedrate scheduler and the PH interpolator are implemented in Matlab and are executed on a personal computer with Intel(R) Core(TM) i5-4300U 1.90 GHz CPU.

For PP_2 and PF_4 both R0 and S2 formulations are reported to show the special points (critical or crossing) selection and to compare the resulting feedrate profiles. The results of all possible configuration of R and S formulations are reported in Table 7, 8 and 9. Beside simulation time, each table illustrates if the imposed bounds related to chordal error (E_c), x-y acceleration (A_x, A_y) and jerk (J_x, J_y) components are satisfied (✓) or not (✗). The feedrate scheduling parameters used for all the executed tests are listed in Table 6, while the parameters p_a and p_j are always both set to $1/\sqrt{2}$ and q_j to $1/2$.

On each PH test curve on the left of Figures 58, 64 and 70 the critical points are marked in red, while in Figures 61 and 67 crossing points are marked in cyan. The related absolute curvature values are marked on the right of the same figures where the curvature profile is reported together with the curvature threshold κ_{cr}, represented as a dashed red line. For all the three considered tests, Figures 59, 65, 62, 68 and 71 show the resulting profiles of $v(t)$, $\dot{v}(t)$ and $\ddot{v}(t)$, while Figure 60, 66, 63, 69 and 72 show Cartesian components of acceleration and jerk. Acceleration, constant and deceleration phases are coloured in green, black and magenta, respectively. The time intervals associated with different curve blocks are highlighted by vertical grey separation lines.

Table 6. Parameter setting.

parameter	**symbols**	PP_2	PF_4	**gear**
Commanded Feedrate (mm/s)	V_m	50	500	250
Maximum acceleration (mm/s^2)	A_m	833	2000	800
Maximum jerk (mm/s^3)	J_m	20825	100000	26000
Maximum chord error (mm)	D	0.001	0.001	0.001
Sampling time (s)	Δt	0.002	0.002	0.002

For each example, in several blocks the maximal feedrate $V_c^{(i)}$ is less than V_m, due to the adaptive behavior of our strategy described in Section 5.3.1, see Figures 59, 65, 62, 68 and 71 (a), (b) and (c). It can be appreciated that the greater is the curvature on the i-th block, more restrictive is the adjusted commanded feedrate. This behavior is highlighted for example on the fourth block of PF_4 curve (see Figure 67 and 68), where the curve presents a high curvature peak and consequently the commanded feedrate drops to 200. The same safety behavior can be observed on the gear curve and in particular on the block corresponding to the top of the quarter of the gear where $V_c^{(i)}$ is set to 125. Concerning Remark 5, we observe that the optimization phase which suitably reduces the initial feedrate values assigned at critical points has been necessary for the gear example. In such case the variation has been as follows:

$$\hat{V} = [0,\ 125.30,\ 166.08,\ 127.02,\ 161.83,\ 148.52,\ 148.52,\ 161.83,\ 127.02,\ 166.08,\ 125.30,\ 0]$$

$$V = [0,\ 118.59,\ 147.24,\ 127.02,\ 140.58,\ 125.35,\ 125.35,\ 140.58,\ 127.02,\ 147.24,\ 118.59,\ 0]\,.$$

Regarding the acceleration control, we observe that for all the considered examples the absolute value of the x-y acceleration components are always below A_m, as shown in the first row of Figures 60, 63, 66, 69 and 72. Furthermore, in all the considered examples the maximum chord error $D = 10^{-3}$ is always fulfilled see Figures 59, 65, 62, 68 and 71 (a). The reference points produced by the PH interpolator with the sampling time Δt reported in Table 6 are shown in Figures 58, 61, 64, 67 and 70 on the right.

For all the three considered tests, Figures 59, 62, 65, 68 and 71 (d) and Figures 60, 63, 66, 69 and 72 show the obtained profiles of $\ddot{v}$ and the x-y jerk components, respectively.

Concerning $\ddot{v}$ it can be observed that the desired control is ensured by our motion in all the cases. Also the x-y jerk components are everywhere controlled for each tests with exception[4] of Figure 60.

Regarding the simulation times, as shown in Table 7, 8 and 9, it can be observed that S strategy always produce slower motion respect R strategy. Furthermore the motion produced by S strategy is more segmented, since the number of crossing points is higher than the number of critical points detected with the R strategy. For sake of brevity the reported numerical results are related to R0 and S2 for PP_2, R0 and S2 for PF_4 and R0 for *gear-quarter*.

Comparing the simulation time with other methods already introduced in the literature (see for example [53, 81]), we may observe that our strategy often produces slightly slower motions since, unlike other methods, it always a priori ensures the desired acceleration control. For a deeper comparison with other feedrate scheduling strategies see [39].

---

[4] It is not surprising that the jerk breaks the threshold since **R0** does not take in account jerk requirements and furthermore it is the relaxed formulation base on the mean of curvature in the considered block.

Table 7. Scheduler features for PP_2 exploiting the R and S strategies.

PP_2 path						
Configuration	Time (s)	E_c	A_x	A_y	J_x	J_y
R0	0.786	✓	✓	✓	✗	✗
R1	0.847	✓	✓	✓	✗	✗
R2	1.039	✓	✓	✓	✗	✗
S0	1.738	✓	✓	✓	✓	✓
S1	1.594	✓	✓	✓	✓	✓
S2	1.594	✓	✓	✓	✓	✓

Table 8. Scheduler features for PF_4 exploiting the R and S strategies.

PF_4 path						
Configuration	Time (s)	E_c	A_x	A_y	J_x	J_y
R0	2.258	✓	✓	✓	✗	✓
R1	2.341	✓	✓	✓	✓	✓
R2	2.362	✓	✓	✓	✓	✓
S0	2.502	✓	✓	✓	✓	✓
S1	2.502	✓	✓	✓	✓	✓
S2	2.678	✓	✓	✓	✓	✓

Table 9. Scheduler features for *gear* exploiting the R and S strategies.

gear path						
Configuration	Time (s)	E_c	A_x	A_y	J_x	J_y
R0	4.723	✓	✓	✓	✓	✓
R1	5.043	✓	✓	✓	✓	✓
R2	5.043	✓	✓	✓	✓	✓
S0	9.437	✓	✓	✓	✓	✓
S1	9.437	✓	✓	✓	✓	✓
S2	9.437	✓	✓	✓	✓	✓

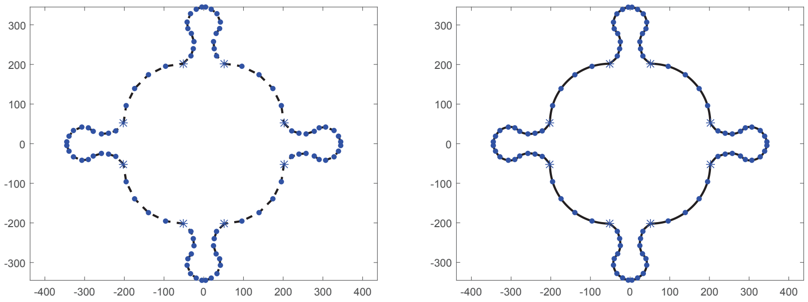


Figure 57. The gear discrete data connected with dashed line (left) and the related G^2 PH quintic spline (right). Interpolation points and breakpoints are depicted respectively as dot and star.

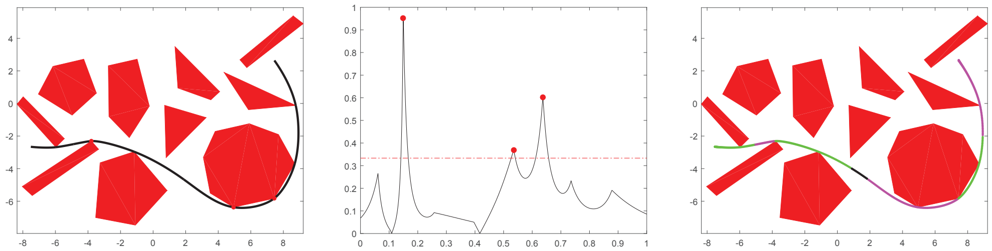
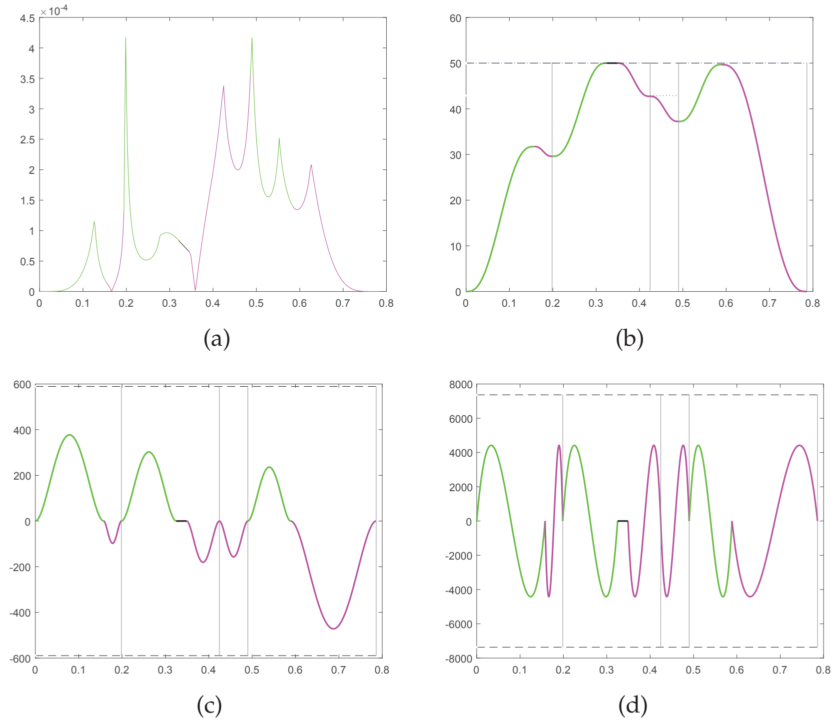


Figure 58. The PP_2 PH spline (left), the related absolute curvature plot with the critical curvature ($3.33 \cdot 10^{-1}$) as red dashed line (center) and the resulting reference points (right). The critical points on the curve and the related peaks on the curvature plot are shown as red dots.



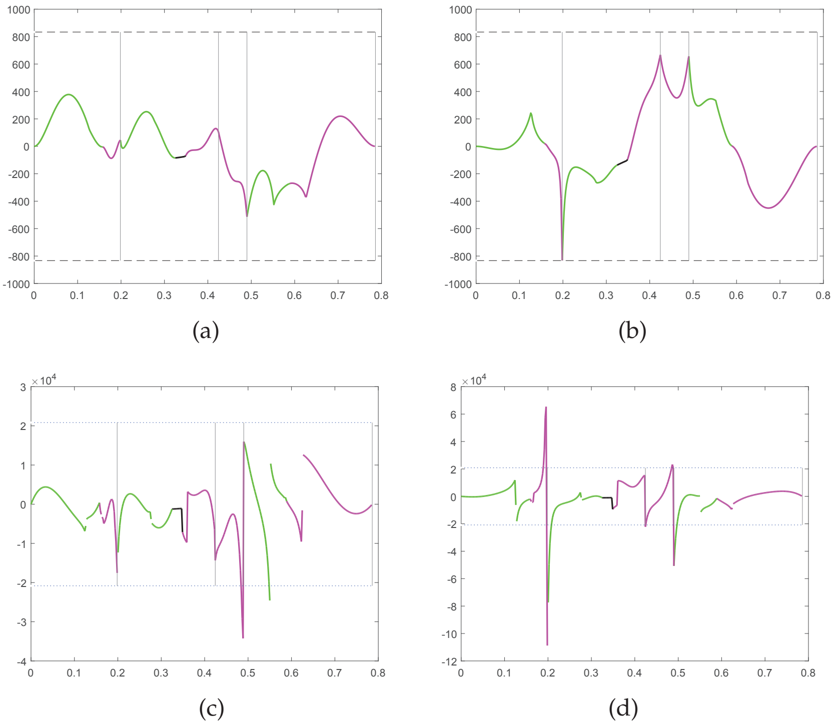
(a)

(b)

(c)

(d)

Figure 59. Feedrate scheduling results for PP_2 curve obtained using **R0** configuration. The chord error (a) and the feedrate $v(t)$ (b) with the related first $\dot{v}(t)$ (c) and second derivative $\ddot{v}(t)$ (d) profiles.



(a) (b)

(c) (d)

Figure 60. Feedrate scheduling results for PP_2 curve obtained using **R0** configuration. The x-y acceleration (a-b) and jerk (c-d) components.

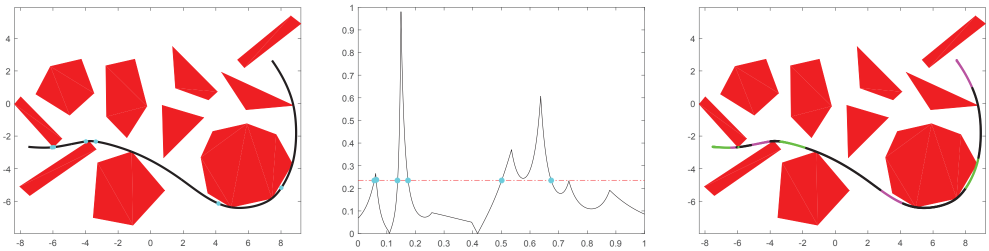
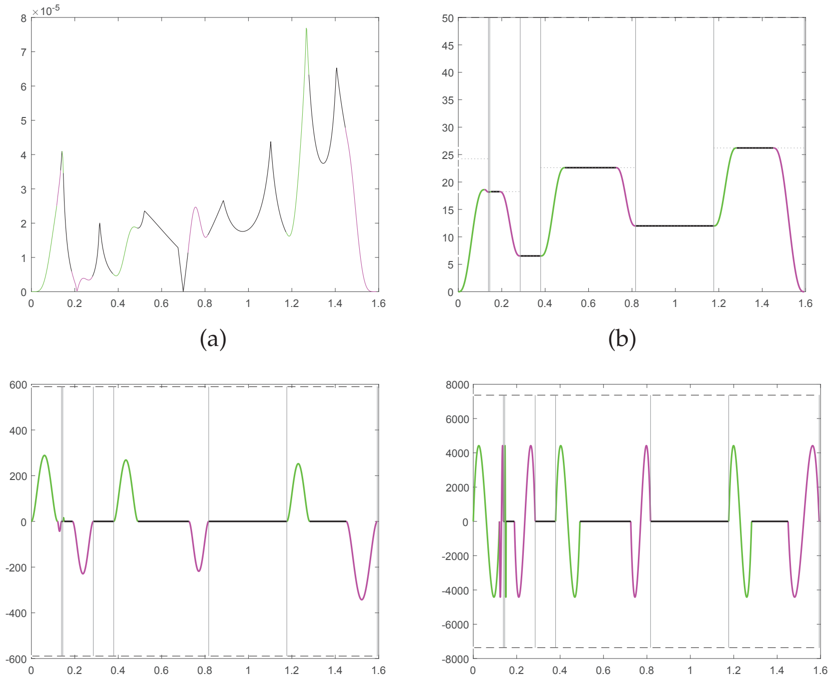


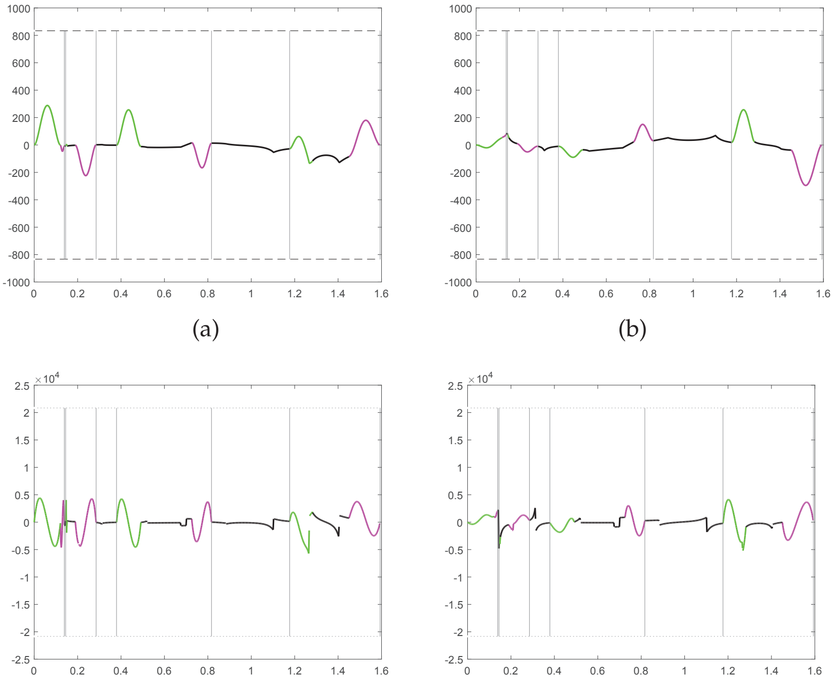
Figure 61. The PP_2 PH spline (left), the related absolute curvature plot with the critical curvature ($2.357 \cdot 10^{-1}$) as red dashed line (center) and the resulting reference points (right). The crossing points on the curve and the related points on the curvature plot are shown as cyan dots.



(a) (b)

(c) (d)

Figure 62. Feedrate scheduling results for PP_2 curve obtained using **S2** configuration. The chord error (a) and the feedrate $v(t)$ (b) with the related first $\dot{v}(t)$ (c) and second derivative $\ddot{v}(t)$ (d) profiles.



(a)

(b)

(c)

(d)

Figure 63. Feedrate scheduling results for PP_2 curve obtained using **S2** configuration. The x-y acceleration (a-b) and jerk (c-d) components.

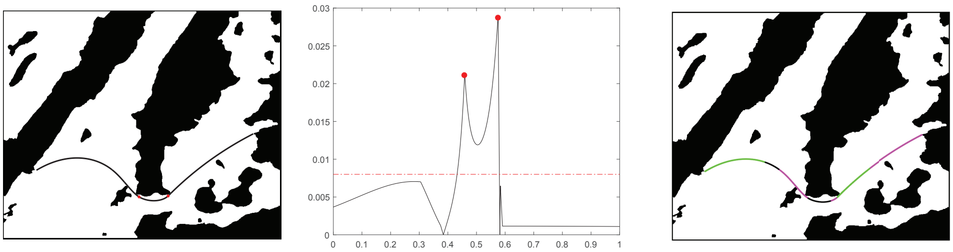
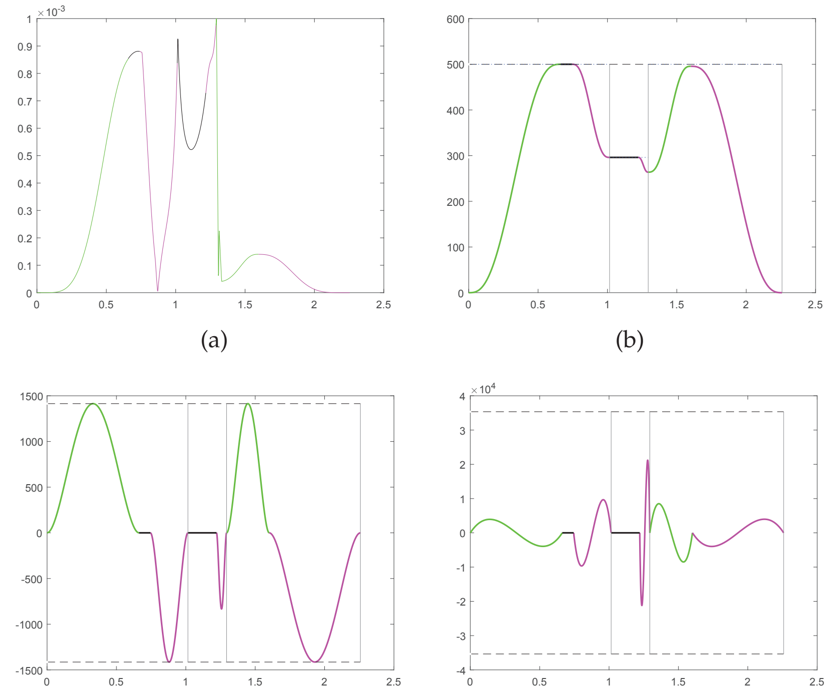


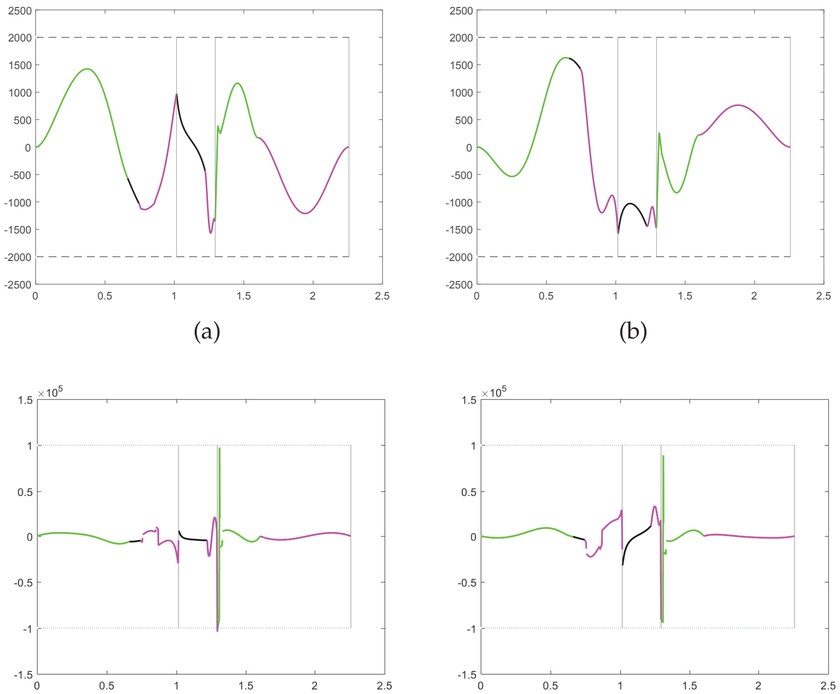
Figure 64. The PF_4 PH spline (left), the related absolute curvature plot with the critical curvature ($8 \cdot 10^{-3}$) as red dashed line (center) and the resulting reference points (right). The critical points on the curve and the related peaks on the curvature plot are shown as red dots.



(a) (b)

(c) (d)

Figure 65. Feedrate scheduling results for PF_4 curve obtained using **R0** configuration. The chord error (a) and the feedrate $v(t)$ (b) with the related first $\dot{v}(t)$ (c) and second derivative $\ddot{v}(t)$ (d) profiles.



(a)

(b)

(c)

(d)

Figure 66. Feedrate scheduling results for PF_4 curve obtained using **R0** configuration. The x-y acceleration (a-b) and jerk (c-d) components.

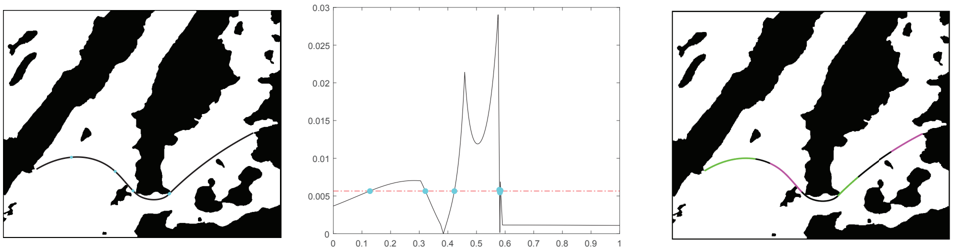
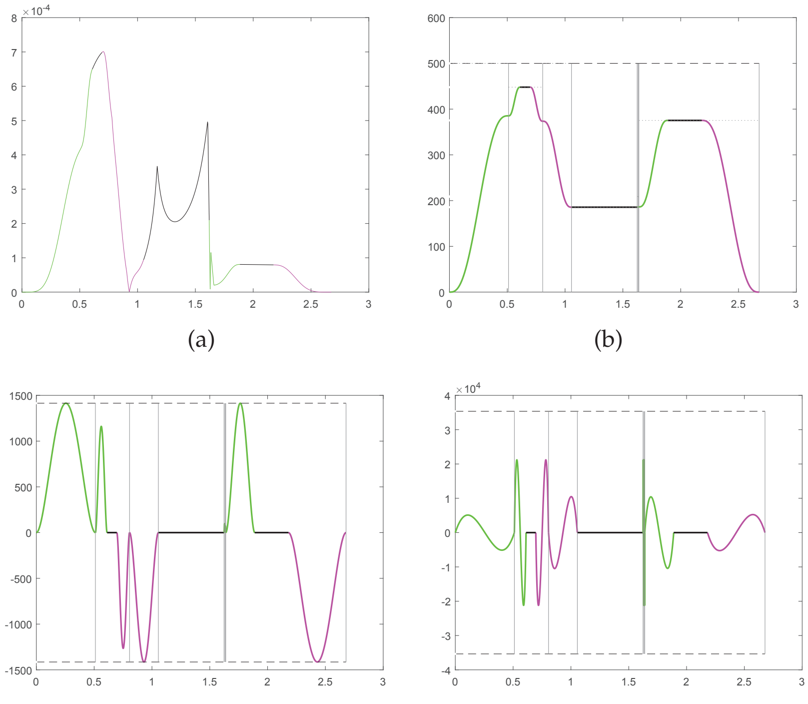


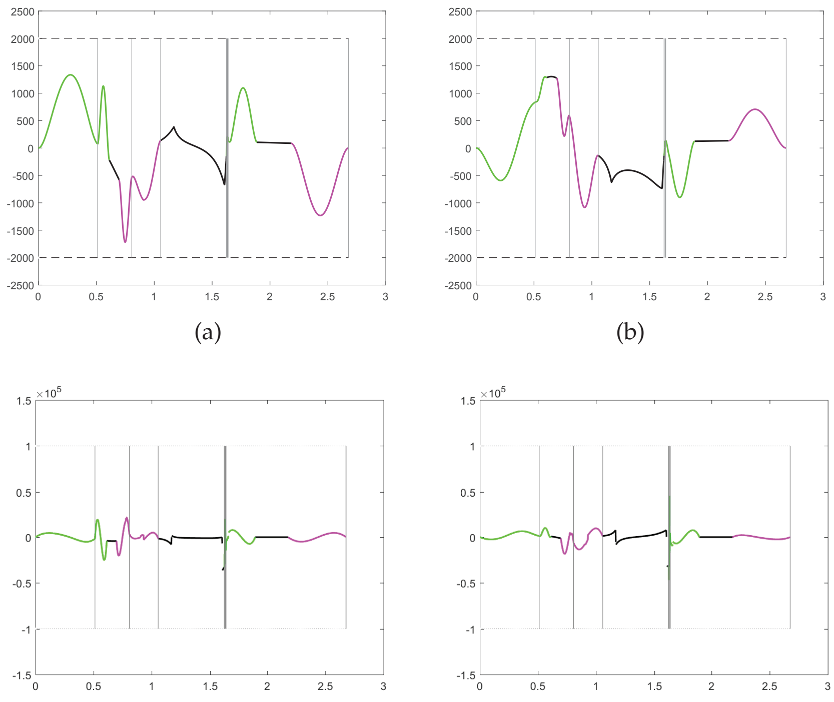
Figure 67. The PF_4 PH spline (left), the related absolute curvature ($5.657 \cdot 10^{-3}$) plot with the critical curvature as red dashed line (center) and the resulting reference points (right). The crossing points on the curve and the points on the curvature plot are shown as cyan dots.



(a) (b)

(c) (d)

Figure 68. Feedrate scheduling results for PF_4 curve obtained using **S2** configuration. The chord error (a) and the feedrate $v(t)$ (b) with the related first $\dot{v}(t)$ (c) and second derivative $\ddot{v}(t)$ (d) profiles.



(a) (b)

(c) (d)

Figure 69. Feedrate scheduling results for PF_4 curve obtained using **S2** configuration. The x-y acceleration (a-b) and jerk (c-d) components.

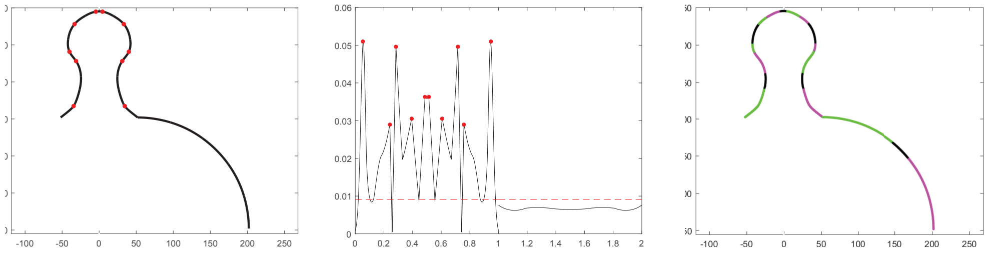
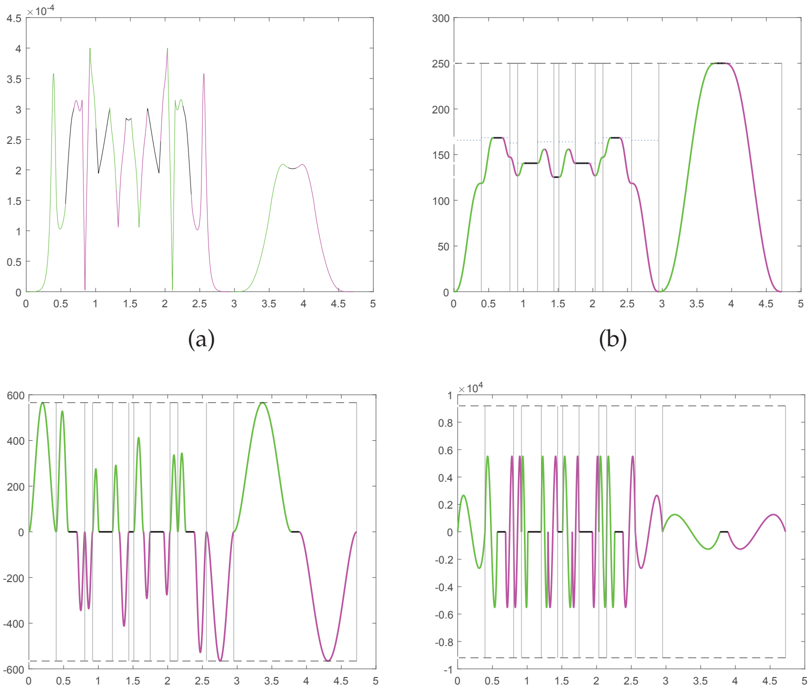


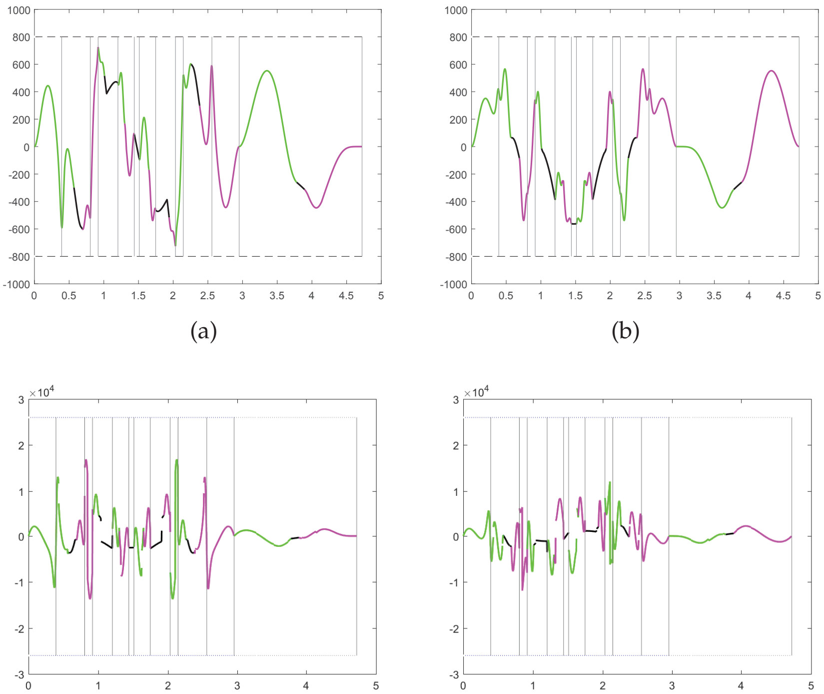
Figure 70. The gear (quarter) PH spline (left), the related absolute curvature plot with the critical curvature as red dashed line (center) and the resulting reference points (right). The critical points on the curve and the related peaks on the curvature plot are shown as red dots.



(a) (b)

(c) (d)

Figure 71. Feedrate scheduling results for gear quarter curve obtained using **R0** configuration. The chord error (a) and the feedrate $v(t)$ (b) with the related first $\dot{v}(t)$ (c) and second derivative $\ddot{v}(t)$ (d) profiles.



(a) (b)

(c) (d)

Figure 72. Feedrate scheduling results for hat curve obtained using **R0** configuration. The x-y acceleration (a-b) and jerk (c-d) components.

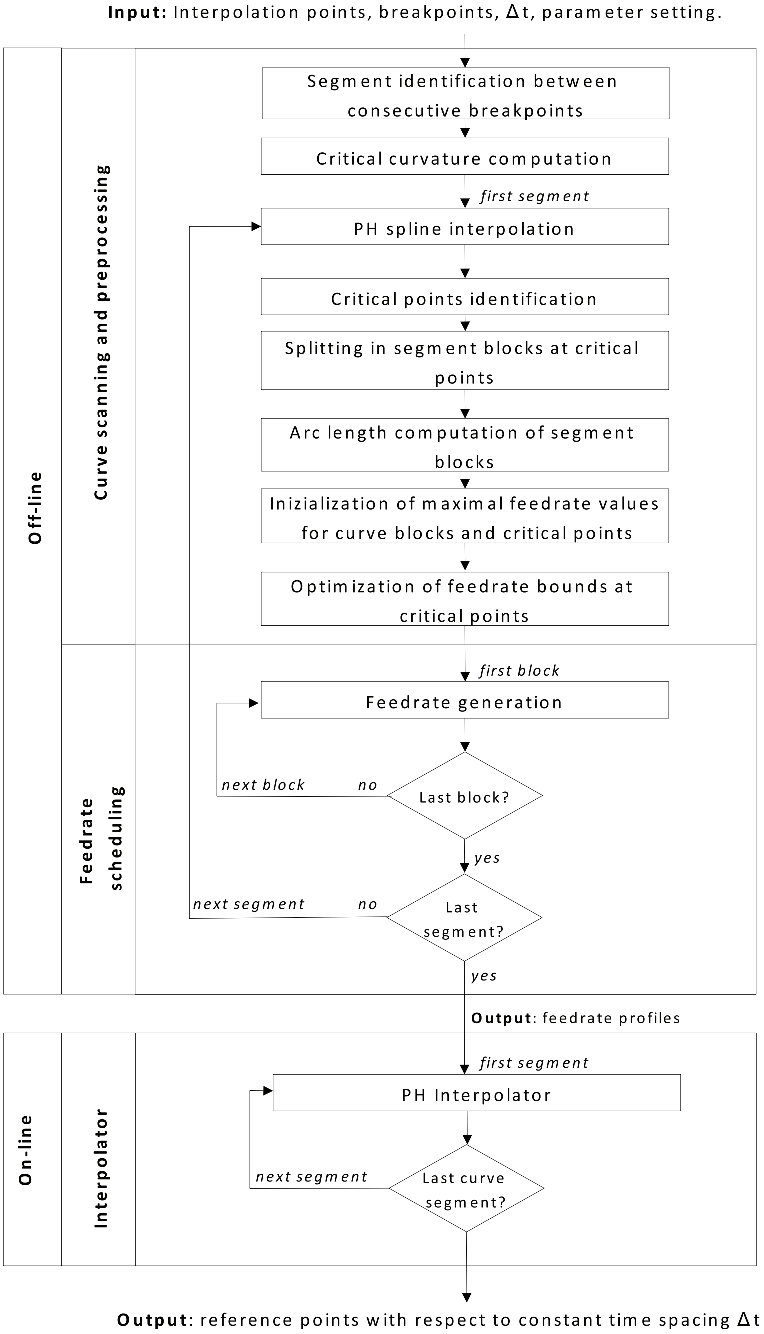


Figure 73. Diagram of the R strategy for feedrate definition.

6

CLOSURE

New methodologies have been developed to enrich with more flexible and accurate paradigms the current state of the art on smooth path and trajectory planning. The aim of smooth path planning is the definition of a reasonable continuous sequence of configurations (path) between an initial and a final one avoiding collisions with a set of static obstacles in a certain environment. Trajectory generation aims at obtaining a time law for the evolution of the configurations. The four main contributions of this work are in the context of planar path and trajectory planning and they are summarized below.

(1) First, algorithms for the construction of a piecewise linear path P_1 that does not intersect any obstacle are discussed and compared. Several variants of the shortest path algorithm applied on the visibility graph are introduced to control the shape (angles and/or length) of the piecewise linear path they produce. This is important because the shape of the final smooth path obtained by interpolation of the vertices of P_1 is highly influenced by the shape of P_1. In order to ensure a better control of the shape of the final path, we associate a suitable dual weighted graph to the primal visibility graph. Dijkstra's algorithm is subsequently applied to this dual configuration. The use of this auxiliary graph allows us to minimize the cumulative angle of the polyline extracted from the visibility graph instead of its length. Furthermore we generalize the possible forms for describing the environment to the path finding context. In this setting, the environment is covered by a set of uniform and non overlapping cells marked as *obstructed* or *free*, by taking into account if they are inside an obstacle or not. This kind of representation allows us to deal with highly complex configurations that contain obstacles not only with polygonal shapes. This strongly increases the versatility of the method. Since the environment can be available only as an image, a picture or a map, we obtain its proper representation by exploiting the effective image segmentation method summarized in Appendix A.1. It might also be worth looking into new navigability structure in order to extend this first contribution of the thesis to the 3D setting. Structures like Rapidly–exploring Random Trees (RRT), Probabilistic Road Maps (PRM) address 3D path planning (see e.g., [52]) and represent a reliable starting point for the development of new strategies.

(2) A new *smooth* path planning is introduced by replacing P_1 with a G^1 or G^2 Pythagorean–hodograph (PH) quintic spline. The set of vertices defining P_1 has been considered as input values for the G^1 and G^2 PH spline interpolation

scheme with tension parameters that produces the final smooth P_2 path. Such schemes are available in the literature and both have a strong point. The first is attractive from the computational point of view, since it is a local scheme, but the second is preferable from the shape point of view, since it produces curvature continuous paths. In order to obtain an efficient application of these schemes to the path planning context, we have extended their asymptotic analysis and consequently developed a technique for the automatic choice of the associated tension parameters. The usage of an interpolation scheme based on PH splines allows us to take full advantage of the key properties of PH curves (fair shapes, polynomial arc length, ...). Also in this case it could be interesting to investigate the spatial counterpart represented by the shape preserving interpolation of spatial data by G^1 PH quintic splines (see e.g., [31]) which rely on the quaternion representation of spatial PH curves (see [22]).

(3) By exploiting the distinctive properties of PH curves, a number of basic utilities are presented in the context of path planning for unmanned or autonomous (air, ground, or undersea) vehicles. Instead of addressing a specific path planning scenario in detail, a family of rigorous, broadly applicable, and easily implemented functions are proposed. The problems treated include: (a) maintenance of minimum safe separations between vehicles that execute different paths at a fixed speed; (b) design of curved paths with identical arc lengths, ensuring simultaneous arrival at fixed speed; (c) path curvature minimization, subject to maximum path length, to ensure compatibility with vehicle steering constraints; and (d) curvature continuous corner rounding of shortest paths through polygonal obstacle fields, consistent with a prescribed maximum path curvature. The procedures can be easily generalized to the three dimensional case using the quaternion representation for spatial PH curves. However other kind of problems in the spatial case deserve attention such as the definition of paths characterized by different constraints e.g. limited climb angle and bounded torsion (see e.g., [67, 25]).

(4) A novel configurable C^2 feedrate scheduler has been introduced. In its more restrictive formulation it ensures full kinematic control. Faster motions are obtained by using its less restrictive formulations, that often still ensure the desired motion control. The developed scheduler has been applied to C^2 PH spline curves to produce acceleration continuous motions. The selection of paths described in terms of PH spline curves is of great interest since it avoids numerical approximations of fundamental geometrical quantities and enhances the accuracy and robustness of the overall scheme. In order to design a jerk continuous scheduler, a C^3 interpolation scheme with PH spline curves (of higher degree) should be investigated. Although the spatial case requires a separate study, also in this case, the procedure should readily extends to the three-dimensional setting.

A

APPENDIX

A.1 IMAGE SEGMENTATION: CONVEX MUMFORD-SHAH VARIANT

A gray scale 8-bit digital image is a matrix with integer entries between 0 (black) and 255 (white), where the gray level of every pixel is given be the numerical value of the corresponding entry in the matrix. The goal of image segmentation is to cluster pixels into salient image regions that correspond to individual surfaces, objects, or natural parts of objects. The classification of pixels into different regions is based on a homogeneity criterion.

The derivation of a mathematical segmentation model requires to see an image as a function $f : \Omega \to R$, with $\Omega \subset R^2$ a bounded open connected set. Without loss of generality, we restrict the range of f to $[0, 1]$. Thus, segmentation can be stated as the process of partitioning the image domain Ω in p regions, $R_1, R_2, \dots, R_p$, such that f has a low variation within each regions and varies rapidly across the boundary compact curve Γ in Ω among different regions R_i, for $i = 1, \dots, p$. One of the most important image segmentation model is the Mumford–Shah model [61] which approximates f by minimizing the following functional,

$$\begin{aligned} E_{MS}(g, \Gamma) = \ & \tfrac{\lambda}{2} \textstyle\int_{\Omega} (f - g)^2 dx + \\ & \tfrac{\mu}{2} \textstyle\int_{\Omega \setminus \Gamma} |\nabla g|^2 dx + \mathrm{Len}(\Gamma)\,, \end{aligned} \tag{A.1}$$

where λ and μ are positive parameters, $\mathrm{Len}(\Gamma)$ is the length of Γ, and $g : \Omega \to \mathbb{R}$ is continuous or differentiable in $\Omega \setminus \Gamma$, but may be discontinuous across Γ. The first term in (A.1) requires that g approximates f, while the second one asks that g does not vary much on each R_i. Since (A.1) is non convex, it is very challenging to find or approximate its minimizer. Several convex approximations or simplifications of the Mumford–Shah energy (A.1) have been proposed in the literature, see, e.g., [12] and reference therein. Here we apply a two–stage image segmentation method recently proposed in [11]. The first stage is aimed to find a smooth image g using a convex variant of (A.1). The second stage is a thresholding of g to obtain the final segmentation. The smoothing phase is justified by the following observation: a binary image can be subsequently recovered from its smoothed version by a suitable thresholding technique. The basic thresholding approach associates two different ranges of gray levels to the image foreground and background. A value of gray level (also called threshold) of separation between the two ranges is calculated in order to assign the value 1 or 0 to the pixels of the two ranges. Here we are interested in minimizing in $W^{1,2}(\Gamma)$ the following variant of the Mumford–Shah functional, recently introduced in [11],

$$E(g) = \frac{\lambda}{2}\int_{\Omega}(f-g)^2 dx + \frac{\mu}{2}\int_{\Omega}|\nabla g|^2 dx + \int_{\Omega}|\nabla g| dx$$

which has a unique minimizer [11, Theorem 2.4] and whose discrete version is

$$\min_{g}\left\{\frac{\lambda}{2}\|f-g\|_2^2 + \frac{\mu}{2}\|\nabla g\|_2^2 + \|\nabla g\|_1\right\}\ , \tag{A.2}$$

where now f is a vector obtained by stacking the columns of the image and whose i–th entry contains the grey level of the corresponding pixel. Accordingly, g and ∇g are vectors with the same number of entries, where $(\nabla g)_i = \sqrt{(\nabla_x g)_i^2 + (\nabla_y g)_i^2}$, with ∇_x and ∇_y defined as the x and y backward difference operators (with periodic boundary conditions). Then, by setting $d_x = \nabla_x g$ and $d_y = \nabla_y g$, (A.2) can be reformulated as the following constrained problem,

$$\begin{aligned}&\min_g \left\{\tfrac{\lambda}{2}\|f-g\|_2^2 + \tfrac{\mu}{2}\|\nabla g\|_2^2 + \|(d_x, d_y)\|_1\right\} \text{ s.t.}\\ &d_x = \nabla_x g\,, \quad d_y = \nabla_y g\,.\end{aligned}$$

By applying the Split Bregman method [42] which uses an additional positive constant ρ to enforce the above constraints, we have at step $k+1$

$$\begin{aligned}(g^{k+1}, d_x^{k+1}, d_y^{k+1}) = \ & \arg\min_{g,d_x,d_y}\left\{\tfrac{\lambda}{2}\|f-g\|_2^2 + \right.\\ & \tfrac{\mu}{2}\|\nabla g\|_2^2 + \|(d_x, d_y)\|_1 + \\ & \tfrac{\rho}{2}\|d_x - \nabla_x g - b_x^k\|_2^2 + \\ & \left.\tfrac{\rho}{2}\|d_y - \nabla_y g - b_y^k\|_2^2\ \right\}\,,\end{aligned} \tag{A.3}$$

and

$$\begin{aligned}b_x^{k+1} &= b_x^k + (\nabla_x g^{k+1} - d_x^{k+1}),\\ b_y^{k+1} &= b_y^k + (\nabla_y g^{k+1} - d_y^{k+1})\,.\end{aligned} \tag{A.4}$$

The initial guesses of every variable are fixed as $g^0 = f$, $d_x^0 = d_y^0 = b_x^0 = b_y^0 = 0$. The minimization problem (A.3) can be splitted in two minimization sub–problems in g^{k+1} and (d_x^{k+1}, d_y^{k+1}), which can be easily computed by alternating minimization. The solution of the sub–problem in g^{k+1} satisfies the system of linear equations

$$\begin{aligned}(\lambda I - (\mu+\rho)\Delta)\, g^{k+1} \ = \ \lambda f \ & + \rho\nabla_x^T(d_x^k - b_x^k)\\ & + \rho\nabla_y^T(d_y^k - b_y^k),\end{aligned} \tag{A.5}$$

where $\Delta = -(\nabla_x^T \nabla_x + \nabla_y^T \nabla_y)$ and hence the coefficient matrix is positive definite. The sub–problem in (d_x^{k+1}, d_y^{k+1}) has the explicit solution

$$\begin{aligned} d_x^{k+1} &= \max\left(s^k - \frac{1}{\rho}, 0\right) \frac{s_x^k}{s^k}, \\ d_y^{k+1} &= \max\left(s^k - \frac{1}{\rho}, 0\right) \frac{s_y^k}{s^k}, \end{aligned} \tag{A.6}$$

where $s_x^k = \nabla_x g^{k+1} + b_x^k$, $s_y^k = \nabla_y g^{k+1} + b_y^k$, and $s_k = ((s_x^k)^2 + (s_y^k)^2)^{1/2}$, the arithmetic operations in (A.6) are intended componentwise. Summarizing, the solution of (A.2) can be iteratively approximated by solving the linear system (A.5) and updating (d_x^{k+1}, d_y^{k+1}) and (b_x^{k+1}, b_y^{k+1}) by (A.6) and (A.4), respectively, see [11, Algorithm 1].

Once the smoothed figure is computed, the p–means clustering method is used to achieve the desired segmentation by thresholding [57]. This is one of the easiest ways to classify a given set in p clusters, $R_1, \dots, R_p$, where p is fixed a priori. The idea is to define the centroid for each cluster. The purpose of the algorithm consists in minimizing the objective function

$$J = \sum_{j=1}^{p} \sum_{x \in R_j} \|x - c_j\|^2 ,$$

where $\|x - c_j\|^2$ is a distance measure between x points and the cluster center c_j. Figure 74 shows the results of the two–stage image segmentation method applied to a sample image.[1] The intermediate result after the smoothing phase is shown together with the final segmentation.

Figure 74. The two–stage image segmentation method applied to a sample image (top). The intermediate result after the smoothing phase is shown (center) together with the final segmentation obtained with $p = 3$ (bottom).

---

[1] Source: `https://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/BSDS300/html/dataset/images/color/test-051-075.html`

A.2 GRAPH SEARCH ALGORITHMS

Algorithm 1.
`modTMSP := MTMSP(obstacles, nodes, edges, start, goal, maxDist, cluDist)`

```
Input:
obstacles: list of polygonal obstacles in the environment;
nodes: list of nodes of the trapezoidal map;
edges: list of edges of the trapezoidal map;
start: start position $\mathbf{q}_s$;
goal: goal position $\mathbf{q}_g$;
maxDist: threshold value for collinear check;
cluDist: threshold value for cluster check;
Output:
modTMSP: node list of the modified trapezoidal map shortest path.

define the two edges $e_s$, $e_g$ connecting start and goal to the nearest nodes
  in nodes;
add $e_s$ and $e_g$ to edges;
shortest := SP(nodes, edges, start, goal);
modTMSP := VisibilityCheck(shortest, obstacles);
modTMSP := CollinearCheck(modTMSP, obstacles, maxDist);
modTMSP := ClusterCheck(modTMSP, obstacles, cluDist);
```

ALGORITHM 2.

`modVGSP` := MVGSP(`nodes, edges, start, goal, tol`)

Input:

`nodes`: list of visibility graph nodes;
`edges`: list of visibility graph edges;
`start`: visibility graph node corresponding to the start position $\mathbf{q}_s$;
`goal`: visibility graph node corresponding to the goal position $\mathbf{q}_g$;
`tol`: threshold for angles.

Output:

`modVGSP`: node list of the modified visibility graph shortest path.

```
assign to modVGSP the empty list;
shortest := SP(nodes, edges, start, goal);
sp_is_ok := false;
while not(sp_is_ok) do
    sp_is_ok := true;
    add to modVGSP the first node of shortest;
    for i from 2 to length(shortest)-1 do
        P := shortest[i-1]; C := shortest[i]; S := shortest[i+1];
        assign to C.angle the angle between $\overline{PC}$ and $\overline{CS}$;
        if C.angle $>$ tol and $\exists$ an edge that defines an angle $\leqslant$ tol with $\overline{PC}$
         then
            sp_is_ok := false;
            assign to da all edges from the initial node in shortest to S;
            remove da from edges;
            sp_temp := SP(nodes, edges, C, goal);
            T := sp_temp[2];
            assign to angle_temp the angle between $\overline{PC}$ and $\overline{CT}$;
            if angle_temp $<$ C.angle then
                shortest := sp_temp; sp_is_ok := false; break;
            else
                add the node C to modVGSP and restore da in edges;
            end
        else
            add the node C to modVGSP;
        end
    end
    add the last node of shortest to modVGSP;
end
```

Algorithm 3.

`visited` := GVGP(`nodes, edges, visited, goal, tol`)

```
Input:
nodes: list of visibility graph nodes;
edges: list of visibility graph edges;
visited: list of visited nodes;
goal: visibility graph node corresponding to the goal position $\mathbf{q}_g$;
tol: threshold for angles.
Output:
visited: list of visited nodes.

initialize neighbours with the empty list;
initialize C with the last node of visited;
initialize P with the second last node of visited;
for each edge $\in$ edges outgoing from C and different from $\overline{PC}$ do
    assign to S the other extreme of edge;
    assign to S.angle the angle between $\overline{PC}$ and $\overline{CS}$;
    add S to neighbours;
end
sort the list neighbours according to an increasing value of S.angle;
if goal $\in$ neighbours then
    add goal to visited;
    return;
else
    for each node $\in$ neighbours do
        if node $\notin$ visited and node.angle $<$ tol then
            add node to visited;
            visited := GVGP(nodes, edges, visited, goal, tol);
            if goal $\in$ visited then
                return;
            else
                delete node from visited;
            end
        end
    end
end
```

Algorithm 4.
`refinedPath := RefinePath(path, grid, maxDist)`

Input:
`grid` (adjacency matrix)
`path` (node list of free cells in `grid`)
`maxDist` (distance threshold)

```
prevSize := number of nodes that compose path;
refinedPath := CollinearElimination(path);
currSize := number of nodes that compose refinedPath;
while prevSize not equal to currSize do
    refinedPath := StringPulling(refinedPath, grid);
    refinedPath := CollinearElimination(refinedPath);
    prevSize := currSize;
    currSize := number of nodes that compose refinedPath;
end
refinedPath := JoinVertices(refinedPath, maxDist);
```

Output:
`refinedPath` (node list of free cells in `grid`).

LIST OF FIGURES

BIBLIOGRAPHY

[1] G. Albrecht and R. T. Farouki. Construction of C^2 Pythagorean–hodograph interpolating splines by the homotopy method. *Adv. Comp. Math.* , 5:417–442, 1996. (Cited on pages 8 and 47.)

[2] G. Ambrosino, M. Ariola, U. Ciniglio, F. Corraro, E. D. Lellis, and A. Pironti. Path generation and tracking in 3-d for uavs. *IEEE Trans. Contr. Sys. Techn.*, 17(4):980–988, 2009. (Cited on page 71.)

[3] M. Annoni, A. Bardine, S. Campanelli, P. Foglia, and C. A. Prete. A real-time configurable NURBS interpolator with bounded acceleration, jerk and chord error. *Comput. Aided Design* , 44:509–521, 2012. (Cited on pages 9, 82, and 85.)

[4] S. Asaturyan, P. Costantini, and C. Manni. G^2 shape-preserving parametric planar curve interpolation. In H. Nowacki and P. D. Kaklis, editors, *Creating Fair and Shape-preserving Curves and Surfaces*, pages 89–98. Stuttgart: Teubner, 1998. (Cited on pages 8, 29, 33, and 34.)

[5] A. Askari, M. Mortazavi, H. A. Talebi, , and A. Motamedi. A new approach in UAV path planning using Bézier–Dubins continuous curvature path. *J. Aerosp. Eng.* , 230:1103–1113, 2016. (Cited on pages 9 and 65.)

[6] M. d. Berg, O. Cheong, M. v. Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag TELOS, Santa Clara, CA, USA, 3rd ed. edition, 2008. (Cited on pages 8, 15, 16, and 18.)

[7] G. Birkhoff and C. de Boor. Piecewise polynomial interpolation and approximation, in *A*pproximation of Functions. *(H. L. Garabedian, ed.), Elsevier, Amsterdam*, pages 164–190, 1965. (Cited on page 72.)

[8] P. Bosetti and E. Bertolazzi. Feed-rate and trajectory optimization for CNC machine tools. *Robot Cim-Int. Manuf.*, 30:667–677, 2014. (Cited on page 9.)

[9] A. Botea, B. Bouzy, M. Buro, C. Bauckhage, and D. S. Nau. Pathfinding in games. In S. M. Lucas, M. Mateas, M. Preuss, P. Spronck, and J. Togelius, editors, *Artificial and Computational Intelligence in Games*, volume 6 of *Dagstuhl Follow-Ups*, pages 21–31. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013. (Cited on pages 16 and 25.)

[10] H. Bruyninckx and D. Reynaerts. Path planning for mobile and hyper–redundant robots using Pythagorean–hodograph curves. *Proceedings, In-*

ternational Conference on Advanced Robotics , pages 595–600, 1997. (Cited on pages 65 and 71.)

[11] X. Cai, R. Chan, and T. Zeng. A Two–Stage Image Segmentation Method Using a Convex Variant of the Mumford–Shah Model and Thresholding. *SIAM J. Imaging Sciences*, 6:368–390, 2013. (Cited on pages 7, 113, 114, and 115.)

[12] T. Chan, S. Esedoglu, and M. Nikolova. Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM J. Appl. Math.*, 66:1632–1648, 2006. (Cited on page 113.)

[13] R. Choe, V. Cichella, E. Xargay, N. Hovakimyan, A. C. Trujillo, and I. Kaminer. A trajectory–generation framework for time–critical cooperative missions. *Proceedings, AIAA Infotech Aerospace Conference*, 2013. (Cited on pages 65 and 68.)

[14] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, Cambridge, MA, June 2005. (Cited on pages 7, 11, 13, 14, and 15.)

[15] K. Chu, J. Kim, K. Jo, and M. Sunwoo. Real–time path planning of autonomous vehicles for unstructured road navigation. *Int. J. Automot. Technol.* , pages 653–688, 2015. (Cited on page 65.)

[16] D. de A. Fernandes, A. J. Søensen, and D. C. Donha. Path generation for high–performance motion of ROVs based on a reference model. *Model. Ident. Control* , 36:81–101, 2013. (Cited on pages 9 and 65.)

[17] E. Dijkstra. A note on two problems in connexion with graphs. *Numer. Math.* , 1:269–271, 1959. (Cited on page 18.)

[18] M. Donatelli, C. Giannelli, D. Mugnaini, and A. Sestini. Curvature continuous path planning and path finding based on PH splines with tension. *Comput. Aided Design* , 88:14–30, 2017. (Cited on page 77.)

[19] C. A. Ernesto and R. T. Farouki. Solution of inverse dynamics problems for contour error minimization in CNC machines. *Int. J. Adv. Manuf. Technol.* , 49:589–604, 2010. (Cited on page 65.)

[20] R. T. Farouki. The conformal map $z \to z^2$ of the hodograph plane. *Comput. Aided Geom. Design* , 11:363–390, 1994. (Cited on pages 31, 66, and 75.)

[21] R. T. Farouki. The elastic bending energy of Pythagorean–hodograph curves. *Comput. Aided Geom. Design* , 13:227–241, 1996. (Cited on page 30.)

[22] R. T. Farouki. *Pythagorean-Hodograph curves: algebra and geometry inseparable*. Springer, Berlin, 2008. (Cited on pages 8, 10, 29, 55, 65, 69, 83, 93, and 112.)

[23] R. T. Farouki. Construction of G^2 rounded corners with Pythagorean-hodograph curves. *Comput. Aided Geom. Design* , 31:127–139, 2014. (Cited on page 66.)

[24] R. T. Farouki. Construction of G^1 planar Hermite interpolants with prescribed arc lengths. *Comput. Aided Geom. Design* , 46:64–75, 2016. (Cited on pages 66, 68, and 69.)

[25] R. T. Farouki. Helical polynomial curves interpolation G^1 data with prescribed axes and pitch angles. *Comput. Aided Geom. Design* , 56:4–15, 2017. (Cited on pages 9 and 112.)

[26] R. T. Farouki, C. Giannelli, C. Manni, and A. Sestini. Identification of spatial PH quintic Hermite interpolants with near–optimal shape measures. *Comput. Aided Geom. Design* , 25:274–297, 2008. (Cited on pages 8 and 71.)

[27] R. T. Farouki, C. Giannelli, D. Mugnaini, and A. Sestini. Path planning with Pythagorean-hodograph curves for unmanned or autonomous vehicles. *J. Aerosp. Eng.* , to appear. (Cited on page 9.)

[28] R. T. Farouki, B. K. Kuspa, C. Manni, and A. Sestini. Efficient solution of the complex quadratic tridiagonal system for C^2 PH quintic splines. *Numer. Algor.* , 27:35–60, 2001. (Cited on pages 8, 34, 66, 77, 83, and 93.)

[29] R. T. Farouki, J. Manjunathaiah, D. Nicholas, G.-F. Yuan, and S. Jee. Variable–feedrate CNC interpolators for constant material removal rates along Pythagorean–hodograph curves. *Comput. Aided Design* , 30:631–640, 2008. (Cited on pages 10 and 65.)

[30] R. T. Farouki, J. Manjunathaiah, and G.-F. Yuan. G codes for the specification of Pythagorean–hodograph tool paths and associated feedrate functions on open–architecture CNC machines. *Inter. J. Mach. Tools Manuf.* , 39:123–142, 1999. (Cited on pages 10, 65, 68, and 83.)

[31] R. T. Farouki, C. Manni, M. L. Sampoli, and A. Sestini. Shape-preserving interpolation of spatial data by Pythagorean-hodograph quintic spline curves. *IMA J. Numer. Anal.*, 35:478–498, 2015. (Cited on page 112.)

[32] R. T. Farouki, C. Manni, and A. Sestini. Shape–preserving interpolation by G^1 and G^2 PH quintic splines. *IMA J. Numer. Anal.*, 23:175–195, 2003. (Cited on pages 8, 29, 32, 45, 46, 47, and 83.)

[33] R. T. Farouki, C. Manni, and A. Sestini. Spatial C^2 PH quintic splines. In T. Lyche, M.-L. Mazure, and L. L. Schumaker, editors, *Curve and Surface*

Design: Saint Malo 2002, pages 147–156. Nashboro Press, 2003. (Cited on page 8.)

[34] R. T. Farouki and C. A. Neff. Hermite interpolation by Pythagorean-hodograph quintics. *Math. Comp.* , **64**:1589–1609, 1995. (Cited on pages 8, 32, 33, and 66.)

[35] R. T. Farouki and K. M. Nittler. Efficient high–speed cornering motions based on continuously–variable feedrates I. Real–time interpolator algorithms. *Int. J. Adv. Manuf. Technol.* , 87:3557–3568, 2016. (Cited on pages 10, 74, 75, and 82.)

[36] R. T. Farouki and T. Sakkalis. Pythagorean hodographs. *IBM J. Res. Develop.* , 34:736–752, 1990. (Cited on pages 29, 65, and 68.)

[37] R. T. Farouki and S. Shah. Real–time CNC interpolators for Pythagorean–hodograph curves. *Comput. Aided Geom. Design* , 13:583–600, 1996. (Cited on pages 9, 65, 68, and 82.)

[38] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni. Trajectory planning in robotics. *Math. in Comp. Sciences*, 6:269–279, 2012. (Cited on page 81.)

[39] C. Giannelli, D. Mugnaini, and A. Sestini. C^2 continuous time–dependent feedrate scheduling with configurable kinematic constraints. *in preparation*. (Cited on pages 83 and 96.)

[40] C. Giannelli, D. Mugnaini, and A. Sestini. Path planning with obstacle avoidance by G^1 PH quintic splines. *Comput. Aided Design* , 75–76:47–60, 2016. (Cited on pages 54, 74, and 77.)

[41] C. Goerzen, Z. Kong, and B. Mettler. A survey of motion planning algorithms from the perspective of autonomous UAV guidance. *J. Intell. Robot Syst.* , 57:65–100, 2010. (Cited on pages 7 and 14.)

[42] T. Goldstein and S. Osher. The Split Bregman Method for L1–Regularized Problems. *SIAM J. Imaging Sciences*, 2:323–343, 2009. (Cited on page 114.)

[43] R. C. Gonzalez and R. E. Woods. *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., 2006. (Cited on page 16.)

[44] P. Hart, N. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE T. Syst. Sci. Cyb.*, SSC-4(2):100–107, 1968. (Cited on page 25.)

[45] J. Hoschek and D. Lasser. *Fundamentals of Computer Aided Geometric Design*. A. K. Peters, Ltd., Natick, MA, USA, 1993. Translator–L. L. Schumaker. (Cited on page 29.)

[46] B. Jüttler. Hermite interpolation by Pythagorean hodograph curves of degree seven. *Math. Comp.* , 70:1089–1111, 2001. (Cited on page 8.)

[47] K. Kendig. *Algebraic Geometry*. Springer–Verlag, New York, 1977. (Cited on page 42.)

[48] T. Kito, J. Ota, R. Katsuki, T. Mizuta, T. Arai, T. Ueyama, and T. Nishiyama. Smooth path planning by using visibility graph–like method. In *Proc. of the 2003 IEEE Int. Conf. on Robotics and Automation, Taipei, Taiwan*, pages 3770–3775, 2003. (Cited on page 8.)

[49] K. K. Kubota. Pythagorean triples in unique factorization domains. *Amer. Math. Monthly*, 79:503–505, 1972. (Cited on page 30.)

[50] J. Lai, K. Lin, S. Tseng, and W. Ueng. On the development of a parametric interpolator with confined chord error, feedrate, acceleration and jerk. *Int. J. Adv. Manuf. Technol.* , 37:104–121, 2008. (Cited on page 9.)

[51] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, New York, 1991. (Cited on page 8.)

[52] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Boston, MA, 2006. (Cited on pages 7 and 111.)

[53] A. C. Lee, M. T. Lin, Y. R. Pan, and W. Y. Lin. The feedrate scheduling of NURBS interpolator for CNC machine tools. *Comput. Aided Design* , 43:612–628, 2011. (Cited on pages 9, 82, 89, and 96.)

[54] Z. Li, D. S. Meek, and D. J. Walton. A smooth, obstacle–avoiding curve. *Comput. Graphics*, 30:581–587, 2006. (Cited on page 9.)

[55] M. Lin, M. Tsai, and H. Yau. Development of a dynamics-based NURBS interpolator with real-time look-ahead algorithm. *Inter. J. Mach. Tools Manuf.* , 47:2246–2262, 2007. (Cited on page 9.)

[56] D. G. Macharet, A. A. Neto, and M. F. M. Campos. On the generation of feasible paths for aerial robots in environments with obstacles. In *Proceedings, 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3380–3385, Saint Louis, MO, 2009. (Cited on pages 9, 65, and 74.)

[57] J. Macqueen. Some methods for classification and analysis of multivariate observations. In *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967. (Cited on page 115.)

[58] T. Maekawa, T. Noda, S. Tamura, T. Ozaki, and K. Machida. Curvature continuous path generation for autonomous vehicle using B–spline curves. *Comput. Aided Design* , 42:350–359, 2010. (Cited on page 9.)

[59] E. Magid, D. Keren, E. Rivlin, and I. Yavneh. Spline–based robot navigation. In *Proc. of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China*, pages 2296–2301, 2006. (Cited on page 9.)

[60] S. Martina. Metodi B-spline per il disegno di percorsi regolari in ambienti tridimensionali contenenti ostacoli. U*niversità degli Studi di Firenze, Tesi Magistrale*, 2017. (Cited on page 9.)

[61] D. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Commun. Pur. App. Math.*, 42:577–685, 1989. (Cited on page 113.)

[62] A. A. Neto and M. F. M. Campos. A path planning algorithm for UAVs with limited climb angle. In *Proceedings, 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3894–3899, Saint Louis, MO, 2009. (Cited on page 65.)

[63] A. A. Neto and M. F. M. Campos. On the generation of feasible paths for aerial robots with limited climb angle. In *Proceedings, 2009 IEEE International Conference on Robotics and Automation*, pages 2872–2877, Kobe, Japan, 2009. (Cited on page 65.)

[64] A. A. Neto, D. G. Macharet, and M. F. M. Campos. Feasible RRT–based path planning using seventh order Bézier curves. In *Proceedings, 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1445–1450, Taipei, Taiwan, 2010. (Cited on page 65.)

[65] A. A. Neto, D. G. Macharet, and M. F. M. Campos. On the generation of trajectories for multiple UAVs in environments with obstacles. *J. Intell. Robot Syst.* , 57:123–141, 2010. (Cited on pages 9, 65, 71, and 74.)

[66] A. A. Neto, D. G. Macharet, and M. F. M. Campos. Feasible path planning for fixed–wing UAVs using seventh order Bézier curves. *J. Braz. Comput. Soc.* , 19:193–203, 2013. (Cited on page 65.)

[67] A. A. Neto, D. G. Macharet, and M. F. M. Campos. 3D path planning with continuous bounded curvature and pitch angle profiles using 7th order curves. In *Proceedings, 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4923–4928, Hamburg, Germany, 2015. (Cited on pages 9, 65, and 112.)

[68] G. Nielson. Some piecewise polynomial alternative to spline under tension. In R. E. Barnhill and R. F. Riesenfeld, editors, *Computer Aided Geometric Design*, pages 209–235. Academic Press, 1974. (Cited on pages 46 and 54.)

[69] S. Rabin. A* aesthetic optimizations. In M. DeLoura, editor, *Game Programming Gems*, pages 272–287. Charles River Media, 2000. (Cited on page 25.)

[70] A. Sestini, L. Landolfi, and C. Manni. On the approximation order of a space data–dependent PH quintic Hermite interpolation scheme. *Comput. Aided Geom. Design* , 30:148–158, 2013. (Cited on page 8.)

[71] M. Shanmugavel, A. Tsourdos, and B. A. White. Collision avoidance and path planning of multiple UAVs using flyable paths in 3D. In *Proceedings, 15th International Conference on Methods and Models in Automation and Robotics*, pages 218–222, Miedzyzdroje, Poland, 2010. (Cited on pages 9, 65, and 74.)

[72] M. Shanmugavel, A. Tsourdos, B. A. White, and R. Zbikowski. Differential geometric path plannings of multiple UAVs. *ASME J. Dyn. Syst. Meas. Control* , 129:620–632, 2007. (Cited on pages 65, 68, and 71.)

[73] M. Shanmugavel, A. Tsourdos, R. Zbikowski, and B. White. Path planning of multiple UAVs in an environment of restricted regions. In *Proceedings, 2005 ASME International Mechanical Engineering Congress and Exposition*, pages 115–122, Orlando, FL, 2005. (Cited on pages 68, 71, and 74.)

[74] M. Shanmugavel, A. Tsourdos, R. Zbikowski, B. A. White, C. A. Rabbath, and N. Léchevin. A solution to simultaneous arrival of multiple UAVs using Pythagorean hodograph curves. In *Proceedings, 2006 American Control Conference*, pages 2813–2818, 2006. (Cited on pages 65, 68, and 71.)

[75] Z. Sír, E. Wings, and B. Jüttler. Rounding spatial G-code tool paths using Pythagorean hodograph curves. *Trans. of the ASME, Journal of Computing and Information Science in Engineering* , 7:186–191, 2007. (Cited on pages 10 and 83.)

[76] S. Subchan, B. A. White, A. Tsourdos, M. Shanmugavel, and R. Zbikowski. Pythagorean hodograph (PH) planning for tracking airborne contaminant using sensor swarm. In *Proceedings, IEEE International Instrumentation and Measurement Control Technology Conference*, Victoria, BC, Canada, 2008. (Cited on pages 9 and 65.)

[77] S. Suh, S. Kang, S. Chung, and D. Stroud. *Theory and Design of CNC Systems*. Springer–Verlag, London, United Kindgom, 2008. (Cited on page 9.)

[78] Y.-F. Tsai, R. T. Farouki, and B. Feldman. Performance analysis of CNC interpolators for time–dependent feedrates along PH curves. *Comput. Aided Geom. Design* , 18:245–265, 2001. (Cited on pages 8 and 10.)

[79] Z. Šír, R. Feichtinger, and B. Jüttler. Approximating curves and their offsets using biarcs and Pythagorean hodograph quintics. *Comput. Aided Design* , 38:608–618, 2006. (Cited on page 32.)

[80] Z. Šír and B. Jüttler. Constructing acceleration continuous tool paths using Pythagorean Hodograph curves. *Mech. and Mach. Theory*, 40:1258–1272, 2005. (Cited on pages 10 and 83.)

[81] L. Xinhua, P. Junquan, S. Lei, and W. Zhongbin. A novel approach for NURBS interpolation through the integration of acc-jerk-continuous-based control method and look-ahead algorithm. *Int. J. Adv. Manuf. Technol.* , 88:961–969, 2017. (Cited on pages 9, 82, and 96.)

[82] S. Yeh and P. Hsu. Adaptive-feedrate interpolation for parametric curves with a confined chord error. *Comput. Aided Design* , 34:229–237, 2002. (Cited on pages 9, 82, and 88.)